SinoDB 管理员指南

星瑞格 SinoDB 产品系列

SinoDB

版本 12.10

目录

插图清单..... 14

表格清单..... 18

简介..... 21
本简介内容..... 21
关于本出版物..... 21
用户类型..... 21
软件依赖性..... 21
演示数据库..... 21
示例代码约定..... 22
语法图..... 22
符合行业标准..... 23

第I部分 数据库服务器..... 24
数据库服务器配置和管理概述..... 24
数据库服务器概念..... 24
环境配置..... 25
数据库服务器配置..... 26
存储空间的创建和管理..... 26
自动性能调优..... 27
功能配置..... 28
连接配置..... 29
限制会话资源..... 29
UNIX™上的自动启动和关闭..... 30
在 Windows™ 上自动启动..... 30
数据库服务器维护任务..... 30
数据库服务器监视..... 31
客户端/服务器通信..... 33
客户端/服务器体系结构..... 34
网络协议..... 34
网络编程接口..... 34
Windows™ 网络域..... 34
数据库服务器连接..... 35
支持多路复用连接..... 35
数据库服务器支持的连接..... 36
本地连接..... 37
共享内存连接 (UNIX™)..... 37
流管道连接 (UNIX™ 和 Linux™)..... 38
命名管道连接 (Windows™)..... 39
本地回送连接..... 39
通信支持服务..... 40
连接文件..... 40
网络配置文件..... 40
网络安全性文件..... 42
sqlhosts 文件和 SQLHOSTS 注册表键..... 46
sqlhosts 信息..... 47
sqlhosts.std 文件中的 IANA 标准服务名称及端口号..... 48
sqlhosts 连接信息..... 49
组信息..... 57

TCP/IP 连接的备用方法..... 59
SinoDB® 对 IPv6 地址的支持..... 61
连接相关的配置参数..... 61
DBSERVERNAME 配置参数中设置的连接信息..... 62
DBSERVERALIASES 配置参数中设置的连接信息..... 62
LIMITNUMSESSIONS 配置参数中设置的连接信息..... 63
NETTYPE 配置参数中设置的连接信息..... 63
在 NS_CACHE 配置参数中设置的名称服务最大保留时间..... 64
NUMFDSERVERS 配置参数中设置的连接信息..... 64
HA_ALIAS 配置参数中设置的连接信息..... 65
网络连接的环境变量..... 65
自动终止空闲连接..... 65
Distributed Relational Database Architecture™ (DRDA®) 通信..... 66
DRDA®概述..... 66
配置 SinoDB® 数据库服务器以连接 IBM® Data Server 客户端..... 66
用NETTYPE配置参数为接口/协议组合分配轮询线程..... 68
使用 DRDA_COMMBUFFSIZE 配置参数指定DRDA® 通信缓冲区的大小..... 69
DRDAEXEC 线程与客户端查询..... 69
SQL与受支持和不受支持的数据类型..... 69
显示 DRDA® 连接信息..... 70
显示DRDA®会话信息..... 70
客户端/服务器配置的示例..... 71
网络连接..... 71
多种连接类型..... 71
访问多个数据库服务器..... 72
数据库服务器初始化..... 73
初始化过程..... 73
数据库服务器运行模式..... 77
允许更改模式的用户..... 78
更改数据库服务器运行模式..... 78

第II部分 磁盘、内存和进程管理..... 81
虚拟处理器和线程..... 81
虚拟处理器..... 81
线程..... 82
虚拟处理器的优势..... 82
虚拟处理器如何服务线程..... 84
控制结构..... 84
上下文切换..... 84
堆栈..... 85
队列..... 85
互斥..... 86
虚拟处理器类..... 87
CPU 虚拟处理器..... 88
用户定义的虚拟处理器类..... 91
租户虚拟处理器类..... 92
Java™ 虚拟处理器..... 93
磁盘 I/O 虚拟处理器..... 93
网络虚拟处理器..... 96
通信支持模块虚拟处理器..... 100
审计虚拟处理器..... 100
综合性虚拟处理器..... 101
Basic Text Search 虚拟处理器..... 101
MQ 消息传递虚拟处理器..... 101
XML 虚拟处理器..... 101

管理虚拟处理器..... 102
设置虚拟处理器配置参数..... 102
启动和停止虚拟处理器..... 102
在联机模式下添加虚拟处理器..... 103
删除 CPU 和用户定义的虚拟处理器..... 103
监视虚拟处理器..... 104
使用命令行实用程序监视虚拟处理器..... 104
使用 SMI 表监视虚拟处理器..... 104
共享内存..... 105
共享内存..... 105
共享内存使用..... 105
共享内存分配..... 106
共享内存大小..... 107
超过 SHMTOTAL 时要执行的操作..... 108
连接共享内存的进程..... 108
客户端如何连接到通讯部份 (UNIX™)..... 108
实用程序如何连接共享内存..... 108
虚拟处理器如何连接共享内存..... 109
共享内存常驻部分..... 110
共享内存头..... 111
逻辑日志缓冲区..... 111
物理日志缓冲区..... 112
高可用性数据复制缓冲区..... 112
琐表..... 112
共享内存的缓冲池部分..... 113
共享内存的虚拟部分..... 114
管理共享内存的虚拟部分..... 114
共享内存虚拟部分的组件..... 114
数据分布高速缓存..... 117
共享内存的通信部分 (UNIX™)..... 118
共享内存的虚拟扩展部份..... 119
并发控制..... 119
共享内存互斥..... 119
共享内存缓冲区锁定..... 119
数据库服务器线程对共享缓冲区的访问..... 120
FIFO/LRU 队列..... 120
预读操作..... 122
数据库服务器线程对缓存页的访问..... 123
将数据清空到磁盘..... 123
清空缓冲池缓冲区..... 123
首先清空前映像..... 123
清空物理日志缓冲区..... 124
同步缓冲区清空..... 124
清空期间的写入类型..... 124
清空逻辑日志缓冲区..... 125
缓冲区大对象数据..... 126
写入简单大对象..... 126
访问智能大对象..... 127
64 位平台上的内存使用..... 127
管理共享内存..... 128
设置操作系统共享内存配置参数..... 128
最大共享内存段大小..... 129
信号 (UNIX™)..... 129
设置数据库服务器共享内存配置参数..... 129
设置 SQL 语句高速缓存参数..... 131
设置共享内存..... 132

打开或关闭常驻共享内存的驻留 132
在联机模式下打开或关闭驻留 132
重启数据库服务器时打开或关闭驻留 132
将段添加到共享内存的虚拟部分 132
为关键活动保留内存 133
配置内存严重过低时的服务器响应 133
目标内存量的维护方案 133
监视共享内存 134
监视共享内存段 134
监视共享内存概要文件和锁存器 134
监视缓冲区 135
服务器故障后删除共享内存段 136
数据存储 138
块 138
块的磁盘分配 139
可扩展块 140
分区和偏移量 140
页 141
BLOB 页 141
智能大对象页 142
扩展数据块 142
数据库空间 144
控制简单大对象数据的存储位置 144
根数据库空间 145
临时数据库空间 145
BLOB 空间 146
智能大对象空间 146
使用智能大对象空间的优势 146
智能大对象空间和 Enterprise Replication 147
元数据，用户数据和保留区域 147
控制智能大对象数据的存储位置 148
智能大对象空间的储存特征 149
继承智能大对象空间特征的继承级别 150
有关智能大对象空间的更多信息 151
临时智能大对象空间 152
物理日志空间 153
外部空间 154
数据库 154
表 155
损坏的表 155
SinoDB® 的表类型 156
标准永久表 156
RAW 表 156
Temp 表 157
表类型的属性 157
临时表 158
表空间 160
表中的最大表空间数 160
表和索引表空间 160
扩展数据块交错 161
表分段存储和数据存储 161
存储数据所需的磁盘空间量 163
根数据库空间的大小 163
数据库所需空间量 164
存储池 164
磁盘布局准则 165

数据库空间和块准则 165
表位置准则 165
样本磁盘布局 166
逻辑卷管理器 168
管理磁盘空间 168
分配磁盘空间 169
指定偏移量 169
在 UNIX™ 上分配熟文件空间 170
在 UNIX™ 上分配原始磁盘空间 170
创建原始设备的符号链接 (UNIX™) 171
在 Windows™ 上分配 NTFS 文件空间 171
在 Windows™ 上分配原始磁盘空间 171
指定存储空间和块的名称 172
指定块的最大大小 172
指定块和存储空间的最大数量 172
更改物理模式后备份 173
监视存储空间 173
管理数据库空间 173
创建使用缺省页大小的数据库空间 173
创建具有非缺省页大小的数据库空间 175
通过使用直接 I/O 提高熟文件数据库空间的性能 176
将多个指定分段存储在单个数据库空间 176
创建临时数据库空间 178
磁盘空间不足时应执行的操作 178
向数据库空间和 BLOB 空间添加块 178
重命名数据库空间 179
管理自动定位和分段化 180
管理 BLOB 空间 180
创建 BLOB 空间 181
准备 BLOB 空间以存储 TEXT 和 BYTE 数据 181
确定 BLOB 页大小 182
管理智能大对象空间 182
创建智能大对象空间 182
计算智能大对象空间元数据的大小 183
向智能大对象空间添加块 183
更改智能大对象的存储特征 184
创建临时智能大对象空间 184
管理物理日志空间 185
自动空间管理 185
创建和管理存储池条目 186
将块标记为可扩展或不可扩展 188
更改可扩展存储空间的大小 188
更改自动添加更多空间的阈值和等待时间 189
配置监视低存储任务的频率 189
手动扩展空间或可扩展块 190
自动添加更多空间的最低配置和测试的示例 190
自动添加更多空间的配置示例 191
删除块 192
验证块是否为空 192
使用 onspaces 从数据库空间删除块 193
从 BLOB 空间删除块 193
使用 onspaces 从智能大对象空间删除块 193
删除存储空间 193
删除存储空间的准备工作 194
删除镜像存储空间 194
使用 onspaces 删除存储空间 194

删除存储空间后备份 194
从存储池创建空间或块 195
将空的空间返还给存储池 196
管理外部空间 196
创建外部空间 197
删除外部空间 197
跳过无法访问的分段 197
DATASKIP 配置参数 197
onspaces 的数据跳过功能 197
使用 onstat 检查数据跳过状态 198
SQL 语句 SET DATASKIP 198
数据跳过功能对事务的影响 198
确定何时使用数据跳过 198
监视分段存储使用 199
显示数据库 199
SMI 表 199
监视磁盘使用 199
监视块 199
监视表空间和扩展数据块 202
监视 BLOB 空间中的简单大对象 203
监视智能大对象空间 204
多租户 208
创建租户数据库 209
管理租户数据库 211
复原租户数据库到某一时间点 212
存储优化 213
存储优化方法 214
自动优化数据存储 215
示例: 根据需要优化数据存储 216
对分区取消分段 217
压缩 218
向表装载数据 222
使用外部表移动数据 222
外部表 222
定义外部表 223
将列映射到其他列 224
从命名管道装载数据和将数据卸载到命名管道 224
使用命名管道装载数据 224
FIFO 虚拟处理器 225
使用命名管道卸载数据 225
使用 PIPE 选项将数据从一个实例复制到另一个实例 226
监视装载或卸载操作 227
监视常用装载或卸载操作 227
监视 FIFO 虚拟处理器 227
高可用性集群环境中的外部表 228
外部表的系统目录条目 229
使用外部表时的性能注意事项 229
管理外部表装载和卸载操作产生的错误 230
拒绝文件 230
外部表错误消息 231
外部表的表类型的可恢复性 231

第III部分 日志记录和日志管理 232
日志记录 232
需要日志记录的数据库服务器进程 232

事务日志记录..... 233
SQL 语句和数据库服务器活动的日志记录。..... 233
始终纪录的活动..... 233
为具有事务日志记录的数据库纪录活动..... 236
未记录的活动..... 236
数据库日志记录状态..... 237
未缓冲的事务日志记录..... 238
已缓冲的事务日志纪录..... 238
符合 ANSI 的事务日志纪录..... 238
无数据库日志记录..... 238
具有不同日志缓冲状态的数据库..... 238
X/Open DTP 环境中的数据库日志记录..... 239
日志记录状态或模式的设置或更改..... 239
管理数据库日志记录模式..... 239
更改数据库日志记录模式..... 239
使用 ondblog 更改数据库日志记录模式..... 240
使用 ondblog 更改缓冲模式..... 240
使用 ondblog 取消日志记录模式更改..... 240
使用 ondblog 结束日志记录..... 241
使用 ondblog 使数据库符合 ANSI 标准..... 241
更改符合 ANSI 标准的数据库的日志记录模式..... 241
用 ontape 更改数据库日志纪录模式..... 241
使用 ontape 打开事务日志纪录..... 241
使用 ontape 结束日志记录..... 241
使用 ontape 更改缓冲模式..... 242
使用 ontape 使数据库符合 ANSI 标准..... 242
修改表日志记录模式..... 242
更改表以关闭日志记录..... 242
更改表以打开日志记录..... 242
禁用对临时表的日志记录..... 243
监视事务..... 243
监视数据库的日志记录模式..... 243
使用 SMI 表监视日志记录模式..... 243
逻辑日志..... 243
什么是逻辑日志？..... 243
逻辑日志文件的位置..... 244
逻辑日志文件的标识..... 244
逻辑日志文件的状态标志..... 245
逻辑日志文件的大小..... 245
逻辑日志文件数量..... 246
性能注意事项..... 246
动态日志分配..... 246
释放逻辑日志文件..... 247
下一个逻辑日志文件未释放时要执行的操作..... 247
下一个日志文件包含上一个检查点时要执行的操作..... 247
记录 BLOB 空间和简单大对象..... 247
切换日志文件以激活 BLOB 空间..... 248
备份日志文件以释放 BLOB 页..... 248
插入或删除 TEXT 和 BYTE 数据之后备份 BLOB 空间..... 248
记录智能大对象空间和智能大对象..... 248
智能大对象空间日志记录..... 248
智能大对象日志记录..... 250
在记录智能大对象数据时防止长事务..... 250
日志记录过程..... 250
数据库空间日志记录..... 250
BLOB 空间日志记录..... 250

管理逻辑日志文件..... 250
估计日志文件的大小和数量..... 251
估计记录智能大对象时的日志大小..... 252
估计逻辑日志文件的数量..... 252
备份逻辑日志文件..... 252
备份 BLOB 空间..... 253
备份智能大对象空间..... 253
切换到下一个逻辑日志文件..... 253
释放逻辑日志文件..... 253
删除状态为 D 的日志文件..... 254
释放状态为 U 的日志文件..... 254
释放状态为 U-B 或 F 的日志文件..... 254
释放状态为 U-C 或 U-C-L 的日志文件..... 254
释放状态为 U-B-L 的日志文件..... 255
监视日志记录活动..... 255
监测逻辑日志的填充度..... 255
监视临时逻辑日志..... 256
SMI 表..... 256
监视日志备份状态..... 256
分配逻辑日志文件..... 256
动态添加逻辑日志文件以防止事务阻塞..... 257
动态添加逻辑日志以提升性能..... 259
手动添加逻辑日志文件..... 259
删除逻辑日志文件..... 261
更改逻辑日志文件大小..... 261
移动逻辑日志文件..... 261
显示逻辑日志记录..... 262
控制长事务..... 262
物理日志记录、检查点和快速恢复..... 264
临界区..... 264
物理日志记录..... 264
快速恢复使用物理记录页..... 264
备份使用物理记录页..... 264
物理记录的数据库服务器活动..... 264
物理日志的大小和位置..... 265
用于估计物理日志大小的策略..... 265
当事务日志记录关闭时物理日志溢出..... 266
检查点..... 266
在检查点之间用于清空缓冲池的 LRU 值..... 267
备份期间的检查点..... 267
快速恢复..... 268
需要快速恢复..... 268
启动快速恢复的情境..... 268
检查点之后的快速恢复..... 268
管理物理日志..... 270
更改物理日志的位置和大小..... 270
监视物理和逻辑日志记录活动..... 270
监视检查点信息..... 271
打开或关闭检查点调整..... 272
强制执行检查点..... 272
服务器提供的检查点统计信息..... 272
SMI 表..... 272
打开或关闭自动 LRU 调整..... 273

第IV部分 容错..... 274

镜像..... 274
镜像..... 274
镜像..... 274
镜像的好处..... 274
镜像成本..... 274
不制作镜像的后果..... 275
要制作镜像的数据..... 275
镜像的替代方法..... 275
镜像过程..... 276
创建镜像块..... 276
镜像状态标志..... 276
恢复..... 276
处理期间执行的操作..... 276
停止镜像的结果..... 277
镜像块的结构..... 277
使用镜像..... 277
准备对数据制作镜像..... 278
启用 MIRROR 配置参数..... 278
为镜像数据分配磁盘空间..... 278
链接块 (UNIX™)..... 278
在磁盘故障后将块重新链接到设备..... 278
使用镜像..... 279
初始化期间为根数据库空间制作镜像..... 279
更改镜像状态..... 279
管理镜像过程..... 279
启动未镜像存储空间的镜像..... 280
启动新存储空间的镜像..... 280
添加镜像块..... 280
使镜像块脱机..... 280
恢复镜像块..... 281
结束镜像..... 281
一致性检查..... 281
执行定期的一致性检查..... 281
验证一致性..... 282
监视数据不一致性..... 283
保留一致的 0 级备份..... 284
处理损坏..... 284
查找损坏的徵兆..... 284
修复索引损坏..... 284
修复块上的 I/O 错误..... 284
收集诊断信息..... 285
禁用 I/O 错误..... 285
监视数据库服务器是否有禁用 I/O 错误..... 286
用于监视禁用 I/O 错误的消息日志..... 286
用于监视禁用 I/O 错误的事件警报..... 286
无坏扇区映射..... 287

第V部分 高可用性和可扩展性..... 288
高可用性和可扩展性策略..... 288
支持高可用性和可扩展性的组件..... 289
数据复制的优点..... 290
透明扩展和工作负载平衡策略..... 291
高可用性策略..... 293
高可用性集群配置..... 294
高可用性集群规划..... 294
配置集群..... 295

集群的硬件和操作系统需求。 295
集群的数据库和数据需求 295
集群的数据库服务器配置需求 296
为高可用性集群配置安全连接 298
首次启动 HDR 298
使用 ontape STDIO 功能来减少设置时间 301
远程独立辅助服务器 302
比较 RS 辅助服务器和 HDR 辅助服务器 302
索引页日志记录 303
首次启动 RS 辅助服务器 303
将脱机主服务器转换为RS 辅助服务器 305
延迟应用日志记录 306
远程独立辅助服务器的流量控制 308
共享磁盘辅助服务器 308
SD 辅助服务器 308
SD 辅助服务器的磁盘需求 309
设置共享磁盘辅助服务器 309
获取 SD 辅助服务器统计信息 310
将 SD 辅助服务器升级为主服务器 310
将主服务器转换为标准服务器 310
共享磁盘辅助服务器的流量控制 310
集群管理 310
数据复制的工作原理 311
数据初始复制的工作原理 311
将主服务器数据复制到辅助服务器 311
数据复制配置示例 315
高可用性集群环境故障诊断 323
设计数据复制组客户端 324
执行基本管理任务 325
更改 HDR 复制对的配置参数 325
备份存储空间和逻辑日志文件 325
更改数据库的日志记录模式 326
添加和删除块与存储空间 326
重命名块 326
辅助数据库服务器上保存块状态 326
使用和更改块的镜像 327
管理物理日志 327
管理逻辑日志 327
管理虚拟处理器 327
管理共享内存 327
设置 SMX 连接 328
将索引复制到 HDR 辅助数据库服务器 328
调整 HDR 服务器对中的 LRU 清空和自动调节 329
克隆主服务器 329
辅助服务器上的数据库更新 331
高可用性群集的备份和复原 335
更改数据库服务器模式 335
更改数据库服务器类型 336
防止 HDR 服务器上的检查点阻塞 336
监视 HDR 状态 337
获取 RS 辅助服务器统计信息 338
移除 RS 辅助服务器 339
创建或更改 RS 辅助服务器上的密码 339
在集群故障转移期间完成事务 339
配置服务器以使事务在故障转移后完成 340
通过连接管理器进行连接管理 340

配置连接管理器..... 341
创建连接管理器配置文件..... 342
为连接管理配置环境与设置配置参数..... 371
定义连接管理的 sqlhosts 信息..... 372
启动 UNIX™ 和 Linux™ 上的连接管理器..... 391
在 Windows™ 上启动连接管理器..... 392
停止连接管理..... 392
连接管理的监视和故障排除..... 393
提高连接管理可用性的策略..... 393
连接管理的配置示例..... 394
高可用性集群的配置连接管理示例..... 394
网格或复制集连接管理配置的示例..... 397
高可用性复制系统连接管理配置的示例..... 400
示例: 优先连接和网络监视的连接管理配置..... 403
集群故障转移、重定向和复原..... 405
高可用集群的故障转移配置..... 405
使用 ISV 集群管理软件进行故障转移..... 406
共享文件系统的 I/O 防护..... 406
集群故障..... 407
数据复制客户端的重定向与连接..... 409
使用 DBPATH 环境变量自动重定向客户端..... 409
使用连接信息重定向客户端..... 410
使用 INFORMIXSERVER 环境变量重定向客户端..... 412
使用应用程序代码重定向客户端..... 412
比较重定向方法..... 413
故障后恢复 HDR 和 RS 集群..... 414
关键数据损坏后恢复集群..... 415
网络故障后重新启动 HDR 和 RS 集群..... 416
辅助服务器发生故障时重新启动 HDR 或 RS 集群..... 416
在辅助服务器成为主服务器后恢复 HDR 集群..... 416
主服务器发生故障时重新启动..... 417
数据损坏后恢复共享磁盘集群..... 419
在辅助服务器成为主服务器后恢复 SD 集群..... 419

第VI部分 分布式数据..... 421
多阶段提交协议..... 421
事务管理器..... 421
与事务管理器搭配的 TP/XA 库..... 421
Microsoft™ Transaction Server (MTS/XA)..... 422
SinoDB® 事务对符合 XA 的外部数据源的支持..... 422
高可用性集群中的 XA..... 423
松耦合和紧耦合方式..... 424
两阶段提交协议..... 424
当使用两阶段提交协议时..... 425
两阶段提交概念..... 425
两阶段提交协议的阶段..... 425
两阶段提交协议如何处理故障..... 426
假定结束的优化..... 427
独立操作..... 427
启动独立操作的情境..... 427
独立操作的可能结果..... 427
启发式回滚场景..... 428
启发式结束事务场景..... 430
监视全局事务..... 431
两阶段提交协议错误..... 432

两阶段提交和逻辑日志记录..... 433
事务提交时的逻辑日志记录..... 433
启发式回滚期间写入的逻辑日志记录..... 434
启发式结束事务后写入的逻辑日志记录..... 434
两阶段提交中使用的配置参数..... 435
DEADLOCK_TIMEOUT 参数的功能..... 435
TXTIMEOUT 参数的功能..... 435
从失败的两阶段提交手动恢复..... 436
确定是否需要手动恢复..... 436
确定事务是否实现不一致..... 436
确定分布式数据库是否包含不一致的数据..... 437
决定是否需要执行操作来更正情况..... 438
手动恢复示例..... 438

第VII部分 自动监视与更正操作概述..... 441
调度程序..... 441
调度程序表..... 442
内置任务和传感器..... 443
创建任务..... 448
创建传感器..... 450
任务和传感器的操作..... 452
创建组..... 453
创建阈值..... 454
创建警报..... 455
监视调度程序..... 456
修改调度程序..... 458
使用 SQL 管理 API 执行远程管理..... 458
SQL 管理 API 的 admin() 与 task() 函数..... 459
查看 SQL 管理 API 历史记录..... 460
控制 command_history 表的大小..... 460
向下钻取查询..... 461
使用 SQLTRACE 配置参数指定启动 SQL 跟踪信息..... 463
全局禁用 SQL 跟踪或禁用对某个会话的 SQL 跟踪..... 464
启用 SQL 跟踪..... 464
启用某个会话的全局 SQL 跟踪..... 465

插图清单

图 1: 客户端应用程序和数据库服务器通过共享内存连接通信..... 38

图 2: 客户端应用程序与名为 river_shm 的数据库服务器之间的共享内存连接。..... 38

图 3: 在名为 river 的计算机上客户端与名为 river_soc 的数据库服务器之间的本地回送连接。..... 39

图 4: sqlhosts 信息到 hosts 和 services 文件的关系..... 51

图 5: 使用多块网络接口卡..... 60

图 6: 网络客户端/服务器配置的示例..... 71

图 7: 使用多种连接类型的 UNIX 客户端/服务器配置示例..... 72

图 8: UNIX上的多个数据库服务器..... 72

图 9: 虚拟处理器..... 81

图 10: 并行处理..... 83

图 11: 上下文切换：虚拟处理器如何从一个线程切换到另一个..... 85

图 12: 虚拟处理器为每个用户隔离非共享数据..... 85

图 13: 数据库服务器如何排队执行数据库 I/O..... 86

图 14: 数据库服务器..... 88

图 15: 处理器专用..... 90

图 16: 侦听线程和轮询线程在与客户端连接时的角色。..... 97

图 17: 侦听线程和 sqlexec 线程在与客户端应用程序通信时的角色..... 98

图 18: 数据库服务器如何使用共享内存..... 106

图 19: 数据库服务器共享内存的内容..... 107

图 20: 共享内存下边界地址概述..... 110

图 21: 逻辑日志缓冲区及其与磁盘上逻辑日志文件的关系..... 111

图 22: 共享内存缓冲池..... 113

图 23: LRU 队列..... 120

图 24: 将简单大对象写入 BLOB 空间..... 127

图 25: 磁盘上的页..... 141

图 26: 块，在逻辑上被分成一系列页..... 141
图 27: 磁盘上的 BLOB 页..... 141
图 28: BLOB 空间中的块，在逻辑上分成一系列 BLOB页..... 142
图 29: 智能大对象空间中的块，在逻辑上分成一系列扩展数据块。..... 142
图 30: 原始磁盘空间上六个连续页构成的扩展数据块..... 143
图 31: 扩展数据块的分配过程..... 143
图 32: 使用 CREATE TABLE...IN 语句来控制表的位置..... 144
图 33: 连接逻辑和物理存储单元的数据库空间..... 145
图 34: 控制智能大对象的放置..... 148
图 35: 连接逻辑和物理存储单元的智能大对象空间..... 148
图 36: 存储特征层次结构..... 150
图 37: stores_demo 数据库..... 154
图 38: 跨多个块的表..... 155
图 39: stores_demo 数据库中的样本表空间..... 161
图 40: 属于一个数据库空间中两个不同表空间的三个扩展数据块..... 161
图 41: 链接逻辑存储单元（包括表分段）和物理存储单元的数据库空间..... 162
图 42: 针对性能优化的磁盘布局..... 167
图 43: 针对可用性优化的磁盘布局..... 168
图 44: SinoDB 服务器实例中的多个租户..... 209
图 45: 压缩和存储优化过程中分段中的数据..... 213
图 46: 打开智能大对象空间的日志记录..... 253
图 47: 将物理日志中所有剩馀的前映像写回磁盘..... 269
图 48: 前滚自最近检查点以来写入的逻辑日志记录..... 269
图 49: 回滚所有未完成事务。..... 269
图 50: 将数据同时写入主块和镜像块..... 274
图 51: 分开读取减少了磁盘头必须移动的最大距离..... 277
图 52: 镜像与集群的比较..... 290
图 53: 数据复制配置中的主数据库服务器和辅助数据库服务器..... 291

图 54: 故障后数据复制配置中的数据库服务器和客户端..... 291
图 55: 如何从主服务器向 HDR 辅助服务器复制数据..... 312
图 56: 管理数据复制的线程..... 312
图 57: 管理 RS 辅助服务器数据复制的线程..... 313
图 58: 使用失物招领文件..... 315
图 59: 配有三台 RS 辅助服务器的主服务器..... 316
图 60: 配有 HDR 辅助服务器和 RS 辅助服务的主服务器..... 316
图 61: 主服务器故障转移到 HDR 辅助服务器..... 316
图 62: 充当 HDR 辅助服务器角色的 RS 辅助服务器..... 316
图 63: 配有两个 SD 辅助服务器的主服务器..... 317
图 64: 充当主服务器角色的 SD 辅助服务器..... 317
图 65: 具有镜像磁盘的主服务器和 SD 辅助服务器..... 317
图 66: 主共享磁盘故障后的共享磁盘镜像..... 317
图 67: 刀片服务器中的主服务器和 SD辅助服务器..... 317
图 68: 多刀片服务器配置，以防止单点故障..... 318
图 69: 刀片服务器故障后的故障转移..... 318
图 70: 刀片服务器配置中的 HDR 辅助服务器..... 318
图 71: 刀片服务器配置中从主服务器故障转移到 SD 辅助服务器..... 318
图 72: 整个刀片服务器故障..... 319
图 73: 配置 Enterprise Replication 作为可恢复组的一部分..... 319
图 74: 三层服务器可用性的配置..... 321
图 75: 第一层保护..... 321
图 76: 第二层保护..... 322
图 77: 第三层保护..... 322
图 78: 语法.....
图 79: cliff_ol 数据库服务器故障前后的连接值..... 411
图 80: 连接的数据库服务器..... 425
图 81: 在提交事务期间的逻辑日志写入..... 433

图 82: 启发式回滚期间写入的逻辑日志记录..... 434
图 83: 启发式结束事务期间写入的逻辑日志记录..... 435

表格清单

表 1: SinoDB 搜索单独指定或与域名一起指定的用户名的位置。..... 35
表 2: 连接类型概括..... 49
表 3: sqlhosts 文件和 SQLHOSTS 注册表键中的服务器选项。..... 52
表 4: sqlhosts 文件和 SQLHOSTS 注册表键中的组选项。..... 53
表 5: 连接重定向选项的设置。..... 53
表 6: 保持活动选项设置..... 54
表 7: 多路复用选项设置..... 55
表 8: PAM 服务设置..... 55
表 9: 可信主机和信任用户查找设置。..... 55
表 10: 集群的安全连接选项。..... 56
表 11: netrc 查找选项设置。..... 56
表 12: 运行模式..... 77
表 13: 更改 Windows 中的运行模式..... 78
表 14: 虚拟处理器类..... 87
表 15: 数据库服务器如何划分磁盘 I/O 的优先级..... 93
表 16: 每个侦听端口的侦听线程..... 97
表 17: 用于侦听单个接口/协议组合的多个端口的 sqlhosts 条目..... 99
表 18: 支持 onsoctcp 接口/协议组合的两个网络接口卡的 sqlhosts 条目示例..... 100
表 19: 配置虚拟处理器的配置参数..... 102
表 20: 配置 SQL 语句高速缓存..... 131
表 21: 查找智能大对象空间任务的信息..... 151
表 22: 临时和标准智能大对象空间..... 152
表 23: 临时和标准智能大对象..... 153
表 24: SinoDB 的表类型..... 156
表 25: 表类型的快速恢复..... 157

表 26: 在 SinoDB 上备份表..... 158
表 27: 在 SinoDB 上复原表..... 158
表 28: 删除块的数据库服务器模式..... 192
表 29: 存储优化方法..... 213
表 30: 存储优化的方法..... 214
表 31: 查看压缩信息的方法..... 221
表 32: 描述外部表文件的系统目录表..... 229
表 33: 在特殊情况下生成日志的 SQL 语句。..... 236
表 34: 日志记录模式转换..... 240
表 35: 逻辑日志文件的编号顺序..... 245
表 36: 逻辑日志状态标志..... 245
表 37: 动态添加日志文件的事件警报..... 258
表 38: DYNAMIC_LOGS 设置..... 259
表 39: 检查数据一致性..... 282
表 40: 共享磁盘辅助服务器的可扩展性..... 291
表 41: 远程独立辅助服务器的可扩展性..... 292
表 42: Enterprise Replication 的可扩展性..... 292
表 43: 首次启动 HDR 的步骤..... 299
表 44: 首次启动带有 RS 辅助服务器的主服务器的步骤..... 304
表 45: 辅助服务器设置信息..... 311
表 46: 各种需求的建议配置..... 322
表 47: 主数据库服务器上的模式更改..... 335
表 48: 辅助数据库服务器上的模式更改..... 336
表 49: 连接管理器事件警报级别..... 347
表 50: DEBUG 连接管理器配置参数的值..... 349
表 51: EVENT_TIMEOUT 连接管理器配置参数值..... 349
表 52: FOC 参数属性..... 350
表 53: FOC 连接管理器配置参数的 ORDER 属性值..... 351

表 54: LOG 连接管理器配置参数值.....355
表 55: SECONDARY_EVENT_TIMEOUT 连接管理器配置参数值..... 359
表 56: SLA 连接管理器配置参数值的属性.....360
表 57: DBSERVERS 属性值..... 361
表 58: MODE 属性的值..... 362
表 59: POLICY 属性的值..... 362
表 60: USEALIASES 属性的值..... 364
表 61: HOST 属性的值..... 364
表 62: NETTYPE 属性的值..... 364
表 63: SERVICE 属性的值..... 364
表 64: SQLHOSTS 连接管理器配置参数值.....369
表 65: DBPATH 连接的重定向方法.....413
表 66: 连接信息的重定向方法.....413
表 67: INFORMIXDIR 连接的重定向方法.....414
表 68: 连接管理器连接的重定向方法..... 414
表 69: 主服务器发生关键介质故障后重启 HDR 的步骤。.....415
表 70: 辅助服务器发生故障后的重新启动 步骤.....416
表 71: 将辅助数据库服务器手动更改为标准服务器后的重新启动步骤..... 417
表 72: 将辅助数据库服务器自动更改为标准服务器后的重新启动步骤..... 418
表 73: 调度程序表.....442
表 74: 内置任务和传感器.....443
表 75: command_history 表中某些信息的示例.....460

简介

本简介内容

本简介概述了本出版物中的信息，并描述了所使用的约定。

关于本出版物

本出版物描述有关配置、管理和使用 SinoDB®的概念与过程。

姐妹篇 《*SinoDB®* 管理员参考》 中包含有关使用 SinoDB® 数据库服务器的参考资料。如果需要调整数据库服务器和 SQL 查询的性能，请参阅 《*SinoDB®* 性能指南》。

用户类型

本出版物是为以下用户编写的：

- 数据库用户
- 数据库管理员
- 数据库服务器管理员
- 性能工程师
- 以下类别的程序员
  - 应用程序开发者
  - DataBlade® 模块开发者
  - 用户定义例程的作者

本出版物编写时假设您已具有下列背景：

- 计算机，操作系统和操作系统提供的实用程序的应用知识
- 关系数据库的一定使用经验或对数据库概念有所了解
- 一些计算机编程经验
- 有数据库服务器管理、操作系统管理或网络管理方面的相关经验

软件依赖性

本出版物在编写时假设您是使用 SinoDB® V12.10 作为数据库服务器。

演示数据库

DB-Access实用程序随SinoDB®数据库服务器产品一起提供，它包括一个或多个以下演示数据库：

- stores_demo 数据库以一家虚构的体育用品批发商的有关信息举例说明了关系模式。星瑞格®出版物中的许多示例均基于 stores_demo 数据库。
- superstores_demo 数据库举例说明了对象关系模式。superstores_demo 数据库包含扩展数据类型、类型和表继承以及用户自定义例程的示例。

有关如何创建和填充演示数据库的信息，请参阅《*SinoDB® DB-Access* 用户指南》。有关数据库及其内容的描述，请参阅《*SinoDB® SQL* 指南*:* 参考》。

用于安装演示数据库的脚本位于 UNIX™ 平台上的 $INFORMIXDIR/bin 目录和 Windows™ 环境中的 %INFORMIXDIR%\bin 目录中。

示例代码约定

SQL 代码的示例出现在整个出版物中。除非另有说明，代码不特定于任何单个的 SinoDB® 应用程序开发工具。

如果示例中仅列出 SQL 语句，那么它们将不用分号定界。例如：您可能看到以下示例中的代码：

```
CONNECT TO stores_demo
...

DELETE FROM customer
   WHERE customer_num = 121
...

COMMIT WORK
DISCONNECT CURRENT
```

要将此 SQL 代码用于特定产品，必须应用该产品的语法规则。例如，如果使用的是 SQL API，那么必须在每条语句的开头使用 EXEC SQL ，并在每条语句的结尾使用分号（或其他合适的定界符）。 如果使用的是 DB-Access，那么必须用分号将多条语句隔开。

> 提示： 代码示例中的省略点表示在整个应用程序中可添加更多的代码，但对于正在讨论的概念是不需要显示的。

有关使用特定应用程序开发工具或 SQL API 的 SQL 语句的详细指导，请参阅您的产品文档。

语法图

语法图使用特殊组件描述 SQL 语句和命令的语法。

沿着线的路径，从左向右，从上往下阅读语法图。

右向双箭头和直线符号►►──表示语法图开始。

直线和右向单箭头符号──►表示语法换行继续。

右向箭头和直线符号►──表示语法接续前一行。

直线、右向和左向箭头符号──►◄表示语法图结束。

语法片段由管线符号|──开始，以线管符号──| 结束。

必选项在水平线（主路径）上。

required_item

可选项在主路径下方。

required_item
[*optional_item*]

如果有两个以上选项，则其会以堆栈形式呈现。

如果必须从选项中选一个选项，那么堆栈的其中一个选项会在主路径上。

required_item
{ *required_choice1required_choice2* }

如果选项为可选的，整个堆栈会在主路径下方。

required_item
[{ *optional_choice1optional_choice2* }]

如果其中一项是默认项，它会在主路径上方，剩余选项则在主路径下方。

required_item
[{ *default_choiceoptional_choiceoptional_choice* }]

在主路径的上方，有个返回到左边的箭头，表示该项可被重复。这种情况下，重复项必须用一个或多个空格间隔。

required_item
repeatable_item

如果重复箭头包含一个逗号，表示重复项目间必须使用逗号分隔。

required_item
,*repeatable_item*

堆栈上方重复箭头表示可以从在堆栈中选择多个选项，或者是重复某一项。

SQL 关键词是以大写来显示（例如，FROM），您必须完全按照显示来拼写。变量是以小写来显示（例如，column-name），它们表示语法中用户提供的字段名或变量值。

如出现标点符号、圆括号、算术运算符，或是其他类似符号，您必须将其视为语法的一部分输入。

有时候，单个变量表示一个语法段。例如，在下图中，变量parameter-block表示标记为 parameter-block的语法段。

required_item <parameter-block>
parameter-block
parameter-block
{ *parameter1* | *parameter2* { *parameter3 parameter4* } }

符合行业标准

SinoDB® 产品符合各种标准。

基于 SinoDB® SQL 的产品完全符合 SQL-92 入门标准（发布为 ANSI X3.135-1992），即 ISO 9075:1992 标准。另外，SinoDB® 数据库服务器的许多功能都遵循 SQL-92 中级和最高级标准，以及 X/Open SQL 公共应用程序环境 (CAE) 标准。

第 I 部分

数据库服务器

数据库服务器配置和管理概述

在安装 SinoDB® 后，配置数据库服务器系统并且开始管理数据库服务器。

安装 SinoDB® 时，请遵循安装指示信息以确保满足所有先决条件 (例如，所有关键文件和目录的许可权均已正确设置)。《*SinoDB*® 安装指南》 有安装指示说明。

您必须有管理数据库服务器的正确权限。对于大多数管理任务，您需要以下权限：

- 在 UNIX™ 上，您必须以 root、 informix 用户身分，或者非根安装的拥有者身分来登录。如果使用了角色分离，那么您必须被授与 DBSA 角色。
- 在 Windows™ 上，您必须是 Informix-Admin 群中一员。

配置数据库服务器时，您有多种选项可选。配置包括定制化环境和数据库服务器。并且可以控制数据库服务器的运行方式和可用功能。

必须配置连接以连接客户端管理工具和应用程序。

必须执行一些初始管理任务来完成数据库服务器系统设置。配置数据库服务器后，您的管理职责将包括一組例程任务。

数据库服务器概念

要管理数据库服务器，您必须对存储、 配置、 日志记录、 CPU 使用、 共享内存使用和自动化等关键概念有所了解。

根数据库空间

根数据库空间是数据库服务器创建的初始数据库空间或存储空间。根数据库空间包含保留页和用来描述并跟踪所有物理和逻辑存储单元的内部表格。根数据库空间是逻辑日志、 物理日志、 数据库和临时表的默认位置。数据库服务器无法在没有根数据库空间的情况下运行。

配置文件 (onconfig)

数据库服务器需要一个配置文件。通常该配置文件的名称是 onconfig.*server_name*。onconfig 文件包含控制数据库服务器属性的配置参数。 数据库服务器会在启动、关机、以及运行中某些操作时读取onconfig文件。许多配置参数可在数据库服务器运行时动态地进行设置。

虚拟处理器

虚拟处理器会运行多个线程来执行查询和其他任务。操作系统将虚拟处理器调用为 CPU 进程。多个虚拟处理器并行运行多个线程。虚拟处理器分为不同类，每个类专门用于处理特定线程类型。

逻辑日志

数据库服务器包含数个逻辑日志文件用于记录数据处理的操作、数据定义操作以及管理信息 (例如，检查点记录，chunk 的添加和删除)。在其他关系数据库管理系统中，逻辑日志类似于事务日志。

物理日志

物理日志存储页的前映像。"前映像"是数据库服务器把变更页记录到磁盘前取得的页的图像。未修改页可用于数据库服务器出现故障，或备份过程需要页面提供数据库服务器数据的准确快照。

缓冲池

缓冲池包含缓冲区用于将来自磁盘的页面高速缓存在共享内存中。对高速缓存页面执行操作会比对从磁盘检索的页面执行操作要快。

高速缓存

数据库服务器使用高速缓存将信息存储在共享内存中，而不是执行磁盘读取或其他操作来获取信息。高速缓存信息可以提高访问同一表的多个查询的性能。

调度程序

调度程序为一个子系统，用于在预定义时间运行一组任务或是运行服务器内部确定的一组任务。任务是 SQL 语句，可收集信息，或者运行特定操作。某些任务为数据库服务器内部任务，会自动运行。如果适用的话，您可以启用其他任务。您可以创建自己的任务并调度何时运行。

系统数据库

系统数据库包含有关数据库服务器的信息。sysmaster 数据库中包含系统监视接口（SMI）表。SMI 表提供有关数据库服务器状态的信息。sysadmin 数据库中包含的表用于包含并组织调度程序任务和传感器，存储由传感器所收集的数据， 记录调度程序作业的结果和 SQL 管理 API 函数。

环境配置

可以通过环境变量的设置，和创建或修改环境变量相关的文件来配置您的环境。可控制环境变量是否针对特定用户或数据库会话在环境等级设置。必须对数据库服务器环境和客户端环境设置环境变量。

如果选择在安装时创建一个数据库服务器实例，安装程序将会设置强制的环境变量。否则，您必须在启动数据库服务器前设置环境变量。以下环境变量是强制的：

- INFORMIXDIR 环境变量指出数据库服务器的安装目录。
- INFORMIXSERVER 环境变量指出数据库服务器的名称。
- ONCONFIG 环境变量指出 INFORMIXDIR/etc 目录中 onconfig 文件的名称。
- PATH 环境变量必须包括 $INFORMIXDIR/bin 目录。

您可以设置其它环境变量来配置数据库服务器环境：

- 如果您计划用非缺省的名称或位置创建一个 sqlhosts 文件，那么将 INFORMIXSQLHOSTS 环境变量设为您的 sqlhosts 文件的名称和路径。
- 如果您计划使用 DB-Access 实用程序来执行 SQL 语句，请使用 INFORMIXTERM 或相似的环境变量来指定终端属性。
- 如果您需要Global Language Support (GLS)，请设置 GLS 环境变量。
- 如果您想启用其它功能，请设置相应的环境变量。某些环境变量控制的功能也可由配置参数控制。环境变量可覆盖配置参数设置。

可设置客户端 API 支持的环境变量来配置客户端环境。有关更多信息，请参阅您的客户端 API 手册。

您有多种环境变量设置方式可选择。例如，可运行 SET ENVIRONMENT 语句来设置当前会话的环境变量。可以在登录脚本、命令提示符或配置文件中添加环境变量设置。

相关链接

《SinoDB SQL 指南: 参考》: 环境变量

《SinoDB GLS 用户指南》: GLS 相关的环境变量

数据库服务器配置

必须通过设置配置参数、创建存储空间以及配置连接来定制数据库服务器属性和功能。并可自动启动。

可以通过设置或修改在 onconfig 文件中的配置参数来定制数据库服务器属性。可以使用 SinoDB® 开放管理工具 (OAT) 来监控或者更新配置。OAT 会对配置参数值提供建议以优化数据库服务器配置。。

如果选择在安装时配置数据库服务器，则会设置许多配置参数和环境变量都，并且自动创建一组存储空间。或者，您也可以手动配置数据库服务器。

第一次启动数据库服务器时，会初始化磁盘空间，并且创建根数据库空间的初始块。存在在那个磁盘空间的任何数据都会被覆盖掉。也会初始化数据库服务器需要的共享内存。之后的数据库服务器启动，则只会初始化共享内存。虽然根数据库空间是日志文件和数据库的缺省位置，但是您可以把日志文件和数据库存储在其他存储空间以防止根数据库空间耗尽。

相关链接

数据库服务器初始化 在第73页

《*SinoDB* 管理员参考》：更改 *onconfig* 文件

《*SinoDB* 安装指南》：安装期间的数据库服务器配置

《*SinoDB* 安装指南》：安装后创建数据库服务器

存储空间的创建和管理

可以创建多个存储空间来存储不同类型的对象，例如：数据，索引，日志，临时对象，而不是把所有对象都存储在根数据库空间中。在磁盘上分配数据的方式会影响数据库服务器的性能。可配置数据库服务器自动最小化数据所需的存储空间，并根据需要自动扩展存储空间。通过配置多租户来分隔多个客户端组织之间的存储和处理资源。

一个存储空间由一个或多个 chunk 组成。最大 chunk 容量是 4 TB。一个实例中可以有高达 32766 个 chunk。

在数据库服务器初始化后，您可以创建存储空间，例如：数据库空间和智能大对象空间。可使用 SinoDB® 开放管理工具 (OAT) 或 onspace 实用程序来创建存储空间和 chunk。

以下为最常见的存储空间：

数据库空间

存储数据库、表、逻辑日志文件，和物理日志文件。

临时数据库空间存储临时表。

智能大对象空间

存储智能大对象。智能大对象包括 CLOB（字符大对象）和 BLOB（二进制大对象）数据类型。用户定义的数据类型也可以使用智能大对象空间。一些 SinoDB® 的功能需要智能大对象空间，例如：Enterprise Replication、J/Foundation、空间数据类型，以及 Basic Text Search 查询。在某些状况下，智能大对象空间会根据需要自动创建。

临时智能大对象空间存储临时智能大对象，但不记录元数据或用户数据。

物理日志空间

存储物理日志。如果没有创建物理日志空间，那么物理日志会存储在数据库空间。

其他类型的存储空间存储特殊类型的数据。

如果在安装时创建服务器，那么就会自动创建某些存储空间。

自动最小化存储空间

通过配置自动数据压缩和合并来最小化数据需要的空间量。可压缩数据，合并和返回空闲空间，以及合并扩展数据块。您可以指定每个操作的发生频率。

可以自动循环消息日志以限制日志的空间量。

自动扩展存储空间

创建存储空间后，可配置服务器根据需要来自动扩展每个存储空间。可为可用的 raw devices、cooked files 和目录创建条目存储池，并且确保 SP_AUTOEXPAND 配置参数为缺省值 1。除了外部空间以外，所有类型的存储空间都是自动扩展的。

自动管理数据位置

可以自动化确定数据库、表和索引位置的过程。可以使数据库服务器为数据库、表和索引选择最佳位置，并且自动将表分段。数据库服务器由有利的非关键空间（具有最有效页面大小和其他因素的空间）来选择位置，而不是以默认方式在根数据库空间创建一个新数据库。数据库服务器以循环制分段新表，并且当表变大时会根据需要增加更多分段。

可以通过指定数据库或表的位置来覆盖自动行为。

多租户

可以在单个 SinoDB® 实例中创建多个租户数据库来分隔多个客户端组织之间的数据、存储空间和处理资源。

相关链接

管理自动定位和分段化 在第180页
数据存储 在第138页
管理磁盘空间 在第168页
自动空间管理 在第185页
存储优化 在第213页
多租户 在第208页
《*SinoDB* 嵌入性指南》：管理嵌入式环境中的消息日志

自动性能调优

可设置配置参数和调度程序任务以使数据库服务器能够自动调整影响性能的值。默认情况下，许多自动调优的配置参数和调度程序任务是用来解决常见的性能问题。

您可以配置数据库服务器来调节资源以提高性能：

- 增加 CPU 虚拟处理器 (VP) 的数目，直至 CPU 处理器的数目或您指定的数量。 将 cpu 类的 VPCLASS 配置参数设置为 autotune=1。
- 增加 AIO VP 的数目。将 aio 类的 VPCLASS 配置参数设置为 autotune=1。
- 增加缓冲池的大小。设置 BUFFERPOOL 配置参数以启用缓冲池的自动扩展。
- 增大或减小 CPU VPs 的私有内存高速缓存的大小。将 VP_MEMORY_CACHE_KB 配置参数设置为私有内存高速缓存的初始大小。
- 增加逻辑日志文件的数目以改善性能。将 AUTO_LLOG 配置参数设置为1，加上用于添加日志文件的数据库空间的名称，以及所有逻辑日志文件可选的最大 KB 数。
- 按需增加物理日志文件的大小以改善性能。创建物理日志空间来存储物理日志。

如果是在安装过程中创建数据库服务器，那么缓冲池、逻辑日志和物理日志将配置为自动扩展。

下面的自动优化选项是缺省启用的。您可以控制是否启用选项。

- 将 CPU 虚拟处理器的数目增加到 CPU 处理器数目的一半以确保最佳性能。通过调度程序的 auto_tune_cpu_vps 任务进行控制。
- 增加 AIO 虚拟处理器和页清理线程的数目以增加 I/O 能力。使用 AUTO_TUNE 配置参数控制。
- 处理预读请求来减少磁盘 I/O 的等待时间。使用 AUTO_TUNE 配置参数控制。
- 必要时频繁触发检查点，并按需添加逻辑日志文件，以避免事务阻塞。使用 AUTO_TUNE 和 DYNAMIC_LOGS 配置参数控制。
- 调优 LRU 清空以提高事务吞吐量。使用 AUTO_TUNE 配置参数控制。
- 在表模式更改后，重新优化 SPL 例程，并重新准备已制备的对象，防止手动进程和错误。使用 AUTO_TUNE 配置参数控制。
- 在预定的时间间隔更新过期或残缺的统计信息，以提高查询性能。使用调度程序中的 Auto Update Statistics 任务和 AUTO_TUNE 配置参数进行控制。

- 对压缩的表、具有行大于页的表，以及具有 VARCHAR、LVARCHAR 和 NVARCHAR 数据的表进行轻扫描。使用 BATCHEDREAD_TABLE 配置参数控制。
- 从索引缓冲区取出一组键以减少缓冲区的读取次数。使用 BATCHREAD_INDEX 配置参数控制。
- 增加共享内存高速缓存以提高查询性能。使用 DS_POOLSIZE、 PC_POOLSIZE、 PLCY_POOLSIZE 以及 USRC_POOLSIZE 配置参数进行控制。

相关链接

内置任务和传感器 在第443页

《*SinoDB* 管理员参考》: 数据库配置参数

《*SinoDB* 管理员参考》: *onspaces -c -P:* 创建物理日志空间

功能配置

您可以配置数据库服务器以支持您所需要的可选功能的类型。

通常启用以下功能：

并行数据库查询

您可以控制数据库服务器用于并行执行决策支持查询的资源。必须平衡联机事务处理 (OLTP) 查询的要求与决策支持查询的要求。必须考虑的资源包括共享内存、 线程、 临时表空间以及扫描带宽等。

数据复制

数据复制是在多个不同的站点表现数据库对象的过程。

高可用性集群配置包含一个主服务器以及一个或多个包含与主服务器相同数据的辅助服务器。高可用性集群可以提供冗余、 故障移转、 负载平衡和可扩展性。您可以通过连接管理器从应用程序直接连接到集群服务器。

Enterprise Replication 在地理分布的数据库服务器之间复制所有数据或指定的数据子集。您可以定义复制服务器组为一个网格来管理与运行跨服务器的查询。可在同一数据库服务器上同时有高可用性集群和 Enterprise Replication。

审计

在安装数据库服务器时，如果启用了角色分离，就可以审计所选的活动。若要使用数据库服务器审计，您必须指定审计记录的存储位置、如何处理错误状况以及其他配置选项。如果您怀疑用户滥用访问权限，那么您可能需要更改用户的审计方式。

安全性

可以通过防止未授权的查看和更改数据或其他数据库对象来维持数据安全。自主访问控制可验证正在尝试执行操作的用户是否被授予所需的权限来执行此操作。可以使用基于标签的访问控制 (LBAC) 来控制谁具有数据各行各列的读取访问权限和写入访问权限。

分布查询

您可以使用数据库服务器查询和更新多个数据库，无论这些数据库是跨多个数据库服务器或在同一数据库服务器实例中。SinoDB® 使用两阶段提交协议，以确保分布式的查询在多个数据库服务器之间统一提交或回滚。

磁盘镜像

当使用磁盘镜像时，数据库服务器会向两个位置写数据。镜像可以消除由于存储设备故障造成的数据丢失。如果镜像数据因任何原因变为不可用，那么数据的镜像可以立即透明地提供给用户。

仓库

您可以创建数据仓库应用程序，并优化您的数据仓库查询。仓库加速器 是一个内存中的数据库，它可提高操作及历史数据解析查询的性能。仓库加速器 使用柱状内存方式来加速复杂的仓库和操作查询，而无需更改或调整应用程序。

相关链接

高可用性和可扩展性 在第288页

镜像 在第274页

《*SinoDB* 性能指南》: 并行数据库查询 *(PDQ)*

《*SinoDB* 安全指南》: 审计数据安全

《*SinoDB* 安全指南》: 保护数据
《*SinoDB* 数据库设计和实现指南》: 分布式查询
《*SinoDB* 仓库加速器管理指南》: 仓库加速器概述
《*SinoDB Enterprise Replication* 指南》: *SinoDB Enterprise Replication* 技术概述

连接配置

连接信息允许客户端应用程序在网络上连接数据库服务器。即使客户端应用程序和数据库服务器在同一台计算机或者节点上，您也必须准备连接信息。

SinoDB® 客户端/服务器连接信息，sqlhost 信息，包括数据库服务器名称，客户端用于连接数据库服务器的连接类型，数据库服务器运行的计算机或者节点的主机名称以及其已知的服务名称。在启动数据库服务器之前，您无需在 sqlhost 信息中指定所有可能的网络连接。但要使一个新的连接可用，您必须关闭数据库服务器然后重新启动它。

sqlhosts 文件包含连接信息。根据您的需要，您可能还需要修改其他的连接和安全文件。

当数据库服务器为联机状态，您可以连接客户端应用程序并且开始创建数据库。在访问数据库的信息之前，客户端应用程序必须连接到数据库服务器环境。您可以从 SinoDB® 客户端软件开发工具包 (Client SDK) 提供的客户端程序 (例如：SinoDB® 开放管理工具 (OAT)、DB-Access 或 API 驱动程序) 发出 SQL 语句，与数据库服务器连接或断开连接。

相关链接

连接文件 在第40页

限制会话资源

您可以限制单个会话的可用资源以更均匀地分配系统的使用，防止资源垄断。

设置以下配置参数以对会话施加限制：

- SESSION_LIMIT_LOCKS 限制一个租户会话可获得的锁数。
- SESSION_LIMIT_MEMORY 限制可分配给一个会话的内存量。
- SESSION_LIMIT_TEMPSPACE 限制可分配给一个会话的临时表空间量。
- SESSION_LIMIT_LOGSPACE 限制一个会话内的事务的大小，基于单个事务填充的日志空间量。
- SESSION_LIMIT_TXN_TIME 限制一个事务在会话中可运行的时间量。

您可以在会话中设置 IFX_SESSION_LIMIT_LOCKS 环境变量来指定比 SESSION_LIMIT_LOCKS 配置参数值更低的锁数。

会话限制不适用于拥有管理特权的用户，如 informix 用户或 DBSA 用户。

session_mgr 线程会终止超过设定限制的事务和会话。session_mgr 线程在数据库服务器启动时启动，保持非活动状态直到发生会话超出限制。

相关链接

《*SinoDB* 管理员参考》: *SESSION_LIMIT_LOCKS* 配置参数
《*SinoDB* 管理员参考》: *SESSION_LIMIT_MEMORY* 配置参数
《*SinoDB* 管理员参考》: *SESSION_LIMIT_TEMPSPACE* 配置参数
《*SinoDB* 管理员参考》: *SESSION_LIMIT_LOGSPACE* 配置参数
《*SinoDB* 管理员参考》: *SESSION_LIMIT_TXN_TIME* 配置参数
《*SinoDB* 管理员参考》: *TENANT_LIMIT_SPACE* 配置参数
《*SinoDB* 管理员参考》: *onstat -g ses* 命令: 显示会话相关信息

UNIX™上的自动启动和关闭

您可以修改 UNIX™ 上的启动或关闭脚本，以自动开启或关闭数据库服务器。

UNIX™ 启动脚本

修改 UNIX™ 启动脚本以在计算机进入多用户模式时自动启动数据库服务器。

1. 在 UNIX™ 启动脚本中添加 UNIX™ 和数据库服务器实用程序命令，使该脚本执行以下动作：
   - 设置 INFORMIXDIR 环境变量为数据库安装目录的全路径名称。
   - 设置 PATH 环境变量包括$INFORMIXDIR/bin 录。
   - 设置 INFORMIXSERVER 环境变量，使 sysmaster 数据库可以更新（如果需要也可以创建）。
   - 运行 oninit 命令，启动数据库服务器并使其处于联机模式。
2. 如果您计划启动多个版本的数据库服务器（多驻留），您必须在脚本中添加命令为每个数据库服务器实例设置 ONCONFIG and INFORMIXSERVER 环境变量，和运行 oninit 命令。
3. 如果不同版本的数据库服务器安装在不同的目录中，您必须在脚本中添加命令为每个版本设置 INFORMIXDIR 环境变量并重复前面的步骤。

UNIX™ 关闭脚本

修改 UNIX™ 关闭脚本，以便在 UNIX™ 关闭时以受控方式关闭数据库服务器。在所有客户端应用程序结束事务并退出之后，数据库服务器关闭命令才会执行。

1. 在 UNIX™ 关闭脚本中添加 UNIX™ 和数据库服务器实用程序命令，使该脚本执行以下动作：
   - 设置 INFORMIXDIR 环境变量为数据库安装目录的全路径名称。
   - 设置 PATH 环境变量包含 $INFORMIXDIR/bin 目录。
   - 将 ONCONFIG 环境变量设置为适当的配置文件。
   - 运行 onmode -ky 命令，从而立即启动关闭命令并使数据库脱机。
2. 如果您运行了多个版本的数据库服务器（多驻留），您必须在脚本中添加命令为每个实例设置 ONCONFIG 环境变量和运行 onmode -ky 命令。
3. 如果不同版本的数据库服务器安装在不同的目录中，您必须在脚本中添加命令为每个版本设置 INFORMIXDIR 环境变量并重复前面的步骤。

相关链接

《SinoDB 管理员参考》: 数据库配置参数

《SinoDB 管理员参考》: oninit 实用程序

《SinoDB SQL 指南: 参考》: SinoDB 产品中的环境变量

在 Windows™ 上自动启动

您可以在 Windows™ 上自动启动数据库服务器。

当 Windows™ 启动时，自动启动数据库服务器：

1. 从 服务 控制应用程序窗口，选择 SinoDB® 服务，并且点击 启动。
2. 在 状态类型 对话框中选择 自动。
3. 在 登录身份 对话框中，选择 此帐户，并且核实 informix 在文本框中。

通过清除 自动 属性，来停止自动启动，

数据库服务器维护任务

除了监测数据库服务器的潜在问题外，定期执行例程维护任务来维持服务器运行平稳和最佳性能。

您可以使用 SinoDB® 开放管理工具 (OAT) 或者 SinoDB® 命令行实用程序来执行以下任务：并非以下所有任务对每个安装都适用。

备份数据和逻辑日志文件

为了确保可以在故障事件中恢复数据库，您需要经常备份存储空间和逻辑日志。可以使用 ON-Bar 实用程序或者 ontape 实用程序来创建备份。可使用 OAT 自动备份。

检查数据的一致性

为了确保数据是一致的，要执行偶然检查。

管理逻辑日志

为了确保数据库服务器性能，要执行逻辑日志管理任务，例如，备份逻辑日志文件、添加和释放逻辑日志文件、调整逻辑日志文件大小，以及指定高水线。当联机时，数据库服务器动态分配逻辑日志文件来防止长交易阻塞了用户交易。

管理物理日志

为了确保数据库服务器的性能，请确保分配了足够空间给物理日志。物理日志的大小和地址可以修改。当数据库服务器启动时，它会检查物理日志是否为空，如果物理日志为空就表示该服务器是在受控模式下关闭的。如果物理日志不为空，数据库服务器就会自动执行快速恢复。系统故障可能会有一个或多个未完成事务，快速恢复会自动将数据库恢复到在系统故障后物理和逻辑上都一致的一个状态。

管理共享内存

为了确保数据库服务器拥有适当的共享内存数量以维持性能目标，请执行以下任务：

- 改变缓冲区的大小或者数量（通过改变逻辑日志或者物理日志缓冲区的大小，或者通过改变共享内存缓冲池的缓冲区的数量）
- 改变共享内存参数值
- 改变强制性驻留（开或关，暂时或者针对某个会话）
- 调整检查点间隔
- 添加段到共享内存的虚拟部分
- 配置 SQL 语句高速缓存来减少内存使用和查询准备时间。

管理虚拟处理器

为了确保数据库服务器的性能，要配置足够多的虚拟处理器 (VP)。虚拟处理器的配置和管理对数据库服务器的性能有直接影响。数据库服务器的虚拟处理器的最佳数量和混合度取决于您的硬件和您的数据库服务器支持的应用程序类型。

管理数据库服务器消息日志

为了确保消息日志空间不被填满，需要监控数据库服务器消息日志的大小。数据库服务器会附加新的条目到该文件。您可以启用数据库服务器消息日志的自动轮转，以限定日志文件的总大小。

相关链接

一致性检查 在第281页
逻辑日志 在第243页
物理日志记录、检查点和快速恢复 在第264页
共享内存 在第105页
虚拟处理器和线程 在第81页
《*SinoDB* 备份和还原指南》：备份与复原的概述
《*SinoDB* 嵌入性指南》：自动循环消息日志文件的任务

数据库服务器监视

您可以使用不同的工具来监控数据库服务器的活动。除了可以使用 SinoDB® 提供的工具和实用程序，也可以使用操作系统提供的工具。

SinoDB® 开放管理工具 (OAT)

OAT 是一个 Web 应用程序用于管理和分析 SinoDB® 数据库服务器性能。您可以从 Web 服务器上安装的单一 OAT 来管理多个数据库服务器实例。您可以通过任何浏览器访问 Web 服务器，以管理所有数据库服务器。SinoDB® 开放管理工具 (OAT) 的 Health Advisor 插件 用于分析 SinoDB® 数据库服务器的状态。Health Advisor 插件搜集关于数据库服务器、数据库，以及操作系统的信息，并可生成包含结果和建议的报告。

您可以利用 SinoDB® 移动开放管理工具，从您的移动设备端检查一个或多个 SinoDB® 数据库服务器的状态和性能。

事件警告

可以启用事件警告以报告需要您立即关注的状况。要使用事件警告功能，请将 ALARMPROGRAM 配置参数设置为执行必要管理操作的可执行文件的完整路径名。您可以在 OAT 中监视事件警告，或者配置接受警告的方式。

数据库服务器消息日志

数据库服务器消息日志是一个操作系统文件。在数据库服务器消息日志中的消息通常不需要立即行动。如果数据库服务器发生故障，则消息日志将作为审计追踪，以回溯导致意外问题的事件。消息日志通常会提供问题的确切性质和建议的纠正措施。

一天监视消息日志一次或两次，以确保处理正常进行并按预期方式记录事件。可使用 OAT 来查看消息日志。

或者，也可运行 onstat -m 命令来获得消息日志的名称以及 20 个最近的条目，或者在文本编辑器中读取完整消息日志。当消息生成时，可以运行操作系统命令行，例如 UNIX™ 命令 tail -f 来查看消息。要查看一条消息的额外的信息，请使用 finderr 实用程序。

onstat 实用程序

onstat 实用程序提供了从命令行中监视数据库服务器的命令。onstat 实用程序从共享内存中读取数据，并且报告到命令运行瞬间的统计信息。即，onstat 提供处理过程中动态改变的信息，包括缓冲区、锁、索引以及用户的改变。您可以在命令行中或者在 OAT 中运行 onstat 命令。

SMI表

系统监视接口 (SMI) 表包含关于数据库服务器状态的动态信息。您可以在 SMI 表上使用 SELECT 语句来确定您想知道的关于数据库服务器的任何内容。SMI 表在 sysmaster 数据库中。

系统控制台

数据库服务器通过系统控制台发送对数据库服务器管理员有用的消息。默认情况下，系统控制台设为 online.con。可设置 CONSOLE 配置参数来改变控制台消息的目标路径名称。

Windows only: 数据库服务器系统管理员可以从任何节点登录控制台来执行系统管理和监视任务。

相关链接

《SinoDB 管理员参考》: 事件警报

《SinoDB 管理员参考》: finderr 实用程序

《SinoDB 管理员参考》: onstat 实用程序

《SinoDB 管理员参考》: sysmaster 数据库

UNIX™ 操作系统工具

数据库服务器依赖于主机的操作系统来提供对系统资源（例如：CPU、内存，以及各种未缓冲的磁盘 I/O 接口和文件）的访问。每个操作系统都有一套自己的实用程序来报告系统资源的使用状况。不同的操作系统可能有具有相同名称的监控实用程序，但他们的选项与信息显示不同。

下表展示了典型的 UNIX™ 操作系统资源监视实用程序。更多关于如何监视操作系统资源的信息，请查询系统管理指南。

UNIX™ 实用程序	描述
vmstat	显示虚拟内存统计信息。
iostat	显示 I/O 利用率统计信息。
sar	显示各种资源统计信息。
ps	显示活动进程信息。
cron	通过系统调度程序（定时运行的命令或程序）来捕捉系统资源状态。还可以使用操作系统提供的其他调度工具。

Windows™ 管理实用程序

您可以使用 SinoDB® 实用程序和 Windows™ 实用程序来管理和监视 Windows™ 操作系统上的数据库服务器。

以下 SinoDB® 实用程序可简化 Windows™ 上的数据库服务器管理。

实用程序	说明和使用
ixpasswd.exe	改变以 informix 用户身份登录的所有服务的登录密码。可以交互地更改密码或使用 -y 选项在命令行上更改密码。使用此实用程序，就无需在每次更改 informix 密码时手动为每个服务更改密码。 如果您本地登录并运行 ixpasswd，它将会更改以本地 informix 用户身份登录的服务的密码。如果您登录到域中并运行 ixpasswd，它将会更改以 ***domain***\informix 身份登录的服务的密码。 用法：ixpasswd [-y *new_password*]
ixsu.exe	开启以指定用户身份运行的命令行窗口。该用户是一个本地用户，除非您指定了一个域名。如果没有指定用户名，则缺省用户为 informix。您不再需要以当前用户身份注销并以 informix 身份登录，以执行必须以 informix 身份运行的数据库管理任务。 ixsu 实用程序需要高级用户权限： • 以操作系统方式操作 • 添加配额 • 替换进程级别记号 有关设置高级用户权限的更多信息，请查看 Windows™ 文件。 用法：ixsu [[*domain*\]*username*]
ntchname.exe	将 SinoDB® 注册表项的旧主机名更改为新主机名。在更改主机名后运行 ntchname。此实用程序不会更改用户环境变量。 执行 ntchname 后，请编辑 %INFORMIXDIR%\%INFORMIXSERVER%.cmd 文件并将 INFORMIXSQLHOSTS 条目更改为新的主机名。 用法：ntchname *old_name new_name*

Windows™ 事件查看器

事件查看器显示有关操作系统、其他应用程序和数据库服务器的信息、警告和错误消息。

要显示数据库服务器消息，请从 管理工具 窗口里选择 事件查看器 >安全性。双击任何事件以获得更加详细的消息。

Windows™ 性能监视器

Windows™ 性能监视器（perfmon.exe）显示资源状况，例如处理器、内存、高速缓存、线程和进程。性能监视器还提供图表、警告和报告功能，以及将信息存储到日志文件以便日后分析。

要显示性能监视器，请从 管理工具 窗口里选择 性能。

客户端/服务器通信

以下主题说明了要配置客户端/服务器通信所必须了解的概念和术语。

相关链接

更改客户端连接信息 在第410页

客户端/服务器体系结构

SinoDB® 产品符合客户端/服务器软件设计模型

应用程序和客户端可以位于数据库服务器的计算机上或者其他计算机上。客户端应用程序将发出请求以获得数据库服务器的服务和数据。数据库服务器通过提供客户端请求的服务和数据来作出响应。

将网络协议连同网络编程接口一起使用可连接客户端和数据库服务器并在它们之间传输数据。

网络协议

网络协议是一组规则，规定了如何在应用程序间传输数据，以及如何在客户端与数据库服务器之间传输数据。协议的规则在网络驱动程序中实施。 网络驱动程序包含将客户端与数据库服务器之间的传输数据格式化的代码。

客户端和数据库服务器通过网络编程接口获得对网络驱动程序的访问权。网络编程接口包含的系统调用或库例程提供了对网络通信设施的访问权。UNIX™ 的网络编程接口的示例是 TLI (传输层接口)。Windows™ 的网络编程接口的示例是 WINSOCK (套接字编程接口)。

网络协议的作用在于它能够启用客户端/服务器通信，尽管客户端和数据库服务器位于具有不同体系结构和操作系统的不同计算机上。

您可以配置数据库服务器来支持多个协议，但是仅当某些客户端使用 TCP/IP 时才考虑该选项。

相关链接

数据库服务器支持的连接 在第36页
sqlhosts 文件和 *SQLHOSTS* 注册表键 在第46页
数据库服务器连接 在第35页
网络配置文件 在第40页
数据库服务器连接 在第35页

网络编程接口

网络编程接口是一种应用程序编程接口 (API)，它包含一组通信例程或系统调用。应用程序可以调用这些例程以便与位于相同或不同计算机上的其他应用程序通信。

在该说明的上下文中，客户端和数据库服务器是调用 TLI 或套接字 API 中例程的应用程序。客户端和数据库服务器都使用网络编程接口根据通信协议发送和接收数据。

如果要使客户端/服务器通信成功，那么客户端和数据库服务器的环境都必须用相同的协议配置。然而，一些网络协议可以通过多个网络编程接口来访问。例如，根据操作系统平台上可用的编程接口，可通过 TLI 或套接字来访问 TCP/IP。

相关链接

网络连接 在第71页
通信支持服务 在第40页
网络安全性文件 在第42页

Windows™ 网络域

Windows™ 网络技术允许您创建网络域。域是一组连接的 Windows™ 计算机，它们共享用户帐户信息和安全策略。

域控制器管理所有域成员的用户帐户信息。域控制器使网络管理更便利。通过管理一个所有域成员的帐户列表，域控制器将使网络管理员不需要同步每台域计算机上的帐户列表。换句话说，创建或者更改用户账户的网络管理员仅须更新域控制器上的帐户列表而不需要更新域中每台计算机上的帐户列表。

其他 Windows™ 计算机上的用户要登录到 Widnows™ 数据库服务器，必须属于同一个域或者属于可信域。可信域是与另一个域建立了信任关系的域。在信任关系中，用户帐户只位于可信域中。

用户可以通过使用本地登录名和概要文件或者域登录名和概要文件，尝试登录到作为域成员的 Windows™ 计算机。然而，如果用户已列为可信用户或者用户尝试登录的计算机已列为可信主机，那么可以在没有概要文件的情况下授予用户登录访问权限。

重要： 仅当运行数据库服务器的 Windows™ 域中有该用户标识的帐户时，客户端应用程序才能连接到该数据库服务器。此规则也适用于可信域。

如果指定了用户标识，但针对需要域名和用户名（域\用户）的工作站连接未指定域名，那么数据库服务器将只在本地工作站和主域上查找该用户帐户。如果明确指定了域名，那么将使用该域来查找该用户帐户。如果在本地工作站上找不到匹配的域\用户帐户，尝试连接将失败，错误为 -951。

使用 CHECKALLDOMAINSFORUSER 配置参数可配置数据库服务器在联网 Windows™ 环境中搜索用户名的方法。

表 1: SinoDB® 搜索单独指定或与域名一起指定的用户名的位置。

	指定了域和用户	仅指定了用户名
CHECKALLDOMAINSFORUSER 未设置	仅在指定域中查找	仅在本地主机查找
CHECKALLDOMAINSFORUSER=0	仅在指定域中查找	仅在本地主机查找
CHECKALLDOMAINSFORUSER=1	仅在指定域中查找	在所有域中查找

重要： 数据库服务器的可信客户端机制与在 Windows™ 域之间建立的信任关系无关。因此，即使客户端从可信 Windows™ 域进行连接，用户也必须在运行数据库服务器的域中具有帐户。

相关链接

CHECKALLDOMAINSFORUSER 配置参数

数据库服务器连接

客户端应用程序使用 CONNECT 或 DATABASE SQL 语句建立与数据库服务器的连接。

应用程序可能包含以下 CONNECT 语句以连接到名为 my_server 的数据库服务器：

```
CONNECT TO '@my_server'
```

提示： 数据库服务器的内部通信设施称为关联服务设施 (ASF)。如果看到包含对 ASF 的引用的错误消息，那么说明连接有问题。

相关链接

网络协议 在第34页

网络协议 在第34页

《SinoDB SQL 指南: 语法》: CONNECT 语句

《SinoDB SQL 指南: 语法》: DATABASE 语句

支持多路复用连接

多路复用连接在数据库服务器和客户端之间使用单个网络连接处理来自客户端的多个数据库连接。

客户端应用程序可以建立多个到数据库服务器的连接以代表单个用户访问多个数据库。 如果连接不是多路复用的，那么每个数据库连接将建立到数据库服务器的单独网络连接。每个额外的网络连接都将使用额外的计算机内存和处理器时间，甚至对于不活动的连接亦如此。多路复用连接使数据库服务器可以创建多个数据库连接，而无需使用额外网络连接所需的额外计算机资源。

配置数据库服务器以支持多路复用连接：

1. 使用 DBSERVERALIASES 配置参数定义别名。
   例如，指定：

```
DBSERVERALIASES ifx_mux
```

2. 为别名添加 sqlhosts 文件条目，并用 onsqlmux 作为 nettype 条目。hostname 和 servicename 必须具有条目，但这些条目会被忽略。连字符 (-) 可作为条目。
   例如：

```
#dbservername   nettype    hostname   servicename   options
ifx_mux         onsqlmux   -          -
```

3. 通过在客户端用于数据库服务器连接的 sqlhosts 条目中指定 m=1，以启用选定连接类型的多路复用。
   例如：

```
#dbservername   nettype    hostname   servicename   options
menlo           ontlitcp   valley     jfkl          m=1
```

4. 在 Windows™ 平台，还必须设置 IFX_SESSION_MUX 环境变量。

以下示例显示 onconfig 文件和 sqlhosts文件条目。

onconfig文件：

```
DBSERVERNAME web_tli
DBSERVERALIASES web_mux
```

sqlhosts文件：

```
#dbservername   nettype    hostname   servicename   options
web_tli         ontlitcp   node5      svc5          m=1
web_mux         onsqlmux   -          -
```

无需更改数据库服务器使用的 sqlhosts 信息。客户端程序无需执行任何特殊的 SQL 调用，即可启用连接多路复用。当 onconfig 文件和 sqlhosts 条目进行了适当配置并且数据库服务器启动时，将自动启用连接多路复用。

多路复用连接不支持：

- 多线程客户端连接
- 共享内存连接
- 与下级数据库服务器的连接（例如，用于分布式查询或数据复制）

如果在应用程序尝试建立连接时出现以上任一情况，数据库服务器就会建立标准连接。数据库服务器不返回 SQL 错误。

多路复用连接期间，不支持 SinoDB® ESQL/C sqlbreak() 函数。

相关链接

sqlhosts 文件和 *SQLHOSTS* 注册表键 在第46页
sqlhosts 文件和 *SQLHOSTS* 注册表键选项 在第51页
sqlhosts 文件和 *SQLHOSTS* 注册表键选项 在第51页
《*SinoDB* 管理员参考》: *DBSERVERNAME* 配置参数
《*SinoDB* 管理员参考》: *DBSERVERALIASES* 配置参数
《*SinoDB* 性能指南》: 多路复用连接和 *CPU* 利用率

数据库服务器支持的连接

数据库服务器支持客户端应用程序的不同连接类型。

数据库服务器支持以下连接类型。

连接类型	Windows™	UNIX™	本地	网络
套接字	X	X	X	X
TLI (TCP/IP)		X	X	X
共享内存		X	X	
流管道		X	X	
命名管道	X		X	

配置连接时，请考虑设置 LISTEN_TIMEOUT 和 MAX_INCOMPLETE_CONNECTION 配置参数。通过这些参数，使得处理连接的侦听器 VP 更难遭到覆盖，这样就能够降低受到恶意的拒绝服务 (DOS) 攻击的风险。

仅限 UNIX: 在许多 UNIX™ 平台上，数据库服务器都支持多个网络编程接口。机器说明显示了数据库服务器在操作系统上支持的接口/协议组合。

设置客户端连接：

1. 在 onconfig 文件中指定连通性和连接配置参数。
2. 在平台上的连接文件中设置适当的条目。
3. 在 UNIX™ 启动脚本或本地及全域范围内的 Windows™ 注册表中指定连通性环境变量。
4. 添加 sqlhosts 条目以定义数据库服务器的数据库服务器组。

相关链接

《SinoDB 管理员参考》: NETTYPE 配置参数
网络协议 在第34页
连接文件 在第40页
sqlhosts 信息 在第47页
连接相关的配置参数 在第61页
网络连接的环境变量 在第65页
《SinoDB 管理员参考》: LISTEN_TIMEOUT 配置参数
《SinoDB 管理员参考》: MAX_INCOMPLETE_CONNECTIONS 配置参数
《SinoDB 安全指南》: 限制拒绝服务攻击
sqlhosts 连接信息 在第49页

本地连接

本地连接是同一台计算机上客户端和数据库服务器之间的连接。

以下主题描述了不同类型的本地连接。

共享内存连接 (UNIX™)

共享内存连接使用共享内存的一块区域作为客户端和数据库服务器相互通信的通道。客户端不能与数据库服务器建立多个共享内存连接。

下图说明了共享内存连接。

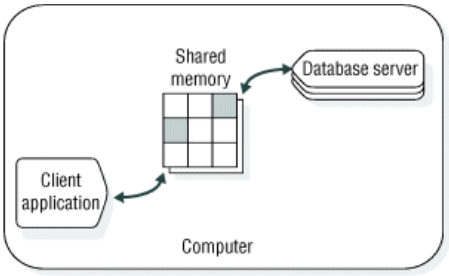

图 1: 客户端应用程序和数据库服务器通过共享内存连接通信

共享内存提供对数据库服务器的快速访问，但是它会引起一些安全性方面的风险。错误或恶意的应用程序可能删除或查看自己的或其他本地用户的消息缓冲区。如果客户端应用程序执行显式内存寻址或过度索引数据数组，那么共享内存通信也容易受到编程错误的影响。如果使用的是网络通信或流管道，那么这些错误不会影响数据库服务器。

共享内存连接的示例

下图显示名为 river 的计算机上的共享内存连接。

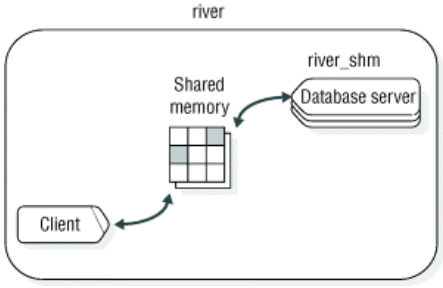

图 2: 客户端应用程序与名为 river_shm 的数据库服务器之间的共享内存连接。

此安装的 onconfig 文件包括以下行：

```
DBSERVERNAME river_shm
```

此安装的 sqlhosts 文件包括以下行：

```
#dbservername   nettype   hostname   servicename options
 river_shm      onipcshm  river      rivershm
```

客户端应用程序使用以下语句连接此数据库服务器：

```
CONNECT TO '@river_shm'
```

对于共享内存连接，网络配置文件无需任何条目。对 sqlhosts 文件的hostname 和 servicename 字段使用任意值。

相关链接

共享内存的通信部分 *(UNIX)* 在第118页
客户端如何连接到通讯部份 *(UNIX)* 在第108页
客户端/服务器配置的示例 在第71页

流管道连接 (UNIX™ 和 Linux™)

流管道是 UNIX™ 进程间通信 (IPC) 设施，该设施允许同一计算机上的进程可以互相通信。

流管道连接有以下优点：

- 与共享内存连接不同，流管道没有被其他明确显示地访问共享内存相同部分的程序覆盖或读取的安全风险。
- 与共享内存连接不同，流管道连接允许在同一台计算机上的数据库服务器之间进行分布式事务。

流管道有以下缺点：

- 在一些计算机上，流管道连接可能比共享内存连接慢。
- 流管道并不适用于所有平台。
- 在客户端/服务器通信中使用共享内存或流管道时，hostname 条目会被忽略。

相关链接

sqlhosts 连接信息 在第49页
sqlhosts 连接信息 在第49页

命名管道连接 (Windows™)

命名管道 是 Windows™ 上双向进程间通信（IPC）的应用程序编程接口（APIs）。

命名管道连接将通过使传输层操作透明而向网络软件提供高级接口。命名管道将数据存储在内存中并且在接到请求时检索这些数据，这种方法类似于从文件系统读取和向其写入数据。

本地回送连接

同一台计算机上客户端应用程序和数据库服务器之间的网络连接称为本地回送连接。

就像客户端应用程序和数据库服务器位于不同的计算机上一样，所用的网络工具是相同的。如果您的计算机具备处理网络事务的功能，那么就可以进行本地回送连接。 本地回送连接不如共享内存连接快，但是本地回送连接不会产生共享内存的安全风险。

在本地回送连接中，数据看上去是从客户端应用程序传出到网络，然后再次传回数据库服务器。内部连接进程是直接在客户端和数据库服务器之间发送信息，而不是把信息传递到网络上去。

本地回送连接示例

下图显示了使用套接字和 TCP/IP 的本地回送连接。

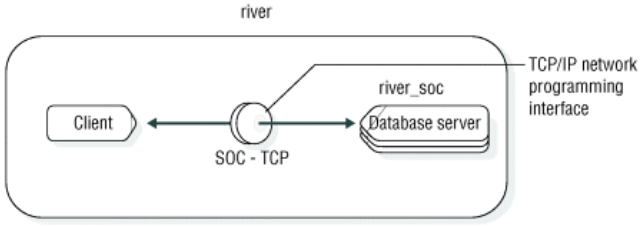

图 3: 在名为 river 的计算机上客户端与名为 river_soc 的数据库服务器之间的本地回送连接。

此安装的 sqlhosts 文件包括以下行：

```
#dbservername   nettype   hostname   servicename   options
 river_soc      onsoctcp  river      riverol
```

如果网络连接使用 TLI 而不是套接字，那么本例中仅 nettype 条目将更改。在这种情况下，nettype 条目是 ontlitcp 而不是 onsoctcp。

此安装的 onconfig 文件包括以下行：

```
DBSERVERNAME river_soc
```

此示例假定 river 条目在 hosts 文件中，riverol 条目在 services 文件中。

相关链接

客户端/服务器配置的示例 在第71页

通信支持服务

通信支持服务包括与连接相关的服务

与连接相关的服务：

- 认证，这是验证用户或应用程序的身份的过程。最常见的认证形式就是要求用户输入名字和密码以获得计算机或应用程序的访问权。
- 消息完整性，用于确保通信消息在到达其目的地时完整无缺且未变更。
- 消息机密性，用于在传输期间保护消息以免受到未经授权的查看。

通信支持服务还可以包括其他处理，如数据压缩或基于流量的记帐。

数据库服务器通过名为“通信支持模块”(CSM) 的插件软件模块来提供额外的安全性相关通信支持服务。当未指定通信支持模块时，数据库服务器将使用此缺省认证策略。

相关链接

网络编程接口 在第34页
网络安全性文件 在第42页

连接文件

连接文件包含支持客户端/服务器之间通信，以及数据库服务器与其他数据库服务器之间通信的信息。

连接配置文件可以分为三组：

- 网络配置文件
- 网络安全性文件
- sqlhosts 文件

  Windows: 在数据库服务器上，连接信息存储在 sqlhosts 文件中；但是在客户端上，连接信息存储在 SQLHOSTS 注册表中。

相关链接

连接配置 在第29页
数据库服务器支持的连接 在第36页

网络配置文件

这些主题标识并说明网络配置文件在 TCP/IP 网络上的使用。

相关链接

网络协议 在第34页

TCP/IP 连接文件

当配置数据库服务器以使用 TCP/IP 网络协议时，可以使用 hosts 和 services 文件内的信息来准备 sqlhosts 信息。

对于连接到在网络上运行 SinoDB® 客户端/服务器产品的计算机的每个网络控制器卡，hosts 文件都需要一个单独的条目。文件中的每个条目都包含 IP 地址（或以太网卡地址）和主机名。您还可以包含主机别名。尽管主机名的长度在 hosts 文件中没有限制，但 SinoDB® 数据库服务器仍将主机名限制为 256 个字节。

以下示例包含两个条目。

```
#address       hostname  alias
98.555.43.21   odyssey
12.34.56.555   illiad    sales
```

services 文件包含在 TCP/IP 上可用的每个服务的条目。每个条目都是一个单独的行，包含以下信息：

- 服务名

SinoDB® 产品使用该名称来确定用于建立客户端/服务器连接的端口号和协议。服务名称限制为 128 个字节。

- 端口号和连接协议，以正斜杠字符分隔（/）

  端口号是计算机端口，而 TCP/IP 的协议是 tcp。

  操作系统会对端口号施加限制。用户 informix 必须使用大于或等于 1024 的端口号。 仅允许 root 用户使用小于 1024 的端口号。
- 主机别名（可选）

服务名称和端口号可以是任意的。然而，它们在文件内必须是唯一的，且在所有运行 SinoDB® 客户端/服务器产品的计算机上都必须相同。以下示例具有一个条目：

```
#servicename   port/protocol
server2        1526/tcp
```

此条目将 server2 作为 TCP 端口 1526 的服务名称。 数据库服务器可以使用此端口来处理连接请求。

重要： 对于与其他数据库服务器通信的数据库服务器，必须为 DBSERVERNAME 配置参数定义 TCP/IP 连接或 IPCSTR（进程间通信流管道）连接。您还可以使用合适的连接协议为协调程序与下级服务器之间的连接定义至少一个 DBSERVERALIASES 配置参数设置。对于跨服务器事务，每个参与服务器必须支持与协调程序间的 TCP/IP 或 IPCSTR 连接，即使两个数据库服务器实例在同一工作站上。

通常，每个与数据库服务器名称关联的连接类型都包含一个单独的 NETTYPE 参数。您可在 DBSERVERNAME 和 DBSERVERALIASES 参数中列出数据库服务器名称条目。通过 sqlhosts 文件或注册表中的条目，可将连接类型与数据库服务器名称相关联。

hosts 和 services 文件必须可用于每台运行 SinoDB® 客户端/服务器产品的计算机。

UNIX:

- hosts 和 services 文件位于 /etc目录中。
- 在使用 NIS 的系统中，hosts 和 services 文件都保留在 NIS 服务器上。 位于本地计算机上的 hosts 和 services 文件可能不会使用，也可能不是最新的。要查看 NIS 文件的内容，请在命令行输入以下命令 ：

  ```
  ypcat hosts
  ypcat services
  ```

Windows:

- hosts 和 services 文件位于 %WINDIR%\system32\drivers\etc\ 中。
- 您可能希望配置 TCP/IP 以使用域名服务 (DNS) 来解析主机名称。
- 动态主机配置协议 (DHCP) 从地址池动态地分配 IP 地址，而不是使用显示地分配给每个工作站的 IP 地址。如果您的系统使用 DHCP，那么 Windows™ 网络名称服务 (WINS) 是必需的。 DHCP 对数据库服务器是透明的。

相关链接

sqlhosts 连接信息 在第49页

当打开 TCP/IP 连接时客户端和服务器的操作
当 TCP/IP 连接打开时，客户端和服务器都会读取信息。

在客户端方读取以下信息 ：

- INFORMIXSERVER 环境变量。
- hosts 文件信息 (INFORMIXSQLHOSTS 环境变量 $INFORMIXDIR/etc/sqlhosts 文件) 和 services 文件信息
- 其他环境变量
- 资源文件

在服务器方读取以下信息 ：

- DBSERVERNAME 配置参数
- DBSERVERALIASES 配置参数
- 服务器环境变量和配置参数，包括管理 TCP/IP 连接的任何 NETTYPE 配置参数设置。

相关链接

《*SinoDB* 管理员参考》: *NETTYPE* 配置参数

多个 TCP/IP 端口
您可以修改服务文件以利用多块以太网卡。

利用多块以太网卡：

- 在 services 文件中，数据库服务器使用的每个端口建立一个条目，如以下示例所示：

```
#servicename   port/protocol   alias
soc1           21/tcp
soc2           22/tcp
```

单个 IP 地址的每个端口必须唯一。 独立的以太网卡可以使用唯一的或共享的端口号。您可能希望连接到同一个数据库服务器的几块以太网卡使用同一个端口号。（在此情况中，服务名称是相同的。）

- 在 hosts 文件中，每个以太网卡有一个条目对应一个独立 IP 地址，如以下示例所示：

```
#address         hostname   alias
192.147.104.19   svc8
192.147.104.20   svc81
```

- 在 onconfig 文件中，为其中一个以太网卡设置 DBSERVERNAME 配置参数，为另一个以太网卡设置 DBSERVERALIASES 配置参数。以下行显示 onconfig 文件中的样本条目：

```
DBSERVERNAME chicago1
DBSERVERALIASES chicago2
```

- 为每个以太网卡添加一个 sqlhosts 条目。即，为 DBSERVERNAME 建立一个条目，并为 DBSERVERALIASES 建立另一个条目。

```
#dbservername   nettype   hostname   servicename   options
chicago1        onsoctcp  svc8       soc1
chicago2        onsoctcp  svc81      soc2
```

此配置完成后，应用程序将通过指定给数据库服务器名称（由 INFORMIXSERVER 环境变量提供）的以太网卡进行通信。

相关链接

《*SinoDB SQL* 指南: 参考》: *INFORMIXSERVER* 环境变量

网络安全性文件

SinoDB® 产品遵循网络安全性文件中所规定的标准安全程序。

对于与远程计算机上数据库服务器连接的客户端应用程序，客户端应用程序的用户必须在远程计算机上具有有效的用户 ID。

相关链接

网络编程接口 在第34页
通信支持服务 在第40页
《*SinoDB* 管理员参考》: *REMOTE_SERVER_CFG* 配置参数
《*SinoDB* 管理员参考》: *REMOTE_USERS_CFG* 配置参数
《*SinoDB* 管理员参考》: *S6_USE_REMOTE_SERVER_CFG* 配置参数

可信主机信息

可信主机上的用户无需密码即可访问本地系统。您可以包含一个可选的用户名，以限制对特定主机上特定用户的身份验证。

使用以下 可信主机文件之一为 rlogin、 rsh、rcp 和 rcmd 的远程认证指定远程主机：

- hosts.equiv
- 由数据库服务器的 REMOTE_SERVER_CFG 配置参数指定的文件。

仅对不提供用户帐户或密码的客户端应用程序使用可信主机信息。如果客户端应用程序提供了无效的帐户名和密码，即使可信主机信息中包含了该客户端计算机条目，数据库服务器仍将拒绝连接。

要使用可信主机信息进行认证，请在 sqlhosts 文件条目中指定 s=1 或 s=3 选项。如果未指定 s 选项，则 s=3 是缺省值。

在 Windows™ 中，可信主机文件位于 \%WINDIR%\system32\drivers\etc 目录中。

在 Linux™ 和 UNIX™ 中， 可信主机文件位于 $INFORMIXDIR/etc/ 目录中。

hosts.equiv 文件包括以下要求：

- 必须为用户 informix 所有
- 属于 informix 组
- 必须限制只有用户 informix 才能修改该文件。使用八进制的权限，以下值之一均可：
  - 644
  - 640
  - 444
  - 440

如果使用 hosts.equiv 文件和 rlogind 守护进程，那么就可以在客户端计算机上执行以下语句来确定客户端是否可信：

```
rlogin hostname
```

如果在没有收到密码提示的情况下成功登录，说明该客户端是可信的。当使用 REMOTE_SERVER_CFG 配置参数指定文件名时，这种用于确定客户端是否可信的方法将失效。

可信主机文件条目

为避免额外的 DNS 查找，请在可信主机文件中指定包含和不包含域名的主机名。例如，如果可信主机名为 host1，且位于 example.com 域中，那么请将以下条目加入可信主机文件：

```
#trustedhost       username
host1              informix
host1.example.com  informix
```

在某些网络中，远程主机用来连接到特定计算机的主机名可能与计算机用于表示其自身的主机名不同。例如，具有完全限定域名 (FQDN) host2.example.com 的网络主机可能使用 viking 作为自己的本地主机名。如果发生这种情形，必须指定两种主机名格式：

```
#trustedhost
host2.example.com
viking
```

使用 REMOTE_SERVER_CFG 配置参数指定的文件而非 hosts.equiv 文件

在以下情形下，使用 REMOTE_SERVER_CFG 配置参数和该参数指定的文件：

- 针对数据库服务器需要的可信主机不同于为操作系统列出的可信主机
- 安装时的安全策略不允许使用 hosts.equiv

- 您是非 root 服务器实例的用户，并且需要控制哪些主机可信

要在 REMOTE_SERVER_CFG 配置参数指定的文件中添加条目，您可以手动输入信息或使用 cdr add trustedhost 参数运行 admin() 或 task() 函数。如果您在高可用性集群的服务器上使用 cdr add trustedhost 参数运行 admin() 或 task() 函数，可信主机信息会添加到集群中所有数据库服务器的可信主机文件中。如果您已在高可用性集群或 Enterprise Replication 域的任一数据库服务器上手动输入了可信主机信息，请不要使用 cdr list trustedhost 参数运行 admin() 或 task() 函数。

相关链接

为高可用性集群配置安全连接 在第298页
sqlhosts 文件和 *SQLHOSTS* 注册表键选项 在第51页
使用连接信息重定向客户端 在第410页
《*SinoDB SQL* 指南: 参考》: *INFORMIXSERVER* 环境变量
《*SinoDB SQL* 指南: 参考》: *DBPATH* 环境变量
《*SinoDB* 管理员参考》: *REMOTE_SERVER_CFG* 配置参数
《*SinoDB* 管理员参考》: *S6_USE_REMOTE_SERVER_CFG* 配置参数
《*SinoDB Enterprise Replication* 指南》: 创建复制服务器的 *sqlhost* 组条目

可信用户信息
用户可以在 .rhosts 文件中列出他们可以作为可信用户进行连接的主机。

.rhosts 文件位于数据库服务器所在计算机上的用户主目录中。 要启用可信用户认证，请在 sqlhosts 条目的选项中指定 s=2 或 s=3。如果未指定 s 选项，则 s=3 是缺省值。

无法使用用户的 .rhosts 文件的原因可能有多种。例如，非 root 安装可能对特定用户的 .rhosts 文件不具有读访问权。通过设置 REMOTE_USERS_CFG 配置参数，可以指定备用文件名。如果设置此参数，数据库服务器只会有一个用于所有用户的可信用户文件。

.rhosts 文件的每一行都是用户可以进行连接的一个主机。必须指定包含和不包含域名的服务器名称，以避免执行额外的 DNS 查找。 例如：

```
#trustedusers
xxx.example.com
xxx

yyy.example.com
yyy
```

由 REMOTE_USERS_CFG 配置参数指定的文件必须是各个 .rhost 文件的组合。 文件中的每个单行条目都具有以下格式：

```
hostname username
```

例如，假设针对用户 John 和 Fred 存在以下两个 .rhosts 文件：

~john/.rhosts

```
#trustedhosts
xxx.example.com
xxx

yyy.example.com
yyy
```

~fred/.rhosts

```
#trustedhosts
xxx.example.com
xxx
```

```
zzz.example.com
zzz
```

John 不信任 zzz.example.com 或 zzz，而 Fred 不信任 yyy.example.com 或 yyy。

这两个 .rhosts 文件可以合并成一个具有以下格式的一个单独文件：

```
#trustedhost      username
xxx.example.com   john
xxx               john

yyy.example.com   john
yyy               john

xxx.example.com   fred
xxx               fred

zzz.example.com   fred
zzz               fred
```

Windows™：在 Windows™ 管理员创建用户身份时，不会自动分配主目录。 管理员可以使用用户管理器应用程序将主目录添加到用户的概要文件中。

相关链接

sqlhosts 文件和 *SQLHOSTS* 注册表键选项 在第51页

使用连接信息重定向客户端 在第410页

《*SinoDB SQL* 指南: 参考》: *INFORMIXSERVER* 环境变量

《*SinoDB SQL* 指南: 参考》: *DBPATH* 环境变量

《*SinoDB* 管理员参考》: *REMOTE_USERS_CFG* 配置参数

《*SinoDB Enterprise Replication* 指南》: 创建复制服务器的 *sqlhost* 组条目

netrc 信息

netrc信息是用于指定标识数据的可选信息。 没有访问数据库服务器的权限的用户或不是数据库服务器所信任的计算机上的用户可以使用此文件提供可信的名称和密码。在远程计算机上拥有不同用户帐户和密码的用户也可以提供此信息。

UNIX™: netrc 信息位于用户的主目录中的 .netrc 文件。 可使用任何标准文本编辑器来准备 .netrc 文件。 netrc 条目的格式为：

```
machine machine_name login user_name password user_password
```

Windows™:使用 setnet32 的 主机信息 选项卡编辑 netrc 信息。

如果您没有在应用程序中为远程服务器明确地提供用户密码 (即，通过 CONNECT 语句的 USER 子句或 DB-Access 中的用户名和密码提示)，客户端应用程序将在 netrc 信息中查找用户名和密码。 如果用户在应用程序中明确地指定密码或者数据库服务器不是远程的，那么将不会参考 netrc 信息。

无论数据库服务器是否使用缺省的认证策略或通信支持模块，它都将使用 netrc 信息。

有关此文件的特定内容的信息，请参阅您的操作系统文档。

仅限 Windows: 在 Windows™ 上，Windows™ 管理员创建用户身份时，不会自动分配主目录。管理员可以使用“用户管理器”应用程序将主目录添加到用户的概要文件中。

相关链接

sqlhosts 文件和 *SQLHOSTS* 注册表键选项 在第51页

用户模仿
对于某些客户端查询或操作行为，数据库服务器必须模仿客户端代表客户端运行进程或程序。

为了模仿客户端，数据库服务器必须得到每个客户端连接的密码。客户端可通过 CONNECT 语句或 netrc 信息提供用户 ID 及密码。

以下示例显示了您可以如何提供密码以模仿客户端。

netrc

```
machine trngpc3 login bruce password im4golf
```

CONNECT 语句

```
CONNECT TO ol_trngpc3 USER bruce USING "im4golf"
```

sqlhosts 文件和 SQLHOSTS 注册表键

SinoDB® 客户端/服务器连接信息（ *sqlhosts* 信息）包含允许客户端应用程序查找并连接到网络上任何 SinoDB® 数据库服务器的信息。

sqlhosts 文件的缺省位置：

UNIX™:

$INFORMIXDIR/etc/sqlhosts

Windows™:

%INFORMIXDIR%\etc\sqlhosts.%INFORMIXSERVER%

如果您将信息存储在其他位置，那么必须设置 INFORMIXSQLHOSTS 环境变量。

如果为使用分布式查询设置了多个数据库服务器，请使用下列方法之一存储所有数据库的 sqlhosts 信息：

- 存储在 INFORMIXSQLHOSTS 环境变量指向的 sqlhosts 文件中
- 在每个数据库服务器目录中存储单独的 sqlhosts 文件

相关链接

使用连接信息重定向客户端 在第410页
网络协议 在第34页
支持多路复用连接 在第35页

使用文本编辑器创建 sqlhosts 文件
每个数据库服务器或客户端的计算机都必须有一个 sqlhosts 文件。

缺省情况下，sqlhosts 文件位于 $INFORMIXDIR/etc 目录中。或者，您可以将 INFORMIXSQLHOSTS 环境变量设置为包含 sqlhosts 信息的文件的完整路径名和文件名。

打开任何标准文本编辑器来创建 sqlhosts 文件。

注:

- 使用空白（空格、制表符或两者）来分隔字段。
- 不要在字段内包含任何空格或制表符。
- 要在 sqlhosts 文件中放置注释，请以注释符 (#) 开始每一行。您还可以将若干行留空以增加可读性。

样本 sqlhosts文件

以下代码块显示样本 sqlhosts 文件。

```
#dbservername    nettype       hostname      servicename      options
 menlo           onipcshm      valley        menlo
 newyork         ontlitcp      hill          dynsrvr2         s=2,b=5120
```

```
payroll         onsoctcp      dewar         py1
asia            group          -             -              e=asia.3
asia.1          ontlitcp      node6         svc8           g=asia
asia.2          onsoctcp      node0         svc1           g=asia
```

使用 Setnet32 设置 SQLHOSTS 注册表键 (Windows™)
客户端应用程序连接到 SinoDB® 数据库服务器，该数据库服务器运行在可通过网络访问的计算机上。要建立此连接，请使用 Setnet32 来指定 SinoDB®数据库服务器在网络上的位置，以及要使用的网络通信协议。您必须从要使用的数据库服务器的管理员处获取此信息。

如果指定共享 SQLHOSTS 注册表键，那么必须将本地计算机上的 INFORMIXSQLHOSTS 环境变量设置为存储注册表的 Windows™ 计算机的名称。数据库服务器首先查找 INFORMIXSQLHOSTS 计算机上的 SQLHOSTS 注册表键。如果数据库服务器在 INFORMIXSQLHOSTS 计算机上找不到 SQLHOSTS 注册表键，或者如果没有设置 INFORMIXSQLHOSTS，那么数据库服务器将在本地计算机上查找 SQLHOSTS 注册表键。

您必须遵循 Windows™ 网络访问约定和文件许可权以确保本地计算机可以访问共享 SQLHOSTS 注册表键。有关网络访问约定和文件许可权的信息，请参阅 Windows™ 文档。

1. 在 Client SDK 产品所在文件夹中双击 Setnet32。

   将打开 SinoDB® Setnet32 窗口。
2. 单击 服务器信息 选项卡以显示 服务器信息页面，其中包含以下元素 ：

   - SinoDB® 服务器

     选择一个现有的 SinoDB® 数据库服务器，或输入一个新的数据库服务器名称。
   - 主机名

     选择您要使用的数据库服务器的主机，或输入新主机的名称。
   - 协议名称

     从安装程序提供的协议列表中选择一个网络协议。
   - 服务名称

     指定与特定数据库服务器关联的服务名称。输入分配给主机上的数据库服务器的服务名称或端口号。您必须从数据库服务器管理员处获取此信息。

     > Requirement: 如果您输入了服务名称，那么它必须定义在客户端上 Windows™ 安装目录的 services 文件中。 此文件位于 system32\drivers\etc\services。服务定义必须与数据库服务器主机上的定义匹配。
   - 选项

     输入特定于数据库服务器的选项。有关更多信息，请参阅 《*SinoDB®* 管理员指南》 。
   - 生成缺省服务器

     将 INFORMIXSERVER 环境变量设置为当前数据库服务器的名称，以使其成为缺省数据库服务器。
   - 删除服务器

     从 Windows™ 注册表中删除数据库服务器的定义。这会同时删除与该数据库服务器关联的主机名、协议名和服务名称。
3. 单击 确定以保存值。

sqlhosts 信息

sqlhosts 信息包含每个数据库服务器的连接信息和组定义。 当启动数据库服务器、客户端应用程序连接到数据库服务器或数据库服务器连接到另一台数据库服务器时，数据库服务器将查找连接信息。

作为数据库服务器或客户端的每台计算机都必须包含连接信息。

在 sqlhosts 文件中，每行包含一个数据库服务器的连接信息或一个组定义。

- 每个数据库服务器的连接信息包含四个必填信息字段和一个可选字段。
- 组定义仅包括三个字段中的信息。

在注册表中，数据库服务器名称将指定给 SQLHOSTS 注册表键中的键，而其他字段是该键的值。

下表总结了用于 SQLHOSTS 信息的字段。

sqlhosts 文件中的字段名称	SQLHOSTS 注册表键中的字段名称	连接信息说明	组信息说明
dbservername	Database server name key 或 database server group key	数据库服务器名称	数据库服务器组名称
nettype	PROTOCOL	连接类型	关键字 group
hostname	HOST	数据库服务器的主机	无信息。在该字段中使用连字符作为占位符
servicename	SERVICE	端口号别名	无信息。在该字段中使用连字符作为占位符
options	OPTIONS	描述或限制连接的选项	组选项

相关链接

定义连接管理的 *sqlhosts* 信息 在第372页
配置连接管理器 在第341页
数据库服务器支持的连接 在第36页
配置 *SinoDB* 数据库服务器以连接 *IBM Data Server* 客户端 在第66页
提高连接管理可用性的策略 在第393页
《*SinoDB* 嵌入性指南》: *syncsqlhosts* 实用程序
为高可用性集群连接管理定义 *sqlhosts* 信息 在第372页
为使用安全端口的高可用性集群的连接管理定义 *sqlhosts* 信息 在第374页
为使用分布式关系数据库结构 *(DRDA)* 的高可用性集群定义 *sqlhosts* 信息 在第376页
为使用分布式关系数据库结构 *(DRDA)* 和安全端口的高可用性集群定义 *sqlhosts* 信息 在第378页
为网格和复制集的连接管理定义 *sqlhosts* 信息 在第381页
为使用安全端口的网格和复制集的连接管理定义 *sqlhosts* 信息 在第382页
为高可用性复制系统的连接管理定义 *sqlhosts* 信息 在第384页
为使用安全端口的高可用性复制系统的连接管理定义 *sqlhosts* 信息 在第387页
为服务器集的连接管理定义 *sqlhosts* 信息 在第390页

sqlhosts.std 文件中的 IANA 标准服务名称及端口号

互联网编号分配机构（IANA）为 SinoDB® 数据库服务器指定服务名称及端口号。

数据库服务器的服务名称及端口号：

端口/服务	IANA 代码	描述
sqlexec	9088/tcp	SinoDB® SQL 接口

这些服务名称在 SinoDB® 的 sqlhosts.std 文件中创建。无需更改已安装的 SinoDB® 系统，因为这些系统将继续使用现有的端口号和服务名称。（此外，无法保证某个其他系统尚未使用分配给 SinoDB® 的服务名称或端口号。）

具有以下标准的策略的组织可以使用这些服务名称和端口号，假如这些组织想要数据库服务器与 IANA 标准保持一致的话。如果安装在同一个工作站的另一个应用程序已使用其中某个服务名称或端口号，那么可以要

求不合规应用程序的发布者注册一个指定的 IANA 端口号，以避免冲突。应用程序不合规时，可以使用非标准端口来运行 SinoDB®。

有关更多信息，请参阅 IANA 组织的 Web 站点。

sqlhosts 连接信息

sqlhosts 文件中的字段或 SQLHOSTS 注册表键描述连接信息。

语法

unique_60_Connect_42_altfrag001

{ *dbservername* { *connection_type* | group } *hostname servicename* [〈选项〉 []] }

元素	用途	限制
dbservername	为连接信息指定的数据库服务器命名。 如果通过 group 关键字而非连接类型进行指定，那么指定一个组以将多个相关数据库服务器条目作为一个逻辑条目进行处理。您可以使用组来建立或更改客户端/服务器连接，或者简化到数据库服务器的连接重定向。	名称必须以小写字母开头，可以包含小写字母、 数字和下划线 (_) 符号。字段长度限制为 128 字节。 数据库服务器必须存在。其名称必须由 onconfig 文件中 DBSERVERNAME 或 DBSERVERALIASES 配置参数指定。 一个数据库服务器组不能嵌套在另一个数据库服务器组中。数据库服务器可以是一个组的成员。
connection_type	描述数据库服务器和客户端应用程序或另一台数据库服务器之间的连接类型。	
hostname	指定数据库服务器所在的计算机。	字段长度限制为 256 字节。 如果指定了 group 关键字，则必须为 null (-)。
servicename	指定端口号的别名。服务名称字段的解释取决于连接类型字段中的连接类型。	字段长度限制为 128 字节。 如果指定了 group 关键字，则必须为 null (-)。

dbservername 字段

所有关联网络中的数据库服务器必须具有唯一的数据库服务器名称。

如果 sqlhosts 文件有多个条目具有相同 dbservername，那么只使用第一个条目。

连接类型字段

连接类型字段在 sqlhosts 文件中称为 nettype，在 SQLHOSTS 注册表键中称为 PROTOCOL。

下表概括了不同操作系统上数据库服务器连接的可能连接类型值。

表 2: 连接类型概括

在 UNIX™ 的值	在 Windows™ 的值	描述	连接类型
drsoctcp	drsoctcp	分布式关系数据库体系结构（DRDA®） — IBM® Data Server 客户端的连接。	网络

在 UNIX™ 的值	在 Windows™ 的值	描述	连接类型
		必须在 sqlhosts 文件或在 SQLHOSTS 注册表中配置使用 drsoctcp 连接协议的新服务器别名。	
drtlitcp	drtlitcp	分布式关系数据库结构（DRDA®）— IBM® Data Server 客户端的连接。 必须在 sqlhosts 文件或在 SQLHOSTS 注册表中配置使用 drtlitcp 连接协议的新服务器别名。	网络
onipcshm		共享内存通信。如果用于服务器和客户端位于不同位置的非 root 安装，那么 sqlhosts 文件中需要使用 cfd 选项。	IPC
onipcstr		流管道通信。如果用于服务器和客户端位于不同位置的非 root 安装，那么 sqlhosts 文件中需要使用 cfd 选项。	IPC
	onipcnmp	命名管道通信	IPC
ontlitcp		使用 TCP/IP 协议的 TLI	网络
onsoctcp	onsoctcp	使用 TCP/IP 协议的套接字	网络
onsqlmux	onsqlmux	多路复用连接	网络

注： 以 "on" 开头的连接类型值可以使用 "ol" 来代替 "on"。例如，如果在 sqlhosts 信息中使用，那么 onipcshm 或 olipcshm 均指共享内存连接。

主机名字段

主机名是在 sqlhosts 文件中的 hostname 字段以及 HOST 注册表键中输入。

如果连接类型为 onsqlmux，那么 hostname 字段绝对不能为空，但其中输入的任何特定值都将被忽略。

以下说明客户端应用程序如何取得主机名字段使用的值。

通过 TCP/IP 进行的网络通信

当使用 TCP/IP 连接协议时，hostname 字段是 hosts 文件中的一个键，用于提供计算机的网络地址。您在 hostname 字段中使用的名称必须与 hosts 文件中的名称相对应。在大多数情况下，hosts 文件中的主机名与计算机的名称相同。

在某些情况下，您可能想要在主机名字段中使用实际的互联网 IP 地址。

UNIX™：共享内存通信和流管道通信

当您将共享内存或流管道用于客户端/服务器通信时，hostname 字段必须包含数据库服务器所在计算机的实际主机名。

多路复用连接

将 onsqlmux 用作连接类型时，hostname 字段必须有一个条目，但该条目将被忽略。连字符（-）可用作条目。

服务名称字段

通过 TCP/IP 进行的网络通信

服务名称字段在 UNIX™ 操作系统上称为 servicename，在 Windows™ 操作系统上称为 SERVICE。使用 TCP/IP 连接协议时，服务名称条目必须与 services 文件中的名称相对应。services 文件中的端口号告诉网络软件如何在指定的主机上查找数据库服务器。

下图显示 sqlhosts 信息和 hosts 文件之间的关系，以及 sqlhosts 信息与 services 文件的关系。

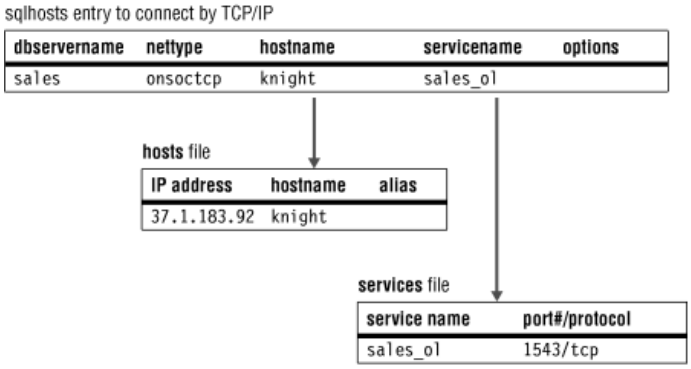

图 4: sqlhosts 信息到 hosts 和 services 文件的关系

在某些情况下，可在服务名称字段中使用实际的 TCP 侦听端口号。

Windows™：命名管道通信

对于命名管道连接（onipcnmp），SERVICE 条目可以是数据库服务器所在主计算机的环境中具有唯一性的任意简短字母组。

UNIX™：共享内存通信和流管道通信

对于共享内存连接（onipcshm）或流管道连接（onipcstr），数据库服务器在内部使用 *servicename* 条目中的值来创建支持连接的文件。对于 onipcshm 和 onipcstr 这两种连接，*servicename* 可以是数据库服务器所在主计算机的环境中具有唯一性的任意简短字母组。

提示： 对于流管道连接，将 *dbservername* 用作 *servicename*。

多路复用连接

对于多路复用连接（onsqlmux），hostname 字段必须有一个条目，但该条目将被忽略。可将连字符（-）用作条目。

相关链接

为高可用性集群配置安全连接 在第298页
连接相关的配置参数 在第61页
组信息 在第57页
流管道连接 *(UNIX 和 Linux)* 在第38页
指定网络连接 在第96页
更改客户端连接信息 在第410页
流管道连接 *(UNIX 和 Linux)* 在第38页
连接相关的配置参数 在第61页
数据库服务器支持的连接 在第36页
TCP/IP 连接文件 在第40页
TCP/IP 连接的备用方法 在第59页

sqlhosts 文件和 SQLHOSTS 注册表键选项
您可以在 sqlhosts 文件或 SQLHOSTS 注册表键中包含服务器选项和组选项。

以下语法分段显示服务器选项。在服务器选项之后说明组选项的语法分段。

重要： 选项必须以逗号分隔，但每个 sqlhosts 条目中列出的第一个选项前面一定不能有逗号。

服务器选项
服务器选项
[a=*token*]
[,b=*size*]
[,cfd=*filepath*]
[,g=*group*]
[,k={ 0 | 1 }]

[,m={ 0 | 1 }]
{ [<查找选项> <PAM 选项>] | [,s = 6] }
查找选项
查找选项
[,r={ 0 | 1 }]
[,s={ 0 | 1 | 2 | 3 }]
PAM 选项
PAM 选项
, s=4, pam_serv =(*name*), pamauth =({ challenge | password })

表 3: sqlhosts 文件和 SQLHOSTS 注册表键中的服务器选项。

元素	用途	限制
a	存储连接到 仓库加速器 所需的认证令牌。此条目由 SinoDB® 在 仓库加速器 的连接设置过程中创建。	重要： 不要手动更改此选项。
b	指定用于 TCP/IP 连接的通信缓冲区空间大小（以字节为单位）。	支持的最大缓冲区大小为 32KB。
cfd	指示共享内存和流管道连接所用的通信文件的存储位置。	cfd 路径长度限制在 70 字节。相对路径字节长度包括 $INFORMIXDIR。
g	指定数据库服务器所属组的名称。	组必须已定义。
k	启用网络服务以定期检查客户端和服务器之间的连接是否仍处于活动状态。如果发现连接中断，网络服务会释放资源。	仅适用于 TCP/IP 连接。
m	使数据库服务器可以创建多个数据库连接，而无需耗用更多网络连接所需的额外计算机资源。	• 不支持多线程客户端连接、共享内存连接，以及与下级数据库服务器的连接。 • 不支持 SinoDB® ESQL/C sqlbreak() 函数。
r	启用对操作系统安全文件查找的控制来控制客户端（用户）获取数据库服务器访问权的方式。s 选项标识数据库服务器端的设置。r 选项标识客户端的设置。	数据库服务器将忽略 r 设置。
s	启用对操作系统安全文件查找的控制来控制客户端（用户）获取数据库服务器访问权的方式。s 选项标识数据库服务器端的设置。r 选项标识客户端的设置。	客户端将忽略 s 设置。
pam_serv	提供数据库使用的 PAM 服务的名称。	必须与 s = 4 选项结合使用。
pamauth	描述 PAM 服务所用的授权方法。	必须与 s = 4 选项结合使用。

以下语法分段显示 sqlhosts 文件中的组选项。

组选项
组选项
[c={ 0 | 1 }]

[,e=*server*]
[,i=*identifier*]

表 4: sqlhosts 文件和 SQLHOSTS 注册表键中的组选项。

元素	用途	限制
c	控制连接重定向。指示客户端应用程序选择数据库服务器或一个数据库服务器组内的别名的顺序。	
e	指定标记数据库服务器组末尾的数据库服务器名称。	
i	指定数据库服务器组的标识号。	该标识号必须是 1 到 32767 之间的正整数且在您的网络环境中必须唯一。 i 选项对于 Enterprise Replication 是必需的。

用法

当更改 sqlhosts 条目中的选项值时，这些更改将影响客户端应用程序建立的下一个连接。服务器将自动识别所做的任何更改。

数据库服务器将 options 条目视为一连串列进行求值。options 条目中的逗号或空格表示一个列的结束。客户端和数据库服务器将检查每一列以确定该选项是否受支持。

您可以在每个条目中以任何顺序放入多个选项。一个 options 条目的最大长度为 256 个字节。

注意： 不受支持或不正确的选项不会触发通知。

缓冲区选项（b）

b 选项仅适用于使用 TCP/IP 连接协议的连接。其他类型的连接将忽略 b 选项。

您可调整缓冲区大小以便更有效地使用系统和网络资源；然而，缓冲区设置得过大，那么用户会因为没有内存可以分配而收到连接拒绝错误。例如，如果您在一个有 1000 位用户的系统上设置 b = 16000，那么该系统可能需要 16 兆字节内存用于通信缓冲区。此设置可能会耗尽计算机的内存资源。使用 TCP/IP 的数据库服务器的缺省缓冲区大小为 4096 字节。

如果您的网络内有多种不同类型的计算机，请谨慎修改通信缓冲区的大小。

提示： 使用通信缓冲区的缺省大小。如果您选择将缓冲区大小设置为其他值，请将客户端通信缓冲区大小和数据库服务器端通信缓冲区大小设置为相同大小。

组连接重定向选项（c）

c 选项仅对服务器组的服务器有效。

将 c 选项用于：

- 平衡多个数据库服务器实例的负载。
- 设置高可用性数据复制（HDR），以便在发生故障的情况下转移到备份数据库服务器。

表 5: 连接重定向选项的设置。

设置	结果
c = 0	这是缺省设置。 客户端应用程序连接到 sqlhosts 信息中服务器组中列出的第一个数据库服务器实例。如果客户端无法连

设置	结果
	接到第一个实例，它将尝试连接到第二个实例，依次类推。
c = 1	客户端应用程序从服务器组中随机选择一个数据库服务器实例为起点来连接。

通信文件目录选项（cfd）

您可以使用通信文件目录选项将共享内存文件或流管道连接通信文件存储到新位置中。如果服务器和客户端位于不同位置，那么必须为 SinoDB® 的非 root 安装指定通信文件目录选项；如果服务器和客户机位于同一位置中，指定此选项将提高系统性能。

cfd 选项可定义绝对路径或相对于 $INFORMIXDIR 的路径以用于存储通信文件：

- cfd=/*location* 定义绝对路径
- cfd=*location* 定义相对于 $INFORMIXDIR 的路径

cfd 路径长度限制为 70 字节。相对路径字节长度包括 $INFORMIXDIR。

SinoDB® 的非 root 安装无写入 /INFORMIXTMP 目录的权限，因此如果在 sqlhosts 信息中未将任何通信文件目录指定为选项，那么共享内存和流管道的连接通信文件将写入 $INFORMIXDIR/etc 目录中。

> 重要： 对于服务器和客户端位于不同位置的 SinoDB® 的非 root 安装必须定义该选项，否则连接将失败。

组末尾选项（e）

如果没有为组指定 e 选项，但是所有 sqlhosts 条目都指定组或组成员，那么网络必须扫描整个文件。可以使用 e 选项指定服务器组的末尾，从而提高系统性能。网络层将扫描 sqlhosts 文件，直至读取到 e 选项指定的条目为止。

如果没有为组指定任何组末尾选项，那么假定组成员是连续的。达到不属于组的条目或处于文件的末尾时（以先达到者为准），即可确定该组的末尾。

在以下示例中，e 选项指定了条目 lx3，所以网络层不会扫描条目 lx4。

```
#dbservername   nettype   hostname   servicename   options
g_x1            group     -          -             i=10,e=lx3
lx1             onsoctcp  apollo11   9810          g=g_x1
lx2             onsoctcp  apollo12   9820          g=g_x1
lx3             onsoctcp  apollo13   9830          g=g_x1

lx4             onsoctcp  apollo14   9840
```

保持活动选项（k）

此选项启用网络服务以定期检查客户端与服务器之间的连接是否仍处于活动状态。 如果连接的接收端未在操作系统的参数所指定的时间内响应，那么网络服务将立即检测到断开的连接并释放资源。

表 6: 保持活动选项设置

设置	结果
k = 0	禁用此服务
k = 1	启用此服务 （缺省）

多路复用选项（m）

此选项使数据库服务器可以创建与客户端应用程序的多个数据库连接，而无需使用更多网络连接所需的额外计算机资源。启用此服务之后，必须重新启动服务器。

表 7: 多路复用选项设置

设置	结果
m = 0	禁用此服务 （缺省）
m = 1	启用此服务

PAM 选项（pam_serv、pam_auth、s = 4）

数据库服务器提供一个接口以使用可插式认证模块（PAM）进行会话认证。要配置此接口，请提供 PAM 服务名称和认证方法。认证可以是连接密码或要求用户回答问题的用户提问。 IBM® Data Server 客户端的分布式关系数据库体系结构 (DRDA®) 连接可以使用密码认证，而不能使用用户提问认证。

SinoDB® PAM 认证调用 pam_authenticate() 和 pam_acct_mgmt() 函数。

表 8: PAM 服务设置

选项	描述	设置
pam_serv	数据库服务器使用的 PAM 服务的名称。	PAM 服务通常位于 /usr/lib/security 目录中，参数列在 /etc/pam.conf 文件中。 在 Linux™ 中，/etc/pam.conf 文件可替换成 /etc/pam.d 目录，每个 PAM 服务在该目录内有一个文件。如果 /etc/pam.d 存在，Linux™ 将忽略 /etc/pam.conf。
pamauth	PAM 服务使用的认证方法。 必须将应用程序设计为正确响应提问提示之后才能连接到数据库服务器。	pamauth=password 使用连接请求密码进行认证。 pamauth=challenge 认证要求用户正确回答问题或提示
s = 4	启用 PAM 认证。	

可信主机和可信用户查找选项（s）

通过这些安全性选项，您可以明确地启用或禁用其中一个或两个文件的使用。

表 9: 可信主机和信任用户查找设置。

设置	结果
s = 0	在 hosts.equiv 或 REMOTE_SERVER_CFG 配置参数指定的文件中禁用可信主机查找。 在 rhosts 文件或由 REMOTE_USERS_CFG 配置参数指定的文件中禁用可信用户查找。 仅接受带有密码的外对内连接。不能用于分布式数据库操作。
s = 1	在 hosts.equiv 或 REMOTE_SERVER_CFG 配置参数指定的文件中启用可信主机查找。

设置	结果
	在 rhosts 文件或由 REMOTE_USERS_CFG 配置参数指定的文件中禁用可信用户查找。
s = 2	在 hosts.equiv 或 REMOTE_SERVER_CFG 配置参数指定的文件中禁用可信主机查找。 在 rhosts 文件或由 REMOTE_USERS_CFG 配置参数指定的文件中启用可信用户查找。 不能用于分布式数据库操作。
s = 3	在 hosts.equiv 或 REMOTE_SERVER_CFG 配置参数指定的文件中启用可信主机查找。 在 rhosts 文件或由 REMOTE_USERS_CFG 配置参数指定的文件中启用可信用户查找。 （缺省）

集群的安全连接选项（s = 6）

sqlhosts 信息中的 s= 6 选项确保集群服务器之间的连接为可信任的。在 sqlhosts 信息中列出的安全端口仅用于集群通信。客户端应用程序不能连接到安全端口。

表 10: 集群的安全连接选项。

设置	结果
s = 6	配置 Enterprise Replication 和高可用性连接安全性。不能与其他任何 s 选项并用。

netrc 查找选项（r）

可使用 r 选项启用或禁用 netrc 查找。

表 11: netrc 查找选项设置。

设置	结果
r = 0	禁用 netrc 查找。
r = 1	启用 netrc 查找（缺省值）

相关链接

组信息 在第57页
可信主机信息 在第43页
可信用户信息 在第44页
netrc 信息 在第45页
支持多路复用连接 在第35页
为高可用性集群配置安全连接 在第298页
《*SinoDB* 安全指南》: 可插拔式认证模块 *(UNIX* 或 *Linux)*
支持多路复用连接 在第35页

组信息

可在 sqlhosts 文件或 SQLHOSTS 注册表键中定义服务器组。创建服务器组时，可以将多个相关的数据库服务器或连接管理器 SLA 条目视为一个条目用于客户端连接，以简化数据库服务器或连接管理器的连接重定向。您必须为参与 Enterprise Replication 的数据库服务器创建组条目。

您可以在以下环境变量或 SQL CONNECT 命令中使用组名称代替数据库服务器名称：

- 客户端应用程序的 INFORMIXSERVER 环境变量值可以是组名称。但是，对于数据库服务器或数据库服务器实用程序，INFORMIXSERVER 环境变量的值不可以使用组名称。
- DBPATH 环境变量值可包含多个组名称。

当指定组的连接信息时，可以在主机名和服务器/端口值中使用连字符（-）字符（ASCII 45）。

高可用性集群组

高可用性集群组的 sqlhosts 为以下格式：

```
#dbservername   nettype   hostname     servicename        options
 group_name     group     -            -                    c=1,e=member_name_n
 member_name_1  protocol  host_name_1  service_or_port_1  g=group_name
 member_name_2  protocol  host_name_2  service_or_port_2  g=group_name
 member_name_n  protocol  host_name_n  service_or_port_n  g=group_name
```

C=1 为可选，指定在组成员列表中选择一个随机起始点用于连接尝试。e=*member_name* 为可选，指定组成员的最终条目，因此不需扫描整个文件。g=*group_name* 选项用于指定该成员所属组，为组成员的必备选项。

Enterprise Replication 服务器组

所有参与复制的数据库服务器必须为数据库服务器组成员。企业中的每一个数据库服务器都需要唯一标识符。Enterprise Replication 节点组具有以下 sqlhosts 格式：

```
#dbservername   nettype   hostname     servicename        options
 group_name_1   group     -            -
 i=identifier_1,e=member_name_1
 member_name_1  protocol  host_name_1  service_or_port_1  g=group_name_1

 group_name_2   -         -
 i=identifier_2,e=member_name_2
 member_name_2  protocol  host_name_2  service_or_port_2  g=group_name_2

 group_name_n   -         -
 i=identifier_n,e=member_name_n
 member_name_n  protocol  host_name_n  service_or_port_n  g=group_name_n
```

i=*identifier* 为 Enterprise Replication 的必备选项。e=*member_name* 为可选，指定组成员的最终条目，因此不需扫描整个文件。g=*group_name* 选项用于指定该成员所属组，为组成员的必备选项。

连接管理器服务级别协议组

连接管理器 SLA 组具有以下 sqlhosts 格式：

```
#dbservername          nettype   hostname   servicename            options
SLA_1_group_name       group     -          -
 c=1,e=SLA_name_1_from_CM_n
SLA_name_1_from_CM_1   protocol  CM_1_host  CM_1_port_or_service_1
 g=SLA_1_group_name
SLA_name_1_from_CM_2   protocol  CM_2_host  CM_2_port_or_service_1
 g=SLA_1_group_name
```

```
SLA_name_1_from_CM_n  protocol  CM_n_host  CM_n_port_or_service_1
 g=SLA_1_group_name

SLA_2_group_name      group     -          -
 c=1,e=SLA_name_2_from_CM_n
SLA_name_2_from_CM_1  protocol  CM_1_host  CM_1_port_or_service_2
 g=SLA_2_group_name
SLA_name_2_from_CM_2  protocol  CM_2_host  CM_2_port_or_service_2
 g=SLA_2_group_name
SLA_name_2_from_CM_n  protocol  CM_n_host  CM_n_port_or_service_2
 g=SLA_2_group_name

SLA_n_group_name      group     -          -
 c=1,e=SLA_name_n_from_CM_n
SLA_name_n_from_CM_1  protocol  CM_1_host  CM_1_port_or_service_n
 g=SLA_n_group_name
SLA_name_n_from_CM_2  protocol  CM_2_host  CM_2_port_or_service_n
 g=SLA_n_group_name
SLA_name_n_from_CM_n  protocol  CM_n_host  CM_n_port_or_service_n
 g=SLA_n_group_name
```

C=1 为可选，指定在组成员列表中选择一个随机起始点用于连接尝试。e=*member_name* 为可选，指定组成员的最终条目，因此不需扫描整个文件。g=*group_name* 选项用于指定该成员所属组，为组成员的必备选项。

相关链接

sqlhosts 连接信息 在第49页
sqlhosts 文件和 *SQLHOSTS* 注册表键选项 在第51页

在 sqlhosts 文件中创建数据库服务器组
通过在 sqlhosts 文件中添加条目来定义组及组成员。

在 sqlhosts 文件中创建数据库服务器组：

1. 添加一个条目来定义数据库服务器组：

   dbservername
   组名称。名称必须以小写字母开头，可包含小写字母、数字及下划线（_）符号。

   nettype
   单词 group。

   hostname
   连字符（-）字符（ASCII 45），用以指示此字段为 null。

   servicename
   连字符（-）字符（ASCII 45），用以指示此字段为 null。

   options
   c、e 或 i 选项（根据情况而定）。

2. 为组成员添加一个或多个条目。包含g=组选项。

示例

以下示例显示了名为 g_asia 的组的定义。该组包含四个成员。

```
#dbservername  nettype   hostname  servicename  options
g_asia         group     -         -            c=1,e=manilla
tokyo          onsoctcp  node_1    service_1    g=g_asia
beijing        onsoctcp  node_2    service_2    g=g_asia
seoul          onsoctcp  node_3    service_4    g=g_asia
```

```
manilla        onsoctcp  node_4    service_5    g=g_asia
```

TCP/IP 连接的备用方法

以下主题将说明一些绕过 TCP/IP 连接的端口及 IP 地址查找的方法。

TCP/IP 连接的 IP 地址

对于 TCP/IP 连接（TLI 和套接字），您可以在 hostname 字段中使用实际 IP 地址，而不是使用 hosts 文件中的主机名或别名。以下示例显示 hosts 文件中的样本 IP 地址和主机。

```
#address       hostname  alias
555.12.12.12   smoke
98.555.43.21   odyssey
12.34.56.555   knight    sales
```

如果对该表中的 knight 使用 IP 地址，以下两个 sqlhosts 条目是等效的：

```
#dbservername   nettype    hostname       servicename   options
sales           ontlitcp   12.34.56.555   sales_ol
```

```
#dbservername   nettype    hostname       servicename   options
sales           ontlitcp   knight         sales_ol
```

在某些情况下，使用 IP 地址可能加快连接时间。 然而，计算机通常以其主机名来标识，因此在主机名字段中使用 IP 地址将不便于标识与条目相关联的计算机。

UNIX: 您可以在 hosts 文件的网络地址字段中查找 IP 地址，或者使用 UNIX™ 的 arp 或 ypmatch 命令来查找 IP 地址。

Windows: 可以将 Windows 配置为使用以下任一机制将域解析为 IP 地址：

- Windows™ 互联网名称服务
- 域名服务器

TCP/IP 连接的通配符寻址

当同时满足以下两个条件时，可以在 hosts 文件的 hostname 字段中使用通配符寻址：

- 您使用的是 TCP/IP 连接。
- 数据库服务器所在的计算机使用多块网络接口卡 (NICs)。

如果上述条件都满足，您可以在数据库服务器使用的 hostname 字段中使用星号 (*) 作为通配符。当您在 hostname 字段中输入一个通配符时，数据库服务器可以接受其主计算机上任何有效 IP 地址的连接。

每个 IP 地址关联到一个唯一的主机名。当计算机使用多块 NICs（如以下表所示），那么在 hosts 文件中对于每一个接口卡都必须有一个条目。例如， 使用双 NICs 的 texas 计算机的 hosts 文件可能包含以下条目：

```
#address       hostname   alias
123.45.67.81   texas1
123.45.67.82   texas2
```

如果客户端应用程序与数据库服务器分享 sqlhosts 信息，那么可以在 hostname 字段中同时指定通配符和主机名或 IP 地址（例如，*texas1 或 *123.45.67.81）。 客户端应用程序会忽略通配符，并使用主机名（或 IP 地址）来建立连接，数据库服务器则使用通配符来接受来自任何 IP 地址的连接。

通配符格式允许数据库服务器侦听线程在所有有效网络接口卡上使用相同的服务端口号等待客户端连接。然而，等待多个 IP 地址的连接可能比等待特定主机名或 IP 地址的连接所需的处理器时间更长。

下图显示具有双网络接口卡的计算机 texas 上的数据库服务器。两个客户端使用不同网络卡与数据库服务器通信。

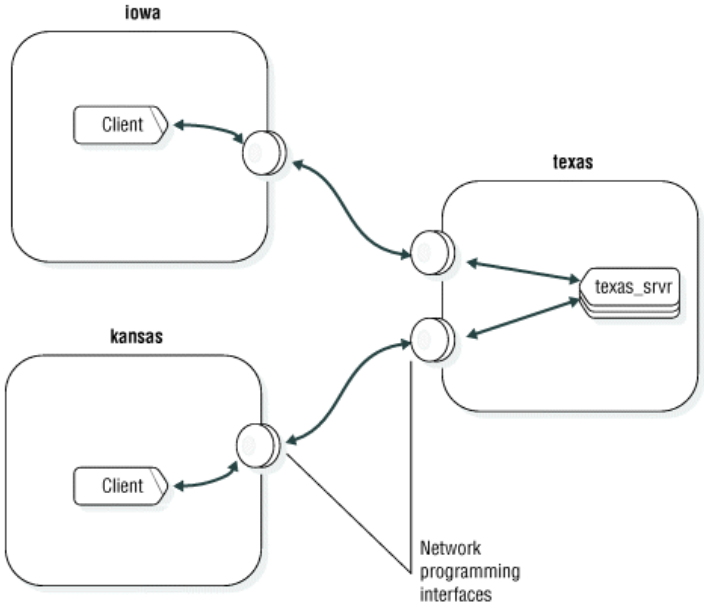

图 5: 使用多块网络接口卡

以下示例显示 texas_srvr 数据库服务器可能的 sqlhosts 连接信息。

```
#dbservername   nettype    hostname        servicename   options
texas_srvr      ontlitcp   *texas1         pd1_on
```

```
#dbservername   nettype    hostname        servicename   options
texas_srvr      ontlitcp   *123.45.67.81   pd1_on
```

```
#dbservername   nettype    hostname        servicename   options
texas_srvr      ontlitcp   *texas2         pd1_on
```

```
#dbservername   nettype    hostname        servicename   options
texas_srvr      ontlitcp   *123.45.67.82   pd1_on
```

```
#dbservername   nettype    hostname        servicename   options
texas_srvr      ontlitcp   *               pd1_on
```

如果连接信息与前面任意一行的内容一致，那么 texas_srvr 数据库服务器就可以接受来自任一网络卡的客户端连接。数据库服务器在 hostname 字段中发现通配符，就会忽略显式的主机名。

> 提示: 为了清晰简便地维护，在主机名字段中使用通配符时，请含括主机名（也就是使用 *host 而非 *）。

客户端应用程序所使用的连接信息必须包含显式的主机名或 IP 地址。在 iowa 上的客户端应用程序可使用以下任一主机名： texas1、*texas1、123.45.67.81 或 *123.45.67.81。如果 hostname 字段中有通配符（*），客户端应用程序将忽略。

在 kansas 上的客户端应用程序可使用以下任一主机名：texas2、*texas2、123.45.67.82 或 *123.45.67.82。

TCP/IP 连接的端口号

对于 TCP/IP 网络协议，您可以在服务名称字段中使用实际的 TCP 侦听端口号。

例如，如果在 services 文件中 sales 数据库服务器的端口号为 1543，那么您可以在 sqlhosts 文件中写入如下条目：

```
#dbservername   nettype    hostname   servicename   options
sales           ontlitcp   knight     1543
```

在某些情况下，使用实际端口号可能节省连接建立的时间。但是，正如 hostname 字段中的 IP 地址一样，使用实际端口号可能会不便于连接信息的管理。

相关链接

sqlhosts 连接信息 在第49页

SinoDB® 对 IPv6 地址的支持

在所有平台上，SinoDB® 都可识别互联网协议第六版 (IPv6) 的地址（长度为128位）与 互联网协议第四版 (IPv4) 的地址（长度为32位）。

数据库服务器在启动时会检查底层操作系统是否支持 IPv6。如果支持 IPv6，就会使用 IPv6。如果底层操作系统不支持 IPv6，就会使用 IPv4 地址。SinoDB® 和 Client SDK 从名称服务中检索 IP 地址。

在同时具有 IPv4 和 IPv6 地址的主机上运行的 SinoDB® 可以使用与多宿主主机上运行的服务器相同的方式来处理。可使用以下任一方式在同时具有 IPv4 和 IPv6 地址的主机上配置 SinoDB®：

- 创建别名（使用 DBSERVERALIASES 配置参数），并分配 IPv6 地址给其中一个，将 IPv4 地址分配给另一个。
- 通过在 sqlhosts 文件中使用带通配符的主机名，指示 SinoDB® 监听主机上配置的所有 IP 地址。

  例如：

  ```
  #dbservername   nettype    hostname   servicename   options
  olserver1       onsoctcp   *myhost    onservice1
  ```

  如果已对主机配置了 IPv6 地址，那么 SQLHOSTS 文件中 hostname 条目将映射到 IPv6。如果未对主机配置 IPv6 地址，那么 hostname 条目将映射到 IPv4 地址。

禁用 IPv6 支持

SinoDB® 也提供了在 IPv4 环境中禁用 IPv6 支持的方法。

禁用所有数据库实例和客户端应用程序的 IPv6 支持：

- 创建空文件 $INFORMIXDIR/etc/IFX_DISABLE_IPV6。

用户 informix 必须具有该文件的读取权限。该文件不需要读写，也不需要包含任何数据。

对单一数据库实例或单一客户端应用程序禁用 IPv6 支持：

- 在数据库服务器实例上，或在应用程序运行的工作站上，创建名为 IFX_DISABLE_IPV6 的环境变量，并将其值设置为 yes，如下所示：

  ```
  IFX_DISABLE_IPV6=yes
  ```

连接相关的配置参数

onconfig 文件中的某些配置参数用于指定连接相关的信息。

当重启数据库服务器时，重启程序将使用这些配置参数中设置的值。

以下为与连接相关的配置参数：

- DBSERVERNAME

- DBSERVERALIASES
- LIMITNUMSESSIONS
- NETTYPE
- NS_CACHE
- NUMFDSERVER
- HA_ALIAS

UNIX: 配置连接时，也要考虑设定 LISTEN_TIMEOUT 与 MAX_INCOMPLETE_CONNECTIONS 配置参数。这些参数能降低恶意拒绝服务 (DOS) 攻击的风险，使得覆盖处理连接的 Listener 虚拟处理器更加困难。更多信息，请参阅《*SinoDB*® 安全指南》。

相关链接

sqlhosts 连接信息 在第49页
数据库服务器支持的连接 在第36页
sqlhosts 连接信息 在第49页

DBSERVERNAME 配置参数中设置的连接信息

客户端应用程序连接到数据库服务器时，必须指定该数据库服务器的名称。与指定的数据库服务器名称关连的 sqlhosts 信息描述了应用程序和数据库服务器之间的连接类型。

例如，要指定数据库服务器的名称为 nyc_research，请在 onconfig 文件或 Windows™ 注册表键中设置 DBSERVERNAME 值：

```
DBSERVERNAME nyc_research
```

客户端应用程序可以在以下其中一个位置指定数据库服务器名称：

- INFORMIXSERVER 环境变量中
- 在用于指定数据库环境的 SQL 语句（例如： CONNECT、DATABASE、CREATE TABLE 和 ALTER TABLE）中
- DBPATH 环境变量中

DBSERVERNAME 必须指定数据库服务器名称，或指定数据库服务器的其中一个别名。名称务必以小写字母开头，且可以包含其他小写字母、数字及下划线。名称不能包含大写字符、字段定界符（空格或制表符）或换行符。基本 ASCII 代码集的其他字符未必可靠。例如，连字符或减号可能会产生问题，冒号也可能不能使用。@ 字符为保留字符，用于分隔数据库和服务器（例如，dbase@server）。

对于 onimcsoc 或 onsoctcp 协议，可更新 DBSERVERNAME 配置参数，以便在 sqlhosts 信息中包含数据库服务器别名的多个侦听线程的数量，如下所示：

```
DBSERVERNAME name-number_of_multiple_listen_threads
```

相关链接

《*SinoDB* 管理员参考》: *DBSERVERNAME* 配置参数

DBSERVERALIASES 配置参数中设置的连接信息

通过 DBSERVERALIASES 配置参数可以对同一数据库服务器指定额外的数据库服务器名称。

最大的别名数为 32。 以下示例显示 onconfig 配置文件中指定三个数据库服务器名称给同一数据库服务器实例的条目设置。

```
DBSERVERNAME          sockets_srvr
DBSERVERALIASES       ipx_srvr,shm_srvr
```

因为每一个数据库服务器名称有一个对应的 sqlhosts 条目，因此，可以将一个数据库服务器与多个连接类型相关连。

```
shm_srvr          onipcshm     my_host            my_shm
sockets_srvr      onsoctcp     my_host            port1
```

```
ipx_srvr        ontlispx     nw_file_server     ipx_srvr
```

使用上述示例中的 sqlhosts 文件，客户端应用程序可以使用以下语句连接到使用共享内存通信的数据库服务器：

```
CONNECT TO '@shm_srvr'
```

客户端应用程序可以使用以下语句对相同的数据库服务器发出 TCP/IP 套接字连接：

```
CONNECT TO '@sockets_srvr'
```

DBSERVERALIASES 务必以小写字母开头，且可包含其他小写字母、数字及下划线。DBSERVERALIASES 不可包含大写字符、字段定界符（空格或制表符）或换行符。基本 ASCII 代码集的其他字符未必可靠。例如，连字符或减号可能会产生问题，冒号也可能不能使用。@ 字符为保留字符，用于分隔数据库和服务器（例如，dbase@server）。

在上述示例中， @shm_srvr 语句连接到该服务器上未识别的数据库上；或者，也可以连接到 dbase1@shm_srvr。

对于 onimcsoc 或 onsoctcp 协议，可更新 DBSERVERALIASES 配置参数，以便在 sqlhosts 信息中包含数据库服务器别名的多个侦听线程的数量。

```
DBSERVERALIASES name-number,name-number
```

相关链接

《*SinoDB* 管理员参考》：*DBSERVERALIASES* 配置参数

LIMITNUMSESSIONS 配置参数中设置的连接信息

LIMITNUMSESSIONS 配置参数是一个可选参数，用于指定连接至 SinoDB® 的最大会话数。如果指定了最大值，那么也可以指定在会话数接近最大值时是否让SinoDB® 将信息打印至 online.log 文件里。

针对服务器进行的分布式查询计数受其限制。

如果数据库服务器即将达到极限值时，您可能需要动态增加或暂时关闭 LIMITNUMSESSIONS 配置参数，以便允许管理实用程序运行。使用 onmode -wf 或 onmode -wm 来动态增加或关闭 LIMITNUMSESSIONS。

如果启用了 LIMITNUMSESSIONS 配置参数，并且由于此限制限制了会话数，那么连接到任何数据库的一般用户线程计数和 DBSA 用户线程计数都会受其限制。然而，即使达到了该限制值，还是允许 DBSA 用户连接到该服务器。

LIMITNUMSESSIONS 配置参数不应用作为遵守许可协议的方法。

> 示例
>
> 以下示例指定了连接至数据库服务器的最大会话数为 100，并且在连接的会话数接近 100 时打印警告消息：LIMITNUMSESSIONS 100,1

NETTYPE 配置参数中设置的连接信息

NETTYPE 配置参数可调整数据库服务器用于通信的虚拟处理器的数量和类型。每种类型的网络连接（例如，ipcshm 或 soctcp）在配置文件中具有单独的 NETTYPE 条目。

> Recommendation: 虽然 NETTYPE 参数并非必需参数，但是如果使用两种或更多连接类型，那么必须设置 NETTYPE。当数据库服务器运行一段时间后，您可以使用 NETTYPE 配置参数来调整数据库服务器以获得更好的性能。

有关 NETTYPE 的更多信息，请参阅网络虚拟处理器 在第96页。有关 NETTYPE 配置参数的信息，请参阅《*SinoDB*® 管理员参考》。

在 NS_CACHE 配置参数中设置的名称服务最大保留时间

NS_CACHE 配置参数定义以下对象中单个条目的最大保留时间：主机名/ IP 地址高速缓存、 服务高速缓存、用户高速缓存和组高速缓存。如果指定最大保留时间，数据库服务器可以从高速缓存中获得主机、服务、用户和组数据库服务器的信息。

每个高速缓存在为特定高速缓存重配时间或该时间重新配置之后过期。

通常网络名称服务提供者（例如 DNS）是位于远程计算机上。为了避免花费时间从网络名称服务提供者返回信息，您可以使用 NS_CACHE 配置参数来指定从某一内部高速缓存获取信息的最大保留时间。SinoDB® 将在高速缓存中查找信息。如果该处没有想要的信息，数据库服务器将查询操作系统以获取信息。

使用 SinoDB® 名称服务高速缓存机制可避免许多操作系统中的查找，从而在可配置的时间内保留并复用检索到的每个信息段。

数据库服务器从高速缓存中获取信息的速度比查询操作系统更快。如果通过把保留时间设置为 0 来禁用一个或多个高速缓存，那么数据库服务器将查询操作系统以获取主机、服务、用户或组信息。

作为 DBA，您可能希望在以下情况下修改 NS_CACHE 配置参数的设置：网络名称服务提供者是运行在远程计算机上，或 MSC VP 运行时的处理器使用量很大。

例如，运行 onstat -g glo 命令，以在输出的 Individual virtual processors 部分中检查 msc VP 的使用情形。在以下的输出样本中，在 usercpu 和 syscpu 列中显示的 msc 处理器使用量非常高。如果怀疑使用量高是因为 DNS 调用花费了过多时间，那么可以使用操作系统命令来确认高使用量，然后修改 NS_CACHE 配置参数的设置。

```
Individual virtual processors:
vp    pid       class       usercpu   syscpu    total     Thread    Eff
1     2036      cpu         76.95     7.14      84.09     99.08     84%
2     2149      adm         0.00      0.00      0.00      0.00       0%
3     2151      LIC         0.00      0.00      0.00      0.00       0%
4     2260      lio         0.00      0.00      0.00      0.03       0%
5     2442      pio         0.00      0.00      0.00      0.00       0%
6     2443      aio         0.00      0.01      0.01      0.11       8%
7     2444      msc         14.18     14.64     28.82     199.91    14%
8     2446      fifo        0.00      0.00      0.00      0.00       0%
```

您可能也希望在以下情况下指定 NS_CACHE 信息：操作系统没有名称服务 (NS) 高速缓存，或禁用了操作系统 NS 高速缓存。

> 示例
>
> 要将主机和服务连接的最大保留时间定义为 600 秒，并对用户和组数据库服务器连接禁用最大保留限制，请指定：
>
> ```
> NS_CACHE host=600,service=600,user=0,group=0
> ```

NUMFDSERVERS 配置参数中设置的连接信息

对于 UNIX™ 上的网络连接，可以使用 NUMFDSERVERS 配置参数指定处理 SinoDB® 虚拟处理器 (VPs) 之间迁移的网络连接所需的最大轮询线程数。

在以下情况下指定 NUMFDSERVERS 信息将非常有用： SinoDB® 的新连接和断开连接的请求比率很高，或网络共享文件 (NSF) 锁之间存在大量的争用。

相关链接

《SinoDB 管理员参考》: NUMFDSERVERS 配置参数

HA_ALIAS 配置参数中设置的连接信息

HA_ALIAS 配置参数定义网络别名，用于高可用性集群中服务器到服务器的通信。连接管理器、ifxclone 实用程序和 onmode -d 命令也可使用设置的网络别名。

为高可用性集群中的每一个服务器设置 HA_ALIAS 配置参数。对于使用共享内存连接的高可用性集群服务器，HA_ALIAS 配置参数为必需的。

HA_ALIAS 配置参数值必须与 sqlhosts 文件中 TCP 条目关联的 DBSERVERNAME 或 DBSERVERALIASES 配置参数相匹配。

相关链接

《SinoDB 管理员参考》: HA_ALIAS 配置参数
《SinoDB 管理员参考》: onmode -d: 设置数据复制类型
《SinoDB 管理员参考》: DBSERVERALIASES 配置参数
《SinoDB 管理员参考》: DBSERVERNAME 配置参数

网络连接的环境变量

INFORMIXCONTIME（连接时间）以及INFORMIXCONRETRY（连接重试）环境变量在客户端试图连接数据库服务器时会影响客户端行为。使用环境变量可使网络流量繁忙导致的连接错误最小化。

若客户端应用程序显式连接到共享内存段，那么您可能需要设置INFORMIXSHMBASE（共享内存库）。

可以使用 INFORMIXSERVER 环境变量来指定客户端连接到的缺省数据库服务器名称。

相关链接

数据库服务器支持的连接 在第36页
客户端如何连接到通讯部份 *(UNIX)* 在第108页
《SinoDB SQL 指南: 参考》: INFORMIXCONTIME 环境变量
《SinoDB SQL 指南: 参考》: INFORMIXCONRETRY 环境变量
《SinoDB SQL 指南: 参考》: INFORMIXSHMBASE 环境变量 (UNIX)
《SinoDB SQL 指南: 参考》: INFORMIXSERVER 环境变量

自动终止空闲连接

通过启用 idle_user_timeout 调度程序任务来自动结束与用户间已经空闲了特定时间的会话。

您必须以用户 informix 或另一个授权用户身份连接到 sysadmin 数据库。

若要启用 idle_user_timeout 任务，请执行下面的语句：

```
UPDATE ph_task
   SET tk_enable = 't';
   WHERE tk_name = 'idle_user_timeout';
```

默认情况下， idle_user_timeout 任务终止已空闲超过 60 分钟的用户会话。属于用户 informix 的会话是不会被终止的。Idle_user_timeout 任务在两小时后开始检查空闲会话，这是任务的默认频率。

> 提示: 当数据库服务器计算机上的系统时间更改时，用户会话空闲的时间量不再准确。例如，如果一个用户会话最后工作是在下午3:14，在下午 3:15 时系统时钟往前调整了一个小时，那么在数据库服务器查看时将发现用户会话已闲置了一个多小时。

若要更改空闲超时时间，您需要更新任务的的运行频率及阈值。所允许的最短空闲超时时间是 5 分钟。例如，要将超时时间设置为 5 分钟，需要执行下面的语句：

```
UPDATE ph_task
  SET tk_frequency = INTERVAL (5) MINUTE TO MINUTE
```

```
  WHERE tk_name = 'idle_user_timeout';

UPDATE ph_threshold
   SET value = '5'
   WHERE task_name = 'idle_user_timeout';
```

Distributed Relational Database Architecture™ (DRDA®) 通信

DRDA是支持多个数据库系统和应用程序一起工作的一组协议。

本节包含有关如何配置 SinoDB® 来使用 Distributed Relational Database Architecture™ (DRDA®) 的信息。

DRDA®概述

分布式关系数据库体系结构（DRDA®）是一组协议，使不同平台上的应用程序和数据库系统之间的可以通信，并且能够将关系数据分布在多个平台之间。

使用 DRDA® 的关系数据库管理产品的任意组合，都可连接形成一个分布式关系数据库管理系统。DRDA® 通过定义交换内容和交换方式来协调各系统之间的通信。

可以将数据库服务器配置为使用 DRDA® 以响应公共 API （例如：IBM® Data Server JDBC Driver 或 IBM® Data Server.NET Provider）的请求。

连接管理器支持 DRDA®，因此可以使用连接管理来将客户端的连接请求重定向到合适的数据库服务器上。对于使用 DRDA® 的高可用性集群，连接管理器还可提供自动故障转移功能。

Enterprise Replication、数据复制和 SinoDB® 实用程序（例如 DB-Access）需要标准的 SinoDB® 连接。Enterprise Replication 实用程序不能在 DRDA® 连接上运行。但是，Enterprise Replication 连接可以与 DRDA® 连接共存。

当在 ANSI 兼容的数据库上使用 DRDA® 时，无缓冲的日志记录和隐式事务是强制执行的。如果将基于非 ANSI 兼容的数据库的应用程序迁移到 DRDA® 环境，则连接必须处理需要事务的语句的应用程序逻辑。例如，在存储过程中的串联运算符之前必须有 BEGIN WORK 语句。

可以通过可插式认证模块的密码验证来确保公共客户端 API 与 SinoDB® 之间的DRDA® 连接。

相关链接

《SinoDB 数据库设计和实现指南》: 事务

《SinoDB SQL 指南: 语法》: BEGIN WORK 语句

通过连接管理器进行连接管理 在第340页

配置 SinoDB® 数据库服务器以连接 IBM® Data Server 客户端

要将 IBM® Data Server 客户端连接至 SinoDB®，请必须按照以下步骤进行配置。

首先必须安装好IBM® Data Server 客户端和适当的驱动程序。

1. 在每个连接管理器和数据库服务器主机上，为每个服务器添加 sqlhosts 文件条目。
   例如：

```
#dbservername    nettype     hostname    servicename   options
 server_1        onsoctcp    host_1      port_1
 server_2        onsoctcp    host_2      port_2
 server_3        onsoctcp    host_3      port_3
```

2. 在每个数据库服务器的 onconfig 文件中，设置 DBSERVERALIASES 参数指定服务器的别名。

   server_1 的 onconfig 文件条目：

```
DBSERVERALIASES drda_1
```

server_2 的 onconfig 文件条目：

```
DBSERVERALIASES drda_2
```

server_3 的 onconfig 文件条目：

```
DBSERVERALIASES drda_3
```

3. 在每个连接管理器的主机上，为 DRDA® 别名添加 sqlhosts 文件条目。指定 drtlitcp 或 drsoctcp 协议，并指定 DRDA® 通信的端口。
例如：

```
#dbservername  nettype    hostname   servicename   options
 server_1      onsoctcp   host_1     port_1
 server_2      onsoctcp   host_2     port_2
 server_3      onsoctcp   host_3     port_3

 drda_1        drsoctcp   host_1     drda_port_1
 drda_2        drsoctcp   host_2     drda_port_2
 drda_3        drsoctcp   host_3     drda_port_3
```

4. 在每个连接管理器的主机上，为数据库服务器组添加组条目，并为 DRDA® 别名组添加组条目。为数据库服务器和DRDA®别名条目添加组选项。使用 c=1 组条目选项，以便组成员列表中的连接尝试起点是随机的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。
例如：

```
#dbservername   nettype    hostname   servicename   options
 my_servers     group      -          -             c=1,e=server_3
 server_1       onsoctcp   host_1     port_1        g=my_servers
 server_2       onsoctcp   host_2     port_2        g=my_servers
 server_3       onsoctcp   host_3     port_3        g=my_servers

 drda_aliases   group      -          -             c=1,e=drda_3
 drda_1         drsoctcp   host_1     port_4        g=drda_aliases
 drda_2         drsoctcp   host_2     port_5        g=drda_aliases
 drda_3         drsoctcp   host_3     port_6        g=drda_aliases
```

5. 在连接管理器配置文件中，添加 DRDA® 服务级别协议。
例如：

第一个连接服务器配置文件具有以下条目：

```
NAME connection_manager_1

CLUSTER my_cluster
{
   INFORMIXSERVER my_servers
   SLA sla_primary_1          DBSERVERS=PRI
   SLA sla_primary_drda_1     DBSERVERS=PRI
   SLA sla_secondaries_1      DBSERVERS=SDS,HDR
   SLA sla_secondaries_drda_1 DBSERVERS=SDS,HDR
   FOC ORDER=ENABLED \
       PRIORITY=1
}
```

第二个连接服务器配置文件具有以下条目：

```
NAME connection_manager_2

CLUSTER my_cluster
{
```

```
    INFORMIXSERVER my_servers
    SLA sla_primary_2           DBSERVERS=PRI
    SLA sla_primary_drda_2      DBSERVERS=PRI
    SLA sla_secondaries_2       DBSERVERS=SDS,HDR
    SLA sla_secondaries_drda_2 DBSERVERS=SDS,HDR
    FOC ORDER=ENABLED \
        PRIORITY=2
}
```

6. 在每个 IBM® Data Server 客户端的主机上，为每个连接管理器配置文件的每个服务级别协议（SLA）创建 sqlhosts 文件条目。为每组 SLA 条目创建组条目，并且将组选项添加至 SLA 条目。

   例如：

```
#dbservername             nettype    hostname   servicename  options
 g_primary                group      -          -            c=1,e=sla_primary_2
 sla_primary_1            onsoctcp   cm_host_1  cm_port_1    g=g_primary
 sla_primary_2            onsoctcp   cm_host_2  cm_port_2    g=g_primary

 g_secondaries            group      -          -            c=1,e=sla_secondaries_2
 sla_secondaries_2        onsoctcp   cm_host_1  cm_port_3    g=g_secondaries
 sla_secondaries_2        onsoctcp   cm_host_2  cm_port_4    g=g_secondaries

 g_primary_drda           group      -          -            c=1,e=sla_primary_2_drda
 sla_primary_1_drda       drsoctcp   cm_host_1  cm_port_5    g=g_primary_drda
 sla_primary_2_drda       drsoctcp   cm_host_2  cm_port_6    g=g_primary_drda

 g_secondaries_drda       group      -          -            c=1,e=sla_secondaries_2_drda
 sla_secondaries_2_drda   drsoctcp   cm_host_1  cm_port_7    g=g_secondaries_drda
 sla_secondaries_2_drda   drsoctcp   cm_host_2  cm_port_8    g=g_secondaries_drda
```

在前面的示例中，IBM® Data Server 客户端至 @g_primary_drda 的连接请求，是通过 drsoctcp 协议发送给其中一个连接管理器。连接管理器接到请求后，使用 SLA 将主服务器的连接信息提供客户端应用程序。

如果通过 DRDA® 协议访问该服务器时收到错误 -23104，那么客户端应用程序可能正在尝试绑定一个值，其编码与数据库语言环境的代码页或代码集不同。在启动 SinoDB® 实例之前，将 GL_USEGLU 环境变量设为 1。此设置使服务器能够初始化适当的 Unicode 转换器，以处理代码集转换。

相关链接

sqlhosts 信息 在第47页
为使用分布式关系数据库结构 *(DRDA)* 的高可用性集群定义 *sqlhosts* 信息 在第376页
为使用分布式关系数据库结构 *(DRDA)* 和安全端口的高可用性集群定义 *sqlhosts* 信息 在第378页
《*SinoDB GLS* 用户指南》: *GL_USEGLU* 环境变量

用NETTYPE配置参数为接口/协议组合分配轮询线程

NETTYPE配置参数为您数据库服务器实例支持的每种连接类型配置轮询线程。您可以使用该配置参数为每个接口/协议组合分配多个轮询线程。

按以下步骤设置 NETTYPE 配置参数：

1. 指定SQLI、drtlitcp 或 drsoctcp 作为连接协议。
2. 添加有关轮询线程数、连接数和虚拟处理器类的信息。

例如，指定：

```
NETTYPE drtlitcp,3,2,CPU
```

一个 NETTYPE 条目可处理具有同一协议类型的多个数据库服务器别名。因此，当使用 DRDA® 时，网络监听器线程（NETTYPE drtlitcp 或 drsoctcp）通常具有至少两个套接字打

开监听连接。 一个套接字打开用于 SQLI 连接，另一套接字打开用于 DRDA® 连接。如果配置了许多独立的服务器别名，那么可能打开更多的套接字。

相关链接

《*SinoDB* 管理员参考》: *NETTYPE* 配置参数

使用 DRDA_COMMBUFFSIZE 配置参数指定DRDA® 通信缓冲区的大小

使用 DRDA_COMMBUFFSIZE 配置参数指定 DRDA® 通信缓冲区的大小。最小为 4 KB，最大为 2 MB，缺省值为 32 KB。

您可以将一兆字节缓冲区指定为 1M、1m、1024K、1024k 或 1024。SinoDB®自动将小于 4 KB 的值重置为 32 KB。

当建立 DRDA® 会话时，该会话将分配当前缓冲区大小的通信缓冲区。

您可以使用 isgetdrdacommbuffsize() 函数来返回 DRDA_COMMBUFFSIZE 的当前值。

当数据库服务器正在运行时，您不可以使用 onmode -wm 命令来更改设置。

DRDAEXEC 线程与客户端查询

对于每个 DRDA® 客户端，SinoDB® 创建一个会话以及一个 DRDAEXEC 线程（该线程等同于 SQLEXEC 线程）以处理和运行查询。该线程还会格式化 DRDA® 协议格式的查询结果，并将结果返回到客户端计算机。

如果设置了 PDQPRIORITY， 那么将可以并行运行 DRDA® 客户端发出的多个查询，以及并行运行查询操作。从 DRDAEXEC 线程运行的查询也可以并行运行。

SQL与受支持和不受支持的数据类型

使用 DRDA® 时，SinoDB® 语法在公共 API 上受支持。使用 DRDA® 连接时，对于 decimal 和 money 数据类型的所有数据检索操作，SinoDB® 会将 decimal 和 money 的值舍入为 32 位精度。

以下数据类型在公共 API 上受支持：

- BIGINT
- BIGSERIAL
- BLOB
- BOOLEAN
- BSON
- BYTE
- CHAR(32k)
- CLOB
- DATE
- DATETIME
- DECIMAL
- FLOAT
- INT
- INT8
- INTERVAL
- JSON
- LVARCHAR(32k)
- MONEY
- NCHAR(32k)
- NVARCHAR(255)
- SERIAL
- SERIAL8
- SMALLFLOAT

- SMALLINT
- TEXT
- VARCHAR(255)

SinoDB® DATETIME 值映射到 DATE、TIME 或 TIMESTAMP 值。

以下数据类型支持与数据库服务器主机变量一起使用：

- CHAR
- DATE
- INT
- SMALLINT
- VCHAR

显示 DRDA® 连接信息

使用 onstat 和 onmode 命令显示包括 DRDA® 线程名称的信息，以及区分 SQLI 和 DRDA® 会话的指示符。

以下命令显示有关线程名称和会话的信息：

- onstat -g ses
- onstat -g sql
- onstat -g ath
- onstat -g stk
- onstat -u
- onstat -x
- onstat -G
- onstat -g ddr
- onstat -g env
- onstat -g stm
- onstat -g ssc
- onmode -D
- onmode -Z

例如，onstat 输出可能将 "drdaexec" 显示为线程名称。

显示DRDA®会话信息

使用 sysmaster 数据库中的 syssesappinfo 表可查看 DRDA® 客户端会话信息。

该表在 sesapp_sid、sesapp_name 和 sesapp_value 列中显示客户端会话 ID、会话应用程序名称，以及会话值。

例如，该表可能显示以下信息：

- sesapp_sid：6
- sesapp_name：Accting
- sesapp_value：db2jcc_application

也可以使用 onstat -g ses 命令来显示客户端会话信息。

相关链接

《SinoDB 管理员参考》: sysmaster 数据库
《SinoDB 管理员参考》: syssesappinfo
《SinoDB 管理员参考》: onstat -g ses 命令: 显示会话相关信息

客户端/服务器配置的示例

接下来的部分将显示多个客户机/服务器连接的正确的 sqlhosts 条目。

即使没有明确提及，但您可以假定网络配置文件 hosts 和 services 都已经正确准备了。下列示例包括：

- 使用网络连接
- 使用多个连接类型
- 访问多个数据库服务器

共享内存和本地回送连接的示例可以在共享内存和本地回送连接的说明中找到。

相关链接

共享内存连接 *(UNIX)* 在第37页
本地回送连接 在第39页

网络连接

下图显示了两个数据库服务器的网络连接配置：

客户端应用程序位于主机 river 上，数据库服务器位于主机 valley 上。

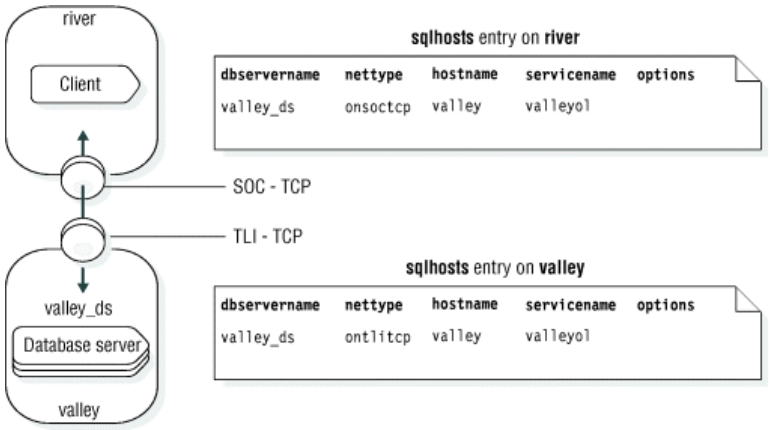

图 6: 网络客户端/服务器配置的示例

在这两台计算机上都有 valley_ds 数据库服务器的 sqlhosts 条目定义。

两台计算机在同一个 TCP/IP 网络上，但是主机 river 对于网络编程接口使用套接字，而主机 valley 对于网络编程接口使用 TLI。nettype 字段必须反映出 sqlhosts 文件所在计算机使用的网络编程接口的类型。此示例中，在主机 river 上对于 valley_ds 数据库服务器的 nettype 字段是 onsoctcp，而在主机 valley 上对于 valley_ds 数据库服务器的 nettype 字段是 ontlitcp。

相关链接

网络编程接口 在第34页

多种连接类型

数据库服务器的单个实例中可以提供多种连接类型。

下图说明了多种连接类型的配置。数据库服务器位于主机 river 上。由于共享内存速度快，所以客户端 A 通过共享内存连接到数据库服务器。而客户端 B 必须使用网络连接，因为客户端和服务器在不同的计算机上。

当您想要数据库服务器接受多种连接类型时，您必须进行下列操作：

- 在 onconfig 文件中添加 DBSERVERNAME 和 DBSERVERALIASES 条目
- 为每对数据库服务器与连接类型，添加一个 sqlhosts 条目。

下图中的配置，数据库服务器具有两个数据库服务器名：river_net 和 river_shm。onconfig 文件包含以下条目：

```
DBSERVERNAME            river_net
DBSERVERALIASES         river_shm
```

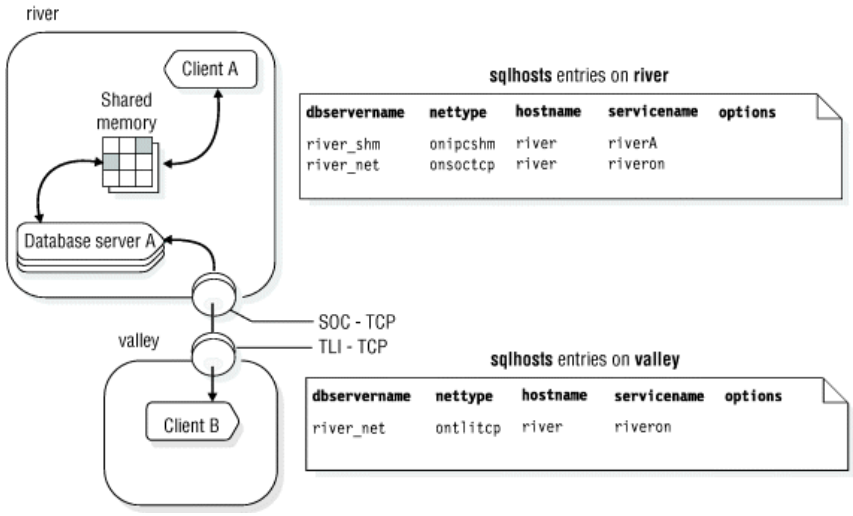

图 7: 使用多种连接类型的 UNIX™ 客户端/服务器配置示例

客户端应用程序使用的数据库服务器名称可用于决定使用的连接类型。客户端 A 使用以下语句来连接数据库服务器：

```
CONNECT TO '@river_shm'
```

在 sqlhosts 文件中，与名称 river_shm 关连的 nettype 指定共享内存连接，因此该连接为共享内存连接。

客户端 B 使用以下语句来连接数据库服务器：

```
CONNECT TO '@river_shm'
```

在 sqlhosts 文件中，与 river_net 关连的 nettype 指定网络（TCP/IP）连接，因此客户端 B 使用网络连接。

访问多个数据库服务器

当一台计算机上有多个数据库服务器处于活动状态，称为多重驻留。

下图显示了主机 river 上两个数据库服务器的配置。

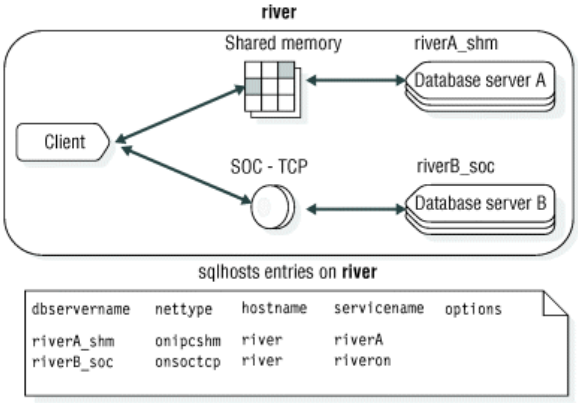

图 8: UNIX™上的多个数据库服务器

对于先前示例中的配置，必须准备一个 onconfig 文件用于数据库服务器 A，并准备另一个文件用于数据库服务器 B。sqlhosts 文件包含这两个数据库服务器的连接信息。

数据库服务器 A 的 onconfig 文件包含以下行：

```
DBSERVERNAME            riverA_shm
```

数据库服务器 B 的 onconfig 文件包含以下行：

```
DBSERVERNAME riverB_soc
```

相关链接

《SinoDB 安装指南》: 创建多处驻留数据库服务器 (Windows)

《SinoDB 安装指南》: 创建多处驻留数据库服务器 (UNIX, Linux)

数据库服务器初始化

数据库服务器需要磁盘空间初始化和共享内存初始化.

