
SQL PROJECT
Presented BY:
PANDIMEENA S

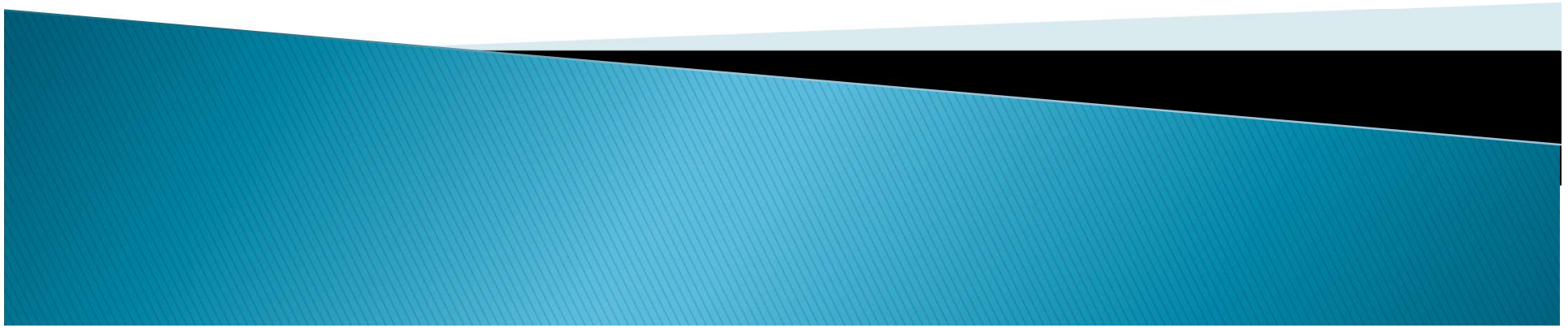
CONTENTS :

Database

- Data Base is the collection of Structured Information or Date Stored in the computer System ,
- The all Database are managed by DBMS (Database Management System).
- Database used to stored small unit and large amount of Data in Table format and also retrieving all data when and whatever we want.

Types of Databases

- Hierarchical Database
- Relational Database
- Non Relational Database
- Object Oriented Databases

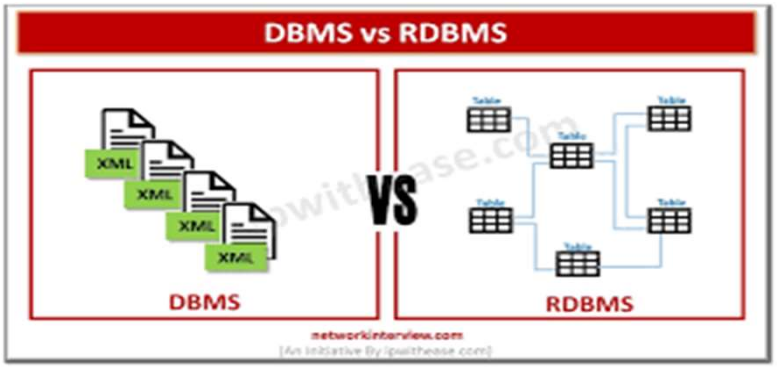


DBMS & RDBMS:

DBMS :

Database Management System:

- All databases managed by the Database
- Management System.
- Used to save data in structure manner,
- Stored, Retrieved and Run SQL query.



RDBMS :

Relational Database Management System.

- Its same as the DBMS in this case we can established the connection between The various tables in the Database.
- Create,Update,Delete and Managed the Datas.

Advantage:

- MySQL is one of the RDBMS
- We can connect the table and retrieving the data which is existing on the tables in databases.

What is SQL:

- Sql is the structure query language which is used to storing ,Manipulating and Retrieving data from the tables from databases.
- Its Easy to Learn and User-friendly
- Its is Interpreted Language
- Its is the most powerful language and Flexible.

Sql servers

- Oracle server
- MySQL workbench
- No SQL
- Mango DB server Microsoft server
- Navigation
- My SQL
- SQL Server

SQL VS MYSQL :

SQL	MYSQL
▪ It's a structure query Language to Manage RDBMS	▪ It's a Relational Database Management system.
▪ Its used to Retrieve the data from tables in databases	▪ Its used to managed tables and databases what we sired on that self.
▪ Sql is the statement query format to Codes the commands.	▪ Its used to established the connection between the Table by using of SQL language.

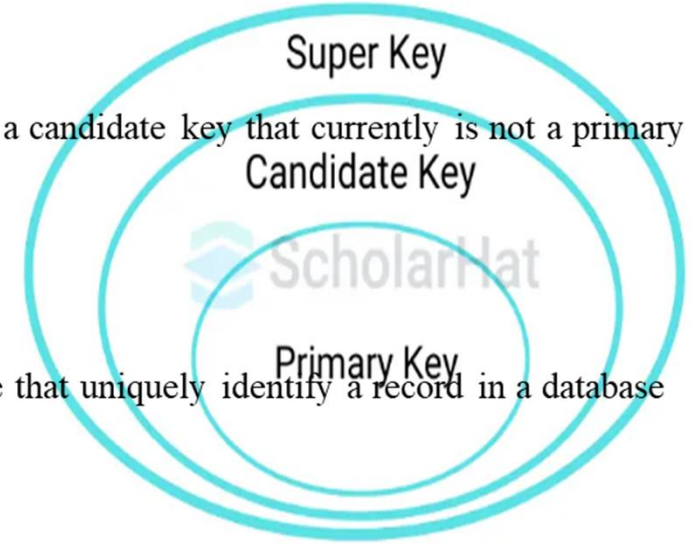
Why I choose the MySQL workbench Platform to Code the SQL:

- MySQL is one of the Relational Database Management System.
- Fast Storing,Retriving,Update,Delete the data easily.
- MySQL Connectively,Speed,and Security make its highly suited for accessing databases on the internet.
- Open sources.
- Scalability.
- Security.
- High performance.
- Reliability

Keys in Sql

- 1. Super key :
- A super key is a set of one or more than one key that can be used to identify a record uniquely in a table.
- **Example:** Primary key, Unique key, and Alternate key are a subset of Super Keys.

- 2.Candinate key :
- A Candidate Key is a set of one or more fields/columns that can identify a record uniquely in a table.
- There can be multiple Candidate Keys in one table. Each Candidate Key can work as a Primary Key.



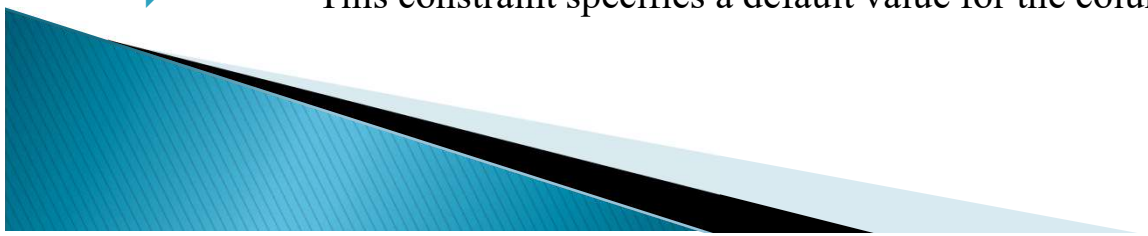
- 3.Alternate key:
- An Alternate key is a key that can work as a primary key. It is a candidate key that currently is not a primary key.
- It is also called a secondary key.

- 4.Primary key:
- A primary key is a set of one or more fields/columns of a table that uniquely identify a record in a database table.
- It can not accept null, or duplicate values.

- 5.Forign key:
- A foreign key is an attribute that is a Primary key in its parent table but is included as an attribute in another host table
- Foreign Key may have duplicate & NULL values if it is defined to accept NULL values.

Constrains in SQL

- **NOT NULL**:
- This constraint tells that we cannot store a null value in a column. That is, if a column is specified as NOT NULL then we will not be able to store null in this particular column any more.
- **UNIQUE**:
- This constraint when specified with a column, tells that all the values in the column must be unique. That is, the values in any row of a column must not be repeated.
- **PRIMARY KEY**:
- A primary key is a field which can uniquely identify each row in a table. And this constraint is used to specify a field in a table as primary key.
- **FOREIGN KEY**:
- A Foreign key is a field which can uniquely identify each row in a another table. And this constraint is used to specify a field as Foreign key.
- **CHECK**:
- This constraint helps to validate the values of a column to meet a particular condition. That is, it helps to ensure that the value stored in a column meets a specific condition.
- **DEFAULT**:
- This constraint specifies a default value for the column when no value is specified by user.



Data Types in Sql

Numeric Data Type :

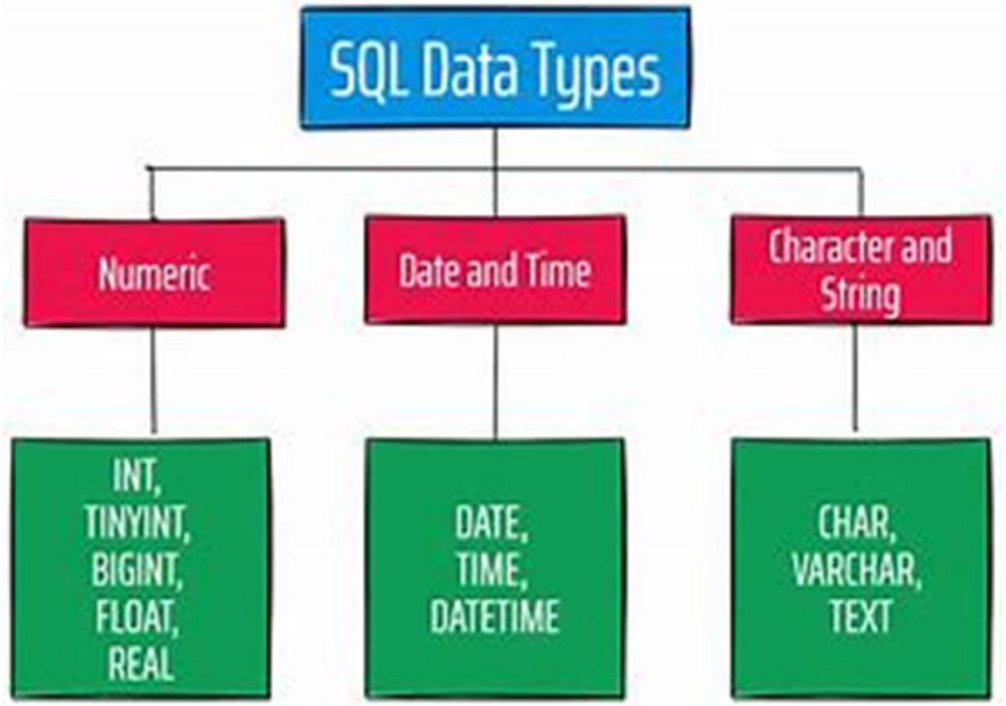
We can Stored Numeric Data Type is used to Represent all of the values, especially intervals and Ratios.

