**SQL PRACTISE**

**ALTER COMMMAND🡪**

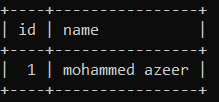
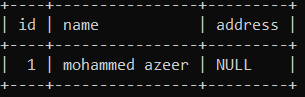
To Change structure of existing table alter command in used, with the help of alter command we can perform following operations

* **Adding extra columns into the table**

There is table name student with id and name if we need to add extra address column then we can add using alter command

**Syntax🡪 Alter** table (table\_name) **add** (new\_table\_name datatype);

**Keyword used🡪 Alter+add**

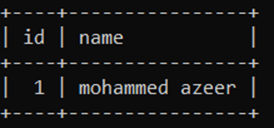
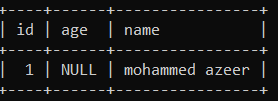
 ****

Alter table student add (address varchar(20));

To add column in a specific position use **after** key word, we will add age column after id in the student table

**Syntax🡪 Alter** table (table\_name) **add** (new\_table\_name datatype) **after** (column\_name);

**Keyword used🡪 Alter+add+after**



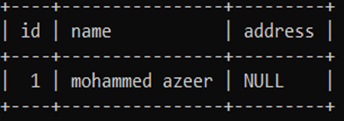
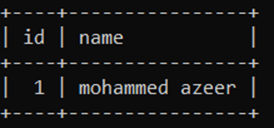
Alter table student add age int after id;

* **Removing a column from the table**

We will remove address column from the student table

**Syntax🡪** **Alter** table (table\_name) **drop column** (column\_name);

**Keyword used🡪 Alter+drop column**

Alter table student drop column address;

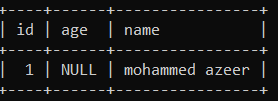
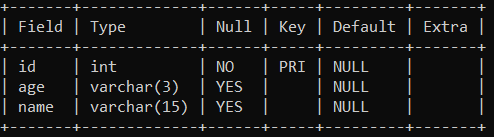
* **Modify the datatype of existing column**

to modify the datatype of a column in a table we use the key word modify

we will modify the age datatype to varchar(3)

**Syntax🡪** **Alter** table (table\_name) **modify** (column\_name with new datatype);

**Keyword used🡪 Alter+modify**

 ****

Alter table student modify age varchar(3);

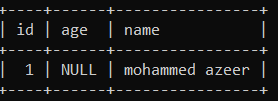
* **Renaming the column name to new column name**

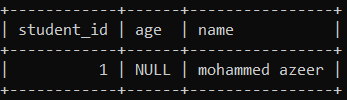
To change the column name make use of rename command to change the column name

We will rename the id to student id in the student table

**Syntax🡪** **Alter** table (table\_name) **rename column**(old column\_name) **to** (new column\_name);

**Keyword used🡪 Alter+rename+to**





Alter table student rename column id to student\_id;

Same for changing the table name

We will change table student to student\_details

**Syntax🡪** **Alter** table (table\_name) **rename to** (new table\_name);

**Keyword used🡪 Alter+rename+to**

**DIFFERENCE BETWEEN ALTER AND UPDATE COMMAND🡪**

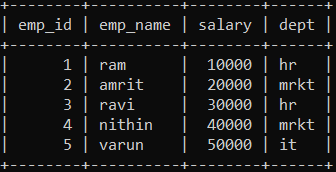
**Alter command🡪** it is a DDL command (data definition language), used to change the structure of the table

**Update command🡪** it is a DML command (data manipulation language), used to make changes only in the data inside the table

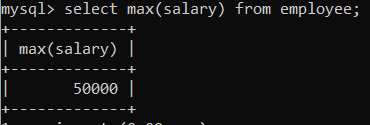
We cannot use alter command to change the data present in the table where as we cannot use update command to make changes in the structure of the table

**SQL QUERIES ON EMPLOYEE TABLE🡪**

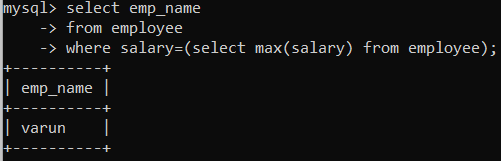
**Employee table**



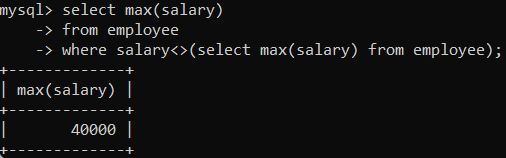
* **write a sql query to find the maximum salary from the employee table?**



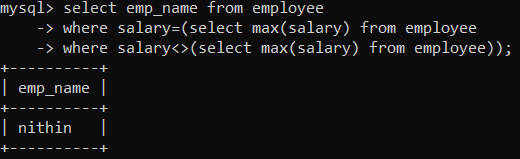
* **write a query to display employee name who is taking the maximum salary?**