共享内存初始化

共享内存初始化或启动服务器建立数据库服务器共享内存的如下内容 ： 内部表、 缓冲区和共享内存通信区。每次数据库服务器启动时都会初始化共享内存。从命令行 使用 oninit 实用程序来初始化数据库服务器共享内存，并使数据库服务器联机。

磁盘空间初始化

磁盘空间初始化使用存储在配置文件中的值在磁盘上创建根数据库空间的初始块。从命令行使用 oninit -i 命令来初始化磁盘空间。初始化磁盘空间时，数据库服务器将自动初始化共享内存作为过程的一部分。磁盘空间将在第一次启动数据库服务器时初始化。

> 警告： 初始化磁盘空间时，将覆盖该磁盘空间上的所有内容。如果重新初始化现有数据库服务器的磁盘空间，那么较早数据库服务器中的所有数据都会被删除。

通过将 FULL_DISK_INIT 配置参数设为 0，以防止意外磁盘初始化。当此参数设为 0 时，如果根数据库空间存在则 oninit -i 命令失效。

共享内存初始化和磁盘空间初始化之间的关键差异是共享内存初始化对于磁盘空间分配或布局没有影响。不会删除任何数据。

相关链接

数据库服务器配置 在第26页

《SinoDB 管理员参考》: oninit 实用程序

《SinoDB 管理员参考》: onmode -k, -m, -s, -u, -j: 更改数据库服务器模式

《SinoDB 管理员参考》: FULL_DISK_INIT 配置参数

初始化过程

启动数据库服务器或初始化磁盘空间时，数据库服务器会执行一组步骤。可以在消息日志中查看每一个步骤的结果。

磁盘空间初始化通常包括共享内存初始化。然而，有些通常在共享内存初始化中发生的活动（如记录配置更改），在磁盘初始化中是不需要的，因为这些活动与新初始化的磁盘无关。

在两种类型的初始化中完成以下主要任务：

- 处理配置文件 在第74页
- 创建共享内存段 在第74页

- 初始化或重启共享内存结构 在第74页
- 初始化磁盘空间 *(必要时)* 在第75页
- 启动所有必需的虚拟处理器 在第75页
- 进行必要的转换 在第75页
- 启动快速恢复和检查点 *(必要时)* 在第75页
- 记录配置变更 在第75页
- 更新 *oncfg_servername.servernum* 文件 在第75页
- 更改为静默模式 在第75页
- 删除临时表空间 *(可选)* 在第76页
- 设置强制驻留 *(如果指定)* 在第76页
- 更改为联机模式, 并将控制权交还用户 在第76页
- 创建或更新 *SMI* 表 *(必要时)* 在第76页
- 在每个检查点监视最大用户连接数 在第76页

处理配置文件

数据库服务器在初始化和重启时使用配置参数分配共享内存段。如果您更改了共享内存配置参数，您必须关闭并重启数据库服务器使更改生效。

ONCONFIG 环境变量指定包含配置参数的 onconfig 文件，必须在您初始化或重启数据库服务器之前设置完毕。

初始化期间，数据库服务器会从以下文件中查找配置值：

- 如果已设置 ONCONFIG 环境变量，数据库服务器会从 onconfig 文件中读取值。

  如果已设置 ONCONFIG 环境变量，但数据库服务器无法访问指定的 onconfig 文件，那么服务器将返回错误信息。
- 如果没有设置 ONCONFIG 环境变量，那么数据库服务器将从 onconfig 文件中读取值。

如果在 onconfig 文件省略了任何配置参数，数据库服务器将使用服务器构建的缺省值。

重启过程将当前配置文件中的值与存储在跟数据库空间保留页 PAGE_CONFIG 中的先前的值（如果有的话）相比较。如果有差异，当数据库服务器重启时将使用当前 onconfig 配置文件中的值。

创建共享内存段

数据库服务器使用配置值来计算数据库服务器的驻留共享内存所需的大小。此外，数据库服务器将计算内部值所需的附加配置要求。将计算并存储空间需求。

为创建共享内存，数据库服务器将从操作系统获取三种不同类型的内存的共享内存空间：

- 常驻部分，用于数据缓冲区和内部表
- 虚拟部分，用于大多数系统和用户会话内存要求
- IPC 通讯部分，用于 IPC 通讯

  仅当您配置了 IPC 共享内存连接时，数据库服务器才会分配该部分共享内存。

接著，数据库服务器将共享内存段连接到其虚拟地址空间并初始化共享内存结构。有关共享内存结构的更多信息，请参阅 共享内存的虚拟部分 在第114页。

在完成初始化且数据库服务器已开始运行之后，服务器可根据需要创建额外的共享内存段。数据库服务器将以页大小的增量方式创建段。

初始化或重启共享内存结构

数据库服务器连接到共享内存后，将清除未初始化数据的共享内存空间。接著，数据库服务器将配置共享内存头信息并初始化共享内存结构中的数据。数据库服务器将配置逻辑日志缓冲区所需的空间，初始化这些结构，并将形成逻辑日志缓冲区的三个单独缓冲区链接起来。

在数据库服务器重映射共享内存空间之后，它会将每个结构的新起始地址和大小注册到新的共享内存头中。

共享内存初始化期间，磁盘结构和磁盘布局不受影响。数据库服务器会从磁盘中读取所需地址信息（如逻辑日志和物理日志的位置），并使用此信息更新共享内存中的指针。

初始化磁盘空间（必要时）

仅在第一次启动服务器时或在运行 oninit -i 命令时，磁盘空间才会进行初始化。数据库服务器重启时不会初始化磁盘空间。在共享内存结构初始化之后，数据库服务器开始进行磁盘初始化。数据库服务器初始化保留在磁盘的根数据库空间中的所有保留页，并将页面 0 的控制信息写入磁盘。

配置参数 FULL_DISK_INIT 指定当根路径位置（第一个块位置的第一个页面处）有页面 0 时，oninit -i 命令是否可以在实例上运行。使用此配置参数可防止意外重新初始化现有服务器实例的磁盘。

FULL_DISK_INIT 配置参数的缺省设置为 0。当此配置参数设置为 0 时，那么仅当根路径位置没有页面 0 时，oninit -i 命令才能运行。

如果根路径位置有页面 0，那么仅当 FULL_DISK_INIT 配置参数设置为 1时才能进行初始化。初始化之后，数据库服务器会自动将 FULL_DISK_INIT 配置参数重置为0 。

启动所有必需的虚拟处理器

数据库服务器启动所有其所需的虚拟处理器。

onconfig 文件中的参数影响启动哪些处理器。例如，NETTYPE 参数可以影响用于处理连接的处理器的启动数量和类型。有关虚拟处理器的更多信息，请参阅 虚拟处理器 在第81页。

进行必要的转换

数据库服务器会检查其内部文件。如果文件来自较早的版本，服务器会将文件更新到当前格式。

启动快速恢复和检查点（必要时）

数据库服务器检查是否需要快速恢复，如果需要则启动快速恢复。快速恢复不会在磁盘空间初始化期间执行，因为还没有任何要恢复的内容。

有关快速恢复的更多信息，请参阅 快速恢复 在第268页。

在完成快速恢复之后，数据库服务器将运行一个检查点来验证是否所有已恢复的事务都已清空到磁盘以便不重复执行这些事务。

作为检查点过程的一部分，数据库服务器将在消息日志中写入检查点完成的消息。有关检查点的更多信息，请参阅 检查点 在第266页。

记录配置变更

数据库服务器将存储在配置文件中的当前值与存储在根数据空间保留页 PAGE_CONFIG 中的以前的值相比较。如有差异，数据库服务器会在消息日志中写入消息记录这两个值（新值和旧值）。

更新 oncfg_servername.servernum 文件

数据库服务器创建 oncfg_*servername.servernum* 文件并在您每次添加或删除数据空间， BLOB 空间、逻辑日志文件或块时更新它。

您不需要以任何方式处理该文件，但是会发现该文件列在 UNIX™ 的 $INFORMIXDIR/etc 目录或 Windows™ 的 %INFORMIXDIR%\etc 目录。数据库服务器会在抢救逻辑日志的全系统恢复期间使用 oncfg_*servername.servernum* 文件。

更改为静默模式

数据库服务器切换为静默模式或联机模式，取决于如何启动初始化或数据库服务器的重启过程。

删除临时表空间（可选）

临时表空间（如果有的话）是由过早终止而未能执行适当清除的用户进程所留下的表空间。数据库服务器将删除任何临时表空间并回收磁盘空间。有关临时表空间的更多信息，请参阅 临时表 在第158页。

该任务在数据库服务器重启时执行，在磁盘空间初始化时不会执行。

如果使用 oninit 的 -p 选项进行数据库服务器的初始化，则数据库服务器会跳过这个步骤。

设置强制驻留（如果指定）

如果 RESIDENT 配置参数值为 -1 或大于 0 的数字，数据库服务器将尝试强制共享内存为驻留状态。

如果主机系统不支持强制驻留，初始化进程会继续进行。驻留未强制执行，所以数据库服务器将向消息日志发送错误消息。

更改为联机模式，并将控制权交还用户

当数据库服务器将 SinoDB® Dynamic Server initialized - complete disk initialized 消息写入消息日志（该消息只有在数据库服务器初始化而非重新启动时才会写入），以及动态分配虚拟共享内存段时，控制权才会交还用户。

服务器将控制权交还用户。初始化过程生成的任何错误消息都会显示在以下位置：

- 命令行
- 由 MSGPATH 配置参数指定的数据库服务器消息日志文件。

可以使用 oninit -w 实用程序迫使服务器在可配置的超时内返回到命令提示符。oninit -w 实用程序对于排除初始化故障很有用。

创建或更新 SMI 表（必要时）

如系统监视界面 (SMI) 表和其他系统数据库不存在，数据库服务器将创建它们。

如果 SMI 表不是最新的，数据库服务器将更新这些表。如果 SMI 表不存在，如同磁盘初始化的状况一样，数据库服务器将创建这些表。在数据库服务器创建 SMI 表之后，将 sysmaster database built successfully 消息写入消息日志。数据库服务器还会在转换和逆转时重建 sysmaster 数据库。有关 SMI 表的更多信息，请参阅 《*SinoDB*® 管理员参考》中的 sysmaster 数据库章节。

如果您在数据库服务器完成 SMI 表构建之前关闭该服务器，表的构建进程便会停止。此情况不会损坏数据库服务器。数据库服务器在您下次使数据库服务器联机时构建 SMI 表。但是，如果您不允许这些 SMI 表完成构建，您将无法对于这些表进行查询，也无法使用 ON-Bar 进行备份。

数据库服务器将在磁盘初始化，转换或逆转期间将删除并重新创建 sysutils 数据库。ON-Bar 将备份与恢复信息存储在 sysutils 数据库。该数据库创建完成后，将 sysutils database built successfully 消息写入消息日志中。

数据库服务器将在初始化期间创建 sysuser 数据库。sysuser 数据库用于 SinoDB® 服务器与服务器通信中的可插式认证模块 (PAM) 认证。

数据库服务器将在初始化期间创建 sysadmin 数据库。sysadmin 数据库提供 SinoDB® 中的远程管理与调度程序 API 功能部件。

在 SMI 表和系统数据库创建完成后，数据库服务器便可以使用了。数据库服务器会一直运行，除非您停止它或遇到故障。

> 建议： 请不要试图通过停止虚拟处理器或关闭另一个数据库服务器进程来停止数据库服务器的运行。相关更多信息，请参阅 启动和停止虚拟处理器 在第102页。

在每个检查点监视最大用户连接数

数据库服务器在每个检查点以下列格式在消息日志中打印最大的用户连接数：maximum server connections *number*。可以监视自从上次重启或磁盘初始化以来连接到数据库服务器的用户数。

当重新初始化数据库服务器时，用户数将重置。

数据库服务器运行模式

可从命令行执行 onstat 实用程序来确定当前的数据库服务器模式。 onstat 标头将显示该模式。

该表显示了数据库服务器的主要运行模式。

表 12: 运行模式

运行模式	描述	允许访问的用户
脱机模式	数据库服务器没有运行。没有分配共享内存。	仅管理员（informix 用户）可以从此模式更改为其他模式。
静默模式	数据库服务器进程正在运行，并且分配了共享内存资源。 管理员使用此模式来执行不需要执行 SQL 和 DDL 语句的维护功能。	仅管理员（informix 用户）能访问数据库服务器。 其他用户可以查看数据库服务器的状态信息，但不能访问数据库服务器。
管理模式	此模式是一种介于静默模式和联机模式之间的中间模式。 管理员使用此模式来执行任何维护任务，其中包括需要执行 SQL 和 DDL 语句的维护任务。管理员还可以执行联机模式中可用的其他所有功能。	以下用户可以在管理模式下连接到数据库服务器： • informix 用户 • 拥有 DBSA 角色的用户 如果想要 DBSA 组成员的用户（除了 informix 用户以外）连接到在管理模式下的数据库服务器，请将 ADMIN_USER_MODE_WITH_DBSA 配置参数设为 1 。 • 一个或多个用户有管理模式访问权限。 Informix 用户或DBSA 可以通过 onmode-j 命令， oninit-U 命令或 ADMIN_MODE_USERS 配置参数动态地授与一个或多个特定用户连接到在管理模式下的数据库服务器。 其他用户可以查看数据库服务器的状态信息，但不能访问数据库服务器。
联机模式	此为数据库服务器的正常运行模式。	任何授权的用户都可以连接到数据库服务器，并执行所有数据库活动。 Informix 用户或 root 用户可以使用命令行实用程序来更改许多数据库服务器的 ONCONFIG 参数值。

此外，数据库服务器还可以是以下其中一种模式：

- 只读模式用于数据复制环境中的辅助数据库服务器。应用程序可以查询在只读模式下的辅助数据库服务器，但无法写入只读数据库。
- 恢复模式 是过渡的。在数据库服务器从系统归档中执行快速恢复或恢复，或执行系统还原时发生。恢复是在从脱机模式转到静默模式时发生。
- 关闭模式 是过渡的。在数据库服务器从联机模式转换到静默模式或从联机模式（或静默模式）转换到脱机模式时发生。当前的用户可以访问系统，但不允许任何新用户访问。

关闭模式启动后，就无法取消。

允许更改模式的用户

仅 UNIX™

以root 或 informix 身份登录的用户和 DBSA 组成员可以更改数据库服务器的运行模式。

如果您想要 DBSA 组的用户在管理模式下连接到数据库服务器，请将 ADMIN_USER_MODE_WITH_DBSA 配置参数设置为 1。若此参数设置为 0，则仅用户 informix 可以访问。如果 $ONCONFIG 中缺少该参数，那么该参数值将视为 0。

用户 Informix 或 DBSA 可以使用 onmode 实用程序、oninit 实用程序或 ADMIN_MODE_USERS 配置参数动态地授与一个或多个特定用户在管理模式下连接到数据库服务器的能力。

> 注： 对于 DBSA 组的成员，必须更改 $INFORMIXDIR/bin/oninit 的权限，以在标准 SinoDB® 安装中允许 public execute permission - root:informix:6755。

仅 Windows™

表 12: 运行模式 在第77页 显示在 Windows™ 中可以更改数据库服务器运行模式的用户。Apache 以用户 informix 身份运行。Apache 服务器以 Informix-Admin 组成员的身份运行。

表 13: 更改 Windows™ 中的运行模式

更改运行模式	Administrators 组	Informix-Admin 组
命令行实用程序，如 starts		X
服务控制面板	X	

更改数据库服务器运行模式

使用 oninit 和 onmode 实用程序来变更数据库服务器的运行模式。使用 ADMIN_MODE_USERS 配置参数来指定哪些用户可以连接到在管理模式下的数据库服务器。

> 仅 Windows: 在 Windows™ 操作系统中，数据库服务器作为服务运行。Windows™ 提供服务控制应用程序（也称为 "服务" 工具）来启动，停止和暂停服务。服务控制应用程序位于控制面板程序组。数据库服务器的服务名称包括数据库服务器名（onconfig 文件中的 DBSERVERNAME 值）。例如，newyork 数据库服务器的 SinoDB® 服务是:
>
> ```
> Sinoregal Database Server - newyork
> ```
>
> 要使用 "服务" 工具来更改模式，请启动该工具并选择数据库服务器服务。然后在服务窗口中选择适当的选项。在这些主题之后显示的表将说明每种模式要选择的选项。
>
> 要启动和停止数据库服务器，您可以使用其他 Windows™ 工具，如 NET 命令和服务器管理器工具。有关这些方法的更多信息，请查询您的 Windows™ 操作系统文档。
>
> 提示: 在更改数据库服务器的模式后，运行 onstat 命令以验证当前服务器状态。

可使用以下方法更换数据库服务器模式:

从脱机模式转为静默模式

将数据库服务器从脱机模式转为静默模式时，数据库服务器将初始化共享内存。仅管理员才能访问数据库服务器，并执行不涉及 SQL 和 DDL 语句的维护功能。

UNIX：运行 oninit-s 命令。

Windows：在命令行，执行 starts *dbservername* -s 命令。

从脱机模式转为联机模式

将数据库服务器从脱机模式转为联机模式时，数据库服务器将初始化共享内存，并且所有用户都可使用。

UNIX：运行 oninit-s 命令。

Windows：在 "服务" 工具中，选择数据库服务器服务，然后单击 开始。或者，在命令行执行 starts ***dbservername*** 命令。

从脱机模式转为管理模式

将数据库服务器从脱机模式转为管理模式时，在该模式下，仅管理员可以执行数据库服务器功能和维护功能（其中包含涉及 SQL 和 DDL 语句的执行）。

运行 oninit-s 命令。

从静默模式转为联机模式

当数据库服务器从静默模式转为联机模式时，所有会话都将获得访问权。如您将数据库服务器从联机模式转为静默模式，再转回到联机模式，那么较早的处理中被中断的用户都必须重新选择他们的数据库，并重新声明他们的游标。

Unix 或 Windows：运行 onmode -m 命令。

仅适用于 Windows：在 "服务" 工具中，选择数据库服务器服务，然后单击 继续。

平稳地从联机模式转为静默模式

在不中断当前处理的情况下，将数据库服务器从联机模式平稳地转为静默模式以限制对数据库服务器的访问。在执行此任务后，数据库服务器将设置一个标志以防止新的会话获得数据库服务器的访问权。但允许当前的会话完成处理。一旦启动模式变更后，就无法取消。在联机模式转为静默模式的过程中，数据库服务器被认为是处于关闭模式。

Unix 或 Windows：运行 onmode -s 或 onmode -sy 命令。

仅适用于 Windows：在 "服务" 工具中，选择数据库服务器服务，然后单击 暂停。

立即从联机模式转为静默模式

将数据库服务器从联机模式立即转为静默模式以限制对数据库服务器的访问。正在进行的工作可能会丢失。提示询问是否确认立即关闭。如果确认，数据库服务器将向连接到共享内存的所有用户发送断开信号。如果会话没有接收到断开信号或不能在 10 秒内自动遵从，数据库服务器将终止该会话。数据库服务器用户将会收到错误消息 -457（指示数据库服务器已关闭）或收到错误消息 -459（指示会话已被意外终止）。数据库服务器将清除数据库服务器已终止的所有会话。活动的事务将回滚。

运行 onmode -u 或 onmode -uy 命令。

从静默或联机模式转为管理模式

将数据库服务器从静默或联机模式转为管理模式时，在该模式下，仅管理员可以使用。如果是从联机模式开始，数据库服务器将自动断开与所有用户（使用非用户 informix 的用户标识连接的用户）的连接，并且这些用户将收到错误消息。如果连接在事务期间被终止，那么数据库服务器将回滚该事务。当您想在没有连接其他用户的情形下运行 SQL 和 DLL 命令时，请更改到管理模式。另请参阅 指定管理模式用户 在第80页。

运行 onmode -j 命令。

从管理模式转为联机模式

当数据库服务器从管理模式转为联机模式，所有用户都可以获取数据库服务器的访问权。

运行 onmode -m 命令。

从管理模式转为静默模式

当数据库服务器从管理模式转为静默模式时，在该模式下，仅管理员可以使用，用于执行不涉及执行 SQL 和 DDL 语句的维护功能。

运行 onmode -s 命令。

立即从任何模式转为脱机模式

可以将数据库服务器从任何模式立即转为脱机模式。提示询问是否确认进入脱机模式。如果确认，数据库服务器会发出检查点请求，并向连接到共享内存的所有用户发送断开信号。如果会话没有接收到断开信号或不能在 10 秒内自动遵从，数据库服务器将终止该会话。数据库服务器用户将会收到错误消息 -457（指示数据库服务器已关闭）或收到错误消息 -459（指示会话已被意外终止）。数据库服务器转为

脱机模式后，可使用静默模式、管理模式或联机模式重新启动数据库服务器。当重新启动数据库服务器时，它会执行快速恢复以确保数据在逻辑上是一致的。数据库服务器将清除数据库服务器已终止的所有会话。活动的事务将回滚。

Unix 或 Windows：运行 onmode -k 或 onmode -ky 命令。

仅适用于 Windows：在 "服务" 工具中，选择数据库服务器服务，然后单击 停止。

如果 onmode 命令无法关闭数据库服务器，可使用 onclean 实用程序强制执行立即关机。

相关链接

《*SinoDB* 管理员参考》: *oninit* 实用程序

《*SinoDB* 管理员参考》: *onmode -k, -m, -s, -u, -j:* 更改数据库服务器模式

指定管理模式用户

您可以指定哪些用户可以在管理模式下连接数据库服务器。

临时管理模式用户

用户 Informix 或 DBSA 可以使用 onmode -j -U 或 oninit -U 命令，授与单个用户的会话访问管理模式下的数据库服务器的权限。

例如，运行以下命令使得三位用户可以连接到数据库服务器并具有数据库服务器访问权直到数据库服务器模式更改为脱机、静默或联机模式为止。

```
onmode -j -U mark,ajay,carol
```

连接后，这些用户可以运行任何 SQL 或 DDL 命令。 当服务器更改到管理模式时，所有在 onmode -j -U 命令中未被识别的用户的会话将丢失与数据库服务器的连接。

在最初运行 onmode -j -U 命令之后，您可以通过 onmode -j -U 移除单个用户，将单个用户名从新名称列表中移除。例如，运行：

```
onmode -j -U mark,carol
```

运行带有空白而非名称的 oninit -U 命令来移除列表中的所有用户，如以下例所示：

```
oninit -U " "
```

永久管理模式用户

不同于 Oninit 和 onmode 命令可使您指定管理模式下的用户直到服务器模式更改为脱机、静默或联机模式为止，ADMIN_MODE_USERS 配置参数可以无限保存管理模式下的用户列表。

若要建立保存在 onconfig 文件中的管理模式用户列表，请指定以逗号分隔的用户名列表作为 ADMIN_MODE_USERS 配置参数值。例如： mark,ajay,carol。

若要在一个会话期间覆盖 ADMIN_MODE_USERS，请使用 onmode -wf 命令，如以下示例所示：

```
onmode -wf ADMIN_MODE_USERS=sharon,kalpana
```

ADMIN_MODE_USERS 配置参数的作用是添加可以访问管理模式下的数据库服务器的人员列表。在 onmode 命令行中列出的人员将覆盖在 onconfig 文件中列出的人员。

相关链接

《*SinoDB* 管理员参考》: *oninit* 实用程序

《*SinoDB* 管理员参考》: *onmode -k, -m, -s, -u, -j:* 更改数据库服务器模式

《*SinoDB* 管理员参考》: *ADMIN_MODE_USERS* 配置参数

第 II 部分

磁盘、内存和进程管理

虚拟处理器和线程

这些主题描述虚拟处理器，说明线程如何在虚拟处理器中运行，并说明数据库服务器如何使用虚拟处理器和进程来提高性能。

相关链接

数据库服务器维护任务 在第30页

虚拟处理器

虚拟处理器是操作系统调度用于处理的进程。

数据库服务器进程叫做 虚拟处理器 ，因为他们的作用方式与计算机中 CPU 的作用方式类似。如同 CPU 运行多个操作系统进程以服务多个用户一样，数据库服务器虚拟处理器运行多个线程以服务多个 SQL 客户端应用程序。

下图说明客户端应用程序与虚拟处理器的关系。少量的虚拟处理器可以服务大量的客户端应用程序或查询。

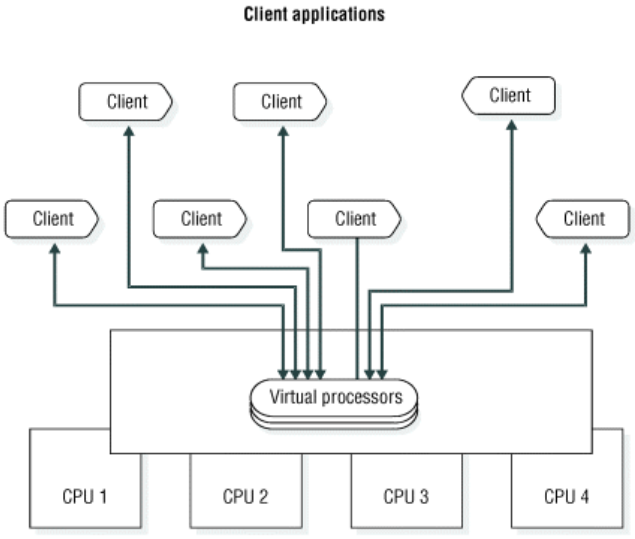

图 9: 虚拟处理器

线程

线程是虚拟处理器的一个任务，如同虚拟处理器是 CPU 的一个任务一样。

虚拟处理器是操作系统调度在 CPU 上执行的任务；数据库服务器线程是虚拟处理器为处理而内部调度的任务。线程有时称为轻量级进程，因为它们类似于进程，但对于操作系统的要求较少。

数据库服务器虚拟处理器是多线程的，因为他们运行多个并发线程。

线程的特性如下。

操作系统	操作
UNIX™	线程是虚拟处理器为处理而内部调度的任务。
Windows™	线程是虚拟处理器为处理而内部调度的任务。因为虚拟处理器作为 Windows™ 线程实施，所以数据库服务器线程在 Windows™ 线程中运行。

> 重要： 在这些主题中，所有对线程的引用都是指由数据库服务器创建、调度和删除的线程。所有对“ Windows 线程”的引用都是指由 Windows™ 创建、调度和删除的线程。

虚拟处理器代表 SQL 客户端应用程序运行线程（会话线程），并运行线程以满足内部需求（内部线程）。在大多数情况下，对于客户端应用程序的每个连接，数据库服务器都会运行一个会话线程。数据库服务器运行内部线程以完成数据库 I/O、日志记录 I/O、页面清除和管理任务。有关数据库服务器为单个客户端运行多个会话线程的情况，请参阅 并行处理 在第83页。

用户线程是用于服务客户端应用程序请求的数据库服务器线程。用户线程包括会话线程（称为 sqlexec 线程），这些线程是数据库服务器为服务客户端应用程序而运行的主线程。

用户线程还包括用于服务 onmode 实用程序请求的线程、恢复线程、B型树扫描程序线程和页清除程序线程。

要显示活动用户线程，请使用 onstat -u。有关监视会话和线程的更多信息，请参阅 《*SinoDB*® 性能指南》。

虚拟处理器的优势

虚拟处理器有很多优势。

相较于数据库服务器为单个客户应用程序提供服务，数据库服务器虚拟处理器的动态多线程的特性将提供以下优势：

- 虚拟处理器可以共享处理。
- 虚拟处理器可以节省内存和资源。
- 虚拟处理器可以执行并行处理。
- 当数据库服务器正在运行时，您可以启动其他的虚拟处理器并终止活动的 CPU 虚拟处理器。
- 可以将虚拟处理器绑定到 CPU。

下列主题将描述这些优势。

共享处理

同一类的虚拟处理器具有相同的代码，并且共享对内存中的数据和处理队列的访问。类中的任何虚拟处理器都可以运行属于该类的任何线程。

通常，数据库服务器会尝试使线程保持在同一虚拟处理器上运行，因为将其移动到不同的虚拟处理器上可能需要在总线上传送处理器内存的一些数据。然而，当线程正在等待运行时，数据库服务器可以将线程移动到另一虚拟处理器上运行，因为平衡处理负载的好处要远大于传输数据时花费的开销。

在一类虚拟处理器中的共享处理是自动发生的，且对数据库用户是透明的。

节省内存与资源
相较于一个客户端进程对应一个服务器进程的体系结构，对于每个客户端，数据库服务器通过运行线程而非进程的方式能够以少量服务器进程对大量客户端提供服务。

多线程可以更有效地使用操作系统资源，因为线程会共享分配给虚拟处理器的资源。虚拟处理器运行的所有线程对虚拟处理器内存、通信端口和文件具有相同访问权。虚拟处理器会协调线程对资源的访问。虽然每个进程都有一组不同的资源，但当多进程需要访问相同资源时，操作系统必须对访问进行协调。

通常，虚拟处理器从一个线程切换到另一个线程的速度比操作系统从一个进程切换到另一个进程还快。当操作系统在进程间切换时，它必须使一个进程停止在处理器上运行，保存其当前的处理状态（或上下文），然后启动另一个进程。两个进程都必须进入和退出操作系统内核，并且可能需要替换部分物理内存内容。然而，线程共享相同的虚拟内存和文件描述符。当虚拟处理器从一个线程切换到另一个线程时，该切换是从一个执行路径切换到另一个执行路径。作为进程的虚拟处理器继续在 CPU 上运行而不中断。有关虚拟处理器如何从一个线程切换到另一个线程的描述，请参阅 上下文切换 在第84页。

并行处理
当客户端启动索引构建、排序或逻辑恢复时，数据库服务器创建多个线程来并行处理该任务，尽可能多地使用计算机资源。当一个线程正在等待 I/O 时，另一个线程就可以工作。

在以下情况中，CPU 类的虚拟处理器可为单个客户端以并行方式运行多个会话线程：

- 索引构建
- 排序
- 恢复
- 扫描
- 连接
- 聚集
- 分组
- 用户定义的例程 (UDR) 执行

有关并行 UDR 执行的更多信息，请参阅《*SinoDB*® 用户自定义例程和数据类型开发者指南》。

下图说明了并行处理。

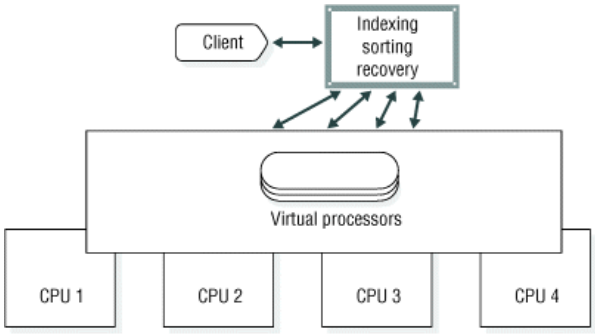

图 10: 并行处理

在联机模式下添加和删除虚拟处理器
您可以添加虚拟处理器以满足数据库服务器运行中对服务日益增加的需求。

如果某类的虚拟处理器变成计算绑定或 I/O 绑定（意思是 CPU 工作或 I/O 请求的积累速度远超出虚拟处理器的当前数量的处理能力范围），您可以为该类启动额外的虚拟处理器，以进一步分散处理负荷。

更多信息，请参阅 在联机模式下添加虚拟处理器 在第103页。

仅 Windows: 在 Windows™ 环境下，不可以删除任何类的虚拟处理器。

当数据库服务器正在运行时，您可以删除 CPU 的虚拟处理器，或用户定义类。更多信息，请参阅 设置虚拟处理器配置参数 在第102页。

将虚拟处理器绑定至 CPU
可以使用某些多处理器系统将进程绑定至特定 CPU。该功能称为处理器专用。

在数据库服务器支持处理器专用的多处理器计算机上，您可以将 CPU 虚拟处理器绑定至该计算机中特定 CPU。将 CPU 虚拟处理器绑定至 CPU时，该虚拟处理器仅在该 CPU 上运行。此操作将提高虚拟处理器的性能，因为它减少了操作系统必须在进程间进行的切换量。将CPU虚拟处理器绑定到特定的CPU还可以隔离计算机上特定处理器上的数据库工作，使剩馀的处理器可以处理其他工作。仅 CPU 虚拟处理器可以绑定至 CPU。

有关如何将 CPU 虚拟处理器指定到硬件处理器的信息，请参阅 处理器专用 在第90页。

虚拟处理器如何服务线程

虚拟处理器通过在多个线程之间切换来并发地为这些线程提供服务。

在给定的时间里，一个虚拟处理器只能运行一个线程。虚拟处理器运行一个线程直到其让步。当线程让步时，虚拟处理器将切换至下一个准备运行的线程。虚拟处理器持续这个过程，最终当原始的线程准备好继续运行时返回到该线程。有些线程完成了它们的工作，虚拟处理器会启动新的线程来处理新的工作。由于虚拟处理器在线程之间持续切换，它可以使得 CPU 持续处理。处理的速度使虚拟处理器看起来就像同时在处理多个任务，而实际上也确实如此。

运行多个并发线程需要调度和同步，以防止一个线程干扰另一个线程的工作。虚拟处理器使用以下结构与方法，以协调多线程的并发处理：

- 控件结构
- 上下文切换
- 堆栈
- 队列
- 互斥

这些主题描述了虚拟处理器如何使用这些结构与方法。

控制结构

当客户端连接至数据库服务器时，数据库服务器将创建 会话 结构（称为会话控制块）以保存联接与用户的信息。

会话在客户机连接到数据库服务器时开始并在连接终止时结束。

接下来，数据库服务器创建了一个线程结构（称为会话的 线程控制块 (TCB)），并启动主线程 (sqlexec) 处理客户端请求。当线程 让步，即当它暂停并允许下一线程运行时，虚拟处理器会将该线程的状态信息保存在线程控制块中。该信息包括进程系统注册的进程内容、程序计数器（下一个要执行的指令的地址），以及堆栈指针。此信息组成线程内容。

大多数情况下，数据库服务器将在每个会话中运行一个主线程。然而，在它执行并行处理的情况下，它会为单个客户端创建多个会话线程，同样地，也创建多个相应的线程控制块。

上下文切换

虚拟处理器通过 上下文切换从运行某个线程切换到运行另一个线程。

当固定的时间量（时间片）逾期时，数据库服务器不会像操作系统抢占进程那样抢占正在运行的线程，而是线程会在以下其中之一的节点上让步：

- 代码中的预定点
- 在某个条件满足前该线程不会执行

当完成任务所需的处理量会造成其他线程等待一段超长时间时，线程将在某个预定点上让步。这种长时间运行的任务的代码包括在处理中的关键点上调用让步函数。当线程执行了这些任务之一时，它会在遇到让步函数调用时让步。在就绪队列中的其他线程将有机会运行。当下一次轮到原来的线程时，它将在调用让步函数之后紧接的点上恢复代码执行。对让步函数的预定调用使数据库服务器可在对性能最有利的点上中断线程。

当某个条件发生前线程无法继续它的任务时，它也会让步。例如，当线程等待磁盘 I/O 完成时、当等待来自客户端的数据时、当等待锁定或其他资源时，它都会让步。

当线程让步时，虚拟处理器将其上下文保存在线程控制块中。接着，虚拟处理器从一队就绪的线程中选择要运行的新线程，从该新线程的线程控制块加载它的上下文，然后在程序计时器中的新地址处开始执行。下图说明虚拟处理器如何完成上下文切换。

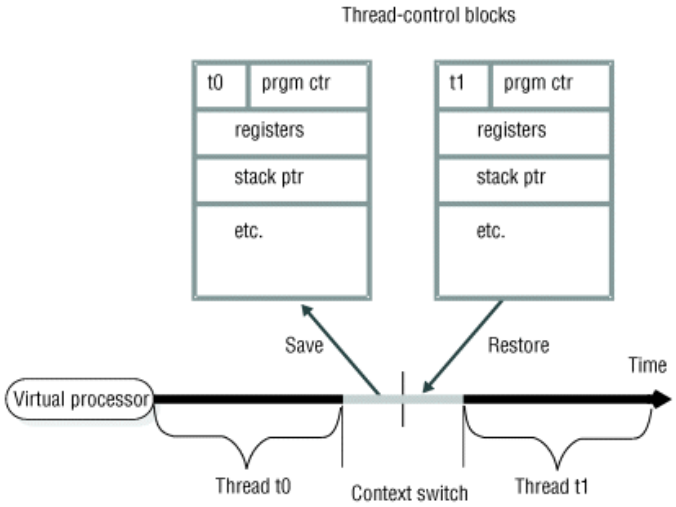

图 11: 上下文切换：虚拟处理器如何从一个线程切换到另一个

堆栈

数据库服务器在共享内存的虚拟部分中分配一个区域以存储线程所执行的函数的非共享数据。该区域称为堆栈。

有关如何设置堆栈大小的信息，请参阅 堆栈 在第117页。

堆栈使虚拟处理器能够保护线程的非共享数据免遭并发执行相同代码的其他线程的覆盖。例如，如果多个客户端应用程序并发执行 SELECT 语句，那么每个客户端的会话线程将在代码中执行许多相同的函数。如果线程没有私有堆栈，一个线程可能会在一个函数中覆盖掉属于另一个线程的本地数据。

当虚拟处理器切换到新的线程时，它会从该线程的线程控制块加载该线程的堆栈指针。堆栈指针储存堆栈的起始地址。虚拟处理器可以指定起始地址的偏移量以访问堆栈中的数据。下图说明虚拟处理器如何使用堆栈隔离会话线程的非共享数据。

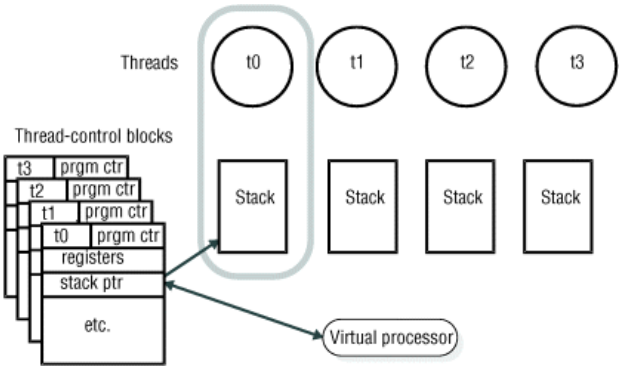

图 12: 虚拟处理器为每个用户隔离非共享数据

队列

数据库服务器使用三种类型的队列来调度多个并发运行的线程的处理。

相同类的虚拟处理器共享队列。在某种程度上，这使線程能夠在需要时可以从类中的一个虚拟处理器迁移到另一个虚拟处理器。

就绪队列
就绪队列容纳当前（正在运行）线程让步时准备运行的线程。

当一个线程让步时，虚拟处理器将从就绪队列中选取下一个具有适当优先级的线程。在该队列中，虚拟处理器以先进先出 (FIFO) 基础来处理具有相同优先级的线程。

在一台多处理器计算机上，如果您注意到某类的虚拟处理器的线程正在就绪队列中堆积（表示工作堆积的速度比虚拟处理器可以处理的速度快），那么您可以启动该类的额外虚拟处理器以分散处理负载。有关如何监视就绪队列的信息，请参阅 监视虚拟处理器 在第104页。有关如何在数据库服务器处于联机模式时添加虚拟处理器的信息，请参阅 在联机模式下添加虚拟处理器 在第103页。

休眠队列
休眠队列容纳在特定时间没有工作要做的线程的内容。可以使线程休眠一段时间或是永久休眠。

虚拟处理器的管理类 (ADM) 运行系统计时器和特殊的实用程序线程。该类的虚拟处理器会自动创建并运行。没有配置参数会影响虚此类的拟处理器。

ADM 虚拟处理器唤醒已休眠了一段时间的线程。ADM 虚拟处理器中运行的线程每隔一秒来检查一次正在休眠的线程。如果某休眠线程已休眠了指定的时间，ADM 虚拟处理器就会将其移至适当的就绪队列。已休眠了指定的时间的线程，也可显示地被另一个线程唤醒。

永久休民的线程会在其有更多的工作要做时被唤醒。例如，当 CPU 虚拟处理器上运行的线程必须访问磁盘时，它会发送 I/O 请求，然后将自身置于 CPU 虚拟处理器的休眠队列中并中止。当 I/O 线程通知 CPU 虚拟处理器 I/O 已经完成时，CPU 虚拟处理器调度原来的线程通过将其从休眠队列移到就绪队列来继续处理。下图说明数据库服务器如何排队执行数据库 I/O。

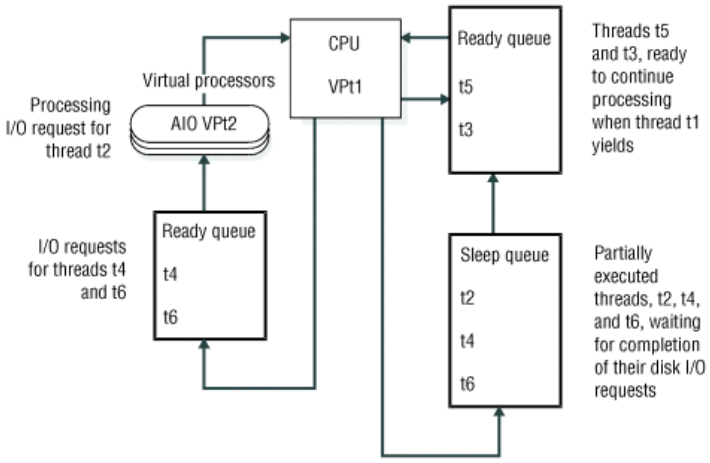

图 13: 数据库服务器如何排队执行数据库 I/O

等待队列
等待队列保存了必须等待特定事件才能继续运行的线程。

等待队列协调线程对共享数据的访问。当用户线程尝试获取逻辑日志锁存器却发现该锁存器正被其他用户占用，遭拒绝的线程将自身放入逻辑日志等待队列中。当占用锁的线程准备释放锁存器时，它将检查等待的线程，如果有线程正在等待，它将唤醒等待队列中的下一个线程。

互斥

互斥（互相排斥），也称为锁存器，是数据库服务器用于同步多个线程对共享资源的访问的一种锁存机制。

互斥与信号相类似，一些操作系统使用后者来控制多个进程对分享数据的访问。然而，互斥比信号允许更高的并行度。

互斥是与共享资源（如缓冲区）关联的变量。线程必须在访问资源前获取该资源的互斥。其他线程将不能访问资源直至拥有者将其释放。在互斥为可用后，线程通过将其设置为使用状态而获得互斥。互斥的同步确保一次只有一个线程写入共享内存的区域。

有关监视互斥的信息，请参阅监视共享内存概要文件和锁存器 在第134页。

相关链接

共享内存的缓冲池部分 在第113页

虚拟处理器类

每类的虚拟处理器专用于处理特定类型的线程。

下表显示虚拟处理器类和其处理类型。

每类虚拟处理器的数量配置取决于物理处理器 (CPU)、硬件内存和使用中的数据库应用程序的可用性。

表 14: 虚拟处理器类

虚拟处理器类	分类	目的
ADM	管理	执行管理功能。
ADT	审计	执行审计功能。
AIO	磁盘 I/O	执行无日志磁盘 I/O。如果使用 KAIO，AIO 虚拟处理器对熟磁盘空间执行 I/O。
BTS	Basic Text Search	运行 Basic Text Search 索引操作和查询。
CPU	中央处理	运行所有会话线程和一些系统线程。运行内核异步 I/O (KAIO) 线程。可以运行单个轮询线程，取决于配置。
CSM	通信支持模块	执行通信支持服务操作。
dwavp	数据仓库	在连接到 仓库加速器 的数据库服务器上运行 仓库加速器 的管理功能和程序。
IDSXMLVP	XML 发布	运行 XML 发布功能。
JVP	Java™ UDR	运行 Java™ UDR。包含 Java™ 虚拟机 (JVM)。
LIO	磁盘 I/O	若逻辑日志文件是在熟磁盘空间中，则写入逻辑日志文件（内部类）。
MQ	MQ 消息	运行 MQ 消息事务。
MSC	杂项	对于需要大量堆栈的系统调用的服务请求。
PIO	磁盘 I/O	若物理日志文件是在熟磁盘空间中，则写入物理日志文件（内部类）。
SHM	网络	运行共享内存通信。
SOC	网络	使用接口执行网络通信。
租户	多租户	运行租户数据库的会话线程。租户虚拟处理器是特定于租户数据库的用户定义处理器的特例。
TLI	网络	使用传输层接口 (TLI) 执行网络通信。
类名	用户定义的	在线程安全模式下运行用户定义例程，若例程失败，也不会影响数据库服务器。

下图说明了数据库服务器的主要组件和可扩展性。

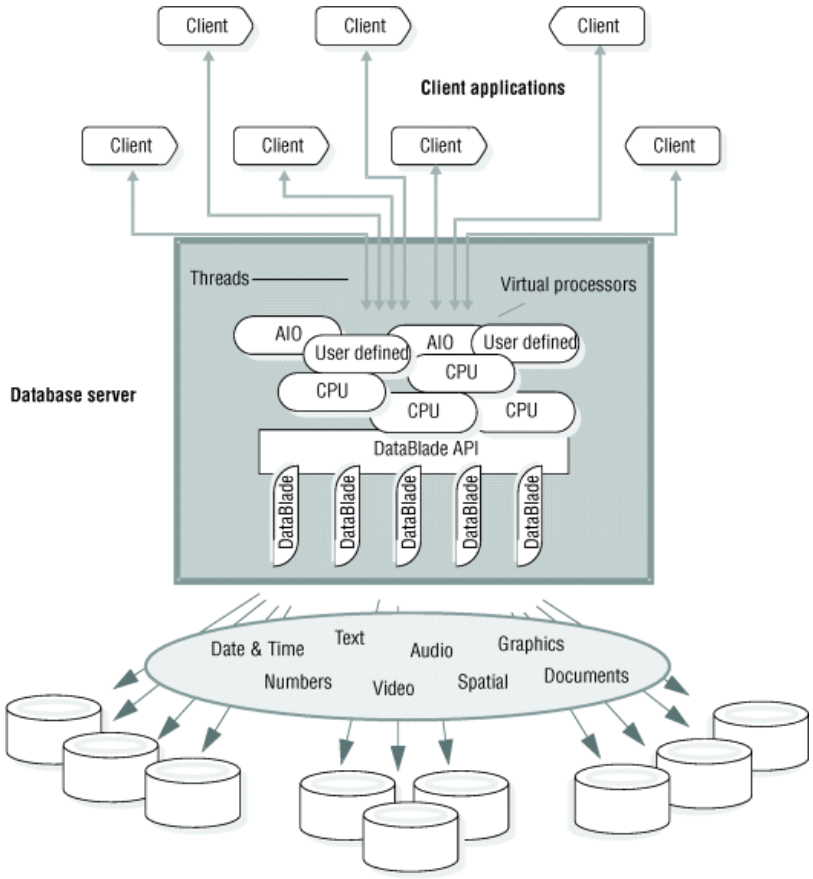

图 14: 数据库服务器

相关链接

《SinoDB 管理员参考》: VPCLASS 配置参数

启动和停止虚拟处理器 在第102页

CPU 虚拟处理器

CPU 虚拟处理器运行所有会话线程（用于处理 SQL 客户端应用程序的请求）和一些内部线程。

内部线程执行数据库服务器的内部服务。例如，侦听客户端应用程序的连接请求的线程就是内部线程。

每个 CPU 虚拟处理器都有一个与之相关联的专用内存高速缓存。每个专用内存高速缓存块由 1 到 32 个内存页组成，每个内存页是 4096 字节。数据库服务器使用专用内存高速缓存来缩短内存块的访问时间。使用 VP_MEMORY_CACHE_KB 配置参数可启用专用内存高速缓存，并指定相关内存高速缓存信息。有关更多信息，请参阅《*SinoDB*® 管理员参考》和《*SinoDB*® 性能指南》。

确定所需 CPU 虚拟处理器数

正确的 CPU 虚拟处理器数目应该是所有处理器均处于忙碌状态，但并未繁忙无法跟上进入的请求。分配的 CPU 虚拟处理器数必定不得超过计算机上硬件处理器数。

除非启用了 SINGLE_CPU_VP 配置参数，否则数据库服务器启动时会自动将 CPU 虚拟处理器的数量增加到数据库服务器计算机中的 CPU 处理器数量的一半。但是，您可以根据您的系统调整 CPU VP 的数量。

您可以配置数据库服务器在需要自动增加 CPU VP 数量，最高可达 CPU 处理器的数量。

要在数据库服务器运行状态下评估 CPU 虚拟处理器性能，请在某段时间内以固定的时间间隔重复以下命令：

```
onstat-g glo
```

如果将累积的 usercpu 和 syscpu 时间加起来接近测试期间 100% 实际耗时时间， 如果有可运行的 CPU，则添加另一个 CPU 虚拟处理器。

使用 VPCLASS 配置参数指定 CPU 虚拟处理器的以下信息：

- 最初类要启动的虚拟处理器数
- 类要运行的最大虚拟处理器数
- CPU 类虚拟处理器的处理器专用
- 禁用优先级迟滞
- 数据库服务器是否依需要自动添加 CPU 虚拟处理器

除了要考虑计算机中的 CPU 数和连接到数据库服务器的用户数外，还要考虑在 CPU 虚拟处理器或用户定义的虚拟处理器上运行的用户定义例程和 DataBlade® 模块（这些是用户定义的例程的集合）。

> 注： 使用 VPCLASS 配置参数而不是以下已停用的配置参数：AFF_SPROC、 AFFNPROCS、 NOAGE、 NUMCPUVPS 和 NUMAIOVPS。

相关链接

《SinoDB 管理员参考》: VPCLASS 配置参数

在 *CPU* 或网络虚拟处理器上运行轮询线程 在第96页

将 *UDR* 指定给用户定义的虚拟处理器类 在第92页

《SinoDB 管理员参考》: VPCLASS 配置参数

《SinoDB 管理员参考》: onstat -g glo 命令: 显示全局多线程信息

在多处理器计算机上运行

如果在多处理器计算机上运行多个 CPU 虚拟处理器，请将 onconfig 文件中的 MULTIPROCESSOR 参数设置为 1。

当 MULTIPROCESSOR 参数设置为 1 时，数据库服务器以适于多处理器计算机的方式执行锁定操作。有关设置多处理器模式的信息，请参阅《*SinoDB*® 管理员参考》 中有关配置参数的章节。

在单处理器计算机上运行

如果在单处理器计算机上运行数据库服务器，请将 MULTIPROCESSOR 配置参数设置为 0。要运行只有一个 CPU 虚拟处理器的数据库服务器，请将 SINGLE_CPU_VP 参数设置为 1。

将 MULTIPROCESSOR 参数设置为0 可使数据库服务器绕过多处理器计算机上多个进程所需的锁定。有关 MULTIPROCESSOR 配置参数的信息，请参阅《*SinoDB*® 管理员参考》。

将 SINGLE_CPU_VP 参数设置为 1 允许数据库服务器绕过运行多 CPU 虚拟处理器时会产生的一些互斥调用。有关 SINGLE_CPU_VP 配置参数信息，请参阅《*SinoDB*® 管理员参考》。

> 重要： 即使在据库服务器仅启动一个 CPU 虚拟处理器，将 *VPCLASS num* 设置为 1，并将 SINGLE_CPU_VP 设置为0 也不会减少互斥调用的数量。您必须将 SINGLE_CPU_VP 设置为 1 以减少运行单个CPU虚拟处理器时执行的锁存量。

SINGLE_CPU_VP 参数设置为 1 将对数据库服务器施加以下两个重要限制：

- 仅允许一个 CPU 虚拟处理器。

  当数据库服务器处于联机模式时，不能添加 CPU 虚拟处理器。
- 禁止用户定义的类。（但是，用户仍然可以定义直接在 CPU VP 上运行的例程。）

有关更多信息，请参阅在联机模式下添加虚拟处理器 在第103页。

在联机模式下添加和删除 CPU 虚拟处理器

当数据库服务器处于联机模式时，您可以添加或删除 CPU 类虚拟处理器。

有关如何添加或删除 CPU 类虚拟处理器的指示信息，请参阅在联机模式下添加虚拟处理器 在第103页和删除 *CPU* 和用户定义的虚拟处理器 在第103页。

防止优先级迟滞
某些操作系统会随着处理时间的累积降低长时间运行的进程的优先级，操作系统的这种功能称为优先级迟滞。

优先级迟滞会导致数据库服务器进程的性能随着时间的推移而降低。然而，在某些情况下，可以通过操作系统禁用此功能并使长时间运行的进程保持在高优先级中运行。

要确定计算机上优先级迟滞是否可用，请参阅安装随附的机器说明文件，在本指南 『简介』 中有说明。

如果可通过操作系统禁用优先级迟滞，那么可在 VPCLASS 配置参数中为优先级条目指定 noage 来禁用该功能。有关更多信息，请参阅《*SinoDB*® 管理员参考》。

处理器专用
数据库服务器支持将 CPU 虚拟处理器自动绑定到支持处理器专用的多处理器计算机上的处理器。

数据库服务器分发手册包含机器说明文件，该文件包含有关您的数据库服务器版本是否支持此功能的信息。当把一个 CPU 虚拟处理器指定给特定的 CPU 时，该虚拟处理器只运行在该 CPU 上，但其他进程也可以运行在该 CPU 上。

使用带有 *aff* 选项的 VPCLASS 配置参数可以在支持处理器专用的多处理器计算机上实现处理器专用。

下图说明了处理器专用的概念。

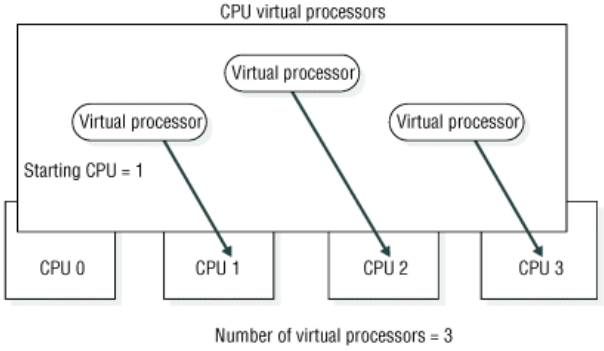

图 15: 处理器专用

仅 UNIX: 请参阅机器说明文件，查看您的 UNIX™ 平台是否支持处理器专用。

使用 VPCLASS 配置参数设置处理器专用
使用 VPCLASS 配置参数來设置处理器专用，您可以指定要为其指定虚拟处理器的单个处理器或某一范围内的处理器。

当指定某一范围内的处理器时，还可以指定一个递增值，用于指示将范围中哪些 CPU 指定给虚拟处理器。例如，可以指定将虚拟处理器分配给范围 0-6 中的每隔一个 CPU，从 CPU 0 开始。

```
VPCLASS CPU,num=4,aff=(0-6/2)
```

虚拟处理器分配给 CPU 0、2、4、6。

如果指定 VPCLASS CPU,num=4,aff=(1-10/3)，将虚拟处理器分配给范围 1-10 中的每隔两个 CPU，从 CPU 1 开始。虚拟处理器分配给 CPU 1、4、7、10。

当指定多个值或范围时，这些值和范围不必是递增的，也不需要有特定顺序。例如，可以指定 aff=(8,12,7-9,0-6/2)。

数据库服务器以循环方式将 CPU 虚拟处理器分配给 CPU，从 aff 选项中指定的第一个处理器编號开始。如果指定的 CPU 虚拟处理器数量多于物理 CPU，那么数据库服务器会继续从第一个 CPU 开始分配 CPU 虚拟处理器。例如，假设您指定下面的 VPCLASS 设置 ：

```
VPCLASS cpu,num=8,aff=(4-7)
```

数据库服务器会进行以下分配：

- CPU 虚拟处理器编号0到 CPU 4
- CPU 虚拟处理器编号1到 CPU 5
- CPU 虚拟处理器编号2到 CPU 6
- CPU 虚拟处理器编号3到 CPU 7
- CPU 虚拟处理器编号4到 CPU 4
- CPU 虚拟处理器编号5到 CPU 5
- CPU 虚拟处理器编号6到 CPU 6
- CPU 虚拟处理器编号7到 CPU 7

有关更多的信息，请参阅《*SinoDB*® 管理员参考》中的 VPCLASS 参数配置。

用户定义的虚拟处理器类

可以定义特殊的虚拟处理器类以运行用户定义的例程或运行 DataBlade® 模块。

用户定义的例程通常是用来支持用户定义的数据类型。如果不想要用户定义的例程运行在 CPU 类 （这是缺省值），您可以将其指定给用户定义的虚拟处理器 (VP) 类。用户定义的虚拟处理器类也称为扩展虚拟处理器 。

这些主题提供以下有关用户定义的虚拟处理器的信息 ：

- 何时在用户定义的 VP 而非 CPU VP 中运行 C 语言 UDR
- 如何将 C 语言 UDR 指定给一个特定的用户定义的 VP 类
- 当数据库服务器处于联机模式时，如何添加和删除用户定义的 VP

确定所需的用户定义的虚拟处理器数

您可以指定与操作系统允许的一样多的用户定义的虚拟处理器。

如果运行多个 UDR 或者以 UDR 运行并行 PDQ 查询，那么必须配置多个用户定义的虚拟处理器。

If you run many UDRs or parallel PDQ queries with UDRs, you must configure more user-defined virtual processors.

用户定义的虚拟处理器

用户定义的虚拟处理器类可保护数据库服务器免遭行为不良的用户定义的例程的破坏。

行为不良的用户定义的例程至少有以下特征之一 ：

- 不会将控制权让给其他线程
- 使用阻绝式操作系统调用
- 修改全局 VP 状态

行为良好的 C 语言 UDR 没有这些特征。仅在 CPU VP 中运行行为良好的 C 语言 UDR 。

> 警告： 在 CPU VP 中执行行为不良的例程会导致对数据库服务器操作的严重干扰，并可能导致其失败或行为反常。此外，例程本身可能不会产生正确的结果。

为确保安全执行，将任何行为不良的用户定义的例程指定给用户定义的虚拟处理器类。用户定义的 VP 除去 CPU VP 类上以下编程限制 ：

- 定期让出处理器
- 消除阻绝式 I/O 调用

运行在用户定义的虚拟处理器类中的函数不需要让出处理器，并且它们可能发出直接的文件系统调用阻挡虚拟处理器的进一步处理直到 I/O 完成。

用户查询的正常处理不受 C 语言 UDR 的不良行为影响，因为这些 UDR 不在 CPU 虚拟处理器中执行。有关行为不良的例程的更多详细信息，请参阅 《*SinoDB*® *DataBlade*® *API* 程序员指南》。

指定用户定义的虚拟处理器
带有 *vpclass* 选项的 VPCLASS 参数定义用户定义的 VP 类。还可以指定非让步的用户定义的虚拟处理器。

有关更多信息，请参阅设置虚拟处理器配置参数 在第102页 和 《*SinoDB*® 管理员参考》 中有关参数配置的主题。

将 UDR 指定给用户定义的虚拟处理器类
SQL CREATE FUNCTION 语句注册用户定义的例程。

以下的 CREATE FUNCTION 语句注册用户定义的例程 GreaterThanEqual()，并指定对该例程的调用由名为 UDR 的用户定义的 VP 类执行：

```
CREATE FUNCTION GreaterThanEqual(ScottishName, ScottishName)
   RETURNS boolean
   WITH (CLASS = UDR  )
   EXTERNAL NAME '/usr/lib/objects/udrs.so'
   LANGUAGE C
```

要执行此函数， onconfig 文件必须定义 UDR 类的 VPCLASS 参数。否则，对 GreaterThanEqual 函数的调用将失败。

> 提示： CLASS 例程修饰符可以为 VP 类指定任何名称。在注册 UDR 时，类名称不需要存在。然而，当尝试运行 UDR 且其指定了用于执行的用户定义 VP 类时，此类必须存在并已为其指定了虚拟处理器。

要配置 UDR 类，请在 onconfig 文件中包含如下类似行。此行将 UDR 类配置为具有两个虚拟处理器且没有优先级迟滞。

```
VPCLASS      UDR  ,num=2,noage
```

前面的行将 UDR VP 类定义为让步 VP 类；即，此 VP 类允许 C 语言 UDR 让步给其他必须访问 UDR VP 类的线程。 有关如何使用 VPCLASS 配置参数的更多信息，请参阅《*SinoDB*® 管理员参考》。

有关 CREATE FUNCTION 语句的更多信息，请参阅《*SinoDB*® *SQL* 指南: 语法》。

相关链接

确定所需 *CPU* 虚拟处理器数 在第88页

在联机模式下添加或删除用户定义的虚拟处理器
您可以在数据库服务器处于联机模式时添加或删除用户定义的类中的虚拟处理器。

有关如何进行这些操作的指示信息，请参阅在联机模式下添加虚拟处理器 在第103页和删除 *CPU* 和用户定义的虚拟处理器 在第103页。

租户虚拟处理器类

租户虚拟处理器类是租户数据库特有的。如果您将 SinoDB® 实例配置多租户，那么您可以指定租户数据库的会话线程在租户虚拟处理器而非 CPU 虚拟处理器中运行。

在创建租户数据库时，可通过定义虚拟处理器的类和数量来创建租户虚拟处理器类。可将租户虚拟处理器类指定给多租户数据库。设置 VP_MEMORY_CACHE_KB 配置参数为每个 CPU 虚拟处理器和租户虚拟处理器创建专用内存高速缓存。

当包含某个租户虚拟处理器类定义的所有租户数据库被删除时，该虚拟处理器类将自动被删除。

相关链接

多租户 在第208页
《*SinoDB* 管理员参考》: *tenant create* 参数: 创建租户数据库 *(SQL* 管理 *API)*
《*SinoDB* 管理员参考》: *tenant update* 参数: 更改租户数据库属性 *(SQL* 管理 *API)*
《*SinoDB* 管理员参考》: *VP_MEMORY_CACHE_KB* 配置参数

《*SinoDB* 管理员参考》: *onstat -g vpcache* 命令: 显示 *CPU* 虚拟处理器和租户虚拟处理器专用内存高速缓存的统计信息

Java™ 虚拟处理器

Java™ UDR 和 Java™ 应用程序在专门的虚拟处理器（称为 *Java*™ 虚拟处理器 (JVP)）上运行。

JVP 在其代码中嵌入 Java™ 虚拟机 (JVM)。JVP 具有與 CPU VP 相同的能力可以處理所有 SQL 查詢。

可以指定操作系统允許的任意數量的 JVP。如果运行多个 Java™ UDR 或者以 Java™ UDR 运行并行 PDQ 查询，您必须配置更多 JVP。有关 Java™ 编写的 UDR 的更多信息，请参阅《*SinoDB*® *J/Foundation* 开发者指南》。

使用带有 jvp 关键字的 VPCLASS 配置参数来配置 JVP。有关更多信息，请参阅《*SinoDB*® 管理员参考》的配置参数章节。

Use the VPCLASS configuration parameter with the jvp keyword to configure JVPs. For more information, see the configuration parameters chapter in the 《*SinoDB*® 管理员参考》.

磁盘 I/O 虚拟处理器

The following classes of virtual processors perform disk I/O: PIO (physical-log I/O), LIO (logical-log I/O), AIO (asynchronous I/O), and CPU (kernel-asynchronous I/O).以下虚拟处理器的类执行磁盘 I/O：PIO（物理日志 I/O）、LIO（逻辑日志 I/O）、AIO（异步 I/O）和CPU（内核异步 I/O）。

除非物理日志文件和逻辑日志文件位于原始磁盘空间，并且数据库服务器已实现 KAIO，否则 PIO 类将执行所有到物理日志文件的 I/O，LIO 类将执行所有到逻辑日志文件的 I/O。

在不支持 KAIO 的操作系统上，数据库服务器使用虚拟处理器的 AIO 类来执行与物理或逻辑日志不相关的数据库 I/O。

当 KAIO 可用于平台时，数据库服务器使用 CPU 类来执行 KAIO。如果数据库服务器实现 KAIO，那么 KAIO 执行所有到原始磁盘空间的 I/O，包含到物理和逻辑日志的 I/O。

仅 UNIX: 要了解您的 UNIX™ 平台是否支持KAIO，请参阅机器说明文件。

仅 Windows: Windows™ 支持KAIO。

有关非日志记录 I/O 的更多信息，请参阅异步 *I/O* 在第94页。

I/O 优先级

数据库服务器通过将不同 I/O 类型指定给不同的虚拟处理器类，并通过分配非日志记录 I/O 队列的优先级来划分磁盘 I/O 的优先级。

例如，划分优先级可确保高优先级日志 I/O 绝不会排列在具有低优先级的临时文件的写操作后面。数据库服务器对其执行的不同类型的磁盘 I/O 划分优先级，如下表所示。

表 15: 数据库服务器如何划分磁盘 I/O 的优先级

优先级	I/O 类型	VP 类
第 1级	逻辑日志 I/O	CPU 或 LIO
第 2 级	物理日志 I/O	CPU 或 PIO
第 3 级	数据库 I/O	CPU 或 AIO
第 3 级	页清除 I/O	CPU 或 AIO
第 3 级	预读取 I/O	CPU 或 AIO

逻辑日志 I/O
虚拟处理器的 LIO 类执行对逻辑日志文件的 I/O。

在以下情况的逻辑日志文件执行 I/O：

- 没有实现 KAIO。
- 逻辑日志文件在熟磁盘空间

仅当实现了 KAIO 并且逻辑日志文件位于原始磁盘空间时，数据库服务器才会使用 CPU 虚拟处理器中的 KAIO 线程来对逻辑日志执行 I/O。

逻辑日志文件存储使数据库服务器能够回滚事务并且从系统故障中恢复的数据。逻辑日志文件的 I/O 是数据库服务器执行的最高优先权的磁盘 I/O。

如果逻辑日志文件位于无镜像的数据库空间中，那么数据库服务器只运行一个 LIO 虚拟处理器。如果逻辑日志文件位于有镜像的数据库空间中，那么数据库服务器运行两个 LIO 虚拟处理器。此虚拟处理器类没有任何与其关联的参数。

物理日志 I/O
虚拟处理器的 PIO 类对物理逻辑文件执行 I/O。

对以下情况中的物理日志文件执行 I/O：

- 没有实现 KAIO。
- 物理日志文件存储在缓冲文件块中。

仅当实现了 KAIO 并且物理日志文件位于原始磁盘空间时，数据库服务器才会使用 CPU 虚拟处理器中的 KAIO 线程来对物理日志执行 I/O。物理日志文件存储自上一个检查点后已更改的数据库空间页的前映像。（有关檢查点的更多信息，请参阅检查点 在第266页。）在恢复开始时，数据库服务器会在处理来自逻辑日志的事务前，使用物理日志文件将前映像还原到自上一个检查点以来已更改的数据库空间页上。物理日志的 I/O 是继逻辑日志文件 I/O 之后第二高优先级的 I/O。

如果物理日志文件位于无镜像的数据库空间中，那么数据库服务器只运行一个 PIO 虚拟处理器。如果逻辑日志文件位于有镜像的数据库空间中，那么数据库服务器运行两个 PIO 虚拟处理器。此虚拟处理器类没有任何与其关联的参数。

异步 I/O
.数据库服务器异步执行数据库 I/O，这意味着 I/O 将独立于请求 I/O 的线程进行排队与执行。异步执行 I/O 允许该发出请求的线程在 I/O 执行时继续工作。

数据库服务器使用下列设备之一异步执行所有数据库 I/O：

- AIO 虚拟处理器
- 平台（该平台支持 KAIO）上的 KAIO

数据库 I/O 包括 SQL语句的 I/O、预读取 I/O、页清除 I/O和检查点 I/O。

内核异步 I/O
The database server implements KAIO by running a KAIO thread on the CPU virtual processor. The KAIO thread performs I/O by making system calls to the operating system, which performs the I/O independently of the virtual processor.数据库服务器通过在 CPU 虚拟处理器上运行 KAIO 线程来实现 KAIO。KAIO 线程通过对操作系统（该操作系统可执行独立于虚拟处理器的 I/O）进行系统调用来执行 I/O。

当以下条件存在时，数据库服务器使用 KAIO：

- 计算机和操作系统对其支持。
- 实现性能提升。
- I/O 是对原始磁盘空间的。

对于磁盘 I/O，KAIO 线程可产生比 AIO 虚拟处理器更好的性能，因为它不需要在 CPU 和 AIO 虚拟处理器之间切换。

仅 UNIX: 当SinoDB® 转移到支持该功能的平台时，SinoDB® 将实现 KAIO。数据库服务器管理者不用配置 KAIO。想了解您的平台是否支持 KAIO，请参阅机器说明文件。

仅 Linux: 内核异步 I/O（KAIO）缺省为启用。可以通过在启动服务器进程的环境中指定 KAIOOFF=1 来将其禁用。

在Linux™平台上，并行 KAIO 请求存在全系统最大数量的限制。/proc/sys/fs/aio-max-nr 文件包含该值。Linux™ 系统管理员能增加该值，例如，通过使用该命令：

```
# echo new_value > /proc/sys/fs/aio-max-nr
```

在 /proc/sys/fs/aio-nr 文件中可以看到所有操作系统进程已分配请求的当前数量。

默认情况下，SinoDB® 将分配最大请求数的一半，并将他们平均分配给已配置的 CPU 虚拟处理器数。可以使用环境变量 KAIOON 来控制分配给每个 CPU 虚拟处理器的请求数。为此，要做到这点，可在启动 SinoDB® 前将 KAIOON 设置为所需的值。

KAIOON 最小值是 100。如果 Linux™ 将耗尽 KAIO 资源，例如动当态添加了许多 CPU 虚拟处理器时，online.log 文件中将打印警告。如果发生这种情况，Linux™ 系统管理员必须按之前描述添加 KAIO 资源。

AIO 虚拟处理器

如果平台不支持 KAIO 或者 I/O 是对缓冲文件块，那么数据库服务器将通过虚拟处理器的 AIO 类执行数据库 I/O。所有 AIO 虚拟处理器平等地服务其类中的所有 I/O 请求。

数据库服务器将根据块的文件名为每个磁盘块指定一个队列（有时候称为*gfd* 队列）。数据库服务器根据最小化磁盘头移动算法来排序队列中的 I/O 请求。AIO 虚拟处理器为循环方式下有暂挂工作的队列提供服务。所有其他的非块 I/O 会排列在 AIO 队列中。

使用带有 aio 关键字的 VPCLASS 参数来指定数据库服务器初次启动的 AIO 虚拟处理器的数量。在数据库服务器处于联机模式时，您可以启动其他的 AIO 虚拟处理器，但不能删除 AIO 虚拟处理器。

当服务器检测到 AIO VP 的处理速度跟不上 I/O 工作负载时，您可以使数据库服务器添加 AIO 虚拟处理器和清空程序线程。在 VPCLASS 配置参数设置中包含 autotune=1 关键字。

手动控制 AIO 虚拟处理器数量

分配 AIO 虚拟处理器的目的是分配足够的虚拟处理器从而可以保持 I/O 请求队列的长度比较短，也就是说，队列中具有的 I/O 请求尽可能少。当 *gfd* 队列总是很短时，表示磁盘设备 I/O 处理速度和请求发生的速度一样快。

onstat-g ioq 命令显示有关 I/O 队列的长度和其他统计信息。可以使用该命令为 AIO 虚拟处理器监视 *gfd* 队列长度。

一个 AIO 虚拟处理器可能已足够：

- 如果数据库服务器在您的平台上实现内核异步 I/O（KAIO），并且您的所有数据库空间都由原始磁盘空间组成。
- 如果您的文件系统支持用于数据库空间块的页大小的直接 I/O，并且您使用直接 I/O。

对于每个由缓冲文件块组成的活动数据库空间分配两个 AIO 虚拟处理器。

- 如果数据库服务器实现 KAIO，但是您正在对块使用一些缓冲文件。
- 如果系统不支持对块的 KAIO。

如果 KAIO 不在您的平台上实现，那么为数据库服务器经常访问的每个磁盘分配两个 AIO 虚拟处理器。

如果您使用熟文件并使用 DIRECT_IO 配置参数启用直接 I/O，那么您可能可以减少 AIO 处理器的数量。

如果数据库服务器实现 KAIO 并且使用 DIRECT_IO 配置参数启用直接 I/O，那么 SinoDB® 将尝试使用 KAIO，这样可能就不需要多个 AIO 虚拟处理器。然而，即使启用了直接 I/O，如果文件系统不支持直接 I/O 或 KAIO，您仍然必须为缓冲文件块组成或不使用 KAIO 的每个活动数据库空间分配两个AIO虚拟处理器。

临时数据库空间不使用直接 I/O。如果有临时数据库空间，那么可能需要多个 AIO 虚拟处理器。

分配足够的 AIO 虚拟处理器以满足 I/O 请求的最大量。一般来说，分配过多 AIO 虚拟处理器不会产生不利影响。

相关链接

《SinoDB 管理员参考》：VPCLASS 配置参数

网络虚拟处理器

客户端能通过以下方式连接数据库服务器：网络连接、管道或共享内存。

网络连接可以由远程计算机上的客户端建立，或油本地计算机上的客户端通过模拟来自远程计算机的连接（称为本地回送连接）建立。

指定网络连接

一般来说，DBSERVERNAME 和 DBSERVERALIASES 参数定义在 sqlhosts 文件或注册表中具有相应条目的 dbservername。sqlhosts 中每个 dbservername 都有指定接口/协议组合的 nettype 条目。数据库服务器为每个唯一的 nettype 条目运行一个或多个轮询线程。

NETTYPE 配置参数为接口/协议组合提供可选配置信息。可将其用于为接口/协议组合分配多个轮询线程，并指定运行轮询线程的虚拟处理器类（CPU 或 NET）。

有关 NETTYPE 配置参数的完整描述，请参阅《*SinoDB*® 管理员参考》。

相关链接

sqlhosts 连接信息 在第49页

在 CPU 或网络虚拟处理器上运行轮询线程

轮询线程可以在 CPU 虚拟处理器或网络虚拟处理器上运行。

通常来说，特别在单处理器计算机上，轮询线程在 CPU 虚拟处理器上运行效率更高。然而，在具有多个远程客户端的多处理器计算机来说，这种情况可能并不是成立。

NETTYPE 参数有个名为 *vp class* 的可选条目，可用于指定 CPU 或 NET 以分别表示 CPU 虚拟处理器或网络虚拟处理器。

如果没有为与 DBSERVERNAME 变量相关联的接口/协议组合（轮询线程）指定虚拟处理器类，缺省类将设置为 CPU。数据库服务器假定与 DBSERVERNAME 相关联的接口/协议组合为主接口/协议组合，并且假定该设置为最有效率的。

对于其他接口/协议组合，如果没有指定 vp class，缺省值将是 NET。

当数据库服务器处于联机模式时，您无法删除正在运行轮询或侦听线程的 CPU 虚拟处理器。

> 重要： 必须仔细区分用于网络连接的轮询线程和用于共享内存连接的轮询线程，每个 CPU 虚拟处理器上只能运行一个。TCP 连接必须只能在网络虚拟处理器中，并必须仅具有维持响应所需的最小连接数。共享内存连接必须只能在 CPU 虚拟处理器中，并在每个 CPU 虚拟处理器中运行。

相关链接

确定所需 *CPU* 虚拟处理器数 在第88页

指定网络虚拟处理器的数量

每个轮询线程需要单独的虚拟处理器，因此当您为接口/协议组合指定轮询线程的数量时，您间接地指定了网络虚拟处理器的数量，并且指定这些虚拟处理器将由 NET 类运行。

如果您将 vp class 指定为 CPU，那么您必须配置足够的 CPU 虚拟处理器来运行轮询线程。如果数据库服务器中没有 CPU 虚拟处理器可以运行 CPU 轮询线程，那么数据库服务器会启动指定类的网络虚拟处理器来运行 CPU 轮询线程。

对于大多数系统来说，每个网络接口/协议组合有一个轮询线程和一个虚拟处理器就足够了。对于有 200 个或更多用户的系统来说，运行额外的网络虚拟处理器可能可以提高吞吐量。在这种情况下，您必须通过试验来确定每个网络接口/协议组合的最佳虚拟处理器数。

为客户端/服务器连接指定侦听和轮询线程

当启动数据库服务器时， oninit 进程会为 onconfig 文件中 DBSERVERNAME 和 DBSERVERALIASES 参数指定的每个 dbservername 启动一个内部线程（称为 侦听线程）。

要为每个 dbservername 条目指定侦听端口，请在 sqlhosts 为其指定 hostname 和 service name 条目的唯一组合。例如，下表中显示的 sqlhosts 文件条目或注册表项使数据库服务器 soc_ol1 为主机或网址 myhost 上的 port1 启动侦听线程，

表 16: 每个侦听端口的侦听线程

dbservername	nettype	hostname	service name
soc_ol1	onsoctcp	myhost	port1

侦听线程开启端口并请求指定的接口/协议组合的一个轮询线程来监视客户端请求。对于正在使用的连接的轮询线程可以在 CPU 虚拟处理器或网络虚拟处理器上运行。有关轮询线程数的信息，请参阅 指定网络虚拟处理器的数量 在第96页。

有关如何指定接口/协议组合的轮询线程是在 CPU 还是网络虚拟处理器上运行的信息，请参阅 在 *CPU* 或网络虚拟处理器上运行轮询线程 在第96页 以及《*SinoDB*® 管理员参考》 中的NETTYPE 配置参数。

当轮询线程接收到来自客户端的连接请求时，它会将请求传递给该端口的侦听线程。侦听线程认证用户、建立到数据库服务器的连接，并启动 sqlexec 线程，该线程是为客户端执行主处理的会话线程。下图说明了侦听线程和轮询线程在建立与客户端应用程序的连接时的角色。

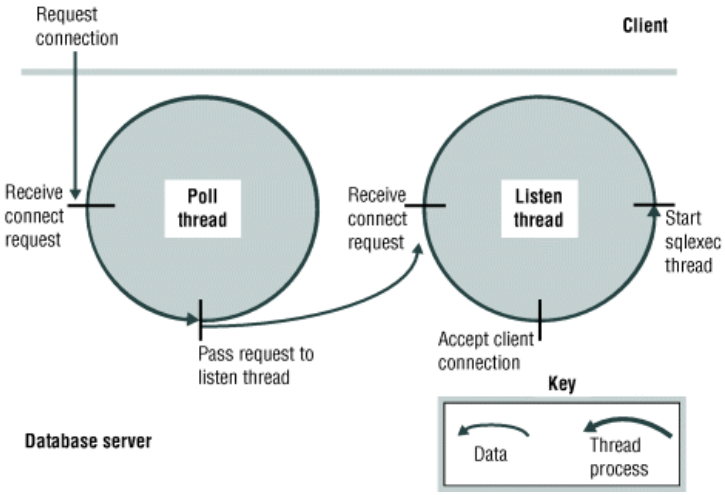

图 16: 侦听线程和轮询线程在与客户端连接时的角色。

轮询线程等待来自客户端的请求，并将这些请求放在共享内存中以便 sqlexec 线程处理。对于网络连接，轮询线程将消息放在共享内存全局池的队列中。然后，轮询线程唤醒客户端的 sqlexec 线程来处理请求。sqlexec 线程将尽可能直接写回客户端而无需轮询线程帮助。通常情况下，轮询线程从客户端读取数据，而 sqlexec 线程将数据发送到客户端。。

仅 UNIX: 对于共享内存连接，轮询线程将消息放在共享内存的通信部分。

下图说明了轮询线程和 sqlexec 线程在与客户端应用程序通信时执行的基本任务。

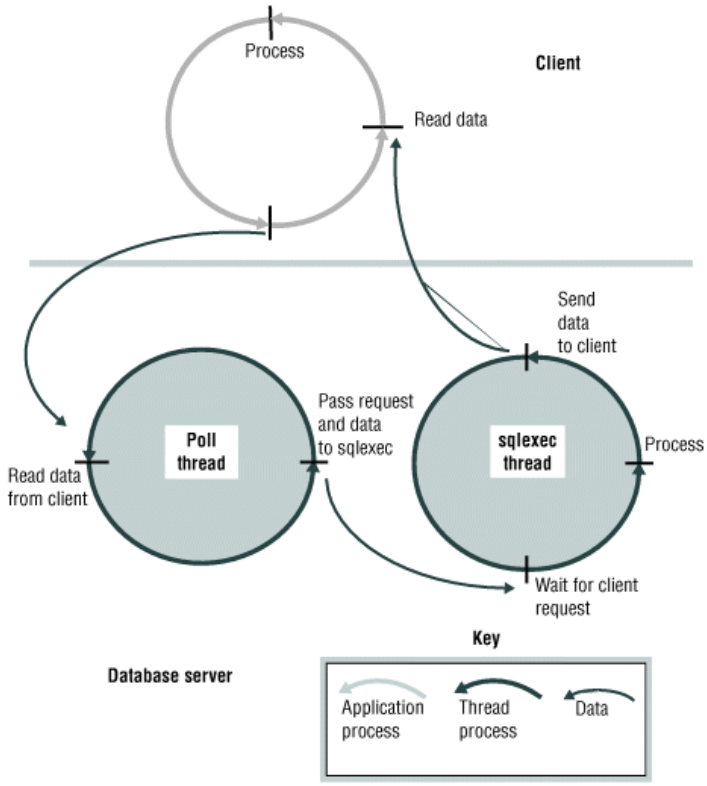

图 17: 侦听线程和 sqlexec 线程在与客户端应用程序通信时的角色

快速轮询

如果操作系统平台支持快速轮询，那么可以使用 FASTPOLL 配置参数来启用或禁用网络快速轮询。

如果有很多的连接，快速轮询将非常有用。例如，如果数据库服务器有 300 多个并发连接，那么可以启用 FASTPOLL 配置参数以获取更好的性能。通过把 FASTPOLL 配置参数设置为 1 来启用快速轮询。

如果操作系统不支持快速轮询，那么SinoDB® 将忽略 FASTPOLL 配置参数。

多个侦听线程

可以通过使用多个侦听线程来提高对连接请求的服务。

如果数据库服务器无法使用单个端口和相应的侦听线程为给定的接口/协议组合提供令人满意的连接请求服务，则可以采用下列方法来改善连接请求服务：

- 为其他端口添加侦听线程。
- 如果有 onimcsoc 或 onsoctcp 协议，可在同一端口添加侦听线程。
- 添加另一个网络接口卡。
- 使用 SQL 管理 API 或 onmode -P 命令来动态启动、停止或重启 SOCTCP 或 TLITCP 网络协议的侦听线程。

如果 onsoctcp 协议的一个端口有多个侦听线程 ，那么当 CPU VP 连接正忙时，数据库服务器可以接受新的连接。

添加侦听线程

当启动数据库服务器时， oninit 进程将为具有 DBSERVERNAME 和 DBSERVERALIASES 配置参数指定的服务器名称和别名的服务器启动侦听线程。您可以为其他端口添加侦听线程。

还可以为 onimcsoc 或 onsoctcp 协议的一个服务（端口）设置多个侦听线程。

要为其他端口添加侦听线程，首先必须使用 DBSERVERALIASES 参数来指定每个端口的 dbservername 。例如，下图中的 DBSERVERALIASES 参数为标识为 soc_ol1 的数据库服务器实例定义了两个额外的 dbservername：soc_ol2 和 soc_ol3。

```
DBSERVERNAME      soc_ol1
DBSERVERALIASES   soc_ol2,soc_ol3
```

在定义了数据库服务器的额外 dbservername 之后，必须在 sqlhosts 文件或注册表中为每个 dbservername 指定接口/协议组合和端口。每个端口由 hostname 和 servicename 条目的唯一组合来标识。 例如，下表中显示的 sqlhosts 条目使得数据库服务器为 onsoctcp 接口/协议组合启动三个侦听线程，每个定义的端口一个侦听线程。

表 17: 用于侦听单个接口/协议组合的多个端口的 sqlhosts 条目

dbservername	nettype	hostname	service name
soc_ol1	onsoctcp	myhost	port1
soc_ol2	onsoctcp	myhost	port2
soc_ol3	onsoctcp	myhost	port3

如果在接口/协议组合中包含了 NETTYPE 参数，它将应用于该接口/协议组合的所有连接。换言之，如果上表中的 onsoctcp 有 NETTYPE 参数，它将应用于显示的所有连接。在这个例子中，数据库服务器为 onsoctcp 接口/协议组合运行一个轮询线程，除非 NETTYPE 参数指定多个。有关 sqlhosts 文件或注册表中条目的更多信息，请参阅见 连接文件 在第40页。

为 onimcsoc 或 onsoctcp 协议的一个端口设置多个侦听线程

要为 onimcsoc 或 onsoctcp 协议的一个服务（端口）设置多个侦听线程，请如下指定 DBSERVERNAME 和 DBSERVERALIASES 信息：

- DBSERVERNAME <*name*>-<*n*>
- DBSERVERALIASES <*name1*>-<*n*>,<*name2*>

例如：

- 要为 DBSERVERNAME 为 ifx 的服务器开启两个侦听线程，请指定：

  ```
  DBSERVERNAME ifx-2
  ```

- 要为 DBSERVERALIASES 为 ifx_a 和 ifx_b 服务器开启两个侦听线程，请指定：

  ```
  DBSERVERALIASES ifx_a-2,ifx_b-2
  ```

添加网络接口卡
可以添加一个网络接口卡来提高性能或将数据库服务器连接到多个网络。

如果主机的网络接口卡不能提供令人满意的连接请求服务，那么您可能想要改善性能。

要支持多个网络接口卡，必须在 sqlhosts中为每个网络接口卡指定一个唯一的 主机名 （网络地址）

例如， 使用 添加侦听线程 在第98页中显示的相同 dbservername，下表中显示的 sqlhosts 文件或注册表条目将使得数据库服务器为同样的接口/协议组合启动三个侦听线程（如同 添加侦听线程 在第98页中的条目所达成的）。但是在此情况下，其中两个线程侦听一个接口卡 (myhost1) 上的端口，第三个线程侦听第二个接口卡 (myhost2) 上的一个端口。

表 18: 支持 onsoctcp 接口/协议组合的两个网络接口卡的 sqlhosts 条目示例

dbservername	nettype	hostname	service name
soc_ol1	onsoctcp	myhost1	port1
soc_ol2	onsoctcp	myhost1	port2
soc_ol3	onsoctcp	myhost2	port1

动态启动、停止、重启侦听线程
可以动态启动、停止或重启 SOCTCP 或 TLITCP 网络协议的侦听线程，而无需中断现有连接。例如，您可能想在其他服务器功能正在正常执行且不关闭服务器的情况下，停止无响应的侦听线程后重新启动新线程。

侦听线程必须在服务器的 sqlhosts 文件中定义。如有必要，可在启动、停止或重启侦听线程之前，修改 sqlhosts 条目。

动态启动、停止或重启侦听线程：

1. 运行以下其中一个 onmode -P 命令：
   - onmode -P start *server_name*
   - onmode -P stop *server_name*
   - onmode -P restart *server_name*
2. 或者，如果您是直接或远程连接到 sysadmin 数据库，则可以运行以下其中一个命令：
   - 带有 start listen 参数的 admin() 或 task() 命令，格式如下 ：

     ```
     EXECUTE FUNCTION task("start listen", "server_name");
     ```
   - 带有 stop listen 参数的 admin() 或 task() 命令 ，格式如下：

     ```
     EXECUTE FUNCTION task("stop listen" ,"server_name");
     ```
   - 带有 restart listen 参数的 admin() 或 task() 命令 ，格式如下：

     ```
     EXECUTE FUNCTION task("restart listen", "server_name");
     ```

例如，以下任一命令将为服务器 ifx_serv2 启动新侦听线程：

```
onmode -P start ifx_serv2
```

```
EXECUTE FUNCTION task("start listen", "ifx_serv2");
```

通信支持模块虚拟处理器

虚拟处理器的通信支持模块 (CSM) 类执行通信支持服务和通信支持模块功能。

数据库服务器启动与启动的 CPU 虚拟处理器数量相同数量的 CSM 虚拟处理器。

有关通信支持服务的更多信息，请参阅 客户端/服务器通信 在第33页。

审计虚拟处理器

通过将 onconfig 文件中的 ADTMODE 参数设置为 1 来打开审计模式时，数据库服务器将启动一个审计类 (ADT) 的虚拟处理器。

有关数据库服务器审计的更多信息，请参阅 《*SinoDB®* 安全指南》。

综合性虚拟处理器

综合性虚拟处理器服务系统调用的请求（如获取有关当前用户或主机系统名的信息），这些调用可能需要很大的堆栈。

该虚拟处理器仅运行一个线程；使用 128KB 的堆栈执行。

Basic Text Search 虚拟处理器

要运行 Basic Text Search 查询需要使用 Basic Text Search 虚拟处理器。

创建 Basic Text Search 索引时，会自动添加 Basic Text Search 虚拟处理器。

Basic Text Search 虚拟处理器运行时不会让步，它一次处理一个索引操作。要同时运行多个 Basic Text Search 索引操作与查询，请创建额外的 Basic Text Search 虚拟处理器。

使用带有 BTS 关键字的 VPCLASS 配置参数来配置 Basic Text Search 虚拟处理器。例如，要增加 5 个 BTS 虚拟处理器，请将以下行添加到 onconfig 并重启数据库服务器：

```
VPCLASS bts,num=5
```

使用 onmode -p 命令可动态添加 BTS 虚拟处理器，例如：

```
onmode -p 5 bts
```

相关链接

《SinoDB 管理员参考》: VPCLASS 配置参数

《SinoDB 管理员参考》: onmode -p: 添加或删除虚拟处理器

《SinoDB 数据库扩展用户指南》: Basic Text Search

MQ 消息传递虚拟处理器

要使用 MQ 消息传递，需要 MQ 虚拟处理器。

执行 MQ 消息传递事务时，将自动创建 MQ 虚拟处理器。

MQ 虚拟处理器运行时不会让步；它一次处理一个操作。要同时执行多个 MQ 消息传递事务，请创建额外的 MQ 虚拟处理器。

使用带有 MQ 关键字的 VPCLASS 配置参数来配置 MQ 虚拟处理器。例如，要添加五个 MQ 虚拟处理器，请将以下行添加到 onconfig，然后重启数据库服务器。

```
VPCLASS mq,noyied,num=5
```

有关 VPCLASS 配置参数的更多信息，请参阅《*SinoDB*® 管理员参考》。有关 MQ 消息传递的更多信息，请参阅《*SinoDB*® 数据库扩展用户指南》

XML 虚拟处理器

要执行 XML 发布，需要 XML 虚拟处理器。

当执行 XML 函数时，将自动创建 XML 虚拟处理器。

一个 XML 虚拟处理器一次执行一个 XML 函数。要同时执行多个 XML 函数，请创建额外的 XML 虚拟处理器。

使用带有 IDSXMLVP 关键字的 VPCLASS 配置参数来配置XML虚拟处理器。例如，要添加五个 XML 虚拟处理器，请将以下行添加到 onconfig，然后重启数据库服务器。

```
VPCLASS idsxmlvp,num=5
```

使用 onmode-p 命令可动态添加 XML 虚拟处理器，例如：

```
onmode -p 5 idsxmlvp
```

有关 VPCLASS 配置参数以及 onmode 实用程序的更多信息，请参阅《*SinoDB*® 管理员参考》。有关 XML 发布的更多信息，请参阅《*SinoDB*® 数据库扩展用户指南》。

管理虚拟处理器

这些主题描述如何设置会影响数据库服务器虚拟处理器的配置参数，以及如何启动和停止虚拟处理器。

有关虚拟处理器类的描述以及有关必须为每个类指定多少虚拟处理器的建议，请参阅 虚拟处理器和线程 在第81页。

设置虚拟处理器配置参数

使用 VPCLASS 配置参数来指定虚拟处理器 (VP) 类、创建用户定义的虚拟处理器，以及指定选项（如服务器启动的 VP 数、类所允许的最大 VP 数，以及当处理器专用可用时 VP 对 CPU 的分配）。

下表列出用于配置虚拟处理器的配置参数。

表 19: 配置虚拟处理器的配置参数

参数	描述
MULTIPROCESSOR	设置为 1 以支持多个 CPU 虚拟处理器；对于单个 CPU VP，设置为 0
NETTYPE	设置网络协议线程和虚拟处理器的参数
SINGLE_CPU_VP	设置为 0 以启用用户定义的 CPU VP；对于单个 CPU VP，设置为任何其他设置值
VPCLASS	每个定义一个 VP 类和其属性（例如：当服务器启动时，该类启动多少个 VP ）
VP_MEMORY_CACHE_KB	通过为每个 CPU 虚拟处理器创建专用内存高速缓存来提升访问内存块的速度。

相关链接

《SinoDB 管理员参考》: VPCLASS 配置参数
《SinoDB 管理员参考》: MULTIPROCESSOR 配置参数
《SinoDB 管理员参考》: SINGLE_CPU_VP 配置参数
《SinoDB 管理员参考》: VP_MEMORY_CACHE_KB 配置参数

启动和停止虚拟处理器

当启动数据库服务器时， oninit 实用程序将启动直接和间接指定的虚拟处理器的数量和类型。

主要通过配置参数来配置虚拟处理器，而对于网络虚拟处理器，则是通过 sqlhosts 信息中的参数来配置。

可以使用数据库服务器来启动最多 1000 个虚拟处理器。

在数据库服务器处于联机模式时，如有必要，您可以启动更多的虚拟处理器来提高性能。

在数据库服务器处于联机模式时，您可以删除 CPU 虚拟处理器和用户定义类的虚拟处理器。

若要关闭数据库服务器并停止所有虚拟处理器，请使用 onmode -k 命令。

相关链接

虚拟处理器类 在第87页

《SinoDB 管理员参考》: onmode -k, -m, -s, -u, -j: 更改数据库服务器模式

在联机模式下添加虚拟处理器

当数据库服务器处于运行中时，您可以使用 onmode 实用程序的 -p 选项来为一些虚拟处理器类启动额外的虚拟处理器。

可以使用 onmode 实用程序的 -p 选项来启动额外的虚拟处理器。可以使用 NETTYPE 配置参数来添加网络虚拟处理器。

還可以启动用户定义类的额外的虚拟处理器以运行用户定义的例程。有关用户定义的虚拟处理器的更多信息，请参阅 将 UDR 指定给用户定义的虚拟处理器类 在第92页。

相关链接

《SinoDB 管理员参考》: onmode -p: 添加或删除虚拟处理器

《SinoDB 管理员参考》: NETTYPE 配置参数

在联机模式下使用 onmode 添加虚拟处理器

当数据库服务器处于联机模式時，使用 onmode 命令的-p选项来添加虚拟处理器。

以正数指定想要添加的虚拟处理器的数量。可以选择在虚拟处理器数量前加上一个加号 （+）。在数字之后，以小写字母指定虚拟处理器的类。例如，以下任一命令都可以启动四个 AIO 类的额外的虚拟处理器：

```
onmode -p 4 aio

onmode -p +4 aio
```

onmode 实用程序立即启动额外的虚拟处理器。

一次仅可以添加一个类的虚拟处理器。若要添加另一个类的虚拟处理器，必须再次运行 onmode 。

添加网络虚拟处理器

当添加网络虚拟处理器时，将添加轮询线程，每个轮询线程都需要在自己的虚拟处理器上运行。

在以下示例中，轮询线程总共处理 240 个连接 ：

```
NETTYPE ipcshm,4,60,CPU # Configure poll thread(s) for nettype
```

对于 ipcshm，轮询线程的数目对应于内存段的数量。例如，如果 NETTYPE 设置为 3100，并且您需要一个轮询线程，请将轮询线程设置为 1300。

删除 CPU 和用户定义的虚拟处理器

当数据库服务器处于联机模式时，可以使用 onmode 实用程序的 -p 选项来删除，或终止 CPU 和用户定义类的虚拟处理器。

删除 CPU 虚拟处理器

在 onmode 命令之后，请指定一个负数，该负数是您想要删除的虚拟处理器的数量，然后以小写字母指定 CPU 类。例如，下面的命令可以删除两个 CPU 虚拟处理器 ：

```
% onmode -p -2 cpu
```

如果尝试删除正在运行轮询线程的 CPU 虚拟处理器，将收到以下消息 ：

```
onmode: failed when trying to change the number of cpu virtual processor by -number.
```

有关更多信息，请参阅 在 CPU 或网络虚拟处理器上运行轮询线程 在第96页。

删除用户定义的虚拟处理器

在 onmode 命令之后，请指定一个负数，该负数是您想要删除的虚拟处理器的数量，然后以小写字母指定用户定义的类。例如，下面的命令删除 *usr* 类的两个虚拟处理器：

```
onmode -p -2 usr
```

仅 Windows: 在 Windows™ 中，一次只有一个用户定义的虚拟处理器类。在 onmode -p ***vpclass*** 命令中省略 *number* 参数。

有关如何创建用户定类的虚拟处理器，及将用户定义的例程指定给该类，请参阅 用户定义的虚拟处理器类 在第91页。

监视虚拟处理器

监视虚拟处理器以确定为数据库服务器配置的虚拟处理器数量对于当前活动的水平来说是否为最佳的。

有关这些 onstat -g 选项的更多信息，请参阅 《*SinoDB*® 性能指南》 中有关配置对 CPU 使用率的影响的主题。

有关 onstat-g 命令的输出示例，请参阅 《*SinoDB*® 管理员参考》中有关 onstat 实用程序的信息。

使用命令行实用程序监视虚拟处理器

使用 onstat-g 选项监视虚拟处理器。

可以使用以下选项来监视虚拟处理器 ：

onstat-g ath 命令

onstat-g ath 命令显示有关系统线程和虚拟处理器类的信息。

Onstat-g glo 命令
使用 onstat-g glo 命令来显示有关当前正在运行的虚拟存储器的信息，以及每个虚拟处理器类的累积统计信息。

有关 onstat-g glo 的输出示例，请参阅 《*SinoDB*® 管理员参考》中有关 onstat 实用程序的信息。

onstat-g ioq 命令
使用 onstat -g ioq 选项可确定是否必须分配更多虚拟处理器。 onstat-g ioq 命令显示有关 I/O 队列的长度和其他统计信息。

如果 I/O 队列的长度不断增加，那么 I/O 请求的积累速度超过 AIO 虚拟处理器处理请求的速度。如果 I/O 队列的长度继续增加显示 I/O 请求不断在积累，那么请考虑添加 AIO 虚拟处理器。

有关 onstat-g ioq 的输出示例，请参阅 《*SinoDB*® 管理员参考》。

onstat-g rea 命令
使用 onstat-g rea 选项来监视就绪队列中的线程数。

如果就绪队列中某一类的虚拟处理器（例如，CPU 类）的线程数不断增加，那么可能需要在配置中添加更多此类的虚拟处理器。

有关 onstat-g rea 的输出示例，请参阅 《*SinoDB*® 管理员参考》。

使用 SMI 表监视虚拟处理器

查询 sysvpprof 表以获得有关当前正在运行的虚拟处理器的信息。

此表包含以下各列。

列	描述
vpid	虚拟处理器的标识号
class	虚拟处理器类
usercpu	用户 CPU 已用分钟数
syscpu	系统 CPU 已用分钟数

共享内存

These topics describe the content of database server shared memory, the factors that determine the sizes of shared-memory areas, and how data moves into and out of shared memory.这些主题描述数据库服务器共享内存的内容，共享内存区域大小的决定因素，以及数据移入和移出共享内存的方式。

有关如何更改确定共享内存分配的数据库服务器配置参数的信息，请参阅 管理共享内存 在第128页。

相关链接

数据库服务器维护任务 在第30页

共享内存

共享内存是操作系统的一种功能，它允许数据库服务器线程和进程通过共享对内存池的访问来共享数据。

数据库服务器将共享内存用于以下用途 ：

- 减少内存使用和磁盘 I/O
- 执行进程之间的高速通信

共享内存使数据库服务器能够减少总体内存使用量，因为参与的进程（在此情况下，即虚拟处理器）不需要保留共享内存中数据的专用副本。

共享内存将减少磁盘 I/O，因为缓冲区（作为公共池受管）将是以整个数据库服务器范围进行清空，而不是对每个进程进行清空。而且，虚拟处理器经常可以避免从磁盘中读取数据，因为数据已经作为较早读取操作的结果存在于共享内存中了。减少磁盘 I/O 将减少执行时间。

Shared memory provides the fastest method of interprocess communication, because it processes read and write messages at the speed of memory transfers.

共享内存提供最快的进程间通信方法，因为它以内存传输速度来处理读写消息。

共享内存使用

数据库服务器使用共享内存以使得虚拟处理器和实用程序可以共享数据，并为使用 IPC 通信的本地客户端应用程序提供快速的通信通道。

下图说明了共享内存模式。

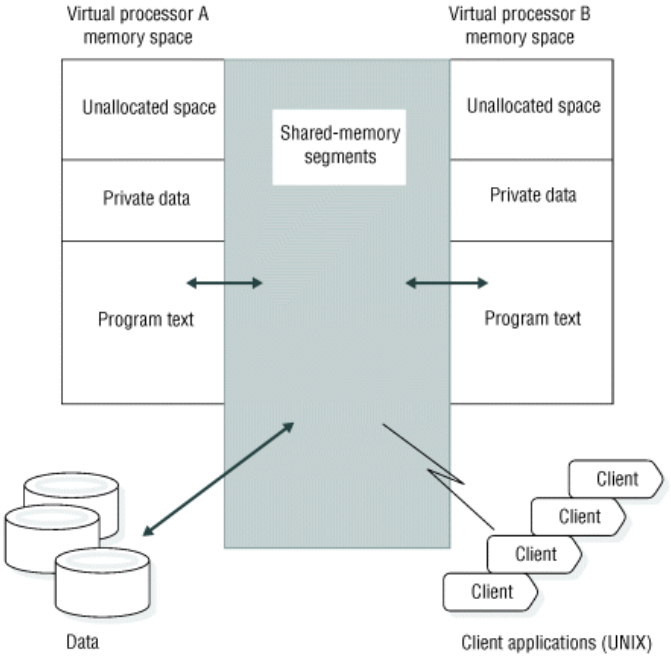

图 18: 数据库服务器如何使用共享内存

共享内存分配

数据库服务器在共享内存里创建不同部分来处理不同进程。

数据库服务器将创建共享内存的以下部分 ：

- 常驻 部分
- 缓冲池 部分
- 虚拟 部分
- IPC 通讯 或消息部分

  如果 sqlhosts 文件指定了共享内存通信，那么数据库服务器为该信部分分配内存。
- 虚拟扩展 部分

数据库服务器根据需要将操作系统段添加到共享内存的虚拟和虚拟扩展部分。

有关平台的共享内存设置的更多信息，请参阅机器说明文件。下图显示了共享内存各部分的内容。

所有的数据库服务器虚拟处理器都可以访问相同的共享内存段。每个虚拟处理器通过保留其自身对共享内存资源（如缓冲区、锁定和锁存器）的指针集来管理其工作。当数据库服务器从脱机模式转换为静默、管理或联机模式时，虚拟处理器将连接共享内存。数据库服务器使用锁定和锁存器来管理多个线程对共享内存资源的并发访问。

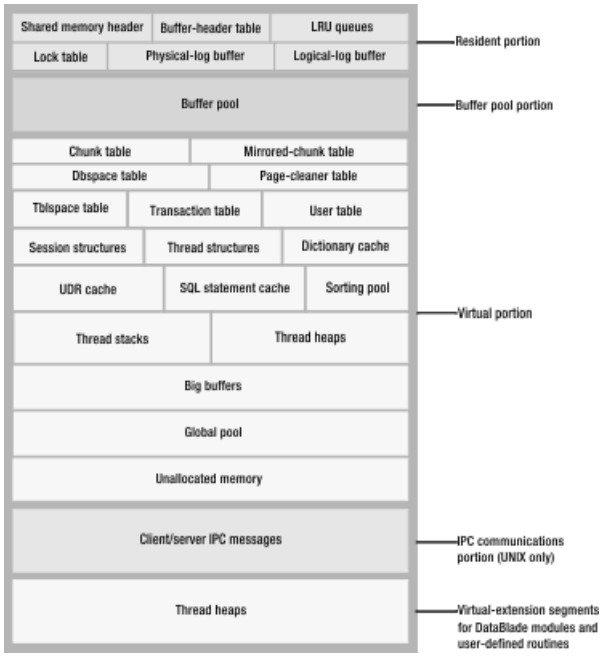

图 19: 数据库服务器共享内存的内容

共享内存大小

数据库服务器共享内存的每个部分是由一个或多个操作系统内存段组成，每个内存段分成一系列大小为 4 KB 的块，并且由位图管理。

onstat 实用程序输出的标题行包含数据库服务器共享内存的大小（以 KB 表示）。还可以使用 onstat -g seg 来监视数据库服务器为共享内存的每一部分分配多少内存。有关如何使用 onstat 的信息，请参阅《*SinoDB*® 管理员参考》。

可以在 onconfig 文件中设置 SHMTOTAL 参数来限制数据库服务器可以在计算机或节点上安排的内存开销量。SHMTOTAL 参数指定数据库服务器可用于所有内存分配的共享内存的总量。但是，如果数据库服务器需要的内存量大于 SHMTOTAL 中设置的量，那么某些操作可能会失败。如果发生这种情况，数据库服务器将在消息日志中显示以下消息 ：

```
size of resident + virtual segments x + y > z
   total allowed by configuration parameter SHMTOTAL
```

此外，数据库服务器会将错误消息返回到启动违规操作的应用程序。例如，如果数据库服务器在尝试执行操作（如索引构建或散列连接）时需要的内存量大于在 SHMTOTAL 中指定的内存量，那么该服务器将向应用程序返回类似于以下消息之一的错误消息：

```
-567    Cannot write sorted rows.
-116    ISAM error: cannot allocate memory.
```

数据库服务器在发送这些消息后会回滚任何由违规查询执行的部分结果。

内部的操作（如页面清除程序或检查点活动）也会导致数据库服务器超过 SHMTOTAL 上限。 当这种情况发生时，数据库服务器会向消息日志发送消息。例如，假设数据库服务器尝试为页面清除程序活动分配其他内存但失败了。 因此，数据库服务器会向消息日志发送类似于以下消息的信息 ：

```
17:19:13          Assert Failed: WARNING! No memory available for page cleaners
17:19:13          Who: Thread(11, flush_sub(0), 9a8444, 1)
17:19:13          Results: Database server may be unable to complete a checkpoint
17:19:13          Action: Make more virtual memory available to database server
17:19:13          See Also: /tmp/af.c4
```

After the database server informs you about the failure to allocate additional memory, it rolls back the transactions that caused it to exceed the SHMTOTAL limit. Immediately after the

rollback, operations no longer fail from lack of memory, and the database server continues to process transactions as usual.

数据库服务器通知您其他内存分配失败后，将回滚导致其超过 SHMTOTAL 限制的事务。一经回滚后，操作不再因内存不足而失败，数据库服务器将继续如常处理事务。

超过 SHMTOTAL 时要执行的操作

当数据库服务器需要的内存量大于 SHMTOTAL 所允许的值时，将出现瞬时状态，这可能是因突发超过正常处理负载的活动所引起的。

只有导致数据库服务器耗尽内存的操作会暂时失败。其他操作仍将以正常方式继续处理。

如果有消息定期指示数据库服务器需要的内存量大于 SHMTOTAL 所允许的值，表示您没有正确配置数据库服务器。降低 BUFFERPOOL 配置参数中的 DS_TOTAL_MEMORY 值或 buffers 值为一种可能的解决方案；增加 SHMTOTAL 的值是另一种解决方案。

连接共享内存的进程

一些进程连接到数据库服务器共享内存。

以下进程连接到数据库服务器共享内存：

- 通过共享内存通讯部分（ipcshm）与数据库服务器通信的客户端应用程序进程。
- 数据库服务器虚拟处理器
- 数据库服务器实用程序

以下主题描述每种类型的进程如何连接到数据库服务器共享内存。

客户端如何连接到通讯部份 (UNIX™)

通过共享内存 (nettype ipcshm) 与数据库服务器通信的客户端应用程序进程透明地连接到共享内存的通讯部份。自动编译到应用程序的系统库函数使其能够连接到共享内存的通讯部份。

有关共享内存连接的信息，请参阅 客户端/服务器通信 在第33页与网络虚拟处理器 在第96页。

如果没有设置 INFORMIXSHMBASE 环境变量，那么客户端应用程序将连接到特定于平台的地址的通讯部份。如果客户端应用程序连接到其他共享内存段（非数据库服务器共享内存），那么用户可将 INFORMIXSHMBASE 环境变量设置为其所连接数据库服务器共享内存通信段的所在地址。当您指定共享内存通信段的地址时，可以防止数据库服务器与应用程序所使用的其他共享内存段冲突。有关如何设置 INFORMIXSHMBASE 环境变量的信息，请参阅 《*SinoDB® SQL* 指南*:* 参考》。

相关链接

共享内存连接 *(UNIX)* 在第37页
网络连接的环境变量 在第65页

实用程序如何连接共享内存

数据库服务器实用程序如 onstat、onmode 与 ontape 可连接共享内存。

实用程序 onstat、onmode 与 ontape 可通过以下文件之一来连接共享内存。

操作系统	文件
UNIX™	$INFORMIXDIR/etc/.infos.*servername*
Windows™	$INFORMIXDIR/etc/.infos.*servername*

变量 servername 是 onconfig 文件中 DBSERVERNAME 参数的值。实用程序从 INFORMIXSERVER 环境变量中获取文件名的 servername 的部分。

oninit 进程在启动数据库服务器时读取 onconfig 文件并创建 .infos.servername 文件。当数据库服务器终止时，将删除该文件。

虚拟处理器如何连接共享内存

数据库服务器虚拟处理器在安装期间连接共享内存。

在此过程中，数据库服务器必须满足以下两点要求:

- 确保所有虚拟处理器能够定位并访问相同的共享内存段。
- 确保共享内存段的物理内存位置与指定给同一台计算机上其他数据库服务器实例的共享内存段的物理内存位置不同。

数据库服务器使用 SERVERNUM 与 SHMBASE 两个配置参数来满足这些要求。

当虚拟处理器连接共享内存时，会执行以下主要步骤：

- 从 onconfig 文件中访问 SERVERNUM 参数
- 使用 SERVERNUM 计算共享内存键值
- 使用共享内存键值请求一个共享内存段

  操作系统返回第一个共享内存段的共享内存标识符。
- 引导操作系统将第一个共享内存段连接到其位于 SHMBASE 的进程空间。
- 如需要，连接其他共享内存段以接续第一段

以下主题描述数据库服务器在连接共享内存段的过程中如何使用 SERVERNUM 与 SHMBASE 配置参数的值。

获取共享内存段的键值

SERVERNUM 配置参数的值与 *shmkey* 的值（内部计算的数字）决定了每个共享内存段的唯一键值。

要查看共享内存段的键值，请运行 onstat -g seg 命令。有关更多信息，请参阅《*SinoDB*® 性能指南》中有关 SHMADD 和缓冲池的章节。

当虚拟处理器请求操作系统连接第一个共享内存段时，它将提供唯一键值来标识该段，而操作系统将传回与键值相关联的共享内存段标识符作为回应。使用此标识符，虚拟处理器将请求操作系统将共享内存段连接至虚拟处理器地址空间。

指定连接第一个共享内存段的位置

onconfig 文件中的SHMBASE 参数指定每个虚拟处理器连接第一个（或基本）共享内存段所在的虚拟地址。

每个虚拟处理器在同一虚拟地址连接第一个共享内存段。此种情况使得同一数据库服务器实例中的所有虚拟处理器能够引用共享内存中的相同位置而无需计算共享内存地址。数据库服务器实例的所有共享内存地址都是相对于 SHMBASE 的。

> 警告: 不可更改 SHMBASE 的值。

SHMBASE 的值易受以下因素影响：

- SHMBASE 的特定值取决于平台以及处理器是 32 位还是 64 位的。SHMBASE 的值不是任意数，它将在虚拟处理器动态获取其他内存空间时保证共享内存段的安全。
- 不同操作系统在不同的虚拟地址上容纳附加的内存。有些架构扩展虚拟处理器数据段的最高虚拟地址以便容纳下一段。在此情况下，数据段可能会发展成共享内存段。
- 某些 UNIX™ 版本要求用户将虚拟地址的 SHMBASE 参数设置为零。零地址会通知 UNIX™ 内核挑选连接共享内存段所在的最佳地址。但并非所有 UNIX™ 架构都支持此选项。而且，在一些系统中，内核做出的选择可能不是最佳选择。

有关 SHMBASE 的信息，请参阅 SinoDB® 的机器说明文件。

连接附加共享内存段

要连接附加共享内存段时，虚拟处理器会向操作系统发出请求，如同其请求第一个段一样。

每个虚拟处理器必须连接到数据库服务器获取的总共享内存量。在虚拟处理器连接每个共享内存段后，将会计算已连接的共享内存的数量以及剩余的数量。数据库服务器通过将共享内存头写入第一个共享内存段中来促进此过程。从共享内存头的16个字节中，虚拟处理器可获取以下数据：

- 此数据库服务器共享内存的总量
- 每个共享内存段的大小

对于附加内存段，虚拟处理器将 *shmkey* 的先前值上加1。虚拟处理器将导引操作系统连接位于从以下计算得到的地址的段：

```
SHMBASE + (段大小 x 已连接的段数)
```

虚拟处理器重复此过程直至它获得共享内存的总量。

假设初始键值为 (SERVERNUM * 65536) + *shmkey* ，那么在数据库服务器可以请求由同一计算机上另一数据库服务器实例使用的共享内存键值前，它可以请求最多 65,536 个共享内存段。

定义共享内存下边界地址
如果操作系统使用参数来定义共享内存的下边界地址，并且参数未正确设置，那么会使共享内存段无法连续地连接。

下图说明了该问题。如果下边界地址小于上一段的结束地址加上当前段的大小，那么操作系统会将当前段连接到上一段结束位置之后的某个点上。此操作将在两段之间创建间隔。由于共享内存必须连接到虚拟处理器使其看起来像是连续内存一样，所以此间隔将产生问题。当此情况出现时，数据库服务器将会接收到错误。

要更正该问题，请检查设置下边界地址的操作系统内核参数，或重新配置内核以允许较大的共享内存段。

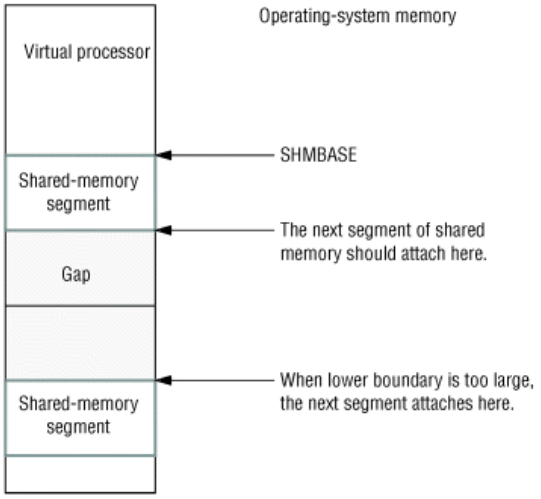

图 20: 共享内存下边界地址概述

共享内存常驻部分

操作系统在系统上运行的进程间切换时，通常将部分内存的内容交换到磁盘。然而当某一部分内存指定为常驻时，它不会交换到磁盘。将经常访问的数据驻留在内存中可以提高性能，因为这样可以减少了访问该数据时所需的磁盘 I/O 操作的数目。

当以下两种条件存在时，数据库服务器将请求操作系统把虚拟部分保持在物理内存中：

- 操作系统支持共享内存驻留。
- onconfig 文件中的 RESIDENT 参数设置为 -1 或大于 0 的值。

  警告: 当考虑是否将 RESIDENT 参数设置为 -1 时，应考虑所有应用程序对共享内存的使用。为使用SinoDB® 而锁定所有共享内存会对同一台计算机上的其他应用程序（如果有）的性能产生不利影响。

数据库服务器共享内存的常驻部分存储了以下数据结构，这些结构的大小在数据库服务器运行时不会变更：

- 共享内存头
- 逻辑日志缓冲区
- 物理日志缓冲区
- 锁表

相关链接

《*SinoDB* 管理员参考》: *RESIDENT* 配置参数

共享内存头

共享内存头包含共享内存中所有其他结构的描述，包括内部表、缓冲池，以及指向这些结构的指针。

当虚拟处理器首次连接共享内存时，它会从共享内存头中读取地址信息以便引导到所有其他结构。

共享内存头的大小大约有 200 KB 左右,但因不同计算机平台而异。您无法调整该头的大小。

逻辑日志缓冲区

数据库服务器使用逻辑日志来存储自上次数据库空间备份以来数据库服务器数据的变更记录。逻辑日志存储代表数据库服务器工作的逻辑单元的记录。

逻辑日志包含以下五种类型日志记录（此外还有许多其他类型）：

- 所有数据库的 SQL 数据定义语句
- 所有使用日志记录创建的数据库的 SQL 数据操作语句
- 数据库日志状态的更改记录
- 检查点记录
- 配置更改记录

The database server uses only one of the logical-log buffers at a time. This buffer is the current logical-log buffer. Before the database server flushes the current logical-log buffer to disk, it makes the second logical-log buffer the current one so that it can continue writing while the first buffer is flushed. If the second logical-log buffer fills before the first one finishes flushing, the third logical-log buffer becomes the current one. This process is illustrated in the following figure.数据库服务器一次只使用一个逻辑日志缓冲区。此缓冲区为当前逻辑日志的缓冲区。在数据库服务器把当前的逻辑日志缓冲区清空到磁盘之前，它将使第二逻辑日志缓冲区成为当前的逻辑日志缓冲区以便其可以在清空第一缓冲区时继续写入。如果在第一缓冲区完成清空之前，第二个逻辑日志缓冲区已满，那么第三逻辑日志缓冲区将成为当前的逻辑日志缓冲区。下图说明了此过程。

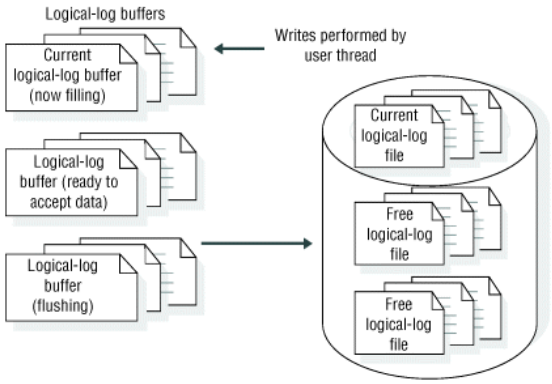

图 21: 逻辑日志缓冲区及其与磁盘上逻辑日志文件的关系

有关数据库服务器如何清空逻辑日志缓冲区的说明，请参阅 清空逻辑日志缓冲区 在第125页。

LOGBUFF 配置参数指定逻辑日志缓冲区的大小。如果存储比缓冲区大的记录（例如，数据库空间中的 TEXT 或 BYTE 数据），那么小缓冲区可能会产生问题。逻辑日志缓冲区大小的建议值为 64 KB。当设置小于建议值时，数据库服务器会在启动时给一个建议值。有关可指定给该配置参数的可能的值，请参阅 《*SinoDB*® 管理员参考》。

有关 TEXT 与 BYTE 数据对共享内存缓冲区的影响的信息，请参阅 缓冲区大对象数据 在第126页。

物理日志缓冲区

数据库服务器使用物理日志缓冲区来容纳一些已修改的数据库空间页的前映像。

物理日志中的前映像以及逻辑日志中的记录使得数据库服务器在系统故障后可以恢复数据库的一致性。

物理日志缓冲区实质上是两个缓冲区。双缓冲允许数据库服务器进程在其他缓冲区正被清空到磁盘上的物理日志时可以写入活动的物理日志缓冲区。有关数据库服务器如何清空物理日志缓冲区的描述，请参阅 清空物理日志缓冲区 在第124页。有关监视物理日志文件的信息，请参阅 监视物理和逻辑日志记录活动 在第270页。

onconfig 文件中的 PHYSBUFF 参数指定物理日志缓冲区的大小。一次物理日志缓冲区的写入正好是一页。如果物理日志缓冲区的指定大小不能够按页的大小均匀分隔，数据库服务器会向下降到能按页的大小平均分隔的最近的值。虽然有些操作要求缓冲区立即清空，但通常数据库服务器是在缓冲区满时将缓冲区清空到磁盘上的物理日志文件。因此，缓冲区的大小决定了数据库服务器将缓冲区清空到磁盘的频率。

物理日志缓冲区大小的缺省值为 512 KB。如果要使用较小的值，数据库服务器将显示一条消息，指示可能无法达到最佳性能。使用小于 512 KB 的物理日志缓冲区仅影响性能，不会影响事务完整性。

有关此配置参数的更多信息，请参阅 《*SinoDB*® 管理员参考》。

高可用性数据复制缓冲区

数据复制需要两个数据库服务器实例（一个主实例和一个辅助实例）分别运行在两台计算机上。

如果为数据库服务器实现数据复制，那么主数据库服务器会先将逻辑日志记录保留在数据复制缓冲区中，再将其发送给辅助数据库服务器。数据复制缓冲区大小总是与逻辑日志缓冲区的大小一样。有关逻辑日志缓冲区的大小的信息，请参阅，逻辑日志缓冲区 在第111页。有关如何使用数据复制缓冲区的更多信息，请参阅数据复制的工作原理 在第311页。

琐表

锁是在用户线程在琐表中写入条目时创建。单个事务可以拥有多个锁。琐表是可用琐的池。

有关锁定和与锁定相关联的 SQL语句说明，请参阅《*SinoDB*® *SQL* 指南: 教程》。

存储在琐表中的以下信息描述锁：

- 拥有锁的事务的地址
- 琐的类型（互斥、更新、共享、字节或意向）
- 已锁定的页或行标识
- 放置锁的表空间
- 关于已锁定字节（智能大对象的字节范围琐）的信息：
  - 智能大对象标识
  - 智能大对象中已锁定字节开始处的偏移量
  - 已锁定的字节数（从偏移处开始）

要指定锁表的初始大小，请设置 LOCKS 配置参数。有关使用 LOCKS 配置参数来指定会话的锁数的信息，请参阅《*SinoDB*® 管理员参考》中有关配置参数的主题，及《*SinoDB*® 性能指南》中有关配置对内存利用率的影响的主题。

如果会话分配的琐数超过 LOCKS 配置参数中指定的值，那么数据库服务器会将琐表的大小加倍，最多可达 15 倍。数据库服务器通过尝试在每次增加时使锁表翻倍来增加锁表的大小。然而，每次增加的量不能超过最大值。对于 32 位平台，每次最多可以增加 100,000 个琐。因此，32 位平台允许的最大锁总数为 8,000,000（最大启动锁数）+ 99（最大动态琐表扩展数）x 100,000（每个琐表扩展可增加的最大锁数）。对于 64 位平台，每次最多可增加 1,000,000 个锁。因此，允许的最大琐总数为 500,000,000（最大启动锁数）+ 99（最大动态琐表扩展数）x 100,000（每个琐表可增加的最大锁数）。

使用 DEF_TABLE_LOCKMODE 配置参数为新表的页或行设置锁定模式。

锁能够使会话在并发事务提交或回滚后才读取数据。对于使用事务日志记录创建的数据库，可使用 onconfig 文件中的 USELASTCOMMITTED 配置参数来指定数据库服务器是否使用上次提交的数据版本。上次提交的数据版本是任何更新发生之前的数据版本。USELASTCOMMITTED 配置参数设置的值会覆盖在 SQL 语句

SET ISOLATION TO COMMITTED READ 中指定的隔离级别。有关使用 USELASTCOMMITTED 配置参数的更多信息，请参阅《*SinoDB*® 管理员参考》中有关配置参数的主题。

有关使用和监视锁定的更多信息，请参阅《*SinoDB*® 性能指南》和《*SinoDB*® *SQL* 指南: 教程》中有关锁定的主题。

共享内存的缓冲池部分

共享内存的缓冲池部分包含缓冲区用于存储从磁盘读取的数据库空间页面。

下图说明了共享内存头和缓冲池。

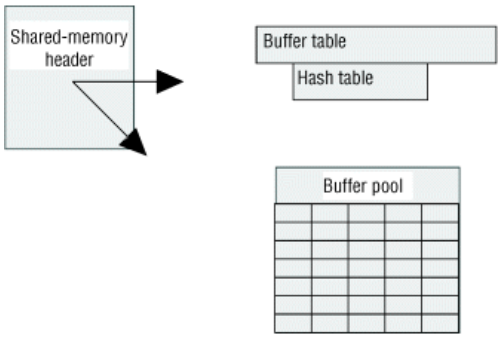

图 22: 共享内存缓冲池

使用 BUFFERPOOL 配置参数来指定缓冲池的信息，包括缓冲池中缓冲区的数目或缓冲池的总体大小。每个缓冲区是一个数据库服务器页面的大小。太少的缓冲区可能严重影响性能。可设置 BUFFERPOOL 配置参数使数据库服务器能够按需自动增加缓冲区的数目以提高性能。否则，必须监视数据库服务器，并调整缓冲区数以确定可接受的值。

一个缓冲池管理一种大小的页。对于数据库服务器中的存储空间使用的每个页面大小，都需要一个不同的缓冲池。数据库服务器会自动创建所需的缓冲池。例如，如果第一个创建的数据库空间的页大小为 6 KB，那么数据库服务器将会创建一个缓冲池来将默认为 6 KB 的页面缓存到内存中。可使用 BUFFERPOOL 配置参数来控制缓冲池的属性。

如果数据库服务器是处于联机、静默、或管理模式，您还可以使用 onparams -b 命令来添加不同大小的缓冲池。使用 onparams -b 命令时所指定的信息将自动传输到 onconfig 文件成为 BUFFERPOOL 配置参数的新条目。

一般情况下，数据库服务器以整页单元（一个缓冲区的大小）执行 I/O 操作。例外情况是从大缓冲区、从 BLOB 空间缓冲区，或从轻量级的 I/O 缓冲区执行 I/O 操作。

自动 LRU（least recently used）调整会影响所有缓冲池，并调整通过 BUFFERPOOL 配置参数显示设置的 lru_min_dirty 和 lru_max_dirty 的值。

通过缓冲表来跟踪缓冲区的状态。在共享内存中，缓冲区被组织成 FIFO/LRU 缓冲队列。缓冲区获取是由互斥锁和锁访问的信息来管理。

onstat -b 命令显示有关缓冲区的信息。

相关链接

线程数据 在第117页
互斥 在第86页
创建 *BLOB* 页缓冲区 在第126页
《*SinoDB* 管理员参考》: *BUFFERPOOL* 配置参数
《*SinoDB* 管理员参考》: *onstat -b* 命令: 显示正在使用的缓冲区的信息
《*SinoDB* 管理员参考》: *onparams -b*: 添加缓冲池
《*SinoDB* 性能指南》: *BUFFERPOOL* 配置参数和内存利用率

共享内存的虚拟部分

共享内存的虚拟区可以由数据库服务器扩展，并可以由操作系统调出磁盘。

在数据库服务器运行时，它会自动按需将其他操作系统段连接到虚拟区域。

管理共享内存的虚拟部分

数据库服务器使用内存池来追踪类型和大小相似的内存分配。

在池中保留相关的内存分配将有助于减少内存分段。它也能使数据库服务器能够一次释放大块内存空间，而不是释放构成内存池的每块。

所有会话都有一个或多个内存池。当数据库服务器需要内存时，它会先查看特定的池。如果池中内存不足以满足需求，数据库服务器将会从系统池中添加内存。如果数据库服务器在系统池中也无法找到足够的内存，它将会动态分配更多段到虚拟部分。

数据库服务器从通过链结表跟踪可用空间的池分配其每个子系统（会话池、堆栈、堆、控制块、系统目录、SPL 例程高速缓存、SQL 语句高速缓存、排序池和消息缓冲区）的虚拟共享内存。当数据库服务器分配内存的部分时，它首先在池的可用列表中搜索足够大小的分段。如果找不到，它会将新的块从虚拟部分带入池中。当释放内存时，被释放的内存将作为可用分段返回到池中，并留在其中直到该池被删除。例如，当数据库服务器启动了客户端应用程序的会话时，它为该会话池分配内存；当会话终止时，数据库服务器会将分配的内存作为可用分段返回。

共享内存虚拟部分的大小
使用配置参数来指定共享内存虚拟部分的初始大小、之后要添加的段的大小，以及可用于 PDQ 查询的内存量。

要指定虚拟共享内存部分的初始大小，请设置 SHMVIRTSIZE 配置参数。要指定之后添加到虚拟共享内存的段的大小，请设置 SHMADD 和 EXTSHMADD 配置参数。

要指定 PDQ 查询可用的内存量，请设置 DS_TOTAL_MEMORY 参数。

若想增加非 PDQ 查询可用的内存量，并且 PDQ 优先级设置为 0（零），可以使用以下任一方式来更改该内存量：

- 设置 DS_NONPDQ_QUERY_MEM 配置参数
- 运行 onmode -wm 或 onmode -wf 命令。

例如，如果使用 onmode 实用程序，请按以下示例所示指定值：

```
onmode -wf DS_NONPDQ_QUERY_MEM=500
```

DS_NONPDQ_QUERY_MEM 最小值为128 KB。所支持的最大值为 DS_TOTAL_MEMORY 值的 25%。

相关链接

将段添加到共享内存的虚拟部分 在第132页
《*SinoDB* 管理员参考》: *DS_TOTAL_MEMORY* 配置参数
《*SinoDB* 管理员参考》: *DS_NONPDQ_QUERY_MEM* 配置参数
《*SinoDB* 管理员参考》: *SHMVIRTSIZE* 配置参数
《*SinoDB* 管理员参考》: *SHMADD* 配置参数
《*SinoDB* 管理员参考》: *EXTSHMADD* 配置参数

共享内存虚拟部分的组件

共享内存的虚拟部分储存各类型数据

共享内存的虚拟部分存储以下数据：

- 内部表
- 大缓冲区

- 会话数据
- 线程数据（堆栈和堆）
- 数据分布高速缓存
- 字典高速缓存
- SPL 例程高速缓存
- SQL 语句高速缓存
- 排序池
- 全局池

共享内存内部表
数据库服务器共享内存包括七种内部表用于追踪分享内存资源。

共享内存内部表如下：

- 缓冲表
- 块表
- 数据空间表
- 页清除程序表
- 表空间表
- 事务表
- 用户表

缓冲表
缓冲表追踪共享内存池中单个缓冲区的地址和状态。

当使用缓冲区时，该缓冲区将包含来自磁盘的数据或索引页的映像。有关磁盘页的目的和内容的更多信息，请参阅页 在第141页。

缓冲表中的每个缓冲区包含了以下缓冲区管理所必须的控制信息 ：

缓冲区状态

缓冲区状态分为空的、未修改的或已修改的。未修改的缓冲区包含可以被覆盖的数据。已修改的（脏）缓冲区包含必须先写入磁盘才能够被覆盖的数据。

当前锁定访问级别

缓冲区根据用户线程正在执行的操作类型接收锁定访问级别。数据库服务器支持两种缓冲区锁定访问级别 ： 共享和互斥。

等待缓冲区的线程

每个缓冲区头都保留一系列正在等待缓冲区的线程以及每个等待的线程所需要的锁定访问级别。

每个数据库服务器缓冲区在缓冲表中都有一个条目。

有关数据库服务器缓冲区的信息，请参阅共享内存常驻部分 在第110页。有关如何监视缓冲区的信息，请参阅监视缓冲区 在第135页。

数据库服务器基于已分配的缓冲区数确定缓冲区表散列表中的条目数。最大散列值是小于 buffers 值的2的最大幂数，它是在某个 BUFFERPOOL 配置参数字段中指定。

块表
块表追踪数据库服务器中的所有块。

如果启用了镜像，那么会在启动共享内存时创建相应的镜像块表。镜像块表追踪所有镜像块。

共享内存中的块表包含了能够使数据库服务器在磁盘上找到块的信息。此信息包括数据库空间中初始块的编号和下一个块的编号。标志描述了块状态：镜像块或主块；脱机、 联机或恢复模式；以及此块是否是 BLOB 空间的一部分。有关监视块的信息，请参阅监视块 在第199页。

块表的最大条目数可能受操作系统所允许每个进程的最大文件描述符数的限制。通常可使用操作系统内核配置参数来指定每个进程的文件描述符数。有关详细信息，请查询操作系统手册。

数据库空间表
数据库空间表跟踪数据库服务器中的存储空间。

数据库空间表信息包含有关每个数据库空间的以下信息 ：

- 数据库空间编号
- 数据库空间名称和所有者
- 数据库空间镜像状态 （已镜像或未镜像）
- 数据库空间的创建日期和时间

如果存储空间是 BLOB 空间，那么标志将指示该 BLOB 空间所在的介质：磁性的介质或可移动的介质。如果存储空间是智能大对象空间，它将包含用于跟踪智能大对象的元数据以及包含用户数据的大型连续的页面块的内部表。

有关监视数据库空间的信息，请参阅监视磁盘使用 在第199页。

页清除程序表
页清除程序表跟踪每个页清除程序线程的状态和位置。

页清除程序线程数由 onconfig 文件中的 CLEANERS 配置参数来指定。有关应指定多少页清除程序线程的建议，请参阅《*SinoDB*® 管理员参考》中有关配置参数的章节。

无论 onconfig 文件中 CLEANERS 配置参数指定的页清除程序线程数是多少，页清除程序表始终包含 128 个条目。

有关监视页清除程序线程活动的信息，请参阅《*SinoDB*® 管理员参考》中有关 onstat -F 选项的信息。

表空间表
表空间表跟踪数据库服务器实例中的所有活动的表空间。

活动的表空间就是当前数据库会话所使用的表空间。每一个活动的表空间占表空间表中的一个条目。活动的表空间包括数据库表、临时表，和内部控制表（如系统目录表）。每一表空间表条目包含表空间的头信息、表空间名以及指向磁盘上数据库空间中表空间 tblspace 的指针。（共享内存的活动的表空间表与表空间 tblspace 不同。）有关监视表空间的信息，请参阅监视表空间和扩展数据块 在第202页。

数据库服务器为每个数据库空间管理一个表空间表。

事务表
事务表跟踪数据库服务器中的所有事务。

来自于事务表的跟踪信息可通过 onstat -x 显示。有关onstat -x 的输出示例，请参阅《*SinoDB*® 性能指南》中的监视事务。

数据库服务器根据当前事务数自动增加事务表中的条目数，最多可达 32,767 个条目。

有关事务和事务中使用的 SQL 语句的更多信息，请参阅《*SinoDB*® *SQL* 指南*:* 教程》、《*SinoDB*® *SQL* 指南*:* 参考》和《*SinoDB*® *SQL* 指南*:* 语法》。

仅 UNIX: 事务表同样明确支持 X/Open 环境。支持 X/Open 环境需要TP/XA。

用户表
用户表追踪所有用户线程和系统线程。

每个客户端会话根据指定的并行级别有一个主线程以及零到多个辅助线程。系统线程包括一个监视和控制检查点的线程、一个处理 onmode 命令的线程、B型树扫描线程，和页清除程序线程。

数据库服务器会按需增加用户表中的条目数。可使用 onstat -u 命令来监视用户线程。

大缓冲区
大缓冲区是由多个页组成的单个缓冲区。实际页数将取决于平台。

数据库服务器分配大缓冲区来提高大型读写操作的性能。无论何时数据库服务器向磁盘写入多个物理上相邻的页时，它都使用大缓冲区。例如，数据库服务器尝试使用大缓冲区来执行一系列顺序读（轻度扫描）或将存储在数据库空间的简单大对象读入共享内存。

用户对大缓冲区没有控制权。如果数据库服务器使用轻度扫描，那么它将从共享内存中分配大缓冲区。

有关使用 onstat 命令监视大缓冲区的信息，请参阅 《*SinoDB*® 性能指南》中有关配置对 I/O 活动的影响的主题。

会话数据
当客户端应用程序请求连接到数据库服务器时，数据库服务器将与客户端开始会话 ，并在共享内存中创建该会话的数据结构。该创建的数据结构称为 会话控制块。

会话控制块存储会话标志、用户标志、客户端进程标志、主机名称，以及各种状态标志。

数据库服务器按需为会话数据分配内存。

可通过配置参数 SESSION_LIMIT_MEMORY 来指定单个会话可以分配的最大内存量以限制分配给会话的内存。对于有管理员权限的用户，例如 informix 用户和 DBSA 用户，并不受此限制。

相关链接

《*SinoDB* 管理员参考》: *SESSION_LIMIT_MEMORY* 配置参数

onstat -g ses 命令: 显示会话相关信息

线程数据
当客户端连接到数据库服务器时，数据库服务器除了启动会话外，还会启动一个主会话线程，并在共享内存中为其创建 线程控制块 。

数据库服务器还代表其自身启动内部线程，并为这些线程创建线程控制块。当数据库服务器从运行一个线程切换为运行另一个线程（上下文切换）时，它在线程控制块中保存有关线程的信息（例如寄存器内容，程序计数器（下一步指令的地址）以及全局指针）。有关线程控制块以及其使用方法的更多信息，请参阅 上下文切换 在第84页。

数据库服务器按需为线程控制块分配内存。

相关链接

共享内存的缓冲池部分 在第113页

堆栈
数据库服务器中的每个线程在共享内存的虚拟部分都有其自己的堆栈区域。

有关线程如何使用堆栈的描述，请参阅 堆栈 在第85页。有关如何监视会话的堆栈的大小，请参阅《*SinoDB*® 性能指南》中有关监视会话和线程的章节。

用户线程的堆栈空间大小是由 onconfig 文件中的 STACKSIZE 参数指定。可以根据需要，通过改变 STACKSIZE 的值来改变所有用户线程的堆栈的大小。

可以使用环境变量 INFORMIXSTACKSIZE 的设置值来覆盖服务器配置文件中的 STACKSIZE 的设置值。在环境中设置 INFORMIXSTACKSIZE 并回收实例。

You can use the INFORMIXSTACKSIZE environment variable to override the STACKSIZE value in the server configuration file. Set INFORMIXSTACKSIZE in the environment and recycle the instance.

