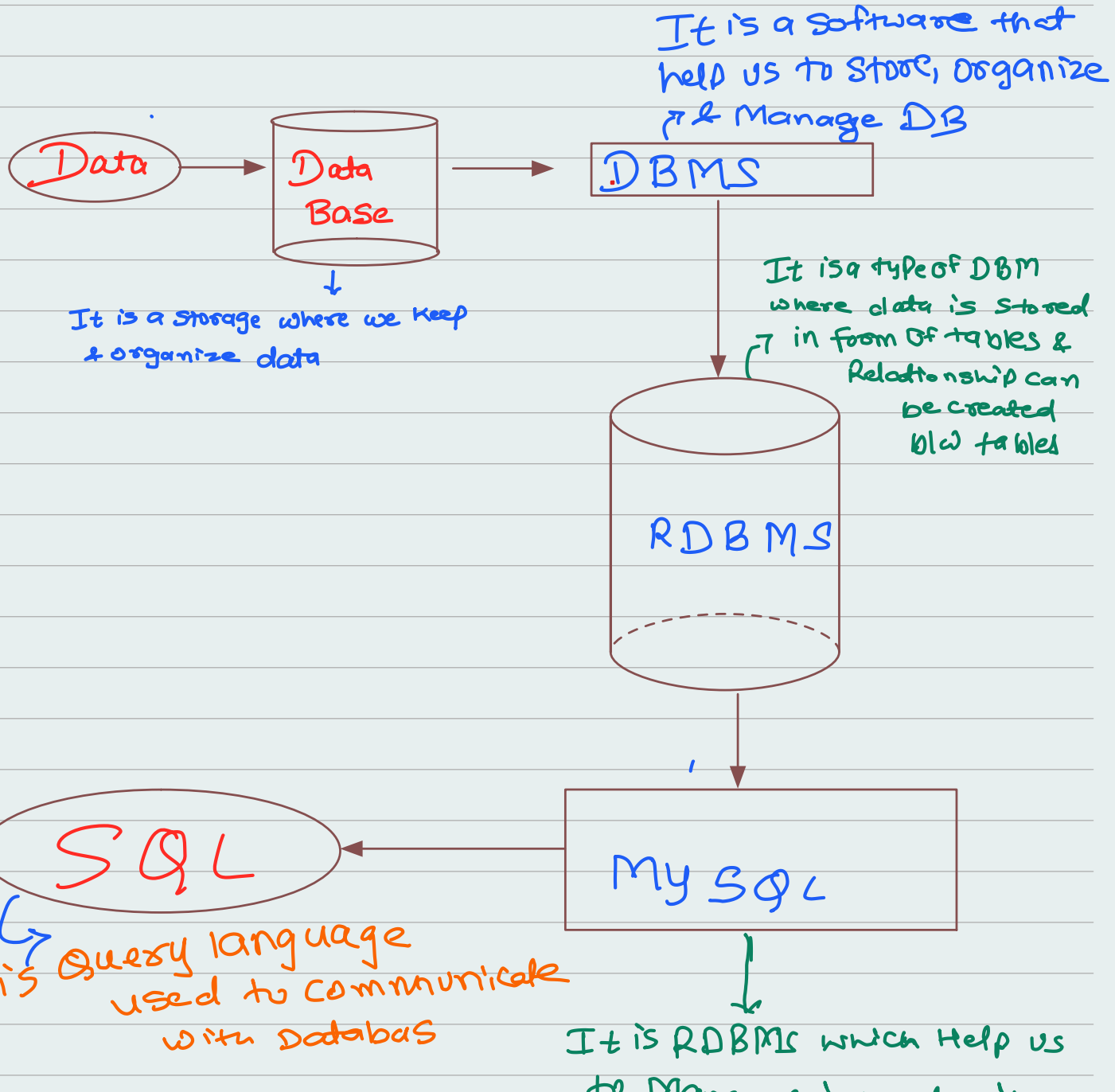
SQL



Char Name Char(10)

K	a	r	t	h	i	K			

7 len | 7 Char

Data Types

Numerical DT

Data Typ	Storage	When to use
INT	4 bytes	Whole Number (age, rollno, quantity)
Small int	2 bytes	Small whole Number (marks)
Big int	8 bytes	very large whole Number (Aadhar NO)
Decimal	Depend on precision as a Decimal	For exactly values (Salary, money)
Number		Exactly Precision
float	4 bytes	Approximate decimal (Scientic cal)
Double	1 Bit	Larger floating with more precision.

Character| String DT

DataType	Storage	When to use
Char(n)	Fixed n bytes	For fixed length (Pincode, Gender
varchar(n)	variable length	For variable length (Name, city, email
Text	upto 65535 char	Large text (Article, Blogs.

Date & Time DT

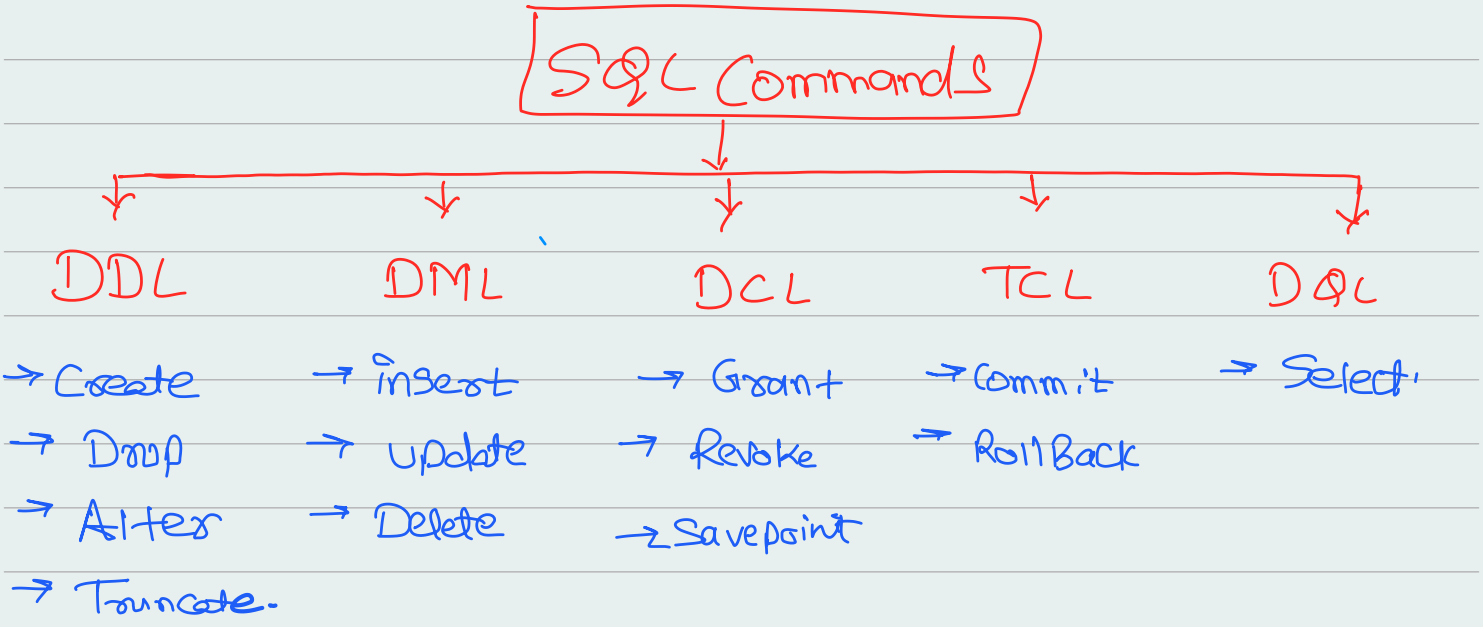
Data type	Storage	When to use
Date	3 bytes	Store only Date ("2025-09-09"
Time	3 bytes	Store only time (' 14:30:00"
Date time	8 bytes	Store both Date & time ('2025-09-09 14:30:00')
Time Stamp	4 bytes	Store date time auto update
Year	1 byte	Store year only ('2025')

