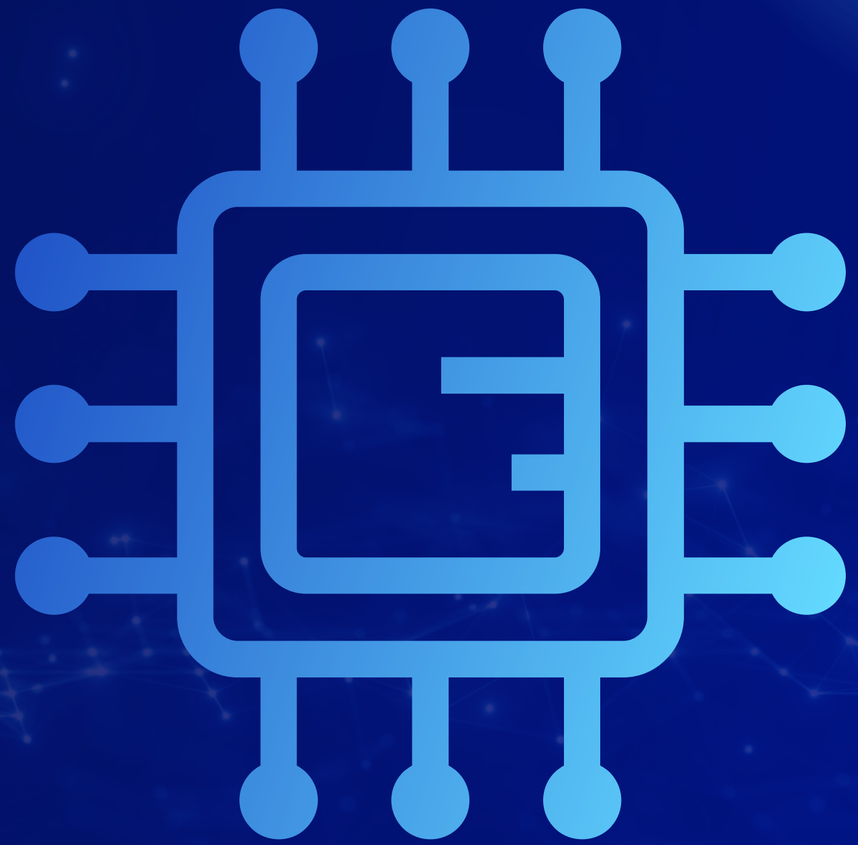
Thynk Unlimited

Data Analysis Presentation

SQL Transaction

An SQL transaction is a sequence of one or more SQL operations (e.g., INSERT, UPDATE, DELETE) executed as a single unit of work. Transactions ensure that either all operations succeed or none are applied, maintaining data integrity.



SQL Transaction Control Commands

Begin SQl Command

The BEGIN TRANSACTION command marks the beginning of a new transaction. All SQL statements that follow this command will be part of the same transaction until a COMMIT or ROLLBACK is encountered. This command doesn't make any changes to the database, it just starts the transaction.ity.

Syntax:

```
BEGIN TRANSACTION transaction_name ;
```

Commit SQl Command

The COMMIT command is used to save all changes made during the current transaction to the database. Once a transaction is committed, the changes are permanent.

Syntax:

```
COMMIT;
```

RollBack SQl Command

The ROLLBACK command is used to undo all changes made in the current transaction. It is used when an error occurs or when the desired changes cannot be completed. The database will revert to the state it was in before the BEGIN TRANSACTION was executed.

Syntax:

```
ROLLBACK;
```

SavePoint SQl Command

Syntax:

```
SAVEPOINT SAVEPOINT_NAME;
```

Example

```
SAVEPOINT SP1;
//Savepoint created.
DELETE FROM Student WHERE AGE = 20;
//deleted
SAVEPOINT SP2;
//Savepoint created.
```

RollBack to save point

The ROLLBACK TO SAVEPOINT command allows us to roll back the transaction to a specific savepoint, effectively undoing changes made after that point.

Syntax:

```
ROLLBACK TO SAVEPOINT SAVEPOINT_NAME;
```

Release save point

This command is used to remove a SAVEPOINT that we have created. Once a SAVEPOINT has been released, we can no longer use the ROLLBACK command to undo transactions performed since the last SAVEPOINT.

