HW 10

A

物理备份

- 优点
  - 备份到文件级别，可以把数据库中的所有文件都进行备份，包括数据库的数据和索引等，所占用的空间较小，当数据库的数据很大时，**恢复速度较快**
- 缺点
  - 无法进行**数据迁移**，如想要将数据从 MySQL 数据库迁移到 Oracle 时，物理备份是不行的
  - 备份时数据库还会存在数据写入的情况，一定程度上会造成数据丢失的可能性
  - 不总是可以跨平台、操作系统和 MySQL 版本

逻辑备份

- 优点
  - 备份粒度小，把数据库里的数据以 SQL 语句的方式导出成文件的形式，简单，易操作，自带工具方便、可靠
  - 与存储引擎无关，可以很容易地跨平台、跨操作系统、跨数据库版本
- 缺点
  - 将数据库的内容转化成脚本，文件大，恢复时候执行**速度较慢**

B

每隔一段时间，对数据库做一次全量备份，对整个数据库的文件进行备份，在下一次全量备份前，每隔一段时间对数据库的修改部分做增量备份。

当数据库系统崩溃后，系统恢复到最近的一次全量备份时的状态，之后再根据这次全量备份之后的增量备份内容恢复到之后的备份状态。

这样通过全量备份和增量备份就可以恢复到任意的时间点了。需要注意的是，无论是全量还是增量备份，都需要存到和这个数据库不同的文件系统上 ， 否则可能一起 crash, 备份就失去意义了。

C

1. 可伸缩性
   - 数据库表太大时，单个文件或者硬盘放不下，文件一次性读进来占用很多时间和内存。通过将数据分区分在不同磁盘，可以解决单磁盘容量瓶颈问题，存储更多的数据，也能解决单磁盘的 IO 瓶颈问题。
2. 提高性能
   - 分区可以减少数据库检索时需要遍历的数据量，在查询时只需要在数据对应的分区进行查询。例如对于聚合函数，例如 sum() 和 count()，可以在每个分区进行并行处理，最终只需要统计所有分区得到的结果即可，从而提升了性能。
3. 方便对数据进行运维管理
   - 对于失去保存意义的数据，通过删除对应的分区，达到快速删除的作用
   - 在某些场景下，单个分区表的备份恢复会更有效率

D

取决于数据库的数据量大小，当数据量很大时 Partion 仍然是需要的，因为在 C 中提到的（2）和（3）仍然起作用。

例如当我们对数据的读写只涉及某个或部分 partion 时，则只需要对这个别的 partion 进行读写即可，减小了数据的读写开销，从而可以提升性能。

但是当数据量较小时，这种性能上的提升很小，Partion 反而增大了不必要的额外开销，这种情况则是不需要的。