1

Topics

- What is a Transaction?
  - Properties of transactions – ACID
  - Transaction states
- Concurrency Control
  - What if not controlled?
  - Concurrency Control in practice – locks, isolation levels
- Concurrency Control in Oracle

RMIT UNIVERSITY

2

2

What is a transaction?

- Transaction:
  Action, or series of actions, carried out by user or application, which reads or updates contents of database.

RMIT UNIVERSITY

3

What is a transaction?

- Transaction:
  Action, or series of actions, carried out by user or application, which reads or updates contents of database.

```
BEGIN TRANSACTION;
SELECT bal INTO :oldBal
    FROM account
    WHERE acctNo = <card_input> AND
          pin = <user_input>;
newBal = oldBal - <user_input>;
INSERT INTO txnLog
    VALUES (<user_input>, timestamp, :oldBal, :newBal);
UPDATE account
    SET bal = :newBal;
    WHERE acctNo = <card_input>;
COMMIT;
```

4

4

A transaction in action

- Can have one of two outcomes:
  - Success - transaction commits and database reaches a new consistent state.
  - Failure - transaction aborts, and database must be restored to consistent state before it started.
- Such a transaction is rolled back or undone.
- Committed transaction cannot be aborted.
- Aborted transaction that is rolled back can be restarted later.

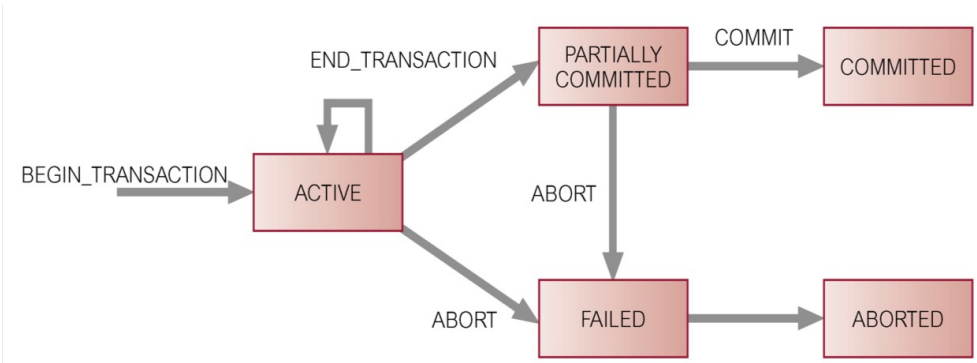
e.g.: ATM had a mechanical failure and couldn't dispense cash.

RMIT UNIVERSITY

5

5

A transaction in action



RMIT UNIVERSITY

6

6

A transaction in action

- Can have one of two outcomes:
  - Success - transaction commits and database reaches a new consistent state.
  - Failure - transaction aborts, and database must be restored to consistent state before it started.
- Apart from these two obvious states, it can be "partially committed" or "Failed".

RMIT UNIVERSITY

7

7

A transaction in action

- **PARTIALLY COMMITTED:**
  - occurs after the final statement has been executed
  - At this point, it may be found that the transaction cannot be committed (serialisability issue, integrity constraint violation, or other failure external to the database).
  - In such cases, the transaction would go into the FAILED state and would have to be aborted.
  - If the transaction has been successful, any updates can be safely recorded and the transaction can go to the COMMITTED state.

RMIT UNIVERSITY

8

8

A transaction in action

- **FAILED:**
  - occurs if the transaction cannot be committed or the transaction is aborted while in the ACTIVE state, perhaps due to the user aborting the transaction or as a result of a failed "partially committed" transaction.
  - All database actions will be rolled back at this stage.