* **write a sql query to display the second highest salary from the employee table?**

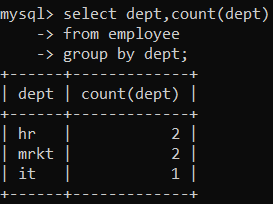
****

* **write a sql query to display the employees name who is taking the second highest salary from the employee table?**

****

In the place of ‘=’ we can use ‘in’ if there are multiple employees having second highest salaries

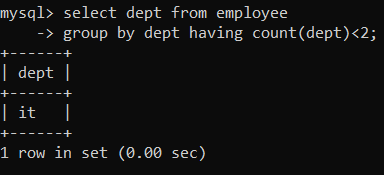
* **write a query to display all the department name along with the number of employees working in that?**



When we are using group by key word the select statement should always have the name which is used with group by, and we can make use of aggregate function along with it

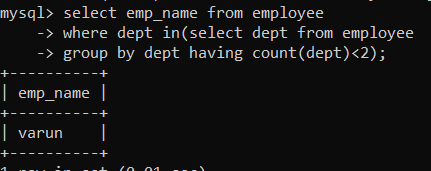
For eg🡪 here in the employee table we are using group by dept, so we should dept after select statement, along with this we can make use of aggregate functions like count, max, avg etc.

* **write a query to display all the depart name where number of employees are less than 2?**



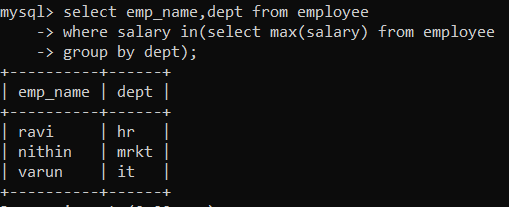
With group by we make using use having clause always

* **write a query to find the employee names who is working in the department where number of employees is lees than 2?**



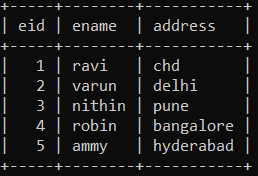
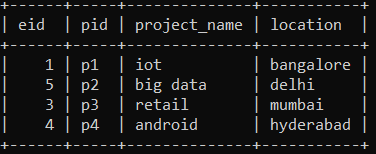
‘in’ key word in used instead of ‘=’ because there could be various employees in the table who are working in the department where number of employees are less than 2

* **write a query to display highest salary department wise and name of the employee who is taking that salary?**

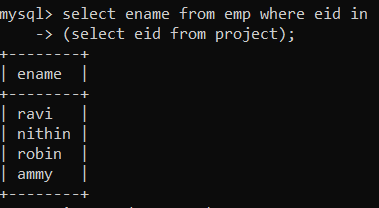


**IN AND NOT IN COMMAND🡪**

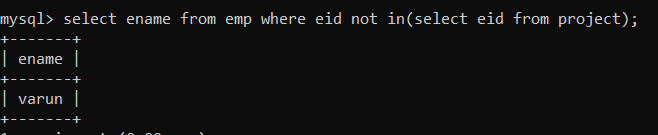
**Creating 2 tables emp and project, project has a foreign key which references to emp table**

** **

* **write a query to find the name of employees who are working on a project from emp and project table?**



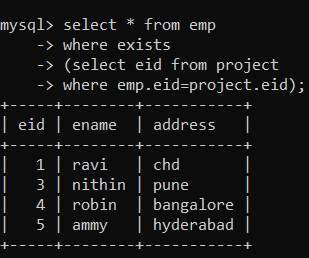
* **write a query to find the name of employees who are not working on a project from emp and project table?**

****

**EXIST AND NOT EXIST COMMAND🡪**

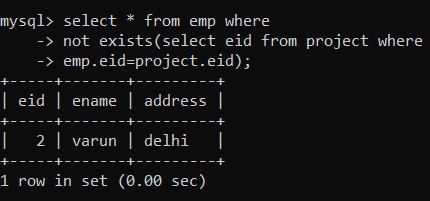
Exist and not exist works on corelated nested queries, in nested query we make use of in and not in where as in exist and not exist is used in corelated nested query

* **write a query to find the detail of employee who is working on at least one project?**

****

Exists is used to check each and every data from the other table, if the data is present it return true or else it return false

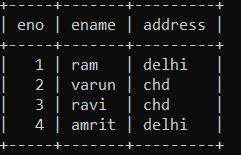
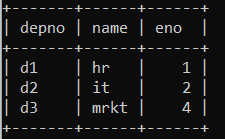
* **write a query to find the detail of employee who is not working on at least one project?**