相关链接

《*SinoDB* 管理员参考》: *STACKSIZE* 配置参数

堆
每个线程都有一个堆来容纳其运行时创建的数据结构。

堆在线程创建时动态分配。线程堆的大小是不可配置的。

数据分布高速缓存

数据库服务器使用由 UPDATE STATISTICS 语句在 MEDIUM 或 HIGH 模式中生成的分布统计信息来确定最低成本的查询计划。

当数据库服务器第一次访问有关特定列的分布统计信息时，它将从磁盘上的 sysdistrib 系统目录表中读取分布统计信息，并将统计信息存储在数据分布高速缓存中。然后可以读取这些统计信息来优化后续访问该列的查询。

如果这些统计信息有效地存储在数据分布高速缓存中，并有效地从数据分布高速缓存中访问这些统计信息，那么将会提高性能。可使用 DS_HASHSIZE 和 DS_POOLSIZE 配置参数来配置数据分布高速缓存的大小。有关更改数据分布高速缓存的缺省大小的信息，请参阅 《*SinoDB*® 性能指南》中有关查询和查询优化器的主题。

字典高速缓存
当会话执行需要访问系统目录表的 SQL 语句时，数据库服务器会从系统目录表中读取数据。

数据库服务器将每个查询的表的目录数据存储在该表的后续查询中可以更有效地访问的结构中。这些结构将创建在共享内存的虚拟部分中以供所有会话使用。这些结构构成了字典高速缓存。

可使用 DD_HASHSIZE 和 DD_HASHMAX 配置参数来配置字典高速缓存的大小。有关这些参数的更多信息，请参阅 《*SinoDB*® 性能指南》中有关配置对内存的影响的章节。

SQL 语句高速缓存
SQL 语句高速缓存减少了查询的内存使用量和准备时间。

数据库服务器使用 SQL 语句高速缓存来存储用户运行的已优化的 SQL语句。当用户运行存储在 SQL 语句高速缓存的语句时，数据库服务器不再需要对该语句进行优化，从而提高了性能。

有关更多信息，请参阅 设置 *SQL* 语句高速缓存参数 在第131页。有关这些参数如何影响 SQL 语句高速缓存性能的详细信息，请参阅 《*SinoDB*® 性能指南》 。

排序内存
数据库服务器为排序分配的虚拟共享内存量取决于要排序的行数、行的大小以及其它因素。

以下数据库操作使用大量的共享内存虚拟部分来排列数据：

- 涉及连接、分组、聚集和排序操作的决策支持查询
- 索引建立
- SQL 中的 UPDATE STATISTICS 语句

有关平行排序的信息，参阅 《*SinoDB*® 性能指南》。

SPL 例程和 UDR 高速缓存
数据库服务器将 SPL 例程转换为可执行格式并将其存储在 UDR 高速缓存中，可以供任何会话访问。

当会话第一次需要访问 SPL例程或其它用户定义例程时， 数据库服务器将从系统目录表中读取定义并将定义存储于 URD 高速缓存中。

可使用 PC_HASHSIZE 和 PC_POOLSIZE 配置参数来配置 UDR 高速缓存的大小。有关更改 UDR 高速缓存的缺省大小的信息，请参阅 《*SinoDB*® 性能指南》中有关查询和查询优化器的章节。

全局池
全局池存储数据库服务器全局的结构。

全局池包括消息队列，供网络通信的轮询线程存储来自客户端的消息。sqlexec 线程将从全局池中取出消息并处理它们。

有关更多信息，请参阅 《*SinoDB*® 性能指南》中有关网络缓冲池和共享内存虚拟部分的章节。

共享内存的通信部分 (UNIX™)

如果至少有一个连接配置为 IPC 共享内存连接，数据库服务器将为共享内存的 IPC 通信部分分配内存。数据库服务器将在您设置共享内存时执行此分配。

通信部分包含使用共享内存与数据库服务器通信的本地客户端应用程序的消息缓冲区。

共享内存通信部分的大小约为 12KB 乘上共享内存通信 (nettype ipcshm) 的期望连接数。如果 (nettype ipcshm) 不存在，那么期望的连接数将缺省为 50。有关客户端如何连接到共享内存的通信部分的信息，请参阅 客户端如何连接到通讯部份 *(UNIX)* 在第108页。

相关链接

共享内存连接 *(UNIX)* 在第37页

共享内存的虚拟扩展部份

共享内存的虚拟扩展部份包含附加的虚拟段和虚拟扩展段。

虚拟扩展段包含 DataBlade® 模块的线程堆和在用户定义的虚拟处理器中运行的用户定义例程的线程堆。

EXTSHMADD 配置参数用于设置虚拟扩展段的大小。SHMADD 和 SHMTOTAL 配置参数适用于共享内存的虚拟扩展部份，正如它们适用于共享内存的其它部分一样。

并发控制

运行在同一虚拟处理器上或运行在个别虚拟处理器上的数据库服务器线程将共享对共享内存中资源的访问权。

当线程写入共享内存时，它使用互斥和锁定的机制来防止其它线程同时写入相同区域。互斥将给予线程访问共享内存资源的权限；锁定将防止其它线程写入缓冲区直到放置锁的线程完成对缓冲区的操作并释放锁定为止。

共享内存互斥

数据库服务器使用 互斥 来协调尝试修改共享内存中数据的线程。每个可修改的共享内存资源都与互斥相关联。

线程可以在修改共享内存资源之前，必须先获得与该资源关联的互斥。线程获取该互斥之后，便可以更改该资源。更改完成后，线程将释放该互斥。

如果线程尝试获取的互斥正由另一线程所持有，那么该线程必须等待该互斥被释放。

例如，两个线程尝试访问块表中的同一个插槽，但是只有一个线程可以取得与块表关联的互斥。只有持有互斥的线程才可以将其条目写入块表。第二个线程必须等待该互斥被释放后才能获取它。

有关监视互斥（也称为锁存器）的信息，请参阅 监视共享内存概要文件和锁存器 在第134页。

共享内存缓冲区锁定

共享内存的主要好处在于数据库服务器线程可以共享对共享内存缓冲池中的磁盘数据页的访问权。数据库服务器在通过锁定数据缓冲区的策略获得增加的并发性的同时保持线程隔离。

缓冲区锁的类型
数据库服务器使用两种类型的锁来管理到共享内存缓冲区的访问权。

两种锁类型：

- 共享锁
- 互斥锁

每种锁定类型在执行过程中会强制施行所需的线程隔离级别。

共享锁
如果多个线程可以访问缓冲区以读取数据但没有一个想要修改数据，那么缓冲区处为共享模式，或具有共享锁。
互斥锁
如果线程要求对缓冲区的互斥访问，那么缓冲区处于互斥模式或具有互斥锁。

请求访问缓冲区的所有其他线程将放置到等待队列中。当正在执行的线程准备释放互斥锁时，它将唤醒等待队列中的下一个线程。

数据库服务器线程对共享缓冲区的访问

数据库服务器线程通过一系列队列访问共享缓冲区，并使用互斥和锁定来同步访问和保护数据。

FIFO/LRU 队列

缓冲区为高速缓存保存数据。数据库服务器使用近期最少使用 (LRU) 队列来替换高速缓存的数据。SinoDB®也支持先进先出 (FIFO) 队列。当设置 LRU 队列的数目时，实际上是在设置 FIFO/LRU 队列的数目。

使用 BUFFERPOOL 配置参数可指定有关缓冲池信息，包括数据库服务器共享内存设置时要创建的 LRU 队列的数量的信息，以及控制共享内存缓冲区清空到磁盘的频率的 lru_min_dirty 和 lru_max_dirty 的值的信息。

要提升事务吞吐量，请增加 lru_min_dirty 和 lru_max_dirty 的值。但是请勿更改 lru_min_dirty 和 lru_max_dirty 值之间的差距。如果启用了 AUTO_LRU_TUNING 配置参数，那么 lru_max_dirty 和 lru_min_dirty 字段的值将按需自动重置以提升性能。

相关链接

《SinoDB 管理员参考》: AUTO_LRU_TUNING 配置参数

《SinoDB 管理员参考》: BUFFERPOOL 配置参数

LRU 队列的组成部分

每个 LRU 队列由一对链表组成。

以下为链表对：

- FLRU (free least-recently used) 列表，追踪队列中可用或未修改的页面
- MLRU (modified least-recently used) 列表，追踪队列中已修改的页面

可用或未修改的页面列表称为队列对中的 FLRU 队列，已修改的页面列表称为 MLRU 队列。这两个不同列表使您无需在队列中搜索可用或未修改的页面。下图说明了 LRU 队列的结构。

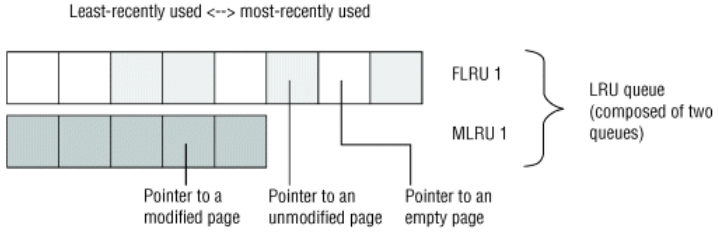

图 23: LRU 队列

按最近最少使用的顺序排序的页

当数据库服务器处理从磁盘读取页的请求时，它必须决定要替换内存中的哪一页。

数据库服务器不是随机选择一个页，而是假定最近引用的页比一段时间未引用的页更有可能在今后得到引用。因此，数据库服务器不会替换最近访问的页，而是替换将替换最近最少访问的页。通过维护按最近最少使用到最近最多使用顺序排序的页，数据库服务器可以容易地定位到内存中最近最少使用的页。

LRU 队列及缓冲池管理

在处理开始之前，所有页缓冲区都是空的，并且每个缓冲区由某个 FLRU 队列的一个条目代表。

缓冲区会均匀分布在各个 FLRU 队列中。要计算每个队列中的缓冲区的数目，请用缓冲区总数除以 LRU 队列的数目。缓冲区和 LRU 队列的数目是在 BUFFERPOOL 配置参数中指定的。

当用户线程需要获取缓冲区时，数据库服务器随机选择一个 FLRU 队列，并使用列表中最旧或最近最少使用的条目。如果可以锁存最近最少使用的页，该页将从队列中除去。

如果 FLRU 队列已锁定且无法锁存结尾页，数据库服务器将随机选择另一个 FLRU 队列。

如果用户线程正在共享内存中搜索特定的页，那么它将从存储在缓冲区表中的控制信息中获取该页的 LRU 队列的位置。

正在执行的线程在完成其工作后将释放缓冲区。如果该页已被修改，那么缓冲区将放置在 MLRU 队列的最近最常使用的末尾。如果读取的页未被修改，那么缓冲区将返回到 FLRU 队列的最近最多使用的末尾。有关如何监视 LRU 队列的信息，请参阅 监视缓冲区 在第135页.

要配置的 LRU 队列数
多个 LRU 队列可减少用户线程争用，并允许多个清除程序从队列中清空页，以确保脏页的百分比维持在可接受的水平。

通过在 BUFFERPOOL 配置参数中设置 lrus 的值来指定 LRU 队列数目。LRU 队列的缺省数取决于计算机上的 CPU 数 ：

- 如果是单处理器计算机，那么 lrus 的缺省值是 8。
- 如果是多处理器计算机，并启用了 MULTIPROCESSOR 配置参数，那么 lrus 的缺省值将大于8 或 CPU 虚拟处理器的数量。

在 BUFFERPOOL 配置参数指定了 lrus 的初始值后，可使用 onstat-R 命令来监视 LRU 队列。如果发现脏 LRU 队列的百分比一直超过 BUFFERPOOL 配置参数中 lru_max_dirty 的值，请增加 lrus 的值以添加更多的 LRU 队列。

例如，如果 lru_max_dirty 设置为 70 ，并且一直发现有 75% 的 LRU 队列是脏的，请增加 lrus 的值。如果增加 LRU 队列的数目，那么就会缩短队列长度，从而降低了页清除程序的工作。然而，您必须使用 CLEANERS 配置参数分配足够数量的页清除程序。

相关链接

《SinoDB 性能指南》: LRU 调整

《SinoDB 管理员参考》: BUFFERPOOL 配置参数

要分配的清除程序数
You must configure one cleaner for each disk that your applications update frequently. However, you must also consider the length of your LRU queues and frequency of checkpoints.必须为应用程序经常更新的每个磁盘都配置一个清除程序。不过，必须考虑 LRU 队列的长度及检查点的频率。

影响页清除程序数是否能跟上需要清除的页数的另一个因素是您是否配置了足够的页清除程序线程数。在某些队列中，因为没有页清除程序可用于清除队列，以致于脏页的百分比可能超过为 lru_max_dirty 指定的 BUFFERPOOL 值。 一段时间后，页清除程序可能落后太多难以追上，这时缓冲池将会比在 lru_max_dirty 中指定的百分比还要脏。

例如，假设 CLEANERS 参数设置为 8，并将 LRU 队列数从 8 增加到 12。您不能期待性能有多大的提升，因为 8 个清除程序必须分担额外 4 个队列的清除工作。如果您将 CLEANERS 数量增加到 12，那么每个清除程序可以更有效率地清除现已缩短的每个队列。

因为页清除程序必须在检查点期间将所有已修改的页清空到磁盘，将 CLEANERS 设置得太少将会导致检查点时性能变差。如果没有配置足够数量的页清除程序，那么检查点会花费更长的时间，导致整体的性能变差。

有关更多信息，请参阅清空缓冲池缓冲区 在第123页.

添加到 MLRU 队列的页数
页清除程序线程将 MLRU 队列中已修改的缓冲区清空到到磁盘。要指定清除开始的点，可使用 BOFFERPOOL 配置参数来指定 lru_max_dirty 的值以限制可附加到 MLRU 队列的页缓冲区数。

lru_max_dirty 的初始设置为 60.00，因此页清除会在队列所管理的缓冲区有 60% 的缓冲区被修改时开始。

实际上，在若干状况下页清除都会开始，其中之一是当 MLRU 队列接近 lru_max_dirty 的值。有关数据库服务器如何执行缓冲池清空的更多信息，请参阅 将数据清空到磁盘 在第123页。

以下示例显示 lru_max_dirty 的值如何应用于 LRU 队列，以指定何时开始页清除，从而限制 MLRU 队列中缓冲区数。

```
Buffers specified as 8000
lrus specified as 8
lru_max_dirty specified as 60 percent

Page cleaning begins when the number of buffers in the MLRU
   queue is equal to lru_max_dirty.

Buffers per lru queue = (8000/8) = 1000

Max buffers in MLRU queue and point at which page cleaning
   begins:1000 x 0.60 = 600
```

MLRU 清除结束
还可指定 MLRU 清除可以结束的点。

在 BUFFERPOOL 配置参数中的lru_min_dirty 的值指定 MLRU 队列中缓冲区的可接受的百分比。例如，如果 lru_min_dirty 设置为 50.00，那么当 LRU 队列中被修改的缓冲区的百分比为 50% 时，就不需要清除页了。实际上，页清除可根据页清除程序线程的指示继续清除超个此点。

以下示例显示 lru_min_dirty 的值如何应用于 LRU 队列，以指定 MLRU 队列中缓冲区的可接受百分比以及页清除结束的点。

```
Buffers specified as 8000
lrus specified as 8
lru_min_dirty specified as 50 percent

The acceptable number of buffers in the MLRU queue and
   the point at which page cleaning can end is equal
   to lru_min_dirty.

Buffers per LRU queue = (8000/8) = 1000

Acceptable number of buffers in MLRU queue and the point
   at which page cleaning can end:1000 x .50 = 500
```

lru_max_dirty 和 lru_min_dirty 的值可以使用小数位数。 例如，如果把 lru_max_dirty 设置为1.0333 并且把 lru_min_dirty 设置为 1.0，这将触发 LRU 在 3,100 脏缓冲区时开始写并在 3,000 脏缓冲区时停止。

有关数据库服务器如何清空缓冲池的更多信息，请参阅 将数据清空到磁盘 在第123页。

预读操作

数据库服务器会自动读取查询操作中正在处理的当前页的之后几页，除非禁用自动预读操作。预读使应用程序能夠运行得更快，因为它們將花費更少的時間等待磁盘 I/O。

当服务器检测到查询（包括 OLTP 查询和索引扫描）遇到 I/O 操作时，自动预读 会在连续扫描数据记录期间请求将页面放入缓冲池高速缓存以提高查询的性能

默认情况下，数据库服务器将根据查询遇到来自磁盘 I/O 的时间，自动确定何时发出预读请求以及何时停止：

- 如果查询遇到 I/O 操作，服务器将发出预读请求以提高查询的性能。由于预读请求通过对相对于 CPU 处理速度较慢的 I/O 处理速度的补偿大大地提高数据库处理的速度，因此使性能得到提升。
- 如果查询大部分已高速缓存，那么服务器将检测到没有发生任何 I/O 操作，也就不会执行预读。

使用 AUTO_READAHEAD 配置参数可更改查询的自动预读模式或禁用查询的自动预读。您可以：

- 通过执行 onmode -wm 或 onmode -wf 命令来动态更改 AUTO_READAHEAD 配置参数的值。
- 执行 SET ENVIRONMENT AUTO_READAHEAD 语句来更改模式，或启用/禁用会话的自动预读。

使用 onstat -p 命令查看数据库服务器读取和写入操作，并监视线程需要等待共享内存锁存器的次数。RA-pgsused 输出字段显示数据库服务器预读使用的页数和监视数据库服务器对预读的使用。

使用 onstat -g rah 命令来显示有关预读请求的统计信息。

相关链接

《*SinoDB* 管理员参考》*: AUTO_READAHEAD* 配置参数

《*SinoDB* 管理员参考》*: onstat -p* 命令*:* 显示概要文件计数

《*SinoDB* 管理员参考》*:onstat -g rah* 命令*:* 显示预读请求统计信息

数据库服务器线程对缓存页的访问

数据库服务器使用共享锁定缓存以允许多个数据库服务器线程可以同时访问共享内存中的同一缓冲区

数据库服务器使用两种类型的缓冲区锁来提供此并发性而不会在线程隔离中有所丢失。两种类型的锁定访问是共享与互斥。（有关更多信息，请参阅 缓冲区锁的类型 在第119页。）

将数据清空到磁盘

将缓冲区写入磁盘称为 缓冲区清空。

当用户线程修改缓冲区中的数据时，它将该缓冲区标记为脏。当数据库服务器将缓冲区清空到磁盘时，它随后会将缓冲区标记作为不脏，并允许覆盖缓冲区中的数据。

数据库服务器会清空以下缓冲区：

- 缓冲池（包含在此节中）
- 物理日志缓冲区

  （请参阅 清空物理日志缓冲区 在第124页 。）
- 逻辑日志缓冲区

  （请参阅 清空逻辑日志缓冲区 在第125页 。）

页清除程序线程管理缓冲区清空。数据库服务器总是运行至少一个页清除程序线程。如果数据库服务器配置多个页清除程序线程，那么 LRU 队列将分配给这些页清除程序以更有效地清空。有关指定数据库服务器运行多少页清除程序线程的信息，请参阅 《*SinoDB*® 管理员参考》中的 CLEANERS 配置参数。

清空物理日志缓冲区、已修改的共享内存页缓冲区以及逻辑日志缓冲区，必须依照维持数据一致性的规则与页清除程序活动同步。

清空缓冲池缓冲区

缓冲区的清空是由特定条件触发的。

当以下状况发生时将启动缓冲区的清空：

- MLRU 队列中缓冲区数达到 BUFFERPOOL 配置参数中 lru_max_dirty 值所指定的数目。
- 页清除程序线程无法跟上。换句话说，用户线程必须获取缓冲区时，却没有未修改的缓冲区可用。
- 数据库服务器必须执行检查点。(请参阅 检查点 在第266页。)

自动 LRU 调节会影响所有缓冲池，并调整 BUFFERPOOL 配置参数中的 lru_min_dirty 和 lru_max_dirty 的值。

首先清空前映像

已修改页的前映像在已修改页本身之前清空到磁盘。

实际上，首先清空物理日志缓冲区，然后清空包含已修改页的缓冲区。因此，即便由于用户线程正在尝试获取缓冲区但无可用的（前台写入）而必须清空共享内存缓冲区页时，也必须等到页的前映像写入磁盘后才能清空缓冲区页。

清空物理日志缓冲区

数据库服务器总是在将任何数据缓冲区清空到磁盘之前首先将物理日志缓冲区的内容清空到磁盘。

数据库服务器临时在物理日志缓冲区中存储一些已修改的磁盘页的前映像。如果前映像是写入物理日志缓冲区而非磁盘上的物理日志，那么服务器会在将已修改页清空到磁盘前先将物理日志缓冲区清空到磁盘。

以下事件将导致活动的物理日志缓冲区清空：

- 活动的物理日志缓冲区变满。
- 共享内存中已修改页必须清空，但其前映像仍然在活动的物理日志缓冲区中。
- 检查点出现。

数据库服务器每次只使用两个物理日志缓冲区中的一个。该缓冲区为活动的（或当前的）物理日志缓冲区。在数据库服务器将活动的物理日志缓冲区清空到磁盘之前，它会使其他缓冲区成为活动的物理日志缓冲区，以便服务器可以在清空第一个缓冲区时继续写入。

物理日志缓冲区和物理日志两者都有助于维护数据在物理上和逻辑上的一致性。有关物理日志记录、检查点以及快速恢复的信息，请参阅 物理日志记录、检查点和快速恢复 在第264页.

同步缓冲区清空

首次设置共享内存时，所有缓冲区都是空的。当处理发生时，数据页将从磁盘读入缓冲区，而用户线程开始修改这些页。

清空期间的写入类型

为提供有关提示缓冲区清空活动的特定条件的信息，数据库服务器定义了三种类型的写入 ，并计算每个写入发生的频率。

以下为写入类型：

- 前台写入
- LRU 写入
- 块写入

若要查看数据库服务器保留的写入计数，请运行 onstat -F 命令。

如果实现数据库服务器的镜像，那么数据总是先写入主块。然后复写到镜像块。对镜像块的写入次数包含在计数中。

相关链接

监视缓冲区 在第135页
《SinoDB 管理员参考》:onstat -F 命令: 显示计数

前台写入

sqlexec 线程将缓冲区写入磁盘，称作 前台写入。当 sqlexec 线程代表用户搜索 LRU 队列但找不到空的或未被修改的缓冲区时将发生前台写入。

为腾出空间， sqlexec 线程一次清空几页以容纳要从磁盘读入的数据。有关 更多信息，请参阅 *FIFO/LRU 队列* 在第120页.

如果 sqlexec 线程必须执行缓冲区清空以获取共享内存缓冲区，那么性能会变差。 必须避免前台写入。要显示前台写入次数的计数，请运行 onstat -F。如果您发现前台写入很规律地发生，请调整页清除参数的值。或增加页清除程序数，或减少 BUFFERPOOL 的 lru_max_dirty 的值。

LRU 写入

LRU 写入是由页清除程序而非 sqlexec 线程执行。数据库服务器以后台写入来执行 LRU 写入，通常当脏缓冲区的百分比超过 BUFFERPOOL 配置参数中 lru_max_dirty 指定的百分比时发生。

前台写入可触发 LRU 写入。当前台写入发生时，已执行写入的 sqlexec线程 将唤醒页清除程序来清除为之执行前台写入的 LRU。

在适当调整的系统中，页清除程序确保有足够的未修改缓冲区页可用于存储从磁盘读取的页。因此，执行查询的 sqlexec 线程在读取查询所需的磁盘页之前，无需将页清空到磁盘。对于不使用前台写入的查询，此种情况可使性能有显著地提高。

由于页清除程序线程执行缓冲区写入比 sqlexec 线程更有效率，因此 LRU 写入比前台写入更受欢迎。若要监视两种类型的写入，请使用 onstat -F。

块写入
块写入通常由页清除程序线程在检查点期间执行，或者可能在共享内存缓冲池里每一页被修改时执行。如同排序写入的块写入为可用于数据库服务器的最有效的写入。

在块写入中，每个页清除程序线程负责一个或多个块。每个页清除程序线程读取缓冲区头，并创建一组指针指向与特定块相关联的页面。（页清除程序对此信息有访问权，因为块编号包含在物理页编号地址中，它是页头的一部分。）此排序使磁头移动（磁盘搜索时间）最小化，并在可能的情况下，使页清除程序线程能够在写入时使用大缓冲区。

此外，由于用户线程必须等待检查点完成，因此页清除程序线程不会与大量线程争用 CPU 时间。因此，页清除程序线程可以通过更少的上下文切换来完成它们的工作。

清空逻辑日志缓冲区

许多事件可导致逻辑日志缓冲区清空。

数据库服务器使用共享内存的逻辑日志缓冲区作为临时存储区，用于记录数据库服务器页描述的修改。这些更改记录将从逻辑日志缓冲区写入磁盘上的当前逻辑日志文件，并最终写入逻辑日志备份介质。有关逻辑日志记录的说明，请参阅 逻辑日志 在第243页.

五个事件导致当前逻辑日志缓冲区清空：

- 当前逻辑日志缓冲区变满。
- 事务在未缓冲日志记录的数据库中准备或提交。
- 无日志记录数据库会话终止。
- 检查点出现。
- 已修改不需要物理日志中的前映像的页。

以下主题将详细说明每个事件。

当事务在未缓冲日志记录数据库中准备或终止之后
许多日志记录将导致使用未缓冲记录的数据库中的逻辑日志缓冲区清仓。

以下日志记录将导致清空：

- COMMIT
- PREPARE
- XPREPARE
- ENDTRANS

有关已缓冲日志记录与未缓冲日志记录的比较，请参阅《*SinoDB® SQL* 指南: 语法》中的 SET LOG 语句。

当使用无日志记录数据库或未缓冲日志纪录数据库的会话终止时
即使对于无日志记录数据库，数据库服务器会记录某些改变数据库模式的活动，如表或扩展的创建。

当数据库服务器终止使用未缓冲日志记录或无日志记录数据库的会话时，会清空逻辑日志缓冲区以确保记录任何日志记录活动。

当检查点出现时
检查点在特定事件之后出现。

有关检查点期间所发生事件的详细说明，请参阅 检查点 在第266页.

当已修改不需要物理日志中的前映像的页时
当已修改不需要物理日志中的前映像的页时，必须在页清空到磁盘前清空逻辑日志缓冲区。

缓冲区大对象数据

简单大对象（TEXT 或 BYTE 数据）可以存储在数据库空间 或 BLOB 空间。智能大对象（CLOB 或 BLOB 数据）只能存储在智能大对象空间。

数据库服务器使用不同的方法来访问每种类型的存储空间。以下主题将说明每种类型的缓冲方法。

写入简单大对象

数据库服务器以写入其他数据类型的相同方式 将简单大对象写入数据库空间中的磁盘页。

有关更多信息，请参阅将数据清空到磁盘 在第123页。

还可以将简单大对象指定给 BLOB 空间。数据库服务器将简单大对象写入 BLOB 空间的方式，与其将其他数据写入共享内存缓冲区然后清空至磁盘的方式不同。有关 BLOB 空间的说明，请参阅《*SinoDB*® 管理员参考》中有关磁盘结构和存储的章节。

BLOB 页和共享内存

BLOB 空间的 BLOB 页能存储大量数据。

数据库服务器不会经由共享内存缓冲池来创建或访问 BLOB 页，也不会将 BLOB 空间的 BLOB 页写入逻辑或物理日志。

如果 BLOB 空间数据通过共享内存池，它可能将通过去除索引页和数据页来降低池的有效性。相反，BLOB 页数据将在创建时直接写入磁盘。

为了减少逻辑日志和物理日志流量，数据库服务器使用与写入数据库空间页不同的方式将 BLOB 页从磁介质写入数去库空间备份磁带和逻辑日志备份磁带。有关如何记录 BLOB 空间的说明，请参阅记录 *BLOB* 空间和简单大对象 在第247页。

创建简单大对象

当简单大对象数据写入磁盘时，其所属的行可能还不存在。

例如，在插入操作期间，简单大对象会在该行的其他数据传送之前传送。在简单大对象存储后，将创建具有指向该位置的 56 位描述符的数据行。有关如何以物理方式存储简单大对象的说明，请参阅《*SinoDB*® 管理员参考》的磁盘存储和结构章节中有关数据库空间 BLOB 页结构的部分。

创建 BLOB 页缓冲区

为从应用程序接收简单大对象数据，数据库服务器创建一对 BLOB 空间缓冲区（一个用于读取，一个用于写入），大小为一个 BLOB 空间的 BLOB 页。每个用户仅拥有一组 BLOB 空间缓冲区，因此，一次只能访问一个简单大对象。

简单大对象数据将以 1KB 从客户端应用程序进程传送到数据库服务器。数据库服务器开始用 1KB 块填充 BLOB 空间缓冲区并尝试一次缓冲两个 BLOB 页。数据库服务器缓冲两个 BLOB 页以便可以确定何时添加从一个页到下一个页的转发指针。当它填满了第一个缓冲区并发现还有更多的数据要传送时，它会在向磁盘写入页之前添加到下一页的转发指针。当不再有数据要传送时，数据库服务器会将不带转发指针的最后一页写入磁盘。

当线程开始将第一个 BLOB 空间缓冲区写入磁盘时，它尝试根据用户定义的 BLOB 页大小来执行 I/O。例如，如果 BLOB 页大小是 32KB，那么数据库服务器尝试以 32,768 字节的增量来读取或写入数据。如果底层硬件（例如磁盘控制器）不能在单个操作中传送这个数据量，那么操作系统内核将在内部（以内核模式）循环直到传送完成。

BLOB 空间缓冲区会保留到创建它们的线程完成。当简单大对象写入磁盘后，数据库服务器将释放 BLOB 空间缓冲区对。下图说明将简单大对象写入 BLOB 空间的过程。

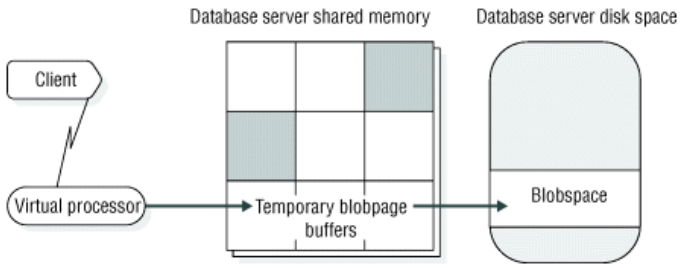

图 24: 将简单大对象写入 BLOB 空间

用自由图页分配与追踪 BLOB 空间的 BLOB 页 。如需要，将创建链接连接 Blob 页和指向下一个 BLOB 页段的指针。

操作记录（插入、更新或删除）将写入逻辑日志缓冲区。

相关链接

共享内存的缓冲池部分 在第113页

访问智能大对象

数据库服务器通过共享内存缓冲区访问智能大对象的方法，与其访问存储在数据库空间的数据的方法相同。然而，智能大对象的用户数据部分是缓存在比正常缓冲区页更低优先级的缓冲区，以防止将更高价值的数据清空出缓冲池。缓冲允许对于经常访问的智能大对象进行更快的访问。

智能大对象存储在智能大对象空间中。您不能将简单大对象存储在智能大对象空间，也不能将智能大对象存储在 BLOB 空间。智能大对象空间由用户数据区域和元数据区域组成。用户数据区包含智能大对象数据。元数据区域包含有关智能大对象空间内容的信息。有关智能大对象空间的更多信息，请参阅 智能大对象空间 在第146页。

因为智能大对象通过共享内存缓冲池并可以被记录，所以您在分配缓冲区时必须考虑到它们。使用 BUFFERPOOL 配置参数来分配共享内存缓冲区。通常，尝试分配足够的缓冲区，以便为每个同时打开的智能大对象保存两个智能大对象页。（附加页可用于预读用途。）有关为智能大对象调整缓冲区的更多信息，请参阅 《*SinoDB*® 性能指南》。

使用 LOGBUFF 配置参数来指定逻辑日志缓冲区的大小。有关设置以下配置参数的信息，请参阅 《*SinoDB*® 管理员参考》。

- BUFFERPOOL
- LOGBUFF

已记录的智能大对象的用户数据区域不会通过物理日志，因此不需要为智能大对象更改 PHYSBUFF 参数。

有关智能大对象空间结构的更多信息，请参阅 《*SinoDB*® 管理员参考》的磁盘结构和存储章节中有关智能大对象空间结构的部份。有关创建智能大对象空间的信息，请参阅 《*SinoDB*® 管理员参考》中有关 onspaces 实用程序的信息。

64 位平台上的内存使用

通过 64 位寻址，您可以有更大的缓冲池以减少从磁盘获取数据的 I/O 操作量。

因为 64 位平台允许更大的内存地址空间，所以以下内存相关的配置参数的最大值在 64 位平台上更大：

- BUFFERPOOL
- CLEANERS
- DS_MAX_QUERIES
- DS_TOTAL_MEMORY
- LOCKS
- SHMADD
- SHMVIRTSIZE

每个 64 位平台的机器说明文件都会列出这些配置参数和特定于平台的参数（例如 SHMMAX）的最大值。有关配置参数的更多信息，请参阅 《*SinoDB*® 管理员参考》和《*SinoDB*® 性能指南》 中有关共享内存的章节。

管理共享内存

这些主题告诉您如何执行有关管理共享内存的下列任务：

- 设置共享内存的配置参数
- 设置共享内存
- 开启或关闭数据库服务器共享内存的驻留部分的驻留
- 添加段到共享内存的虚拟部分
- 为关键活动预留内存
- 在具有内存限制的应用中保留目标内存量
- 监视共享内存

These topics do not cover the DS_TOTAL_MEMORY configuration parameter. This parameter places a ceiling on the allocation of memory for decision-support queries. For information about this parameter, see your 《*SinoDB*® 性能指南》.

这些主题不包括 DS_TOTAL_MEMORY 配置参数。此参数为决策支持查询的内存分配设置上限。有关此参数的信息，请参阅 《*SinoDB*® 性能指南》。

设置操作系统共享内存配置参数

一些操作系统配置参数可以响数据库服务器使用共享内存。

由于参数名会随平台而变，而且并不是所有的参数都存在于所有平台上，因此不提供参数名。以下列表按功能描述这些参数：

- 最大操作系统共享内存段大小（以字节或 KB 表示）
- 最小共享内存段大小（以字节表示）
- 最大共享内存标识符数
- 共享内存的下边界地址
- 每个进程连接的最大共享内存段数
- 整个系统的最大共享内存量

  仅 UNIX:

  - 最大信号标识符数
  - 最大信号数
  - 每个标识符的最大信号数

  在 UNIX™ 平台上，机器说明文件包含了可用于配置操作系统资源的推荐值。当配置操作系统时请使用这些推荐值。有关如何设置这些操作系统参数的信息，请查阅操作系统手册。

  有关操作系统环境的特定信息，请参阅数据库服务器随附的机器说明文件。

相关链接

《*SinoDB* 性能指南》: 影响 *CPU* 利用率的 *UNIX* 配置参数

《*SinoDB* 性能指南》: 影响 *CPU* 利用率的 *Windows* 配置参数

最大共享内存段大小

当数据库服务器创建所需的共享内存段时，它将尝试获取尽可能大的操作系统段。

数据库服务器尝试获取的第一数据段大小等于其正在分配的部分（驻留、虚拟或通讯）的大小，并四舍五入到 8KB 的最近倍数。

如果所请求的段大小超过所允许的最大大小，那么数据库服务器将从操作系统接收错误。如果数据库服务器接收到错误，那么它将请求的大小除以二，然后再试一次。获取尝试将持续直到可以创建 8 KB 倍数的最大分段大小为止。然后，数据库服务器将根据需要创建多个附加段。

使用 2 GB 以上内存 (Windows™)
在 Windows™ 上数据库服务器可以访问大于 2 GB 的共享内存段。

Windows™ 2003 及更早的版本，必须在 Windows™ 启动文件中使用一个条目来启动此功能。启用更大的共享内存段，Microsoft™ 称为 4 GB 调优（4GT）。

要添加条目，请编辑 boot.ini 文件（位于顶层或根目录）。您可以添加启动选项或使用目前现有的启动选项。要启用对大于 2GB 的支持，请将以下文本添加到启动行的末尾：

```
/3GB
```

以下示例启用大于 2 GB 的支持：

```
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Windows NT
Workstation Version 4.00"
/3GB
```

The maximum size of the shared-memory segment depends on the operating system, but it is approximately 3 gigabytes for Windows™ without additional drivers.

共享内存段的最大大小取决于操作系统，但是对于没有附加驱动程序的 Windows™ 系统来说，大小约为 3 GB。

最大共享内存标识符数 (UNIX™)
操作系统通过共享内存标识符来标识每个共享内存段。共享内存标识符会在虚拟处理器尝试连接到共享内存时影响数据库服务器的运行。

对于大多数操作系统来说，虚拟处理器以先到先服务的原则接收标识符，直到达到为操作系统所定义的限制。有关共享内存标识符的更多信息，请参阅 虚拟处理器如何连接共享内存 在第109页。

或许可通过将共享内存标识符数乘以最大共享内存段大小来计算出操作系统可以分配的最大共享内存量。

信号 (UNIX™)

数据库服务器操作需要每个虚拟处理器配备一个 UNIX™ 信号，每个通过共享内存 (ipcshm 协议) 连接到数据库服务器的用户配备一个，数据库服务器实用程序配备六个，其他用途配备十六个。

设置数据库服务器共享内存配置参数

可以修改影响共享内存的常驻、缓冲池或虚拟部分的配置参数。

设置缓冲池共享内存的参数

onconfig 文件中的 BUFFERPOOL 配置参数指定有关缓冲池的信息。数据库服务器所使用的每个页大小都需要一个缓冲池（由 onconfig 文件中的 BUFFERPOOL 配置参数条目表示）。

设置常驻共享内存的参数

以下列表包含 onconfig 文件中的参数，这些参数指定了缓冲池和内部表在共享内存常驻部分中的配置。要使变更的配置参数生效，您必须关闭并重新启动数据库服务器。

LOCKS
: 指定数据库对象（例如：行、键值、页和表）的初始锁数。

LOGBUFF
: 指定逻辑日志缓冲区的大小。

PHYSBUFF
: 指定物理日志缓冲区的大小。

RESIDENT
: 指定数据库服务器共享内存常驻部分的驻留。

SERVERNUM
: 指定本地主计算机上数据库服务器的唯一标识号。

SHMTOTAL
: 指定将由数据库服务器使用的总内存量。

设置虚拟共享内存的参数

以下列表包含用于配置共享内存的虚拟部分的配置参数：

DS_HASHSIZE
: 数据分布高速缓存中列表的散列存储区数。

DS_POOLSIZE
: 数据分布高速缓存中的最大条目数。

PC_HASHSIZE
: 指定 UDR 高速缓存和数据库服务器使用的其他高速缓存的散列存储区数。

PC_POOLSIZE
: 指定可以存储在 UDR 高速缓存中的 UDR（SPL 例程和外部例程）的数目。此外，此参数指定其他数据库服务器高速缓存（如 typename 高速缓存和 opclass 高速缓存）的大小。

SHMADD
: 指定动态添加的共享内存段的大小。

SHMNOACCES
: 指定不用于连接共享内存的虚拟内存地址范围的列表。使用此参数可避免与其他进程发生冲突。

EXTSHMADD
: 指定用户定义的例程或 DataBlade® 例程在用户定义的虚拟处理器运行时，添加的虚拟扩展段的大小。

SHMTOTAL
: 指定将由数据库服务器使用的总内存量。

SHMVIRTSIZE
: 指定共享内存的虚拟部分的初始大小。

STACKSIZE
: 指定数据库服务器用户线程的堆栈大小。

设置共享内存性能的参数

以下配置参数影响共享内存性能。

AUTO_READAHEAD

为查询指定自动预读模式或禁用自动预读操作。当数据库服务器检测到查询遇到 I/O 时，自动预读操作可以通过发出异步页请求来提高查询性能。异步页请求通过将查询处理过程与从磁盘检索数据并将数据放入缓冲池所需的处理相叠加来提高查询性能。

CKPTINTVL

在 RTO_SERVER_RESTART 配置参数没有设置为打开自动检查点调整，用于指定在数据库服务器检查是否需要检查点之前经过的最大秒数。

CLEANERS

指定数据库服务器将运行的页清除程序线程数。

相关链接

《*SinoDB* 管理员参考》：数据库配置参数

《*SinoDB* 管理员参考》：更改 *onconfig* 文件

《*SinoDB* 性能指南》：影响内存利用率的配置参数

《*SinoDB* 性能指南》：共享内存

设置 SQL 语句高速缓存参数

SQL 语句高速缓存有不同的配置方式。

下表显示可以配置 SQL 语句高速缓存的不同方法。

表 20: 配置 SQL 语句高速缓存

配置参数	用途	onmode 命令
STMT_CACHE	打开、启用或禁用内存中的 SQL 语句高速缓存。如果已打开，请指定 SQL 语句缓存能否保存已分析和优化的 SQL 语句。	onmode -e ***mode***
STMT_CACHE_HITS	指定将语句完全插入 SQL 语句高速缓存前，命中（引用）该语句的次数。	onmode -W STMT_CACHE_HITS
STMT_CACHE_NOLIMIT	控制是否在 SQL 语句高速缓存的大小大于 STMT_CACHE_SIZE 值后将语句插入 SQL 语句高速缓存中。	onmode -W STMT_CACHE_NOLIMIT
STMT_CACHE_NUMPOOL	定义 SQL语句高速缓的内存池数。	无
STMT_CACHE_SIZE	指定 SQL 语句高速缓存的大小。	无

使用以下的 onstat 选项可监视 SQL 语句高速缓存：

- onstat -g ssc
- onstat -g ssc all
- onstat -g ssc pool

有关这些配置参数、onstat -g 选项和 onmode 命令的更多信息，请参阅 《*SinoDB*® 管理员参考》。

For more information about using the SQL statement cache, monitoring it with the onstat -g options, and tuning the configuration parameters, see improving query performance in the 《*SinoDB*® 性能指南》. For details on qualifying and identical statements, see the 《*SinoDB*® *SQL* 指南: 语法》.

有关使用 SQL语句高速缓存、使用 onstat -g 选项监视它，并调整配置参数的更多信息，请参阅 《*SinoDB*® 性能指南》 中有关提高查询性能的部份。有关限定和恒等语句的详情信息，请参阅《*SinoDB*® 性能指南》。

设置共享内存

要设置共享内存时，先将数据库服务器脱机再联机。

有关如何将数据库服务器从联机模式转换为脱机模式的信息，请参阅 更改数据库服务器运行模式 在第78页。

打开或关闭常驻共享内存的驻留

您可以打开或关闭共享内存常驻部分的驻留。

可以使用以下任一方式打开或关闭驻留：

- 在数据库服务器处于联机模式时，使用 onmode 实用程序可立即逆转共享内存驻留状态。
- 更改 onconfig 文件中的 RESIDENT 参数，可在下次设置数据库服务器共享内存时打开或关闭共享内存驻留。

有关共享内存常驻部分的说明，请参阅 共享内存常驻部分 在第110页。

在联机模式下打开或关闭驻留

在数据库服务器处于联机模式下，请使用 onmode 实用程序来打开或关闭驻留。

要立即打开共享内存常驻部分的驻留，请运行以下命令：% onmode -r

要立即关闭共享内存常驻部分的驻留，请运行以下命令：% onmode -n

这些命令不会更改 onconfig 文件中的 RESIDENT 参数值。也就是说，这种更改不是永久的，驻留状态会在您下一次设置共享内存时将回复到 RESIDENT 参数所指定的状态。在 UNIX™ 平台上，您必须是 root 或 informix 用户才能打开或关闭驻留。在 Windows™ 平台上，您必须是 Informix-Admin 组的用户才能打开或关闭驻留。

重启数据库服务器时打开或关闭驻留

可以使用文本编辑器打开或关闭驻留。

要更改驻留的当前状态，请使用文本编辑器找到 RESIDENT 参数。将 RESIDENT 设置为 1 以打开驻留或者将其设置为 0 以关闭驻留，然后将此文件写入磁盘。在更改生效之前，必须关闭并重新启动数据库服务器。

将段添加到共享内存的虚拟部分

可以使用 onmode 实用程序的 -a 选项将指定大小的段添加到虚拟共享内存。

正常情况下，无需要将段添加到虚拟共享内存，因为数据库服务器会按需自动添加段。

如果操作系统段的数目受到限制，并且相对于所需的量，初始段的大小很小以致于快要超过操作系统对共享内存段的限制，这时使用 onmode 实用程序添加段是有用的。

相关链接

共享内存虚拟部分的大小 在第114页

为关键活动保留内存

保留特定的内存量，以供需要执行关键活动（如回滚活动）且数据库服务器可用内存有限时使用。如果数据库服务器在关键活动期间耗尽可用内存，则可防止数据库服务器崩溃。

如果通过将 LOW_MEMORY_RESERVE 配置参数设置为指定的值（以 KB为单位）来启用此配置参数，那么用户即使收到内存不足错误，也可完成回滚之类的关键活动。如果 LOW_MEMORY_RESERVE 的值为 0，将关闭低内存保留功能。

例如，合理的保留内存量为 512 KB。要保留 512 KB，请指定：

```
LOW_MEMORY_RESERVE 512
```

还可使用 onmode -wm 或 onmode-wf 命令来动态调整 LOW_MEMORY_RESERVE 配置参数的值。

使用 onstat -g seg 命令来监视 LOW_MEMORY_RESERVE 的值。查找输出的最后两行，其中包含短语 "low memory reserve"。其中的第一行显示保留内存的大小（KB）。第二行显示数据库服务器已使用此内存的次数以及所需的最大内存量。当服务器重启时，这两个值都将重置。

相关链接

《SinoDB 管理员参考》: LOW_MEMORY_RESERVE 配置参数

《SinoDB 管理员参考》: onstat -g seg 命令: 显示共享内存段的统计信息

《SinoDB 管理员参考》: onmode -wf#-wm: 动态更改某些配置参数

配置内存严重过低时的服务器响应

您可配置主数据库服务器或标准数据库服务器内存严重过低时所采取的操作。可以根据空闲时间、内存使用和其他因素来指定终止会话的条件，这样目标应用程序可继续处理。对于具有内存限制的嵌入式应用程序，低内存响应是很有用的。

在主服务器或标准服务器上设置自动低内存管理：

- 将 LOW_MEMORY_MGR 配置参数设置为1，在数据库服务器启动时，将启用低内存管理。
- 通过使用带有 scheduler lmm enable 参数的 SQL 管理 API 命令为需要维护的内存量设置阈值参数。
- 确认 SHMTOTAL 配置参数设置值为正整数。

要禁用自动低内存管理，请运行带有 scheduler lmm disable 参数的 SQL 管理 API 命令

相关链接

《SinoDB 管理员参考》: LOW_MEMORY_MGR 配置参数

《SinoDB 管理员参考》: scheduler lmm enable 参数: 指定自动低内存管理设置 (SQL 管理 API)

《SinoDB 管理员参考》: scheduler lmm disable 参数: 停止自动低内存管理 (SQL 管理 API)

《SinoDB 管理员参考》: onstat -g lmm 命令: 显示低内存管理信息

目标内存量的维护方案

本主题中的方案显示在具有内存限制的应用程序中如何维护目标内存量。

假设你想要指定数据库服务器在可用内存等于或小于 10 MB 时开始运行低内存管理进程，以停止应用程序并释放内存。假设你还希望指定服务器在可用内存等于或大于 20 MB 时停止运行低内存管理进程：

1. 将 LOW_MEMORY_MGR 配置参数设置为1 并重新启动服务器，或者运行 onmode -wf 命令来改变 LOW_MEMORY_MGR 配置参数的值。
2. 运行带有 scheduler lmm enable 参数和低内存参数的 SQL 管理 API 命令，如下所示：

```
XECUTE FUNCTION task("scheduler lmm enable",
 "LMM START THRESHOLD", "10MB",
 "LMM STOP THRESHOLD", "20MB",
```

```
"LMM IDLE TIME", "300");
```

3. 运行 onstat -g lmm 命令来显示有关自动低内存管理的设置，包括服务器尝试维持的内存量、服务器当前使用的内存量、低内存启动与停止阈值，以及内存有关的其他统计信息。

   还可在 online.log 文件中查看低内存管理信息。

相关链接

《SinoDB 管理员参考》: LOW_MEMORY_MGR 配置参数

《SinoDB 管理员参考》: scheduler lmm enable 参数: 指定自动最低内存管理设置 (SQL 管理 API)

《SinoDB 管理员参考》: scheduler lmm disable 参数: 停止自动最低内存管理 (SQL 管理 API)

《SinoDB 管理员参考》: onstat -g lmm 命令: 显示低内存管理信息

监视共享内存

这些主题描述如何监视共享内存段、共享内存概要文件，和特定的共享内存资源（缓冲区、锁存器和锁定）的使用。

可使用 onstat -o 实用程序捕获数据库服务器共享内存的静态快照用于以后分析和比较。

监视共享内存段

监视共享内存段以确定数据库服务器创建的段的数目和大小。

数据库服务器动态分配共享内存段，因此这些数字会改变。如果数据库服务器正在分配过多的共享内存段，那么您可以增加 SHMVIRTSIZE 配置参数的值。有关更多信息，请参阅 《*SinoDB*® 管理员参考》 中有关配置参数的主题。

onstat -g seg 命令列出每个共享内存段的信息，包括段的地址和大小，以及可用或正使用的内存量。有关 onstat -g seg 输出的示例，请参阅 《*SinoDB*® 管理员参考》 中有关 onstat 实用程序的信息。

监视共享内存概要文件和锁存器

监视数据库服务器概要文件以分析性能和共享内存资源的使用。

您可以获取锁存器使用的统计信息，以及有关特定锁存器的信息。这些统计数据可用作衡量系统活动的标准。

要将这些统计数据重置为零，请使用 onstat -z 选项。有关 onstat 显示的所有字段的描述，请参阅 《*SinoDB*® 管理员参考》 中有关 onstat 实用程序的信息。

用于监视共享内存和锁存器的命令行实用程序
使用命令行实用程序来监视共享内存和锁存器

可以使用以下的命令行实用程序来监视共享内存和锁存器：

onstat -s

使用 onstat -s 命令获取锁存器信息。

onstat -p

运行 onstat -p 来显示有关数据库服务器活动以及等待锁存器的统计信息（在 lchwaits 字段中）。有关 onstat -p 输出的示例，请参阅 《*SinoDB*® 管理员参考》 中有关 onstat 实用程序的信息。

SMI 表
查询 sysprofile 表以获取共享内存统计信息。

此表包含 onstat -p 输出中的所有统计信息，除 ovbuff、usercpu 和 syscpu 的统计信息以外。

监视缓冲区

您可以同时获取缓冲区使用的统计信息以及有关特定缓冲区的信息。

统计信息包括高速缓存到缓冲区的数据写入的百分比，以及线程需要等待获取缓冲的次数。高速缓存写入的百分比是重要的性能指标。缓冲区的等待次数提供了系统并发性的测量方法。

特定缓冲区的信息包括共享内存中线程持有的所有缓冲区的列表。您可以使用这些信息来跟踪特定缓冲区的状态。例如，可以确定是否另一线程正在等待该缓冲区。

可以获取有关缓冲区可用性的统计信息，以及每个 LRU 队列中的缓冲区信息。统计信息包括数据库服务器尝试超越最大缓冲区数的次数和磁盘写入数的次数（按导致缓冲区清空的事件分类）。这些数据可以帮助您确定缓冲区数是否适当。每个 LRU 队列中的缓冲区信息包含队列长度，以及队列中已修改的缓冲区百分比。

可以从 onstat 实用程序、sysprofile SMI 表或 SinoDB® 开放管理工具 (OAT) 获取有关缓冲池活动的信息。

使用 onstat 命令监视缓冲区

可以使用以下 onstat 命令监视缓冲区：

onstat -g buf

运行 onstat -g buf 命令以获取有关每个缓冲区的积极性和效率的统计信息。显示以下统计类型：

- 页读取和写入
- 高速缓存百分比
- 等待缓冲区
- 清空
- 缓冲池扩展
- 缓冲池段
- 快速高速缓存

onstat -B

运行 onstat -B 命令以获取有关所有不在可用列表中的缓冲区的信息，包括：

- 缓冲区的共享内存地址
- 当前持有缓冲区的线程的地址
- 正在等待每个缓冲区的第一个线程的地址
- 有关缓冲池的信息

onstat -b

运行 onstat -b 命令以获取以下有关每个缓冲区的信息：

- 当前线程持有的每个缓冲区的地址
- 缓冲区中所容纳的页的页号
- 缓冲区所容纳的页类型（如数据页、表空间页等）
- 放置在缓冲区上的锁定类型（互斥或共享）
- 当前正持有缓冲区的线程的地址
- 正在等待每个缓冲区的第一个线程的地址
- 有关缓冲池的信息

您可以将用户线程地址和 onstat -u 输出中显示的地址相比较以获取会话标识号。

onstat -X

运行 onstat -X 命令获取与 onstat -b 相同的信息，以及所有正在等待缓冲区的线程的列表，不单单只是第一个等待的线程。

onstat -R

运行 onstat -R 命令显示有关缓冲池的信息、每个 LRU 队列中的缓冲区数，以及已修改或可用缓冲区的数目与百分比。

onstat -F

运行 onstat-F 命令获取已执行的写入类型的计数，以及以下有关页清除程序的信息：

- 页清除程序数
- 页清除程序共享内存地址
- 页清除程序的 Current 状态
- 分配给页清除程序的 LRU 队列

sysprofile SMI 表

查询 sysprofile 表以获取高速缓存读取与写入、写入类型和总缓冲区等待数的统计信息。以下为相关行。

bufreads

从缓冲区读取的次数

bufwrites

写入缓冲区的次数

buffwts

任何线程需要等待缓冲区的次数

chunkwrites

块写入数

dskreads

从磁盘读取的次数

dskwrites

写入磁盘的次数

fgwrites

前台写入数

lruwrites

LRU 写入数

相关链接

清空期间的写入类型 在第124页

《*SinoDB* 管理员参考》*: onstat -g buf* 命令*:* 显示缓冲池的概要文件信息

《*SinoDB* 管理员参考》*: onstat -b* 命令*:* 显示正在使用的缓冲区的信息

《*SinoDB* 管理员参考》*: onstat -B* 命令*:* 显示已用缓冲区的信息

《*SinoDB* 管理员参考》*: onstat -X* 命令*:* 显示线程信息

《*SinoDB* 管理员参考》*: onstat -R* 命令*:* 显示 *LRU*、*FLRU* 和 *MLRU* 队列信息

《*SinoDB* 管理员参考》*:onstat -F* 命令*:* 显示计数

《*SinoDB* 管理员参考》*: sysprofile*

服务器故障后删除共享内存段

数据库服务器故障后，必须关闭共享内存段。

重要： 该过程必须由具有 SinoDB® 使用经验的 DBA 来执行。请咨询技术支持以获取协助。此过程仅适用于 UNIX™ 系统。

当数据库服务器实例发生故障时，请遵循以下步骤删除共享内存段：

1. 以用户 informix 的身份登录：

2. 使用 onmode -k 命令 将数据库服务器切换为脱机模式，然后删除共享内存。
3. 如果 onmode -k 命令失败并且服务器未脱机，请执行 onclean -k 命令，或是执行以下步骤：
   a) 使用 onstat -g glo 命令显示多线程信息。
   b) 从上面命令的输出中，找到与 class 列中第一个 cpu 实例关联的进程标识符 (pid) 。
   例如，onstat -g glo 命令的以下输出中的 class 列内出现 4 次 cpu，其 pids 分别为 2599、2603、2604 和 2605：

```
MT global info:
sessions threads  vps      lngspins
0        49       14       1
          sched calls     thread switches yield 0   yield n   yield forever
total:    900100          898846          1238      27763     423778
per sec:327             325             2         12        151
Virtual processor summary:
 class       vps       usercpu   syscpu    total
 cpu         4         0.92      0.10      1.02
 aio         4         0.02      0.02      0.04
 lio         1         0.00      0.00      0.00
 pio         1         0.00      0.00      0.00
 adm         1         0.00      0.01      0.01
 msc         1         0.00      0.00      0.00
 fifo        2         0.00      0.00      0.00
 total       14        0.94      0.13      1.07
Individual virtual processors:
 vp    pid       class       usercpu   syscpu    total
 1     2599      cpu         0.25      0.06      0.31
 2     2602      adm         0.00      0.01      0.01
 3     2603      cpu         0.23      0.00      0.23
 4     2604      cpu         0.21      0.03      0.24
 5     2605      cpu         0.23      0.01      0.24
 6     2606      lio         0.00      0.00      0.00
 7     2607      pio         0.00      0.00      0.00
 8     2608      aio         0.02      0.02      0.04
 9     2609      msc         0.00      0.00      0.00
 10    2610      fifo        0.00      0.00      0.00
 11    2611      fifo        0.00      0.00      0.00
 12    2612      aio         0.00      0.00      0.00
 13    2613      aio         0.00      0.00      0.00
 14    2614      aio         0.00      0.00      0.00
                 tot         0.94      0.13      1.07
```

   c) 使用 kill 命令（按顺序）终止进程标识符为 2599、 2603、 2604 和 2605 的进程。
4. 如果尚未移除共享段，请遵循以下步骤：
   a) 确定服务器编号。
   可以通过检查 SinoDB® 实例的 onconfig 文件来找到服务器编号
   b) 将服务器编号加上 21078。
   例如，如果服务器编号是 1，加上 21078 得到 21079。
   c) 将上一步骤得到数值转换为十六进制。
   前面示例中，21079 的十六进制为 5257.
   d) 将上一步骤的十六进制值末尾添上 48。
   例如，525748。
   e) 以 root 身分运行 ipcs 实用程序来显示服务器上尚未关闭的共享内存段（如果有的话）。搜索 key 列以查找 *4.d* 在第137页 中得出的数字。
   f) 删除与 *4.d* 在第137页 中得出的数字关联的每个共享内存标识符。

获取有关 onclean 实用程序的更多信息，请参阅 《*SinoDB®* 管理员参考》。

参阅您的操作系统手册，获取与您的系统相匹配的 ipcm语法。

数据存储

数据库服务器使用物理存储单元分配磁盘空间。它以逻辑单元来存储数据。与大小会产生变动的逻辑单元不同，每个物理单元的大小都是固定的或指定的，它们的大小是由磁盘体系结构所确定。

以下主题将定义术语，并且说明磁盘空间管理必须了解的概念。这些主题涵盖以下领域：

- 数据库服务器用于在磁盘上存储数据的物理和逻辑单元的定义
- 关于如何计算储存数据所需的磁盘空间量的指示信息
- 关于如何配置磁盘空间以及放置数据库与表的指导
- 关于使用外部表的指示信息

数据库服务器使用以下物理存储单元管理磁盘空间。

块 在第138页
页 在第141页
BLOB 页 在第141页
智能大对象页 在第142页
扩展数据块 在第142页

数据库服务器在以下逻辑单元中储存数据：

数据库空间 在第144页
临时数据库空间 在第145页
BLOB 空间 在第146页
智能大对象空间 在第146页
临时智能大对象空间 在第152页
物理日志空间 在第153页
外部空间 在第154页
数据库 在第154页
表 在第155页
表空间 在第160页
分区和偏移量 在第140页

数据库服务器保持以下储存结构以确保数据的物理和逻辑一致性：

逻辑日志
物理日志
保留页

相关链接

《*SinoDB* 管理员参考》: *SinoDB* 中的限制
存储空间的创建和管理 在第26页
管理磁盘空间 在第168页
《*SinoDB* 管理员参考》: 保留页

块

块 是专用于数据库服务器数据储存的物理磁盘的最大单元。

块可以为管理员提供用于分配磁盘空间的特别大的单元。单个块的最大大小为 4TB。允许的块数为 32,766 。

以下的存储空间由块组成：

- 数据库空间
- BLOB 空间
- 智能大对象空间
- 临时数据库空间
- 临时智能大对象空间

创建块时，请指定其路径、大小和关联的储存空间名称。

数据库服务器也将块用于镜像。对块制作镜像时，数据库服务器将对该块上数据维持两个数据副本。每次对主块的写入操作，都会自动向镜像块执行同样的写入操作。读取操作在两个块之间平均分配。如果主块或镜像块故障，故障的块将标记为关闭，另一块会执行所有操作而不会中断用户对数据的访问。

创建表、索引和其他数据对象时，会将块空间分配或指定给那些对象。分配的空间并不一定会使用到。例如，创建表时会为其分配空间，但直到向该表添加数据时才使用了该空间。当数据库空间中的所有块报告可用页数为 0 时，将无法在该数据库空间中创建新的数据对象。但是，只要现有数据库对象有未使用的空间，您就可以继续向这些数据库对象添加数据。可以通过 onstat -d 命令或 SinoDB® 开放管理工具 (OAT) 来监视块。

相关链接

智能大对象空间 在第146页
BLOB 空间 在第146页
数据库空间 在第144页
指定存储空间和块的名称 在第172页
镜像 在第274页
《*SinoDB* 管理员参考》: *onstat -d* 命令: 显示块信息
《*SinoDB* 管理员参考》: *onmode -BC*: 允许大块模式

块的磁盘分配

数据库服务器可以使用常规操作系统文件或原始磁盘设备来储存数据。在 UNIX™ 上，只要性能是重要的，就必须使用原始磁盘设备来储存数据。在 Windows™ 上，推荐使用 NTFS 文件储存数据以便于管理。

存储空间可在挂载的 NFS 文件系统上，该文件系统使用常规操作系统文件。

Windows™ 上的磁盘访问

在 Windows™ 上，原始磁盘和 NTFS 都使用内核异步 I/O (KAIO)。 Windows™ 文件系统管理器会将额外开销添加到磁盘 I/O，因此使用原始磁盘可稍许提高性能。由于 NTFS 文件是储存数据的更标准方式，因此必须使用 NTFS 文件而不是原始磁盘。如果数据库服务器需要大量磁盘访问，请考虑使用原始磁盘。

Windows™ 上的原始磁盘空间

在 Windows™ 上，原始磁盘空间可以是不带盘符的物理驱动器，或是使用磁盘管理器配置了盘符的逻辑磁盘分区。该空间可以是格式化的或非格式化的。如果它包含数据，那么在该空间分配给数据库服务器后，将覆盖掉其上的数据。有关更多信息，请参阅 在 *Windows*上分配原始磁盘空间 在第171页。

NTFS 文件

对于 Windows™ 上的磁盘空间，必须使用 NTFS 文件而不是 FAT 文件。有关更多信息，请参阅 在 *Windows* 上分配 *NTFS* 文件空间 在第171页。

UNIX™ 上未缓冲或已缓冲磁盘的访问

You can allocate disk space in two ways. You can either use files that are buffered through the operating system, or you can use unbuffered disk access. 可以通过两种方式分配磁盘空间。可以使用通过操作系统缓冲的文件或使用未缓冲的磁盘访问。

通过操作系统缓冲的文件通常称为熟 文件 。

未缓冲的磁盘访问也称为原始磁盘空间。

当数据库空间位于 原始磁盘设备 （也称为 字符专用设备）上时，数据库服务器使用未缓冲的磁盘访问。

要创建原始设备，请以原始界面配置块设备（硬盘）。该设备提供的存储空间称为原始磁盘空间。原始磁盘空间的块在物理上是连续的。

块的名称是 /dev 目录中字符专用文件的名称。在许多操作系统中，您可以通过文件名的第一个字母（通常是 r）来区别字符专用文件与块专用文件。例如， /dev/rsd0f 是与 /dev/sd0f 块专用设备相应的的字符专用设备。

有关更多信息，请参阅 在 *UNIX* 上分配原始磁盘空间 在第170页。

熟文件 为操作系统管理的常规文件。熟文件块和原始磁盘块是同等可靠的。与原始磁盘空间不同，熟文件的逻辑连续块在物理上可能是不连续的。

可以比原始磁盘空间更方便地分配熟文件。要分配熟文件，必须在任何现有分区上创建该文件。块的名称是该文件的完整路径名。有关这些步骤的描述，请参阅 在 *UNIX*上分配熟文件空间 在第170页。

对于性能要求不高的学习环境，或对于静态数据而言，熟文件会很方便。如果必须使用UNIX™ 熟文件，请将最少访问的数据存储在那些文件中。将文件存储在最少活动的文件系统中。

对于熟文件块，操作系统将处理来自其自身缓冲池的所有块 *I/O*，并且确保所有对块的写入都在物理上写入磁盘。

> 重要： 虽然通常必须在 UNIX™ 上使用原始磁盘设备以获得更好的性能，但如果启用了 DIRECT_IO 配置参数，那么熟文件的性能可接近用于数据库空间块的原始设备的性能。这是由于直接 I/O 会绕过文件系统缓冲区的使用。在 AIX® 操作系统上，还可以为 SinoDB® 启用并发 I/O，以便在对使用熟文件的块执行读取和写入时使用直接 I/O。有关使用直接 IO 或并发 IO 的更多信息，请参阅《*SinoDB*® 性能指南》。

要针对性能确定最佳设备，请在系统上对数据库空间和表布局使用两种类型的设备执行基准测试。

使用原始磁盘时，无需采取任何特殊操作来创建大于 2GB 的块和文件。如果要在熟文件中创建大块，或者要将各种数据库导出和导入实用程序用于大文件，那么必须确保正确配置将保存大文件的文件系统。

可扩展块

可扩展快是SinoDB® 可自动扩展的块，或当应用程序需要额外存储空间时可手动扩展的块。如果有可扩展块，那么无需添加新的块或耗费时间尝试确定哪个存储空间将耗尽，以及何时将耗尽空间。

配置 SinoDB® 自动添加更多存储空间，可以防止发生以下情况的错误：分区需要额外存储空间时，却无法在该分区所在空间内任一块找到该空间。

可扩展块必须位于非镜像数据库空间或临时数据库空间中。

可以使用带有 modify space sp_sizes 参数的 SQL 管理 API 命令来修改可扩展块所在空间的扩展大小与创建大小。

相关链接

自动空间管理 在第185页
存储池 在第164页
自动空间管理 在第185页
将块标记为可扩展或不可扩展 在第188页
手动扩展空间或可扩展块 在第190页

分区和偏移量

系统管理员可以将物理磁盘分成多个分区，这些分区是一个磁盘的不同部分，并具有不同路径名。虽然在原始磁盘设备上分配块时必须使用整个磁盘分区，但可以使用偏移量将分区或熟文件分成更小的块。

> 提示： 有了对块大小的 4TB 限制，可以通过为每个磁盘驱动器指定单个块来避免对磁盘进行分区。

可以使用偏移量来标识块在磁盘分区、文件或设备上的位置。例如，假设您要创建 1000 KB 的块，并将其分为两块，每块 500 KB。您可以使用 0 KB 偏移量来标记第一个块的开始，然后用 500 KB 偏移量来标记第二个块的开始。

在数据库空间、BLOB 空间或智能大对象空间中创建、添加或删除块时，都可以指定偏移量。

还可能需要指定偏移量来防止数据库服务器重写分区信息。

相关链接

磁盘布局准则 在第165页
在 *UNIX* 上分配原始磁盘空间 在第170页

页

页 是数据库服务器用于在 SinoDB® 中读取和写入的磁盘存储物理单元。

下图说明页的概念，在磁盘片中以加深部分表示。

图 25: 磁盘上的页

在大多数 UNIX™ 平台上，页大小是 2 KB。在大多数 Windows™ 平台上，页大小是 4 KB。页的大小是由硬件决定，无法更改。

块包含一定数量的页，如下图所示。页是完全包含在一个块中。也就是说，页不能跨过块的边界。

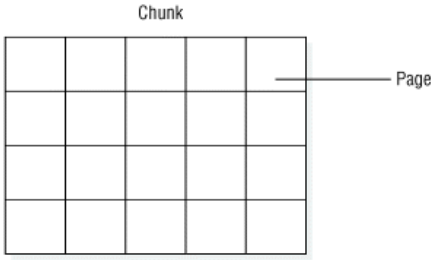

图 26: 块，在逻辑上被分成一系列页

有关数据库服务器如何在页内构造数据的信息，请参阅 《*SinoDB*® 管理员参考》 中有关磁盘结构和存储的章节。

BLOB 页

BLOB 页 是数据库服务器用于在 BLOB 空间中存储简单大对象（TEXT 或 BYTE 数据）的磁盘空间分配单元。

可将 BLOB 页大小大小指定为数据库服务器页大小的倍数。由于数据库服务器将 BLOB 空间分配为连续的空间，所以在与数据大小尽可能接近的 BLOB 页中存储简单大对象将更加有效。下图说明了 BLOB 页概念，以数据页倍数（三倍）表示。

图 27: 磁盘上的 BLOB 页

有关 SinoDB® 如何结构存储在 BLOB 页中的数据的信息，请参阅 《*SinoDB*® 管理员参考》的磁盘结构和存储主题中有关 BLOB 空间 BLOB 页的结构。

与块中的页一样，一定数量的 BLOB 页构成 BLOB 空间中的块，如下图所示。BLOB 页是完全包含在一个块中，且不能跨过块的边界。

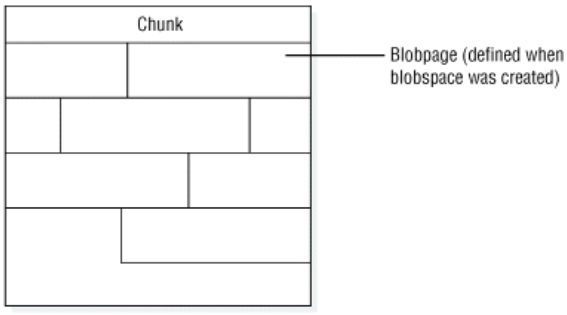

图 28: BLOB 空间中的块，在逻辑上分成一系列 BLOB页

您可以选择将简单大对象数据存储在数据库空间中而不是 BLOB 空间中。然而，对于大于两页的简单大对象，存储在 BLOB 空间可提高性能。存储在数据库空间中的简单大对象可以共享页，但是存储在 BLOB 空间中的简单大对象不能共享页。

有关如何确定 BLOB 页大小的信息，请参阅 确定 *BLOB* 页大小 在第182页。有关 BLOB 空间的描述，请参阅 *BLOB* 空间 在第146页。

智能大对象页

智能大对象页是数据库服务器用于在智能大对象空间中存储智能大对象的页类型。与 BLOB 页不同，智能大对象页是不可配置的。智能大对象页与数据库服务器页的大小相同，在 UNIX™ 上通常是 2 KB，在 Windows™ 上通常是 4 KB。

智能大对象空间中的分配单元是扩展数据块，而 BLOB 空间中的分配单元是 BLOB 页。与块中的页一样，一定数量的智能大对象扩展数据块构成智能大对象空间中的块，如下图所示。扩展数据块是完全包含在一个块中，且不能跨过块的边界。

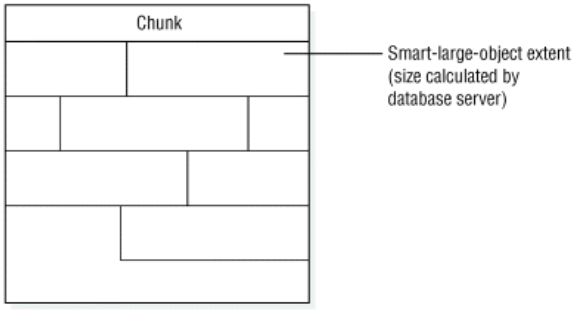

图 29: 智能大对象空间中的块，在逻辑上分成一系列扩展数据块。

智能大对象不能存储在数据库空间或 BLOB 空间中。有关更多信息，请参阅 智能大对象空间 在第146页 和《*SinoDB*® 管理员参考》的磁盘结构和存储章节中有关智能大对象空间结构的部份。

数据库服务器根据一组启发方法（如写操作中的字节数）来计算智能大对象的扩展数据块大小。有关更多信息，请参阅 智能大对象空间的扩展数据块大小 在第149页。

扩展数据块

扩展数据块由存储指定表的数据的连续页的集合组成。

当创建表时，数据库服务器会分配固定数量的空间以包含要存储在该表中的数据。（请参阅 表 在第155页。）当此空间填满时，数据库服务器必须分配额外的存储空间。数据库服务器用于分配初始存储空间和后续存储空间的物理存储单元称为 扩展数据块。

下图说明了扩展数据块的概念。

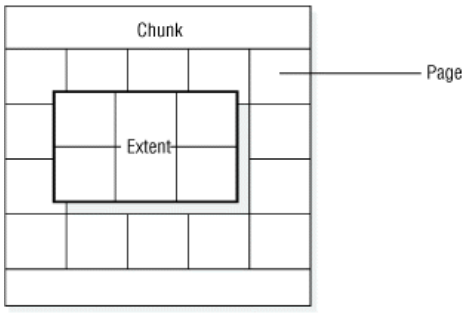

图 30: 原始磁盘空间上六个连续页构成的扩展数据块

每个永久数据库表有两个与之关联的扩展数据块大小。 初始扩展数据块大小是第一次创建表时分配给该表的 KB 数。 下一个扩展数据块大小是当初始扩展数据块（以及任和后续的扩展数据块）填满时分配给该表的 KB 数。对于永久表和用户定义的临时表，下一个扩展数据块大小会在每个扩展数据块之后开始加倍。对于系统创建的临时表，下一个扩展数据块大小会在添加了 4 个扩展数据块之后开始加倍。

创建表时，可以指定初始扩展数据块大小以及表增长时要添加的扩展数据块大小。还可修改数据库空间中表的扩展数据库大小和新的后续扩展数据块的大小。要指定初始扩展数据块大小和下一个扩展数据块大小，请使用 CREATE TABLE 和 ALTER TABLE 语句。有关更多信息，请参阅 《*SinoDB® SQL* 指南*:* 语法》 和《*SinoDB®* 管理员参考》 中有关磁盘结构的部份。

创建带有 CLOB 或 BLOB 数据类型的列的表时，还应为智能大对象空间定义扩展数据外。有关更多信息，请参阅 智能大对象空间的储存特征 在第149页。

下图显示数据库服务器如何为扩展数据块分配六个页：

- 扩展数据块是完全包含在一个块中，且不能跨过块的边界。
- 如果数据库服务器无法找到指定给下一个扩展数据块大小的连续磁盘空间，它将在数据库空间的下一个块中搜索连续空间。

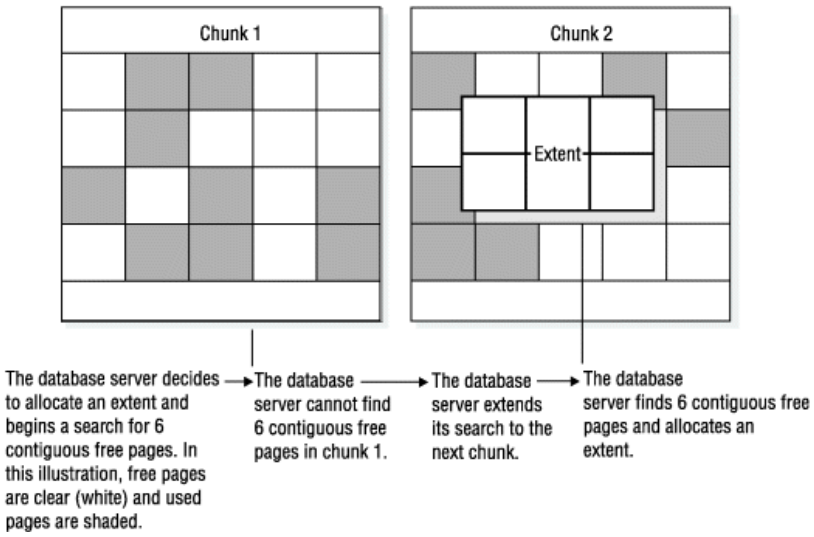

图 31: 扩展数据块的分配过程

相关链接

表 在第155页

《*SinoDB* 管理员参考》*:* 扩展数据块大小加倍

数据库空间

数据库空间 是一种可包含 1 至 32,766 个块的逻辑单元。数据库服务器使用数据库空间来存储数据库和表。将数据库、表、逻辑日志文件和物理日志放置在数据库空间中。

当创建标准或临时数据库空间时，可指定数据库空间的页大小。您不能指定 BLOB 空间、智能大对象空间或外部空间的页大小。如果不指定页大小，那么根数据库空间的大小就是缺省页大小。

当创建标准数据库空间时，可指定数据库空间中表空间 tblspace 的第一个和下一个扩展数据块大小。当必须将表空间 tblspace 扩展数据块放入非主块中，指定扩展数据块大小可减少表空间 tblspace 扩展数据块的数量，以及这些情况发生的频率。

可以为镜像数据库空间中的每一块建立镜像。一旦数据库服务器分配了镜像块，便会立刻将镜像块中的所有空间都标记为已满。

相关链接

块 在第138页

控制简单大对象数据的存储位置

数据库服务器管理员的主要职责之一就是控制数据库服务器存储数据的位置。

通过在最快的磁盘驱动器上存储访问频率很高的表或关键数据库空间（根数据库空间、物理日志和逻辑日志），可以提高性能。通过在不同的物理设备上存储关键数据，可确保当保存非关键数据的磁盘故障时，该故障只会影响该磁盘上数据的可用性。

如下图所示，可以使用 CREATE DATABASE 或 CREATE TABLE 语句的 IN *dbspace* 选项来控制数据库或表的位置。

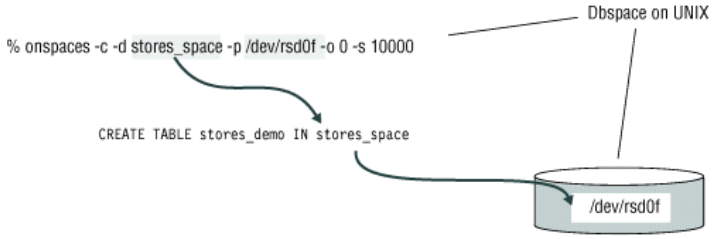

图 32: 使用 CREATE TABLE...IN 语句来控制表的位置

在数据库空间创建数据库或表之前，必须先创建该数据库空间。

如下图所示，数据库空间包含一个或多个块。您可以随时添加更多块。数据库服务器管理员必须监控数据库空间块的填充度并预期是否有必要向数据库空间分配更多的块。若一个数据库空间包含多个块，您不能指定数据所在的块。

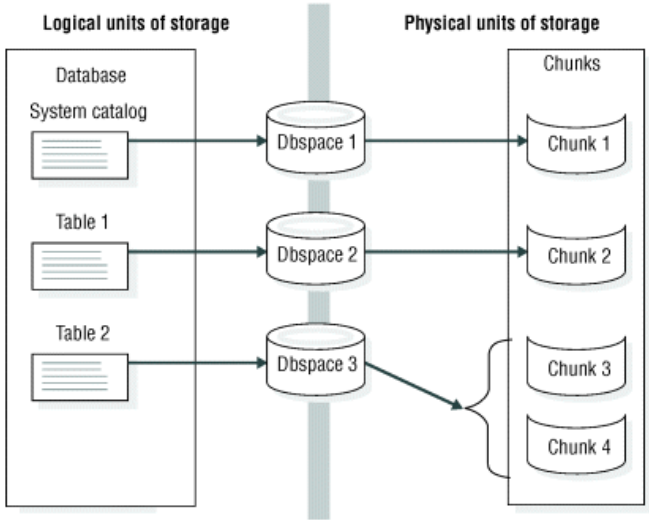

图 33: 连接逻辑和物理存储单元的数据库空间

相关链接

表 在第155页
管理数据库空间 在第173页
监视磁盘使用 在第199页

根数据库空间

根数据库空间 是数据库服务器创建的初始数据库空间。

根数据库空间是特殊的，因为它包含了保留页和内部表，它们将描述和追踪所有物理和逻辑存储单元。（有关这些主题的更多信息，请参阅 表 在第155页和《*SinoDB*® 管理员参考》中的磁盘结构和存储章节。）根数据库空间的初始块及其镜像是唯一在磁盘空间设置期间创建的块。您可以在磁盘空间设置后将其他块添加到根数据库空间。

onconfig 配置文件中的以下磁盘配置参数指的是根数据库空间的第一个（初始）块。

- ROOTPATH
- ROOTOFFSET
- ROOTNAME
- MIRRORPATH
- MIRROROFFSET
- TBLTBLFIRST
- TBLTBLNEXT

根数据库空间也是使用 CREATE DATABASE 语句创建的任何数据库的缺省数据库空间位置。

根数据库空间是数据库服务器为管理所请求的数据而创建的所有临时表的缺省位置。

有关应分配多少空间给根数据库空间的信息，请参阅 根数据库空间的大小 在第163页 。还可以在设置了数据库服务器磁盘空间后将额外块添加到根数据库空间。

临时数据库空间

临时数据库空间 是专门为临时表的存储而保留的数据库空间。您不能为临时数据库空间建立镜像。

数据库服务器不会删除临时数据库空间，除非有明确指示要这么做。临时数据库空间是临时的，仅就是指当数据库服务器异常关闭时不保留任何数据库空间内容。

无论何时设置数据库服务器，所有临时数据库空间都将设置。数据库服务器会清除上次数据库服务器关闭后可能保留的任何表。

数据库服务器不对临时数据库空间执行逻辑或物理日志记录。由于没有对临时数据库空间执行物理日志记录，因此检查点和 I/O 操作发生的较少，从而提高性能。

对于标准数据库空间的临时表，数据库服务器会记录表的创建、扩展数据块的分配和表的删除。相反地，数据库服务器不会记录存储在临时数据库空间的表。在临时数据库空间中禁止使用逻辑日志可减少逻辑恢复期间要前滚的日志记录数，从而提高关键停机时间内的性能。

使用临时数据库空间存储临时表可以减少存储空间备份的大小，因为数据库服务器不会备份临时数据库空间。

数据库服务器使用临时磁盘空间来存储备份时被覆盖的数据的前映像，以及内存中查询处理的溢出。请确保正确地设置了 DBSPACETEMP 环境变量或参数，以便为您的需求指定具有足够空间的数据库空间。如果指定的数据库空间的空间不足，备份将失败，并将使用根数据库空间，或者在根数据库空间填满后备份将失败。

如果您有多个临时数据库空间，并在临时表中执行了 SELECT 语句，那么查询结果将依循环顺序插入。

有关如何创建临时数据库空间的详细说明，请参阅 创建临时数据库空间 在第178页。

重要： 当数据库服务器作为辅助数据库服务器运行时，它需要临时数据库空间来存储由只读查询生成的内部临时表。

BLOB 空间

BLOB 空间 是由一个或多个块组成的逻辑存储单元，仅用于存储 TEXT 和 BYTE 数据。

BLOB 空间会以可能的最有效的方式存储 TEXT 和 BYTE 数据。您可以将与不同表（请参阅 表 在第155页）关联的 TEXT 和 BYTE 列存储在相同的 BLOB 空间中。

数据库服务器将存储在 BLOB 空间中的数据直接写入磁盘。这些数据不会穿过常驻共享内存。如果穿过，数据量可能会占据很多缓冲池页，而导致强制退出其他数据和索引页。基于同样的原因，数据库服务器不会将指定给 BLOB 空间的 TEXT 或 BYTE 对象写入逻辑日志或物理日志。当备份逻辑日志时，数据库服务器会通过将 BLOB 空间对象直接从磁盘写入逻辑日志备份磁带的方式来记录这些 BLOB 空间对象。BLOB 空间对象永远不会穿过逻辑日志文件。

当您创建 BLOB 空间时，您会指定一个或多个块给它。您可以随时添加块。数据库服务器管理员的任务之一就是监视块的填充度并预期是否有必要分配更多块给 BLOB 空间。有关如何监视块的填充度的说明，请参阅 管理磁盘空间 在第168页。有关如何创建 BLOB 空间以及如何为其添加或删除块的说明，请参阅 管理磁盘空间 在第168页。

For information about the structure of a blobspace, see the topics about disk structures and storage in the 《*SinoDB*® 管理员参考》.

有关 BLOB 空间结构的信息，请参阅 《*SinoDB*® 管理员参考》 中有关磁盘结构和存储的主题。

相关链接

块 在第138页

智能大对象空间

智能大对象空间 是由一个或多个块组成的逻辑存储单元，用于存储智能大对象。

智能大对象包含 CLOB（字符大对象）和 BLOB（二进制大对象）数据类型。用户定义的数据类型也可以使用智能大对象空间。有关数据类型的更多信息，请参阅 《*SinoDB*® *SQL* 指南*:* 参考》。

相关链接

块 在第138页

使用智能大对象空间的优势

智能大对象空间优于 BLOB 空间。

以下为使用智能大对象空间的优势：

- 类似于标准 UNIX™ 文件，它们有读取、写入和搜索的属性。

  程序员可以使用类似于 UNIX™ 和 Windows™ 函数的函数来读取、写入和搜索智能大对象。SinoDB® 在 DataBlade® API 和 SinoDB® ESQL/C 编程接口提供该智能大对象接口。
- 它们是可恢复的。

  您可以记录对存储在智能大对象空间中的数据进行的所有写入操作。在事务期间，您可以提交事务或回滚失败的事务。
- 它们遵循事务隔离模式。

  可以锁定不同粒度级别的智能大对象，而锁定持续时间将遵循事务隔离级别的规则。有关锁定与并发的更多信息，请参阅 《*SinoDB®* 性能指南》。
- 表行中的智能大对象无需在一个语句中检索。

  应用程序可以使用 DataBlade® API 或 SinoDB® ESQL/C 编程接口来逐块存储或检索智能大对象。有关 DataBlade® API 函数的更多信息，请参阅 《*SinoDB® DataBlade® API* 函数参考》。有关 SinoDB® ESQL/C 函数的更多信息，请参阅 《*SinoDB® ESQL/C* 程序员指南》。

智能大对象空间和 Enterprise Replication

在为 Enterprise Replication 定义复制服务器前，必须先创建智能大对象空间。Enterprise Replication 会将已复制数据假脱机为智能大对象。

在 CDR_QDATA_SBSPACE 配置参数中指定智能大对象空间名称。Enterprise Replication 使用创建智能大对象空间的缺省日志模式来假脱机行数据。CDR_QDATA_SBSPACE 配置参数接受多个智能大对象空间 ，最多可达 32 个。Enterprise Replication 可以支持日志记录和非日志记录智能大对象空间的组合来存储假脱机的行数据。有关更多信息，请参阅《*SinoDB® Enterprise Replication* 指南》。

在定义复制服务器，可以使 Enterprise Replication 自动从存储池中配置磁盘空间，并在设置相应的配置参数。如果 CDR_QDATA_SBSPACE 或 CDR_DBSPACE 配置参数未设置或设置为空，cdr define server 命令可自动创建必要的磁盘空间并将这些配置参数设为相应值。

元数据，用户数据和保留区域

As with blobspaces and dbspaces, when you create an sbspace, you assign to it one or more chunks. The first chunk of an sbspace always has three areas.与 BLOB 空间和数据库空间一样，当您创建智能大对象空间时，可以为其指定一个或多个块。智能大对象空间的第一个块总是有三个區域。

以下是智能大对象空间的第一个块的三个区域：

元数据区域

元数据标识智能大对象空间和每个存储在其中的智能大对象的关键特性，并使数据库服务器能够操纵和恢复存储其中的智能大对象。

用户数据区域

用户数据是用户应用程序存储在智能大对象空间里的智能大对象数据。块最多可有两个用户数据区域。

保留区域

当需要更多的空间时，数据库服务器会将保留区域中的空间分配给元数据区域或用户数据区域。块最多可有两个保留区域。

有关为智能大对象空间正确分配元数据和用户数据的信息，请参阅 计算智能大对象空间元数据的大小 在第183页和《*SinoDB®* 性能指南》。

当您向智能大对象空间添加块时，可以指定其是否包含元数据区域和用户数据区域，或者是否专门为用户数据保留块。您可以随时添加块。如果您正在更新智能大对象，那么对原始磁盘上的用户数据的 I/O 要比对熟块文件上的用户数据的 I/O 更快。有关如何创建智能大对象空间，及如何为其添加或删除块的指示信息，请参阅 管理磁盘空间 在第168页。

重要： 无论数据库的日志记录设置如何，总是记录智能大对象空间的元数据。

控制智能大对象数据的存储位置

您可在创建表时指定列的数据类型。

对于智能大对象，可以指定 CLOB、BLOB 或用户定义数据类型。如下图所示，您可以使用 CREATE TABLE 语句的 PUT 子句的 IN *sbspace* 选项来控制智能大对象的放置。

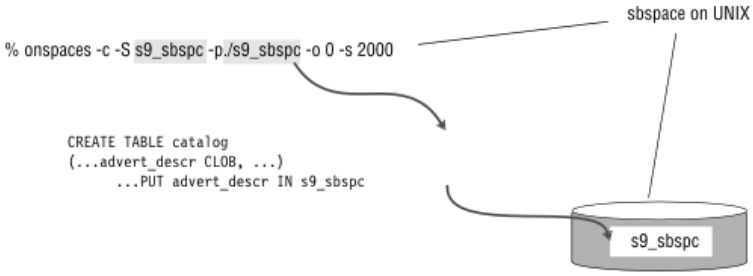

图 34: 控制智能大对象的放置

在 PUT 子句中指定智能大对象空间之前，必须先创建该智能大对象空间。有关如何使用 onspaces -c -S 命令来创建智能大对象空间的更多信息，请参阅 向数据库空间和 *BLOB* 空间添加块 在第178页。有关如何在 PUT 子句中指定智能大对象特征的更多信息，请参阅《*SinoDB® SQL* 指南: 语法》中的 CREATE TABLE 语句。

如果没有指定 PUT 子句，那么数据库服务器将在 SBSPACENAME 配置参数指定的缺省智能大对象空间中储存智能大对象。有关 SBSPACENAME 的更多信息，请参阅 《*SinoDB®* 管理员参考》 中有关配置参数的主题。

正如下图所示，智能大对象空间可包含一个或多个块。当智能大对象空间包含多个块，就不能指定数据所在的块。

您可以随时添加更多的个块。监控智能大对象空间块的填充度并预期是否有必要向智能大对象空间分配更多的块，此为数据库服务器管理员的高优先级任务。有关监控智能大对象空间的更多信息，请参阅 《*SinoDB®* 性能指南》。

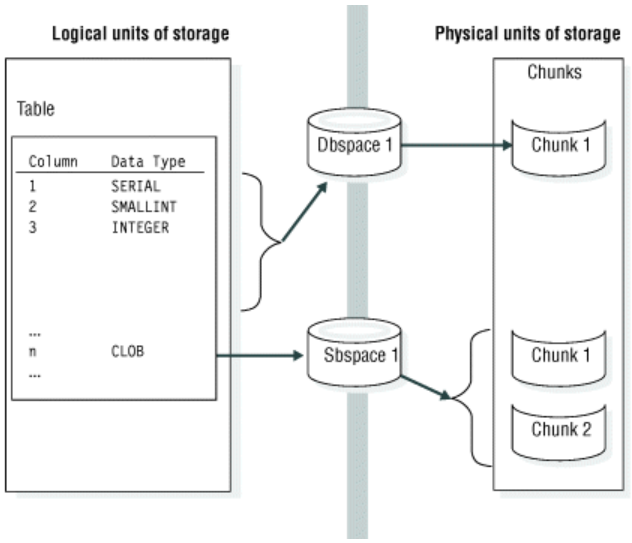

图 35: 连接逻辑和物理存储单元的智能大对象空间

数据库服务器使用智能大对象空间存储包含智能大对象的表列。数据库服务器使用数据空间来存储其余的表列。

您可以为智能大对象空间建立镜像以便在介质故障时加速恢复。有关更多信息，请参阅 镜像 在第274页。

有关使用 onspaces 来执行以下任务的信息，请参阅 管理磁盘空间 在第168页。

- 创建智能大对象空间
- 向智能大对象空间添加块
- 更改智能大对象的存储特征

- 创建临时智能大对象空间
- 删除智能大对象空间

智能大对象空间的储存特征

作为数据库服务器管理员，您可以使用这些储存特征的系统缺省值，或者可以在用 onspaces -c 创建智能大对象空间时使用 -Df 标记来指定这些特征。之后，可以使用 onspaces -ch 选项来改变这些智能大对象空间特征。

The administrator or programmer can override these default values for storage characteristics and attributes for individual tables.

管理员或程序员可以覆盖储这些存特征的缺省值和单个表的属性。

智能大对象空间的扩展数据块大小

智能大对象空间中的扩展数据块由储存智能大对象数据的连续页的集合组成。

智能大对象空间内的分配单位是扩展数据块。数据库服务器根据一组启发式搜索（如写操作中的字节数）计算智能大对象的扩展数据块大小。例如，如果操作要求写入 30KB，数据库服务器会试图分配一个大小为 30KB 的扩展数据块。

> 重要： 对于大部分应用程序，您必须使用数据库服务器计算为扩展数据块的大小计算的值。

如果您知道智能大对象的大小，那么可以使用下列函数之一来设置扩展数据块大小。数据库服务器将整个智能大对象分配为一个扩展数据块（如果在该块有该大小的扩展数据块可用）：

- DataBlade® API mi_lo_specset_estbytes() 函数。

  有关智能大对象的 DataBlade® API 函数的更多信息，请参阅《*SinoDB® DataBlade® API* 函数参考》。
- SinoDB® ESQL/C ifx_lo_specset_estbytes 函数。

  有关智能大对象的 SinoDB® ESQL/C 函数的更多信息，请参阅《*SinoDB® ESQL/C* 程序员指南》。

有关调整扩展数据块大小的信息，请参阅《*SinoDB®* 性能指南》中有关配置对 I/O 利用率的影响的章节中的智能大对象。

智能大对象平均大小

智能大对象的长度经常会变化。您可以提供智能大对象的平均大小来计算智能大对象空间的空间大小。

可以用 onspaces -c -Df 选项的 AVG_LO_SIZE 标记来指定平均大小。

要指定元数据区域的大小和位置，请在 onspaces 命令中指定 -Ms 和 -Mo 标记。如果您不使用 -Ms 标记，那么数据库服务器会使用 AVG_LO_SIZE 的值来估算分配给元数据区域的空间大小。有关更多信息，请参阅 计算智能大对象空间元数据的大小 在第183页。

缓冲模式

当您创建智能大对象空间时，缺省的缓冲模式打开，表示在共享内存的常驻部分使用缓冲池。

作为数据库管理员，您可以用 onspaces -c -Df 选项的 BUFFERING 标记来指定缓冲模式。缺省值为“buffering=ON”，表示使用缓冲池。如果关闭缓冲，数据库服务器将在共享内存的虚拟部分中使用专用缓冲区。

> 重要： 通常，如果对智能大对象的读写操作小于 8KB，那么创建智能大对象空间时不要指定缓冲模式。如果正在读写短的数据块（如 2KB 或 4KB），那么保留缺省值“buffering=ON”以得到更好的性能。

有关何时使用专用缓冲区的信息，请参阅《*SinoDB®* 性能指南》中有关配置对 I/O 利用率影响的主题中有关轻量级 I/O 操作部分。

上次访问时间
创建智能大对象空间时，可以指定是否必须保存上次用 onspaces -c -Df 选项的 ACCESSTIME 标记读取或更新的智能大对象的时间。

系统缺省值为 “ACCESSTIME=OFF”。数据库服务器会将上次访问时间保留在元数据区域。

有关程序员如何使用上次访问时间的更多信息，请参阅《*SinoDB® DataBlade® API* 程序员指南》和《*SinoDB® ESQL/C* 程序员指南》。

锁定模式
创建智能大对象空间时，可以指定数据库服务器否使用 onspaces -c -Df 选项的 LOCK_MODE 标记来锁定整个智能大对象还是智能大对象中一定范围的字节。

缺省值为“LOCK_MODE=BLOB”，表示锁定整个智能大对象。有关更多信息，请参阅《*SinoDB®* 性能指南》的锁定章节。

日志记录
创建智能大对象空间时，可以指定是否开启智能大对象的日志记录。

缺省为无日志记录。有关更多信息，请参阅 记录智能大对象空间和智能大对象 在第248页。

> 重要： 当您使用日志记录数据库时，请开启智能大对象空间的日志记录。如果发生故障需要日志恢复，您可以让智能大对象与数据库的其它部分保持一致。

可以使用 onspaces -c -Df 选项的 LOGGING 标记来指定日志状态。缺省值为 "LOGGING=off"。可以用 onspaces -c -Df 选项更改日志状态。可以用 SQL 语句 CREATE TABLE 或 ALTER TABLE 中的 PUT 子句来覆盖日志状态。有关这些 SQL 语句的更多信息，请参阅《*SinoDB® SQL* 指南: 语法》。

程序员可以用 DataBlade® API 和 SinoDB® ESQL/C 提供的函数来覆盖日志状态。有关智能大对象的 DataBlade® API 函数 ，请参阅《*SinoDB® DataBlade® API* 函数参考》。有关智能大对象的 SinoDB® ESQL/C 函数，请参阅《*SinoDB® ESQL/C* 程序员指南》。

当智能大对象空间的日志记录为开启时，智能大对象会穿过共享内存的常驻部分。尽管应用程序可以逐块检索智能大对象，但仍必须考虑可能穿过缓冲池和逻辑日志缓冲区的更大数据大小。有关更多信息，请参阅 访问智能大对象 在第127页。

继承智能大对象空间特征的继承级别

智能大对象空间特征的四种继承级别为系统，智能大对象空间，列和智能大对象。您可以使用智能大对象空间属性的系统缺省值，或覆盖特定的智能大对象空间、表中的列或智能大对象的系统缺省值。

下图显示了智能大对象的存储特征层次结构。

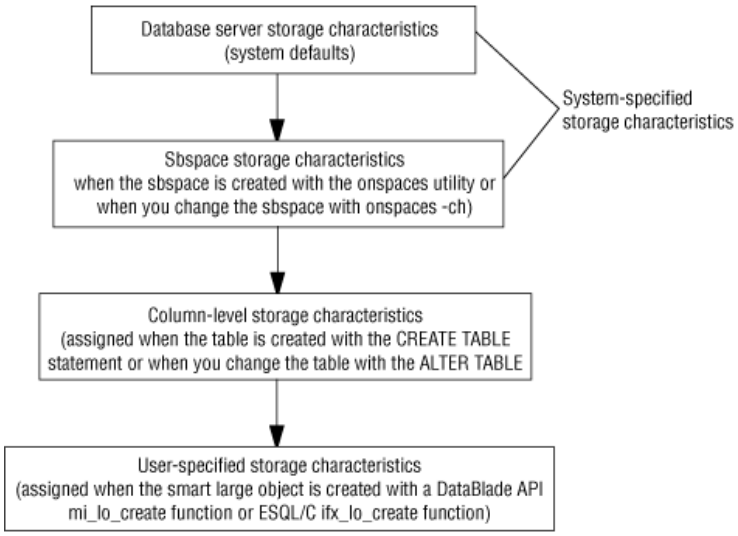

图 36: 存储特征层次结构

该图显示您可用以下方法覆盖系统缺省值：

- 使用 onspaces -c -S 命令的 -Df 标记来覆盖特定智能大对象空间的系统缺省值。

  可稍后使用 onspaces -ch 选项更改智能大对象空间的这些智能大对象空间的属性。有关 -Df 标记的有效范围的更多信息，请参阅《*SinoDB*® 管理员参考》中的 onspaces 主题。
- 在 CREATE TABLE 或 ALTER TABLE 语句中的 PUT 子句指定这些属性，可覆盖特定列的系统缺省值。

  有关这些 SQL 语句的更多信息，请参阅《*SinoDB*® *SQL* 指南: 语法》。
- 程序员可以用 DataBlade® API 和 SinoDB® ESQL/C 编程接口提供的函数来覆盖特定智能大对象的智能大对象空间属性的缺省值。

有关智能大对象空间的更多信息

有关使用和管理智能大对象空间的各种任务。

下表列出了关于使用和管理智能大对象空间的任务的信息来源。

表 21: 查找智能大对象空间任务的信息

任务	参阅
设置智能大对象的内存配置参数	管理共享内存 在第128页
了解智能大对象页	智能大对象页 在第142页
指定智能大对象空间的 I/O 特征	智能大对象空间的储存特征 在第149页中的 onspaces 选项
为智能大对象空间分配空间	创建智能大对象空间 在第182页
向智能大对象空间 添加块	向智能大对象空间添加块 在第183页
定义或改变智能大对象的存储特征	更改智能大对象的存储特征 在第184页 《*SinoDB*® *SQL* 指南: 语法》中的 CREATE TABLE 或 ALTER TABLE 语句的 PUT 子句
监视智能大对象空间	监视智能大对象空间 在第204页 《*SinoDB*® 性能指南》中有关表性能注意事项的主题
为智能大对象空间设置日志记录	记录智能大对象空间和智能大对象 在第248页
备份智能大对象空间	备份智能大对象空间 在第253页
检查智能大对象空间的一致性	验证元数据 在第283页
了解智能大对象空间的结构	《*SinoDB*® 管理员参考》中有关磁盘结构的主题
使用智能大对象空间的 onspce	《*SinoDB*® 管理员参考》中有关实用程序的主题
使用 CLOB 或 BLOB 数据类型创建表	《*SinoDB*® *SQL* 指南: 语法》
在应用程序中访问智能大对象	《*SinoDB*® *DataBlade*® *API* 程序员指南》 、《*SinoDB*® *ESQL/C* 程序员指南》
计算元数据区域大小 改进元数据的 I/O 改变存储特征	《*SinoDB*® 性能指南》中有关表性能的主题
了解智能大对象锁定	《*SinoDB*® 性能指南》中有关锁定的主题
为临时智能大对象配置智能大对象空间	《*SinoDB*® 性能指南》中有关配置对 I/O 活动的影响的主题

临时智能大对象空间

使用 临时智能大对象空间 可在不用元数据记录和用户数据记录的情况下存储临时智能大对象。

如果您将临时智能大对象存储在标准智能大对象空间中，那么将记录元数据。临时智能大对象空间类似于临时数据库空间。使用带有 -t 选项的 onspaces -c -S 命令来创建临时智能大对象空间。有关更多信息，请参阅 创建临时智能大对象空间 在第184页 。

您可以将临时大对象存储在标准智能大对象空间或临时智能大对象空间。

- 如果在 SBSPACETEMP 参数中指定临时智能大对象空间，那么可在该临时智能大对象空间存储临时智能大对象。
- 如果在 SBSPACENAME 参数中指定标准智能大对象空间，那么可在该标准大对象空间存储临时和永久智能大对象。
- 如果在 CREATE TEMP TABLE 语句中指定临时智能大对象空间名称，那么可在该临时智能大对象空间存储临时智能大对象。
- 如果在 CREATE TABLE 参数中指定永久智能大对象空间名称，那么可在该永久智能大对象空间存储临时智能大对象。
- 如果省略了 SBSPACETEMP 和 SBSPACENAME 参数，并创建智能大对象，那么可能会显示错误信息 -12053 。
- 如果在 SBSPACENAME 参数中指定临时智能大对象空间，那么就不能将永久智能大对象存储在该智能大对象空间中，但可将临时智能大对象存储在该智能大对象空间中。

比较临时和标准智能大对象空间

下表比较了标准和临时智能大对象空间。

表 22: 临时和标准智能大对象空间

特征	标准智能大对象空间	临时智能大对象空间
存储智能大对象	是	否
存储临时智能大对象	是	是
记录元数据	总是记录元数据	不记录元数据
记录用户数据	如果LOGGING=ON，对于临时智能大对象，不记录用户数据；对于永久智能大对象，会记录用户数据	不记录用户数据 会记录空间的创建和删除，以及块的添加
快速恢复	是	No (the sbspace is emptied when the database server restarts) To set up shared memory without cleaning up temporary smart large objects, specify oninit -p. If you keep the temporary large objects, their state is indeterminate.否（数据库服务器重新启动时会清空智能大对象空间）。要设置共享内存而无需清空临时智能大对象，请指定 onInit-p。如果保留临时大对象，那么它们的状态是不确定的。
备份和还原	是	否
添加和删除块	是	是
配置参数	SBSPACENAME	SBSPACETEMP

临时智能大对象
使用 临时智能大对象 来存储不需要从备份还原或在快速恢复中记录重放文本或映像数据（CLOB 或 BLOB）。

临时智能大对象将在用户会话中持续，并且其更新速度比智能大对象块。

除了在 ifx_lo_specset_flags 或 mi_lo_specset_flags 函数中设置 LO_CREATE_TEMP 标志外，创建临时智能大对象的方式与创建永久智能大对象的方式相同。使用 mi_lo_copy 或 ifx_lo_copy 从临时智能大对象来创建永久智能大对象。有关创建临时智能大对象的详细信息，请参阅《*SinoDB® DataBlade® API* 程序员指南》 。

> 重要： 只在临时表中存储指向临时大对象的指针。如果将它们存储在标准表中并重新启动数据库服务器，那么将会导致大对象不存在的错误。

下表比较了标准和临时智能大对象。

表 23: 临时和标准智能大对象

特征	智能大对象	临时智能大对象
创建标志	LO_CREATE_LOG 或 LO_CREATE_NOLOG	LO_CREATE_TEMP
回滚	是	否
日志记录	是，如果开启	否
持续时间	永久（直到用户将其删除）	在用户会话或事务结束时删除
存储的表类型	永久或临时表	临时表

物理日志空间

物理日志空间是由一个块组成的逻辑存储单元，用于存储物理日志。当物理日志存于 物理日志空间时，数据库服务器按需增加物理日志的大小以提高性能。

如果在安装过程中没有创建服务器，那么系统将在根数据库空间内创建物理日志。然而，您可以创建物理日志空间以将物理日志移动到不同的数据库空间，从而避免物理日志占满根数据库空间。为获得最佳性能，请在与根数据库空间或逻辑日志位置不同的磁盘上创建物理日志空间。如果在安装过程中创建服务器，那么将会自动创建一个默认大小取决于 AUTO_TUNE_SERVER_SIZE 配置参数的值的物理日志空间。

默认情况下，分配给物理日志空间的块是可扩展的，因此，初始块的大小可以很小。当物理日志需要更多空间的时候，数据库服务器会自动扩展该块。

物理日志空间具有以下限制：

- 数据库服务器实例只能有一个物理日志空间。
- 物理日志空间只可包含一个物理日志。
- 物理日志空间只可有一个块。
- 该块必须与根数据库空间具有相同的页大小。

相关链接

管理物理日志空间 在第185页
物理日志的大小和位置 在第265页
《*SinoDB* 管理员参考》: *AUTO_TUNE_SERVER_SIZE* 配置参数

外部空间

外部空间 是与表示外部数据位置的随机字符串关联的逻辑名。外部空间所引用的资源取决于访问其内容的用户定义的访问方法。

例如，数据库用户可能需要访问以专有格式编码的二进制文件。首先，开发人员创建 访问方式，它是一组访问数据的例程。这些例程负责数据库服务器和外部文件之间的所有交互。然后，DBA 将添加一个外部空间，该外部空间会将该文件作为数据库的目标。DBA 在该外部空间中创建了表之后，用户可通过 SQL 语句访问该专有文件中的数据。要找到这些文件，请使用外部空间信息。

外部空间不必是文件名。例如，它可以是网络位置。访问数据的例程可以用任何方式使用与外部空间关联的字符串中找到的信息。

有关用户定义的访问方法的更多信息，请参阅《*SinoDB*® 虚表接口程序员指南》 。有关创建函数和主要访问方法的更多信息，请参阅《*SinoDB*® *SQL* 指南: 语法》 。

数据库

数据库是包含表和索引的逻辑存储单元。每个数据库还包含跟踪有关数据库中许多元素（包括表、索引、SQL 例程和完整性约束）的信息的系统目录。

数据库是存储于 CREATE DATABASE 语句中的 IN 子句指定的数据库空间中。当未在 CREATE DATABASE 语句中明确指定数据库空间时，除非启用自动选位，否则数据库将存储于根数据库空间。可以通过将 AUTOLOCATE 配置参数或会话环境变量设为正整数来启用自动选位。数据库服务器将选择创建新数据库和新表的数据库空间，而无需指定存储位置。在数据库服务器所选的数据库空间中，表将通过循环方式自动分段。

当在 CREATE DATABASE 语句中指定数据库空间时，数据库空间位置为以下表的位置：

- 数据库系统目录表
- 属于该数据库的任何表

下图显示了 stores_demo 数据库中包含的表。

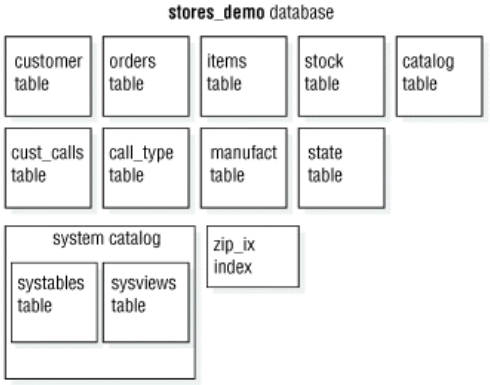

图 37: stores_demo 数据库

数据库的大小限制与其所在数据库空间中的位置有关。确定数据库中所有的表都是在特定物理设备上创建的，只分配一个块给该设备，并且创建仅包含该块的数据库空间。将数据库放置在该数据库空间中。当将数据库放置在分配给特定物理设备的块中时，该数据库的大小将受限于该块的大小。

相关链接

管理自动定位和分段化 在第180页
显示数据库 在第199页

表

在关系数据库系统中， 表 是一行列标题，以及零行或多行数据值。列标题行标识了一个或多个列，以及每列的数据类型。

当创建表时，数据库服务器会为页块（称为扩展数据块）中的表分配磁盘空间。您可以指定第一个扩展数据块和任何后续扩展数据块的大小。

可以在创建表（使用 CREATE TABLE 语句的 IN *dbspace* 子句）时，通过指名数据库空间将表放置到特定的数据库空间中。当没有指定数据库空间时，数据库服务器会将表放置在数据库所在的数据库空间中。通过指定的分段分布方案，可在多个数据库空间或一个数据库空间内进行表分段。然而，如果将 AUTOLOCATE 配置参数设置为正整数，那么数据库服务器将在最适于表的数据库空间中以循环方式自动分段新表。

表或表分段完全位于创建它的数据库空间。数据库服务器管理员可使用此事实，藉由将表放置在数据库空间中然后当数据库空间变满时拒绝向其添加块的方式来限制表的增长。

由扩展数据块组成的表可以跨多个块，如下图所示。

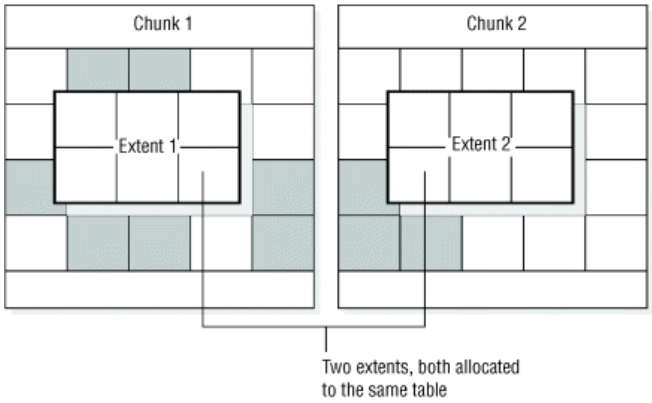

图 38: 跨多个块的表

简单大对象位于 BLOB 页中，这些 BLOB 页可位于带有表的数据页的数据库空间中或位于单独的 BLOB 空间中。

相关链接

扩展数据块 在第142页
表分段存储和数据存储 在第161页
磁盘布局准则 在第165页
控制简单大对象数据的存储位置 在第144页

损坏的表

有许多方法可以损坏表。

以下项目会损坏表：

- 不正确的缓冲区清空
- 用户错误
- 在块的顶部挂载文件系统或另一个块
- 当变更范围超出您的要求时进行删除或更新。

已损坏的索引会导致表看起来像是已损坏，即使该表并没有损坏。

oncheck 命令无法修复大多数已损坏的表。如果某页受损， oncheck 可检测并尝试修复该页，但不能改正该页中的数据。

SinoDB® 的表类型

您可以在 SinoDB® 上的日志记录数据库中创建日志记录或非日志记录表。两种表类型为 STANDARD（日志记录表）和 RAW（非日志记录表）。可以创建 STANDARD 表或 RAW 表，并且可以将表从一个类型更改为另一个类型。

在非日志记录数据库中， STANDARD 表和 RAW 表都是非日志记录的。在非日志数据库中，STANDARD 表和 RAW 表之间的唯一区别为 RAW 表不支持主键约束、唯一约束、引用约束或回滚。然而，可以对这些表进行索引和更新。

下表列出了 SinoDB® 中表类型的属性。每个表类型在 systables 的 flags 列的标志值都是十六进制值。

表 24: SinoDB® 的表类型

特征	STANDARD	RAW	TEMP
永久	是	是	否
记录	是	否	是
索引	是	是	是
约束	是	无引用约束或唯一约束 允许 NULL 和 NOT NULL 约束	是
回滚	是	否	是
可恢复	是	是，如果未更新	否
可复原	是	是，如果未更新	否
可载入	是	是	是
Enterprise Replication 服务器	是	否	否
高可用性集群中的主服务器	是	是，不能更改日志记录模式	是
高可用性集群中的辅助服务器	是	是，但任何操作都不可访问	是
标志值	无	0x10	无

标准永久表

STANDARD 表与数据库服务器创建的记录的数据库中的表相同。

STANDARD 表不使用轻量级追加。所有操作都会一条条记录下来，因此 STANDARD 表可以恢复和回滚。可备份和复原 STANDARD 表。日志记录使您能够在执行热复原或时间点复原时应用自上次物理备份起的更新。在 STANDARD 表上允许使用 Enterprise Replication。

STANDARD 表是日志记录和非日志记录数据库上的缺省类型。STANDARD 表如果存储在日志记录数据库中，将被记录；如果存储在非日志记录数据库中，将不被记录。

RAW 表

RAW 表是非日志记录的永久表，类似于非日志记录数据库中的表。

以下的语句创建一个名为 r_tab 的 RAW 表：

```
CREATE RAW TABLE IF NOT EXISTS r_tab1 (col1 INT, col2 CHAR(40) NOT NULL);
```

支持但不记录 RAW 表中行内的更新、插入和删除操作。可以在 RAW 表中定义索引，但不能在 RAW 表中定义唯一约束，主键约束和引用约束。

无论是存储在日志记录数据库还是非日志记录数据库中，RAW 表都具有相同的属性。如果更新了 RAW 表，除非在更新后执行 0 级备份，否则不能可靠地复原数据。如果自备份后没有更新过表，那么可以从上次的物理备份中复原 RAW 表，但仅备份逻辑日志并不足以复原 RAW 表。快速恢复可回滚 STANDARD 表中未完成的事务，但不能回滚 RAW 表中未完成的事务。有关创建和更改 RAW 表的信息，请参阅 《*SinoDB® SQL* 指南: 语法》 。

RAW 表适用于初始装载和数据验证。可以使用任何装载实用程序（包括 dbexport 和 DB-Access 的 LOAD 语句）来装载 RAW 表。如果在装载 RAW 表时发生错误或故障，所得数据是发生故障时磁盘上的任何数据。

限制: 不要在事务中使用 RAW 表。装载数据之后，使用 ALTER TABLE 语句把表类型改为 STANDARD 类型并执行 0 级备份，然后才在事务中使用该表。

限制: 不要在 RAW 表和 TEMP 表上使用 Enterprise Replication 。

在高可用性集群环境中使用 RAW 表时，有一些限制。因为不记录对 RAW 表进行的修改，并且辅助服务器（包括 HDR、RSS 和 SDS）使用日志记录与主服务器保持同步，因此将限制对 RAW 表执行某些操作：

- 在主服务器上，可以创建，删除和访问 RAW 表。然而，不允许将表模式从非记录更改为记录，或从记录更改为非记录。在高可用性集群环境中更改表的日志记录模式会生成错误 -19845。
- 在辅助服务器（HDR、SDS 或 RSS）中，任何操作都不能访问 RAW 表。尝试从 SQL 访问 RAW 表会生成错误 -19846。

Temp 表

Temp 表是临时的记录表，它们将在用户会话关闭时、数据库服务器关闭，或者故障后重新启动时删除。

Temp 表支持索引、约束和回滚。您无法恢复、备份或复原 Temp 表。Temp 表支持批量操作，例如轻量级追加，可以快速将行添加到每个表分段的末尾。有关轻量级追加的更多信息，请参阅 《*SinoDB®* 性能指南》 。

有关更多信息，请参阅 临时表 在第158页 。

表类型的属性

这些主题说明装载表，快速恢复，表类型的备份和复原。

向表装载数据

可以使用 CREATE TABLE RAW 语句来创建 RAW 表，或在装载表之前使用 ALTER TABLE 语句把 STANDARD 表更改为 RAW 表。在装载表之后，请对该表运行 UPDATE STATISTICS。

SinoDB® 创建默认使用日志记录的 STANDARD 表。数据仓库应用程序可以有需要花费很长时间装载的巨型表。装载非日志记录表比装载日志记录表要快。

有关如何提高装载非常大的表的性能的更多信息，请参阅《*SinoDB®* 性能指南》。有关使用 ALTER TABLE 把日志记录表更改为非日志记录表的更多信息，请参阅《*SinoDB® SQL* 指南: 语法》。

表类型的快速恢复

SinoDB® 中可用表类型的快速恢复方案。

下表显示了表类型的快速恢复方案。

表 25: 表类型的快速恢复

表类型	快速恢复行为
Standard	快速恢复成功。所有提交的日志记录将前滚，而所有未完成的事务将回滚。
RAW	如果自上次 RAW 表更改后已完成检查点，那么所有数据都可恢复。 上一个检查点之后发生的插入、更新和删除操作都将丢失。

表类型	快速恢复行为
	在 RAW 表中未完成的事务将不回滚。

备份和复原 RAW 表
SinoDB® 中可用的表类型的备份方案。

下表显示了表类型的备份方案。

表 26: 在 SinoDB® 上备份表

表类型	是否允许备份？
Standard	是。
Temp	否。
RAW	是。如果更新了 RAW 表，必须对其进行备份，这样就可以复原其中的所有数据。仅备份逻辑日志是不够的。

重要: 装入 RAW 表或将 RAW 表更改为 STANDARD 类型后，必须执行 0 级备份。

下表显示了这些表类型的复原方案。

表 27: 在 SinoDB® 上复原表

表类型	是否允许复原？
Standard	是。热复原，冷复原，以及时间点复原都可。
Temp	否。
RAW	当复原 RAW 表时，它仅包含上次备份时位于磁盘上的数据。由于不记录 RAW 表，因此自上次备份以来发生的任何更改都不会复原。

临时表

数据库服务器必须为特定类型的临时表提供磁盘空间。

以下临时表需要磁盘空间：

- 使用 SQL 语句（如 CREATE TEMP TABLE. . . ）创建的临时表。
- 数据库服务器处理查询时创建的临时表

请确保您的数据库服务器为用户创建的和数据库服务器创建的临时表配置了足够的临时空间。数据库服务器的一些使用可能需要与永久存储空间一样多的临时存储空间，或更多。

默认情况下，数据库服务器在根数据库空间中存储临时表。如果不想在根数据库空间中存储临时表，请使用 DBSPACETEMP 环境变量或 DBSPACETEMP 配置参数来指定临时表的数据库空间列表。

创建的临时表
可以用一些 SQL 语句来创建临时表。

可以用以下 SQL 语句来创建临时表：

- CREATE TABLE 语句的 TEMP TABLE 选项
- SELECT 语句的 INTO TEMP 子句，例如，SELECT * FROM customer INTO TEMP *cust_temp*

只有创建临时表的会话才能使用该表。当会话退出时，该表将自动删除。

当您创建临时表时，数据库服务器将使用以下条件：

- 如果用于填充 TEMP 表的查询没有生成任何行，那么数据库服务器将创建一个空的且未分段的表。
- 如果该查询生成的行未超过 8KB ，那么临时表将只位于一个数据库空间中。

- 如果生成的行超过 8KB，数据库服务器将创建多个分段并使用循环分段方式来进行填充，除非您指定了表的分段方法和存储位置。

如果使用 CREATE TEMP 和 SELECT ... INTO TEMP SQL 语句，并且设置了 DBSPACETEMP：

- 列表中的日志记录数据库空间将用于创建指定或暗示 WITH LOG 子句的表。
- 列表中的非日志记录临时数据库空间将用于创建指定 WITH NO LOG 子句的表。

当使用 CREATE TEMP 和 SELECT...INTO TEMP SQL 语句，且未设置 DBSPACETEMP 或未包含正确的数据库空间类型时，SinoDB® 将使用数据库的数据库空间来存储临时表。有关更多信息，请参阅《*SinoDB® SQL* 指南*:* 语法》。

存储用户创建的临时表的位置
如果应用程序允许您指定临时表的位置，您可以指定日志记录空间或您专门为临时表创建的非日志记录空间。

有关创建临时数据库空间的信息，请参阅《*SinoDB®* 管理员参考》 中的 onspaces 主题。

如果您没有指定临时表的位置，那么数据库服务器会将临时表存储在指定为 DBSPACETEMP 配置参数或环境参数的自变量的其中一个空间中。数据库服务器会记住上一个用于存储临时表的数据库空间的名称。当数据库服务器接收到另一个临时存储空间需求时，它将使用下一个可用数据库空间均匀地将 I/O 分布在临时存储空间中。

有关您未列出作为 DBSPACETEMP 的自变量的任何空间时数据库存储临时表位置的信息，请参阅《*SinoDB®* 管理员参考》 中 DBSPACETEMP 一节。

当您使用应用程序创建临时表，您可以使用该临时表直到应用程序退出，或执行以下操作之一为止：

- 关闭创建该表的数据库，并开启其他数据库服务器中的数据库
- 关闭创建该表的数据库
- 显示地删除该临时表

数据库服务器创建的临时表
数据库服务器有时会在对数据库运行查询或备份该数据库时创建临时表。

数据库服务器可能在以下任一情况下创建临时表：

- 包含 GROUP BY 或 ORDER BY 子句的语句
- 使用带有 UNIQUE 或者 DISTINCT 关键字的聚集函数的语句
- 使用自动索引或者散列连接的 SELECT 语句
- 复杂的 CREATE VIEW 语句
- 创建滚动游标的 DECLARE 语句
- 包含相关子查询的语句
- 包含 IN 或 ANY 子句的子查询的语句
- CREATE INDEX 语句

当启动表创建的进程结束后，数据库服务器会删除其创建的临时表。

如果数据库服务器没有删除临时表就关闭，那么数据库服务器在下次启动时将删除临时表。想要启动数据库服务器时不删除临时表，请运行带有 -p 选项的 oninit 命令。

应用程序和分析工具可以定义其中派生表包含与基本表连接的多个视图的查询，可能包含数百列。数据库服务器尝试将视图或派生表摺叠放入主查询。无法摺叠的任何视图或派生表都会具体化到临时表中。临时表会排除主查询中未引用的所有列。临时表仅使用在 Projection 子句引用的列和在父查询的其他子句（包含 WHERE、HAVING、GROUP BY 和 ON 子句）中引用的列来创建。

通过从系统生成的临时表中排除主查询中未引用的列，这一简化模式可节省存储资源、避免未使用列的不必要数据 I/O，从而提升查询性能。

然而，在嵌套查询中，只会检查父查询中来自视图和派生表的投影列，但不会检查高于直接父查询的级别的投影列。

重要: 除了临时表，数据库服务器使用临时磁盘空间来存储发生备份时覆盖的数据记录的前映像，以及内存中发生的查询处理的溢出。请确保正确设置了 DBSPACETEMP 环境变量或 DBSPACETEMP 配置参数，以指定具有满足您需要的足够空间的数据库空间。如果在指定的数据库空间中空间不足，备份将失败，并将使用根数据空间，或者在填满根数据空间之后，备份将失败。

存储数据库服务器创建的临时表的位置
当数据库服务器创建临时表时，它会将该临时表存储在您在 DBSPACETEMP 配置参数或 DBSPACETEMP 环境变量中指定的数据库空间之一。环境变量取代了配置参数。

当未在 DBSPACETEMP 中指定任何临时数据库空间时，或者指定的临时数据库空间的空间不足时，数据库服务器将根据以下规则在标准数据库空间中创建该表：

- 如果使用 CREATE TEMP TABLE 来创建临时表，那么数据库服务器会将该表存储在包含该表所属数据库的数据库空间中。
- 如果使用 SELECT 语句的 INTO TEMP 选项来创建临时表，那么数据库服务器会将该表存储在根数据空间中。

有关更多信息，请参阅创建临时数据库空间 在第178页。

表空间

数据库服务器管理员有时必须追踪特定表的磁盘使用情况。表空间包含分配到给定表或表分段（如果表已分段）的所有磁盘空间。单独表空间包含已分配给关联索引的磁盘空间。

例如，表空间不对应于块的任何特定部分，或者任何特定块。组成表空间的索引和数据可能会分散在所有块中。然而，表空间表示用于特定表的块的空间的方便记帐实体。（请参阅表 在第155页。）

表中的最大表空间数

您可以在表中指定最多达 2**20 （或 1,048,576）的表空间数。

表和索引表空间

表表空间和索引表空间包含特定页类型。

表表空间包含以下页类型：

- 分配给数据的页
- 分配给索引的页
- 用于在数据库空间中存储 TEXT 或 BYTE 数据的页（不是用于在 BLOB 空间中存储 TEXT 或 BYTE 数据的页）
- 追踪页在表扩展数据块内使用的位图页

索引表空间包含以下页类型：

- 分配给索引的页
- 追踪页在索引扩展数据块内使用的位图页

下表说明用于构成部分 stores_demo 数据库的三个表的表空间。每个表空间上只有一个表（或者表分段）。BLOB 页代表存储在数据库空间中的 TEXT 或者 BYTE 数据。

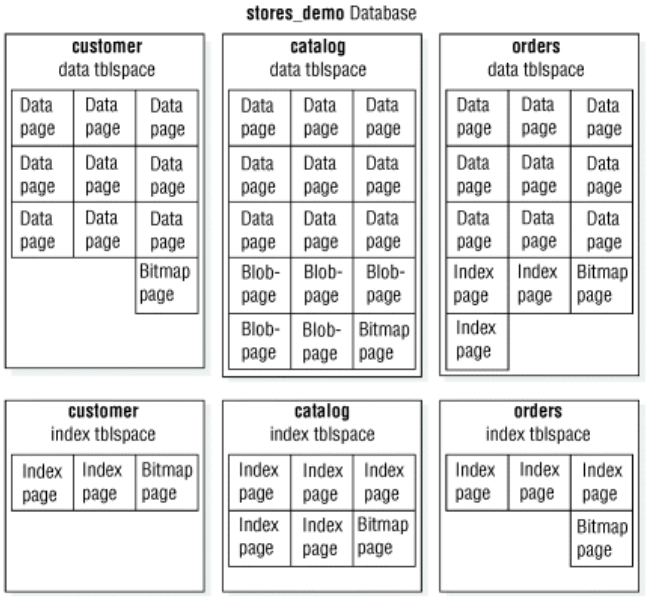

图 39: stores_demo 数据库中的样本表空间

扩展数据块交错

数据库服务器将属于表空间的页分配为扩展数据块。虽然扩展数据块内的页是连续的，但扩展数据块还是可能会分散在表所在的数据库空间中（甚至在不同的块上）。

下图描绘的情境是：两个非连续的扩展数据块属于 table_1 的表空间，而第三个扩展数据块属于 table_2 的表空间。table_2 扩展数据块位于第一个 table_1 扩展数据块和第二个 table_1 扩展数据块之间。此种情况下，扩展数据块发生交错。因为在 table_1 的顺序访问搜索需要磁盘头搜寻过 table_2 扩展数据块，因此性能会比 table_1 扩展数据块为连续时要差。有关如何避免与消除交错的扩展数据块的指示信息，请参阅《*SinoDB*® 性能指南》。

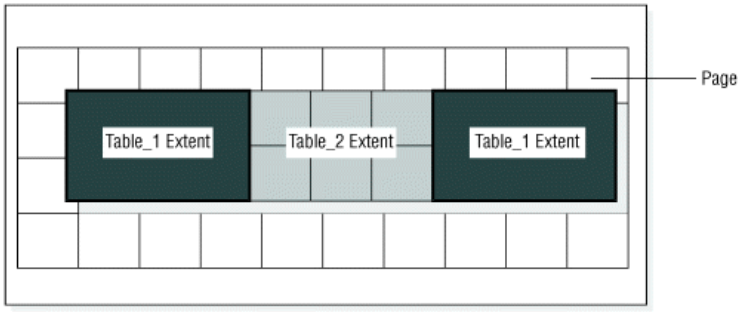

图 40: 属于一个数据库空间中两个不同表空间的三个扩展数据块

表分段存储和数据存储

分段存储功能使您更能掌控数据库存储数据的位置。不但可以指定单个表与索引的位置，还可以指定表与索引分段的位置（这些分段是位于不同存储空间的表或索引的不同部分）。

可以使用以下方式将表分段化：

- 分段跨越一个以上数据库空间的表。然而，您不能将分段放置于具有不同页大小的数据库空间中。所有的分段都必须有相同的页大小。
- 如果分段表使用基于表达式或循环分布方案，就可在单个数据库空间中创建分段表的多个分区。

可以将以下存储空间分段：

- 数据库空间

- 智能大对象空间

通常您在最初创建表时将表分段。CREATE TABLE 语句可采用以下形式之一：

```
CREATE TABLE tablename ... FRAGMENT BY ROUND ROBIN IN dbspace1,
 dbspace2, dbspace3;

CREATE TABLE tablename ...FRAGMENT BY EXPRESSION
   <Expression 1> in dbspace1,
   <Expression 2> in dbspace2,
   <Expression 3> in dbspace3;
```

FRAGMENT BY ROUND ROBIN 和 FRAGMENT BY EXPRESSION 关键字指的是两种不同的分布方案。两种语句都将分段与数据库空间关联。

如果将 AUTOLOCATE 配置参数或会话环境变量设置为正整数，但未指定表的位置，那么将会在数据库服务器选择的数据库空间中按循环顺序方式将新表分段。

对表进行分段时，还可在相同的数据库空间内为表创建多个分区，如下例所示：

```
CREATE TABLE tb1(a int)
   FRAGMENT BY EXPRESSION
         PARTITION part1 (a >=0 AND a < 5) in dbs1,
         PARTITION part2 (a >=5 AND a < 10) in dbs1
       ;
```

下图说明了分段在指定数据位置中的角色。

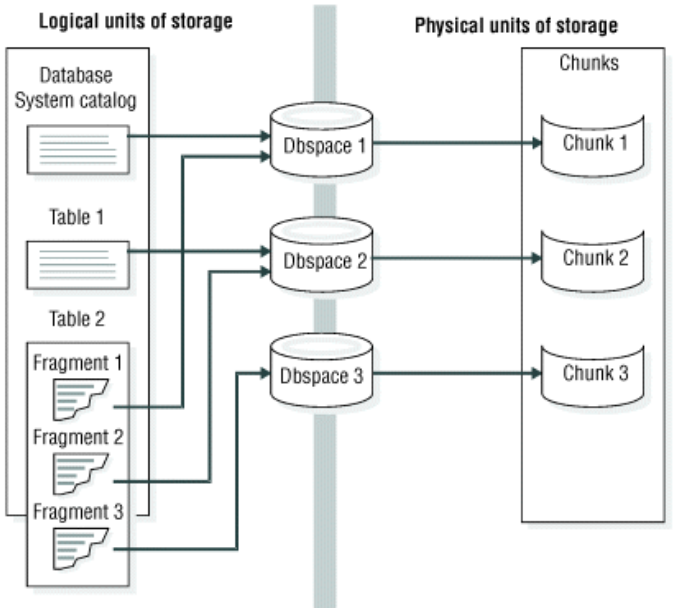

图 41: 链接逻辑存储单元（包括表分段）和物理存储单元的数据库空间

相关链接

管理磁盘空间 在第168页
管理自动定位和分段化 在第180页
《SinoDB 数据库设计和实现指南》: 表分段存储策略
《SinoDB 性能指南》: 分段存储准则
表 在第155页

存储数据所需的磁盘空间量

请遵循以下步骤来确定所需的磁盘空间量：

1. 计算所需的根数据空间大小
2. 估算要分配给所有的数据库服务器数据库的总磁盘空间量（包含开销与增长的空间）。

以下主题将说明这些步骤。

根数据库空间的大小

可以计算根数据库空间的大小，此空间存储了描述数据库服务器的信息。

以下存储結構可以存储在根数据空间中：

物理和逻辑日志（每一类型最少 200 KB）

虽然根数据空间是物理日志和逻辑日志文件的缺省存储位置，但还是要把逻辑文件移动到其他的数据库空间中。可以设置 AUTO_LLOG 配置参数来指定逻辑日志文件的数据库空间。可以将物理日志存储在物理日志空间。

推荐: 设置一个小型物理日志和一些小型逻辑日志。例如，创建三个 1000 KB 的逻辑日志文件，或者 3000 KB 的总日志空间。初始安装完成后，在不与其他数据库空间竞争 I/O 的区域创建逻辑日志的数据库空间，并将 AUTO_LLOG 配置参数指向那个数据库空间。在该数据库空间中为逻辑日志创建一组更大的逻辑日志文件，并从根数据库空间中删除原始日志。然后创建物理日志的物理日志空间 。使物理日志空间足够大以容纳最终的物理日志，并尽可能将它与其它数据库空间隔离。此配置优化日志记录性能和根数据空间，原因如下：

- 在移动这些日志记录后，将使剩馀在根数据库空间中未使用的空间最小化。
- 物理和逻辑日志彼此间，以及与根数据库空间之间，不会在相同磁盘上相互竞争空间与 I/O。
- 如果增加日志可以显著提高性能，服务器会自动增加总逻辑日志空间和物理日志的大小。

临时表

分析用户应用程序以估计数据库服务器对临时表可能需要的磁盘空间数量。尝试估计同时运行的语句的数量。返回的行和列占用的空间为估计所需空间量提供了良好的基础。数据库服务器在热复原期间创建的最大的临时表等于逻辑日志的大小。可以通过加总所有逻辑日志文件的大小来计算逻辑日志的大小。还必须分析用户应用程序以估计数据库服务器对显示临时表可能需要的磁盘空间量。

数据

尽管根数据库空间是数据库的缺省位置，但请不要将数据库和表存储在根数据库空间中。

系统数据库（大小依版本而变）

sysmaster、sysutils、syscdr 和 sysuuid 数据库，以及系统目录必须存储在根数据库空间。sysadmin 数据库默认情况下存储在根数据库空间中，但是，您可以在将 sysadmin 数据库移动到不同的数据库空间。

保留页（~ 24KB）

保留页包含数据库服务器使用的控制和跟踪信息。保留页必须存储在根数据库空间。

表空间 tblspace （最小 100 - 200KB）

tblspace 表空间包含所有表空间的信息。tblspace 表空间必须存储在根数据库空间。

该估计是初始化数据库服务器之前的根数据库空间的大小。根数据库空间的大小取决于是否将物理日志、逻辑日志，以及临时表存储在根数据库空间或另一个数据库空间。在磁盘初始化期间，根数据库空间必须足够大以能够进行最小大小的配置。

允许在根数据库空间的额外空间用于系统数据库增长、扩展保留页和充足的可用空间。扩展保留页数取决于数据库服务器中主块、镜像块、逻辑日志文件和存储空间的数目。

如果在服务器已初始化后，需要将根数据库空间增大，可以向根数据库空间添加新块。可以启用自动空间管理按需扩展根数据库空间。

重要： 为根数据库空间以及其他包含关键数据（例如，物理日志和逻辑日志）的数据库空间建立镜像。

相关链接

自动空间管理 在第185页
管理数据库空间 在第173页
移动逻辑日志文件 在第261页
更改物理日志的位置和大小 在第270页
创建临时数据库空间 在第178页
《*SinoDB* 管理员参考》: *ROOTSIZE* 配置参数
《*SinoDB* 管理员参考》: *reset sysadmin* 参数: 移动 *sysadmin* 数据库 *(SQL* 管理 *API)*

数据库所需空间量

数据库服务器数据存储所需要的额外的磁盘空间量取决于用户的需要，以及开销和增长。

用户运行的每一个应用程序都有不同的存储需求。下表提出了一些您可以用来计算要分配的磁盘空间（根数据库空间除外）量的步骤：

- 决定必须存储多少数据库和表。计算每个需要的空间量。
- 计算每个表的增长率，并为每个表分配一些磁盘空间以容纳增长。
- 决定为那些数据库和表建立镜像。

有关计算表大小的指示说明，请参阅《*SinoDB*® 性能指南》。

存储池

数据库服务器的每个实例都具有存储池。存储池包含有关以下对象的信息：目录、熟文件和原始设备，以便服务器可在必要时用于自动扩展现有数据库空间、临时数据库空间、智能大对象空间、临时智能大对象空间或 BLOB 空间。

当存储空间低于 SP_THRESHOLD 配置参数中定义的阈值时，数据库服务器会自动运行空间扩展任务，通过扩展空间中的现有块或添加新块的方法来扩展。

可以使用 SQL 管理 API 命令来执行以下操作：

- 添加、删除或修改用于描述存储池中一个目录、熟文件或原始设备的条目。必要时，服务器可使用指定的目录、熟文件或原始设备来向现有存储空间自动添加空间。
- 通过修改与扩展存储空间相关的两个不同的数据库空间大小（扩展大小和创建大小）来控制存储池条目的使用方式。
- 将块标记为可扩展或不可扩展
- 当不希望数据库服务器自动扩展空间时，立即扩展空间的大小。
- 立即按指定的最小的大小来扩展块。
- 通过存储池的条目创建存储空间或块。
- 将空的空间从已删除的存储空间或块返还给存储池。

sysadmin 数据库中的 storagepool 表包含有关数据库服务器实例的存储池中所有条目的信息。

相关链接

可扩展块 在第140页
自动空间管理 在第185页
创建和管理存储池条目 在第186页
自动空间管理 在第185页
创建和管理存储池条目 在第186页

磁盘布局准则

有效磁盘布局具有以下典型目标：

- 限制磁头移动
- 减少磁盘争用
- 平衡负载
- 最大化可用性

在设计磁盘布局时必须对这些目标做一些折中。例如，将系统目录表，逻辑日志和物理日志分开可以帮助减少对这些资源的争用。但是，此操作也会增加必须执行系统复原的机率。有关详细的磁盘布局准则，请参阅《*SinoDB*® 性能指南》。

相关链接

分区和偏移量 在第140页
表 在第155页

数据库空间和块准则

本主题列出了一些不需要有关特定数据库特征信息的磁盘布局的常规策略。

- 将磁盘分区与块相关联，并为根数据库空间分配至少一个附加块。

  已分区的磁盘可能需要使用偏移量。有关详细信息，请参阅 在 *UNIX* 上分配原始磁盘空间 在第170页。

  提示： 以一个块的最大大小 4TB，您可以通过为每个磁盘驱动器指定块来避免进行分区。
- 为关键数据空间（根数据空间、包含物理日志和逻辑日志文件的数据空间）建立镜像。还可为使用率高的的数据库和表建立镜像。

  可以在数据库空间级别上指定镜像。对属于数据库空间的所有块来说，镜像可以打开也可以关闭。将主服务器和镜像的数据库空间放置在不同磁盘上。理想情况下，不同的控制器处理不同的磁盘。
- 将临时表和排序文件分布在多个磁盘上。

  要为临时表和排序文件定义数个数据库空间，可使用 onspaces -t。当将这些数据库空间放置在不同磁盘上并将他们罗列在 DBSPACETEMP 配置参数中时，就可以将与临时表和排序文件关连的 I/O 分布在多个磁盘上。有关使用 DBSPACETEMP 配置参数或环境变量的信息，请参阅《*SinoDB*® 管理员参考》中有关配置参数的章节。
- 将物理日志保留在根数据库空间，而将逻辑日志从根数据库空间移出。但是，如果打算将系统目录存储在根数据库空间中，请将物理日志移到另一个数据库空间中。

  有关日志存储位置的建议，请参阅 逻辑日志文件的位置 在第244页。也请参阅 移动逻辑日志文件 在第261页和更改物理日志的位置和大小 在第270页。
- 提升备份和复原的性能，请执行以下操作：
  - 将系统目录与其追踪的数据集群起来。
  - 如果使用 ON-Bar 对高速磁带机执行并行备份，请将这些数据库存储在数个小的数据库空间中。

    有关附加的性能建议，请参阅《*SinoDB*® 备份和还原指南》。

表位置准则

本主题在假定数据库中具有关于表的某些特征的前提下，列出了一些优化磁盘布局的策略。

您可以使用表分段存储通过更高程度的控制来实现这些策略中的许多策略：

- 隔离独立磁盘上高使用率的表。

  要隔离自身磁盘设备上的高使用率的表，请将该设备指定给一个块，并将该块指定给一个数据库空间 。最后，将频繁使用的表放置在使用 CREATE TABLE 的 IN *dbspace* 选项创建的数据库空间中。

  要显示每个块的 I/O 操作级别，请运行 onstat -g iof 选项。
- 将高使用率的表在多个磁盘上分段。

- 將数据库空间中的相关表分組。

  如果包含数据库空间的设备发生故障，该数据库空间的所有表都是不可访问的。然而，其他数据库空间的表仍然是可访问的。尽管包含关键信息的数据库空间故障时必须执行执行冷复原，但非关键数据库空间发生故障时只须执行热复原。
- 将使用率高的表放置于磁盘的中间分区上。
- 优化表扩展数据块的大小。

有关更多信息，请参阅《*SinoDB*® 性能指南》中有关表性能注意事项的章节。有关 onstat 选项的信息，请参阅《*SinoDB*® 管理员参考》。

样本磁盘布局

当开始组织磁盘空间时，数据库服务器管理员通常会想到下列目标中的一个或多个目标：

- 高性能
- 高可用性
- 简单和频繁的备份和复原

满足任一目标都有取舍。例如，将系统配置成高性能通常会导致数据可用性降低。以下章节提供了一个示例，该示例中数据库管理员必须在磁盘资源有限的情况下作出磁盘布局的选择。这些章节描述了两种不同的磁盘布局解决方案。第一个解决方案代表性能优化，而第二种解决方案代表可用性和复原优化。

样本磁盘布局的设置是使用 stores_demo 数据库结构（但不是卷）的虚构的体育商品数据库。在此示例中，数据库服务器配置成可处理大约 350 用户和 3GB 的数据。磁盘空间资源如下表所示。

磁盘驱动器	驱动器大小	高性能
磁盘 1	2.5GB	否
磁盘 2	3GB	是
磁盘 3	2GB	是
磁盘 4	1.5GB	否

数据库包含两个大表：cust_calls 和 items。假设这些表中都包含 1,000,000 多行。cust_calls 表提供了客户致电经销商的所有通话记录。items 表经销商已发货的每张订单上的每个项目。

数据库包含两个使用率高的表：items 和 orders。这些表经常被全国各地的用户访问。

其余的表为低容量表，数据库服务器使用这些表来查找数据，如：邮政编码或制造商。

表名	最大大小	访问率
cust_calls	2.5 GB	低
items	0.5 GB	高
orders	50 MB	高
customers	50 MB	低
stock	50 MB	低
catalog	50 MB	低
manufact	50 MB	低
state	50MB	低
call_type	50MB	低

性能优先级最高时的样本布局

要优化性能，请使用多个存储空间和多个磁盘。下图显示了一种针对性能优化的磁盘布局。此磁盘布局使用以下策略来提高性能：

- 迁移根数据库空间中的逻辑日志和物理日志文件

  此策略将逻辑日志和物理日志分开，从而减少对根数据空间的争用。为了获得最佳性能，请利用逻辑和物理日志的自动性能优化：

  - 创建物理日志空间以启用物理日志的自动扩展。
  - 设置 AUTO_LLOG 配置参数以启用逻辑日志在指定数据库空间中的自动扩展。

  如果在安装过程中创建服务器，则会创建物理日志空间，并将 AUTO_LLOG 配置参数设置为非关键数据库空间 。
- 在独立磁盘上的数据库空间中使用率最高的两个表的位置

  这些磁盘都不存储逻辑日志或物理日志。理想情况下，可将 items 和 orders 表分别存储在单独的高性能磁盘上。然而，在当前方案中，此策略无法使用，因为需要其中一个高性能磁盘来存储大的 cust_calls 表（对于此任务来说，其他两个磁盘太小）。

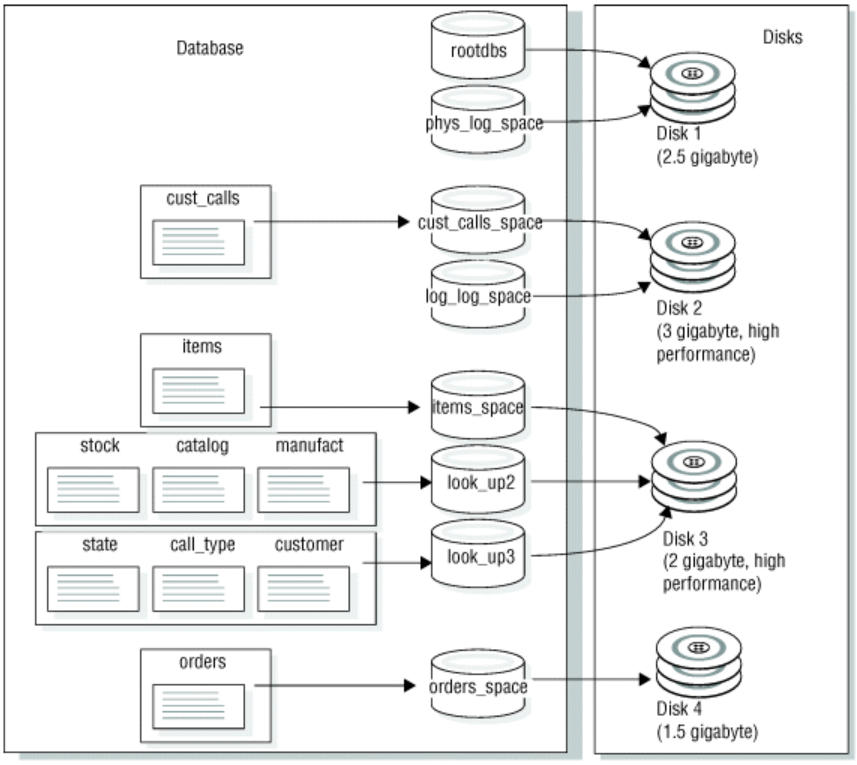

图 42: 针对性能优化的磁盘布局

可用性优先级最高时的样本布局

前面的磁盘布局的缺点在于如果 Disk 1 或 Disk 2 发生故障，那么整个数据库服务器将会停止运行直到您将这些磁盘上的数据库空间从备份中复原为止。换句话说，该磁盘布局在可用性方面很差。

下图为另一种磁盘布局，在优化可用性以及镜像。这种布局将所有关键数据空间（系统目录表、物理日志和逻辑日志）镜像到独立的磁盘上。理想情况下，可以将逻辑日志和物理日志分开（如同之前的布局），并将每个磁盘镜像到其自身的镜像磁盘。然而，在此方案中，所需的磁盘数不存在；因此，逻辑日志和物理日志都位于根数据库空间上。

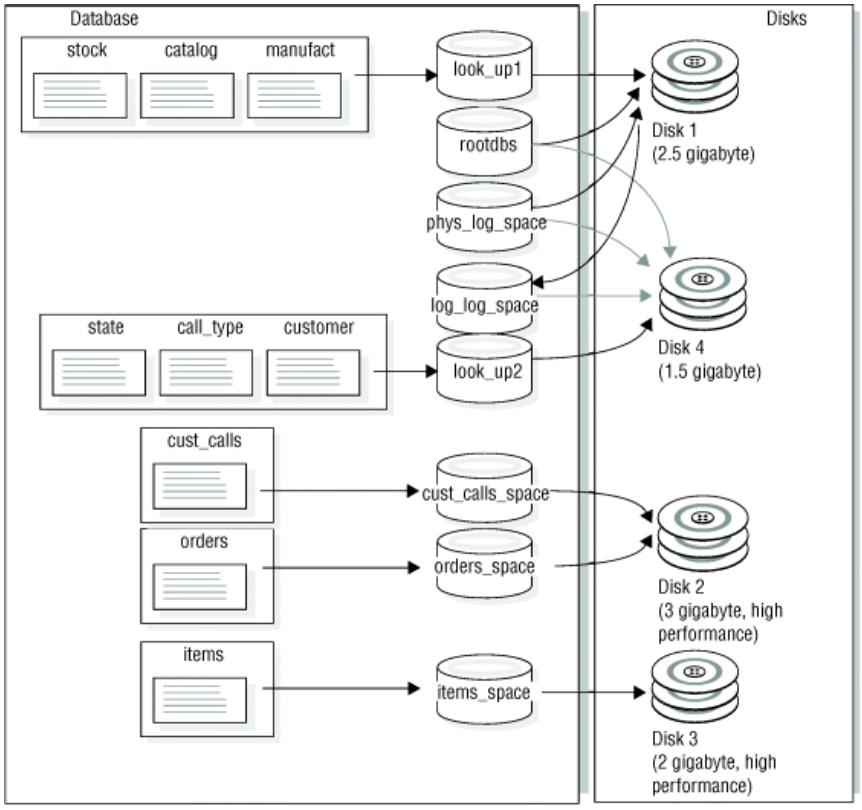

图 43: 针对可用性优化的磁盘布局

逻辑卷管理器

可以使用逻辑卷管理器 (LVM) 实用程序通过用户定义的逻辑卷来管理磁盘空间。

很多计算机制造商提供带专有 LVM 的计算机。可使用数据库服务器在由大多数专有 LVM 管理的磁盘上存储和检索数据。本节的后续部分将说明逻辑卷管理器的优势与劣势。

大多数 LVM 可以管理多个 GB 的磁盘空间。数据库服务器块大小限制为 4TB，且只有当分配的块的偏移量为零时才能获取这个大小。因此，你必须将分配为块的卷的大小限制为 4TB。

由于可使用 LVB 将磁盘驱动器分为多个卷，因此可控制数据放置在给定磁盘上的位置。可通过定义由磁盘驱动器最中间柱面组成的卷并在该卷中放置使用率高的表来提高性能。（从技术上讲，您不能把表直接放置在卷中。您必须先分配一个块作为卷，然后将块指定到数据库空间 ，最后将表放置到数据库空间中。有管更多信息，请参阅 控制简单大对象数据的存储位置 在第144页。）

提示： 如果选择使用大的磁盘驱动器，那么可将块指定给一个驱动器，并且不必将该磁盘分区。

还可通过使用逻辑卷管理器定义跨多个磁盘的卷并且将表放置到该卷中来提高性能。

很多逻辑卷管理器允许标准操作系统格式实用程序所不允许的的灵活性程度。其中一项就是在您定义逻辑卷之后重新将其定位。因此，与操作系统格式实用程序相比，首次获取磁盘空间的布局就不那么重要了。

LVM 经常提供操作系统级的镜像工具。有关更多信息，请参阅 镜像的替代方法 在第275页。

管理磁盘空间

可使用数个实用程序和工具来管理数据库服务器控制的磁盘空间和数据。

可以使用以下实用程序来管理存储空间：

- onspaces 实用程序命令
- OAT
- SQL 管理 API 命令

《*SinoDB*® 性能指南》 也包含有关管理磁盘空间的信息。尤其它描述了如何消除交错的扩展数据快、如何回收空的扩展数据块中的空间，以及如何提高磁盘 I/O。

可以生成 SQL 管理 API 或 onspaces 命令再现以 dbschema 实用程序存在于文件中的存储空间、块和日志。

相关链接

表分段存储和数据存储 在第161页
存储空间的创建和管理 在第26页
《*SinoDB* 迁移指南》: 存储空间、块和日志的创建
《*SinoDB* 管理员参考》: *SQL* 管理 *API* 函数
《*SinoDB* 性能指南》: 管理扩展数据块
《*SinoDB* 性能指南》: 管理智能大对象空间
数据存储 在第138页

分配磁盘空间

本节解释如何分配数据库服务器的磁盘空间。

请在分配的磁盘空间之前阅读以下章节：

- *UNIX* 上未缓冲或已缓冲磁盘的访问 在第139页
- 存储数据所需的磁盘空间量 在第163页
- 磁盘布局准则 在第165页

在可以创建存储空间或块，或对现有存储空间制作镜像之前，必须为块文件分配磁盘空间。可以为数据库服务器磁盘空间分配一个空文件或部分原始磁盘。

> 仅 UNIX: 在 UNIX™ 上，如果分配原始磁盘空间，就必须使用 UNIX™ 的 ln 命令来创建特殊字符设备名称与另一个文件名之间的链接。有关本主题的更多信息，请参阅 创建原始设备的符号链接 *(UNIX)* 在第171页。
>
> 将 UNIX™ 文件和其固有的操作系统接口用于数据库服务器磁盘空间，称为使用 熟空间。
>
> 仅 Windows: 在 Windows™ 上，数据库服务器磁盘空间必须使用 NTFS 文件格式。有关此建议的更多信息，请参阅 *UNIX* 上未缓冲或已缓冲磁盘的访问 在第139页。

您可以在磁盘和控制器上平衡块。在单个磁盘上放置多个块可以提高吞吐量。

指定偏移量

当将磁盘空间块分配给数据库服务器时，请指定偏移量。

为以下两个目的之一指定偏移量：

- 防止数据库服务器覆盖分区信息
- 在分区、磁盘设备或熟文件上定义多个块

偏移量的最大值是 4TB。

许多计算机系统和某些磁盘驱动器制造商会将物理磁盘驱动器的信息保存在驱动器本身中。此信息有时称为卷目录表（VTOC）或磁盘标签。VTOC 通常存储在驱动器的第一个轨道上。备用扇区和坏扇区映射表（也称为 重指向表 ）也可能存储在第一个轨道上。

如果计划从磁盘起始处分配分区，那么可能需要使用偏移量以防止数据库服务器覆盖操作系统所需的关键信息。有关所需的精确偏移，请参阅磁盘驱动器手册。

重要： 如果您运行了两个或更多个数据库服务器实例，那么必须格外小心，不要定义重叠的块。重叠块会导致数据库服务器用重叠块中的不相关数据覆盖另一个块中的数据。此覆盖事实上会删除重叠数据。

为数据库空间的初始块指定偏移量
对于根数据库空间的初始块及其镜像（如果有的话），分别用 ROOTOFFSET 和 MIRROROFFSET 参数指定偏移量。

有关更多信息，请参阅 《*SinoDB*® 管理员参考》中有关参数配置的主题。

为附加块指定偏移量
要为数据库服务器空间的附加块指定偏移量，必须在为数据库服务器分配空间时提供偏移量作为参数。

有关更多信息，请参阅 创建使用缺省页大小的数据库空间 在第173页。

使用偏移量创建多个块
通过指定偏移量和分配小于用总空间的块，您可以从磁盘分区、磁盘设备或文件中创建多个块。

偏移量指定块的开始位置。数据库服务器通过将块的大小添加到偏移量，来确定数据块最后字节的位置。

对第一个块，指定任何初始偏移量（如果有需要），并将其大小指定为少于所分配磁盘空间总大小的数量。对于每个附加块，指定偏移量时要包含所有先前分配块的大小，并加上初始偏移量，然後分配大小少于或等于分配中剩余的空间量大小。

在 UNIX™上分配熟文件空间

以下是为熟文件分配磁盘空间的过程示例。

在 UNIX™ 上为名为 usr/data/my_chunk 的熟文件分配磁盘空间：

1. 以用户 informix 身份登录：su informix
2. 将目录切换至熟空间将位于的目录：cd /usr/data
3. 通过将 null 连接至数据库服务器将用于磁盘空间的文件名来创建块：cat /dev/null > my_chunk
4. 将文件许可权设置为 660 (rw-rw----)：chmod 660 my_chunk
5. 必须将组和文件所有者均设置为 informix：

```
ls -l my_chunk  -rw-rw----
  1  informix   informix
  0  Oct 12 13:43 my_chunk
```

6. 使用 onspaces 创建存储空间或块

有关如何使用已分配的文件来创建存储空间的信息，请参阅： 创建使用缺省页大小的数据库空间 在第173页、 创建 *BLOB* 空间 在第181页和 创建智能大对象空间 在第182页

在 UNIX™ 上分配原始磁盘空间

要分配原始空间，您必须具有专用于原始空间的可用磁盘分区。要创建原始磁盘空间，您可以将磁盘重新分区或卸载现有文件系统。在卸载设备前，请备份所有文件。

分配原始磁盘空间

1. 创建并安装原始设备

   有关如何在 UNIX™ 上分配原始磁盘空间的特定指示信息，请参阅操作系统文档和 *UNIX* 上未缓冲或已缓冲磁盘的访问 在第139页。
2. 将字符专用设备的所有权和许可权更改到 informix。

   字符专用设备的文件名通常以字母 r 开头。有关过程，请参阅 在 *UNIX*上分配熟文件空间 在第170页中的 *4* 在第170页 和 *5* 在第170页 。
3. 验证字符专用设备的操作系统许可权是否为 crw-rw----。
4. 使用 UNIX™ 链接命令 ln -s创建字符专用设备名称与另一个文件名称之间的符号链接。

有关详细信息，请参阅 创建原始设备的符号链接 *(UNIX)* 在第171页。

限制： 在创建了数据库服务器用于磁盘空间的原始设备之后，请不要在分配为数据库服务器磁盘空间的同一原始设备上创建文件系统。也不要将同一原始设备作为数据库服务器磁盘空间分配的交换空间。

相关链接

分区和偏移量 在第140页

创建原始设备的符号链接 (UNIX™)

使用符号链接分配标准设备名称并指向设备。

使用 UNIX™ 链接命令（通常是 ln），在字符专用设备名与另一个文件名之间创建符号链接。要验证设备和链接是否都存在，请在设备目录下运行 UNIX™ 命令 ls -l (BSD 上的ls -lg)。以下示例显示原始设备的链接。如果操作系统不支持符号链接，那么硬链接也可起作用。

```
ln -s /dev/rxy0h /dev/my_root # orig_device link to symbolic_name
ln -s /dev/rxy0a /dev/raw_dev2
ls -l
crw-rw---  /dev/rxy0h
crw-rw---  /dev/rxy0a
lrwxrwxrwx /dev/my_root@->/dev/rxy0h
lrwxrwxrwx
/dev/raw_dev2@->/dev/rxy0a
```

为何使用符号链接？如果是在原始设备上创建块，但该设备发生了故障，那么在更换该原始设备并使用相同路径名之前，无法从备份复原。在上次备份时可访问的所有块，执行复原时均必须为可访问的。

符号链接简化了磁盘故障的恢复，并使您可迅速替换块所处的磁盘。可用另一个设备替换故障设备，将新设备路径名链接到先前为故障设备创建的同一文件名，然后复原数据。无需等待原始设备修复。

在 Windows™ 上分配 NTFS 文件空间

在 Windows™ 上，数据库服务器默认使用 NTFS 文件。在 NTFS 文件系统中，可为未缓冲文件使用标准文件名。如果所有分区都是 FAT 文件，可将其中一个转换为 NTFS。

要为数据库服务器磁盘空间或镜像空间分配 NTFS 文件空间，第一步是创建空（0 字节）文件。

分配 NTFS 文件空间：

1. 以 Informix-Admin 组成员的身份登录。
2. 打开 MS-DOS 命令 shell。
3. 切换至您希望分配空间的目录，如以下示例所示：

```
c:> cd \usr\data
```

4. 如必要，请运行以下命令将分区转换为 NTFS：convert /fs:ntfs
5. 使用以下命令创建空文件：c:> copy nul my_chunk
6. 如果要验证文件是否已创建，请使用 dir 命令进行验证。