INTIGER DATA TYPES :

- Int -11 digit
- Tinyint - 4 digit
- Smallint - 5 digits
- Mediumint - 9 digit
- Bigint -20 digit

FLOAT DATA TYPES :

- Float -24 decimal point
- Double - 53 decimal point

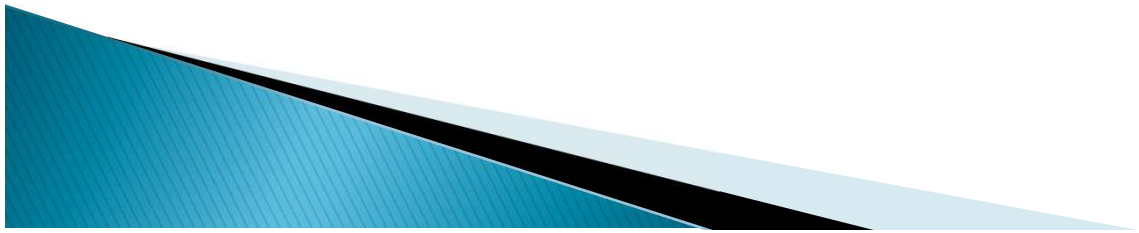


Sting data type :

- Varchar - 225 characters
- Tiny text - 225 characters
- Small text - 5 characters
- medium text - 9 digits
- Long Text - 4gb
- Binary - 225 characters
- Varbinary - 225 characters

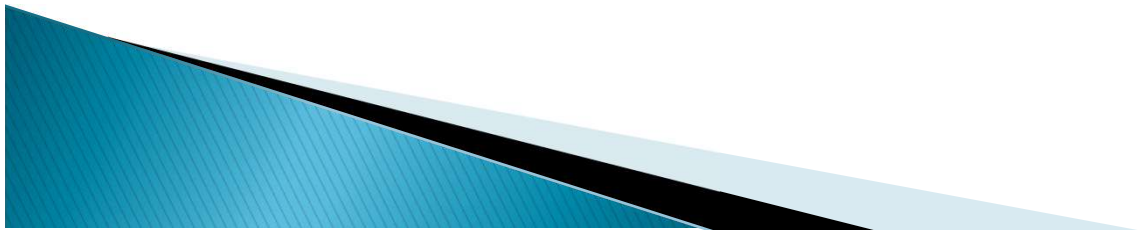
Object Data Types:

TinyBlob - 225 bytes
Blob - 65,535 bytes
Mediumblob - 16,777,215 bytes



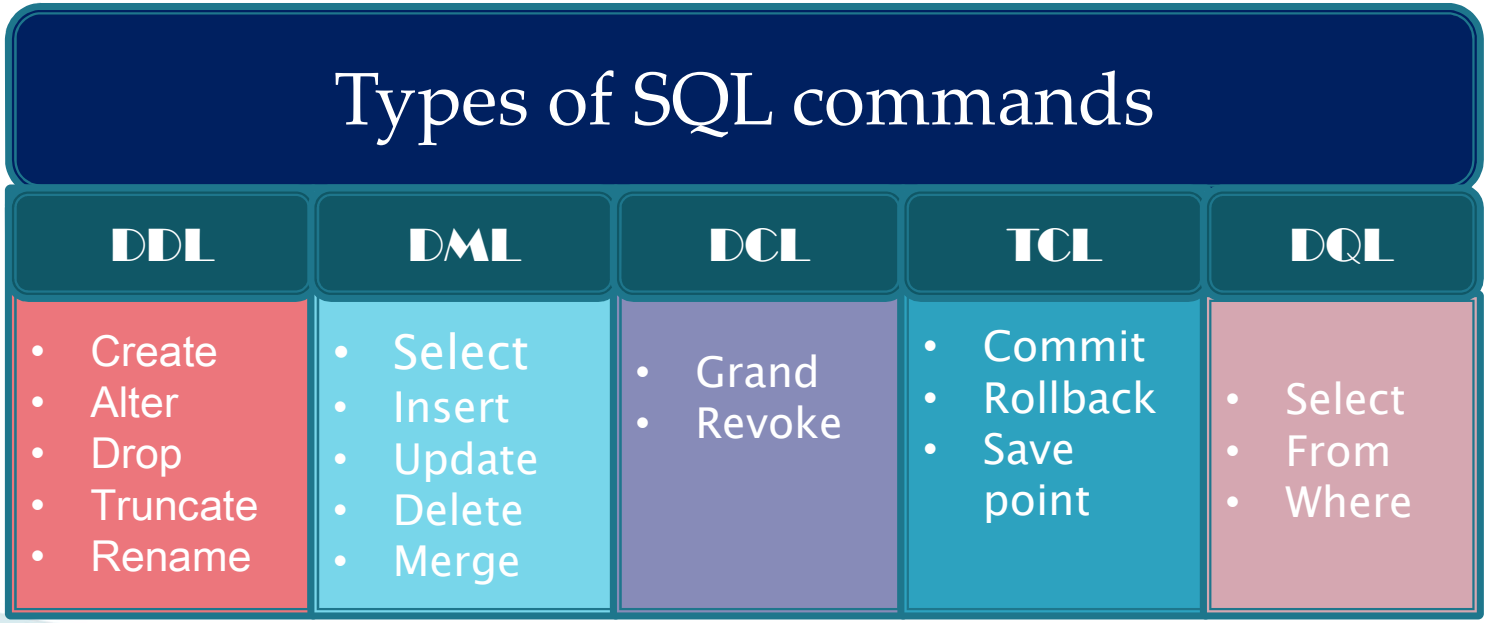
- **Date Data Type:**

- Date - yyyy-mm-dd-2024-03-18
- DateTime - yyyy-mm-dd hh-mm-ss -2024-03-18 7:56:56
- Timestamp - yyyy-mm-dd
- Timestampdiff - yyyy-mm-dd to yyyy_mm_dd = 2024-03-18
- Time - hh:mm:ss
- Year - 2 digit yy or 4 digit yyyy (default 4 digit)

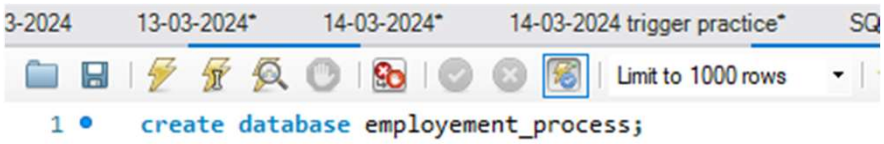


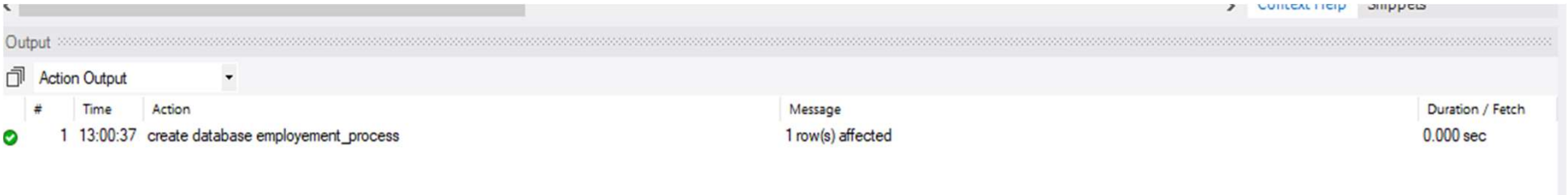
SQL Language:

- DDL - Data Definition Language.
- DML - Data Manipulating Language.
- DCL - Data Control Language.
- TCL - Transaction Control Language
- DQL - Data Query Language.



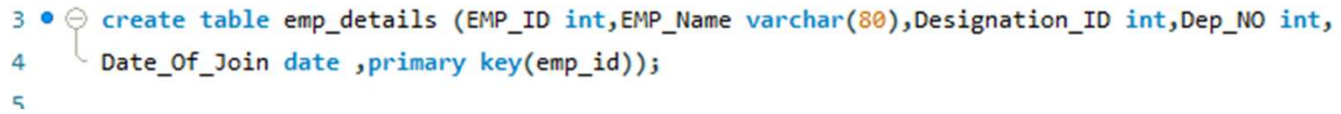
- **DDL:**
- Create Database:
- Create the new data base in MySQL workbench.

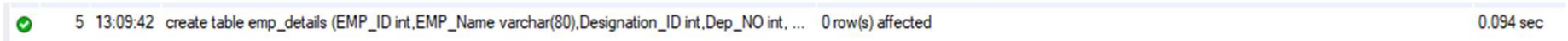




Create Table:

create table to stored a values as a data in databases.

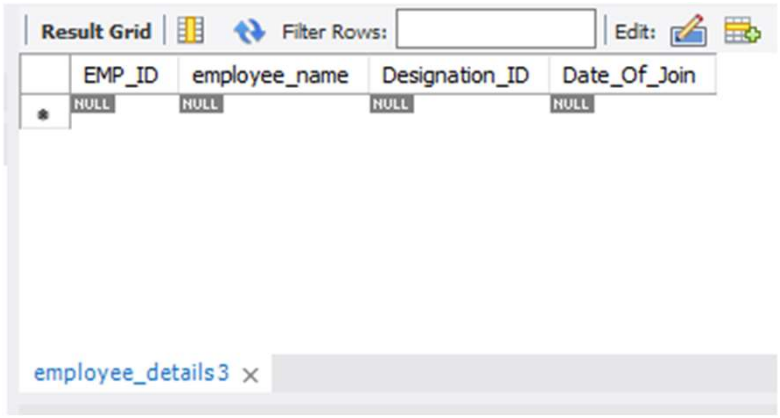




Alter Table :

- Modify
- Rename table
- Add
- Drop
- Change column name

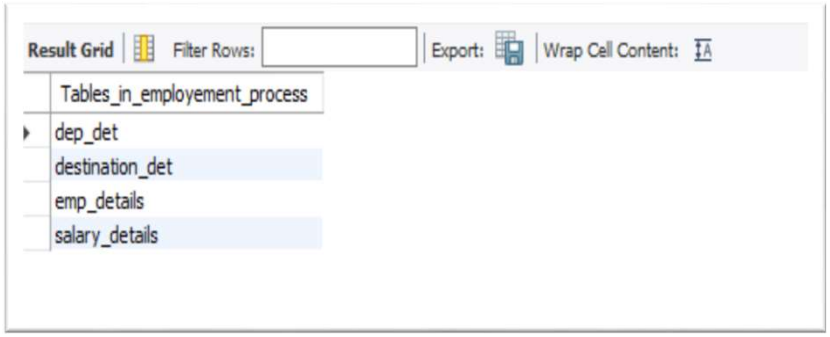
```
10 •  alter table emp_details modify  emp_name varchar(50);
11 •  alter table emp_details change emp_name  employee_name varchar(50);
12 •  alter table emp_details rename employee_details;
13 •  alter table employee_details drop dep_no;
```



Show Tables:

- We can see the all table in the database.

