SQL 10: TRANSACTIONS - 1

24/05/24

AGENDA

1. Transactions
2. ACID properties
3. Commits & Rollbacks
4. Isolation levels
5. Read Uncommitted

TRANSACTION

Table: Accounts

c_id	c_name	balance
1	Harish	1000
2	Devesh	1000
3		
4		
5		

Transfer Money.

Harish $\xrightarrow{500}$ Devesh

1. Get Balance of Harish → Hb → DBcall
2. Hb > 500
3. Hb - 500
4. Write (Hb) = 500 → DBcall
5. Get Balance of Devesh → Db → DBcall
6. Db + 500
7. Write (Db) = 1500 → DBcall.

Pseudo Code :-

```
transfer_money( from, to , amount) :
        x = read ( from)
        if x >= amount :
            x = x - amount
            write ( from) = x
            y = read ('To')
            y = y + amount
            write ('To') = Y.
```

Ques 2 **Transaction :-**

Raushan $\xrightarrow{500}$ Arun
Vishal $\xrightarrow{200}$ Arun

Initial		Final	
Raushan	1000	Raushan	500
Arun	1000	Arun	1700
Vishal	1000	Vishal	800

transfer (from, to, amount)

```
x = read ( from )
if x >= amount :
        x = x - amount
        write ( from ) = x
        y = read ( To )
        y = y + amount
        write ( To ) = y.
```

$R \xrightarrow{500} A$	$V \xrightarrow{200} A$
x = 1000	x = 1000
True	True
x = 1000 - 500 = 500	x = 800
R = 500	V = 800
A = 1000	A = 1000
A = 1000 + 500 = 1500	A = 1000 + 200 = 1200
A = 1500	A = 1200
COMMIT :	COMMIT :

Arun = 1500 , 1200

Initial		Final	
Raushan	1000	Raushan	500
Arun	1000	Arun	1500, 1200
Vishal	1000	Vishal	800
Total	3000	Total	2800 , 2500

Issues

① Illogical state / Inconsistent.

② Complete operation may NOT execute.

TRANSACTIONS

A set of DB operations - logically grouped together.

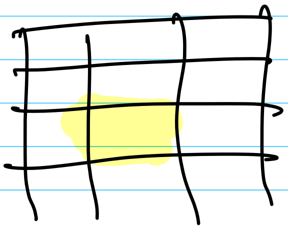
eg

```
BEGIN TRANSACTION
        SELECT
        UPDATE
        SELECT
        UPDATE

END TRANSACTION.
```

A →	Atomicity
C →	Consistency
I →	Isolation
D →	Durability.

① **Atomicity.**



Atom - cannot be divided further.

All changes to data are performed as if they are one single operation.

ie ALL changes are done or NONE!

② **CONSISTENCY**

→ same, exact
→ logically correct
→ accurate

account.

c_id	c_name	balance
1	Harish	1000 − 200 800
2	Arun	1000 + 200 1200
3		

#12345

→ transaction detail

trans_id	amount
# 12345	200

Banking / Finance — Consistency -- 100%

③ **ISOLATION** (Separate)

The intermediate state of a transaction is "invisible" to another transaction.

NOTE: Each transaction will set its own isolation level.

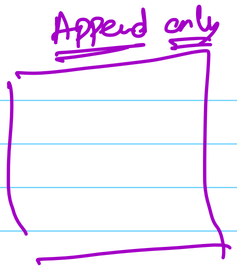
④ **DURABILITY**

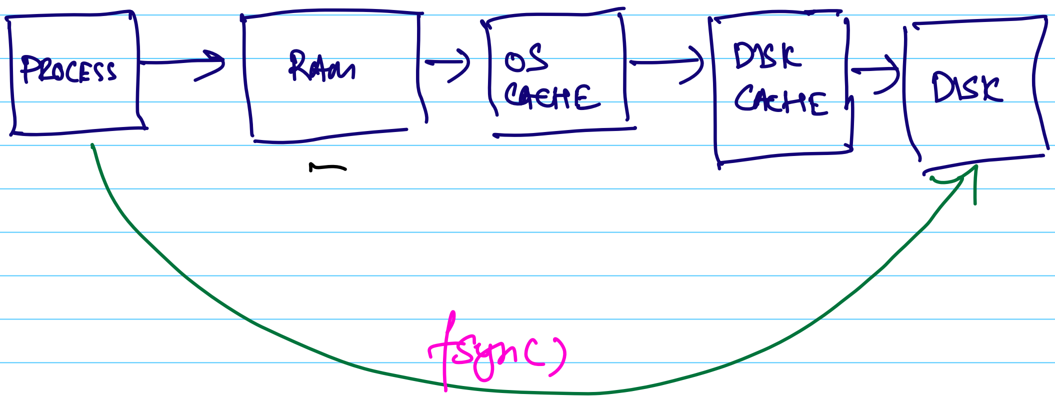
→ long lasting.