分配了文件空间后，通常可使用 onspaces 创建数据库空间或其他存储空间。有关如何创建数据库空间或 BLOB 空间的信息，请参阅 创建使用缺省页大小的数据库空间 在第173页和创建 *BLOB* 空间 在第181页。

在 Windows™上分配原始磁盘空间

可将 Windows™ 上的原始磁盘空间配置为逻辑驱动器或物理驱动器。

要找到驱动器盘符或磁盘号，请运行 Disk Administrator。如果必须分割驱动器（将多个物理磁盘组合成一个逻辑磁盘），那么仅逻辑驱动器规范可起作用。

您必须是 Informix-Admin 组的成员才能创建存储空间或添加块。原始磁盘空间可以是格式化的或未格式化的磁盘空间。

重要: 如果将格式化的驱动器或磁盘分区分配为原始磁盘空间，而它包含数据，那么数据库服务器会在开始使用磁盘空间时覆盖数据。在将磁盘空间分配给数据库服务器前，必须确保原始磁盘空间上的所有数据是可清除的。

要指定逻辑驱动器，请执行以下操作：

1. 将驱动器盘符指定给磁盘分区。
2. 为 onconfig 文件中的 ROOTDBS 指定以下值：\\.*drive_letter*
3. 要创建存储空间或添加块，请指定逻辑驱动器分区。

   以下示例在 e: 驱动器上 5200 KB 偏移量处向数据库空间 dpspc3 添加 5000 KB 的块。

```
onspaces -a dbspc3 \\.\e:-o 5200 -s 5000
```

要指定物理驱动器，请执行以下操作：

1. 如果磁盘分区尚未指定驱动器盘符，请为 onconfig 文件中的 ROOTDBS 指定以下值：\\.\PhysicalDrive<*number*>
2. 要创建存储空间或添加块，请指定物理驱动器分区。

   以下示例在 PhysicalDrive0 上 5200 KB 偏移量处向数据库空间 dpspc3 添加 5000 KB 的块。

```
onspaces -a dbspc3 \\.\PhysicalDrive0 :-o 5200 -s 5000
```

指定存储空间和块的名称

块名称与存储空间名称遵循相同的规则。

如下所示，为存储空间或块指定明确的路径名：

- 如果要在 UNIX™ 上使用原始磁盘，那么必须使用已链接的路径名。 （请参阅 创建原始设备的符号链接 *(UNIX)* 在第171页。）
- 如果要在 Windows™ 上使用原始磁盘，那么路径名将采用以下格式，其中 *x* 指定磁盘驱动器或分区：

```
\\.\x:
```

- 如果要使用文件作为数据库服务器磁盘空间，那么路径名将是完整的路径和文件名。

在创建存储空间或添加块时，请使用这些命名规则。文件名必须具有以下特征：

- 唯一，且不超过 128 字节
- 以字母或下划线开头
- 仅包含字母、数字、下划线或 $ 字符

除非名称用引号括起，否则名称不区分大小写。默认情况下，数据库服务器将名称中的大写字符转换为小写。如果希望在名称中使用大写，请用引号将其括起，并将 DELIMIDENT 环境变量设置为 ON。

相关链接

块 在第138页

指定块的最大大小

在大多数平台上，最大块大小为 4 TB；但在其他平台上，最大块大小为 8 TB。

要确定平台所支持的块大小，请参阅机器说明文件。

指定块和存储空间的最大数量

可在数据库服务器系统上指定存储空间的最大块数为 32,766 以及最大存储空间数为 32,766。

存储空间可以是数据库空间、BLOB 空间和智能大对象空间的任何组合。

考虑到可应用于数据库服务器实例大小的所有限制，实例的最大大小约为 8 PB。

更改物理模式后备份

必须对根数据库空间和修改过的存储空间执行 0 级备份，从而确保在某些情况下可复原数据。

执行 0 级备份，从而确保在执行以下操作时可复原数据：

- 添加或删除镜像
- 删除逻辑日志文件
- 更改物理日志的大小或位置
- 更改存储管理器配置
- 添加、移动或删除数据库空间、BLOB 空间或智能大对象空间
- 对数据库空间、BLOB 空间或智能大对象空间添加、移动或删除块

  重要： 添加新的逻辑日志时，不再需要为使用新逻辑日志而对根数据库空间和修改过的数据库空间执行 0 级备份。但必须执行 0 级备份以防止 1 级和 2 级备份失败。

必须对修改过的存储空间执行 0 级备份，以确保在转换到日志记录表类型之前可以复原未记录的数据：

- 将非日志记录数据库转换为日志记录数据库
- 将 RAW 表转换为标准表

监视存储空间

您可监视存储空间状态，并配置当存储空间变满时如何通知。

当存储空间或分区变满时，消息会显示在联机消息日志文件中。

可以使用 STORAGE_FULL_ALARM 配置参数配置存储空间变满时触发的警报。可指定警报发送频率，以及要发送的警报的最低严重性级别。 默认情况下，警报时间间隔是 600 秒，警报严重性级别为 3。有关 STORAGE_FULL_ALARM 配置参数和事件警报的更多信息，请参阅 《*SinoDB*® 管理员参考》。

如果高可用性集群中的主服务器遇到了空间不足的情况，且启用了 STORAGE_FULL_ALARM 配置参数，那么将触发事件警报，并在主服务器上返回错误状态，但是不会在任何辅助服务器上返回错误状态。这是预期行为，因为主服务器遇到空间不足情况时，不会再将日志记录从主服务器发送到辅助服务器。在这种情况下，辅助服务器永远不会超过其存储限制，因此也不会触发事件警报或返回错误状态。

可使用 SinoDB® 调度程序设置自动监视存储空间状态的任务。任务的属性定义调度程序收集的信息，并指定任务运行的频率。例如，可将任务定义为一周五天、每隔一小时监视存储空间。有关更多信息，请参阅 调度程序 在第441页和创建任务 在第448页。

管理数据库空间

此章节包含有关创建具有和不具有缺省页大小的标准和临时数据库空间的信息、有关创建数据库空间时在数据库空间中为表空间 tblspace 指定第一个和下一个扩展数据块大小的信息，以及有关向数据库空间或 BLOB 空间添加块的信息。

有关监视数据库空间的信息，请参阅 监视存储空间 在第173页。

相关链接

控制简单大对象数据的存储位置 在第144页
根数据库空间的大小 在第163页

创建使用缺省页大小的数据库空间

可使用 onspaces 创建标准数据库空间和临时数据库空间。

有关创建具有非缺省页大小的数据库空间的信息，请参阅 创建具有非缺省页大小的数据库空间 在第175页。

任何新添加的数据库空间以及其镜像（如果有）立即可用。如果使用镜像，可在创建数据库空间时对其制作镜像。镜像可立即生效。

要使用 onspaces 创建标准数据库空间：

1. 在 UNIX™ 上，必须以用户 informix 或 root 身份登录来创建数据库空间。

   在 Windows™ 上，Informix-Admin 组中的用户可创建数据库空间。
2. 请确保数据库服务器处于联机、管理或静默模式中。
3. 按 分配磁盘空间 在第169页中所述，为数据库空间分配磁盘空间。
4. 要创建数据库空间，请使用 onspaces -c -d 选项。

   KB 是 -s *size* 和 -o offset 选项的缺省单位。要将 KB 转换为 MB，可将该单位数量乘以 1024（例如， 10 MB = 10 * 1024 KB）。

   如果要创建具有非缺省页大小的数据库空间，请参阅 创建具有非缺省页大小的数据库空间 在第175页以获取有关 onspaces 附加选项的信息。
5. 如果不想为数据库空间中的表空间 tblspace 指定第一个和下一个扩展数据块大小，请转至 *6* 在第174页。

   如果想为数据库空间中的表空间 tblspace 指定第一个和下一个扩展数据块大小，请参阅 为表空间 *tblspace* 指定第一个和下一个扩展数据块大小 在第174页以获取其他信息。
6. 创建数据库空间后，必须对根数据库空间和新数据库空间执行 0 级备份。

以下示例显示了如何在 UNIX™ 上使用原始磁盘空间创建对主块和镜像块的偏移量均为 5000 KB 的 10 MB 镜像数据库空间 dbspce1。

```
onspaces -c -d dbspce1 -p /dev/raw_dev1 -o 5000 -s 10240 -m /dev/raw_dev2
 5000
```

以下示例显示了如何从 Windows™ 上的原始磁盘空间（驱动器 e:）创建偏移量为 200 KB 的 5 MB 数据库空间 dbspc3：

```
onspaces -c -d dbspc3 \\.\e:-o 200 -s 5120
```

有关使用 onspaces 创建数据库空间的更多信息，请参阅 数据库空间 在第144页和《*SinoDB*® 管理员参考》中有关 onspaces 实用程序的信息。

为表空间 tblspace 指定第一个和下一个扩展数据块大小

如果要减少表空间 tblspace 的扩展数据块数，并降低发生必须将表空间 tblspace 扩展数据块放入非主块的情况的频率，可以指定第一个和下一个扩展数据块大小。（ 主块 是数据库空间中的初始块。）

可选择指定第一个扩展数据块大小、下一个扩展数据块大小、第一个和下一个扩展数据块大小，或都不指定。如果不为表空间 tblspace 指定第一个或下一个扩展数据块大小， 那么 SinoDB® 将使用现有的缺省扩展数据块大小。

可使用 TBLTBLFIRST 和 TBLTBLNEXT 配置参数为服务器初始化时创建的根数据库空间 中的表空间 tblspace 指定第一个和下一个扩展数据块大小。

可使用 onspaces 实用程序为非根数据库空间中的表空间 tblspace 指定第一个和下一个扩展数据块大小。

创建数据库空间时，仅可以指定第一个和下一个扩展数据块大小。数据库空间创建后，不能改变第一个和下一个扩展数据块大小的规格。此外，不能指定临时数据库空间、智能大对象空间、BLOB 空间或外部空间的扩展数据块大小。

要指定第一个和下一个扩展数据块大小，请执行以下操作：

1. 确定在表空间 tblspace 中所需的总页数。

   页数等于表数 + 拆离索引数 + 可能位于数据库空间的表分段数 + 表空间 tblspace 的一个页面。

2. 计算页数所需的 KB 数。

   此数字将取决于系统上一个页的 KB 数。

3. 通过考虑创建数据库空间期间分配的表空间 tblspace 的所有扩展数据块的重要性以及是否扩展数据块的分配必须连续来确定系统上空间管理需求。

   这些问题越重要，那么第一个扩展数据块大小就必须越大。如果不是很在意可能在辅助块中存在不连续的扩展数据块，那么第一个和下一个扩展数据块大小可以小一些。

4. 如下所示，指定扩展数据块大小：

   - 如果空间需求用于根数据库空间，那么在 TBLTBLFIRST 配置参数指定第一个扩展数据块大小，在 TBLTBLNEXT 配置参数指定下一个扩展数据块大小。然后，初始化数据库服务器实例。
   - 如果空间需求用于非根数据库空间，使用 onspaces 实用程序在命令行上指示第一个和下一个扩展数据块大小以创建数据库空间。

Extent sizes must be in KB and must be multiples of the page size. When you specify first and next extent sizes, follow these guidelines:扩展数据块大小必须以 KB 计，并且必须是页大小的倍数。指定第一个和下一个扩展数据块大小时，请遵循以下准则：

扩展数据块类型	最小大小	最大大小
非根数据库空间中的第一个扩展数据块	相当于 50 页 （以 KB 指定）。这是系统缺省值。例如，对于 2 KB 页面的系统，最小长度为 100。	初始块大小，减去任何系统对象（如保留页、数据库表空间，以及物理和逻辑日志）所需的空间。
根数据库空间中的第一个扩展数据块	相当于 250 页 （以 KB 指定）。这是系统缺省值。	初始块大小，减去任何系统对象（如保留页、数据库表空间，以及物理和逻辑日志）所需的空间。
下一个扩展数据块	系统上磁盘页大小的 4 倍。在任何类型的数据库空间上，缺省值均为 50 页。	最大块大小减去 3 页。

可使用以下 onspaces 实用程序的 -ef 和 -en 选项为非根数据库空间中的表空间 tblspace 指定第一个和下一个扩展数据块大小。

- 第一个扩展数据块大小：-ef *size_in_kbytes*
- 下一个扩展数据块大小：-ef *size_in_kbytes*

例如，可以指定：

```
onspaces -c -d dbspace1 -p /usr/data/dbspace1 -o 0 -s 1000000 -e 2000 -n 1000
```

可使用 oncheck -pt 和 oncheck -pT 显示表空间 tblspace 的第一个和下一个扩展数据块大小。

如果正在使用数据复制并且在主数据库服务器上创建数据库空间，那么第一个和下一个扩展数据块大小将通过 ADDCHK 日志记录传递到辅助数据库服务器。

有关 onspaces 实用程序、oncheck 命令，以及为表空间 tblspace 指定第一个和下一个扩展数据块大小的更多信息，请参阅 《*SinoDB*® 管理员参考》。

相关链接

《*SinoDB* 管理员参考》：*TBLTBLFIRST* 配置参数

《*SinoDB* 管理员参考》：*TBLTBLNEXT* 配置参数

创建具有非缺省页大小的数据库空间

如果需要的密钥长度比可用于缺省页大小的更长，那么可以指定标准或临时数据库空间的页大小。

根数据库空间使用缺省页大小。如果要创建具有不同页大小的数据库空间，其大小必须为缺省页大小的整数倍，且不能超过 16 KB。

对于具有足够存储空间的系统，较大页大小的性能优势包括：

- 减少 B 型树索引的深度，即使是较小的索引键。
- 减少检查点时间，通常出现在使用较大页大小时。

其他的性能优势还体现在可以：

- 将当前跨多个页（大小为缺省页大小）的长行在同一页上分组。
- 为临时表定义不同的页大小，因此临时表具有单独缓冲池。

表在一个数据库空间中，而该表的索引可在另一个数据库空间中。这些分区的页大小可以不同。

创建具有非缺省页大小的数据库空间：

1. 如果已从早于 V10.00 的版本升级，请运行 onmode -BC 2 命令启用大块模式。
   缺省情况下，当首次初始化或重新启动 SinoDB® 时，SinoDB® 将启用大块模式。
2. 可选： 创建对应于数据库空间页大小的缓冲池。可使用 onparams 实用程序或 BUFFERPOOL 配置参数。
   如果创建的数据库空间其页大小没有对应的缓冲池，那么 SinoDB® 将使用在 onconfig 文件定义的 BUFFERPOOL 配置参数的缺省值自动创建缓冲池。
   无法使多个缓冲池具有相同的页大小。
3. 创建数据库空间时定义数据库空间的页大小。可使用 onspaces 实用程序。

> 提示： 如果使用非缺省页大小，可能需要增加物理日志的大小。如果对非缺省页执行许多更新，可能需要将物理日志大小增大 150% 到 200%。要调整物理日志，可能需要进行一些试验。可根据物理日志填满触发检查点的频率，按需调整物理日志的大小。

通过使用直接 I/O 提高熟文件数据库空间的性能

在 UNIX™ 系统上，可通过使用 I/O 提高用于数据库空间块的熟文件的性能。

直接 I/O 必须可用，且文件系统必须支持用于数据库空间 块的页大小的直接 I/O。

可使用 SinoDB® 将原始设备或熟文件用于数据库空间块。通常，熟文件比较慢，因为文件系统提供的额外的开销和缓冲。直接 I/O 绕过文件系统缓冲区的使用，因此对磁盘的读写效率更高。使用 DIRECT_IO 配置参数指定直接 I/O。如果文件系统支持用于数据库空间块的页大小的直接 I/O 并使用直接 I/O，那么熟文件的性能可能接近用于数据库空间块的原始设备的性能。

要通过使用直接 I/O 提高熟文件数据库空间的性能，请执行以下操作：

1. 确认您是否具有直接 I/O，且文件系统是否支持用于数据库空间块的页大小的直接 I/O。
2. 将 DIRECT_IO 配置参数设置为 1 启用直接 I/O。

如果有 AIX® 操作系统，那么还可以 SinoDB® 启用并行 I/O，在对使用熟文件的块执行读写时与直接 I/O 一起使用。

有关使用直接 IO 或并行 IO 的更多信息，请参阅 《*SinoDB*® 性能指南》。

将多个指定分段存储在单个数据库空间

对于使用基于表达式、时间间隔、列表或循环分布方案的分段表，可以创建可位于单个数据库空间中的指定分段。

将多个表或索引分段存储在单个数据库空间中时的查询性能将优于将每个分段存储在不同数据库空间中时的查询性能，并且简化了数据库空间的管理。

假设使用基于表达式的分布方案创建分段表，该方案中的每个表达式指定放至于特定分段中的数据集。您可决定使用某个数据库空间中的一个月的数据，以及其他 11 个数据库空间中的 11 个月的数据来分割表中的数据。但是，如果您想要对所有年度数据只使用一个数据库空间，那么您可以创建指定分段，使每个月的数据存储在一个数据库空间中。

如果创建具有指定分段的分段表，那么 sysfragments 系统目录表中每一行的 partition 列中存放的是分段名。如果创建不具有指定分段的分段表，那么在 partition 列中存放的是数据库空间名。sysfragments 系统目录表中的 flags 列标识分段存储方案是否具有指定分段。

可以创建具有指定分段的表或索引，并且可以使用 PARTITION 关键字和分段名来创建、删除和更改指定分段。

要创建具有指定分段的分段表，请使用以下示例的 SQL 语法：

```
CREATE TABLE tb1(a int)
   FRAGMENT BY EXPRESSION
     PARTITION part1 (a >=0 AND a < 5) IN dbspace1,
     PARTITION part2 (a >=5 AND a < 10) IN dbspace1
          ...
       ;
```

如果创建的表或索引分段包含指定分段，那么在使用 ALTER FRAGMENT 语句时必须使用包含分段名的语法，如以下示列所示：

```
ALTER FRAGMENT ON TABLE tb1 INIT FRAGMENT BY EXPRESSION
    PARTITION part_1 (a >=0 AND a < 5) IN dbspace1,
    PARTITION part_2 (a >=5 AND a < 10) IN dbspace1;
```

```
ALTER FRAGMENT ON INDEX ind1 INIT FRAGMENT BY EXPRESSION
    PARTITION part_1 (a >=0 AND a < 5) IN dbspace1,
    PARTITION part_2 (a >=5 AND a < 10) IN dbspace1;
```

可以使用 PARTITION BY EXPRESSION 子句替代在 CREATE TABLE、CREATE INDEX 和 ALTER FRAGMENT ON INDEX 语句中的 FRAGMENT BY EXPRESSION 子句，如以下示例所示：

```
ALTER FRAGMENT ON INDEX idx1 INIT PARTITION BY EXPRESSION
           PARTITION part1   (a <= 10) IN idxdbspc1,
           PARTITION part2   (a <= 20) IN idxdbspc1,
           PARTITION part3   (a <= 30) IN idxdbspc1;
```

使用 ALTER FRAGMENT 语法将不具有指定分段的分段化表及索引更改为具有指定分段的表和索引。以下语法显示了如何将具有多个数据库空间的分段表转换为具有指定分段的分段表：

```
CREATE TABLE t1 (c1 int) FRAGMENT BY EXPRESSION
     (c1=10) IN dbs1,
     (c1=20) IN dbs2;
ALTER FRAGMENT ON TABLE t1 MODIFY dbs2 TO PARTITION part_3 (c1=20)
 IN dbs1
```

以下语法显示了如何将分段索引转换为具有指定分段的索引：

```
CREATE TABLE t1 (c1 int) FRAGMENT BY EXPRESSION
     (c1=10) IN dbs1, (c1=20) IN dbs2, (c1=30) IN dbs3
CREATE INDEX ind1 ON t1 (c1) FRAGMENT BY EXPRESSION
     (c1=10) IN dbs1, (c1=20) IN dbs2, (c1=30) IN dbs3
```

```
ALTER FRAGMENT ON INDEX ind1 INIT FRAGMENT BY EXPRESSION
     PARTITION part_1 (c1=10) IN dbs1, PARTITION part_2 (c1=20) IN dbs1,
     PARTITION part_3 (c1=30) IN dbs1,
```

请参阅《*SinoDB*® 性能指南》以获取有关分段存储的更多信息，包括分段存储指南、分段索引的过程、创建具有指定分段的连接和拆离索引的过程，以及用于创建具有指定分段的连接和拆离索引的 SQL 语句的示例。

请参阅《*SinoDB® SQL* 指南: 语法》以获取更多的语法详细信息，包括 GRANT FRAGMENT 和 REVOKE FRAGMENT 语句中有关指定分段的信息，以及有关使用 DROP、DETACH 和 MODIFY 的详细信息。

创建临时数据库空间

要指定临时文件的分配位置，创建临时数据库空间。

定义临时数据库空间：

1. 使用带有 -c -d -t 选项的 onspaces 实用程序 。

   有关更多信息，请参阅 创建使用缺省页大小的数据库空间 在第173页。
2. 使用 DBSPACETEMP 环境变量，DBSPACETEMP 配置参数指定数据库服务器可用于临时存储的数据库空间。

   DBSPACETEMP 配置参数可包含非缺省页大小的数据库空间。尽管可以在DBSPACETEMP 参数列表中包含不同页大小的数据库空间，数据库服务器却只使用页大小与所列出的第一个数据库空间的页大小相同的数据库空间。

   有关 DBSPACETEMP 的更多信息，请参阅 《*SinoDB®* 管理员参考》中有关配置参数的主题。
3. 如果创建了多个临时数据库空间，那么这些数据库空间必须位于不同磁盘上以优化 I/O。

如果创建了临时数据库空间，那么必须通过设置 DBSPACETEMP 配置参数和/或 DBSPACETEMP 环境变量让数据库服务器知道存在该新建的临时数据库空间。数据库服务器直到您采取了以下两个步骤后才会开始使用临时数据库空间：

- 设置 DBSPACETEMP 配置参数和/或 DBSPACETEMP 环境变量。
- 重启数据库服务器。

以下示例显示了如何创建名为 temp_space 的 5 MB 临时数据库空间，偏移量为 5000 KB：

```
onspaces -c -t -d temp_space -p /dev/raw_dev1 -o 5000 -s 5120
```

有关更多信息，请参阅 临时数据库空间 在第145页。

相关链接

根数据库空间的大小 在第163页

磁盘空间不足时应执行的操作

如果所创建的数据库空间初始块是熟文件（ 在 UNIX™ 上）或 NTFS 文件（在 Windows™ 上），数据库服务器会验证该磁盘空间是否足够用于该初始块。

如果块的大小大于磁盘上的可用空间，将会显示消息且不会创建任何数据库空间。但是，数据库服务器为初始块所创建的熟文件不会移除。它的大小表示在您创建数据库空间之前文件系统上剩馀的空间。移除该文件以回收该空间。

向数据库空间和 BLOB 空间添加块

当数据库空间、BLOB 空间或智能大对象空间将满了或需要更多的磁盘空间时添加块。

> 重要： 新添加的块及其相关联的镜像（如果有的话）立即可用。如果是向镜像存储空间添加块，那么还必须添加镜像块。

使用 onspaces 添加块：

1. 在 UNIX™ 上，必须以用户 informix 或 root 身分登录来添加块。

   在 Windows™ 上，Informix-Admin 组中的用户可添加块。
2. 请确保数据库服务器处于联机、管理或静默模式，或快速恢复模式的清除阶段。
3. 按分配磁盘空间 在第169页所述，为块分配磁盘空间，。
4. 要添加块，请使用 onspaces 的 -a 选项。

   如果存储空间已镜像，则必须指定主块和镜像块的路径名。

如果指定了不正确的路径名、偏移量或大小，那么数据库服务器不会创建块，并且会显示错误消息。另参阅 磁盘空间不足时应执行的操作 在第178页。

5. 在创建块之后，必须对根数据库空间，以及包含该块的数据库空间、BLOB 空间或智能大对象空间执行 0 级备份。

以下示例将 10 MB 镜像块添加到 blobsp3。对主块和镜像块都指定了 200 KB 的偏移量。如果不添加镜像块，那么可省略 -m 选项。

```
onspaces -a blobsp3 -p /dev/raw_dev1 -o 200 -s 10240  -m /dev/raw_dev2 200
```

下一个示例以 5200 KB 的偏移量向 dbspc3 数据库空间添加 5 MB 的原始磁盘空间块。

```
onspaces -a dbspc3 \\.\e:-o 5200 -s 5120
```

还可以定义当应用程序需要更多存储空间时，SinoDB® 可用于自动扩展块大小的信息。如果有可扩展块，就无需添加新块或耗费时间来尝试确定哪个存储空间的空间将耗尽以及何时将耗尽。

相关链接

自动空间管理 在第185页
向智能大对象空间添加块 在第183页

重命名数据库空间

如果您是用户 informix 或具有 DBA 权限，并且数据库服务器处于静默模式（不是任何其他模式），那么可使用 onspaces 实用程序重命名数据库。

要重命名数据库空间，请使用以下 onspaces 实用程序命令：

```
onspaces -ren old_dbspace_name-n new_dbspace_name
```

可重命名标准数据库空间以及所有其他数据库空间，包括 BLOB 空间、智能 BLOB 空间、临时空间，和外部空间。但是，您不能重命名任何关键数据库空间，例如根数据库空间或是包含物理日志的数据库空间。

可重命名数据库空间以及智能大对象空间：

- 当 Enterprise Replication 启用时
- 当数据复制启用时，在主数据库服务器上

在辅助数据库服务器上或当辅助数据库服务器上是 Enterprise Replication 配置的一部分时，不能重命名数据库空间或智能大对象空间。

重命名数据库空间只更改数据库空间的名称，它不会重新组织数据。

重命名数据库空间命令将在所有存储该名称的地方更新数据库空间名称。这包括磁盘上的保留页、系统目录、ONCONFIG 配置文件，和内存中数据结构。

重要： 重命名数据库空间之后，为重命名的数据库空间以及根数据库空间执行 0 级归档。有关更多信息，请参阅 《*SinoDB® 备份和还原指南*》。

重命名数据库空间后可能需要执行的其他操作

如果重命名数据库空间，那么必须重写和重新编译引用旧数据库空间命名的所有存储过程代码。

如果存储过程包含 ALTER FRAGMENT 关键字且引用了该数据库空间名，那么必须重写和重新编译该存储过程。

如果重命名了在 DATASKIP 配置参数中指定的数据库空间，那么必须在重命名该数据库空间后手动更新 DATASKIP 配置参数。

管理自动定位和分段化

可控制数据库服务器能否自动选择数据库、索引和表的位置，以及能否自动切分表。可控制数据库服务器用于存储数据库、索引和表分段的数据库空间列表。

如果启用自动定位和分段化，数据库服务器执行以下任务：

- 如未指定位置的新数据库存储在最佳的数据库空间中，而不是根数据库空间中。缺省情况下，除了租户数据库的数据库空间外，所有其他数据库空间均可用。
- 将未指定位置的新表和索引存储在最佳数据库空间中，而非与数据库相同的数据库空间中。
- 为新表分配循环分段的初始号。除非在 CREATE TABLE 语句中包含 FIRST EXTENT 子句，否则表分段将直到一行插入后才会有扩展数据块。
- 表增长时添加更多表分段。

要启用自动定位和分段化，请将 AUTOLOCATE 配置参数或 AUTOLOCATE 会话环境变量设置为正整数。

自动定位不适用于租户数据库，或租户数据库中的表、分段和索引。

要查看可用数据库空间列表，请查询 sysautolocate 系统目录表。

要添加数据库空间到可用数据库空间列表上，请运行带有 autolocate database 、 autolocate database add 或 autolocate database anywhere 参数的 SQL 管理 API 函数 task() 或 admin()。

要从可用数据库空间列表上移除数据库空间，请运行带有 autolocate database remove 参数的 SQL 管理 API 函数 task() 或 admin()。

要禁用特定数据库中表的自动定位和分段化，请运行带有 autolocate database off 参数的 SQL 管理 API 函数 task() 或 admin()。

要禁用所有数据库中表的自动定位和分段化，请将 AUTOLOCATE 配置参数或 AUTOLOCATE 会话环境变量设置为 0。

相关链接

表分段存储和数据存储 在第161页
存储空间的创建和管理 在第26页
《SinoDB 管理员参考》: AUTOLOCATE 配置参数
《SinoDB 管理员参考》: autolocate database 参数: 指定自动定位和分段的数据库空间 (SQL 管理 API)
《SinoDB 管理员参考》: autolocate database add 参数: 添加数据库空间到数据库空间列表 (SQL 管理 API)
《SinoDB 管理员参考》: autolocate database remove 参数: 从数据库空间列表中移除数据库空间 (SQL 管理 API)
《SinoDB 管理员参考》: autolocate database anywhere 参数: 添加所有数据库空间到数据库列表 (SQL 管理 API)
《SinoDB 管理员参考》: autolocate database off 参数: 禁用数据库的自动分片 (SQL 管理 API)
《SinoDB SQL 指南: 语法》: AUTOLOCATE 会话环境选项
《SinoDB SQL 指南: 参考》: SYSAUTOLOCATE
数据库 在第154页

管理 BLOB 空间

本节说明如何创建 BLOB 空间和确定 BLOB 页的大小。

数据库服务器将 TEXT 和 BYTE 数据存储在数据库空间或 BLOB 空间中，但 BLOB 空间更有效率。有关添加块的的信息，请参阅向数据库空间和 *BLOB* 空间添加块 在第178页。

有关监视 BLOB 空间的信息，请参阅 监视存储空间 在第173页

创建 BLOB 空间

可以使用 onspaces 创建 BLOB 空间。

创建 BLOB 空间之前：

1. 如分配磁盘空间 在第169页中所述，为 BLOB 空间分配磁盘空间。
2. 确定对您环境最佳的 BLOB 页大小。

   有关指示信息，请参阅 确定 *BLOB* 页大小 在第182页。

指定最多为 128 个字节的 BLOB 空间名称。该名称必须唯一的，并且必须以字母或下划线开头。您可以在名称中使用字母、数字、下划线和 $ 字符。

> 重要： 如果数据库服务器启用了镜像，那么您可以在创建 BLOB 空间时对其制作为镜像。镜像可立即生效。

使用 onspaces 创建 BLOB 空间：

1. 要在 UNIX™ 上创建 BLOB 空间，您必须以用户 informix 或 root 身分登录。

   要在 Windows™ 上创建 BLOB 空间，您必须是 Informix-Admin 组的成员。
2. 请确保数据库服务器处于联机、管理或静默模式，或快速恢复方式的清除阶段。
3. 使用 onspaces -c -b 选项，添加 BLOB 空间。
   a) 为 BLOB 空间指定明确的路径名称。如果已对 BLOB 空间制作镜像，那么必须指定主块和镜像块的路径名和大小。
   b) 使用 -o 选项指定 BLOB 空间的偏移量。
   c) 使用 -s 选项指定 BLOB 空间块的大小（以 KB 计）。
   d) 使用 -g 选项根据每个 BLOB 页的磁盘页数指定 BLOB 页大小。

   请参阅 确定 *BLOB* 页大小 在第182页。例如，如果数据库服务器实例的磁盘页大小为 2 KB，而您需要的 BLOB 页大小为 10 KB，那么在该字段中输入 5。

   如果指定了不正确的路径名、偏移量或大小，那么数据库服务器不会创建 BLOB 空间，并且会显示错误消息。另请参阅 磁盘空间不足时应执行的操作 在第178页。
4. 创建了 BLOB 空间之后，必须对根数据库空间以及新 BLOB 空间执行 0 级备份。

以下示例显示了如何在数据库服务器页大小为 2 KB 时，创建 BLOB 页大小为 10 KB 的 10 MB 镜像 BLOB 空间 blobsp3。指定主块和镜像块的偏移量为 200 KB。BLOB 空间从 UNIX™ 上的原始磁盘空间创建。

```
onspaces -c -b blobsp3 -g 5 -p /dev/raw_dev1 -o 200 -s 10240 -m /dev/raw_dev2
 200
```

有关使用 onspaces 创建 BLOB 空间的参考信息，请参阅 《*SinoDB*® 管理员参考》 中关于 onspaces 实用程序的信息。

准备 BLOB 空间以存储 TEXT 和 BYTE 数据

新创建的 BLOB 空间无法立即用于存储 TEXT 和 BYTE 数据。BLOB 空间日志记录和恢复需要在不同的逻辑日志文件中创建用于创建 BLOB 空间的语句，以及用于将 TEXT 和 BYTE 数据插入该 BLOB 空间的语句。

此项要求适用于所有 BLOB 空间，无论数据库的日志记录状态。要符合此要求，请在创建了 BLOB 空间后切换到下一逻辑日志文件。（有关指示信息，请参阅 备份日志文件以释放 *BLOB* 页 在第248页。）

确定 BLOB 页大小

在创建 BLOB 空间时，使用最常出现的简单大对象的大小作为 BLOB 页的大小。换言之，选取浪费最少空间量的 BLOB 页大小。

有关计算最佳 BLOB 页大小的信息，请参阅 《*SinoDB*® 性能指南》中有关配置对 I/O 活动影响的主题中的 BLOB 页大小注意事项。

如果一个表具有多个 TEXT 或 BYTE 列，并且各个对象的大小并不接近，请将每个列存储在不同的 BLOB 空间中，且每个列用适当大小的 BLOB 页来存储。请参阅 表 在第155页。

确定数据库服务器页大小

在指定 BLOB 页大小时，请根据数据库服务器基页大小来指定。

可以使用以下方式之一来确定您系统的数据库服务器页大小：

- 运行 onstat -b 实用程序以显示系统页大小，在输出的最后一行中显示其为缓冲区大小。
- 要查看 PAGE_PZERO 保留页的内容，请运行 oncheck -pr 实用程序。

获取 BLOB 空间存储统计信息

可使用数据库服务器实用程序命令来帮助确定每个 BLOB 空间的最佳 BLOB 页大小。

使用以下数据库服务器实用程序命令来确定最佳的 BLOB 页大小：

- oncheck -pe
- oncheck -pB

oncheck -pe 命令提供有关存储在 BLOB 空间中的对象的背景信息。

- 每个有数据存储在 BLOB 空间块中的表的完整的所有权信息（显示为 *database:owner.table*）
- 每个表用于存储与其相关联的 TEXT 和 BYTE 数据的总页数
- BLOB 空间中的总可用页数和总开销页数

oncheck -pB 命令列出每个表或数据库的以下统计信息：

- 表或数据库在每个 BLOB 空间中使用的 BLOB 页数
- 作为表或数据库的一部分而存储的简单大对象所用的 BLOB 页的平均填充度。

有关更多信息，请参阅 使用 *oncheck -pe* 监视 *BLOB* 空间使用情况 在第203页、使用 *oncheck -pB* 确定 *BLOB* 页充满度 在第203页和《*SinoDB*® 性能指南》中有关表性能注意事项主题中的优化 BLOB 空间 BLOB 页大小的部份。

管理智能大对象空间

本节描述如何创建标准的或临时的智能大对象空间、监视元数据和用户数据区域、向智能大对象空间添加块以及变更智能大对象的存储特征。

有关监视智能大对象空间的信息，请参阅 监视存储空间 在第173页。

创建智能大对象空间

使用 onspaces 实用程序创建智能大对象空间。

使用 onspaces 创建智能大对象空间。

要使用 onspaces 创建智能大对象空间，请执行以下操作：

1. 要在 UNIX™ 上创建智能大对象空间，必须以用户 informix 或 root 身分登录。

   要在 Windows™ 上创建智能大对象空间，您必须是 Informix-Admin 组的成员。
2. 请确保数据库服务器处于联机、管理或静默模式，或快速恢复方式的清除阶段。
3. 使用 onspaces -c -S 选项创建智能大对象空间。

   a) 使用 -p 选项指定路径名、使用 -o 选项指定偏移量，以及使用 -s 选项指定智能大对象空间大小。

b) 如果想要对智能大对象制作镜像，请使用 -m 选项指定镜像路径和偏移量。
c) 如果您想使用智能大对象空间的缺省存储特征，请省略 -Df 选项。

如果您想指定不同的存储特征，请使用 -Df 选项。有关更多信息，请参阅 智能大对象空间的储存特征 在第149页。

d) 智能大对象空间的第一个块必须具有元数据区。

您可以为智能大对象空间指定元数据区域，或让数据库服务器计算元数据区域的大小。有关更多信息，请参阅 计算智能大对象空间元数据的大小 在第183页。

4. 在创建智能大对象空间之后，必须对根数据库空间和新智能大对象空间执行 0 级备份。
5. 要在该智能大对象空间中开始存储智能大对象，请在 SBSPACENAME 配置参数中指定空间名称。
6. 使用 onstat -d、onstat -g smb s，以及 oncheck -cs、-cS、-ps 或 -pS 来显示有关智能大对象空间的信息。

有关更多信息，请参阅 监视智能大对象空间 在第204页。

这显示了如何创建 20 MB 的镜像智能大对象空间 sbsp4。主块和镜像块的偏移量均指定为 500 KB，并且指定偏移量为 200 KB 且大小为 150 KB 的元数据。AVG_LO_SIZE -Df 标记指定期望的平均智能大对象大小为 32 KB。

```
onspaces -c -S sbsp4 -p /dev/rawdev1 -o 500 -s 20480 -m /dev/rawdev2 500
-Ms 150 -Mo 200 -Df "AVG_LO_SIZE=32"
```

有关为创建智能大对象空间和智能大对象缺省选项的信息，请参阅《*SinoDB*® 管理员参考》中有关 onspaces 实用程序的信息。有关创建智能大对象的信息，请参阅《*SinoDB*® *DataBlade*® *API* 程序员指南》和《*SinoDB*® *ESQL/C* 程序员指南》。

计算智能大对象空间元数据的大小

智能大对象空间的第一个块必须有元数据区域。为智能大对象空间正确估算元数据区域的大小以确保智能大对象空间不会耗尽元数据空间，这一点很重要。

当向智能大对象空间添加智能大对象和块时，元数据区域会增大。另外，数据库服务器保留 40% 的用户区域，在元数据区域空间耗尽的情况下使用。

为确保这种情况不会发生，您可以：

- 让数据库服务器为新智能大对象子空间块计算元数据区域的大小。
- 明确指定元数据区域的大小。

有关估计智能大对象空间大小和元数据区域大小的指示信息，请参阅《*SinoDB*® 性能指南》中的表性能的注意事项。另可参阅 监视元数据及用户数据区域 在第208页。

向智能大对象空间添加块

可以向智能大对象空间或临时智能大对象空间添加块。

可以为块指定元数据区域、让数据库服务器计算元数据区域，或将块仅用于用户数据。

要使用 onspaces 向智能大对象空间添加块，请执行以下操作：

1. 请确保数据库服务器处于联机、管理、静默模式或者是快速恢复模式的清理阶段。
2. 使用 onspaces -a 选项创建智能大对象空间块。
   a) 使用 -p 选项指定路径名、使用 -o 选项指定偏移量，以及使用 -s 选项指定块的大小。
   b) 如果想对块制作镜像，请使用 -m 选项指定镜像路径和偏移量。
   c) 要指定元数据空间的大小与偏移量，请使用 -Mo 和 -Ms 选项。

   数据库服务器在新块上分配指定量的元数据区域。
   d) 要让数据库服务器为新块计算元数据的大小，请省略-Mo 和 -Ms 选项。

数据库服务器将智能大对象的估计平均大小除以用户数据区域的大小。

e) 要将块仅用于用户数据，请指定 -U 选项。

如果您使用 -U 选项，那么数据库服务器就不会在该块分配元数据空间。相反地，子空间会使用其他块里的元数据区。

3. 在向智能大对象空间添加块之后，数据库服务器将会写 CHRESERV 和 CHKADJUP 日志记录。
4. 对根数据库空间以及智能大对象空间执行 0 级备份。
5. 使用 onstat -d 和 oncheck -pe 可监视智能大对象空间块中的可用空间量。

此示例向 sbsp4 添加 10 MB 的镜像块。对主块和镜像块都指定 200 KB 的偏移量。如果不添加镜像块，可省略 -m 选项。-U 选项指定新块仅包含用户数据。

```
onspaces -a sbsp4 -p /dev/rawdev1 -o 200 -s 10240 -m /dev/rawdev2 200 -U
```

还可定义当应用程序需要额外存储空间时，SinoDB® 可用于自动扩展块大小的信息。如果有可扩展块，就无需添加新块或耗费时间来尝试决定哪一个存储空间（数据库空间、临时数据库空间、智能大对象空间、临时智能大对象空间或 BLOB 空间）将耗尽以及何时耗尽。

相关链接

监视智能大对象空间 在第204页
自动空间管理 在第185页
向数据库空间和 *BLOB* 空间添加块 在第178页

更改智能大对象的存储特征

使用 onspaces -ch 命令更改智能大对象空间的以下缺省存储特征：

- 扩展数据块大小
- 智能大对象平均大小
- 缓冲模式
- 上次访问时间
- 锁定模式
- 日志记录

有关更多信息，请参阅 智能大对象空间的储存特征 在第149页和《*SinoDB*® 性能指南》中有关表性能注意事项主题中的管理智能大对象空间部分。

创建临时智能大对象空间

有关确定临时智能大对象存储位置的背景信息和规则，请参阅 临时智能大对象空间 在第152页。可以将临时智能大对象存储在标准的或临时的智能大对象空间中。可以向临时智能大对象空间添加或删除块。

要创建具有智能大对象的临时智能大对象空间：

1. 为临时智能大对象空间分配空间。

   有关详细信息，请参阅 分配磁盘空间 在第169页。

   有关 SBSPACETEMP 的信息，请参阅 《*SinoDB*® 管理员参考》中有关配置参数的主题。
2. 如以下示例所示，创建临时智能大对象空间：

   ```
   onspaces -c -S tempsbsp -t -p ./tempsbsp -o 0 -s 1000
   ```
3. 可以指定以下任一 onspaces 选项：

   a) 指定元数据区域及偏移量（-Ms 和 -Mo）。

   b) 指定存储特征（-Df）。

   无法为临时智能大对象空间打开日志记录。

4. 将 SBSPACETEMP 配置参数设置为缺省临时智能大对象空间存储区域的名称。

   重新启动数据库服务器。
5. 使用 onstat -d 显示临时智能大对象空间。

   有关 onstat -d 输出的信息和示例，请参阅《*SinoDB*® 管理员参考》 中的 onstat 实用程序。
6. 在创建智能大对象时，指定 LO_CREATE_TEMP 标记。

   使用 DataBlade® API：

   ```
   mi_lo_specset_flags(lo_spec,LO_CREATE_TEMP);
   ```

   使用 SinoDB® ESQL/C：

   ```
   ifx_lo_specset_flags(lo_spec,LO_CREATE_TEMP);
   ```

有关创建智能大对象的信息，请参阅《*SinoDB® DataBlade® API* 程序员指南》和《*SinoDB® ESQL/C* 程序员指南》。

管理物理日志空间

可使用 onspaces 实用程序或等效的 SQL 管理 API 命令来创建或移动物理日志空间。

要创建物理日志空间，请运行 onspaces -c -P 命令或带有 create plogspace 参数的 SQL 管理 API 函数 admin() 或 task()。

如果想要将物理日志空间的位置更改为不同块，请创建新的物理日志空间。将物理日志移动到新的物理日志空间，并删除旧的物理日志空间。

可以通过以下方式修改物理日志空间中的块：

- 将块标记为不可扩展。运行带有 modify chunk extendable off 参数的 SQL 管理 API 函数 admin() 或 task()。
- 更改块的扩展数据块大小。默认扩展数据块大小是 10000 KB。运行带有 modify space sp_sizes 参数的 SQL 管理 API 函数 admin() 或 task()。

相关链接

物理日志空间 在第153页

《SinoDB 管理员参考》: onspaces -c -P: 创建物理日志空间

《SinoDB 管理员参考》: create plogspace: 创建物理日志空间 *(SQL 管理 API)*

《SinoDB 管理员参考》: modify chunk extendable off 参数*:* 标记块为不可扩展 *(SQL 管理 API)*

《SinoDB 管理员参考》: modify space sp_sizes 参数*:* 更改可扩展的存储空间的大小 *(SQL 管理 API)*

自动空间管理

可将服务设置为需要更多的空间时，自动添加更多存储空间。这样就可以更有效地使用空间，并确保空间是按需分配的。同时减少空间不足的错误，以及减少手动监视空间和确定哪些存储空间将耗尽可用空间所需的时间。将服务器配置为自动添加空间时，还是可以手动扩展空间或块。

当服务器扩展以下存储空间时：数据库空间、临时数据库空间、智能大对象空间、临时智能大对象空间或 BLOB 空间，服务器可向存储空间添加块。服务器还可扩展数据库空间、物理日志空间或临时数据库空间中未镜像的块。如果存储空间是非镜像的数据库空间或临时数据库空间，服务器还可扩展该存储空间中的块。

要对自动或手动空间管理进行配置，请运行 SQL 管理 API 命令或 SinoDB® 开放管理工具 (OAT) 来执行以下任务：

1. 在存储池中创建、修改和删除一个或多个条目。存储池包含 SinoDB® 用于扩展存储空间的可用原始设备、熟文件，和目录的条目。

2. 将块标记为可扩展。
3. 修改存储空间的创建和扩展大小（可选）。
4. 更改自动添加更多空间的阈值和等待时间（可选）。
5. 配置 "监视低存储" 任务的频率（可选）。

如果存储池中包含条目，还可运行 SQL 管理 API 命令以执行以下操作：

- 不想等待自动扩展空间任务运行时，可手动扩展存储空间或块。
- 通过存储池条目手动创建存储空间，并将空存储空间中的空间返还给存储池。

默认情况下，SP_AUTOEXPAND 配置参数设置为 1 以启用存储空间的自动扩展。如果不希望服务器自动扩展存储空间，请将 SP_AUTOEXPAND 配置参数设置为 0 以禁用块的自动创建或扩展。也可指定某个块为不可扩展。

提示:

某些情况下，将服务器配置为自动扩展现有存储空间后，该数据库服务器可能没有自动地扩展 DBSPACETEMP 配置参数中列出的临时数据库空间。如果使用临时数据库空间的操作（例如索引建构或排序）耗尽空间，那么您将收到空间不足的错误。要解决此问题，您必须手动向临时数据库空间添加块，或者使用更大的临时数据库空间。

如果您有存储池，且数据库服务器参与到 Enterprise Replication 中，那么当定义复制服务器时将按需自动创建复制所需的存储空间。

相关链接

存储空间的创建和管理 在第26页
根数据库空间的大小 在第163页
向数据库空间和 *BLOB* 空间添加块 在第178页
向智能大对象空间添加块 在第183页
可扩展块 在第140页
可扩展块 在第140页
存储池 在第164页
《*SinoDB* 管理员参考》: *SP_WAITTIME* 配置参数
存储池 在第164页
创建租户数据库 在第209页

创建和管理存储池条目

可在存储池中添加，修改和删除条目。

存储池中的每个条目都包含有关数据库服务器实例必要时可用于自动扩展现有存储空间的目录、熟文件或原始设备的信息。

创建存储池条目

要创建存储池条目，请运行带有 storagepool add 参数的 admin() 或 task() 函数，如下所示：

```
EXECUTE FUNCTION task("storagepool add", "path", "begin_offset",
"total_size", "chunk size", "priority");
```

指定以下信息：

- 当需要额外存储空间时服务器可使用的文件、目录或设备的路径。
- SinoDB® 可以开始分配空间的设备的偏移量（以 KB 计）。
- 此条目中 SinoDB® 可用的总空间。服务器可从该空间量中分配多个块。
- 可从设备、文件或目录分配的块的最小大小（以 KB 计）。可创建的最小块为 1000 KB。因此，可以指定的最小块大小为 1000 KB。
- 优先级数字，从 1 到 3。（1=高；2=中；3=低）。服务器会尝试先从高优先级的条目分配空间，然后才会从较低优先级的条目分配空间。

存储池的大小和偏移量的缺省单位为 KB。但是，可以按以下示例中显示的任一方式指定信息：

- "100000"
- "100000 K"
- "100 MB"
- "100 GB"
- "100 TB"

修改存储池条目

要修改存储池条目，请运行带有 storagepool modify 参数的 admin() 或 task() 函数，如下所示：

```
EXECUTE FUNCTION task("storagepool modify", "storage_pool_entry_id",
"new_total_size", "new_chunk size", "new_priority");
```

删除存储池条目

要删除存储池条目，请运行带有 storagepool delete 参数的 admin() 或 task() 函数，如下所示：

```
EXECUTE FUNCTION task("storagepool delete", "storage_pool_entry_id");
```

要删除所有的存储池条目，请运行带有 storagepool purge all 参数的 admin() 或 task() 函数，如下所示：

```
EXECUTE FUNCTION task(" storagepool delete", "storage_pool_entry_id");
```

要删除所有已满的存储池条目，请运行带有 storagepool purge full 参数的 admin() 或 task() ，如下所示：

```
EXECUTE FUNCTION task("storagepool purge full");
```

要删除有错误的存储池条目，请运行带有 storagepool purge errors 参数的 admin() 或 task() ，如下所示：

```
EXECUTE FUNCTION task("storagepool purge errors");
```

示例

以下命令添加一个名为 /region2/dbspaces 的目录，其开始偏移量为 0，总大小为 0，初始块大小为 20 MB 且具有高优先级。在此示例中，对目录而言，只能接受起始偏移量为 0 和总大小为 0 这两个条目。

```
EXECUTE FUNCTION task("storagepool add", "/region2/dbspaces", "0", "0",
"20000", "1");
```

以下命令将存储池条目 8 的总大小、块大小和优先级更改为 10 GB、10 MB 和中等优先级。

```
EXECUTE FUNCTION task("storagepool modify", "8", "10 GB", "10000", "2");
```

以下命令删除条目 ID 为 7 的存储池条目。

```
EXECUTE FUNCTION task("storagepool delete", "7");
```

相关链接

存储池 在第164页

《*SinoDB* 管理员参考》*: storagepool purge* 参数*:* 删除存储池条目 *(SQL* 管理 *API)*

《*SinoDB* 管理员参考》*: storagepool modify* 参数*:* 更改存储池条目 *(SQL* 管理 *API)*

《SinoDB 管理员参考》: *storagepool delete* 参数: 删除存储池条目 *(SQL 管理 API)*
存储池 在第164页

将块标记为可扩展或不可扩展

将块标记为可扩展以启用块的自动或手动扩展。您可将标记更改为不可扩展以防止块的自动或手动扩展。

如果块标记为不可扩展：

- 块中只有少量或没有可用空间时，服务器不能自动扩展块。（但是，如果存储池中包含条目，那么服务器可通过向存储空间添加块来扩展存储空间。）
- 您不能手动扩展块的大小。

先决条件：可扩展块必须位于未镜像的数据库空间或临时数据库空间中。

要将块标记为可扩展：

1. 运行带有 modify chunk extendable 参数的 admin() 或 task() 函数，如下所示：
2.
```
EXECUTE FUNCTION task("modify chunk extendable", "chunk number");
```

要将块标记为不可扩展：

1. 运行带有 modify chunk extendable off 参数的 admin() 或 task() 函数，如下所示：

```
EXECUTE FUNCTION task("modify chunk extendable off", "chunk number");
```

以下命令指定块 12 为可扩展。

```
EXECUTE FUNCTION task("modify chunk extendable", "12");
```

相关链接

可扩展块 在第140页
《SinoDB 管理员参考》: *modify chunk extendable* 参数: 标记块为可扩展的 *(SQL 管理 API)*
《SinoDB 管理员参考》: *modify chunk extendable off* 参数: 标记块为不可扩展 *(SQL 管理 API)*

更改可扩展存储空间的大小

可通过修改创建大小和扩展大小来控制存储池中的可扩展存储空间的增长。缺省情况下，可扩展的存储空间的最大大小是不受限制的。可通过设置最大大小来限制可扩展存储空间的大小。

要修改存储空间的创建、扩展或最大的大小：

运行带有 modify space sp_sizes 参数的 SQL 管理 API 函数 admin() 或 task()，如下所示：

```
EXECUTE FUNCTION task("modify space sp_sizes", "space_name",
 "new_create_size", "new_extend_size", "max_size");
```

space_name 是存储空间的名称。

new_create_size 是服务器在指定的数据库空间、临时数据库空间、智能大对象空间、临时智能大对象空间或 BLOB 空间中可创建新块的最小大小。

new_extend_size 是服务器在指定的未镜像数据库空间或临时数据库空间中可扩展块的最小大小。

使用数字（KB 数）或百分比（占总空间的百分比）指定大小。

max_size 为服务器在存储空间中可扩展的最大大小（以 KB 计）。0 值表示无限制。

以下命令将名为 dbspace3 的空间的创建大小设置为 60 MB、扩展大小设置为 10 MB，以及最大大小设置为 200 MB：

```
EXECUTE FUNCTION task("modify space sp_sizes", "dbspace3", "60000",
   "10000", "200000");
```

以下命令将名为 logdbs 的空间的创建大小和扩展大小分别设置为 20% 和 1.5%：

```
EXECUTE FUNCTION task(" modify space
 sp_sizes ", " logdbs ", " 20 ", " 1.5 ");
```

相关链接

《SinoDB 管理员参考》: modify space sp_sizes 参数: 更改可扩展的存储空间的大小 (SQL 管理 API)

更改自动添加更多空间的阈值和等待时间

尽管 SinoDB® 可以在存储空间已满时通过自动扩展或添加块来应对空间不足的情况，而您还可以配置服务器以在存储空间变满之前扩展或添加块。

在存储空间中指定最小可用 KB 量的阈值以触发扩展空间的任务。

还可以使用 SP_WAITTIME 配置参数指定线程返回空间不足错误之前等待空间扩展的最大秒数。

要更改阈值和等待时间：

1. 将 SP_THRESHOLD 配置参数中指定的阈值从零（禁用）更改为非零值 。指定 1 到 50 的百分比数，或指定 1000 到块最大大小（以 KB 计）。
2. 更改 SP_WAITTIME 配置参数的值，该配置参数指定线程返回空间不足错误之前等待空间扩展的最大秒数。

相关链接

《SinoDB 管理员参考》: SP_THRESHOLD 配置参数

《SinoDB 管理员参考》: SP_WAITTIME 配置参数

配置监视低存储任务的频率

可更改 mon_low_storage 任务的频率，该任务定期扫描数据库空间列表，以查找低于 SP_THRESHOLD 配置参数指示的阈值的空间。如果任务发现了低于阈值的空间，将尝试通过扩展可扩展块或使用存储池添加块的方式来扩展空间。

mon_low_storage 任务的缺省频率是每小时一次，但是可配置该任务以增加或减少运行频率。

先决条件：在 SP_THRESHOLD 配置参数中指定存储空间最小可用 KB 量。

要将 mon_low_storage 任务配置为以更高或更低的频率运行：

运行以下 SQL 语句，其中 *minutes* 是两次运行之间的分钟数：

```
DATABASE sysadmin;
UPDATE ph_task set tk_frequency = INTERVAL (minutes)
 MINUTE TO MINUTE WHERE tk_name = "mon_low_storage";
```

例如，要将任务配置为每 10 分钟运行一次，请执行以下 SQL 语句：

```
DATABASE sysadmin;
UPDATE ph_task set tk_frequency = INTERVAL (10) MINUTE TO MINUTE
 WHERE tk_name = "mon_low_storage";
```

手动扩展空间或可扩展块

必要时可手动扩展空间或块，而无须等待 SinoDB® 自动扩展空间或块。

先决条件：

- 仅可扩展位于非镜像数据库空间或临时数据库空间中的块。
- 扩展块之前需先将其标记为可扩展。否则，必须运行带有 modify chunk extendable 参数的 admin() 或 task() 函数以指定该块为可扩展的。
- 如果不能通过扩展块来扩展空间，那么存储池中必须包含服务器可用于创建新块的活动条目。

要立即增加存储空间：

执行以下任一操作：

- 通过运行带有 modify space expand 参数的 admin() 或 task() 函数手动扩展空间，如下所示：

```
EXECUTE FUNCTION task("modify space expand", "space_name", "size");
```

例如，以下命令将编号 8 的空间扩展 1GB：

```
EXECUTE FUNCTION task("modify space expand", "8", "1000000");
```

服务器通过扩展空间中的块或添加新块来扩展空间。服务器可能对于请求的大小项上取整，其取决于存储空间的页大小和扩展过程中任何存储池条目所使用的配置块大小。

- 运行带有 modify chunk extend 参数的 admin() 或 task() 函数手动扩展块，如下所示：

```
EXECUTE FUNCTION task("modify chunk extend", "chunk_number", "extend_amount");
```

例如，以下命令将编号 12 的块扩展 5000 KB：

```
>EXECUTE FUNCTION task("modify chunk extend", "12", "5000");
```

服务器可能会将请求的大小向上取整，取决于存储空间的页大小。

相关链接

可扩展块 在第140页

《SinoDB 管理员参考》: modify space expand 参数: 扩大空间的大小 (SQL 管理 API)

《SinoDB 管理员参考》: modify chunk extend 参数: 扩展块的大小 (SQL 管理 API)

自动添加更多空间的最低配置和测试的示例

此示例显示可如何最低程度配置和测试自动添加更多空间。可通过创建数据库空间、填充空间、向 SinoDB® 存储池添加条目和向空间装载表来执行此操作。当空间填满时，SinoDB® 将自动扩展该空间。

要最低程度配置和测试自动添加更多空间：

1. 创建数据库空间

   例如，创建名为 expandable_dbs 的数据库空间，并将名为 /my_directory/my_chunk 的熟文件的前 10000 KB 分配给初始块，如下所示：

   ```
   onspaces -c -d expandable_dbs -p /my_directory/my_chunk -o 0 -s 10000
   ```

2. 填充数据库空间。

   例如，在不装载数据行的情况下填充数据库空间。而是，创建表并为第一个扩展数据块分配一大组连续的可用页，如下所示：

   ```
   CREATE TABLE large_tab (col1 int) IN expandable_dbs EXTENT SIZE 10000000;
   ```

   可使用 onstat -d 命令或 SinoDB® 开放管理工具 (OAT) 来监视块中的可用页。 如果数据库空间已满，那么在尝试创建数据并将其装载到新表时将会收到空间不足的错误。

3. 向 SinoDB® 存储池添加条目。

   例如，将 $INFORMIXDIR/tmp 目录添加到存储池中，如下所示：

   ```
   DATABASE sysadmin;
   EXECUTE FUNCTION task("storagepool add", "$INFORMIXDIR/tmp",
    "0", "0", "10000", "2");
   ```

4. 在 SP_THRESHOLD 配置参数中，设置在 SinoDB® 自动运行任务以扩展空间之前，存在存储空间中最小可用 KB 量的阈值。
5. 创建新表并将其加载到数据库。

   如果存储空间现在变满，不会再收到空间不足错误，而是 SinoDB® 会自动在 $INFORMIXDIR/tmp 文件中创建熟文件并使用新的熟文件向 expandable_dbs 数据库添加块。如果继续填充此块，服务器将自动对其进行扩展。向数据库空间添加新块之前，如果可能，服务器将始终扩展块。
6. 减少存储空间中的可用空间，以测试 SP_THRESHOLD 配置参数中的值。

   在存储空间中分配足够的页来减少自由空间，以使得其低于 SP_THRESHOLD 指示的阈值。但是，不要完全填满该空间。

   下次 mon_low_storage 任务运行时，您必须确保看到开空间自动扩展。
7. 创建空间不足的条件。

   分配存储空间中的所有页。然后尝试分配更多的页。必须确保分配成功，且您不会收到空间不足的错误。

   SinoDB® 每次扩展或添加块，以及将新块标记为可扩展时，都会向日志写入消息。

   运行 onstat -d 命令显示实例中的所有块。查找带有 E 标志的可扩展块。此命令的输出显示服务器通过添加新块或扩展现有块的大小自动扩展空间。

自动添加更多空间的配置示例

此示例显示如何通过以下方式对自动添加更多空间进行充分配置：更改某些配置参数设置、更改用于监视低存储的任务的频率，以及指定可扩展空间和块的信息。

为自动添加更多空间配置：

1. 向存储池添加条目。

   例如，将 $INFORMIXDIR/tmp 目录添加到存储池中，如下所示：

   ```
   DATABASE sysadmin;
   EXECUTE FUNCTION task("storagepool add", "$INFORMIXDIR/tmp",
    "0", "0", "10000", "2");
   ```

2. 将未镜像数据库空间和临时数据库空间中的一些块标记为可扩展，以便将来必要时，服务器可扩展这些块。

   例如，指定第 12 个块为可扩展：

   ```
   EXECUTE FUNCTION task("modify chunk extendable", "12");
   ```

   也可将可扩展块的标记更改为不可扩展。 例如，指定第 10 个块为不可扩展：

   ```
   EXECUTE FUNCTION task("modify chunk extendable off", "10");
   ```

3. 在 SP_THRESHOLD 配置参数中，设置在 SinoDB® 自动运行任务以扩展空间之前，存在存储空间中最小可用 KB 量的阈值。指定以下任一项：

   - 1 到 50 的百分比值，
   - 1000 到块的最大大小（以 KB 计）。

   如果单个存储空间的填充程度超过了您定义的阈值而且持续到下次空间监视任务（mon_low_storage）运行时都是如此，那么服务器将通过扩展可扩展块或通过使用存储池添加块来尝试扩展空间。

例如，假设 SP_THRESHOLD 值为 5.5，服务器会将该值是为 5.5%。如果某个空间中的可用页很少，且可用空间百分比低于 5.5%，并持续到下次 mon_low_storage 任务执行时仍是如此，那么该任务将会尝试扩展空间。如果 SP_THRESHOLD 值设为 50000，并且某个空间的可用空间低于 50000 KB，那么在下次 mon_low_storage 运行时将扩展该空间。

4. 可选： 更改 mon_low_storage 的运行频率。该任务定期扫描数据库空间列表以查找低于 SP_THRESHOLD 配置参数指示阈值的空间。

   例如，要配置该任务每 10 分钟跑一次，请执行以下 SQL 语句：

   ```
   DATABASE sysadmin;
   UPDATE ph_task set tk_frequency = INTERVAL (10) MINUTE TO MINUTE
    WHERE tk_name = "mon_low_storage";
   ```

5. 可选： 更改 SP_WAITTIME 配置参数的值，该配置参数指定在线程返回空间不足错误之前，线程等待空间扩展的最大秒数。
6. 可选： 更改与扩展存储空间相关的两个大小：
   - 扩展大小，这是在数据库空间、临时数据库空间或物理日志空间中扩展块时使用的最小大小。
   - 创建大小，这是在非镜像的数据库空间、临时数据库空间、智能大对象空间、临时智能大对象空间或 BLOB 空间中创建新块时使用的最小大小。

     例如，以下命令将编号 3 的空间的创建大小和扩展大小分别设置为 60 MB 和 10 MB。

     ```
     EXECUTE FUNCTION task("modify dbspace sp_sizes",
      "3", "60000", "10000");
     ```

在配置了自动扩展存储空间之后，还可必要时手动扩展空间或扩展空间中的块。

删除块

使用 onspaces 实用程序从数据库空间中删除块。

在删除块之前，请使用下表作为指导方针，确保数据库服务器处于正确的模式。

表 28: 删除块的数据库服务器模式

块类型	数据库服务器为联机模式	数据库服务器为管理或静默模式	数据库服务器为脱机模式
数据库空间块	是	是	否
临时数据库空间块	是	是	否
BLOB 空间块	否	是	否
智能大对象空间或临时智能大对象空间块	是	是	否

验证块是否为空

要使用这些实用程序从数据库空间成功删除块，块必须不包含任何数据。除开销页外的所有页必须全部释放。

如果有任何页仍然分配给非开销的实体，那么实用程序将返回以下错误：Chunk is not empty.

此外，当数据库空间由两个或多个块组成，且附加块不包含用户数据时，如果附加块包含表空间 tblspace 则这些块无法删除。

如果收到 Chunk is not empty 消息，必须通过运行 oncheck -pe 来列出扩展数据块的内容，以确定哪个表或其他实体仍占用该块的空间。

通常，当你删除拥有这些页的表时就能除去这些页。然后重新输入实用程序指令。

使用 onspaces 从数据库空间删除块

以下示例在 UNIX™ 平台上从 dbsp3 中删除块。指定 300 KB 的偏移量。

```
onspaces -d dbsp3 -p /dev/raw_dev1 -o 300
```

不能使用前面示例中的语法来删除数据库空间的初始块。而是，必须删除数据库空间。使用 onstat -d 的 fchunk 列来确定哪个块是数据库空间的初始块。有关 onstat 的更多信息，请参阅 《*SinoDB*® 管理员参考》 中有关 onspaces 实用程序的信息。

有关使用 onspaces 从数据库空间删除块的信息，请参阅 《*SinoDB*® 管理员参考》。

从 BLOB 空间删除块

从 BLOB 空间删除块的过程与使用 *onspaces* 从数据库空间删除块 在第193页中所述的从数据库空间删除块的过程相同，除了数据库服务器必须处于静默或管理模式外，还必须将任何出现对数据库空间的引用替换为 BLOB 空间的名称。

使用 onspaces 从智能大对象空间删除块

以下示例在 UNIX™ 平台上从 sbsp3 删除块。指定 300 KB 的偏移量。当从智能大对象空间或临时智能大对象空间删除块时，数据库服务器必须处于联机管理或静默模式。

```
onspaces -d sbsp3 -p /dev/raw_dev1 -o 300
```

不能使用前面示例的语法来删除智能大对象空间的初始块。而是，必须删除智能大对象空间。使用 onstat -d 的 fchunk 列来确定哪个块是智能大对象空间 的初始块。

-f (force) 选项

可使用 onspaces 的 -f 选项来删除其中没有分配元数据的删除智能大对象空间块。如果块包含智能大对象空间的元数据，那么必须删除整个智能大对象空间。使用 onstat -d 的 块 部分来确定那个智能大对象空间包含元数据。

```
onspaces -d sbsp3 -f
```

警告： 如果强制删除智能大对象空间，可能会导致表和智能大对象空间之间的一致性问题。

删除不带任何指针的智能大对象

每个智能大对象都有引用计数，即智能大对象指针数。当引用计数大于 0 时，数据库服务器假定该智能大对象正在使用中，就不会删除它。

引用计数为 0 的智能大对象很少会保留下来。可使用 onspaces -cl 命令删除所有引用计数为 0 的智能大对象（如果未被任何应用程序打开）。

有关使用 onspaces -cl 的信息，请参阅 《*SinoDB*® 管理员参考》 中有关 onspaces 实用程序的信息。

删除存储空间

使用 onspaces 删除数据库空间、临时数据库空间、BLOB 空间、智能大对象空间、临时智能大对象空间或外部空间。

在 UNIX™ 上，您必须以 root 或 informix 的身份登录才能删除存储空间。在 Windows™ 上，您必须是 Informix-Admin 组中的成员才能删除存储空间。

仅当数据库服务器处于联机、管理或静默模式时，才可以删除存储空间。

删除存储空间的准备工作

在删除数据库空间之前，您必须先删除先前在该数据库空间中创建的所有数据库和表。您不能删除根数据库空间 。

在删除 BLOB 空间之前，必须先删除具有引用该 BLOB 空间的 TEXT 或 BYTE 列的所有表。

运行 oncheck -pe 以验证没有任何表或日志文件位于数据库空间或 BLOB 空间 中。

在删除智能大对象空间之前，必须先删除具有引用该智能大对象空间中对象的 CLOB 或BLOB 列的所有表。对于智能大对象空间，您无需删除指向智能大对象空间的列，但是这些列必须为 null；也就是说，所有的智能大对象必须取消智能大对象空间的分配。

> 提示： 如果在发生轻量级追加的数据库空间中删除表 ，这些轻量级追加可能会比您期望的要慢。此问题的症状就是物理日志记录活动。如果轻量级追加比您预期的要慢，请确保在轻量级追加之前或期间没有表从数据库空间中删除。如果删除了表，那么在执行轻量级追加之前请使用 onmode -c 强制执行检查点。
>
> 重要： 删除块或数据库空间将触发阻塞检查点，将强制所有数据库更新等待所有缓冲池清空到磁盘之后才执行。此更新阻塞在阻塞检查点期间所用时间会比非阻塞检查点期间所用时间长的多，特别是在缓冲池很大的情况下。

删除镜像存储空间

如果删除已镜像的存储空间，那么镜像空间也会被删除。

如果希望只删除存储空间的镜像，请关闭镜像。（请参阅 结束镜像 在第281页。）此操作删除数据库空间、BLOB 空间或智能大对象空间的镜像，并释放块以作其他用途。

使用 onspaces 删除存储空间

要使用 onspaces 删除存储空间，请按以下示例使用 -d 选项。

此示例删除名为 dbspce5 的数据库空间及其镜像。

```
onspaces -d dbspce5
```

此示例删除名为 blobsp3 的数据库空间及其镜像。

```
onspaces -d blobsp3
```

如果要删除包含数据的智能大对象空间，请将 -d 选项和 -f 选项一起使用。如果省略 -f 选项，就无法删除包含数据的智能大对象空间。此示例删除名为 sbspc4 的数据库空间及其镜像。

```
onspaces -d sbspc4
```

> 警告： 如果使用 -f 选项， 那么数据库服务器中的表可能包含指向已删除的智能大对象的无效指针。

有关使用 onspaces 删除存储空间的信息，请参阅《*SinoDB*® 管理员参考》中有关 onspaces 实用程序的信息。

删除存储空间后备份

如果创建了名称与已删除的存储空间相同的存储空间，请执行 0 级备份以确保以后的复原不会将新存储空间与旧存储空间混淆。有关更多信息，请参阅《*SinoDB*® 备份和还原指南》。

> 重要： 在删除数据库空间、BLOB 空间或智能大对象空间之 后，新释放的块可用于重新分配到其他数据库空间、BLOB 空间或智能大对象空间。但是，在重新分配新释放的块之前，必须对根数据库空间和已修改过的存储空间执行 0 级备份。如果未执行此备份，而随后必须执行复原，那么复原可能会失败，因为备份保留页不是最新的。

从存储池创建空间或块

如果存储池中包含条目，那么可从存储池的可用空间中创建存储空间或块。

先决条件：存储池必须包含条目（目录、熟文件或原始设备）。

要从存储池创建存储空间或块：

运行带有以下参数之一的 admin() 或 task() 函数，以从存储池创建空间。命令中使用的元素取决于要创建的空间类型。

- ```
  EXECUTE FUNCTION task("create dbspace from storagepool", "space_name",
   "size", "page_size", "mirroring_flag", "first_extent", "next_extent");
  ```
- ```
  EXECUTE FUNCTION task("create tempdbspace from storagepool",
   "space_name", "size", "page_size");
  ```
- ```
  EXECUTE FUNCTION task("create blobspace from storagepool", "space_name",
   "size", "page_size", "mirroring_flag",);
  ```
- ```
  EXECUTE FUNCTION task("create sbspace from storagepool", "space_name",
   "size", "log_flag", "mirroring_flag",);
  ```
- ```
  EXECUTE FUNCTION task("create tempsbspace from storagepool",
   "space_name", "size",);
  ```
- ```
  EXECUTE FUNCTION task("create chunk from storagepool",
   "space_name", "size",);
  ```

示例

以下命令创建名为 blobspace1 的镜像 BLOB 空间。新建的 BLOB 空间的大小为 100 GB，BLOB 页大小为 100 页。

```
EXECUTE FUNCTION task("create blobspace from storagepool", "blobspace1", "100
 GB",
 "100", "1");
```

以下命令向名为 logdbs 的数据库空间添加块。新块的大小为 200 MB。

```
EXECUTE FUNCTION task("create chunk from storagepool", "logdbs", "200 MB");
```

相关链接

《SinoDB 管理员参考》: create dbspace from storagepool 参数: 从存储池创建数据库空间 (SQL 管理 API)

《SinoDB 管理员参考》: create tempdbspace from storagepool 参数: 从存储池创建临时数据库空间 (SQL 管理 API)

《SinoDB 管理员参考》: create blobspace from storagepool 参数: 从存储池创建 BLOB 空间 (SQL 管理 API)

《SinoDB 管理员参考》: create sbspace from storagepool 参数: 从存储池创建智能大对象空间 (SQL 管理 API)

《SinoDB 管理员参考》: create tempsbspace from storagepool 参数: 从存储池创建临时智能大对象空间 (SQL 管理 API)

《SinoDB 管理员参考》: create chunk from storagepool 参数: 从存储池创建块 (SQL 管理 API)

将空的空间返还给存储池

可将空块或空存储空间中的空间返还给存储池。

要将空的块、数据库空间、临时数据库空间、BLOB 空间、智能大对象空间或临时智能大对象空间中的存储空间返还给存储池：

运行带有以下参数之一的 admin() 或 task() 函数，将空间返还给存储池。命令中使用的元素取决于要删除的对象类型。

- ```
  EXECUTE FUNCTION task("drop chunk to storagepool", "space_name",
   "chunk_path", "chunk_offset")
  ```
- ```
  EXECUTE FUNCTION task("drop dbspace to storagepool", "space_name");
  ```
- ```
  EXECUTE FUNCTION task("drop tempdbspace to storagepool", "space_name");
  ```
- ```
  EXECUTE FUNCTION task("drop blobspace to storagepool", "space_name");
  ```
- ```
  EXECUTE FUNCTION task("drop sbspace to storagepool", "space_name");
  ```
- ```
  EXECUTE FUNCTION task("drop tempsbspace to storagepool", "space_name");
  ```

示例

以下命令删除名为 blob4 的空 BLOB 空间，并将释放的所有空间添加到存储池。

```
EXECUTE FUNCTION task("drop blobspace to storagepool", "blob4");
```

以下命令删除名为 health 的数据库空间中的空块，并将释放的所有空间添加到存储池。

```
EXECUTE FUNCTION task("drop chunk to storagepool", "health",
 "/health/rawdisk23", "100 KB");
```

相关链接

《SinoDB 管理员参考》: drop chunk to storagepool 参数: 将空块的空间归还到存储池 (SQL 管理 API)

《SinoDB 管理员参考》: drop dbspace to storagepool 参数: 从空数据库空间归还空间到存储池 (SQL 管理 API)

《SinoDB 管理员参考》: drop tempdbspace to storagepool 参数: 从空临时数据库空间归还空间到存储池 (SQL 管理 API)

《SinoDB 管理员参考》: drop blobspace to storagepool 参数: 从空 BLOB 空间归还空间到存储池 (SQL 管理 API)

《SinoDB 管理员参考》: drop sbspace to storagepool 参数: 从空智能大对象空间归还空间到存储池 (SQL 管理 API)

《SinoDB 管理员参考》: drop tempsbspace to storagepool 参数: 从空临时智能大对象空间归还空间到存储池 (SQL 管理 API)

管理外部空间

外部空间不需要分配磁盘空间。可使用 onspaces 实用程序创建和删除外部空间。有關外部空间的更多信息，請參阅 外部空间 在第154页。

创建外部空间

可使用 onspaces 实用程序创建外部空间。但是，首先必须有一个有效的数据源和可用来访问该数据源的有效访问方法。虽然您可以创建不具有效访问方法或有效数据源的外部空间，但对该外部空间的任何检索数据尝试都将生成错误。有关访问方法的信息，请参阅《*SinoDB*® 虚表接口程序员指南》。

要使用 onspaces 创建外部空间，请按以下示例使用 -c 选项。以下示例显示如何创建与 UNIX™ 密码文件相关联的外部空间 pass_space。

```
onspaces -c -x pass_space -l /etc/passwd
```

指定最多 128 字节的外部空间名称。该名称必须是唯一的，并以字母或下划线开始。在名称中可使用字母、数字、下划线和 $ 字符。

> 重要: 前面的示例假设您已经编写了提供正确访问文件 passwd 的函数的例程，并且假定该文件本身已存在。在创建外部空间之后，必须使用适当命令以允许访问文件 passwd 中的数据。有关用户定义的访问方法的更多信息，请参阅《*SinoDB*® 虚表接口程序员指南》。

有关使用 onspaces 创建外部空间的参考信息，请参阅《*SinoDB*® 管理员参考》中有关 onspaces 实用程序的信息。

删除外部空间

要使用 onspaces 删除外部空间，请按以下示例使用 -d 选项。如果外部空间与现有的表或索引相关联则不能删除.

此示例删除名为 pass_space 的外部空间。

```
onspaces -d pass_space
```

跳过无法访问的分段

分段存储的一个好处是可以在 I/O 操作期间跳过不可用的表分段。例如，即使分段位于当前因磁盘故障而脱机的块上时，查询仍可继续进行。发生这种情况时，磁盘故障仅影响分段表中的部分数据。想较之下，对于未分段的表，如果它们存在于故障磁盘上时将变成完全无法访问。

此功能的控制如下所示：

- 由数据库服务器管理员使用 DATASKIP 配置参数
- 由各个应用程序使用 SET DATASKIP 语句。

DATASKIP 配置参数

可将 DATASKIP 参数设置为 OFF、ALL 或 ON *dbspace_list*。OFF 表示数据库服务器不跳过任何分段。如果某个分段不可用，查询将返回错误。ALL 表示跳过任何不可用的分段。ON *dbspace_list* 指示数据库服务器跳过任何位于指定的数据库空间中的分段。

onspaces 的数据跳过功能

使用 onspaces 实用程序的数据跳过功能来指定在不可用时要跳过的数据库空间。例如，以下命令设置 DATASKIP 参数，使数据库服务器跳过 dbspace1 和 dbspace3 中的分段，但不跳过 dbspace2 中的分段：

```
onspaces -f ON dbspace1 dbspace3
```

有关 onspaces 选项的完整语法，请参阅《*SinoDB*® 管理员参考》中有关 onspaces 实用程序的信息。

使用 onstat 检查数据跳过状态

使用 onstat 实用程序列出当前受数据跳过功能影响的数据库空间。-f 选项列出使用 DATASKIP 配置参数设置的数据库空间和使用 onspaces 实用程序的 -f 选项设置的数据库空间。

当运行 onstat -f 时，您会收到消息，通知您 DATASKIP 配置参数是对所有数据库空间都设置为 on、对所有数据库空间设置为 off，还是特定数据库空间设置为 on。

SQL 语句 SET DATASKIP

应用程序可使用 SQL 语句 SET DATASKIP 来控制在某个分段不可用时是否跳过该分段。应用程序必须仅在有限情况下包含此语句，因为它会使查询根据底层分段的可用性而返回不同的结果。与配置参数 DATASKIP 类似，SET DATASKIP 语句接受数据库空间列表向数据库服务器指示要跳过哪些数据库空间的分段。例如，假设某个应用程序的编程人员将以下语句包含在应用程序的开头：

```
SET DATASKIP ON dbspace1, dbspace5
```

该语句将使数据库服务器在以下两个条件都满足时跳过 dbspace1 或 dbspace5：

- 应用程序尝试访问其中一个数据库空间。
- 数据库服务器发现其中一个数据库空间为不可用。

如果数据库服务器发现 dbspace1 和 dbspace5 都不可用，它会跳过这两个数据库空间 。

数据库服务器管理员可使用 SET DATASKIP 语句的 DEFAULT 设置来控制数据跳过功能。假设应用程序开发者将以下语句包含在应用程序中：

```
SET DATASKIP DEFAULT
```

当在此 SQL 语句之后运行查询时，数据库服务器会检查配置参数 DATASKIP 的值。一旦数据库服务器管理员意识到一个或多个数据库空间不可用时，数据库服务器管理员可以鼓励用户使用此设置来指定要跳过的数据库空间。

数据跳过功能对事务的影响

如果打开数据跳过功能，那么 SELECT 语句将始终执行。此外，如果表由循环分段且至少一个分段联机，那么 INSERT 语句将始终成功执行。但是，当写入数据库的操作可能会影响数据库的完整性时，数据库服务器不会完成这类操作。以下操作会失败：

- 数据库服务器无法消除脱机的分段的所有 UPDATE 和 DELETE 操作

  如果数据库服务器可消除脱机的分段，那么更新或删除就会成功，但结果与 DATASKIP 设置无关。
- 根据基于表达式的分布方案已分段的表，其 INSERT 操作的相应分段已脱机
- 任何涉及引用约束检查的操作，且该约束涉及脱机的分段中的数据

  例如，如果应用程序删除具有子行的行，子行也必须可用于删除。
- 任何影响索引值的操作（例如，更新已建立索引的列），且该索引位于脱机的块中。

确定何时使用数据跳过

要少用该功能且谨慎使用，因为该结果始终是不可靠的。在以下情况中可考虑使用：

- 可接受事务完整性受影响。
- 可确定事务完整性不会受影响。

后一任务可能困难且耗时。

确定何时跳过选择的分段

在某些情况下，可能希望数据库服务器跳过某些分段，而不是其他的分段。这通常发生在以下情况中：

- 跳过对于查询结果没有显著作用的分段。

- 某些分段脱机，且您断定跳过这些分段并返回有限数量的数据比取消查询要好。

当要跳过分段时，请使用 ON *dbspace-list* 设置来指定包含数据库服务器必须跳过的分段的数据库空间的列表。

确定何时跳过所有分段

将 DATASKIP 配置参数设置为 ALL 将使数据库服务器跳过所有不可用的分段。请谨慎使用此选项。如果数据库空间变为不可用，那么在执行应用程序发出的所有查询之前未发出 SET DATASKIP OFF 语句的话将发生错误。

监视分段存储使用

数据库管理员可能会发现分段存储的以下方面对监视有用：

- 分段上的数据分布
- 分段上的 I/O 请求平衡
- 包含分段的块的状态

管理员可以监视表分段上的数据分布。如果分段存储的目的是提升管理响应时间，那么数据要均匀地分布在分段上是很重要的。要监视分段存储磁盘使用，您必须监视数据库服务器表空间，因为分段的磁盘存储单位是表空间。（有关如何监视分段表的数据分布的信息，请参阅 监视表空间和扩展数据块 在第202页。）

管理员必须监视包含在分段中的数据的 I/O请求队列。当 I/O 队列变得不平衡时，管理员必须与 DBA 一起来调整分段存储策略。（有关如何监视块使用的说明，包括监视每个块的 I/O 队列，请参阅 监视块 在第199页。）

管理员必须监视分段的可用性，并在包含一个或多个分段的数据库空间发生故障时采取适当的步骤。有关如何确定块是否脱机，请参阅 监视块 在第199页。

显示数据库

可以显示通过 SMI 表所创建的数据库。

相关链接

数据库 在第154页

SMI 表

查询 sysdatabases 表以显示由数据库服务器管理的每个数据库，一行一个数据库。有关此表中的列的描述，请参阅《*SinoDB*® 管理员参考》中有关 sysmaster 数据库的主题中的 sysdatabases 信息。

监视磁盘使用

这些主题描述跟踪各种数据库服务器存储单元所使用磁盘空间的方法。

有关本节中提及的内部数据库服务器存储单元的背景信息，请参阅《*SinoDB*® 管理员参考》中有关磁盘结构和存储的章节。

相关链接

控制简单大对象数据的存储位置 在第144页

监视块

可以监视块的以下信息：

- 块大小
- 可用页数
- 块中的表

可以使用此信息来跟踪块所使用的磁盘空间、监视块 I/O 活动和检查分段存储。

onstat -d 实用程序

onstat -d 实用程序列出所有数据库空间、BLOB 空间和智能大对象空间，以及以下有关这些空间中的块的信息。

- 块地址
- 块号和相关联的数据库空间号
- 设备中的偏移量（以页计）
- 块大小（以页计）
- 块中的可用页数
- 物理设备的路径名

如果在具有 BLOB 空间块的实例上发出 onstat -d 命令，那么所显示的可用页数是过时的。在 free 值之前的波浪号 (~) 表示该数值为近似值。在备份其中记录了删除的逻辑日志且释放 BLOB 页之后，onstat -d 命令才会将 BLOB 页注册为可用页。因此，如果您删除 25 个简单大对象并立即执行 onstat -d，那么 onstat 的输出中将不会包含新释放的空间。

要获取 BLOB 空间块中准确的可用 BLOB 页数，可发出 onstat -d update 命令。有关详细信息，请参阅 *onstat -d* 更新选项 在第200页。

在 onstat -d update 输出中，chunk 部分的 flags 列提供以下信息：

- 块是主块或镜像块
- 块是联机、脱机、恢复中或是新块

有关 onstat -d 输出的示例，请参阅《*SinoDB*® 管理员参考》中有关 onstat 实用程序的信息。

> 重要： 在镜像可活动之前和关闭镜像之后，必须对根数据库空间和已修改的数据库空间执行 0 级备份。

onstat -d 更新选项

onstat -d update 选项显示与 onstat -d 所显示信息相同的信息，以及显示每个 BLOB 空间块中准确的可用 BLOB 页数。

onstat -D 选项

onstat -D 选项显示与 onstat -d 所显示信息相同的信息，另外还显示从块中读取的页数（在 page Rd 字段中）。

使用 onstat -g iof 命令监视块 I/O 活动
使用 onstat -g iof 命令监视块 I/O 活动，包括对分段表的不同分段上的 I/O 请求的分布。

onstat -g iof 命令显示：

- 每个块的读取数和写入数
- 显示单个操作的各服务级别的 I/O
- 操作类型
- 操作发生的次数
- 完成操作所用平均时间

如果某个块有不成比例的 I/O 活动量，那么该块可能会成为系统的瓶颈。

有关 onstat -g iof 输出的实例，请参阅《*SinoDB*® 管理员参考》中有关 onstat 实用程序的信息。

oncheck -pr 命令

数据库服务器将块信息存储在保留页 PAGE_1PCHUNK 和 PAGE_2PCHUNK。

要列出保留页的内容，请运行 oncheck -pr。以下示例显示 oncheck -pr 的输出样本。此输出基本上与 onstat -d 的输出是相同的；但如果块信息自上一个检查点之后更改过，那么在 oncheck -pr 输出中将不会包含这些更改。

```
Validating PAGE_1DBSP & PAGE_2DBSP...
          Using dbspace page PAGE_2DBSP.

    DBspace number                1
    DBspace name                  rootdbs
    Flags                         0x20001          No mirror chunks
    Number of chunks              2
    First chunk                   1
    Date/Time created             07/28/2008 14:46:55
    Partition table page number   14
    Logical Log Unique Id         0
    Logical Log Position          0
    Oldest Logical Log Unique Id  0
    Last Logical Log Unique Id    0
    Dbspace archive status        No archives have occurred
.
.
Validating PAGE_1PCHUNK & PAGE_2PCHUNK...
          Using primary chunk page PAGE_2PCHUNK.

    Chunk number                  1
    Flags                         0x40       Chunk is online
    Chunk path                    /home/server/root_chunk
    Chunk offset                  0 (p)
    Chunk size                    75000 (p)
    Number of free pages          40502
    DBSpace number                1
.
.
.
```

oncheck -pe 命令

要获取块中信息的物理布局，请运行 oncheck -pe。将列出数据库空间、BLOB 空间和智能大对象空间。以下示例显示 oncheck -pe 的输出样本。

将显示以下信息 ：

- 数据库空间的名称、 所有者和创建日期
- 块大小（以页计）、已用页数和可用页数
- 块中所有表的列表，包含初始页数和页中的表长度

按顺序列出块中的表。该输出對於确定块分段存储是有用的。如果尽管有足够的可用页数但数据库服务器仍无法在块中分配扩展数据块，那么该块可能被错误地分段。

```
DBSpace Usage Report:  rootdbs          Owner:  informix  Created: 08/08/2006

Chunk  Pathname                                 Size     Used     Free
    1  /home/server/root_chunk                 75000    19420    55580

Description                                            Offset    Size
-----------------------------------------        ---------------------
RESERVED PAGES                                            0        12
CHUNK FREELIST PAGE                                      12         1
rootdbs:'informix'.TBLSpace                              13       250
PHYSICAL LOG                                            263      1000
FREE                                                   1263      1500
LOGICAL LOG: Log file 2                                2763      1500
LOGICAL LOG: Log file 3                                4263      1500
```

```
...
sysmaster:'informix'.sysdatabases                    10263         4
sysmaster:'informix'.systables                       10267         8
...

Chunk  Pathname                                Size      Used      Free
    2  /home/server/dbspace1                   5000        53      4947

Description                                          Offset    Size
-------------------------------------------        --------------------
RESERVED PAGES                                            0         2
CHUNK FREELIST PAGE                                       2         1
dbspace1:'informix'.TBLSpace                              3        50
FREE                                                     53      4947
```

相关链接

更改物理日志的位置和大小 在第270页

SMI 表

查询 syschunks 表以获取块状态。 以下是相关列。

chknum
: 数据库空间中的块数

dbsnum
: 数据库空间数

chksize
: 总的块大小（以页计）

nfree
: 可用页数

is_offline
: 块是否脱机

is_recovering
: 块是否正在恢复

mis_offline
: 镜像块是否脱机

mis_recovering
: 镜像块是否正在恢复

syschkio 表中包含以下列。

pagesread
: 从块中读取的页数

pageswritten
: 向块中写入的页数

监视表空间和扩展数据块

监视表空间和扩展数据块以确定数据库、表或表分段的磁盘使用情形。在使用表分段存储情形下，并希望确保表数据和表索引数据是适当分布在分段上之时，监视表的磁盘使用情形就特别重要。

运行 oncheck -pt 以获取扩展数据快信息。 oncheck -pT 选项将返回 oncheck -pt 的选项所有信息，以及有关页和索引使用的其他信息。

SMI 表

查询 systabnames 表以获取有关每个表空间的信息。systabnames 表具有指示相应的表、数据库和每个表空间的表所有者的列。

查询 systabnames 表以获取有关每个扩展数据块的信息。 sysextents 表具有指示扩展数据块所属数据库和表，以及扩展数据块的物理地址和大小的列。

监视 BLOB 空间中的简单大对象

监视 BLOB 空间以确定可用空间和 BLOB 页 大小是否为最佳大小。

使用 oncheck -pB 确定 BLOB 页充满度

oncheck -pB 命令显示描述 BLOB 页平均充满度的统计信息。 如果您发现大量简单大对象的统计信息中显示充满度百分比很低，那么更改 BLOB 空间中 BLOB 页大小可提高数据库服务器性能。

使用数据库名称或表名称作为参数运行 oncheck -pB。以下示例检索存储在 stores_demo 数据库的 sriram.catalog 表中的所有简单大对象的存储信息：

```
oncheck -pB stores_demo:sriram.catalog
```

有关解释 oncheck-pB 输出的详细信息，请参阅《*SinoDB*® 性能指南》中有关表性能注意事项的章节中的优化 BLOB 空间 BLOB 页大小。

使用 oncheck -pe 监视 BLOB 空间使用情况

oncheck -pe命令提供有关 BLOB 空间使用情况的信息：

- 存储 TEXT 和 BYTE 数据的表名（按块）
- 所使用的磁盘页（非 BLOB 页）数（按表）
- 剩余的可用磁盘页数（按块）
- 所用的开销页数（按块）

以下示例显示 oncheck -pe 的输出样本。

```
BLOBSpace Usage Report:  fstblob           Owner:  informix  Created:
 03/01/08
    Chunk: 3   /home/server/blob_chunk                 Size     Used
 Free
                                                       4000      304
 3696
        Disk usage for Chunk 3                       Total Pages
        ---------------------------------------------------------
        OVERHEAD                                               8
        stores_demo:chrisw.catalog                           296
        FREE                                                3696
```

使用 oncheck -pT 监视数据库空间中的简单大对象

使用 oncheck -pT 监视数据库空间以确定 TEXT 和 BYTE 数据使用的数据库空间的页数。

该命令使用数据库名称或表名称作为参数。对于数据库中的每个表，或对于特定表，数据库服务器显示常规表空间报告。

常规报告之后是按页类型的详细列出的扩展数据块中页的使用情况。有关 TEXT 和 BYTE 数据的信息，请看 Type 列。

数据库服务器可在同一 BLOB 页上存储多个简单大对象。因此，您可以计算表空间中存储 TEXT 或 BYTE 数据的页数，但您无法估计表中简单大对象的数目。

以下示例显示样本输出。

```
TBLSpace Usage Report for mydemo:chrisw.catalog

    Type                 Pages      Empty  Semi-Full       Full  Very-Full
    ---------------- ---------- ---------- ---------- ---------- ----------
    Free                     7
    Bit-Map                  1
    Index                    2
    Data (Home)              9
    Data (Remainder)         0          0          0          0          0
    Tblspace BLOBs           5          0          0          1          4
                     ----------
    Total Pages             24

    Unused Space Summary

        Unused data bytes in Home pages               3564
        Unused data bytes in Remainder pages             0
        Unused bytes in Tblspace Blob pages           1430

Index Usage Report for index  111_16 on mydemo:chrisw.catalog

                    Average    Average
    Level    Total No. Keys Free Bytes
    ----- -------- -------- ----------
        1        1       74       1058
    ----- -------- -------- ----------
    Total        1       74       1058

Index Usage Report for index  111_18 on mydemo:chrisw.catalog

                    Average    Average
    Level    Total No. Keys Free Bytes
    ----- -------- -------- ----------
        1        1       74        984
    ----- -------- -------- ----------
    Total        1       74        984
```

监视智能大对象空间

要在智能大对象空间中监视的最重要区域之一是元数据页使用。在创建智能大对象空间时，指定元数据区域的大小。而且，在任何时间向智能大对象空间添加块时，可指定要添加到块的元数据空间。

如果您尝试插入新的智能大对象，但无可用的元数据空间，那么您会收到错误。管理员必须监视元数据空间的可用性，以防止这种情况发生。

使用以下命令来监视智能大对象空间。

命令	描述
onstat -g smbs	显示系统中所有智能大对象空间的存储属性： • 智能大对象空间的名称、标志、所有者 • 日志记录状态 • 平均智能大对象大小 • 第一个扩展数据块大小、下一个扩展数据块大小，和最小扩展数据块大小 • 最大 I/O 存取时间 • 锁定模式

命令	描述
onstat -g smb c	显示每个智能大对象空间块以下信息： • 块号和智能大对象空间名称 • 块大小和路径名称 • 总用户数据页数和可用用户数据页数 • 每个用户数据区域中和元数据区域中页的位置和数量
oncheck -ce oncheck -pe	显示以下有关智能大对象空间使用的信息： • 存储智能大对象的表名（按块） • 所使用的磁盘页（非智能大对象页）数（按表） • 剩余的可用用户数据页数（按块） • 剩余的保留用户数据页数（按块） • 所使用的元数据页数（按块） 输出提供以下总计 ： • 所有用户数据区域和元数据区域的总已用页数。系统将保留区域的 53 页加到用户数据区域和和元数据区域的总数上。 • 在元数据区域中剩余的可用页数 • 在所有用户数据区域中剩余的可用页数
onstat -d	显示以下有关每个智能大对象空间中块的信息： • 在元数据区域和用户数据区域中的每个智能大对象空间块中的可用智能大对象页数 • 在元数据区域和用户数据区域中的每个智能大对象空间块中的总智能大对象页数
oncheck -cs oncheck -ps	验证并显示有关智能大对象空间的元数据区域的信息。
oncheck -cS	显示有关智能大对象空间的智能大对象扩展数据块和用户数据区域的信息。
oncheck -pS	显示有关智能大对象空间的智能大对象扩展数据块、用户数据区域和元数据区域的信息。有关 oncheck-cS 和 -pS 的更多信息，请参阅《*SinoDB*® 性能指南》中有关表性能注意事项主题中的管理智能大对象空间的部份。

相关链接

向智能大对象空间添加块 在第183页
oncheck -ce 和 *oncheck -pe* 选项 在第206页
监视元数据及用户数据区域 在第208页
onstat -d 选项 在第205页
oncheck -ps 选项 在第207页
oncheck -cs 选项 在第207页
《*SinoDB* 管理员参考》*: onstat -g smb* 命令*:* 显示智能大对象空间信息

onstat -d 选项

使用 onstat-d 选项显示以下有关每个智能大对象空间中块的信息：

- 在元数据区域和用户数据区域中的每个智能大对象空间块中的可用智能大对象页数
- 在元数据区域和用户数据区域中的每个智能大对象空间块中的总智能大对象页数

有关 onstat -d 输出的示例，请参阅《*SinoDB*® 管理员参考》中有关 onstat 的信息。

要找出总已用空间量，请运行 oncheck -pe命令。 有关更多信息，请参阅 *oncheck -ce* 和 *oncheck -pe* 选项 在第206页.

备份其中记录删除操作的逻辑日志并释放智能大对象页之后，onstat -d 选项才会将智能大对象页注册为可用的。因此，如果您删除 25 个智能大对象并立即运行 onstat -d，那么在 onstat 的输出中不会包含新释放的空间。

相关链接

监视智能大对象空间 在第204页

oncheck -ce 和 oncheck -pe 选项

运行 oncheck -ce 以显示每个用户数据区域中每个智能大对象空间块大小、已使用空间的总量和可用空间量。oncheck -pe 选项显示与 oncheck -ce 相同的信息，另外还显示块使用的详细清单。首先列出数据库空间，然后列出智能大对象空间。-pe 输出提供以下有关智能大对象空间使用的信息：

- 存储智能大对象的表名（按块）
- 所使用的磁盘页（非智能大对象页）数（按表）
- 剩余的可用用户数据页数（按块）
- 已使用的元数据页数（按块）

输出提供以下总计 ：

- 用户数据区域、元数据区域和保留区域的已使用页的总数。

  系统将保留区域的 53 页加到用户数据区域和和元数据区域的总数上。
- 在元数据区域中剩余的可用页数
- 在用户数据区域中剩余的可用页数

  提示： oncheck -pe 选项以数据库服务器页的形式，而非智能大对象页的形式，提供有关智能大对象空间使用的信息。

以下示列显示样本输出。在此示例中，智能大对象空间 s9_sbspc 的已用总页数为 214、元数据区域中的可用页数为 60，以及用户数据区域中的可用页数为 726 。

```
Chunk Pathname                                      Size      Used      Free
      2 /ix/ids9.2/./s9_sbspc                         1000       940        60

    Description                                        Offset     Size
    ------------------------------------------------ -------- --------
    RESERVED PAGES                                          0        2
    CHUNK FREELIST PAGE                                     2        1
    s9_sbspc:'informix'.TBLSpace                            3       50
    SBLOBSpace LO [2,2,1]                                  53        8
    SBLOBSpace LO [2,2,2]                                  61        1
...
SBLOBSpace LO [2,2,79]                                    168        1
    SBLOBSpace FREE USER DATA                             169      305
    s9_sbspc:'informix'.sbspace_desc                      474        4
    s9_sbspc:'informix'.chunk_adjunc                      478        4
    s9_sbspc:'informix'.LO_hdr_partn                      482        8
    s9_sbspc:'informix'.LO_ud_free                        490        5
    s9_sbspc:'informix'.LO_hdr_partn                      495       24
    FREE                                                  519       60
    SBLOBSpace FREE USER DATA                             579      421

    Total Used:                                 214
    Total SBLOBSpace FREE META DATA:             60
    Total SBLOBSpace FREE USER DATA:            726
```

可使用 CHECK EXTENTS 作为等效于 oncheck -ce 的 SQL 管理 API 命令。有关使用 SQL API 命令的信息，请参阅 使用 *SQL* 管理 *API* 执行远程管理 在第458页以及《*SinoDB*® 管理员参考》。

相关链接

监视智能大对象空间 在第204页

oncheck -cs 选项

oncheck -cs 及 oncheck -Cs 选项验证智能大对象空间的元数据区域。以下示例显示 s9_sbspc 的 -cs 的输出示例。如果不在命令行上指定智能大对象空间名称，那么 oncheck 会检查并显示所有智能大对象空间的元数据。

使用 oncheck -cs 输出来查看元数据区域中的剩余空间量。如果该区域已满，那么为元数据区域分配另一具有足够空间的块。要查元数据区域中已用页数，可将 Used 列中的数值进行总计。要查元数据区域中可用页数，可将 Free 列中的数值进行总计。

例如，基于下图中显示的字段值，s9_sbspc 的元数据区域中已用总页数为 33 个 2 KB 大小的页（或 66 KB）。元数据区域包含的可用总页数为 62（或 124KB）。

```
Validating space 's9_sbspc' ...

SBLOBspace Metadata Partition                    Partnum      Used      Free
s9_sbspc:'informix'.TBLSpace                     0x200001        6        44
s9_sbspc:'informix'.sbspace_desc                 0x200002        2         2
s9_sbspc:'informix'.chunk_adjunc                 0x200003        2         2
s9_sbspc:'informix'.LO_hdr_partn                 0x200004       21        11
s9_sbspc:'informix'.LO_ud_free                   0x200005        2         3
```

相关链接

监视智能大对象空间 在第204页

oncheck -ps 选项

oncheck -ps 选项验证并显示智能大对象空间分区中元数据区域的信息。 以下示例显示 s9_sbspc 的 -ps 输出示例。如果不在命令行上指定智能大对象空间名称，那么 oncheck 会验证并显示所有存储空间的表空间信息。

要监视可用元数据空间量，请运行以下命令 ：

```
oncheck -ps spacename
```

-ps 输出包括了有关元数据区域中的锁定粒度、 partnum、已分配与已使用的页数、扩展数据块大小以及行数的信息。使用 oncheck -ps 输出来查看元数据区域中的剩余空间量。如果该区域已满，那么为元数据区域分配另一具有足够空间的块。

如果您为包含存除了智能大对象的表的数据库空间运行 oncheck -ps，那么您可查到表中的行数。

```
Validating space 's9_sbspc' ...

TBLSpace Report for
    TBLspace Flags                2801       Page Locking
                                             TBLspace use 4 bit bit-maps
                                             Permanent System TBLspace

    Partition partnum             0x200001
    Number of rows                92
    Number of special columns     0
    Number of keys                0
    Number of extents             1
    Current serial value          1
    First extent size             50
    Next extent size              50
    Number of pages allocated     50
    Number of pages used          6
    Number of data pages          0
    Number of rows                0
    Partition lockid              2097153

    Current SERIAL8 value         1
```

```
Current REFID value             1
Created                         Thu May 24 14:14:33 2007
```

相关链接

监视智能大对象空间 在第204页

监视元数据及用户数据区域

数据库服务器保留 40% 的用户数据区域作为保留区域。数据库服务器将此保留空间用于元数据或用户数据。随著智能大对象添加到该智能大对象空间将耗尽元数据区域。当数据库服务器耗尽元数据或用户数据空间时，它会将一块保留空间移至相应区域。

当所有的保留区域耗尽时，即使用户数据区域有可用空间，数据库服务器也无法将空间移至元数据区域。

1. 当您向智能大对象空间添加智能大对象时，请使用 oncheck -pe 或 onstat -g smb c 来监视元数据区域、用户数据区和保留区域中的空间。
   有关示例，请参阅 *oncheck -ce 和 oncheck -pe 选项* 在第206页。
2. 使用消息日志来监视元数据盗用。

   数据库服务器打印有关从保留区域分配到元数据区域的页数消息。
3. 在智能大对象空间耗尽元数据和保留区域空间之前，向智能大对象空间添加另一个块。

   有关更多信息，请参阅 向智能大对象空间添加块 在第183页。
4. 数据库服务器在将空间从保留区域移至元数据或用户数据区域时写入 FREE_RE 和 CHKADJUP 日志记录。

有关更多信息，请参阅 计算智能大对象空间元数据的大小 在第183页 。

相关链接

监视智能大对象空间 在第204页

多租户

通过在单个 SinoDB® 实例中创建多个租户数据库来分隔多个客户机构的数据、 存储空间和处理资源。

例如，假设您想要为小企业提供工资单服务，将工资单应用程序视为服务销售小型企业客户。您可在单个 SinoDB® 实例中为每个客户端配置一个租户数据库，而不是为每个客户端提供一个单独的 SinoDB® 实例。

配置多租户时，可以分隔数据库服务器的以下方面 ：

数据

为每个客户端创建单独的租户数据库。

存储空间

每个租户数据库把存储空间专用于数据存储。在租户数据库中所创建的表、片段和索引必须创建在专用存储空间中。仅租户数据库可以使用该专用的存储空间。

您可以限制租户数据库可用的永久存储空间量以节省系统资源。

临时存储空间可供特定租户数据库专用或供数据库之间共享。

用户

可以设置客户端用户访问每个租户数据库的权限。可以授予某些用户创建、 修改或删除租户数据库的权限。在默认情况下，仅 DBA 或用户 informix 可以创建租户数据库。

处理资源

可通过定义租户虚拟处理器类和创建执行租户数据库的会话线程的虚拟处理器来分隔租户数据库的 CPU 资源。

会话限制

可为租户会话设置以下限制 ：

- 租户会话可以获得的锁定数。

- 可配置给会话的内存量。
- 可分配给会话的临时存储空间量。
- 会话中事务的大小（基于各个事务填充的日志空间量）。
- 允许会话中事务可运行的时间量。
- 连接到租户数据库的所有会话的分享内存量。
- 租户数据库的客户端连接数。

下图显示 SinoDB® 服务器实例中两个客户端的可能配置。每个客户端具有一个数据库和可访问租户数据库的用户。每个租户数据库拥有自己的存储空间。这两个租户数据库共享缺省的临时智能大对象空间。Tenant A 具有带有两个虚拟处理器的租户虚拟处理器类，而 Tenant B 具有带有一个虚拟处理器的虚拟处理器类。

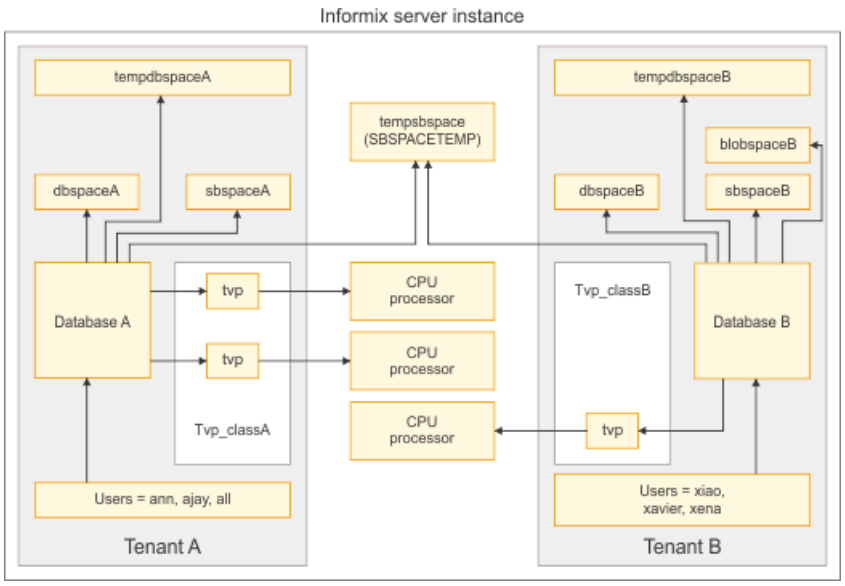

图 44: SinoDB® 服务器实例中的多个租户

复制与租户数据库

可使用 Enterprise Replication 和高可用性集群来复制租户数据库。

可通过 Enterprise Replication 网格运行命令来创建、 修改或删除租户数据库。

不可从高可用性集群中可更新的辅助服务器运行命令来创建、 修改或删除租户数据库。

备份与复原租户数据库

可将租户数据库备份为数据库服务器备份的一部分，或者通过在备份命令中指定租户存储空间来备份租户数据库。可通过在 onbar -r 命令中指定 -T 选项，以 ON-Bar 来复原单个租户数据库。

相关链接

存储空间的创建和管理 在第26页
租户虚拟处理器类 在第92页

创建租户数据库

您可以创建一个租户数据库来隔离一个客户端组织的数据、 存储和处理资源。

您必须是 informix 用户、 DBSA 或具有 TENANT 权限才可创建租户数据库。

不可将现有数据库转换为租户数据库。也不可将租户数据库转换为非租户数据库。不可使用 CREATE DATABASE 语句来创建租户数据库。

要创建租户数据库：

1. 为租户数据库创建存储空间。创建租户数据库时，所有专用的存储空间必须为空。

必须为租户数据库创建以下类型的专用空间 ：

数据库空间

必须为租户数据库创建至少一个数据库空间 。租户数据库必须存储在一个或多个专用数据库空间中。

BLOB 空间

如果租户数据库会包含简单大对象，那么必须创建一个或多个 BLOB 空间。

智能大对象空间

如果租户数据库会包含智能大对象，那么必须创建一个或多个智能大对象空间。智能大对象可以包含 BLOB 或 CLOB 数据，或者是太大而无法在一行中容纳的数据和表的统计信息。某些 SinoDB® 功能，如 Enterprise Replication、空间数据，和基本文本搜索都需要智能大对象空间。

临时数据库空间

可选 ：创建一个或多个临时数据库空减以存储临时表。否则，临时表将存储在 DBSPACETEMP 配置参数或环境变量所指定的临时数据库空间中。

临时智能大对象空间

可选 ：创建一个或多个临时智能大对象空間以存储临时智能大对象。否则，临时智能大型对象將存储在 SBSPACETEMP 配置参数所指定的临时智能大对象空间中。

2. 可选： 为租户数据库设置限制，使其不能垄断系统资源。

T租户数据库限制不适用于拥有管理特权的用户，如 informix 用户或 DBSA 用户。可为租户数据库设置以下限制 ：

会话可用的锁定

设置 session_limit_locks 属性以指定会话最大可用的锁定数。

会话中事务可用的日志空间

设置 session_limit_logspace 属性来指定会话用于各个事务的最大日志空间量。

会话可用内存量

设置 session_limit_memory 属性来指定会话可分配的最大内存量。

会话可用的临时表空间

设置 session_limit_tempspace 属性来指定会话可分配的最大临时表空间量。

事务可运行的时间量

Set the session_limit_txn_time property to specify the maximum amount of time that a transaction can run in a session

设置 session_limit_txn_time 属性来指定会话中事务可运行的最大时间量

租户数据库可用空间总量

设置 tenant_limit_space 属性以指定租户用户可用的最大存储空间量。

3. 设置存储池以便存储空间可自动增长。可指定可扩展存储空间的最大大小以限制租客数据库的增长。
4. 向特定用户提供 TENANT 特权，以创建、 修改和删除租户数据库。
例如，以下命令将 TENANT 特权给予用户 jsmith ：

```
EXECUTE FUNCTION task("grant admin", "jsmith", "tenant");
```

5. 通过运行带有 tenant create 参数的 SQL 管理 API 函数 admin() 或 task() 来创建租户数据库并定义其属性。
例如，以下语句创建名为 companyA 的租户数据库 ：

```
EXECUTE FUNCTION task('tenant create', 'companyA',
      '{dbspace:"companyA_dbs1,companyA_dbs2", sbspace:"companyA_sbs1",
        vpclass:"tvp_A,num=2", logmode:"ansi"}');
```

租户数据库具有两个数据库空间、一个智能大对象空间、两个租户虚拟处理器和 ANSI 日志记录模式。

在租户数据库中创建或更改表和索引期间明确指定存储位置时，必须指定列在租户数据库定义中的数据库空间。否则，该语句会失败。如果未明确指定表或索引的存储位置，它们将创建在租户数据库定义中列出的第一个数据库空间中。

注： 通过执行以下任务来提高数据库的安全性 ：

- 运行 GRANT 和 REVOKE 语句来控制用户对数据库的访问。
- 设置 DBCREATE_PERMISSION 配置参数来限制创建非租户数据库的能力。

相关链接

自动空间管理 在第185页

《SinoDB 管理员参考》: tenant create 参数: 创建租户数据库 (SQL 管理 API)

《SinoDB 管理员参考》: The tenant 表

管理租户数据库

可查看和更新租户数据库属性，以及删除租户数据库。

查看租户数据库属性

使用 SinoDB® 开放管理工具 (OAT) 查看有关租户数据库的空间使用、存储空间、租户虚拟处理器、块、表和索引等的信息。

要查看租户数据库定义，请查询 sysadmin 数据库中的 tenant 表。例如，以下语句列出租户数据库及其属性 ：

```
SELECT hex(tenant_id),tenant_dbsname,tenant_resources::json,
       tenant_create_time,tenant_last_updated
FROM tenant;
```

包含租户属性的 tenant_resources 列的类型为 BSON，因此必须将该列转换为 JSON 以查看这些属性。

更新租户数据库的属性

要更新属性，请运行带有 tenant update 参数的 SQL 管理 API 函数 admin() 或 task()。更新将在新会话中生效。

可向租户数据库的存储空间的现有列表追加数据库空间、 BLOB 空间或智能大对象空间。存储空间必须为空。您必须具有 DBA 或 TENANT 权限才能更改租户数据库属性。

不可从租户数据库中删除专用存储空间，除非是删除该数据库。

为以下租户数据库属性指定新值时，现有值就会被替换掉了。

- dbspacetemp （分配给租户的临时数据库空间）
- session_limit_logspace （单个事务的日志空间限制）
- session_limit_memory （每个会话分配的内存的限制）
- session_limit_tempspace （每个会话的临时表空间限制）
- session_limit_txn_time （事务可运行的时间长度限制）
- sbspacetemp （分配给租户的临时智能大对象空间）
- session_limit_locks （每个会话的锁定数限制）
- tenant_limit_space （总存储空间限制）
- vpclass （虚拟处理器类名称和数量）

删除租户数据库

要删除租户数据库，请运行带有 tenant drop 参数的 SQL 管理 API 函数 admin() 或 task()。您必须具有 DBA 或 TENANT 权限才能删除租户数据库。不可使用 DROP DATABASE 语句删除租户数据库。租户数据库的所有专用存储空间是空的且可用的。将删除不与其他租户数据库共享的任何租户虚拟处理器。

相关链接

《SinoDB 管理员参考》: *tenant update* 参数: 更改租户数据库属性 *(SQL* 管理 *API)*
《SinoDB 管理员参考》: *tenant drop* 参数: 删除租户数据库 *(SQL* 管理 *API)*
《SinoDB 管理员参考》: *The tenant* 表

复原租户数据库到某一时间点

可使用 ON-Bar 实用程序将租户数据库复原到某一个时间点。

在开始复原租户数据库之前，必须满足以下条件：

- 租户数据库存储空间必须具有物理备份。
- 数据库服务器必须联机。
- 租户数据库定义中的所有租户存储空间必须存在。
- 对于现有的备份而言，租户存储空间必须有唯一名称。例如，如果删除具有备份的租户数据库，并在新租户数据库中使用具有相同名称的存储空间，则必须删除已删除租户数据库的备份。
- 没有其他热复原或租户复原正在进行。

对于高可用性集群中的租户数据库，租户数据库时间点复原有以下附加条件：

- 该集群不能包含共享磁盘的辅助服务器或可更新的辅助服务器。只可包含只读的 HDR 辅助服务器和远程独立的辅助服务器。
- 所有辅助服务器必须联机。
- 仅可以在主服务器上运行复原命令。

  在租户时间点复原的过程中，将在辅助服务器上运行内部产生的命令以复原辅助服务器上租户数据库空间的新状态。租户时间点复原命令包括在租户空间新状态的新物理备份。主服务器上运行的租户时间点复原命令必须等到所有辅助服务器确认自动生成的新租户空间备份的复原完成后，才会返回。

如同其他热复原，租户数据库复原所需的逻辑日志将复原到 DBSPACETEMP 配置参数所指定的临时空间。如果未设置 DBSPACETEMP 配置参数，那么将会在根数据库空间中创建临时文件。

要复原租户数据库到某一时间点，请运行以下命令：

```
onbar -r -T tenant_database -t "time" -0
```

以租户数据库名称替代 *tenant_database*。以从逻辑日志中复原的最后事务的时间替代 *time*。

如果省略 -0 选项，在执行复原命令之前所有永久性租户空间必须标记为脱机。临时空间从不备份或复原。

如果复原失败，需解决问题并重新执行复原。在复原成功完成前，租户数据库无法接受连接。在复原期间，sysadmin:tenant 表的 tenant_resources 列的 tenant_state 字段值是设为 restoring。当租户数据库是被阻挡的，tenant_state 字段值是设为 blocked。可执行以下查询来查看 tenant_state 字段值：

```
SELECT bson_value_lvarchar(tenant_resources, 'tenant_state') AS tenant_state
   FROM sysadmin:tenant
   WHERE tenant_dbsname = 'tenant_dbname';
```

以租户数据库名称替代 *tenant_dbname*。

> 示例
>
> 以下命令将名为 tenant1 的租户数据库复原到指定的时间点：
>
> ```
> onbar -r -T tenant1 -t "2015-10-10 11:35:57" -0
> ```

相关链接

《SinoDB 管理员参考》: *tenant* 表
《SinoDB 备份和还原指南》: *onbar -r* 语法: 复原数据

存储优化

数据压缩和合并过程可以最小化数据和索引使用的磁盘空间。

下表描述可用于减少以下对象使用的磁盘空间量的方法：行中数据、数据库空间中的简单大对象，和索引键。可自动执行任一或所有方法，或者按需操作。

表 29：存储优化方法

存储优化方法	目的	何时使用
压缩	压缩表和分段中的数据、压缩数据库空间 中的简单大对象，以及压缩索引中的键。减少所需的磁盘空间。 启用压缩之后，会自动压缩新数据或索引键。	当想要减少 2000 行或更多的数据行、数据库空间中的简单大对象、或 2000 个或更多的索引键的大小时
重新打包	合并表、分段和索引中的可用空间。	在压缩之后或者当想要合并可用空间时
收缩	将可用空间返还给数据库空间。	在压缩或重新打包之后，或者想要将可用空间返还给数据库空间时。
消除碎片	使邻接的合并扩展数据块中的数据行靠得更紧密	当频繁更新的表或索引在多个非连续的扩展数据块中分散时

下图显示使用分段中大部分空间的未压缩数据、数据压缩时创建的可用空间、在重新打包操作后移动到分段末端的可用空间，以及在收缩操作后留在分段中的数据。索引存储优化的方法是相同的。

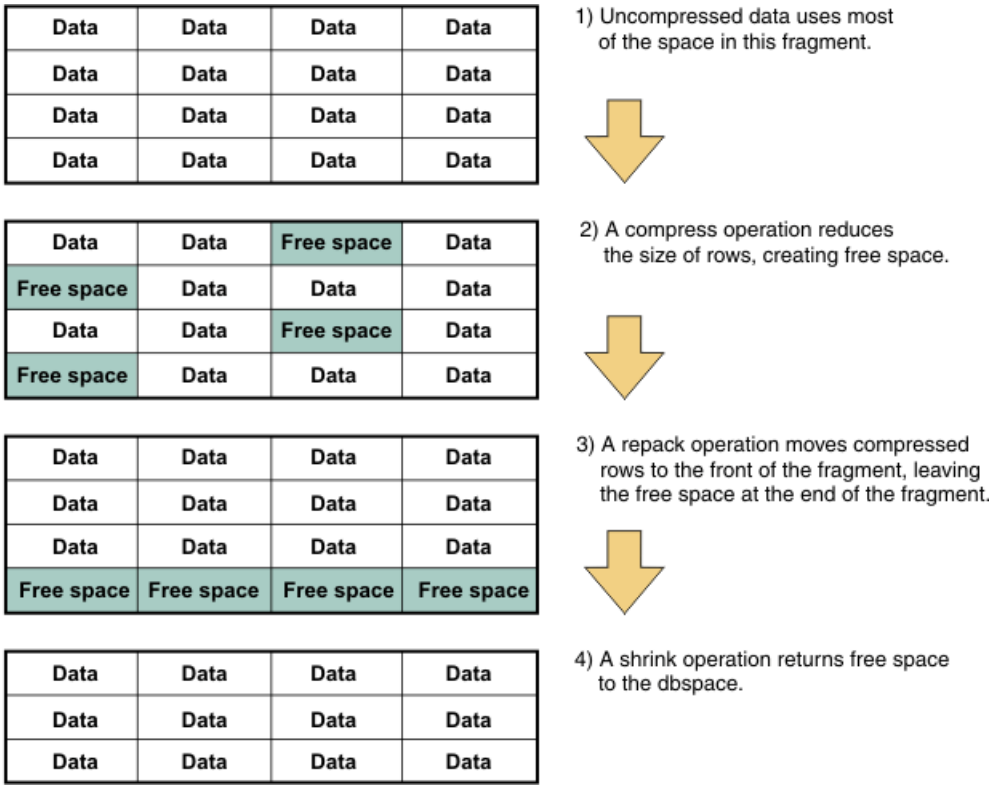

图 45：压缩和存储优化过程中分段中的数据

相关链接

存储空间的创建和管理 在第26页

存储优化方法

可优化各表、分段或索引。也可为所有的表和分段安排自动优化。

可使用 CREATE TABLE 语句的 COMPRESSED 选项在表至少有 2000 行时启用表自动压缩。

如果索引具有 2000 个或更多的键，则可使用 CREATE INDEX 语句的 COMPRESSED 选项来启用索引自动压缩。如果少于 2000 个键，则不会启用压缩。

可使用 SQL 管理 API 函数 task 或 admin 对表、分段或索引执行任何类型的存储优化。

可启用 auto_crsd 调度程序任务以自动对所有表和表分段进行压缩、重新打包、收缩和消除碎片。

可使用 OAT 来优化任何表、分段或索引，或者自动优化所有的表和表分段。

表 30: 存储优化的方法

目的	SQL 语句	SQL 管理 API 参数	调度程序任务	OAT 页
自动压缩表或分段的数据。	带有 COMPRESSED 选项的 CREATE TABLE 语句	table compress 或 fragment compress		Storage
自动压缩所有的表和分段的数据。			auto_crsd	Server Optimization Policies
重新打包和收缩表或分段		table repack shrink 或 fragment repack shrink		Storage
自动重新打包和收缩所有的表和分段			auto_crsd	Server Optimization Policies
自动压缩B型树索引	带有 COMPRESSED 选项的 CREATE INDEX 语句	index compress		Storage
重新打包和收缩B型树索引		index repack shrink		Storage
对表或分段进行消除碎片		defragment		Storage
对所有的表和分段进行自动消除碎片			auto_crsd	Server Optimization Policies

相关链接

可压缩的数据 在第218页
查看压缩信息 在第221页
《SinoDB SQL 指南: 语法》: 表的 COMPRESSED 选项
《SinoDB SQL 指南: 语法》: 索引的 COMPRESSED 选项
《SinoDB 管理员参考》: defragment 参数: 动态地对分区扩展块取消分段 (SQL 管理 API)
《SinoDB 管理员参考》: 表和分段压缩和解压缩操作 (SQL 管理 API)

《SinoDB 管理员参考》: index compress repack shrink 参数: 优化 B 型树索引的存储 (SQL 管理 API)
压缩 在第218页

自动优化数据存储

可通过启用 auto_crsd 调度器任务，为所有表和扩展数据块配置自动压缩、收缩、重新打包和消除碎片。

可通过 SinoDB® 开放管理工具 (OAT) 中的 Server Optimization Policies 页或通过更新 sysadmin 数据库的调度程序表来启用和配置 auto_crsd 任务。

通过更新调度程序表启用 auto_crsd 任务：

1. 以用户 informix 或其他授权用户身分连接到 sysadmin 数据库。
2. 通过在 ph_task 表上使用 UPDATE 语句启用 auto_crsd 调度程序任务将 tk_enable 列的值设为 T。例如，以下语句启用 auto_crsd 任务：

```
UPDATE ph_task
  SET tk_enable = 'T'
  WHERE tk_name = 'auto_crsd';
```

3. 可选： 通过在 ph_task 表上使用 UPDATE 语句更改 tk_frequency 列的值，以更改该任务的的运行频率。缺省值为 7 00:00:00，指示每周运行一次该任务。例如，以下语句将频率更改成每天一次：

```
UPDATE ph_task
  SET tk_frequency = '1 00:00:00'
  WHERE tk_name = 'auto_crsd';
```

4. 可选： Disable individual operations by using an UPDATE statement on the ph_threshold table to set the value column for a threshold to F:通过在 ph_threshold 表上使用 UPDATE 语句将阈值的 value 列设置为 F，以禁用个别操作：

   - AUTOCOMPRESS_ENABLED：控制压缩
   - AUTOREPACK_ENABLED：控制重新打包
   - AUTOSHRINK_ENABLED：控制收缩
   - AUTODEFRAG_ENABLED：控制消除碎片

   例如，以下语句仅禁用 auto_crsd 任务的消除碎片操作：

```
UPDATE ph_threshold
  SET value = 'F'
  WHERE name = 'AUTODEFRAG_ENABLED';
```

5. 可选： 通过在 ph_threshold 表上使用 UPDATE 语句更改阈值 value 列的值，以更改个别操作的阈值。

   - AUTOCOMPRESS_ROWS：压缩的阈值是未压缩行数。缺省阈值是 50,000 行。当未压缩行数超过 50,000 时，将压缩表。
   - AUTOREPACK_SPACE：重新打包表的阈值是非连续空间的百分比。缺省值为 90%。表所占用的空间中超过 90% 以上的空间为不连续时，将重新打包该表。
   - AUTOSHRINK_UNUSED：压缩表或分段的阈值是已分配的未使用空间的百分比。缺省值为 50%。当已分配空间中有超过 50% 以上未使用时，将会收缩表或分段。
   - AUTODEFRAG_EXTENTS：表或扩展数据块的消除碎片阈值是扩展数据块的数量。缺省值为 100。当扩展数据块数超过 100 时，将会对表或表分段进行消除碎片操作。

   例如，以下语句将压缩阈值更改为 5000 行：

```
UPDATE ph_threshold
  SET value = '5000'
  WHERE name = 'AUTOCOMPRESS_ROWS';
```

当超出已启用的操作的阈值时，调度程序将会运行该操作。

相关链接

对分区取消分段 在第217页
调度程序 在第441页
《*SinoDB* 管理员参考》：调度程序表

示例：根据需要优化数据存储

在此例中，您将学习如何通过 SQL 管理 API 命令确定压缩表从而节省的空间量、压缩表的方法，以及根据需要优化存储的方法。您也将学习解压缩表和移除压缩字典的方法。

假设在名为 music 的数据库中有一个名为 rock 的表，且该数据库的所有者是用户 mario。rock 表没有分段化。您可在表分段上执行与在整个表上执行的相同操作，但是语法规则略有不同。

先决条件 ：

- 表中每个分段中必须至少有 2,000 行，而不是整个表总共有 2,000 行。
- 您必须可以连接到 sysadmin 数据库（缺省情况下只有用户 informix 可连接），且您必须是 DBSA。
- 逻辑和物理日志足够大，可以处理正常处理和压缩操作。压缩、重新打包和解压缩操作会使用大量日志空间。

要压缩行数据和数据库空间中的简单大对象：

1. 可运行以下指令来检查压缩该表从而节省的空间量：

```
EXECUTE FUNCTION task("table estimate_compression", "rock", "music", "mario");
```

查看生成的报告，其中指示可为 rock 表当前使用的空间节省约 75%。您决定压缩该表。

2. 在压缩数据之前，要创建压缩字典，其中包含 SinoDB® 用于压缩 rock 表中的数据的信息。请运行以下命令。

```
EXECUTE FUNCTION task("table create_dictionary", "rock", "music", "mario");
```

提示： 如果不以单独步骤创建压缩字典，那么 SinoDB® 将在压缩数据时自动创建字典。

3. 您决定压缩 rock 表中的数据以及 数据库空间中的简单大对象，合并数据，然后将可用空间返还给数据库空间。请运行以下命令：

```
EXECUTE FUNCTION task("table compress repack shrink", "rock", "music", "mario");
```

通过并行运行，可以更快地执行相同的操作。请运行以下命令：

```
EXECUTE FUNCTION task("table compress repack shrink parallel", "rock",
"music", "mario");
```

可通过指定要压缩或收缩的项目来调整命令。例如：

- 要仅压缩行数据，请指定：

```
EXECUTE FUNCTION task("table compress rows parallel","rock","music","mario");
```

- 要仅压缩行数据，然后重新打包并收缩数据，请指定：

```
EXECUTE FUNCTION task("table compress repack shrink rows parallel",
"rock","music","mario");
```

- 要仅压缩数据库空间中的简单大对象，请指定：

```
EXECUTE FUNCTION task("table compress blobs parallel","rock","music","mario");
```

在压缩现有行和简单大对象之后，SinoDB® 会合表末端留下的可用空间，然后将其从表中移除并返还给数据库空间。

如果简单大对象或行压缩之后不会变小，那么数据库服务器不会对其进行压缩。

4. 要解压缩数据。请运行以下命令：

```
EXECUTE FUNCTION task("table uncompress parallel", "rock", "music", "mario");
```

5. 要移除压缩字典。
   a) 验证 Enterprise Replication 是否不需要该压缩字典。

      如果 Enterprise Replication 确实需要这些已解压缩或已删除的表和分段的压缩字典，请勿除去这些字典。
   b) 将包含具有压缩字典的表或分段的数据库空间归档。
   c) 运行以下命令：

```
EXECUTE FUNCTION task("table purge_dictionary", "rock", "music", "mario");
```

要在表片段上运行压缩和其他存储优化命令，应包含 fragment 参数而非 table 参数，以及分段分区号而非表名称。

```
EXECUTE FUNCTION task("fragment command_arguments", "partnum_list");
```

相关链接

压缩 在第218页

《SinoDB 管理员参考》: table 或 fragment 参数: 压缩数据和优化存储 (SQL 管理 API)

对分区取消分段

可通过对分区取消分段来合并非连续空间从而提高性能。

随著时间推移，经常更新的表可能变得支离破碎，这会服务器每次访问表的性能。消除表的碎片可使数据行更紧密，并可避免分区标题页溢出问题。消除索引的碎片可使条目更紧密，以提升访问表信息的速度。

在对表、索引或分区进行碎片消除之前，请确保没有以下任一冲突操作正在进行：

- 表、索引或数据库空间上有现存消除碎片操作。
- 表或分区上正在运行 DDL 语句，如 DROP TABLE 或 ALTER FRAGMENT。
- 表正在进行截断。
- 表正在进行压缩或解压缩。
- 正在运行联机索引创建。

不能对以下对象进行消除碎片操作：

- 伪表，如虚拟表接口 (VTI) 表
- 具有虚拟索引接口 (VII) 索引的表
- 具有函数索引的表
- 临时表
- 排序文件
- 具有互斥存取设置的表

要确定表、索引或分区的扩展数据块数，可运行 oncheck -pt 命令。

要消除表、索引或分区的碎片，可运行带有 defragment 参数或 defragment partnum 参数的 SQL 管理 API 函数 task() 或 admin()，并指定要进行碎片消除的表名称、索引或分区号。

运行命令之后，就不能停止消除碎片请求。

如果在完成消除碎片请求中发生问题，将向联机日志文件发送错误消息。

相关链接

《SinoDB 管理员参考》: oncheck -pt 和 -pT: 显示表或分段的表空间

《SinoDB 管理员参考》: defragment 参数: 动态地对分区扩展块取消分段 (SQL 管理 API)

自动优化数据存储 在第215页

压缩

可压缩并解压表和分段中的行数据，以及数据库空间中的简单大对象。可压缩 B-型树索引。还可合并表或分段中的可用空间，并将该可用空间返还给数据库空间。在压缩数据之前，可估计可以节省的磁盘空间量。

压缩数据、简单大对象或者索引、合并数据，以及返还可用空间有以下益处：

- 显著节省磁盘存储空间
- 减少压缩的分段的磁盘使用量
- 显著节省逻辑日志的使用量，其在完成压缩操作后可以节省更多空间，并防止高吞吐量 OLTP 的瓶颈。
- 页面读取更少，因为更多行可置于一个页面中
- 缓冲池更小，因为更多数据可置于同一大小的池中
- 减少 I/O 活动
  - 与未压缩行相较，更多压缩行置于一个页面中
  - 压缩行的插入、更新和删除操作的日志记录更小
- 能够压缩按时间分段不常访问的数据的较旧分段，而让最近频繁访问的数据处于解压缩的格式
- 能够释放表不再需要的空间
- 备份和复原更快

如果应用程序运行时具有很高的缓冲区高速缓存命中率，并且高性能比空间使用更重要，那么可能不希望压缩数据，因为压缩可能会稍微降低性能。

可并行压缩数据和索引。

查询可以访问压缩表中的数据。

由于与未压缩的数据相比，压缩数据占用更少页面且每页中有更多行，所以查询优化器可能在压缩后选择不同的方案。

如果使用 Enterprise Replication，那么压缩一个复制服务器上的数据不会影响其他复制服务器上的数据。

如果使用高可用性集群，源表中压缩的数据，在目标表中也是压缩的。不能在辅助服务器上执行压缩操作，因为辅助服务器必须与主服务器拥有相同的数据和物理布局。

压缩的主要替代方法是购买更多的物理存储器。减少 IO 绑定工作负载中的瓶颈的主要替代方法是购买更多物理内存以便扩展缓冲池。

相关链接

存储优化方法 在第214页
示例: 根据需要优化数据存储 在第216页

可压缩的数据

可以压缩行中的数据以及数据库空间中的简单大对象。但是，您可能不希望压缩可压缩的所有类型的数据。

可压缩以下数据类型：

- 数据行的内容，包括跨多页的行的任何剩馀片段，以及在逻辑日志记录中的行的映像。
- 存储在数据库空间中的简单大对象（TEXT 或 BYTE 数据类型）。

具有频繁重复的长模式的表或表分段数据非常适合压缩。特定类型的数据（如文本）可能比其他类型数据（如数字数据）更适合压缩，因为文本之类的数据可能包含更长和更频繁重复的模式。

I/O 绑定表，例如：具有不良高速缓存命中率的表，是很适于压缩的。在 OLTP 环境中，压缩 I/O 绑定表可以提升性能。

SinoDB® 可以压缩任意数据类型的组合，因为它将所有数据视为未结构化的字节序列来压缩。因此，服务器可以压缩跨多个列的模式，如城市、州和邮政编码的组合。（服务器按照数据压缩前存在的相同序列解压字节序列。）

相关链接

存储优化方法 在第214页

不可压缩的数据
不能压缩某些类型的表和片段中的行的数据。

不能压缩以下数据库对象中的行中的数据：

- 在 sysmaster、sysutils、sysuser、syscdr 和 syscdcv1 数据库中的表和分段
- 目录
- 临时表
- 虚拟表接口表
- 表空间 tblspace
- 内部分区表
- 字典表（每个数据库空间一个字典表，这些表包含该数据库空间中已压缩分段或表的压缩字典，以及有关这些字典的元数据。）

不能压缩正在执行联机索引建构的表。

不能压缩 BLOB 空间中的简单大对象。

仅当压缩行的映像小于未压缩映像时，SinoDB® 才会压缩行的映像。即便压缩行仅比未压缩映像稍小一些，但节省的少量框间也可以让服务器在页中放入更多行。

非常小的表不适于压缩，因为可能无法从压缩的行中获得足够空间以抵消压缩字典的存储成本。

SinoDB® 无法将单个行压缩到小于 4 个字节长。服务器必须保留空间，以防之后行映像增长超过页所能容纳。因此，请务必不要尝试压缩具有包含 4 字节或短于 4 字节的行的分段或非分段的表。

B 型树索引压缩
可以压缩拆离的 B 型树索引。还可以合并索引中的可用空间，并且将索引末尾的可用空间返还给数据库空间。在压缩索引之前，可估计可以节省的磁盘空间量。

可以压缩分段或非分段表上的拆离的 B 型树索引。

索引必须要有至少 2000 个键才可进行压缩。

不能压缩以下类型的索引：

- 不是 B 型树索引的索引
- 连接的 B 型树索引
- 虚拟 B 型树索引
- 键数少于 2000 的索引

压缩操作仅压缩索引的叶子（底层）。

不能将已压缩的索引解压缩。如果不再需要已压缩的索引，可以删除该索引，然后将其重新创建为非压缩的索引。

使用带有 COMPRESSED 选项的 CREATE INDEX 语句创建索引时，可以对其进行压缩。可使用 SQL 管理 API 命令来压缩现有的索引。

相关链接

《SinoDB 管理员参考》: index compress repack shrink 参数: 优化 B 型树索引的存储 (SQL 管理 API)
《SinoDB 管理员参考》: index estimate_compression 参数: 估计索引压缩 (SQL 管理 API)
《SinoDB 管理员参考》: 根节点与叶节点的创建
《SinoDB SQL 指南: 语法》: CREATE INDEX 语句

估算压缩率
压缩率取决于压缩的数据。在压缩表或表分段之前，可估算数据压缩后将可节省的空间量。压缩估算是根据行数据样本。实际节省的空间比率可能略有不同。

SinoDB® 使用的压缩算法是基于字典的算法，该算法对在字典建构时采样的数据中最常按长度加权的数据的模式执行操作。

如果典型数据分布偏离字典创建时的采样数据，那么压缩率会降低。

最大压缩率为 90%。任何字节序列的最大压缩率是通过将每组 15 个字节替换为一个 12 位符号数，生成原映像 10% 大小的压缩映像。但是，由于 SinoDB® 会向每个压缩映像添加一个字节的元数据，因此无法达到 90% 的比率。

SinoDB® 通过对行数据的随机采样然后计算以下项目大小总和来估算压缩率：

- 未压缩的行映像
- 压缩的行映像，根据由估算压缩命令所临时创建的新压缩字典
- 压缩的行映像，根据现存字典（如果有的话）。如果没有现存字典，该值将与未压缩行映像大小总和相同。

空间实际节省比率可能与压缩估算有所差异，这是由于采样错误、数据类型、数据置入数据页的方式，或运行了其他存储优化操作而导致。

某些数据类型的压缩率高于其他数据类型：

- 即使文本储存于 CHAR 或 VARCHAR 列中，不同语言或字符集的文本可能有不同的压缩率。
- 主要由零组成的数字数据可能压缩的很好，而更多变化的数字数据可能无法压缩的很好。
- 具有长空白的数据可压缩的很好。

压缩估算取决于行的原始压缩率。通常情况下，服务器把一行放在一个数据页内。行在数据页的放置方式会影响实际压缩率与估算压缩率之间的差异：

- 每个未压缩行几乎填满一整页且压缩率低于 50% 时，那么每个压缩行仍将填充半页以上。服务器仍将每个压缩行放置于单独页。在这种情况下，虽然估算压缩率可能是 45%，但是实际上没有节省到空间。
- 每个未压缩行填充略多于半页且压缩率低，那么每个压缩行可能小到足以放置于半页中。服务器可将两个压缩行置于一页上。在这种情况下，即使估算压缩率可能低至 5%，但是实际空间节省为 50%。

SinoDB® 在一页中存储行数不超过 255 行。因此，小行或大页会减少压缩可获得的节省总量。例如，如果压缩前一页中有 200 行，由于压缩后一页仅可有 255 行，因此，无论压缩时行有多小，最大有效压缩率约为 20%。

如果使用的页大小大于最小页大小，那么可通过切换到较小的页的方式来增加实际压缩空间节省量，从而：

- 不再达到 255 行的限制。
- 如果仍然达到该限制，页中的未使用空间将更少。

如果压缩操作加上重新打包操作、缩紧操作，或重新打包和收缩操作，那么与估算相比可以节省更多（或更少）空间。仅当一页中的压缩行数多于未压缩行数，重新打包操作才能节约更多空间。如果重新打包操作释放空间，那么收缩操作才可在数据库空间级别节约空间。

相关链接

《*SinoDB* 管理员参考》：估计压缩操作的输出 *(SQL* 管理 *API)*

压缩字典

压缩字典是一个库，该库是由数据 或索引键中频繁出现的模式，以及替换这些模式的符号数构成。

对于每个压缩分段、每个压缩的非分段表、数据库空间中的每个压缩简单大对象，和每个压缩索引分区都存在一个压缩字典。

压缩字典是使用从至少包含 2,000 行的分段或非分段表随机采样的数据，或从至少包含 2,000 个键的索引随机采取的数据所构建的。通常情况下，存储压缩字典需要大约 100 KB 的空间。

压缩字典最多可存储 3,840 个模式，每个模式的长度为 2 到 15 个字节。（超过 7 个字节的模式将减少压缩字典中能存储的模式数。）每一个模式是由一个压缩行中的 12 位符号数来表示。输入行映像中的一序列字节必须精确与字典中某个完整模式匹配才可压缩。如果行与字典匹配的模式不足，那么可能不是可压缩的，因为不完全匹配的输入行的每个字节在压缩映像中将替换为 12 位（1.5 字节）。

SinoDB® 尝试捕获最佳的可压缩模式（模式频率乘以长度）。数据通过以下方式压缩：将出现的模式替换为字典中的相应符号数，并将出现的不匹配任何模式的字节替换文特殊保留的符号数。

数据库空间中所有表或分段的压缩字典存储在该数据库空间中的隐藏字典表中。sysmaster 数据库中的 syscompdicts_full 表和 syscompdicts 视图提供有关压缩字典的信息。

相关链接

《SinoDB 管理员参考》: syscompdicts_full

移动压缩数据工具

可使用 SinoDB® 数据迁移实用程序在数据库间移动压缩数据。

不能使用 onunload 和 onload 实用程序将压缩数据从一个数据库移动到另一个数据库。在使用 onunload 和 onload 实用程序之前，您必须将压缩表和分段中的数据解压缩。

dbexport 实用程序用于解压缩压缩数据。因此，必须在使用 dbimport 实用程序导入数据之后重新压缩数据。

相关链接

《SinoDB 迁移指南》: 数据迁移实用程序

查看压缩信息

可查看压缩统计信息、有关压缩字典的信息，以及压缩字典。

下表描述可用于查看压缩信息的方法。

表 31: 查看压缩信息的方法

方法	描述
SinoDB® 开放管理工具 (OAT)	显示是否压缩了表、分段或索引，以及存储优化后的估算存储节省。
oncheck -pT 或 oncheck -pt 命令	在输出的 "Compressed Data Summary" 部分显示任何压缩项的统计信息。如果未压缩任何项，那么输出中不会出现 "Compressed Data Summary" 部分。 例如，对于行数据，oncheck -pT 显示表或表片段中任何压缩行的数量，以及表或表分段的行压缩百分比。
onlog -c 选项	使用压缩字典扩展压缩数据，并显示压缩日志记录的未压缩内容。
onstat -g dsk 选项	显示有关当前运行的压缩操作的进度的信息。
onstat -g ppd 选项	显示有关当前打开的压缩分段（也称为分区）的现有活动的压缩字典的信息。此选项显示的信息与 sysmaster 数据库中 syscompdicts 视图显示的信息相同。
sysmaster 数据库中的 syscompdicts_full 表	显示有关压缩字典和压缩字典二进制对象的元数据。 仅用户 informix 可访问该表。
sysmaster 数据库中的 syscompdicts 视图	显示的信息与 syscompdicts_full 表中显示的信息相同，但出于安全原因，其中不包括 dict_dictionary 列，该列中包含压缩字典二进制对象。
UNLOAD TO 'compression_dictionary_file' SELECT * FROM sysmaster:syscompdicts_full; SQL 语句	查看文件中的压缩字典。

相关链接

存储优化方法 在第214页

《*SinoDB* 管理员参考》: *onstat -g dsk* 命令: 显示当前正在运行的压缩操作的进度
《*SinoDB* 管理员参考》: *onstat -g ppd* 命令: 显示分区压缩字典信息
《*SinoDB* 管理员参考》: *oncheck -pt* 和 *-pT*: 显示表或分段的表空间
《*SinoDB* 管理员参考》: *onlog* 实用程序
《*SinoDB* 管理员参考》: *syscompdicts_full*

向表装载数据

可以按以下方式向现有的表装载数据。

装载数据的方法	TEXT 或 BYTE 数据	CLOB 或 BLOB 数据	参阅
DB-Access LOAD 语句	是	是	《*SinoDB® SQL* 指南: 语法》中的 LOAD 语句
dbload 实用程序	是	是	《*SinoDB®* 迁移指南》
dbimport 实用程序	是	是	《*SinoDB®* 迁移指南》
SinoDB® ESQL/C 程序	是	是	《*SinoDB® ESQL/C* 程序员指南》
使用 EXTERNAL 源表插入 MERGE	是	是	《*SinoDB® SQL* 指南: 语法》
onload 实用程序	否	否	《*SinoDB®* 迁移指南》

重要: 数据库服务器把数据装载到数据库后，不包含任何压缩 TEXT 和 BYTE 数据的机制。

使用外部表移动数据

可以使用外部表来装载或卸载数据库数据。

您可以发出一系列用于执行以下功能的 SQL 语句：

- 将运行数据高效传输到其他系统或从其他系统传入
- 以 SinoDB® 内部数据格式跨平台传输数据文件
- 使用数据库服务器在定界 ASCII、固定 ASCII 和 SinoDB® 内部（原始）表示法之间转换数据。
- 使用 SQL INSERT 和 SELECT 语句来指定数据库表中新列的数据映射
- 提供平行标准 INSERT 操作，即可在不删除索引的情况下装载数据
- 使用命名管道来支持在存储设备（包括磁带机和直接网络连接）中装载和卸载数据
- 维护运行期间的装载和卸载数据统计信息的记录
- 执行快速（高速）和高级（数据检查）传输

可使用 DB-Access 发出 SQL 语句或将其嵌入 ESQL/C 程序中。

外部表

外部表是不受 SinoDB® 数据库服务器管理的数据文件。外部表的定义包括数据格式类型、外部数据描述字段和全局参数。

为了将外部数据映射到内部数据，数据库服务器将外部数据视为外部表。将外部数据视为表提供了将数据移入或移出数据库并指定数据转换的有效方法。

数据库服务器运行装载任务时，将从外部源读取数据，并执行创建行所需的转换，然后将行插入表中。数据库服务器将错误写入拒绝文件。

如果不能转换外部表中的数据，可指定数据库服务器将记录以及失败原因写入拒绝文件中。要做到这个，请在 CREATE EXTERNAL TABLE 语句中指定 REJECTFILE 关键字。

数据库服务器提供了若干不同的转换机制，这些机制在数据库服务器内部执行，因而在转换任务期间提供最大性能。数据库服务器优化固定格式和定界格式的 ASCII 和 SinoDB® 数据表示法之间的数据转换。

要执行定制化的转换，可创建将转换数据写入命名管道的过滤器程序。然后，数据库服务器以一种常见格式从命名管道中读取其输入。

定义外部表

要定义外部表，需要使用 SQL 语句来描述数据文件、定义表，然后指定要装载或卸载的数据。

要设置装载和卸载任务，请发出以下一系列 SQL 语句：

- CREATE EXTERNAL TABLE 用于描述要装载或卸载的数据文件
- CREATE TABLE 用于定义要装载的表
- INSERT...SELECT 用于装载或卸载

以下步骤概述装载过程：

1. CREATE EXTERNAL TABLE 语句描述各种外部文件的位置和外部数据的格式，其位置可以是磁盘或管道（磁带机或直接网络连接）。以下示例为 CREATE EXTERNAL TABLE 语句：

```
CREATE EXTERNAL TABLE emp_ext
   ( name CHAR(18) EXTERNAL CHAR(18),
     hiredate DATE EXTERNAL CHAR(10),
     address VARCHAR(40) EXTERNAL CHAR(40),
     empno INTEGER EXTERNAL CHAR(6) )
USING (
   FORMAT 'FIXED',
    DATAFILES
      ("DISK:/work2/mydir/emp.fix")
    );
```

2. CREATE TABLE 语句定义要装载的表。以下样本 CREATE TABLE 语句定义 employee 表：

```
CREATE TABLE employee
   FRAGMENT BY ROUND ROBIN IN dbspaces;
```

3. INSERT...SELECT 语句用于映射外部数据与数据库表之间的移动。以下样本 INSERT 语句从外部表中装载 employee 表：

```
INSERT INTO employee SELECT * FROM emp_ext
```

   重要： 如果指定多个 *INSERT...SELECT* 语句来卸载数据，那么每个后续 INSERT 语句将覆盖数据文件。请对数据文件使用绝对路径。

向数据库装载数据时，SELECT 子句中的 FROM *table* 部分是 CREATE EXTERNAL 语句定义的外部表。向外部文件卸载数据时，SELECT 子句控制从数据库检索数据。

与 TEMP 表不同，外部表在删除前会在目录中保留定义。创建外部表时，可保存数据的外部描述以便复用。当将表卸载到 SinoDB® 内部数据表示法时，此操作特别有用，因为稍后可使用相同的外部表描述来重新装载该数据。

在 Windows™ 系统上，如果使用 DB-Access 实用程序或 dbexport 实用程序来将数据库表卸载到文件中，然后计划使用该文件作为外部表数据文件，那么必须在 CREATE EXTERNAL TABLE 语句中将 RECORDEND 定义为 '\012'。

外部表定义中包含定义外部数据文件中数据所需的所有信息，如下所示：

- 外部数据中的字段的描述。

- DATAFILES 子句。

  该子句指定：

  - 数据文件是位于磁盘或命名管道上。
  - 文件的路径名。
- The FORMAT clause.
- FORMAT 子句。

  该子句指定外部数据文件中的数据格式类型。数据库服务器转换多种数据格式的外部数据，这些格式包括定界和固定 ASCII 以及 SinoDB® 内部格式。
- 任何影响数据格式的全局参数。

如果将外部表直接映射到界定格式的内部数据库表，可使用 CREATE EXTERNAL TABLE 语句来定义列，并增添 SAMEAS *internal-table* 子句，而无需显示枚举列。

将列映射到其他列

如果数据文件要有不同的字段顺序（例如 empno、name、address、hiredate），可使用 INSERT 语句来映射列。首先，创建与外部表的列顺序相同的表。

```
CREATE EXTERNAL TABLE emp_ext
   (
   f01 INTEGER,
   f02 CHAR(18),
   f03 VARCHAR(40),
   f04 DATE
   )
USING (
   DATAFILES ("DISK:/work2/mydir/emp.dat"),
   REJECTFILE "/work2/mydir/emp.rej"
   );
INSERT INTO employee (empno, name, address, hiredate)
   SELECT * FROM emp_ext;
```

通过此方法，映射插入列以匹配外部表的字段顺序。

另一种对列重新排序的方法是使用 SELECT 子句来匹配数据库表的顺序。

```
INSERT INTO employee
   SELECT f02, f04, f03, f01 FROM emp_ext;
```

从命名管道装载数据和将数据卸载到命名管道

可以使用命名管道（也称为先进先出，FIFO）数据文件，以从非标准设备（如磁带机）装载和卸载到非标准设备。

与普通操作系统文件不同，命名管道没有 2 GB 的大小限制。操作系统打开并检查命名管道的文件末尾的方式与普通文件不同。

使用命名管道装载数据

可以使用命名管道从外部表来装载数据。

要使用命名管道从外部表装载数据，请执行以下步骤：

1. 在 SQL CREATE EXTERNAL TABLE 语句的 DATAFILES 子句中指定命名管道。
2. 创建在 DATAFILES 子句中指定的命名管道。

   使用操作系统命令来创建命名管道。

使用带有 -p 选项的 mknod UNIX™ 命令来创建命名管道。要避免在 UNIX™上出现妨碍管道打开的问题，请为管道读取程序和管道写入程序启动单独的 UNIX™ 进程，或使用 O_NDELAY 标记集打开管道。

3. 使用读取命名管道的程序打开命名管道。
4. 执行 SQL 的 INSERT 语句。

```
INSERT INTO employee SELECT * FROM emp_ext;
```

重要： 如果在执行 INSERT 语句之前没有创建和打开命名管道， INSERT 会成功，但不会装载任何行。

FIFO 虚拟处理器

数据库服务器使用 FIFO 虚拟处理器 (VP) 来读取和写入命名管道上的外部表。

FIFO 虚拟处理器的缺省数量为 1。

数据库服务器为您在 CREATE EXTERNAL TABLE 语句的 DATAFILES 子句中指定的每个命名管道使用依各 FIFO VP。例如，假设您使用以下 SQL 语句定义外部表：

```
CREATE EXTERNAL TABLE ext_items
   SAMEAS items
   USING (
      DATAFILES("PIPE:/tmp/pipe1",
                "PIPE:/tmp/pipe2",
                "PIPE:/tmp/pipe3"
      ));
```

如果对 FIFO VP 使用缺省值 1 ，那么数据库服务器在读取完 pipe1 中的所有数据之前不会读取 pipe2 中的数据；并且在读取完 pipe2 中的所有数据之前不会读取 pipe3 中的数据。

使用命名管道卸载数据

可以使用命名管道将数据从数据库卸载到外部表。

要使用命名管道将数据卸载到外部表，请执行以下步骤：

1. 在 SQL 的 CREATE EXTERNAL TABLE 语句或 SELECT INTO EXTERNAL 语句的 DATAFILES 子句中指定命名管道。

```
DATAFILES ("PIPE:/usr/local/TAPE")
```

2. 创建在 DATAFILES 子句中指定的命名管道。使用操作系统命令来创建命名管道。
3. 使用写入命名管道的程序打开命名管道。
4. 将数据卸载到命名管道

```
CREATE EXTERNAL TABLE emp_ext
   ( name CHAR(18) EXTERNAL CHAR(20),
     hiredate DATE EXTERNAL CHAR(10),
     address VARCHAR(40) EXTERNAL CHAR(40),
     empno INTEGER EXTERNAL CHAR(6) )
   USING (
      FORMAT 'FIXED',
      DATAFILES
      ("PIPE:/usr/local/TAPE")
       );

INSERT INTO emp_ext SELECT * FROM employee;
```

重要： 如果在执行 SELECT 或 INSERT 语句之前没有创建和打开命名管道，卸载将失败，并生成 *ENXIO* 错误消息（no such device or address）。

使用 PIPE 选项将数据从一个实例复制到另一个实例

可使用命名管道将数据从一个 SinoDB® 实例复制到另一个实例，而无需将数据写入中间文件。

可使用命名管道将数据从一个 SinoDB® 实例卸载，然后装载到另一个实例，而无需将数据写入中间文件。还可使用命名管道将数据从一个表复制到同一 SinoDB® 实例上的另一个表。在以下示例中，将数据从一个实例的源表复制到另一个实例的目标表。

首先必须根据所使用的硬件平台使用以下命令之一创建命名管道。在本例中，命名管道称为 pipe1。

```
% mkfifo /work/pipe1
% mknod /work/pipe1
```

执行以下步骤，将数据从源实例上的表复制到同一台计算机上目标实例中的表。

1. 在源实例上创建源表。在本例中，源表称为 source_data_table：

```
CREATE TABLE source_data_table
(
  empid    CHAR(5),
  empname  VARCHAR(40),
  empaddr  VARCHAR(100)
);
```

2. 在源实例上创建外部表。在本例中，外部表称为 ext_table：

```
CREATE EXTERNAL TABLE ext_table
(
  empid    CHAR(5),
  empname  VARCHAR(40),
  empaddr  VARCHAR(100)
)
USING
(DATAFILES
  (
    'PIPE:/work/pipe1'
  )
);
```

3. 在目标实例上创建目标表。在本例中，目标表称为 destin_data_table：

```
CREATE TABLE destin_data_table
(
  empid    CHAR(5),
  empname  VARCHAR(40),
  empaddr  VARCHAR(100)
);
```

4. 在目标实例上创建外部表。在本例中，外部表称为 ext_table：

```
CREATE EXTERNAL TABLE ext_table
(
  empid    CHAR(5),
  empname  VARCHAR(40),
  empaddr  VARCHAR(100)
)
USING
(DATAFILES
  (
    'PIPE:/work/pipe1_1'
  )
);
```

5. 从 UNIX™ shell 运行以下命令。该命令将数据从 /work/pipe1 重定向到 /work/pipe1_1

```
cat /work/pipe1 > /work/pipe1_1
```

6. 在目标实例上运行以下命令，将数据从命名管道定向到目标表：

```
INSERT INTO destin_data_table SELECT * FROM ext_table;
```

7. 在源实例上运行以下命令，将数据假脱机到命名管道：

```
INSERT INTO ext_table SELECT * FROM source_data_table;
```

可以通过在 DATAFILES 子句中插入多个 PIPE 语句并为每个语句创建一个命名管道的方式来使用多个管道。

监视装载或卸载操作

可监视外部表装载或卸载操作的状态。

您可能希望在以下情况监视装载或卸载操作：

- 如果希望经常装载和卸载相同表，以建构数据集市或数据仓库，那么监视作业进度可估算类似作业的时间，以供未来使用。
- 如果通过命名管道装载或卸载，那么监视 I/O 队列可确定是否有足够数量的 FIFO 虚拟处理器。

监视常用装载或卸载操作

可使用 onstat -g iof 命令查找要查看的文件中的全局文件描述符 (gfd)。然后，使用 onstat -g sql 命令监控装载和卸载操作。

The following example shows sample onstat -g iof command output.以下示例显示 onstat -g iof 命令输出样本。

```
AIO global files:
gfd path name      bytes read   page reads   bytes write   page writes  io/s
3   rootdbs        1918976      937          145061888     70831        36.5

      op type     count         avg. time
      seeks       0             N/A
      reads       937           0.0010
      writes      4088          0.0335
      kaio_reads  0             N/A
      kaio_writes 0             N/A
```

