**A COMPLETE GUIDE ON SQL DATABASE**

Database: collection of our data at a place

1. **Relational Database:** Structured Data – data is stored in the form of rows and columns.

* MySQL
* SQL
* PostgreSQL

1. **NoSQL Database:** Data is not structured.

eg. JSON files: {key: value} = unstructured.

* MongoDB
* Cassandra
* HBASE

**CRUD OPERATION: CREATE, READ (Select), UPDATE(update for records, alter for schema) , DELETE.**

Commands**:**

**Notes: if u want to run multiple commands at once, then use semicolon (;).**

CREATE DATABASE techforallwithpriya; // for creating a new database.

if database if already existing then use,

CREATE DATABASE IF NOT EXISTS techforallwithpriya;

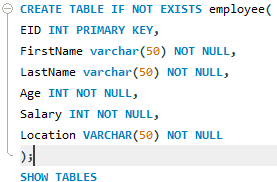
If u want to start working with any specific database: use the command

USE techforallwithpriya;

Datatypes in SQL:

Int – numeric data

varchar(20) – textual string. // 20 is the length of the string.

**How to create table inside a database:**

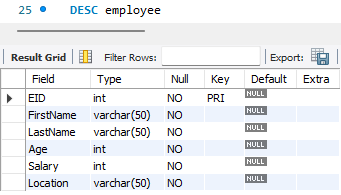
primary key is a constraint and it is used for unique identification and by default it is not null.

primary key(EID) = another way to declare primary key.

NOT NULL 🡺 it tells that this is a mandatory field.

If you want to see the entire schema of the table then use keyword DESC table\_name

eg,

DESC employee

* Inserting data in the table (manually);



Notes: if we use AUTO\_INCREMENT in the schema, then it will start with one and with each insertion it will increment the value automatically. Also we can set our own preferred starting value.

* To delete the table from the database.

**DROP TABLE table\_name;**

eg,

**Note: while inserting value in the table, if you are writing less than the no of field available then make sure to provide in which manner you are writing the values.**

**eg-**

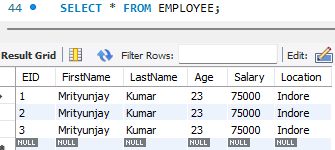
****

* Command to read the table : select \* from table\_name

**eg:**

****

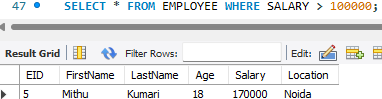
**table:**

****

* **Command used for filtering operation in sql**

SELECT \* FROM employee WHERE salary > 100000

eg:

how the flow of code will work here:

1. FROM

2. WHERE (this will filter the data based on the input.)

3. SELECT

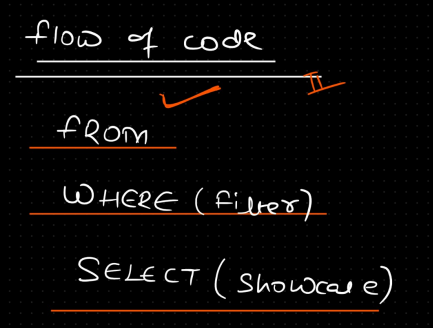
Notes: Difference between update command and alter command

**Update:** update the record of the table. e.g- content of the table.

And it is called dml: data manipulation language

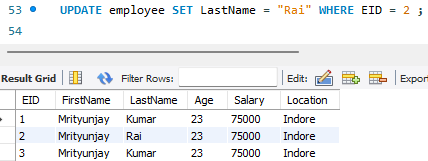
**Alter:** it is used to update the schema of the table. **eg,**

And it is ddl command: data definition language.

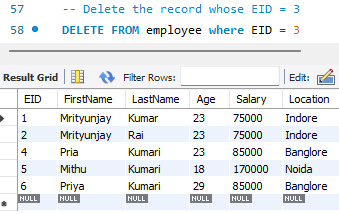
* **Flow of the code.**

****

* Command used for updating the records of the table.

use **primary key** in this process, it increases the readability and it also matches with the new practices.

* Command used for **deleting records** of the table



* **Differentiate between primary and unique** (it is a constraint**)**

primary key = unique, not null

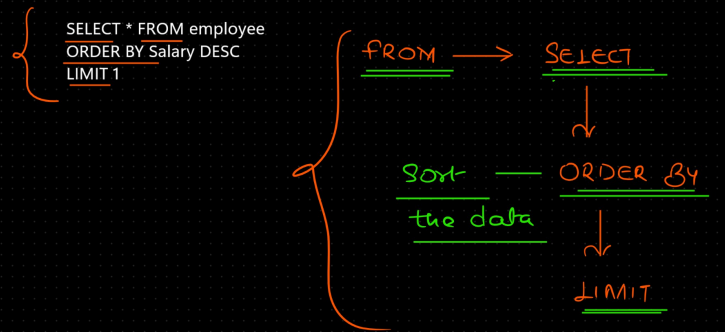
unique key = unique but it can be null

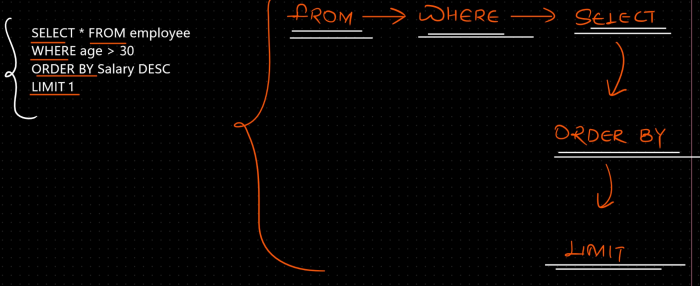
* **Foreign key 🡺 it is used for referencing other table in the database.**

It is used for linking two table.

