INFO20003 Database Systems

Xiuge Chen

Tutorial 10

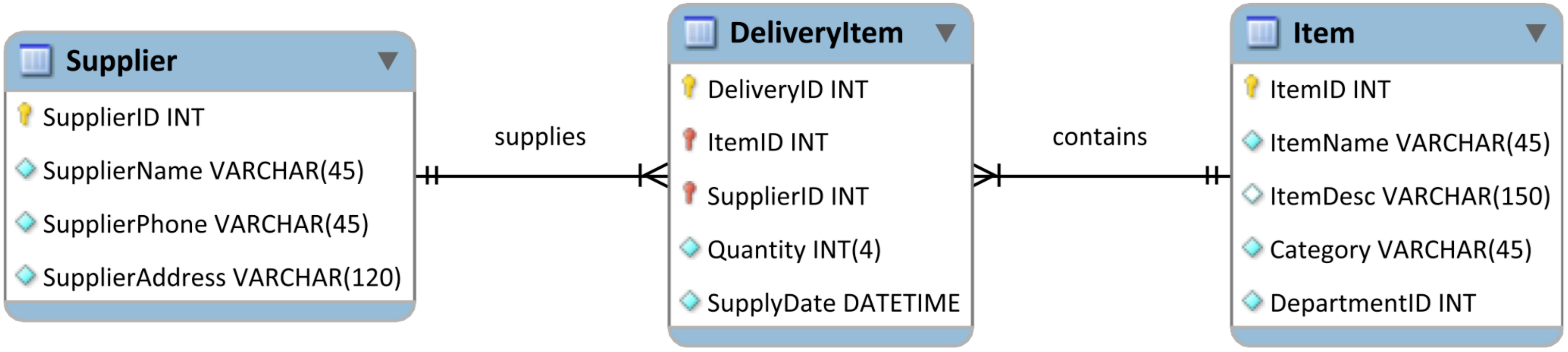
2021.05.10

Agenda

1. **Apply capacity planning concepts - 10min**
2. **Review of backup & recovery concepts - 15min**
3. **Apply backup & recovery concepts to case studies - 10min**
4. **Review transactions concept - 10min**
5. **Apply transactions concepts - 15min**

Apply capacity planning concepts

1. Exercise



50 distinct suppliers that provide 2000 distinct items. The average delivery is of 40 distinct items and each supplier delivers approximately once a week (this to be 50 deliveries a year). For each delivery, there are on average 40 rows added to the DeliveryItem table. While the Item and Supplier tables stay constant in size, the DeliveryItem table grows by 100,000 rows every year.

This assumes that suppliers and items stay constant; however, if the business is successful, the suppliers and frequency of deliveries and number of distinct items delivered can be expected to grow. If we know the length of each row, we can estimate how big each table will be year by year.

Apply capacity planning concepts

1. Exercise

Using the MySQL information about data type storage and information from the data dictionary, the analyst has determined the following average row lengths of each table:

Table	Number of rows	Average row length
Supplier	50 rows	144 bytes
Item	2000 rows	170 bytes
DeliveryItem	0 rows	19 bytes

Table 1: Row volume and row length for the Supplier delivers Item entities.

Note: only Deliver grow: 100,000 rows / year

Assume that the number of suppliers and number of items do not change from year to year, and that the delivery schedule remains the same. Calculate the size of the three tables:

THE UNIVERSITY OF MELBOURNE

Apply capacity planning concepts

1. Exercise

Table	**Number of rows**	**Average row length**
Supplier	50 rows	144 bytes
Item	2000 rows	170 bytes
DeliveryItem	0 rows	19 bytes

Table 1: Row volume and row length for the Supplier delivers Item entities.

Note: only Deliver grow: 100,000 rows / year

a. When database use begins (year 0)

Size(R) = # initial rows × avg length of row

Supplier = 50 × 144 bytes = 7200 bytes (approx. 7 KB
– *Note: 1 KB = 1024 bytes)*

Item = 2000 × 170 bytes = 340,000 bytes (approx. 332 KB)

DeliveryItem = 0 × 19 bytes = 0 bytes

Apply capacity planning concepts

1. Exercise

Table	Number of rows	Average row length
Supplier	50 rows	144 bytes
Item	2000 rows	170 bytes
DeliveryItem	0 rows	19 bytes

Table 1: Row volume and row length for the Supplier delivers Item entities.