要确定装载或卸载操作是否可并行执行，请在 INSERT 语句之前执行 SET EXPLAIN ON 语句。SET EXPLAIN 输出显示以下计数：

- 优化器为 INSERT 语句选择的并行 SQL 运算符的数量
- 每个 SQL 运算符处理的行数

要监视装载操作，请运行 onstat -g sql 以获取会话 标识。

监视 FIFO 虚拟处理器

可以使用 onstat 命令监视 FIFO VP 的有效使用。

使用 onstat -g ioq 选项以显示正在等待执行 I/O 要求的每个 FIFO 队列的长度。以下示例显示样本输出。

```
AIO I/O queues:
q name/id    len maxlen totalops  dskread dskwrite  dskcopy
```

```
  fifo   0      0      0         0         0         0         0
   adt   0      0      0         0         0         0         0
   msc   0      0      1       153         0         0         0
   aio   0      0      9      3499      1013        77         0
   pio   0      0      2         3         0         2         0
   lio   0      0      2      2159         0      2158         0
   gfd   3      0     16     39860        38     39822         0
   gfd   4      0     16     39854        32     39822         0
   gfd   5      0      1         2         2         0         0
   gfd   6      0      1         2         2         0         0
...
   gfd  19      0      1         2         2         0         0
```

以上示例的样本输出中的 q name 字段显示队列的类型，如 fifo 表示 FIFO VP，或 aio 表示 AIO VP。如果 q name 字段显示 gfd 或 gfdwq，表示这是与输出的 id 字段匹配的全局文件描述符的文件的队列。磁盘文件在一个队列中同时有读取和写入请求。每个磁盘文件在 onstat -g ioq 输出中显示一行。管道有单独的读取和写入队列。每个管道在输出中显示两行：gfd 为读取请求和 gfdwq 为写入请求。

len 或 maxlen 字段的值最高为 4（对于装载）或 4 * number_of_writer_threads（对于卸载）。xuwrite 运算符控制写程序线程的数量。

使用 totalops 字段的值，而非 len 或 maxlen 字段的值，来监视文件或管道上完成的读或写请求数。totalops 字段表示从该文件中读取或写入了 34 KB 数据。如果 totalops 没有增加，表示文件或管道上的读或写操作已停止（因为 FIFO VP 正忙）。

要提高性能，请使用 onmode -p 命令来添加更多 FIFO VP。FIFO VP 的缺省数量为 1。在此样本输出中，FIFO 队列不包含任何数据。例如，如果通常定义两个以上的管道来装载或卸载，请使用以下样本 onmode 命令来增加 FIFO VP 的数量：

```
onmode -p +2 FIFO
```

也可以使用 onmode -p 命令来移除 FIFO VP。但不能将 FIFO VP 的数量设置为低于 1。

有关更多信息，请参阅《*SinoDB*® 管理员参考》。

高可用性集群环境中的外部表

在辅助服务器上使用外部表与在主服务器上使用的方式大致相同。

可在主服务器和辅助服务器上执行以下操作：

- 将数据从数据库表卸载到外部表：

  ```
  INSERT INTO external_table SELECT * FROM base_table WHERE ...
  ```

- 将数据从外部表装载到数据库表：

  ```
  INSERT INTO base_table SELECT * FROM external_table WHERE ...
  ```

在 SDS、RSS 或 HDR 辅助服务器上装载数据比在主服务器上装载数据的速度慢。

辅助服务器不支持 CREATE EXTERNAL TABLE 语句和 SELECT ... INTO EXTERNAL ... 语句。

将数据从数据库表中卸载到外部表时，将在辅助服务器上，而非主服务器上，创建数据文件。在辅助服务器上创建的外部表数据文件不会自动传输到主服务器上，在主服务器上创建的外部表数据文件也不会自动传输到辅助服务器上。

在主服务器上创建外部表时，仅外部表的模式会复制到辅助服务器，而不会复制数据文件。

要在主服务器和辅助服务器之间同步外部表，可以将外部表文件从主服务器复制到辅助服务器，或者使用以下步骤 ：

1. 在主服务器上：

a. 创建与外部表的模式相同的临时表。
b. 填充临时表：

```
INSERT INTO dummy_table SELECT * FROM external_table
```

2. 在辅助服务器上：

使用以下命令来填充外部表：

```
INSERT INTO external_table SELECT * FROM dummy_table
```

外部表的系统目录条目

可以查询系统目录表以确定外部表的状态。

每次创建外部表时，SinoDB® 都会更新 sysexternal 和 sysextdfiles 系统目录表。外部格式类型（fmttype）指定为 FIXED 时，将更新 sysextcols 系统目录表。

表 32: 描述外部表文件的系统目录表

表名	描述
sysexternal	存储每个外部表文件的名称和属性
sysextdfiles	存储每个外部表文件的文件路径和目录
sysextcols	存储 FIXED-类型的外部表的每个列的属性

有关更多信息，请参阅 《*SinoDB® SQL* 指南: 参考》

创建外部表时，将会在 systables 系统目录中插入一行，但是，nrows（行数）和 npused （所用数据页数）列可能无法精确反映外部表的行数和所用数据页数，除非在外部表创建时指定 NUMROWS 子句。

创建外部表时没有为 NUMROWS 子句指定值时，由于数据位于数据库外的数据文件中，所以 SinoDB® 不能确定外部表的行数。SinoDB® 将会使用一个很大的值（MAXINT - 1）来更新 systables 系统目录中的 nrows 列，并基于 nrows 值来计算所用的数据页数。之后，优化器将使用 npused 和 nrows 中存储的值来确定最有效的执行计划。尽管无需精确指定 NUMROWS 子句，但是指定的越精确，nrows 和 npused 的值就越精确。

使用外部表时的性能注意事项

想要使用 SQL 命令处理 ASCII 文件中的数据或将数据从外部数据文件中装载到 RAW 数据库表时，可使用外部表。

将信息装载到数据库的方法有几种，包括 ：

- LOAD FROM ... INSERT INTO... DB-Access 命令
- dbimport 实用程序
- 外部表

外部表为装载数据到无索引的 RAW 表提供最佳性能。

注: 装载数据之前锁定外部表将提高装载性能。

管理外部表装载和卸载操作产生的错误

可以管理外部表装载和卸载操作期间发生的错误。

这些主题描述如何使用拒绝文件和错误消息来管理错误，以及如何恢复已装载到数据库的数据。

拒绝文件

装载期间发生转换错误的行会写入执行转换操作的服务器上的拒绝文件中。

CREATE EXTERNAL TABLE 语句的 REJECTFILE 关键字用于确定指定给拒绝文件的名称。

可不使用拒绝文件，而是使用 CREATE EXTERNAL TABLE 语句的 MAXERRORS 关键字指定数据库服务器停止装载数据之前允许的错误数。（如果不设置 MAXERRORS 关键字，数据库服务器将无视错误数，处理所有数据。）

数据库服务器将在开始执行装载之前删除拒绝文件（如果有）。仅当装载期间发生错误时，才会重新创建和写入拒绝文件。

拒绝文件条目是包含以逗号分隔的以下字段的单行：

```
file name, record, reason-code, field-name: bad-line
```

file name

输入文件的名称

record

输入文件中发现到错误的记录号

reason-code

错误说明

field-name

外部字段名称（行中的第一个错误位置）或 <none>（如果拒绝不特别针对特定列）

bad-line

坏行本身（仅对定界或固定 ASCII 文件）

装载操作以 ASCII 格式写入 *file name* 、*record*、*ield-name* 和 *reason-code*。

bad-line 信息由输入文件的类型决定：

- 对于定界文件或固定文本文件，整个坏行将直接复制到拒绝文件中。但是如果定界格式表具有 TEXT 或 BYTE 列，那么拒绝文件中将不包含任何坏数据。装载操作将仅为每个拒绝的行生成一个标题。
- 对于 SinoDB® 内部数据文件，坏行不会放入拒绝文件中，因为您无法编辑文件中的二进制表示。但是，拒绝文件中仍然会报告 *file name*、*record*、*reason-code* 和 *field-name*，以便可以隔离该问题。

以下类型的错误可以导致行被拒绝。

CONSTRAINT ***constraint name***

违反该约束。

CONVERT_ERR

任何字段遇到转换错误。

MISSING_DELIMITER

找不到定界符。

MISSING_RECORDEND

找不到记录结尾。

NOT NULL

field-name 中为 null。

ROW_TOO_LONG

输入记录长度超过 2 GB。

外部表错误消息

与外部表相关的大多数错误消息都在 -26151 到 -26199 范围内。

其他消息为 -615、-999、-23852 和 -23855。在消息中， *n macro* 和 *r macro* 指从替换字符 %r(*first..last*) 生成的值。有关错误消息列表，请参阅《*SinoDB*® 错误消息》或使用 finderr 实用程序。有关违例表错误消息的信息，请参阅《*SinoDB*® 管理员参考》。

外部表的表类型的可恢复性

装载数据时，数据库服务器会将检查表的可恢复性级别。

- 如果表的记录类型是 RAW，那么数据库服务器可以使用轻量级追加（或 EXPRESS）方式装载数据并处理检查约束。但是，如果数据库服务器在 EXPRESS 方式下插入数据行到 RAW 表时崩溃，这种未记录的轻量级追加操作是不会回滚，且表可能处于某种未知状态。
- 仅 DELUXE 方式支持数据可恢复性。DELUXE 方式使用记录的常规插入。要恢复 EXPRESS 方式装载失败的数据，请恢复到最近的 0 级备份。对于此级别的可恢复性，表类型必须为 STANDARD。

有关恢复 RAW 或 STANDARD 记录类型的表的信息，请参阅《*SinoDB*® 备份和还原指南》。

第 III 部分

日志记录和日志管理

日志记录

These topics describe logging of SinoDB® databases and addresses the following questions:这些主题描述了 SinoDB® 数据库日志记录并解决了以下问题：

- 哪些数据库服务器进程需要日志记录？
- 什么是事务日志记录？
- 记录哪种数据库服务器活动？
- 什么是数据库日志记录状态？
- 谁可以设置或更改数据库日志记录状态？

单一数据库服务器实例所管理的所有数据库均将其日志记录存储在同一逻辑日志中，无论它们是否使用事务日志记录。大部分数据库用户可能会关心事务日志记录是否已缓冲，或表是否使用日志记录。

如果想要更改数据库日志记录的状态，请参阅 日志记录状态或模式的设置或更改 在第239页。

需要日志记录的数据库服务器进程

在 SinoDB® 运行时（处理事务、追踪数据存储和确保数据一致性），SinoDB® 自动为所执行的一些操作生成逻辑日志记录 。大多数情况下，数据库服务器不会进一步使用逻辑日志记录。但是，当数据库服务器需要回滚事务时，例如在系统故障后运行快速恢复时，逻辑日志记录将会很关键。逻辑日志记录是数据恢复机制的核心。

数据库服务器将逻辑日志记录存储在逻辑日志中。逻辑日志由 逻辑日志文件 组成，数据库服务器在硬盘上管理它们直到这些文件已安全地转为脱机（备份）。数据库服务器管理员将保留备份的逻辑日志文件以供数据复原期间的使用需要，或直到管理员决定不再需要这些文件用于数据复原。有关逻辑日志的更多信息，请参阅 逻辑日志 在第243页 。

逻辑日志本身长度为变动的。此种方式增加了可写入逻辑日志缓冲区中一页的逻辑日志记录数。但是，数据库服务器经常在页满之前清空逻辑日志缓冲区。有关逻辑日志记录格式的更多信息，请参阅《*SinoDB®* 管理员参考》中有关解释逻辑日志记录的主题。

数据库服务器在执行各种恢复数据和确保数据一致性的功能时使用逻辑日志记录，如下所示：

事务回滚

如果数据库正在使用事务日志记录，并且必须回滚事务时，那么数据库服务器将使用逻辑日志记录来逆转事务期间所做的更改。有关更多信息，请参阅 事务日志记录 在第233页。

快速恢复

如果数据库服务器以不受控方式关闭，那么数据库服务器将使用逻辑日志记录来恢复自尚未清空到磁盘的最旧更新以来所发生的所有事务，并回滚任何未提交的事务。（当共享内存中和磁盘上的所有数据相

同时，它们可说是物理一致性。）数据库服务器在快速恢复中使用逻辑日志记录，它将把整个数据库服务器返回到最近逻辑日志记录的逻辑一致性状态。（有关更多信息，请参阅 检查点之后的快速恢复 在第268页。）

数据复原

数据库服务器使用最近的存储空间和逻辑日志备份来重新创建数据库服务器系统，直至最近备份的逻辑日志记录点为止。逻辑复原将应用自上次存储空间备份以来的所有日志记录。

延迟检查

如果事务使用 SET CONSTRAINTS 语句将检查设置为 DEFERRED，那么数据库服务器直到事务提交后才会检查约束。如果事务提交时发生约束错误，那么数据库服务器会使用逻辑日志记录来回滚事务。有关更多信息，请参阅 《*SinoDB® SQL* 指南: 语法》中的 SET Database Object Mode 语句。

级联删除

引用约束的级联删除使用逻辑日志记录以确保如果父行删除并且在删除子行之前系统故障可以回滚事务。有关表继承的信息，请参阅 《*SinoDB®* 数据库设计和实现指南》。有关主键和外键约束的信息，请参阅 《*SinoDB® SQL* 指南: 教程》。

分布式事务

分布式事务所涉及的每个数据库服务器将保留事务的逻辑日志记录。即使在执行该事务的其中一个数据库服务器发生故障，此过程也可确保数据完整性和一致性。有关更多信息，请参阅 两阶段提交和逻辑日志记录 在第433页。

数据复制

使用 HDR 辅助服务器、SD 辅助服务器和 RS 辅助服务器的数据复制环境使用逻辑日志记录来维护主服务器和辅助服务器的数据一致性，从而使其中一个数据库服务器在另一个数据库服务器故障时可以迅速用作备份数据库服务器。有关更多详细信息，请参阅 数据复制的工作原理 在第311页。

Enterprise Replication

必须对 Enterprise Replication 使用数据库日志记录，因为它可以从逻辑日志记录中复制数据。有关更多信息，请参阅 《*SinoDB® Enterprise Replication* 指南》。

事务日志记录

当数据库中的 SQL 数据操作语句生成逻辑日志记录时，则会认为该数据库或表具有或使用事务日志记录。

数据库日志记录 状态 指示数据库是否使用事务记录。 日志缓冲模式 指示数据库是使用已缓冲还是未缓冲日志记录，或是使用符合 ANSI 标准的日志记录。有关更多信息，请参阅 数据库日志记录状态 在第237页和管理数据库日志记录模式 在第239页。

在创建数据库时，指定数据库是否使用 事务日志记录 ，如果使用，将使用哪种日志缓冲机制。数据库创建之后，您可以关闭数据库日志记录或更改成已缓冲日志记录。即使关闭了所有数据库的事务日志记录，数据库服务器始终会记录一些事件。有关更多信息，请参阅 始终纪录的活动 在第233页 和 *X/Open DTP* 环境中的数据库日志记录 在第239页。

可以使用数据库中的日志记录表或非日志记录表。创建表的用户指定表的类型。即使使用非日志记录表，数据库服务器始终会记录一些事件。有关更多信息，请参阅 *SinoDB* 的表类型 在第156页。

SQL 语句和数据库服务器活动的日志记录。

数据库服务器中可能有以下三类记录活动：

始终纪录的活动

有些数据库操作始终会生成逻辑日志记录，即使关闭事务日志记录或使用非日志记录表。

对于永久表，以下操作始终记录：

- 某些 SQL 语句，包括 SQL 数据定义语句

- 存储空间备份
- 检查点
- 对数据库服务器配置的管理更改，例如添加块或数据库空间
- 为表分配新扩展数据块
- 对数据库的日志记录状态的更改
- 智能大对象操作：
  - 创建
  - 删除
  - 分配和取消分配扩展数据块
  - 截断
  - 组合和分割块可用列表页
  - 更改 LO 头和 LO 引用计数
- 智能大对象空间元数据
- BLOB 空间

下表列出了即使事务日志记录关闭也记录的操作语句。

```
ALTER ACCESS_METHOD
ALTER FRAGMENT
ALTER FUNCTION
ALTER INDEX
ALTER PROCEDURE
ALTER ROUTINE
ALTER SECURITY LABEL COMPONENT
ALTER SEQUENCE
ALTER TABLE
ALTER TRUSTED CONTEXT
ALTER USER
CLOSE DATABASE
CREATE ACCESS_METHOD
CREATE AGGREGATE
CREATE CAST
CREATE DATABASE
CREATE DISTINCT TYPE
CREATE EXTERNAL TABLE
CREATE FUNCTION
CREATE FUNCTION FROM
CREATE INDEX
CREATE OPAQUE TYPE
CREATE OPCLASS
CREATE PROCEDURE
CREATE PROCEDURE FROM
CREATE ROLE
CREATE ROUTINE FROM
CREATE ROW TYPE
CREATE SCHEMA
CREATE SECURITY LABEL
CREATE SECURITY LABEL COMPONENT
CREATE SECURITY POLICY
CREATE SEQUENCE
CREATE SYNONYM
```

```
CREATE TABLE
CREATE TEMP TABLE
CREATE TRIGGER
CREATE TRUSTED CONTEXT
CREATE USER
CREATE VIEW
CREATE XADATASOURCE
CREATE XADATASOURCE TYPE
DROP ACCESS_METHOD
DROP AGGREGATE
DROP CAST
DROP DATABASE
DROP FUNCTION
DROP INDEX
DROP OPCLASS
DROP PROCEDURE
DROP ROLE
DROP ROUTINE
DROP ROW TYPE
DROP SECURITY
DROP SEQUENCE
DROP SYNONYM
DROP TABLE
DROP TRIGGER
DROP TRUSTED CONTEXT
DROP TYPE
DROP USER
DROP VIEW
DROP XADATASOURCE
DROP XADATASOURCE TYPE
GRANT
GRANT FRAGMENT
RENAME COLUMN
RENAME DATABASE
RENAME INDEX
RENAME SECURITY
RENAME SEQUENCE
RENAME TABLE
RENAME TRUSTED CONTEXT
RENAME USER
REVOKE
REVOKE FRAGMENT
TRUNCATE
UPDATE STATISTICS
SAVE EXTERNAL DIRECTIVES
SET CONSTRAINTS
SET Database Object Mode
SET INDEXES
SET TRIGGERS
START VIOLATIONS TABLE
```

STOP VIOLATIONS

为具有事务日志记录的数据库纪录活动

如果数据库使用了事务日志记录，那么以下的 SQL 语句将生成一个或多个日志记录。如果回滚这些语句，那么回滚也会生成日志记录。

DELETE
FLUSH
INSERT
LOAD
MERGE
PUT
SELECT INTO TEMP
UNLOAD
UPDATE

以下 SQL 语句在特殊情况下才生成日志。

表 33: 在特殊情况下生成日志的 SQL 语句。

SQL 语句	语句生成的日志记录
BEGIN WORK	除非数据库使用了事务日志记录，否则返回错误。如果事务进行了其他日志记录工作，就会产生一条日志记录。
COMMIT WORK	除非数据库使用了事务日志记录，否则返回错误。如果事务进行了其他日志记录工作，就会产生一条日志记录。
ROLLBACK WORK	除非数据库使用了事务日志记录，否则返回错误。如果事务进行了其他日志记录工作，就会产生一条日志记录。
EXECUTE	该语句是否生成日志记录取决于运行的命令。
EXECUTE FUNCTION	该语句是否生成日志记录取决于执行的函数。
EXECUTE IMMEDIATE	该语句是否生成日志记录取决于运行的命令。
EXECUTE PROCEDURE	该语句是否生成日志记录取决于执行的过程。

未记录的活动

一些 SQL 语句是不记录的。

无论数据库的日志记录模式，以下 SQL 语句均不产生日志记录。

ALLOCATE COLLECTION
ALLOCATE DESCRIPTOR
ALLOCATE ROW
CLOSE
CONNECT
DATABASE
DEALLOCATE COLLECTION
DEALLOCATE DESCRIPTOR
DEALLOCATE ROW
DECLARE
DESCRIBE
DISCONNECT
FETCH

FREE
GET DESCRIPTOR
GET DIAGNOSTICS
INFO
LOCK TABLE
OPEN
OUTPUT
PREPARE
RELEASE SAVEPOINT
SAVEPOINT
SELECT
SET AUTOFREE
SET COLLATION
SET CONNECTION
SET DATASKIP
SET DEBUG FILE
SET DEFERRED_PREPARE
SET DESCRIPTOR
SET ISOLATION
SET LOCK MODE
SET LOG
SET OPTIMIZATION
SET PDQPRIORITY
SET ROLE
SET SESSION AUTHORIZATION
SET STATEMENT CACHE
SET TRANSACTION
SET Transaction Mode
SET USER PASSWORD
UNLOCK TABLE
WHENEVER
SET ENVIRONMENT
SET EXPLAIN

对于临时数据库空间的临时表，不记录任何内容，甚至对于其他表类型记录的 SQL 语句也不记录。如果在 DBSPACETEMP 配置参数的值中包含了临时（非日志记录）数据库空间，那么数据库服务器会先将非日志记录表放在这些临时数据库空间中。

数据库日志记录状态

必须对数据库使用事务日志记录以利用需要日志记录的数据库服务器进程 在第232页中列出的功能。

数据库服务器管理的每个数据库均有日志记录状态。日志记录状态指示了数据库是否使用事务日志记录，以及（如果使用）数据库是运用哪一种日志缓冲机制。要查找出数据库的事务日志记录状态，请按监视数据库的日志记录模式 在第243页中的说明，使用数据库服务器实用程序。数据库日志记录状态指示以下日志记录类型中的任意一种：

- 未缓冲的事务日志记录
- 已缓冲的事务日志记录
- 符合 ANSI 标准的事务日志记录
- 无日志记录

所有逻辑日志记录在数据库服务器将其写入磁盘上的逻辑日志之前，均会经过共享内存中的逻辑日志缓冲区。但是，数据库服务器清空逻辑日志缓冲区的时刻对于已缓冲的事务日志记录和未缓冲的事务日志记录是不同的。有关更多信息，请参阅图 18: 数据库服务器如何使用共享内存 在第106页和清空逻辑日志缓冲区 在第125页。

未缓冲的事务日志记录

如果事务是对使用未缓冲日志记录的数据库进行的，那么逻辑日志缓冲区的记录保证在事务提交处理期间写入磁盘。当控制权在 COMMIT 语句之后（并且，对于分布式事务，在 PREPARE 语句之前）返回应用程序时，逻辑日志记录在磁盘上。一旦提交缓冲区中的任何事务（即，将提交记录写入逻辑日志缓冲区），数据库服务器就立即清空记录。

数据库服务器清空缓冲区时，仅将已使用的页写入磁盘。已使用的页包括仅部分满的页，但这样就浪费了一些空间。由于这个原因，与同一数据库服务器上的所有数据库均使用已缓冲的日志记录相比，磁盘上的逻辑日志文件会更快填满。

对于大部分数据库而言，未缓冲的日志记录是最好的选择，因为它保证所有已提交的事务可以恢复。发生故障时，仅故障发时未提交的事务才会丢失。但是，如使用未缓冲的日志记录，数据库服务器会更频繁地将逻辑日志缓冲区清空到磁盘，且缓冲区会包含更多部分填满的页，因此，它比已缓冲的日志记录能更快填满逻辑日志。

已缓冲的事务日志纪录

如果事务是对使用已缓冲日志纪录的数据库进行的，那么该记录会尽可能久地保留（已缓冲）在逻辑日志缓冲区。直到发生以下状况之一，这些记录才会从共享内存中的逻辑日志缓冲区清空到磁盘上的逻辑日志：

- 缓冲区已满。
- 具有未缓冲日志记录的数据库上的事务提交会清空缓冲区。
- 出现检查点。
- 关闭连接。

如果使用已缓冲日志记录，发生故障时，不能期望数据库服务器恢复故障发生时在逻辑日志缓冲区中的事务。因此，可能会丢失一些已提交的事务。作为此风险的回报，就是变更期间性能会稍有提高。只要您在故障发生的情况下可重新创建更新，那么已缓冲日志记录最适用于频繁更新的数据库（当更新速度很重要时）。您可调整逻辑日志缓冲区的大小以便为您的系统在性能和因故障而丢失事务的风险之间找到可接受的平衡点。

符合 ANSI 的事务日志纪录

符合 ANSI 标准的数据库记录状态表明该数据库所有者使用 MODE ANSI 关键字。符合 ANSI 标准的数据库始终使用未缓冲事务日志记录，对事务处理强制执行 ANSI 规则。您无法更改符合 ANSI 标准的数据库的缓冲状态。

无数据库日志记录

如果关闭数据库的日志记录，那么将不会记录事务，但会记录其他操作。有关更多信息，请参阅始终纪录的活动 在第233页。通常，在装载数据或执行查询时关闭数据库的日志记录。

如果您对恢复源满意，那么您可以决定数据库不使用事务日志记录以减少数据服务器的进程量。例如，如果要将许多行从可恢复的源（如磁带机或 ASCII 文件）装载到数据库，那麼可能不需要事务日志記錄，并且在不使用事务日志记录的情况下装载会进行的更快。但是，如果数据库中有其他活动的用户，那么直到重新启动日志记录（这必须等待 0 级备份）之前将不会有其事务的逻辑日志记录 。

具有不同日志缓冲状态的数据库

数据库服务器上的所有数据均使用相同的逻辑日志和逻辑日志缓冲区。因此，具有不同日志缓冲状态的各数据库的事务可以写入相同的逻辑日志缓冲区。在这样的情况下，如果具有已缓冲日志记录的数据库上和具有未缓冲日志记录的数据库上均存在事务，那么数据库服务器在缓冲区满时或在具有未缓冲日志记录的数据库上的事务完成时清空缓冲区。

X/Open DTP 环境中的数据库日志记录

在 X/Open 分布式事务处理环境中的数据库（DTP）必须使用未缓冲日志记录。未缓冲日志记录确保数据库服务器逻辑日志始终处于一致状态，并可与事务管理器同步。如果在 X/Open DTP 环境中打开使用缓冲日志记录的数据库，那么数据库的状态会自动更改为未缓冲日志记录。数据库服务器支持符合 ANSI 标准和非 ANSI 标准的数据库。有关更多信息，请参阅 事务管理器 在第421页。

日志记录状态或模式的设置或更改

使用 CREATE DATABASE 语句创建数据库的用户为该数据库建立了日志记录状态或缓冲模式。有关 CREATE DATABASE 语句的更多信息，请参阅《*SinoDB® SQL* 指南: 语法》。

如果 CREATE DATABASE 语句没有指定日志记录状态，那么创建无日志记录的数据库。

仅数据库服务器管理员可以更改日志记录状态。管理数据库日志记录模式 在第239页描述了该主题。普通用户无法更改数据库日志记录状态。

如果数据库不使用日志记录，那么无需考虑是已缓冲日志记录还是未缓冲日志记录更为合适。如果为数据库指定日志记录但未指定缓冲模式，那么默认为未缓冲日志记录。

用户可以在会话期间从未缓冲日志记录切换为已缓冲（但不符合 ANSI 标准）日志记录，也可以从已缓冲日志记录切换为未缓冲日志记录。在应用程序中 SET LOG 语句将执行此变更。有关 SET LOG 语句的更多信息，请参阅《*SinoDB® SQL* 指南: 语法》。

管理数据库日志记录模式

可监视和更改数据库日志记录模式。

本节中主题提供有关以下内容的信息：

- 了解数据库日志记录模式
- 使用 ondblog 更改数据库日志记录模式
- 使用 ontape 更改数据库日志记录模式
- 监视事务日志记录

作为数据库服务器管理员，您可以如下所示更改数据库的日志记录模式：

- 将事务日志记录从已缓冲更改为未缓冲。
- 将事务日志记录从未缓冲更改为已缓冲。
- 使数据库符合 ANSI 标准。
- 向数据库添加事务日志记录（已缓冲或未缓冲）。
- 结束数据库的事务日志记录。

有关数据库日志记录模式、何时使用事务日志记录，以及何时缓冲事务日志记录的信息，请参阅日志记录 在第232页。要查明数据库的当前日志记录方式，请参阅监视数据库的日志记录模式 在第243页。

有关使用 SQL 管理 API 命令（而不是一些 ondblog 和 ontape 命令）的信息，请参阅使用 *SQL* 管理 *API* 执行远程管理 在第458页和《*SinoDB®* 管理员参考》。

更改数据库日志记录模式

可以使用 ondblog 或 ontape 来添加或更改日志记录。然后使用 ON-Bar 或 ontape 来备份数据。使用 ON-Bar 或 ontape 时，数据库服务器必须处于联机，管理或静默模式。

有关 ON-Bar 和 ontape 的信息，请参阅《*SinoDB®* 备份和还原指南》。

下表显示数据库服务器管理员能如何更改数据库日志记录模式。某些日志记录模式更改是立刻生效的，而另一些更改需要 0 级备份。

表 34: 日志记录模式转换

转换源:	转换为无日志记录	转换为未缓冲日志记录	转换为已缓冲日志记录	转换为符合 ANSI 标准
无日志记录	不适用	（受影响的存储空间的）0 级备份	（受影响的存储空间的）0 级备份	（受影响的存储空间的）0 级备份
未缓冲的日志记录	是	不适用	是	是
已缓冲的日志记录	是	是	不适用	是
符合 ANSI 标准	非法	非法	非法	不适用

更改数据库日志记录模式有以下影响：

- 在更改日志记录状态时，数据库服务器会对数据库加上互斥锁定以防止其他用户访问该数据库，并在更改完成时释放该锁定。
- 如果在日志记录模式更改期间发生故障，那么在恢复数据库服务器数据后，请检查 sysmaster 数据库的 sysdatabases 表中标志的日志记录模式。有关更多信息，请参阅监视数据库的日志记录模式 在第243页。然后重新尝试更改日志记录模式。
- 在选择了已缓冲或未缓冲日志记录之后，应用程序可以使用 SQL 语句的 SET LOG 将日志记录模式从一种改为另一种。此更改在会话期间一直持续有效。有关 SET LOG 的信息，请参阅《*SinoDB® SQL* 指南: 语法》。
- 如果向数据库添加日志记录，该更改将直到该数据库的所有存储空间的下一次 0 级备份后才完成。

使用 ondblog 更改数据库日志记录模式

可以使用 ondblog 实用程序为一个或多个数据库更改日志记录模式。如果向数据库添加日志记录，那么必须在更改生效之前对包含该数据库的数据空间创建 0 级备份。有关更多信息，请参阅《*SinoDB®* 管理员参考》中有关使用 ondblog 的主题。

使用 ondblog 更改缓冲模式

要将 stores_demo 数据库的缓冲模式从已缓冲日志记录更改为未缓冲日志记录，请运行以下命令：

```
ondblog unbuf stores_demo
```

要将 stores_demo 数据库的缓冲模式从未缓冲日志记录更改为已缓冲日志记录，请运行以下命令：

```
ondblog buf stores_demo
```

使用 ondblog 取消日志记录模式更改

在下一次 0 级备份之前，请运行以下命令以取消日志记录模式更改要求：

```
ondblog cancel stores_demo
```

无法取消立即执行的日志记录更改。

使用 ondblog 结束日志记录

要结束名为 dbfile 的文件中所列出的两个数据库的日志记录，请运行以下命令：

```
ondblog nolog -f dbfile
```

使用 ondblog 使数据库符合 ANSI 标准

要使用 ondblog 使名为 stores_demo 数据库成为符合 ANSI 标准的数据库，请运行以下命令：

```
ondblog ansi stores_demo
```

更改符合 ANSI 标准的数据库的日志记录模式

在将数据库创建或转换为 ANSI 模式之后，就无法轻易将其更改为任何其他日志记录模式。如果意外地将数据库转换为 ANSI 模式，请遵循以下步骤来更改日志记录模式：

要更改日志记录模式：

1. 要卸载数据，请使用 dbexport 或其他迁移实用程序。

   dbexport 实用程序创建 schema 文件。

   有关如何装载和卸载数据的信息，请参阅《*SinoDB*® 迁移指南》。
2. 要重建具有已缓冲日志记录的数据库并载入数据，请使用 dbimport -l buffered 命令。

   要重建具有未缓冲日志记录的数据库并载入数据，请使用 dbimport -l 命令。

用 ontape 更改数据库日志纪录模式

如果使用 ontape 作为备份工具，那么可以使用 ontape 来更改数据库的日志纪录模式。

使用 ontape 打开事务日志纪录

在更改数据库日志记录模式之前，请参阅更改数据库日志记录模式 在第239页。

在创建 0 级备份同时使用 ontape 向数据库添加日志记录。

例如，使用 ontape 向名为 stores_demo 的数据库添加已缓冲日志记录，请运行以下命令：

```
ontape -s -B stores_demo
```

使用 ontape 向名为 stores_demo 的数据库添加未缓冲日志记录，请运行以下命令：

```
ontape -s -U stores_demo
```

除了打开事务日志记录外，这些命令还创建整个系统存储空间备份。当 ontape 提示您提供备份级别时，请指定 0 级备份。

> 提示： 使用 ontape，您必须对所有存储空间执行 0 级备份。

使用 ontape 结束日志记录

要使用 ontape 结束名为 stores_demo 的数据库的日志记录，请运行以下命令：

```
ontape -N stores_demo
```

使用 ontape 更改缓冲模式

要使用 ontape 将名为 stores_demo 的数据库上的缓冲模式从已缓冲日志记录更改为未缓冲日志记录，但不创建存储空间备份，请运行以下命令：

```
ontape -U stores_demo
```

要使用 ontape 将名为 stores_demo 的数据库上的缓冲模式从未缓冲日志记录更改为已缓冲日志记录，但不创建存储空间备份，请运行以下命令：

```
ontape -B stores_demo
```

使用 ontape 使数据库符合 ANSI 标准

要使用 ontape 使已使用事务日志纪录（未缓冲或已缓冲）的名为 stores_demo 的数据库成为符合 ANSI 标准的数据库，请运行以下命令：

```
ontape -A stores_demo
```

要使用 ontape 使尚未使用事务日志记录的名为 stores_demo 的数据库成为符合 ANSI 标准的数据库，请运行以下命令：

```
ontape -s -A stores_demo
```

除了使数据库符合 ANSI 标准外，该命令同时还创建存储空间备份。当提示您指定级别时，请指定 0 级备份。

> 提示： 将日志记录模式更改为符合 ANSI 标准后，就无法轻易再对其进行更改。要更改符合 ANSI 标准的数据库的日志记录模式，请卸载数据，使用新的日志记录模式重建数据库并重新载入数据。详细信息，请参阅更改符合 *ANSI* 标准的数据库的日志记录模式 在第241页。

修改表日志记录模式

在缺省情况下，数据库服务器创建使用日志记录的标准表。要创建无日志记录表，请使用带有 WITH LOG 子句的 CREATE TABLE 语句。有关 CREATE TABLE 和 ALTER TABLE 语句的信息，请参阅《*SinoDB® SQL* 指南*:* 语法》。有关更多信息，请参阅*SinoDB* 的表类型 在第156页。

更改表以关闭日志记录

要将表从日志记录切换为无日志记录，请使用 TYPE 选项为 RAW 的 SQL 语句 ALTER TABLE。例如，以下语句将 tablog 表更改为 RAW 表：

```
ALTER TABLE tablog TYPE (RAW)
```

更改表以打开日志记录

要将表从无日志记录切换为日志记录，请使用 TYPE 选项为 STANDARD 的 SQL 语句 ALTER TABLE。例如，以下语句将 tabnolog 表更改为 STANDARD 表：

```
ALTER TABLE tabnolog TYPE (STANDARD)
```

> 重要： 在将表更改为 STANDARD 时也打开了表的日志记录。更改表之后，如果必须能后复原该表，请执行 0 级备份。

禁用对临时表的日志记录

可以禁用对临时表的日志记录以提高性能，并防止 SinoDB® 在数据复制环境（含有 HDR 辅助服务器、RS 辅助服务器 和 SD 辅助服务器）中使用主服务器时传输临时表。

要禁用对临时表的日志记录，请将 TEMPTAB_NOLOG 配置参数设置为 1。

对于高可用性集群中的 HDR、RS 和 SD 辅助服务器，必须通过将 TEMPTAB_NOLOG 配置参数设置为 1 以始终禁用对临时表的逻辑日志记录。

可以使用 onmode -wf 命令来更改 TEMPTAB_NOLOG 的值。

监视事务

This topic contains references for information about ways to monitor transactions.

该主题包含有关监视事务方法的信息的参考。

命令	描述	参考
onstat -x	监视事务	监视全局事务 在第431页
onstat -g sql	监视按会话标识和数据库列出的 SQL 语句	《*SinoDB*® 性能指南》中的性能监视
onstat -g spf	显示有关 SQL 查询的详细信息。	
onstat -g stm	监视预编译 SQL 语句的内存使用情况。	《*SinoDB*® 性能指南》中的内存利用率

监视数据库的日志记录模式

这些主题说明了监视数据库和表的日志记录模式的方法。

使用 SMI 表监视日志记录模式

可查询 sysmaster 数据库中的 sysdatabases 表以确定日志记录模式。该表中的一行对应于数据库服务器管理的一个数据库。flags 字段指示数据库的日志记录模式。is_logging、is_buff_log 和 is_ansi 字段指示是否日志记录为活动的，以及是使用已缓冲日志记录还是符合 ANSI 标准的日志记录。有关该表的列描述，请参阅《*SinoDB*® 管理员参考》中有关 sysmaster 数据库章节中的 sysdatabases 部分。

逻辑日志

日志记录 在第232页中的信息和以下主题说明了数据库服务器如何使用逻辑日志。有关如何执行逻辑日志任务的信息，请参阅管理逻辑日志文件 在第250页和管理数据库日志记录模式 在第239页。

相关链接

数据库服务器维护任务 在第30页

什么是逻辑日志？

为了保留自上次存储空间备份以来的事务和数据库服务器更改的历史记录，数据库服务器生成日志记录。数据库服务器将日志记录存储在逻辑日志中，这是由三个或更多逻辑日志文件组成的循环文件。将该日志称为逻辑的是因为日志记录代表数据库服务器的逻辑操作（而不是物理操作）。在任何时候，存储空间备份加上逻辑日志备份的组合包含了数据库服务器数据的完整副本。

作为数据库服务器管理员，必须配置和管理逻辑日志。例如，如果没有定期备份日志文件，那么逻辑日志填满了，数据库服务器将会暂挂处理。

这些职责包含以下任务：

- 为逻辑日志选择适当的位置

  请参阅 逻辑日志文件的位置 在第244页。
- 监视逻辑日志文件状态

  请参阅 逻辑日志文件的标识 在第244页。
- 为逻辑日志分配适当的磁盘空间量

  请参阅 逻辑日志文件的大小 在第245页。
- 按需分配附加日志文件

  请参阅 分配逻辑日志文件 在第256页。
- 将逻辑日志文件备份到介质上

  请参阅 备份逻辑日志文件 在第252页和释放逻辑日志文件 在第247页。
- 管理 BLOB 空间和智能大对象空间的日志记录

  请参阅 记录 *BLOB* 空间和简单大对象 在第247页和记录智能大对象空间和智能大对象 在第248页。

逻辑日志文件的位置

数据库服务器初始化磁盘空间时，将逻辑日志文件和物理日志放置在根数据库空间中。

要通过减少对根数据库空间的写入次数和最小化争用来提高性能，可将逻辑日志文件从根数据库空间移到磁盘上未被活动表或物理日志共享的数据库空间中。

进一步提高性能，可将逻辑日志文件分成两组，并将其存储在两个不同的磁盘上（两个磁盘均不包含数据）。例如，如果有 6 个逻辑日志文件，那么可能可将文件 1、3 和 5 放在磁盘 1 上，文件 2、4 和 6 放在磁盘 2 上。这种安排可提高性能，因为不会需要同一磁盘驱动器同时处理写入当前逻辑日志文件和备份。

逻辑日志文件包含关键信息，因此必须制作镜像以获取最大数据保护。如果将逻辑日志文件移到其他数据库空间中，请计划在该数据库空间上启动镜像。

相关链接

移动逻辑日志文件 在第261页
移动逻辑日志文件 在第261页

逻辑日志文件的标识

每个逻辑日志文件（不论是否备份到介质）都有一个唯一标识号。以初始化数据库服务器磁盘空间后所填充的第一个逻辑日志文件为 1 而开始该序列。当前逻辑日志文件已满时，数据库服务器切换到下一个逻辑日志文件，并将新日志文件的唯一标识号增加 1。新添加的或标记为删除的日志文件的唯一标识号为 0。

分配给每个逻辑日志文件的实际磁盘空间有一称为日志文件号 的标识号。例如，如果配置 6 个逻辑日志文件，那么这些文件具有从 1 到 6 的日志号。日志号可能是不按顺序的。在备份并释放逻辑日志文件时，数据库服务器会将磁盘空间重新用于逻辑日志文件。

下表说明了日志号和唯一标识号之间的关系。在日志 7 在日志 5 之后插入，并在第二次循环交替中第一次使用。

表 35: 逻辑日志文件的编号顺序

日志文件号	第一次循环交替唯一标识号	第二次循环交替唯一标识号	第三次循环交替唯一标识号
1	1	7	14
2	2	8	15
3	3	9	16
4	4	10	17
5	5	11	18
7	0	12	19
6	6	13	20

逻辑日志文件的状态标志

所有逻辑日志文件在第一位置均具有以下状态标志之一：已添加 (A)、已删除 (D)、可用 (F) 或已使用 (U)。下表显示了可能的日志状态标志组合。

表 36: 逻辑日志状态标志

状态标志	逻辑日志文件状态
A------	日志文件已添加且可用，但尚未使用。
D------	如果删除具有状态 U-B 的日志文件，那么它会标记为已删除。该日志文件被删除，其空间得以释放，可在您为所有存储空间进行 0 级备份时重新使用。
F------	日志文件已释放且可供使用。 逻辑日志文件在备份后得以释放，逻辑日志文件中的所有事务均关闭，存储在该文件中的最旧更新会清空到磁盘。
U	日志文件已使用但尚未备份。
U-B----	日志文件已备份但仍需用于恢复。（当不再需要该日志文件用于恢复时，将释放该文件。）
U-B---L	日志已备份但仍需用于恢复。包含上一个检查点记录。
U---C	数据库服务器当前正在填充该日志文件。
U---C-L	当前日志文件包含上一个检查点记录。

使用 onstat -l 命令按号码列出日志文件，并监视已使用日志空间的状态标识和百分比。有关更多信息，请参阅 *onstat -l* 命令 在第255页。

逻辑日志文件的大小

逻辑日志文件的最小大小为 200 KB。

逻辑日志文件的最大大小为 524288 页（等于 0x7ffff + 1），其基页大小为 2 KB 或 4 KB，具体取决于操作系统。要确定操作系统上数据库服务器的基页大小，请执行 onstat -d，然后检查根数据库空间的 pgsize 值。

决定要使用的日志文件的大小和数量。如果分配了多于所需的磁盘空间，那么会浪费空间。然而，如果未分配足够的磁盘空间，那么性能可能会受到不利影响。许多用户同时写入日志时，请使用较大的日志文件。

注: 较小的日志文件意味着如果包含该日志文件的磁盘脱机，可稍后进行恢复。如果设置了连续日志备份，那么日志文件会在填满时将自动备份。较小的日志导致逻辑恢复时间稍长。

逻辑日志文件数量

在考量逻辑日志文件的数目时，请考虑以下要点：

- 必须始终有至少 3 个逻辑日志文件，最多为 32,767 个日志文件。
- 日志文件的数目影响逻辑日志备份的频率。
- 逻辑日志文件的数目影响 BLOB 空间 BLOB 页可回收的速率。请参阅备份日志文件以释放 *BLOB* 页 在第248页。

性能注意事项

对于给定的系统活动级别，分配的逻辑日志磁盘空间越少，逻辑日志空间越快填满，且用户活动就越有可能因备份和检查点而阻塞。调整逻辑日志大小以找出您系统的最佳值。

- 逻辑日志备份

  逻辑日志文件填满时，必须对其进行备份。备份过程可能会阻碍涉及与逻辑日志文件位于同一磁盘上的数据的事务处理。将物理日志、逻辑日志和用户数据放在不同的磁盘上。（请参阅《*SinoDB*® 备份和还原指南》。）
- 逻辑日志的大小

  较小的逻辑日志比较大的逻辑日志更快填满。可以按照手动添加逻辑日志文件 在第259页中的說明添加较大的逻辑日志文件。
- 个别逻辑日志记录的大小

  逻辑日志记录的大小根据处理操作和数据库服务器环境而变化。通常，数据行越长，逻辑日志记录就越大。逻辑日志包含已插入、更新或已删除的行的映像。更新可使用插入和删除所用空间两倍的空间，因为更新可能包含前映像和后映像。（插入仅存储后映像，而删除仅存储前映像。）
- 逻辑日志记录的数量

  向逻辑日志写入的逻辑日志记录越多，逻辑日志就越快填满。具有事务日志记录的数据库比不具有事务日志记录的数据库的事务更快地填充逻辑日志。
- 日志缓冲的类型

  使用未缓冲事务日志记录的数据库比使用已缓冲事务日志记录的数据库更快地填充逻辑日志。
- 对表进行的 Enterprise Replication

  因为 Enterprise Replication 会生成复制表的前映像和后映像，因而可能使逻辑日志填满。
- 回滚频率

  回滚越多，逻辑日志越快填满。尽管回滚记录很小，但回滚本身也需要逻辑日志文件空间。
- 智能大对象的数目

  已启用用户数据日志记录且具有大量用户数据更新的智能大对象会以极快速率填充逻辑日志。如果不想记录元数据，可使用临时智能大对象。

动态日志分配

动态日志分配防止在长事务回滚期间日志文件填满并挂起系统。仅当下一个日志文件包含打开的事务时，该功能才处于活动状态。（如果 事务 在到达长事务高水位标志时未提交或回滚，则该事务为 长事务 。）

当下一个日志文件包含打开的事务时，数据库服务器会在当前日志文件之后自动（动态）分配日志文件。对于以下操作可使用动态日志分配：

- 当系统活动时添加日志文件
- 在当前日志文件后插入日志文件
- 立即访问新日志文件（即使根数据库空间未备份）

测试动态日志分配的最佳方法是产生跨所有日志文件的事务，然后使用 onstat -l 检查新添加的日志文件。有关更多信息，请参阅分配逻辑日志文件 在第256页。

重要： 仍必须备份日志文件以防止其填满。如果日志文件填满，那么系统将挂起直到您执行备份。

释放逻辑日志文件

每次数据库服务器提交或回滚事务时，会尝试释放事务开始所在的逻辑日志文件。在数据库服务器释放逻辑日志文件以便重新使用之前，必须满足以下条件：

- 备份日志文件。
- 逻辑日志文件中没有记录与打开的事务相关联。
- 逻辑日志文件不包含尚未清空到磁盘的最旧更新。

下一个逻辑日志文件未释放时要执行的操作

当数据库服务器尝试切换到下一个逻辑日志文件时发现顺次的下一个日志文件仍在使用，那么数据库服务器立即暂挂所有处理。即使其他逻辑日志文件为可用的，数据库服务器不会跳过使用中的文件而写入非顺次的可用文件。处理停止以保护逻辑日志文件中的数据。

逻辑日志文件可能由于以下任一原因而处于使用中：

- 文件包含尚未清空到磁盘的最近检查点或最旧更新。

  发出 onmode -c 命令以执行检查点并释放逻辑日志文件。有关更多信息，请参阅强制执行检查点 在第272页。
- 文件包含打开的事务。

  打开的事务为控制长事务 在第262页中所说明的长事务。
- 文件未备份。

  如果逻辑日志文件未备份，那么当使用 ON-Bar 或 ontape 来备份逻辑日志文件时，处理会恢复进行。

下一个日志文件包含上一个检查点时要执行的操作

在下一个日志文件包含上一个检查点或最旧更新时，数据库服务器不暂挂处理。如果尚未清空到磁盘的上一个检查点记录或最旧更新位于紧随最后可用日志后的日志中，那么数据库服务器切换到最后可用日志时，将始终强制执行检查点。例如，如果 4 个逻辑日志文件具有以下列表中显示的状态，那么数据库服务器在切换到逻辑日志文件 3 时强制执行检查点。

1

U-B----

2

U---C--

3

F

4

U-B---L

记录 BLOB 空间和简单大对象

简单大对象数据（TEXT 和 BYTE 数据类型）可能太庞大，无法包含在逻辑日志记录中。如果始终记录简单大对象，那么它们可能会很大以致减缓逻辑日志。

数据库服务器假设您将数据库设计为较小简单大对象存储在数据库空间中，而较大简单大对象存储在 BLOB 空间中：

- 对于存储在数据库空间中的简单大对象，数据库服务器将简单大对象数据包含在日志记录中。

- 对于存储在 BLOB 空间中的简单大对象，数据库服务器不会将简单大对象数据包含在日志记录中。仅当备份逻辑日志时，逻辑日志才记录 BLOB 空间数据。

为了那些对 BLOB 空间中的简单大对象执行频繁更新的应用程序能获得更好的整体性能，可减少逻辑日志的大小。较小的日志可改进对必须重新使用的简单大对象的访问。有关更多信息，请参阅《*SinoDB*® 性能指南》中有关配置对 I/O 利用率的影响的章节。

切换日志文件以激活 BLOB 空间

在以下情况下，必须切换到下一个逻辑日志文件：

- 在创建 BLOB 空间后，如果想要立即将简单大对象插入 BLOB 空间
- 向现有的 BLOB 空间添加新块后，如果想要将简单大对象插入到使用该新块的 BLOB 空间

数据库服务器要求在不同逻辑日志文件中创建用于創建 BLOB 空间的语句、在 BLOB 空间中创建块的语句，以及將簡單大對象插入 BLOB 空间的语句。此要求与数据库日志状态无关。

有关切换到下一个逻辑日志文件的指示信息，请参阅切换到下一个逻辑日志文件 在第253页。

备份日志文件以释放 BLOB 页

删除存储在 BLOB 空间页中的数据时，不必为了重新使用而释放这些页。仅当以下两个操作均发生时，BLOB 空间页是可用的：

- 通过对列的 UPDATE 操作或通过删除行将 TEXT 或 BYTE 数据删除
- 备份存储了行（包含 TEXT 或 BYTE 数据）的 INSERT 操作的逻辑日志

插入或删除 TEXT 和 BYTE 数据之后备份 BLOB 空间

请确保备份所有的 BLOB 空间和包含 BLOB 空间中简单大对象上的事务的逻辑日志。在日志备份期间，数据库服务器在逻辑日志中使用数据指针将更改过的 text 和 byte 数据从 BLOB 空间复制到逻辑日志中。

记录智能大对象空间和智能大对象

智能大对象空间 在第146页中所描述的智能大对象空间包含两个组件：元数据和用户数据。在缺省情况下，不记录智能大对象空间。

智能大对象空间的元数据组件描述存储在特定智能大对象空间中的智能大对象的关键特征。元数据包含指向智能大对象的指针。如果元数据损坏或变得不可访问，那么智能大对象空间也将损坏，并且智能大对象空间中的智能大对象将无法恢复。

即使关闭数据库的日志记录，标准智能大对象空间中的元数据始终得以记录。记录智能大对象空间元数据确保元数据始终可以恢复到一致的事务状态。但是，不记录临时智能大对象空间中的元数据。

智能大对象空间日志记录

当记录智能大对象空间时，数据库服务器会变慢，逻辑日志会迅速填满。如果对智能大对象空间使用日志记录，那么必须确保逻辑日志大到足以容纳日志记录数据。有关更多信息，请参阅估计记录智能大对象时的日志大小 在第252页。

在打开数据库的日志记录后，数据库服务器到等到您执行 0 级备份后才会开始日志记录。然而，在打开智能大对象的日志记录时，数据库服务器会立即开始记录更改。为了减少日志条目的数量，载入智能大对象时关闭日志记录，之后再打开以捕获智能大对象的更新。

重要： 打开智能大对象的日志记录时，必须立即执行 0 级备份以便能够恢复和复原智能大对象。

有关更多信息，请参阅 备份智能大对象空间 在第253页和《*SinoDB*® 备份和还原指南》.。

智能大对象的日志记录

如果用户频繁更新数据或恢复任何已更新数据的能力至为重要，可为智能大对象使用日志记录。数据库服务器将操作（插入、 更新、 删除、 读取或写入）记录写入逻辑日志缓冲区。CLOB 或 BLOB 数据的已修改部分包含在日志记录中。

要提高性能，可关闭智能大对象的日志记录。如果用户主要在分析数据且不常更新数据，或如果数据对于恢复并不重要，也可以关闭日志记录。

已更新的智能大对象的日志记录

更新智能大对象时，数据库服务器不会记录整个对象。假设在为智能大对象启用日志记录的状态下，用户在偏移量 Y 写入 X 字节的数据。数据库服务器会记录以下信息：

- 如果 Y 设置到大对象的末尾，那么数据库服务器记录 X 字节 （更新的字节范围）。
- 如果 Y 是在大对象的开头或中间，那么数据库服务器记录以下选项中最小的：
  - 新旧映像之间的差异
  - 前映像和后映像
  - 如前映像和后映像相同，则不记录任何值

开启或关闭智能大对象空间的日志记录

可以使用几种不同的方法来开启或关闭智能大对象空间的日志记录。

如果想要在智能大对象空间中使用日志记录，请在创建智能大对象空间时需指定 onspaces 命令的 -Df "LOGGING=ON" 选项。如果在智能大对象空间中关闭日志记录，您可以在特定列中对智能大对象打开日志记录。可开启包含智能大对象的列的日志记录，但关闭其他列的日志记录。

要验证智能大对象空间中的智能大对象是否已记录，请使用 oncheck -pS *sbspace_name* | grep "Create Flags" 命令。

如果在具有缺省日志记录选项的智能大对象空间中创建智能大对象且在输出看到 LO_NOLOG 标志，表示不记录该智能大对象空间中的智能大对象。如果在输出看到 LO_LOG 标志，表示记录该智能大对象空间中的所有智能大对象。

可以按以下任一方式来修改智能大对象空间的日志记录状态。

指定的函数或语句	日志记录操作	参考
onspaces -ch -Df "LOGGING=ON" onspaces -ch -Df "LOGGING=OFF"	打开或关闭现有智能大对象空间的日志记录	更改智能大对象的存储特征 在第184页 *onspaces -ch:*更改智能大对象空间缺省规范
使用 set sbspace logging on 或 set sbspace logging off 参数的 SQL 管理 API task() 或 admin() 函数	打开或关闭现有智能大对象空间的日志记录	设置智能大对象空间日志记录参数#更改智能大对象空间的日志记录 *(SQL 管理 API)*
CREATE TABLE 或 alter table 语句的 PUT 子句中的 LOG 选项	将载入列中的所有的智能大对象的日志记录打开	日志记录 在第150页 *PUT 子句*
mi_lo_create DataBlade® API 函数	在最初载入智能大对象时关闭其日志记录	《*SinoDB® DataBlade® API* 函数参考》
mi_lo_alter DataBlade® API 函数	完成载入后打开日志记录	《*SinoDB® DataBlade® API* 函数参考》
ifx_lo_create SinoDB® ESQL/C 函数	在最初载入智能大对象时关闭其日志记录	《*SinoDB® ESQL/C* 程序员指南》

指定的函数或语句	日志记录操作	参考
ifx_lo_alter SinoDB® ESQL/C 函数	在完成载入后打开日志记录	《*SinoDB® ESQL/C* 程序员指南》

智能大对象日志记录

当用 LOG 选项创建智能大对象时，逻辑日志将创建智能 *BLOB* 日志记录。智能 BLOB 日志记录跟踪对用户数据或元数据的更改。更新智能大对象时，日志记录中仅包含智能大对象页的修改部分。仅当智能大对象启用日志记录时，才会在逻辑日志中创建用户数据日志记录。

警告： 对于频繁更新的智能大对象，请谨慎开启其日志记录。此日志记录开销可能会显著减慢数据库服务器的速度。

有关智能大对象的日志记录的信息，请参阅《*SinoDB®* 管理员参考》中有关解释逻辑日志记录的章节。

在记录智能大对象数据时防止长事务

可在单个智能大对象的数据收集过程持续较长时间的情况下使用智能大对象。例如，记录了许多个小时低质量音频信息的应用程序。虽然收集的数据量可能不大，但记录会话可能很长，导致长事务状况。

提示： 要防止长事务的发生，请定期提交对智能大对象的写入操作。

日志记录过程

这些主题详细描述了数据库空间、 BLOB 空间和智能大对象空间的日志记录过程。这些信息对于执行常规的数据库服务器管理任务不是必需的。

数据库空间日志记录

数据库服务器对于涉及数据库空间中所存储数据的操作使用以下日志记录过程：

1. 将数据页从磁盘读到共享内存页缓冲区
2. 如需要，将未更改页复制到物理日志缓冲区
3. 如需要，将新数据写入页缓冲区并创建事务的逻辑日志记录
4. 将物理日志缓冲区清空到磁盘上的物理日志
5. 将逻辑日志缓冲区清空到磁盘上的逻辑日志文件
6. 清空页缓冲区并将其写回到磁盘上

BLOB 空间日志记录

数据库服务器记录 BLOB 空间数据，但数据不会经过共享内存或磁盘上的逻辑日志文件。数据库服务器直接将BLOB 空间所存储的数据从磁盘复制到磁带上。BLOB 空间开销页（自由图页和 位图页）的修改记录是到达逻辑日志的唯一 BLOB 空间数据。

管理逻辑日志文件

即使您的数据库不使用事务日志记录，还是必须管理逻辑日志文件。有关逻辑日志的背景信息，请参阅逻辑日志 在第243页。

在 UNIX™ 上，您必须以 informix 或 root 身份登录才能进行本章中所述的任何更改。在 Windows™ 上，您必须是 Informix-Admin 组的成员。

设置逻辑日志时要执行以下任务：

- 在初始化或重新启动数据库服务器之前，请使用 LOGFILES 参数指定要创建的逻辑日志文件的数量。
- 在数据库服务器联机后，请估算系统所需逻辑日志文件的大小和数量。

请参阅估计日志文件的大小和数量 在第251页。
- 如果不想使用缺省值，请更改 LOGSIZE 和 LOGBUFF 配置参数。
- 添加估算的逻辑日志文件数。

  请参阅分配逻辑日志文件 在第256页。

请例行执行以下任务 ：

- 备份逻辑日志文件
- 切换到下一个逻辑日志文件
- 释放逻辑日志文件
- 监视日志记录活动和日志备份状态

请视需要执行以下任务：

- 添加逻辑日志文件
- 删除逻辑日志文件
- 更改逻辑日志文件的大小
- 移动逻辑日志文件
- 更改逻辑日志配置参数
- 监视逻辑日志的事件警报
- 设置事务的高水位标志

有关使用 SQL 管理 API 命令（而不是一些 oncheck、onmode、onparams 和 onspaces 命令）的信息，请参阅使用 *SQL* 管理 *API* 执行远程管理 在第458页和 《*SinoDB*® 管理员参考》。

估计日志文件的大小和数量

使用 LOGSIZE 配置参数设置逻辑日志文件的大小。

对于您的数据库服务器系统的最佳的日志空间量取决于以下因素：

- 应用程序需求和应用程序的更新活动量。更新活动量增加需要增加日志空间。
- 恢复时间目标（RTO）为给予服务器在重新启动之后从问题中恢复并进入联机或静默模式的时间量（以秒计）。

  在灾难性事件的情况中，要考虑您可以容忍多少数据丢失。更频繁的日志备份（为减少数据和事务丢失的风险）需要更大的日志空间。
- 是使用 Enterprise Replication 还是使用数据复制配置（例如：HDR 辅助服务器、 SD 辅助服务器 或 RS 辅助服务器）。

  这些复制服务影响日志文件的数量和大小。如果您的系统使用其中任何一个复制服务，请参阅高可用性集群配置 在第294页或《*SinoDB*® *Enterprise Replication* 指南》中的指南。

以下为决定日志大小的指南：

- 通常，管理较少的大日志文件比管理大量的小日志文件要容易得多。
- 日志空间过多不会影响性能。但是，日志文件和日志空间不足却会影响性能，因为数据库服务器将频繁触发检查点。
- BLOB 空间中的智能大对象不会被记录，但这些对象会包含在创建对象的日志备份中。这意味着直到服务器备份创建对象的日志才会释放对象。因此，如果 BLOB 空间中的智能大对象频繁更新，那么可能需要更频繁地进行日志备份以在 BLOB 空间内获得更多可用空间。
- 对于生成少量日志数据的应用程序，可从使用 10 个日志文件，每个文件 10 MB 开始。
- 对于生成大量日志数据的应用程序，可从使用 10 个日志文件，每个文件 100 MB 开始。

以下两种维护 RTO 策略的方式决定了在灾难性事件（例如：数据服务器丢失）中所容许的数据丢失：

- 维护 RTO 策略的一种方法是使用自动日志备份，即一旦日志文件填满就触发日志备份。这样就将数据丢失限制到在备份期间包含在日志文件中的事务以及日志备份期间发生的任何其他事务。

- 维护 RTO 策略的另一种方法是使用调度程序。您可以创建一个任务，自上次日志备份以来，以固定时间间隔自动备份任何新的日志数据。这样就将数据丢失限制到在时间间隔之间尚未备份的事务。有关使用调度程序的信息，请参阅调度程序 在第441页。

如果 RTO 策略是必需的，那么可以使用调度程序插入一个按适当频率执行的任务来维护该策略。这样就会在日常周期内的特定时间自动备份日志文件。如果日志空间在日志备份和回收之前就填满了，那么可以备份这些日志并添加一个新的日志文件以允许事务处理继续进行，或者可以使用调度程序添加新的任务来检测此情况并自动执行其中任一操作。

您可以随时添加日志文件，并且数据库服务器会在事务一致性（例如，可能占用大量日志空间的长事务）需要时自动添加日志文件。

增加逻辑日志空间量的最简单的方法是添加另一个逻辑日志文件。请参阅手动添加逻辑日志文件 在第259页。

以下表达式提供总日志空间配置示例（以 KB 计）：

```
LOGSIZE = (((connections * maxrows) * rowsize) / 1024) / LOGFILES
```

表达式元素	解释
LOGSIZE	指定每个逻辑日志文件的大小（以 KB 计）
connections	为在 sqlhosts 文件或注册表以及 NETTYPE 参数中指定的所有网络类型指定最大连接数。如果通过在配置文件中设置多个 NETTYPE 配置参数来配置多个连接，那么请为每个 NETTYPE 添加用户字段，并以此总数替代前公式中的 *connections*。
maxrows	指定在单个事务中要更新的最大行数。
rowsize	指定表行的平均大小（以字节计）。要计算 *rowsize*，请将行中各列的长度（来自 syscolumns 系统目录表）相加。
1024	将 LOGSIZE 转换为指定的 KB 单位。
LOGFILES	指定逻辑日志文件的数量。

估计记录智能大对象时的日志大小

如果打算记录智能大对象用户数据，那必须确保日志大小比写入的数据量大得多。如果将智能大对象存储在标准智能大对象空间中，那么即使不记录智能大对象也始终记录元数据。如果将智能大对象存储在临时智能大对象空间中，那么将不会有任何日志记录。

估计逻辑日志文件的数量

LOGFILES 参数提供系统初始化或重新启动时逻辑日志文件的数量。如果所有逻辑日志文件的大小相同，那么可如下所示计算分配给逻辑日志文件的总空间量：

```
总逻辑日志空间量 = LOGFILES * LOGSIZE
```

如果数据库服务器包含不同大小的日志文件，那么就不能使用 (LOGFILES * LOGSIZE) 表达式来计算逻辑日志的大小。而是必须将磁盘上每个日志文件的大小相加。检查 onstat -l 输出中的 size 字段。有关更多信息，请参阅*onstat -l* 命令 在第255页。

有关 LOGSIZE、LOGFILES 与 NETTYPE 的信息，请参阅《*SinoDB*® 管理员参考》 中有关配置参数的主题。

备份逻辑日志文件

逻辑日志包含已执行事务的历史记录。将逻辑日志文件复制到介质的过程称为备份逻辑日志文件。备份逻辑日志文件实现以下两个目标：

- 将逻辑日志记录储存在介质上，以便需要数据复原时可前滚这些记录。
- 将逻辑日志文件空间标记为可用于新的逻辑日志记录。

如果忽略了备份日志文件，那么会耗尽日志空间。

您可以启动手动逻辑日志备份，或设置连续逻辑日志备份。在复原存储空间之后，必须复原逻辑日志以使数据处于一致状态。有关日志备份的更多信息，请参阅《*SinoDB*® 备份和还原指南》。

备份 BLOB 空间

先备份逻辑日志还是先备份 BLOB 空间是无关紧要的。

备份 BLOB 空间数据：

1. 如果当前逻辑日志包含有关 BLOB 空间中简单大对象的事务时，请关闭该日志。
2. 在更新简单大对象数据后，尽快对逻辑日志及 BLOB 空间执行备份。

> 警告： 如果不备份这些 BLOB 空间和逻辑日志，那么可能无法复原 BLOB 空间数据。若等到 BLOB 空间不可用时才执行日志备份，那么数据库服务器将无法访问 BLOB 空间以将更改过的数据复制到逻辑日志中。

备份智能大对象空间

当打开智能大对象的日志记录时，必须执行该智能大对象空间的 0 级备份。

下图显示在未备份的智能大对象空间中打开日志记录时会发生的状况。尽管已记录的更改是可恢复，但在故障期间智能大对象 LO1 中未记录的更改将丢失。您将无法完全复原 LO1。

在快速恢复期间，数据库服务器将前滚 LO1 中所有已提交的事务。如果未记录 LO1，那么数据库服务器将无法回滚未提交的事务。因此，LO1 内容将是不正确的。有关更多信息，请参阅快速恢复 在第268页。

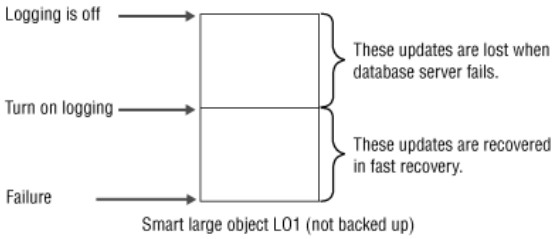

图 46: 打开智能大对象空间的日志记录

切换到下一个逻辑日志文件

由于以下原因，可能希望在当前日志文件填满之前切换到下一个逻辑日志文件：

- 备份当前日志
- 激活新的 BLOB 空间和 BLOB 空间块

数据库服务器可处于联机模式进行此更改。运行以下命令切换到下一个可用日志文件：onmode -l

更改立即生效。（确保在命令行中输入小写 L，而不是数字 1。）

释放逻辑日志文件

如果日志文件是新添加的（状态 A），那么可立即使用。也可立即删除。

可能由于以下原因想要释放逻辑日志文件：

- 使数据库服务器不停止处理
- 释放已删除 BLOB 页所用的空间

释放日志文件的过程根据日志文件的状态而变化。在以下主题将描述每一个过程。要查明逻辑日志文件状态，请参阅逻辑日志文件的状态标志 在第245页和监视日志记录活动 在第255页。

提示： 有关使用 ON-Bar 或 ontape 来备份存储空间和逻辑日志的信息，请参阅《*SinoDB®* 备份和还原指南》。

删除状态为 D 的日志文件

当删除已使用的日志文件时，该文件将标记为已删除（状态为 D）并不能再使用，同时 onparams 将打印以下信息：

```
Log file log_file_number has been pre-dropped. It will be
deleted from the log list and its space can be reused
once you take level 0 archives of all BLOBspaces,
Smart BLOBspaces and non-temporary DBspaces.
```

必需具有 0 级归档，以确保日志文件本身和不同数据库空间中的所有关联信息都已归档。在 0 级归档结束时将删除日志文件；但是，由于除去日志文件本身属于磁盘上根保留页结构中的更改，因此，下一次执行的归档也必须是 0 级归档。必须先执行行 0 级归档，才能执行 1 级或 2 级归档。

释放状态为 U 的日志文件

如果日志文件包含记录但尚未备份时（状态为 U），请使用通常使用的备份工具备份文件。

如果备份日志文件不会将状态更改为可用 (F)，则其状态将更改为 U-B 或 U-B-L。请参阅释放状态为 *U-B* 或 *F* 的日志文件 在第254页或释放状态为 *U-B-L* 的日志文件 在第255页。

释放状态为 U-B 或 F 的日志文件

如果日志文件已备份但仍在使用中（状态为 U-B），那么表示日志文件中的某些事将仍在进行，或日志文件包含快速恢复所需的最旧更新。由于过去已使用状态为 F 的日志文件，所以它将与状态为 U-B 的日志文件一样遵循相同的规则。

释放使用中的已备份日志文件：

1. 如果不想等到事务完成，请将数据库服务器置于静默模式。

   请参阅 更改数据库服务器运行模式 在第78页。将回滚任何活动事务。
2. 使用 onmode -c 命令以强制产生一个检查点。之所以这样做是因为状态为 U-B 的日志文件可能包含最旧的更新。

已备份但不在使用中（状态为 U-B）的日志文件无需释放。在以下示例中，日志 34 无需释放，但日志 35 和 36 需要释放。日志 35 包含上一个检查点，日志 36 是已备份但仍在使用中。

```
34 U-B--   Log is used, backed up, and not in use
35 U-B-L   Log is used, backed up, contains last checkpoint
36 U-B--   Log is used, backed up, and not in use
37 U-C--   This is the current log file, not backed up
```

提示： 仅当逻辑日志没有被活动事务跨越，也不包含最旧的更新，那么您才能释放状态为 U-B（并非 L）的逻辑日志。

释放状态为 U-C 或 U-C-L 的日志文件

请遵循以下步骤来释放当前日志文件。

释放当前日志文件（状态为 C）：

1. 运行以下命令将当前日志文件切换到下一个可用日志文件：onmode -l
2. 使用 ON-Bar 或 ontape 来备份原本的日志文件。
3. 在备份了所有填满的日志文件后，会提示您切换到下一个可用日志文件，并备份新的当前日志文件。

   如果刚刚切换到此日志文件，所以无需执行备份。

在释放当前日志文件之后，如果日志文件状态为 U-B 或 U-B-L，请参阅释放状态为 *U-B* 或 *F* 的日志文件 在第254页或释放状态为 *U-B-L* 的日志文件 在第255页。

释放状态为 U-B-L 的日志文件

如果日志文件已备份，并且其中所有事务都已关闭，但未释放该文件（状态为 U-B-L），那么该逻辑日志文件包含最近的检查点记录。您可以释放状态为 U-B-L 的日志文件。

要释放状态为 U-B-L 的日志文件，数据库服务器必须创建一个新的检查点。您可以运行以下命令以强制一个检查点：onmode -c

监视日志记录活动

监视逻辑日志文件以确定总可用空间（在所有文件中）、当前文件中的可用空间，以及文件状态（例如，日志是否已备份）。有关监视逻辑日志缓冲区的信息，请参阅监视物理和逻辑日志记录活动 在第270页。

监测逻辑日志的填充度

可以使用以下命令行实用程序来监视逻辑日志文件：

onstat -l 命令

onstat -l 命令显示有关物理及逻辑日志的信息。

包含每个逻辑日志文件信息的输出部分包括以下信息：

- 逻辑日志文件描述符的地址
- 日志文件号
- 指示每个日志状态（可用、已备份、当前等等）的状态标识
- 日志文件的唯一标识
- 文件的开始页
- 文件大小（以页计）、已用页数，及已用页的百分比

如果您删除列表中间的几个日志或数据库服务器动态添加一些日志文件，那么在 numbers 字段中的日志文件号将失序。

有关 onstat -l 输出的示例的更多信息，请参阅《*SinoDB*® 管理员参考》。

oncheck -pr 命令

数据库服务器将逻辑日志文件信息存储在专用于检查点信息的保留页中。因为数据库服务器仅在检查点期间更新此信息，所以该信息不如 onstat -l 选项所显示的信息新。有关使用这些选项来显示保留页信息的更多详细信息，请参阅《*SinoDB*® 管理员参考》。

可以使用 oncheck -pr 命令来查看检查点保留页。以下示例显示某个逻辑日志文件的样本输出。

```
...
Log file number             1
Unique identifier           7
Log contains last checkpoint Page 0, byte 272
Log file flags              0x3  Log file in use
                                 Current log file
Physical location           0x1004ef
Log size                    750 (p)
Number pages used           1
Date/Time file filled       01/29/2001 14:48:32
...
```

监视临时逻辑日志

因为在热复原期间永久日志不可用，所以数据库服务器使用 临时逻辑日志 前滚事务。当前滚完成，数据库服务器将释放临时日志文件。如果您在热复原期间发出 onstat -l， 那么输出包含临时日志文件（其格式与常规日志文件相同）的第四部分。临时日志文件仅使用 B、C、F 及 U 状态标志。

SMI 表

Query the syslogs table to obtain information about logical-log files. This table contains a row for each logical-log file. The columns are as follows.查询 syslogs 表以获取有关逻辑日志文件的信息。该表中一行对应一个逻辑日志文件。各列如下所示：

number
: 逻辑日志文件的标识号

uniqid
: 日志文件的唯一标识

size
: 文件大小（以页计）

used
: 已用页数

is_used
: 指示日志文件是否正使用的标志

is_current
: 指示日志文件是否为当前文件的标志

is_backed_up
: 指示日志文件是否已备份的标志

is_new
: 指示日志文件是否为自上次存储空间备份以来新添加的标志

is_archived
: 指示日志文件是否已写入归档磁带的标志

is_temp
: 指示日志文件是否标记为临时日志文件的标志

监视日志备份状态

要监视日志状态和查看哪些日志已备份，请使用 onstat -l 命令。状态标志 B 表示日志已备份。

分配逻辑日志文件

当初始化或重新启动数据库服务器时，它将创建在 LOGFILES 配置参数所指定数量的逻辑日志文件。逻辑日志文件大小由 LOGSIZE 配置参数所指定。

您可以手动添加逻辑日志文件或配置数据库服务器根据需要添加日志文件。当添加逻辑日志文件时，数据库服务器将动态更新 LOGFILES 配置参数的值。

以下配置参数也会影响逻辑日志文件。除非另有说明，否则可以在服务器运行时更新这些配置参数的值。

AUTO_LLOG
: 自动添加逻辑日志以提高性能，并限制逻辑日志文件总大小。

DYNAMIC_LOGS
: 自动添加逻辑日志以防止事务阻塞。

LOGBUFF

在共享内存中设置三个逻辑日志缓冲区的大小。更改 LOGBUFF 配置参数的值时，必须重新启动数据库服务器。

LTXEHWM

设置可用日志空间的百分比，在填满时触发数据库服务器以给予当前正在回滚的长事务对逻辑日志的互斥访问。

LTXHWM

设置可用日志空间的百分比，在填满时触发数据库服务器检查长事务。

相关链接

《SinoDB 管理员参考》: LOGFILES 配置参数
《SinoDB 管理员参考》: LOGSIZE 配置参数
《SinoDB 管理员参考》: AUTO_LLOG 配置参数
《SinoDB 管理员参考》: DYNAMIC_LOGS 配置参数
《SinoDB 管理员参考》: LOGBUFF 配置参数
《SinoDB 管理员参考》: LTXEHWM 配置参数
《SinoDB 管理员参考》: LTXHWM 配置参数

动态添加逻辑日志文件以防止事务阻塞

DYNAMIC_LOGS 配置参数决定数据库服务器何时动态添加逻辑日志文件以防止事务阻塞。

当使用 DYNAMIC_LOGS 的缺省值 2 时，如果下一个活动日志文件包含最早打开事务的开头，那么数据库服务器会动态添加新的日志文件并发出警报。

数据库服务器将在以下时刻检查逻辑日志空间：

- 切换到新的日志文件后
- 在逻辑恢复的事务清除阶段开始时

如果 DYNAMIC_LOGS 参数设置为 1 并且下一个活动日志文件包含来自打开事务的记录，那么数据库服务器将提示您手动添加日志文件并发出警报。在添加日志文件之后，数据库服务器将重新开始处理事务。

如果 DYNAMIC_LOGS 参数设置为 0 并且在长事务回滚期间逻辑日志耗尽了空间，那么数据库服务器将挂起。（长事务防止第一个逻辑日志文件变为可用并防止其可用于重新使用。）要解决该问题并完成长事务，请将 DYNAMIC_LOGS 设置为 2 并且重新启动数据库服务器。

相关链接

监视动态添加日志的事件 在第258页
控制长事务 在第262页
《SinoDB 管理员参考》: DYNAMIC_LOGS 配置参数

动态添加的日志文件的大小和数量
用 DYNAMIC_LOGS 配置参数启用动态日志的目的是为了添加足够的日志空间以允许回滚事务。

动态添加日志文件时，数据库服务器使用以下因素来计算日志文件的大小：

- 日志平均大小
- 可用连续空间量

如果逻辑日志的空间不足，数据库服务器将根据需要添加多个日志文件以允许回滚事务。日志文件的数量受限于：

- 支持的最大日志文件数
- 日志文件的磁盘空间量
- 根块中可用的连续空间量

如果数据库服务器因为磁盘空间耗尽而停止添加新的日志文件，它会写入错误消息并发出警报。向现有数据库空间添加数据库空间或块。然后，数据库服务器会自动重新开始处理事务。

在根块的保留页存储有关每个日志文件的信息。随着添加更多的日志文件，包含该信息的扩展数据快将会扩展。根块需要两个 1.4 MB 的扩展数据块 ，各用于追踪 32,767 个日志文件 （支持的最大数目）。

如果在恢复期间从非根块分配块保留页扩展数据块，那么服务器会尝试将其放回到根块中。如果根块中的空间不足，那么恢复将失败。包含所需空间的消息会显示在联机日志中。再次尝试复原之前，必须从根块中释放所需的空间。

动态添加的逻辑日志文件的位置
如果 DYNAMIC_LOGS 配置参数设置为 2，动态添加的日志文件的缺省位置是包含最新日志文件的数据空间。

数据库服务器按照以下搜索顺序在数据库空间分配日志文件。如果数据库空间包含逻辑日志文件或物理日志，那该数据库空间变得至关重要。

1
包含最新日志文件的数据库空间
（如果该数据库空间已满，那么数据库服务器会搜索其他数据库空间。）

2
包含日志文件的镜像数据库空间（但根数据库空间除外）

3
已包含日志文件的所有数据库空间（但根数据库空间除外）

4
包含物理日志的数据库空间

5
根数据库空间

6
任何镜像数据库空间

7
任何数据库空间

如果不想使用这个搜索顺序来分配新的日志文件，那么必须把 DYNAMIC_LOGS 参数设置为 1，并使用要用于新日志的位置运行 onparams -a -i 。有关详细信息，请参阅监视动态添加日志的事件 在第258页。

监视动态添加日志的事件
可监视数据库服务器动态添加逻辑日志文件时触发的事件警报以防止事务阻塞。DYNAMIC_LOGS 配置参数值必须是 1 或 2。

当每个警报触发时，就会向消息日志写入一条消息。

可以在事件类标识 27（需要日志文件）的警报脚本中包括 onparams 命令以添加日志文件。也可在脚本中运行 onstat -d 命令来检查是否有足够空间，并对具有足够空间的位置运行 onparams a -i 命令。您必须使用 -i 选项在当前日志文件后添加新的日志。

表 37: 动态添加日志文件的事件警报

类标识	严重性	类消息	消息
26	3	动态添加日志文件 *log_number*	当数据库服务器动态添加日志文件时显示该消息。 动态向数据库空间 *dbspace_number* 添加日志文件 *log_number*。
27	4	需要日志文件	当 DYNAMIC_LOGS 设为 1 且数据库服务器等待您添加日志文件显示该消息。 警报：最旧逻辑日志 *log_number* 包含来自打开的事务 *transaction_address* 的记录。逻辑日志将保持阻塞，直到添加了日志文件为止。如下所示，使用 onparams -a 命令和

类标识	严重性	类消息	消息
			-i（插入）选项添加日志文件：onparams -a -d *dbspace* -s *size*-i 然后尽快完成事务。
28	4	没有可用于日志文件的空间	警报：因为最旧的逻辑日志 *log_number* 包含来自打开的事务 *transaction_address* 的记录，所以服务器将试图动态添加日志文件。但是没有可用空间。添加数据库空间或块，然后尽快完成事务。

下表显示数据库服务器对于每个 DYNAMIC_LOGS 配置参数的设置所采取的动作。

表 38: DYNAMIC_LOGS 设置

DYNAMIC_LOGS 值	含义	事件警报	等待添加日志？	添加动态日志？
2（缺省）	允许自动分配新日志文件，以防止打开的事务挂起系统。	是（26、28）	否	是
1	允许手动添加新日志文件	是（27）	是	否
0	不分配日志文件，但发布以下有关打开的事务的消息： 警告： 最旧的逻辑日志文件 *log_number* 包含来自打开的事务 *transaction_address* 的记录 ， 但动态日志功能为关闭的。	否	否	否

相关链接

动态添加逻辑日志文件以防止事务阻塞 在第257页

《*SinoDB* 管理员参考》：事件警报

《*SinoDB* 管理员参考》：*DYNAMIC_LOGS* 配置参数

动态添加逻辑日志以提升性能

可设置 AUTO_LLOG 配置参数，以使数据库服务器可以动态地添加逻辑日志来提升性能。

在设置 AUTO_LLOG 配置参数时，还需指定用于创建新逻辑日志文件的数据库空间，以及服务器停止添加日志的逻辑日志文件大小的整体上限以防止影响性能。

AUTO_LLOG 和 DYNAMIC_LOGS 配置参数在不直接交互影响的不同条件下添加逻辑日志。启用 AUTO_LLOG 配置参数时，将添加逻辑日志以提高性能。启用 DYNAMIC_LOGS 配置参数时，将在更紧急的情况下添加逻辑日志，例如当长事务使用所有可用的日志空间而威胁到服务器运行时。两个配置参数的设置不会相互约束。例如，在 AUTO_LLOG 配置参数中指定的最大大小并不影响 DYNAMIC_LOGS 配置参数可以添加的日志空间量。同样地，AUTO_LLOG 配置参数的值并不影响您可以手动添加的日志空间量。

相关链接

《*SinoDB* 管理员参考》：*AUTO_LLOG* 配置参数

手动添加逻辑日志文件

可使用 onparams 命令来添加逻辑日志文件。

您可能由于以下原因而需手动添加逻辑日志文件：

- 增加分配给逻辑日志的磁盘空间

- 改变逻辑日志文件的大小
- 使打开的事务可以回滚
- 作为将逻辑日志文件移动到不同数据库空间的一部分

  限制: 您不能采取以下操作：

  - 向 BLOB 空间或智能大对象空间添加日志文件。
  - 向具有非缺省页大小的数据库空间添加逻辑日志或物理日志。

一次添加一个逻辑日志文件，最多向任一数据库空间添加 32,767 个逻辑日志文件。一旦向数据库空间添加一个日志文件，该数据库空间就成为关键数据库空间。您可以在存储空间备份期间添加逻辑日志文件。

您可以在以下任一位置添加逻辑日志文件：

- 文件列表的末尾，使用 onparams -a 命令
- 在当前逻辑日志文件之后，使用 onparams -a -i 命令

使用 onparams 添加逻辑日志文件

1. 在 UNIX™ 上，以用户 informix 或 root 的身分登录，或在 Windows™ 上，以 Informix-Admin 组成员的身分登录。
2. 确保数据库服务器处于联机模式、管理模式、静默模式，或者快速恢复方式的清除阶段。

   数据库服务器在清除阶段将以下消息写入日志：

   ```
   Logical recovery has reached the transaction cleanup phase.
   ```
3. 决定要将日志文件添加到日志文件列表的末尾还是当前日志文件之后。

   You can insert a log file after the current log file regardless of the DYNAMIC_LOGS parameter value. Adding a log file of a new size does not change the value of LOGSIZE.

   无论 DYNAMIC_LOGS 参数值为多少，您都可以在当前的日志文件后插入日志文件。添加新大小的日志文件不会更改 LOGSIZE 的值。

   - 以下命令使用 LOGSIZE 配置参数指定的日志文件的大小将逻辑日志文件添加到 logspace 数据库空间中日志文件列表的末尾：

     ```
     onparams -a -d logspace
     ```
   - 以下命令将 1000 KB 的逻辑日志文件插入到 logspace 数据库空间中当前的日志文件之后：

     ```
     onparams -a -d logspace -s 1000 -i
     ```
   - 要添加具有新大小（此例中为 250 KB）的逻辑日志文件，请运行以下命令：

     ```
     onparams -a -d logspace -s 250
     ```
4. 使用 onstat -l 检查日志文件的状态。

   新的日志文件的状态为 A，且立即可用。
5. 下次必须备份数据时，请对根数据库空间和包含新日志文件的数据库空间执行 0 级备份。

   虽然添加日志文件之后不再需要立即备份，但数据结构已更改，所以下一次备份必须为 0 级备份。有关备份数据的信息，请参阅《*SinoDB*® 备份和还原指南》。

有关使用 onparams 添加逻辑日志文件的更多信息，请参阅《*SinoDB*® 管理员参考》。

相关链接

《*SinoDB* 管理员参考》: *onstat -g iov* 命令: 显示 *AIO VP* 统计信息

移动逻辑日志文件 在第261页

删除逻辑日志文件

可使用 onparams 命令来删除逻辑日志文件。

要删除逻辑日志文件并增加数据库空间中的磁盘空间量，可以使用 onparams。数据库服务器始终需要最少三个逻辑日志文件。如果逻辑日志仅由三个日志文件组成，那么就无法删除日志文件。

删除日志文件的规则已更改：

- 如果您删除一个从未被写入的日志文件（状态 A），数据库服务器将其删除，并立即释放空间。
- 如果您删除一个已使用的日志文件（状态 U-B），数据库服务器将其标记为已删除（D）。在您对包含日志文件的数据库空间和根数据库空间进行了 0 级备份之后，数据库服务器将删除该日志文件并释放空间。
- 您无法删除当前正在使用或包含最后检查点记录（状态 C 或 L）的日志文件。

使用 onparams 删除逻辑日志文件：

1. 确保数据库服务器是联机模式、管理模式或静默模式。
2. 运行以下命令删除日志文件号为 21 的逻辑日志文件：onparams -d -l 21

   一次删除一个日志文件。您必须知道要删除的逻辑日志文件的日志文件号。
3. 如果日志文件为新添加（A）的状态，那么可立即删除。

   如果日志文件为已使用（U）的状态，那么将标记为已删除（D）。
4. 要删除已使用日志文件，请对所有数据库空间进行 0 级备份。

   该备份可防止数据库服务器在复原期间使用已删除的日志文件，并确保保留页包含有关当前日志文件数的信息。

有关使用 onparams 删除逻辑日志文件的信息，请参阅《*SinoDB*® 管理员参考》。

有关使用 onlog 显示逻辑日志文件和其唯一标识号的信息，请参阅显示逻辑日志记录 在第262页。

相关链接

《*SinoDB* 管理员参考》: *onparams -d -l lognum:* 删除逻辑日志文件

《*SinoDB* 管理员参考》: *onlog* 实用程序

移动逻辑日志文件 在第261页

更改逻辑日志文件大小

如果想要改变日志文件大小，添加适当大小的新日志文件并随后删除旧的日志文件是较为容易的方式。可使用以下方式来更改逻辑日志文件的大小：

- 使用带有 -s 选项的 onparams 来添加不同大小的新日志文件。

  请参阅手动添加逻辑日志文件 在第259页。
- 若想要数据库服务器创建更大的日志文件，请增加 onconfig 文件中的 LOGSIZE 值。

移动逻辑日志文件

您可能由于性能的原因或为使数据库空间中有更多的空间而想要移动逻辑日志文件。

想查明逻辑日志文件的位置，请运行 onstat -l 命令。虽然移动逻辑日志文件不难，但可能很耗时。

移动逻辑日志文件是由两个简单的操作组合而成：

- （可选）从逻辑日志文件当前所在的数据库空间中删除这些文件。
- 将逻辑日志文件添加到其新的数据库空间

  限制： 不能将逻辑日志文件移动到非缺省页大小的数据库空间中。

数据库服务器必须是联机模式、静默模式或快速恢复模式。

可以通过将 AUTO_LLOG 配置参数设置为 1 和数据库空间名称来更改新的逻辑日志的位置。AUTO_LLOG 配置参数使数据库服务器根据需要添加逻辑日志以提高性能。

示例

以下过程提供如何将 6 个逻辑日志文件从根数据库空间移到另一个数据库空间 dbspace_1 的示例：

1. 执行以下命令，将 6 个新的逻辑日志文件添加到 dbspace_1 ：

```
onparams -a -d dbspace_1
```

2. 对所有存储空间进行 0 级备份以释放当前日志文件以外的所有日志文件。例如，您可以运行以下命令来备份包括当前日志文件的所有日志文件：

```
onbar -l -b -c
```

3. 运行 onmode -l 命令切换到新的当前日志文件。
4. 通过对每个日志文件运行带有日志文件号的 onparams -d -l 命令以删除根数据库空间中的 6 个逻辑日志文件。您不能删除当前的逻辑日志文件。
5. 创建根数据库空间和 dbspace_1 的 0 级备份。例如，运行以下命令：

```
onbar -b root dbspace_1
```

相关链接

根数据库空间的大小 在第163页
逻辑日志文件的位置 在第244页
删除逻辑日志文件 在第261页
手动添加逻辑日志文件 在第259页
逻辑日志文件的位置 在第244页
《*SinoDB* 管理员参考》*: AUTO_LLOG* 配置参数
《*SinoDB* 管理员参考》*: onstat -l* 命令*:* 显示物理和逻辑日志信息
《*SinoDB* 备份和还原指南》*: onbar -b* 语法*:* 备份

显示逻辑日志记录

使用 onlog 实用程序来显示和解释逻辑日志记录。有关使用 onlog 的信息，请参阅《*SinoDB*® 管理员参考》。

控制长事务

有多种方式来控制长事务：

- 调整长事务高水位标志设置
- 调整互斥存取长事务高水位标志设置
- 调整日志文件大小
- 设置可用于事务的日志空间的限制
- 设置事务可运行的时间量的限制

数据库服务器使用 LTXHWM 和 LTXEHWM 配置参数来设置长事务的高水位标志。如果 DYNAMIC_LOGS 设置为 1 或 2，那么 LTXHWM 缺省值为 80%，而 LTXEHWM 缺省值为 90%。如果 DYNAMIC_LOGS 设置为 0，那么 LTXHWM 缺省值为 50%，而 LTXEHWM 缺省值为 60%。

如果您减少高水位标志值，就增加了长事务的可能性。要进行补偿，须分配额外的日志空间。

长事务高水位标志（LTXHWM）

长事务高水位标志 是事务在回滚之前允许跨越的总日志空间的百分比。

如果数据库服务器在最旧已使用的日志文件中发现打开的事务，它会动态添加日志文件。由于日志空间持续增长，所以高水位标志会向外扩展。当日志空间到达高水位标志，数据库服务器会回滚事务。事务回滚和其他进程也会生成逻辑日志记录。数据库服务器持续添加日志文件直到回滚完成以防止逻辑日志耗尽空间。如果存在多个长事务，则回滚多个事务。

例如，数据库服务器有 10 个逻辑日志，且 LTXHWM 设为 98。事务从日志文件 1 开始，更新活动填满日志 1 到 9。数据库服务器在日志文件 10 之后动态添加日志文件 11。只要事务未完成，该过程将持续直到数据库服务器添加了 40 个日志文件。当数据库服务器添加第 50 个日志文件时，事务已赶上高水位标志，因此数据库服务器将回滚该事务。

互斥存取长事务高水位标志（LTXEHWM）

在当前正在回滚的长事务获得对逻辑日志的互斥存取时，发生互斥存取长事务高水位标识。数据库服务器将显著减少日志记录的生成。仅允许当前正在回滚事务的线程和当前正在写入 COMMIT 记录的线程可以访问逻辑日志。限制对逻辑日志的访问，以尽可能地为用户线程（正回滚事务的线程）正在写入的回滚记录保留空间。

> 重要： 如果将 LTXHWM 和 LTXEHWM 均设为 100，那么数据库服务器将永远不会停止长事务。因此，对于正常的数据库服务器操作，必须将 LTXHWM 设置为低于100。要运行未知长度的调度事务，须将 LTXHWM 设置为 100。如果您有足够的磁盘空间，并且不希望当正在回滚长事务时阻挡到其他用户时，可将 LTXEHWM 设置为 100。

调整日志文件大小以防止长事务

当许多用户同时写入日志时，使用较大的日志文件。如果您使用小日志，而且长事务可能会发生，那么降低高水位标志。将 LTXHWM 的值设为 50 ， LTXEHWM 的值设为 60。

如果日志文件太小，数据库服务器可能会在回滚长事务时耗尽日志空间。在这种情况下，数据库服务器无法足够快速地阻挡以在最后一个日志文件填满之前添加日志文件。如果最后一个日志文件填满了，系统会挂起并且显示错误消息。要解决此问题，请关闭并且重新启动数据库服务器。

设置可用于事务的日志空间的限制

SESSION_LIMIT_LOGSPACE 配置参数限制会话可用于每个事务的日志空间量，以防止超过特定大小的事务发生，以及防止来自单个会话的大型事务垄断系统资源。

数据库服务器终止超过日志空间限制的事务，并在数据库消息日志中生成错误。

该大小限制不适用于拥有管理特权的用户，例如 informix 用户或 DBSA 用户。

设置事务可运行的时间量的限制

SESSION_LIMIT_TXN_TIME 配置参数限制会话中事务可运行的时间量，以防止需要大量时间的事务发生，以及防止来自单个会话的长事务垄断系统资源。

数据库服务器终止超过时间限制的事务，并在数据库消息日志中生成错误。

该时间限制不适用于拥有管理特权的用户，例如 informix 用户或 DBSA 用户。

相关链接

动态添加逻辑日志文件以防止事务阻塞 在第257页

《SinoDB 管理员参考》: SESSION_LIMIT_LOGSPACE 配置参数

《SinoDB 管理员参考》: SESSION_LIMIT_TXN_TIME 配置参数

物理日志记录、检查点和快速恢复

这些主题涵盖数据库服务器用来实现数据一致性的三个过程：

- 物理日志记录
- 检查点
- 快速恢复

物理日志 是一个磁盘页集，数据库服务器在此存储页的未修改副本（称为 前映像）。物理日志记录 是存储数据库服务器将要更改的页的前映像的过程。检查点 是数据库服务器将磁盘上的页和共享内存中的页进行同步的时间点。快速恢复 是一种自动进程，可在数据库服务器在不受控制状态下脱机后将其恢复到一致状态。

这些过程确保多个逻辑相关的写入记录为一个单元，并定期地使共享内存中的数据与磁盘上的数据保持一致。

有关管理和监视物理日志和检查点的任务，请参阅管理物理日志 在第270页。

相关链接

数据库服务器维护任务 在第30页

临界区

临界区 是必须作为单个单元执行的代码段（或机器指令）。临界区通过允许线程在其交换出去之前运行一系列指令，可确保线程的完整性。

物理日志记录

物理日志记录 是在更改页实际记录在磁盘上之前存储数据库服务器将要更改的页的过程。在数据库服务器更改共享内存缓冲池中的某些页之前，它将页的前映像存储在共享内存中的物理日志缓冲区。

数据库服务器将这些页的前映像页保留在共享内存中的物理日志缓冲区中，直到某个或多个页清除程序将页清空到磁盘。在数据库发生故障或备份过程中需要这些未修改页时，未修改页可用于提供数据库服务器数据的准确快照。快速恢复和数据库服务器备份会使用这些快照。

除了特殊情况外，数据库服务器在每个检查点回收物理日志。有关检查点的更多信息，请参阅检查点 在第266页。

快速恢复使用物理记录页

在数据库服务器故障后，数据库服务器将使用页的前映像来将磁盘上的这些页复原到它们上一个检查点时的状态。然后，数据库服务器使用逻辑日志记录来将所有数据返回到最近完成的事务之时，其时的物理与逻辑状态是一致的。有关此过程的更详细说明，请参阅快速恢复 在第268页。

备份使用物理记录页

当执行备份时，数据库服务器会执行检查点，并搭配物理日志以标识出备份上页的所属正确版本。在 0 级备份中，数据库服务器备份所有磁盘页。有关更多详细信息，请参阅《*SinoDB*® 备份和还原指南》。

物理记录的数据库服务器活动

如果在检查点之间对一个页进行了多次修改，通常在物理日志中只记录第一个前映像。

物理日志是一种循环日志，其中每个检查点使用一次物理日志中的页。如果设置了 RTO_SERVER_RESTART 配置参数，那么将出现其他物理日志记录来提高快速恢复性能。

物理恢复消息

当快速恢复开始时，数据库服务器会记录以下带有块名称和偏移量的消息：

```
Physical recovery started at page chunk:offset.
```

当快速恢复完成时，数据库服务器会记录以下带有已检查和已恢复页数的消息：

```
Physical recovery complete: number pages examined, number pages restored.
```

物理日志记录和简单大对象

物理日志中的数据库服务器页可以是任何一种数据库服务器页，包括表空间（tblspaces）中的简单大对象。甚至开销页（例如块可用列表页、BLOB 空间自由图页和 BLOB 空间位图页）也会在页上数据修改和清空到磁盘之前复制到物理日志。

BLOB 空间 BLOB 页不在物理日志中记录。有关 BLOB 空间日志记录的更多信息，请参阅记录 *BLOB* 空间和简单大对象 在第247页。

物理日志记录和智能大对象

智能大对象的用户数据部分没有物理记录。但是，元数据有物理记录。有关智能大对象的信息，请参阅智能大对象空间 在第146页。

物理日志的大小和位置

当数据库服务器初始化磁盘空间时，它将物理日志置于根数据库空间中。物理日志的初始大小由 PHYSFILE 配置参数设置。

在首次初始化数据库服务器之后，您可以使用 onparams 实用程序来更改物理日志的大小和位置。

为了提高性能（特别是，要减少对根数据库空间的写入数量，以及减少磁盘竞争），您可以将物理日志从根数据库空间移到另一个数据库空间，最好是不包含活动表或逻辑日志文件的磁盘。为获得最佳性能，可创建物理日志空间来存储物理日志，并且允许数据库服务器按需扩展物理日志大小以提高性能。

> 建议： 把极重要的数据库空间放至于容错存储设备上。如果物理日志所在的存储设备不是容错存储设备，请使用 SinoDB® 对包含物理日志的数据库空间进行镜像。这将在存储设备发生故障时可保护数据库。但是，如果对物理日志空间进行了镜像，那么它将无法扩展。

相关链接

物理日志空间 在第153页
更改物理日志的位置和大小 在第270页

用于估计物理日志大小的策略

物理日志大小取决于两个因素：事务生成物理日志活动的速率和是否设置了 RTO_SERVER_RESTART 配置参数。

事务生成物理日志活动的速率可以影响检查点性能。在检查点处理期间，如果因事务持续生成物理日志数据而导致物理日志开始变得太满，数据库服务器将阻塞事务以使检查点完成并避免物理日志溢出。

为避免事务阻塞，数据库服务器必须具有足够的物理日志空间以容纳检查点处理期间发生的所有事务活动。物理日志达到 75% 满时触发检查点。当物理日志达到 75% 满时，必须等检查点处理完成之后才能够使用剩馀的 25% 的物理日志。由于每个活动事务都可能生成物理日志活动，因此，一旦系统检测到有发生物理日志溢位的可能性，就会发生事务阻塞。

例如，假设您有 1 GB 物理日志和 1000 个活动事务。如果每个事务同时处于临界区，那么 1000 个活动事务就有生成近 80 MB 物理日志活动的可能性。当物理日志填满 750 MB 时，数据库服务器会触发检查点。如果在物理日志填满 920 MB 之前不能完成检查点处理，那么将发生事务阻塞直到检查点完成。如果发生事务阻塞，服务器将自动触发更频繁的检查点以避免事务阻塞。您可以禁用自动检查点的生成。

如果存在许多脏分区，那么即使物理日志填充度未达到 75%，服务器也可能触发检查点，因为将已修改的分区数据清空到磁盘需要物理日志空间。服务器在检查物理日志填充度是否达到为 75% 时，还会检查以下条件是否为真：

```
（已使用物理日志页 + 脏分区数）>=（物理日志大小 * 9）/10
```

有关检查点处理和自动检查点的更多信息，请参阅检查点 在第266页。

估算物理日志大小时要考虑的第二个因素取决于是否使用 RTO_SERVER_RESTART 配置参数指定快速恢复的目标时间量。如果无需考虑快速恢复时间，就不需要启用 RTO_SERVER_RESTART 配置参数。如果指定了 RTO_SERVER_RESTART 配置参数的值，那么事务活动将生成附加物理日志活动。

通常，该附加物理日志活动对事务性能的影响很小或几乎没有影响。额外的日志记录用于在快速恢复期间辅助缓冲池，从而以最佳方式执行日志重播。如果物理日志比所有缓冲池的总大小要大很多，那么在快速恢复期间将出现页清空和页故障。页清空和页故障会大幅地降低快速恢复的性能，而且数据库服务器将不能维持 RTO_SERVER_RESTART 的策略。

对于缓冲池空间小于 4 GB 的系统，物理日志的大小可定在所有缓冲池总大小的 110%。对于较大的缓冲池，以 4 GB 的物理日志空间开始，然后监视检查点活动。如果检查点发生过于频繁，并且似乎会影响性能，请增加物理日志的大小。

当数据库服务器配置了小的物理日志并且具有大量用户，那么可能发生一种称为物理日志溢出的罕见情况。请遵照上述有关大小的准则可帮助避免物理日志溢出。每当数据库服务器检测到次优配置时，就会向消息日志生成性能警告。

如果检测到次优配置，您可以使用 onstat -g ckp 命令来显示配置建议

当事务日志记录关闭时物理日志溢出

如以下示例所示，如果在事务日志记录关闭的数据库中使用简单大对象或智能大对象，物理日志将会溢出。

在数据库服务器处理简单大对象时，数据库服务器存储在磁盘上的简单大对象的每个部分均可分别记录，从而允许线程退出每个部分之间的代码临界区。但如果关闭日志记录，那么数据库服务器必须对一个临界区中的简单大对象执行所有操作。如果简单大对象很大而物理日志很小，那么该情况可能导致物理日志填满。如果发生这种情况，数据库服务器会将以下消息发送到消息日志：

```
Physical log file overflow
```

数据库服务器将启动关机。有关建议的更正操作，请查看消息日志。

检查点

数据库服务器会定期将缓冲池内的事务和数据清空到磁盘。在事务和数据清空到磁盘之前，数据和事务是处于流出的状态。数据库服务器将事务写入逻辑日志，而不是每个事务完成之后立即强迫写入磁盘。数据库服务器在事务发生时记录事务。如出现系统故障，服务器将执行以下操作：

- 重放日志，以重做和复原事务。
- 将数据库返回至与发生故障时数据库系统的状态一致的状态。

为了便于数据库系统的复原和逻辑恢复，数据库服务器会生成一致性点，称为检查点。检查点是日志中数据库系统的已知和一致状态建立的时间点。通常，检查点涉及记录特定量的信息，因此，如果发生故障，数据库服务器可以在已建立的点上重新启动。

检查点的目的在于定期地将逻辑日志中的重新启动点向前移动。如果检查点不存在且发生故障，那么数据库服务器需要处理自系统重新启动以来逻辑日志中记录的所有事务。

检查点可在以下任一情况中出现：

- 特定事件发生时。例如，每当服务器添加一个数据库空间或执行数据库备份时，将出现检查点。

  通常，这类事件会触发阻塞事务处理的检查点。因此，这些检查点称为阻塞检查点。

- 当资源限制发生时。例如，每个逻辑日志空间跨度都需要检查点，以保证日志具有开始快速恢复的检查点。数据库服务器将在物理日志填充度达到 75% 时触发检查点以避免物理日志溢出。

  资源限制触发的检查点通常不会阻塞事务。因此，这些检查点称为非阻塞检查点。

  但是，如果在检查点处理期间数据库服务器将要耗尽资源时，那么在检查点处理的中段将发生事务阻塞，以保证在资源耗尽之前完成检查点。如果事务遭到阻塞，那么服务器会更频繁地触发检查点，以避免在检查点处理期间的事务阻塞。有关更多信息，请参阅用于估计物理日志大小的策略 在第265页。

如果发生故障且辅助服务器成为主服务器时，则两台服务器上检查点之间的差异会影响重新连接尝试。如果在新辅助服务器上的检查点不存在于新主服务器上，则辅助服务器对主服务器的连接尝试会失败。辅助服务器必须完全复原之后才能连接到主服务器。

自动检查点会使数据库服务器触发更频繁的检查点，以避免事务阻塞。自动检查点会尝试监视系统活动和资源使用情况（物理日志和逻辑日志的使用情况以及缓冲池脏的程度）以能够及时触发检查点，以便在物理日志或逻辑日志耗尽之前完成检查点的处理。数据库服务器为每个逻辑日志空间跨度生成至少一个自动检查点。这保证可以开始快速恢复的检查点的存在。使用 AUTO_CKPTS 配置参数，可在数据库服务器启动时启用或禁用自动检查点。（可以使用 onmode -wm 或 onmode -wf 动态地启用或禁用自动检查点。）

提示:

- 如果自动检查点由于物理日志活动而被频繁触发，那么您可以增加物理日志大小或使用物理日志空间，以自动调整物理日志资源。
- 如果自动检查点由于物理日志活动而被频繁触发，那么您可以在 onconfig 文件中设置 AUTO_LLOG 参数，以允许服务器自动增加逻辑日志空间来减少检查点的触发频率。

手动检查点是您可以启动的基于事件的检查点。数据库服务器提供两种方式来确定发生意外中断时快速恢复所需的时间。

- 使用 CKPTINTVL 配置参数设置服务器触发检查点的频率。
- 使用 RTO_SERVER_RESTART 配置参数设置快速恢复所需的时间。

  当使用 RTO_SERVER_RESTART 配置参数时：

  - 数据库服务器将忽略 CKPTINTVL 配置参数。
  - 数据库服务器将监视物理日志和逻辑日志的使用情况，以估计快速恢复所需的时间。如果服务器估计的快速恢复所需的时间超过 RTO_SERVER_RESTART 配置参数中所指定的时间，那么服务器将自动触发检查点。

RTO_SERVER_RESTART 配置参数设置的是目标时间量，不是保证的时间量。可增加重新启动所需时间的多个因素也会影响快速恢复所需时间。这些因素包括意外中断时正在回滚的活动长事务。有关 RTO_SERVER_RESTART 和 AUTO_CKPTS 配置参数的更多信息，请参阅《*SinoDB*® 管理员参考》中有关配置参数的主题。

在检查点之间用于清空缓冲池的 LRU 值

在检查点之间用于清空缓冲池的 LRU 值对检查点的性能不是特别重要。BUFFERPOOL 配置参数中设置的 lru_max_dirty 值和 lru_min_dirty 值通常仅对于维护页替换的足够清洁页是必需的。以将 lru_min_dirty 设置为 70，lru_max_dirty 设置为 80 作为开始。

如果在检查点期间事务被阻塞，那么数据库服务器随后将尝试增加检查点频率，以消除事务阻塞。当服务器搜索可用页以进行页替换时发生了前台写入，那么服务器随后将自动增加 LRU 清空频率，以防止该事件再次发生。当数据库服务器完成页替换并找到频繁访问的页时，那么服务器将自动增加 LRU 清空频率。对于 LRU 清空的任何自动调整都不会存留在 onconfig 文件。

有关监视和调整检查点参数的更多信息，以及有关 LRU 调整的信息，请参阅《*SinoDB*® 性能指南》。

备份期间的检查点

执行备份时，数据库服务器会运行检查点并将所有更改过的页清空到磁盘。启动复原时，数据库服务器会重新应用所有逻辑日志记录。

有关 ON-Bar 或 ontape 的信息，请参阅《*SinoDB*® 备份和还原指南》。

快速恢复

快速恢复是一种自动的、具有容错性的功能，每当数据库服务器从脱机模式转为静默模式、管理模式或联机模式时就会执行该功能。无需为快速恢复执行任何管理操作；它是自动执行的功能。

快速恢复过程会检测数据库服务器上次脱机是否在不非受控条件下进行的。如果是，快速恢复将数据库服务器返回到物理与逻辑一致性状态。

如果快速恢复过程发现数据库服务器是在受控方式下脱机的，那么快速恢复过程终止并将数据库服务器转为联机模式。

请参阅检查点之后的快速恢复 在第268页。

需要快速恢复

快速恢复在任何导致数据库服务器内存内容丢失的故障之后将数据库服务器复原至物理和逻辑一致性的状态。例如，操作系统在无警告的情况下发生故障。系统故障不会毁坏数据库，但会影响故障发生时正在进行的事务。

启动快速恢复的情境

每当管理员将数据库服务器从脱机模式转为静默模式、管理模式或联机模式时，数据库服务器都会检查是否需要快速恢复。

作为共享内存初始化的一部分，数据库服务器将检查物理日志的内容。当数据库服务器在受控状况下关闭时，物理日志为空。从联机模式转为静默模式的过程包含一个清空物理日志的检查点。因此，如果数据库服务器在物理日志中找到页，很明显地数据库服务器是在非受控条件下脱机的，并且快速恢复开始。

快速恢复和已缓冲日志记录

如果数据库使用已缓冲日志记录（如已缓冲的事务日志纪录 在第238页中所述），那么与已提交事务相关联的一些逻辑日志记录可能不会在发生故障时写入逻辑日志。如果发生这种情况，快速恢复无法恢复这些事务。快速恢复仅能恢复在磁盘上的逻辑日志中存储了相关联 COMMIT 记录的事务。（由于这个原因，已缓冲日志记录代表性能与数据脆弱性之间的平衡。）

快速恢复期间可能的物理日志溢出

在快速恢复期间，物理日志可能会溢出。如果发生溢出，数据库服务器将尝试向名为 plog_extend.*servernum* 的磁盘文件扩展物理日志空间。该文件缺省位置为 $INFORMIXDIR/tmp。

使用 ONCONFIG 的 PLOG_OVERFLOW_PATH 参数可定义用于创建此文件的位置。

当在快速恢复期间执行第一个检查点时，数据库服务器将移除 plog_extend.*servernum* 文件。

快速恢复和无日志记录

对于不使用日志记录的数据库或表，快速恢复将数据库复原到其在最近检查点之时的状态。自上一次检查点以来对数据库所做的所有更改都将丢失。

检查点之后的快速恢复

作为共享内存初始化的一部分，快速恢复将数据库服务器恢复到一致性状态。恢复所有已提交事务，并回滚所有未提交事务。

快速恢复按以下步骤发生：

1. 数据库服务器使用物理日志中的数据将所有磁盘页返回到它们在最近检查点时的状态。该点称为物理一致性。
2. 数据库服务器在逻辑日志文件中查找最近的检查点记录。
3. 数据库服务器前滚在最近检查点记录之后写入的所有逻辑日志记录。
4. 数据库服务器回滚所有未提交事务。在 XA 资源管理器可用之前，某些 XA 事务可能尚未解决。

服务器恢复到上一个检查点状态

要将所有磁盘页返回到它们在最近检查点时的状态，数据库服务器需将存储在物理日志中的前映像写入共享内存中，然后写回磁盘。每个物理日志中的前映像都包含在检查点之后更新的页的地址。当数据库服务器将物理日志中的前映像页写入共享内存并又写回磁盘时，自最近检查点以来对数据库服务器数据的更改将撤销。此时，数据库服务器在物理上是一致的。下图说明了此步骤。

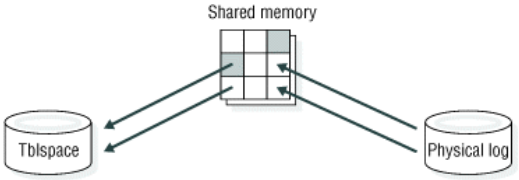

图 47: 将物理日志中所有剩馀的前映像写回磁盘

服务器在逻辑日志中定位检查点记录

在返回到上一个检查点状态之后，数据库服务器将在逻辑日志中定位最近检查点记录的地址。最近检查点记录可保证其位于磁盘上的逻辑日志中。

服务器前滚逻辑日志记录

After locating the checkpoint record in the logical log, the database server rolls forward the logical-log records that were written after the most recent checkpoint record. This action reproduces all changes to the databases since the time of the last checkpoint, up to the point at which the uncontrolled shutdown occurred. The following figure illustrates this step.在逻辑日志中定位检查点记录之后，数据库服务既将前滚在最近检查点记录之后写入的逻辑日志记录。此操作重新再现自上一个检查点以来直到发生不受控的关机之时对数据库的所有更改。下图说明了此步骤。

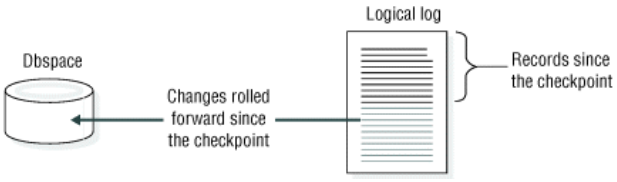

图 48: 前滚自最近检查点以来写入的逻辑日志记录

服务器回滚未提交事务

在前滚逻辑日志记录之后，数据库服务器将回滚在系统故障时尚未提交的事务的所有逻辑日志记录。由于前滚所有已提交事务并回滚所有为提交事务，因此所有数据库在逻辑上是一致的。在 XA 资源管理器可用之前，某些 XA 事务可能尚未解决。

已完成两阶段提交的第一阶段的事务属于例外情况。有關更多信息，请参阅两阶段提交协议如何处理故障在第426页。

由于一个或多个事务可能跨及几个检查点而未提交，所以该回滚程序可能跨越最近检查点记录往回读出逻辑日志。包含打开的事务的记录的所有逻辑日志文件均可用于数据库服务器，因为日志文件要等到在其所包含的所有事务关闭之后才会被释放。

下图说明了回滚过程。此处，未提交的更改从逻辑日志回滚到特定磁盘上的数据库空间 。当快速恢复完成时，数据库服务器将转回静默模式、管理模式或联机模式。

图 49: 回滚所有未完成事务。

管理物理日志

这些主题描述了以下过程：

- 更改物理日志的位置和大小
- 监视物理日志、物理日志缓冲区和逻辑日志缓冲区
- 监视并强制执行检查点

有关背景信息，请参阅物理日志记录、检查点和快速恢复 在第264页。

更改物理日志的位置和大小

可使用 onparams 实用程序来更改物理日志的位置和大小。

您可以移动物理日志文件来尝试提高性能。当数据库服务器初始化磁盘空间时，它会将分配给物理日志的磁盘页放置于根数据库空间。您可以通过将物理日志移动到另一个数据库空间来提高性能。

您可以将物理日志移动到数据库空间或物理日志空间。当物理日志位于物理日志空间时，数据库服务器可以按需增加物理日志的大小以提高性能。当物理日志位于数据库空间时，您必须手动增加物理日志的大小。

要将物理日志移至物理日志空间，请运行 onspaces -c -P 命令或运行带有 create plogspace 参数的 SQL 管理 API admin() 或 task() 函数来创建此物理日志空间。要更改物理日志空间的位置，请创建一个新的物理日志空间。将物理日志移至新的物理日志空间，然后删除旧的物理日志空间。

将物理日志移至数据库空间的先决条件：

- 在 UNIX™ 上以用户 informix 或 root 身份登录，或者在 Windows™ 上以 Informix-Admin 组成员身份登录。
- 通过运行 oncheck -pe 命令检查目标块是否有足够的连续空间可用。

  分配给物理日志的空间必须是连续的。当更改物理日志的大小或位置时，如果目标数据库空间没有足够的连续空间，那么服务器将不更改物理日志。此外，如果在初始化数据库服务器时物理日志没有足够的资源，那么初始化将失败。数据库空间必须使用缺省页大小。

要将物理日志移至数据库空间，请运行 onparams -p -s 命令或 运行带有 create dbspace 参数的 SQL 管理 API admin() 或 task() 函数。

以下示例更改物理日志的大小和位置。新的物理日志大小是 400 KB，且位于 dbspace6 数据库空间中：

```
onparams -p -s 400 -d dbspace6
```

相关链接

根数据库空间的大小 在第163页
监视物理和逻辑日志记录活动 在第270页
物理日志的大小和位置 在第265页
oncheck -pe 命令 在第201页
《*SinoDB* 管理员参考》*: onparams -p:* 更改物理日志参数
《*SinoDB* 管理员参考》*: onspaces -c -P:* 创建物理日志空间

监视物理和逻辑日志记录活动

监视监控物理日志以确定在检查点出现前物理日志文件的使用百分比。您可以使用此信息来找出物理日志文件的最佳大小。该大小必须足够大，使数据库服务器不需过于频繁地强迫执行检查点，也必须足够小，以节省磁盘空间并保证快速恢复。

监视物理日志和逻辑日志缓冲区以确定它们对于当前处理级别是否为最佳大小。要监视的重要统计信息就是每个磁盘的写入页数的统计信息。有关调整物理日志和逻辑日志缓冲区的更多信息，请参阅《*SinoDB*® 性能指南》。

要监视物理日志文件、物理日志缓冲区和逻辑日志缓冲区，请使用以下命令。

实用程序	命令	其他信息
命令行	onstat -l	第一行显示每个物理日志缓冲区的以下信息： • 已用的缓冲区页数 (bufused) • 每个物理日志缓冲区以页计的大小 (bufsize) • 向缓冲区写入的页数 (numpages) • 从缓冲区向磁盘写入的次数 (numwrits) • 向缓冲区写入的页数与向磁盘写入的次数之比率 (pages/IO) 第二行显示物理日志的以下信息： • 物理日志文件中首页的页码 (phybegin) • 物理日志文件以页计的大小 (bufsize) • 日志中的当前位置，即发生下次写入的位置，以页码指定 (physpos) • 日志中已用的页数 (phyused) • 已用的总物理日志页数的百分比 (%used) 第三行显示每个逻辑日志缓冲区的以下信息： • 已用的缓冲区页数 (bufused) • 每个逻辑日志缓冲区以页计的大小 (bufsize) • 向缓冲区写入的记录数 (numrecs) • 向缓冲区写入的页量 (numpages) • 从缓冲区向磁盘写入的次数 (numwrits) • 缓冲区中记录数与页数的比率 (recs/pages) • 向缓冲区写入的页数与向磁盘写入的次数之比率 (pages/IO)
命令行	onparams -p	移动物理日志或调整物理日志大小
命令行	onmode -l	前进到下一个逻辑日志文件

有关 onstat -l 输出的示例的更多信息，请参阅《*SinoDB*® 管理员参考》。

有关使用 SQL 管理 API 命令（而不是一些 onparams 和 onmode 命令）的信息，请参阅使用 *SQL* 管理 *API* 执行远程管理 在第458页和《*SinoDB*® *SQL* 指南: 语法》。

相关链接

更改物理日志的位置和大小 在第270页

监视检查点信息

监视检查点活动以查看各种信息，包含线程需要等待检查点完成的次数。这些信息对于确定检查点间隔是否适合非常有用。

要监视检查点，请使用以下命令。

实用程序	命令	其他信息
onstat 实用程序	onstat -m	查看消息日志中的最近 20 行。

实用程序	命令	其他信息
		如果最近 20 行中不包含检查点消息，那么请直接使用文本编辑器读取消息日志。数据库服务器在检查点结束时将个别检查点消息写入日志。 如果出现检查点，但数据库服务器没有页可写入磁盘，那么数据库服务器不会向消息日志写入任何消息。
onstat 实用程序	onstat -p	获取这些检查点统计信息： • numckpts：自数据库服务器联机以来发生的检查点数。 • ckptwaits：用户线程等待检查点完成的次数。数据库服务器将防止用户线程在检查点期间进入临界区。

有关调整检查点时间间隔的信息，请参阅《*SinoDB*® 性能指南》。

打开或关闭检查点调整

要打开自动检查点调整，请使用 onmode -wf AUTO_CKPTS=1 命令。要关闭自动检查点调整，请使用 onmode -wf AUTO_CKPTS=0 命令。

强制执行检查点

必要时，可使用 onmode 或 SQL 管理 API 命令来强制执行检查点。

在以下任一情况下强制执行检查点：

- 要释放包含最近检查点记录且已备份但尚未释放（onstat -l 的 U-B-L 或 U-B 状态）的逻辑日志文件。
- 在发出 onmode -sy 将数据库服务器置于静默模式之前。
- 在构建大型索引后，如果数据库服务器在下一个检查点之前终止。索引构建将在下一次重新启动数据库服务器时重新启动。
- 如果检查点有很长时间未出现，而您要尝试进行可能中断数据库服务器的系统操作
- 如果前台写入耗用比预期更多的资源（强制执行检查点暂时将资源使用量降为零）
- 运行 dbexport 或卸载表之前，请确保在导出或卸载数据之前所有数据在物理上的一致性。
- 使用 PUT 或 INSERT 语句执行大量表装载后 （因为表加载使用缓冲区高速缓存，强制执行检查点可清除缓冲区高速缓存。）

要强制执行检查点，请运行 onmode -c。

有关使用 SQL 管理 API 命令（而不是一些 onmode 命令）的信息，请参阅使用 *SQL* 管理 *API* 执行远程管理 在第458页和《*SinoDB*® *SQL* 指南: 语法》.

服务器提供的检查点统计信息

数据库服务器提供有关前 20 个检查点的历史信息。可以通过 SMI sysckptinfo 表来访问这些信息。

相关链接

《SinoDB 管理员参考》: sysckptinfo

SMI 表

查询 sysprofile 表以获取物理日志和逻辑日志缓冲区的统计信息。sysprofile 表还提供与 onstat -p 选项所提供信息相同的检查点统计信息。这些行包含以下统计信息。

plgpagewrites
: 写入物理日志缓冲区的页数

plgwrites
: 从物理日志缓冲区向物理日志文件写入的次数

llgrecs
写入逻辑日志缓冲区的记录数

llgpagewrites
写入逻辑日志缓冲区的页数

llgwrites
从逻辑日志缓冲区向逻辑日志文件写入的次数

numckpts:
自数据库服务器联机以来所出现的检查点数

ckptwaits
线程在检查点期间等待检查点完成进入临界区的次数

value
numckpts 和 ckptwaits 的值

打开或关闭自动 LRU 调整

使用 AUTO_LRU_TUNING 配置参数可在数据库服务器启动时启用或禁用自动 LRU 调整。

如果已设置 RTO_SERVER_RESTART 配置参数，数据库服务器将自动触发检查点，以便能在指定的时间内将服务器转为联机。如果服务器不能符合 RTO_SERVER_RESTART 策略，则数据库服务器将在消息日志中打印警告消息。

要关闭特定会话的自动 LRU 调整，请发出 onmode -wm AUTO_LRU_TUNING=*0* 命令。

要在会话期间关闭自动 LRU 调整后再将其打开，请发出 onmode -wm AUTO_LRU_TUNING=*1* 命令。

自动 LRU 调整更改会影响所有缓冲池，并调整 BUFFERPOOL 配置参数中的 lru_min_dirty 和 lru_max_dirty 的值。

For more information about LRU tuning, see the 《*SinoDB*® 性能指南》.

有关 LRU 调整的更多信息，请参阅《*SinoDB*® 性能指南》。

第 IV 部分

容错

镜像

这些主题描述数据库服务器的镜像功能。有关如何执行镜像任务的指示信息，请参阅使用镜像 在第277页。

相关链接

功能配置 在第28页

块 在第138页

镜像

镜像是一种策略，用于将已定义的数据库空间、BLOB 空间或智能大对象空间的主块与同等大小的镜像块联结成对。

每次写入主块将自动伴随向镜像块的相同写入。下图说明了此概念。如果主块发生故障，镜像可使您从镜像块读取和向镜像块写入，直到您恢复主块，而不会中断用户对数据的访问。

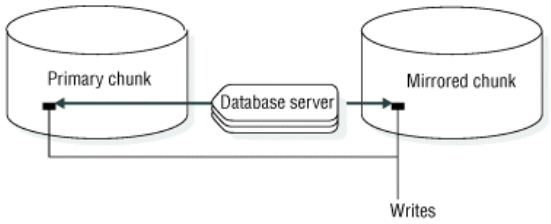

图 50: 将数据同时写入主块和镜像块

在通过网络管理的磁盘上不支持镜像。同一数据库服务器实例必须管理镜像集的所有块。

镜像的好处

如果发生介质故障，镜像将为数据库服务器管理员提供不必使数据库服务器脱机就可恢复数据的方法。该功能将带来更大的可靠性和更少的系统停机时间。而且，如果镜像数据的块位于不同介质上，那么应用程序可以继续对主块位于受影响介质上的数据库进行读取和写入。

任何关键数据库都必须位于镜像数据库空间中。必须对包含数据库服务器保留页的根数据库空间制作镜像。

镜像成本

磁盘空间成本和性能成本与镜像相关。磁盘空间成本是由于需要额外的空间存储镜像数据。性能成本源于对主块和镜像块都执行写入。对于磁盘写入使用多个虚拟处理器可降低该性能成本。根据块中数据的位置，使

用分开读取（数据库服务器从主块或镜像块读取数据）实际上对于只读数据可使性能提高。有关数据库服务器如何执行对镜像块的读写的更多信息，请参阅处理期间执行的操作 在第276页。

不制作镜像的后果

如果未对数据库空间制作镜像，那么在介质故障后从存储空间备份进行恢复的频率会增加。

如果镜像块受到介质故障影响，数据库服务器仅从仍联机的块中读取，直到您将脱机块恢复为联机。当镜像对的第二个块脱机时，数据库服务器将无法访问存储在该块上的数据。如果该块包含逻辑日志文件、物理日志或根数据库空间，那么数据库服务器将立即脱机。如果该块不包含逻辑日志文件、物理日志或根数据库空间，那么数据库服务器可以继续运行，但是线程无法从脱机块中读取或写入脱机块。如果无镜像的块脱机，那么您必须通过从备份恢复数据库空间来恢复该块。

要制作镜像的数据

理想情况下必须对所有数据制作镜像。如果磁盘空间是个问题，那么您可能就无法如此做。在这种情况下，就选择某些关键块来制作镜像。

关键块始终包括作为根数据空间一部分的块、存储逻辑日志文件的块和存储物理日志的块。如果其中任一关键块发生故障，那么数据库服务器会立即脱机。

如果某些块存储了对您的业务很关键的数据，请优先为这些块制作镜像。

此外，优先为存储常用数据的块制作镜像。该操作可以确保当一个广泛使用的块脱机时许多用户的活动不会停止。

镜像的替代方法

正如本手册中的说明，镜像是一种数据库服务器的功能。您的操作系统或硬件可能提供替代解决方案。

如果您正在考虑操作系统提供的镜像功能而不是数据库服务器提供的镜像功能，请在决定使用哪一种之前将两者的功能实现进行比较。镜像过程中最慢的步骤是将数据写入磁盘。数据库服务器以并行方式对镜像块执行写入的策略可帮助减少该步骤所需要时间。（请参阅磁盘写入镜像块 在第276页。）而且，数据库服务器镜像使用分开读取来提高读取性能。（请参阅磁盘从镜像块读取 在第277页。）未使用并行镜像写入和分开读取的操作系统镜像功能可能会降低性能。

您可以同时运行数据库服务器镜像和操作系统镜像。他们各自独立运行。在某些情况下，您可能决定同时使用数据库服务器镜像和操作系统提供的镜像功能。例如，您可能在同一个磁盘驱动器上同时有数据库服务器数据和其他数据。您可以使用操作系统镜像对其他数据进行镜像，使用数据服务器镜像为数据库服务器数据进行镜像。

逻辑卷管理器

逻辑卷管理器是替代镜像方案。某些操作系统供应商提此类型的实用程序以便将多个磁盘视为一个文件系统。将数据保存至两个以上的磁盘将给您更多保护以防止介质故障，但是额外的写入会增加性能成本。

硬件镜像

另一方案是使用硬件镜像，例如低价磁盘冗余阵列（RAID）。此类型的硬件镜像的优势在于它需要比数据库服务器镜像更少的磁盘空间来存储相同数量的数据以防止介质故障。

某些硬件镜像系统支持 热交换。您可以在保持数据库服务器联机交换坏磁盘。建议在执行热交换之前减少 I/O 活动。

> 重要： 如果在使用硬件镜像时数据库服务器发生问题，请参阅操作系统或磁盘文件或技术支持以获取协助。

外部备份与复原

如果使用硬件磁盘镜像，则使用外部备份和复原使系统恢复联机的速度将比使用传统 ON-Bar 命令更快。有关外部备份和复原的更多信息，请参阅《*SinoDB*® 备份和还原指南》。

镜像过程

本节详细地描述了镜像的过程。有关如何执行诸如创建镜像块、启动镜像、更改镜像块状态等等的镜像操作指南，请参阅使用镜像 在第277页。

创建镜像块

当您指定镜像块时，数据库服务器就将所有数据从主块复制到镜像块。这个复制过程称为恢复。一旦恢复完成，镜像就立即开始。

如果您开始对包含逻辑日志文件的数据库空间中的块制作镜像，则会延迟标记镜像开始的恢复过程。在您创建根数据库空间的 0 级备份后，才会开始对包含逻辑日志文件的数据库空间制作镜像。延迟可确保如果包含逻辑日志文件的主块在数据库空间复原期间变为不可用时，数据库服务器可以使用镜像的逻辑日志文件。

0 级备份将更新的数据库服务器配置信息（包括有关新镜像块的信息）从根数据库空间保留页复制备份。如果执行数据复原时，那么备份开头的更新配置信息将在主块不可用时指示数据库服务器去查找逻辑日志文件的镜像副本。如果该新的存储空间备份信息不存在，那么数据库服务器就无法利用镜像的日志文件。

由于类似原因，您无法在创建数据库空间备份时对包含逻辑日志文件的数据库空间制作镜像。 新的信息必须在数据库空间备份磁带的第一个块中，其无法在备份开始后复制到那个位置。

有关创建镜像块的详细信息，请参阅使用镜像 在第277页。

镜像状态标志

数据库空间、BLOB 空间和智能大对象空间具有指示它们已镜像还是未镜像的状态标志。

镜像开始之前，您必须执行根数据库空间的 0 级备份。

块具有指示以下信息的状态标志：

- 块是主块还是镜像块
- 块当前是联机、脱机、需根数据库空间 0 级备份的新镜像块，还是处于恢复过程中

有关这些块的状态标志的描述，请参阅《*SinoDB*® 管理员参考》中有关 onstat -d 选项的描述。有关如何显示这些状态标志的信息，请参阅监视磁盘使用 在第199页。

恢复

当数据库服务器恢复镜像块时，它执行与镜像开始时所使用过程相同的恢复过程。镜像恢复过程包括将数据从现有联机块上复制到新的已修复块上，直到两个块完全相同。

当您启动恢复时，数据库服务器将脱机块置于恢复模式，并将信息从联机块复制到恢复块。当恢复完成后，该块自动变成联机状态。无论您正在恢复镜像对的主块还是镜像块，均执行相同步骤。

> 提示： 在恢复过程中，您仍可以使用联机块。如果将数据写入已复制到恢复块的页，那么数据库服务器会在继续恢复过程之前在恢复块上更新相应页。

有关如何恢复脱机块的信息，请参阅恢复镜像块 在第281页中有关恢复镜像块的信息。

处理期间执行的操作

这些主题说明了镜像块磁盘 I/O 的详细信息以及数据库服务器如何处理这些块的介质故障。

磁盘写入镜像块

在数据库服务器处理期间，数据库服务器通过执行两个并行写入来进行镜像：一个写入主块，一个写入镜像块。

磁盘从镜像块读取

因为数据的两个版本位于不同的磁盘上，所以数据库服务器使用镜像来提高读取的性能。根据块的那一半包含数据页的地址，将决定是从主块或镜像块读取数据页。此功能称为分开读取。分开读取通过减少磁盘搜索时间来提高性能。由于磁盘头必须移动的最大距离减半，所以减少磁盘搜索时间。下图说明了分开读取。

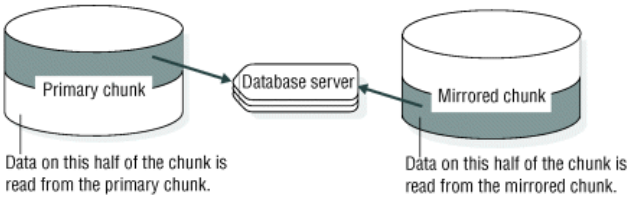

图 51: 分开读取减少了磁盘头必须移动的最大距离

检测介质故障

数据库服务器在第一次打开块时检查返回码，然后在任何读取或写入后均检查返回码。无论何时数据库服务器检测到主 （或镜像） 块设备发生故障，它会将块状态标志设置为脱机 (D)。有关块状态标志的信息，请参阅镜像状态标志 在第276页。

如果数据库服务器检测到主 （或镜像） 块设备发生故障，对保持联机的那个块仍继续读取和写入。即使管理员故意是其中一个块脱机，情况也是如此。

管理员恢复脱机块并将其返回至联机状态之后，会再次在主块和镜像块之间分开读取，并对两个块均进行写入。

块恢复

数据库服务器使用异步 I/O 以最小化恢复块所需的时间。对联机块的读取可以与对脱机块的写入重覆进行，而不是这两个过程循序发生。也就是说，执行读取的线程不需要等待执行写入的线程完成，就可读取更多数据。

停止镜像的结果

当结束镜像时，数据库服务器会立即释放镜像块，并使空间可用于重新分配。结束镜像的过程仅需几秒钟。

在结束镜像后创建根数据库空间的 0 级备份，以确保将包含更新镜像块信息的保留页复制到备份。此操作可防止恢复过程假设镜像数据仍可用。

镜像块的结构

镜像块包含与主块相同的控制结构，如下所示：

- BLOB 空间块的镜像包含 BLOB 空间开销页。
- 数据库空间块的镜像包含数据库空间开销页。
- 智能大对象空间的镜像包含元数据页。

有关这些结构的信息，请参阅《*SinoDB*® 管理员参考》中的磁盘结构和存储章节中有关镜像块结构的部分。

由监视块 在第199页中所述方法之一提供的磁盘空间使用显示将始终指示镜像块是已满，即使主块有可用空间。已满镜像块表示除了作为主块的镜像外，块中无空间可用。只要主块和镜像块联机，状态会一直保持为已满。

如果主块脱机而镜像块成为主块，那么磁盘空间分配报告就会准确地描述新主块的填充度。

使用镜像

这些主题描述了使用数据库服务器镜像功能所需的各种镜像任务。它概述对数据制作镜像所需的步骤。

准备对数据制作镜像

本节说明如何在未启用镜像功能的数据库服务器上开始镜像数据。

准备对数据镜像：

1. 将数据库服务器脱机并启用镜像。

   请参阅启用 *MIRROR* 配置参数 在第278页。

2. 将数据库服务器恢复联机。
3. 为镜像块分配磁盘空间。

   只要在下一步中指定镜像块时磁盘空间可用，您就可以随时分配此磁盘空间。镜像块必须与相应的主块在不同的磁盘上。请参阅为镜像数据分配磁盘空间 在第278页。

4. 选择您想要镜像的数据库空间、BLOB 空间或智能大对象空间，并为该存储空间中每个主块指定镜像块路径名称和偏移量。

   镜像过程在执行这一步后开启。对于所有要镜像的存储空间重复此步骤。请参阅使用镜像 在第279页。

启用 MIRROR 配置参数

可以设置 MIRROR 配置参数以启用 （或禁用） 镜像。

启用镜像将启动镜像任务所需的数据库服务器功能。然而，当您启用镜像时，您并未启动镜像过程。直到您为数据库空间、 BLOB 空间或智能大对象空间创建镜像块，镜像才会实际开始。请参阅使用镜像 在第279页。

如果您计划为根数据库空间创建镜像，将其作为初始化的一部分，那么在初始化数据库服务器时启用镜像；否则，保持镜像禁用。如果您稍后决定对存储空间制作镜像，您可以更改 MIRROR 配置参数的值。

要启用数据库服务器的镜像，必须将 onconfig 中的 MIRROR 参数设为 1。MIRROR 的缺省值为 0，表示禁用镜像。

如果不使用镜像，就不要把 MIRROR 参数设为 1 。

要更改 MIRROR 的值，您可以在数据库服务器处于联机模式时用文本编辑器编辑 onconfig 文件。在改变 onconfig 文件之后，将数据库服务器转为脱机，然后转为静默以使更改生效。

为镜像数据分配磁盘空间

在创建镜像块之前，必须为此目的分配磁盘空间。您可以为镜像块分配原始磁盘空间或熟文件空间。有关分配磁盘空间的说明，请参阅分配磁盘空间 在第169页。

始终在与相应主块不同的磁盘上（理想情况下，使用不同控制器）为镜像块分配磁盘空间。可以使用此设置在主块所在磁盘脱机时访问镜像块，反之亦然。

链接块 (UNIX™)

使用 UNIX™ 的链接（ln）命令将镜像块的实际文件或原始设备链接到镜像路径名称。如果磁盘发生故障，您可以将新文件或原始设备链接到该路径名，而不必实际更换故障磁盘后再将块恢复联机。

在磁盘故障后将块重新链接到设备

在 UNIX™上，如果实际镜像文件或原始设备所在磁盘脱机，您可将块重新链接到不同磁盘上的文件或原始设备。这样可以在故障磁盘重新联机之前恢复镜像块。以下示例显示了可用于重链接的典型 UNIX™ 命令。

原始设置包括链接到实际的原始磁盘设备的主块根和镜像根块，如下所示：

```
ln -l
```

```
lrwxrwxrwx 1 informix 10 May 3 13:38 /dev/root@->/dev/rxy0h
lrwxrwxrwx 1 informix 10 May 3 13:40 /dev/mirror_root@->/dev/rsd2b
```

假设原始设备 /dev/rsd2b 所在的磁盘已脱机。您可使用 rm 命令来删除相应的符号链接，如下所示：

```
rm /dev/mirror_root
```

现在您可将镜像块路径名重新链接到正在运行磁盘上的原始磁盘设备，并继续恢复块，如下所示：

```
ln -s /dev/rab0a /dev/mirror_root
```

使用镜像

当您为数据库空间、BLOB 空间或智能大对象空间中的每个主块创建镜像块时，镜像开始。

当您创建镜像块时，数据库服务器将数据从主块复制到镜像块。当此过程完成，数据库服务器开始镜像数据。如果主块包含逻辑日志文件，数据库服务器不会立即复制数据，而是一直等到您执行 0 级备份后。有关此行为的说明，请参阅创建镜像块 在第276页。

> 重要： 对于整个数据库空间、BLOB 空间或智能大对象空间，您必须始终启动镜像。数据库服务器不允许您在数据库空间、BLOB 空间或智能大对象空间中选择特定块进行镜像。您必须为空间中的每个块创建镜像块。

当执行以下操作时，启动对存储空间的镜像 ：

- 在系统初始化期间创建镜像根数据库空间
- 将数据库空间状态从未镜像更改为已镜像
- 创建镜像的数据库空间、BLOB 空间或智能大对象空间

这些操作均需要您为存储空间中的现有块创建镜像块。

初始化期间为根数据库空间制作镜像

如果您在初始化数据库服务器时启用镜像，您还可以为根块指定镜像路径名和偏移量。当初始化服务器时，数据库服务器会创建镜像块。但因根块包含逻辑日志文件，所以直至您执行 0 级备份镜像才会实际开始。

要指定根镜像路径名和偏移量，请在启动数据库服务器之前，在 onconfig 文件中设置 MIRRORPATH 和 MIRROROFFSET 的值。

如果您没提供镜像路径名和偏移量，但想要启动对根数据库空间进行镜像，那么您必须在数据库服务器初始化后，更改根数据库空间的镜像状态。

更改镜像状态

您可以对镜像块状态进行以下两种更改：

- 将镜像块从联机更改为脱机
- 将镜像块从脱机更改为恢复

仅当块是镜像对的一部分时才可以使块脱机或复原。您可以使主块或镜像块脱机，只要该对中的另一块处于状态。

有关如何确定块状态的信息，请参阅监视磁盘使用 在第199页。

管理镜像过程

可以使用 onspaces 实用程序管理镜像过程。

有关 onspaces 语法的完整说明，请参阅《*SinoDB*® 管理员参考》 中的 *onspaces* 实用程序 。

启动未镜像存储空间的镜像

您可以随时准备为数据库空间、BLOB 空间或智能大对象空间制作镜像。但直到您执行 0 级备份后镜像才会开始。

使用 onspaces 启动未镜像的数据库空间的镜像

可以使用 onspaces 实用程序来启动对数据库空间、BLOB 空间或智能大对象空间的镜像 。例如，下面的 onspaces 命令启动对数据库空间 db_project（包含 data1 和 data2 两个块）的镜像：

```
onspaces -m db_project\
-p /dev/data1 -o 0 -m /dev/mirror_data1 0\
-p /dev/data2 -o 5000 -m /dev/mirror_data2 5000
```

下面的示例显示如何启动名为 sp1 的数据库空间的镜像。您可以在命令中或文件中指定主路径、主偏移、镜像路径和镜像偏移量。

```
onspaces -m sp1 -f mirfile
```

mirfile 文件包含以下行：

```
/ix/9.3/sp1 0 /ix/9.2/sp1mir 0
```

在此行中，/ix/9.3/sp1 是主路径，0 是主偏移量，/ix/9.3/sp1mir 是镜像路径，0 是镜像偏移量。

启动新存储空间的镜像

您还可在创建新的数据库空间、BLOB 空间或智能大对象空间时启动镜像。

使用 onspaces 启动新空间的镜像

可以使用 onspaces 实用程序来创建镜像数据库空间。例如，以下命令创建带有初始块 /dev/chunk1 和镜像块 /dev/mirror_chk1 的 数据库空间 db_acct。

```
onspaces -c -d db_acct -p /dev/chunk1 -o 0 -s 2500 -m /dev/mirror_chk1 0
```

另一种启动镜像的方法是选择 Index by Utility >onspaces -m。

添加镜像块

如果您向已镜像的数据库空间、BLOB 空间或智能大对象空间添加块，您必须也添加相应的镜像块。

使用 onspaces 添加镜像块

可以使用 onspaces 实用程序向数据库空间、BLOB 空间、或智能大对象空间添加主块及其镜像块 。以下示例将块 chunk2 添加到 db_acct 数据库空间。由于数据库空间已镜像，所以也要添加镜像块 mirror_chk2。

```
onspaces -a db_acct -p /dev/chunk2 -o 5000 -s 2500 -m /dev/mirror_chk2 5000
```

使镜像块脱机

当镜像块脱机时，数据库服务器不能向其中写入也不能从其中读取。您可能将镜像块脱机，然后重新链接到不同设备。(请参阅在磁盘故障后将块重新链接到设备 在第278页。)

使镜像块脱机与结束镜像不同。结束整个数据库空间的镜像将使数据库服务器删除该数据库空间 的所有镜像块。

使用 onspaces 使镜像块脱机

可以使用 onspaces 实用程序使块脱机。以下例子使一个块（数据库空间 db_acct 的一部分）脱机：

```
onspaces -s db_acct -p /dev/mirror_chk1 -o 0 -D
```

恢复镜像块

要开始对联机块中的数据制作镜像，您必须恢复脱机块。

使用 onspaces 恢复镜像块

可以使用 onspaces -s 实用程序来恢复脱机块。例如，要恢复路径名为 /dev/mirror_chk1 且偏移量为 0 KB 的块，可以发出以下命令：

```
onspaces -s db_acct -p /dev/mirror_chk1 -o 0 -O
```

结束镜像

当结束数据库空间、BLOB 空间或智能大对象空间的镜像时，数据库服务器会立即释放此空间的镜像块。这些块立即可用于重新分配给其他储存空间。

仅 UNIX™ 系统中用户 informix 和 root，或 Windows™ 系统中 Informix-Admin 组的成员可以结束镜像。

如果数据库空间中的任一主块脱机，那么您无法结束镜像。结束镜像时系统可以处于联机模式。

使用 onspaces 结束镜像

可以使用 onspaces 实用程序结束镜像。例如，要结束根数据库空间的镜像，可输入以下命令：

```
onspaces -r rootdbs
```

另一种结束镜像的方法是选择 Index by Utility >onspaces -r。

一致性检查

SinoDB® 数据库服务器用于检测硬件或操作系统错误所导致的数据库服务器故障或问题。通过在其许多关键功能中执行 数据库服务器通过在其多个关键函数上执行 断言 来检测问题。断言是一种一致性检查，验证页、结构或其他实体的内容与其他假定是否一致。

若任一检查发现内容有误，数据库服务器将报告断言失败，并在数据库服务器消息日志中写入文本以描述失败的检查。数据库服务器还会在另外的文件中收集进一步的诊断信息，该文件可能对 SinoDB® 技术支持人员有用。

这些主题提供了一致性检查措施和处理不一致的方法的概述。

相关链接

数据库服务器维护任务 在第30页

执行定期的一致性检查

要从一致性检查获取最大利益并确保数据库空间备份的完整性，您必须定期执行以下操作：

- 验证所有数据和数据库服务器开销信息是一致的。
- 检查消息日志中一致性验证的断言失败。
- 在一致性验证后创建 0 级数据库空间备份。

以下主题描述了每个操作。

验证一致性

由于检查所需要时间与此检查导致的可能争用，因此请将此检查调度为在活动最少的时后进行。您必须在创建 0 级备份前执行此检查。

执行下表中显示的命令以作为一致性检查的一部分。

表 39: 检查数据一致性

验证类型	命令
系统目录表	oncheck -cc
数据	oncheck -cD ***dbname***
扩展数据块	oncheck -ce
索引	oncheck -cI ***dbname***
保留页	oncheck -cr
逻辑日志和保留页	oncheck -cR
元数据和智能大对象	oncheck -cs

您可以在数据库服务器处于联机模式时，运行这些命令。有关每个指令如何在检查对象时锁定它们以及那些用户可执行确认的信息，请参阅《*SinoDB*® 管理员参考》 中的 oncheck。

In most cases, if one or more of these validation procedures detects an error, the solution is to restore the database from a dbspace backup. However, the source of the error might also be your hardware or operating system.

大多数情况中，如果一个或多个验证过程检测出错误，那么解决方案是从数据库空间备份复原数据库。但是，错误也可能来自于硬件或操作系统。

验证系统目录表

要验证系统目录表，使用 oncheck -cc 命令。

每个数据库有自己的系统目录，其中包含有关数据库表、列、索引、视图、约束、存储过程及权限的信息。

如果验证完成时显示警告，该警告的唯一用途是提醒您找不到特定类型的记录。这些警告并非指示数据、系统目录或甚至数据库设计有任何问题。警告仅指示不存在任何表的同义词；即，系统目录没有 syssyntable 表中的记录。例如：如果为没有为任何表定义同义词的数据库验证系统目录表，则可能会显示以下警告：

```
WARNING: No syssyntable records found.
```

但是，如果您在验证系统目录表时接收到错误消息，那么情况就完全不同了。请立即与 SinoDB® 技术支持联系。

验证数据页

要验证数据页，使用 oncheck -cD 命令。

如果数据页验证检测到错误，请尝试从指定表中卸载数据、删除该表、重新创建该表，并重新装载数据。有关装载和卸载数据的信息，请参阅《*SinoDB*® 迁移指南》 。如果此过程失败，请从存储空间备份执行数据复原。

验证扩展数据块

要验证每个数据库的扩展数据块，使用 oncheck -ce 命令。

扩展数据块决不可重叠。如果此命令检测到错误，请从存储空间备份执行数据复原。

验证索引
如果索引损坏，数据库服务器在查询中将不能使用索引。

可以使用 oncheck -cI 命令来验证数据库中每个表的索引。

此外，调度程序任务 bad_index_alert 将查找已被服务器标记为损坏的索引。该任务在夜间运行。此任务找到的每个已损坏索引，都会在 sysadmin:ph_alert 表中建立一个条目。

如果索引已损坏，请将其删除，然后重新创建。

验证逻辑日志

要验证逻辑日志和保留页，使用 oncheck -cR 命令。

验证保留页

要验证保留页，使用 oncheck -cr 命令。

保留页是位于根数据库空间初始块开始处的页。这些页包含主数据库服务器开销信息。如果此命令检测到错误，请从存储空间备份中执行数据复原。

此命令可能提供警告。在大多数情况下，这些警告是让您注意您已经知道的情况。

验证元数据

对每个数据库运行 oncheck -cs 以验证数据库中所有智能大对象的元数据。必要时，从数据库空间备份复原数据。

监视数据不一致性

如果一致性检查代码在数据库服务器运行期间检测到不一致，将向数据库服务器消息日志报告断言失败。（请参阅《*SinoDB*® 管理员参考》 中的消息日志主题 。）

消息日志和转储文件中的断言失败

以下显示断言失败在消息日志中采用的格式：

```
Assert Failed: 简短说明是什么失败
   Who: 当时正在运行的用户/会话/线程的描述
   Result: 受影响的数据库服务器实体的状态
   Action: 数据库服务器管理员应该采取的操作
   See Also: 包含其他诊断信息的文件
```

See Also: 行包含以下一个或多个文件名称：

- af.xxx
- shmem.xxx
- gcore.xxx
- gcore.xxx
- */path name*/core

在所有情况下，xxx 对与单个线程的断言失败相关联的所有文件而言是一个公共的十六进制数。af.xxx、shmem.xxx 和 gcore.xxx 文件位于 ONCONFIG 的参数 DUMPDIR 所指定的目录下。

af.xxx 文件包含发送到消息日志的断言失败消息的副本，以及当前相关的结构和数据缓冲区的内容。

仅当 ONCONFIG 的参数 DUMPSHMEM 设置为 1 或 2，shmem.xxx 文件才会包含数据库服务器在断言失败时共享内存的完整副本。

> 仅限 UNIX: 在 UNIX™ 上，仅当 ONCONFIG 的参数 DUMPGCORE 设置为 1 且操作系统支持 gcore 实用程序，gcore.xxx 才会包含当时运行线程的数据库服务器虚拟进程的核心转储。仅当 ONCONFIG 的参数 DUMPCORE 设置为 1，core 才会包含当时运行线程的数据库服务器虚拟进程的核心转储。 core 文件的路径名是上次调用数据库服务器的目录。

验证表和表空间数据

要验证表和表空间数据，请在数据库或表上使用 oncheck -cD 命令。

大多数一般断言失败消息后面跟随着其他信息，通常包含检测到错误的表空间。如果此检查确认存在不一致情况，请从表中卸载数据、删除该表、重新创建该表，并重新装载数据。除此之外，无需执行任何其他操作。

在许多情况下，当断言失败时数据库服务器会立即停止。但是，当失败看起来是特定于某个表或较小的实体时，数据库服务器会继续运行。

当断言是由于数据库服务器代表用户访问的数据页上的不一致而失败时，错误还会发送到应用程序进程。SQL 错误取决于正在进行的操作。但是，ISAM 错误几乎始终是 -105 或 -172，如下所示：

```
-105 ISAM error: bad isam file format
-172 ISAM error: Unexpected internal error
```

有关消息的目的和内容的其他详细信息，请参阅《*SinoDB*® 管理员参考》中有关消息日志消息的主题。

保留一致的 0 级备份

在您执行验证一致性 在第282页中描述的检查且没有发现错误后，创建 0 级备份。保留此存储空间备份和所有后续的逻辑日志备份带，直到您完成下一次一致性检查。在每次 0 级备份之前执行一致性检查。如果不这样做，那么至少保留从存储空间备份复原所需的所有磁带，该备份是在验证数据库服务器一致后立即创建的。

处理损坏

本节描述了数据库服务器系统损坏的几种特征，以及数据库服务器或作为管理员的您为解决问题可采取的操作。数据库的损坏可能是硬件问题、操作系统问题或一些未知数据库服务器问题的结果。损坏可能影响数据或数据库服务器开销信息。

查找损坏的徵兆

您可以通过几种不同方式法查找有关损坏的信息。

数据库服务器通过以下方式警告用户和管理员可能有损坏：

- 向应用程序报告的错误消息说明找不到页、表或数据库。如果操作是由于数据或开销信息中的不一致而失败，那么以下错误之一将始终返回应用程序：

  ```
  -105 ISAM error: bad isam file format
  -172 ISAM error: Unexpected internal error
  ```
- 断言失败报告会写入数据库服务器消息日志。它们指示包含可帮助确定问题来源的其他诊断信息的文件。请参阅验证一致性 在第282页。
- oncheck 实用程序返回错误。
- ph_alert 表显示了有关已损坏索引的信息。

修复索引损坏

在第一次指示损坏时，运行 oncheck -cI 命令以确定损坏是否存在索引中。

如果您在数据库服务器处于联机模式时检查索引，oncheck 会检测损坏但不会提示您进行修复。如果损坏存在，您可在处于联机模式时使用 SQL 语句删除和重新创建索引（数据库服务器会锁定表和索引）。如果您在数据库服务器处于静默模式时运行 oncheck -cI 并检测到损坏，将提示您确认实用程序是否尝试修复该损坏。

修复块上的 I/O 错误

如果在数据库服务器运行期间发生 I/O 错误，那么发生错误的块的状态将更改为脱机。

如果块为脱机，那么 onstat -d 将显示主块的块状态为 PD-，而镜像块的块状态为 MD-。有关 onstat -d 输出的示例，请参阅《*SinoDB*® 管理员参考》。

此外，消息日志列出了带有出错位置和建议的解决方案的消息。列出的解决方案是可能的解决方法但不一定能够纠正问题。

如果脱机块已镜像，那么数据库服务器将继续使用镜像块进行操作。使用操作系统实用程序来确定脱机块的问题并纠正该问题。然后，您必须指示数据库服务器恢复镜像块数据。

有关恢复镜像块的信息，请参阅恢复镜像块 在第281页。

如果脱机块未镜像且包含逻辑日志文件、物理日志或根数据空间，那么数据库服务器会立即启动停止操作。否则，数据库服务器可继续运行，但将无法写入或读取该脱机或该脱机块所在数据空间中的任何其他块。您必须采取步骤以确定 I/O 错误发生的原因、纠正问题，并从备份复原数据库空间。

如果当某块标记为脱机而将数据库服务器变为脱机模式时，只要该标记为脱机的块不包含关键数据（逻辑日志文件、物理日志或根数据空间），那么您可以重新启动数据库服务器。

收集诊断信息

某些 ONCONFIG 参数会影响数据库服务器收集诊断信息的方式。由于断言失败通常为不可预见问题的指示，所以无论何时发生都要通知 SinoDB® 技术支持。所采集的诊断信息旨在供 SinoDB® 技术人员使用。有关 af.*xxx* 文件和共享内核的内容和使用不再进一步说明。

为确定触发断言失败的问题原因，那么您不能删除诊断信息（直到 SinoDB® 技术支持指示您可以这样做），这一点是极为重要。af.xxx 文件通常包含解决问题所需的信息。

一些 ONCONFIG 参数指示数据库服务器无论何时检测到断言失败或数据库服务器进入结束序列，都保留诊断信息：

- DUMPDIR
- DUMPSHMEM
- DUMPCNT
- DUMPCORE
- DUMPGCORE

有关配置参数的更多信息，请参阅《*SinoDB*® 管理员参考》。

您可决定是否设置这些参数。诊断输出会使用大量的磁盘空间。（具体内容取决于环境变量设置和您的操作系统。）输出元素包含共享内存的副本和核心转储。

> 提示： 核心转储是指在断言失败时，内存中的进程映像。在某些系统上，核心转储包含共享内存的副本。核心转储仅在此类情况下才是有用。

对于有磁盘空间约束的数据库服务器管理员可能更愿意编写脚本，以检测指定目录中诊断输出的存在，并将该输出文件发送到磁带。此方法可以保存诊断信息，并使磁盘空间使用最小化。

禁用 I/O 错误

SinoDB® 将禁用 I/O 错误分为两类：破坏性与非破坏性。当包含数据库的磁盘以某种方式损坏时，禁用 I/O 错误是破坏性的。该类型的事件会威胁数据的完整性，并且数据库服务器会将块和数据空间标记为脱机。数据库服务器会禁止访问已损坏磁盘，直至您修复或更换磁盘并执行物理和逻辑复原。

当错误不会威胁到数据的完整性时，禁用 I/O 错误是非破坏性的。当某人意外地断开电缆连接时、当您不小心删除您设置指向块的符号链接时或当磁盘控制器受损时，就会发生非破坏性的错误。

在数据库服务器将某 I/O 错误视为禁用 I/O 错误之前，该错误必须符合两个条件。首先，错误必须是在数据库服务器尝试在具有至少以下特征之一的块上执行操作时发生：

- 块没有镜像。
- 与有问题块成对的主块或镜像块是处于脱机状态。

其次，错误必须在数据库服务器尝试执行下列操作之一失败时发生：

- 在块上搜索、读取或写入
- 打开块
- 验证有关首次使用的页的块信息是否有效

  数据库服务器在打开一个块后立即执行此验证作为稳定性检查。

可以防止数据库服务器在您调查禁用 I/O 错误时将数据库空间标记为脱机。如果发现是小问题（如电缆松动），您可将数据库服务器变为脱机，然后再变为联机而无需从备份复原受影响的数据库空间。如果发现是严重的问题（如磁盘损坏），您可使用 onmode -O 将受影响的数据库空间标记为脱机并继续处理。

监视数据库服务器是否有禁用 I/O 错误

数据库服务器用两种方式通知您禁用 I/O 错误：

- 消息日志
- 事件警报

用于监视禁用 I/O 错误的消息日志

当发生禁用 I/O 错误时，数据库服务器会向消息日志发送一条消息。

消息为：

```
Assert Failed: Chunk {chunk-number} is being taken OFFLINE.
Who: Description of user/session/thread running at the time
Result: State of the affected database server entity
Action: What action the database server administrator should take
See Also: DUMPDIR/af.uniqid containing more diagnostics
```

如下表所述，结果和操作取决于 ONDBSPACEDOWN 的当前设置。

ONDBSPACEDOWN 设置	结果	操作
0	禁用数据库空间 {*space_name*} 。	恢复数据库空间{*space_name*}。
	禁用 BLOB 空间 {*space_name*} 。	恢复 BLOB 空间{*space_name*}。
1	数据库服务器必须停止。	关机并重新启动数据库服务器。
2	数据库服务器在下一个检查点时阻塞。	使用 onmode -k 关机， 或使用 onmode -O 覆盖。

ONDBSPACEDOWN 的值对临时数据库空间没有影响。对于临时数据库空间，无论如何设置 ONDBSPACEDOWN，数据库服务器将继续处理。如果临时数据库空间需要修复，可将其删除并重新创建。

有关解释数据库服务器发送至消息日志的消息的更多信息，请参阅《*SinoDB®* 管理员参考》中有关消息日志消息的主题。

用于监视禁用 I/O 错误的事件警报

当数据库空间引发禁用 I/O 错误时，数据库服务器会将特定值 作为参数传递给事件警报可执行文件。

事件警报值：

严重性

4（应急）

类

5

类消息

Dbspace is disabled: 'dbspace-name'

特定消息

Chunk {chunk-number} is being taken OFFLINE.