SQL

SQL is like a Query Language we used to talk to a database. Just like we use english to communicate with each other, we use SQL to communicate with Data.

To do this, SQL gives us diffesents types of Commands. each with a Specific purpose.

Commands

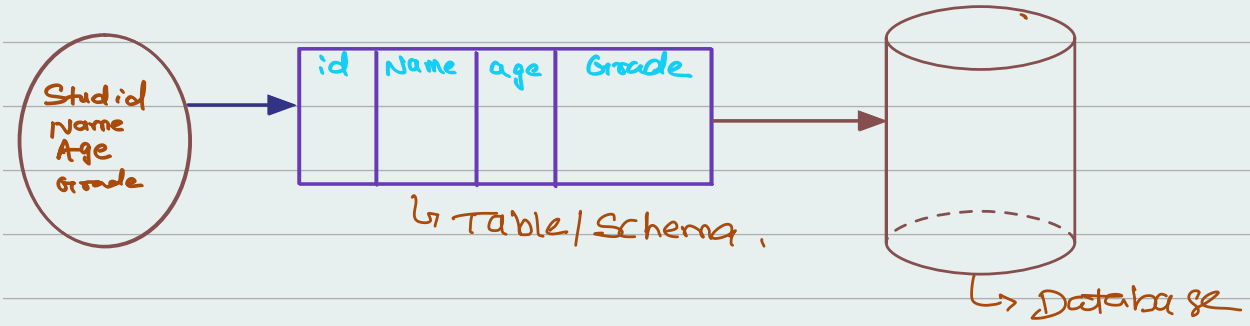


① DDL → Data Defination Language

Used to define and change Structure of Data base Objects (tables, Schema, idexes).

→ Once we executed, Changes are permanent.

① Create → Create a table & database



Query:- Create table table_name (col1, col2, ---

Create table Students (id, name, age, Grade);

② Alter

Used to modify Structure of a table.

add, delete, modify a column

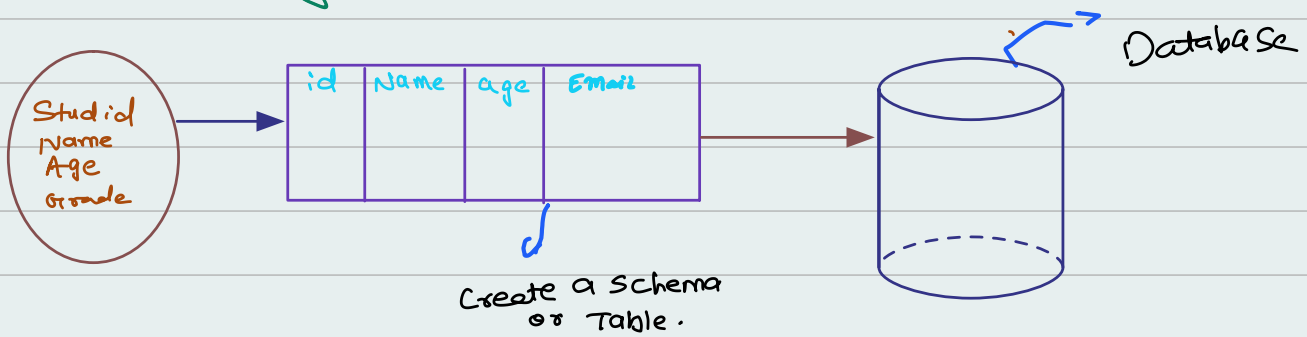
It dosen't delete Data.

→ Add a New Column

→ Modify a Column Size

→ Delete a Column.

Let's consider for now we are adding New column Email & Removing Gender



① Add a New Column to table:

Syntax :- Alter table table Add column Datatype ;

→ Alter table Students Add email varchar(100);

Add Multiple Columns

Syntax → Alter Table Students
Add city varchar(50);
Add gender varchar(10);

Add Phone_N

→ Alter table students Add phone_N varchar(15

② Modify the existing column

Syntax :- Alter table tablename Modify columnname Datatype ;

Alter table Students Modify age varchar(3);

Alter table students Modify name varchar(100);

③ Change column name

Syntax :- Alter table tablename Change

Alter table Students change id enrollment_id;

③ Drop → Delete a Entire table (Structure + data) permently.

Syntax :- Alter table tablename Drop column Column name;

Drop city → Alter table students Drop colum city;

Drop entire table → Drop table students;

④ Truncate :- Delete all Rows from a table but keep the structure

→ faster to delete

Syntax :- Truncate table tablename;

→ Truncate table students;

② DML → Data Manipulation Language

Commands that manipulate (add, update delete) data inside a tables

① Insert

→ Insert a Data into table./Add a new Data

Two types of Insert we have :-

① Explicit Insert → Safer, flexible

② Implicit Insert.

① Explicit Insert

Syntax:- Insert into table_name (col_1, col2, col3 ...)
Values (val1, val2, val3 ...) ;

→ Insert into students (id, name, age, email, city, Gender);
values (1, "vivek", 24, "vivek@gmail.com", "Hyd", "M");

② Implicit Insert → No need to mention columns
value must be in exact order

Syntax:- Insert into table_name
values (value1, value2, value3 ....) ;

Insert into students
values (2,'charan', 23 , "charan@gmail.com",'| Hyd", "M"),
Values (3,'shouthi', 24 , "shouthi@gmail.com", "MH", "F"),
values (4, 'Akasha', 23, 'Akasha@gmail.com' 'MH', "F");

② update :- Modify the existing Data

Syntax :- update table name
set column1 = value1
where condition;

```
Update Students
Set age = 21
Where id = 1;
```

③ Delete → used to Remove rows from a table.

