MySQL Backup, Recovery & Partition

Physical Backup versus Logical Backup

- 物理备份
  - 物理备份由存储数据库内容的目录和文件的原始副本组成。
  - 这种类型的备份适用于需要在出现问题时快速恢复的大型重要数据库。
- 逻辑备份
  - 逻辑备份保存以逻辑数据库结构（CREATE DATABASE、CREATE TABLE 语句）和内容（INSERT 语句或分隔文本文件）表示的信息。
  - 这种类型的备份适用于较小的数据量，您可以在其中编辑数据值或表结构，或者在不同的机器架构上重新创建数据
- 物理备份方法具有以下特点
  - 备份由数据库目录和文件的精确副本组成。 通常，这是全部或部分 MySQL 数据目录的副本。
  - 物理备份方法比逻辑备份更快，因为它们只涉及文件复制而无需转换。
  - 输出比逻辑备份更紧凑。
  - 由于备份速度和紧凑性对于繁忙的重要数据库很重要，因此 MySQL Enterprise Backup 产品执行物理备份。
  - 备份和恢复粒度范围从整个数据目录级别到单个文件级别。 这可能会或可能不会提供表级粒度，具体取决于存储引擎。
    - 例如，InnoDB 表可以每个都在一个单独的文件中，或者与其他 InnoDB 表共享文件存储； （即表不是最小单位）
    - 每个 MyISAM 表都唯一对应于一组文件。
  - 除了数据库之外，备份还可以包括任何相关文件，例如日志或配置文件。
  - 内存中的表数据很难以这种方式备份，因为内容并没有被存储到硬盘上。
  - **备份仅可移植到具有相同或相似硬件特性的其他机器上。**
  - 可以在 MySQL 服务器未运行时执行备份。
    - 如果服务器正在运行，则需要执行适当的锁定，以便服务器在备份期间不会更改数据库内容。
    - MySQL Enterprise Backup 会自动为需要锁定的表执行此锁定。
  - 物理备份工具包括 MySQL Enterprise Backup for InnoDB 或任何其他表的 mysqlbackup，或 MyISAM 表的文件系统级命令（如 cp、scp、tar、rsync）。
- 逻辑备份方法有以下特点
  - 备份是通过查询 MySQL 服务器获取数据库结构和内容信息来完成的。
  - 备份比物理方法慢，因为服务器必须访问数据库信息并将其转换为逻辑格式。
  - 输出比物理备份大，尤其是在以文本格式保存时。
  - 备份和恢复粒度在服务器级别（所有数据库）、数据库级别（特定数据库中的所有表）或表级别可用。
  - 备份不包括日志或配置文件，或不属于数据库的其他与数据库相关的文件。
  - 以逻辑格式存储的备份独立于机器且高度便携。
  - 在运行 MySQL 服务器的情况下执行逻辑备份。(Online)
  - 逻辑备份工具包括 **mysqldump** 程序和 SELECT ... INTO OUTFILE 语句。 这些适用于任何存储引擎，甚至 MEMORY。

    - 要恢复逻辑备份，可以使用 mysql 客户端处理 SQL 格式的转储文件。 要加载分隔文本文件，请使用 **LOAD DATA** 语句或 **mysqlimport** 客户端。

Online Backup versus Offline Backup

- 离线备份的特点
    - 客户端可能会受到不利影响，因为备份期间服务器不可用。 因此，此类备份通常取自可以脱机而不会损害可用性的副本。
    - 备份过程更简单，因为客户端活动不可能干扰。
    - 在线和离线之间的类似区别适用于恢复操作，并且适用类似的特征。
        - 但是，与在线备份相比，客户端更可能受到在线恢复的影响，因为恢复需要更强的锁定。
        - 在备份期间，客户端可能能够读取正在备份的数据。 恢复修改数据而不仅仅是读取数据，因此必须防止客户端在恢复数据时访问数据

Local Backup versus Remote Backup

- 本地备份是在运行 MySQL 服务器的同一台主机上执行的，而远程备份是从不同的主机完成的。 对于某些类型的备份，即使输出是本地写入服务器主机上的，也可以从远程主机启动备份。
- mysqldump 可以连接到本地或远程服务器。 – 对于 SQL 输出（CREATE 和 INSERT 语句），可以完成本地或远程转储并在客户端生成输出。 – 对于分隔文本输出（使用 --tab 选项），在服务器主机上创建数据文件。
- SELECT ... INTO OUTFILE 可以从本地或远程客户端主机启动，但输出文件是在服务器主机上创建的。
- 物理备份方法通常在 MySQL 服务器主机上本地启动，以便服务器可以脱机，尽管复制文件的目的地可能是远程的

Snapshot Backup

获取当前数据库的快照，利用Copy-on-write机制，只有当数据改变的时候，才将改变前的数据存储到一个新的数据块。这样，没有被修改的空间就可以共享。

Full Backup versus Incremental Backup

应该将全量备份和增量备份两者结合。每隔一段时间（一般而言相对长一点，而且可以挑在服务器负载比较低的时间段进行，假设每隔6小时一次）就在另一台服务器上进行全量备份，将数据库的所有内容进行备份。增量备份将数据库变化写入bin-log文件（bin-log文件也应该位于另一台服务器上，或者存储在不同的物理设备上）。当数据库服务器挂掉的时候，应该先通过全量备份恢复到某个时间段数据库的状态，然后删除该时间段之前的所有bin-log文件（因为已经用了全量备份，该时间段之前的增量备份就没有必要存储了），接着再用该时间段之后的所有增量备份对数据进行恢复。

MySQL dump SQL format/ Delimited-Text format

SQL format:

- 不是纯数据，包含很多额外的内容（很多insert...）；
- 但是易于迁移到支持标准SQL语句的数据库，比如Oracle。

Delimited-Text format:

- 会生成.sql和.txt文件，前者包含建表和数据库的SQL语句，后者包含数据；
- 不易于迁移，需要mysqlimport读取txt中的文件。

MySQL dump也可以用于检测MySQL的兼容性。

Partition

- 水平切割（无法分割字段）
- 分区在很多情况下能够加速增删改查（可以在SQL语句中指定分区）。
- RANGE分区：基于一个给定连续区间范围，把数据分配到不同的分区；
- LIST分区：类似RANGE分区，区别在LIST分区是基于枚举出的值列表分区，RANGE是基于给定连续区间范围分区；
- HASH分区：基于用户定义的表达式返回值来选择分区，该表达式对要插入到表的行中列值操作；
- KEY分区：类似HASH，但是HASH允许使用用户自定义表达式，而KEY分区不允许，它需要使用MySQL服务器提供的HASH函数，同时HASH分区只支持整数分区，而KEY分区支持除BLOB和TEXT类型外其他列；
- 都用LESS THAN写，删除分区不会影响存储。

More Info

- 如果有大量数据需要处理而不需要呈现数据，（比如统计成绩，只要最终平均成绩，不需要所有的成绩），可以在数据源用脚本执行。
- Logrotate 日志切割