Note: only Deliver grow: 100,000 rows / year

b. After one year of database use:

Size(R) = initial size + grow speed × time × avg length of row

DeliveryItem = 0 + 100,000 × 1 × 19 bytes = 1,900,000 bytes
(approx. 1.8 MB – *Note: 1 MB = 1024 KB = 1,048,576 bytes*)

Supplier = 7200 + 0 bytes = 7 KB unchanged
Item = 340,000 + 0 bytes = 332 KB unchanged

Apply capacity planning concepts

1. Exercise

Table	Number of rows	Average row length
Supplier	50 rows	144 bytes
Item	2000 rows	170 bytes
DeliveryItem	0 rows	19 bytes

Table 1: Row volume and row length for the Supplier delivers Item entities.

Note: only Deliver grow: 100,000 rows / year

c. After five years of database use:

Size(R) = initial size + grow speed × time × avg length of row

DeliveryItem = 0 + 100,000 × 5 × 19 bytes = 9,500,000 bytes
(approx. 9.1 MB)

Supplier = 7200 + 0 bytes = 7 KB unchanged
Item = 340,000 + 0 bytes = 332 KB unchanged

Apply capacity planning concepts

1. Exercise

Table	Number of rows	Average row length
Supplier	50 rows	144 bytes
Item	2000 rows	170 bytes
DeliveryItem	0 rows	19 bytes

Table 1: Row volume and row length for the Supplier delivers Item entities.

Note: only Deliver grow: 100,000 rows / year

Tables such as: DeliveryItem table, Order-Item table

are known as ***event tables***

because they record events (sales, deliveries, orders, academic results).

The event tables need to be given **special attention** when applying capacity planning concepts, because they are expected to grow **rapidly**.

Apply capacity planning concepts

Any questions?

Review of backup & recovery concepts

2. Key Concepts of backup & recovery

a. Why do we need a backup of our database?

Backups of databases are required so that in the event of failure of the database system there can be a full recovery of the database so that there is no loss of data. *(If we cannot make a full database recovery, we may end up with data integrity errors and data mismatch)*

Review of backup & recovery concepts

2. Key Concepts of backup & recovery

b. Backup Types

i. Logical

- Save **rows** as they are displayed to users and the **associated metadata information** (column name, data types, indexes columns, type of indexes).
- Include the **structure** of the table – its relationship to other entities, index information, data types.

- Useful when we want to **move data** from one to another operating system. *(Physical backup cannot do that as the file format is usually unique to each operating system and database engines)*.

- Very good for **migrating data** from one database to a completely different database and environment.

Review of backup & recovery concepts

2. Key Concepts of backup & recovery

b. Backup Types

ii. Physical

- **Direct image copy** of the physical database files on the disk.
- **Fastest way** to make a copy of the database using the operating system.

- In database failure, the physical copies can be restored to their original location, and the DBA can then replay all the transactions using the **Crash Recovery log**. (The Crash Recovery log is an area in memory that records every change we make in the database)

- To make a physical copy of the database we can either shut down the MySQL server or perform an open (sometimes called 'hot') backup.

Review of backup & recovery concepts

2. Key Concepts of backup & recovery

c. Backup Modes

i. online

- Online backups mean that the users are **still connected** and are **unaffected** by the backup operations. There is **no loss of availability** of the database.

ii. offline

- Offline backups mean that the database server process is **shut down** while the physical copy of the file is made. No users can connect to the database or process any queries while a database is offline.

Review of backup & recovery concepts

2. Key Concepts of backup & recovery

d. Backup Location

i. onsite

- Onsite backups are stored on the **same premises** – **but not the same machine** – as the database.

ii. offsite

- Offsite backups are stored in a **remote location** (usually **more than 160 km away** from the primary site).