Syntax → Delete from table name
Where condition;

```
Delete from Students
Where id = 3;
```

Commands	Removes	Keep Structure	Can use where
Delete	Selected Rows	Yes	Yes
Truncate	All rows	Yes	NO
Drop	Table + data	NO	NO

3. DQL (Data Query Language)
→ used only to ~~Retrive~~ (fetch) Data.

4. DCL (Data control Language)
Deals with permission & acces

1. Grant → Give permission
2. Revoke → Take Back permission

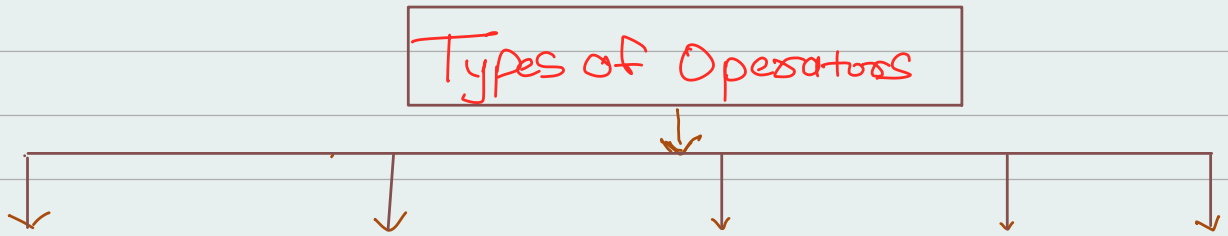
5. TCL (Transaction control Language)
Deals with Transactions (a set of SQL queries executed as one unit)

1. Commit → Save permission
2. Rollback → undo changes

Operators

Operators are special symbols / keywords used SQL queries to perform operations on data.

Types of Operators



Arthmetics Operator

+, −, *, /, %

Comparision / Relational operator

=, !=/<>, >, <, >=, <=

Logical Operator

And or NOT

Special Operators

IN, Like, isNULL/isNOT NULL, Between, Exists

Set Operator

Union, Union all, Intersect, Except / Minus

SQL Constraints

What is Constraints?

Constraints in SQL are Rules applied to column in a table.

→ They make sure that the Data stored in database is accurate, reliable & meaningful

Imagin a school Database ;

We have two tables :

1. Students → Stores Students details
2. departments → Stores department details.

But Here are the challanges :-

① How do we make sure each student has unique id?

② How do we stop duplicate email or phone no?

③ How do we ensure a student's percentage : never more than 100?

This is where constraints comes in :-

Constraints are like Rules that the database follows to keep d — accurate, valid and meaningful.

Data for Constraints.

① department table

dept_id	department name

2. Students table

Stud_id	Name	gender	dob	email	phone no	percentage	dep-id

Types of Constraints

1. Not Null
2. Unique
3. Check
4. Default
5. Primary key
6. Foreign key.

① **Not Null:** A record cannot have NULL values.

Prevents blank entries.

Why do we need it?

→ Mandatory field (eg. Student name, DoB)

→ Data completness.

eg:- A student must always have a name & date of Birth

```
Create table Students (
    id int
    Name varchar(100) Not NULL,
    dob Date Not NULL,
    );
```

② Unique:- We cannot have duplicate values.
NULL values are allowed

Why do we need it?
→ Maintain Uniqueness in data like email phone_no,
→ Supports authentication.
eg:- No two students can register with same email.

```
Create table Students (
  id int,
  email varchar(100) Unique );
```

③ Auto Increment
→ Automatically increase numeric value by 1 whenever a new record is inserted.
→ usually applied on primary key.

Why do we need it?
→ Automatic ID Generation
→ Avoid Humain Error

④ Primary Key:-

A primary key uniquely identifies each row in
It automatically combines two Rules :-

Not NULL + Unique.

Why do we need it?

① Unique Identification
② Data Integrity
③ Relationship
④ Performance.

	Not NULL	Unique	Primary Key
NULL	✗	✓	✗
Duplicate	✓	✗	✗

Syntax:- id primary key

```
Create table students (
        id int Primarykey key Auto-increment,
        Name varchar (100));
```

⑤ Foreign Key

A foreign keys links one table to another
It refers to the primary key of another table.

Why do we need it?

① Maintain Data integrity
② Build Relationships
③ Prevents Invalid Data

— . Foreign key (id) Reference departments (id);

What is a Key?

Create few Rules for table.

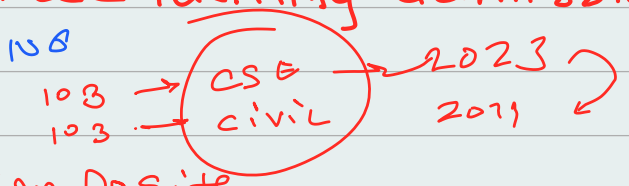
Can we identify a student's admission Just by Studen_id?

NO

2 → 2019

2 → 2024

Can we identify admission just by department_id?



Composite

→ we create a id by combine one or more column.

→ duplicate & Non-Null

Uni

Student_id	department_id	year
1	101 → CS	2019

Difference b/w Delete, Truncate & Drop

Feature	Delete	Truncate	Drop
Defination	Remove al selected rows from table	Remove all Rows from table	Remove the entire table
Where clause	Allowed	Not allowed	Not applicable
Data Removed	Selected Rows	All Rows	Data + Table Structure
Table structure	Remain	Remain	Removed
Speed	Slow	faster	faster
Auto Increment	No	Yes	Not appliable

Key Type	Definition	Rules / Properties	Example
Primary Key	A column (or set of columns) that **uniquely identifies each row** in a table.	- Only **one** primary key per table. - Cannot have `NULL` values. - Automatically **unique**.	`student_id` in **Students** table
Unique Key	Ensures that the column values are **unique**, but it's **not the main identifier**.	- A table can have **multiple** unique keys. - Can have **one NULL** (in most DBs).	`email` in **Students** table
Foreign Key	A column that **links two tables**. It refers to the **Primary Key** in another table.	- Used for **relationships**. - Can have duplicate values. - Can have NULL (if optional).	`department_id` in **Students** → references `department_id` in **Departments**
Composite Key	A **Primary Key made up of 2 or more columns**.	- Combination must be unique. - Useful when no single column is unique.	`(student_id, course_id)` in **Enrollments** table

Clauses

Imagine I am a principal of college
→ Show me the names of all the Students from cs department who scored above 80, arranged by their percentages
Can we just say this one word like Students!?
→ No
The Database needs step by step instructions.

