**[MySQL MyISAM和InNodb备份与恢复技巧](http://blog.csdn.net/nightelve/article/details/17397063)**

**1. 为什么要备份数据库**

对数据库来说，最重要也最容易被忽视的就是备份。由于不可预测性，偶然的事件可能会导致非常惨重的损失。

数据越是重要，数据的变化越频繁，备份越发需要经常进行。

备份周期根据不同业务的需要可以调整，但是不能忽视备份。

备份时最好也备份my.cnf或my.ini，这样可以保存你以前的配置参数。

**2. MyISAM 表备份/恢复策略**

**2.1. 文件热备份**

**2.1.1. 拷贝文件**

因为MySQL表保存为文件方式，很容易备份。要想保持备份的一致性，对相关表执行LOCK TABLES操作，然后对表执行FLUSH TABLES。

你只需要读锁定；这样当你复制数据库目录中的文件时，允许其它客户继续查询表。需要FLUSH TABLES语句来确保开始备份前将所有激活的索引页写入硬盘。

标准流程：锁表-》刷新表到磁盘-》拷贝文件-》解锁。

**2.1.2. 使用sql 语句备份**

如果你想要进行SQL级别的表备份，你可以使用SELECT INTO ...OUTFILE或BACKUP TABLE。对于SELECT INTO ...OUTFILE， 输出的文件不能先存在。

对于BACKUP TABLE也如此，因为覆盖完整的文件会有安全风险。

这两种备份方法，如果输出文件有重名的话，最好把重名文件移除。

BACKUP TABLE备份时注意输出目录的权限，改方法只是备份MYD和frm文件，不备份索引。

**2.1.3. 使用mysqlhotcopy 备份**

mysqlhotcopy 是一个 Perl脚本，最初由Tim Bunce编写并提供。它使用LOCK TABLES、FLUSH TABLES和cp或scp来快速备份数据库。它是备份数据库或单个表的最快的途径，但它只能运行在数据库目录所在的机器上。mysqlhotcopy 只用于备份MyISAM。它运行在Unix和NetWare中。

shell> mysqlhotcopy db\_name [/path/to/new\_directory ]

shell> mysqlhotcopy db\_name\_1 ... db\_name\_n /path/to/new\_directory

备份给定数据库中的匹配正则表达式的表：

shell> mysqlhotcopy db\_name./regex/

加上发音符(‘~’)前缀，表名的正则表达式可以被否定：

shell> mysqlhotcopy db\_name./~regex/

mysqlhotcopy支持下面的选项：

·         ---help，-？

显示帮助消息并退出。

·         --allowold

如果目标存在不放弃(加上一个\_old后缀重新命名它)。

·         --checkpoint=db\_name.tbl\_name

在指定的db\_name.tbl\_name插入检查点条目。

·         ---debug

启用调试输出。

·         --dryrun，-n

报告动作而不执行它们。

·         --flushlog

所有表锁定后刷新日志。

·         --keepold

完成后不删除以前(重新命名的)的目标。

·         -- method=command

复制方法(cp或scp)。

·         --noindices

备份中不包括全部索引文件。这样使备份更小、更快。可以在以后用myisamchk -rq重新构建索引。

·         --password=password，-p password

当连接服务器时使用的密码。请注意该选项的密码值是不可选的，不象其它MySQL程序。

·         --port=port\_num，-P port\_num

当连接本地服务器时使用的TCP/IP端口号。

·         --quiet，-q

除了出现错误时保持沉默。

·         --regexp=expr

复制所有数据库名匹配给出的正则表达式的数据库。

·         --socket=path，-S path

用于连接的Unix套接字文件。

·         --suffix=str

所复制的数据库名的后缀。

·         --tmpdir=path

临时目录(代替/tmp)。

·         --user=user\_name，-u user\_name

当连接服务器时使用的MySQL用户名。

mysqlhotcopy从选项文件读取[client]和[mysqlhotcopy]选项组。

因为mysqlhotcopy一般是用来做完全备份，所以推荐使用—flushlog选项来产生增量更新日志。

**2.1.4. 使用mysqldump 备份**

可以备份表结构和数据，可以同时支持MyISAM和InnoDB引擎数据库。

mysqldump可以备份单个表、单个库或所有库。

Mysqldump 还可以只导出表结构。

mysqldump是逻辑备份，输出的是sql语句文件，还可以输出其他数据库兼容的格式。

有3种方式来调用mysqldump：

shell> mysqldump [options] db\_name [tables]

shell> mysqldump [options] ---database DB1 [DB2 DB3...]