Syntax:

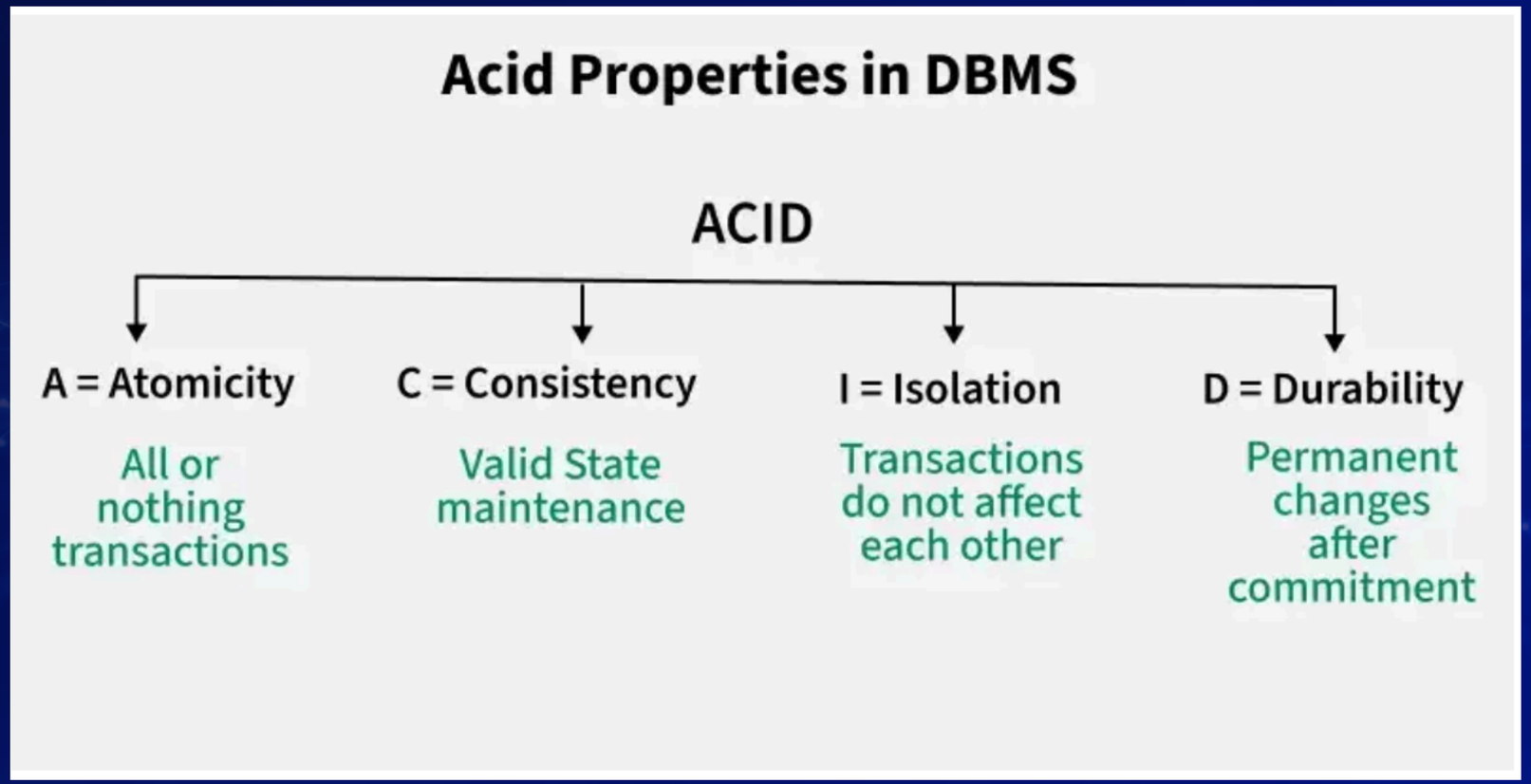
```
RELEASE SAVEPOINT SAVEPOINT_NAME;
```

Example

Once the savepoint `SP2` is released, we can no longer roll back to it.

```
RELEASE SAVEPOINT SP2; -- Release the second savepoint.
```

ACID principle



Atomicity

Atomicity means a transaction is all-or-nothing either all its operations succeed, or none are applied. If any part fails, the entire transaction is rolled back to keep the database consistent.

- Commit: If the transaction is successful, the changes are permanently applied.
- Abort/Rollback: If the transaction fails, any changes made during the transaction are discarded.

Consistency

Consistency in transactions means that the database must remain in a valid state before and after a transaction.

- A valid state follows all defined rules, constraints, and relationships (like primary keys, foreign keys, etc.).
- If a transaction violates any of these rules, it is rolled back to prevent corrupt or invalid data.
- If a transaction deducts money from one account but doesn't add it to another (in a transfer), it violates consistency.

Isolation

Isolation ensures that transactions run independently without affecting each other. Changes made by one transaction are not visible to others until they are committed.
It ensures that the result of concurrent transactions is the same as if they were run one after another, preventing issues like:

- Dirty reads: reading uncommitted data
- Non-repeatable reads: data changes between two reads
- Phantom reads: new rows appear during a transaction

Durability

Durability ensures that once a transaction is committed, its changes are permanently saved, even if the system fails. The data is stored in non-volatile memory, so the database can recover to its last committed state without losing data.

ThankYou

-