What is clause?

A clause is like an instructions or keyword in SQL that tell the database what to do we the Data.

Each clause plays a role in Building the full instructions

Type of clauses

1. Select
2. From
3. Where
4. Group by
5. Having
6. Order By
7. Limit
8. Offset
9. Set clause

} Order of writing SQL Query.

Department table

PK

department_id	department_name
5	Civil
1	Computer Science
3	Electronics
2	Information Technology
4	Mechanical

Students Table

PK

FK

student_id	full_name	gender	dob	email	phone_number	percentage	total_fees	fees_paid	department_id
1	Deepraj Patil	Male	1998-09-07	deepraj@gmail.com	7689065437	90.78	100000.00	95000.00	1
2	Vivek Patil	Male	2019-09-09	vivek@gmail.com	8756789534	90.67	90000.00	85000.00	2
3	Rahul Sunchu	Male	2018-06-04	rahul@gmail.com	7856789504	98.67	95000.00	95000.00	1
4	Shivraj Patil	Male	2017-05-01	shivraj@gmail.com	9076789534	80.67	88000.00	80000.00	3
5	Shruthi Deshmukh	Female	2017-03-01	shruthi@gmail.com	7864678950	98.67	87000.00	87000.00	2
6	Vaishnavi Kulkarni	Female	2019-06-01	vaishnavi@gmail.com	9090789534	80.67	88000.00	85000.00	4

① Can you please provide students Names who scored more than 90 percentage

② Can you please give me only Male candidate data.

① Select → Specific Column to Retrive/fretch

Syntax :- Select col1, col2 From table_name;

Select * from table_name;

↳ Retrive all Rows x Columns

Select a table

② From → specifies the table(s) to Retrive the data From.

Syntax :- Select * from table name.

③ Where :- Filter the Rows Based on Condition.

Syntax :- Select * from table_name

Where condition ;

Types of Operators

Arithmetics Operator	Comparision/Relational operator	Logical Operator	Special Operator	Set Operator
+, −, *, /, %	=, !=/<>, >, <, >=, <=	And or NOT	IN, Like, isNULL/isNOT NULL, Between, Exists	Union, Union all, Intersect, Except / Minus

④ Order By :- Sort results set by one or more columns.

Syntax :- Select * from table nam

Order By col1 [ASC, DSC] ;

Where clause conditions

```
Select * from Students
        Gender = "Male"
And     percentage >85
And     department_id != 3
And     dob >= '2018-01-01'
And     full_name like 'VY.'
And     email like 'y.gmail.com'
And     department_id in (1,2,4)
And     departmentid not in (5,6)
And     percentage Between 90 And 100
And     phone number is Not NULL ;
```

Group By → It is powerful function used to group rows from a table based on values of one or more columns.
It is often use in conjunction with Aggregate functions.

student_id	full_name	gender	dob	email	phone_number	percentage	total_fees	fees_paid	department_id
1	Deepraj Patil	Male	1998-09-07	deepraj@gmail.com	7689065437	90.78	100000.00	95000.00	1
2	Vivek Patil	Male	2019-09-09	vivek@gmail.com	8756789534	90.67	90000.00	85000.00	2
3	Rahul Sunchu	Male	2018-06-04	rahul@gmail.com	7856789504	98.67	95000.00	95000.00	1
4	Shivraj Patil	Male	2017-05-01	shivraj@gmail.com	9076789534	80.67	88000.00	80000.00	3
5	Shruthi Deshmukh	Female	2017-03-01	shruthi@gmail.com	7864678950	98.67	87000.00	87000.00	2
6	Vaishnavi Kulkarni	Female	2019-06-01	vaishnavi@gmail.com	9090789534	80.67	88000.00	85000.00	4

Execution of Groupby

1. Split
2. Apply
3. Combine

Agg
- → Sum
- → Count
- → Avg
- → max
- → min

1. Split → mf
2. Apply → count
3. Combine

↑ Categorical / distict

```
Select Gender, Count(Gender)
From Students
Group by Gender;
```

↳ Aggregation fu

Alias ↑

```
Select Gender, max(percentage) as max_percentage
From Students
Group by Gender;
```

Min, mean, Count, sum.

```
Select department_id, Gender, max(percentage) as max_per
From Students
Where Gender = 'male' and Fees_paid > 90,000
Groupby department_id, Gender
```

Having :- It will Help us to filter a Data based on Groupby.

```
Select Gender, Count(Gender) as Gender_count
from student
Group by Gender
Having Count(Gender) > 4;
```

Where	Having
→ Filter a Data based on Column	Filter a Data based on agg fun
→ Before Groupby	After Group by

Limit

SQL Limit in a SQL clause used to restrict the No of Rows Returning by query.

It is particularly useful when you want to control the size of your result set to avoid excessive data Retrival.

When to use Limit ?

You would use the SQL Limit Clause when you need to :

1. Retrive only specific No of Result Set.
2. Implement pagination for displaying data in chunks.
3. Optimize query performance by reducing the amount of Data transferred.

Syntax:-

```
Select column1, column2, ...
From table
Limit number_of_rows;
```

```
Select * from Students
Limit 5;
```

```
Select * from Students
order by percentage
limit 5;
```