shell> mysqldump [options] --all--database

如果没有指定任何表或使用了---database或--all--database选项，则转储整个数据库。

要想获得你的版本的mysqldump支持的选项，执行mysqldump ---help。

如果运行mysqldump没有--quick或--opt选项，mysqldump在转储结果前将整个结果集装入内存。如果转储大数据库可能会出现问题。该选项默认启用，但可以用--skip-opt禁用。

mysqldump支持下面的选项：

·         ---help，-？

显示帮助消息并退出。

·         --add-drop--database

在每个CREATE DATABASE语句前添加DROP DATABASE语句。

·         --add-drop-tables

在每个CREATE TABLE语句前添加DROP TABLE语句。

·         --add-locking

用LOCK TABLES和UNLOCK TABLES语句引用每个表转储。重载转储文件时插入得更快。

·         --all--database，-A

转储所有数据库中的所有表。与使用---database选项相同，在命令行中命名所有数据库。

·         --allow-keywords

允许创建关键字列名。应在每个列名前面加上表名前缀。

·         ---comments[={0|1}]

如果设置为 0，禁止转储文件中的其它信息，例如程序版本、服务器版本和主机。--skip—comments与---comments=0的结果相同。 默认值为1，即包括额外信息。

·         --compact

产生少量输出。该选项禁用注释并启用--skip-add-drop-tables、--no-set-names、--skip-disable-keys和--skip-add-locking选项。

·         --compatible=name

产生与其它数据库系统或旧的MySQL服务器更兼容的输出。值可以为ansi、mysql323、mysql40、postgresql、oracle、mssql、db2、maxdb、no\_key\_options、no\_tables\_options或者no\_field\_options。要使用几个值，用逗号将它们隔开。这些值与设置服务器SQL模式的相应选项有相同的含义。

该选项不能保证同其它服务器之间的兼容性。它只启用那些目前能够使转储输出更兼容的SQL模式值。例如，--compatible=oracle 不映射Oracle类型或使用Oracle注释语法的数据类型。

·         --complete-insert，-c

使用包括列名的完整的INSERT语句。

·         --compress，-C

压缩在客户端和服务器之间发送的所有信息（如果二者均支持压缩）。

·         --create-option

在CREATE TABLE语句中包括所有MySQL表选项。

·         ---database，-B

转储几个数据库。通常情况，mysqldump将命令行中的第1个名字参量看作数据库名，后面的名看作表名。使用该选项，它将所有名字参量看作数据库名。CREATE DATABASE IF NOT EXISTS db\_name和USE db\_name语句包含在每个新数据库前的输出中。

·         ---debug[=debug\_options]，-# [debug\_options]

写调试日志。debug\_options字符串通常为'd:t:o,file\_name'。

·         --default-character-set=charset

使用charsetas默认字符集。如果没有指定，mysqldump使用utf8。

·         --delayed-insert

使用INSERT DELAYED语句插入行。

·         --delete-master-logs

在主复制服务器上，完成转储操作后删除二进制日志。该选项自动启用--master-data。

·         --disable-keys，-K

对于每个表，用;和;语句引用INSERT语句。这样可以更快地装载转储文件，因为在插入所有行后创建索引。该选项只适合MyISAM表。

·         --extended-insert，-e

使用包括几个VALUES列表的多行INSERT语法。这样使转储文件更小，重载文件时可以加速插入。

·         --fields-terminated-by=...，--fields-enclosed-by=...，--fields-optionally-enclosed-by=...，--fields-escaped-by=...，--行-terminated-by=...