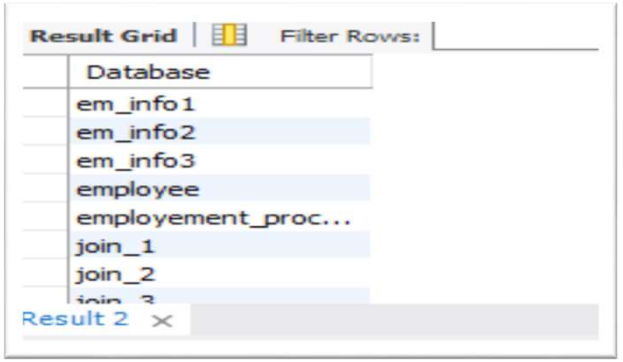
```
show tables;
```



Show Databases:

We can see All the Databases in the Tool.

```
show databases;
```



Drop:

Query: Drop database X ;

- Database
- Table

```
1 •  drop database employement_process;
```

```
1  14:10:29  drop database employement_process    4 row(s) affected    0.032 sec
```

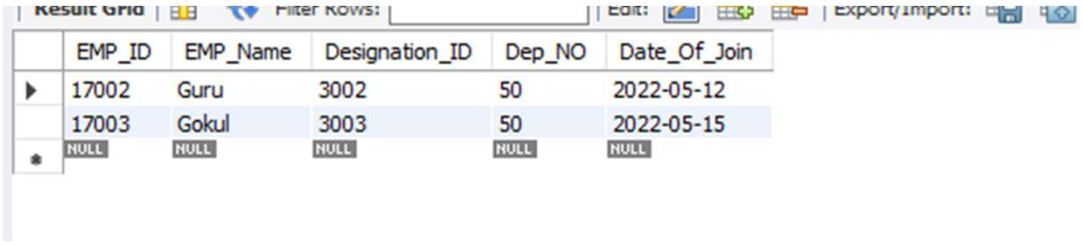
- **Truncate:**
- Query: truncate employement_process1;
- Database
- Table

```
6 •  truncate table emp_details;
```

```
5  22:19:44  truncate table emp_details    0 row(s) affected    0.031 sec
```

Dml :

- **Select all :**
- Select the data from database table what we want
- Query: select*from employee_details;

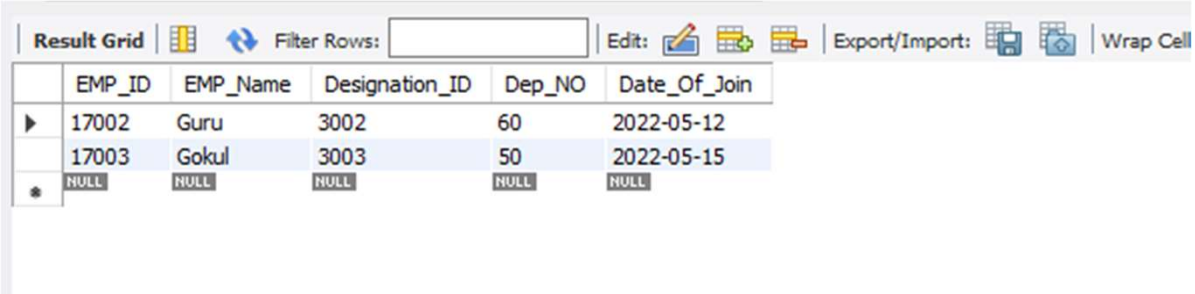


EMP_ID	EMP_Name	Designation_ID	Dep_NO	Date_Of_Join
17002	Guru	3002	50	2022-05-12
17003	Gokul	3003	50	2022-05-15
NULL	NULL	NULL	NULL	NULL

- **Insert:**
- Insert the values in to the database table .
- Query: insert into employee_details values (17002,'Guru',3002,50,'2022-0512'),(17003,'Gokul',3003,50,'2022-05-15');

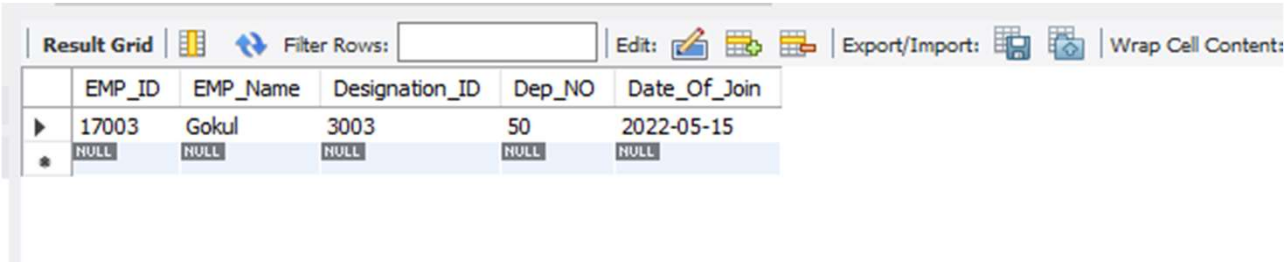
11	22:30:11	insert into employee_details values (17002,'Guru',3002,50,'2022-05-12'), (17003,'Gokul',300...	2 row(s) affected Records: 2 Duplicates: 0 Warnings: 0	0.000 sec

- **Update:**
- we can update the value of the table after insert the value.
- Query: update employee_details set dep_no =60 where emp_id=17002;



EMP_ID	EMP_Name	Designation_ID	Dep_NO	Date_Of_Join
17002	Guru	3002	60	2022-05-12
17003	Gokul	3003	50	2022-05-15
NULL	NULL	NULL	NULL	NULL

- **Delete**:
- we can delete the value of the data from table
- Query: delete from employee_details where emp_id=17002;

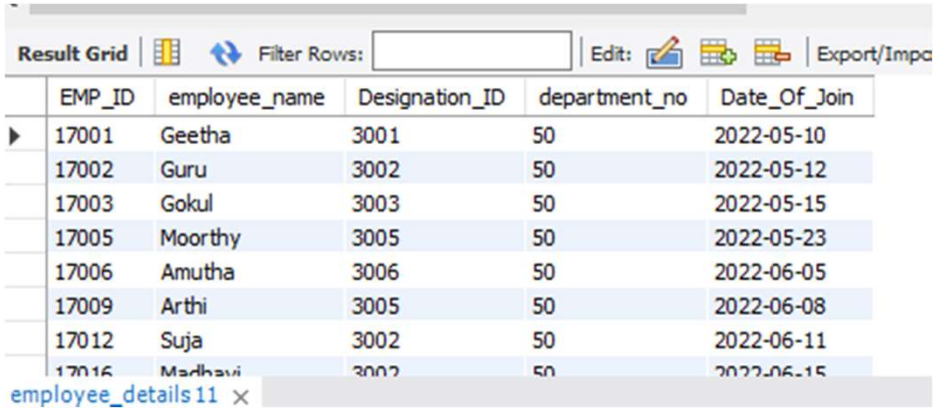


EMP_ID	EMP_Name	Designation_ID	Dep_NO	Date_Of_Join
17003	Gokul	3003	50	2022-05-15
NULL	NULL	NULL	NULL	NULL

General Functions in SQL :

- Where
- Or
- And
- In
- Not in
- greater than (>)
- less than (<)
- less tan or equal(<=)
- grater tan or equal (>=)
- not equal(<>)
- !
- Count
- Distinct count
- Order by
- Group by
- Limit
- Like
- Not like
- Between

- **Where:**
- Retrieve data from the table what using in the specified where condition
- query: select*from employee_details where department_no=50;

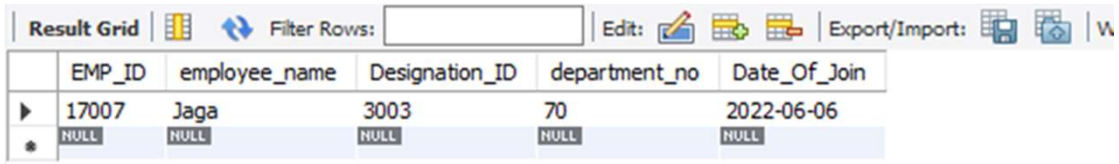


Result Grid | Filter Rows: | Edit: | Export/Impo

EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join
17001	Geetha	3001	50	2022-05-10
17002	Guru	3002	50	2022-05-12
17003	Gokul	3003	50	2022-05-15
17005	Moorthy	3005	50	2022-05-23
17006	Amutha	3006	50	2022-06-05
17009	Arthi	3005	50	2022-06-08
17012	Suja	3002	50	2022-06-11
17016	Madhavi	3002	50	2022-06-15

employee_details 11 ×

- **And:**
- Select muliple critiria value of data from table
- Query:select*from employee_details where department_no =70 and employee_name="jaga";



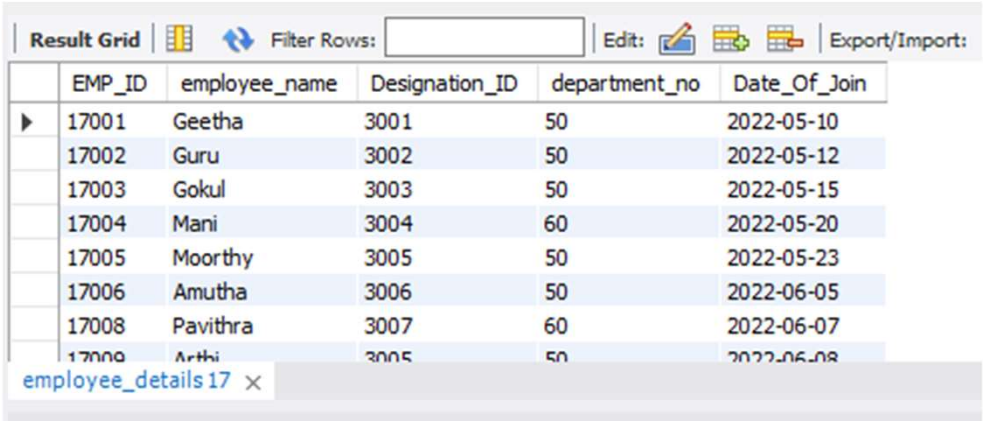
Result Grid | Filter Rows: | Edit: | Export/Import: | W

EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join
17007	Jaga	3003	70	2022-06-06
NULL	NULL	NULL	NULL	NULL

Or :

Select multiple criteria value by using or conditons that estimates one will be true return both.

Query : select*from employee_details where (department_no=50 or department_no=60) or date_of_join<"2022-06-06";



Result Grid | Filter Rows: | Edit: | Export/Import:

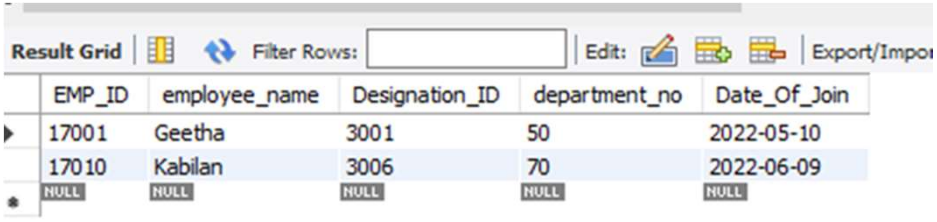
EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join
17001	Geetha	3001	50	2022-05-10
17002	Guru	3002	50	2022-05-12
17003	Gokul	3003	50	2022-05-15
17004	Mani	3004	60	2022-05-20
17005	Moorthy	3005	50	2022-05-23
17006	Amutha	3006	50	2022-06-05
17008	Pavithra	3007	60	2022-06-07
17009	Arthi	3005	50	2022-06-08

employee_details 17 ×

In :

Select a table data from some range that means what we specified in range .