Order of writing SQL Query

Select
Distict
From
Join
Where
Group by
Having
Order by
Limit / offset

Order of Execution SQL Query

From
Join
Where
Group By
Having
Select
Distict
Order By
Limit / offs

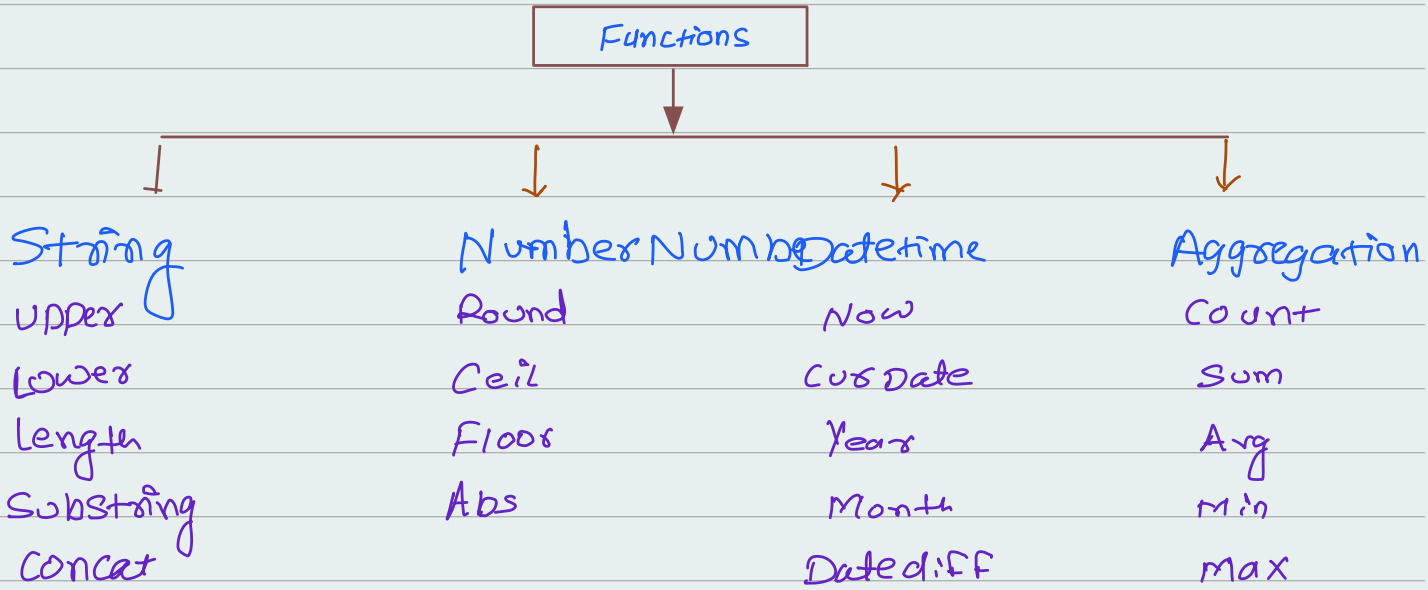
SQL Functions

SQL functions are Built in operations in sql that takes input, perform & actions & Return a value

Two main types:

① Single row (scalar) function → work on each individually

② Group (Aggregation) function → work on set of Rows & Return a Single value



Revisit

Constraints → Apply a Rules → Database / Table

(1) unique → No duplicate

(2) Not Null → Null / missing values

(3) Primary key → Unique + Not Null

(4) Foreign Key → Unique + Duplicate values
↳ Connect with another table
↳ Reference of primary key.

(5) Composite → Two or more colums → unique colm

Clauses → Step by step instructions to add or table to get output.

(1) select (2) from

Sta

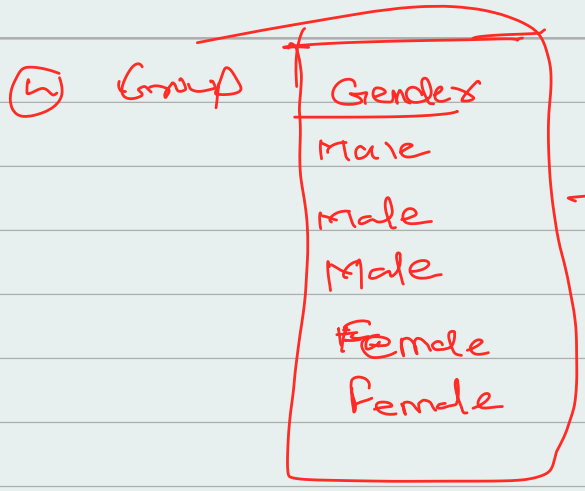
(2) where

Select fullname from Students
where Maths > 10;

<, >=, <=, =, !=

Between, in, not in, like

→ and or not



① split → [M, F]

② Apply → count

③ combine

M 3
F 2

Male → 3 [M1, M2, M3]
Female → 2 [F1, F2]

Mde 3
Female

Gender	Marks	Dept
M	5	CSE
M	4	civil
M	5	CSE
F	6	civil
F	6	CSE

Student

```
Select Gender, max(Marks) as Max_marks
From Student
Groupby Gender ;
```

↳ Split M F

Apply M [5, 5, 4] } Max
F [6, 6]

Combine { M 5
F 6

Mde 3

Female

Student

Gender	Marks	Dept
M	5	CSE
M	4	civil
M	5	CSE
F	6	civil
F	6	CSE

Dept, Gender max(Marks)

```
Select Dept, Gender, max(Marks)
from student
group Dept, Gender;
```

```
Select Dept, Gender, max(Marks)
from student
where Gender = 'M'
group Dept, Gender
(having max(Marks) ≥ 5);
```

Relationship

When we store Data in single table

→ Data duplication → names were repeated
→ Data inconsistency
→ Data updation
→ Handling missing values

we have spited the Data into

1. Students
2. Teachers
3. subject
4. Reference table

—

What is Relational Data base?

→ Rows columns
↳ Tables

Relationship

→ Connecting one two or tables together

Types of Relationship

1. one to one
2. one to many
3. many to many

1. one to one Relationship

When Table A record is connect with Record in table B is called one to one.

1. User → Account SBI
2. Student → Id
3. Kukatpally → Pin code.

one to Many

Where one Record in table A is connecting with multiple Records in TaBble B-

One teacher → DA
→ DS

Deepak → SQL
→ PBI
→ Excel
→ EDA
} DA

Student → DA
→ DS

Many to One

Student1 → One teacher
Student2 → One teacher
Student3 → One teacher

4139 → Deepay
410 → Deepay
421 → Deepay

Many to Many

- When a many Records in table A connect with many Records in table B

Students ↔ Courses

Students ↔ Subjects

Joins

What is Joins?

A Joins is used to Combine data from two ore more tables based on a Related colum Between them

Why we need Joins?

1. Data Duplication / Redudancy.
2. Data updation
3. Storage
4. Data consistency.

Types of Joins

1. Inner Join / Join
2. Left Join
3. Right Join
4. Cross Join
5. Self Join

Apply a Rules.

Primary → Unique + Not null

Foreign key → Unique

↓

Connect with Second table / another table.

Inner Join

Return only Rows/Records with matching values in Both table.