这些选项结合-T选项使用，与LOAD DATA INFILE的相应子句有相同的含义。

·         --first-slave，-x

不赞成使用，现在重新命名为--lock-all-tables。

·         --flush-logs，-F

开始转储前刷新MySQL服务器日志文件。该选项要求RELOAD权限。请注意如果结合--all--database(或-A)选项使用该选项，根据每个转储的数据库刷新日志。例外情况是当使用--lock-all-tables或--master-data的时候：在这种情况下，日志只刷新一次，在所有 表被锁定后刷新。如果你想要同时转储和刷新日志，应使用--flush-logs连同--lock-all-tables或--master-data。

·         --force，-f

在表转储过程中，即使出现SQL错误也继续。

·         --host=host\_name，-h host\_name

从给定主机的MySQL服务器转储数据。默认主机是localhost。

·         --hex-blob

使用十六进制符号转储二进制字符串列(例如，'abc' 变为0x616263)。影响到的列有BINARY、VARBINARY、BLOB。

·         --lock-all-tables，-x

所有数据库中的所有表加锁。在整体转储过程中通过全局读锁定来实现。该选项自动关闭--single-transaction和--lock-tables。

·         --lock-tables，-l

开始转储前锁定所有表。用READ LOCAL锁定表以允许并行插入MyISAM表。对于事务表例如InnoDB和BDB，--single-transaction是一个更好的选项，因为它不根本需要锁定表。

请注意当转储多个数据库时，--lock-tables分别为每个数据库锁定表。因此，该选项不能保证转储文件中的表在数据库之间的逻辑一致性。不同数据库表的转储状态可以完全不同。

·         --master-data[=value]

该选项将二进制日志的位置和文件名写入到输出中。该选项要求有RELOAD权限，并且必须启用二进制日志。如果该选项值等于1，位置和文件名被写入CHANGE MASTER语句形式的转储输出，如果你使用该SQL转储主服务器以设置从服务器，从服务器从主服务器二进制日志的正确位置开始。如果选项值等于2，CHANGE MASTER语句被写成SQL注释。如果value被省略，这是默认动作。

--master-data选项启用--lock-all-tables，除非还指定--single-transaction(在这种情况下，只在刚开始转储时短时间获得全局读锁定。又见--single-transaction。在任何一种情况下，日志相关动作发生在转储时。该选项自动关闭--lock-tables。

·         --no-create-db，-n

该选项禁用CREATE DATABASE db\_name语句，如果给出---database或--all--database选项，则包含到输出中。

·         --no-create-info，-t

不写重新创建每个转储表的CREATE TABLE语句。

·         --no-data，-d

不写表的任何行信息。如果你只想转储表的结构这很有用。

·         --opt

该选项是速记；等同于指定 --add-drop-tables--add-locking --create-option --disable-keys--extended-insert --lock-tables --quick --set-charset。它可以给出很快的转储操作并产生一个可以很快装入MySQL服务器的转储文件。该选项默认开启，但可以用--skip-opt禁用。要想只禁用确信用-opt启用的选项，使用--skip形式；例如，--skip-add-drop-tables或--skip-quick。

·         --password[=password]，-p[password]

连接服务器时使用的密码。如果你使用短选项形式(-p)，不能在选项和密码之间有一个空格。如果在命令行中，忽略了--password或-p选项后面的 密码值，将提示你输入一个。

·         --port=port\_num，-P port\_num

用于连接的TCP/IP端口号。

·         --protocol={TCP | SOCKET | PIPE | MEMORY}

使用的连接协议。

·         --quick，-q

该选项用于转储大的表。它强制mysqldump从服务器一次一行地检索表中的行而不是检索所有行并在输出前将它缓存到内存中。

·         --quote-names，-Q

用‘`’字符引用数据库、表和列名。如果服务器SQL模式包括ANSI\_QUOTES选项，用‘"’字符引用名。默认启用该选项。可以用--skip-quote-names禁用，但该选项应跟在其它选项后面，例如可以启用--quote-names的--compatible。

·         --result-file=file，-r file

将输出转向给定的文件。该选项应用在Windows中，因为它禁止将新行‘/n’字符转换为‘/r/n’回车、返回/新行序列。

·         --routines，-R

在转储的数据库中转储存储程序(函数和程序)。使用---routines产生的输出包含CREATE PROCEDURE和CREATE FUNCTION语句以重新创建子程序。但是，这些语句不包括属性，例如子程序定义者或创建和修改时间戳。这说明当重载子程序时，对它们进行创建时定义者应设置为重载用户，时间戳等于重载时间。