- Query: select*from employee_details where emp_id in(17001,17010);

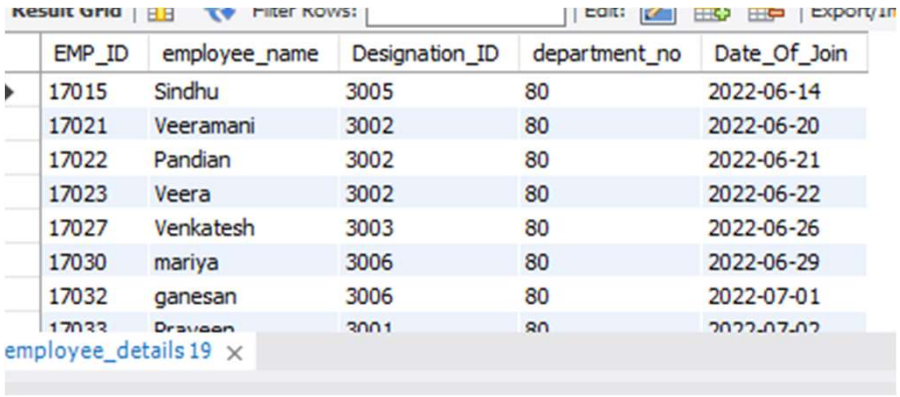


Result Grid | Filter Rows: | Edit: | Export/Impo

EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join
17001	Geetha	3001	50	2022-05-10
17010	Kabilan	3006	70	2022-06-09
NULL	NULL	NULL	NULL	NULL

Not in

- Select a exists of a table date what we pun in the range in I condition
- Query:select*from employee_details where department_no not in ("50","60","70")
  ;

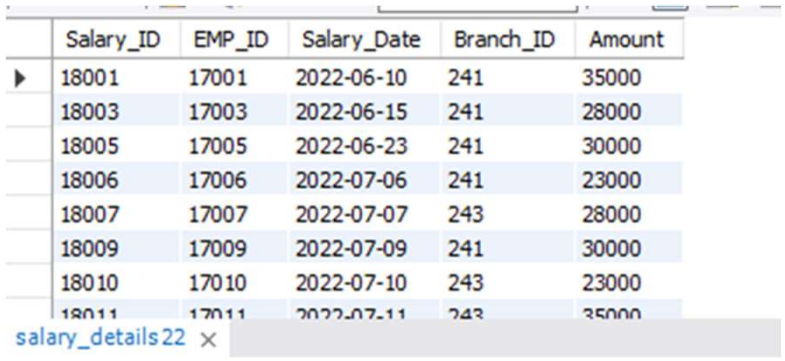


EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join
17015	Sindhu	3005	80	2022-06-14
17021	Veeramani	3002	80	2022-06-20
17022	Pandian	3002	80	2022-06-21
17023	Veera	3002	80	2022-06-22
17027	Venkatesh	3003	80	2022-06-26
17030	mariya	3006	80	2022-06-29
17032	ganesan	3006	80	2022-07-01

- **Grater than :**

  Select a value grater than the range what we given value.

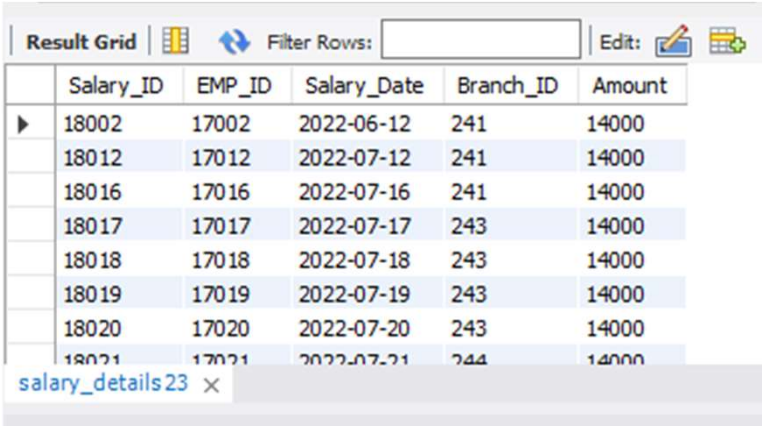
Query : select *from salary_details where amount>20000;



Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount
18001	17001	2022-06-10	241	35000
18003	17003	2022-06-15	241	28000
18005	17005	2022-06-23	241	30000
18006	17006	2022-07-06	241	23000
18007	17007	2022-07-07	243	28000
18009	17009	2022-07-09	241	30000
18010	17010	2022-07-10	243	23000

Less than (<):

- Select the values less than our conditions
- **Query :** select *from salary_details where amount<15000;

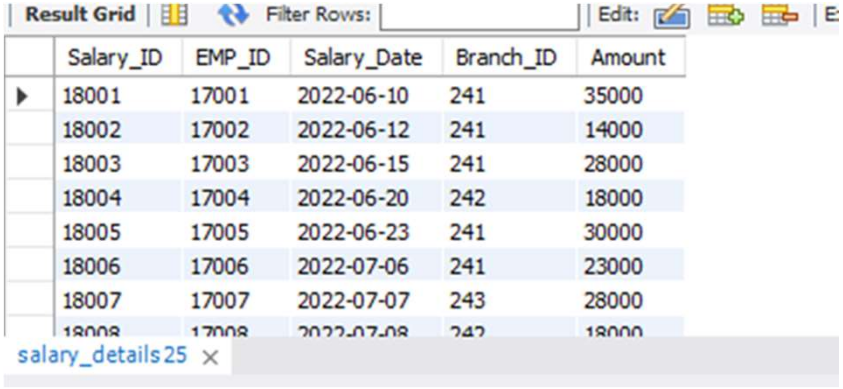


Result Grid | Filter Rows: | Edit:

Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount
18002	17002	2022-06-12	241	14000
18012	17012	2022-07-12	241	14000
18016	17016	2022-07-16	241	14000
18017	17017	2022-07-17	243	14000
18018	17018	2022-07-18	243	14000
18019	17019	2022-07-19	243	14000
18020	17020	2022-07-20	243	14000
18021	17021	2022-07-21	244	14000

salary_details23 ×

- **Grater than equal (>=):**
- select the valiues fro the greater and then equals in same sql satatement.
- **Query :** select*from salary_details where amount>=14000;



Result Grid | Filter Rows: | Edit:

Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount
18001	17001	2022-06-10	241	35000
18002	17002	2022-06-12	241	14000
18003	17003	2022-06-15	241	28000
18004	17004	2022-06-20	242	18000
18005	17005	2022-06-23	241	30000
18006	17006	2022-07-06	241	23000
18007	17007	2022-07-07	243	28000
18008	17008	2022-07-08	242	18000

salary_details25 ×

Less than equal (<=):

select the values less and then the equals value from the table of the databases.

Query:select*from salary_details where amount<=15000;

Result Grid | Filter Rows: | Edit:

Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount
18018	17018	2022-07-18	243	14000
18019	17019	2022-07-19	243	14000
18020	17020	2022-07-20	243	14000
18021	17021	2022-07-21	244	14000
18022	17022	2022-07-22	244	14000
18023	17023	2022-07-23	244	14000
NULL	NULL	NULL	NULL	NULL

salary_details 27 ×

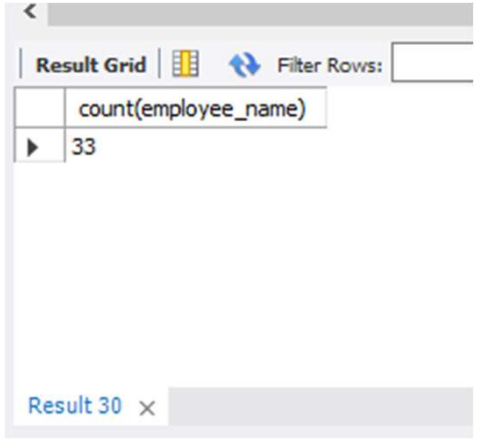
- **Not equal :**
- Retrieve data from exists data from table what we give in not equals .
- **Query:** select*from employee_details where department_no <>70;

Result Grid | Filter Rows: | Edit: | Export/Import

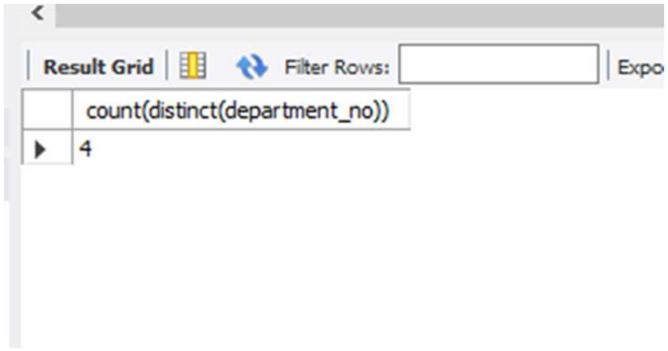
EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join
17001	Geetha	3001	50	2022-05-10
17002	Guru	3002	50	2022-05-12
17003	Gokul	3003	50	2022-05-15
17004	Mani	3004	60	2022-05-20
17005	Moorthy	3005	50	2022-05-23
17006	Amutha	3006	50	2022-06-05
17008	Pavithra	3007	60	2022-06-07
17009	Arthi	3005	50	2022-06-08

employee_details 29 ×

- **Count:**
- Count the values or Data from the table
- Query: count(employee_name)from employee_details ;



- **Distinct count:**
- Select the counted value without duplicate by using distinct count.
- Query: select count(distinct(department_no)) from employee_details;

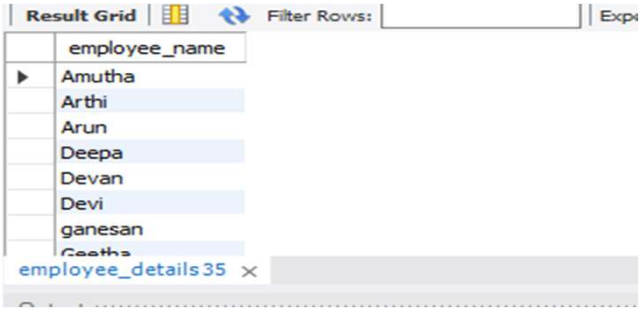


Order by :

Types By Shorts:

Ascending order – short by ascending order

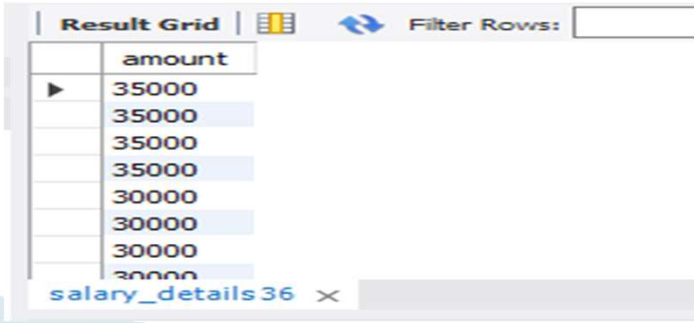
- **Query :** select employee_name from employee_details order by employee_name asc;



Descending order – shorts by descanting order.