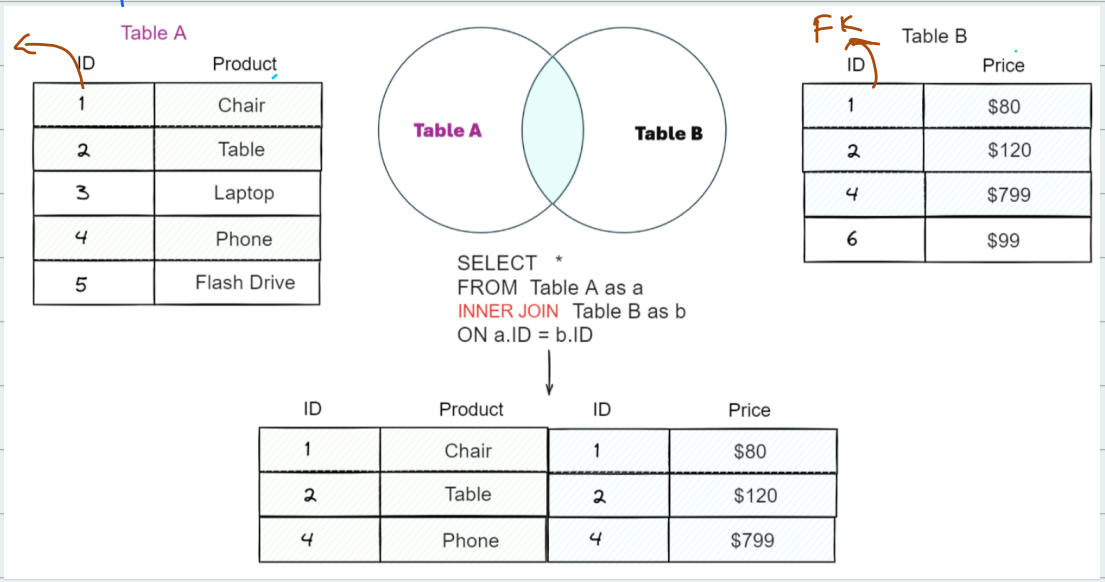
or

The Result of the Inner join is only the Common Values between 2 tables

Product

Price

PK



→ The inner join is fetching a data using primary key & foreign key which are common Records.

Query :-

```
Select * from product as p
Join / innerjoin price as Pr
on p.id = pr.id;
```

- ```
  Select p.product, pr.price from product P
  Join price as pr
  On p.id = pr.id
  ```
  ```

Left Join

Return all the Rows from the table, and matched Rows from the Right table. Unmatched rows from table shows as NULL.

Table A

ID	Product
1	Chair
2	Table
3	Laptop
4	Phone
5	Flash Drive

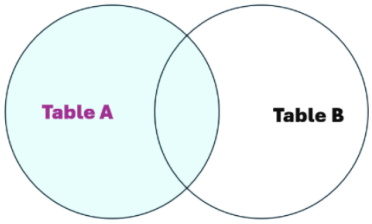


Table B

ID	Price
1	$80
2	$120
4	$799
6	$99

```
SELECT *
FROM Table A as a
LEFT JOIN Table B as b
ON a.ID = b.ID
```

ID	Product	ID	Price
1	Chair	1	$80
2	Table	2	$120
3	Laptop	NULL	NULL
4	Phone	4	$799
5	Flash Drive	NULL	NULL

```
Select * from product as P
Left Join price as Pr
  On p.id = pr.id ;

Select p.id, P.product, Pr.price
from product as p
Left Join price as pr
  on p.id = pr.id ;
```

Right Join

Return all Rows from Right table, and matched Rows from Left table

unmatched rows from the Left show as NULL.

Table A

ID	Product
1	Chair
2	Table
3	Laptop
4	Phone
5	Flash Drive

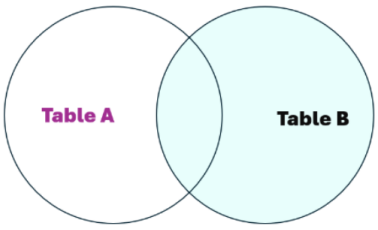


Table B

ID	Price
1	$80
2	$120
4	$799
6	$99

```
SELECT *
FROM Table A as a
RIGHT JOIN Table B as b
ON a.ID = b.ID
```

↓

ID	Price	ID	Product
1	$80	1	Chair
2	$120	2	Table
4	$799	4	Phone
6	$99	NULL	NULL

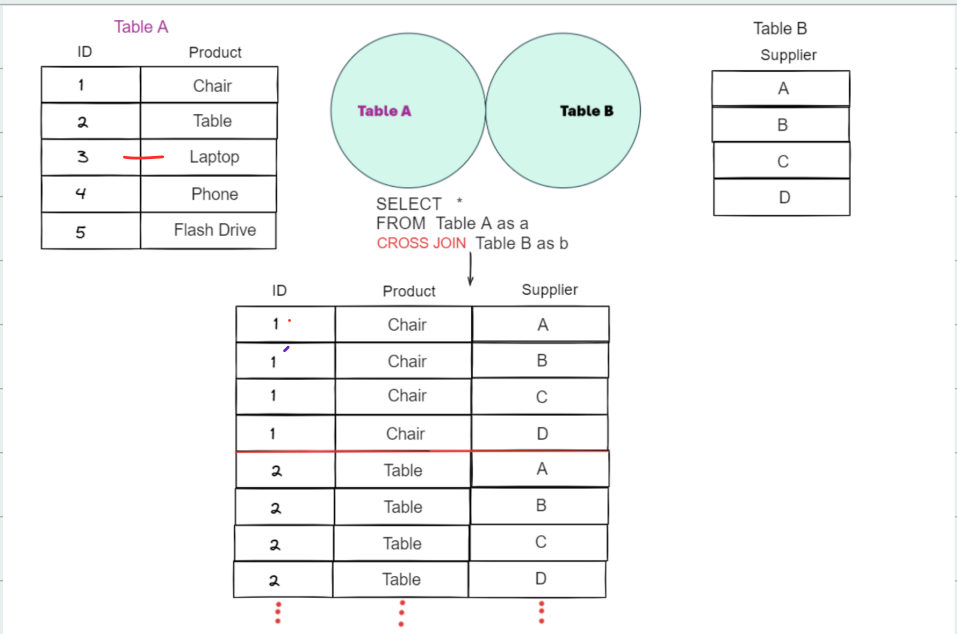
```
Select * from product as P
Right Join price as Pr
    On p.id = pr.id.
```

```
Select p.id, P.product, pr.price
from product as p
  Right Join price as pr
  on p.id = pr.id;
```

Cross Join :-

Return a Cartesian product - every combination of Rows from both table.

Cross tab



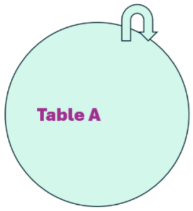
Self Join

A Self Join is a regular Join where a table is Joined with itself.

It is useful for comparing rows within the same table - especially relational structure like employees & Managers, products & categories or friends & followers.

Table A

ID	Employee	Manager_ID
1	Hang	Null
2	Connie	1
3	Le	1
4	Afnan	2
5	Nitesh	3