如果你需要创建的子程序使用原来的定义者和时间戳属性，不使用--routines。相反，使用一个具有mysql数据库相应权限的MySQL账户直接转储和重载mysql.proc表的内容。

该选项在MySQL 5.1.2中添加进来。在此之前，存储程序不转储。

·         --set-charset

将SET NAMES default\_character\_set加到输出中。该选项默认启用。要想禁用SET NAMES语句，使用--skip-set-charset。

·         --single-transaction

该选项从服务器转储数据之前发出一个BEGIN SQL语句。它只适用于事务表，例如InnoDB和BDB，因为然后它将在发出BEGIN而没有阻塞任何应用程序时转储一致的数据库状态。

当使用该选项时，应记住只有InnoDB表能以一致的状态被转储。例如，使用该选项时任何转储的MyISAM或HEAP表仍然可以更改状态。

--single-transaction选项和--lock-tables选项是互斥的，因为LOCK TABLES会使任何挂起的事务隐含提交。

要想转储大的表，应结合--quick使用该选项。

·         --socket=path，-S path

当连接localhost(为默认主机)时使用的套接字文件。

·         --skip--comments

参见---comments选项的描述。

·         --tab=path，-T path

产生tab分割的数据文件。对于每个转储的表，mysqldump创建一个包含创建表的CREATE TABLE语句的tbl\_name.sql文件，和一个包含其数据的tbl\_name.txt文件。选项值为写入文件的目录。

默认情况，.txt数据文件的格式是在列值和每行后面的新行之间使用tab字符。可以使用--fields-xxx和--行--xxx选项明显指定格式。

注释：该选项只适用于mysqldump与mysqld服务器在同一台机器上运行时。你必须具有FILE权限，并且服务器必须有在你指定的目录中有写文件的许可。

·         --tables

覆盖---database或-B选项。选项后面的所有参量被看作表名。

·         --triggers

为每个转储的表转储触发器。该选项默认启用；用--skip-triggers禁用它。

·         --tz-utc

在转储文件中加入SET TIME\_ZONE='+00:00'以便TIMESTAMP列可以在具有不同时区的服务器之间转储和重载。(不使用该选项，TIMESTAMP列在具有本地时区的源服务器和目的服务器之间转储和重载）。--tz-utc也可以保护由于夏令时带来的更改。--tz-utc默认启用。要想禁用它，使用--skip-tz-utc。该选项在MySQL 5.1.2中加入。

·         --user=user\_name，-u user\_name

连接服务器时使用的MySQL用户名。

·         --verbose，-v

冗长模式。打印出程序操作的详细信息。

·         --version，-V

显示版本信息并退出。

·         --where='where-condition', -w 'where-condition'