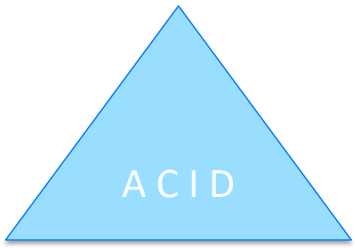
RMIT UNIVERSITY

9

9

Properties of a transaction

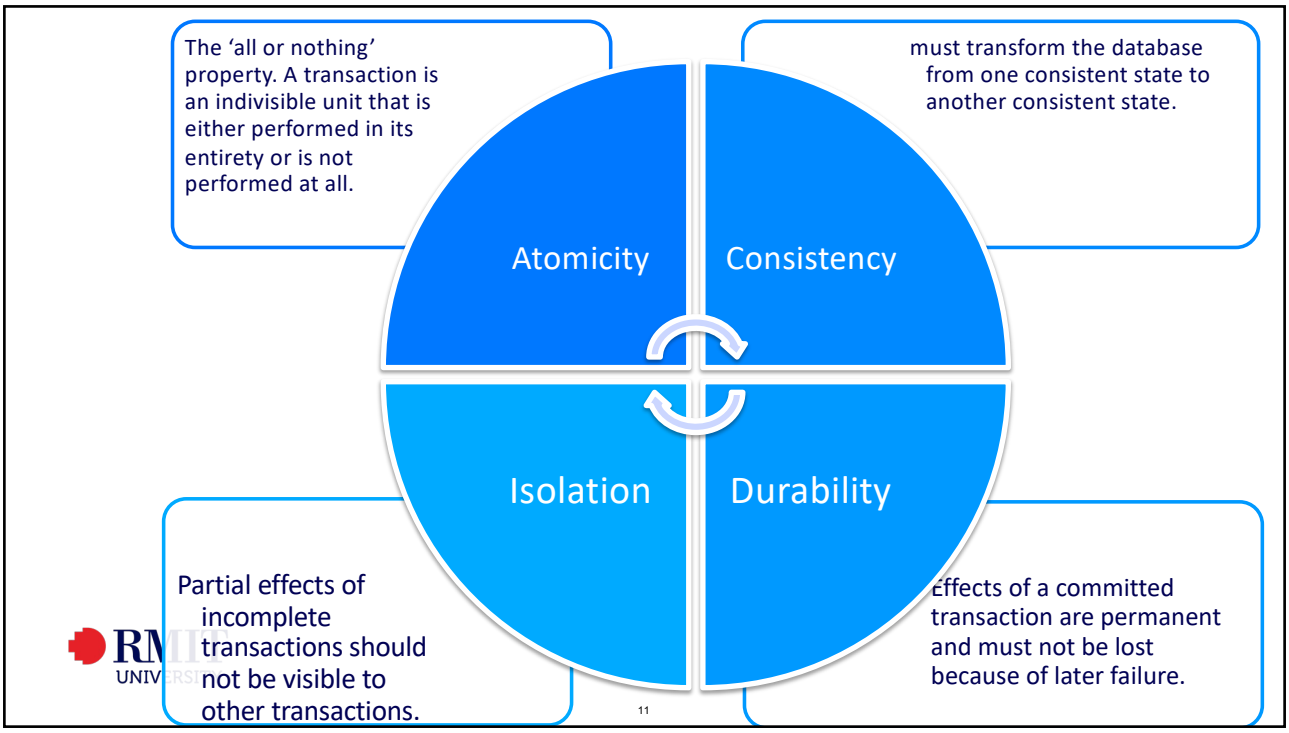
- In order to safeguard data and ensure the integrity of the whole database system, any transaction management system should have FOUR properties.



RMIT UNIVERSITY

10

10



11

Concurrency Control

- You can ensure "I" of ACID, if there is only one transaction in action in any given time.
- However, it is not practical in large multi-user database environments – one ATM transaction at any point in time?
- Concurrency is essential, but, must be well controlled, in order to ensure "I" as well as high efficiency.

RMIT UNIVERSITY

12

12

Concurrency (non-)Control

13

13

Concurrency Control

- Process of managing simultaneous operations on the database without having them interfere with one another.
- Prevents interference when two or more users are accessing database simultaneously and at least one is updating data.
- Although two transactions may be correct in themselves, interleaving of operations may produce an incorrect result.

RMIT UNIVERSITY

14

14

A Need for Concurrency Control

- Three examples of potential problems caused by concurrency:
  - Lost Update Problem
  - Dirty-Read Problem
  - Non-repeatable Read Problem

RMIT UNIVERSITY

15

15

Lost Update Problem

- Successfully completed update is overridden by another user.

Time	Transaction 1	Transaction 2	Bank Balance (BB) (X)
t1		BEGIN TXN	100
t2	BEGIN TXN	Read (BB(X))	100
t3	Read (BB(X))	BB(X) = BB(X) +100	100
t4	BB(X) = BB(X) -10	Write (BB(X))	200
t5	Write (BB(X))	COMMIT	90
t6	COMMIT		90

If T1 started after completing T2 (serial schedule), final balance would be 190.