Review of backup & recovery concepts

2. Key Concepts of backup & recovery

e. Backup Choices

i. Full backups

- Back up **all** data in our database

ii. Incremental backups

- Only backs up the **changes** since the last backup.

smaller in size
shorter in duration

iii. Partial backups

back up only chosen tables

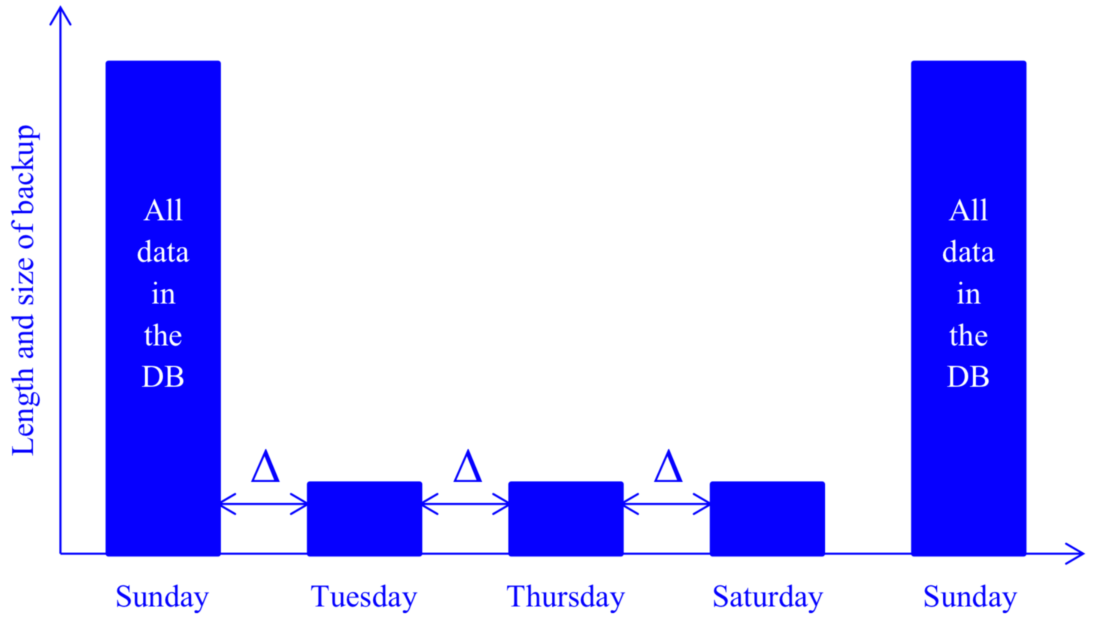


Figure S1: Full versus incremental backups. Full backups are larger in size and therefore take longer to complete.

Review of backup & recovery concepts

2. Key Concepts of backup & recovery

f. How does database recovery work?

Recovery of databases is nearly always in two phases: restore the backup then recover to the point of failure using the recovery log.

1. The first phase is the restoration of the backup to the database server machine.
2. The second phase is the recovery up until the point of the database failure (known as a full recovery). No committed data in the database should be lost.

Review of backup & recovery concepts

Any questions?

Apply backup & recovery concepts to case studies

3. Backup and recovery case study

ACME Manufacturing makes widgets in its factory. The factory runs 24 hours a day, 7 days a week in three shifts. The quietest shift is the Sunday night shift, which runs from midnight Sunday to 8am Monday. While ACME manufactures widgets, the database must run. This is ACME's only widget factory.

The database administrator has implemented a backup policy that takes a full backup every Sunday at 3am during the night shift, and then an incremental backup on Tuesday, Thursday and Saturday mornings at 3am.

The backup strategy has determined that if there is a database failure, restoration of the database is time-critical. ACME must have the shortest outage time to restore and recover the database. This means the database must be restored quickly so that the manufacturing can continue. ACME must have the smallest elapsed time from the point of failure to the database being fully operational and useable.

Apply backup & recovery concepts to case studies

3. Backup and recovery case study

a. Given the business requirements and the database administrator's backup policy, what database backup type, mode and site would you recommend?

online, *offsite*, *physical* backup.