After a transaction is successful, changes to data persist, and are NOT UNDONE, even in the event of system failure.

Write Ahead log (WAL)

fsync()
Append only file.



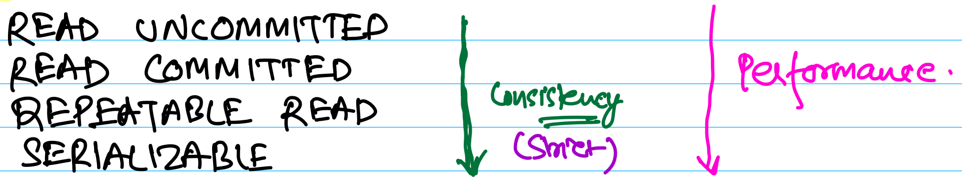


BREAK TILL – 8:14 AM.

COMMIT & ROLLBACK

Commit	==	Marriage
Rollback	==	Break-up.

ISOLATION LEVELS



NOTE: Isolation level of other transactions DO NOT matter!

① READ UNCOMMITTED

→ Problem: **Dirty Read.**

Allows transaction to read even "uncommitted" data from another transaction.

Pros → Fast

Cons → Inconsistency. →(if rollback)

ISOLATION LEVELS

READ UNCOMMITTED	(RU)
READ COMMITTED	(RC)
REPEATABLE READ	(RR)
SERIALIZABLE	(S)

(severity increases ⟶)

$$\begin{tikzcd} RU \arrow[r, dashed] & RC \arrow[r, dashed] & RR \arrow[r, dashed] & S \\ & \text{Postgres} \arrow[u] & \text{MySQL} \arrow[u] & \text{CockroachDB.} \arrow[u] \end{tikzcd}$$

```sql
-- SQL 10 TRANSACTIONS 1

-- SESSION 1
USE SAKILA;

SELECT * FROM FILM WHERE FILM_ID = 1;

UPDATE FILM
SET TITLE = "IT"
WHERE FILM_ID = 1;

SET AUTOCOMMIT = 0;
BEGIN;

UPDATE FILM
SET TITLE = "ABCD"
WHERE FILM_ID = 1;

SELECT * FROM FILM WHERE FILM_ID = 1;

COMMIT;

-- ROLLBACK;
SET AUTOCOMMIT = 0;
BEGIN;

UPDATE FILM
SET TITLE = "JAWAAN"
WHERE FILM_ID = 1;

SELECT * FROM FILM WHERE FILM_ID = 1;

ROLLBACK;

SELECT * FROM FILM WHERE FILM_ID = 1;
```

```sql
-- SHOW ISOLATION LEVEL

SHOW VARIABLES LIKE '%ISOLATION%';

-- SESSION - 1

SET AUTOCOMMIT = 0;
BEGIN;
SELECT * FROM FILM WHERE FILM_ID = 1;

UPDATE FILM
SET TITLE = "YJHD"
WHERE FILM_ID = 1;

UPDATE FILM
SET TITLE = "DDLJ"
WHERE FILM_ID = 1;
```

```
-- SESSION 2
USE SAKILA;
SELECT * FROM FILM WHERE FILM_ID = 1;

-- SHOW ISOLATION LEVEL

SHOW VARIABLES LIKE '%ISOLATION%';

-- READ UNCOMMITTED
SELECT * FROM FILM WHERE FILM_ID = 1;

SET SESSION TRANSACTION ISOLATION LEVEL READ
UNCOMMITTED;
SELECT * FROM FILM WHERE FILM_ID = 1;

SET SESSION TRANSACTION ISOLATION LEVEL READ
COMMITTED;
SELECT * FROM FILM WHERE FILM_ID = 1;
```