It can be more than one in a table.



* **Sort the data points**
* Use **ORDER BY** – this sorts the data in ascending order.
* Use **ORDER BY DESC** – this sorts the data in descending order.
* **Flow of code:**

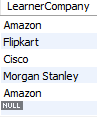
****

* count(\*) == it is trying to count all the rows record. And filter out based on the condition

 as is used to rename the column with readable name.

notes:

count will count all the no null enteries.

List:

SELECT COUNT( LearnerCompany) FROM LEARNER;

ans: 5;

but if you want, no duplicate entry then use DISTINCT keyword.

SELECT COUNT(DISTINCT LearnerCompany) FROM LEARNER;

ans: 4

* wild card or say pattern matching

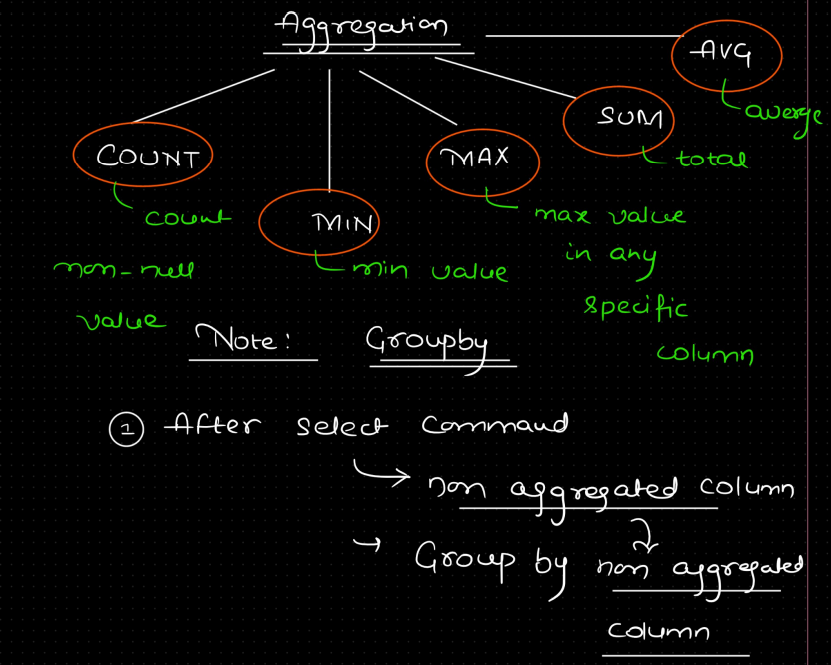
Using the LIKE keyword we can do this.

this is used to match a patten that look similar to this.

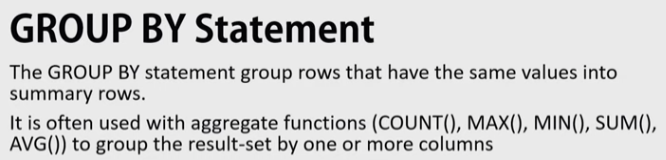
eg.,

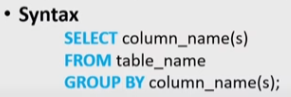


Ans: as date is written in ‘2024-01-29’ in sql we can use LIKE to match with the sql format.

Aggregation

* **GroupBy statement.**



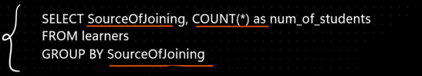
Syntax:

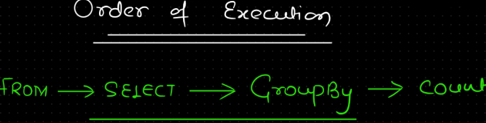
Eg:



Note: GroupBy can only be used by aggegate function. Also, if there are two column after select statement, then you have to use both the column after GroupBy as well.

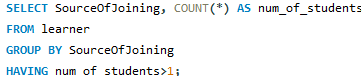
* Order of execution of commands:



explanation: first from as it will fetch on which table it has to work and then select which tells on what to work and then groupBy as count(\*) will count after the grouping is done.

NOTE: if filteration is required on aggregation HAVING clause

Eg.



Note: never use WHERE clause after using GROUP BY.