RMIT UNIVERSITY

16

16

Dirty-Read Problem

- Occurs when one transaction can see intermediate results of another transaction before it has committed.

Time	Transaction 3	Transaction 4	Bank Balance (BB) (X)
t1		BEGIN TXN	100
t2		Read (BB(X))	100
t3		BB(X) = BB(X) +100	100
t4	BEGIN TXN	Write (BB(X))	200
t5	Read (BB(X))		200
t6	BB(X) = BB(X) -10	ROLLBACK	100
t7	Write (BB(X))		190
t8	COMMIT		190

If T3 waited until T4 is completed (i.e. aborted), it would have read the correct balance.

17

17

Non-repeatable Read Problem

- Occurs when one transaction reads same data twice (or more) and values differ between reads.

Time	Transaction 5	Transaction 6	Bank Balance (BB) (X)
t1		BEGIN TXN	100
t2	BEGIN TXN	Read (BB(X))	100
t3	Read (BB(X))	BB(X) = BB(X) +100	100
t4	...	Write (BB(X))	200
t5	...	COMMIT	200
t6	Read (BB(X))		200
t7	COMMIT		200

Two reads in T5 will give you two values, even if you didn't change it within T5.

18

18

Solution

- Concurrency Control
  - Use of locks
  - Most DBMSs offer a number of transaction isolation levels – a compromise between "I" and efficiency.

RMIT UNIVERSITY

19

19

Locks

- Two types: READ (Shared) or WRITE (Exclusive) locks
- If transaction has **shared lock** on an item, can read but not update the item. Other transactions can read the same item, but cannot update.
- If transaction has **exclusive lock** on an item, can both read and update the item. Other transactions can neither read nor update the same item.

More than one transaction can hold shared locks simultaneously on the same item.

RMIT UNIVERSITY

20

20

Isolation Levels

- Most DBMSs offer a number of transaction isolation levels, which control the degree of locking that transactions require to acquire for various operations.
  - Serialisable
  - Repeatable reads
  - Read committed
  - Read uncommitted

Highest isolation level. Most database texts will only mention about this level of isolation.
For many database applications, the majority of database transactions can be constructed to avoid requiring SERIALISABLE level, thus reducing the locking overheads.

RMIT UNIVERSITY

21

21

Isolation Levels – Serialisable

- In a **Serialisable Schedule**, conflicting operations (Read-Write, Write—Read, or Write-Write) are executed in the same order as they would execute in a serial schedule.
- This ensures "I" among transactions in the schedule.
- Locks are used to enforce this.

A Serial Schedule is a schedule where there aren't any interleaving at all. Each transaction will execute one after another.

RMIT UNIVERSITY

22

22

Isolation Levels – Serialisable

Time	Transaction 1	Transaction 2	Bank Balance (BB) (X)
t1		BEGIN TXN	100
t2	BEGIN TXN	Write-Lock(X)	100
t3	Write-Lock(X)	Read (BB(X))	100
t4		BB(X) = BB(X) +100	200
t5	Waiting to accquire	Write (BB(X))	200
t6	the lock ...	Unlock (X)	200
t7	Read (BB(X))	COMMIT	200
t8	BB(X) = BB(X) -10		
t9	Write (BB(X))		190
t10	Unlock (X)		190
t11	COMMIT		190

T1 goes into a "wait" state until Write-lock on X becomes available. Makes everything a bit slow, but ensure accurate result!

23

23

Isolation Levels – Repeatable Reads

- In this isolation level, DBMS keeps read and write locks until the end of the transaction.
- However, range-locks are not managed, so phantom reads can occur.

A range lock will lock a whole bunch of rows (say, rows matching a WHERE clause) , not just one row like other row-level locks.

A phantom read occurs when new rows are added or removed by another transaction while first transaction still reads data.

RMIT UNIVERSITY

24

24

Isolation Levels – Read Committed

- In this isolation level, DBMS keeps write locks until the end of the transaction, but read locks are released as soon as the SELECT operation is performed.
- So the non-repeatable reads phenomenon can occur in this isolation level.

Most DBMS's allow the user to decide the isolation level for their application.
Critical txn – serializable
Non-critical (e.g. summarrising) – read committed

RMIT UNIVERSITY

25

25

Isolation Levels – Read Uncommitted

- This is the lowest isolation level.
- In this level, dirty reads are allowed, so one transaction may see not-yet-committed changes made by other transactions.