事件标识

5001

如果您想要数据库服务器使用事件警报通知您有关禁用 I/O 错误，那么您可编写脚本以便数据库服务器检测到禁用 I/O 错误时可执行该脚本。有关如何设置所编写的可执行文件的信息，请参阅《*SinoDB*® 管理员参考》中有关事件警报的附录和配置参数的主题。

无坏扇区映射

SinoDB® 依赖于您的主机操作系统来进行坏扇区映射。数据库服务器从系统调用中接收到的失败返回代码得知坏扇区和坏磁道。当发生这种情况时，数据库服务器将尝试重新访问若干次，以确定情况属实。如果确认了情况，数据库服务器会将尝试读取或写入的块标记为脱机。

数据库服务器无法采取任何操作来识别坏的柱面、磁道或扇区位置，因为唯一可用信息是尝试 I/O 操作的块中的字节位移。

如果数据库服务器在未镜像的块检测到 I/O 错误，它会将该块标记为脱机。如果脱机块包含逻辑日志文件、物理日志或根数据库空间，数据库服务器将会立即启动停止操作。否则，数据库服务器可继续操作，但直到其数据库空间复原之前应用程序将无法访问该脱机块。

第 V 部分

高可用性和可扩展性

成功的生产环境要求数据库系统始终可用、将计划的中断（如有的话）降到最低，并可随着业务需要的变化而快速轻松地扩展。

在计划和非计划的中断期间，企业必须提供对数据库资源的持续访问。计划的中断包括软件或硬件的定期维护。非计划的中断是不可预期的系统故障，例如断电、断网、硬件故障，操作系统或其他软件错误。发生灾难时，例如地震或海啸，可能出现大范围的系统故障。

企业想避免系统中服务器过载，以确保数据可用性，并防止服务拒绝攻击。

企业还想在业务增长时、季节性业务高峰期和月末或年末时，快速轻松地扩展系统。

具有以下一项或多项功能的系统可轻松应对中断，并可提高数据可用性：

冗余
: 系统维护辅助服务器的能力。辅助服务器为主服务器的副本，且在主服务器发生故障时接管主服务器。

故障转移
: 系统将所有工作负载从故障服务器转移到另一台服务器上的能力。

工作负载均衡
: 系统自动将客户机请求转向工作负载能力最高的服务器的能力。

可扩展性
: 系统利用其他资源（例如数据库服务器、处理器、内存或磁盘空间）的能力。

对系统维护的影响最小化
: 快速轻松维护所有服务器的能力，以便尽可能降低用户应用程序受到的影响的能力。

相关链接

功能配置 在第28页
高可用性集群中的 *XA* 在第423页

高可用性和可扩展性策略

可定制 SinoDB® 数据库软件以创建适当的高可用性和可扩展性解决方案，以满足业务目标和环境。

要确定为获得高可用性和可扩展性而定制您的数据库系统的最佳方式，您必须确定有助于实现业务目标的策略。您可以使用适当的 SinoDB® 技术和组件来支持这些策略。

支持高可用性和可扩展性的组件

可定制 SinoDB® 数据库软件以创建具有以下功能的系统：提供不间断服务、最小化维护和停机时间、自动将客户端连接请求重定向到最合适的数据库服务器，以及跨硬件分配处理和存储。

高可用性集群

高可用性集群是由通过安全网络连接的单个主服务器和一个或多个辅助服务器组成；主服务器包含了数据的主副本，而辅助服务器是与主服务器同步或有访问主服务器数据的权限。事务在主服务器提交之后，将会发送到辅助服务器，所以数据库数据是可靠的。如果主服务器故障，辅助服务器会变成主服务器，并将客户连接请求重定向到新的主服务器上。

高可用性集群服务器的配置了相同的硬件和软件。高可用性集群服务器彼此可以是很邻近的或在地理上相距很远，而应用程序可以安全地连接到任何一台集群服务器。

有三种类型的辅助服务器：

- 共享磁盘（SD）辅助服务器，其与主服务器共享磁盘空间。
- 高可用性数据复制（HDR）辅助服务器，其维护整个主服务器同步或异步更新的副本，并且如果主服务器发生故障，则可快速接替提供客户端访问。
- 远程独立（RS）辅助服务器，其维护整个主服务器异步更新的副本，并且可在灾难恢复场景中充当远程备份服务器。

Enterprise Replication

使用 Enterprise Replication，可通过复制事务在多个服务器上维护完整或部分数据副本。数据通过从逻辑日志捕获的事务进行异步复制。如果远程数据库服务器或网络发生故障，本地数据库服务器将继续服务本地用户并存储要复制的事务直到远程服务器可用。在每个数据库服务器上，Enterprise Replication 读取逻辑日志以捕获本地发起的事务，然后将这些事务复制到 Enterprise Replication 域中的其他数据库服务器。

分片集群

SinoDB® 可以跨多个数据库服务器水平分割（分片）一个表或集合。被分片数据所在的数据库服务器的集合称为分片集群，集合中的每个数据库服务器是一个分片服务器。在分片集群中分布行或文件会减少每个分片服务器上相关索引的大小，并在硬件中分配性能。随着数据库增长，您可通过添加多个数据库服务器来扩展。水平分割也成为分片。

根据您指定的分片规则，插入到一个分片服务器的行或文件可以复制或发送到其他分片服务器。在分片服务器上执行的查询可以选择分片集群中其他分片服务器上的数据。当数据基于指定特定分段特征的复制密钥分片时，查询可跳过不包含相关数据的分片服务器。这种查询优化是数据分片的另一个好处。

连接管理

连接管理器是一个实用程序，用于监视高可用性集群、Enterprise Replication 域、网格和服务器集中数据库服务器的工作负载和状态，并使用重定向策略将客户端连接请求发给最合适的数据库服务器。连接管理器还可作为代理服务器来处理客户端/服务器的通信并规避有关防火墙的连接问题。

连接管理器可以控制高可用性集群的故障转移，如果原来的主服务器故障，将自动把辅助服务器升级成主服务器。

如果发生部分网络故障，连接管理器可以优先考虑应用程序服务器与高可用性集群的主服务器之间的连接，以更好地定义故障转移。

网格

网格是互相连接的复制服务器，其中 SQL 语句可以从一个服务器传播到其他所有服务器。网格提供了一种更简单的方法来管理大量服务器、更新数据库架构、运行存储过程和用户定义的例程，以及管理复制。

相关链接

通过连接管理器进行连接管理 在第340页
《*SinoDB JSON* 兼容性指南》 : *JSON* 数据分片
《*SinoDB Enterprise Replication* 指南》: 分片集群建立
《*SinoDB Enterprise Replication* 指南》: 网格建立与管理
《*SinoDB Enterprise Replication* 指南》: *Enterprise Replication* 建立与管理

数据复制的优点

数据复制的优势并非不需成本。数据复制显然需要更多存储，并且更新复制数据比更新单个对象花费更多处理时间。

您可以在客户端应用程序的逻辑中通过明确指定何处数据需要更新来实现数据复制。但是这种数据复制的方法成本很高，容易出错且难以维护。相反，数据复制概念通常与复制透明性相结合。复制透明性构建在数据库服务器（而不是客户端应用程序）中，以自动处理定位和维护数据复制的详细信息。

集群与镜像

集群和镜像是提高容错性的透明方式。

镜像 在第274页中所述的镜像是单个数据库服务器维护不同磁盘上特定数据库空间副本的机制。该机制保护镜像数据库空间中的数据免受磁盘故障影响，因为数据库服务器自动更新两个磁盘上的数据，并且在一个数据库空间故障时自动使用另一磁盘。

或者，集群在完全独立的数据库服务器上复制数据库服务器管理的所有数据，而不仅仅是复制特定的数据库空间。因为集群涉及两个独立的数据库服务器，所以它不仅保护这些数据库服务器管理的数据免受磁盘故障影响，还保护这些数据免受所有类型的数据库服务器故障（包括计算机故障或整个站点灾难性的故障）的影响。

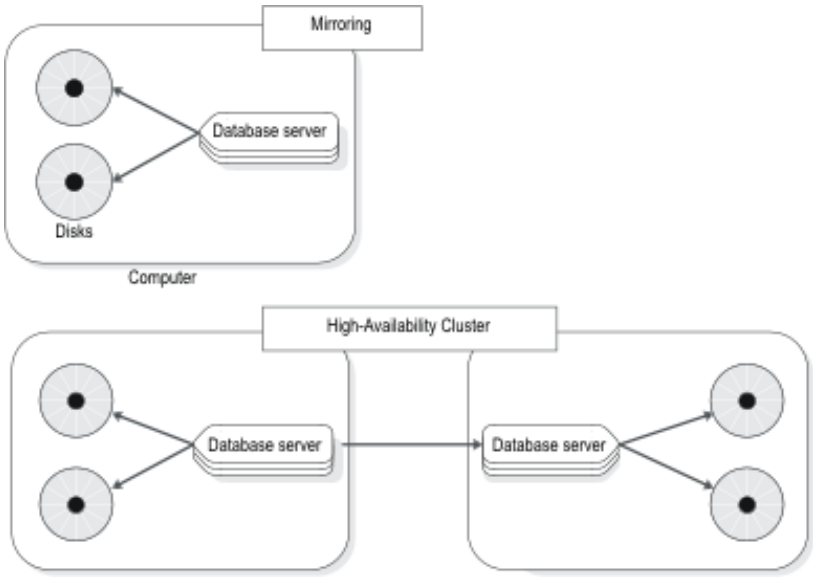

图 52: 镜像与集群的比较

集群与两阶段提交

在多阶段提交协议 在第421页中详细描述的两阶段提交协议，确保跨多个数据库服务器的事务统一提交或回滚。

理论上，您可以配置具有相同数据的两个数据库服务器，利用两阶段提交来复制数据，并在其中一格数据库服务器上定义触发器以将更新复制到另一数据库服务器。但是，这种实现在不同故障场景中有很多同步问题。并且，分布式事务的性能比集群的性能要差。

集群中复制数据的类型

高可用性集群可复制数据库空间和智能大对象空间中的数据，但不会复制 BLOB 空间中的数据。

所有内置和扩展数据类型都会复制到辅助服务器。必须记录用户定义类型（UDT）且这些类型是位于单个数据库服务器中。如果行外数据储存在智能大对象空间或同一数据库服务器上的不同表中，那么可复制带有行外数据的数据类型。对于要复制存储在智能大对象空间中的数据，必须记录智能大对象空间。

不复制储存在操作系统文件、持续的外部文件或与用户定义例程关联的内存对象中的数据。

用户定义类型、用户定义例程和 DataBlade® 模块有特殊的安装和注册需求。有关指示信息，请参阅数据初始复制的工作原理 在第311页。

主要和辅助数据库服务器

当配置一组数据库服务器集以使用数据复制时，其中一个数据库服务器称为主数据库服务器，其余称为辅助数据库服务器。（在这样的情况下，不使用数据复制的数据库服务器称为标准数据库服务器。）辅助服务器可包括 SD 辅助服务器、RS 辅助服务器和 HDR 辅助服务器的任何组合。

如下图所示，主数据库服务器管理的数据发生更改后，辅助数据库服务器会动态随之更新。

图 53: 数据复制配置中的主数据库服务器和辅助数据库服务器

如下图所示，如果其中一个数据库服务器故障，那么可以将使用该数据库服务器的客户端重定位到该对中成为主服务器的另一 数据库服务器。

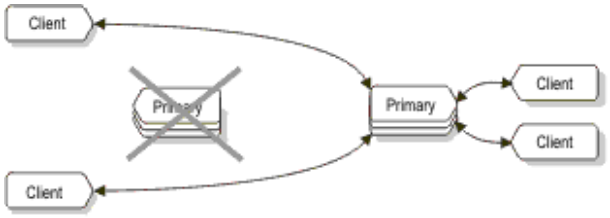

图 54: 故障后数据复制配置中的数据库服务器和客户端

透明扩展和工作负载平衡策略

SinoDB® 服务器可轻松地扩展并可动态平衡工作负载以确保对资源的最优使用。

SinoDB® 可实现以下目标：

- 定期增加容量 在第291页
- 地理上分散处理和增加容量 在第292页
- 均衡工作负载以优化资源的使用 在第293页

定期增加容量

如果业务环境遇到高峰期，那么可能需要定期增加容量。可以通过添加远程独立辅助服务器来增加容量。那种类型的辅助服务器维护数据的完整副本，包含通过安全网络连接从主服务器异步传输来的更新。如果数据量很大且，难以制作多份个副本，那么可以使用共享磁盘辅助服务器，而不是远程独立辅助服务器。如果只需为想报告（只读）工作负载增加容量，那么可以使用高可用性数据复制（HDR）辅助服务器。

表 40: 共享磁盘辅助服务器的可扩展性

优点	潜在缺点
• 高可用性。辅助服务器与主服务器共享磁盘。	• 无故障转移。辅助服务器可能与主服务器配置在同一计算机硬件上运行。

优点	潜在缺点
	• 无数据冗余。辅助服务器没有维护数据的副本。（使用 SAN 设备进行磁盘存储） • 主服务器和辅助服务器需要相同的硬件、操作系统和数据库服务器产品版本。

以下为使用 SD 辅助服务器的原因：

- 增强报告容量

  多台辅助服务器使您可以在不影响主服务器的情况下卸载报告功能。
- 服务器故障备份

  如果主服务器故障，SD 辅助服务器可以快速且轻易地提升为主服务器。例如，如果您使用能提供足够且可靠磁盘存储 SAN（存储区域网络）设备，但是却担心服务器故障，那么 SD 辅助服务器可提供可靠的备份。

表 41: 远程独立辅助服务器的可扩展性

优点	潜在缺点
• 数据冗余。辅助服务器维护数据的副本。 • 故障转移。辅助服务器在地理上远离主服务器，例如在另一建筑物、另一个城市或另一个国家。 • 无需更改应用程序。在遇到服务器故障时，可自动切换客户端与主服务器或辅助服务器的连接。	• 主服务器和辅助服务器需要相同的硬件、操作系统和数据库服务器产品版本。

地理上分散处理和增加容量

对于在多处都有办公室的企业，可能希望本地服务器处理本地的请求，而不是依赖单个集中式的服务器。在这种情况下，您可以设置 Enterprise Replication 服务器网络。远程数据库服务器中断是可以容忍的。如果数据库服务器或网络发生故障，本地数据库服务器将继续服务本地用户。本地数据库服务器将复制的事务存储在持久存储中，直到远程服务器可用。本地服务器上的 Enterprise Replication 通过读取逻辑日志捕获要复制的事务、存储事务，然后以可靠方式将每个事务作为复制数据传输到目标服务器。

您还可使用 Enterprise Replication 设置分片集群，其中数据在多个服务器之间水平切分（分片）。随着容量需求增加，您可以向分片集群添加额外的数据库服务器，以提升总容量。

表 42: Enterprise Replication 的可扩展性

优点	潜在缺点
• 服务器可位于另一建筑物、另一个城市或另一个国家。 • 服务器可在不同硬件上。 • 服务器可在不同操作系统上运行。 • 服务器可以运行不同版本的数据库服务器产品。 • 可复制少量数据（基于日志的异步复制 ） • 使用多个连接管理器进行自动客户端重定向，并可以添加共享磁盘辅助服务器来协助复制服务器。 • 虽容量需求的增加，您可向已建立的分片集群添加额外的分片服务器。	• 可能存在冲突。 • 事务可能失败。如果发生，您必须修复不一致的数据。

以下为使用 RS 辅助服务器的原因：

- 增加服务器可用性

  通过维护可用于增加可用性的一个或多个 RS 辅助服务器来提供进一步的保证。
- 地理位置上的远程备份支持

  为了应对最糟的灾害恢复场景，通常需要在距离站点一段距离的位置设置辅助服务器。RS 辅助服务器是理想的远程备份解决方法。如果辅助服务器位于 WAN（广域网）上，那么主 HDR 和辅助 HDR 之间的高级别协作可能会导致性能问题。使主服务器和辅助服务器保持相对密切关系可简化维护并将对性能的影响降至最低。
- 提高报告性能

  多台辅助服务器可在不影响主服务器的情况下卸载报告功能。此外，RS 辅助服务器配置使得分离报告需求和 HA 需求更容易，从而为两种环境提供更好的解决方案。
- 在不稳定网络上的可用性

  如果检查点同步到达，那么较慢或不稳定的网络环境可导致主服务器和辅助服务器都延迟。RS 辅助服务器配置使用全双工网络，并且不需要协调。如果主服务器和 RS 辅助服务器之间的网络性能未达最佳状态，那么 RS 服务器是一种极具吸引力的解决方案。

均衡工作负载以优化资源的使用

在创建或修改服务级别协议 SLA 时，可配置工作负载均衡。SinoDB® 从集群中的每个服务器收集信息并自动将客户端应用程序连接到活动量最少的服务器。

可在集群内创建特定于某些应用程序类型的组，如用于联机事务处理 (OLTP) 或仓库的应用程序。应用程序可以选择连接到特定组，以优化每种类型查询的性能。

相关链接

《SinoDB JSON 兼容性指南》 : JSON 数据分片

《SinoDB Enterprise Replication 指南》: 分片集群建立

《SinoDB Enterprise Replication 指南》: Enterprise Replication 建立与管理

高可用性策略

可配置 SinoDB®，以在各种业务情况中最大化可用性。

目标	策略	优点	潜在缺点
服务器故障保护	使用与主服务器共享磁盘空间的辅助服务器。（共享磁盘辅助服务器）	• 可用性极高。此辅助服务器和主服务器具有相同的数据访问权。如果主服务器故障，辅助服务器可以快速接管。 • 由于此辅助服务器和主服务器具有相同的数据的访问权，所以数据库始终处于同步状态。 • 无需更改应用程序。服务器故障时，可自动切换客户端与主服务器或辅助服务器的连接。	• 此辅助服务器和主服务器在同一台计算机上。 • 无数据冗余。此辅助服务器没有维护数据的副本。（ 使用 SAN 设备进行磁盘存储 ） • 主服务器和辅助服务器需要相同的硬件、操作系统和数据库服务器产品版本。 • 辅助服务器的硬件必须能够处理与主服务器相同的负载。如果辅助服务器太小，它可能会影响主服务器的性能。

目标	策略	优点	潜在缺点
站点故障保护	• 使用维护数据库服务器和数据副本的辅助服务器。（ 高可用性数据复制服务器 ） • （ 也可使用 RSS 和 ER ）	• 可用性极高。应用程序如果无法连接主服务器，可快速访问此服务器。 • 数据同步复制。 • 提升可扩展性 • 无需更改应用程序	• 位于主服务器本地 • 需要数据的精确副本（包括表和数据库模式）。 • 主服务器和辅助服务器需要相同的硬件、操作系统和数据库服务器产品版本。
多站点故障保护	• 使用在地理上远离主服务器且从主服务器异步更新的辅助服务器。（ 远程独立辅助服务器 ） • （ 也可以使用 ER ）	• 可用性极高。应用程序如果无法连接到主服务器，可以快速访问此服务器。 • 数据异步复制。 • 提升可扩展性 • 无需更改应用程序	
地理分散处理与站点故障保护	ER 与 HDR	ER 和 ER 的备份	需要多个连接管理器

高可用性集群配置

这些主题描述如何为 SinoDB® 规划、配置、启动和监视高可用性集群，以及如何在介质发生故障后复原数据。如果您计划使用高可用性集群，请先阅读本节中的所有主题。如果您计划使用 SinoDB® Enterprise Replication，请参阅《*SinoDB® Enterprise Replication* 指南》。

高可用性和可扩展性 在第288页说明了高可用性集群是什么、其工作原理，以及如何为集群环境设计客户端应用程序。

高可用性集群规划

在开始设置计算机和数据库服务器以使用高可用性集群之前，您可能想要做一些初步规划。以下列表包含了要执行的规划任务：

- 选择并获取适当的硬件。
- 如果使用多个数据库服务器来存储要复制的数据，那么需要迁移并重新分布数据使其可以由单一数据库服务器管理。
- 确保要复制的所有数据库都使用事务日志记录。要开启事务日志记录，请参阅管理数据库日志记录模式 在第239页。
- 开发使用复制对中数据库服务器的客户端应用程序。有关设计注意事项的说明，请参阅数据复制客户端的重定向与连接 在第409页和设计数据复制组客户端 在第324页。
- 为首次启动 HDR 创建调度。
- 为主数据库服务器设计存储空间和逻辑日志备份调度。
- 为如何处理其中一个数据库服务器故障以及如何在故障后重新启动 HDR 生成一个计划，请阅读数据复制客户端的重定向与连接 在第409页。

配置集群

通过确认硬件、操作系统、和数据库需求来配置集群。还可以设置安全协议和安全连接。

要将系统配置为高可用性集群，必须执行以下操作：

- 满足硬件和操作系统要求。
- 满足数据库和数据要求。
- 满足数据库服务器配置要求。
- 配置连接

本节将说明每个主题。

连接管理器还支持 Distributed Relational Database Architecture™ (DRDA®) 连接。有关更多信息，请参阅*Distributed Relational Database Architecture (DRDA)* 通信 在第66页。

集群的硬件和操作系统需求。

要使高可用性集群运行，硬件必须满足一些需求。

硬件必须满足以下需求：

- 主服务器和辅助服务器必须可运行相同的 SinoDB® 可执行映像，即使他们的硬件或操作系统不相同。例如，使用具有不同的 Linux™ 32 位操作系统的服务器，因为这些操作系统都可运行相同的 SinoDB® 可执行映像。在这种情况下，不能添加 Linux™ 64 位操作系统的服务器， 因为该操作系统需要不同的 SinoDB® 可执行映像。查看机器说明文件：可使用相同机器说明文件中列为支持的硬件和操作系统的任意组合。
- 运行主数据库服务器和辅数据库服务器的硬件必须支持网络能力。
- 分配给主数据服务器和辅助数据服务器的数据库空间的磁盘空间量必须相等。磁盘空间的类型是无关的；在两个数据库服务器上可使用任何原始或熟空间组合。
- 在每台计算机上的块必须具有相同的路径名。允许在 UNIX™ 平台使用符号链接，但不允许在 Windows™ 平台。

集群的数据库和数据需求

要使高可用性集群运行，数据库和数据必须满足一些需求。

您的数据库及数据必须满足以下需求：

- 所有数据必须记录。

  要复制的所有数据库必须将事务日志记录打开。

  这要求很重要，因为辅助数据库服务器使用来自主数据库服务器的逻辑日志记录更新其所管理的数据。如果主数据库服务器管理的数据库不使用日志记录，那么对这些数据库的更新就不会生成日志记录，这样辅助数据库服务器将无法更新复制的数据。日志记录可以是已缓冲日志记录，也可以是未缓冲日志记录。

  如果必须在启动 HDR 之前打开事务日志记录，请参阅使用 *ontape* 打开事务日志纪录 在第241页。
- 数据必须位于数据库空间或智能大对象空间。

  如果主数据库服务器具有存在 BLOB 空间的简单大对象，那么对 BLOB 空间中数据的修改将不会作为正常 HDR 处理的一部份进行复制。但会复制数据库空间中的简单大对象数据。

  复制存在智能大对象空间中的智能大对象。必须记录智能大对象空间。复制用户定义类型（UDT），除非它们有存储在操作系统文件中的行外数据。如果行外数据存储在智能大对象空间或同一数据库服务器上的不同表中，那么复制带有行外数据的数据类型。
- 辅助数据库服务器不得使用磁盘压缩。

  如果使用 SinoDB® 磁盘压缩功能，源表中压缩的数据在目标表中也是处于压缩状态。在 HDR 辅助服务器、RS 辅助服务器，或 SD 辅助服务器上无法执行压缩操作，因为 HDR 目标服务器必须具有与源服务器相同的数据及物理布局。

集群的数据库服务器配置需求

要使高可用性集群服务器对运行，必须完整配置每一个数据库服务器。有关配置数据库服务器的信息，请参阅数据库服务器配置和管理概述 在第24页。然后您可以使用该配置的相关方面来配置该数据库服务器对中另一个数据库服务器。有关配置参数的更多信息，请参阅《*SinoDB*® 管理员参考》。

这些主题描述了一下有关集群数据库服务器对的配置注意事项：

数据库服务器版本

主数据服务器和辅助数据服务器的服务器版本必须是相同的。

存储空间和块的配置

在主服务器和辅助服务器上，数据库空间的数量、块的数量、它们的大小、路径名、以及它们的偏移量都必须相同。此外，如果 HDR 辅助服务器用于创建活动报告，那么配置必须至少包含一个临时数据库空间。请参阅 用于排序的临时数据库空间和临时表的使用 在第325页。

仅限 UNIX™:

如在 *UNIX* 上分配原始磁盘空间 在第170页中所述，必须对块路径名使用符号链接。

> 重要： 如果块路径名不使用符号链接，就不能轻易更改块的路径名。有关更多信息，请参阅重命名块 在第326页。

以下 ONCONFIG 参数在每个数据库服务器上必须具有相同值：

- ROOTNAME
- ROOTOFFSET
- ROOTPATH
- ROOTSIZE

HDR 环境中的非缺省页大小

数据库空间的页大小和缓冲池规范自动从主数据库服务器传播至辅数据库服务器。虽然主数据库服务器和辅助数据库服务器都必须具有相同的缓冲池，但是缓冲池中的缓冲区数量无需匹配。

镜像

无需将两个数据库服务器上的 MIRROR 参数设置为相同值；您可以启用一个数据库服务器上的镜像，而禁用另一个数据库服务器上的镜像。但是，如果您为主数据库服务器的根块指定了镜像块，那么您必须也为辅数据库服务器上的根块指定镜像块。因此，以下 ONCONFIG 参数在两个数据库服务器上必须具有相同值：

- MIRROROFFSET
- MIRRORPATH

物理日志配置

两个数据库服务器的物理日志必须是相同的。以下 ONCONFIG 参数在每个数据库服务器上都必须具有相同值：

- PHYSBUFF
- PHYSFILE

数据库空间及逻辑日志磁带备份设备

可以为主数据服务器和辅数据服务器指定不同的磁带设备。

如果使用 ON-Bar，那么将两个数据库服务器上 ON-Bar 配置参数设置为相同值。有关 ON-Bar 参数的信息，请参阅《*SinoDB*® 备份和还原指南》。

如果使用 ontape，那么存储空间和逻辑日志备份设备的磁带大小和磁带块大小必须相同。以下 ONCONFIG 参数在每个数据库服务器上都必须具有相同值：

- LTAPEBLK

- LTAPESIZE
- TAPEBLK
- TAPESIZE

要用足磁带的物理容量，请将 LTAPESIZE 和 TAPESIZE 设置为 0。

逻辑日志配置

复制所有日志记录到辅助服务器。您必须在两个数据库服务器上配置相同数目的逻辑日志文件和相同的逻辑日志大小。以下 ONCONFIG 参数在每个数据库服务器上都必须具有相同值：

- LOGBUFF
- LOGFILES
- LOGSIZE
- DYNAMIC_LOGS

数据库服务器记录逻辑日志文件的添加。主服务器上动态添加的逻辑日志文件将自动复制到辅助服务器上。尽管辅助服务器上的 DYNAMIC_LOGS 值无效用，请保持 DYNAMIC_LOGS 与主服务器上的值同步，以防其角色切换。

高可用性集群配置参数

在每个参与高可用性集群的数据库服务器上设置 HA_ALIAS 配置参数。

在高可用性集群中的每一个辅助服务器上设置 TEMPTAB_NOLOG 配置参数。高可用性集群中的辅助服务器必须禁用临时表上的逻辑日志记录。

将 HDR 对中的两个数据库服务器的以下配置参数设置成相同值：

- DRAUTO
- DRINTERVAL
- DRTIMEOUT

集群事务协调
可配置高可用性集群，使其当客户端会话发出提交时，服务器将拦阻该会话，直到该事务在该会话中、辅助服务器上或整个集群上应用完成。可设置 CLUSTER_TXN_SCOPE 配置参数或运行 SET ENVIRONMENT CLUSTER_TXN_SCOPE 语句来配置此行为。

在不同的高可用性集群服务器或在高可用性集群服务器的不同会话中执行的多步骤客户端操作可能会因异步日志处理而失败。如果客户端应用程序将数据载入集群服务器，并在数据复制到辅助服务器之前试图在辅助集群服务器上处理相同的数据时，该操作将失败。客户端事务必须先应用到辅助服务器上，才可进一步处理其数据。

集群事务协调使客户端应用程序在事务提交返回之前等待事务的集群范围或辅助服务器应用。这个过程可防止操作失败并确保多步骤过程的步骤依序发生。

集群事务的不同范围：

- SESSION：当客户端会话发送提交时，数据库服务器将拦阻该会话直到事务提交结果返回该会话。在控制权返回该会话之后，同一数据库服务器或集群中其他数据库服务器上的其他会话可能还不知晓该事务的提交及其结果。
- SERVER：当用客户端会话发出提交时，数据库服务器将拦阻该会话，直到该事务在客户端会话发出提交的数据库服务器上完成应用。在该数据库服务器上的其他会话知晓该事务的提交及其结果。集群中其他数据库服务器的会话可能不知晓该事务的提交及其结果。此为高可用性集群服务器的缺省行为。
- CLUSTER：当客户端会话发出提交时，数据库服务器将拦阻该会话，直到该事务在高可用性集群的所有数据库服务器（使用 DELAY_APPLY 或 STOP_APPLY 的 RS 辅助服务器除外）上完成应用。在高可用性集群的所有数据库服务器（使用 DELAY_APPLY 或 STOP_APPLY 的 RS 辅助服务器除外）上的其他会话都知晓该事务的提交及其结果。

将 CLUSTER_TXN_SCOPE 值设为 CLUSTER 不会改变 DRINTERVAL 配置参数值指定的行为。当客户端应用程序在主服务器上提交事务时，主服务器会按照 DRINTERVAL 配置参数指定的最大间隔时间发送逻辑日志缓冲区

到 HDR 辅助服务器。主服务器将逻辑日志缓冲区发送到 HDR 辅助服务器之后，控制权将返回会话，但是该会话要等到该事务在所有集群服务器上完成应用后才会收到提交完成。

相关链接

《SinoDB 管理员参考》: CLUSTER_TXN_SCOPE 配置参数

《SinoDB 管理员参考》: DELAY_APPLY 配置参数

《SinoDB 管理员参考》: STOP_APPLY 配置参数

《SinoDB SQL 指南: 语法》: SET ENVIRONMENT 语句

为高可用性集群配置安全连接

要使高可用性集群运行，数据库服务器彼此之间必须建立可信连接。通过在每一个集群服务器上使用可信主机文件并在 sqlhosts 文件条目中包含连接安全选项来确保集群服务器之间的连接。

sqlhosts 文件中指定的安全端口仅用于数据库服务器之间的通信。客户端应用程序不可连接到安全端口。

为复制配置可信环境，请在每个集群服务器上执行以下操作：

1. 在运行集群服务器的每个主机上编辑 sqlhosts 文件：
   a) 为该主机上运行的每个集群服务器添加一个条目，并包含 s=6 选项。
   b) 为集群中的每一个其他集群服务器添加一个条目。添加的条目中不要包含 s=6 选项。
2. 设置 sqlhosts 文件或注册表的 nettype 字段，并将 NETTYPE 配置参数设置为网络协议（例如，ontlitcp 或 onsoctcp），以便两台不同计算机上的数据库服务器可以相互通信。
   不要指定非网络协议，例如 onipcshm、onipcstr 或 onipcnmp。
3. 指定可信主机信息。
   可使用以下方式指定可信主机信息：
   - 在 $INFORMIXDIR/etc 目录中创建 hosts.equiv 文件，并手动添加条目到该文件中。
   - 在 $INFORMIXDIR/etc 目录中创建可信主机文件，并手动添加条目到该文件中。必须将 REMOTE_SERVER_CFG 配置参数设置为可信主机文件名，并将 S6_USE_REMOTE_SERVER_CFG 配置参数设置为 1。
   - 运行带有 cdr add trustedhost 参数的 admin() 或 task() 函数，并指定可信主机信息。通过 cdr add trustedhost 指定的可信主机信息将传播到高可用性集群中所有服务器。如果您已经在高可用性集群或 Enterprise Replication 域中的任一数据库服务器上手动输入了可信主机信息，那么不要运行此函数。
4. 创建服务器别名，用于运行实用程序和客户端应用程序。
   例如，将 INFORMIXSERVER 环境变量设置为该别名，用于运行实用程序（例如， onstat 和 ontape ）和客户端应用程序（例如，DB-Access）。

相关链接

sqlhosts 连接信息 在第49页

可信主机信息 在第43页

《SinoDB 管理员参考》: S6_USE_REMOTE_SERVER_CFG 配置参数

《SinoDB 管理员参考》: REMOTE_SERVER_CFG 配置参数

sqlhosts 文件和 *SQLHOSTS* 注册表键选项 在第51页

更改客户端连接信息 在第410页

首次启动 HDR

完成 HDR 配置后，即可准备启动 HDR。本主题描述了启动 HDR 的必需步骤。

希望在两个数据库服务器 ServerA 和 ServerB 上启动 HDR。以下步骤描述了启动 HDR 的过程（将 ServerA 作为主数据库服务器，ServerB 作为辅助数据库服务器）。下表列出执行每个步骤所需的命令和发送至消息日志的消息。您可使用 ontape 或 ON-Bar 执行备份和复原。您必须在整个过程中使用相同的实用程序。

重要： 即使您使用 ON-Bar 执行备份和复原，两个数据库服务器上仍然需要 ontape 实用程序来执行备份和应用逻辑日志。不要参与 HDR 集群环境的数据库服务器上移除 ontape 实用程序。

还可使用外部备份和复原来设置 HDR。请参阅《*SinoDB*® 备份和还原指南》，以获取如何执行外部备份和复原。请参阅使用 *ontape STDIO* 功能来减少设置时间 在第301页，以获取直接从 HDR 主服务器设置 HDR 辅助服务器的最快方法。

启动 HDR 的操作：

1. 在两个数据库服务器上安装用户定义的类型、用户定义例程和 DataBlade® 模块，然后仅在 ServerA 上注册它们。
2. 创建 ServerA 的 0 级备份。
3. 使用 onmode -d 命令将 ServerA 类型设置为主类型，并指示相关联的辅助数据库服务器的名称（在本例中为 ServerB）。

   当发出 onmode -d 命令时，数据库服务器会尝试与 HDR 对中的另一个数据库服务器建立连接并启动 HDR 操作。仅当该对中的另一个数据库服务器已经设置为正确类型时，建立连接的尝试才会成功。

   此时，ServerB 未联机，且类型也未设置为辅助类型，因此不会建立 HDR 连接。
4. 从在步骤 *1* 在第299页 中创建的 0 级备份执行对 ServerB 的物理复原。不要执行逻辑复原。

   如果您使用：

   - ON-Bar，请使用 onbar -r -p 命令来执行物理复原。
   - ON-Bar 并正在执行外部复原，请使用 onbar -r -p -e 命令来执行物理复原。
   - ontape，请使用 ontape -p 选项。您不能使用 ontape -r 选项，因为它同时执行物理复原和逻辑复原。

     注： 如果主服务器和辅助服务器位于两台不同机器上，那么您必须将主服务器的物理复原放置在辅助服务器上。物理复原的位置是由 onconfig 参数 TAPE 定义。在运行 ontape -p 前，必须在辅助服务器上设置 *IFX_ONTAPE_FILE_PREFIX* 变量。
   - ontape 并正在执行外部复原，请使用 ontape -p -e 命令来执行物理复原。
5. 使用 onmode -d 命令将 ServerB 的类型设置为辅助类型，并指示相关联的主数据库服务器。

   ServerB 尝试与主数据库服务器（ServerA）建立 HDR 连接，并启动操作。必须成功建立连接。

   在 HDR 开始前，辅助数据库服务器使用自步骤 2 以来写入主数据库服务器的逻辑日志记录来执行逻辑恢复。如果所有这些逻辑日志记录仍然位于主数据库服务器磁盘上，那么主数据库服务器将直接通过网络把这些记录发送到辅助数据库服务器上，并自动执行逻辑恢复。

   如果已经备份并释放主数据库服务器上的逻辑日志文件，那么这些文件中的记录将不再在磁盘上。辅助数据库服务器会提示您从磁带中恢复这些文件。在这种情况下， 必须执行步骤 *6* 在第299页。

   重要： 在同一会话期间，必须完成步骤 *4* 在第299页 到 *5* 在第299页 。如果在步骤 *4* 在第299页 之后必须关闭并重新启动辅助数据库服务器，那么您必须重新执行步骤 *4* 在第299页。
6. 如果写入主数据库服务器的逻辑日志记录不再在主磁盘上，那么辅助数据库服务器会提示您从磁带备份恢复这些文件。

   如果辅助数据库服务器必须通过网络读取备份的逻辑日志文件，那么将辅助数据库服务器的磁带设备参数设置为主数据库服务器所在计算机上的设备或与主数据库服务器位于位于相同位置的设备。

   在恢复磁带上的所有逻辑日志文件之后，逻辑恢复将使用主数据库服务器磁盘上的逻辑日志文件完成。

下表说明了上述步骤，这样您可以清楚地确定在主服务器上执行的步骤和在辅助服务器上执行的步骤。该表还显示了在执行每个步骤之后写入日志文件的信息。

表 43: 首次启动 HDR 的步骤

步骤	在主服务器上 (ServerA)	在辅助服务器上 (ServerB)
1.	安装 UDR、UDT 和 DataBlade® 模块。	安装 UDR、UDT 和 DataBlade® 模块。

步骤	在主服务器上 (ServerA)	在辅助服务器上 (ServerB)
	注册 UDR、UDT 和 DataBlade® 模块。	

2.	ontape 命令：
	运行 ontape -s -L 0
	ON-Bar 命令
	：运行 onbar -b -L 0
	发送至消息日志的消息：
	`Level 0 archive started on rootdbs`
	`Archive on rootdbs completed`	

3.	onmode 命令
	onmode -d primary *sec_name*
	发送至消息日志的消息：
	`DR: new type = primary`
	`server name = sec_name`
	`DR: Trying to connect to secondary server`
	`DR: Cannot connect to secondary server`	

4.		ontape 命令：
		运行 ontape -p 或 ontape -p -e
		提示备份日志时回答 no 。
		ON-Bar 命令
		运行 onbar -r -p 或 onbar -r -p -e
		发送至消息日志的消息：
		`Sinoregal Database Server Initialized --`
		`Shared Memory Initialized`
		`Recovery Mode`
		`Physical restore of rootdbs started`
		`Physical restore of rootdbs completed`

5.		运行 onmode -d secondary *prim_name* 发送至消息日志的消息： `DR: new type = secondary, primary server name = prim_name` 如果自步骤 1 以来写入主数据库服务器的逻辑日志记录仍然位于主数据库服务器磁盘上，那么辅助数据库服务器将读取这些记录以执行逻辑恢复。（否则，必须执行步骤 6 ）。
	发送至消息日志的消息： `DR: Primary server connected`	发送至消息日志的消息： `DR: Trying to connect to primary server`

步骤	在主服务器上 (ServerA)	在辅助服务器上 (ServerB)
	DR: Primary server operational	DR: Secondary server connected DR: Failure recovery from disk in progress *n* recovery worker threads will be started Logical Recovery Started Start Logical Recovery - Start Log *n*, End Log ? Starting Log Position - *n 0xnnnnn* DR: Secondary server operational
6.		ontape 命令： ontape -l ON-Bar 命令 onbar -r -l
	发送至消息日志的消息： DR: Primary server connected DR: Primary server operational	发送至消息日志的消息： DR: Secondary server connected DR: Failure recovery from disk in progress *n* recovery worker threads will be started Logical Recovery Started Start Logical Recovery - Start Log *n*, End Log ? Starting Log Position - *n 0xnnnnn* DR: Secondary server operational

相关链接

高可用性群集的备份和复原 在第335页
关键数据损坏后恢复集群 在第415页

使用 ontape STDIO 功能来减少设置时间

通过使用 ontape STDIO 功能可以显著提高设置 HDR 的速度。使用此功能，ontape 会在备份期间将数据写入 shell 的标准输出，然后在复原期间从标准输入读取该数据。使用远程命令解释器（如 rsh 或 ssh）将 STDIO 备份和同时发生的 STDIO 复原组合在一个管道中，以允许使用一个命令行来执行 HDR （或 RSS）辅助服务器的初始设置。由于不对磁带或磁盘执行写入或读取操作，所以可以节省存储空间，而且无需等待备份完成即可开始复原。

请参阅 《*SinoDB*® 备份和还原指南》 以获取有关使用 STDIO 值的详细信息。

不管使用何种备份实用程序 (ontape 或 ON-Bar)，都可使用这种使用 ontape 设置 HDR 的方法。

> 重要： 以这种方法使用 STDIO 时，不会在任何位置保存可用于执行复原的永久备份。如果在源（备份）端上使用 -F （假）选项，那么不会将备份记录在数据库服务器的保留页中。此外，还会禁止任何交互式对话，并且不会显示任何提示或询问。您还必须确保管道的远程端为远程 SinoDB® 实例挑选合适的环境。脚本不可产生备份数据以外的任何输出，因为备份数据由复原过程读取（例如，不启用跟踪）。

下表中的步骤必须由用户 informix 执行，脚本必须是可执行的，如果不是用完整路径调用该脚本，那么该脚本必须位于主目录中。如果您需要通过网络进行安全的数据传输，那么使用 ssh 而非 rsh 。

1. 在辅助服务器上，安装UDR、UDT 和 DataBlade® 模块。
2. 在主服务器上，安装UDR、UDT 和 DataBlade® 模块。
3. 在主服务器上，注册 UDR、UDT 和 DataBlade® 模块。

4. 在主服务器上，执行以下命令：

```
onmode -d primary secondary_server
```

5. 在主服务器上，执行以下命令：

```
ontape -s -L 0 -t STDIO -F | rsh secondary_host ontape_HDR_restore.ksh
```

6. 在辅助服务器上，执行以下命令：

```
onmode -d secondary primary_server
```

在上表中，辅助服务器上的 ontape_HDR_restore.ksh 脚本必须包含以下命令：

```
#!/bin/ksh
# first get the proper Sinoregal DS environment set
. hdr_sec.env
# redirecting stdout and stderr required since otherwise command might never return
ontape -p -t STDIO > /dev/null 2>&1
```

以下步骤显示如何从辅助服务器设置 HDR：

1. 在辅助服务器上，安装UDR、UDT 和 DataBlade® 模块。
2. 在主服务器上，安装UDR、UDT 和 DataBlade® 模块。
3. 在主服务器上，注册 UDR、UDT 和 DataBlade® 模块。
4. 在主服务器上，执行以下命令：

```
onmode -d primary secondary_server
```

5. 在辅助服务器上，执行以下命令：

```
rsh primary_host ontape_HDR_backup.ksh | ontape -p -t STDIO
```

6. 在辅助服务器上，执行以下命令：

```
onmode -d secondary primary_server
```

在上表中，主服务器上的 ontape_HDR_backup.ksh 脚本必须包含以下命令：

```
#!/bin/ksh
# first get the proper Sinoregal DS environment set
. hdr_pri.env
ontape -s -L 0 -F -t STDIO
```

远程独立辅助服务器

这些主题概述了在高可用性集群环境中设置和配置远程独立（RS）辅助服务器。

比较 RS 辅助服务器和 HDR 辅助服务器

RS 辅助服务器在许多方面都与 HDR 辅助服务器相似。将日志发送到 RS 辅助服务器的方式与主服务器将日志发送到 HDR 辅助服务器的方式很相似。但是， RS 辅助服务器用于完全在异步通信框架内运行，因此对对主服务器的影响达到最小。不论是事务提交还是检查点在主服务器和 RS 辅助服务器之间均不同步。在主服务器上提交的任何事务不保证同时在 RS 辅助服务器上提交。

在高可用性集群中，HDR 辅助服务器的日志必须领先于任何 RS 辅助服务器的日志。如果 HDR 辅助服务器脱机，主服务器将继续把日志发送到 RS 辅助服务器。但是，当 HDR 辅助服务器恢复联机时，SinoDB® 会停止向 RS 辅助服务器发送日志，并优先向 HDR 辅助服务器发送日志，直到 HDR 辅助服务器的日志重播领先

于 RS 辅助服务器。HDR 辅助服务器日志的优先性是必要的，因为 HDR 辅助服务器是集群中故障转移的首选。如果故障转移发生时，RS 辅助服务器日志领先于 HDR 辅助服务器日志，那么 RS 辅助服务器无法与新的主服务器同步。

尽管 RS 辅助服务器与 HDR 辅助服务器相似，但有些功能是 HDR 辅助服务器支持的而 RS 辅助服务器不支持的：

- SYNC 模式
- DRAUTO 参数
- 同步检查点

对于高可用性集群中的 HDR、RSS 和 SDS 辅助服务器，必须将 TEMPTAB_NOLOG 配置参数设置为 1，以始终禁用临时表的逻辑日志记录。

索引页日志记录

必须启用索引页日志记录以使用 RS 辅助服务器。

相关链接

将主服务器数据复制到辅助服务器 在第311页

索引页日志记录的工作原理

创建索引时，索引页日志记录将所有页写入逻辑日志，以使高可用性环境中服务器之间的索引创建同步。

索引页日志记录将完整索引写入志文件中，然后将该日志文件异步地传输到辅助服务器。辅助服务器可以是 RS 辅助服务器或 HDR 辅助服务器。然后，将日志文件事务读入辅助服务器的数据库。在恢复期间，辅助服务器无需重新构建索引。对于 RS 辅助服务器，其允许立即访问主服务器上的索引，主服务器无需等待来自辅助服务器的确认。

可使用 ONCONFIG 的 LOG_INDEX_BUILDS 参数来控制索引日志记录。将 LOG_INDEX_BUILDS 参数设置为 1（启用），可在主服务器上创建索引并发送给辅助服务器。

启用或禁用索引日志记录

使用 LOG_INDEX_BUILDS 配置参数在数据库服务器启动时启用或禁用索引页日志记录。可通运行 onmode -wf LOG_INDEX_BUILDS=1（启用）或 0（禁用）来更改 onconfig 文件中 LOG_INDEX_BUILDS 的值。

当 RS 辅助服务器存在于高可用性环境中时，必须启用索引页日志记录。

查看索引页日志记录统计信息

可以使用 onstat 实用程序或系统监视接口（SMI）来查看是启用还是禁用索引页日志记录。统计信息也显示索引页日志记录启用或禁用的日期和时间。

要查看索引页日志记录的统计信息，请使用 onstat -g ipl 命令，或查询 sysipl 表。

有关 onstat -g ipl 输出的示例，请参阅《*SinoDB*® 管理员参考》中有关 onstat 实用程序的信息。

首次启动 RS 辅助服务器

在完成 RS 辅助服务器的硬件配置后，就可以随时启动 RS 辅助服务器并将其连接到主服务器。

假设您想要启动主服务器 和 RS 辅助服务器（ServerA 和 ServerB）。以下步骤描述了启动服务器的过程（以 ServerA 作为主数据库服务器，而 ServerB 作为 RS 辅助数据库服务器）。上表列出了执行每一步骤所需的命令。

此过程要求对主服务器进行备份，然后将备份复原到辅助服务器上。您可以使用 ontape 或 ON-Bar 执行备份和复原。您必须在整个过程中使用相同的实用程序。

> 重要： 即使使用 ON-Bar 执行备份和复原，两个数据库服务器仍然需要 ontape 实用程序来执行备份和应用逻辑日志。不要从参与 HDR 集群环境的数据库服务器移除 ontape 实用程序。

您也可以用于外部备份和复原的标准 ON-Bar 或 ontape 命令来设置 RS 辅助服务器。

启动带有 RS 辅助服务器的主服务器：

1. 在两个数据库服务器上均安装用户定义类型、用户定义例程和 DataBlade® 模块，然后仅在 ServerA 上注册它们。

   有关如何安装用户定义类型和用户定义例程的信息，请参阅《*SinoDB®* 用户自定义例程和数据类型开发者指南》。有关如何安装 DataBlade® 模块的信息，请参阅《*SinoDB® DataBlade®* 模块安装和注册指南》。
2. 激活主服务器上的索引页日志记录。
3. 在主服务器上记录 RS 辅助服务器的身份。首次建立主服务器与辅助服务器之间的连接时，可选密码在这两个服务器之间提供认证。
4. 创建 ServerA 的 0 级备份。
5. 从您在步骤 *4* 在第304页 中创建的 0 级备份执行对 ServerB 的物理复原。不要执行逻辑复原。

   使用适当的命令：

   - 使用 onbar -r -p 命令执行物理复原。
   - 使用 onbar -r -p -e 命令执行物理外部复原。
   - 使用 ontape -p 选项。（不要使用 ontape -r 选项因为它同时执行物理复原和逻辑复原。）
   - 使用 ontape -p -e 命令执行物理外部复原。
6. 使用 onmode -d RSS ServerA *password* 命令将 ServerB 的类型设置为 RS 辅助服务器，并指示相关联的主数据库服务器。在此示例中，ServerA 的 DBSERVERNAME 和 HA_ALIAS 配置参数均设置为 ServerA，而 ServerB 的 DBSERVERNAME 和 HA_ALIAS 配置参数均设置为 ServerB。

   通过使用数据库服务器的 HA_ALIAS 配置参数的值，ServerB 试图与主服务器（ServerA）建立连接并启动操作。必须成功建立连接。

   辅助服务器使用自步骤 *4* 在第304页 以来写入主服务器的逻辑日志记录执行逻辑恢复。如果所有这些逻辑日志记录仍然位于主数据库服务器磁盘上，那么主数据库服务器将通过网络直接把这些记录发送到 RS 辅助服务器，并自动执行逻辑恢复。

   如果您已经备份并释放了主数据库服务器上的逻辑日志文件，那么这些文件中的记录不再在磁盘上。辅助数据库服务器会提示您从磁带中恢复这些文件。在这种情况下，您必须执行步骤 *7* 在第304页。

   > 重要： 在同一会话中，您必须完成步骤 *5* 在第304页 到 *6* 在第304页。如果必须在步骤 *5* 在第304页 之后关闭并重新启动辅助数据库服务器，那么必须重新执行步骤 *5* 在第304页。
7. 如果写入主数据库服务器的逻辑日志记录已经不再在主磁盘上，辅助数据库服务器会提示您从磁带备份恢复这些文件。

   如果辅助数据库服务器必须通过网络读取备份的逻辑日志文件，那么将辅助数据库服务器上的磁带设备参数设置为正运行主服务器的计算机的设备，或设置为与主服务器处于相同位置的设备。

   在恢复磁带上所有逻辑日志文件之后，逻辑恢复将使用主数据库服务器磁盘上的逻辑日志文件来完成。

表 44: 首次启动带有 RS 辅助服务器的主服务器的步骤

步骤	在主服务器上	在 RS 辅助服务器上
1.	安装 UDR、UDT 和 DataBlade® 模块。 注册 UDR、UDT 和 DataBlade® 模块。	安装 UDR、UDT 和 DataBlade® 模块。
2.	onmode 命令 onmode -wf LOG_INDEX_BUILDS=1	
3.	onmode 命令 onmode -d add RSS *rss_ha_alias password*	
4.	ontape 命令 ontape -s -L 0 ON-Bar 命令	

步骤	在主服务器上	在 RS 辅助服务器上
	onbar -b -L 0	
5.		ontape 命令 ontape -p 或 ontape -p -e 当提示您备份日志时回答 no。 ON-Bar 命令 onbar -r -p 或 onbar -r -p -e
6.		onmode 命令 onmode -d RSS *primary_ha_alias password* 如果自步骤 1 以来写入主数据库服务器的所有逻辑日志记录仍然位于主数据库服务器上，那么辅助数据库服务器将读取这些记录以执行逻辑恢复。（否则，必须执行步骤 *7* 在第304页）。
7.		ontape 命令 ontape -l ON-Bar 命令 onbar -r -l 仅当数据库服务器提示您从磁带备份恢复逻辑日志文件时才需要执行此步骤。

相关链接

高可用性群集的备份和复原 在第335页

通过备用备份方法来减少设置时间

通过使用 ontape STDIO 功能，可显著提升设置辅助服务器的速度。请参阅使用 *ontape STDIO* 功能来减少设置时间 在第301页以获取更多信息。

请参阅《*SinoDB*® 备份和还原指南》以获取有关使用 STDIO 值的详细信息。

将脱机主服务器转换为RS 辅助服务器

将主服务器计划的或意外的故障移转到 RS 辅助服务器后，可以将旧的主服务器转换为 RS 辅助服务器。

例如，假设您有一个名为 srv1 的主服务器，此服务器已故障移转到一个名为 srv2 的 RS 辅助服务器。以下步骤显示如何将旧的主服务器转换为 RS 辅助服务器。

1. 在新的主服务器（srve2）上将旧的主服务器（srv1）注册成 RS 辅助服务器。

```
onmode -d add RSS srv1
```

2. 如果要将旧的主服务器转换为 RS 辅助服务器且该服务器处于脱机状态，那么请使用此章节所示备份和恢复命令初始化服务器：第一次启动一台 *RS* 辅助服务器。此外，可通过以下命令初始化旧的主服务器：

```
oninit -PHY
```

请参阅 *oninit* 实用程序以获得更多信息。

3. 使用以下命令将服务器转换为 RS 辅助服务器：

```
onmode -d RSS srv2
```

延迟应用日志记录

为了帮助灾难恢复方案，可以将 RS 辅助服务器配置为先等待指定的时间段，然后应用从主服务器接收的日志。

通过延迟日志文件的应用，您可以从 RS 辅助服务器的数据库复原来快速地从错误的数据库修改中恢复。还可以在指定时间停止 RS 辅助服务器上的日志应用。

例如：假定数据库管理员想要根据行的存在时间删除表中的某些行。表中的每行都有一个时间戳记，指示该行的创建时间。如果数据库管理员不小心将过滤器设置为错误日期，则可能会删除比预期更多的行。通过延迟日志文件的应用，这些行将仍然会存在 RS 辅助服务器上。这样，数据库管理员可以从辅助服务器上抽取这些行，并将其插入到主服务器中。

现在，假设数据库管理员需要通过重新命名来对模式进行更改，但数据库管理员输入了错误的命令并删除了表 orders，而不是将表名称改为 store_orders。如果将 RS 辅助服务器配置为延迟日志应用，则数据库管理员可以从辅助服务器来恢复 orders 表。

当配置了日志文件延迟应用时，在达到指定时间段之前，将不应用从主服务器发送的事务。从主服务器接收到的日志文件将在 RS 辅助服务器上的指定安全目录中暂存，然后在指定的时间段之后应用。延迟日志文件的应用有两种方法：

- 在指定时间间隔之后应用暂存的日志文件
- 在指定时间停止应用日志文件

可通过在 RS 辅助服务器的 onconfig 文件中设置配置参数来启用日志文件的延迟应用。在启用日志文件的延迟应用之前，必须通过 LOG_STAGING_DIR 配置参数来指定日志文件暂存的目录。在指定 LOG_STAGING_DIR 配置参数之后，可通过您编辑 onconfig 文件或动态使用 onmode -wf 命令来配置 DELAY_APPLY 或 STOP_APPLY 配置参数。

日志记录的存储位置

服务器在 LOG_STAGING_DIR 指定的目录中创建其他名为 ifmxlog_## 的目录，其中 ## 是 SERVERNUM 指定的实例。这些目录用于存储逻辑日志，也在 RS 辅助服务器恢复期间使用。如果需要恢复 RS 辅助服务器，并且主服务器上日志已覆盖，那么可使用 ifmxlog_## 中的日志来恢复该服务器。ifmxlog_## 内的文件不再需要时将清除。

延迟触发的条件

BEGIN WORK、COMMIT WORK 和 ROLLBACK WORK 日志记录中的时间值用于计算延迟或停止日志文件的应用时间。在将日志页递到恢复过程之前，计算这些时间值。

当发出 BEGIN WORK 语句时， BEGIN WORK 日志记录将直到事务执行第一次更新活动之后才写入；因此，在发出 BEGIN WORK 语句的时间和写入 BEGIN WORK 日志的时间之间存在延迟。

与辅助服务器更新交互

您必须考虑辅助服务器更新与日志文件延迟应用之间的交互。如果启用了更新，并且更新了辅助服务器，在到达 DELAY_APPLY 指定的时间之前将不会应用更新。但是，禁用辅助服务器更新也会禁用 Committed Read，这将保证检索行时，在表中提交每个检索的行。

要保持 Committed Read 的隔离级别，应考虑使用 UPDATABLE_SECONDARY 配置参数启用辅助服务器更新，但是从连接管理器服务级别协议列表中移除用于日志文件延迟应用的 RS 辅助服务器。此外，可考虑将 RS 辅助服务器移到新的 SLA。

请参阅辅助服务器上的数据库更新 在第331页和《*SinoDB*® 管理员参考》以获取更多的信息。

指定日志暂存目录

配置日志暂存目录，以指定 RS 辅助服务器的日志文件在应用于数据库之前的暂存位置。

在启用日志文件的延迟应用之前，必须指定来自主服务器的日志文件的暂存目录。未定义任何缺省暂存目录。服务器在 LOG_STAGING_DIR 指定的目录中创建其他名为 ifmxlog_## 的目录，其中 ## 是 SERVERNUM 指定的实例。这些目录用于存储逻辑日志，也在 RS 辅助服务器恢复期间使用。暂存的日志文件在不再需要

时将会自动移除。如果丢失了 LOG_STAGING_DIR 内的文件，而且主服务器覆盖了日志，那么必须重建 RS 辅助服务器。

必须确保 LOG_STAGING_DIR 指定的目录存在且安全。该目录的所有者必须是用户 informix，必须属于组 informix，并且不得具有公共读、写或执行许可权。如果启用了角色隔离，那么 LOG_STAGING_DIR 指定的目录必须属于拥有 $INFORMIXDIR/etc 的用户或组。如果 LOG_STAGING_DIR 指定的目录不安全，那么不能初始化该服务器。如果该目录不安全，以下消息将写入联机消息日志：

```
The log staging directory (directory_name) is not secure.
```

还必须确保磁盘有足够空间可存储来自主服务器的所有日志，并且确保目录中没有来自不再使用的早期实例的暂存日志。

要查看有关发送到 RS 辅助服务器设置的日志暂存目录的数据的信息，请在 RS 辅助服务器上运行 onstat -g rss verbose 命令。

如果暂存文件写入失败，RS 辅助服务器会发出事件警报 40007。

请参阅《*SinoDB*® 管理员参考》以获取更多信息。

设置 LOG_STAGING_DIR：

1. 确保存储日志的目录存在且安全。
2. 编辑 RS 辅助服务器的 onconfig 文件。
3. 按以下方法指定暂存目录：LOG_STAGING_DIR *directory_name*，其中*directory_name* 是存储日志的目录名称。
4. 重启服务器。

也可通过使用 onmode -wf 命令，在不重启服务器的情况下就能设置 LOG_STAGING_DIR 配置参数；但是，在执行该命令时，日志文件的延迟应用不得处于活动状态。

延迟应用 RS 辅助服务器上的日志记录
可在 RS 辅助服务器上延迟应用日志记录，以准备灾难恢复方案。

通过设置 DELAY_APPLY 配置参数来启用日志文件的延迟应用。可以手动编辑 onconfig 文件并重启服务器，或可以使用 onmode -wf 命令动态地更改该值。在设置 DELAY_APPLY 的值时，还必须设置 LOG_STAGING_DIR。如果配置了DELAY_APPLY 而未将 LOG_STAGING_DIR 设置为正确且安全的目录，那么无法初始化服务器。

使用数目和修饰符 来设置 DELAY_APPLY。数目最多包含三个数字，用于指示修饰符单位的数量。修饰符是以下项之一：

- D（或 d），代表天
- H（或 h）， 代表小时
- M（或 m），代表分钟
- S（或 s），代表秒

请参阅《*SinoDB*® 管理员参考》以获取更多信息。

要在 RS 辅助服务器上延迟 4 小时应用日志文件，请运行：

```
onmode  -wf DELAY_APPLY=4H
```

要延迟 1 天应用日志文件，请运行：

```
onmode -wf DELAY_APPLY=1D
```

要禁用日志文件的延迟应用，请运行：

```
onmode  -wf DELAY_APPLY=0
```

停止应用日志记录
可在 RS 辅助服务器上停止应用日志记录，以准备灾难恢复方案。

可通过设置 STOP_APPLY 配置参数来停止 RS 辅助服务器上的日志文件应用。可以手动编辑 onconfig 文件并重启服务器，或可以使用 onmode -wf 命令动态地更改该值。在设置 STOP_APPLY 的值时，还必须设置 LOG_STAGING_DIR。如果配置了STOP_APPLY 而未将 LOG_STAGING_DIR 设置为正确且安全的目录，那么无法初始化该服务器。

请参阅《*SinoDB*® 管理员参考》以获取更多信息。

要立即在 RS 辅助服务器上停止应用日志文件，请运行以下命令：

```
onmode  -wf STOP_APPLY=1
```

要在 2009 年 4 月 15 日晚上 11 点停止应用日志文件，请运行：

```
onmode  -wf STOP_APPLY=2009:04:15-23:00:00
```

要恢复正常应用日志文件，请运行：

```
onmode  -wf STOP_APPLY=0
```

远程独立辅助服务器的流量控制

流量控制提供了限制主服务器上的日志活动的方法，以便集群中的远程独立（RS）辅助服务器的事务处理不会落后太多。启用流量控制可确保位于忙碌或间歇性网络的 RS 辅助服务器上的日志保持最新。

在主服务器上设置 RSS_FLOW_CONTROL 配置参数以启用流量控制。集群中的所有 RS 辅助服务器均受主服务器的 RSS_FLOW_CONTROL 配置参数影响。当流量控制处于激活状态时，连接到主服务器的用户可能会感受到响应时间较慢。

日志始终按照接收的顺序发送到 RS 辅助服务器。

要检查 RS 辅助服务器的流量控制是否处于激活状态，请使用 onstat -g rss verbose 命令，并将 RSS flow control 的值和 Approximate Log Page Backlog 的值进行比较。如果 Approximate Log Page Backlog 的值比 RSS flow control 的第一个值高，那么流量控制是激活的。如果 Approximate Log Page Backlog 的值比 RSS flow control 的第二个值低，那么流量控制是禁用的。

相关链接

《SinoDB 管理员参考》: RSS_FLOW_CONTROL 配置参数
《SinoDB 管理员参考》: SDS_FLOW_CONTROL 配置参数

共享磁盘辅助服务器

这些主题概述了在高可用性环境中 SD（共享磁盘）辅助服务器的设置和配置。SD 辅助服务器选项可用于 SinoDB® 标准版本。

SD 辅助服务器

共享磁盘（SD）辅助服务器 参与高可用性集群配置。在这种配置中，主服务器和 SD 辅助服务器共享同一磁盘或磁盘阵列。

SD 辅助服务器不在自己的磁盘空间中维护物理数据库副本。而是与主服务器共享磁盘。

必须将 SD 辅助服务器配置为可访问允许并行访问的共享磁盘设备。不要配置使用操作系统缓冲（如 NFS 交叉安装文件系统）的 SD 辅助服务器。如果 SD 辅助服务器实例和主服务器实例位于一台机器上，那么两个服务器均可访问本地磁盘。如果 SD 辅助服务器和主服务器位于不同的物理机器上，那么必须将它们配置为可访问本地连接的共享磁盘设备，如 Veritas 或 GPFS™。

SD 辅助服务器可以与 HDR 辅助服务器、RS 辅助服务器和 Enterprise Replication 结合使用。

SD 辅助服务器可以快速地添加至高可用性环境，因为它们不需要单独的磁盘副本。由于 SD 辅助服务器共享主服务器的磁盘存储资源，所以建议您提供一些其它磁盘备份方式，如磁盘镜像，或者使用 RS 辅助服务器或 HDR 辅助服务器。

以下限制会影响属于共享磁盘辅助服务器的数据库服务器实例的提升：

- SD 辅助服务器不能提升为 RS 辅助服务器。
- SD 辅助服务器不能提升为可存在于主要高可用性环境之外的标准服务器。

SD 辅助服务器的磁盘需求

除磁盘需求外（与主服务器共享），硬件和软件的需求通常与 HDR 辅助服务器的需求是相同的（请参阅特定支持平台的机器说明 ）。此外，具有数据库服务器的计算机之间必须共享主磁盘系统。也就是说，从 SD 辅助服务器到数据库空间的路径和主服务器的数据库空间路径相同。请参阅配置集群 在第295页。

设置共享磁盘辅助服务器

通过配置主服务器、SD 辅助服务器和启动 SD 辅助服务器来设置共享磁盘系统。

相关链接

《*SinoDB* 管理员参考》：数据库配置参数

设置 SD 主服务器

设置主服务器：

1. 设置 SDS_TIMEOUT 配置参数， 指定主服务器将等待从 SD 辅助服务器发送的日志位置确认的时间量（以秒为单位）。
2. 执行以下命令来配置 SD 主服务器的别名：

```
onmode -d set SDS primary ha_alias
```

由 *ha_alias* 指定的服务器名称将成为共享磁盘环境的主服务器和 SD 辅助服务器的日志源。

设置 SD 辅助服务器

设置 SD 辅助服务器：

1. 在配置文件中设置以下配置参数：
   - HA_ALIAS：设置别名用于高可用性集群中服务器间通信。
   - SDS_ENABLE：在辅助服务器上设置为 1（启用）以启用共享磁盘环境。
   - SDS_PAGING：设置两个文件（用于存储可能需要在检查点之间清空的页）的路径。每个文件作为任意页大小的块的临时磁盘存储。
   - SDS_TEMPDBS：设置 SD 辅助服务器首次启动时动态创建的临时数据库空间。
   - SDS_LOGCHECK：设置主服务器与辅助服务器之间网络通信暂时不可用时延迟故障移转的秒数。
   - TEMPTAB_NOLOG：设为 1，以防止对临时表进行逻辑日誌记录。
   - UPDATABLE_SECONDARY：设置为正整数，以使得客户端应用程序可以在辅助服务器上执行更新、插入和删除操作。
2. 设置以下配置参数以与主服务器上那些参数匹配：
   - ROOTNAME
   - ROOTPATH
   - ROOTOFFSET
   - ROOTSIZE
   - PHYSFILE
   - LOGFILES
   - LOGSIZE

   除了DBSERVERALIASES、DBSERVERNAME 和 SERVERNUM以外，可设置其他配置参数以与主服务器上的那些参数匹配。

3. 在 sqlhosts 文件中添加标识主服务器的条目：

```
#dbservername   nettype   host   servicename   options
 name           protocol  name   port
```

4. 使用 oninit 命令启动 SD 辅助服务器。

   SD 辅助服务器以静默模式下的快速恢复处理任何打开的事务。在快速恢复期间 LGR 内存池的内存使用量会增加。

5. 检查辅助服务器上的 online.log 文件，确认已完成处理打开的事务并回到联机模式。
6. 允许客户端应用程序连接到 SD 辅助服务器。

获取 SD 辅助服务器统计信息

使用 onstat 实用程序或系统监视接口（SMI）表来查看 SD 辅助服务器统计信息。

使用 onstat -g sds 来查看 SD 辅助服务器统计信息。onstat 实用程序的输出取决于该程序是在主服务器还是辅助服务器上运行。

查询 syssrcsds 表可获取有关主服务器上共用磁盘统计信息的信息。

查询 systrgsds 表可获取有关辅助服务器上共用磁盘统计信息的信息。

有关 onstat 和 SMI 表的信息，请参阅《*SinoDB*® 管理员参考》。

将 SD 辅助服务器升级为主服务器

通过在 SD 辅助服务器上发出以下命令将 SD 辅助服务器转换为主服务器：

```
onmode -d set SDS primary <alias>
```

SD 辅助服务器不能转换为标准服务器。

将主服务器转换为标准服务器

通过在主服务器发出以下命令将主服务器转换为标准服务器，并断开与共用磁盘环境的连接：

```
onmode -d clear SDS primary <alias>
```

共享磁盘辅助服务器的流量控制

流量控制提供了限制主服务器上日志活动的方法，以便集群中的共享磁盘（SD）辅助服务器在事务处理上不会落后太多。

在主服务器上设置 SDS_FLOW_CONTROL 配置参数以启用流量控制。集群中的所有 SD 辅助服务器都会受主服务器的 SDS_FLOW_CONTROL 配置参数的设置影响。当流量控制处于活动状态时，连接到主服务器的用户可能会感受到响应时间较慢的状况。

日志始终以接收的顺序发送到 SD 辅助服务器。

相关链接

《SinoDB 管理员参考》: SDS_FLOW_CONTROL 配置参数

《SinoDB 管理员参考》: RSS_FLOW_CONTROL 配置参数

集群管理

此章描述了各种用于监视和维护集群的管理任务（其中部分可选）。例如，用于优化性能的负载均衡任务，以及确保安全性任务。

数据复制的工作原理

这些主题描述数据库服务器向辅助服务器复制数据的机制。有关设置、启动和管理各种类型辅助服务器的指示信息，请参阅下表。

表 45: 辅助服务器设置信息

辅助服务器类型	请参阅
HDR 辅助服务器	请参阅高可用性集群配置 在第294页。有关使用外部备份和储存启动 HDR 对的信息，请参阅《*SinoDB*® 备份和还原指南》。
RS 辅助服务器	请参阅远程独立辅助服务器 在第302页。
SD 辅助服务器	请参阅共享磁盘辅助服务器 在第308页

数据初始复制的工作原理

HDR 辅助和 RS 辅助服务器使用存储空间备份和逻辑日志备份（备份到磁带上和磁盘上的备份），来执行从主数据库服务器到辅助数据库服务器的数据初始复制。

SD 辅助服务器不需要来自主服务器的备份与复原，因为 SD 辅助服务器与主服务器共享同一磁盘。

复制数据：

1. 在两个数据库服务器上均安装用户定义类型、用户定义例程和 DataBlade® 模块。
2. 仅在主数据库服务器上注册用户定义类型、用户定义例程和 DataBlade® 模块。
3. 要使两个数据库服务器管理的数据同步，请在主数据库服务器上创建所有储存空间的 0 级备份。
4. 从复制对中的辅助数据库服务器上的备份复原所有存储空间。

   然后，从之前步骤中储存空间备份复原的辅助数据库服务器会从主数据库服务器读取自那个备份以来生成的所有逻辑日志记录。

   数据库服务器会首先从已不再磁盘上的任一备份逻辑日志文件读取逻辑日志记录，然后从磁盘上任一逻辑日志文件中读取逻辑日志记录。

有关复制数据的详细指示，请参阅首次启动 *HDR* 在第298页。 《*SinoDB*® 备份和还原指南》说明了如何使用 ON-Bar 启动复制。

您必须使用存储空间备份执行初始备份。您不能使用数据迁移实用程序，如 onload 和 onunload 来复制数据，因为在每个数据库服务器上表的物理页布局必须相同，以便数据复制可以运行。

将主服务器数据复制到辅助服务器

所有辅助服务器类型都使用日志来复制主服务器数据。主服务器将其整个逻辑日志发送到 HDR 和 RS 辅助服务器，但逻辑页位置仅发送到 SD 辅助服务器。

索引页日志记录可供所有辅助服务器使用，但对于到 RS 辅助服务器的复制而言是必需的。

数据库必须使用事物日志记录来进行复制。

> 警告： 如果主服务器和辅助服务器彼此断开连接，并允许作为标准服务器或主服务器独立运行，那么高可用性数据复制可能需要重新建立。

复制到 HDR 辅助服务器

主数据库服务器可以使用三种同步模式将数据复制到 HDR 辅助服务器：

- 完全同步模式，其中事务需要 HDR 辅助服务器上的完成确认，然后才能完成。

  当您使用完全同步模式时，数据完整性最高，但如果客户端应用程序使用未缓冲日志记录且有许多小事务，那么系统性能可能会受到负面影响。

- 异步模式，其中事务无需 HDR 辅助服务器上的已接收或已完成确认即可完成。

  当您使用异步模式时，系统性能最佳，但如果服务器出现故障，那么数据可能会丢失。
- 接近同步模式，其中事务需要 HDR 辅助服务器上的已接收确认，然后才能完成。

  接近同步模式可提供比完全同步模式更优异的性能，比异步模式更优异的数据完整性。如果SYNC 模式（在 DRINTERVAL 设置为 -1 时开启）使用未缓冲日志记录，那么 SYNC 模式与接近同步模式相同。

同步模式是由 DRINTERVAL 配置参数值、HDR_TXN_SCOPE 配置参数值和数据库日志类型的组合所控制。

以下两个图演示了从主服务器至 HDR 辅助服务器的复制。

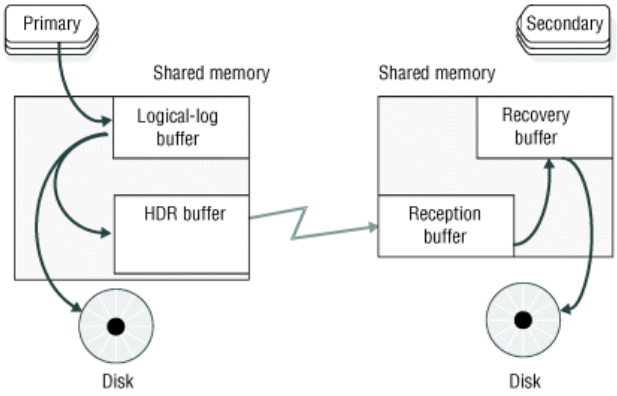

图 55: 如何从主服务器向 HDR 辅助服务器复制数据

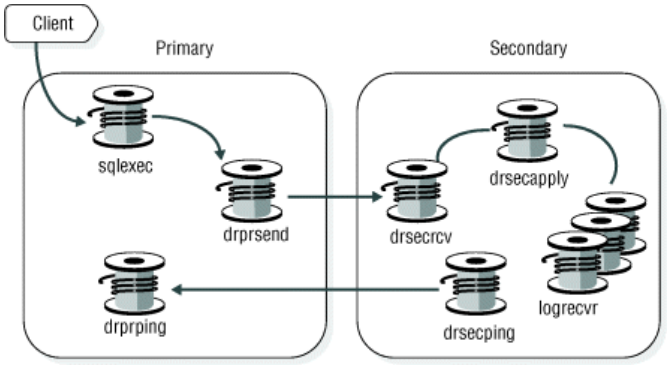

图 56: 管理数据复制的线程

主服务器逻辑日志缓冲区的内容复制到共享内存数据复制缓冲区并清空到磁盘。如果主服务器使用完全同步或接近同步模式，那么它必须接收来自 HDR 辅助服务器的确认，然后才能完成逻辑日志清空。主服务器启动 drprsend 线程以将数据复制缓冲区通过网络传输到辅助服务器的 drsecrcv 线程，然后该线程将数据写入共享内存接收缓冲区。drsecapply 线程将接收缓冲区复制到恢复缓冲区。HDR 和 RS 辅助服务器均使用 logrecvr 线程来将逻辑日志记录应用到其数据库空间。您可以通过更改 OFF_RECVRY_THREADS 的配置参数的值来调整 logrecvr 线程的数量。

drprping 和 drsecping 线程发送和接收消息以监视两个服务器之间的连接。

复制到 RSS 辅助服务器

因为主服务器和 RS 辅助服务器之间的检查点是异步的，所以RS 辅助服务器需要索引页日志。

以下图演示了从主服务器到 RS 辅助服务器的复制。

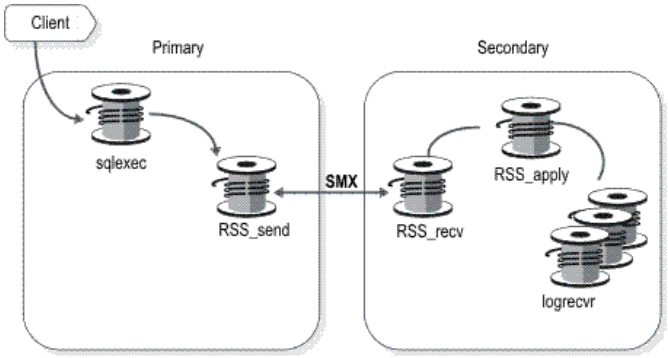

图 57: 管理 RS 辅助服务器数据复制的线程

如果主服务器可以验证其已连接到 RS 辅助服务器，那么 RSS_send 线程将页从磁盘或逻辑日志缓冲区复制到数据复制缓冲区。RSS_send 线程使用“服务器多路复用组”（SMX）连接将数据复制缓冲区发送到 RS 辅助服务器的 RSS_recv 线程。然后，RSS_recv 线程将数据写入接受缓冲区。RSS_apply 线程将接收缓冲区复制到恢复缓冲区。

与 HDR 完全同步模式或接近同步模式不同，主服务器在发送下一缓冲区之前不需要来自辅助服务器的确认。在 RSS_send 线程等待 RSS_Recv 线程接收来自 RS 辅助服务器的确认之前，主服务器最多可发送 32 个未确认的数据复制缓冲区。

复制到 SD 辅助服务器

SD 辅助服务器从磁盘读取逻辑日志页，然后将数据应用到其内存数据缓冲区。

相关链接

集群故障 在第407页
索引页日志记录 在第303页
《SinoDB 管理员参考》: DRINTERVAL 配置参数
《SinoDB 管理员参考》: HDR_TXN_SCOPE 配置参数
《SinoDB SQL 指南: 语法》: HDR_TXN_SCOPE 会话环境选项

HDR 复制的完全同步模式
HDR 完全同步模式确保在主服务器上已提交的事务也已在 HDR 辅助服务器上提交，这样可以在发生故障时保护事务的一致性。

主数据库服务器将逻辑日志缓冲内容写入 HDR 缓冲区后，主数据库服务器从缓冲区将记录发送到 HDR 辅助数据库服务器。仅当主服务器接收到来自 HDR 辅助数据库服务器已收到记录的确认之后，主数据库服务器上的逻辑日志缓冲清空才会完成。

为跟踪同步，主服务器和 HDR 辅助服务器都其保留页中存储以下信息：

- 包含最后完成的检查点的逻辑日志文件的标识
- 逻辑日志文件内检查点记录的位置
- 最后发送或接收的逻辑文件的标识
- 最后发送或接受的逻辑日志记录的页码

要查看此信息，请运行 onstat -g dri ckpt 命令。

HDR 复制对中数据库服务器之间的检查点是同步的。主服务器会等待 HDR 辅助服务器确认收到了检查点日志记录后，再完成其检查点。如果检查点未在 DRTIMEOUT 配置参数指定的时间内完成，那么主数据库服务器将假定发生了故障。

HDR 完全同步模式有以下需求：

- 主服务器和 HDR 辅助服务器上的 DRINTERVAL 配置参数必须设为 0。
- 主服务器和 HDR 辅助服务器上的 DRTIMEOUT 配置参数的值必须相同。

如果主服务器和 HDR 辅助服务器上的操作系统时间是同步的，那么管理会更容易。

要开启完全同步数据复制，请将 DRINTERVAL 配置参数设置为 0，然后使用以下其中一种方法：

- 将 HDR_TXN_SCOPE 配置参数设置为 FULL_SYNC。
- 运行 SET ENVIRONMENT HDR_TXN_SCOPE 'FULL_SYNC';

日志记录会按照接收的顺序应用。当日志传输缓冲区包含很多日志记录时，在 HDR 辅助服务器上应用这些日志记录需要更多时间，并且可能受到负面影响。如果发生这种情况，对于 HDR 数据复制可考虑使用接近同步模式。

相关链接

《SinoDB 管理员参考》: DRINTERVAL 配置参数

《SinoDB 管理员参考》: onstat -g dri 命令: 显示高可用性数据复制信息

《SinoDB SQL 指南: 语法》: HDR_TXN_SCOPE 会话环境选项

HDR 复制的接近同步模式

对 HDR 复制使用接近同步模式时，主服务器在接收到 HDR 辅助服务器已收到所传输的事务后，将逻辑日志缓冲区清空到磁盘。主服务器不会等待事务在 HDR 辅助服务器上已提交的确认。

当日志传输缓冲区包含许多日志记录时，在 HDR 辅助服务器上应用这些日志记录需要更多时间。HDR 复制的接近同步模式提供比完全同步模式更优异的性能，比异步模式更优异的数据完整性。

主服务器在其保留页存储以下接近同步信息：

- 已排队至 drprsend 线程的未处理数据复制缓冲区的数量 。
- 日志唯一值，最近分页日志的页码。
- 线程控制块（TCB）指针、括号里的线程标识，以及由该线程执行的提交的日志序列号（LSN）。
- 等待 HDR 辅助服务器已接收确认的提交的 LSN。

要查看此信息，请运行 onstat -g dri que 命令。

HDR 接近同步模式有以下需求：

- 主服务器和 HDR 辅助服务器上的 DRINTERVAL 配置参数必须设置为 -1，或者主服务器上的 DRINTERVAL 配置参数必须设置为 0。
- 主服务器和 HDR 辅助服务器的 DRTIMEOUT 配置参数必须设置为相同值。
- 主服务器和 HDR 辅助服务器的操作系统时间必须是同步的。

要开启接近同步数据复制，请将 DRINTERVAL 配置参数设置为 0，然后使用以下其中一种方法：

- 将 HDR_TXN_SCOPE 配置参数设置为 NEAR_SYNC。
- 运行 SET ENVIRONMENT HDR_TXN_SCOPE 'NEAR_SYNC';

相关链接

《SinoDB 管理员参考》: DRINTERVAL 配置参数

《SinoDB 管理员参考》: onstat -g dri 命令: 显示高可用性数据复制信息

《SinoDB SQL 指南: 语法》: HDR_TXN_SCOPE 会话环境选项

HDR 复制的异步模式

异步 HDR 复制表示主服务器不等待 HDR 辅助服务器的响应，就将逻辑日志清空到磁盘。异步 HDR 复制可提高复制速度，但事务可能会丢失。

可通过多种方式打开 HDR 复制的异步模式：

- 将 DRINTERVAL 配置参数设置为正整数。
- 将 DRINTERVAL 配置参数设置为 0，并将 HDR_TXN_SCOPE 配置参数设置为 ASYNC。
- 运行以下语句：

```
SET ENVIRONMENT HDR_TXN_SCOPE 'ASYNC';
```

在异步模式中，主数据库服务器在将逻辑日志缓冲区的内容复制到数据复制缓冲区后，就会将逻辑日志清空到磁盘。当以下条件之一发生时，主数据库服务器会通过网络发送 HDR 缓冲区的内容：

- HDR 缓存区已满。
- 自记录发送到 HDR 辅助数据库服务器以来的时间间隔已超过主服务器的 DRINTERVAL 配置参数值。

为降低使用异步复制的集群中丢失事务的风险，请对所有的数据库使用无缓冲的日志记录。无缓冲日志记录减少了事务记录写入与传输之间的时间量。如果您的主服务器使用缓冲日志记录，并且您接收到 error -7350 Attempt to update a stale version of a row 消息，请切换到无缓冲日志记录。

如果发生故障转移，但主服务器带着数据复制重启，那么在主服务器上已提交但在辅助服务器上未提交的事务会存储到由 DRLOSTFOUND 配置参数指定的文件中。

相关链接

《SinoDB 管理员参考》: DRINTERVAL 配置参数

《SinoDB 管理员参考》: onstat -g dri 命令: 显示高可用性数据复制信息

《SinoDB SQL 指南: 语法》: HDR_TXN_SCOPE 会话环境选项

失物招领事务

使用异步更新，主数据库服务器上已提交的事务可能未复制到辅助数据库服务器上。如果在主数据库服务器将提交记录复制到 HDR 缓冲区之后但在主服务器将提交记录发送到辅助数据库服务器之前发生故障，那么就会发生这种情况。

如果辅助数据库服务器在主数据库服务器故障之后转变为标准数据库服务器，辅助数据库服务器建会回滚任何打开的事务。这些事务包含了任何在主数据库服务器上已提交但辅助数据库服务器未接收到其提交记录的事务。结果，事务在主数据库服务器上但未在辅助数据库服务器上提交。当您在故障后重启数据复制，数据库服务器在主数据库服务器的逻辑恢复期间将丢失事务的所有逻辑日志记录放入某文件（由 DRLOSTFOUND 配置参数指定）中。下图说明了该过程。

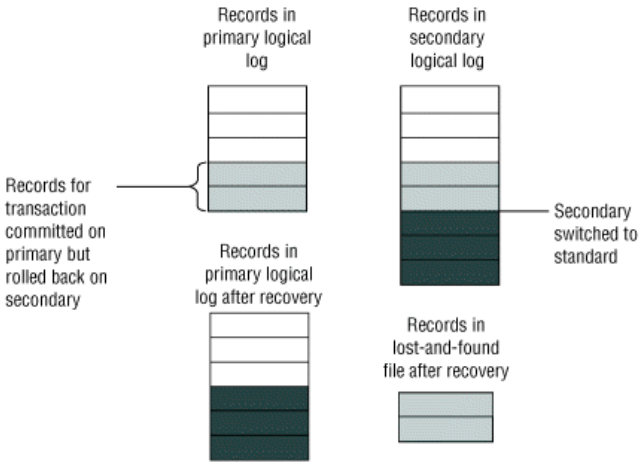

图 58: 使用失物招领文件

如果在主数据库服务器重新启动数据复制后在运行主数据库服务器的计算机上创建失物招领文件，那么表示事务已丢失。数据库服务器无法重新应用失物招领文件中的事务记录，因为在辅助数据库服务器充当标准数据库服务器时可能发生有冲突的更新。

为了降低不用同步模式运行数据复制而丢失事务的风险，请对所有的数据库使用无缓冲日志记录。该方法减少了从主数据库服务器到辅助数据库服务器的事务记录的写入和传输之间的时间量。

数据复制配置示例

这些主题描述了如何配置数据复制环境的一些示例。

远程独立辅助服务器配置示例

下图说明了由多个 RS 辅助服务器组成的配置的示例。当主服务器距离与 RS 辅助服务器很远 之间的距离较远时，或者当主服务器和辅助服务器之间的网速较慢或不稳定时，这种配置将非常有效。由于 RS 辅助服务器使用全双工通信协议，并且不要求同步检查点处理，因此主服务器的性能通常不受影响。

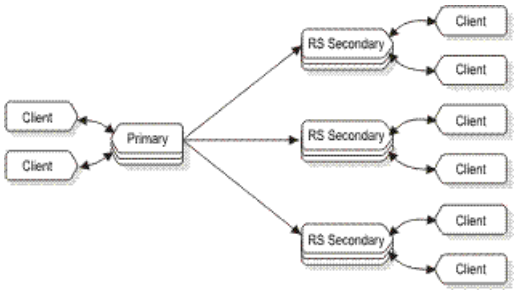

图 59: 配有三台 RS 辅助服务器的主服务器

下一个图显示了 RS 辅助服务器和 HDR 辅助服务器的配置示例。在该此示例中，HDR 辅助服务器提供高可用性，而 RS 辅助服务器提供当主服务器和 HDR 辅助服务器都故障时作为备用灾难恢复。RS 辅助服务器在地理位置上可距离主服务器和 HDR 辅助服务器很远，因此区域性破坏（如地震或洪水）不会对 RS 辅助服务器造成影响。

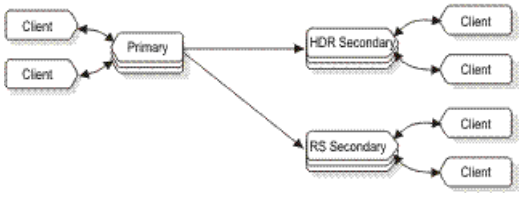

图 60: 配有 HDR 辅助服务器和 RS 辅助服务的主服务器

如果主数据库服务器发生故障，那么可将现有的 HDR 辅助服务器转变为主服务器，如下图所示：

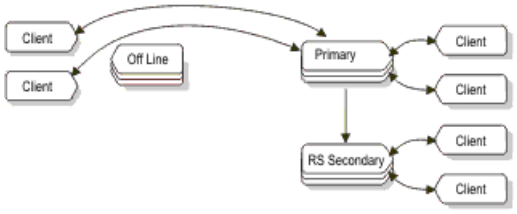

图 61: 主服务器故障转移到 HDR 辅助服务器

如果原主服务器将长时间处于脱机状态，那么 RS 辅助服务器可转换为 HDR 辅助服务器。之后，当原主服务器恢复为联机状态时，可将其配置为 RS 辅助服务器，如下图所示：

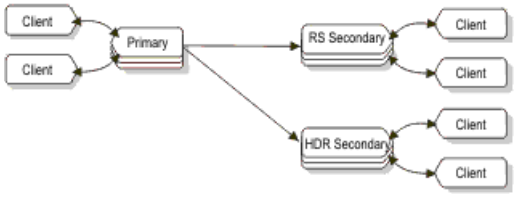

图 62: 充当 HDR 辅助服务器角色的 RS 辅助服务器

共享磁盘辅助服务器配置示例

下图显示配有两个 SD 辅助服务器的主服务器的示例。在这种情况中，主服务器的角色可以转移到两个 SD 辅助服务器中的任一个。无论是由于计划的中段还是主服务器故障而导致主服务器必须停止服务，均是如此。

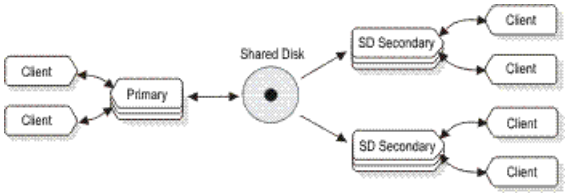

图 63: 配有两个 SD 辅助服务器的主服务器

由于两台 SD 辅助服务器都是读取同一磁盘子系统，因此它们都同样能够充当主服务器角色。下图说明了主服务器处于脱机状态时的情况。

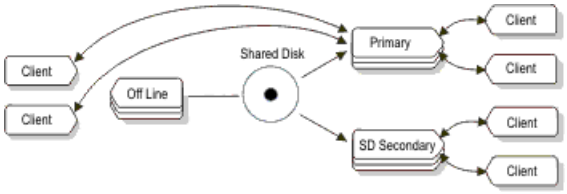

图 64: 充当主服务器角色的 SD 辅助服务器

有多种方式可以防止共享磁盘的硬件发生故障。可能最常见的方法就是基于 RAID 技术（如 RAID-5）来配置磁盘阵列。另一种防止磁盘故障的方法是使用 SAN（存储区域技术）来包含远程磁盘镜像的某些格式。由于 SAN 磁盘可位于距离主磁盘和其镜像很近的位置，因此可为服务器或磁盘子系统的计划中断和意外中断提供高度可用性。下图说明了此种配置 ：

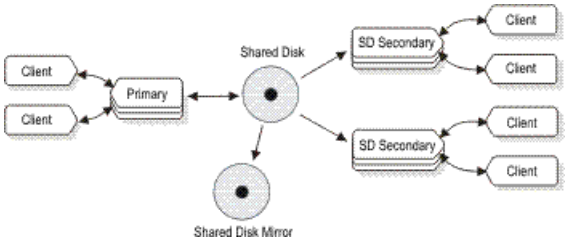

图 65: 具有镜像磁盘的主服务器和 SD 辅助服务器

如果磁盘发生故障，服务器可以重新配置，如下图所示 ：

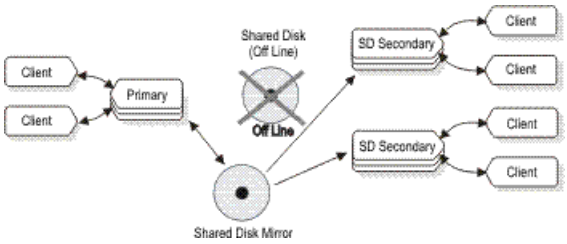

图 66: 主共享磁盘故障后的共享磁盘镜像

除了配置镜像的磁盘子系统，如上图所述，您可能想要配置其他服务器。例如，您可能想要在单个刀片服务器机柜内使用主服务器和两个 SD 辅助服务器。通过将服务器组放置到单台刀片服务器中，刀片服务器本身可成为故障点。必须定期地增加读取的处理能力时（例如，执行大型报告任务时），下图中的配置是一种具吸引力的解决方案。

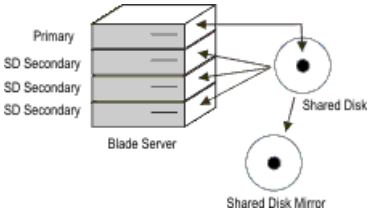

图 67: 刀片服务器中的主服务器和 SD辅助服务器

您可以决定使用多个刀片服务器来避免单个刀片服务器可能引起的故障点，如下图所示。

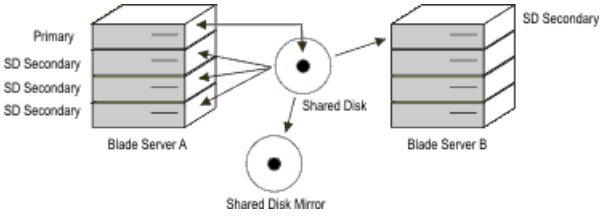

图 68: 多刀片服务器配置，以防止单点故障

在上图中，如果刀片服务器 A 故障，那么它将把主服务器角色转移到刀片服务器 B上的 SD 辅助服务器。由于它是可以快速地使额外的 SD 辅助服务器联机，因此可以动态地将额外的 SD 辅助服务器添加到刀片服务器 B，如下图所示。

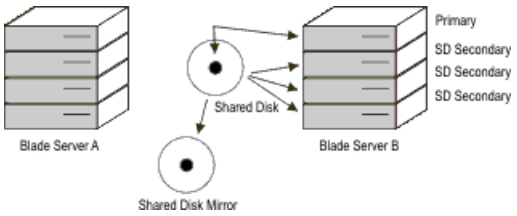

图 69: 刀片服务器故障后的故障转移

由于主服务器与磁盘镜像可支持的镜像磁盘之间的距离存在限制，因此您可能会对使用共享磁盘并依靠共享磁盘镜像来提供磁盘可用性有所考虑。例如，您可能希望磁盘子系统的两个副本之间有著相当显著的距离。在这种情况下，您可能会选择使用 HDR 辅助服务器或 RS 辅助服务器来维护磁盘子系统的辅助副本。如果网络连接很快（即，如果辅助服务器对于 ping 操作的响应时间短于 50 毫秒），那么必须考虑使用 HDR 辅助服务器。对于速度较慢的网络连接，请考虑使用 RS 辅助服务器。下图显示了刀片服务器配置中 HDR 辅助服务器的示例。

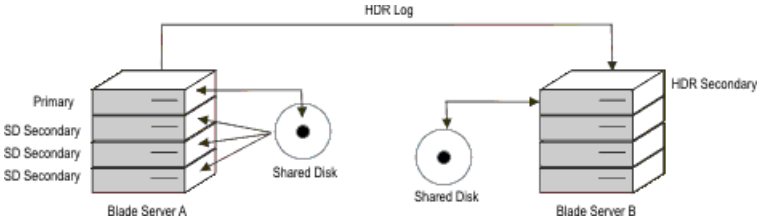

图 70: 刀片服务器配置中的 HDR 辅助服务器

在上图所示的配置中，如果主节点故障，但共享磁盘未受损且刀片服务器仍然可运行，那么可将主服务器角色从刀片服务器 A 的第一台服务器转移到同一刀锋服务器的另一台服务器。更改主服务器会导致远程 HDR 辅助服务器的源自动重新路由到新的主服务器，如下图所示 ：

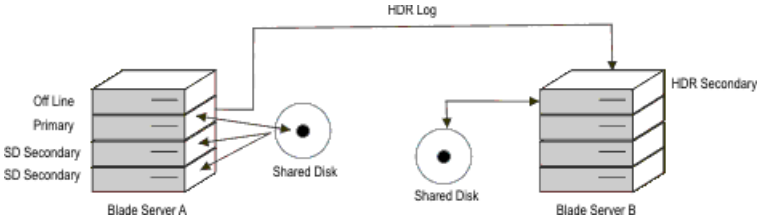

图 71: 刀片服务器配置中从主服务器故障转移到 SD 辅助服务器

然而，假设在上图中所描述的故障不是刀片服务器中的某个刀片，而是整个刀片服务器。在这种情况下，那么可能需要故障转移到 HDR 辅助服务器。由于 SD 辅助服务器的启动非常快，所以可以轻松地添加额外的 SD 辅助服务器。请注意，SD 辅助服务器仅能与主节点配合使用；当主节点转移到刀片服务器 B 时，那么需要启动刀片服务器 B 上的 SD 辅助服务器，如下图所示。

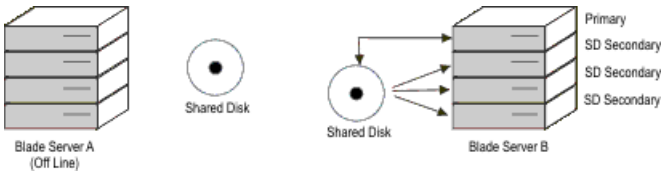

图 72: 整个刀片服务器故障

Enterprise Replication 作为可恢复组的一部分

虽然 Enterprise Replication 不支持 SYNC （同步）操作模式，但它提供了支持具有多台活动服务器的环境的能力。在故障转移事件期间， Enterprise Replication 可以协调两个服务器之间的数据库差异。必须将 Enterprise Replication 视为提高服务器之间同步的方法，因为每个 Enterprise Replication 系统都维护一个独立的日志记录系统。下图显示了使用 Enterprise Replication 的配置。

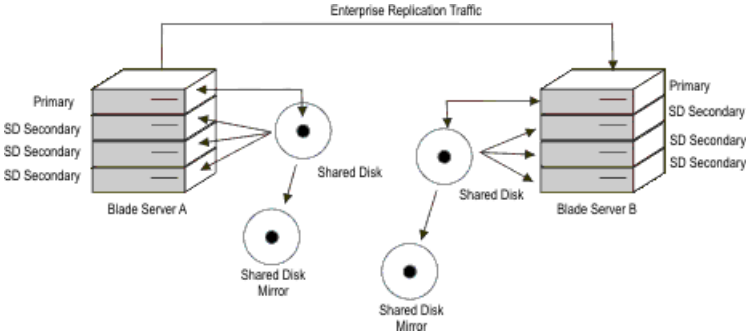

图 73: 配置 Enterprise Replication 作为可恢复组的一部分

使用 Enterprise Replication 的高可用性集群配置示例

假设在如下配置的两个高可用性服务器集群之间需要 Enterprise Replication：

集群 1 ：

- 主服务器
- HDR 辅助服务器
- SD 辅助服务器 1
- SD 辅助服务器 2
- RS 辅助服务器 1
- RS 辅助服务器 2

集群 2 ：

- 主服务器
- HDR 辅助服务器
- SD 辅助服务器 1
- SD 辅助服务器 2
- RS 辅助服务器 1
- RS 辅助服务器 2

进一步假设每台服务器按以下约定命名 ：

- 前三个字符：企业名称
- 第 4 个字符：主机的简短编号
- 第5、 6 和 7 个字符：集群编号
- 第8、 9 和 10 个字符：服务器类型："pri" 表示主服务器，"sec" 表示辅助服务器
- 第11、 12 和 13 个字符：连接类型："shm" 或 "tcp"

例如，名为 srv4_1_pri_shm 的服务器描述如下 ：

- srv = 企业名称

- 4 = 主机的简短编号
- _1_ = 集群编号
- pri = 这是主服务器
- shm = 连接类型使用共享内存通信

sqlhosts 文件中的以下条目将支持上面的配置 ：

```
srv4_1_pri_shm onipcshm sun-mach4 srv4_1_pri_shm
srv4_1_sec_shm onipcshm sun-mach4 srv4_1_sec_shm
srv5_1_rss_shm onipcshm sun-mach5 srv5_1_rss_shm
srv5_1_sds_shm onipcshm sun-mach5 srv5_1_sds_shm
srv6_1_rss_shm onipcshm sun-mach6 srv6_1_rss_shm
srv6_1_sds_shm onipcshm sun-mach6 srv6_1_sds_shm
srv_1_cluster group - - i=1
srv4_1_pri_tcp ontlitcp sun-mach4 21316 g=srv_1_cluster
srv4_1_sec_tcp ontlitcp sun-mach4 21317 g=srv_1_cluster
srv5_1_rss_tcp ontlitcp sun-mach5 21316 g=srv_1_cluster
srv5_1_sds_tcp ontlitcp sun-mach5 21317 g=srv_1_cluster
srv6_1_rss_tcp ontlitcp sun-mach6 21316 g=srv_1_cluster
srv6_1_sds_tcp ontlitcp sun-mach6 21317 g=srv_1_cluster

srv4_2_pri_shm onipcshm sun-mach4 srv4_2_pri_shm
srv4_2_sec_shm onipcshm sun-mach4 srv4_2_sec_shm
srv5_2_rss_shm onipcshm sun-mach5 srv5_2_rss_shm
srv5_2_sds_shm onipcshm sun-mach5 srv5_2_sds_shm
srv6_2_rss_shm onipcshm sun-mach6 srv6_2_rss_shm
srv6_2_sds_shm onipcshm sun-mach6 srv6_2_sds_shm
srv_2_cluster group - - i=2
srv4_2_pri_tcp ontlitcp sun-mach4 21318 g=srv_2_cluster
srv4_2_sec_tcp ontlitcp sun-mach4 21319 g=srv_2_cluster
srv5_2_rss_tcp ontlitcp sun-mach5 21318 g=srv_2_cluster
srv5_2_sds_tcp ontlitcp sun-mach5 21319 g=srv_2_cluster
srv6_2_rss_tcp ontlitcp sun-mach6 21318 g=srv_2_cluster
srv6_2_sds_tcp ontlitcp sun-mach6 21319 g=srv_2_cluster
```

复杂故障移转恢复策略示例

本主题描述了在大区域范围灾难情况下，用于达成最大可用性的三层服务器的方法。

通常，HDR 辅助服务器为 SD 辅助服务器提供备份，并为在地理上与主机系统相距很远的高度可用系统提供支持。RS 辅助服务器为 HDR 辅助服务器提供附加可用性，并视为灾难可用性的解决方案。如果必须使用 RS 辅助服务器确保可用性，那么将强制您通过执行备份和复原来手动重新构建其他系统，以便回到正常运行。为进一步理解，将呈现一个大区域范围灾难的场景，例如暴风。

为提供最大可用性以免受区域灾难影响需要分层可用性。第一层提供的可用性解决方案可处理短暂的本地故障。例如，这可能包括将一些刀片服务器连接到运行 SD 辅助服务器的单个磁盘子系统。将 SD 辅助服务器放置在校园内多个位置，可在发生本地中段时提供无缝故障转移。

您可能想要添加第二层，通过包含带有其自身磁盘副本的备用位置来提高可用性。为防止大范围的灾难，您可以考虑在较远位置（可能数百英里）配置一台 HDR 辅助服务器。也可能希望使远程系统成为刀片服务器或某个其他多服务器系统。如果发生故障转移并且远程 HDR 辅助服务器成为主服务器，那么通过提供的第二层可轻松地启动远程站点上的 SD 辅助服务器。

然而，即使是两层的方法也可能是不够的。某个區域的飓风可能造成几百英里之外的龙卷风。要防护这种情况，请考虑添加第三层保护，例如位数千英里以外的 RS 辅助服务器。此三层方法提供了可以显著降低中断风险的额外冗余。

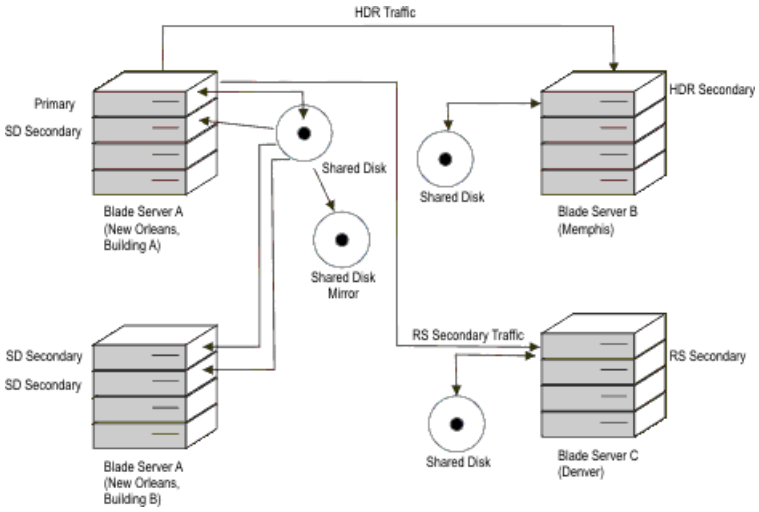

图 74: 三层服务器可用性的配置

现在假设在新奥尔良校园的建筑物 A 发生本地中断。可能是机房里一根水管爆炸，导致刀片服务器和共享磁盘子系统的主服务器副本遭受水害。您可以通过在建筑物 B 的刀片服务器上运行的其中一个 SD 辅助服务器上运行 onmode -d 使其具有主服务器名称，以将主服务器角色切换到建筑物 B。这将导致所有其他辅助节点自动连接到新的主节点。

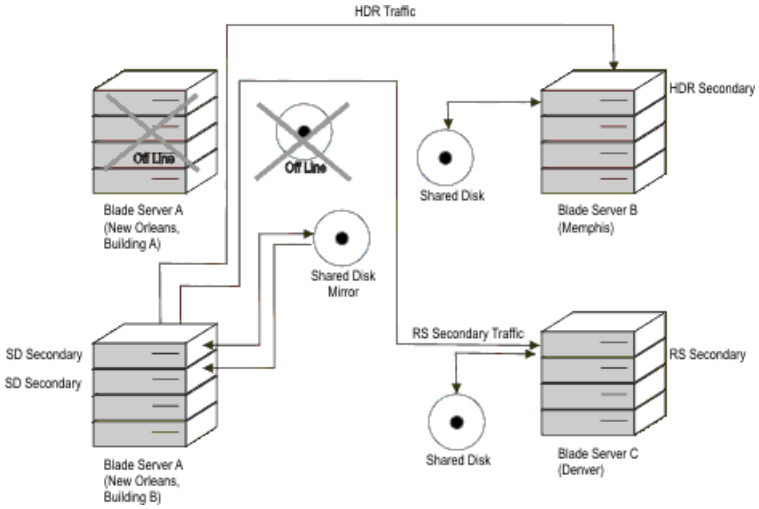

图 75: 第一层保护

如果新奥尔良发生区域中断，以致建筑物 A 和 B 都丢失，那么可将主服务器角色切换到孟菲斯。此外，可能希望使丹佛的服务器成为 HDR 辅助服务器，并可能在孟菲斯的机器上添加额外的 SD 辅助服务器。

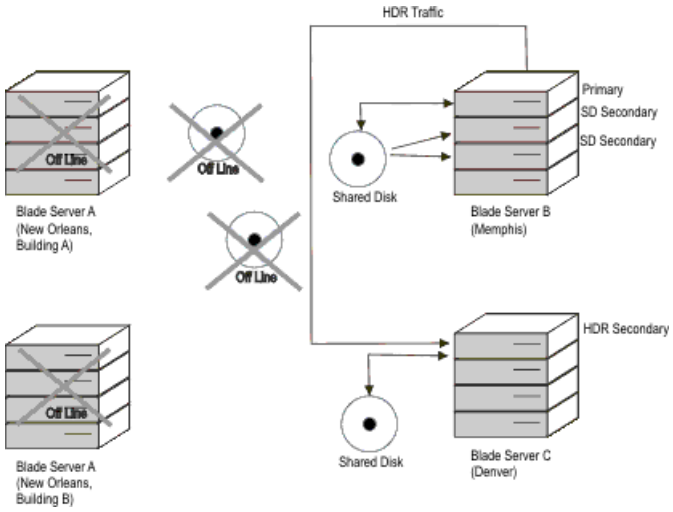

图 76: 第二层保护

遭遇到影响两个站点的更大中断将需要切换到最远程的系统。

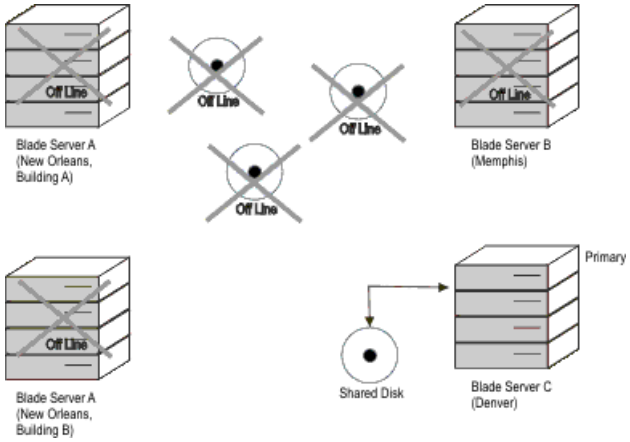

图 77: 第三层保护

表 46: 各种需求的建议配置

需求	建议配置
必须定期提高报告容量	使用 SD 辅助服务器
您使用的是提供足够磁盘硬件可用性的 SAN 设备，但担心发生服务器故障	使用 SD 辅助服务器
您使用的是提供足够磁盘硬件镜像的 SAN 设备，但还需要主服务器操作丢失（并且镜像磁盘的限制不是问题）时能够联机的第二组服务器。	请考虑在两个站点使用两个刀片服务器中心来运行 SD 辅助服务器
您想要在适度的距离位置有一个备份网站，但不能容忍故障转移期间任何数据丢失。	考虑使用两个刀片服务器中心，SD 辅助服务器在主刀片服务器中心，HDR 辅助服务器在远程刀片服务器中心。
您需要具有未曾丢失数据的高度可用系统，但也必须在世界的另一边设置远程系统。	请考虑使用针对数据复制运行完全同步模式或接近同步模式的本地 HDR 辅助服务器，并在世界的另一边使用 RS 辅助服务器。
您需要高可用性解决方案，但在您所在区域的网络，ping 的最佳响应时间约为 200 毫秒。	请考虑使用 RS 辅助服务器

需求	建议配置
您需要备份网站，但不具有与备份站点的任何直接通信	请考虑使用具有备份和恢复的 "连续日志复原"
只要数据最终到达目的地，您能夠容忍數據交付中的延迟；但在任何情况下，都必须可以快速故障转移	请考虑使用具有与 ER 结合的硬件磁盘镜像的 SD 辅助服务器。
您需要额外写入处理能力，能够容忍这些写入交付中的延迟，需要高度可用的东西，并且可以分割工作负载。	请考虑在 SD 辅助服务器使用 ER

高可用性集群环境故障诊断

与独立服务器环境相比，高可用性集群环境需要很少甚至不需要额外的故障诊断。此主题说明用于描述高可用性集群环境的术语，并提供一些常见的故障诊断过程。

可以使用诊断工具显示高可用性环境的配置，并验证是否正确设置辅助服务器来更新数据。

服务器处理事务的速度很快。onstat 命令只显示命令运行瞬间的状态信息。

为了更新辅助数据库上的数据，SinoDB® 会在主数据库服务器和辅助数据库服务器上创建 代理分发器。为每个代理分发者分配一个在集群内唯一的标识。代理分发器负责将 DML 更新请求从辅助服务器发送至主服务器。辅助服务器基于其 onconfig 文件中的 UPDATABLE_SECONDARY 设置确定要创建的代理分发器的实例数。

在高可用性集群中初始化可更新辅助服务器时，该服务器将保持在快速恢复方式，直到所有打开的事务（包括打开和已准备的 XA 事务）完成为止。在快速恢复方式时，应用程序无法连接至该服务器。该服务器将保持在快速恢复方式，直到所有打开的事务提交或回滚。

使用主服务器上的 onstat -g proxy 命令查看有关高可用性集群中所有代理分发器的信息。

```
onstat -g proxy

Secondary    Proxy      Reference     Transaction  Hot Row
Node         ID         Count         Count        Total
serv2        392        0             2            112
serv2        393        0             2            150
```

检查前示例中 onstat 命令的输出，其中有两个代理分发器，标识分别为 392 和 393。Transaction Count 显示每个代理分发器正在处理的事务数。

可在辅助服务器上运行 onstat -g proxy 来查看有关能够为来自辅助服务器的更新请求提供服务的代理分发服务器的信息。

```
onstat -g proxy

Primary    Proxy     Reference     Transaction  Hot Row
Node       ID        Count         Count        Total
serv1      392       0             2            112
serv1      393       0             2            150
```

在此示例中，服务器是共享磁盘（SD）辅助服务器，并配置为更新数据。另外，该服务器连接到名为 serv1 的主服务器，并有两个代理分发器，每个的事务计数为 2。

在主服务器上使用 onstat -g proxy all 命令显示有关代理分发器和 代理线程 的信息。代理分发器创建一个或多个代理线程，以用于处理来自辅助服务器的数据更新。

```
onstat -g proxy all

Secondary  Proxy      Reference     Transaction  Hot Row
Node       ID         Count         Count        Total
```

```
serv2      392        0             2           1
serv2      393        0             2           0

TID  Flags       Proxy  Source  Proxy   Current  sqlerrno iserrno
                 ID     SessID  TxnID   Seq
63   0x00000024  392    22      1       5        0        0
64   0x00000024  392    19      2       5        0        0
62   0x00000024  393    23      1       5        0        0
65   0x00000024  393    21      2       5        0        0
```

在 onstat -g proxy all 命令的输出中， TID 表示在主服务器上运行的代理线程的标识。Source SessID 表示辅助服务器上的用户会话的标识。Proxy TxnID 显示当前事务的序号。每个 Proxy TxnID 对于代理分发器而言都是唯一的。Current Seq 表示正在处理的事务中当前操作的序号。每个发送至辅助服务器的数据库事务，在内部都分成一个或多个操作才发送至主服务器。最后两个字段 sqlerrno 和 iserrno 显示在事务中遇到的任何 SQL 或 ISAM/RSAM 错误。错误号为 0 表示完成且没有错误。

在辅助服务器上运行 onstat -g proxy all 显示有关当前可更新辅助服务器上数据的所有会话的信息。

```
onstat -g proxy all

Primary   Proxy      Reference     Transaction  Hot Row
Node      ID         Count         Count        Total
serv1     3466       0             0            1
serv1     3465       0             1            0

Session  Proxy  Proxy  Proxy  Current  Pending  Reference
         ID     TID    TxnID  Seq      Ops      Count
19       3465   67     1      23       0        1
```

在辅助服务器上执行的 onstat -g proxy all 命令生成的输出中，Session 表示辅助服务器上用户会话的标识。Proxy ID 和 Proxy TID 与主服务器上显示的一样。Pending Ops 显示等待传送至主服务器的操作数。Reference Count 显示事务正在使用的线程数。当 Reference Count 显示 0 表示事务处理完成。

要显示由给定分发器执行的当前工作的详细信息，请使用：

```
onstat -g proxy <proxy id> [proxy transaction id] [operation number]
```

proxy transaction id 和 operation number 都是可选参数。如提供，第一个数字视为代理事务标识。如在后面跟著第二数字，其视为操作编号。如果不存在任何代理事务标识，此命令的执行与 onstat - 相同。相似地，如果给定代理事务标识的操作编号不存在，此命令的执行与 onstat - 相同。

使用以下命令显示有关服务器是否配置为允许更新数据的信息。此命令可在主服务器或辅助服务器上运行：

```
onstat -g <server_type>
```

Examples:

```
onstat -g rss
onstat -g sds
onstat -g dri
```

设计数据复制组客户端

此主题针对连接到正在运行数据复制的数据库服务器的客户端的各种设计注意事项进行说明。

另请参阅辅助服务器上的隔离级别 在第333页以获取有关辅助服务器上 committed read 和 committed read last committed 隔离级别的信息。

用于排序的临时数据库空间和临时表的使用

即使辅助服务器为只读模式，但当它必须进行排序及创建临时表时也会进行写入操作。临时数据库空间 在第145页说明了数据库服务器从何处找到临时空间以在排序期间使用或用于临时表。

为防止辅助服务器写入处于逻辑恢复模式的数据库空间，您必须采取以下操作：

1. 确保一个或多个临时数据库空间存在。

   有关创建临时数据库空间的指示信息，请参阅创建使用缺省页大小的数据库空间 在第173页。
2. 执行以下操作之一：
   - 将辅助数据库服务器的 onconfig 文件中的 DBSPACETEMP 参数设置为临时数据库空间或数据库空间。
   - 将客户端应用程序的 DBSPACETEMP 环境变量设置为临时数据库空间或数据库空间。

在辅助服务器（SD 辅助服务器、 RS 辅助服务器和 HDR 辅助服务器）上必须使用 WITH NO LOG 选项来创建临时表。或者，将辅助服务器的 TEMPTAB_NOLOG 配置参数设为 1，以便将临时表的缺省日志记录方模式改为无日志记录。在日志记录启用情况下创建表将导致 ISAM 错误。

对于 SD 辅助服务器，设置 SDS_TEMPDBS 配置参数以配置 SD 辅助服务器使用的临时数据库空间。

对于 SD 辅助服务器，无需显示添加临时数据库空间，因为启动服务器时，辅助服务器会分配由 SDS_TEMPDBS 指定的块。它只需要准备用于接收块的设备。

如果高可用性集群中的主服务器故障，并且 SD 辅助服务器接手成为主服务器，那么 SD 辅助服务器上 SDS_TEMPDBS 配置参数所设置的值将用于临时数据库空间，直到服务器重新启动。必须确保辅助服务器上 SDS_TEMPDBS 配置参数所指定的值 ，与主服务器上所指定的值不同。SD 辅助服务器重新启动之后，将使用 DBSPACETEMP 配置参数。

执行基本管理任务

这些主题包含系统运行 HDR 时如何执行数据库服务器管理任务的指示信息。

更改 HDR 复制对的配置参数

一些配置参数必须在复制对中的两个数据库服务器上均设置为相同的值（如 集群的数据库服务器配置需求 在第296页中所列）。在每个数据库服务器上可以有不同的值的配置参数可以单独更改。

更改 onconfig 文件，请执行以下操作：

1. 使用 onmode -k 选项使每个数据库服务器脱机。

   如果启用连接管理器的自动故障转移或自动将 DRAUTO 设置切换为 1 或 2，请先使 HDR 辅助服务器脱机。
2. 更改每个数据库服务器上的参数。
3. 从最后脱机的数据库服务器开始，将每个数据库服务器恢复联机。

   例如，如果最后脱机的是 HDR 辅助数据库服务器，那么首先使该 HDR 辅助服务器恢复联机。表 *43: 首次启动 HDR 的步骤* 在第299页列出使主数据库服务器和辅助数据库服务器恢复联机的过程。

备份存储空间和逻辑日志文件

当使用 HDR 时，必须仅在主数据库服务器上备份逻辑日志文件和存储空间。但是，请做好在辅助数据库服务器上执行存储空间和逻辑日志备份的准备，以防万一数据库服务器类型更改为标准的情况发生。

必须在两个数据库服务器上使用相同的备份和复原工具。

所使用的块大小和磁带大小（用于存储空间备份和逻辑日志备份）必须在主数据库服务器和辅助数据库服务器上均相同。

可使用 ontape 将磁带大小设置为 0 ，以便自动使用磁带的全部物理容量。

更改数据库的日志记录模式

当使用 HDR 时，不可开启主数据库服务器上数据库的事务日志记录。您可以关闭数据库的日志记录；但是，随后对该数据库所做的更改将不会复制到辅助数据库服务器上。

打开数据库日志记录，请执行以下操作：

1. 要关闭 HDR，请关闭辅助数据库服务器。
2. 打开数据库日志记录。

   在开启数据库日志记录后，如果未在主数据库服务器上执行 0 级备份就启动数据复制，然后在辅助数据库服务器进行复原，那么主数据库服务器和辅助数据库服务器上数据库的数据可能不相同。这种情况可能会导致数据复制问题。
3. 在主数据库服务器上执行 0 级备份，并在辅助数据库服务器上进行复原。

   在首次启动 *HDR* 在第298页中描述了该过程。

添加和删除块与存储空间

仅可从主数据库服务器执行磁盘布局操作，例如添加或删除块和数据库空间。在辅助数据库服务器上复制该操作。这种安排确保复制对中的两个数据库服务器的磁盘布局保持一致。

块的目录路径名或实际文件在您创建前块之前就必须存在。请确保在主数据库服务器上创建块之前路径名（和偏移量，如果适用）存在于辅助数据库服务器上，否则数据库服务器将关闭数据复制。

> 提示： 在具有一个或多个 SD 辅助服务器的高可用性集群的主服务器上添加数据库空间时，SD 辅助服务器的 online.log 可能会显示以下错误："Assert Failed: Page Check Error"。如果发生这种情况，关闭并重新启动 SD 辅助服务器。在重新启动 SD 辅助服务器后，新添加的数据库空间将变为可用和完全发挥作用。

重命名块

如果您的块路径名是使用符号链接，那么可在 HDR 运行时重命名块。有关重命名块的指示信息，请参阅《*SinoDB*® 备份和还原指南》。

如果您的块路径名不是使用符号链接，那么在重命名块时，必须使两个数据库服务器保持脱机直到完成数据库服务器的冷复原。

在故障的 HDR 服务器上重命名块，请执行以下操作：

1. 将未损坏的服务器更改为标准模式。
2. 对标准服务器执行 0 级备份。
3. 关闭标准服务器。
4. 从新的 0 级归档进行冷复原期间重命名标准服务器上的块（有关指示信息，请参阅 《*SinoDB*® 备份和还原指南》）。
5. 启动标准服务器
6. 对标准服务器再次执行 0 级归档。请确保服务器处于标准模式。
7. 使用新的 0 级备份复原发生故障的服务器，并重新建立 HDR 对。

辅助数据库服务器上保存块状态

对于高可用性集群服务器，如果在辅助数据库服务器上更改块状态（脱机或联机），并且该辅助服务器在检查点完成前重新启动，那么不保存更新的块状态。

要确保新的块状态清空到辅助数据库服务器的保留页上，可在主数据库服务器上强制执行检查点，并验证检查点也已在辅助数据库服务器上完成。即使辅助数据库服务器重新启动，此时也将保留新的块状态。

如果辅助数据库服务器上的主块脱机，您可以从镜像块中恢复该主块。

要从镜像块中恢复主块，请执行以下操作：

1. 在辅助数据库服务器上运行 onspaces -s 以使主块联机。

2. 在主数据库服务器上运行 onmode -c 以强制执行检查点。
3. 在主数据库服务器上运行 onmode -m 以验证检查点已完成。
4. 在辅助数据库服务器上运行 onmode -m 以验证检查点也已在辅助数据库服务器上完成。

完成这些步骤后，重新启动辅助数据库服务器时，主块将联机。

使用和更改块的镜像

在添加镜像块之前，必须已经在主数据库服务器和辅助数据库服务器上分配了块的磁盘空间。如果想在复制对中的其中一个数据库服务器上制作数据库空间镜像，那么您必须在两个数据库服务器上创建该数据库空间的镜像块。有关分配磁盘空间的一般信息，请参阅分配磁盘空间 在第169页。

除非正在使用镜像，否则请不要将 MIRROR 配置参数设置为 1 。

仅可从主数据库服务器执行磁盘布局操作。因此，仅可以从主数据库服务器添加或删除镜像块。您向主数据库服务器添加或删除的镜像块，也会向辅助数据库服务器添加或删除。您必须在辅助数据库服务器上为新添加的镜像块执行镜像恢复。有关更多信息，请参阅恢复镜像块 在第281页。

当您从主数据库服务器删除块时，SinoDB® 自动删除辅助数据库服务器上相应的块。这对主块和镜像块均适用。

当您关闭主数据库服务器上数据库空间的镜像时， SinoDB® 不会关闭辅助数据库服务器上相应数据库空间的镜像。要关闭独立于主服务器的辅助数据库服务器上数据库空间的镜像，请使用 onspaces -r。有关关闭镜像的更多信息，请参阅结束镜像 在第281页。

可在主数据库服务器或辅助数据库服务器上关闭镜像块或恢复镜像块。这些过程对于 HDR 来说是透明的。

管理物理日志

两个数据库服务器上的物理日志大小必须相同。如果更改了主数据库服务器上物理日志的大小和位置，那么此更改将复制到辅助数据库服务器上。辅助数据库服务器上的 ONCONFIG 值将自动更新。

有关更改物理日志的大小和位置的信息，请参阅管理物理日志 在第270页。

管理逻辑日志

两个数据库服务器上的逻辑日志大小必须相同。您可以使用 onparams 实用程序添加或删除逻辑日志文件，如管理逻辑日志文件 在第250页中所述。SinoDB® 将在辅助数据库服务器上复制此更改；但是，不会更新辅助数据库服务器上的 LOGFILES 参数。因此，当您从主数据库服务器发出 onparams 命令后，必须手动将辅助服务器的 LOGFILES 参数更改为适当值。最后，为使更改生效，您必须对主数据库服务器上的根数据库空间执行 0 级备份。

如果在主数据库服务器上添加逻辑日志文件，那么该文件在您执行 0 级备份后立即可用并标记为 F 。在辅助数据库服务器上新的逻辑日志文件仍标记为 A。然而，这种情况不会阻止辅助数据库服务器写入该文件。

管理虚拟处理器

虚拟处理器的数量对数据复制没有影响。可以分别配置和调整对中的每个数据库服务器。

管理共享内存

如果更改了一个数据库服务器上的共享内存 ONCONFIG 参数，那么必须同时对另一个数据库服务器上的共享内存 ONCONFIG 参数进行相同更改。有关进行该更改的过程，请参阅更改 *HDR* 复制对的配置参数 在第325页。

设置 SMX 连接

服务器多路复用器组（SMX）为一通信接口，支持高可用性环境中服务器之间的多路复用网络连接。SMX 在数据库服务器实例之间提供可靠的、安全的、高性能的通信机制。

降低服务器间的延迟

可通过增加服务器间用于 SMX 连接的管道数来降低高可用性服务器间的延迟。SMX_NUMPIPES 配置参数用于设置管道数。

获取 SMX 统计信息

可以使用 onstat 实用程序或系统监视界面（SMI）表来检视 SMX 连接统计信息或 SMX 会话统计信息。

要查看 SMX 连接统计信息，请使用 onstat -g smx 命令。

要查看 SMX 会话统计信息，请使用 onstat -g smx ses 命令。

设置服务器之间 SMX 活动的等待时间

可设置 SMX_PING_INTERVAL 和 SMX_PING_RETRY 配置参数来调整高可用性集群中辅助服务器等待主服务活动的时间间隔。使用 SMX_PING_INTERVAL 配置参数来指定超时间隔的秒数，即辅助服务器在此期间通过 SMX 连接等待主服务器活动。

使用 SMX_PING_RETRY 配置参数来指定辅助服务器未接收到主服务器响应时重复 SMX_PING_INTERVAL 参数指定的超时间隔的最大次数。如果达到最大数也未接收到响应，那么辅助服务器将在 online.log 中输出错误消息并关闭服务器多路复用器组（SMX）连接。

通过 SMX 连接压缩数据

将数据从源数据库服务器发送到目标数据库服务器之前，使用 SMX_COMPRESS 配置参数来指定数据库服务器使用的压缩级别。网络压缩可节省慢速链接的网络带宽，但会使用较多的 CPU 来压缩和解压缩数据。

相关链接

《SinoDB 管理员参考》: SMX_COMPRESS 配置参数
《SinoDB 管理员参考》: SMX_PING_INTERVAL 配置参数
《SinoDB 管理员参考》: SMX_NUMPIPES 配置参数
《SinoDB 管理员参考》: SMX_PING_RETRY 配置参数
《SinoDB 管理员参考》: onstat -g smx 命令

将索引复制到 HDR 辅助数据库服务器

如果启用索引页日志记录，索引会自动复制到 HDR 辅助数据库服务器上（请参阅索引页日志记录 在第303页）。如果禁用索引页日志记录，且 HDR 辅助数据库服务器上的索引已损坏而必须重建，那么可以使用以下方法之一：

- 手动将索引从主服务器复制到辅助服务器。
- 使辅助服务器自动复制索引（如果您使辅助服务器能够执行此操作）。

要使辅助服务器能够自动复制索引，请执行以下操作之一：

- 将 onmode -d idxauto 设置为 on。
- 将 DRIDXAUTO 配置参数的值设置为 1。

设置这些值的任意一个后，当辅助数据库服务器上的某个线程检测到损坏的索引时，索引将自动复制到辅助数据库服务器。重新启动索引复制可能需要 DRTIMEOUT 配置参数中指定的时间（以秒计）。

有时，可能希望手动复制索引，例如：由于表被锁定，所以想要延迟索引修复。如果希望能够在 HDR 辅助服务器上手动复制索引，请关闭自动复制功能。

要关闭自动索引复制功能，请执行以下操作之一：

- 将 onmode -d idxauto 设置为 off。
- 将 DRIDXAUTO 配置参数设置为 0。

如果 onmode -d idxauto 设置为 off 或 DRIDXAUTO 设置为 0 ，并且辅助服务器检测到损坏的索引，那么可以通过发出以下格式的 onmode -d index 命令在 HDR 辅助服务器上手动复制索引：onmode -d index database:[ownername].table#index

例如：onmode -d index cash_db:user_dx.table_12#index_z

在分段索引具有一个损坏分段的情况下，onmode -d idxauto 选项仅传输单个受影响的分段，而 onmode -d index 选项将会传输整个索引。

> 重要： 可以使用 onmode 命令或 DRIDXAUTO 配置参数来打开或关闭自动索引复制功能。如果使用 onmode 命令，那么就不需停止并重新启动数据库服务器。使用 DRIDXAUTO 参数时，数据库服务器将使用您指定的设置值重新启动。onmode 命令不会更改 DRIDXAUTO 值。如果使用 onmode 命令，那么您必须手动更改 DRIDXAUTO 的值。

辅助服务器生成的 online.log 文件包含所有已复制索引的信息。

相关链接

《SinoDB 管理员参考》: onmode -d 命令: 使用数据复制来复制索引

《SinoDB 管理员参考》: DRIDXAUTO 配置参数

调整 HDR 服务器对中的 LRU 清空和自动调节

当服务器是为 HDR 配置时，由辅助数据库服务器触发的检查点是非阻塞的。这些类型的检查点很少出现。如果辅助数据库服务器触发了非阻塞检查点，那么将阻塞主服务器上的事务，以确保辅助服务器的完整性不受影响。如果在您系统上的辅助数据库服务器触发了非阻塞检查点，那么必须将主服务器上的 LRU 清空调整得更加积极，以减少事务阻塞。

要增加 LRU 清空，请减少 BUFFERPOOL 配置参数中 lru_min_dirty 和 lru_max_dirty 的值。

可以独立地在每个 HDR 节点上打开或关机自动 LRU 调整。每台 HDR 数据库服务器上的设置可以不同。有关关闭自动 LRU 调整的信息，请参阅打开或关闭自动 *LRU* 调整 在第273页.

