2023-11-11

应用系统体系架构 — 作业7

学号：521030990006

姓名：VAHAGN GHAZARYAN

## **Please describe in detail the advantages and disadvantages of physical backup and logical backup.**

## **Physical Backup**

## **Pros:**

## **Fast**: Physical backups usually save time, especially for big databases, as they copy files and folders directly.

## **Complete**: They grab the whole database, including important system details, crucial for getting data back.

## **Easy Restoration**: Putting back data from a physical backup is generally simple because it just involves putting back the database files, great for urgent situations.

## **Cons**:

## **Needs More Space**: These backups can take up a lot of room since they copy the entire database, including unused space.

## **Less Flexible**: You can't easily change or pick out parts of the data with these backups.

## **Logical Backup**

## **Pros**:

## **Works on Different Systems**: Logical backups can be used on various systems. They save the database's structure and data in a way that can be set up on different machines.

## **Allows Data Changes**: You can change data or how the database is set up before putting it back, giving you more control.

## **Choose What to Backup**: You can select specific parts of the database to save or bring back, which is good for smaller data sets.

## **Cons**:

## **Slower**: They usually take longer, especially for big databases, because they involve exporting and importing data, which isn't quick.

## **More Complex Restoration**: Putting data back can be trickier, especially if there are certain rules or links within the database that need careful handling.

## **Please refer to the example in class to describe in detail how to restore the system state through full backup and incremental backup .**

## ***Full Backup Restoration:***

## First, on Sunday, I take a complete backup of everything in the system. This is like making a copy of every single file and piece of data. When I need to restore, I use this full backup to get the system back to how it was on Sunday.

## For this, I use a command in the terminal like **mysql < backup\_sunday.sql**. This command puts everything from the backup into the system.

## ***Incremental Backup Restoration:***

## After the full backup, every day, I take a smaller backup that only saves what changed since Sunday. These are called incremental backups. Let's say the system has a problem on Wednesday. I would have incremental backups for Monday and Tuesday.

## To get the system up to date, I first apply Monday's backup with a command like **mysqlbinlog monday\_backup | mysql.** This adds Monday's changes.

## Then, I do the same for Tuesday's backup with a similar command.

## This process brings the system up to how it was on Tuesday night, just before the problem happened on Wednesday.

## So, I start with the full backup to get a base, then add the small changes from each day. This way, the system is restored to the latest point possible before any trouble occurred.

## **Could you please explain the benefits of the Partition mechanism according to your understanding?**

## The main advantage is of course partitioning makes it possible to store more data in one table than can be held on a single disk or file system partition. Besides that it -

## **Makes Things Faster**

## I find that partitioning helps speed up searching and managing data. It's like having a big book split into smaller chapters, so it's easier and quicker to find what I need.

## **Organizes Data Better**

## When I partition data, it's like putting similar things together in neat boxes. This makes it easier for me to handle and understand my data.

## **Improves Performance**

## Since each partition is smaller, doing tasks like adding or changing data becomes faster. It's like working on a small puzzle piece instead of the whole puzzle.

## **Maintenance Gets Easier**

## With partitioning, I only have to focus on a small part of the data at a time for things like backups or fixing issues. This is much less overwhelming than dealing with the entire database.

## **Limits Problems**

## If something goes wrong in one partition, it doesn't mess up everything. This is great because it means less stress if there’s a small problem.

## **If the data file has enough storage space on one machine, is it still necessary to partition it? Why?**

## **Makes Data Handling Faster**

## Like I said in question C, partitioning is like breaking a big book into chapters. Even if I have space for the whole book, working with chapters is quicker and easier. This means I can access and use my data faster.

## **Keeps Things Organized**

## Partitioning helps me keep my data sorted in a way that makes sense. For example, I might divide sales data by year or customer type. This organization makes it easier for me to find and work with specific data.

## **Easier to Maintain**

## Doing things like backing up or updating data is easier with partitions. I only need to focus on a small part at a time, not the whole database. It's less work and takes less time.

## **Faster Searches and Queries**

## When I’m looking for specific information, it’s quicker to search through a smaller, relevant partition than the entire database. This saves time, especially when dealing with large amounts of data.

## **Less Risk During Problems**

## If there's an issue in one partition, it doesn't affect the whole database. It’s easier to fix problems in smaller sections, and the rest of my data stays safe.

## So, even with plenty of space on my machine, partitioning is still a good idea. It helps me manage my data more effectively and makes many tasks quicker and easier.