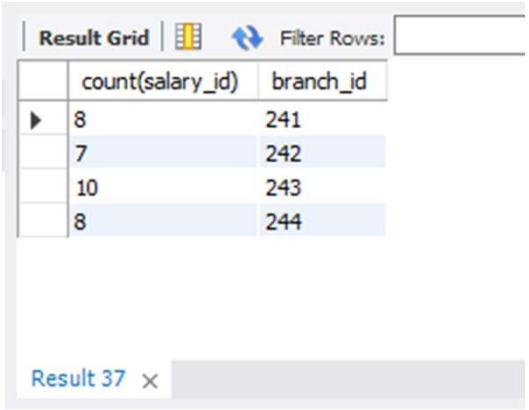
We can get retrive the data from last of the Table by ordering the data descending.

- **Query:** select amount from salary_details order by amount desc;



Group by:

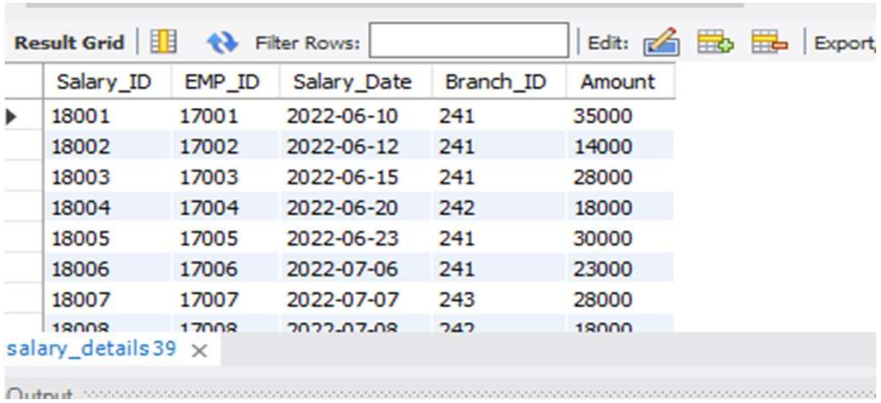
- Grouping the data to remove the dublicate
- **Query**: select count(salary_id),branch_id from salary_details group by branch_id;



Limit:

Get the table data with the specific range limit.

Query: select*from salary_details limit 0,8;



Limit from descending order:

- Get the data of the table in reverse by set the data in descenting order.
- Query: select*from salary_details order by salary_id desc limit 0,6;

Result Grid | Filter Rows: | Edit:

Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount
18033	17033	2022-08-02	244	35000
18032	17032	2022-08-01	244	23000
18031	17031	2022-07-31	243	30000
18030	17030	2022-07-30	244	23000
18029	17029	2022-07-29	243	30000
18028	17028	2022-07-28	242	18000
NULL	NULL	NULL	NULL	NULL

salary_details 41 ×

- Like:
- Get the data from table what we set in the like
- Query: select employee_name from employee_details where employee_name like"A%";

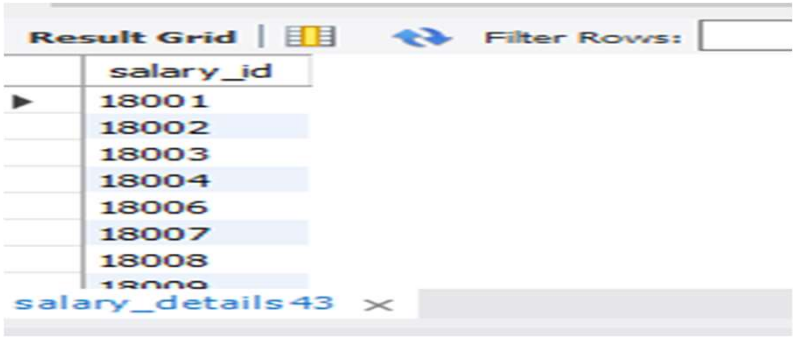
Result Grid | Filter Rows:

employee_name
Amutha
Arthi
Arun

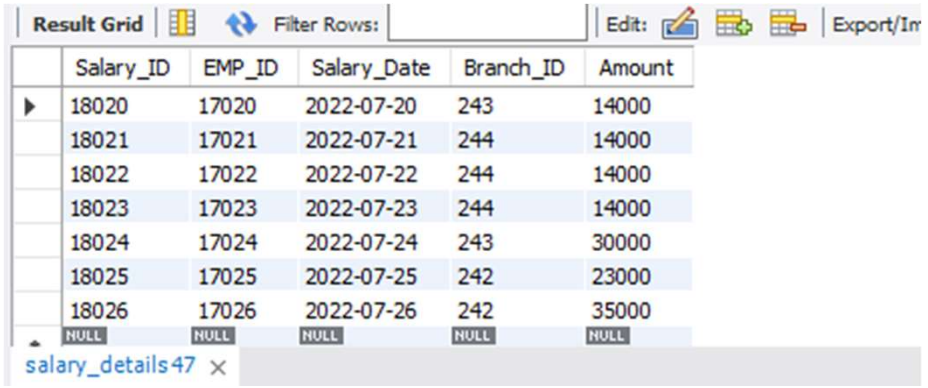
employee_details 42 ×

Not like:

- Get the data exists of the not like data.
- **Query:** select salary_id from salary_details where salary_id not like "%5";



- **Between :**
- Get the data from the in between Range.
- **Query:** select*from salary_details where salary_id between 18020 and 18026;

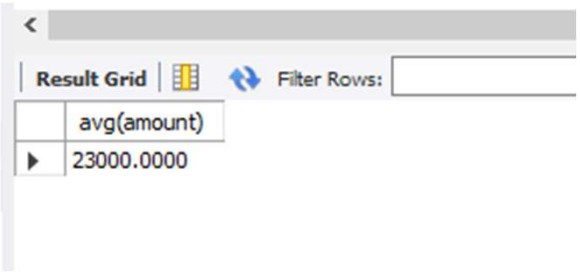


Aggregate functions in SQL

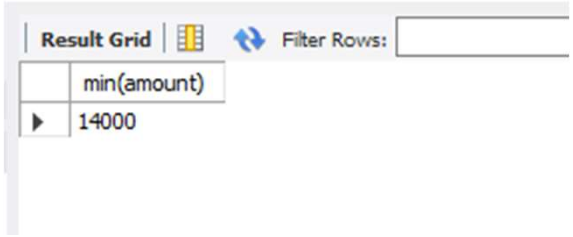
- **Sum:**
- Get the data from table by sum value of particular Data.
- **Query**: select sum(amount) from salary_details1 ;

Average:

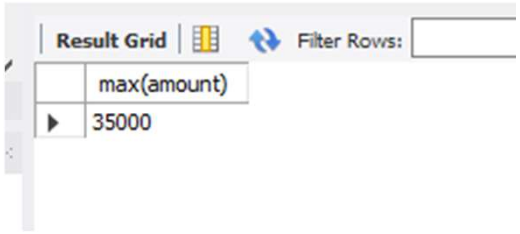
- Get the values of the data by average value of particular Data in Table
- **Query:** select avg(amount) from salary_details1;
-



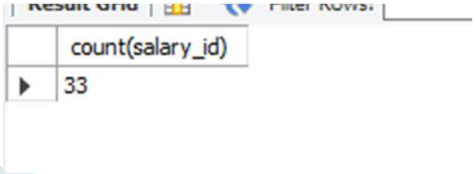
- **Min:**
- Get the data as a minimum value of the data
- **Query:** select min(amount) from salary_details1;



- **Max:**
- Get the data as the maximum values of the data
- **Query:** select max(amount) from salary_details1;

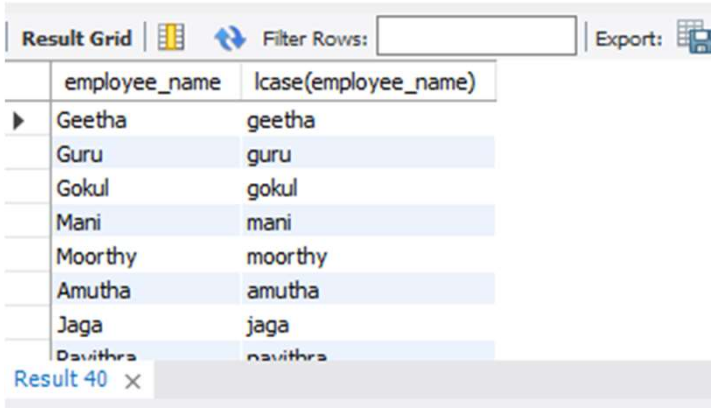


- **Count:**
- Count the value using grouping the data
- **Query:** select count(salary_id) from salary_details1;

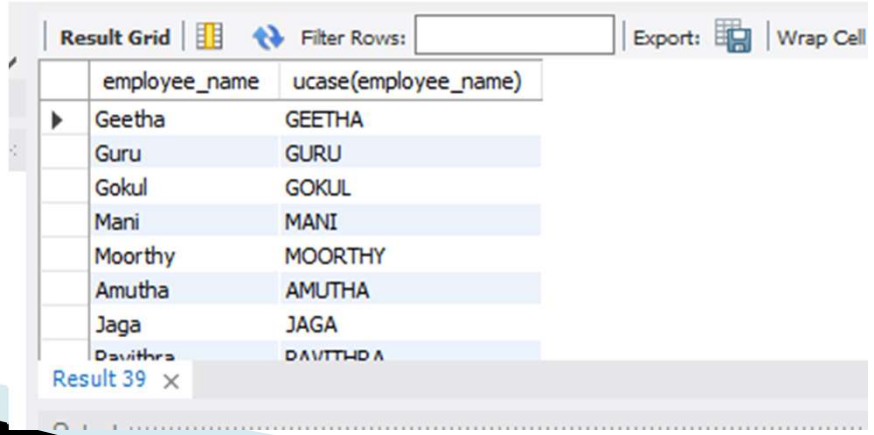


String Functions In SQL

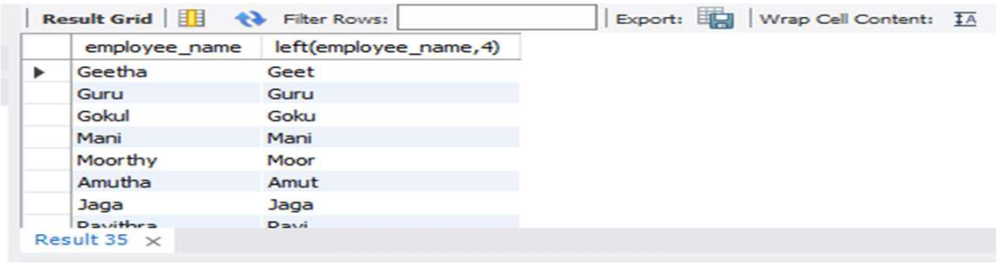
- **L case:**
- Format the Table of Data in Lowercases.
- **Query:** select employee_name , lcase(employee_name) from employee_details;