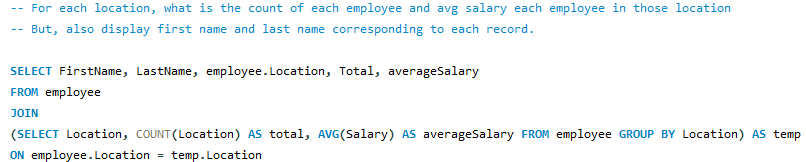
**Logical Operator in SQL**

**AND, OR, NOT, BETWEEN.**

* **ALTER COMMAND IN SQL – it is a DDL command.**

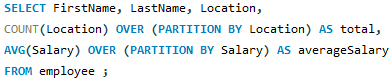
**i.e., it is used for changing the scheme of the table.**

 **note:** PRIMARY KEY should not have AUTO INCREMENT property elsewise, it will not drop. Coz, AUTO INCREMENT property can only be given to PRIMARY KEY or UNIQUE KEY.

**NOTE: UPDATE & DELETE always works with primary key in safe mode.**

For the required result, we wanted, GROUP BY was alone not sufficient to perform the task, so we used join(default inner join) but joins are computationally high.

So, **PARTITION BY** came in picture, which allows to use non aggregated columns.



WINDOW FUNCTION – give extra time. You have to use OVER

1. Row\_number() vs
2. Rank() vs
3. DenseRank()

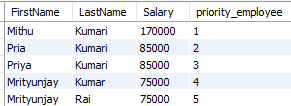
* Priority Setting, I want to sort the data as per the highest salary.

ROW\_NUMBER() – map the values with number on the basis of sorted data

So, always use ORDER BY with ROW\_NUMBER()

This provides unique priority to each employee even if their salary is same. It decides priority if same salary then it uses who ever comes first in the company.

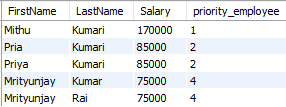
****

**Output:**

**To solve this issue, RANK() came into picture,**

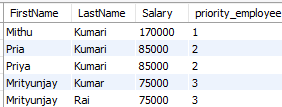
****

**Output:**

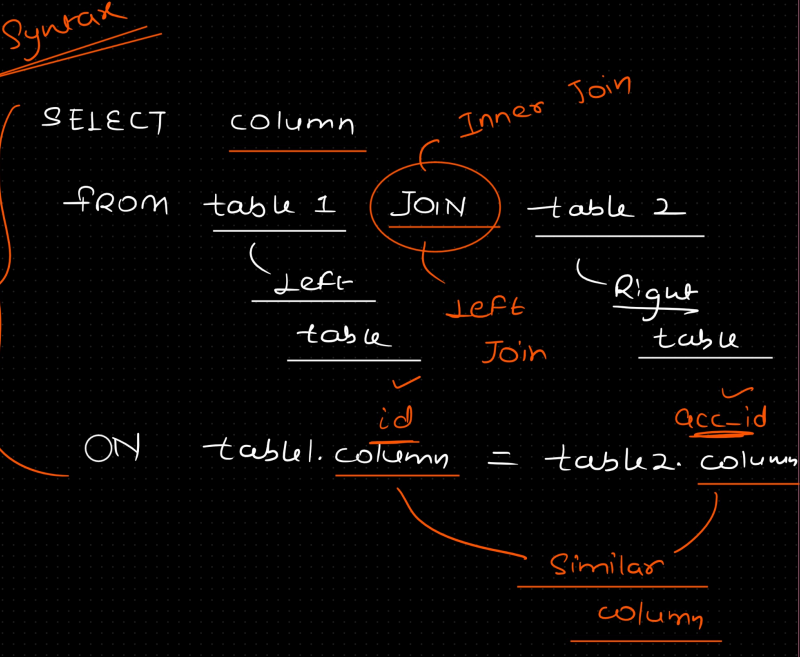
** This also has problem, it internally skips 3 and 5.**

**To remove this issue, DENSE\_RANK() comes in picture.**

****

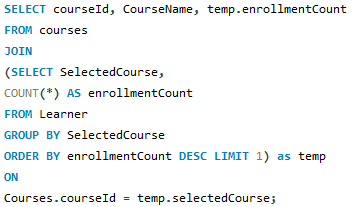
**Output:**

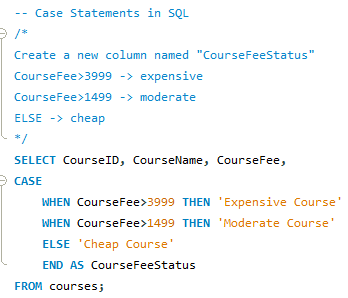
**NOTE: if there is no duplicate entry [here, salary two or more employee is same] then ROW\_NUMBER(), RANK(), DENSE\_RANK(), this will give same result.**

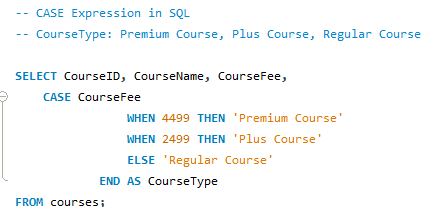
**JOIN AND SUBQUERIES**

**NOTE: IF YOU HAVE SUBQUERIES, THEN SUBQUERIES IS RESOLVED FIRST.**

**Few examples to demonstrate it better.**

****

**Case Statement in SQL**

**Case Expression in SQL**

**Common Table Expression**