```
SELECT a1.ID as Employee_ID, a1.Employee
as Employee, a1.Manager_ID,
a2.ID, a2.Employee as Manager
FROM  Table A as a1
LEFT JOIN   Table A as a2
ON a1.ManagerID= a2.ID
```

a1.ID (Employee_ID)	a1.Employee (Employee)	a1.Manager_ID	a2.ID (Manager_ID)	a2.Employee (Manager)
1	Hang	Null	Null	Null
2	Connie	1	1	Hang
3	Le	1	1	Hang
4	Afnan	2	2	Connie
5	Nitesh	3	3	Le

Set Operations in SQL

Set operations are used to combine the results of two or more Select queries.

This work similarly to set operations in mathematics (like union, intersection, difference).

Types of Set operators.

1. Union
2. Unionall

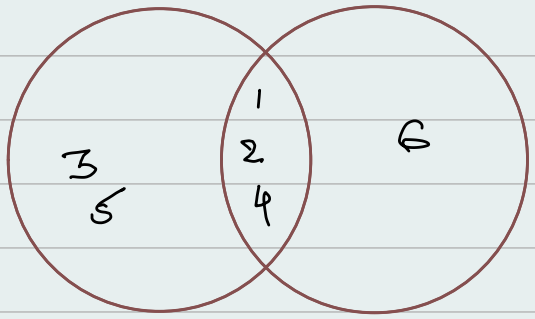
1. Union :-

Union in SQL combines the results of two or more Select statements into a single Result set, Removing duplicate rows to ensure that each row is unique.

Products

Table A

ID	Product
1	Chair
2	Table
3	Laptop
4	Phone
5	Flash Drive



Price

Table B

ID	Price
1	$80
2	$120
4	$799
6	$99

```
Select id
From Products
union
select id from prices.
```

1
2
3
4
5
6

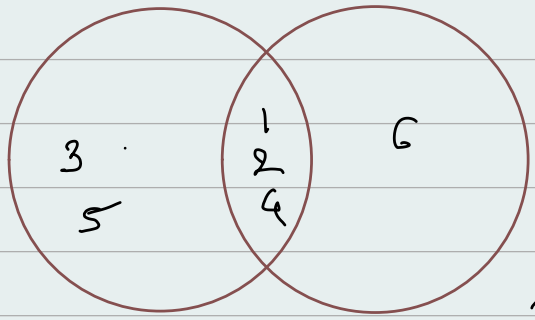
(1) Union all

Union all in SQL combines the results of two or more Select statements into a single Result Set with duplicate values.

Products

Table A

ID	Product
1	Chair
2	Table
3	Laptop
4	Phone
5	Flash Drive



Pri

Table B

ID	Price
1	$80
2	$120
4	$799
6	$99

1
2
3
4
5
1
2
4
6

Union	Union all
Remove the duplicates (Only unique rows are returned)	Keep duplicates (all rows from both queries)
Slower Because SQL must check & Remove duplicates	Faster, since No duplicates Check is done
we use when we want unique Rows	we use when we want all Records.
{ 1, 2, 3, 4	{ 1, 2, 4, 1, 3, 4 } .

1. Joins
2. Grouping
3. Subquery
4. Execution / writing
5. CTE
6. Windows
7. Commands
5. Datatypes

SubQuery

→ A Subquery, also known Inner query, It is a query within another SQL Statement

→ It allows us to use the result of one query input in another query.

→ Subquery will be enclosed in parenthesis '( )' & Can be used in various parts of SQL Statements including select, from, where.

→ For Subquery we use operators
- ↳ Comparative
- ↳ in, not in

Why to use Subquery?

→ Retrieved a data based on a Results of another query: use the output of one query as a filter or Criteria of a another query.

→ Breaking down complex problem into smaller
- → more Redable & manage

Syntax

outer query

```
Select Col1, Col2, Col2
From table
Where Col operator ( Select Column From
                     another_table where cond^n);
```

inner query

Col, Col2. → The colums which we want to retrieve

table → Main table/ outer table

Col → The column you want to filter or compare

operator → >, <, >=, <=, in, Not in etc

name	Marks
A	10
B	20
C	10
I	15

What is ~~the~~ max maximum marks of a student.

Select max(marks) from Student;

↳ maxium max marks of a students

Select name
from Student
where marks = (Select max(marks) from Student);

↑ first inner query will run

(20)

name	marks		
A	10	= 20	X
B	20	= 20	✓
C	10	= 20	X

Or 15 = 20 X

B

outer query will iterate will we get a Result/ file

Types of Sub Query

1. Single subquery
2. Multi subquery (in, all, any)
3. Correlated subquer

1. Singk query

A query inside another query that returns one Result set (single colom or single row)

2. Multi-Row Subquery (in, Any, All)

1. Subquery with IN

Check if a value matches any value in a list

2. Subquery with Any

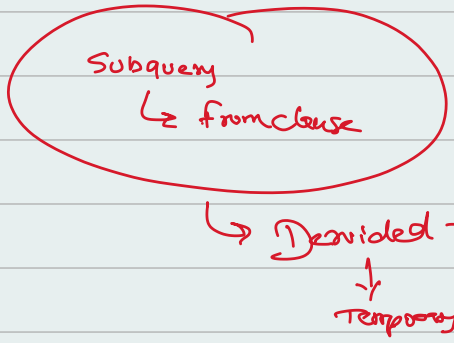
Condition is true if it matches any value returned by subquery

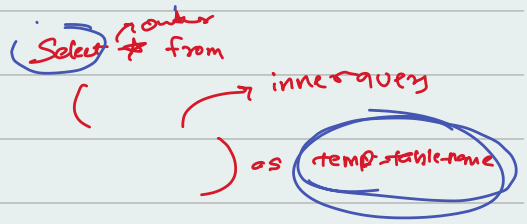
3. Subquery in All

Condition must be true for all values in Subquery.

3. Correlated Subquery

A correlated subquery depends on the outer query.
It Runs Row by Row.





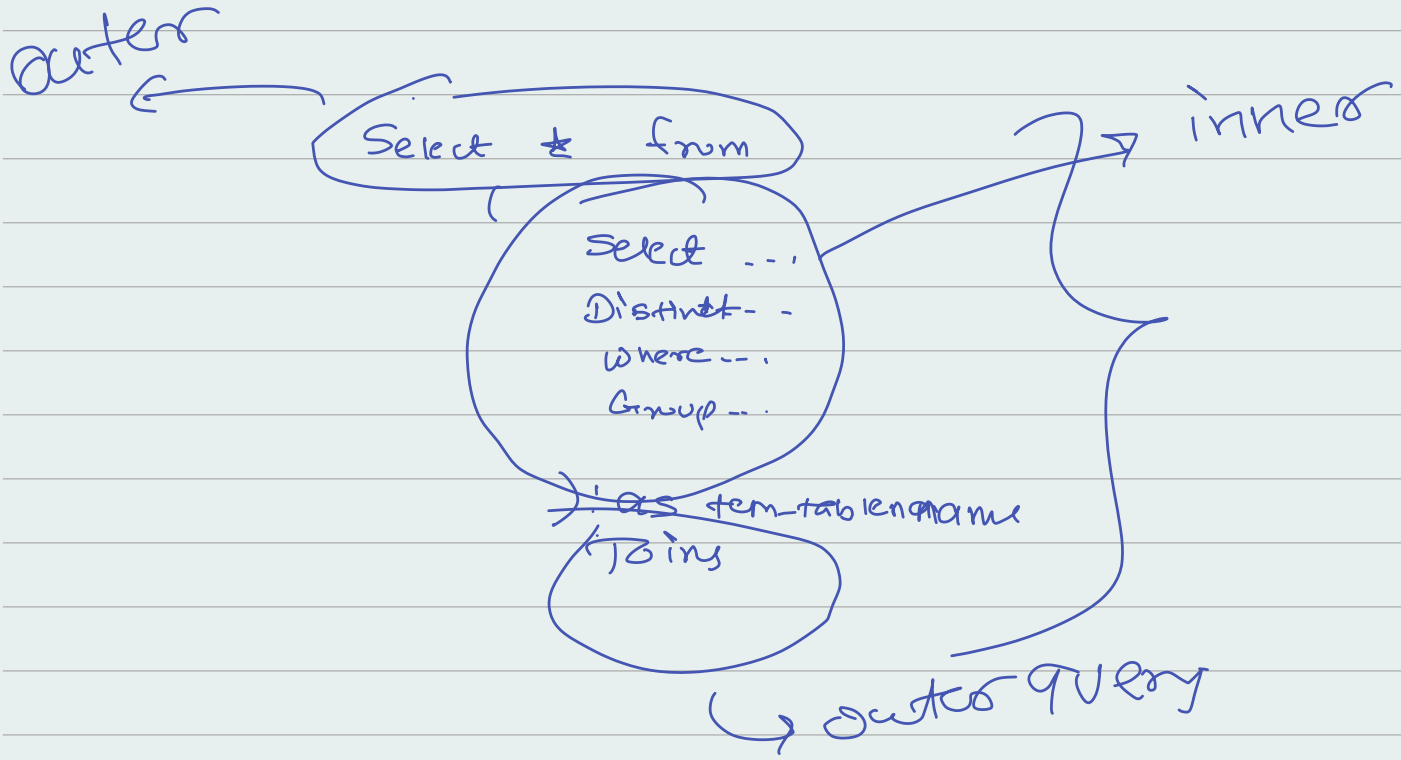
Derived Table.

A derived table is a just a subquery inside the from clause It behaves like a temporary table that only exists while the query Runs.

→ It must have alias (name).

Syntax :-