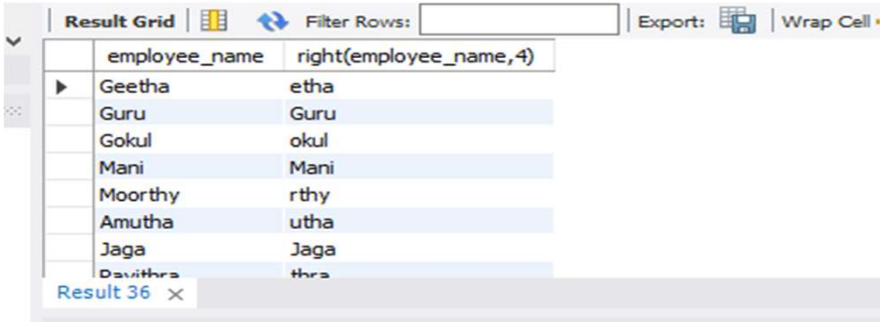
- **U case:**
- Format the table of the data in the Uppercases
- **Query:** select employee_name , ucase(employee_name) from employee_details;



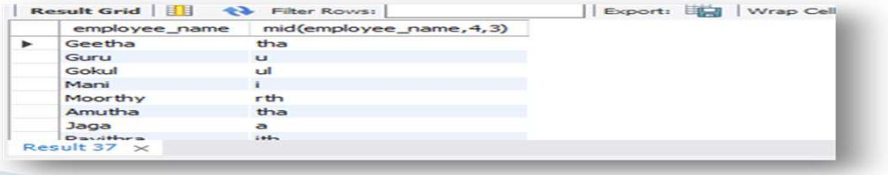
- **Left:** we can get the data from the table that's like in left area by which character length given by we.
- **Query:** select employee_name ,left(employee_name,4) from employee_details;



- **Right:**
- We can get the data as a right from number of character which is given by we.
- **Query:**select employee_name ,right(employee_name,4) from employee_details;

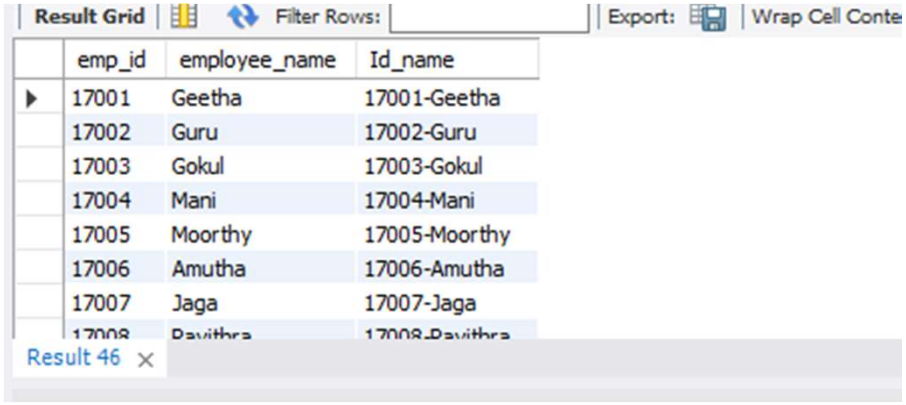


- **Mid:**
- **Query:**
- select employee_name , mid(employee_name,4,3) from employee_details;
- We can get the data in the middle of the data

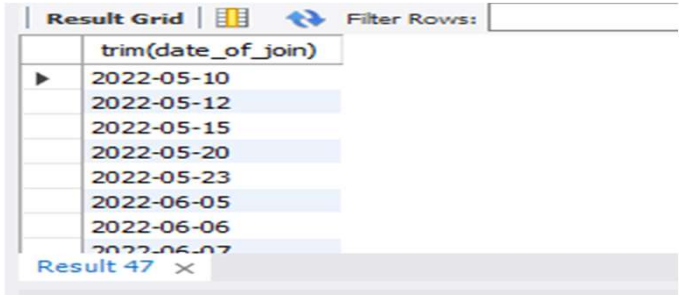


Concatenation:

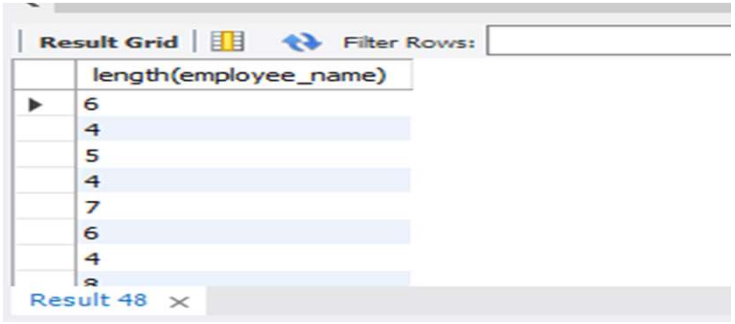
- Concatenate Two Data from the Tables.
- **Query:**select emp_id, employee_name ,concat(emp_id,"-",employee_name) as Id_name from employee_details;



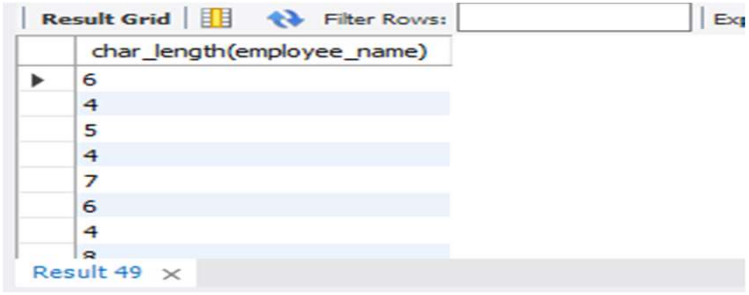
- **Trim:**
- Trim the data from the table that means shape the data in the format of the Data
- **Query:** select trim(date_of_join) from employee_details;



- **Length:**
- To find out the length that means number of the characters in the data by bytes.
- **Query:**select length(employee_name) from employee_details;

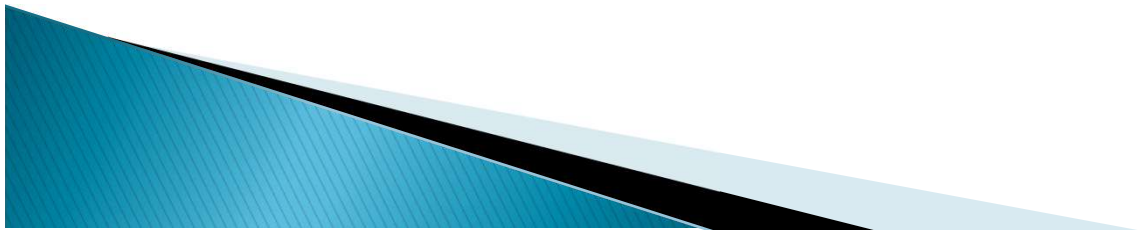


- **Char length:**
- To find out the exact number of the character in the table .
- **Query:**select char_length(employee_name) from employee_details;

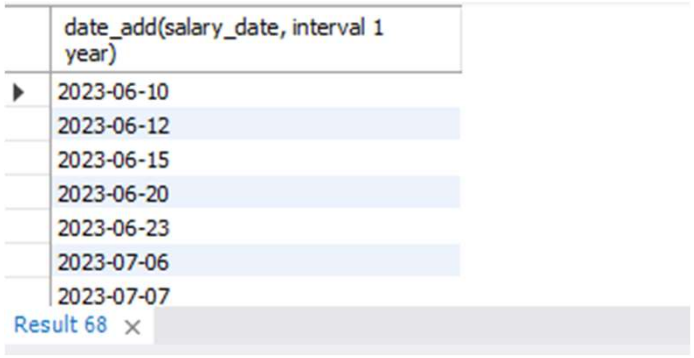


Date Functions in SQL

- Date add
- Datediff
- Date format
- Year
- Day
- Month
- Now &current date
- Timemstampdiff



- **Date add:**
- Used and find a add value of the date Data of any Date by adding with another Date.
- **Query:**select date_add(salary_date, interval 1 year) from salary_details1 ;



	date_add(salary_date, interval 1 year)
▶	2023-06-10
	2023-06-12
	2023-06-15
	2023-06-20
	2023-06-23
	2023-07-06
	2023-07-07

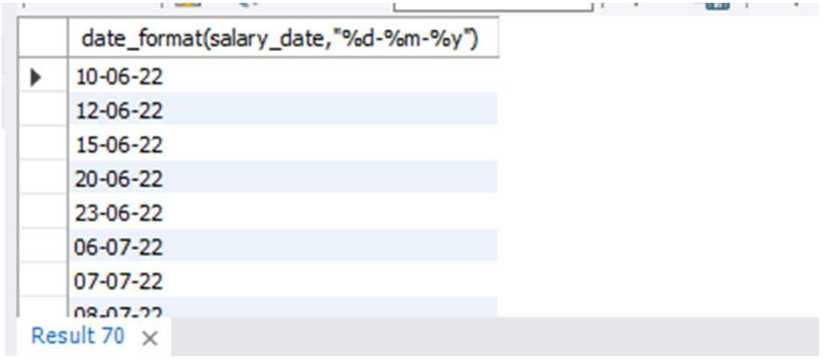
Result 68 ×

- **Datediff:**
- Used to subtract the two Dates of the Data.
- **Query:**select datediff(curdate(),salary_date) from salary_details1 ;

	datediff(curdate(),salary_date)
▶	644
	642
	639
	634
	631
	618
	617
	616

Result 69 ×

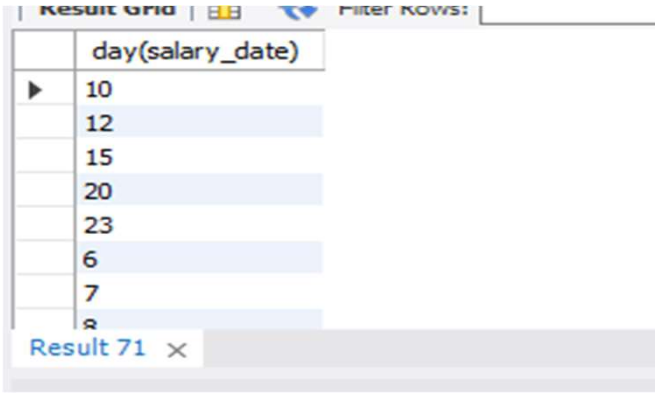
- **Date format:**
- We can modify the formation of the date as we want
- **Query**:select date_format(salary_date,"%d-%m-%y") from salary_details1 ;



	date_format(salary_date,"%d-%m-%y")
▶	10-06-22
	12-06-22
	15-06-22
	20-06-22
	23-06-22
	06-07-22
	07-07-22
	08-07-22