有关 LRU 调整的更多信息，请参阅《*SinoDB*® 性能指南》。

克隆主服务器

可使用 ifxclone 实用程序来执行一步式服务器安装，以使用最低设置或配置来克隆高可用性集群中的主服务器。

可使用 ifxclone 实用程序来创建以下数据库服务器类型之一：

- 独立服务器
- 远程独立（RS）辅助服务器
- 高可用性数据复制（HDR）辅助服务器
- 共享磁盘（SD）辅助服务器
- Enterprise Replication 服务器

使用 ifxclone 实用程序，数据库管理员可以快速、轻松和安全地通过正在运行的 SinoDB® 实例创建克隆服务器，而无需备份源服务器上的数据以及将其传输并复原到克隆服务器上。使用 ifxclone 实用程序，备份和复原过程可以同时开始，而无需对磁盘或磁带执行数据读取或写入。

数据通过使用服务器多路复用器组（SMX）连接的网络从源服务器传输到目标服务器。

您可以通过从脚本调用 ifxclone 实用程序来自动创建克隆实例。

创建主服务器的克隆

使用 ifxclone 实用程序来创建主服务器的克隆。

下面是创建服务器克隆的一般步骤：

1. 确认源服务器上的 ENABLE_SNAPSHOT_COPY 配置参数设置为 1。

2. 确认 CDR_AUTO_DISCOVER 配置参数设置为 1，并且在所有集群服务器上均有设置 REMOTE_SERVER_CFG 文件。
3. 在目标服务器上设置以下环境变量：
   - INFORMIXDIR
   - INFORMIXSERVER
   - ONCONFIG
   - INFORMIXSQLHOSTS
4. 在目标服务器上，创建存在于源服务器上的所有块。可使用 --createchunkfile 选项 (-k) 在目标服务器上自动创建熟块 。请遵循以下步骤创建这些块：
   a) 在源服务器上，运行 onstat -d 命令以显示块列表
   b) 在目标服务器上，以用户 informix 的身份登录，并使用命令 touch、chown 和 chmod 创建这些块。例如，要创建名为 /usr/informix/chunks/rootdbs.chunk 的块，请执行以下步骤：

```
$ su informix
Password:
$ touch /usr/informix/chunks/rootdbs.chunk
$ chown informix:informix /usr/informix/chunks/rootdbs.chunk
$ chmod 660 /usr/informix/chunks/rootdbs.chunk
```

   c) 对 onstat -d 命令报告的每个块重复执行上一步中的所有命令。
5. 在目标服务器主机上创建 sqlhosts 文件，并将目标服务器的连接性信息添加到该文件中。
6. 在源服务器上，运行带有 cdr add trustedhost 参数（包括目标服务器的可信主机信息）的 admin() 或 task() 函数。
7. 在仍然以用户 informix 的身份登录的情况下，使用 --autoconf 选项 (-a) 和在启动克隆服务器的目标系统上的其他适当参数来运行 ifxclone 实用程序。

   注： 如果手动添加可信主机信息，请不要使用 --autoconf 选项；必须手动配置可信主机和 sqlhosts 文件信息。

示例：克隆主服务器

此例中，有以下系统：

- server_1 是源服务器，其 sqlhosts 文件具有以下条目：

```
#dbservername  nettype   hostname           servicename  options
 server_1      onsoctcp  host1.example.com  123
```

- server_1 具有以下配置参数设置：
  - ENABLE_SNAPSHOT_COPY 1
  - CDR_AUTO_DISCOVER 1
  - REMOTE_SERVER_CFG authfile.server_1
- server_1 具有以下可信主机条目：

```
#trustedhost
host1
host1.example.com
host2
host2.example.com
```

- server_2 是目标服务器，位于 host2.example.com 上，并使用端口 456。

1. 在目标服务器上，以用户 informix 的身份登录，并使用 touch、chown 和 chmod 命令来创建块、更改块的所有者，以及块的更改权限。块路径必须与源服务器上块的路径匹配。可以使用 --createchunkfile 选项 (-k) 在目标服务器上自动创建熟块。

2. 以用户 informix 身份，在目标服务器上运行 ifxclone 实用程序：

```
ifxclone -T -S server_1 -I host1.example.com -P 123 -t server_2
         -i host2.example.com -p 456 -a -k
```

ifxclone 实用程序修改源服务器上的 sqlhosts 文件，并在新的目标服务器上创建文件副本。

```
#dbservername  nettype   hostname           servicename  options
 server_1      onsoctcp  host1.example.com  123
 server_2      onsoctcp  host2.example.com  456
```

ifxclone 实用程序还会将源服务器上的可信主机文件传播到目标服务器上。

相关链接

《*SinoDB* 管理员参考》：*ifxclone* 实用程序

辅助服务器上的数据库更新

您可以启用连接到辅助服务器的应用程序来更新数据库数据。如果在辅助服务器上启用写操作，那么会将 DELETE、INSERT、MERGE 和 UPDATE 操作传播到主服务器。

使用 UPDATABLE_SECONDARY 配置参数来控制辅助服务器是否可以更新数据，并可配置更新操作使用的连接数。

辅助服务器上同时支持数据定义语言（DDL）语句和数据操作语言（DML）语句。

所有可更新辅助服务器上都支持 dbimport 实用程序。

在 HDR 辅助服务器上或共享磁盘（SD）辅助服务器上不可使用 dbexport 实用程序。仅当远程独立（RS）辅助服务器设置为停止应用日志文件时，才可支持 dbexport 实用程序。使用 STOP_APPLY 配置参数来停止日志文件的应用。

所有可更新辅助服务器上都支持 dbschema 实用程序。

在只读辅助服务器上也支持 dbschema 实用程序。但是，dbschema 实用程序在这些服务器上运行时，会显示警告消息。

使用 DDL 或 DML 的大多数应用程序可在高可用性集群中的任一辅助服务器上运行；但是，不支持以下 DDL 语句：

- CREATE DATABASE（无日志记录）
- CREATE EXTERNAL TABLE
- CREATE RAW TABLE
- CREATE TEMP TABLE（有日志记录）
- CREATE XADATASOURCE
- CREATE XADATASOURCE TYPE
- DROP XADATASOURCE
- DROP XADATASOURCE TYPE
- UPDATE STATISTICS

在集群环境中，可更新辅助服务器上不支持 SET CONSTRAINTS、SET INDEXES 和 SET TRIGGERS 语句。对于辅助服务器数据库中表对象的 UPDATE 操作，不会重定向 SET Database Object Mode 语句指定的任何会话层索引、触发器或约束模式。

仅当辅助服务器映像和主服务器映像匹配时，客户端应用程序才可在辅助服务器上插入、更新和删除行。支持以下数据类型：

- BIGINT
- BIGSERIAL
- BLOB

- BOOLEAN
- BSON
- BYTE (stored in the table)
- CHAR
- CLOB
- DATE
- DECIMAL
- DATETIME
- FLOAT
- INT
- INT8
- INTERVAL
- JSON
- MONEY
- NCHAR
- NVCHAR
- SERIAL
- SERIAL8
- SMALLFLOAT
- SMALLINT
- TEXT (stored in the table)
- VARCHAR

不支持存储于 BLOB 空间中的 BYTE 和 TEXT 数据类型，因为不能复制 BLOB 空间数据。

如果以下数据没有指向另一分区的指针引用，那么也支持它们：

- COLLECTION
- LIST
- LVARCHAR
- MULTISET
- ROW
- SET
- UDTVAR

任何主服务器和辅助服务器映像之间的差异都会导致 SQL 错误并且回滚所有更改。

您不能在 HDR 辅助服务器、远程独立（RS）辅助服务器或共享磁盘（SD）辅助服务器上使用以下实用程序：

- archecker
- dbload
- ondblog
- onload
- onparams
- onspaces
- onunload

Windows™ 环境不支持 SD 辅助服务器。

在配置为更新的辅助服务器上不支持字节范围锁定。辅助服务器上的字节范围锁定会升级为全对象锁。

复制智能大对象

使用可更新辅助服务器时，可能会收到以下一条或或多条错误消息：

- 12014

- 12015
- 12233

这些错误通常表示智能大对象文件描述符有问题。以下任一状况都可导致这些错误：

- 在提交事务之前，将智能大对象标识传递到另一事务或进程。因为所有对象（包括智能大对象）在事务提交之前都不会提交，所以不允许其他事务使用智能大对象。特别是脏读取可以访问锁定的智能大对象。
- 智能大对象在开启之后没有关闭。在事务结束后，必须关闭辅助服务器上的所有智能大对象，尤其是那些已创建并随后回滚了事务的智能大对象。如果智能大对象文件标识保持打开状态，将导致在会话终止之前内存会保持已分配状态。
- 另一进程删除了主服务器上的智能大对象。共享锁未自动地从辅助服务器传播到主服务器，因此其他辅助服务器可能会访问主服务器上实际已删除的智能大对象。这些访问将一直有效，直到在辅助服务器上重放包含该删除操作的日志记录或主服务器更新辅助服务器。

处理脏读取信息时，可能还会返回其他三个错误代码。

- -126 (ISAM error: bad row id)
- -244 (SQL error:Could not do a physical-order read to fetch next row)
- -937

如果收到以上任一代码，请重试您的查询。

辅助服务器上的 LOCK TABLE 语句行为

可以从高可用性集群中的可更新服务器设置对表的互斥锁定。对于辅助服务器请求的互斥模式锁定，会话可以读取表但是不能更新表。这种行为类似于辅助服务器上的共享访问模式，也就是说，当一个会话在给定表上有互斥锁，那么其它会话就不能获取该表上的共享或互斥锁。

在只读辅助服务器上，发出 LOCK TABLE 语句的会话不会锁定表，并且数据库服务器也不会向客户端返回错误。

集群中的共享模式锁定的行为和独立服务器上的相同。LOCK TABLE 语句运行成功之后，用户可以读取，但在释放锁定之前不能修改该表。

相关链接

高可用性集群中的 *XA* 在第423页

辅助服务器上的隔离级别

所有类型的辅助服务器上都支持以下语句：

```
Set isolation to committed read
Set isolation to committed read last committed
```

设置 Committed Read 隔离的辅助服务器上可以读取本地提交的数据。对于在主服务器上已提交并在辅助服务器上可用且已提交的数据，它们也可读取。连接到辅助服务器上的应用程序可接收当前已在辅助服务器上提交的数据。请参阅 设计数据复制组客户端 在第324页，以获取有关连接到运行数据复制的数据库服务器的客户端的设计注意事项的更多信息。

辅助服务器上的缺省隔离级别是 Dirty Read；但设置显式隔离级别将启用正确的隔离级别：Dirty Read、Committed Read 或 Committed Read Last Committed。

不支持 Repeatable Read 和 Cursor Stability 隔离级别。带有 Repeatable Read 和 Cursor Stability 级别的 SET ISOLATION 语句将被忽略。

启动辅助服务器之后，只有在启动检查点打开的所有事务提交或回滚时，客户端应用程序才与服务器连接。

如果禁用 UPDATABLE_SECONDARY 配置参数（通过取消配置或设置为零），那么辅助数据复制服务器将为只读。在这种情况下，辅助服务器上只能使用 DIRTY READ 或 READ UNCOMMITTED 事务隔离级别。

如果启用了 UPDATABLE_SECONDARY 参数（通过将其设置为大于零的有效连接数），那么辅助数据复制服务器将可支持 COMMITTED READ、COMMITTED READ LAST COMMITTED 或 COMMITTED READ 事务隔离级别，或

USELASTCOMMITTED 会话环境变量。只有 SQL 的 DML 语句（DELETE、INSERT、UPDATE 和 MERGE 语句）和 dbexport 实用程序 支持可更新辅助服务器上的写操作。（除 UPDATABLE_SECONDARY 配置参数之外，还必须设置 STOP_APPLY 和 USELASTCOMMITTED 配置参数，才能在辅助数据服复制服务器上启用 dbexport 的写操作。）

使用 onstat -g ses 或 onstat -g sql 来查看隔离级别设置。有关更多信息，请参阅 《*SinoDB®* 管理员参考》 。

设置锁定方式

如同在主服务器上一样，在辅助服务器上使用 SET LOCK MODE TO WAIT 或 SET LOCK MODE TO WAIT *n* 语句来设置会话的锁定等待超时值。当执行来自辅助服务器的更新时，在主服务器上为当前会话创建的代理线程将使用 SET LOCK MODE 设置的值。如果 SET LOCK MODE 值大于 ONCONFIG 的 DEADLOCK_TIMEOUT 参数值，就会使用 DEADLOCK_TIMEOUT 的值。

高可用性集群辅助服务器上的瞬态类型

无论高可用性集群的辅助服务器是只读的或可更新的，都可在其上使用瞬态未命名复合数据类型（ROW、SET、LIST 和 MULTISET）。在辅助服务器上支持以下使用瞬态类型的操作类型：

- 使用瞬态类型的 SQL 查询
- 使用派生表、集合子查询和 XML 函数的 SQL 查询（这些语句都隐式地使用瞬态类型）
- 由使用瞬态类型的 CREATE TEMP 语句创建的临时表

有关复合数据类型的信息，请参阅 《*SinoDB® SQL* 指南*:* 参考》 和 《*SinoDB® SQL* 指南*:* 语法》。

行版本控制

使用行版本控制可确定行是否改变并检测冲突。在启用行版本控制后，表的每一行将配置为包含一个效验和与一个版本号。当首次插入行时，会自动生成校验和，并且版本设置为 1。每次更新该行时，版本都会加 1，但校验和不变。如果删除了一行并在表中重新插入另一行，那么通过行版本控制就可识别出新插入的行不同于删除的行。通过比较辅助服务器和主服务器之间的行校验和与行版本，就可检测出数据冲突。

Web 应用程序可以使用版本列以确定之前检索对象中包含的信息是否仍处于最新状态。例如，Web 应用程序可能会向客户显示待售商品。当客户决定购买某个商品时，应用程序可以检查商品行的版本列以确定有关该商品的信息是否改变。

如果环境中的客户端应用程序可以更新辅助服务器上的数据，请使用行版本控制以减少网络使用，特别是表中包含许多列。否则，辅助服务器上的所有行和主服务器上的所有行进行比较，以确定是否发生更新。

要将行版本控制添加到现有表，请使用以下语法：

```
ALTER TABLE tablename add VERCOLS;
```

同样地，您可以使用以下语法来删除表中的行版本控制：

```
ALTER TABLE tablename drop VERCOLS;
```

要创建具有行版本控制的新表，请使用以下语法：

```
CREATE TABLE tablename (
   Column Name   Datatype
   Column Name   Datatype
   Column Name   Datatype
   ) with VERCOLS;
```

启用了行版本控制之后，每次行更新，ifx_row_version 都会递增 1；但是，Enterprise Replication 执行的行更新不会递增行版本。要更新使用 Enterprise Replication 的服务器上的行版本，您必须在复制参与者定义中包含 ifx_row_version 列。

高可用性群集的备份和复原

大多数的备份和复原操作都不能在辅助服务器上执行。

必须先在服务器上执行冷复原，然后才能将该服务器建立为 HDR（高可用性数据复制）或 RS（远程独立）辅助服务器。

将服务器设置为 HDR 或 RSS 辅助服务器之后，您只能执行以下备份和复原操作：

- 当设置高可用性集群时，可在 HDR 或 RS 辅助服务器上执行逻辑复原。
- 可在 RS 辅助服务器上执行外部备份。有关更多信息，请参阅 《*SinoDB*® 备份和还原指南》。

主服务器的冷复原期间，必须关闭辅助服务器，但是 SD 辅助服务器在关闭并重新启动之后，可以在热复原期间联机。

相关链接

首次启动 *RS* 辅助服务器 在第303页
首次启动 *HDR* 在第298页
数据损坏后恢复共享磁盘集群 在第419页

更改数据库服务器模式

如果更改高可用性集群中数据库服务器的模式，复制将停止。

使用 onmode 实用程序更改数据库服务器模式。

下表总结了更改主数据库服务器模式产生的影响。

表 47: 主数据库服务器上的模式更改

在主服务器上	在辅助服务器上	重新启动 HDR
从任何模式更改为脱机模式： (onmode -k)	辅助服务器显示：DR: Receive error. HDR 关闭。 模式保持为只读。 如果 DRAUTO 设置为 0，模式将保持为只读。 如果 DRAUTO 设置为 1，那么辅助服务器将切换到标准类型，并可接受更新。（如果 DRAUTO 设置为 2 ，那么当旧的主服务器发生故障时，在连接结束时，辅助服务器将立即变为主服务器。）	将其视为主服务器的故障来处理。根据主数据库服务器脱机时您对辅助数据库服务器所进行的操作不同，可能有两种不同的情况。请参阅以下部分以获取信息： • 辅助服务器未更改为主服务器 • 辅助服务器已自动更改为主服务器 请参阅主服务器发生故障时重新启动 在第417页。
更改为联机、静默或管理模式： (onmode -s / onmode -u) (onmode -j)	辅助服务器不会收到错误。 HDR 保持打开。 模式保持为只读。	在主服务器上使用 onmode -m 。

下表总结了更改辅助数据库服务器模式产生的影响。

表 48: 辅助数据库服务器上的模式更改

在辅助服务器上	在主服务器上	重新启动 HDR
只读脱机 (onmode -k)	主服务器显示：DR: Receive error. HDR 关闭。	将其作为辅助服务器的故障来处理。请遵循 辅助服务器发生故障时重新启动 *HDR* 或 *RS* 集群 在第416页中的过程。
静默 (onmode -s)	主服务器不会收到错误。 HDR 保持打开。	在辅助服务器上使用 onmode -m 。

管理模式在 HDR 辅助数据库服务器上的操作方式与其在主数据库服务器上是一样的。

更改数据库服务器类型

可更改主数据库服务器或辅助数据库服务器的类型

可更改主数据库服务器或辅助数据库服务器的类型。

仅当辅助数据库服务器关闭 HDR 时，才可将数据库服务器类型从辅助更改为标准。当数据复制到主数据库服务器的连接断开或在辅助数据库服务器上的数据复制失败时，关闭 HDR。 当您使标准数据库服务器脱机并随后将其恢复联机时，它不会尝试连接至复制对中的另一台数据库服务器。

使用以下命令切换类型：

- hdrmksec.[sh|bat] 和 hdrmkpri.[sh|bat] 脚本

要使用 hdrmkpri 和 hdrmksec 脚本切换数据库服务器类型，请执行以下操作：

1. 关闭主数据库服务器（ServerA）：onmode -ky
2. 辅助数据库服务器（ServerB）联机时，在 UNIX™ 上运行 hdrmkpri.sh 脚本，或在 Windows™ 上运行 hdrmkpri.bat 脚本，将 ServerB 切换为主数据库服务器。
3. 对于 ServerA，在 UNIX™ 上运行 hdrmksec.sh 脚本，或在 Windows™ 上运行 hdrmksec.bat 脚本，将 ServerA 切换为辅数据库服务器。
4. 使 ServerB （主数据库服务器）联机。

还可以使用以下命令切换服务器类型：

1. 运行以下命令，将 ServerA 更改为主服务器：

   ```
   onmode -d make primary ServerA
   ```

   此命令使 ServerA 成为主服务器，并将集群中其他任何辅助服务器重定向为指向新的主服务器。此命令还关闭旧的 HDR 主服务器（ServerB），因为高可用性环境里只能存在一个主服务器，
2. 运行以下命令，将 ServerB 初始化为 HDR 辅助服务器：
   - 在 UNIX™ 系统上：

     ```
     $INFORMIXDIR/bin/hdrmksec.sh ServerB
     ```
   - 在 Windows™ 系统上：

     ```
     hdrmksec.bat ServerB
     ```

相关链接

手动切换 在第408页

防止 HDR 服务器上的检查点阻塞

在 HDR 辅助服务器上，检查点处理必须等到缓冲池清仓完成。您可以在 HDR 辅助服务器上配置非阻塞检查点，以便在检查点处理完成之前，将从主服务器发送的日志数据储存或暂存在目录中。

可通过 LOG_STAGING_DIR 和 LOG_INDEX_BUILDS 配置参数，在 HDR 辅助服务器上配置非阻塞检查点。在配置了非阻塞检查点之后，从主服务器发送的日志数据会暂存在 LOG_STAGING_DIR 配置参数所指定的目录中。在 HDR 辅助服务器完成检查点处理后，将读取并应用存储在暂存区域的日志数据。如暂存目录为空时，HDR 辅助服务器将读取并应用从主服务器接收的日志数据。

可通过在 HDR 辅助服务器上设置 LOG_STAGING_DIR 配置参数，并同时在主服务器和 HDR 辅助服务器上设置 LOG_INDEX_BUILDS 配置参数来启用非阻塞检查点。LOG_INDEX_BUILDS 的值在主服务器和 HDR 辅助服务器上必须相同。

HDR 辅助服务器在遇到检查点时，将进入缓冲模式。在缓冲模式中，辅助服务器将来自主服务器的任何日志页数据暂存到暂存目录中的文件。

HDR 辅助服务器在完成检查点处理后，进入排出模式。在此模式下，HDR 辅助服务器从暂存文件中读取数据，同时从主服务器接收新数据。暂存区域为空后，HDR 辅助服务器将恢复正常运行。

日志记录在 HDR 服务器上的存储位置

HDR 辅助服务器将在 LOG_STAGING_DIR 指定的目录中额外创建名为 ifmxhdrstage_## 的目录，其中 ## 为 SERVERNUM 指定的实例。这些目录用于存储在检查点处理期间从主服务器发送的逻辑日志。ifmxhdrstage_## 中的文件不再需要时将清除。

非阻塞检查点和辅助服务器更新的交互

您必须考虑辅助服务器更新与 HDR 辅助服务器上的非阻塞检查点之间的交互。如果 HDR 辅助服务器收到更新请求，在 HDR 辅助服务器处理相应的日志记录之前，将不会应用更新。如果 HDR 辅助服务器上启用了非阻塞检查点，那么由于检查点处理导致日志数据在辅助服务器上暂存，因此可能会发生辅助服务器上的数据应用延迟。

查看 HDR 服务器上非阻塞检查点的统计信息

使用 onstat 实用程序查看在主服务器和 HDR 辅助服务器上有关非阻塞检查点的信息。

要查看有关暂存日志的信息，请使用 onstat -g dri ckpt 命令。

有关 onstat -g dri ckpt 的输出示例，请参阅《*SinoDB*® 管理员参考》中有关 onstat 实用程序的信息。

监视 HDR 状态

监视数据库服务器的 HDR 状态以确定以下信息：

- 数据库服务器类型（主服务器、辅助服务器或标准服务器）
- 复制对中另一个数据库服务器名称
- HDR 是否打开
- HDR 参数的值

命令行实用程序

每次运行 onstat 命令时显示的头信息均有字段指示数据库服务器是正在作为主数据库服务器还是辅助数据库服务器运行。

以下示例显示作为复制对中主数据库服务器且处于联机模式的数据库服务器的头信息：

```
Sinoregal Dynamic Server Version 12.10.UC1 -- online(Prim) -- Up 45:08:57
```

以下示例显示作为复制对中辅助数据库服务器且处于只读模式的数据库服务器。

```
Sinoregal Dynamic Server Version 12.10.UC1 -- Read-Only (Sec) -- Up 45:08:57
```

以下示例显示不包含在 HDR 中的数据库服务器的头信息。该数据库服务器类型为标准类型。

```
Sinoregal Dynamic Server Version 12.10.UC1 -- online -- Up 20:10:57
```

onstat -g dri 选项

要获得完整的 HDR 监视信息，请运行 onstat -g dri 选项。显示以下字段：

- 数据库服务器类型（主类型、辅助类型或标准类型）
- HDR 状态（开启或关闭）
- 成对的数据库服务器
- 最近一次 HDR 检查点
- HDR 配置参数值

有关 onstat -g dri 输出的示例，请参阅 《*SinoDB*® 管理员参考》。

oncheck -pr 选项

如果您的数据库服务器正在运行 HDR，那么保留页 PAGE_1ARCH 和 PAGE_2ARCH 将保存 HDR 用于同步主数据库服务器和辅助数据库服务器的检查点信息。以下示例提供了相关 oncheck -pr 输出的示例。

```
Validating Sinoregal Database Server reserved pages - PAGE_1ARCH &
 PAGE_2ARCH
         Using archive page PAGE_1ARCH.

    Archive Level                 0
    Real Time Archive Began       01/11/95 16:54:07
    Time Stamp Archive Began      11913
    Logical Log Unique Id         3
    Logical Log Position          b018

    DR Ckpt Logical Log Id        3
    DR Ckpt Logical Log Pos       80018
    DR Last Logical Log Id        3
    DR Last Logical Log Page      128
```

SMI 表

sysdri 表提供有关数据库服务器高可用性数据复制状态的信息。

在《*SinoDB*® 管理员参考》中有关 sysmaster 数据库的这些主题中描述的 sysdri 表包含以下列：

type
  HDR 服务器类型

state
  HDR 服务器状态

name
  数据库服务器名称

intvl
  HDR 缓冲区清空的时间间隔

timeout
  网络超时

lostfound
  HDR 失物招领的路径名称

获取 RS 辅助服务器统计信息

使用 onstat 命令显示有关 RS 辅助服务器状态的信息。要显示 RS 辅助服务器的数量、有关已发送到 RS 辅助服务器的数据的信息，以及有关 RS 辅助服务器已确认接收的内容的信息，请使用 onstat -g rss 命令。

该命令具有显示单台服务器或多台辅助服务器的扩展信息的选项。有关 onstat -g rss 输出的示例，请参阅《*SinoDB*® 管理员参考》中有关 onstat 实用程序的信息。

移除 RS 辅助服务器

可通过发出以下命令，从高可用性集群中移除 RS 辅助服务器：

```
onmode -d delete RSS rss_servername
```

创建或更改 RS 辅助服务器上的密码

可以为 RS 辅助服务器创建密码，以便在创建集群时，提供主服务器和辅助服务器之间的认证。密码为可选的。密码仅在主服务器和辅助服务器首次连接时有效。

密码可以防止网络上不需要的数据库实例初始化与接收来自主服务器的事务数据。密码与 informix 用户密码无关。

可使用指定 RS 辅助服务器名称的相同命令来创建密码。每个 RS 辅助服务器的名称和密码可在主服务器 0 级备份之前或之后定义。

要设置 RS 辅助服务器的名称和密码，请在主服务器上运行命令 onmode -d add RSS *rss_ha_alias password*。在辅助服务器设置期间，当您在辅助服务器上运行命令 onmode -d RSS *rss_ha_alias password* 时，请含括密码。

仅当服务器未连接到高可用性环境时，您才可以更改密码。试图更改已连接的 RS 辅助服务器的密码将返回错误。

要在 RS 辅助服务器连接之前更改该服务器密码，请在主服务器上使用 onmode -d change RSS *password* 命令。

示例

以下命令建立 RS 辅助服务器 ServerB 并使用密码 s#cure。该服务器的 HA_ALIAS 配置参数设置为 ServerB。

在主服务器上：

```
onmode -d add RSS ServerB s#cure
```

在辅助服务器上：

```
onmode -d RSS ServerB s#cure
```

以下命令将未连接到主服务器的 RS 辅助服务器 ServerB 的密码更改为 s@fest。该服务器的 HA_ALIAS 配置参数设置为 ServerB。

```
onmode -d change RSS ServerB s@fest
```

在集群故障转移期间完成事务

可在高可用性集群环境中配置服务器，以便在主服务器故障转移之后继续处理事务。

故障的主服务器以外的任何其他服务器上运行的事务将继续运行。配置集群环境，以便：

- 在辅助服务器上运行的事务不受影响。
- 在成为主服务器的辅助服务器上运行的事务不受影响。

- 终止故障的主服务器上运行的事务。

智能大对象、XA 事务和在辅助服务器上运行 DDL 语句时，不支持故障转移后的事务完成。

当发生故障转移时，集群中的辅助服务器暂停运行用户事务直到新主服务器开始运行。故障转移后，辅助服务器将保存的事务重新发送到新主服务器。新主服务器将恢复执行来自尚存的辅助服务器的事务。

当分布式事务（跨多个数据库服务器的事务）正在运行时发生故障，此时在主服务器上正在运行的任何事务将终止。

当发生故障转移时，不管是手动还是由连接管理器自动指定，接收故障转移的数据库服务器必须是具有集群中所有尚存服务器的最高级日志重放的服务器。如果接收故障转移的数据库服务器不具有最高级的日志重放位置，那么集群中的所有事务都将终止并回滚。由于主服务器与 SD 辅助服务器读取相同的物理磁盘，所以故障转移到 SD 辅助服务器应先发生。如果故障转移服务器是 HDR 辅助服务器，那么 SD 辅助服务器将关闭。

为了获得最佳的连接管理器故障转移性能，请使用连接管理器配置文件中的 FOC ORDER=ENABLED 设置，并设置集群主服务器上的 HA_FOC_ORDER 配置参数。

配置服务器以使事务在故障转移后完成

可使用 FAILOVER_TX_TIMEOUT 配置参数来配置高可用性集群中的服务器，以便在故障转移之后完成事务。

FAILOVER_TX_TIMEOUT 的值指示在主服务器故障之后，回滚事务之前服务器等待的最大秒数。请在集群中所有服务器都设置相同的 FAILOVER_TX_TIMEOUT 值。例如，指定 20 秒完成事务，那么请将 onconfig 文件中 FAILOVER_TX_TIMEOUT 配置参数的值设置为 20。

要禁用故障转移之后完成事务，请将集群中所有服务器的 FAILOVER_TX_TIMEOUT 配置参数设置为 0。

相关链接

《SinoDB 管理员参考》: FAILOVER_TX_TIMEOUT 配置参数

通过连接管理器进行连接管理

连接管理器可控制高可用性集群的自动故障转移、 监视客户端连接并将请求定向到合适的数据库服务器、充当代理服务器处理客户端/服务器通信，并定出应用服务器与高可用性集群中的主服务器之间的连接优先次序。连接管理器支持高可用性集群、 复制集、 服务器集和网格。

数据库服务器的自动故障转移

如果连接管理器检测到高可用性集群的主服务器发生故障，它可以将辅助服务器提升为主服务器的角色。

如果使用多个连接管理器来管理集群故障转移，可以通过在集群的主服务器上的 onconfig 文件设置 HA_FOC_ORDER 配置参数来强制执行一致的故障转移策略。onconfig 文件中 HA_FOC_ORDER 配置参数的值会替换连接到主服务器的每个连接管理器的配置文件中的 FOC 参数的 ORDER 属性的值。

基于规则的连接重定向和负载平衡

客户端应用程序连接到连接管理器，就如同连接到数据库服务器。连接管理器从连接单元的每个服务器中收集工作负载的统计数据，并使用服务级别协议 (SLAs) 来管理客户端连接请求并将客户端连接请求定向到适当的服务器。当客户端应用程序通过重定向模式 SLA 发出连接请求时，连接管理器会返回一个数据库服务器的 IP 地址和端口号给客户端应用程序。然后，客户端应用程序使用该信息连接到指定的数据库服务器。

连接管理器的重定向发生在通信层，因此客户端应用程序不需要额外操作。连接管理器可以使用基于工作负载、 延迟、请求失败或请求积压的重定向策略。重定向也可以配置为循环发生。

重定向策略 SLAs 不支持使用数据服务器驱动程序（支持 JDBC 和 SQLJ 3.5.1 版或更早版本）或 SinoDB® 客户端软件开发工具包 (Client SDK) 3.00 或更早版本编译的应用程序的连接

代理服务器连接管理

代理模式 SLAs 和重定向模式 SLAs 是相似的；在这两种情况下，连接管理器从连接单元的服务器收集工作负载的统计数据，并控制哪些服务器接收客户端连接请求。当客户端应用程序通过代理模式 SLA 发出连接请求时，客户端/服务器通信将通过连接管理器进行。当数据库服务器位于防火墙后面时，连接管理器可以充当代理服务器来处理客户端/服务器通信。

代理策略 SLA 不像重定向政策 SLA 有版本限制。其支持使用任何版本的数据服务器驱动程序（支持 JDBC 和 SQLJ）或任何版本的 SinoDB® 客户端软件开发工具包 (Client SDK) 编译的应用程序的连接。

应用程序服务器的故障转移优先级

您可以将连接管理器安装在与应用程序服务器相同的主机上，然后定出每个应用程序服务器与高可用性集群的主服务器之间的连接优先级。这可以确保在部分网络故障时，最高优先级的应用程序服务器与集群的主服务器的保持连接。

相关链接

数据复制客户端的重定向与连接 在第409页
高可用集群的故障转移配置 在第405页
*DRDA*概述 在第66页
支持高可用性和可扩展性的组件 在第289页

配置连接管理器

要配置连接管理，必须安装软件、设置环境和连接信息、创建连接管理器配置文件，并运行 oncmsm 实用程序。

要配置并启动连接管理，请完成以下步骤 ：

1. 作为 SinoDB® 客户端软件开发工具包 (Client SDK) 安装的一部分，至少安装一个连接管理器。
   a) 如果连接管理器安装在未安装数据库服务器的主机上，则将每个连接管理器的主机 INFORMIXDIR 环境变量设置为连接管理器的安装目录。
2. 修改所有客户端、 数据库服务器和连接管理器主机上的 sqlhosts 文件的连接信息。
   a) 若 sqlhosts 文件位于 $INFORMIXDIR/etc 以外的主机目录中，则将主机 INFORMIXSQLHOSTS 环境变量设置为相应的 sqlhosts 文件位置。
   b) 如果连接管理器或客户端安装在未安装数据库服务器的主机上，则在每个主机上创建 sqlhosts 文件，并将每个连接管理器或客户端的主机 INFORMIXSQLHOSTS 环境变量设置为 sqlhosts 文件的位置。
3. 在连接管理器主机上为每个已安装的连接管理器创建一个配置文件。
   a) 若该配置文件在 $INFORMIXDIR/etc 以外的目录中，则将 CMCONFIG 环境变量设置为该文件的位置。
   b) 如果您定义了使用事务延迟或请求失败重定向策略的服务级别协议时，那么可运行cdr define qod 和 cdr start qod 命令来启动数据质量 (QOD) 监视。
4. 在集群数据库服务器上设置 onconfig 参数。
   a) 如果连接管理器控制高可用性集群的故障转移，请将所有受管集群数据库服务器上的 DRAUTO 配置参数设置为 3，并且为每个集群主服务器上的 HA_FOC_ORDER 配置参数设置一个故障转移顺序。
   b) 对于每个使用多个端口的数据库服务器，将 DBSERVERALIASES 配置参数设置为连接管理器和数据库服务器 sqlhosts 文件条目中列出的别名。
5. 可选： 配置与连接管理器一起安装的 cmalarmprogram 脚本。
6. 运行 oncmsm 实用程序来启动连接管理器。

相关链接

sqlhosts 信息 在第47页
《*SinoDB* 管理员参考》: *oncmsm* 实用程序
《*SinoDB* 管理员参考》: *onpassword* 实用程序

《*SinoDB* 管理员参考》: *DBSERVERALIASES* 配置参数
《*SinoDB* 管理员参考》: *DRAUTO* 配置参数
《*SinoDB* 管理员参考》: *HA_FOC_ORDER* 配置参数
《*SinoDB SQL* 指南: 参考》: *INFORMIXSQLHOSTS* 环境变量
《*SinoDB SQL* 指南: 参考》: *INFORMIXDIR* 环境变量
《*SinoDB Enterprise Replication* 指南》: *cdr define qod*
《*SinoDB Enterprise Replication* 指南》: *cdr start qod*

创建连接管理器配置文件

要配置连接管理器，您必须创建一个配置文件，然后运行 oncmsm 实用程序来载入配置文件。

连接管理器配置文件由两部分组成：

- 头，其中包含特定于连接管理器的连接管理器参数。
- 正文，由一个或多个连接单元部分组成，其中包含参数和特定于所定义的连接单元的属性。

以下步骤适用于所有连接单元类型：

1. 在安装了连接管理器的主机上的 $INFORMIXDIR/etc 目录中，创建一个 ASCII 文本文件。
2. 在该文件的第一行，指定 NAME 参数，其后为连接管理器的名称。
   连接管理器的名称在受管的连接单元的域中必须是唯一的。
   例如：

```
NAME my_connection_manager_1
```

3. 指定可选的头参数。
   例如：

```
NAME my_connection_manager_1
LOG 1
LOGFILE $INFORMIXDIR/tmp/my_cm_1.log
```

4. 通过指定至少一个连接单元类型来创建配置文件的正文。在连接单元类型后面紧跟着连接单位的名称，然后跟着左右大括号。
   例如：

```
NAME my_connection_manager_1
LOG 1
LOGFILE $INFORMIXDIR/tmp/my_cm_1.log

CLUSTER my_cluster
{
}
```

5. 将连接单元 INFORMIXSERVER 参数设置为连接单元参与者的 sqlhosts 文件条目。
   例如：

```
NAME my_connection_manager_1
LOG 1
LOGFILE $INFORMIXDIR/tmp/my_cm_1.log

CLUSTER my_cluster
{
    INFORMIXSERVER group_name
}
```

6. 如果连接管理器管理连接请求，请指定 SLA 参数、SLA 名称和 DBSERVER 属性。

例如：

```
NAME my_connection_manager_1
LOG 1
LOGFILE $INFORMIXDIR/tmp/my_cm_1.log

CLUSTER my_cluster
{
   INFORMIXSERVER group_name
   SLA sla_1 DBSERVERS=PRI
   SLA sla_2 DBSERVERS=HDR,SDS,RSS
}
```

7. 指定 FOC 参数和 PRIORITY 属性。如果连接管理器管理故障转移，也要指定 ORDER 属性。
例如：

```
NAME my_connection_manager_1
LOG 1
LOGFILE $INFORMIXDIR/tmp/my_cm_1.log

CLUSTER my_cluster
{
   INFORMIXSERVER group_name
   SLA sla_1 DBSERVERS=PRI
   SLA sla_2 DBSERVERS=HDR,SDS,RSS
   FOC ORDER=ENABLED \
       PRIORITY=1
}
```

8. 指定可选的 SLA 参数属性。
例如：

```
NAME my_connection_manager_1
LOG 1
LOGFILE $INFORMIXDIR/tmp/my_cm_1.log

CLUSTER my_cluster
{
   INFORMIXSERVER group_name
   SLA sla_1 DBSERVERS=PRI
   SLA sla_2 DBSERVERS=(HDR,SDS,RSS) \
             POLICY=ROUNDROBIN
   FOC ORDER=ENABLED \
       PRIORITY=1
}
```

9. 如果连接管理器管理多个连接单元，请将其他连接单元添加到配置文件的正文。
例如：

```
NAME my_connection_manager_1
LOG 1
LOGFILE $INFORMIXDIR/tmp/my_cm_1.log

CLUSTER my_cluster
{
   INFORMIXSERVER group_name
   SLA sla_1 DBSERVERS=PRI
   SLA sla_2 DBSERVERS=(HDR,SDS,RSS) \
             POLICY=ROUNDROBIN
   FOC ORDER=ENABLED \
       PRIORITY=1
}
```

```
CLUSTER my_cluster_2
{
   INFORMIXSERVER group_name_2
   SLA sla_3 DBSERVERS=PRI
   SLA sla_4 DBSERVERS=(HDR,SDS) \
             POLICY=ROUNDROBIN
   FOC ORDER=ENABLED \
       PRIORITY=1
}
```

连接管理器配置文件的参数和格式
以下示例显示了连接管理器配置文件的格式，以及每个连接单元类型可设置的参数和属性。

```
#****************************** HEADER ******************************

NAME connection_manager_instance_name

# Optional Parameters
MACRO name_1=value
MACRO name_2=value
MACRO name_n=value_n
.
.
.
LOCAL_IP ip_address_list
LOG value
LOGFILE path_and_filename
DEBUG value
CM_TIMEOUT seconds
EVENT_TIMEOUT seconds
SECONDARY_EVENT_TIMEOUT seconds
SQLHOSTS value

#****************************** BODY ******************************

# Replicate set connection-unit example

REPLSET unit_name_1
{
   INFORMIXSERVER sqlhosts_group_names_list

   #Optional Parameters and Attributes
   SLA sla_name_1 DBSERVERS=value_list \

                  #Optional SLA Attributes
                  MODE=value \
                  USEALIASES=value \
                  POLICY=value \
                  WORKERS=number_of_threads \
                  HOST=host_name \
                  NETTYPE=network_protocol \
                  SERVICE=service_name \
                  SQLHOSTSOPT="options"
   SLA sla_name_2 DBSERVERS=value_list ...
   SLA sla_name_n DBSERVERS=value_list ...
   .
   .
   .
}

# High-availability cluster connection-unit example
```

```
CLUSTER unit_name_2
{
   INFORMIXSERVER sqlhosts_group_name
   FOC ORDER=value \
       PRIORITY=value \
       TIMEOUT=value

   #Optional Parameters and Attributes
   SLA sla_name_1 DBSERVERS=value_list \

                  #Optional SLA Attributes
                  MODE=value \
                  USEALIASES=value \
                  POLICY=value \
                  WORKERS=number_of_threads \
                  HOST=host_name \
                  NETTYPE=network_protocol \
                  SERVICE=service_name \
                  SQLHOSTSOPT="options"
   SLA sla_name_2 DBSERVERS=value_list ...
   SLA sla_name_n DBSERVERS=value_list ...
.
.
.
   CMALARMPROGRAM path_and_filename
}

# Server set connection-unit example

SERVERSET unit_name_3
{

   INFORMIXSERVER sqlhosts_group_names_and_standalone_servers_list

   #Optional Parameter and Attributes
   SLA sla_name_1 DBSERVERS=value_list \

                  #Optional SLA Attributes
                  MODE=value \
                  USEALIASES=value \
                  POLICY=value \
                  WORKERS=number_of_threads \
                  HOST=host_name \
                  NETTYPE=network_protocol \
                  SERVICE=service_name \
                  SQLHOSTSOPT="options"
   SLA sla_name_2 DBSERVERS=value_list ...
   SLA sla_name_n DBSERVERS=value_list ...
   .
   .
   .
}

# Grid connection-unit example

GRID unit_name_4
{
   INFORMIXSERVER server_list

   #Optional Parameter and Attributes
   SLA sla_name_1 DBSERVERS=value_list \
```

```
                #Optional SLA Attributes
                MODE=value \
                USEALIASES=value \
                POLICY=value \
                WORKERS=number_of_threads \
                HOST=host_name \
                NETTYPE=network_protocol \
                SERVICE=service_name \
                SQLHOSTSOPT="options"
   SLA sla_name_2 DBSERVERS=value_list ...
   SLA sla_name_n DBSERVERS=value_list ...
   .
   .
   .
}

# Connection Unit n
connection_unit_type unit_name_n
{
   INFORMIXSERVER values
   .
   .
   .
}
#*************************************************************************
```

提示： 为提高可读性，可以用反斜杠（ \ ）续行字符来分隔较长的配置文件行。

下面的示例显示了一个使用两个文本行但作为一行读取的宏定义：

```
MACRO servers=node1,node2,node3,node4,node5,node6,node7,node8, \
              node9,node10,node11,node12,node13,node14,node15
```

CMALARMPROGRAM 连接管理器配置参数
CMALARMPROGRAM 参数用来指定故障转移处理发生错误时要运行的程序或脚本的路径和文件名。

语法
语法
CMALARMPROGRAM*path_and_filename*

使用

CMALARMPROGRAM 参数是可选的，并受到以下连接单元的支持 ：

- CLUSTER
- GRID
- REPLSET
- SERVERSET

如果连接管理器无法找到能够接收故障转移的服务器，它将以增量间隔（最多 60 秒）搜索 ORDER 属性的服务器，最长两天。如果故障转移处理在尝试八次后失败，连接管理器将调用 CMALARMPROGRAM 参数所指定的程序。前八次故障转移尝试大约需要一分钟。

在使用 cmalarmprogram 脚本之前，您必须编辑该文件，并设置脚本参数。

ALARMADMIN 与 ALARMPAGER 参数用于确定发送到指定的电子邮件地址的连接管理器事件警报的级别。

表 49: 连接管理器事件警报级别

连接管理器事件警报级别	警报类型
0 （缺省值）	无
1	不重要的信息警报
2	信息警报
3	需要注意的警报
4	紧急警报
5	致命错误警报

示例

在以下示例中，如果故障转移处理在尝试八次后失败，将调用 cmalarmprogram.sh：

```
CMALARMPROGRAM $INFORMIXDIR/etc/cmalarmprogram.sh
```

相关链接

FOC 连接管理器配置参数 在第350页

CM_TIMEOUT 连接管理器配置参数

CM_TIMEOUT 参数指定数据库服务器集群等待从连接管理器（故障转移仲裁者）接收事件的秒数。如果在指定的时间内集群都未收到任何事件，那么主服务器将具有最高优先级的连接管理器提升为故障转移仲裁者的角色。

语法

语法

CM_TIMEOUT { 60 *seconds* }

使用

CM_TIMEOUT 参数是可选的，并且适用于 CLUSTER 连接单元。

如果未指定 CM_TIMEOUT 参数，那么超时设置为 60 秒。

示例

在以下示例中，如果在 100 秒之内，高可用性集群的服务器都未从当前的故障转移仲裁者接收到任何事件，那么集群主服务器将把活跃的连接管理器提升为故障转移仲裁者的角色：

```
CM_TIMEOUT 100
```

CLUSTER 连接管理器配置参数

CLUSTER 参数指定连接单元由高可用性集群组成，并指定该连接单元名称。一个高可用性集群是一个主服务器加上一个或多个辅助服务器；辅助服务器可为高可用性数据复制辅助服务器、共享磁盘辅助服务器，或远程独立辅助服务器。

语法

语法

CLUSTER*connection_unit_name*

使用

每个 CLUSTER 参数值在连接管理器配置文件中必须是唯一的。

CLUSTER 参数值不可使用多字节字符。

CLUSTER 连接单元可以使用以下重定向策略：

- 循环
- 辅助应用积压
- 工作负载

引用特定的高可用性集群的 CLUSTER 连接单元定义必须具有唯一的 PRIORITY 属性值。例如，以下由 server_1 和 server_2 组成的高可用性集群，在每个定义中，必须具有唯一的 PRIORITY 值。

my_connection_manager_1 配置文件：

```
NAME my_connection_manager_1
LOG 1
LOGFILE $INFORMIXDIR/tmp/my_cm_1.log

CLUSTER my_cluster
{
   INFORMIXSERVER server_1,server_2
   FOC ORDER=ENABLED \
       PRIORITY=1
}
```

my_connection_manager_2 配置文件：

```
NAME my_connection_manager_2
LOG 1
LOGFILE $INFORMIXDIR/tmp/my_cm_2.log

CLUSTER my_cluster
{
   INFORMIXSERVER server_1,server_2
   FOC ORDER=ENABLED \
       PRIORITY=2
}
```

示例 1：指定一个高可用性集群作为 CLUSTER 连接单元

在以下示例中，将由 my_server_group 中的服务器组成的高可用性集群指定为 CLUSTER 连接单元 my_cluster：

```
CLUSTER my_cluster
{
   INFORMIXSERVER my_server_group
   SLA sla_1 DBSERVERS=ANY
   FOC ORDER=ENABLED
       PRIORITY=1
}
```

DEBUG 连接管理器配置参数
DEBUG 参数指定是启用还是禁用 SQL 和 ESQL/C 错误消息的日志记录。

语法
语法
DEBUG{ 0 | 1 }

表 50: DEBUG 连接管理器配置参数的值

DEBUG 参数值	描述
1	启用 SQL 和 ESQL/C 错误消息的日志记录。
0 （缺省值）	禁用 SQL 和 ESQL/C 错误消息的日志记录。

使用

DEBUG 参数是可选的，并适用于以下连接单元：

- CLUSTER
- GRID
- REPLSET
- SERVERSET

调试消息创建在连接管理器配置文件中 LOGFILE 参数指定的位置。如果未指定 LOGFILE 参数，则连接管理器会创建 $INFORMIXDIR/tmp/*connection_manager_name*.*pid*.log。

连接管理器调试日志记录是在配置文件中启用；您无法从命令行来启用调试模式。

示例：

在以下示例中，my_log 是创建在 $INFORMIXDIR/tmp 目录中，并启用 SQL 和 ESQL/C 错误消息的日志记录。

```
LOGFILE $INFORMIXDIR/tmp/my_log
DEBUG 1
```

EVENT_TIMEOUT 连接管理器配置参数
EVENT_TIMEOUT 指定了活跃仲裁者连接管理器开启故障转移处理之前，等待主服务器事件的秒数。连接管理器等待主服务器事件或来自辅助服务器的主服务器脱机通知。主服务器事件是主服务器发出的该服务器仍在运行的指示，例如性能统计数据，或管理消息，或节点更改。

语法
语法
EVENT_TIMEOUT{ -1 | 60*seconds* }

表 51: EVENT_TIMEOUT 连接管理器配置参数值

EVENT_TIMEOUT 参数值	描述
-1	连接管理器无限期等待
0 至 30	连接管理器等待 30 秒
> 30	连接管理器等待指定的秒数

使用

EVENT_TIMEOUT 参数是可选的，且适用于 CLUSTER 连接单元。

如果未在连接管理器配置文件中设置 EVENT_TIMEOUT ，则连接管理器开始故障转移处理之前，将会等待主服务器事件或来自辅助服务器的主服务器的脱机消息 60 秒。

示例

在以下示例中，活跃仲裁者连接管理器等待 200 秒都未收到主服务器事件之后，将开启故障转移。

```
EVENT_TIMEOUT 200
```

相关链接

FOC 连接管理器配置参数 在第350页

FOC 连接管理器配置参数

FOC 参数和属性指定高可用性集群故障转移配置，以及连接管理器和主服务器之间连接的优先级。

语法

语法

FOC { ORDER = ENABLED | ORDER = DISABLED }
PRIORITY = *value*
[TIMEOUT = *seconds*]

CLUSTER 连接单元必需要有 FOC 参数。

FOC 连接管理器配置参数属性

表 52: FOC 参数属性

FOC 参数属性	描述
ORDER	禁用自动故障转移或使用主服务器 HA_FOC_ORDER 的配置参数值进行自动故障转移。 如果未指定 ORDER 属性，将使用主服务器 HA_FOC_ORDER 的配置参数值进行自动故障转移。 如果未指定 ORDER 属性，也未设置主服务器的 HA_FOC_ORDER 配置参数，那么故障转移顺序依次为 SDS、HDR 和 RSS。
PRIORITY	指定连接管理器和集群主服务器之间连接的优先级。 PRIORITY 值必须为正整数，并且在高可用性集群指定的所有连接管理器 CLUSTER 单元中是唯一的。值越低，优先级越高。 对于CLUSTER 连接单元，即使 ORDER 设置为 DISABLED，也必须指定 PRIORITY 值。
TIMEOUT	指定连接管理器在开始故障转移处理之前，必须等待主服务器事件的额外秒数。 在超过 EVENT_TIMEOUT 参数值后才会应用 TIMEOUT 属性值。例如，如果 EVENT_TIMEOUT 参数设置为 60，TIMEOUT 值设置为 10，那么必须在等待 70 秒都没有收到任何主服务器事件之后，才会开始故障转移。 如果未指定 TIMEOUT 属性，则故障转移将在超过 EVENT_TIMEOUT 参数指定的时间量后立即开始。

FOC 连接管理器配置参数的 ORDER 属性值

表 53: FOC 连接管理器配置参数的 ORDER 属性值

ORDER 属性值	描述
ENABLED	指定连接管理器可以执行自动故障转移。连接管理器使用主服务器的 onconfig 文件中 HA_FOC_ORDER 配置参数值来确定故障转移顺序。 如果 ORDER 属性设置为 ENABLED，但未指定主服务器的 HA_FOC_ORDER 配置参数，那么故障转移顺序依次为 SDS、HDR 和 RSS。

ORDER 使用方法

ORDER 属性指定是否启用自动故障转移。如果故障转移不能完成，连接管理器会以增量间隔（最多 60 秒）重新尝试故障转移，最多两天。

要修改活跃仲裁者连接管理器启动故障转移处理之前必须等待的秒数，请设置 EVENT_TIMEOUT 参数。

如果配置故障转移，可设置 CMALARMPROGRAM 参数来指定故障转移处理尝试八次后仍未完成时运行的程序或脚本。

> 重要： 如果启用了自动故障转移且故障转移处理已开始，那么在故障转移处理完成之前绝对不可以手动重新启动故障的主服务器。

PRIORITY 使用方法

如果在应用程序服务器主机上安装连接管理器，则可以使用 PRIORITY 属性来确定应用程序服务器和集群主服务器之间连接的优先级。

如果某特定连接管理器和主数据库服务器之间的网络连接失败，PRIORITY 确定连接管理器是否启动故障转移。如果故障转移会提升无法与活跃的高优先级连接管理器连接，则不会发生故障转移。如果故障转移会提升无法与活跃的低优先级连接管理器连接，则会发生故障转移。

示例 1：为故障转移配置多个连接管理器

以下为 my_primary_server_1 的 onconfig 文件中 HA_FOC_ORDER 配置参数的设置示列：

```
HA_FOC_ORDER SDS,HDR,RSS
```

安装三个连接管理器并配置以下配置文件：

cm_configuration_file_1

```
NAME my_connection_manager_1

CLUSTER my_cluster_1
{
   INFORMIXSERVER my_server_group_1
   SLA sla_1 DBSERVERS=ANY
   FOC ORDER=ENABLED \
       PRIORITY=1
   CMALARMPROGRAM $INFORMIXDIR/etc/cmalarmprogram.sh
}
```

cm_configuration_file_2

```
NAME my_connection_manager_2

CLUSTER my_cluster_1
{
   INFORMIXSERVER my_server_group_1
```

```
    SLA sla_2 DBSERVERS=ANY
    FOC ORDER=ENABLED \
        PRIORITY=2
    CMALARMPROGRAM $INFORMIXDIR/etc/cmalarmprogram.sh
}
```

cm_configuration_file_3

```
NAME my_connection_manager_3

CLUSTER my_cluster_1
{
    INFORMIXSERVER my_server_group_1
    SLA sla_3 DBSERVERS=ANY
    FOC ORDER=ENABLED \
        PRIORITY=3
    CMALARMPROGRAM $INFORMIXDIR/etc/cmalarmprogram.sh
}
```

当连接管理器完成初始化后，各自会搜索各自主机的 sqlhosts 文件中的 my_server_group_1 条目并连接到该组中的服务器。在 my_server_group_1 中主服务器的 onconfig 文件中的 HA_FOC_ORDER 配置参数值是为每个连接管理器设置的，以便所有连接管理器有一致的故障转移策略。主服务器的 HA_FOC_ORDER 配置参数值替换集群中所有辅助服务器的 onconfig 文件中的 HA_FOC_ORDER 配置参数值。如果主服务器故障，然后某个连接管理器重新启动，此过程将保持故障转移顺序的一致性。

如果连接管理器检测到主服务器故障，则首先尝试将最适合的 SD 辅助服务器转换成为主服务器。如果没有可用的 SD 辅助服务器，则连接管理器会尝试将 HDR 辅助服务器转换成为主服务器。如果 HDR 辅助服务器也不可用，那么连接管理器会尝试将最合适的 RS 辅助服务器转换成为主服务器。

如果故障转移处理尝试八次后仍失败，连接管理器将调用 $INFORMIXDIR/etc/cmalarmprogram.sh。

示例 2：配置多个连接管理器来确定应用程序服务器和集群主服务器之间连接的优先级

以下为 my_primary_server_2 的 onconfig 文件的 HA_FOC_ORDER 配置参数设置示例：

```
HA_FOC_ORDER HDR,RSS
```

安装与配置 my_important_app_cm 和 my_non_essential_app_cm 连接管理器：

cm_configuration_file_4

```
NAME important_app_cm
LOCAL_IP 192.0.2.0, 192.0.2.256

CLUSTER my_cluster
{
    INFORMIXSERVER my_server_group_1
    SLA sla_1 DBSERVERS=ANY
    FOC ORDER=ENABLED \
        PRIORITY=1
}
```

cm_configuration_file_5

```
NAME non_essential_app_cm
LOCAL_IP 192.0.2.257, 192.0.2.258

CLUSTER my_cluster
```

```
{
   INFORMIXSERVER my_server_group_1,
   SLA sla_2 DBSERVERS=ANY
   FOC ORDER=ENABLED \
       PRIORITY=2
}
```

- 网络状态是：
- important_app_cm 可以连接到主服务器和 RS 辅助服务器，但无法连接到 HDR 辅助服务器。
- non_essential_app_cm可以连接到主服务器和 HDR 辅助服务器，但无法连接到 RS 辅助服务器。

如果 non_essential_app_cm 和 my_primary_server 之间的网络出现故障，non_essential_app_cm 尝试故障转移到 HDR 辅助服务器。由于 important_app_cm 不能连接到 HDR 辅助服务器，且其优先级高于 non_essential_app_cm，所以故障转移的尝试遭到阻挡。

相反，如果 important_app_cm 和 my_primary_server 之间的网络出现故障，important_app_cm 尝试故障转移到 HDR 辅助服务器。由于 important_app_cm 不能连接到 HDR 辅助服务器，其将尝试故障转移到 RS 辅助服务器。non_essential_app_cm 不能连接到 RS 辅助服务器，但其优先级低于 important_app_cm，因此发生故障转移。

在这两种情况下， important_app_cm 和主服务器之间的连接优先于 non_essential_app_cm 和主服务器之间的连接。因为连接管理器位于应用程序服务器主机，应用程序服务器和主服务器之间的连接本质上是优先的。

相关链接

LOCAL_IP 连接管理器配置参数 在第355页
CMALARMPROGRAM 连接管理器配置参数 在第346页
EVENT_TIMEOUT 连接管理器配置参数 在第349页
《*SinoDB* 管理员参考》: *HA_FOC_ORDER* 配置参数
示例: 优先连接和网络监视的连接管理配置 在第403页

GRID 连接管理器配置参数

GRID 参数指定连接单元为 Enterprise Replication 网格，并指定网格名称。网格是可作为一个单元来管理的复制服务器集。

语法

语法

GRID*unit_name*

使用

在连接管理器配置文件中，每个连接单元的名称必须是唯一的。

GRID 连接单元可以使用以下重定向策略：

- 请求失败
- 循环机制
- 事务延迟
- 工作负载

网格名称可以使用多字节字符。

示例

以下示例中，网格连接单元命名为 my_grid：

```
GRID my_grid
{
   INFORMIXSERVER my_server_group_1,my_server_group_2
   SLA sla_1 DBSERVERS=ANY
}
```

INFORMIXSERVER 连接管理器配置参数

INFORMIXSERVER 参数指定连接管理器启动后连接的数据库服务器或 Enterprise Replication 节点。当连接管理器的 SQLHOSTS 参数设置为 REMOTE 或 LOCAL+REMOTE 时，INFORMIXSERVER 参数的值也用于为服务级别协议提供数据库服务器信息。

语法

CLUSTER 连接单元

CLUSTER 连接单元

INFORMIXSERVER { , *server_name sqlhosts_group_name* }

GRID 连接单元

GRID 连接单元

INFORMIXSERVER { , *sqlhosts_group_name* }

REPLSET 连接单元

REPLSET 连接单元

INFORMIXSERVER { , *sqlhosts_group_name* }

SERVERSET 连接单元

SERVERSET 连接单元

INFORMIXSERVER { , *sqlhosts_group_name stand_alone_server_name* }

使用

INFORMIXSERVER 参数为必需的，且以下连接单元支持：

- CLUSTER
- GRID
- REPLSET
- SERVERSET

对于 Enterprise Replication 域，列出接受来自连接管理器的客户端请求的节点的组名。

对于服务器集复制域，列出接受来自连接管理器的客户端请求的节点的组名。同时列出单机服务器名。

对于高可用性集群，列出集群的组名，或列出数据库服务器的名称。

示例

在以下示例中，连接管理器连接至连接管理器 sqlhosts 文件中 my_server_group 组中的所有数据库服务器。

```
NAME my_connection_manager


CLUSTER my_cluster
{
   INFORMIXSERVER my_server_group
   FOC ORDER=ENABLED \
       PRIORITY=1
}
```

相关链接

组信息 在第57页

LOCAL_IP 连接管理器配置参数
LOCAL_IP 参数指定运行连接管理器的计算机上要监视的 IP 地址。LOCAL_IP 参数与 FOC 参数的 PRIORITY 属性一起使用，以确定部分网络失效时是否发生数据库故障转移。

语法
语法
LOCAL_IP { , *ip_address* }

使用

LOCAL_IP 参数是可选的，且适用于 CLUSTER 连接单元。

您必须列出要监视的网络接口卡的 IP 地址。

示例

在以下示例中，连接管理器监视其主机与集群主服务器之间通过 192.0.2.0 和 192.0.2.1 IP 地址的网络接口卡的网络连接。

```
LOCAL_IP 192.0.2.0,192.0.2.1
```

相关链接

FOC 连接管理器配置参数 在第350页
示例: 优先连接和网络监视的连接管理配置 在第403页

LOG 连接管理器配置参数
LOG 配置参数指定连接管理器的日志记录方式。

语法
语法
LOG{ 0 | 1 | 2 | 3 }

表 54: LOG 连接管理器配置参数值

LOG 参数值	描述
0 （缺省）	指定关闭日志记录
1	连接管理器记录代理模式和重定向模式 SLA 信息。
2	连接管理器仅记录代理模式 SLA 信息。记录客户端和连接管理器之间的数据发送和接收活动。
3	连接管理器仅记录代理模式 SLA 信息。记录客户端和连接管理器之间的数据内容。

使用方式

LOG 参数是可选的，且适用于以下连接单元：

- CLUSTER
- GRID
- REPLSET
- SERVERSET

设置 LOGFILE 参数以指定日志文件的存储位置。如果未设置 LOGFILE 参数，连接管理器将在 $INFORMIXDIR/tmp 目录下创建名为 *connection_manager_name.process_ID*.log 的日志文件。

示例

以下示例中，连接管理器将代理模式和重定向模式 SLA信息记录到 $INFORMIXDIR/tmp/my_cm.log。

```
LOG 1
LOGFILE $INFORMIXDIR/tmp/my_cm.log
```

LOGFILE 连接管理器配置参数
LOGFILE 参数指定连接管理器日志文件的名称和位置。

语法
语法
LOGFILE*path_and_filename*

使用

LOCAL_IP 参数是可选的，且适用于以下连接单元：

- CLUSTER
- GRID
- REPLSET
- SERVERSET

日志文件的路径和文件名称在连接管理器启动时显示，且日志文件在连接管理器运行时会持续更新状态信息。如果同一主机上安装了多个连接管理器，请为每个连接管理器指定一个不同的路径和文件名。

请确定日志文件的目录存在，且启动连接管理器的用户可以访问此文件。

如果未设置 LOGFILE 参数，连接管理器将在 $INFORMIXDIR/tmp 目录下创建名为 *connection_manager_name.process_ID*.log 的日志文件。

示例

以下示例中，连接管理器记录有关客户端和连接管理器之间的数据发送和接收活动的代理模式 SLA 信息。此信息记录在 $INFORMIXDIR/tmp/my_cm.log。

```
LOG 2
LOGFILE $INFORMIXDIR/tmp/my_cm.log
```

MACRO 连接管理器配置参数
MACRO 参数用于指定宏的名称和可在连接管理器配置文件中重复使用的值。

Macro 定义格式
Macro 定义格式
MACRO*name=value*

Macro 使用格式
Macro 使用格式
${*macro_name*}

使用

MACRO 参数是可选的，且适用于连接管理器配置文件。

宏可以包含空格，但是不能含有换行符。

可设置多个 MACRO 参数以创建多个宏。

已定义的宏，可用于其他宏中。例如：

```
MACRO WA=wa_server_1,wa_server_2,wa_server_3,wa_server_4
MACRO OR=or_server_1,or_server_2,or_server_3,or_server_4
MACRO ID=id_server_1,id_server_2,id_server_3,id_server_4
MACRO PNW=${WA),${OR),${ID)
```

示例 1：在服务级别协议使用宏

以下示例中，宏 CA 包含八个服务器名称。

```
NAME my_connection_manager_1
MACRO CA=ca_server_1,ca_server_2,ca_server_3,ca_server_4, \
         ca_server_5,ca_server_6,ca_server_7,ca_server_8

CLUSTER my_cluster_1
{
   INFORMIXSERVER group_name_1
   SLA sla_1 DBSERVERS=${CA}
   FOC ORDER=ENABLED PRIORITY=1
}
```

通过以下方式扩展该宏：

```
{
   INFORMIXSERVER group_name_1
   SLA sla_1 DBSERVERS=ca_server_1,ca_server_2,ca_server_3,ca_server_4, \
         ca_server_5,ca_server_6,ca_server_7,ca_server_8
   FOC ORDER=ENABLED PRIORITY=1
}
```

示例 2：在服务级别协议使用多个宏

以下示例中，宏 WA、OR 和 ID 每个都包含服务器名称。宏 ID 包含括号以创建重定向策略组。

```
NAME my_connection_manager_2
MACRO WA=wa_server_1,wa_server_2,wa_server_3
MACRO OR=or_server_1,or_server_2,or_server_3
MACRO ID=(id_server_1,id_server_2,id_server_3)

CLUSTER my_cluster_2
{
   INFORMIXSERVER group_name_2
   SLA sla_1 DBSERVERS=PRI
   SLA sla_2 DBSERVERS=${WA},${OR}
   SLA sla_3 DBSERVERS=${ID} POLICY=ROUNDROBIN
   FOC ORDER=ENABLED PRIORITY=1
}
```

通过以下方式扩展这些宏：

```
{
   INFORMIXSERVER group_name_2
   SLA sla_1 DBSERVERS=PRI
   SLA sla_2
 DBSERVERS=wa_server_1,wa_server_2,wa_server_3,or_server_1,or_server_2,or_server_3
   SLA sla_3 DBSERVERS=(id_server_1,id_server_2,id_server_3)
 POLICY=ROUNDROBIN
   FOC ORDER=ENABLED PRIORITY=1
```

```
}
```

NAME 连接管理器配置参数
NAME 参数指定连接管理器实例的名称。

语法
语法
NAME*connection_manager_name*

使用方式

NAME 参数是必需的，且适用于连接管理器实例。

连接管理器实例的名称对于监视、关闭或重新载入连接管理器是必需的。

连接管理器名称在管理的连接单元域中是唯一的。

示例

以下示例中，连接管理器实例命名为 my_connection_manager。

```
NAME my_connection_manager
```

REPLSET 连接管理器配置参数
REPLSET 参数指定连接单元为 Enterprise Replication 复制集，并指定复制集的名称。一个复制集结合数个复制形成可作为一个单元来管理的集合。

语法
语法
REPLSET*unit_name*

使用

每个连接单元的名称在连接管理器配置文件中必须是唯一的。

REPLSET 连接单元可以使用以下重定向策略：

- 请求失败
- 循环机制
- 事务延迟
- 工作负载

复制集名可以使用多字节字符。

示例

以下示例中，定义名为 my_replicate_set 复制集连接单元：

```
REPLSET my_replicate_set
{
   INFORMIXSERVER my_group_1,my_group_2
   SLA sla_1 DBSERVERS=ANY
}
```

SECONDARY_EVENT_TIMEOUT 连接管理器配置参数
SECONDARY_EVENT_TIMEOUT 指定了连接管理器与辅助服务器的连接断开之前，等待辅助服务器事件的秒数。辅助服务器事件是辅助服务器发出的该服务器仍在运行的指示，例如性能统计数据，或管理消息。

语法
语法

SECONDARY_EVENT_TIMEOUT{ -1 | 60*seconds* }

表 55: SECONDARY_EVENT_TIMEOUT 连接管理器配置参数值

SECONDARY_EVENT_TIMEOUT 参数值	描述
-1	连接管理器无限期等待
0 至 30	连接管理器等待 30 秒。
> 30	连接管理器等待指定的秒数。

使用

SECONDARY_EVENT_TIMEOUT 参数是可选的，且适用于 CLUSTER 连接单元。

如果未在连接管理器配置文件中设置 SECONDARY_EVENT_TIMEOUT 参数，则连接管理器将在等待辅助服务器事件 60 秒后，断开与该服务器的连接。

示例

以下示例中，连接管理器等待 300 秒都未收到辅助服务器事件之后，将断开与服务器的连接。

```
SECONDARY_EVENT_TIMEOUT 300
```

SERVERSET 连接管理器配置参数
SERVERSET 参数指定连接单元为服务器集，并指定服务器集的名称。服务器集包含不使用复制或故障转移的不相关的标准服务器。

语法
语法
SERVERSET*unit_name*

使用

每个连接单元名称在连接管理器配置文件中必须是唯一的。

SERVERLSET 连接单元可以使用以下的重定向策略

- 循环机制
- 工作负载

服务器集的名称不能使用多字节字符。

示例：

以下示例中，定义名为 my_server_set 的服务器集的连接单位：

```
SERVERSET my_server_set
{
   INFORMIXSERVER my_group_1,my_group_2
   SLA sla_1 DBSERVERS=ANY
}
```

SLA 连接管理器配置参数
SLA 参数定义服务级别协议，将客户端请求定向到数据库服务器。

语法
SLA *sla_name*
{ <CLUSTER SLA 语法> <GRID 或 REPLSET SLA 语法> <SERVERSET SLA 语法> }

[MODE = { REDIRECT | PROXY }]
[USEALIASES = { ON | OFF }]
[WORKERS = { 4 *Number_of_threads* }]
[HOST = { *host_name ip_address* }]
[NETTYPE = { *drsoctcp onsoctcp* }]
[SERVICE = { *port_number service_name* }]
[MODE = { REDIRECT | PROXY }]
[USEALIASES = { ON | OFF }]
[WORKERS = { 4 *Number_of_threads* }]
[HOST = { *host_name ip_address* }]
[NETTYPE = { *drsoctcp onsoctcp* }]
[SERVICE = { *port_number service_name* }]
[SQLHOSTSOPT =" { , *option* } "]
CLUSTER SLA 语法
CLUSTER SLA 语法
DBSERVERS = { ANY [POLICY = { ROUNDROBIN | WORKLOAD [+ SECAPPLYBACKLOG: *pages*] }] | { , *group* | HDR | PRI | primary | RSS | SDS *server* } [] | { , | { , *group* | HDR | PRI | primary | RSS | SDS *server* } | ({ , *group* | HDR | PRI | primary | RSS | SDS *server* }) } [POLICY = { ROUNDROBIN | WORKLOAD [+ SECAPPLYBACKLOG: *pages*] }] }
GRID 或 REPLSET SLA 语法
GRID 或 REPLSET SLA 语法
DBSERVERS = { ANY [POLICY = { { + | [*weight* *] FAILURE | [*weight* *] LATENCY | [*weight* *] WORKLOAD } | ROUNDROBIN }] | { , *group* } | [] { , | { , *group* } | ({ , *group* }) } [POLICY = { { + | [*weight* *] FAILURE | [*weight* *] LATENCY | [*weight* *] WORKLOAD } | ROUNDROBIN }] }
SERVERSET SLA 语法片段
SERVERSET SLA 语法片段
DBSERVERS = { ANY [POLICY = { WORKLOAD | ROUNDROBIN }] | { , *group server* } | [] { , | { , *group server* } | ({ , *group server* }) } [POLICY= { WORKLOAD | ROUNDROBIN }] }
unique_823_Connect_42_asdf

SLA 参数属性

表 56: SLA 连接管理器配置参数值的属性

属性名	描述
DBSERVERS	指定定向连接请求的服务器、服务器别名、服务器组或服务器类型。使用 ANY 关键字、SDS 或 RSS 集群关键字，或括号括起来的一组值来启用此组的重定向策略。
HOST	指定数据库服务器的主机。使用 SLA 中的值，而不是连接的管理器的主机 sqlhosts 文件中的值。
MODE	指定连接请求是通过连接管理器还是连接管理器向连接请求源提供连接信息。 缺省值是 REDIRECT。
NETTYPE	指定数据服务器的网络协议。使用 SLA 中的值，而不是连接管理器的主机 sqlhosts 文件中的值。
POLICY	指定连接管理器如何将客户端连接请求重定向至 DBSERVER 属性指定的服务器。 重定向策略适用于 ANY 关键词、SDS 和 RSS 集群关键词，和括号括起来的一组值。括号内的括号被忽略。 缺省值是 WORKLOAD。

属性名	描述
	可以给定工作负载、应用失败和事务延迟策略相对权重。
SERVICE	指定数据库服务器的端口号或服务名。使用 SLA 中的值，而不是连接管理器的主机 sqlhosts 文件中的值。
SQLHOSTSOPT	指定 SLA 中指定的数据库服务器的连接选项。所有连接选项放在单一对引号中。使用 SLA 中的值，而不是连接管理器的主机 sqlhosts 文件中的值。
USEALIASES	指定连接管理器是否可以将客户端连接请求重定向至 DBSERVER 配置参数指定的数据库服务器别名。 缺省值是 ON。
WORKERS	指定分配给 SLA 的工作线程数量。当指定服务级别协议时，连接管理器将创建一个 SLA 监听器程序来拦截客户端连接请求。SLA 监听器程序可以有一个或多个工作线程。 缺省值是 4。

DBSERVERS 属性值

表 57: DBSERVERS 属性值

属性值	描述
ANY	指定连接请求可以发送至指定集群、网格或复制集中的任何可用数据库服务器。 对于 SERVERSET 连接单元，ANY 指定连接请求可以发送至 SERVERSET 连接单元的 INFORMIXSERVER 参数指定的任何可用数据库服务器。 将重定向策略应用到 ANY 时不需将 ANY 放进括号中。如果没有指定重定向策略，ANY 将使用工作负载重定向策略。
group	在连接管理器的主机 sqlhosts 文件中指定一个组条目。连接请求可以发送至组成员。
HDR	指定可以将连接请求发送至高可用性数据复制服务器的集群关键字。仅 CLUSTER 连接单元支持 HDR。
PRI 或 PRIMARY	指定可以将连接请求发送至主服务器的集群关键字。仅 CLUSTER 连接单元支持 PRI 和 PRIMARY 。
RSS	指定可以将连接请求发送至远程独立辅助服务器的集群关键字。仅 CLUSTER 连接单元支持 RSS。 不需要将 RSS 放在括号中，以将重定向策略应用于指定的服务器。如果没有指定重定向策略，RSS 将使用工作负载重定向策略。
SDS	指定可以将连接请求发送至共享磁盘辅助服务器的集群关键字。仅 CLUSTER 连接单元支持 SDS。 不需要将 SDS 放在括号中，以将重定向策略应用于指定的服务器。如果没有指定重定向策略，SDS 将使用工作负载重定向策略。
server	在连接管理器主机 sqlhosts 文件中指定服务器或别名条目。连接请求可以发送至服务器。

MODE 属性值

表 58: MODE 属性的值

属性值	值
PROXY	指定连接管理器作为客户端连接的代理服务器。 在以下情况中使用代理模式： • 防火墙阻止客户端应用连接数据库服务器。 • 对于使用 JDBC 和 SQLJ 版本 3.5.1（或较早版本）的数据服务器驱动，或 SinoDB® 客户端软件开发工具包 (Client SDK) 3.00 或更早版本编译的应用程序，不想重新编译。 由于作为代理服务器的连接管理器处理所有客户端/服务器的通讯，所以配置多个连接管理器实例，以避免连接管理器成为单点故障。 注： 对于代理模式，您必须将操作系统设置为允许最大数量的文件描述。 例如，在 UNIX™ 操作系统上使用 ulimit 命令。
REDIRECT （缺省）	指定客户端连接使用重定向模式，该模式将连接管理器配置为将相应的数据库服务器名称、IP 地址和端口号返回至发出请求的客户端应用程序。然后，客户端应用程序使用返回的 IP 地址和端口号来连接到指定的数据库服务器。 重定向策略 SLA 不支持来自使用 JDBC 和 SQLJ 版本 3.5.1（或较早版本）数据服务器驱动，或 SinoDB® 客户端软件开发工具包 (Client SDK) 3.00 或更早版本编译的应用程序的连接。

POLICY 属性值

表 59: POLICY 属性的值

属性值	值
FAILURE	指定将连接请求定向或代理到请求失败次数最少的复制服务器。 以下连接单元支持请求失败策略： • REPLSET • GRID 要使用请求失败策略，您必须通过运行 cdr define qod 和 cdr start qod 命令来启用数据质量监控（QOD）。网格要使用请求失败策略，必须具有启用复制的表。
LATENCY	指定将连接请求定向或代理到事务延迟最低的复制服务器。 以下连接单元支持事务延迟策略： • REPLSET • GRID 此参数值未指明具体的事务延迟时间段。连接管理器使用公式以相对值来决定将客户端连接请求重定向到那边去。 要使用事务延迟策略，您必须通过运行 cdr define qod 和 cdr start qod 命令来启用数据质量监控。网格要使用事务延迟策略，网格必须具有启用复制的表。
ROUNDROBIN	指定将连接请求以重复、有序方式（轮询）定向或代理到一组服务器。

属性值	值

	如果使用轮询策略，将使用 DBSERVERS 属性值来创建轮询组。在 DBSERVERS 属性组值中出现多次的服务器在轮询组中将视为单一参与者。例如，如果 SLA 具有以下定义：
	`SLA sla_1 DBSERVERS=(server_1,server_3,server_1,server_2) \`
	`POLICY=ROUNDROBIN`
	则轮询组成员为：
	• server_1
	• server_2
	• server_3
	以下连接单元支持轮询策略：
	• CLUSTER
	• REPLSET
	• GRID
	• SERVERSET
	以下软件支持轮询策略：
	• 所有 SinoDB® 服务器版本。
	• SinoDB® 客户端软件开发工具包 (Client SDK) 3.70.xC8 及以后版本，和 4.10xC2 及以后版本的连接管理器。

SECAPPLYBACKLOG:*number_of_pages*	指定如果辅助服务器的请求积压超出了 *number_of_pages*， 则连接管理器不会将新的连接重定向或代理到服务器。例如：
	`SLA sla_1 DBSERVERS=(server_1,server_2,server_3) \`
	`POLICY=SECAPPLYBACKLOG:500`
	连接管理器将连接请求发送到 server_1、server_2 或 server_3 中请求积压低于 500 页且工作负载最低的服务器。
	要查看集群中所有服务器的请求积压，请运行 onstat -g 集群命令。
	要查看特定辅助服务器的请求积压，请运行以下任一命令：
	• onstat -g dri
	• onstat -g sds
	• onstat -g rss
	CLUSTER 连接单元支持请求积压策略：
	以下软件支持请求积压策略：
	• SinoDB® 12.10.xC5 及以后版本。
	• SinoDB® 客户端软件开发工具包 (Client SDK) 3.70.xC8 及以后版本，和 4.10xC2 及以后版本的连接管理器。

WORKLOAD （缺省）	指定将连接请求定向或代理到工作负载最低的数据库服务器。 工作负载计算是基于服务器拥有的虚拟服务器数和服务器就绪队列中线程数。 以下连接单元支持 WORKLOAD 策略： • CLUSTER

属性值	值
	• GRID • REPLSET • SERVERSET

USEALIASES 属性值

表 60: USEALIASES 属性的值

属性值	值
ON (缺省)	指定连接管理器可以将客户端连接请求定向到数据库服务器的 DBSERVERALIASES 配置参数指定的服务器别名。
OFF	指定连接管理器不可以将客户端连接请求定向到数据库服务器的 DBSERVERALIASES 配置参数指定的服务器别名。

HOST 属性值

表 61: HOST 属性的值

属性值	值
host_name	指定数据库服务器的主机名或主机别名。
ip_address	指定数据库服务器的 TCP/IP 地址

NETTYPE 属性值

表 62: NETTYPE 属性的值

属性值	值
drsoctcp	为分布式关系数据库结构指定 TCP/IP 协议。
onsoctcp	指定使用 TCP/IP 协议的套接字

SERVICE 属性值

表 63: SERVICE 属性的值

属性值	值
port_number	指定端口号
service_name	指定服务名称

使用

SLA 参数为可选的，且以下连接单元支持该参数：

- CLUSTER
- GRID
- REPLSET
- SERVERSET

客户端应用程序使用 SLA 名称连接数据服务器或 DBSERVERS 属性值指定的数据服务器类型。对于每一个 SLA，有一个监听线程安装在服务器上的指定端口来检测传入的客户端请求。SLA 参数可以在相同的配置文件中出现多次，但是每个 SLA 名称必须是唯一的。

示例 1：来自服务级别协议的连接请求重定向

以下示例展示了一个简单的连接管理器配置。该配置指定了一个单一的 SLA。

```
NAME my_connection_manager_1

CLUSTER my_cluster_1
{
   INFORMIXSERVER my_server_group_1
   SLA sla_1 DBSERVERS=SDS,HDR,PRI
   FOC ORDER=ENABLED PRIORITY=1
}
```

CONNECT TO @sla_1 连接请求按以下方式定向：

1. 连接到任一可用 SD 辅助服务器
2. 如果 SD 辅助服务器不可用，连接至 HDR 辅助服务器。
3. 如果 HDR 辅助服务器不可用，连接至主服务器。

示例 2：定义多个服务级别协议

以下示例展示了一个简单的连接管理器配置。该配置指定了两个 SLA。

```
NAME my_connection_manager_2

CLUSTER my_cluster_2
{
   INFORMIXSERVER my_server_group_2
   SLA sla_1 DBSERVERS=server_1
   SLA sla_2 DBSERVERS=server_2,server_3
   FOC ORDER=ENABLED PRIORITY=1
}
```

本示例是为一个高可用性集群配置连接管理器，并定义了两个 SLA：

- CONNECT TO @sla_1 连接请求定向到 server_1。
- CONNECT TO @sla_2 连接请求定向到 server_2。如果 server_2 不可用，则连接请求定向到 server_3。

示例 3：服务级别协议的重定向策略

以下示例展示了使用工作负载平衡重定向策略的连接管理器配置。该配置指定了一个单一的 SLA。

```
NAME my_connection_manager_3

CLUSTER my_cluster_3
{
   INFORMIXSERVER my_server_group_3
   SLA sla_1 DBSERVERS=(server_1,server_2) \
             POLICY=WORKLOAD
   SLA sla_2 DBSERVERS=(server_3,server_4,server_5) \
             POLICY=ROUNDROBIN
   SLA sla_3 DBSERVERS=(server_6,server_7,server_8) \
             POLICY=ROUNDROBIN+SECAPPLYBACKLOG:400
   FOC ORDER=ENABLED PRIORITY=1
```

```
}
```

- CONNECT TO @sla_1 连接请求定向到 server_1 和 server_2 中具有最低工作负载的服务器。WORKLOAD 是缺省的重定向策略，所以不需要在 SLA 中指定 POLICY=WORKLOAD。
- CONNECT TO @sla_2 连接请求轮询定向到 server_3、server_4 和 server_5。
- CONNECT TO @sla_3 连接请求轮询定向到 server_6、server_7 和 server_8。 如果服务器的请求积压为 400 页或更多，则服务器将在轮询顺序中被忽略，且在其请求积压低于 400 页前都不会收到连接请求。

示例 4：在服务级别协议中的代理和重定向模式

在重定向模式下，连接管理器通过向客户端应用程序返回指定的数据库服务器的 IP 地址和端口号来回应重定向请求。客户端应用程序使用该 IP 地址和端口号连接数据库服务器。代理模式下，连接管理器充当代理服务器，客户端请求将通过连接管理器进行路由。如果未指定 SLA 模式，则重定向模式为缺省模式。

```
NAME my_connection_manager_4

CLUSTER my_cluster_4
{
   INFORMIXSERVER my_server_group_4
   SLA sla_1 DBSERVERS=ANY \
             MODE=REDIRECT #Default value, so is not required for the SLA
 definition
   SLA sla_2 DBSERVERS=ANY \
             MODE=PROXY
   FOC ORDER=ENABLED PRIORITY=1
}
```

- CONNECT TO @sla_1 连接请求会使连接管理器将集群服务器的 IP 地址和端口号返回给客户端应用程序。
- CONNECT TO @sla_1 连接请求通过连接管理器定向到一个集群服务器。

示例 5：调整连接管理器和集群服务器的超时值

以下示例展示更改了缺省超时值的连接管理器配置：

```
NAME my_connection_manager_5
CM_TIMEOUT 300
EVENT_TIMEOUT 45
SECONDARY_EVENT_TIMEOUT 50

CLUSTER my_cluster_5
{
   INFORMIXSERVER my_server_group_5
   SLA sla_1 DBSERVERS=ANY
   FOC ORDER=ENABLED PRIORITY=1
}
```

- 如果集群在 300 秒内未收到来自连接管理器的任何事件，则集群的主服务器会将下一个可用的连接管理器提升为故障转移仲裁者的角色。
- 如果连接管理器在 45 秒内未收到来自主服务器的任何事件，则连接管理器开始故障转移处理，并试图将辅助服务器提升为主服务器。
- 如果连接管理器在 50 秒内未收到来自辅助服务器的任何事件，则连接管理器将断开与辅助服务器的连接。

示例 6：服务级别协议中的宏和负载均衡

以下示例展示了使用 SLA 中已定义的宏的连接管理器配置。

```
NAME my_connection_manager_6
MACRO CA=ca_server_1,ca_server_2,ca_server_3
MACRO NY=ny_server_1,ny_server_2,ny_server_3

REPLSET my_replicate_set_1
{
   INFORMIXSERVER my_er_group_1,my_er_group_2
   SLA sla_1 DBSERVERS=${CA}
   SLA sla_2 DBSERVERS=(${NY}) \
             POLICY=ROUNDROBIN
}
```

本示例中定义了两个宏：

- CA，由 ca_server_1、ca_server_2 和 ca_server_3 组成。
- NY，由 ny_server_1、ny_server_2 和 ny_server_3 组成。

连接管理器如下重定向客户端连接请求：

- CONNECT TO @sla_1 连接器请求定向到 ca_server_1。如果 ca_server_1 不可用，则连接请求定向到 ca_server_2。如果 ca_server_2 不可用，则连接请求定向到 ca_server_3。
- CONNECT TO @sla_2 连接请求轮询定向到 ny_server_1、ny_server_2 和 ny_server_3。

示例 7：服务级别协议的数据质量重定向策略

以下示例展示使用事务延迟和请求失败重定向策略定向网格中连接请求的连接管理器配置。使用 cdr define qod 和 cdr start qod 命令开启数据质量（QOD）监控。

```
NAME my_connection_manager_7

GRID my_grid_1
{
   INFORMIXSERVER my_server_group_1,my_server_group_2,my_server_group_3
   SLA sla_1 DBSERVERS=ANY \
             POLICY=LATENCY
   SLA sla_2 DBSERVERS=ANY \
             POLICY=FAILURE
   SLA sla_3 DBSERVERS=ANY \
             POLICY=2*Failure+LATENCY
}
```

- CONNECT TO @sla_1 连接请求定向到事务延迟最低的服务器。
- CONNECT TO @sla_2 连接请求定向到请求失败次数最少的服务器。
- CONNECT TO @sla_3 连接请求定向到请求失败次数最少和事务延迟最低的服务器。在连接管理计算中，最小请求失败次数是最低事务延迟的两倍重要。

示例8：服务级别协议中的 sqlhosts 连接信息

以下示例展示了使用属性值而不是主机 sqlhosts 文件中的值来定向连接请求的连接管理器配置。

```
NAME my_connection_manager_8
```

```
SERVERSET my_server_set
{
   INFORMIXSERVER server_1,server_2,server_3
   SLA sla_1 DBSERVERS=server_1 \
             NETTYPE=onsoctcp
             HOST=host_1 \
             SERVICE=port_1 \
   SLA sla_2 DBSERVERS=server_2 \
             NETTYPE=onsoctcp
             HOST=host_2 \
             SERVICE=port_2 \
}
```

连接管理器使用 HOST、SERVICE 和 NETTYPE 属性的值来定向连接请求，而不是其主机 sqlhosts 文件内的值。

示例 9：用别名控制连接请求

以下示例展示了指定 SLA 可使用的数据库服务器别名的连接管理器配置。

server_1 的 onconfig 文件有以下参数：

```
DBSERVERALIASES server_1_alias_1,server_1_alias_2
```

server_1 使用的 sqlhosts 文件有以下条目：

```
#dbservername      nettype   hostname   servicename  options
my_group_9         group     -          -            e=server_1_alias_2
server_1           onsoctcp  my_host_1  my_port_1    g=my_group_9
server_1_alias_1   onsoctcp  my_host_1  my_port_2    g=my_group_9
server_1_alias_2   onsoctcp  my_host_1  my_port_3    g=my_group_9
```

server_1 的连接管理器配置文件有以下条目：

```
NAME my_connection_manager_9

SERVERSET my_server_set_2
{
   INFORMIXSERVER my_group_9
   SLA sla_1 DBSERVERS=server_1
   SLA sla_2 DBSERVERS=server_1 \
             USEALIASES=OFF
   SLA sla_3 DBSERVERS=server_1_alias_1
   SLA sla_4 DBSERVERS=server_1_alias_1 \
             USEALIASES=OFF
}
```

连接管理器按以下方式定向客户端请求：

- CONNECT TO @sla_1 和 CONNECT TO @sla_3 请求 通过 my_port_1、my_port_2、或 my_port_3 定向到 server_1。
- CONNECT TO @sla_2 请求仅通过 my_port_1 定向到 server_1。
- CONNECT TO @sla_4 请求仅通过 my_port_2 定向到 server_1。

相关链接

组信息 在第57页

SQLHOSTS 连接管理器配置参数
SQLHOSTS 参数指定了连接管理器可以到那里搜索由 INFORMIXSERVER 参数和 DBSERVERS 属性指定的数据库服务器。

Syntax
Syntax
SQLHOSTS={ LOCAL | REMOTE | LOCAL+REMOTE }

表 64: SQLHOSTS 连接管理器配置参数值

SQLHOSTS 参数值	描述
LOCAL	连接管理器在本地 sqlhosts 文件中搜索请求的数据库服务器实例。
REMOTE	连接管理器在远程 sqlhosts 文件中搜索请求的数据库服务器实例。连接管理器在连接单元的 INFORMIXSERVER 参数指定的数据库服务器的 sqlhosts 文件中搜索。
LOCAL+REMOTE （缺省）	连接管理器在本地 sqlhosts 文件中搜索请求的数据库服务器实例。如果无法找到数据库服务器，则连接管理器在连接单元的 INFORMIXSERVER 参数指定的数据库服务器的 sqlhosts 文件中搜索。

使用

SQLHOSTS 参数是可选的，且适用于以下连接单元：

- CLUSTER
- GRID
- REPLSET
- SERVERSET

当您想要限制客户端应用程序访问高可用性集群中的一个或多个数据库服务器时，SQLHOSTS 选项是十分有用的。

示例

以下示例中， SQLHOSTS 参数用于限制连接管理器可连接的服务器。

```
NAME my_connection_manager
SQLHOSTS LOCAL

CLUSTER my_cluster
{
   INFORMIXSERVERS my_servers
   SLA my_sla DBSERVERS=server_1,server_2,server_3,server_4
   FOC ORDER=ENABLED PRIORITY=1
}
```

本地 hosts 文件有以下条目：

```
#dbservername       nettype    hostname   servicename   options
 my_servers         group      -          -             c=1,e=server_3
 server_1           onsoctcp   host_1     port_1        g=my_servers
 server_2           onsoctcp   host_2     port_2        g=my_servers
 server_3           onsoctcp   host_3     port_3        g=my_servers
```

远程数据库服务器的 sqlhosts 文件有以下条目：

```
#dbservername       nettype    hostname   servicename   options
 my_servers         group      -          -             c=1,e=server_4
 server_1           onsoctcp   host_1     port_1        g=my_servers
```

```
server_2          onsoctcp   host_2     port_2        g=my_servers
server_3          onsoctcp   host_3     port_3        g=my_servers
server_4          onsoctcp   host_3     port_3        g=my_servers
```

server_4 未定义在本地 sqlhosts 文件中，且连接管理器被配置为不搜索远程 sqlhosts 文件，因此连接管理器不会向 server_4 发送连接请求。

相关链接

组信息 在第57页

更改连接管理器配置文件

使用 oncmsm 实用程序将修改的配置文件加载到连接管理器并更改连接管理器的配置。

如果您使用多个连接管理器，那么您可以运行 onstat -g cmsm 命令来显示连接管理器实例的名称。

1. 修改现有的连接管理器配置文件

   配置文件的缺省位置是 $INFORMIXDIR/etc 目录。
2. 在连接管理器的主机上，运行带有 r 参数和连接管理器实例名称的 oncmsm 实用程序。

   示例：

```
oncmsm -r connection_manager_name
```

   由于连接管理器的多个实例可以同时处于活跃状态，因此您必须指定连接管理器实例的名称。

连接管理器配置文件重新加载后，新的配置立即生效。

相关链接

《*SinoDB 管理员参考*》*: oncmsm 实用程序*

《*SinoDB 管理员参考*》*: onstat -g cmsm command:* 显示连接管理器信息

将旧格式的连接管理器配置文件转换为当前格式

在 3.70.xC3版本之前的 SinoDB® 客户端软件开发工具包 (Client SDK) 版本中的连接管理器配置文件，与当前版本的连接管理器无法兼容。您必须通过运行 oncmsm 实用程序从 3.70.xC3 之前的版本转换配置文件。

如果有多个配置文件需要转换，可将相关的 SLA 定义结合为单个配置文件。

即使连接管理器正在运行，也可以转换连接管理器的配置文件。

将连接管理器文件转换为当前格式 ：

1. 登录到安装了更新的 Client SDK 的计算机。
2. 运行 oncmsm 实用程序，指定旧的和新的配置文件名称。

配置文件转换后，可以加载到连接管理器。

示例：转换配置文件并将加载到连接管理器

对于以下示例 ：

- 使用的是 UNIX™ 操作系统。
- 连接管理器从 3.50.xC9 更新到了 4.10xC1，则必须更新配置文件。
- 连接管理器使用位于 $INFORMIXDIR/etc 目录中名为 configuration_file_old 的配置文件。
- 想要将该文件转换为新格式并重命名为 configuration_file_new。

运行以下命令 ：

1. oncmsm -c configuration_file_old -n configuration_file_new
2. oncmsm -c configuration_file_new

相关链接

oncmsm 实用程序

为连接管理配置环境与设置配置参数

启动连接管理器之前，必须配置其环境。

设置环境变量 ：

- 对于 UNIX™ C 使用适当的 shell 命令。下面的示例使用 C shell (csh)。
- 对于 Windows™ 系统，使用 setnet32 实用程序的 Environment 选项卡。

1. 如果客户端或连接管理器安装在未安装数据库服务器的主机上，将每个主机的 INFORMIXDIR 环境变量设置为客户端或连接管理器所安装的目录。
   运行以下命令 ：

   ```
   setenv INFORMIXDIR path
   ```

2. 如果客户端或连接管理器安装在未安装数据库服务器的主机上，将每个主机的 INFORMIXSQLHOSTS 环境变量设置为主机所使用的 sqlhosts 文件的所在位置。
   运行以下命令 ：

   ```
   setenv INFORMIXSQLHOSTS path_and_filename
   ```

3. 如果连接管理器配置文件是位于 $INFORMIXDIR/etc 以外的目录中，将 CMCONFIG 环境变量设置为该目录。
   运行以下命令 ：

   ```
   setenv CMCONFIG path_and_filename
   ```

4. 如果连接管理器控制高可用性集群的故障转移，请将所有托管的集群数据库服务器上的 onconfig 文件的 DRAUTO 配置参数设置为 3。
   例如：

   ```
   DRAUTO 3
   ```

5. 如果连接管理器控制高可用性集群的故障转移，请将每个集群主服务器上的 onconfig 文件的 HA_FOC_ORDER 配置参数设置故障转移顺序。
   例如：

   ```
   HA_FOC_ORDER SDS，HDR，RSS
   ```

6. 在每个使用多个端口的数据库在服务器上，将 onconfig 文件的 DBSERVERALIASES 配置参数设置为连接管理器和数据库服务器 sqlhosts 文件条目中所列的别名。
   例如：

   连接管理器主机具有以下 sqlhost 文件条目 ：

   ```
   #dbservername   nettype    hostname   servicename   options
    my_servers     group      -          -             c=1,e=server_3
    server_1       onsoctcp   host_1     port_1        s=6,g=my_servers
    server_2       onsoctcp   host_2     port_2        s=6,g=my_servers
    server_3       onsoctcp   host_3     port_3        s=6,g=my_servers

    my_aliases     group      -          -             c=1,e=a_server_3
    a_server_1     onsoctcp   host_1     port_4        g=my_aliases
    a_server_2     onsoctcp   host_2     port_5        g=my_aliases
    a_server_3     onsoctcp   host_3     port_6        g=my_aliases
   ```

   在 server_1 的 onconfig 文件中设置以下的值 ：

   ```
   DBSERVERALIASES a_server_1
   ```

在 server_2 的 onconfig 文件中设置以下的值 ：

```
DBSERVERALIASES a_server_2
```

在 server_3 的 onconfig 文件中置以下的值 ：

```
DBSERVERALIASES a_server_3
```

相关链接

《*SinoDB SQL* 指南: 参考》: *INFORMIXDIR* 环境变量
《*SinoDB SQL* 指南: 参考》: *INFORMIXSQLHOSTS* 环境变量
《*SinoDB SQL* 指南: 参考》: *CMCONFIG* 环境变量
《*SinoDB* 管理员参考》: *DBSERVERALIASES* 配置参数
《*SinoDB* 管理员参考》: *DRAUTO* 配置参数
《*SinoDB* 管理员参考》: *HA_FOC_ORDER* 配置参数

定义连接管理的 sqlhosts 信息

您必须为连接到连接管理器的客户端应用程序，连接到数据库服务器的连接管理器以及属于连接管理器连接单元的数据库服务器定义 sqlhosts 网络连接信息。

如果连接管理器或客户端是安装在未安装数据库服务器的主机上，那么您必须在每个主机上创建 sqlhosts 文件。

sqlhosts 文件的条目可以指定以下连接单元组件的连接信息 ：

- 数据库服务器
- 使用安全端口的数据库服务器的别名
- 连接管理器服务级别协议（SLA）
- 包含数据库服务器、 数据库服务器别名或 SLAs 的组。

连接管理器连接的所有数据库服务器都必须列在连接管理器使用的 sqlhosts 文件中。如果连接管理器监控高可用性集群，那么连接管理器使用的 sqlhosts 文件必须有所有集群服务器的条目。

1. 在每个数据库服务器主机的 sqlhosts 文件中创建条目。
   您可以修改其中一个 sqlhosts 文件，然后将它分发给其他数据库服务器主机。
2. 在每个连接管理器主机的 sqlhosts 文件中创建条目。
   您可以创建其中一个 sqlhosts 文件，然后将它分发给其他连接管理器主机。
3. 在每个客户端应用程序主机的 sqlhosts 文件中创建条目。
   您可以创建其中一个 sqlhosts 文件，然后将它分发给其他客户端应用程序主机。
4. 如果主机上安装了多个数据库服务器、如果 sqlhosts 文件是在 $INFORMIXDIR/etc 以外的目录中，或者，如果您使用 $INFORMIXDIR/etc/sqlhosts 以外的网络连接文件，请将主机 INFORMIXSQLHOSTS 环境变量设置为 sqlhosts 文件的位置。

如果 sqlhosts 文件条目使用 s = 6 选项来定义安全端口，则使用 sqlhosts 文件中的信息来创建密码文件。

相关链接

组信息 在第57页
sqlhosts 信息 在第47页
《*SinoDB SQL* 指南: 参考》: *INFORMIXSQLHOSTS* 环境变量

为高可用性集群连接管理定义 sqlhosts 信息
您必须为高可用性集群连接管理定义 sqlhosts 网络连接信息。

若要在特定主机上使用 $INFORMIXDIR/etc/sqlhosts 以外的文件，请将主机的 INFORMIXSQLHOSTS 环境变量设置为该替代文件。

1. 在每个连接管理器和数据库服务器的主机上，为集群中的每个数据库服务器创建 sqlhosts 文件条目。
例如：

```
#dbservername   nettype    hostname   servicename   options
 server_1       onsoctcp   host_1     port_1
 server_2       onsoctcp   host_2     port_2
 server_3       onsoctcp   host_3     port_3
```

2. 在每个连接管理器的主机上，添加一个包含集群中每个数据库服务器的组条目，并对数据库服务器条目添加组选项。使用 c=1 组条目选项，以便组成员列表中的连接尝试起点是随机的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。
例如：

```
#dbservername   nettype    hostname   servicename   options
 my_servers     -          -                        c=1,e=server_3
 server_1       onsoctcp   host_1     port_1        g=my_servers
 server_2       onsoctcp   host_2     port_2        g=my_servers
 server_3       onsoctcp   host_3     port_3        g=my_servers
```

3. 在每个客户端应用程序的主机上，为每个连接管理器配置文件中的每个服务级别协议（SLA）创建一个 sqlhosts 文件条目。

   第一个连接管理器配置文件具有以下条目：

```
NAME connection_manager_1

CLUSTER my_cluster
{
   INFORMIXSERVER my_servers
   SLA sla_primary_1     DBSERVERS=PRI
   SLA sla_secondaries_1 DBSERVERS=SDS,HDR
   FOC ORDER=ENABLED \
       PRIORITY=1
}
```

   第二个连接管理器配置文件具有以下条目：

```
NAME connection_manager_2

CLUSTER my_cluster
{
   INFORMIXSERVER my_servers
   SLA sla_primary_2     DBSERVERS=PRI
   SLA sla_secondaries_2 DBSERVERS=SDS,HDR
   FOC ORDER=ENABLED \
       PRIORITY=2
}
```

   将以下条目添加到每个客户端应用程序的主机 sqlhosts 文件中：

```
#dbservername       nettype    hostname   servicename   options
 sla_primary_1      onsoctcp   cm_host_1  cm_port_1
 sla_primary_2      onsoctcp   cm_host_2  cm_port_2

 sla_secondaries_2  onsoctcp   cm_host_1  cm_port_3
 sla_secondaries_2  onsoctcp   cm_host_2  cm_port_4
```

4. 在每个客户端应用程序的主机上，为每组 SLA 条目创建 sqlhosts 文件条目，并且将组选项添加至 SLA 条目。使用 c=1 组条目选项，以便组成员列表中的连接尝试起点是随机的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。

```
#dbservername       nettype    hostname   servicename  options
 g_primary          group      -          -            c=1,e=sla_primary_2
 sla_primary_1      onsoctcp   cm_host_1  cm_port_1    g=g_primary
 sla_primary_2      onsoctcp   cm_host_2  cm_port_2    g=g_primary

 g_secondaries      group      -          -            c=1,e=sla_secondaries_2
 sla_secondaries_2  onsoctcp   cm_host_1  cm_port_3    g=g_secondaries
 sla_secondaries_2  onsoctcp   cm_host_2  cm_port_4    g=g_secondaries
```

到 @g_primary 的客户端连接请求将被定向到其中一个连接管理器。如果 connection_manager_1 连接管理器接收到请求，会使用 sla_primary_1 向客户端应用程序提供主服务器的连接信息。如果 connection_manager_2 连接管理器接收到请求，会使用 sla_primary_2 向客户端应用程序提供主服务器的连接信息。

相关链接

组信息 在第57页
sqlhosts 信息 在第47页
《*SinoDB SQL* 指南: 参考》: *INFORMIXSQLHOSTS* 环境变量

为使用安全端口的高可用性集群的连接管理定义 sqlhosts 信息

您必须为高可用性集群连接管理定义 sqlhosts 网络连接信息。

若要在特定主机上使用 $INFORMIXDIR/etc/sqlhosts 以外的文件，请将主机的 INFORMIXSQLHOSTS 环境变量设置为该替代文件。

1. 在每个连接管理器和数据库服务器的主机上，为集群中的每个数据库服务器创建 sqlhosts 文件条目，并指定 s=6 安全端口选项。
   例如：

```
#dbservername   nettype    hostname   servicename   options
 server_1       onsoctcp   host_1     port_1        s=6
 server_2       onsoctcp   host_2     port_2        s=6
 server_3       onsoctcp   host_3     port_3        s=6
```

2. 在每个连接管理器和数据库服务器的主机上，为每个数据库服务器创建 sqlhosts 文件别名条目。
   例如：

```
#dbservername   nettype    hostname   servicename   options
 server_1       onsoctcp   host_1     port_1        s=6
 a_server_1     onsoctcp   host_1     port_4

 server_2       onsoctcp   host_2     port_2        s=6
 a_server_2     onsoctcp   host_2     port_5

 server_3       onsoctcp   host_3     port_3        s=6
 a_server_3     onsoctcp   host_3     port_6
```

3. 在每个连接管理器的主机上，向组条目添加个别条目。向数据库服务器和数据库服务器别名条目添加组选项。使用 c=1 组条目选项，以便组成员列表中的连接尝试起点是随机的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。
   例如：

```
#dbservername   nettype    hostname   servicename   options
 my_servers     group      -          -             c=1,e=a_server_3
 server_1       onsoctcp   host_1     port_1        s=6,g=my_servers
 a_server_1     onsoctcp   host_1     port_4        g=my_servers
 server_2       onsoctcp   host_2     port_2        s=6,g=my_servers
```

```
 a_server_2     onsoctcp   host_2     port_5        g=my_servers
 server_3       onsoctcp   host_3     port_3        s=6,g=my_servers
 a_server_3     onsoctcp   host_3     port_6        g=my_servers
```

4. 在每个数据库服务器的 onconfig 文件中，将 DBSERVERALIASES 参数设为该数据库服务器的别名。

   server_1 的 onconfig 文件条目：

```
DBSERVERALIASES a_server_1
```

   server_2 的 onconfig 文件条目：

```
DBSERVERALIASES a_server_2
```

   server_3 的 onconfig 文件条目：

```
DBSERVERALIASES a_server_3
```

5. 在每个客户端应用程序的主机上，为连接管理器配置文件中的每个服务级别协议（SLA）创建 sqlhosts 文件条目。

   第一个连接管理器配置文件具有以下条目：

```
NAME connection_manager_1

CLUSTER my_cluster
{
   INFORMIXSERVER my_servers
   SLA sla_primary_1     DBSERVERS=PRI
   SLA sla_secondaries_1 DBSERVERS=SDS,HDR
   FOC ORDER=ENABLED PRIORITY=1
}
```

   第二个连接管理器配置文件具有以下条目：

```
NAME connection_manager_2

CLUSTER my_cluster
{
   INFORMIXSERVER my_servers
   SLA sla_primary_2     DBSERVERS=PRI
   SLA sla_secondaries_2 DBSERVERS=SDS,HDR
   FOC ORDER=ENABLED PRIORITY=2
}
```

   向每个客户端应用程序的主机 sqlhosts 文件添加以下条目：

```
#dbservername       nettype    hostname   servicename   options
 sla_primary_1      onsoctcp   cm_host_1  cm_port_1
 sla_primary_2      onsoctcp   cm_host_2  cm_port_2

 sla_secondaries_2  onsoctcp   cm_host_1  cm_port_3
 sla_secondaries_2  onsoctcp   cm_host_2  cm_port_4
```

6. 在每个客户端应用程序的主机上，为每组 SLA 条目创建 sqlhosts 文件组条目，并且将组选项添加至 SLA 条目。使用 c=1 组条目选项，以便组成员列表中的连接尝试起点是随机的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。

```
#dbservername       nettype    hostname   servicename   options
 g_primary          group      -          -             c=1,e=sla_primary_2
 sla_primary_1      onsoctcp   cm_host_1  cm_port_1     g=g_primary
```

```
 sla_primary_2      onsoctcp   cm_host_2  cm_port_2     g=g_primary

 g_secondaries      group      -          -             c=1,e=sla_secondaries_2
 sla_secondaries_2  onsoctcp   cm_host_1  cm_port_3     g=g_secondaries
 sla_secondaries_2  onsoctcp   cm_host_2  cm_port_4     g=g_secondaries
```

到 @g_primary 的客户端连接请求将被定向到其中一个连接管理器。如果 connection_manager_1 连接管理器接收到请求，会使用 sla_primary_1 向客户端应用程序提供主服务器的连接信息。如果 connection_manager_2 连接管理器接收到请求，会使用 sla_primary_2 向客户端应用程序提供主服务器的连接信息。

相关链接

组信息 在第57页
sqlhosts 信息 在第47页
《*SinoDB SQL* 指南: 参考》: *INFORMIXSQLHOSTS* 环境变量
《*SinoDB* 管理员参考》: *DBSERVERALIASES* 配置参数

为使用分布式关系数据库结构 (DRDA®) 的高可用性集群定义 sqlhosts 信息

连接管理器支持高可用性集群的 Distributed Relational Database Architecture™ (DRDA®) (分布式关系数据库结构) 连接。必须为使用 DRDA® 的高可用性集群的连接管理定义 sqlhosts 网络连接信息。

要在特定主机上使用 $INFORMIXDIR/etc/sqlhosts 以外的文件，请将主机的 INFORMIXSQLHOSTS 环境变量设置为该替代文件。

1. 在每个连接管理器和数据库服务器主机上，为集群中每个数据库服务器创建 sqlhosts 文件条目。
   例如：

```
#dbservername   nettype    hostname   servicename   options
 server_1       onsoctcp   host_1     port_1
 server_2       onsoctcp   host_3     port_2
 server_3       onsoctcp   host_5     port_3
```

2. 在每个数据库服务器的 onconfig 文件中，使用 DBSERVERALIASES 参数设置数据库服务器的别名。

   server_1 的 onconfig 文件条目：

```
DBSERVERALIASES drda_1
```

   server_2 的 onconfig 文件条目：

```
DBSERVERALIASES drda_2
```

   server_3 的 onconfig 文件条目：

```
DBSERVERALIASES drda_3
```

3. 在每个连接管理器的主机上，为 DRDA® 别名添加条目。nettype 值使用 DRDA® 协议。
   例如：

```
#dbservername   nettype    hostname   servicename   options
 server_1       onsoctcp   host_1     port_1
 server_2       onsoctcp   host_2     port_2
 server_3       onsoctcp   host_3     port_3

 drda_1         drsoctcp   host_1     port_4
 drda_2         drsoctcp   host_2     port_5
 drda_3         drsoctcp   host_3     port_6
```

4. 在每个连接管理器的主机上，为该组数据库服务器添加组条目，为该组 DRDA® 别名添加组条目。向数据库服务器和 DRDA® 别名条目添加组选项。使用 c=1 组条目选项，以便组成员列表中的连接尝试起点是随机的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。
例如：

```
#dbservername   nettype    hostname   servicename   options
 my_servers     group      -          -             c=1,e=server_3
 server_1       onsoctcp   host_1     port_1        g=my_servers
 server_2       onsoctcp   host_2     port_2        g=my_servers
 server_3       onsoctcp   host_3     port_3        g=my_servers

 drda_aliases   group      -          -             c=1,e=drda_3
 drda_1         drsoctcp   host_1     port_4        g=drda_aliases
 drda_2         drsoctcp   host_2     port_5        g=drda_aliases
 drda_3         drsoctcp   host_3     port_6        g=drda_aliases
```

5. 在每个客户端应用程序的主机上，为每个连接管理器配置文件中的每个服务级别协议 (SLA) 创建 sqlhosts 文件条目。

   第一个连接管理器配置文件具有以下条目：

```
NAME connection_manager_1

CLUSTER my_cluster
{
   INFORMIXSERVER my_servers
   SLA sla_primary_1          DBSERVERS=PRI
   SLA sla_primary_drda_1     DBSERVERS=PRI
   SLA sla_secondaries_1      DBSERVERS=SDS,HDR
   SLA sla_secondaries_drda_1 DBSERVERS=SDS,HDR
   FOC ORDER=ENABLED \
       PRIORITY=1
}
```

   第二个连接管理器配置文件具有以下条目：

```
NAME connection_manager_2

CLUSTER my_cluster
{
   INFORMIXSERVER my_servers
   SLA sla_primary_2          DBSERVERS=PRI
   SLA sla_primary_drda_2     DBSERVERS=PRI
   SLA sla_secondaries_2      DBSERVERS=SDS,HDR
   SLA sla_secondaries_drda_2 DBSERVERS=SDS,HDR
   FOC ORDER=ENABLED \
       PRIORITY=2
}
```

   将以下条目添加到每个客户端应用程序的主机 sqlhosts 文件：

```
#dbservername            nettype    hostname   servicename  options
 sla_primary_1           onsoctcp   cm_host_1  cm_port_1
 sla_primary_2           onsoctcp   cm_host_2  cm_port_2

 sla_secondaries_2       onsoctcp   cm_host_1  cm_port_3
 sla_secondaries_2       onsoctcp   cm_host_2  cm_port_4

 sla_primary_1_drda      drsoctcp   cm_host_1  cm_port_5
 sla_primary_2_drda      drsoctcp   cm_host_2  cm_port_6

 sla_secondaries_2_drda  drsoctcp   cm_host_1  cm_port_7
```

```
 sla_secondaries_2_drda  drsoctcp   cm_host_2  cm_port_8
```

6. 在每个客户端应用程序的主机上，为每组 SLA 条目创建 sqlhosts 文件组条目，并且将组选项添加至 SLA 条目。使用 c=1 组条目选项，以便组成员列表中的连接尝试起点是随机的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。

```
#dbservername            nettype    hostname   servicename  options
 g_primary               group      -          -            c=1,e=sla_primary_2
 sla_primary_1           onsoctcp   cm_host_1  cm_port_1    g=g_primary
 sla_primary_2           onsoctcp   cm_host_2  cm_port_2    g=g_primary

 g_secondaries           group      -          -            c=1,e=sla_secondaries_2
 sla_secondaries_2       onsoctcp   cm_host_1  cm_port_3    g=g_secondaries
 sla_secondaries_2       onsoctcp   cm_host_2  cm_port_4    g=g_secondaries

 g_primary_drda          group      -          -            c=1,e=sla_primary_2_drda
 sla_primary_1_drda      drsoctcp   cm_host_1  cm_port_5    g=g_primary_drda
 sla_primary_2_drda      drsoctcp   cm_host_2  cm_port_6    g=g_primary_drda

 g_secondaries_drda      group      -          -            c=1,e=sla_secondaries_2_drda
 sla_secondaries_2_drda  drsoctcp   cm_host_1  cm_port_7    g=g_secondaries_drda
 sla_secondaries_2_drda  drsoctcp   cm_host_2  cm_port_8    g=g_secondaries_drda
```

到 @g_primary_drda 的客户端连接请求由 drsoctcp 协议发送到其中一个连接管理器。如果 connection_manager_1 接收到请求，会使用 sla_primary_1_drda 向客户端应用程序提供主服务器的连接信息。如果 connection_manager_2 接收到请求，会使用 sla_primary_2_drda 向客户端应用程序提供主服务器的连接信息。

相关链接

组信息 在第57页
配置 *SinoDB* 数据库服务器以连接 *IBM Data Server* 客户端 在第66页
sqlhosts 信息 在第47页
《*SinoDB SQL* 指南: 参考》: *INFORMIXSQLHOSTS* 环境变量
《*SinoDB* 管理员参考》: *DBSERVERALIASES* 配置参数

为使用分布式关系数据库结构 (DRDA®) 和安全端口的高可用性集群定义 sqlhosts 信息

连接管理器支持高可用性集群的 Distributed Relational Database Architecture™ (DRDA®) (分布式关系数据库结构) 连接。您必须为使用 DRDA® 的高可用性集群的连接管理定义 sqlhosts 网络连接信息。

连接管理器的 sqlhosts 文件必须包含其连接的所有数据库服务器条目。

要在特定主机上使用 $INFORMIXDIR/etc/sqlhosts 以外的文件，请将主机的 INFORMIXSQLHOSTS 环境变量设置为该替代文件。

1. 在每个连接管理器和数据库服务器的主机上，为集群中每个数据库服务器创建 sqlhosts 文件条目，并指定 s=6 安全端口选项。
   例如：

```
#dbservername  nettype   hostname  servicename  options
 server_1      onsoctcp  host_1    port_1       s=6
 server_2      onsoctcp  host_3    port_2       s=6
 server_3      onsoctcp  host_5    port_3       s=6
```

2. 在每个连接管理器和数据库服务器的主机上，添加 DRDA® 别名条目。nettype 值设置为 DRDA® 协议，并指定 s=6 安全端口选项。
   例如：

```
#dbservername  nettype   hostname  servicename  options
 server_1      onsoctcp  host_1    port_1       s=6
 server_2      onsoctcp  host_2    port_2       s=6
 server_3      onsoctcp  host_3    port_3       s=6
```

```
 drda_1         drsoctcp  host_1    port_4       s=6
 drda_2         drsoctcp  host_2    port_5       s=6
 drda_3         drsoctcp  host_3    port_6       s=6
```

3. 在每个连接管理器和数据库服务器的主机上，为每个数据库服务器和每个 DRDA® 别名创建 sqlhosts 文件别名条目。
   例如：

```
#dbservername  nettype   hostname  servicename  options
 server_1      onsoctcp  host_1    port_1       s=6
 a_server_1    onsoctcp  host_1    port_7

 server_2      onsoctcp  host_2    port_2       s=6
 a_server_2    onsoctcp  host_2    port_8

 server_3      onsoctcp  host_3    port_3       s=6
 a_server_3    onsoctcp  host_3    port_9

 drda_1        drsoctcp  host_1    port_4       s=6
 a_drda_1      drsoctcp  host_1    port_10

 drda_2        drsoctcp  host_2    port_5       s=6
 a_drda_2      drsoctcp  host_2    port_11

 drda_3        drsoctcp  host_3    port_6       s=6
 a_drda_3      drsoctcp  host_3    port_12
```

4. 在每个连接管理器的主机上，为该组数据库服务器和 DRDA® 条目添加组条目。添加组选项到个别条目。使用 c=1 组条目选项，以便组成员列表中的连接尝试起点是随机的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。
   例如：

```
#dbservername  nettype   hostname  servicename  options
 g_servers     group     -         -            c=1,e=a_server_3
 server_1      onsoctcp  host_1    port_1       s=6,g=g_servers
 a_server_1    onsoctcp  host_1    port_7       g=g_servers
 server_2      onsoctcp  host_2    port_2       s=6,g=g_servers
 a_server_2    onsoctcp  host_2    port_8       g=g_servers
 server_3      onsoctcp  host_3    port_3       s=6,g=g_servers
 a_server_3    onsoctcp  host_3    port_9       g=g_servers

 g_drda        group     -         -            c=1,e=a_drda_3
 drda_1        drsoctcp  host_1    port_4       s=6,g=g_drda
 a_drda_1      drsoctcp  host_1    port_10      g=g_drda
 drda_2        drsoctcp  host_2    port_5       s=6,g=g_drda
 a_drda_2      drsoctcp  host_2    port_11      g=g_drda
 drda_3        drsoctcp  host_3    port_6       s=6,g=g_drda
 a_drda_3      drsoctcp  host_3    port_12      g=g_drda
```

5. 在每个数据库服务器的 onconfig 文件中，将 DBSERVERALIASES 参数设置为该数据库服务器的别名。

   server_1 的 onconfig 文件条目：

```
DBSERVERALIASES a_server_1,drda_1,a_drda_1
```

   server_2 的 onconfig 文件条目：

```
DBSERVERALIASES a_server_2,drda_2,a_drda_2
```

server_3 的 onconfig 文件条目：

```
DBSERVERALIASES a_server_3,drda_3,a_drda_3
```

6. 在每个客户端应用程序的主机上，为每个连接管理器配置文件中的每个服务级别协议（SLA）创建一个 sqlhosts 文件条目。

   第一个连接管理器配置文件具有以下条目：

```
NAME connection_manager_1

CLUSTER my_cluster
{
   INFORMIXSERVER g_servers,g_drda
   SLA sla_primary_1          DBSERVERS=PRI
   SLA sla_primary_drda_1     DBSERVERS=PRI
   SLA sla_secondaries_1      DBSERVERS=SDS,HDR
   SLA sla_secondaries_drda_1 DBSERVERS=SDS,HDR
   FOC ORDER=ENABLED \
       PRIORITY=1
}
```

   第二个连接管理器配置文件具有以下条目：

```
NAME connection_manager_2

CLUSTER my_cluster
{
   INFORMIXSERVER g_servers,g_drda
   SLA sla_primary_2          DBSERVERS=PRI
   SLA sla_primary_drda_2     DBSERVERS=PRI
   SLA sla_secondaries_2      DBSERVERS=SDS,HDR
   SLA sla_secondaries_drda_2 DBSERVERS=SDS,HDR
   FOC ORDER=ENABLED \
       PRIORITY=2
}
```

   将以下条目添加到每个客户端应用程序的主机 sqlhosts 文件：

```
#dbservername            nettype   hostname   servicename  options
 sla_primary_1           onsoctcp  cm_host_1  cm_port_1
 sla_primary_2           onsoctcp  cm_host_2  cm_port_2

 sla_secondaries_2       onsoctcp  cm_host_1  cm_port_3
 sla_secondaries_2       onsoctcp  cm_host_2  cm_port_4

 sla_primary_1_drda      drsoctcp  cm_host_1  cm_port_5
 sla_primary_2_drda      drsoctcp  cm_host_2  cm_port_6

 sla_secondaries_2_drda  drsoctcp  cm_host_1  cm_port_7
 sla_secondaries_2_drda  drsoctcp  cm_host_2  cm_port_8
```

7. 在每个客户端应用程序的主机上，为每组 SLA 条目创建 sqlhosts 文件组条目，并且将组选项添加至 SLA 条目。使用 c=1 组条目选项，以便组成员列表中的连接尝试起点是随机的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。

```
#dbservername            nettype    hostname   servicename  options
 g_primary               group      -          -            c=1,e=sla_primary_2
 sla_primary_1           onsoctcp   cm_host_1  cm_port_1    g=g_primary
 sla_primary_2           onsoctcp   cm_host_2  cm_port_2    g=g_primary

 g_secondaries           group      -          -            c=1,e=sla_secondaries_2
```

```
sla_secondaries_2       onsoctcp   cm_host_1   cm_port_3    g=g_secondaries
sla_secondaries_2       onsoctcp   cm_host_2   cm_port_4    g=g_secondaries

g_primary_drda          group      -           -            c=1,e=sla_primary_2_drda
sla_primary_1_drda      drsoctcp   cm_host_1   cm_port_5    g=g_primary_drda
sla_primary_2_drda      drsoctcp   cm_host_2   cm_port_6    g=g_primary_drda

g_secondaries_drda      group      -           -            c=1,e=sla_secondaries_2_drda
sla_secondaries_2_drda  drsoctcp   cm_host_1   cm_port_7    g=g_secondaries_drda
sla_secondaries_2_drda  drsoctcp   cm_host_2   cm_port_8    g=g_secondaries_drda
```

到 @g_primary_drda 的客户端连接请求由 drsoctcp 协议发送到其中一个连接管理器。如果 connection_manager_1 接收到请求，会使用 sla_primary_1_drda 向客户端应用程序提供主服务器的连接信息。如果 connection_manager_2 接收到请求，会使用 sla_primary_2_drda 向客户端应用程序提供主服务器的连接信息。

相关链接

组信息 在第57页
配置 *SinoDB* 数据库服务器以连接 *IBM Data Server* 客户端 在第66页
sqlhosts 信息 在第47页
《*SinoDB SQL* 指南: 参考》: *INFORMIXSQLHOSTS* 环境变量
《*SinoDB* 管理员参考》: *DBSERVERALIASES* 配置参数

为网格和复制集的连接管理定义 sqlhosts 信息
您必须为复制集或网格的连接管理定义 sqlhosts 网络连接信息。

要在特定主机上使用 $INFORMIXDIR/etc/sqlhosts 以外的文件，请将主机的 INFORMIXSQLHOSTS 环境变量设置为该替代文件。

1. 在每个连接管理器和复制服务器的主机上，为每个复制服务器创建 sqlhosts 文件条目。
   例如：

```
#dbservername  nettype   hostname   servicename  options
 server_1      onsoctcp  host_1     port_1
 server_2      onsoctcp  host_2     port_2
 server_3      onsoctcp  host_3     port_3
 server_4      onsoctcp  host_4     port_4
```

2. 在每个连接管理器和复制服务器的主机上，为每个复制服务器创建一个 sqlhosts 文件组条目。向每个复制服务器条目添加组选项。使用 i=*unique_number* 组条目选项，为 Enterprise Replication 分配组标识符。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。
   例如：

```
#dbservername  nettype   hostname   servicename  options
 g_server_1    group     -          -            i=1,e=server_1
 server_1      onsoctcp  host_1     port_1       g=g_server_1

 g_server_2    group     -          -            i=2,e=server_2
 server_2      onsoctcp  host_2     port_2       g=g_server_2

 g_server_3    group     -          -            i=3,e=server_3
 server_3      onsoctcp  host_3     port_3       g=g_server_3

 g_server_4    group     -          -            i=4,e=server_4
 server_4      onsoctcp  host_4     port_4       g=g_server_4
```

3. 在每个客户端应用程序的主机上，为每个连接管理器配置文件中的每个服务级别协议（SLA）创建一个 sqlhosts 文件条目。

第一个连接服务器配置文件具有以下条目：

```
NAME connection_manager_1

REPLSET my_replset
{
   INFORMIXSERVER g_server_1,g_server_2,g_server_3,g_server_4
   SLA sla_1 DBSERVERS=ANY
}
```

第二个连接服务器配置文件具有以下条目：

```
NAME connection_manager_2

REPLSET my_replset
{
   INFORMIXSERVER g_server_1,g_server_2,g_server_3,g_server_4
   SLA sla_2 DBSERVERS=ANY
}
```

向每个客户端应用程序的主机 sqlhosts 文件添加以下条目：

```
#dbservername  nettype   hostname   servicename  options
 sla_1         onsoctcp  cm_host_1  cm_port_1
 sla_2         onsoctcp  cm_host_2  cm_port_2
```

4. 在每个客户端应用程序的主机上，为每个 SLA 条目组创建 sqlhosts 文件组条目，并且将组选项添加至 SLA 条目中。使用 c=1 组条目选项，使得在组成员列表中连接尝试的起点是随机决定的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。
例如：

```
#dbservername  nettype   hostname   servicename  options
 g_sla         onsoctcp  -          -             c=1,e=sla_2
 sla_1         onsoctcp  cm_host_1  cm_port_1     g=g_sla
 sla_2         onsoctcp  cm_host_2  cm_port_2     g=g_sla
```

向 @g_sla 发出的客户端连接请求将定向至其中一个连接管理器。如 connection_manager_1 连接管理器接到该请求，将使用 sla_1 向客户端应用程序提供主服务器的连接信息。如 connection_manager_2 连接管理器接到请求，将使用 sla_2 向客户端应用程序提供复制服务器的连接信息。

相关链接

组信息 在第57页
sqlhosts 信息 在第47页
《*SinoDB SQL* 指南: 参考》: *INFORMIXSQLHOSTS* 环境变量

为使用安全端口的网格和复制集的连接管理定义 sqlhosts 信息
您必须为复制集或网格连接管理定义 sqlhosts 网络连接信息。

要在特定主机上使用 $INFORMIXDIR/etc/sqlhosts 以外的文件，请将主机的 INFORMIXSQLHOSTS 环境变量设置为该替代文件。

1. 在每个连接管理器和复制服务器的主机上，为每个复制服务器创建 sqlhosts 文件条目，并指定 s=6 安全端口选项。
例如：

```
#dbservername  nettype   hostname   servicename  options
 server_1      onsoctcp  host_1     port_1       s=6
 server_2      onsoctcp  host_2     port_3       s=6
 server_3      onsoctcp  host_3     port_5       s=6
 server_4      onsoctcp  host_4     port_7       s=6
```

2. 在每个连接管理器和复制服务器的主机上，为每个复制服务器创建一个 sqlhosts 文件别名条目。
例如：

```
#dbservername  nettype   hostname  servicename  options
 server_1      onsoctcp  host_1    port_1       s=6
 a_server_1    onsoctcp  host_1    port_2

 server_2      onsoctcp  host_2    port_3       s=6
 a_server_2    onsoctcp  host_2    port_4

 server_3      onsoctcp  host_3    port_5       s=6
 a_server_3    onsoctcp  host_3    port_6

 server_4      onsoctcp  host_4    port_7       s=6
 a_server_4    onsoctcp  host_4    port_8
```

cdr 实用程序使用的别名，不能连接到安全端口。

3. 在每个连接管理器和复制服务器的主机上，为每个复制服务器和别名对创建一个 sqlhosts 文件组条目。使用 i=*unique_number* 组条目选项，为 Enterprise Replication 分配组标识符。向复制服务器和别名条目添加组选项。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。
例如：

```
#dbservername  nettype   hostname   servicename  options
 g_server_1    group     -          -            i=1,e=a_server_1
 server_1      onsoctcp  host_1     port_1       g=g_server_1,s=6
 a_server_1    onsoctcp  host_1     port_2       g=g_server_1

 g_server_2    group     -          -            i=2,e=a_server_2
 server_2      onsoctcp  host_2     port_3       g=g_server_2,s=6
 a_server_2    onsoctcp  host_2     port_4       g=g_server_2

 g_server_3    group     -          -            i=3,e=a_server_3
 server_3      onsoctcp  host_3     port_5       g=g_server_3,s=6
 a_server_3    onsoctcp  host_3     port_6       g=g_server_3

 g_server_4    group     -          -            i=4,e=a_server_4
 server_4      onsoctcp  host_4     port_7       g=g_server_4,s=6
 a_server_4    onsoctcp  host_4     port_8       g=g_server_4
```

4. 在每个复制服务器的 onconfig 文件中，将 DBSERVERALIASES 参数设置为服务器的别名。

server_1 的 onconfig 文件条目：

```
DBSERVERALIASES a_server_1
```

server_2 的 onconfig 文件条目：

```
DBSERVERALIASES a_server_2
```

server_3 的 onconfig 文件条目：

```
DBSERVERALIASES a_server_3
```

server_4 的 onconfig 文件条目：

```
DBSERVERALIASES a_server_4
```

5. 在每个客户端应用程序的主机上，为每个连接管理器配置文件的每一服务级别协议（SLA）创建一个 sqlhosts 文件条目。

第一个连接服务器配置文件具有以下条目：

```
NAME connection_manager_1

REPLSET my_replset
{
   INFORMIXSERVER g_server_1,g_server_2,g_server_3,g_server_4
   SLA sla_1 DBSERVERS=ANY
}
```

第二个连接服务器配置文件具有以下条目：

```
NAME connection_manager_2

REPLSET my_replset
{
   INFORMIXSERVER g_server_1,g_server_2,g_server_3,g_server_4
   SLA sla_2 DBSERVERS=ANY
}
```

向每个客户端应用程序的主机 sqlhosts 文件添加以下条目：

```
#dbservername  nettype   hostname   servicename  options
 sla_1         onsoctcp  cm_host_1  cm_port_1
 sla_2         onsoctcp  cm_host_2  cm_port_2
```

6. 在每个客户端应用程序的主机上，为 SLA 条目的每个组创建 sqlhosts 文件组条目，并且向 SLA 条目添加组选项。使用 c=1 组条目选项，以使组成员列表中连接尝试的起点是随机决定的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。
例如：

```
#dbservername  nettype   hostname   servicename  options
 g_sla         group     -          -            c=1,e=sla_2
 sla_1         onsoctcp  cm_host_1  cm_port_1    g=g_sla
 sla_2         onsoctcp  cm_host_2  cm_port_2    g=g_sla
```

向 @g_sla 发出的客户端连接请求将定向至其中一个连接管理器。如 connection_manager_1 连接管理器接到请求，将使用 sla_1 向客户端应用程序提供主服务器的连接信息。如 connection_manager_2 连接管理器接到请求，将使用 sla_2 向客户端应用程序提供主服务器的连接信息。

相关链接

组信息 在第57页
sqlhosts 信息 在第47页
《*SinoDB SQL* 指南: 参考》: *INFORMIXSQLHOSTS* 环境变量
《*SinoDB* 管理员参考》: *DBSERVERALIASES* 配置参数

为高可用性复制系统的连接管理定义 sqlhosts 信息
您必须为高可用性复制系统的连接管理定义 sqlhosts 网络连接信息。

要在特定的主机上使用 $INFORMIXDIR/etc/sqlhosts 以外的文件，请将主机的 INFORMIXSQLHOSTS 环境变量设置为该替代文件。

在此示例中，您将三个高可用性集群的主服务器之间设置 Enterprise Replication。

集群 1：

- server_1 （主服务器）
- server_2 （SD 辅助服务器）
- server_3 （HDR 辅助服务器）
- server_4 （RS 辅助服务器）

集群 2：

- server_5（主服务器）
- server_6（SD 辅助服务器）
- server_7（HDR 辅助服务器）
- server_8 （RS 辅助服务器）

集群 3：

- server_9 （主服务器）
- server_10 （SD 辅助服务器）
- server_11 （HDR 辅助服务器）
- server_12 （RS 辅助服务器）

1. 在每个连接管理器和数据库服务器的主机上，为每个服务器创建 sqlhosts 文件条目。
   例如：

```
#dbservername  nettype   hostname   servicename  options
server_1       onsoctcp  host_1     port_1
server_2       onsoctcp  host_1     port_2
server_3       onsoctcp  host_2     port_3
server_4       onsoctcp  host_3     port_4

server_5       onsoctcp  host_4     port_5
server_6       onsoctcp  host_4     port_6
server_7       onsoctcp  host_5     port_7
server_8       onsoctcp  host_6     port_8

server_9       onsoctcp  host_7     port_9
server_10      onsoctcp  host_7     port_10
server_11      onsoctcp  host_8     port_11
server_12      onsoctcp  host_9     port_12
```

2. 在每个连接管理器和数据库服务器的主机上，为每个复制服务器和每个集群的创建 sqlhosts 文件组条目。添加组选项到每个数据库服务器条目。使用 i=*unique_number* 组条目选项，为 Enterprise Replication 分配组标识符。对集群组使用 c=1 组条目选项，以使在组成员名单中连接尝试的起始点是随机决定的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。
   例如：

```
#dbservername  nettype   hostname   servicename  options
 cluster_1     group     -          -            i=1,c=1,e=server_4
 server_1      onsoctcp  host_1     port_1       g=cluster_1
 server_2      onsoctcp  host_1     port_2       g=cluster_1
 server_3      onsoctcp  host_2     port_3       g=cluster_1
 server_4      onsoctcp  host_3     port_4       g=cluster_1

 cluster_2     group     -          -            i=2,c=1,e=server_8
 server_5      onsoctcp  host_4     port_5       g=cluster_2
 server_6      onsoctcp  host_4     port_6       g=cluster_2
 server_7      onsoctcp  host_5     port_7       g=cluster_2
 server_8      onsoctcp  host_6     port_8       g=cluster_2

 cluster_3     group     -          -            i=3,c=1,e=server_12
 server_9      onsoctcp  host_7     port_9       g=cluster_3
 server_10     onsoctcp  host_7     port_10      g=cluster_3
 server_11     onsoctcp  host_8     port_11      g=cluster_3
 server_12     onsoctcp  host_9     port_12      g=cluster_3
```

3. .在每个客户端应用程序的主机上，为每个连接管理器配置文件中的每个服务级别协议（SLA）创建一个 sqlhosts 文件条目。

第一个连接管理器配置文件具有以下条目：

```
NAME connection_manager_1

REPLSET my_replset
{
   INFORMIXSERVER cluster_1,cluster_2,cluster_3
   SLA sla_1 DBSERVERS=ANY
}

CLUSTER my_cluster_1
{
   INFORMIXSERVER cluster_1
   FOC ORDER=ENABLED \
       PRIORITY=1
}

CLUSTER my_cluster_2
{
   INFORMIXSERVER cluster_2
   FOC ORDER=ENABLED \
       PRIORITY=1
}

CLUSTER my_cluster_3
{
   INFORMIXSERVER cluster_3
   FOC ORDER=ENABLED \
       PRIORITY=1
}
```

第二个连接管理器配置文件具有以下条目：

```
NAME connection_manager_2

REPLSET my_replset
{
   INFORMIXSERVER cluster_1,cluster_2,cluster_3
   SLA sla_2 DBSERVERS=ANY
}

CLUSTER my_cluster_1
{
   INFORMIXSERVER cluster_1
   FOC ORDER=ENABLED \
       PRIORITY=2
}

CLUSTER my_cluster_2
{
   INFORMIXSERVER cluster_2
   FOC ORDER=ENABLED \
       PRIORITY=2
}

CLUSTER my_cluster_3
{
   INFORMIXSERVER cluster_3
   FOC ORDER=ENABLED \
       PRIORITY=2
}
```

将以下条目添加到每个客户端应用程序的主机 sqlhosts 文件：

```
#dbservername  nettype   hostname   servicename  options
 sla_1         onsoctcp  cm_host_1  cm_port_1
 sla_2         onsoctcp  cm_host_2  cm_port_2
```

4. 在每个客户端应用程序的主机上，为 SLA 条目的每个组创建 sqlhosts 文件组条目，并添加组选项到 SLA 条目。使用 c=1 组条目选项，以使在组成员名单中连接尝试的起始点是随机决定的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。
例如：

```
#dbservername  nettype   hostname   servicename  options
 g_sla         onsoctcp  -          -            c=1,e=sla_2
 sla_1         onsoctcp  cm_host_1  cm_port_1    g=g_sla
 sla_2         onsoctcp  cm_host_2  cm_port_2    g=g_sla
```

向 @g_sla 发出的客户端连接请求，将定向到其中一个连接管理器。如 connection_manager_1 接到该请求，将使用 sla_1 向客户端应用程序提供主服务器的连接信息。如 connection_manager_2 接到该请求，将使用 sla_2 向客户端应用程序提供主服务器的连接信息。

相关链接

组信息 在第57页
sqlhosts 信息 在第47页
《*SinoDB Enterprise Replication* 指南》: 高可用性复制系统
《*SinoDB SQL* 指南: 参考》: *INFORMIXSQLHOSTS* 环境变量

为使用安全端口的高可用性复制系统的连接管理定义 sqlhosts 信息
您必须为高可用性复制系统的连接管理定义 sqlhosts 网络连接信息。

要在特定主机上使用 $INFORMIXDIR/etc/sqlhosts 以外的文件，请将主机的 INFORMIXSQLHOSTS 环境变量设置为该替代文件。

在此示例中，您将设置两个高可用性集群的主服务器之间的 Enterprise Replication。

集群 1：

- server_1 （主服务器）
- server_2 （SD 辅助服务器）
- server_3 （HDR 辅助服务器）
- server_4 （RS 辅助服务器）

集群 2：

- server_5 （主服务器）
- server_6 （SD 辅助服务器）
- server_7 （HDR 辅助服务器）
- server_8 （RS 辅助服务器）

1. 在每个连接管理器和数据库服务器的主机上，为每个数据库服务器创建 sqlhosts 文件条目。
例如：

```
#dbservername  nettype   hostname  servicename  options
 server_1      onsoctcp  host_1    port_1       s=6
 server_2      onsoctcp  host_1    port_2       s=6
 server_3      onsoctcp  host_2    port_3       s=6
 server_4      onsoctcp  host_3    port_4       s=6

 server_5      onsoctcp  host_4    port_5       s=6
 server_6      onsoctcp  host_4    port_6       s=6
 server_7      onsoctcp  host_5    port_7       s=6
 server_8      onsoctcp  host_6    port_8       s=6
```

2. 在每个连接管理器和数据库服务器的主机上，为每个数据库服务器创建 sqlhosts 文件别名。
例如 ：

```
#dbservername  nettype   hostname  servicename  options
 server_1      onsoctcp  host_1    port_1       s=6
 a_server_1    onsoctcp  host_1    port_9

 server_2      onsoctcp  host_1    port_2       s=6
 a_server_2    onsoctcp  host_1    port_10

 server_3      onsoctcp  host_2    port_3       s=6
 a_server_3    onsoctcp  host_2    port_11

 server_4      onsoctcp  host_3    port_4       s=6
 a_server_4    onsoctcp  host_3    port_12

 server_5      onsoctcp  host_4    port_5       s=6
 a_server_5    onsoctcp  host_4    port_13

 server_6      onsoctcp  host_4    port_6       s=6
 a_server_6    onsoctcp  host_4    port_14

 server_7      onsoctcp  host_5    port_7       s=6
 a_server_7    onsoctcp  host_5    port_15

 server_8      onsoctcp  host_6    port_8       s=6
 a_server_8    onsoctcp  host_6    port_16
```

cdr 实用程序使用的别名不能连接安全端口。

3. 在每个连接管理器和数据库服务器的主机上，为每个集群创建一个 sqlhosts 文件组条目。添加组选项到每个数据库服务器条目。使用 i=*unique_number* 组条目选项，为 Enterprise Replication 分配组标识符。对集群组使用 c=1 组条目选项，以使在组成员名单中连接尝试的起始点是随机决定的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。
例如：

```
#dbservername  nettype   hostname  servicename  options
 cluster_1     group     -          -           i=1,c=1,e=a_server_4
 server_1      onsoctcp  host_1    port_1       s=6,g=cluster_1
 a_server_1    onsoctcp  host_1    port_9       g=cluster_1
 server_2      onsoctcp  host_1    port_2       s=6,g=cluster_1
 a_server_2    onsoctcp  host_1    port_10      g=cluster_1
 server_3      onsoctcp  host_2    port_3       s=6,g=cluster_1
 a_server_3    onsoctcp  host_2    port_11      g=cluster_1
 server_4      onsoctcp  host_3    port_4       s=6,g=cluster_1
 a_server_4    onsoctcp  host_3    port_12      g=cluster_1

 cluster_2     group     -          -           i=1,c=1,e=a_server_8
 server_5      onsoctcp  host_4    port_5       s=6,g=cluster_2
 a_server_5    onsoctcp  host_4    port_13      g=cluster_2
 server_6      onsoctcp  host_4    port_6       s=6,g=cluster_2
 a_server_6    onsoctcp  host_4    port_14      g=cluster_2
 server_7      onsoctcp  host_5    port_7       s=6,g=cluster_2
 a_server_7    onsoctcp  host_5    port_15      g=cluster_2
 server_8      onsoctcp  host_6    port_8       s=6,g=cluster_2
 a_server_8    onsoctcp  host_6    port_16      g=cluster_2
```

4. 在每个数据库服务器的 onconfig 文件中，将 DBSERVERALIASES 参数设置为该数据库服务器的别名。
例如 ：

server_1 的 onconfig 文件条目为：

```
DBSERVERALIASES a_server_1
```

5. 在每个客户端应用程序的主机上，为每个连接管理器配置文件中的每个服务级别协议（SLA）创建一个 sqlhosts 文件条目。

   第一个连接管理器配置文件具有以下条目 ：

```
NAME connection_manager_1

REPLSET my_replset
{
   INFORMIXSERVER cluster_1,cluster_2
   SLA sla_1 DBSERVERS=ANY
}

CLUSTER my_cluster_1
{
   INFORMIXSERVER cluster_1
   FOC ORDER=ENABLED \
       PRIORITY=1
}

CLUSTER my_cluster_2
{
   INFORMIXSERVER cluster_2
   FOC ORDER=ENABLED \
       PRIORITY=1
}
```

   第二个连接管理器配置文件具有以下条目：

```
NAME connection_manager_2

REPLSET my_replset
{
   INFORMIXSERVER cluster_1,cluster_2
   SLA sla_2 DBSERVERS=ANY
}

CLUSTER my_cluster_1
{
   INFORMIXSERVER cluster_1
   FOC ORDER=ENABLED \
       PRIORITY=2
}

CLUSTER my_cluster_2
{
   INFORMIXSERVER cluster_2
   FOC ORDER=ENABLED \
       PRIORITY=2
}
```

   将以下条目添加到每个客户端应用程序的主机 sqlhosts 文件：

```
#dbservername  nettype   hostname   servicename  options
 sla_1         onsoctcp  cm_host_1  cm_port_1
 sla_2         onsoctcp  cm_host_2  cm_port_2
```

6. 在每个客户端应用程序的主机上，为 SLA 条目的每个组创建 sqlhosts 文件组条目，并向 SLA 条目添加组选项。使用 c=1 组条目选项，以使在组成员名单中连接尝试的起始点是随机决定的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。
例如 ：

```
#dbservername  nettype   hostname   servicename  options
 g_sla         onsoctcp  -          -             c=1,e=sla_2
 sla_1         onsoctcp  cm_host_1  cm_port_1     g=g_sla
 sla_2         onsoctcp  cm_host_2  cm_port_2     g=g_sla
```

向 @g_sla 发出的客户端连接请求，将定向到其中一个连接管理器。如 connection_manager_1 接到该请求，将使用 sla_1 向客户端应用程序提供主服务器的连接信息。果 connection_manager_2 接到该请求，将使用 sla_2 向客户端应用程序提供主服务器的连接信息。

相关链接

组信息 在第57页
sqlhosts 信息 在第47页
《*SinoDB Enterprise Replication* 指南》: 高可用性复制系统
《*SinoDB SQL* 指南: 参考》: *INFORMIXSQLHOSTS* 环境变量
《*SinoDB* 管理员参考》: *DBSERVERALIASES* 配置参数

为服务器集的连接管理定义 sqlhosts 信息
您必须为服务器集的连接管理定义 sqlhosts 网络连接信息

连接管理器的 sqlhosts 文件必须包含其连接的所有数据库服务器条目。

要在特定主机上使用 $INFORMIXDIR/etc/sqlhosts 以外的文件，请将主机的 INFORMIXSQLHOSTS 环境变量设置为该替代文件。

1. 0在每个连接管理器和数据库服务器主机上，为每个数据库服务器创建 sqlhosts 文件条目。
例如：

```
#dbservername   nettype    hostname   servicename   options
 standalone_1   onsoctcp   host_1     port_1

 standalone_2   onsoctcp   host_2     port_2

 standalone_3   onsoctcp   host_3     port_3
```

2. 在每个客户端应用程序的主机上，为每个连接管理器配置文件中的每个服务级别协议（SLA）创建一个 sqlhosts 文件条目。

第一个连接服务器配置文件具有以下条目：

```
NAME connection_manager_1
MACRO servers=standalone_1,standalone_2,standalone_3

SERVERSET ${servers}
{
   INFORMIXSERVER ${servers}
   SLA sla_1a DBSERVERS=standalone_1
   SLA sla_2a DBSERVERS=standalone_2
   SLA sla_3a DBSERVERS=standalone_3
}
```

第二个连接服务器配置文件具有以下条目：

```
NAME connection_manager_2
MACRO servers=standalone_1,standalone_2,standalone_3

SERVERSET ${servers}
```

```
{
   INFORMIXSERVER ${servers}
   SLA sla_1b DBSERVERS=standalone_1
   SLA sla_2b DBSERVERS=standalone_2
   SLA sla_3b DBSERVERS=standalone_3
}
```

将以下条目添加到每个客户端应用程序的主机 sqlhosts 文件：

```
#dbservername  nettype   hostname   servicename  options
 sla_1a        onsoctcp  cm_host_1  cm_port_1
 sla_1b        onsoctcp  cm_host_2  cm_port_2

 sla_2a        onsoctcp  cm_host_1  cm_port_3
 sla_2b        onsoctcp  cm_host_2  cm_port_4

 sla_3a        onsoctcp  cm_host_1  cm_port_5
 sla_3b        onsoctcp  cm_host_2  cm_port_6
```

3. 在每个客户端应用程序的主机上，为 SLA 条目的每个组创建 sqlhosts 文件组条目，并且将组选项添加至 SLA 条目中。使用 c=1 组条目选项，以使组成员名单中连接尝试的起始点是随机决定的。使用 e=*last_member* 组条目选项，以便不会为搜索组成员而扫描整个 sqlhosts。
例如：

```
#dbservername  nettype   hostname   servicename  options
 sla_1         group     -          -            c=1,e=sla_1b
 sla_1a        onsoctcp  cm_host_1  cm_port_1    g=sla_1
 sla_1b        onsoctcp  cm_host_2  cm_port_2    g=sla_1

 sla_2         group     -          -            c=1,e=sla_2b
 sla_2a        onsoctcp  cm_host_1  cm_port_3    g=sla_2
 sla_2b        onsoctcp  cm_host_2  cm_port_4    g=sla_2

 sla_3         group     -          -            c=1,e=sla_3b
 sla_3a        onsoctcp  cm_host_1  cm_port_5    g=sla_3
 sla_3b        onsoctcp  cm_host_2  cm_port_6    g=sla_3
```

向 @sla_1 发出的连接请求，将定向至其中一个连接管理器。如 connection_manager_1 接到该请求，将使用 sla_1a 向客户端应用程序提供主服务器的连接信息。如 connection_manager_2 接到该请求，将使用 sla_1b 向客户端应用程序提供复制服务器的连接信息。

相关链接

组信息 在第57页
sqlhosts 信息 在第47页
《*SinoDB SQL* 指南: 参考》: *INFORMIXSQLHOSTS* 环境变量

启动 UNIX™ 和 Linux™ 上的连接管理器

使用 oncmsm 实用程序来启动连接管理器。

连接管理器能在其管理的数据库服务器启动之前启动。如果连接单元由连接管理器管理，则在数据库服务器启动后，连接管理器将连接到连接单元中的数据库服务器。

如果连接单元的数据库服务器在连接管理器初始化之前为联机状态，请启动连接管理器，然后将客户端应用程序请求定向到连接管理器而不是数据库服务器。

1. 通过运行带有 c 参数和连接管理器配置文件名称的 oncmsm 实用程序来启动连接管理器。

```
oncmsm -c configuration_file
```

配置文件的缺省位置是 $INFORMIXDIR/etc 目录。

2. 如果您通过在连接管理器的配置文件中设置 LOG 和 LOGFILE 参数启用了连接管理器日志记录，那么请检查日志以确认连接管理器已成功启动。
   如果连接管理器启动，将显示以下消息：

   ```
   Connection Manager started successfully
   ```

   连接管理器初始化后，它将尝试连接到连接管理器配置文件中 INFORMIXSERVER 参数所指定的数据库服务器或数据库服务器组。然后连接管理器会搜索并连接到每个指定服务器的集群、 网格或复制集中的所有数据库服务器。

相关链接

《*SinoDB 管理员参考*》：*oncmsm 实用程序*

在 Windows™上启动连接管理器

使用 oncmsm 实用程序启动连接管理器。

连接管理器可在其管理的数据库服务器启动之前启动。如果连接单元由连接管理器管理，则在数据库服务器启动后，连接管理器将连接到连接单元中的数据库服务器。

如果连接单元的数据库服务器在连接管理器初始化之前处于联机状态，请启动连接管理器，然后将客户端应用程序的请求定向至连接管理器而非数据库服务器。

1. 使用 i 参数、c 参数和连接管理器配置文件名运行 oncmsm 实用程序，将连接管理器安装成为一个服务。

   ```
   oncmsm -i -c configuration_file
   ```

   配置文件的缺省位置是 %INFORMIXDIR%\etc 目录。
2. 使用连接管理器配置文件指定的连接管理器名运行 oncmsm 实用程序来初始化连接管理器。

   ```
   oncmsm connection_manager_name
   ```

   显示以下消息：

   ```
   Specify the user and password to run this service.
   Press <ENTER> to run Connection Manager as 'localsystem'
   ```

3. 输入 informix 用户 ID 和密码，或按下 Enter 以自动创建 informix-admin 组，并为其分配访问权限。
4. 如果已通过连接管理器配置文件里的 LOG 和 LOGFILE 参数设置启用了连接管理器日志记录，请检查日志确认已成功启动连接管理器。
   如果连接管理器已启动，将显示以下消息：

   ```
   Connection Manager started successfully
   ```

   连接管理器初始化后，将尝试连接管理器配置文件中 INFORMIXSERVER 参数指定的数据库服务器或数据库服务器组。然后，连接管理器将搜索与连接每个指定服务器的集群、网格或复制集中的所有 数据库服务器。

相关链接

《*SinoDB 管理员参考*》：*oncmsm 实用程序*

停止连接管理

当不再想用连接管理器来管理连接单元时，运行 oncmsm 实用程序停止连接管理器实例。

如果有多个连接管理器，您可以运行 onstat -g cmsm 命令以显示连接管理器实例的名称。

1. 登录到连接管理器实例正在运行的计算机。
2. 使用 -k 参数运行 oncmsm 实用程序。
   例如：

   ```
   oncmsm -k connection_manager_name
   ```

3. 如果数据质量（QOD）监测功能处于开启状态，而您想要停止该功能，请在主服务器上运行 cdr stop qod 命令。
   例如：

```
cdr stop qod -c server_1
```

连接管理器停止。

要从 Windows™ 操作系统中卸载已停止的连接管理器，请在安装了连接管理器的计算机上运行以下命令 ：

```
oncmsm -u connection_manager_name
```

相关链接

《*SinoDB* 管理员参考》*: oncmsm* 实用程序

连接管理的监视和故障排除

工具可用来监控连接管理，并帮助您诊断潜在问题。

以下选项可用于连接管理监视和故障排除 ：

- 使用 SinoDB® 开放管理工具 (OAT) 来监控连接管理。
- 在连接管理器配置文件中设置 LOG 和 LOGFILE 参数。日志文件包含有关服务级别协议、 故障转移配置和状态信息的信息。连接管理器启动时会显示日志文件的位置。
- 运行 onstat -g cmsm 以显示有关连接管理器实例的信息。
- 在连接管理器配置文件中设置 CMALARMPROGRAM 参数，并配置 cmalarmprogram 脚本来处理事件警报。
- 如果连接管理器发出事件警报 ：
  - INFORMIXCMNAME 环境变量存储了发出警报的连接管理器实例的名称。
  - INFORMIXCMCONUNITNAME 环境变量存储了发出警报的连接管理器连接单元的名称。

相关链接

《*SinoDB* 管理员参考》*: oncmsm* 实用程序

《*SinoDB* 管理员参考》*: onstat -g cmsm command:* 显示连接管理器信息

提高连接管理可用性的策略

可以增加客户端/服务器通信环境的弹性。

在客户端、连接管理器和数据库服务器主机上安装多个网络接口卡（NIC）

可以通过在连接管理域中每台主机上安装多个 NIC 以防止网络连接失败。如果某个 NIC 出现故障，主机可以使用其他可用的 NIC。

安装多个连接管理器

可以通过安装多个连接管理器来管理域以防止连接管理器成为您的系统中的一个单点故障。

安装独立于数据库服务器的连接管理器

在没有运行数据库服务器的主机上安装连接管理器，以防止如果主机出现故障时连接管理器和数据库服务器同时故障。

在连接管理器服务级别协议中使用关键字

如果连接管理器管理高可用性集群，您可以使用集群关键字在故障转移之后保持连接的一致性。如果集群中的数据库服务器在故障转移之后切换角色，连接管理器可以使用集群关键字继续将客户端连接请求定向到适当的集群服务器类型。

集群关键字为：

- HDR - 高可用性数据复制服务器
- RSS - 远程独立辅助服务器
- SDS - 共享磁盘辅助服务器
- PRI - 主服务器
- PRIMARY - 主服务器

在 sqlhosts 文件和连接管理器配置文件中使用组条目

在管理高可用性集群的每个连接管理器的主机 sqlhosts 文件中创建数据库-服务器组。然后，在连接管理器配置文件中指定 sqlhosts 组，而不是单个服务器名称。如果您将连接单元的 INFORMIXSERVER 参数指定为数据库服务器名称，而且这些指定的服务器在连接管理器重启时处于脱机状态，那么连接管理器是无法重新连接到集群的。如果您将连接单元的 INFORMIXSERVER 参数指定为 sqlhosts 组，而且在连接管理器重启时组的数据库服务器至少有一个是处于联机状态，那么连接管理器可以重新连接到集群。

为 sqlhosts 组条目指定 c=1 选项，以使组成员名单中的连接尝试起始点是随机决定的。例如:

```
#dbservername   nettype    hostname   servicename   options
 my_servers     -          -                        c=1
 server_1       onsoctcp   host_1     port_1        g=my_servers
 server_2       onsoctcp   host_2     port_2        g=my_servers
 server_3       onsoctcp   host_3     port_3        g=my_servers
```

相关链接

组信息 在第57页
《SinoDB 管理员参考》:HA_FOC_ORDER 配置参数
sqlhosts 信息 在第47页

连接管理的配置示例

以下示例演示各种连接单元和系统的连接管理设置步骤。

您可以使用这些示例作为基础，开发您自己的连接管理系统。

高可用性集群的配置连接管理示例

此示例演示高可用性集群配置连接管理的步骤。

在此例子中，在可信任网络上有一个高可用性集群。该集群由三个服务器组成：

- 主服务器（server_1）
- 共享磁盘辅助服务器（server_2）
- HDR 辅助服务器（server_3）

该集群支持以下应用程序服务：

- 联机事务处理（OLTP），只能在主服务器上运行
- Payroll 服务，可在主服务器或 HDR 辅助服务器上运行
- Reporting 服务，可在任一辅助服务器上运行

您的系统有以下需求 ：

- 平衡的数据库服务器工作负载。

- 连接管理器控制故障转移。
- 如果发生故障转移，SD 辅助服务器优先于 HDR 辅助服务器。
- 如果主服务器发生故障，连接管理器在重新启动后仍可连接到集群。
- 该系统可以承受连接管理器故障。
- 该系统可以承受每个主机上的网络接口卡（NIC）故障。

配置连接管理 ：

1. 在每台主机上安装至少两个网络接口卡。

   这可以防止网络接口卡的故障导致连接管理器或数据库服务器故障。
2. 安装两个连接管理器。每个连接管理器安装到不同的主机，并且不要安装到数据库服务器所在的主机上。

   此安装策略可防止连接管理器成为一个单点故障，且防止如果主机出现故障时数据库服务器和连接管理器同时故障。

   如果您要考虑应用程序服务器对主集群服务器的连结的优先级，则可以在应用程序服务器主机上安装连接管理器。
3. 在每个连接管理器主机上，将 INFORMIXDIR 环境变量设置为连接管理器的安装目录。
   运行以下命令 ：

```
setenv INFORMIXDIR path
```

4. 在每个连接管理器安装的 $INFORMIXDIR/etc 目录中创建一个配置文件。

   第一个连接管理器配置文件命名为 cm_1.cfg 且具有以下条目 ：

```
NAME connection_manger_1
LOG 1
LOGFILE $INFORMIXDIR/tmp/my_cm1_log.log
LOCAL_IP 192.0.2.0,192.0.2.1

CLUSTER cluster_1
{
   INFORMIXSERVER servers_1
   SLA oltp_1    DBSERVERS=primary
   SLA payroll_1 DBSERVERS=(PRI,HDR) \
                 POLICY=WORKLOAD
   SLA report_1  DBSERVERS=(SDS,HDR) \
                 POLICY=WORKLOAD
   FOC ORDER=ENABLED \
       PRIORITY=1
   CMALARMPROGRAM $INFORMIXDIR/etc/CMALARMPROGRAM.sh
}
```

   第二个连接管理器配置文件命名为 cm_2.cfg 且具有以下条目 ：

```
NAME connection_manger_2
LOG 1
LOGFILE $INFORMIXDIR/tmp/my_cm2_log.log
LOCAL_IP 192.0.2.2,192.0.2.3

CLUSTER cluster_1
{
   INFORMIXSERVER cluster_1
   SLA oltp_2    DBSERVERS=primary
   SLA payroll_2 DBSERVERS=(PRI,HDR)\
                 POLICY=WORKLOAD
   SLA report_2  DBSERVERS=(SDS,HDR) \
                 POLICY=WORKLOAD
   FOC ORDER=ENABLED \
       PRIORITY=2
```

```
    CMALARMPROGRAM $INFORMIXDIR/etc/CMALARMPROGRAM.sh
}
```

配置文件指定以下信息和行为 ：

- 启用日志记录，日志文件为 $INFORMIXDIR/tmp/my_cm1_log.log 和 $INFORMIXDIR/tmp/my_cm2_log.log。
- connection_manager_1 监视 192.0.2.0 和 192.0.2.1 的网络故障，connection_manager_2 监视 192.0.2.2 和 192.0.2.3 的网络故障。
- 每个连接管理器启动时，将搜索其 sqlhosts 文件以查找 cluster_1 条目，然后连接到那个组中的服务器。
- CONNECT TO @oltp_1 和 CONNECT TO @oltp_2 连接请求将定向到主服务器。
- CONNECT TO @payroll_1 和 CONNECT TO @payroll_2 连接请求将定向到具有最低工作负载的主服务器或 HDR 辅助服务器。
- CONNECT TO @report_1 和 CONNECT TO @report_2 连接请求将定向到具有最低工作负载的辅助服务器。
- connection_manager_1 和主服务器之间的连接优先于 connection_manager_2 和主服务器之间的连接。中断 connection_manager_1 和主服务器之间连接的故障转移会被阻止。
- 如果故障转移处理在尝试八次都失败后，将调用 $INFORMIXDIR/etc/CMALARMPROGRAM.sh。

某些参数和属性未包含在此配置文件中，因此连接管理器具有以下缺省行为：

- 未设置 EVENT_TIMEOUT 参数，因此连接管理器在故障转移处理开始之前等待主服务器事件60 秒。
- 未设置 SLA 参数的 MODE 属性，因此连接管理器将 server_1、server_2 和 server_3 的连接信息返回给客户端应用程序，而不是充当代理服务器。
- 未设置 SECONDARY_EVENT_TIMEOUT 参数，因此连接管理器与辅助服务器断开之前会等待辅助服务器事件 60 秒。
- 未设置 SLA 参数的 HOST、NETTYPE、SERVICE 和 SQLHOSTSOPT 属性，因此每个连接管理器使用本地和远程的 sqlhosts 文件中的连接信息。
- 未设置 SQLHOSTS 参数，因此每个连接管理器会先搜索其本地 sqlhosts 文件，然后再搜索远程数据库服务器 sqlhosts 文件以查找关于 server_1、server_2 和 server_3 的连接信息。
- 未设置 SLA 参数的 WORKERS 属性，因此分配四个四个工作线程给每个 SLA。

5. 将所有数据库服务器上 onconfig 文件中的 DRAUTO 配置参数都设置为 3

```
DRAUTO 3
```

此设置指定连接管理器控制故障转移仲裁。

6. 将 server_1 上 onconfig 文件中的 HA_FOC_ORDER 配置参数设置为 SDS,HDR

```
HA_FOC_ORDER SDS,HDR
```

在连接管理器启动并连接到 server_1 后，HA_FOC_ORDER 值将替换每个连接管理器配置文件中的 ORDER 属性的值。

如果 server_1 发生故障，连接管理器将尝试故障转移到 SD 辅助服务器。如果 SD 辅助服务器也不可用，那么连接管理器将尝试故障转移到 HDR 辅助服务器。

7. 可选： 在每个连接管理器主机上配置 cmalarmprogram 脚本。

事件警报可以发送到指定的电子邮件地址。

8. 将条目添加到 server_1、server_2 和 server_3 主机上的 sqlhosts 文件。

```
#dbservername  nettype    hostname   servicename   options
 server_1      onsoctcp   host_1     port_1
 server_2      onsoctcp   host_1     port_2
 server_3      onsoctcp   host_2     port_3
```

9. 在每个连接管理器创建一个 sqlhosts 文件。

```
#dbservername  nettype    hostname   servicename   options
 cluster_1     group       -          -            c=1,e=server_3
 server_1      onsoctcp    host_1     port_1       g=cluster_1
 server_2      onsoctcp    host_1     port_2       g=cluster_1
 server_3      onsoctcp    host_2     port_3       g=cluster_1
```

如果连接管理器在主服务器发生故障后重新启动，由于 cluster_1 组已定义，因此它将能够连接到集群中的其他数据库服务器。

10. 在每个客户端主机上创建一个 sqlhosts 文件。

```
#dbservername  nettype    hostname   servicename   options
 oltp          group       -          -            c=1,e=oltp_2
 oltp_1        onsoctcp    cm_host_1  cm_port_1     g=oltp
 oltp_2        onsoctcp    cm_host_2  cm_port_2     g=oltp

 report        group       -          -            c=1,e=report_2
 report_1      onsoctcp    cm_host_1  cm_port_3     g=report
 report_2      onsoctcp    cm_host_2  cm_port_4     g=report

 payroll       group       -          -            c=1,e=payroll_2
 payroll_1     onsoctcp    cm_host_1  cm_port_5     g=payroll
 payroll_2     onsoctcp    cm_host_2  cm_port_6     g=payroll
```

如果连接管理器发生故障，由于 oltp、 report 和 payroll 组已定义，因此客户端应用程序仍然可以连接到其他连接管理器。

- CONNECT TO @oltp 连接请求将通过其中一个连接管理器定向到主服务器。
- CONNECT TO @payroll 连接请求将通过其中一个连接管理器定向到具有最低工作负载的主服务器或 HDR 辅助服务器。
- CONNECT TO @payroll 连接请求将通过其中一个连接管理器定向到具有最低工作负载的辅助服务器。

11. 通过在每个连接管理器和客户端主机上运行 setenv 命令，将每个 INFORMIXSQLHOSTS 环境变量设置为 sqlhosts 文件的位置。

```
setenv INFORMIXSQLHOSTS path_and_file_name
```

12. 在每个连接管理器主机上运行 oncmsm 实用程序来启动连接管理器。

在 connection_manager_1 的主机上：

```
oncmsm -c cm_1.cfg
```

在 connection_manager_2 的主机上：

```
oncmsm -c cm_2.cfg
```

13. 检查每个连接管理器日志文件以确认连接管理器已正确启动。

相关链接

组信息 在第57页

《SinoDB 管理员参考》: oncmsm 实用程序

《SinoDB 管理员参考》: HA_FOC_ORDER 配置参数

《SinoDB SQL 指南: 参考》: INFORMIXSQLHOSTS 环境变量

网格或复制集连接管理配置的示例

您可以使用连接管理器来路由网格或复制集的复制服务器的客户端连接。

在此示例中，复制集由四个复制服务器组成：

- server_1
- server_2
- server_3
- server_4

复制集支持可运行在任何复制服务器上的报告服务。

您的系统有以下需求：

- 客户端请求定向至最少请求失败的复制服务器。
- 系统可以承受连接管理器故障。
- 系统可以承受每个主机上的网络接口卡（NIC）故障。

要配置连接管理：

1. 在每台主机上安装至少两个网络接口卡。
   这可以防止网络接口卡的故障导致连接管理器或数据库服务器故障。
2. 安装两个连接管理器。将每个连接管理器安装在不同的主机，并且不要安装在数据库服务器所在的主机上。
   此安装策略防止连接管理器成为一个单点故障，并防止主机故障时数据库服务器和连接管理器同时故障。
3. 在每个连接管理器主机上，将 INFORMIXDIR环境变量设置为连接管理器所安装的目录。
   运行以下命令：

   ```
   setenv INFORMIXDIR path
   ```

4. 在每个连接管理器安装的 $INFORMIXDIR/etc 目录中创建一个配置文件。

   第一个连接管理器配置文件命名为 cm_1.cfg，其包含以下条目：

   ```
   NAME connection_manger_1
   LOG 1
   LOGFILE $INFORMIXDIR/tmp/my_cm1_log.log

   REPLSET replicate_set_1
   {
    INFORMIXSERVER g_server_1,g_server_2,g_server_3,g_server_4
    SLA report_1 DBSERVERS=ANY \
                POLICY=FAILURE
   }
   ```

   第二个连接管理器配置文件命名为 cm_2.cfg，其包含以下条目：

   ```
   NAME connection_manger_2
   LOG 1
   LOGFILE $INFORMIXDIR/tmp/my_cm2_log_.log

   REPLSET replicate_set_1
   {
    INFORMIXSERVER g_server_1,g_server_2,g_server_3,g_server_4
    SLA report_2 DBSERVERS=ANY \
                POLICY=FAILURE
   }
   ```

   配置文件指定以下信息和行为：

   - 启用日志记录，且日志文件为 $INFORMIXDIR/tmp/my_cm1_log.log and $INFORMIXDIR/tmp/my_cm2_log.log.
   - 每一连接管理器启动时，将搜索其 sqlhosts 文件以查找 g_server_1、g_server_2、g_server_3 和 g_server_4 条目，然后连接到这些组中的 server_1、server_2、server_3 和 server_4 服务器。
   - CONNECT TO @report_1 和 CONNECT TO @report_2 连接请求将定向到最少请求失败的复制服务器。

某些参数和属性不包含在此配置文件中，因此连接管理器具有以下缺省行为 ：

- 未设置 SLA 参数的 MODE 属性，因此连接管理器会将 server_1、server_2、server_3 和 server_4 的连接信息返回客户端应用程序，而不是担任代理服务器。
- 未设置 SLA 参数的 HOST、NETTYPE、SERVICE 和 SQLHOSTSOPT 属性，因此每个连接管理器使用本地和远程 sqlhosts 文件中的连接信息。
- 未设置 SQLHOSTS 参数，因此每个连接管理器先搜索其本地 sqlhosts 文件，然后搜索远程数据库服务器 sqlhosts 文件中有关 server_1、server_2、server_3 和 server_4 的连接信息。
- 未设置 SLA 参数的 WORKERS 属性，因此分配四个工作线程给每个 SLA。

5. 添加条目到每个数据库服务器（connection_manger_1 和 connection_manger_2）的主机上的 sqlhosts 文件。

```
#dbservername  nettype   hostname   servicename  options
 g_server_1    group     -          -            i=1,e=server_1
 server_1      onsoctcp  host_1     port_1       g=g_server_1

 g_server_2    group     -          -            i=2,e=server_2
 server_2      onsoctcp  host_2     port_2       g=g_server_2

 g_server_3    group     -          -            i=3,e=server_3
 server_3      onsoctcp  host_3     port_3       g=g_server_3

 g_server_4    group     -          -            i=4,e=server_4
 server_4      onsoctcp  host_4     port_4       g=g_server_4
```

6. 在每个客户端主机上创建一个 sqlhosts 文件。

```
#dbservername  nettype    hostname   servicename   options
 report        group      -          -             c=1,e=report_2
 report_1      onsoctcp   cm_host_1  cm_port_3     g=report
 report_2      onsoctcp   cm_host_2  cm_port_4     g=report
```

由于已定义了 report 组，因此如果连接管理器故障，客户端应用程序仍然可以连接到其他连接管理器。

CONNECT TO @report 连接请求将通过其中一个连接管理器定向到最少请求失败的复制服务器。

7. 在每个连接管理器和客户端主机上运行 setenv 命令将其 INFORMIXSQLHOSTS 环境变量设置为 sqlhosts 文件的位置。

```
setenv INFORMIXSQLHOSTS path_and_file_name
```

8. 运行 cdr define qod 命令打开数据质量（QOD）监视。

```
cdr define qod -c server_1 --start
```

该命令连接到 server_1，将其定义为监视数据的主服务器，并打开数据质量监视。

server_1 为复制集中的服务器记录失败事务计数。失败事务计数决定连接管理器将连接请求发送给哪个复制服务器。

9. 在每个连接管理器主机上运行 oncmsm 实用程序来启动连接管理器。

在 connection_manager_1 的主机上：

```
oncmsm -c cm_1.cfg
```

在 connection_manager_2 的主机上：

```
oncmsm -c cm_2.cfg
```

10.检查每个连接管理器的日志文件以确认连接管理器已正确启动。

相关链接

组信息 在第57页

《SinoDB 管理员参考》: oncmsm 实用程序

《SinoDB 管理员参考》: HA_FOC_ORDER 配置参数

《SinoDB SQL 指南: 参考》: INFORMIXSQLHOSTS 环境变量

《SinoDB Enterprise Replication 指南》: cdr define qod

《SinoDB Enterprise Replication 指南》: cdr start qod

高可用性复制系统连接管理配置的示例

可使用连接管理器来路由复制集参与者的客户端连接，并控制参与 Enterprise Replication 的高可用性集群的故障转移。

在此示例中，网格是由四个节点组成。其中一个节点是高可用性集群中的主服务器，该高可用性集群是由一个主服务器、一个 SD 辅助服务器、一个 HDR 辅助服务器和一个 RS 辅助服务器组成：

- server_1a - ER 节点 1，主服务器
- server_1b - SD 辅助服务器
- server_1c - HDR 辅助服务器
- server_1d - RS 辅助服务器
- server_2 - ER 节点 2
- server_3 - ER 节点 3
- server_4 - ER 节点 4

该网格支持可在任何一个 ER 节点上运行的报告服务。

您的系统有以下需要：

- 将客户端请求定向到具有最低事务延迟的 ER 节点。
- 系统可以承受连接管理器的故障。
- 系统可以承受每台主机上一个网络接口卡（NIC）故障。
- 连接管理器控制集群的故障转移。
- 发生故障转移时，SD 辅助服务器优先于 HDR 辅助服务器。HDR 辅助服务器优先于 RS 辅助服务器。
- 如果主服务器发生故障，连接管理器在重新启动后仍连接到集群。

配置连接管理：

1. 在每台主机上安装至少两个网络接口卡。

   防止网络接口卡故障导致连接管理器或数据库服务器故障。
2. 安装两个连接管理器。每个连接管理器安装在不同的主机上，但不要安装在数据库服务器所在的主机上。

   此安装策略能够防止连接管理器成为单点故障，也能防止主机故障时数据库服务器和连接管理器同时故障。
3. 在每台连接管理器主机上，将 INFORMIXDIR 环境变量设置为连接管理器所安装的目录。
   运行以下命令：

   ```
   setenv INFORMIXDIR path
   ```
4. 在每个连接管理器安装的 $INFORMIXDIR/etc 目录中创建一个配置文件。

   第一个连接服务器配置文件名为 cm_1.cfg，其具有以下条目：

   ```
   NAME connection_manger_1
   LOG 1
   LOGFILE $INFORMIXDIR/tmp/my_cm1_log.log
   LOCAL_IP 192.0.0.2,192.0.2.1
   ```

```
REPLSET replicate_set_1
{
 INFORMIXSERVER g_server_1,g_server_2,g_server_3,g_server_4
 SLA report_1 DBSERVERS=ANY \
              POLICY=LATENCY
}

CLUSTER cluster_1
{
   INFORMIXSERVER g_server_1
   FOC ORDER=ENABLED \
       PRIORITY=1
   CMALARMPROGRAM $INFORMIXDIR/etc/CMALARMPROGRAM.sh
}
```

第二个连接服务器配置文件名为 cm_2.cfg，其具有以下条目：

```
NAME connection_manger_2
LOG 1
LOGFILE $INFORMIXDIR/tmp/my_cm2_log_.log
LOCAL_IP 192.0.2.2,192.0.2.3

REPLSET replicate_set_1
{
 INFORMIXSERVER g_server_1,g_server_2,g_server_3,g_server_4
 SLA report_2 DBSERVERS=ANY \
              POLICY=LATENCY
}
CLUSTER cluster_1
{
   INFORMIXSERVER g_server_1
   FOC ORDER=ENABLED \
       PRIORITY=2
   CMALARMPROGRAM $INFORMIXDIR/etc/CMALARMPROGRAM.sh
}
```

配置文件指定以下信息和行为：

- 启用日志记录，日志文件为 $INFORMIXDIR/tmp/my_cm1_log.log 和 $INFORMIXDIR/tmp/my_cm2_log.log。
- connection_manager_1 监视 192.0.2.0 和 192.0.2.1 的网络故障，和 connection_manager_2 监视 192.0.2.2 和 192.0.2.3 的网络故障。
- 每一连接管理器启动时，将搜索 sqlhosts 文件以查找 g_server_1、g_server_2、g_server_3 和 g_server_4 条目，然后连接到那些组中的 server_1a、server_1b、server_1c、server_1d、server_2、server_3 和 server_4 服务器。
- CONNECT TO @report_1 和 CONNECT TO @report_2 连接请求定向到具有最低事务延迟的复制服务器。
- connection_manager_1 和主服务器之间的连接优先于 connection_manager_2 和主服务器之间的连接。中断 connection_manager_1 和主服务器之间连接的故障转移会被阻止。
- 如果故障转移处理在八次尝试后失败，将调用 $INFORMIXDIR/etc/CMALARMPROGRAM.sh 。

某些参数和属性未包含在此配置文件中，因此连接管理器具有以下缺省行为：

- 未设置 EVENT_TIMEOUT 参数，因此连接管理器在故障转移处理开始之前将等待主服务器事件 60 秒。未设置 SECONDARY_EVENT_TIMEOUT 参数，因此连接管理器与辅助服务器断开连接之前将等待辅助服务器事件 60 秒。
- 未设置 SLA 参数的 HOST、NETTYPE、SERVICE 和 SQLHOSTSOPT 属性，因此每个连接管理器使用本地和远程 sqlhosts 文件中的连接信息。
- 未设置 SQLHOSTS 参数，因此每个连接管理器先搜索其本地 sqlhosts 文件，然后搜索远程数据库服务器 sqlhosts 文件中有关 server_1、server_2、server_3 和 server_4 的连接信息。

- The WORKERS attributes of the SLA parameters are not set, so four worker threads are allocated to each of the SLAs.
- 未设置 SLA 参数的 WORKERS 属性，因此分配四个工作线程给每个 SLA 。

5. 将 server_1a、 server_1b、server_1c 和 server_1d 上的 onconfig 文件的 DRAUTO 配置参数设置为 3。

```
DRAUTO 3
```

此设置指定连接管理器控制故障转移仲裁。

6. 将 server_1a 上的 onconfig 文件的 HA_FOC_ORDER 配置参数设置为 SDS,HDR,RSS。

```
HA_FOC_ORDER SDS,HDR,RSS
```

连接管理器启动并连接到 server_1a 后，HA_FOC_ORDER 值将替换每个连接管理器配置文件中的 ORDER 属性值。

如果 server_1a 发生故障，连接管理器将尝试故障转移到 SD 辅助服务器。如果 SD 辅助服务器也不可用，连接管理器将尝试故障转移到 HDR 辅助服务器。如果 HDR 辅助服务器也不可用，连接管理器将尝试故障转移到 RS 辅助服务器。

7. 添加条目到每个数据库服务器和连接管理器的主机上的 sqlhosts 文件。

```
#dbservername  nettype   hostname  servicename  options
 g_server_1    group     -         -            i=1,c=1,e=server_1d
 server_1a     onsoctcp  host_1    port_1       g=g_server_1
 server_1b     onsoctcp  host_1    port_2       g=g_server_1
 server_1c     onsoctcp  host_2    port_3       g=g_server_1
 server_1d     onsoctcp  host_3    port_4       g=g_server_1

 g_server_2    group     -         -            i=2,e=server_2
 server_2      onsoctcp  host_4    port_5       g=g_server_2

g_server_3    group     -         -            i=3,e=server_3
 server_3      onsoctcp  host_5    port_6       g=g_server_3

 g_server_4    group     -         -            i=4,e=server_4
 server_4      onsoctcp  host_6    port_7       g=g_server_4
```

8. 在每个客户端主机上创建一个 sqlhosts 文件。

```
#dbservername  nettype    hostname   servicename   options
 report        group      -          -             c=1,e=report_2
 report_1      onsoctcp   cm_host_1  cm_port_3     g=report
 report_2      onsoctcp   cm_host_2  cm_port_4     g=report
```

由于已定义了 report 组，在连接管理器发生故障时，客户端应用程序仍然可以连接到另一个连接管理器。

CONNECT TO @report 连接请求通过其中一个连接管理器定向到具有最低事务延迟的复制服务器。

9. 在每个连接管理器和客户端主机上运行 setenv 命令，将其 INFORMIXSQLHOSTS 环境变量设置为 sqlhosts 文件的位置。

```
setenv INFORMIXSQLHOSTS path_and_file_name
```

10. 运行 cdr define qod 命令打开数据质量（QOD）监视。

```
cdr define qod -c server_1a --start
```