```
Select Col1, Col2
From (
        Select ...      - subquery
        From table name
) as temp table ;  → alias is requir
```



CTE

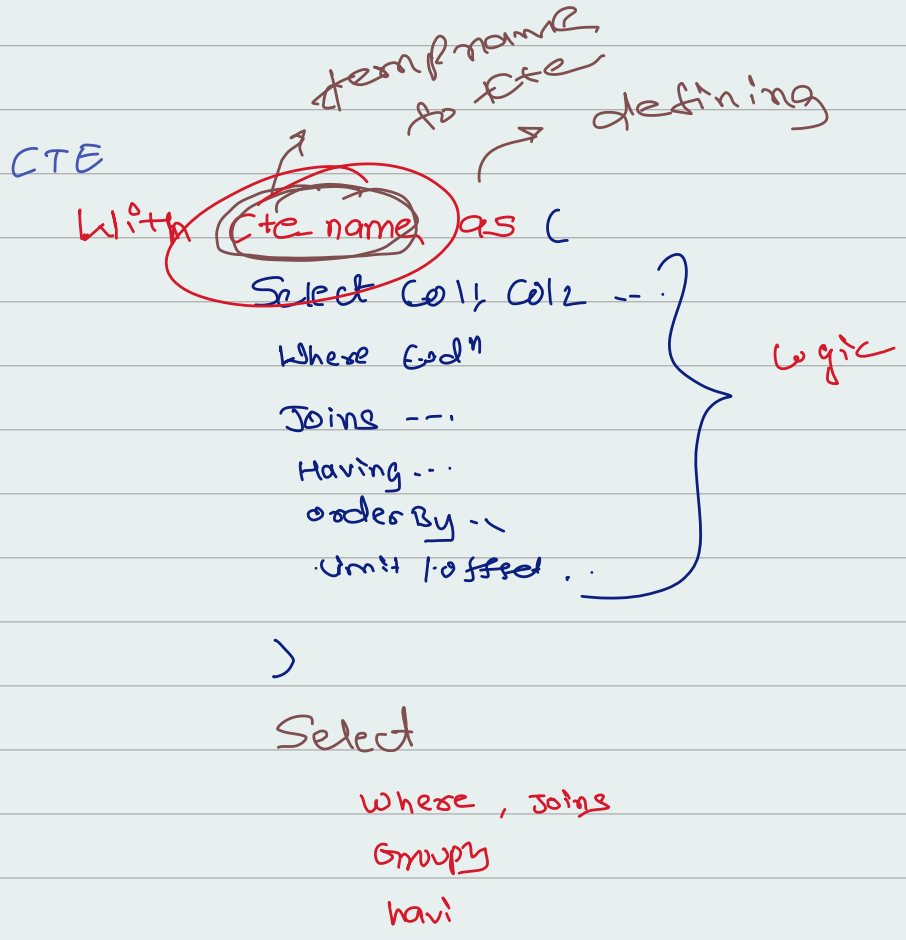
Common Table Expression -

A Common table Expression (CTE) in SQL is a temporary result set that can be referenced within a Select, Insert, Update, Delete Statement

CTE are defined using WITH keyword & allows you to create a named, reusable Subquery within SQL Statement.

Why to use CTE ?

1. Simplifying complex queries by breaking them into Smaller Reusable parts.
2. Reusable Subqueries within a single SQL Statement to avoid duplicates.
3. Organizing SQL statements better Readibility & maintainability.



```
With Customers_cte AS
(
Select customer_id, first_name
last_name
From customers
)
Select * from customers_cte
```

Views

A view is a virtual table based on Select query.
It does not store any physical Data.
You can Select, filter, aggregate and Join tables inside a view

Use case:

1. Simplify Complex queries
2. Enchance Security
3. Maintain consistency.

Syntax

```
Create view view_name as
  Select column1, column2, ....
  From table_name
  Where condition ;
```