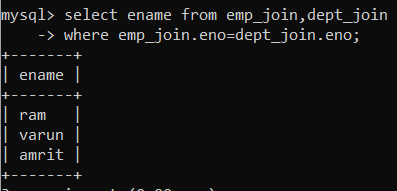
**TYPES OF JOINS🡪**

**NATURAL JOIN:**

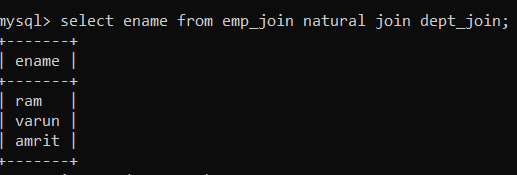
Natural join is used when we want to equalize common attributes between two tables

** **

* **write a sql query to find the employees name who is working in a department from emp\_join and dept\_join table?**

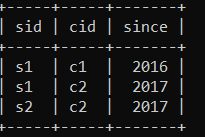


Using natiral join key word

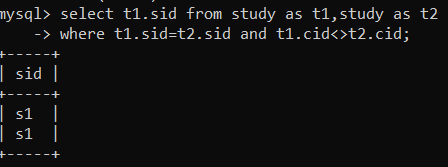


**SELF JOIN:**

**Study table🡪**

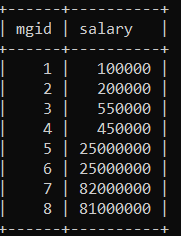
****

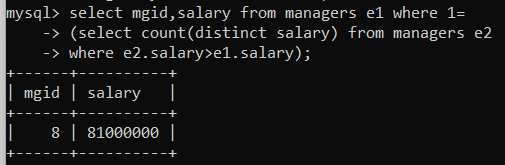
* **write a sql query to find the student id from study table who is enrolled in at least two courses?**

****

**FINDING N TH LARGEST SALARY FROM A TABLE🡪**

**Managers table with salaries🡪**

****

****

**Second highest salary**

**Coomon query🡪**

Select column\_name from table\_name e1 where n-1=(select count(distinct salary)from table\_name e2 where e2.salary>e1.salary);

**NORMALIZATION**

It is technique to remove or reduce redundancy (duplicacy) from a table, there two types of duplicacy row level and column level

**Row level duplicacy🡪**

When two or more rows are exactly same this results in row level duplicacy, we can eliminate the row level duplicacy using **primary key**

**Column level duplicacy🡪**

When two or more columns have same values this results in column level duplicacy, this results in **insertion anomaly, deletion anomaly and updation anomaly**

**FIRST NORMAL FORM**

The rule of 1NF is the table should not contain any multivalued attribute, for example a student can have two or more mobile numbers, two or more addresses and etc

|  |  |  |
| --- | --- | --- |
| **Roll no** | **name** | **Courses** |
| 1 | Sai | c/c++ |
| 2 | Harish | Java |
| 3 | omkar | c/dbms |

This table has multivalued attribute courses according to 1NF there shouldn’t be any multi valued attribute, this table can be represented as follows

|  |  |
| --- | --- |
| **Roll no** | **name** |
| 1 | Sai |
| 2 | Harish |
| 3 | omkar |

|  |  |
| --- | --- |
| **Roll no** | **courses** |
| 1 | C |
| 1 | C++ |
| 2 | Java |
| 3 | C |
| 3 | Dbms |

**Primary key: roll no**

**Primary key: roll no, courses (**composite key)

**Foreign key: roll no (**reference roll)

**SECOND NORMAL FORM**

Table should in 1NF and all the non prime attributes should be fully functional dependant on candidate key

**VIEWS IN SQL**

Virtual table, view is a result set of stored query, view can be read only view and updatable view, materialized views (updated version of view, making a copy of view from a server)

**Advantages of view🡪**

Restrict data access, to make complex query easy, to provide data independence, to present different views of the same data

**INDEXING IN SQL**

**Indexing** is used to speed up the searching operation using pointers, this reduces IO cost which required to search the data

Data is present in secondary memory(database) in form of blocks/pages indexing reduces the blocks which comes into the RAM for searching the data

**Types of indexes**

* primary index
* clustered index
* secondary index

|  |  |
| --- | --- |
| **Primary index** | **Clustered index** |
| **Secondary index** | **Secondary index** |

**Ordered file**

**Unordered file**

**Non key**

**key**

Ordered file🡪the data stored are in sorted order

Unordered file🡪the data stored are un sorted

Key🡪the table has unique values like primary key

Non key🡪the table doesn’t have any unique key

**Primary index🡪**

Index table is created in such way that a pointer is mapped to one single block(sparse)

Because primary index stores data in sorted order and unique values it easy to search

**Clustered index🡪**

Index table is created in such way that a pointer is mapped to single value, since clustered index stored data in sorted order and has non unique values there could multiple data with same value

A concept of **block hanker** is used in clustered index because the same data could in different blocks