只转储给定的WHERE条件选择的记录。请注意如果条件包含命令解释符专用空格或字符，一定要将条件引用起来。

**2.2. 文件冷备份**

当mysql服务器停止时，通过复制所有表文件(\*.frm、\*.MYD和\*.MYI文件)来备份MyISAM数据库。

**2.3. MyISAM 表恢复**

1、 如果是通过mysqldump备份的，就执行：mysql –u root < 备份文件名。

2、 如果通过mysqlhotcopy或文件冷/热拷贝来备份的，停止mysql服务，使用备份文件来覆盖现有文件。

3、 如果是采用BACKUP TABLE备份的，使用restore table来恢复。因为backup table不会备份索引文件，恢复表注意重建索引。

4、 如果是采用SELECT INTO ...OUTFILE备份的，使用load data恢复数据，也可以使用mysqlimport命令来代替。

**3. 日志备份**

**3.1. 日志备份的好处及如何启用**

日志备份可以支持MyISAM和InnoDB，这跟有些备份工具只支持MyISAM不一样。

另外，日志可以做增量备份，这是其他方法无法做到的。

启动日志备份：用--log-bin[=file\_name]选项来启动mysql服务。

**3.2. 增量备份**

先执行FLUSH LOGS刷新日志，同步日志缓存到磁盘，关闭当前的日志并产生新的日志文件。

拷贝上一次完整备份或增量备份后的一个或一些日志文件到一个安全的地方。

这样就制作了一个增量备份。

**3.3. 日志恢复**

日志恢复可以选择恢复某个数据库、某些操作点或时间范围，非常灵活。

通过mysqlbinlog工具可以恢复二进制日志，可以一次恢复多个日志文件，命令如下：

shell> mysqlbinlog [options] hostname-bin.[0-9]\* | mysql

mysqlbinlog命令选项

·         ---help，-？

显示帮助消息并退出。

·         ---database=db\_name，-d db\_name

只列出该数据库的条目(只用本地日志)。

·         --force-read，-f

使用该选项，如果mysqlbinlog读它不能识别的二进制日志事件，它会打印警告，忽略该事件并继续。没有该选项，如果mysqlbinlog读到此类事件则停止。

·         --hexdump，-H

在注释中显示日志的十六进制转储。该输出可以帮助复制过程中的调试。在MySQL 5.1.2中添加了该选项。

·         --host=host\_name，-h host\_name

获取给定主机上的MySQL服务器的二进制日志。

·         --local-load=path，-l pat

为指定目录中的LOAD DATA INFILE预处理本地临时文件。

·         --offset=N，-o N

跳过前N个条目。

·         --password[=password]，-p[password]

当连接服务器时使用的密码。如果使用短选项形式(-p)，选项和 密码之间不能有空格。如果在命令行中--password或-p选项后面没有 密码值，则提示输入一个密码。

·         --port=port\_num，-P port\_num

用于连接远程服务器的TCP/IP端口号。

·         --position=N，-j N

不赞成使用，应使用--start-position。

·         --protocol={TCP | SOCKET | PIPE | -position

使用的连接协议。

·         --read-from-remote-server，-R

从MySQL服务器读二进制日志。如果未给出该选项，任何连接参数选项将被忽略。这些选项是--host、--password、--port、--protocol、--socket和--user。

·         --result-file=name, -r name

将输出指向给定的文件。

·         --short-form，-s

只显示日志中包含的语句，不显示其它信息。

·         --socket=path，-S path

用于连接的套接字文件。

·         --start-datetime=datetime

从二进制日志中第1个日期时间等于或晚于datetime参量的事件开始读取。datetime值相对于运行mysqlbinlog的机器上的本地时区。该值格式应符合DATETIME或TIMESTAMP数据类型。例如：

shell> mysqlbinlog --start-datetime="2004-12-25 11:25:56" binlog.000003

该选项可以帮助点对点恢复。

·         --stop-datetime=datetime

从二进制日志中第1个日期时间等于或晚于datetime参量的事件起停止读。关于datetime值的描述参见--start-datetime选项。该选项可以帮助及时恢复。

·         --start-position=N

从二进制日志中第1个位置等于N参量时的事件开始读。

·         --stop-position=N

从二进制日志中第1个位置等于和大于N参量时的事件起停止读。

·         --to-last-logs，-t

在MySQL服务器中请求的二进制日志的结尾处不停止，而是继续打印直到最后一个二进制日志的结尾。如果将输出发送给同一台MySQL服务器，会导致无限循环。该选项要求--read-from-remote-server。

·         --disable-logs-bin，-D

禁用二进制日志。如果使用--to-last-logs选项将输出发送给同一台MySQL服务器，可以避免无限循环。该选项在崩溃恢复时也很有用，可以避免复制已经记录的语句。注释：该选项要求有SUPER权限。

·         --user=user\_name，-u user\_name

连接远程服务器时使用的MySQL用户名。

·         --version，-V

显示版本信息并退出。

还可以使用--var\_name=value选项设置下面的变量：

·         open\_files\_limit

指定要保留的打开的文件描述符的数量。

**4. InnoDB 表备份/恢复策略**

**4.1. 使用商业的在线热备份工具**

InnoDB Hotbackup 是一个在线备份工具，你可以用它来在InnoDB数据库运行之时备份你的InnoDB数据库。InnoDB 热备份工具 不要求你关闭数据库，并且它不设置任何锁定或干扰你正常的数据库处理。InnoDB 热备份工具 是非免费（商业的）附加软件，它每年的证书费用是每台MySQL服务器运行的计算机390欧元。