Again, some applications may not too strict about the absolute accuracy of results (say in market research application).

RMIT UNIVERSITY

26

26

Isolation Levels – Summary

Isolation level	Dirty reads	Lost updates	Non-repeatable reads	Phantoms
Read Uncommitted	may occur	may occur	may occur	may occur
Read Committed	don't occur	may occur	may occur	may occur
Repeatable Read	don't occur	don't occur	don't occur	may occur
Serializable	don't occur	don't occur	don't occur	don't occur

Source: https://en.wikipedia.org/wiki/Isolation_(database_systems)

RMIT UNIVERSITY

27

27

Concurrency Control in Oracle

- Oracle allows: Serialisable, Read Committed, and Read-only isolation levels.
- You can set them for individual transactions, or at the beginning for the session (preferred).

```
ALTER SESSION SET ISOLATION_LEVEL = SERIALIZABLE;

ALTER SESSION SET ISOLATION_LEVEL = READ COMMITTED;
```

RMIT UNIVERSITY

28

28

Demo

Concurrency Control in Oracle

RMIT UNIVERSITY

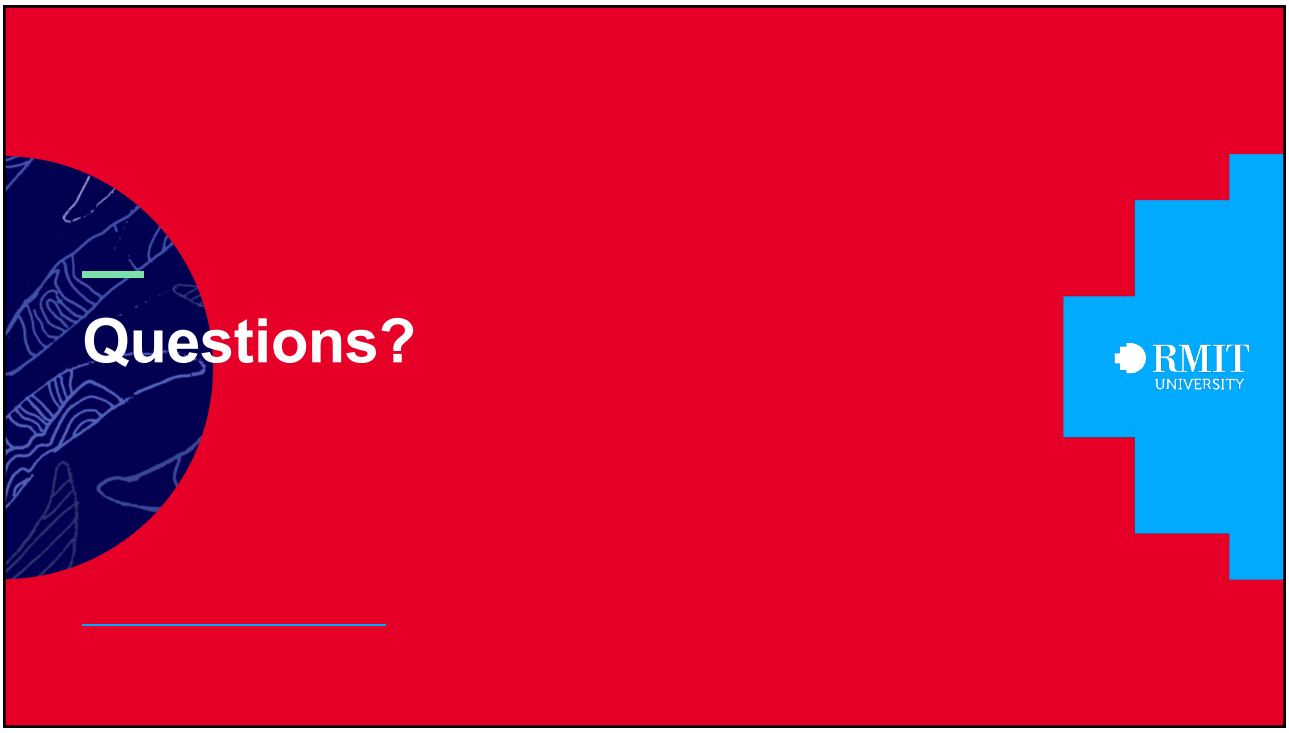
29

Menti-time!

Menti.com
Code: to be supplied

RMIT UNIVERSITY

30



31