应用系统体系架构 — 作业7

# 学号： 522031910213 姓名： 朱涵 得分：

A.请你详细叙述物理备份和逻辑备份各自的优缺点。

答：

物理备份是指直接复制数据库的数据文件、日志文件等内容来进行备份的方式。

优点：

1.备份和恢复速度非常快，因为是直接复制，且操作很简单，不需要额外的管理操作；

2.不需要数据库处于运行状态也可以备份，因为是对于磁盘文件的复制。

缺点：

1. 平台依赖性强，通常需要同一数据库的同一版本才能进行恢复；

2. 备份文件较大，包含所有数据和日志文件；

3.恢复粒度较大，无法针对单个表或行进行恢复，必须全部恢复。

逻辑备份则是指导出像SQL语句这样的操作逻辑来进行备份的方式。

优点：

1. 平台无关性强，可以适用于不同版本甚至不同的数据库，只要支持SQL语句即可；

2. 备份文件较小，只包含数据和结构的SQL文件；

3.灵活性强，可以恢复单个表或行，也可以只创建数据库的结构

缺点：

1. 备份和恢复速度较慢，因为需要解析SQL语句一步步执行，对于大型数据库比较繁琐；

2. 需要数据库处于启动状态才能进行。

B.请你参照上课的举例，详细描述如何通过全量备份和增量备份来实现系统状态的恢复。

答：

1. 开启全量备份和增量备份的支持：我们使用mysqldump进行定期的全量备份，使用MySQL的binlog功能实现增量备份，具体而言，定期（比如每周一的凌晨）使用mysqldump --all-databases --master-data --single-transaction > ebook\_backup\_{date}.sql 指令来进行全量备份，把数据导出到sql文件中，其中--all-databases参数会备份包括系统数据库在内的所有数据库，--master-data会设置主从复制恢复的相关设置，--single-transaction则会在不锁读表的情况下进行备份；在启动mysql时加上--log-bin参数即可启动binlog功能，语句会写入binlog并定期截断为多个日志文件，当进行全量备份时，可以加上--flush-logs参数来强制把当前的binlog截断并落盘，加上--delete-master-logs参数来删除当前主数据库所有的binlog，以此节省磁盘空间。
2. 当ebook遇到了导致数据库损坏的情况，可以通过以下方式恢复数据：

mysql < ebook\_backup\_{date}.sql 指令会通过全量备份将数据库恢复到指定日期的状态，在这里应该是上一个周一的凌晨。

接下来找到binlog的存储位置，可能会有形如gbichot2-bin.000001的日志文件。如果我们提前通过--log-bin参数设置好binlog的存储位置，使其和数据库文件并不在一个磁盘的话，此时日志文件应该不会遭到损坏。假设从上一次全量备份到现在产生了001到003三个binlog文件，那么使用形如mysqlbinlog gbichot2-bin.000001 gbichot2-bin.000002 gbichot2-bin.000003 | mysql的指令将会在一个线程内恢复这三个日志文件的内容，使得数据库恢复到了最新的状态。注意不能同时启动三个线程分别恢复，可能会因为语句之间的依赖关系导致报错。

至此，数据库通过了全量和增量备份实现了系统状态的恢复。

C.请你按照你的理解，阐述Partition机制有什么好处？

答：

1. 扩展数据量。数据分区可以分布式的存放在不同磁盘或设备上，但是却可以被视为一张数据表，这样的做法打破了单个设备的容量限制。
2. 方便添加或删除批量数据。如果需要添加或删除的批数据仅仅存储在一个分区（这种情况在时间分片下非常常见），那么就可以通过DROP TABLE的方式删除或添加分区，而不用DELETE来一条条进行删除，性能将会大大提升。
3. 查询优化。MySQL在执行query plan时会检查WHERE条件是否可以将查询的分区进行裁剪，从而避免访问不必要的分区，显著提高了查询等语句的性能。当然，用户也可以显式指定分区来优化查询，这是同理的
4. 提高并发性能。原本的插入和查询操作只能在一张表上进行，可能会有并发瓶颈。而分区则可以让操作在不同分表上进行，可以提高并发。
5. 提供磁盘利用率。单一数据表过于庞大，可能会导致磁盘碎片化，而分区后的数据文件可能可以在磁盘上分散分布，提高资源利用率。

D.如果数据文件在一台机器上有足够的存储空间存储，是否还需要进行Partition？为什么？

答： 根据实际的需求来判断。

如果数据量很大，用户的查询又经常落在某一个字段的连续范围内或是离散值上，比如经常需要查询某一年的数据，这时可以将数据表按照时间字段分区来优化查询的性能。同理的，有其他大量的查询需求时，也可以考虑什么样的分区可以带来性能的提升。

如果数据具有时效性，并且需要频繁插入，比如天气数据，需要时刻更新，并且应该经常把过期的历史数据归档到其他地方，那么采用时间分片可以让归档和更新操作以分区的形式进行，避免大量的批数据迁移删除与插入。

如果系统对于并发性有较高要求，比如需要支持千万级用户的同时写入，那么分区也可以适当的提高并发性，因为原本一张表的操作被分散到多个分区。

如果系统有分布式的需求，需要把数据分散部署，也可以考虑数据分区，把不同的分区分布到不同设备，但是仍可以作为一张数据表来操作。

综上，当数据分区可以解决系统的需求时，就可以使用，并非只有数据存储空间不足时才能使用partitioning。