具体使用方法可以参考相关文档，我也没有使用过这个工具。

**4.2. 使用mysqldump 热备份**

Mysqldump提供对InnoDB非物理的在线逻辑热备份。

使用方法与备份MyISAM时一样。

**4.3. 使用select into 热备份单个/多个表**

可以使用select into备份一个或多个表，用法与MyISAM表相同。

**4.4. 二进制冷备份**

如果你可以关闭你的MySQL服务器，你可以生成一个包含InnoDB用来管理它的表的所有文件的二进制备份。使用如下步骤：

1.    关闭MySQL服务器，确信它是无错误关闭。

2.  复制你所有数据文件（ibdata文件和.ibd文件）到一个安全的地方。

3.   复制你所有ib\_logfile文件到一个安全的地方。

4.    复制my.cnf配置文件或文件到一个安全的地方。

5.    为你InnoDB表复制.frm文件到一个安全的地方。

**4.5. InnoDB 表恢复**

1、 先尝试使用InnoDB的日志自动恢复功能，方法是重启mysql服务。

2、 在一些情况下，明显地数据库损坏是因为操作系统损坏它自己的文件缓存，磁盘上的数据可能完好，最好是首先重启计算机。它可以消除那些显得是数据库页损坏的错误。

如果不行，则采用下面的方法进行恢复。

3、 如果是mysqldump做的完全备份，先恢复完全备份，然后再恢复完全备份后的增量日志备份。

4、 如果是采用select into备份表的话，则采用load data或mysqlimport恢复。

5、 如果是采用二进制冷备份做的完全备份，则先停止mysql服务，覆盖备份的二进制文件，然后执行上次完全备份后的增量日志备份。

6、 注意：InnoDB二进制文件没有MyISAM那么好，必须在相同的浮点数的cpu机器上移植。

**5. MyISAM/InnoDB 备份/恢复策略**

因为在一个数据库中有可能要同时使用MyISAM和InnoDB两种引擎，因此统一考虑他们的备份/恢复策略。

1、 完整备份采用mysqldump。

2、 增量备份采用bin-log日志。

3、 单表备份采用select into。

4、

**6. 定期维护表**

虽然MySQL提供了多种备份/恢复手段，但是定期维护表大大可以降低表毁坏的可能性，并且可以提高查询性能。

1、对于InnoDB引擎，你可以使用innodb\_tablespace\_monitor来检查表空间文件内文件空间管理的完整性。

2、对于MyISAM引擎，可以通过sql语句或myisamchk 工具来维护表 。

在许多情况下，你会发现使用SQL语句实现MyISAM表的维护比执行myisamchk操作要容易地多：

·         要想检查或维护MyISAM表，使用CHECK TABLE或REPAIR TABLE。

·         要想优化MyISAM表，使用OPTIMIZE TABLE。

·         要想分析MyISAM表，使用ANALYZE TABLE。

这些语句比myisamchk有利的地方是服务器可以做任何工作。使用myisamchk，你必须确保服务器在同一时间不使用表。否则，myisamchk和服务器之间会出现不期望的相互干涉。

一般建议在停止mysql服务时执行myisamchk，如果是在线执行则最好先flush tables（把所有更新写入磁盘）。

**7. 备份策略摘要**

1、一定用--log-bin或甚至--log-bin=log\_name选项运行MySQL服务器，其中日志文件名位于某个安全媒介上，不同于数据目录所在驱动器。如果你有这样的安全媒介，最好进行硬盘负载均衡(这样能够提高性能)。

2、定期进行完全备份，使用mysqldump命令进行在线非块备份。

在负载比较低的时候进行，并且建议采用--single-transaction参数来保证事务数据的一致性，同时不影响其他用户的正常读写。

3、完整备份时采用FLUSH LOGS，便于产生增量备份日志。

4、用FLUSH LOGS或mysqladmin flush-logs 刷新日志进行定期增量备份。

5、定期维护表，这样既可以提高性能，并且可以减少数据丢失和出错的可能性。

6、关键表可以另外再做备份，根据需要。