The online backup would mean there would be **no interruption** of operations at ACME.

The physical backup is the preferred type of backup because it is the **fastest** to backup and restore.

While having an onsite backup creates a risk of a single point of failure, it is ACME's only widget factory, so if it was to be destroyed ACME does not have anywhere else to make widgets. An offsite backup provides little advantage.

Apply backup & recovery concepts to case studies

3. Backup and recovery case study

b. Consider the Full and Incremental backup timeline in Figure 1. If the database suffered a media failure on Friday at 9:23am, how many backups would need to be restored?

Three backups would need to be restored:

1. The Sunday full backup
2. The Tuesday incremental backup
3. The Thursday incremental backup

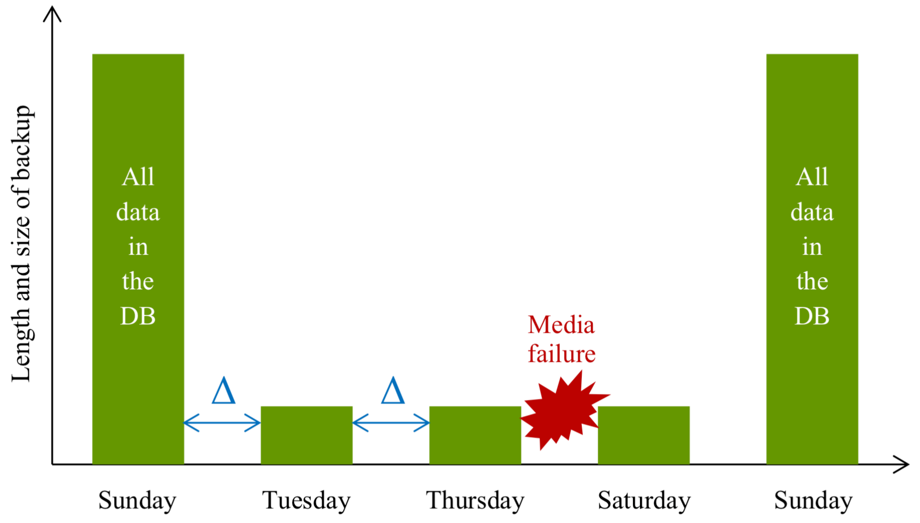


Figure 1. A timeline of full and incremental backups showing the media failure on Friday morning.

The DBA would also replay the crash recovery logs from Friday morning until Friday 9:23am.

Apply backup & recovery concepts to case studies

3. Backup and recovery case study

c. Given the same failure, what would be the benefits and costs of changing the backup strategy to do full backups on Sunday, Tuesday, Thursday and Saturday mornings at 3am?

Benefits: In the restore and recovery phase, this strategy would reduce the time to fully restore the database from the last full backup and apply the crash recovery logs. This would mean ACME are going to be fully operational faster than if the Full and Incremental strategy was used.

Costs: More space would be required to back up the database as the backup will take everything and not just the changes. As there is more data to copy, the backup will take longer to complete. There is a risk that if we don't remove the backups from the database server, we could fill up the file system.

Apply backup & recovery concepts to case studies

Any questions?

Review transactions concept

4. Transaction Key Concepts

a. What is a transaction?

A transaction is a **logical unit of work** that must either be entirely completed or aborted.

A transaction usually corresponds to a single "action" that involves several changes to the database, for example, removing an employee and all their related data from the database, or an actual financial transaction that removes funds from one account and adds them to another.

Transactions adhere to the **ACID principles**.

Review transactions concept

4. Transaction Key Concepts

b. What is ACID

Four desirable properties of transactions. DBMS that implements transactions that achieve these properties is “**ACID-compliant**”

Atomic – Each transaction either entirely succeeds or entirely fails. If a failure occurs mid- transaction, the DBMS must discard (roll back) all of the transaction’s changes up to that point.

Consistent – Upon completion, a transaction must respect data integrity rules (constraints) and leave the database in a consistent state.

Review transactions concept

4. Transaction Key Concepts

b. What is ACID

Isolated – Changes made in one transaction cannot be seen from within other transactions. Transactions give the impression of being executed side-by-side simultaneously, although in practice, the need for data locking means that some transactions might be delayed by other transactions as they wait for locks to be released.