Result 70 ×

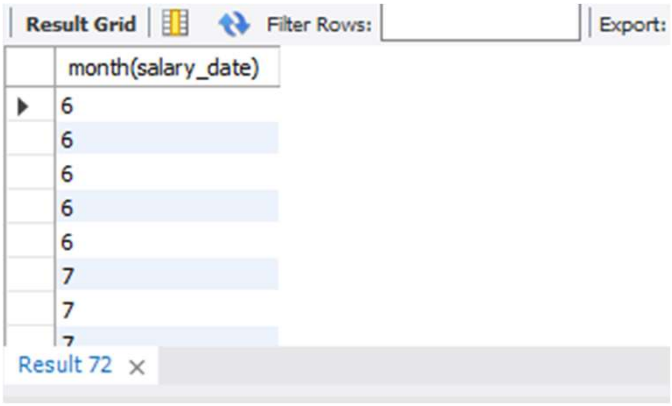
- **Day:**
- Day function used to get the date data only in the days values
- **Query**:select day(salary_date) from salary_details1;



	day(salary_date)
▶	10
	12
	15
	20
	23
	6
	7
	8

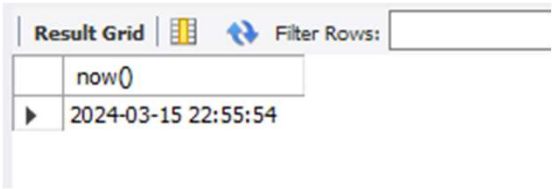
Result 71 ×

- **Month:**
- To used to get the date as a only months or month format
- **Query:**select month(salary_date) from salary_details1;

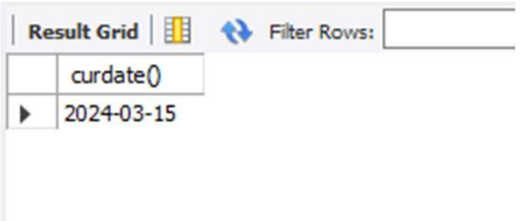


- **Year:**
- Get the data from the date data as a year format or only year of the date.
- **Query:** select year(salary_date) from salary_details1;

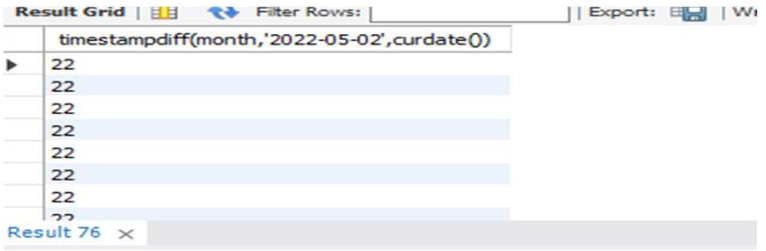
- **Now:**
- Used to define the current date of the today in only date format
- **Query:** select* now();



- **Current date:**
- Can get the date of today with date & time format
- **Query:** select curdate();

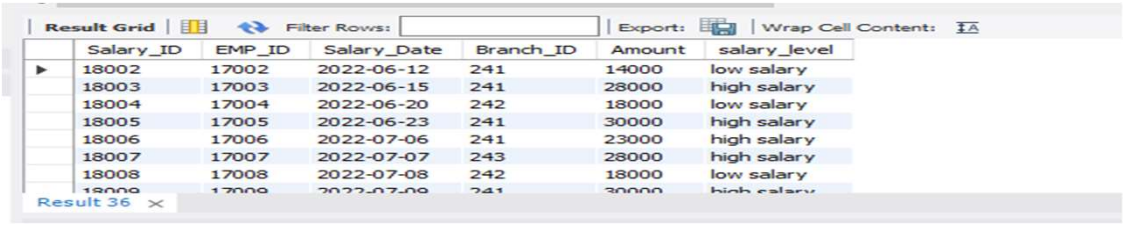


- **Timestamdiff:**
- We can subtract the two date an get the day or month or year format at the same time .
- **Query:** select timestampdiff(month,'2022-05-02',curdate()) from salary_details1;



Logical Functions in SQL

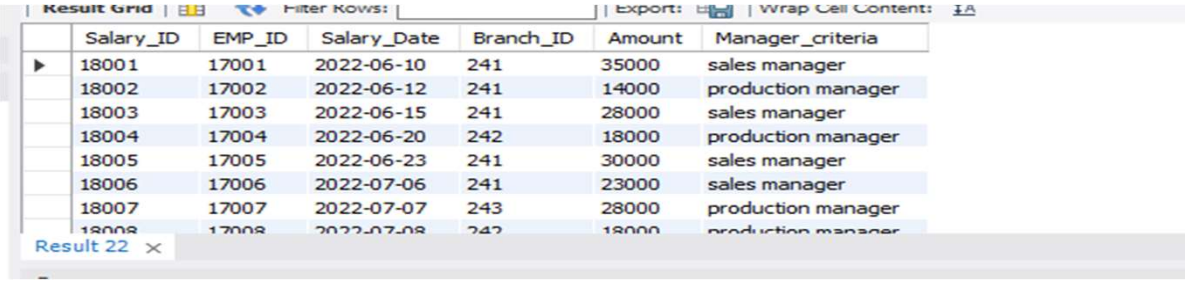
- **If statement:** If the given condition is true that returns the true value else the false value.
- **Query**:select*,if(amount>=20000,"high salary","low salary") as salary_level from salary_details1;



Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount	salary_level
18002	17002	2022-06-12	241	14000	low salary
18003	17003	2022-06-15	241	28000	high salary
18004	17004	2022-06-20	242	18000	low salary
18005	17005	2022-06-23	241	30000	high salary
18006	17006	2022-07-06	241	23000	high salary
18007	17007	2022-07-07	243	28000	high salary
18008	17008	2022-07-08	242	18000	low salary

Result 36

- **If with and statement:**
- Define the specific condition over the multiple criteria of the data or give any criteria range for the data
- **Qurey:** select*,if(amount>=20000 and branch_id=241,'sales manager','production manager')as Manager_criteria from salary_details1;



Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount	Manager_criteria
18001	17001	2022-06-10	241	35000	sales manager
18002	17002	2022-06-12	241	14000	production manager
18003	17003	2022-06-15	241	28000	sales manager
18004	17004	2022-06-20	242	18000	production manager
18005	17005	2022-06-23	241	30000	sales manager
18006	17006	2022-07-06	241	23000	sales manager
18007	17007	2022-07-07	243	28000	production manager

Result 22

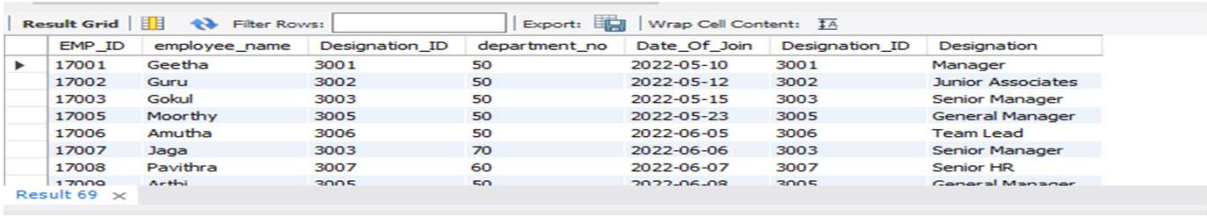
- **If with or statement:**
- That check the satement and returens te value Either the condition is true .
- **Query**:select*,if (amount >=30000,'A',if(amount>=20000,'B',if(amount>=10000,'C','Other'))) as Salary_Grade from salary_details1;

Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount	Salary_Grade
18001	17001	2022-06-10	241	35000	A
18002	17002	2022-06-12	241	14000	C
18003	17003	2022-06-15	241	28000	B
18004	17004	2022-06-20	242	18000	C
18005	17005	2022-06-23	241	30000	A
18006	17006	2022-07-06	241	23000	B
18007	17007	2022-07-07	243	28000	B

Result 23

Table Connections in SQL

- **Two Table Connections:**
- We can connect two Table of data
- **Query:**select*from employee_details join designation_det on employee_details.designation_id=designation_det.designation_id;

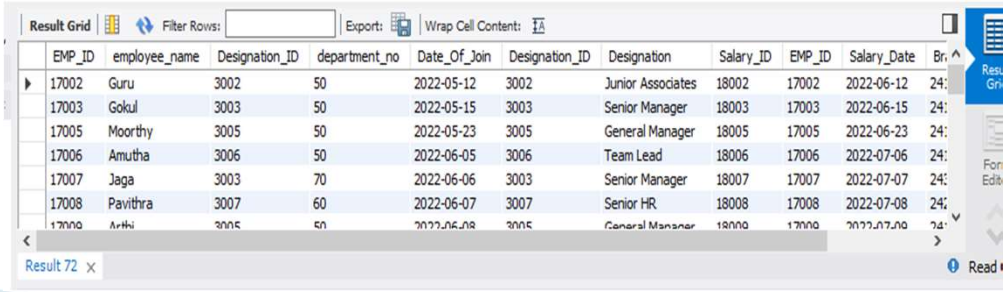


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join	Designation_ID	Designation
17001	Geetha	3001	50	2022-05-10	3001	Manager
17002	Guru	3002	50	2022-05-12	3002	Junior Associates
17003	Gokul	3003	50	2022-05-15	3003	Senior Manager
17005	Moorthy	3005	50	2022-05-23	3005	General Manager
17006	Amutha	3006	50	2022-06-05	3006	Team Lead
17007	Jaga	3003	70	2022-06-06	3003	Senior Manager
17008	Pavithra	3007	60	2022-06-07	3007	Senior HR
17009	Arthi	3005	50	2022-06-08	3005	General Manager

Result 69

- **Three Table Connections:**
- We can connect the Three table.
- **Query:**select* from employee_details join designation_det on employee_details.designation_id=designation_det.designation_id join salary_details1 on salary_details1.emp_id=employee_details.emp_id;