此命令连接 server_1a，将 server_1a 定义为监视数据的主服务器，且打开数据质量监视。

server_1a 监视网格中复制服务器的事务延迟。

11. 在每台连接管理器主机上运行 oncmsm 实用程序，以启动连接管理器。

    在 connection_manager_1 的主机上：

    ```
    oncmsm -c cm_1.cfg
    ```

    在 connection_manager_2 的主机上：

    ```
    oncmsm -c cm_2.cfg
    ```

12. 检查每个连接管理器的日志文件以确认连接管理器已正确启动。

相关链接

组信息 在第57页

《SinoDB 管理员参考》: oncmsm 实用程序

《SinoDB 管理员参考》: HA_FOC_ORDER 配置参数

《SinoDB SQL 指南: 参考》: INFORMIXSQLHOSTS 环境变量

《SinoDB Enterprise Replication 指南》: cdr define qod

《SinoDB Enterprise Replication 指南》: cdr start qod

示例: 优先连接和网络监视的连接管理配置

您可以在应用程序服务器的主机上安装连接管理器，然后理出特定应用程序服务器和高可用性集群的主服务器之间的连接优先顺序。这种配置使优先级别最高的应用程序服务器即使部分网络发生故障仍可保持与集群主服务器的连接。

在下列情况中，您可以配置故障转移 ：

- 主服务器失效。
- 应用程序服务器与主服务器之间网络连接断开。

如果启用了网络监视和故障转移优先级，网络故障发生时，失去与主服务器的连接但保持与辅助服务器的连接的应用程序服务器可以通过共享主机连接管理器来启动到辅助服务器的故障转移。然而，如果故障转移导致具有高优先级别的应用程序服务器和主服务器失去连接，那么连接管理器会阻止故障转移。

通过在每个连接管理器配置文件中设置以下参数和属性，可以启用网络监视和故障转移优先级：

- 连接管理器的 LOCAL_IP 参数
- CLUSTER 连接单元的 SLA 参数
- CLUSTER 连接单元的 FOC 参数
- FOC 参数的 ORDER 属性
- FOC 参数的 PRIORITY 属性

1. 在每台主机上至少安装两个网络接口卡。

   这种方法可以防止网络接口卡故障导致客户端、 连接管理器或数据库服务器的连接失效。

2. 在每个应用程序服务器主机上安装和配置连接管理器。

   a) 将每个连接管理器配置文件中的 LOCAL_IP 参数设置为主机 NIC 卡的 IP 地址。
   例如：

   第一个连接管理器配置文件名为 cm_1.cfg ， 其具有以下条目 ：

   ```
   NAME connection_manager_1
   LOCAL_IP 192.0.2.0,192.0.2.1
   ```

   第二个连接管理器配置文件名为 cm_2.cfg， 其具有以下条目 ：

   ```
   NAME connection_manager_2
   ```

```
LOCAL_IP 192.0.2.2,192.0.2.3
```

b) 为每个连接管理器创建服务级别协议。
例如：

cm_1.cfg 有以下条目：

```
NAME connection_manager_1
LOCAL_IP 192.0.2.0,192.0.2.1

CLUSTER my_cluster

{
   INFORMIXSERVER my_servers
   SLA sla_1 DBSERVERS=PRI
   SLA sla_2 DBSERVERS=SDS,HDR,RSS
}
```

cm_2.cfg 有以下条目：

```
NAME connection_manager_1
LOCAL_IP 192.0.2.2,192.0.2.3

CLUSTER my_cluster

{
   INFORMIXSERVER my_servers
   SLA sla_3 DBSERVERS=PRI
   SLA sla_4 DBSERVERS=SDS,HDR,RSS
}
```

c) 为每个共享主机连接管理器设置 FOC 参数的 PRIORITY 属性，以指定每个应用程序服务器的优先级。
PRIORITY 值必须是一个正整数，并且在已配置管理特定集群的所有连接管理器中是唯一的。如果您在连接单元定义中指定一个 PRIORITY 值，那么就必须将 ORDER 属性设置为 ENABLED，并在主服务器的 HA_FOC_ORDER 配置参数中指定故障转移顺序。
例如：

主服务器有以下 onconfig 文件条目：

```
HA_FOC_ORDER SDS,HDR,RSS
```

cm_1.cfg 有以下条目：

```
NAME connection_manager_1
LOCAL_IP 192.0.2.0,192.0.2.1

CLUSTER my_cluster

{
   INFORMIXSERVER my_servers
   SLA sla_1 DBSERVERS=PRI
   SLA sla_2 DBSERVERS=SDS,HDR,RSS
   FOC ORDER=ENABLED PRIORITY=1
}
```

cm_2.cfg 有以下条目：

```
NAME connection_manager_2
LOCAL_IP 192.0.2.2,192.0.2.3

CLUSTER my_cluster
```

```
{
   INFORMIXSERVER my_servers
   SLA sla_3 DBSERVERS=PRI
   SLA sla_4 DBSERVERS=SDS,HDR,RSS
   FOC ORDER=ENABLED PRIORITY=2
}
```

3. 创建包含每个连接管理器和数据库服务器主机的网络连接信息的 sqlhosts。
4. 可选： 如果您为数据库服务器配置安全端口，请创建一个密码文件。
5. 在每个连接管理器主机上运行 oncmsm 实用程序以启动连接管理器。

   在 connection_manager_1 主机上：

   ```
   oncmsm -c cm_1.cfg
   ```

   在 connection_manager_2 主机上：

   ```
   oncmsm -c cm_2.cfg
   ```

6. 检查每个连接管理器的日志文件，以确认连接管理器已正确启动。

如果现在网络发生故障，连接管理器将启动故障转移，并且故障转移不会导致较高优先级的应用程序服务器丢失与集群主服务器的连接。

相关链接

组信息 在第57页

FOC 连接管理器配置参数 在第350页

LOCAL_IP 连接管理器配置参数 在第355页

《*SinoDB* 管理员参考》: *oncmsm* 实用程序

《*SinoDB* 管理员参考》: *HA_FOC_ORDER* 配置参数

集群故障转移、重定向和复原

为了保持可用性，您必须规划主服务器的故障转移，重定向来自不可用服务器的客户端连接，并在发生故障后将集群复原到其原始配置。

高可用集群的故障转移配置

高可用集群中的故障就是其中一台服务器不再可用。如果主服务器故障，那么您必须将一台辅助服务器升级成主服务器角色。可以将连接管理器配置成执行自动故障转移。

您必须设置 DRAUTO 配置参数，以指定故障转移的执行方式，并使用以下故障转移选项之一：

- 连接管理器
- ISV 集群管理软件
- 手动切换

相关链接

通过连接管理器进行连接管理 在第340页

《*SinoDB* 管理员参考》: *DRAUTO* 配置参数

使用 ISV 集群管理软件进行故障转移

您可以使用独立软件供应商（ISV）的集群管理软件，而不是连接管理器来管理高可用性集群环境中的故障转移处理。

如果高可用性集群的主服务器遇到问题需要辅助服务器充当主服务器的角色，那么在执行实际的故障切换之前，在故障的主服务器上禁止磁盘 I/O ，而在新的主服务器上允许磁盘 I/O 是非常重要的。此外，必须防止对故障的主服务器的网络访问。尤其是对 SD 辅助服务器，如果没有正确完成这些步骤，有可能损坏磁盘。

从高可用性集群环境中的服务器启用磁盘 I/O 操作的机制称为 *I/O* 防护。I/O 防护通过使用回调脚本进行配置。当主服务器发生故障时，在辅助服务器充当主服务器的角色之前，故障转移进程会在辅助服务器上执行回调脚本。该脚本调用特定于 I/O 的任何命令，以启用或禁用磁盘访问。该脚本启用要成为主服务器的服务器上的共享磁盘的写入访问，并禁用发生故障的服务器上的共享磁盘的写入访问。

使用 FAILOVER_CALLBACK 配置参数来指定数据库服务器从辅助服务器转换为主服务器，或从辅助服务器转换为标准服务器时，要执行的脚本名称。在 $INFORMIXDIR/etc 目录中提供了一个名为 ifx_failover_callback.sh 的模板脚本（UNIX™）或 ifx_failover_callback.bat（Windows™）。配置后，在辅助服务器切换为主服务器或标准服务器之前，将执行由 FAILOVER_CALLBACK 指定的脚本。

依据您的高可用性集群类型，您可以通过执行以下操作之一来测试故障转移脚本 ：

- 将 SD 辅助服务器转换成主服务器。
- 如果 DRAUTO 配置参数设置为 0，那么关闭主服务器并将 HDR 辅助服务器转换为标准模式。
- 如果 DRAUTO 配置参数设置为 1，那么关闭 HDR 对中的主服务器。
- 关闭远程独立集群中的主服务器，并将 RS 辅助服务器转换为标准模式。

在 online.log 中包含 Invoking Failover Callback 消息，在故障转移脚本运行后，其中列出该故障转移脚本的路径和文件名。

参阅 《*SinoDB*® 管理员参考》 中有关 FAILOVER_CALLBACK 配置参数的信息。

如果 FAILOVER_CALLBACK 指定的脚本失败 （也就是说，如果它返回一个非零退出码），那么辅助服务器到主服务器（或标准服务器）的故障转移也将失败。在这种情况下，DBA 必须手动执行故障转移。

共享文件系统的 I/O 防护

您可以配置 I/O 防护以保护高可用性集群环境中的共享资源。

软件或硬件故障可能导致未完成的操作写入共享存储设备。I/O 防护可用来隔离服务器并防止其访问共享存储。如果在高可用集群的服务器上执行维护或测试，那么必须使用 I/O 防护。如果在服务器上或在应用程序内检测到问题，集群管理器可检测问题并防止该服务器连接到共享数据。

您你可以配置主服务器故障时运行的脚本。通过 FAILOVER_CALLBACK 配置参数设置故障转移时调用的防护命令脚本。

尽管无需 I/O 防护就可使用 SinoDB® 数据库软件，但仍必须配置 I/O 防护来保护共享磁盘系统，以免遭受意外损失。

I/O 防护的类型

有几种类型的 I/O 防护可用，包括：

- Power 防护 - 检测到问题时关闭节点电源。
- 光纤通道开关防护（需要 SCSI-3 persistent group reservation）- 通过移除问题节点的保留来阻塞光纤通道设备上的端口。

实施 I/O 防护的最常用的方法可能是使用光纤管道防护。光纤通道开关支持业界标准的 SCSI-3 persistent group reservation（PR）技术。PR 技术允许一组系统有临时注册到磁盘，并且与包含数据的磁盘协调写互斥保留。

在大多数情况下，必须安装集群管理软件。集群管理软件提供了向光纤管道开关发送命令的驱动程序和实用程序。例如，Linux™ Cluster Suite 提供了一个名为 fence_scsi 的脚本。Sun Cluster 提供了一个名为scdidadm 的命令。

也可根据所使用的集群管理软件和不同的硬件能力来使用其他防护方法。

IBM® General Parallel File System (GPFS™) 是由 IBM® 开发的高性能共享磁盘集群文件系统。该文件系统可运行于 AIX®、Linux™ 和 Windows™ 平台上。该文件系统可以与 IBM® HACMP™ Cluster Management Software 一起使用。GPFS™ 使用 SCSI PR 防护机制来支持 I/O 防护。防护方面的问题必须由IBM® GPFS™ 支持团队解决。

实施 I/O 防护

可在几种平台上配置 I/O 防护，包括：

- Linux™
- Solaris
- AIX®
- Windows™

有关配置 I/O 防护的特定信息，请参阅设备生产商提供的文档。

相关链接

《*SinoDB* 管理员参考》: *FAILOVER_CALLBACK* 配置参数

集群故障

高可用性集群故障是指集群中的数据库服务器间的连接丢失，有很多种情况可造成。

以下任一情况都可能导致集群故障：

- 设备故障或损坏
- 网络故障
- 数据库服务器之一过渡的处理延迟

数据库服务器会将以下任一状况认定为集群故障：

- 超出 DRTIMEOUT 配置参数值而未与集群中的其它服务器进行通信确认。
- 集群中的数据库服务器未响应通过网络进行的定期消息传递尝试。集群中服务器运行 ping 侦测彼此存在，即使主服务器并没有向辅助数据库服务器发送记录。

  集群中服务器在 DRTIMEOUT 配置参数指定的时间间隔，运行 ping 对集群中其它服务器进行侦测。

当数据库服务器检测到集群故障后，它会在消息日志中写入一个消息（例如 DR: receive error）并关闭数据复制。如果集群发生故障，那么两台数据库服务器之间的连接会断开，辅助数据库服务器会保持在只读模式。

您可以通过将 DRAUTO 配置参数设置为 1 或 2 来配置 HDR 复制对的自动切换。

您可以通过配置连接管理器来配置高可用性集群的自动故障转移。与自动切换相比，连接管理器有许多优势，并且可以自动故障转移至 SD 和 RS 辅助服务器。

相关链接

将主服务器数据复制到辅助服务器 在第311页

自动切换

如果 HDR 可用性系统的主服务器发生故障，HDR 辅助服务器会自动转换为标准或主数据库服务器。

如果发生自动切换，则 HDR 辅助服务器将回滚打开的事务，然后作为主数据库服务器或标准数据库服务器切换至联机模式。自动切换并不会将客户端应用程序重新定向至新的主数据库服务器或标准数据库服务器。

自动切换要优于手动切换，但使用连接管理器来控制高可用性集群的故障转移更有优势。

自动切换需要非常稳定的网络，且只能适用于主服务器和 HDR 辅助服务器。连接管理器可配置为能够承受网络的不稳定，并可以执行故障转移至 HDR、RS 和 SD 辅助服务器。 如果发生网络故障，连接管理器还可以优先考虑应用程序服务器和主服务器之间的连接。

自动切换不会将客户端连接从原主服务器重新定向至新的标准服务器或主服务器。连接管理器会将客户端重新定向到成为主服务器的辅助服务器。

如果使用自动切换，必须确保备份逻辑日志文件，以及 HDR 服务器有足够的逻辑日志磁盘空间以允许在不备份逻辑日志文件的情况下继续进行处理。

要配置自动切换或连接管理器的故障转移，您必须在高可用性集群的所有数据库服务器上设置 DRAUTO 配置参数。

相关链接

数据复制客户端的重定向与连接 在第409页
没有可靠网络时的自动切换 在第408页
主服务器发生故障时重新启动 在第417页

没有可靠网络时的自动切换

尽管自动切换可能是最佳的解决方法，但它并不适用于所有情况。

试想如果主数据库服务器不是真正发生故障，但辅助数据库服务器注册该主服务器已故障时会发生什么情况。例如，如果辅助服务器由于网速慢或网络不稳定而向主服务器发送信号（ping）时未收到响应，那么辅助服务器将假设主数据库服务器故障，并自动切换至标准类型。如果主数据库服务器向辅助数据库服务器发送信号时也未收到响应，那么它将假设辅助数据库服务器故障，并关闭数据复制但保持联机模式。此时，主数据库服务器和辅助数据库服务器（切换至标准类型）都处于联机模式。

如果客户端可以独立地在两台数据库服务器上更新数据，那么此对中的数据库服务器将处于各自具有另一数据库服务器所需的逻辑日志记录的状态。在此情况下，您必须重新启动，并使用整个数据库服务器的 0 级数据库空间备份执行初始数据复制，如 首次启动 *HDR* 在第298页 中所述。因此，如果网络不是十分稳定，那么您可能不希望使用自动切换。HDR 无法在之前的辅助服务器上恢复而没有丢失事务的风险。

相关链接

自动切换 在第407页

手动切换

手动切换即辅助数据库服务器管理员将辅助数据库服务器的类型更改为标准类型。

辅助数据库服务器回滚所有打开的事务，然后作为标准数据库服务器进入联机模式，以便接受客户端应用程序的更新。

相关链接

更改数据库服务器类型 在第336页

将脱机服务器连接到新的主服务器

故障转移后，您必须将任何脱机的辅助服务器重新连接到新的主服务器。

如果高可用性集群中的主服务器故障，系统将通知所有联机辅助服务器。辅助服务器连接到新的主服务器并继续运作。然而，在故障转移时未联机的 RS 辅助服务器和 HDR 辅助服务器不会收到故障转移通知，而且这些服务器恢复联机时将尝试连接原始（故障的）主服务器。在这种情况下，必须在这些辅助服务器上手动重置主服务器。

在故障转移发生时为脱机的辅助服务器上运行以下命令。

- 对于 HDR 辅助服务器：

```
oninit -PHY
onmode -d secondary new_primary
```

  new_primary 为当前主服务器的名称。
- 对于 RS 辅助服务器：

```
oninit -PHY
onmode -d RSS new_primary
```

new_primary 为当前主服务器的名称。

数据复制客户端的重定向与连接

当高可用性集群中的任何服务器变为不可用时，任何到它的客户端连接都应重定向至可用服务器。处理连接重定向的最好方式就是使用连接管理器，将客户端连接自动重新连接至您在服务级别协议中指定的服务器，。或者，可以使用环境变量或连接信息来控制连接重定向。

如果不使用连接管理器，那么可以通过将应用程序配置为连接到服务器所属的服务器组，自动将客户端重定向到集群中的其它数据库服务器。当创建与服务器组的连接时，缺省状态下，将建立与组中当前主服务器的连接。如果由于某个服务器故障导致复制停止，则将连接到处于联机的服务器（处于标准模式或主服务器模式而无需辅助服务器）。您还可以从应用程序里自动执行此操作。SinoDB® 客户端软件开发工具包 (Client SDK) 里的某些客户端连接驱动程序有特定的自动重定向机制。更多信息，请参阅 SinoDB® 客户端软件开发工具包 (Client SDK) 文件。

您在设计客户端应用程序时，就必须对重定向策略做出一些决策。尤其，您必须决定是否在应用程序里处理重定向，以及使用哪种重定向机制。三种重定向机制如下：

- 用 DBPATH 环境变量进行自动重定向
- 管理员用连接信息控制重定向
- 用户使用 INFORMIXSERVER 环境变量控制重定向

您所用的机制决定您可在应用程序中使用的 CONNECT 语法。

相关链接

通过连接管理器进行连接管理 在第340页
自动切换 在第407页

使用 DBPATH 环境变量自动重定向客户端

可以在应用程序里使用 DBPATH 环境变量重定向客户端。

管理员必须执行的操作

管理员不执行任何操作来重定向客户端，但是可能需要关注数据库服务器的类型。

用户必须执行的操作

您的应用程序包含测试连接是否失败的代码，以及发出重新连接语句的代码（如需要），那么重定向得以自动处理。用户没有任何职责。

如果您的应用程序没有包含这样的代码，那么正在运行客户端的用户必须退出并重新启动所有应用程序。

DBPATH 重定向方法的工作原理

当应用程序没有在 CONNECT 语句中明确指定数据库服务器，以及 INFORMIXSERVER 环境变量指定的数据库服务器不可用时，客户端将使用 DBPATH 环境变量来定位数据库（和数据库服务器）。

复制对中的一台数据库服务器为不可用时，如果使用该数据库服务器的应用程序的 DBPATH 环境变量设置为该对中的另一台数据库服务器，则这些应用程序无需重置 INFORMIXSERVER 环境变量。它们的 INFORMIXSERVER 环境变量必须始终包含其经常使用的数据库服务器的名称，且其 DBPATH 环境变量必须始终包含该对中另一台数据库服务器的名称。

例如，如果应用程序通常使用名为 cliff_ol 的数据库服务器，且在复制对中与 cliff_ol 配对的数据库服务器名为 beach_ol，那么那些应用程序的环境变量将如下所示：

```
INFORMIXSERVER  cliff_ol
DBPATH          //beach_ol
```

因为当应用程序发出 CONNECT 语句时 DBPATH 环境变量是只读变量（如果需要），所以应用程序必须重新启动才能进行重定向。

应用程序可包含测试连接是否失败的代码，如果是，则尝试重新连接。如果应用程序有该代码，就无需重新启动。

在该重定向方法中，您可以使用具有此重定向方法的 CONNECT TO *database* 语句。若要使该方法起作用，您不可以使用以下任一语句：

- CONNECT TO DEFAULT
- CONNECT TO *database@dbserver*
- CONNECT TO *@dbserver*

此限制的原因是当 CONNECT 语句指定数据库服务器时应用程序就不会使用 DBPATH。有关 DBPATH 的更多信息，请参阅 《*SinoDB® SQL* 指南*:* 参考》。

使用连接信息重定向客户端

您可以使用 sqlhosts 信息将客户端连接重定向到新的主服务器。

连接信息重定向方法依赖于这样的事实：当应用程序连接数据库服务器时，它使用连接信息来找到该数据库服务器。

如果复制对中的一台数据库服务器不可用，管理员可以更改连接信息中不可用数据库服务器的定义。正如 更改客户端连接信息 在第410页 中所述的，不可用数据库服务器的字段（dbservername 字段除外）更改为指向复制对中另一台数据库服务器。

由于 CONNECT 语句发出时会读取连接信息，所以可能需要重新启动应用程序才能执行重定向。应用程序可包含测试连接是否失败的代码，如果需要，也可包含发出重连接语句的代码。如果连接失败，将自动进行重定向，并且您无需重新启动应用程序就可进行重定向。

应用程序可使用以下连接语句来支持此重定向方法：

- CONNECT TO *database@dbserver*
- CONNECT TO *@dbserver*

假设 INFORMIXSERVER 环境变量始终设置为同一数据库服务器名，而 DBPATH 环境变量未设置，那么应用程序也可以使用以下连接语句：

- CONNECT TO DEFAULT
- CONNECT TO *database*

UNIX™ 上，INFORMIXSQLHOSTS 环境变量指定 $INFORMIXDIR/etc/sqlhosts 中连接信息的全路径名和文件名。有关 INFORMIXSQLHOSTS 的更多信息，请参阅 《*SinoDB® SQL* 指南*:* 参考》。

Windows™ 上，连接信息位于 Windows™ 注册表的键中。

相关链接

sqlhosts 文件和 *SQLHOSTS* 注册表键 在第46页
可信主机信息 在第43页
可信用户信息 在第44页

更改客户端连接信息

若要使用连接信息重定向客户端，您必须更改客户端连接信息，以及更改其他连接文件（如有必要）。

更改有关客户端计算机的连接信息：

1. 在 sqlhosts 文件或注册表中，注释掉故障数据库服务器的条目。
2. 添加条目在 servername 字段中指定发生故障的数据库服务器的 dbservername，并在 nettype、hostname 和 servicename 字段中指定客户端要重定向的数据库服务器的信息。
3. 在 sqlhosts 文件或注册表中使用以下选项，以便故障发生时将应用程序重定向到其他数据库服务器：
   a) 连接重定向选项

b) 组末尾选项
c) 组选项

4. 如果需要，在 UNIX™ 的 /etc/hosts 文件中，或 Windows™ 的 hosts 文件中，为用于客户端重定向的数据库服务器所在计算机的 hostname 添加一个条目。
5. 如果需要，在 UNIX™ 的 /etc/services 文件中，或 Windows™ 的 services 文件中，为用于客户端重定向的数据库服务器的 servicename 添加一个条目。

下图显示可以如何修改连接值以重定向客户端。

无需在正运行数据库服务器的任一计算机上更改连接信息中的条目。

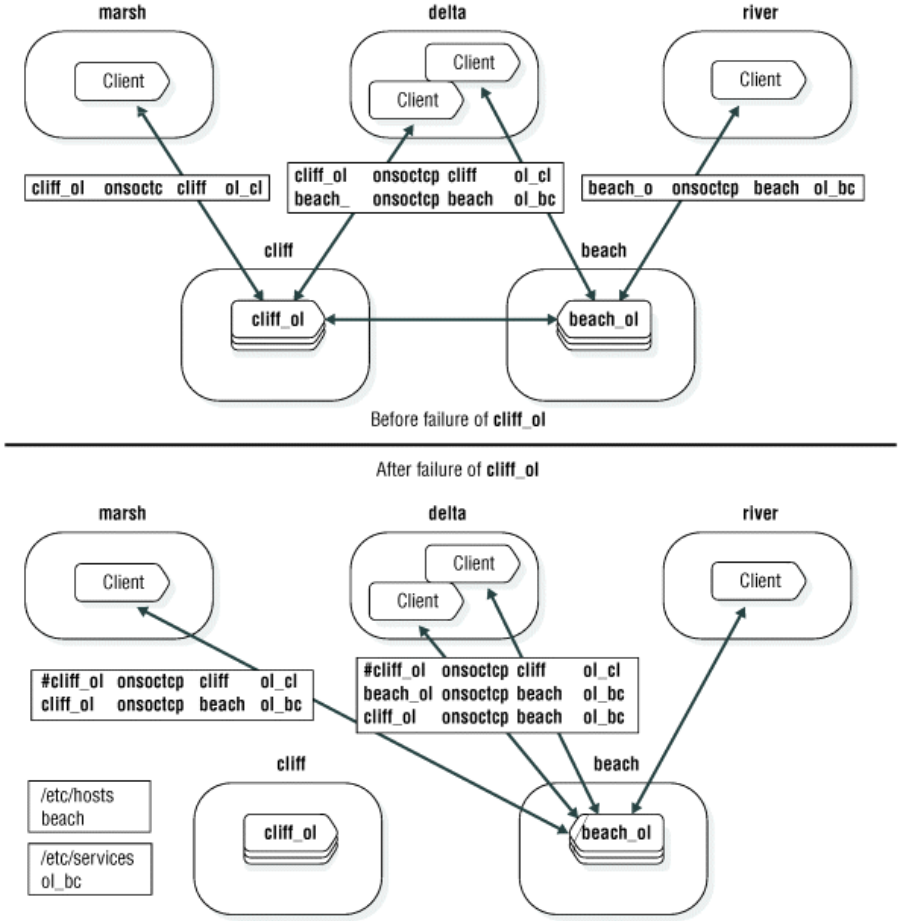

图 79: cliff_ol 数据库服务器故障前后的连接值

相关链接

sqlhosts 连接信息 在第49页
为高可用性集群配置安全连接 在第298页
客户端/服务器通信 在第33页

连接至数据库服务器

在管理员更改了连接信息以及其他连接文件（如果需要）后，当客户端发出下一个 CONNECT 语句时，客户端将连接至管理员重定向到的数据库服务器。

如果您的应用程序包含测试连接是否失败以及发出重新连接语句（如需要）的代码，则会自动处理重定向。用户无责。如果您的应用程序没有这样的代码，那么正在运行客户端的用户必须退出并重启所有应用程序。

使用服务器组自动重定向

可以使用 sqlhosts 文件中的组选项来指定应用程序连接到的服务器组而不是单个数据库服务器。要使连接重定向自动化，请将主服务器和辅助服务器的数据库服务器定义添加到服务器组定义。缺省情况下，当向

HDR 服务器组发出连接请求时，会将该连接路由到主服务器。如果主服务器不可用，那么该连接请求将路由到故障转移处理后提升为主服务器的辅助服务器。

例如，以下 sqlhosts 条目表示具有主服务器定义 hdr_prim 和辅助服务器定义 hdr_sec 的 HDR 服务器组 g_hdr 。

```
#dbservername   nettype    hostname      servicename   options
g_hdr           group      -             -             i=1
hdr_prim        ontlitcp   machine1pri   port1         g=g_hdr
hdr_sec         ontlitcp   machine1sec   port1         g=g_hdr
```

应用程序可使用下列连接语句，以支持重定向方法：

- CONNECT TO *database@dbserver_group*
- CONNECT TO *@dbserver_group*

如果您的应用程序包含测试连接是否失败以及发出重新连接语句（如需要）的代码，则会自动处理重定向。用户无责。如果您的应用程序没有这样的代码，那么正在运行客户端的用户必须退出并重启所有应用程序。

使用 INFORMIXSERVER 环境变量重定向客户端

当应用程序未在 CONNECT 语句中显式指定数据库服务器时，使用 INFORMIXSERVER 环境变量重定向方法，以便将客户端连接到 INFORMIXSERVER 环境变量指定的数据库服务器。

如果集群中某个数据库服务器无法使用，那么使用该数据库服务器的应用程序可将 INFORMIXSERVER 环境变量重置为集群中的另一数据库服务器，以访问相同数据。

应用程序仅在启动时读取 INFORMIXSERVER 环境变量的值。因此，应用程序必须重新启动才能识别环境变量中的更改。

可以使用下列连接语句支持该重定向方法：

- CONNECT TO DEFAULT
- CONNECT TO *database*

此方法不可使用 CONNECT TO *database@dbserver* 或 CONNECT TO *@dbserver* 语句。当数据库服务器已明确指明，CONNECT 语句就不会使用 INFORMIXSERVER 环境变量来查找找数据库服务器。

管理员不会执行任何操作来重定向客户端，但他们可能需要变更数据库服务器的类型。

正在运行客户端应用程序的用户在决定用 INFORMIXSERVER 环境变量重定向客户端时，必须执行以下三个步骤。

使用 INFORMIXSERVER 环境变量重定向客户端：

1. 退出应用程序。
2. 将 INFORMIXSERVER 环境变量更改为复制对中的另一数据库服务器名。
3. 重启应用程序。

使用应用程序代码重定向客户端

如果使用 DBPATH 环境变量或连接信息重定向连接，那么可以在客户端中包含一个例程，用于处理客户端遇到集群故障时的错误。该程序可以调用另一个函数，该函数包含一个重复尝试连接集群中另一数据库服务器的循环。该例程不需要用户退出应用程序并重启应用程序就可重定向客户端。

以下示例显示的是在客户端应用程序中使用 DBPATH 重定向机制循环尝试重新连接的一个函数。在建立连接之后，它还会测试数据库服务器类型，以确保其不是辅助服务器。如果数据库服务器仍是辅助类型，它会调用另一函数警告用户（或数据库服务器管理员）该数据库服务器无法接受更新。

```
/* The routine assumes that the INFORMIXSERVER environment
 * variable is set to the database server that the client
 * normally uses and that the DBPATH environment variable
 * is set to the other database server in the pair.
 */
```

```
#define SLEEPTIME 15
#define MAXTRIES 10

main()
{
   int connected = 0;
   int tries;
   for (tries = 0;tries < MAXTRIES && connected == 0;tries++)
   {
      EXEC SQL CONNECT TO "superstores";
      if (strcmp(SQLSTATE,"00000"))
      {
         if (sqlca.sqlwarn.sqlwarn6 != 'W')
         {
            notify_admin();
            if (tries < MAXTRIES - 1)
               sleep(SLEEPTIME);
            }
         else connected =1;
         }
      }
   return ((tries == MAXTRIES)? -1:0);
   }
```

本示例采用 DBPATH 重定向机制，并使用支持 DBPATH 重定向方法的 CONNECT 语句格式。如果使用连接信息重定向，那么可能会有不同的连接语句，如下所示：

```
EXEC SQL CONNECT TO "superstores@cliff_ol";
```

此示例中， superstores@cliff_ol 为客户端计算机所识别的数据库服务器上的数据库。要可以重定向，管理员必须更改连接信息，以使该名称指向不同的数据库服务器。您可能需要调整客户端在尝试连接前等待的时间或函数尝试连接的次数。为数据库服务器上的管理操作提供足够时间（去更改连接信息或是将辅助数据库服务器的类型更改为标准类型）。

比较重定向方法

不同的重定向方法有不同的需求。

下表总结了这些重定向机制之间的不同：

表 65: DBPATH 连接的重定向方法

DBPATH	自动重定向	用户重定向
何时重定向客户端？	当客户端下次尝试连接指定的数据库	当客户端下次尝试连接指定的数据库
客户端重定向是否需要重启？	否	是
重定向的范围是哪些？	重定向个别客户端	重定向个别客户端
是否需要更改环境变量？	否	否

表 66: 连接信息的重定向方法

连接信息	自动重定向	用户重定向
何时重定向客户端？	在管理员更改连接信息后，当客户端下次尝试与数据库建立连接时	在管理员更改连接信息后，当客户端下次尝试与数据库建立连接时
客户端重定向是否需要重启？	否	是

连接信息	自动重定向	用户重定向
重定向的范围是哪些?	重定向所有使用给定数据库服务器的客户端	重定向个别客户端
是否需要更改环境变量?	否	否

表 67: INFORMIXDIR 连接的重定向方法

INFORMIXDIR	自动重定向
何时重定向客户端?	当客户端重启并读取 INFORMIXSERVER 环境变量的新值时
客户端重定向是否需要重启?	是
重定向的范围是哪些?	重定向个别客户端
是否需要更改环境变量?	是

表 68: 连接管理器连接的重定向方法

连接管理器	自动重定向
何时重定向客户端?	当获得已配置服务级别协议时。
客户端重定向是否需要重启?	否
重定向的范围是哪些?	重定向个别客户端
是否需要更改环境变量?	否

故障后恢复 HDR 和 RS 集群

将 HDR 或 RS 集群恢复到原始配置时，可能需要恢复关键介质，以及重启和配置集群中的服务器。

磁盘故障的结果取决于磁盘故障发生在主服务器还是辅助服务器、磁盘上的块是否包含关键介质（根数据库空间、逻辑日志文件或物理日志）以及块是否已镜像。

如果块已镜像，您可以执行恢复，如同您对使用镜像的标准数据库服务器所做的一样。

如果块未镜像，那么主服务器的恢复过程取决于故障磁盘是否包含关键介质：

- 如果磁盘包含关键介质，则主服务器将失败。您必须使用主数据库空间备份（或辅助数据库空间备份，如果辅助数据库服务器切换为标准模式并重定向了活动）进行完全复原。
- 如果磁盘不包含关键介质，您可以使用热复原来分别复原受影响的数据库空间。热复原包含两个部分：首先从备份来复原故障的数据库空间，然后对数据库空间备份以来写入的所有逻辑日志记录进行逻辑恢复。在执行热复原前，必须备份所有的逻辑日志文件。

在块未镜像的情况下，如果磁盘包含关键介质，辅助服务器将失败；但如果磁盘不包含关键介质，则辅助服务器将保持联机。在这两种情况下，您必须使用主服务器上的数据库空间备份进行完全复原。在第二种情况下，不能从辅助数据库空间备份来复原所选的数据库空间，因为它们目前可能偏离主服务器上相应的数据库空间。您必须执行完全复原。

相关链接

关键数据损坏后恢复集群 在第415页

关键数据损坏后恢复集群

如果高可用性集群中的某个数据库服务器发生造成根数据库空间、包含逻辑日志文件的数据库空间或包含物理日志的数据库空间损坏的故障，那么您必须将该故障的数据库服务器视为在首次启动时磁盘上没有数据一样。将正在运行的带有完整磁盘的数据库服务器作为带有数据的数据库服务器来使用。

主服务器故障

对于以下步骤，假设配置中包含名为 srv_A 的主服务器和名为 sev_B 的 HDR 辅助服务器。重新启动 RS 集群的步骤与此相类似。

要在关键介质发生故障后重新启动 HDR：

1. srv_B 上的 DRAUTO 配置参数将影响您下一步的操作
   - 如果此参数设置为 0，则必须运行 onmode -d make primary 命令将服务器转换为主服务器。
   - 如果此参数设置为 1，则必须运行 onmode -d make primary 命令将服务器转换为主服务器。
   - 如果此参数设置为 2，那么一旦原主服务器发生故障导致连接结束，辅助数据库服务器将立即变为主数据库服务器。
2. 从上次数据库空间备份复原 srv_A （主数据库服务器）。
3. 使用 onmode -d 命令，将 srv_A 设置为 HDR 辅助服务器并启动 HDR。

   onmode -d 命令在 srv_B 上从逻辑日志文件启动逻辑恢复。如果因为在 srv_B 上备份并释放了逻辑日志文件而无法完成逻辑恢复，那么在您执行下一步前 HDR 不会启动。
4. 应用来自 srv_B（新的主数据库服务器）已备份到磁带的逻辑日志文件。现在 HDR 对可运行了，但是 srv_A 和 srv_B 的角色将交换。要将 srv_A 和 srv_B 的交换回原来的角色，请依照指令进行：在辅助服务器成为主服务器后恢复 *HDR* 集群 在第416页。

表 69: 主服务器发生关键介质故障后重启 HDR 的步骤。

步骤	在主数据库服务器（svr_A）上	在辅助数据库服务器（svr_B）上
1.		onmode 命令 onmode -d make primary srv_A
2.	ontape 命令 ontape -p ON-Bar 命令 onbar -r -p	
3.	onmode 命令 onmode -d secondary srv_B	
4.	ontape 命令 ontape -l ON-Bar 命令 onbar -r -l	

辅助服务器故障

如果辅助服务器发生关键介质故障，请按照以下首次启动集群的步骤来恢复集群。

主服务器和辅助服务器故障

正在运行复制对中数据库服务器的计算机不幸同时发生造成根数据库空间、包含逻辑日志文件或物理日志的数据库空间损坏的故障时，您必须重新启动集群。

要在两个数据库服务器发生关键介质故障后重新启动高可用性集群：

1. 从存储空间和逻辑日志备份来复原主数据库服务器。
2. 复原主数据库服务器后，请将另一发生故障的数据库服务器视为在磁盘上没有任何数据，并且假定是首次启动高可用性集群。

相关链接

故障后恢复 *HDR* 和 *RS* 集群 在第414页
首次启动 *HDR* 在第298页

网络故障后重新启动 HDR 和 RS 集群

发生网络故障后，可能需要重新启动 HDR 和 RS 集群。发生网络故障后，主数据库服务器处于联机模式，辅助服务器处于只读模式。关闭两台数据库服务器上的复制 (state = off)。

可能不需要重新启动集群，因为主数据库服务器每10秒会尝试重新连接一次，并且每2分钟显示一次有关无法连接的消息。

如果集群未自动重新启动，那么重新建立连接时，可以在辅助服务器上运行以下命令来重新启动该集群：

- onmode -d secondary *primary_name* 以使辅助服务器成为主服务器。

辅助服务器发生故障时重新启动 HDR 或 RS 集群

如果在辅助服务器发生故障后必须重新启动 HDR 或 RS 集群，那么除了启动辅助服务器外，可能还需要在辅助服务器上执行逻辑日志复原。

这些步骤假定自从辅助数据库服务器发生故障以来，您已根据需要备份主数据库服务器上的逻辑日志文件。

表 70: 辅助服务器发生故障后的重新启动 步骤

步骤	在主服务器上	在辅助服务器上
1.	主数据库服务器必须处于联机模式	oninit 如果在消息日志中接收到以下消息，请继续步骤2： DR: Start Failure recovery from tape
2.		ontape 命令 ontape -l ON-Bar 命令 onbar -r -l

在辅助服务器成为主服务器后恢复 HDR 集群

如果 HDR 集群中的辅助服务器在原主服务器发生故障后成为主服务器，那么可以使用脚本来重新建主服务器，然后将当前的主服务器转换成辅助服务器。

假设名为 srv_pri 的主服务器遇到错误，以至于故障转移为名为 srv_hdr_sec 的 HDR 辅助服务器，此时，主服务器是 srv_hdr_sec，并且集群中的其他辅助服务器现在都指向 srv_hdr_sec。

要将集群复原至 srv_pri 发生故障转移之前的状态，请遵循以下步骤：

1. 通过运行适当命令，将 srv_pri 初始化为 HDR 辅助服务器：

UNIX™ 系统：

```
$INFORMIXDIR/bin/hdrmksec.sh srv_hdr_sec
```

Windows™ 系统：

```
hdrmksec.bat srv_hdr_sec
```

2. 运行以下命令，将 srv_pri 变更为主服务器：

```
onmode -d make primary srv_pri
```

此命令使 srv_pri 成为主服务器，并重定向集群中的其他辅助服务器以指向新的主服务器。该命令也会关闭原 HDR 主服务器（srv_hdr_sec），因为高可用性环境中只能存在一个主服务器。

3. 通过运行以下命令，将 srv_hdr_sec 初始化为 HDR 辅助服务器：

UNIX™ 系统：

```
$INFORMIXDIR/bin/hdrmksec.sh srv_pri
```

Windows™ 系统：

```
hdrmksec.bat srv_pri
```

主服务器发生故障时重新启动

主服务器发生故障后，重新启动 HDR 或 RS 集群的步骤取决于辅助服务器是否已成为主服务器，以及该辅助服务器成为主服务器的方法。

辅助数据库服务器未更改为标准数据库服务器

如果必须在主数据库服务器发生故障且辅助数据库服务器未更改为标准服务器的情况下重新启动 HDR 或 RS 集群，请使用 oninit 命令启动主数据库服务器。

辅助数据库服务器已手动更改为标准数据库服务器

如果在主数据库服务器发生故障后必须重新启动 HDR 或 RS 集群，并且已将辅助数据库服务器手动更改为标准数据库服务，请完成下表中的步骤。

表 71: 将辅助数据库服务器手动更改为标准服务器后的重新启动步骤

步骤	在主数据库服务器上	在辅助数据库服务器上
1.		onmode -s 此步骤将辅助服务器（即现为标准服务器）变为静默模式。所有连接至该数据库服务器的客户端必须断开连接。执行更新的应用程序必须重定向到主服务器。
2.		onmode -d secondary *prim_name*
3.	oninit 如果所有写入辅助数据库服务器的逻辑日志记录仍在辅助数据库服务器的磁盘上，那么当您发出 oninit 命令时，主数据库服务器将从该磁盘恢复这些记录。	

步骤	在主数据库服务器上	在辅助数据库服务器上
	如果已经备份并释放了辅助数据库服务器上的逻辑日志文件，那么这些文件中的记录将不再在磁盘上。在这种情况下，系统将提示您从磁带中恢复这些逻辑日志文件（步骤 4）。 对于 ontape 用户： 如果您想通过网络读取逻辑日志记录，请将逻辑日志磁带设备设置为正在运行辅助数据库服务器的计算机上的设备。	
4.	如果提示您从磁带中复原逻辑日志记录，请执行此步骤。 ontape 命令 ontape -l ON-Bar 命令 onbar -r -l	

辅助数据库服务器自动更改为标准数据库服务器

如果在主数据库服务器发生故障后必须重新启动 HDR 或 RS 集群，并且已将辅助数据库服务器自动更改为标准数据库服务，请完成下表中的步骤。

表 72: 将辅助数据库服务器自动更改为标准服务器后的重新启动步骤

步骤	在主数据库服务器上	在辅助数据库服务器上
1.	% oninit 如果 DRAUTO = 1，那么此数据库服务器类型将设置为主数据库服务器。 如果 DRAUTO = 2，那么当此数据库服务器重启时，其类型将设置为辅助服务器。 如果所有写入辅助数据库服务器的逻辑日志记录仍在辅助数据库服务器的磁盘上，那么当您发出 oninit 命令时，主数据库服务器将从该磁盘恢复这些记录。 如果已经备份并释放了辅助数据库服务器上的逻辑日志文件，那么这些文件中的记录将不再在磁盘上。在这种情况下，系统将提示您从磁带中恢复这些逻辑日志文件（步骤 2）。 对于 ontape 用户： • 请将逻辑日志磁带设备设置为正在运行辅助数据库服务器的计算机上的设备。	如果 DRAUTO = 1，那么当您进行主服务器备份时，辅助数据库服务器将自动正常关机。这可确保所有客户端连接断开。然后，类型将切换回辅助服务器。执行更新的所有应用程序必须重定向到主数据库服务器。 如果 DRAUTO = 2，那么辅助数据库服务器将自动切换为主数据库服务器。原主数据库服务器在重新启动并连接到其他确认为主数据库服务服的服务器后将切换为辅助数据库服务器。
2.	如果提示您从磁带中复原逻辑日志记录，请执行此步骤。 ontape 命令 % ontape -l ON-Bar 命令 onbar -r -l	

相关链接

自动切换 在第407页

数据损坏后恢复共享磁盘集群

如果共享磁盘集群发生故障，那么必须对受影响的数据库空间执行复原。需要执行的复原类型取决于关键数据是否损坏。

相关链接

高可用性群集的备份和复原 在第335页

关键数据损坏

如果主服务器发生故障，导致根数据库空间、包含逻辑日志文件的数据库空间，或包含物理日志的数据库空间的损坏，那么必须将发生故障的数据库服务器视为其磁盘上没有数据。必须对主服务器执行完全复原。在这种情况下，主服务器和 SD 辅助服务器处于脱机状态。

要在严重介质故障之后恢复共享磁盘集群：

1. 对主服务器执行完全复原。根据备份是使用 ON-Bar 或 ontape 实用程序执行的，运行以下命令之一：
   - onbar -r
   - ontape -r

   复原完成后，主服务器重新启动。
2. 重新启动 SD 辅助服务器。

此外，也可以对主服务器上关键数据库空间执行冷复原，重新启动 SD 辅助服务器，然后对非关键数据库空间执行热复原。

关键数据未损坏

如果没有包含关键介质的磁盘发生故障，则可以使用热复原来复原受影响的数据库空间。在这种情况下，主服务器和 SD 辅助服务器处于联机状态。

要复原共享磁盘集群上的非关键数据：

1. 关闭并重新启动 SD 辅助服务器。
2. 对受影响的数据库空间执行热复原。根据备份是使用 ON-Bar 或 ontape 实用程序执行的，运行以下命令之一：
   - onbar -r 加上要复原的数据库空间的名称。
   - ontape -r -D 加上要复原的数据库空间的名称。

在辅助服务器成为主服务器后恢复 SD 集群

如果在原主服务器出现故障后，SD 集群中的辅助服务器成为主服务器，那么可以使用脚本重新建立主服务器，然后将当前的主服务器转换成辅助服务器。

在此示例中，主服务器 srv_pri 故障转移至 SD 辅助服务器 srv_sds_sec。此时，主服务器是 srv_sds_sec，并且集群中的其他辅助服务器现在都指向 srv_sds_sec。要将集群复原至 srv_pri 发生故障转移之前的状态，请遵循以下步骤：

1. 如果必要，在 srv_pri 的onconfig 文件中设置以下参数：

```
SDS_ENABLE 1
SDS_PAGING <path 1>,<path 2>
SDS_TEMPDBS <dbsname>,<dbspath>,<pagesize>,<offset>,<size>
```

   dbsname 值必须唯一。此外，*dbsname* 在所有现有的数据库空间、Blob 空间，和智能大对象空间中必须是唯一，包括从主服务器上继承来的（可能已禁用的）临时空间。如果有多个 SD 辅助服务器，每个服务器的 *dbsname* 值必须是唯一，并且不得与其他 SD 辅助服务器或主服务器共享。请参阅 设置共享磁盘辅助服务器 在第309页以获取有关设置这些参数的信息。
2. 在 srv_pri 上运行 oninit 命令，将 srv_pri 初始化为 SD 辅助服务器。

3. 对 srv_pri 执行手动故障转移，使其成为主服务器：

```
onmode -d make primary srv_pri
```

上面的命令将 srv_sds_sec 从集群中移除，并使 srv_pri 成为主服务器。

4. 在 srv_sds_sec 上运行 oninit 命令，将 srv_sds_sec 复原成 SD 辅助服务器。

第 VI 部分

分布式数据

多阶段提交协议

两阶段提交协议 可确保跨多个数据库服务器的事务统一提交和回滚。您可以将 SinoDB® 数据库服务器与事务管理器一起使用，以便处理非SinoDB® 数据库中的数据。跨 SinoDB® 数据库服务器的分布式查询支持两阶段提交。

这些主题包含使用两阶段提交协议的信息。有关手动恢复失败的两阶段提交事务，请参阅 从失败的两阶段提交手动恢复 在第436页。

这些主题还包含有关使用支持符合 XA 的外部数据源的事务的信息，这些数据源可参与两阶段提交事务。请参阅*SinoDB* 事务对符合 *XA* 的外部数据源的支持 在第422页。

事务管理器

事务管理器支持两阶段提交和回滚。例如，您的数据库为 SinoDB®，会计系统为 Oracle，而汇款系统为 Sybase，那么可以使用事务管理器在不同的数据库之间进行通信。您还可以使用事务管理器通过分布式事务而非 Enterprise Replication 或高可用性数据复制，确保 SinoDB® 数据库或非SinoDB® 数据库之间的数据一致性。

与事务管理器搭配的 TP/XA 库

全局事务 为查询中涉及多个数据库服务器的分布式查询。全局事务环境包含以下部分：

- 客户端应用程序
- 资源管理器（SinoDB® 数据库服务器）
- 事务管理器（供应商软件）

TP/XA 是一个函数库，使数据库服务器充当 X/Open DTP 环境中的资源管理器。将 TP/XA 库作为 SinoDB® ESQL/C 的一部分安装，可启动第三方事务管理器和数据库服务器之间的通信。X/Open 环境支持大型的高性能 OLTP 应用程序。

当您的数据库具备以下特征时，请使用 TP/XA：

- 数据分布在多供应商数据库中
- 事务包括 SinoDB® 和非SinoDB® 数据。

相关链接

高可用性集群中的 *XA* 在第423页

Microsoft™ Transaction Server (MTS/XA)

数据库服务器支持 Microsoft™ Transaction Server (MTS/XA) 在 XA 环境中充当事务管理器。要使用 MTS/XA，请安装 SinoDB® 客户端软件开发工具包，最新版本的 SinoDB® ODBC 驱动程序，和 MTS/XA。MTS/XA 在 Windows™ 上运行。如需更多信息，请联系SinoDB® 技术支持，及参阅 《*SinoDB®* 客户端产品安装指南》 和 MTS/XA 文件。

相关链接

高可用性集群中的 *XA* 在第423页

SinoDB® 事务对符合 XA 的外部数据源的支持

SinoDB® 事务管理器为 SinoDB® 的主要部分，而不是独立模块，可以识别符合 XA 的外部数据源。这些数据源可参与两阶段提交事务。

对于参与特殊事务事件（例如：准备、提交或回滚）的分布式事务的每个符合 XA 的外部数据源，事务管理器都会对其运行支持例程。该交互符合 X/Open XA 接口标准。

符合 XA 的外部数据源（也被称为 资源管理器）的事务支持使您能够执行以下操作：

- 创建符合 XA 的外部数据源的类型和实例。
- 创建或修改用户定义例程、虚拟表接口或虚拟索引接口以启用符合 XA 的外部数据源为来自符合 XA 的数据源的外部数据提供数据访问机制。

  MQ 扩展是提供此类外部数据访问的用户定义例程中的一个示例。
- 使用 SinoDB® 注册符合 XA 的外部数据源。
- 注销符合 XA 的外部数据源。
- 在同一个全局事务内使用多个符合 XA 的外部数据源。

仅具有日志的 SinoDB® 数据库和符合 ANSI 的数据库支持与符合 XA 的外部数据源的事务协调，因为这些数据库支持事务。无日志的数据库不支持与符合 XA 的外部数据源的事务协调。

您可以使用以下的 DDL 语句，它们是 SQL语句的扩展用于管理 XA 数据源类型和数据源：

语句	描述
CREATE XADATASOURCE TYPE	创建符合 XA 的外部数据源类型
CREATE XADATASOURCE	创建符合 XA 的外部数据源实例
DROP XADATASOURCE	删除符合 XA 的外部数据源实例
DROP XADATASOURCE TYPE	删除符合 XA 的外部数据源类型

有关这些语句的更多信息，请参阅 《*SinoDB® SQL* 指南: 语法》。

SinoDB® 和符合 XA 的外部数据源的交互是通过一组用户定义且支持 XA 的例程，例如xa_open、xa_end、xa_commit 和 xa_prepare。在使用 CREATE XADATASOURCE TYPE 语句前，创建这些支持例程。有关更多信息，请参阅 《*SinoDB® DataBlade® API* 程序员指南》。

在创建符合 XA 的外部数据源后，可使用 mi_xa_register_xadatasource()（或 ax_reg()）和 mi_xa_unregister_xadatasource()（或 ax_unreg()）函数将该数据源注册到当前事务中和注销该数据源。在分布式环境中，必须在本地协调者服务器上注册数据源。注册是暂时的，仅在事务期间有效。有关使用这些函数的更多信息，请参阅 《*SinoDB® DataBlade® API* 函数参考》 和 《*SinoDB® DataBlade® API* 程序员指南》。

使用以下 onstat 选项显示涉及符合 XA 的数据源的事务信息。

onstat 选项	此命令显示的符合 XA 的数据源信息
onstat -x	显示事务中 XA 参与者信息。
onstat -G	显示全局事务中 XA 参与者信息。

onstat 选项	此命令显示的符合 XA 的数据源信息
onstat -g ses ***session id***	显示会话信息，包括参与事务的 XA 数据源的信息。

SinoDB® MQ 扩展提供符合 XA 数据源的外部数据访问机制。

相关链接

高可用性集群中的 *XA* 在第423页
《*SinoDB* 数据库扩展用户指南》: *MQ* 消息传递

高可用性集群中的 XA

X/Open 分布式事务处理（DTP）模型允许高可用性集群中可更新的辅助服务器在分布式事务中充当资源管理器。

XA 全局事务中的参与者有三种：

- 应用程序（AP）：定义事务边界，并指定构成事务分支的操作。
- 资源管理器（RM）：提供对资源（如数据库）的访问。
- 事务管理器（TM）：指定事务分支的标识符（XID）、监视事务进度，协调事务分支以完成操作和故障恢复。

在高可用性集群中，会话可以从集群中任何服务器创建事务分支或连接事务分支。例如，可以从 server_2 连接已从 server_1 分离的事务分支。应用程序可以使用连接管理器连接到集群，而无需追踪事务起始服务器。对于松散和紧密耦合事务，事务管理器也可以使用连接管理器连接到集群通过提交、回滚或忽视来完成 XA 事务（请参阅松耦合和紧耦合方式 在第424页）。

所有从辅助服务器启动的全局事务分支都使用现有代理接口重定向到主服务器。主服务器启动和维护所有事务分支，并执行与分支关联的所有请求工作。

在可更新辅助服务器上启动了一个 XA 事务时，主服务器将启动一个相应的 XA 事务。在主服务器上的 XA 事务执行 XA 事务的整个生命周期（启动、 结束、 准备，以及提交或回滚）。辅助服务器上的 XA 事务用于支持未重定向到主服务器的查询。当发出对 xa_end() 函数的调用时，系统将释放 XA 事务，并将用户会话从 XA 事务分离。所有的 XA 事务请求和 XA 事务内发出的所有写操作都将重定向到主服务器。

以下功能特定于 SinoDB® XA 实施：

- 所有 XA 接口请求在可更新辅助服务器上都可使用（请参阅辅助服务器上的数据库更新 在第331页）。
- 支持从可更新辅助服务器启动、准备以及提交或回滚 XA 事务。
- 支持 xa_recover() 函数，该函数用于从资源管理器获取已就绪事务分支的列表。
- 支持在高可用性集群服务器之间迁移 XA 事务分支。集群中的任何服务器都可以连接到 XA 事务分支，不论该事务分支是否源自该服务器。
- XA 客户端和事务管理器可以使用连接管理器连接到任何高可用性集群服务器（请参阅通过连接管理器进行连接管理 在第340页）。
- 支持将 XA 请求从辅助服务器重定向到主服务器。
- 支持对 XA 事务执行事务挽救，但故障切换后完成事务除外（参阅在集群故障转移期间完成事务 在第339页）。
- 如果运行已重定向的 XA 事务的辅助服务器故障，则回滚该事务。
- 支持在 XA 环境内运行，但位于 XA 事务外的 SQL 事务。

在辅助服务器上运行的 XA 事务有以下限制：

- 不支持从其他用户会话 （在相同或不同的辅助服务器上）恢复暂停的全局事务分支。
- 用户会话不能连接到与另一辅助服务器上的其他用户会话关联的全局事务分支。
- XA 事务与辅助服务器上的其他数据具有相同的限制。请参阅 辅助服务器上的数据库更新 在第331页。
- XA 事务不能在只读辅助服务器上启动。如果应用程序试图在只读辅助服务器上创建 XA 事务，将会收到 XA 错误代码 XAER_RMERR。此外，在只读辅助服务器上运行 xa_prepare()、xa_commit() 或 xa_rollback() 将返回错误代码 XA_NOTA (-4)。
- 只读辅助服务器支持以下的 XA API:

- xa_open()
- xa_close()

重要: If you are using the .NET Framework with the Microsoft™ Transaction Server to manage XA transactions on a high availability cluster, you must use the TransactionScope class instead of the ServiceConfig class. The TransactionScope class is available in .NET Framework 3.5.

重要: 如果您要将 .NET Framework 用于 Microsoft™ Transaction Server 以管理高可用性集群上的 XA 事务，则必须使用 TransactionScope 类而非 ServiceConfig 类。 TransactionScope 类可用于 .NET Framework 3.5。

相关链接

高可用性和可扩展性 在第288页
辅助服务器上的数据库更新 在第331页
与事务管理器搭配的 *TP/XA* 库 在第421页
Microsoft Transaction Server (MTS/XA) 在第422页
SinoDB 事务对符合 *XA* 的外部数据源的支持 在第422页
松耦合和紧耦合方式 在第424页

松耦合和紧耦合方式

数据库服务器支持松耦合和紧耦合方式的 XA 全局事务：

- 松耦合方式 表示不同的数据库服务器可协调事务，但不可共享资源。来自事务的所有分支的记录在逻辑日志中以独立事务显示。
- 紧耦合方式支持不同的数据库服务器可协调事务并且共享资源，例如锁定和日志记录。事务的所有分支的记录在逻辑日志中以单一事务显示。

由 BEA Systems 提供的 Tuxedo 事务管理器支持松耦合方式。Tuxedo 可在UNIX™ 和 Windows™ 上运行。

Windows only: 仅在 Windows™ 上运行的 MTS/XA 事务管理器支持紧耦合方式。数据库服务器上的 MTS 紧耦合事务支持包括：

- 支持两层架构（业务逻辑层和数据访问层）的应用程序。
- 连接池和会话池。

MTS 紧耦合事务支持不会影响现有松耦合事务支持。同一数据库服务器能同时使用松耦合和紧耦合事务支持。

MTS 紧耦合事务支持具有以下限制：

- 临时表限于一个事务分支。在同一全局事务中的不同事务分支不能共享临时表。
- 在同一全局事务中的不同事务分支不能共享游标。
- 在同一全局事务中的不同事务分支不能共享隔离级别或锁等待方式。必须单独设置每个事务分支的隔离级别和锁等待方式或设置为缺省级别。如果要把所有事务分支设置成同一隔离级别，则必须使用 SQL 语句来位每个事务分支指定此信息。

有关受支持事务管理器的完整列表，请与您的销售代表联系。

相关链接

高可用性集群中的 *XA* 在第423页

两阶段提交协议

两阶段提交协议提供自动恢复机制，防止事务执行过程中出现系统或介质故障。两阶段提交协议确保所有参与的数据库服务器接收并执行同一指令（提交或回滚事务），而无论是否有本地或网络故障。

如果任何数据库服务系统无法提交它这一部分的事务，那么一定会阻止参与事务的所有数据库服务器提交它们的工作。

当使用两阶段提交协议时

数据库服务器对在多个数据库服务器中修改数据的任何事务自动使用两阶段提交协议。

例如，假设三个名为 australia、italy 和 france 的服务器连接如下图所示。

图 80: 连接的数据库服务器

如果执行以下示例中显示的命令，其结果是在三台不同数据库服务器上执行一次更新和两次插入。

```
CONNECT TO stores_demo@italy
BEGIN WORK
   UPDATE stores_demo:manufact SET manu_code = 'SHM' WHERE manu_name = 'Shimara'
   INSERT INTO stores_demo@france:manufact VALUES ('SHM', 'Shimara', '30')
   INSERT INTO stores_demo@australia:manufact VALUES ('SHM', 'Shimara', '30')
COMMIT WORK
```

两阶段提交概念

每个全局事务都有一个协调者 和一个或多个参与者，定义如下：

- 协调者可指导全局事务的解决方案。它决定是否提交或终止全局事务。

  两阶段提交协议总是将协调者角色分配给当前数据库服务器。在单个事务期间，协调者角色不可更改。在当使用两阶段提交协议时 在第425页的样本事务中，italy 为协调者。如果把示例中的第一行改为以下语句，那么两阶段提交协议将把协调者角色分配给 france。

  ```
  CONNECT TO stores_demo@france
  ```

  使用 onstat -x 选项显示分布式事务的协调者。有关更多信息，请参阅 监视全局事务 在第431页。
- 每个参与者指导一个 事务分支的执行，该事务分支为全局事务中涉及单个本地数据库的部分。在以下情况时全局事务包括多个事务分支：
  - 应用程序使用多个进程为全局事务工作。
  - 多个远程应用程序为同一个全局事务工作。

  在当使用两阶段提交协议时 在第425页中，参与者是 france 和 australia。协调者数据库服务器 italy 也作为参与者，因为它也在进行更新。

两阶段提交协议依赖两种通信， 消息 和 逻辑日志记录:

- 消息在协调者和每个参与者之间传递。来自协调者的消息包括事务标识号及指令（如 prepare to commit、commit 或 roll back）。来自每个参与者的消息包括事务状态和所采取动作的报告(如 can commit 或 cannot commit、committed 或 rolled back）。
- 事务逻辑日志记录储存在磁盘或磁带中以确保数据的完整性和一致性，即使在参与的数据库服务器（参与者或协调者）上发生故障。

  有关更多的详细信息，请参阅 两阶段提交和逻辑日志记录 在第433页。

两阶段提交协议的阶段

在两阶段提交事务中，协调者向所有参与者发送所有数据修改指令（如，插入）。接着，协调者启动两阶段提交协议。两阶段提交协议分为两部分： 预提交阶段和后决策阶段。

预提交阶段

在预提交阶段，协调者和参与者执行以下对话：

协调者

协调者指导每个参与者数据库服务器准备提交事务。

参与者

每个参与者通知协调者它是否可以提交其事务分支。

协调者

协调者基于每个参与者的响应来决定是否提交或回滚事务。它仅当所有参与者指示它们可以提交各自事务分支时才决定提交。如果有任何参与者指示它尚未准备好提交它的事务分支（或如果它未响应），那么协调者将结束全局事务。

后决策阶段

在后决策阶段，协调者和参与者执行以下对话：

协调者

协调者将提交记录或回滚记录写入协调者的逻辑日志中，然后指导每个参与者数据库服务器提交或回滚事务。

参与者

如果协调者发出提交消息，则参与者通过将提交记录写入逻辑日志然后向协调者发送事务已提交的确认消息来提交事务。如果协调者发出回滚消息，则参与者将回滚事务，但不会向协调者发送确认消息。

协调者

如果协调者发出提交事务的消息，它将在收到每个参与者的确认消息后才会结束全局事务。如果协调者发出回滚事务的消息，它将不会等待参与者的确认消息。

两阶段提交协议如何处理故障

两阶段提交协议的设计是以在所有参与的数据库服务器上保持数据完整性的方式来处理系统和介质故障。如果发生故障，两阶段提交协议将执行自动恢复。

自动恢复处理的故障类型

以下事件可导致协调线程或参与者线程终止或挂起，因此需要自动恢复：

- 协调者的系统故障
- 参与者的系统故障
- 网络故障
- 系统管理员终止协调线程
- 系统管理员终止参与者线程

管理员在自动恢复中的角色

自动恢复中管理员的唯一角色是在系统或网络故障后将协调者和/或参与者恢复联机。

> 重要： 慢速网路无法触发自动恢复。除非协调者系统发生故障、网络发生故障或管理员终止协调线程，否则此处描述的恢复机制均不会生效。

协调者故障的自动恢复机制

如果协调线程发生故障，每个参与者必须在提交或回滚事务前或在回滚事务后决定是否启动自动恢复。此责任是假定结束优化的一部分。（请参阅假定结束的优化 在第427页。）

参与者故障的自动恢复机制

只要参与者线程预提交在两阶段提交协议完成之前终止的工作项，就会发生参与者恢复。参与者恢复的目标是根据协调者的决定来完成两阶段提交协议。

参与者恢复机制是由协调者还是参与者驱动，是根据协调者决定提交还是回滚全局事务而决定。

重要： 要在跨服务器事务打开的同时支持下级服务器关闭或重新启动之后的自动恢复，sqlhosts 文件必须包含每一个可能启动分布式操作的数据库服务器的条目。在自动恢复期间，从逻辑日志来恢复协调者名称，且下级服务器与协调者重新连接以完成该事务。由于协调者总是使用自己的 onconfig 文件的 DBSERVERNAME 配置参数中的名称来向各参与者标识它自己，因此协调者的 DBSERVERNAME 设置必须是参与者都已知的网际协议连接名称，但是也可使用正确的连接协议，为协调者和下级服务器之间的连接至少定义一个 DBSERVERALIASES 设置。下级服务器必须可以使用协调者的 DBSERVERNAME 设置或 DBSERVERALIASES 设置来连接协调者。

假定结束的优化

假定结束的优化是描述两阶段协议如何处理事务回滚的术语。

回滚是按以下方式处理的。当协调者决定事务必须回滚时，那么它会向所有参与者发送消息以回滚它们的工作片段。协调者不会等待此消息的确认，而是继续进行以关闭事务并将其从共享内存中除去。如果参与者尝试确定该事务的状态，即查明事务已提交还是已回滚（例如：在参与者恢复期间）— 它将在共享内存中找不到任何事务状态。参与者必须将此解释为表示事务已回滚。

独立操作

两阶段提交中的独立操作是一种独立于两阶段提交协议而发生的操作。独立操作可能与两阶段提交协议指定的操作相对立，也可能不对立。如果独立操作与两阶段提交协议相对立，那么操作会导致错误或启发式决策。启发式决策可能导致不一致的数据库，并且需要手动的两阶段提交恢复。手动恢复是相当复杂的管理过程，应尽量避免。（有关手动恢复过程的说明，请参阅 从失败的两阶段提交手动恢复 在第436页。）

启动独立操作的情境

两阶段提交协议期间的独立操作很少见，但它可能发生在以下情况中：

- 参与者的工作片段发展成长事务错误并回滚。
- 管理员在协议的后决策阶段使用 onmode -z 停止参与者线程。
- 管理员在协议的后决策阶段使用 onmode -Z 结束参与者事务（工作片段）。
- 在协调者发出提交决策并发现参与者故障后，管理员使用 onmode -z 或 onmode -Z 结束协调者数据库服务器上的全局事务。此操纵总是导致错误，尤其是错误 -716。

独立操作的可能结果

如前所述，不是所有的独立操作与两阶段提交协议相对立 。独立操作可能导致以下三种结果：

- 成功完成两阶段提交协议
- 错误状况
- 启发式决策

如果独立操作不是与两阶段提交协议相对立，那么事务将正常提交或回滚。如果操作过早结束全局事务，会导致错误状况。在协调者上结束全局事务不会被视为启发式决策。如果操作与两阶段提交协议相对立，那么会导致启发式决策。所有这些情况均在随后各节中进行说明。

允许事务成功完成的独立操作

独立操作不一定要与两阶段提交协议相对立。例如，如果在参与者数据库服务器上的工作片段因为发展成为长事务而回滚，且协调者发出回滚全局事务的决策，那么数据将保持一致性。

导致错误状况的独立操作

如果协调者数据库服务器管理员在协调者发出 提交决策后，运行 onmode -z（停止协调线程）或 onmode -Z（停止全局事务），那么所有事务的信息将从协调者数据库服务器上的共享内存中移除。

该操作不会被视为启发式决策，因为它不干扰两个阶段协议。它可能是可接受的，或干扰了参与者恢复并导致错误。

在所有参与者可以毫无困难地提交事务的任何时候，该操作是可接受的。在这种情况下，强制结束事务的操作是多余的。仅当协调者准备终止事务时，才会收到您运行 onmode -Z 的指示。

但实际上，仅当您试图加快结束保持打开时间异常长的全局事务时，才可能会考虑在协调者数据库服务器上运行 onmode -z 或 onmode -Z 。在这种情况下，问题的根源可能是某些参与者数据库服务器发生故障。协调者尚未收到参与者提交其工作片段的确认，而协调者将尝试与参与者建立通信以进行调查。

如果您在协调者正主动尝试重新建立通信的同时运行 onmode -z 或 onmode -Z，那么协调线程会遵循您的指令而终止，并在终止之前将错误 -716 写入数据库服务器消息日志。该操作将被视为错误，因为两个阶段提交协议被强迫中断，使协调者无法确定数据库是否一致。

在协调者数据库服务器上停止全局事务不会被视为启发式决策，但它可能导致不一致的数据库。例如，如果参与者最终恢复联机但在协调者共享内存中找不到全局事务，则它将回滚其工作片段，从而导致数据库不一致。

导致启发式决策的独立操作

当以下 两个 条件都成立时，某些独立操作会发展成启发式决策：

- 参与者数据库服务器已向协调者发送 can commit 消息并随后回滚。
- 协调者的决定是提交事务。

当这两个条件都成立时，最终结果就是未一致实现的全局事务（由一个或多个数据库服务器提交但由另一个数据库服务器回滚）。该数据库变得不一致。

以下是两个可能的启发式决策：

- 启发式回滚（在启发式回滚场景 在第428页中有描述）
- 启发式结束事务（在启发式结束事务场景 在第430页中有描述）

在发生启发式回滚或结束事务之后，您可能需要执行手动恢复，这是一个复杂且耗时的过程。必须完全了解启发式决策，以便避免这些问题。在两阶段提交的情况中运行 onmode -z 或 onmode -Z 务必谨慎。

启发式回滚场景

在启发式回滚中，数据库服务器或管理员回滚已发送 can commit 消息的工作。

导致启发式回滚的情况

以下两种情况可导致启发式回滚：

- 逻辑日志填充至 LTXEHWM 配置参数定义的点。（参阅 《*SinoDB®* 管理员参考》 中关于配置参数的主题。）长事务状况的来源是代表全局事务执行的工作片段。
- 管理员执行 onmode -z ***session_id*** 以停止代表全局事务正在执行一项工作片段的数据库服务器线程。

在任一情况中，如果该工作片段已向其协调者发送了 can commit 消息，那么该操作将被视为启发式决策。

情况 1：逻辑日志填充至高水位标志

在两阶段提交中，将阻拦正在等待协调者指令的参与者数据库服务器完成其事务。因为事务保持打开状态，所以包含与该事务相关联的记录的逻辑日志文件无法释放。结果是逻辑日志由于并发用户的活动而继续填充。

如果在参与者等待期间，逻辑日志填充至长事务高水位标志的值 （LTXHWM），那么数据库服务器将指导所有拥有长事务的数据库服务器线程开始回滚这些长事务。如果预提交的工作片段就是造成问题的长事务，那么数据库服务器会启动启发式回滚。也就是说，该数据库服务器在没有协调者指令或协调者不知情的情况下回滚预提交的工作片段。

在两阶段提交中，包含与该工作片段相关联的记录的逻辑日志文件是视为打开的，直到写入 ENDTRANS 逻辑日志记录。该类型的事务与涉及单个数据库服务器的事务（其回滚是实际关闭事务）不同。

逻辑日志可能继续填充直到达到互斥高水位标志（LTXEHWM）。如果发生这种情况，除了那些正在回滚或正在提交的线程外，所有用户线程都将暂挂。在两阶段提交场景中，打开的事务使您无法备份逻辑日志文件与释放逻辑日志中的空间。在这些特定情况下，逻辑日志可完全填充满。如果发生这种情况，参与者数据库服务器会关闭，而您必须执行数据复原。

情况 2：系统管理员执行 onmode -z

您（作为管理员）可以决定通过运行 onmode -z 来启动预提交工作片段的启发式回滚。您可能因为想释放该工作片段拥有的资源而作出该决定。（如果通过运行 onmode -z 停止参与者线程，那么即使您没有结束事务，也能释放参与者线程拥有的所有锁定和共享内存资源。）

启发式回滚的结果

这些主题描述了在发生启发式回滚时协调者和参与者上所发生的事件，以及此过程如何导致不一致的数据库：

1. 在回滚发生的参与者数据库服务器上，会在数据库服务器逻辑日志（HEURTAUX 类型）中放置记录。释放事务拥有的锁和资源。参与者线程将以下消息写入数据库服务器消息日志中，指示发生了长事务状况和回滚：

   ```
   Transaction Completed Abnormally (rollback):
   ```

   ```
   tx=address flags=0xnn
   ```
2. 协调者发出后决策阶段的指令来提交事务。

   在发生启发式回滚的数据库服务器上的参与者线程将如下所示错误消息 -699 返回至协调者：

   ```
   -699 Transaction heuristically rolled back.
   ```

   此时该错误消息不会返回至应用程序；它对协调者来说是条内部通知。协调者将等待所有参与者响应该提交指令。直到所有参与者报告后，协调者才能确定数据库的一致性。
3. 接下去的步骤取决于发生在其他参与者上的操作。可能有两种情境。

情境 1：协调者发出提交并且所有参与者报告启发式回滚

协调者收集来自参与者的所有响应。如果每个参与者报告启发式回滚，那么结果是发生以下事件：

1. 协调者将以下消息写入到自己的数据库服务器消息日志：

   ```
   Transaction heuristically rolled back.
   ```
2. 协调者向所有参与者发送消息以结束事务。
3. 每个参与者其逻辑日志缓冲区写入 ENDTRANS 记录。（从事务表中除去该事务条目。）
4. 协调者将如下所示的错误 -699 返回至应用程序：

   ```
   -699 Transaction heuristically rolled back.
   ```
5. 在这种情况下，所有数据库保持一致。

情境 2：协调者发出提交；一个参与者提交和一个参与者报告启发式回滚

协调者收集来自参与者的所有响应。如果有至少一个参与者报告启发式回滚，以及至少一个参与者报告了确认提交，那么该结果称为 混合事务结果 。结果是发生以下事件：

1. 协调者将以下消息写入到自己的数据库服务器消息日志：

   ```
   Mixed transaction result. (pid=nn user=userid)
   ```

   pid 值是协调者进程的用户进程标识号。user 值是与协调者进程相关联的用户 ID。与此消息相关联的是附加信息，该附加信息列出每个报告了启发式回滚的参与者数据库服务器。附加信息的格式如下：

   ```
   Participant database server dbservername heuristically rolled back.
   ```
2. 协调者向每个启发式回滚其工作片段的参与者发送消息，指导各参与者结束事务。
3. 每个参与者在其逻辑日志缓冲区写入 ENDTRANS 消息。（从事务表中除去该事务条目。）
4. 协调者在其逻辑日志缓冲区写入 ENDTRANS 消息。（从共享内存事务表中除去该事务条目。）
5. 协调者将如下所示的错误 -698 返回至应用程序：

   ```
   -698 Inconsistent transaction. Number and names of servers rolled back.
   ```
6. 与此错误消息相关联的是报告启发式回滚的参与者数据库服务器的列表。如果太多数据库服务器回滚该事务，那么该列表可能会被截断。完整的列表始终包含在协调者数据库服务器的消息日志中。

在这种情况下，检查每个参与者数据库服务器站点上的逻辑日志并确定您的数据库系统是否一致。（请参阅确定事务是否实现不一致 在第436页 。）

启发式结束事务场景

启发式结束事务是管理员采取的一种独立操作，以回滚工作片段并从事务表中除去有关该事务的所有信息。当管理员执行 onmode -Z ***address*** 命令时会启动启发式结束事务进程。

每当通过运行 onmode -Z 来启动启发式结束事务时，数据库服务器支持两阶段提交协议及其自动恢复功能所需要的关键信息都会被移除。如果运行 onmode -Z，那么您应负责确定您的网络数据库系统是否一致。

何时执行启发式结束事务

仅在一种很少见的情况下，才必须运行 onmode -Z 选项来启动启发式结束事务。当已启发式回滚的工作片段保持打开时会发生这个情况，它会防止您的逻辑日志文件成为可用的。结果，逻辑日志是接近填满的危险状况。

通常，协调者会在合理时间内发出其提交或回滚决策。然而，如果协调者发生故障，并且无法返回联机以结束在参与者数据库服务器上启发式回滚的事务时，您可能将面临严重的问题。

问题场景以如下方式开始：

1. 代表全局事务执行工作片段的参与者线程已向协调者发送了 can commit 响应。
2. 该工作片段等待来自协调者的指令。
3. 当该工作片段正在等待时，逻辑日志填充超过长事务高水位标志。
4. 正在等待指令的工作片段是长事务的来源。该参与者数据库服务器指导执行线程回滚该工作片段。此操作是启发式回滚。
5. 该参与者持续等待协调者指导它结束事务。该事务保持打开。逻辑日志继续填充。

如果协调者在合理时间内联系参与者并指导它结束事务，就没有问题产生。如果在参与者数据库服务器上发生启发式回滚，并且随后协调者故障，致使协调者不能指导参与者结束事务，那么就会发生严重问题。

结果，事务保持打开。打开的事务使您无法备份逻辑日志文件及释放逻辑日志中的空间。当逻辑日志继续填充时，它可能会达到互斥存取长事务高水位标志（LTXEHWM）指定的点。如果达到该点，正常的处理将暂挂。在到达高水位标志之后的某个时间，您必须判定该打开的事务是否危及您的逻辑日志。危险就是如果逻辑日志完全填满，那么数据库服务器关闭，并且您必须执行数据复原。

尽管所有问题与运行 onmode -Z 相关联，但您必须判定是否结束事务并保护您的系统免于逻辑日志填满的可能性，还是等着看能否及时重新建立与协调者的通讯，以便在逻辑日志填满前结束事务。

如何使用 onmode -Z

仅当协调者和参与者之间的通信中断时，才会使用 onmode -Z ***address*** 命令。为了确保该通信真正中断，除非正在执行该工作片段的线程已经死亡超过 TXTIMEOUT 指定的时间，否则 onmode -Z 命令是不会运行的。有关此选项的更多信息，请参阅 《*SinoDB*® 管理员参考》。

address 参数值来自于 onstat -x 的输出。 有关 onstat -x 的更多信息，请参阅 《*SinoDB*® 管理员参考》。

启发式结束事务时执行的操作

当运行 onmode -Z 时，您指示 onmode 实用程序从事务表中移除位于指定地址的参与者事务条目。

两条记录写入逻辑日志中以记录该操作。记录类型为 ROLLBACK 和 ENDTRANS ，或如果事务已经启发式回滚，那么仅有 ENDTRANS 类型。以下消息写入参与者数据库服务器消息日志：

```
(time_stamp) Transaction Completed Abnormally (endtx): tx=address flags:0xnn
 user username tty ttyid
```

协调者接收来自发生了 onmode -Z 的参与者的错误消息以作为其对 COMMIT 指令的响应。协调者查询不再有有关该事务的信息的参与者数据库服务器。参与者数据库服务器上缺少事务表条目就表明事务已提交。协调者假定参与者已发送确认消息，但由于某种原因未接收到该消息。因为协调者不知道参与者的工作未提

交，所以它不会生成指示全局性事务不一致实现的消息。只有运行 onmode -Z 命令的管理员才知道实现不一致。

监视全局事务

使用 onstat -x 命令追踪开放事务并判定它们是否已经启发式回滚。

例如，在输出中，flags 字段中的 H 标志标识启发式回滚，G 标志标识全局事务，L 标志指示松耦合方式，T 标志标识紧耦合方式。

curlog 和 logposit 字段提供逻辑日志记录的准确位置。如果事务未回滚，那么 curlog 和 logposit 描述最近写入的日志记录的位置。当事务正在回滚，那么这些字段描述最近“undone”日志记录的位置。随着事务回滚，curlog 和 logposit 值会减少。在长事务中，curlog 和 logposit 值收敛的速度可以帮助您估算回滚需要多久时间。

有关 onstat -x 输出的更多信息及其示例，请参阅 《*SinoDB*® 管理员参考》。

您也可以使用 onstat -u 和 onstat -k 命令来追踪以及它们拥有的锁。有关详细信息，请参阅 《*SinoDB*® 性能指南》中的监视事务主题。有关 onstat -x 显示的字段的描述，请参阅 《*SinoDB*® 管理员参考》。

在辅助服务器上，当启用了故障转移（通过设置 FAILOVER_TX_TIMEOUT 配置参数 ）后完成事务时，两个全局性事务可能有相同的全局事务标识：一个是本地临时全局事务，另一个是属于恢复线程的全局性事务。快速区分真正的全局事务与临时事务的方法是执行了任何操作的真正的全局事务将有一个 B 标志。也可以使用 onstat -g ath 命令来检查事务的所有者。在调用了 xa_end() 函数之后，将删除辅助服务器上的临时全局性事务。

以下 onstat 实用程序示例输出说明了在高可用性集群环境中主服务器和辅助服务器上对 XA 事务的支持。 onstat -x、onstat -G 和 onstat -ath 命令是单独记录，但是组合的 onstat -xG 命令的输出对全局事务是特别重要。示例显示了重定向事务的每个状态。

在示例中，所显示的在辅助服务器上运行的全局事务是临时事务。临时事务用于支持在辅助服务器 上执行的 SQL 语句（不是重定向到主服务器的事务）。仅当用户线程主动和全局事务分支关联时才会显示临时事务。

以下示例显示运行 xa_start() 函数之后在辅助服务器上运行的 onstat -xG 命令的输出：

```
Transactions
                                                                                                  est.
address          flags userthread       locks  begin_logpos      current logpos    isol    rb_time  retrys
 coord
7000000104d4190  AT--G 7000000104a7b68  0      -                 -                 LC      -        0
7000000104d8bd0  ALB-G 7000000104a5aa8  1      180:0x0           180:0x4eb018      DIRTY   0:00     0

Global Transaction Identifiers
address           flags  isol    timeout  fID        gtl  bql  data
7000000104d4190   AT--G  COMMIT  0        5067085    15   4    000102030405060708090A0B0C0D0E0F000000
7000000104d8bd0   ALB-G  DIRTY   0        5067085    15   4    000102030405060708090A0B0C0D0E0F000000
```

在辅助服务器上运行的 onstat -g ath 的输出：

```
Threads:
 tid     tcb              rstcb            prty status               vp-class       name
 317     7000001500902c8  7000000104a7b68  1    cond wait  netnorm      1cpu         sqlexec
 84      7000001403a7dc0  7000000104a5aa8  3    sleeping secs: 1        5cpu         xchg_2.0
```

在主服务器上运行的 onstat -g ath 的输出：

```
Transactions
                                                                                                  est.
address          flags userthread       locks  begin_logpos      current logpos    isol    rb_time  retrys
 coord
7000000104d9e60  ATB-M 7000000104a8bc8  2      180:0x4ea018      180:0x4eb018      COMMIT  0:00     0

Global Transaction Identifiers
address           flags  isol    timeout  fID        gtl  bql  data
7000000104d9e60   AT--M  COMMIT  0        5067085    15   4    000102030405060708090A0B0C0D0E0F000000
```

以上示例中的 M 标志指示全局事务是从辅助服务器启动的。在主服务器上启动的全局事务会显示 G 标志。M 标志只在主服务器上显示。

onstat -g ath|grep 7000000104a8bc8 命令的输出：

```
 196     70000013012d3a8  7000000104a8bc8  1    sleeping secs: 1        4cpu         proxyTh
```

以下示例显示运行 xa_end() 函数之后在辅助服务器上运行的 onstat -xG 命令的输出：

```
Transactions
                                                                                  est.
address          flags userthread       locks  begin_logpos      current logpos    isol    rb_time  retrys
 coord
7000000104d8bd0  ALB-G 7000000104a5aa8  1      180:0x0           180:0x4ee018      DIRTY   0:00     0

Global Transaction Identifiers
address           flags  isol    timeout  fID        gtl  bql  data
7000000104d8bd0   ALB-G  DIRTY   0        5067085    15   4    000102030405060708090A0B0C0D0E0F000000
```

在主服务器上运行的 onstat -xG 命令的输出：

```
Transactions
                                                                                  est.
address          flags userthread       locks  begin_logpos      current logpos    isol    rb_time  retrys
 coord
7000000104d9e60  -TB-M 0               2      180:0x4ea018      180:0x4ee018      COMMIT  0:00     0


Global Transaction Identifiers
address           flags  isol    timeout  fID        gtl  bql  data
7000000104d9e60   -T--M  COMMIT  -1       5067085    15   4    000102030405060708090A0B0C0D0E0F000000
```

运行 xa_prepare() 函数之后在辅助服务器上运行的 onstat -xG 命令的输出：

```
Transactions
                                                                                  est.
address          flags userthread       locks  begin_logpos      current logpos    isol    rb_time  retrys
 coord
7000000104d8bd0  ALX-G 7000000104a5aa8  1      180:0x0           180:0x4ef018      DIRTY   0:00     0

Global Transaction Identifiers
address           flags  isol    timeout  fID        gtl  bql  data
7000000104d8bd0   ALX-G  DIRTY   0        5067085    15   4    000102030405060708090A0B0C0D0E0F000000
```

在主服务器上运行的 onstat -xG 命令的输出：

```
Transactions
                                                                                  est.
address          flags userthread       locks  begin_logpos      current logpos    isol    rb_time  retrys
 coord
7000000104d9e60  -TX-M 0               2      180:0x4ea018      180:0x4ef018      COMMIT  0:00     0

Global Transaction Identifiers
address           flags  isol    timeout  fID        gtl  bql  data
7000000104d9e60   -TX-M  COMMIT  -1       5067085    15   4    0
```

两阶段提交协议错误

以下两阶段提交协议错误需要管理员特别加以注意。

-698

如果收到错误 -698 ，那么说明已发生启发式回滚，并且已造成不一致实现的事务。启发式回滚的结果 在第429页中描述了导致该事件的情况。有关不一致事务如何发展的说明以及了解可用的选项，请参阅此信息。

-699

如果收到错误 -699，那么说明已发生启发式回滚。启发式回滚的结果 在第429页中描述了导致该事件的情况。有关不一致事务如何发展的说明，请参阅此信息。

-716

如果收到错误 -716，那么说明管理员在协调线程发出最终的决策之后已终止了该线程。在导致错误状况的独立操作 在第427页中描述了该场景。

两阶段提交和逻辑日志记录

数据库服务器使用逻辑日志记录来实现两阶段提交协议。您可以使用这些逻辑日志记录来检测启发式决策，并且如果需要的话可以帮助您执行手动恢复。（请参阅 从失败的两阶段提交手动恢复 在第436页。）

分布式事务中包含以下逻辑日志记录：

- BEGPREP
- PREPARE
- TABLOCKS
- HEURTX
- ENDTRANS

有关这些逻辑日志记录的信息，请参阅《*SinoDB*® 管理员参考》中解释逻辑日志的章节。

本节将检查以下数据库服务器场景期间写入的逻辑日志记录的顺序：

事务提交时的逻辑日志记录

下图显示了在成功提交事务的两阶段提交协议期间逻辑日志记录的写入顺序。

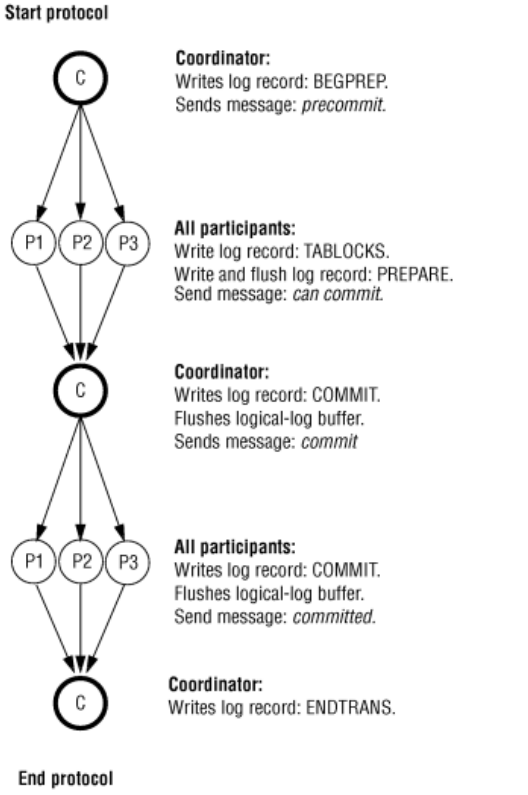

图 81: 在提交事务期间的逻辑日志写入

某些逻辑日志记录必须立即从逻辑日志缓冲区清空；对其他的来说，清空并不是关键的。

协调者的提交工作记录（ COMMIT记录）包含了启动两阶段提交所需的所有信息。它还可作为协调者主机发生故障时自动恢复的起点。因为该记录对于恢复是很关键的，所以不允许它保留在逻辑日志缓冲区中。协调者必须立即清空 COMMIT 逻辑日志记录。

上图中的参与者必须立即同时清空 PREPARE 和 COMMIT 逻辑日志记录。清空 PREPARE 记录可以确保快速恢复能在参与者的主机发生故障时，确定此参与者是否为全局事务的一部分。作为恢复的一部分，参与者可以查询协调者以了解该事务的最终处置。

清空参与者的 COMMIT 记录可以确保参与者的主机发生故障时，参加者具有关于该事务所执行的操作的记录。要理解该信息为何至关重要，请考虑参与者在写入 PREPARE 记录之后但在清空 COMMIT 记录之前崩溃的情况。在快速恢复之后，恢复了 PREPARE 记录，但丢失了 COMMIT 记录（因为在故障发生时它是在逻辑日志缓冲区中）。PREPARE 记录的存在将对协调者启动有关该事务的查询。然而，协调者对该事务一无所知，因为它在收到已提交的确认后就结束了该事务。在这种情况下，参与者将把信息缺乏解释为回滚该事务的最终指示。两阶段提交协议需要立即清空参与者的 COMMIT 记录，以防止此种误解。

启发式回滚期间写入的逻辑日志记录

下图说明了数据库服务器在启发式回滚期间写入逻辑日志记录的顺序。因为启发式回滚仅发生在参与者发送它可以提交的消息以及协调者发送要提交的消息之后发生，所以协议的第一阶段与 图 *81: 在提交事务期间的逻辑日志写入* 在第433页所示相同。当发生启发式回滚时，会假设该回滚是由于参与者 P1 数据库服务器上发生长事务状况而导致的结果。最终的结果是未一致实现的事务。请参阅启发式回滚场景 在第428页。

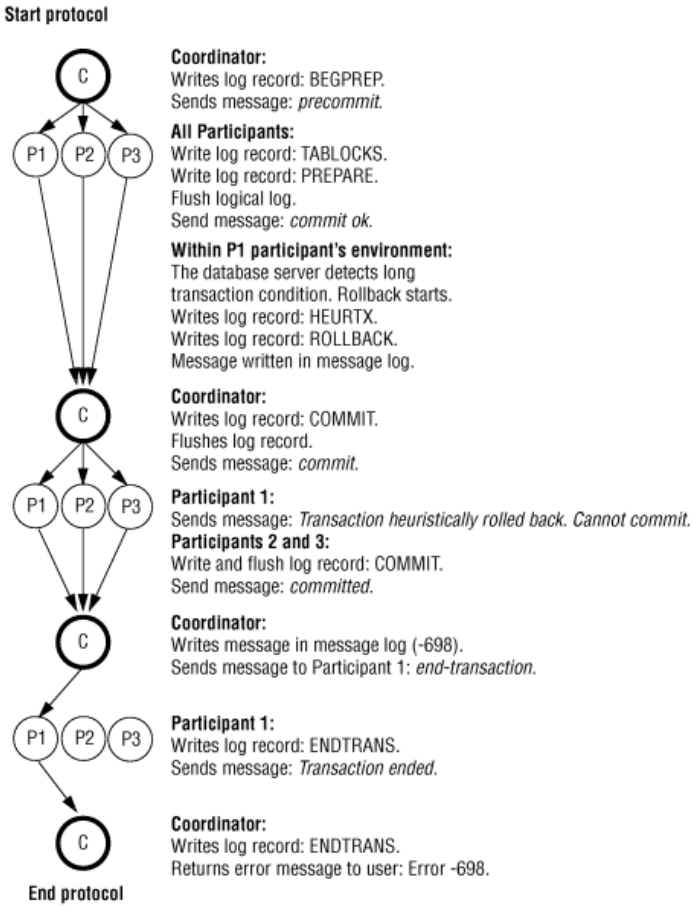

图 82: 启发式回滚期间写入的逻辑日志记录

启发式结束事务后写入的逻辑日志记录

下图说明了启发式结束事务期间逻辑日志记录的写入顺序。事件始终在参与者已发送 can commit 消息之后，数据库服务器管理员在该参与者数据库服务器上结束事务（请参阅 《*SinoDB*® 管理员参考》中有关 onmode 实用程序 的信息）的结果 。在下图中，假定在参与者 P1 数据库服务器上已经发生启发式结束事务。结果是未一致实现的事务。请参阅启发式结束事务场景 在第430页。

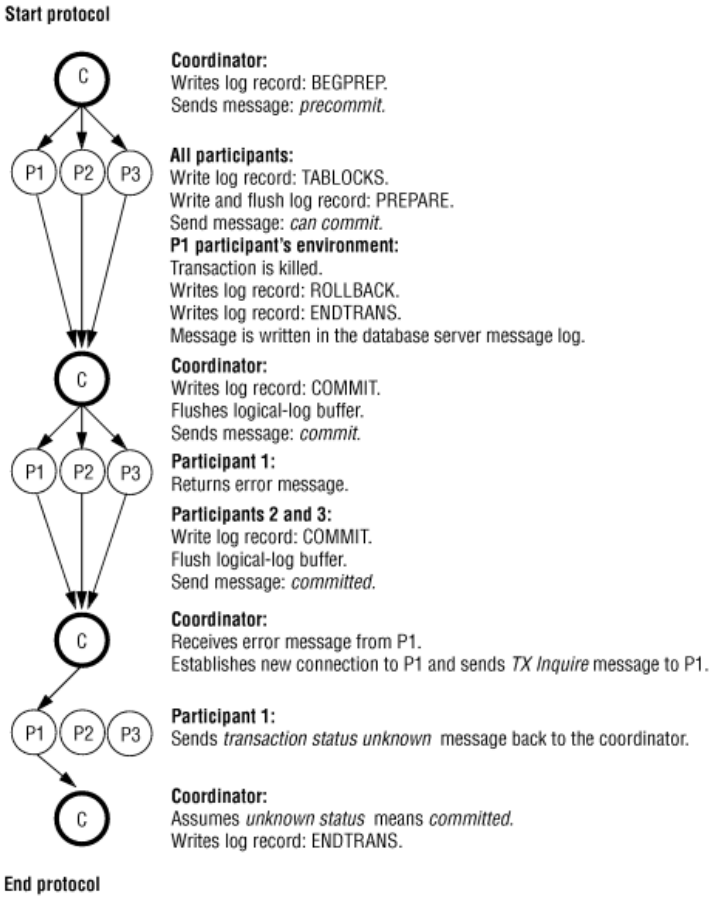

图 83: 启发式结束事务期间写入的逻辑日志记录

两阶段提交中使用的配置参数

以下两个配置文件参数是特定于分布式环境：

- DEADLOCK_TIMEOUT
- TXTIMEOUT

虽然两个参数都是指定超时时间，但是这两个参数是独立的。有关这些配置参数的更多信息，请参阅《*SinoDB*® 管理员参考》。

DEADLOCK_TIMEOUT 参数的功能

如果强制分布式事务等待共享内存资源的时间超过 DEADLOCK_TIMEOUT 指定的秒数，那么拥有该事务的线程会假设存在多服务器死锁。返回以下错误消息：

```
-154 ISAM error: deadlock timeout expired - Possible deadlock.
```

DEADLOCK_TIMEOUT 的默认值是 60 秒。请谨慎调整该值。如果将其设置得过低，各个数据库服务器会结束非死锁的事务。如果将其设置得过高，多服务器死锁可能会减少并发性。

TXTIMEOUT 参数的功能

TXTIMEOUT 配置参数是特定于两阶段提交协议。仅当事务协调者和参与者之间的通信中断，且必须重新建立时，才会使用该参数。

TXTIMEOUT 参数指定参与者数据库服务器在分布式事务期间等待接收来自协调者数据库服务器的提交指令的时间。如果已经过了 TXTIMEOUT 指定的时间，那么参与者数据库服务器会检查该事务的状态，以确定参与者是否必须启动自动参与者恢复。

TXTIMEOUT 以秒为单位指定。默认值是 300 （5 分钟）。此参数的最佳值根据您的特定的环境和应用程序而变化。修改此参数之前，请阅读 两阶段提交协议如何处理故障 在第426页中的说明。

从失败的两阶段提交手动恢复

分布式事务遵循两阶段提交协议。某些操作的发生独立于两阶段提交协议，就导致事务不一致地实现。（请参阅 独立操作 在第427页。）这些情况中，可能需要从事务进行手动恢复。

确定是否需要手动恢复

以下主题概述确定数据库一致性与是否需要更正状况的过程中的步骤。

以下主题中描述了每一步骤。

确定事务是否实现不一致

首要任务是确定事务是否由于独立操作而不一致地实现。

全局事务过早结束

如果您运行 onmode -z 命令来结束协调者上的全局事务，那么事务可能会不一致地实现。（有关该情况是如何发生的说明，请参阅 导致错误状况的独立操作 在第427页。）您可以通过首先检查协调者的数据库服务器消息日志来检查不一致的事务。查找以下错误消息：

```
-716 Possible inconsistent transaction.
Unknown servers are server-name-list.
```

此消息列出了所有作为参与者的数据库服务器。检查每一参与者的逻辑日志。如果至少一个参与者执行了提交和一个参与者执行了回滚，则该事务的实现不一致。

启发式结束事务

如果您运行 onmode -Z ***address*** 命令来结束某个参与者执行的工作片段，并且协调者决定提交事务，那么事务实现将不一致。（有关该场景的说明，请参阅 启发式结束事务场景 在第430页。）检查每一参与者的逻辑日志。如果至少一个参与者执行了提交和一个参与者执行了回滚，那么该事务实现是不一致。

启发式回滚

您可以用以下方法来确定启发式决策所影响的特定数据库服务器参与者，从而回滚事务：

- 检查应用程序中 COMMIT WORK 语句的返回代码。

  以下消息表示有一个参与者执行了启发式回滚：

  ```
  -698 Inconsistent transaction. Number and names of servers rolled back.
  ```
- 检查数据库服务器消息日志文件中的消息。

  如果由于参与数据库服务器上的启发式决策导致数据库不一致，那么协调者数据库服务器消息日志文件中会出现以下消息：

  ```
  Mixed transaction result. (pid=nn user=user_id)
  ```

  只要返回错误 -698，就会写入此消息。与此消息相关联的是涉及 事务回滚的参与者数据库服务器的列表。此为完整列表。 如果大量参与者回滚了该事务，那么以 -698 错误消息创建的列表可能会被截断。
- 检查每一参与者的逻辑日志。

如果至少一个参与者回滚了其工作片段，且一个参与者提交了其工作片段，那么事务实现将不一致。

确定分布式数据库是否包含不一致的数据

如果确定事务实现是不一致的，那么您必须确定这种情况对您的分布式数据库系统意味着什么。尤其，您必须确定数据完整性是否受到影响。

无论何时当一个参与者回滚的工作片段与另一参与者更新的工作片段相关联，那么实现不一致的事务将造成问题。由于分布式事务不支持引用多个数据库服务器上数据的约束，因此无法使用 SQL 定义这些相关性。只有当数据可以在两个独立的事务中更新时，这些工作才是独立的（不存在相关性）。否则，这些工作将视为是相关联的。

在您继续之前，请考虑导致错误的事务。更新的数据与回滚的数据是否相互相关？单个事务可能基于除了维持数据完整性之外的原因而包含多个更新。例如，以下三种可能的原因：

- 减少事务开销
- 简化编码
- 程序员喜好

并验证每个假定已提交事务的参与者数据库服务器实际修改了数据。只读数据库服务器可能作为已提交事务的参与者列出。

如果不一致的事务不会导致数据完整性的违反，此时您就可以退出该过程。

从逻辑日志获取信息

要确定数据完整性是否受到不一致实现的全局事务的影响，就必须重建该全局事务并确定事务的哪些部分已提交，哪些部分已回滚。使用 onlog 实用程序来获取必要信息。过程如下：

1. 在包含 HEURTX 记录的参与者上重建事务。
   a) 参与者数据库服务器的逻辑日志是您进行信息搜索的开始点。日志中的每个记录都有一个本地事务标识号（xid）。获取 HEURTX 记录的 xid。
   b) 使用本地 xid 找出所有相关联的日志记录，这些记录作为此项工作片段的一部分而回滚。
2. 确定哪一个数据库服务器充当全局事务的协调者。
   a) 在参与者上查找包含同一本地 xid 的 PREPARE 记录。PREPARE 记录标记参与者两阶段提交协议的起点。
   b) 使用 onlog -l 选项来获取 PREPARE 记录的详细输出。
      该记录包含全局事务标识（GTRID）与协调数据库服务器的名称。有关 GTRID 的信息，请参阅 获取全局事务标识 在第437页。
3. 从协调者日志获取其他参与者列表。
   a) 检查协调者数据库服务器的日志记录。查找 BEGPREP 记录。
   b) 检查 BEGPREP 记录的详细输出。
      如果该记录中 GTRID 的前 32 字节与参与者的 GTRID 相匹配，那么 BEGPREP 记录就是同一全局事务的一部分。请注意，BEGPREP 详细输出的 ASCII 部分中显示的参与者。
4. 重建每一参与者上的事务。
   a) 在每个参与者数据库服务器上，读取逻辑日志以查找包含与该事务关联的 GTRID 的 PREPARE 记录，并获得该参与者执行工作片段的本地 xid。
   b) 在每个参与者数据库服务器上，使用本地 xid 查找与此事务 （已提交或已回滚） 相关联的所有逻辑日志记录。

在您遵循该过程之后，您将知道事务的所有参与者是哪些，分配给每个参与者哪些工作片段，以及每个工作片段是已回滚还是已提交。根据该信息，您可确定该独立的操作是否影响数据完整性。

获取全局事务标识

当全局事务开始时，它会接收到称为全局事务标识（GTRID）的唯一标识号。GTRID 包含协调者的名称。GTRID 会写入协调者的 BEGPREP 逻辑日志记录，与每个参与者的 PREPARE 逻辑日志记录。

要查看 GTRID，请使用 onlog -l 选项。GTRID 在记录的数据部分偏移 20 个字节的位置，其长度为 144 个字节。以下示例显示了 BEGPREP 记录的 onlog -l 输出。协调者为 chrisw。

```
4a064    188  BEGPREP  4        0  4a038       0    1
         000000bc 00000043 00000004 0004a038 .......C .......8
         00087ef0 00000002 63687269 73770000 ..~..... chrisw..
         00000000 00000000 00000000 00087eeb ........ ......~.
         00006b16 00000000 00000000 00000000 ..k..... ........
         00000000 00000000 00000000 00000000
         00000000 00000000 00000000 00000000
         00000000 00000000 00000000 00000000
         00000000 00000000 00000000 00000000
         00000000 00000000 00000000 00000000
         00000000 00000001 6a756469 74685f73 ........ judith_s
         6f630000 736f6374 63700000          oc..soct cp..
```

协调者的 BEGPREP 记录与参与者的 PREPARE 记录（这些是同一全局事务的一部分）的 GTRID 的前 32 个字节是相同的。例如，将以下示例中 PREPARE 记录的 GTRID 与上一个示例中 BEGPREP 记录的 GTRID 相比较。

```
c7064    184  PREPARE  4        0  c7038    chrisw
         000000b8 00000044 00000004 000c7038 .......D ......p8
         00005cd6 00000002 63687269 73770000 ......  chrisw..
         00000000 00000000 00000069 00087eeb ........ ...i..~.
         00006b16 00000000 00000010 00ba5a10 ..k..... ......Z.
         00000002 00ba3a0c 00000006 00000000 ......:. ........
         00ba5a10 00ba5a1c 00000000 00000000 ..Z...Z. ........
         00ba3a0e 00254554 00ba2090 00000001 ..:..%ET .. .....
         00000000 00ab8148 0005fd70 00ab8148 .......H ...p...H
         0005fe34 0000003c 00000000 00000000 ...4...< ........
         00000000 00ab80cc 00000000 00ab80c4 ........ ........
         00ba002f 63687269 73770000 00120018 .../chrisw......
         00120018 00ba0000                   ........
```

决定是否需要执行操作来更正情况

如果不一致的事务创建了不一致的数据库，那么您可以有以下三个选项 ：

- 让网络数据库保持处于不一致的状态。
- 除去提交事务之处受到的事务影响，从而回滚整个事务。
- 在回滚事务之处重新应用事务的影响，从而提交事务。

如果事务没有严重影响数据库数据，那么您可以让数据库保持处于不一致的状态。如果执行事务的应用程序可以按原样继续，并且您判定通过除去事务影响或重新应用事务将数据库返回到一致状态的成本（时间与工作量方面的成本）过高，那么您就可能遇到这种情况。

您无需立即做出此决定。您可以使用以下所述的方法来确定事务更新了哪些数据，以及哪些记录受到了影响。

当您做出决定时，请考虑到没有自动过程或实用程序可以回滚已提交事务或提交已回滚事务的一部分。以下将说明如何查看数据库服务器消息日志和逻辑日志来找出受影响的记录。如果不具备详细的应用程序知识，那么仅凭消息并不足以确定所发生的事。基于您对于应用程序和系统的知识，您必须基于您对应用程序和系统的了解来确定是回滚还是提交事务。您还必须编写执行回滚或提交的补偿事务。

手动恢复示例

此示例说明了手动恢复所涉及的工作。以下为用户 nhowe 执行的 SQL 语句。返回错误 -698。

```
dbaccess
```

```
CREATE DATABASE tmp WITH LOG;
CREATE TABLE t (a int);
CLOSE DATABASE;
CREATE DATABASE tmp@apex WITH LOG;
CREATE TABLE t (a int);
CLOSE DATABASE;
DATABASE tmp;
BEGIN WORK;
INSERT INTO t VALUES (2);
INSERT INTO tmp@apex:t VALUES (2);
COMMIT WORK;
### return code -698
```

以下为从当前数据库服务器的逻辑日志中摘录的片段：

```
addr      len   type      xid      id   link
.....
17018    16         CKPOINT  0       0     13018    0

18018    20         BEGIN    2       1     0        08/27/91 10:56:57
   3482     nhowe

1802c    32         HINSERT  2       0     18018    1000018  102
   4

1804c    40         CKPOINT  0       0     17018    1

   begin    xid    id addr     user

   1        2          1  1802c    nhowe

19018    72         BEGPREP  2       0     1802c   6d69    1

19060    16         COMMIT   2       0     19018   08/27/91 11:01:38

1a018    16         ENDTRANS 2       0     19060   580543
```

以下为从 apex 数据库服务器的逻辑日志中摘录的片段：

```
addr      len   type      xid      id   link
.....
16018    20         BEGIN    2            1   0        08/27/91
   10:57:07 3483     pault

1602c    32         HINSERT  2       0     16018    1000018  102
   4

1604c    68         PREPARE  2       0     1602c      eh

17018    16         HEURTX   2       0     1604c          1

17028    12         CLR      2       0     1602c

17034    16         ROLLBACK 2       0     17018    08/27/91 11:01:22

17044    40         CKPOINT  0       0     15018    1

      begin    xid      id addr     user
      1        2        1  17034    --------

18018    16         ENDTRANS 2       0     17034     8806c3
   ....
```

首先，您要尝试将当前数据库服务器日志中的事务与 apex 数据库服务器日志中的事务相匹配。BEGPREP 与 PREPARE 日志记录均包含 GTRID.可以通过使用 onlog -l 与查看 BEGPREP 与 PREPARE日志记录的数据部分来抽取 GTRID。 GTRID 在数据部分偏移 22 个字节，长为 68 个字节。一个较为简单但不太精确的方法是查看 COMMIT 或 ROLLBACK 记录的时间。虽然由于将提交（或回滚）消息从协调者传送到参与者所花费的时间而稍有延迟，但这两个时间必须是很接近的。（第二种方法缺乏准确度，因为虽然来自同一协调者的并发事务很可能不在同一时间提交，但并发事务可以在同一时间提交的。）

更正该样本情况

1. 查找所有已更新的记录。
2. 使用 onlog 与记录类型表来识别记录类型（插入、 删除、 更新）。
3. 使用每个记录的 onlog -l 输出来获取本地 xid、表空间号和 rowid。
4. 通过比较表空间号与 systables 系统目录表的 partnum 列中的值，从而将表空间号映射到表名。
5. 运用您的应用程序知识来决定需要哪一种操作来更正该情况。

在本示例中，不同日志中的 COMMIT 与 ROLLBACK 记录的时间戳很接近。没有其他活动事务造成另一并发提交或回滚的可能性。在此示例中，在当前数据库服务器上提交了已分配 十六进制 rowid 102（十进制 258）的插入（HINSERT）。因此，补偿事务如下：

```
DELETE FROM t WHERE rowid = 258
```

第 VII 部分

自动监视与更正操作概述

可以使用 SQL 管理 API，调度程序，向下钻取查询来管理自动维护、监视和管理任务。

SinoDB® 的这些组件简化了复杂系统中的信息收集和服务器维护。

SQL 管理 API

SQL 管理 API 通过 SQL 函数执行远程管理。因为 SQL 管理 API 操作完全在 SQL 中执行，所以可以在客户端工具中使用这些函数来管理数据库服务器。

调度程序

调度程序是一组任务，用于在预定义时间执行 SQL 语句或按照服务器内部决定执行 SQL 语句。SQL 语句可以收集信息，或监视和调整服务器。

向下钻取查询

向下钻取查询提供关于最近执行的 SQL 语句的统计信息，以便追踪各个 SQL 语句的性能并分析语句历史。

可以在 HDR 服务器对的主服务器上使用 SQL 管理 API 与调度程序。

每个工具均需要额外的磁盘空间来存储信息。

还可以使用基于 PHP 的 web 浏览器管理工具， SinoDB® 开放管理工具 (OAT)（OAT）从单一位置来管理多个数据库服务器实例。可使用 OAT 执行的任务包括:

- 通过 SQL 管理 API 和调度程序来定义并管理自动执行的任务
- 创建和显示性能柱状图，以便分析和调整 SQL 语句

相关链接

查看 *SQL* 管理 *API* 历史记录 在第460页
调度程序 在第441页
使用 *SQL* 管理 *API* 执行远程管理 在第458页
向下钻取查询 在第461页

调度程序

可以使用调度程序来创建工作，用于在可预测的时间运行管理任务或收集信息。调度程序使用SQL语句，而不是操作系统的工作调度工具。

调度程序由 sysadmin 数据库中的一组表控制的。

调度程序有四种不同的作业类型可供选择：

任务

在特定时间和频率下运行操作。

传感器

在特定时间和频率下运行操作以收集数据、创建结果表、将数据存储在结果表中，以及在指定的时间之后清除旧数据。

启动任务

仅当服务器从静默模式转换为联机模式时才运行的任务。

启动传感器

仅当服务器从静默模式转换为联机模式时才运行的传感器。

任务或传感器的操作可以是一个或多个 SQL 语句、用户定义例程，或存储过程。

除了定义任务或传感器的操作，还可以使用调度程序执行以下操作：

- 将任务与传感器关联到功能组
- 每次运行任务或传感器时跟踪执行时间和返回值
- 定义不同级别的警告
- 定义阈值以控制何时运行任务或传感器

调度程序包含自动运行的内置任务与传感器。可修改内置任务与传感器，与定义自己的任务与传感器。

磁盘空间需求

调度程序表与传感器结果表可能使用大量磁盘空间。

可以使用以下公式来估计一个传感器的磁盘使用情况：

所收集的行数 * 所收集的行大小 * 每日的数据收集频率 * 保留期

对所有传感器重复此估计，然后您可以确定所需空间的接近估计值。

可以通过降低数据收集的频率来减少存储的数据量，或通过更新 ph_task 表以缩短保留期。

可以使用 SQL 管理 API 将 sysadmin 数据库移动到其他的数据库空间，但是该数据库中的所有现有数据都将丢失。

有关 sysadmin 数据库的更多信息，请参阅 《*SinoDB*® 管理员参考》。

相关链接

自动监视与更正操作概述 在第441页
《*SinoDB* 管理员参考》：调度程序表
自动优化数据存储 在第215页

调度程序表

调度程序表位于 sysadmin 数据库，其包含有关任务和传感器的信息。

sysadmin 数据库包括下表中列出的调度程序表。ph_task 表与其他各表均有直接联系。

表 73: 调度程序表

表	描述
ph_alert	包含与必须监视的任务关联的错误、 警告或参考消息的列表。ph_alert 表包含数据库服务器自动使用的内置警报。可以添加您自己的警报。
ph_group	包含组名称的列表。每个任务和传感器都是组的成员。ph_group 表包含数据库服务器使用的内置组。可以添加您自己的组。
ph_run	包含有关每个任务和传感器的运行方式与运行时间的信息。

表	描述
ph_task	列出任务和传感器，并包含有关数据库服务器运行这些任务和传感器的方式和时间的信息。ph_alert 表包含数据库服务器自动使用的内置任务与传感器。可以添加您自己的任务与传感器。
ph_threshold	包含与任务或传感器关联的阈值的列表。如果达到某个阈值，关联的任务可执行某个操作，例如在 ph_alert 表中插入警报。ph_threshold 表包含数据库服务器使用的内置阈值。可以添加您自己的阈值。
results	包含传感器收集的历史数据的多个表。这些表的结构是由 ph_task 表的传感器定义中的 CREATE TABLE 语句确定。

有关这些表的详细信息，请参阅 《*SinoDB*® 管理员参考》。

相关链接

《*SinoDB* 管理员参考》：调度程序表

内置任务和传感器

调度程序包含自动运行的内置任务和传感器。

下表显示内置调度程序任务和传感器。传感器具有用于储存所收集的信息的结果表，和用于确定信息存储时间长度的保留期。可以通过更新 ph_task 表来更改任务和传感器的属性，如频率。某些任务通过阈值触发。可以通过更新 ph_threshold 表来更改阈值。可以通过将 ph_task 表中 tk_enable 列的值更改为 f 来禁用任务或传感器。

可以通过查询 ph_task 表中的 run_duration 列来确定任务所需的时间。

表 74: 内置任务和传感器

任务或传感器	描述	结果表	频率	保留期
add_storage	该任务在配置了自动空间管理时自动添加更多存储空间。		按需要	
Alert Cleanup	该任务从 ph_alert 表中清除存在时间超过了阈值（15 天）的所有旧警报条目。该阈值在 ph_threshold 表中的名称为 ALERT HISTORY RETENTION。		每天一次	
auto_compress	该任务压缩配置为自动压缩的表。			
auto_crsd	该任务对表和分段执行压缩、收缩、重新打包和取消分段。 默认情况下，禁用该任务。必须通过更新 ph_task 表启用该任务。		每周一次	

任务或传感器	描述	结果表	频率	保留期
	每个操作在 ph_threshold 表中有两行：一行控制是否启用该操作，另一行控制该操作的阈值。 有关更多信息，请参阅 自动优化数据存储 在第215页。			
autoreg exe	此任务注册首次使用的数据库扩展。		按需要	
autoreg migrate-console	内部。该任务使用日志或缓冲日志的日志记录选项来检查每个数据库，并在必要时将所有内置数据库扩展迁移到数据库服务器的正确版本。该任务在必要时为各个数据库创建子任务。		服务器启动时	
autoreg vp	该任务在必要时为数据库扩展创建专用虚拟处理器。		按需要	
auto_tune_cpu_vps	如果计算机上分配的虚拟处理器数少于 CPU 处理器数的一半时，该任务将自动添加 CPU 虚拟处理器。		服务器启动时	
Auto Update Statistics Evaluation	该任务根据当前自动更新统计（AUS）策略来分析记录的所有数据库中的所有表、标识必须更新分布的表，并生成这些表的 UPDATE STATISTICS 语句。AUS 策略由 ph_threshold 表的阈值设置： • AUS_AGE：统计信息在 30 天之后更新。 • AUS_CHANGE：统计信息在更改数据量达到 10% 之后更新。 • AUS_AUTO_RULES：按照准则更新统计信息。 • AUS_SMALL_TABLES：行数少于 100 的表始终更新其统计信息。 有关更多信息，请参阅 *Automatic statistics updating*。		每天一次	
Auto Update Statistics Refresh	该任务运行 Auto Update Statistics Evaluation 任务生成的 UPDATE STATISTICS 语句。通过 ph_threshold 表中的阈值 AUS_PDQ 将更新统计信息的 PDQ 优先级设置为 10。		每周六和周日的 1:00 AM 到 5:00 AM	

任务或传感器	描述	结果表	频率	保留期
bad_index_alert	该任务检查损坏的索引。如果找到任何损坏的索引，将向 ph_alert 表添加警告警报。 有关更多信息，请参阅 *Validate indexes*。		每天一次	
bar_act_log_rotate	该任务对 BAR_ACT_LOG 配置参数中指定的 ON-Bar 活动日志文件进行循环交替。 当 ON-Bar 活动日志文件循环交替时，服务器将切换到新联机消息日志文件，并将之前的日志文件的标识号加 1。达到日志文件最大数量之后，会删除具有最高标识号的日志文件。 要循环交替的最大日志数的阈值在 ph_threshold 表中指定。		每 30 天的 3 A.M.（最大日志文件数为 12）	
bar_debug_log_rotate	该任务对 BAR_DEBUG_LOG 配置参数中指定的 ON-Bar 调试日志文件进行循环交替。 当 ON-Bar 调试日志循环交替时，服务器将切换到新联机消息日志文件，并将之前的日志文件的标识号加 1。达到日志文件最大数量之后，会删除具有最高标识号的日志文件。 要循环交替的最大日志数的阈值在 ph_threshold 表中指定。		每 30 天的 3 A.M.（最大日志文件数为 12）	
check_backup	该任务将检查以确保备份自 ph_threshold 表中的阈值指定的时间以来已运行： • REQUIRED LEVEL BACKUP：任何级别备份之间最大为 2 天 • REQUIRED LEVEL 0 BACKUP：0 级备份之间最大为 2 天 如果未进行备份，将向 ph_alert 表添加警告警报。		每天一次	
check_for_ipa	对于每个具有一个或多个未完成就定位的更改操作的表，该任务为其在 ph_alert 表中添加一个条目。		每周一次	

任务或传感器	描述	结果表	频率	保留期
idle_user_timeout	该任务将结束空闲时间超过了 60 分钟的用户会话。 该任务默认为禁用。您必须通过更新 ph_task 表来启用该任务。 有关更多信息，请参阅 自动终止空闲连接 在第65页。		每 2 小时	
ifx_ha_monitor_log_replay_task	该任务监视高可用性集群的重放位置。		未设置	
ifx_TrickleFeed_load_*ID*	该任务持续刷新数据集市中的数据。数据集市和加速器的名称列在任务描述中。在为数据集市启用缓慢更新之后，在调度程序中会出现该任务。每个启用缓慢更新的数据集市都有一个单独的任务。任务名称中的 *ID* 是唯一的。 有关更多信息，请参阅 *ifx_setupTrickleFeed() function*。		启用缓慢更新时指定的每秒数	
Job runner	该任务使用专用 dbWorker 线程在后台运行 SinoDB® 开放管理工具 (OAT) 的服务器任务。仅供 SinoDB® 开放管理工具 (OAT) 内部使用。		按需要	
Job results clean up	该任务除去 存在时间超过了阈值（30 天）的SinoDB® 开放管理工具 (OAT) 作业结果条目。仅供 SinoDB® 开放管理工具 (OAT) 内部使用。		每天一次	
mon_checkpoint	该传感器保存有关检查点的信息。	mon_checkpoint	每小时	7 天
mon_chunk	该传感器保存有关块使用情况和 I/O 块性能的常规信息。	mon_chunk	每小时	30 天
mon_command_history	该任务从 command_history 表中删除存在时间超过了阈值（30 天）的行。该阈值在 ph_threshold 表中的名称为 COMMAND HISTORY RETENTION。		每天一次	
mon_compression_estimates	该传感器保存有关数据压缩时可节省的空间量的信息。	mon_compression_estimates	每周一次	30 天

任务或传感器	描述	结果表	频率	保留期
mon_config	该传感器保存 onconfig 文件中每个配置参数的最新值。	mon_config	每天一次	
mon_config_startup	该传感器保存当服务器启动时 onconfig 文件中每个配置参数的值。	mon_config	服务器启动时	99 天
mon_iohistory	该传感器保存有关 I/O 块的性能信息。您可以更改 ph_threshold 表中的 IO_SAMPLES_PER_HOUR 参数以更频繁地收集信息。		每小时	30 天
mon_low_storage	该任务扫描数据库空间列表，以查找低于 SP_THRESHOLD 配置参数指定的阈值的空间。然后，该任务通过使用存储池中的条目来扩展块或增加块，以扩展空间。 有关更多信息，请参阅 自动空间管理 在第185页。	mon_low_storage	每小时	7 天
mon_memory_system	该传感器收集有关服务器所用内存量的信息。	mon_memory_system	每小时	7 天
mon_page_usage	该传感器保存有关存储空间中已用和可用的页面的信息。	mon_page_usage	每天一次	7 天
mon_profile	该传感器保存服务器概要文件信息。	mon_prof	每 4 小时	30 天
mon_sysenv	该启动传感器保存数据库服务器启动时有关环境的信息。	mon_sysenv	服务器启动时	60 天
mon_table_names	该传感器保存表名称及其创建时间。	mon_table_names	每天一次	30 天
mon_table_profile	该传感器保存表概要文件信息，其中包括该表上发生的更新、插入和删除操作总数。	mon_table_profile	每天一次	7 天
mon_users	该传感器保存有关每个用户的概要文件信息。	mon_users	每 4 小时	7 天
mon_vps	该传感器收集虚拟处理器信息。	mon_vps	每 4 小时	15 天
online_log_rotate	该任务对 MSGPATH 配置参数中指定的联机消息日志文件进行循环交替。 当联机消息日志循环交替时，服务器将切换到新的联机消息日志文件并将之前日		每 30 天的 3 A.M.（最大日志文件数为 12）	

任务或传感器	描述	结果表	频率	保留期
	志文件的 标识号加 1。当达到日志文件的最大数量时，会删除具有最高标识号的日志文件。 要循环交替的最大日志数的阈值在 ph_threshold 表中指定。			
post_alarm_message	该任务发布警报。		每小时	
purge_tables	该任务标识已超过其清除策略的滚动窗口表。它将根据每个清除策略来放弃或分离符合条件的分段，直至满足该策略或者直至没有分段可以除去。		每天 00:45	
保存 SQL 追踪	当用户在 SinoDB® 开放管理工具 (OAT) 中启用该传感器后，该传感器将当前 SQL 历史记录跟踪缓冲区的内容保存到表中。仅供 SinoDB® 开放管理工具 (OAT) 内部使用。	sql_savesnap	每 15 分钟	1 天
SET tk_enable	该任务用于启动对日志文件进行循环交替的任务。		每 30 天的 3 A.M.	
sync_registry	该任务自动在 sqlhosts 文件格式和 Windows™ 注册表格式之间转换连接信息。		每 15 分钟	

相关链接

自动性能调优 在第27页
修改调度程序 在第458页
《*SinoDB* 管理员参考》：调度程序表

创建任务

您可创建调度任务，以用于在特定时间执行特定操作。

必须以 informix 用户或其他授权用户身份连接到 sysadmin 数据库。

要创建任务，请使用 INSERT 语句在 ph_task 表中添加一行：

1. 包含以下列的值：
   a) tk_name：为任务指定唯一名称。
   b) tk_type：将作业类型更改为 TASK 或 STARTUP TASK。
   c) tk_description：添加任务所执行操作的说明。
   d) tk_execute：添加任务所执行的操作。
      该操作可以是用户定义的函数、单个 SQL 语句，或使用 PREPARE SQL 创建的多语句就绪对象以便在运行时组合一个或多个 SQL 语句。
      命令长度限制为 2048 个字节。
2. 可选： 更改以下列的默认值：

- tk_start_time：默认启动时间为 8:00:00。对于启动任务，请设置为 NULL。
- tk_stop_time：默认停止时间为 19:00:00。对于启动任务，请置为 NULL。
- tk_frequency：默认频率为一天一次。对于启动任务，请设置为 NULL。
- tk_group：默认组为 MISC。
- tk_monday 至 tk_sunday：默认为每天运行

任务会在指定的开始时间以及随后通过频率计算的时间运行。

示例

以下任务使用 SQL 管理 API，在周一、周三和周五的上午 8:00 至下午 7:00 之间每两分钟执行一个检查点。

```
INSERT INTO ph_task
( tk_name,
tk_description,
tk_type,
tk_group,
tk_execute,
tk_start_time,
tk_stop_time,
tk_frequency,
tk_Monday,
tk_Tuesday,
tk_Wednesday,
tk_Thursday,
tk_Friday,
tk_Saturday,
tk_Sunday)
VALUES
( "Example Checkpoint",
"Example to do a checkpoint every 2 minutes.",
"TASK",
"EXAMPLES",
"EXECUTE FUNCTION admin('checkpoint')",
DATETIME(08:00:00) HOUR TO SECOND,
DATETIME(19:00:00) HOUR TO SECOND,
INTERVAL ( 2 ) MINUTE TO MINUTE,
't',
'f',
't',
'f',
't',
'f',
'f');
```

以下示例显示每天凌晨 2 点执行一次的任务的代码，以确保 command_history 表中仅包含最近的数据。在该示例中，最近的数据的定义存储在 ph_threshold 表的 Command History Interval 列中。

```
INSERT INTO ph_task
(
tk_name,
tk_group,
tk_description,
tk_type,
tk_execute,
tk_start_time,
tk_frequency
)
VALUES
```

```
(
"mon_command_history",
"TABLES",
"Monitor how much data is kept in the command history table",
"TASK",
"delete from command_history where cmd_exec_time < (
        select current - value::INTERVAL DAY to SECOND
        from ph_threshold
        where name = 'COMMAND HISTORY INTERVAL' ) ",
"2:00:00",
"1 0:00:00"
);
```

相关链接

任务和传感器的操作 在第452页
修改调度程序 在第458页

创建传感器

您可以创建调度程序传感器来收集和存储有关数据库服务器的数据

您必须以用户 informix 或其他授权用户身份连接到 sysadmin 数据库。

要创建传感器，请使用 INSERT 语句在 ph_task 表中添加一行。

1. 包含以下列的值：
   - tk_name：为任务指定唯一的名称。
   - tk_description：添加任务所执行的操作的描述。
   - tk_result_table：添加用于保存传感器所收集数据的表的名称。
   - tk_create：添加 CREATE 语句以创建结果表。结果表必须具有名为 ID 的 INTEGER 列，用于保存传感器标识。可向该表添加其他列。
   - tk_execute：添加传感器所执行的操作。该操作可以是用户定义的函数、单个 SQL 语句，或使用 PREPARE SQL 创建的多语句就绪对象以便在运行时组合一个或多个 SQL 语句。
2. （可选）更改以下列的默认值：
   - tk_type：默认值为 SENSOR。对于启动传感器，请将值改为 STARTUP SENSOR。
   - tk_delete：删除传感器数据的默认间隔时间是一天。
   - tk_start_time：默认开始时间是 8:00:00。对于启动传感器，请设置为 NULL。
   - tk_stop_time：默认结束时间是 19:00:00。对于启动传感器，请设置为 NULL。
   - tk_frequency：默认频率是一天一次。对于启动传感器，请设置为 NULL。
   - tk_group：默认组是 MISC。
   - tk_monday 到 tk_sunday：默认为每天运行。
   -

传感器会在指定的开始时间以及随后通过频率计算的时间运行。

示例

以下示例显示跟踪数据库服务器的启动环境的传感器的代码。$DATA_SEQ_ID 变量为当前传感器的执行。

```
INSERT INTO ph_task
(
tk_name,
tk_type,
tk_group,
tk_description,
```

```
tk_result_table,
tk_create,
tk_execute,
tk_stop_time,
tk_start_time,
tk_frequency,
tk_delete
)
VALUES
(
"mon_sysenv",
"STARTUP SENSOR",
"SERVER",
"Tracks the database servers startup environment.",
"mon_sysenv",
"create table mon_sysenv (ID integer, name varchar(250), value
 lvarchar(1024))",
"insert into mon_sysenv select $DATA_SEQ_ID, env_name, env_value
FROM sysmaster:sysenv",
NULL,
NULL,
NULL,
"60 0:00:00"
);
```

以下示例显示收集有关正在使用的内存量的信息并在 mon_memory_system 表中存储该信息的传感器的代码。如果该表不存在，此任务将创建该表。此任务是每 30 分钟运行一次，运行时将 mon_memory_system 表中存在超过了 30 天的数据删除。

```
INSERT INTO ph_task
(
tk_name,
tk_group,
tk_description,
tk_result_table,
tk_create,
tk_execute,
tk_stop_time,
tk_start_time,
tk_frequency,
tk_delete
)
VALUES
("mon_memory_system",
"MEMORY",
"Server memory consumption",
"mon_memory_system",
"create table mon_memory_system (ID integer, class smallint, size int8,
     used int8, free int8 )",
"insert into mon_memory_system select $DATA_SEQ_ID, seg_class, seg_size,
     seg_blkused, seg_blkfree FROM sysmaster:syssegls t",
NULL,
NULL,
INTERVAL ( 30 ) MINUTE TO MINUTE,
INTERVAL ( 30 ) DAY TO DAY
);
```

相关链接

任务和传感器的操作 在第452页
修改调度程序 在第458页

任务和传感器的操作

任务或传感器的操作是用于执行一个或多个操作的 SQL 语句或例程。

如果操作中仅包含单个操作，则 SQL 语句很有用。如果操作中包含 多个操作，则使用 C 或 Java™ 语言编写的存储过程或用户定义例程很有用。操作存储在 ph_task 表的 tk_execute 列。

创建操作时，您可有很大的灵活性。可能的操作类型包括：

- 执行 DML 操作。可使用传感器在表中插入或更新数据。可以使用任务从表中删除较旧的数据。
- 执行管理操作。可以使用任务来运行 SQL 管理 API 函数以管理数据库服务器。例如，创建一个任务以每 2 分钟执行一次检查点。
- 依据阈值执行操作。您可以使用 ph_threshold 表中的阈值来决定是否必运行某个任务操作。例如，可以创建一个任务，该任务在可用共享内存量低于阈值时添加共享内存段。
- 创建警报以报告操作或警告潜在问题。例如，可以创建一个终止空闲用户的任务，该任务终止用户会话时在 ph_alert 表中插入一行。还可以创建一个任务来监视备份，并在备份未执行时在 ph_alert 表中插入一个警告。

在任务或传感器操作中使用以下变量：

- $DATA_TASK_ID：用于指示当前任务或传感器。此变量对应于 ph_task 表中 tk_id 字段的值。
- $DATA_SEQ_ID：用于指示任务或传感器的当前执行。此变量对应于 ph_task 表中 tk_sequence 字段和 ph_run 表中 run_task_sequence 字段的值。

示例

以下操作是使用 SQL 语句，供内置的 mon_command_history 任务用于从 command_history 表中移除较旧的行。

```
DELETE FROM command_history
WHERE cmd_exec_time < (
SELECT CURRENT - value::INTERVAL DAY to SECOND
FROM ph_threshold
WHERE name = 'COMMAND HISTORY RETENTION' )
```

以下示例是一个 SQL 语句，供内置的 mon_vps 传感器用于向 mon_vps 结果表添加数据：

```
INSERT INTO mon_vps
SELECT $DATA_SEQ_ID, vpid, num_ready,
class, usecs_user, usecs_sys
FROM sysmaster:sysvplst
```

以下示例是一个存储过程，用于终止空闲超过阈值设置值的用户会话，并且向 ph_alert 表添加警报。

```
/*
 ************************************************************
 *  Create a function that will find all users that have
 *   been idle for the specified time. Call the SQL admin API to
 *   terminate those users.  Create an alert to track which
 *   users have been terminated.
 ************************************************************
 */
CREATE FUNCTION idle_timeout( task_id INT, task_seq INT)
RETURNING INTEGER

DEFINE time_allowed INTEGER;
DEFINE sys_hostname CHAR(16);
DEFINE sys_username CHAR(257);
DEFINE sys_sid      INTEGER;
DEFINE rc           INTEGER;
```

```
    {*** Get the maximum amount of time to be idle ***}
    SELECT value::integer
   INTO time_allowed
   FROM ph_threshold
          WHERE name = "IDLE TIMEOUT";

    {*** Find all users who are idle longer than the threshold ***}
    FOREACH SELECT admin("onmode","z",A.sid), A.username, A.sid, hostname
           INTO rc, sys_username, sys_sid, sys_hostname
           FROM sysmaster:sysrstcb A , sysmaster:systcblst B,
                sysmaster:sysscblst C
           WHERE A.tid = B.tid
           AND C.sid = A.sid
           AND lower(name) in  ("sqlexec")
           AND CURRENT - DBINFO("utc_to_datetime",last_run_time) >
 time_allowed UNITS MINUTE
           AND lower(A.username) NOT IN( "informix", "root")

           {*** If a user is successfully terminated, log ***}
           {*** the information into the alert table.    ***}
           IF rc > 0 THEN
               INSERT INTO ph_alert
                  (
                   ID, alert_task_id,alert_task_seq,
                   alert_type, alert_color,
                   alert_state,
                   alert_object_type, alert_object_name,
                   alert_message,
                   alert_action
                  ) VALUES (
                   0,task_id, task_seq,
                   "INFO", "GREEN",
                   "ADDRESSED",
                   "USER","TIMEOUT",
     "User "||TRIM(sys_username)||"@"||TRIM(sys_hostname)||
       " sid("||sys_sid||")"||
                              " terminated due to idle timeout.",
                   NULL
                  );
          END IF

   END FOREACH;

   RETURN 0;

END FUNCTION;
```

相关链接

创建任务 在第448页
创建传感器 在第450页
创建阈值 在第454页
创建警报 在第455页

创建组

可创建组来管理调度程序任务和传感器。

必须以 informix 用户或其他授权用户身份连接到 sysadmin 数据库。

创建任务或传感器时，可在 ph_task 表的 tk_group 列中指定 ph_group 表中的组名。

创建组：

使用 INSERT 语句向 sysadmin 数据库中的 ph_group 表添加一行。

必须在 group_name 列中包含组的名称，并在 group_description 列中包含组的描述。数据库服务器在 group_id 列中生成组的标识。

示例

以下示例添加一个名为 TABLES 的组：

```
INSERT INTO ph_group
(
group_name,
group_description
)
VALUES
(
"TABLES",
"Tasks that trim history and results tables."
);
```

创建阈值

可创建阈值来确定运行调度程序任务或传感器的条件。

必须以用户 informix 或其他授权用户身份连接到 sysadmin 数据库。

阈值指定一个值，用于与当前值进行比较，以决定是否必须运行任务或传感器。

创建阈值：

1. 使用 INSERT 语句添加 ph_threshold 表中以下列的值：
   - name：阈值的名称
   - task_name：ph_task 表中的任务的名称
   - value：阈值的值
   - value_type：阈值的数据类型（STRING 或 NUMERIC）
   - description：阈值作用的描述
2. 编写使用该阈值的任务或传感器操作。

示例

以下示例为名为 Idle_timeout 的任务添加一个名为 IDLE TIMEOUT 的阈值：

```
INSERT INTO ph_threshold
(
name,
task_name,
value,
value_type,
description
)
VALUES
(
"IDLE TIMEOUT",
"Idle_timeout",
"60",
"NUMERIC",
```

```
"Maximum amount of time in minutes for non-informix users to be idle."
);
```

```
INSERT INTO ph_threshold
(
name,
task_name,
value,
value_type,
description
)
VALUES
(
"IDLE TIMEOUT",
"Idle_timeout",
"60",
"NUMERIC",
"Maximum amount of time in minutes for non-informix users to be idle."
);
```

该任务操作是从当前时间中减去上一个用户操作的时间，并将该值与 ph_threshold 表中的值列进行比较。

相关链接

任务和传感器的操作 在第452页

创建警报

创建警报可以是调度程序任务或传感器操作的一部分。

必须以用户 informix 或其他授权用户身份连接到 sysadmin 数据库。

创建警报：

使用 INSERT 语句在 ph_alert 表中添加一行。包含以下列的值：

- ID：生成的系统；使用 0 作为值。
- alert_task_id：必须引用 ph_task 表中的作业标识。
- alert_task_seq：必须引用 ph_task 表中的作业序号。
- alert_type：选择 INFO、WARNING 或 ERROR。
- alert_color：选择 GREEN、YELLOW 或 RED。
- alert_state：选择 NEW、IGNORED、ACKNOWLEDGED、ADDRESSED。
- alert_object_type：警报描述的对象类别，如 SERVER。
- alert_object_name：对象的名称。
- alert_message：描述警报的消息。
- alert_action：用于执行更正操作的 SQL 语句或函数，或者为 NULL。

示例

以下示例添加报警以警告备份未执行。此代码片段是存储过程的一部分，该存储过程以 task_id 和 task_seq 作为其参数。

```
INSERT INTO ph_alert
(
ID,
alert_task_id,
alert_task_seq,
alert_type,
alert_color,
```

```
alert_state,
alert_object_type,
alert_object_name,
alert_message,
alert_action
)
VALUES
(
0,
task_id,
task_seq,
"WARNING",
"RED",
"NEW",
"SERVER",
"dbspace_name",
"Dbspace ["||trim(dbspace_name)|| "] has never had a level-0 backup.
    Recommend taking a level-0 backup immediately.",
NULL
);
```

相关链接

任务和传感器的操作 在第452页

监视调度程序

可以使用 onstat -g dbc 命令监视正在运行的调度程序线程。可在 ph_run 表中查看有关已完成的任务和传感器的信息。

调度程序运行时使用以下两种线程 ：

- dbWorker：这些线程运行已调度的任务和传感器。
- dbScheduler：此线程准备已安排要运行的下一个任务或传感器。

要查看当前正在运行的任务和传感器以及将运行的下一个任务或传感器的信息，请使用 onstat -g dbc 命令。

要查看已完成的任务和传感器的信息，请查询 sysadmin 数据库的 ph_run 表。您必须以用户 informix 或其他已授权用户身份连接到 sysadmin 数据库。

示例

以下为 onstat -g dbc 命令的输出，显示了 dbWorker 线程和 dbScheduler 线程：

```
Worker Thread(0)    46fa6f10
==================================
Task:                47430c18
Task Name:           mon_config_startup
Task ID:             3
Task Type:           STARTUP SENSOR
Last Error
    Number               -310
    Message              Table (informix.mon_onconfig)
                           already exists in database.
    Time                 09/11/2007 11:41
    Task Name            mon_config_startup

Task Execution:      onconfig_save_diffs

WORKER PROFILE
```

```
    Total Jobs Executed            10
    Sensors Executed                8
    Tasks Executed                  2
    Purge Requests                  8
    Rows Purged                     0

Worker Thread(1)    46fa6f80
====================================
Task:                4729fc18
Task Name:           mon_sysenv
Task ID:             4
Task Type:           STARTUP SENSOR
Task Execution:      insert into mon_sysenv select 1, env_name,
                       env_value FROM sysmaster:sysenv


WORKER PROFILE
    Total Jobs Executed             3
    Sensors Executed                2
    Tasks Executed                  1
    Purge Requests                  2
    Rows Purged                     0


Scheduler Thread    46fa6f80
====================================
Run Queue
    Empty
Run Queue Size        0
Next Task             7
Next Task Waittime    57
```

以下输出显示了 ph_run 表中的四个调度程序作业的历史记录：

```
SELECT * FROM ph_run;

RUN_ID          1
RUN_TASK_ID     2
RUN_TASK_SEQ    1
RUN_RETCODE     0
RUN_TIME        2009-07-20 13:04:59
RUN_DURATION    0.131850300007433
RUN_ZTIME       1248109468
RUN_BTIME       1248109468
RUN_MTIME       1248109499

RUN_ID          2
RUN_TASK_ID     3
RUN_TASK_SEQ    1
RUN_RETCODE     0
RUN_TIME        2009-07-20 13:04:59
RUN_DURATION    0.120845244247991
RUN_ZTIME       1248109468
RUN_BTIME       1248109468
RUN_MTIME       1248109499

RUN_ID          3
RUN_TASK_ID     4
RUN_TASK_SEQ    1
RUN_RETCODE     0
RUN_TIME        2009-07-20 13:04:59
RUN_DURATION    0.00254887164461759
RUN_ZTIME       1248109468
```

```
RUN_BTIME       1248109468
RUN_MTIME       1248109499

RUN_ID          2087
RUN_TASK_ID     7
RUN_TASK_SEQ    742
RUN_RETCODE     0
RUN_TIME        2009-09-09 11:09:51
RUN_DURATION    0.00489335523104662
RUN_ZTIME       1248109468
RUN_BTIME       1248109468
RUN_MTIME       1252508991
```

修改调度程序

可以修改调度程序任务、传感器、警报、阈值或组的属性。可修改内置属性与您添加的属性。

必须以用户 informix 或其他授权用户身份连接到 sysadmin 数据库。

修改调度程序属性：

将 UPDATE 语句用于 sysadmin 数据库中相应的调度程序表。

示例

以下示例停止正在运行的名为 task1 的任务。

```
UPDATE ph_task
  SET tk_enable = "F"
    WHERE tk_name = "task1";
```

以下示例将内置传感器 mon_profile 收集数据的时间量更改为 99 天：

```
UPDATE ph_task
SET tk_delete = "INTERVAL ( 99 ) DAY TO DAY"
WHERE tk_name = "mon_profile";
```

以下示例将名为 COMMAND HISTORY RETENTION 阈值更改为 20，这样 command_history 表会将有关 SQL 管理 API 命令的信息保留 20 天：

```
UPDATE ph_threshold SET value = "20 0:00:00"
WHERE name = "COMMAND HISTORY RETENTION";
```

相关链接

内置任务和传感器 在第443页
创建任务 在第448页
创建传感器 在第450页

使用 SQL 管理 API 执行远程管理

可使用 SQL 管理 API 利用 SQL 语句执行远程管理任务。

SQL 管理 API 函数采用一个或多个定义任务的参数。许多任务也可使用命令行实用程序来完成。使用 SQL 管理 API 函数的优点是能够从其他数据库服务器远程运行这些任务。而运行命令行实用程序命令时，必须直接连接到该数据库服务器。

可以使用 SQL 管理 API 来执行以下类型的管理任务：

- 控制数据压缩
- 更新配置参数
- 检查数据、分区和扩展数据块的一致性，控制 B 型树扫描程序，以及强制执行检查点
- 设置和管理 Enterprise Replication
- 设置和管理高可用性集群
- 控制日志记录和逻辑日志
- 控制共享内存和添加缓冲池
- 控制镜像过程
- 控制决策支持查询
- 更改服务器模式
- 添加、删除和配置存储空间
- 控制 SQL 语句高速缓存
- 控制和配置 SQL 跟踪
- 动态启动和停止监听控制线程
- 执行其他任务，例如移动 sysadmin 数据库、终止会话或添加虚拟处理器

有关 SQL 管理 API 的更多信息，参阅 《*SinoDB*® 管理员参考》。

相关链接

自动监视与更正操作概述 在第441页
《*SinoDB* 管理员参考》：*SQL* 管理 *API* 概述

SQL 管理 API 的 admin() 与 task() 函数

SQL 管理 API 中包含两个定义在 sysadmin 数据库中的函数： admin() 与 task()。

这两个函数执行相同的任务，但是返回不同格式的结果。task() 函数返回描述命令结果的字符串。admin() 函数返回整数。

默认情况下，仅用户 informix 可以连接到 sysadmin 数据库。如果授权用户 root 或 DBSA 组的成员连接到 sysadmin 数据库，那么用户 root 或 DBSA 组的成员也可运行 SQL 管理 API 的 task() 与 admin() 函数。

在不包括任何其他语句的事务中运行 task() 或 admin() 函数。

可以使用 EXECUTE FUNCTION 语句来执行 admin() 与 task() 函数。例如，以下等同于 oncheck -ce 命令的 SQL 语句可指示数据库服务器检查扩展数据块：

```
BEGIN WORK;
EXECUTE FUNCTION admin("check extents");
END WORK;
```

可在调度程序任务操作中使用 SQL 管理 API 函数。例如，在任务操作中使用以下语句定义用于创建数据库空间的任务：

```
EXECUTE FUNCTION admin("create dbspace","dbspace2","/work/dbspace2","20 MB");
```

相关链接

《*SinoDB* 管理员参考》：*SQL* 管理 *API* 概述

查看 SQL 管理 API 历史记录

可查看 sysadmin 数据库的 command_history 表中前 30 天运行的所有 SQL 管理 API 函数的历史记录。

必须以用户 informix 或其他授权用户身份连接到 sysadmin 数据库。

command_history 表显示管理任务是通过 admin() 还是 task() 函数执行的，并显示有关运行命令的用户、运行命令的时间、命令，以及数据库服务器完成运行命令时返回的消息的信息。

显示命令历史记录：

使用SELECT 语句来返回 command_history 表中的数据。

以下示例显示了过去 30 天所有命令的历史记录：

```
SELECT * FROM command_history;
```

下表显示示例命令以及示例 command_history 表中的相关联结果。有关 command_history 表中所有信息的描述，请参阅 《*SinoDB*® 管理员参考》。

表 75: command_history 表中某些信息的示例

命令	返回消息示例
set sql tracing on	SQL tracing on with 1000 buffers of 2024 bytes.
create dbspace	Space 'space12' added.
checkpoint	Checkpoint completed.
add log	Added 3 logical logs to dbspace logdbs.

相关链接

自动监视与更正操作概述 在第441页
《*SinoDB* 管理员参考》: *command_history* 表

控制 command_history 表的大小

可缩短 command_history 表中行的保留期或从该表中删除行，以防止其变得过大。

必须以用户 informix 或其他授权用户身份连接到 sysadmin 数据库。

默认情况下， command_history 表中的行在 30 天之后会自动删除。保留期是由 ph_threshold 表中的 COMMAND HISTORY RETENTION 行来控制。

缩短保留期：

使用 UPDATE 语句来修改 ph_threshold 表中 COMMAND HISTORY RETENTION 行的值。

以下示例将保留期设置为 25 天 ：

```
UPDATE ph_threshold
SET value = "25"
WHERE name = "COMMAND HISTORY RETENTION";
```

可以使用 DELETE 或TRUNCATE TABLE 之类的SQL 语句来手动删除该表中的数据。也可在 ph_task 表中创建任务来清除 command_history 表中的数据。

以下示例显示了监视 command_history 表中数据数量并当表太旧时清除数据的任务。

```
INSERT INTO ph_task
( tk_name, tk_type, tk_group, tk_description, tk_execute,
tk_start_time, tk_stop_time, tk_frequency )
VALUES
("mon_command_history",
"TASK",
"TABLES",
"Monitor how much data is kept in the command history table",
"delete from command_history where cmd_exec_time < (
        select current - value::INTERVAL DAY to SECOND
        from ph_threshold
        where name = 'COMMAND HISTORY RETENTION' ) ",
DATETIME(02:00:00) HOUR TO SECOND,
NULL,
INTERVAL ( 1 ) DAY TO DAY);
```

相关链接

《SinoDB 管理员参考》: command_history 表

向下钻取查询

可使用向下钻取查询或 SQL 跟踪来收集有关运行的每个 SQL 语句的统计信息，以及分析语句历史记录。

SQL 跟踪能帮助您回答如下问题：

- SQL 语句耗费多少时间？
- 单个语句使用多少资源？
- 语句执行耗费多少时间？
- 等待每个资源耗费多少时间？

统计信息存储在循环缓冲区，该缓冲区为存储在 sysmaster 数据库中名为 syssqltrace 的内存伪表。您可以动态调整循环缓冲区的大小。

默认情况下，SQL 跟踪是关闭的，但是您可以对所有用户或一组特定用户打开此功能。当启用具有默认配置的 SQL 跟踪时，数据库服务器会跟踪最后运行的 1000 条 SQL 语句，以及这些语句的概要统计信息。您还可以全局禁用 SQL 跟踪或对特定用户禁用 SQL 跟踪。

如果您计划保存大量历史信息，那么 SQL 跟踪所需的内存相当大。SQL 跟踪所需的默认空间量为 2 MB。您可以根据需求增加或减少存储量。

显示的信息包括：

- 运行命令的用户的用户标识
- 数据库会话标识
- 数据库名称
- SQL 语句类型
- SQL 语句执行的持续时间
- 该语句完成的时间
- 具有该语句类型的 SQL 语句文本或函数调用列表（也称为堆栈跟踪），例如：

  ```
  procedure1() calls procedure2() calls procedure3()
  ```

- 统计信息包括：
  - 缓冲区读取和写入的次数
  - 页面读取和写入的数目

  - 排序和磁盘排序的数目
  - 锁请求和等待的次数
  - 逻辑日志记录数
  - 索引缓冲区读取的次数
  - 估计的行数
  - 优化器估计成本
  - 返回的行数
- 数据库隔离级别。

还可指定要包含在跟踪中的信息的升级级别，如下所示：

- 低级别跟踪，默认情况下已启用，用于捕捉以下示例中显示的信息。该信息包括语句统计信息、语句文本和语句迭代器。
- 中级别跟踪，捕捉包含在低级别跟踪中的所有信息外，还捕捉表名、数据库名和存储过程堆栈的列表。
- high-level tracing captures all of the information included in medium-level tracing, plus host variables.
- 高级别跟踪，捕捉包含在中级别跟踪中的所有信息外，还捕捉主机变量。

所跟踪的信息量影响该历史数据所需的内存量。

您可在任何时间启用和禁用跟踪，并可在数据库服务器运行时更改跟踪缓冲区的数量和大小。如果调整跟踪缓冲区的大小，那么数据库服务器将尝试维护缓冲区的内容。如果增大这些参数，数据不会被截断。但是，如果减小缓冲区的数量或大小，那么跟踪缓冲区中的数据可能会被截断或丢失。

缓冲区的数量决定所跟踪的 SQL 语句数。每个缓冲区包含单个 SQL 语句的信息。默认情况下，单个跟踪缓冲区大小是固定的。如果存储在缓冲区中的文本信息超过了跟踪缓冲区的大小，那么数据将被截断。

以下示例显示 SQL 跟踪信息：

```
select * from syssqltrace where sql_id = 5678;

sql_id            5678
sql_address       4489052648
sql_sid           55
sql_uid           2053
sql_stmttype      6
sql_stmtname      INSERT
sql_finishtime    1140477805
sql_begintxtime   1140477774
sql_runtime       30.86596333400
sql_pgreads       1285
sql_bfreads       19444
sql_rdcache       93.39127751491
sql_bfidxreads    5359
sql_pgwrites      810
sql_bfwrites      17046
sql_wrcache       95.24815205913
sql_lockreq       10603
sql_lockwaits     0
sql_lockwttime    0.00
sql_logspace      60400
sql_sorttotal     0
sql_sortdisk      0
sql_sortmem       0
sql_executions    1
sql_totaltime     30.86596333400
sql_avgtime       30.86596333400
sql_maxtime       30.86596333400
sql_numiowaits    2080
sql_avgiowaits    0.014054286131
sql_totaliowaits  29.23291515300
sql_rowspersec    169.8958799132
```

```
sql_estcost       102
sql_estrows       1376
sql_actualrows    5244
sql_sqlerror      0
sql_isamerror     0
sql_isollevel     2
sql_sqlmemory     32608
sql_numiterators  4
sql_database      db3
sql_numtables     3
sql_tablelist     t1
sql_statement     insert into t1 select {+ AVOID_FULL(sysindices) } 0, tabname
```

有关所有表行的解释，请参阅《*SinoDB*® 管理员参考》 的 sysmaster 数据库部分中有关 syssqltrace 表的信息。

相关链接

自动监视与更正操作概述 在第441页

使用 SQLTRACE 配置参数指定启动 SQL 跟踪信息

使用 SQLTRACE 配置参数可控制数据库服务器启动时的缺省跟踪行为。缺省情况下，不设置此参数。设置的信息包括要跟踪的 SQL 语句数和跟踪模式。

可以修改 onconfig 文件的任何用户均可以修改 SQLTRACE 配置参数的值，并可影响启动配置。但是，只有用户 informix、root 或具有连接sysadmin 数据库特权的 DBSA 可以使用 SQL 管理 API 命令修改 SQL 跟踪的运行时状态。

要在数据库服务器启动时指定 SQL 跟踪信息，请执行以下操作 ：

1. 在 onconfig 文件中设置 SQLTRACE 配置参数。
2. 重启数据库服务器。

示例

onconfig 文件中的以下设置指定数据库服务器收集有关系统上所有用户执行的最多 2000 条 SQL 语句的低级别信息，并分配大约 4 MB 内存（2000 * 2 KB）。

```
SQLTRACE level=LOW,ntraces=2000,size=2,mode=global
```

如果仅使用部分已分配的缓冲区空间（例如，缓冲区空间的 42% ），那么所分配的内存量仍然是 2 KB。

如果不想设置 SQLTRACE 配置参数并重新启动服务器，那么可以运行以下 SQL 管理 API 命令，其提供的功能与为当前会话设置 SQLTRACE 的功能相同 ：

```
EXECUTE FUNCTION task("set sql tracing on", 100,"1k","med","user");
```

在用户模式下启用 SQL 跟踪系统之后，可以启用对每个用户的跟踪。请参阅 启用 *SQL* 跟踪 在第464页。

有关使用 task() 和 admin() 函数的更多信息，请参阅《*SinoDB*® 管理员参考》。

有关 SQLTRACE 配置参数的更多信息（包括某些字段的最小值和最大值），请参阅《*SinoDB*® 管理员参考》。

全局禁用 SQL 跟踪或禁用对某个会话的 SQL 跟踪

即使 SQLTRACE 配置参数中指定的模式是 global 或 user，如果您想要完全关闭所有用户和全局的跟踪并取消 SQL 跟踪当前正在使用的资源，您可以禁用 SQL 跟踪。默认情况下，禁用对所有用户的 SQL 跟踪。

必须以用户 informix 或其他授权用户身份连接到 sysadmin 数据库。

要禁用全局 SQL 跟踪，请使用 set sql tracing off 参数运行 SQL 管理 API task() 或 admin() 函数。

要禁用特定会话的 SQL 跟踪，请以 set sql tracing off 为第一个参数和会话标识号为第二个参数来运行 SQL 管理 API task() 或 admin() 函数。

示例

以下示例禁用全局 SQL 跟踪：

```
EXECUTE FUNCTION task('set sql tracing off');
(expression) SQL tracing off.

1 row(s) retrieved.
```

以下示例对标识为 47 的会话禁用 SQL 跟踪：

```
EXECUTE FUNCTION task(“set sql user tracing off”,47);
```

有关使用 task() 或 admin() 函数的更多信息，请参阅《*SinoDB® SQL* 指南: 语法》。

启用 SQL 跟踪

在 SQLTRACE 配置参数指定模式为 user 之后，必须运行 SQL 管理 API task() 或 admin() 函数来开启特定用户的 SQL 历史记录跟踪。

必须以用户 informix 或其他授权用户身份连接到 sysadmin 数据库。

对特定用户进行 SQL 跟踪不需要启用全局 SQL 跟踪。

要启用特定用户的 SQL 跟踪，请以 set sql tracing on 为第一个参数和该用户会话标识为第二个参数来运行 SQL 管理 API task() 或 admin()函数。

要对 root 或 informix 以外所有用户启用用户 SQL 跟踪，请使用 set sql tracing on 参数和定义这些用户的信息来运行 task() 或 admin() 函数。

示例

以下示例对会话标识为 74 的用户启用 SQL 跟踪：

```
EXECUTE FUNCTION task("set sql user tracing on", 74);
```

以下示例对当前连接到系统的用户（非以用户 root 或 informix 身份登录）启用 SQL 语句跟踪。

```
dbaccess sysadmin -<<END
 execute function task("set sql tracing on", 1000, 1,"low","user");
 select task("set sql user tracing on", sid)
  FROM sysmaster:syssessions
  WHERE username not in ("root","informix");
END
```

有关 task() 或 admin() 函数的更多信息，请参阅《*SinoDB*® 管理员参考》。

启用某个会话的全局 SQL 跟踪

可以通过运行 SQL 管理 API 的 task() 或 admin() 函数来启用当前会话的全球 SQL 跟踪。

必须以用户 informix 或其他授权用户身份连接到 sysadmin 数据库。

默认情况下，不启用全局 SQL 跟踪。可以设置 SQLTRACE 配置参数以永久启用全局跟踪 。

要启用当前数据库服务器会话的全局用户 SQL 历史记录跟踪，请使用 set sql tracing on 参数运行 SQL 管理 API task() 或 admin() 函数。

示例

以下示例启用所有用户的全局地级别 SQL 跟踪：

```
EXECUTE FUNCTION task("set sql tracing on", 1000, 1,"low","global");
```

如果在该语句运行之后有新用户登录到系统，那么可以启用对该新用户的跟踪。请参阅启用 *SQL* 跟踪 在第464页。

有关 task() 或 admin() 函数的更多信息，请参阅《*SinoDB*® *SQL* 指南*:* 语法》。