Durable – Once transaction is committed, the changes carried out in that transaction persist permanently in database.

THE UNIVERSITY OF MELBOURNE

Review transactions concept

4. Transaction Key Concepts

c. Concurrency

Concurrency

Ability to allow many users to connect to and work with the database **simultaneously**.

It is possible for a database to allow concurrent access and still satisfy the ACID properties, but the isolation property creates particular challenges.

Review transactions concept

4. Transaction Key Concepts

d. The lost update problem

When two users attempt to update the **same piece of data** in the database **at the same time**, they might conflict with each other. One user changes the data value, and the other user does not see this update before writing their own update to that database. The first user's update is lost.

THE UNIVERSITY OF MELBOURNE

Review transactions concept

4. Transaction Key Concepts

d. The lost update problem

Example: two people are accessing the same bank account. Suppose my bank account contains $500. I am at an electronics store buying a $300 TV; at exactly the same time, my employer is paying me $120 in wages.

My transaction at the electronics store	**My employer's transaction**
Read account balance ($500)	
	Read account balance ($500)
Write account balance less $300 ($200)	
	Write account balance plus $120 ($620)

Instead of being $180 poorer, I am now $120 richer! This is not my fault – it's up to the bank to make sure the isolation property of ACID is satisfied and lost updates cannot happen.

Review transactions concept

Any questions?

Apply transactions concepts

5. Transaction Exercises

It’s class registration day, when UniMelb students register in tutorial classes for the upcoming semester. In one particular subject, each tutorial class can fit a maximum of 24 students.

Eamonn and Jacqueline both wish to register in the Wednesday 10am tutorial class for this subject. This class already has 23 students enrolled – just one place remains.

Suppose the database contains tables like this:

$$\text{TutorialClass}\ (\overset{\text{FK}}{\underline{\text{SubjectCode}}},\ \underline{\text{TutorialNumber}},\ \text{TotalEnrolments})$$

$$\text{TutorialEnrolment}\ (\overset{\text{FK}}{\underline{\text{SubjectCode}}},\ \overset{\text{FK}}{\underline{\text{TutorialNumber}}},\ \overset{\text{FK}}{\underline{\text{StudentNumber}}})$$

Apply transactions concepts

5. Transaction Exercises

a. Describe how a lost update could occur in this database when Eamonn and Jacqueline try to simultaneously register in the Wednesday 10am tutorial.

Eamonn	**Jacqueline**
Read TutorialClass.TotalEnrolments (23)	
	Read TutorialClass.TotalEnrolments (23)
Insert row into TutorialEnrolment	
	Insert row into TutorialEnrolment
	Write TutorialClass.TotalEnrolments (24)
Write TutorialClass.TotalEnrolments (24)	

Even though there are now 25 students enrolled in the class, the value of TotalEnrolments for this class is equal to 24. A lost update has occurred.

Apply transactions concepts

5. Transaction Exercises

b. How could the lost update problem be avoided in this situation?

1. Enforce **serial execution**, only one transaction is executed at a time.

   very inefficient, but will guarantee that the isolation of ACID is satisfied.

2. Use **locking**. When a transaction wishes to read the TotalEnrolments value for a class, it takes out a lock on that row of the TutorialClass table. This prevents other transactions from modifying that row. Once the transaction has finished writing to the row, it releases the lock.

   more efficient, as a student's enrolment request only has to wait for the completion of other requests to enrol *in the same class*, instead of waiting for all other requests in the system.

Apply transactions concepts

5. Transaction Exercises

b. How could the lost update problem be avoided in this situation?

3. Use the other concurrency control methods outlined in the lecture

I. **timestamps** (if the timestamp of TotalEnrolments changes between when Eamonn reads it and when he is about to write it, Eamonn's transaction would abort and restart)

II. **optimistic concurrency control** (if TotalEnrolments is no longer equal to its original value when Eamonn is about to write it, Eamonn's transaction would abort and restart).

Apply transactions concepts

Any questions?