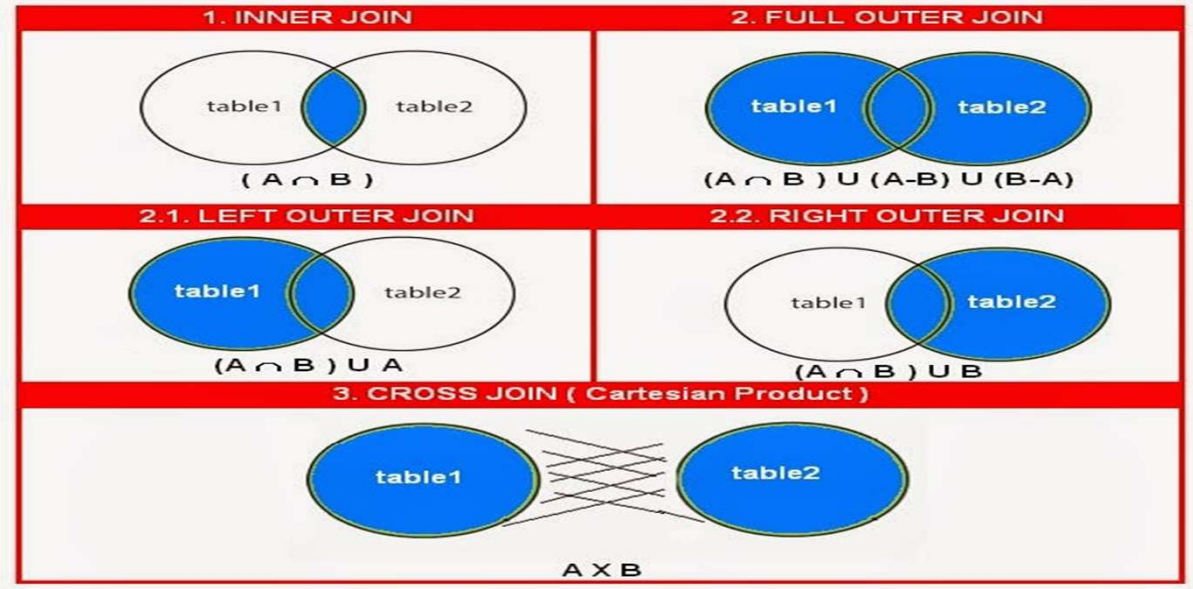
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join	Designation_ID	Designation	Salary_ID	EMP_ID	Salary_Date	Br
17002	Guru	3002	50	2022-05-12	3002	Junior Associates	18002	17002	2022-06-12	24:
17003	Gokul	3003	50	2022-05-15	3003	Senior Manager	18003	17003	2022-06-15	24:
17005	Moorthy	3005	50	2022-05-23	3005	General Manager	18005	17005	2022-06-23	24:
17006	Amutha	3006	50	2022-06-05	3006	Team Lead	18006	17006	2022-07-06	24:
17007	Jaga	3003	70	2022-06-06	3003	Senior Manager	18007	17007	2022-07-07	24:
17008	Pavithra	3007	60	2022-06-07	3007	Senior HR	18008	17008	2022-07-08	24:
17009	Arthi	3005	50	2022-06-08	3005	General Manager	18009	17009	2022-07-09	24:

Result 72

JOINS in SQL

- Inner join
- Right join
- Left join
- Full outer join
- Cross join



- **Inner join:**
- To connect the two or more tables that includes what are Rows exact match from Two tables
- **Query :** select employee_details.emp_id,employee_details.employee_name,employee_details.
- date_of_join,salary_details1.salary_date,salary_details1.amount from employee_details inner join salary_details1 on employee_details.emp_id=salary_details1.emp_id;

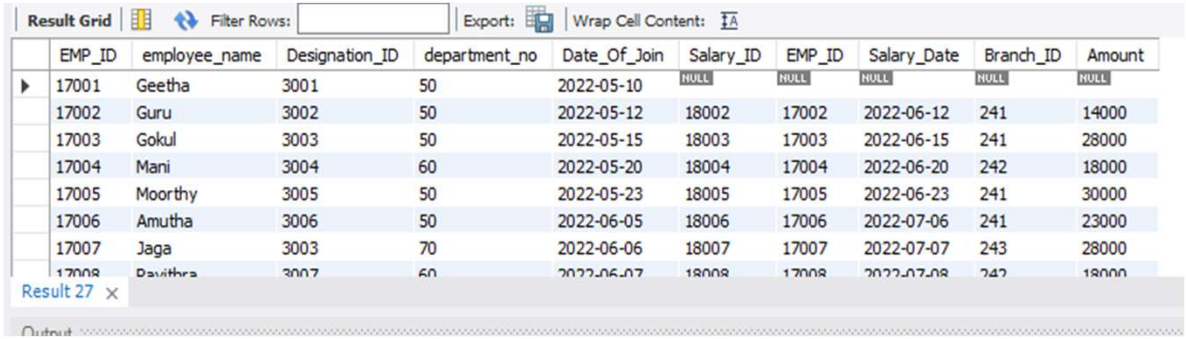
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

emp_id	employee_name	date_of_join	salary_date	amount
17027	Venkatesh	2022-06-26	2022-07-27	28000
17028	Raja	2022-06-27	2022-07-28	18000
17029	Priya	2022-06-28	2022-07-29	30000
17030	mariya	2022-06-29	2022-07-30	23000
17031	srinivasan	2022-06-30	2022-07-31	30000
17032	ganesan	2022-07-01	2022-08-01	23000
17033	Praveen	2022-07-02	2022-08-02	35000

Result 24

Left join:

- Connect the table even the right table has no the data for matches the left table
- **Query:** select *from employee_details left join salary_details1 on employee_details.emp_id=salary_details1.emp_id order by employee_details.emp_id asc ;

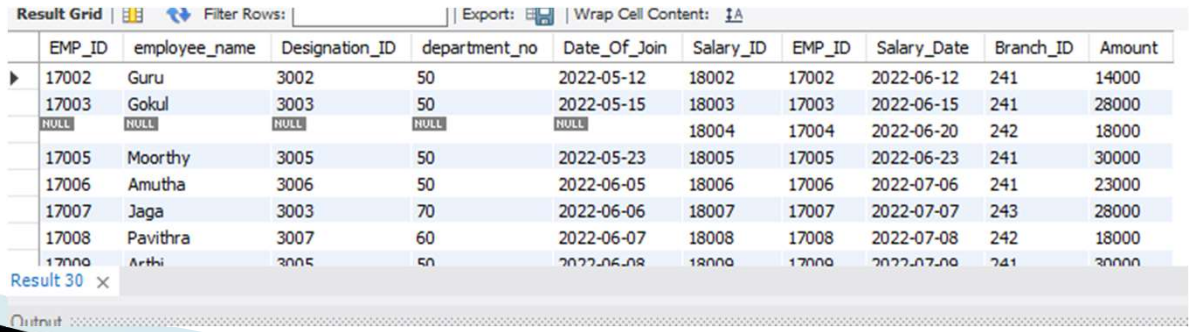


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join	Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount
17001	Geetha	3001	50	2022-05-10	NULL	NULL	NULL	NULL	NULL
17002	Guru	3002	50	2022-05-12	18002	17002	2022-06-12	241	14000
17003	Gokul	3003	50	2022-05-15	18003	17003	2022-06-15	241	28000
17004	Mani	3004	60	2022-05-20	18004	17004	2022-06-20	242	18000
17005	Moorthy	3005	50	2022-05-23	18005	17005	2022-06-23	241	30000
17006	Amutha	3006	50	2022-06-05	18006	17006	2022-07-06	241	23000
17007	Jaga	3003	70	2022-06-06	18007	17007	2022-07-07	243	28000
17008	Pavithra	3007	60	2022-06-07	18008	17008	2022-07-08	242	18000

Result 27

- **Right join:**
- Connect the table to left table even that left table has no data for matches the right table
- **Query:** select *from employee_details right join salary_details1 on employee_details.emp_id=salary_details1.emp_id order by salary_details1.emp_id asc ;



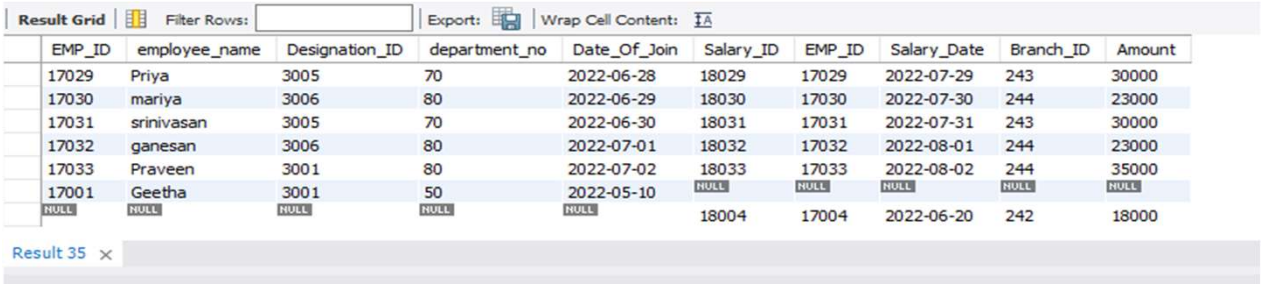
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join	Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount
17002	Guru	3002	50	2022-05-12	18002	17002	2022-06-12	241	14000
17003	Gokul	3003	50	2022-05-15	18003	17003	2022-06-15	241	28000
NULL	NULL	NULL	NULL	NULL	18004	17004	2022-06-20	242	18000
17005	Moorthy	3005	50	2022-05-23	18005	17005	2022-06-23	241	30000
17006	Amutha	3006	50	2022-06-05	18006	17006	2022-07-06	241	23000
17007	Jaga	3003	70	2022-06-06	18007	17007	2022-07-07	243	28000
17008	Pavithra	3007	60	2022-06-07	18008	17008	2022-07-08	242	18000

Result 30

Full Outer Join :

- Two connect two or more table in the union form even that has the mix matches data from the tables .
- **Qeury:(**select*from employee_details left join salary_details1 on employee_details.emp_id=salary_details1.emp_id) union(select*from employee_details right join salary_details1 on employee_details.emp_id=salary_details1.emp_id);

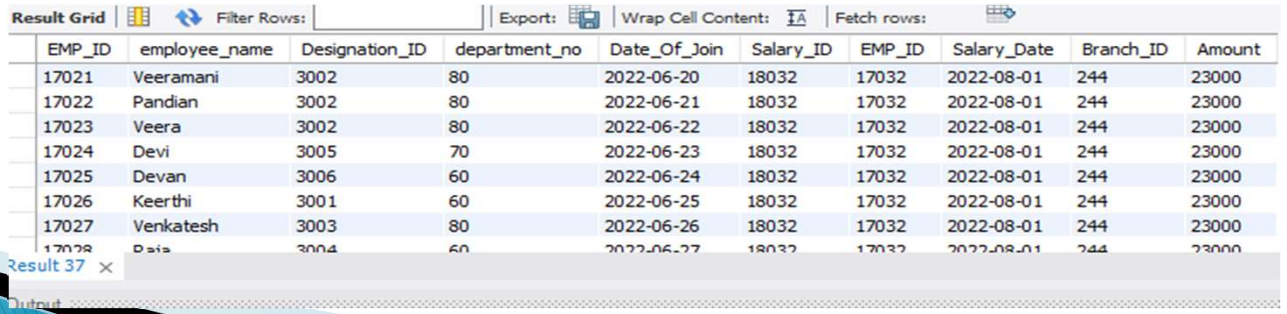


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join	Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount
17029	Priya	3005	70	2022-06-28	18029	17029	2022-07-29	243	30000
17030	mariya	3006	80	2022-06-29	18030	17030	2022-07-30	244	23000
17031	srinivasan	3005	70	2022-06-30	18031	17031	2022-07-31	243	30000
17032	ganesan	3006	80	2022-07-01	18032	17032	2022-08-01	244	23000
17033	Praveen	3001	80	2022-07-02	18033	17033	2022-08-02	244	35000
17001	Geetha	3001	50	2022-05-10	NULL	NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	18004	17004	2022-06-20	242	18000

Result 35

- **Cross join:**
- Connect the table data in each table data in the separate form .
- **Query:** select*from employee_details cross join salary_details1 ;



Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join	Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount
17021	Veeramani	3002	80	2022-06-20	18032	17032	2022-08-01	244	23000
17022	Pandian	3002	80	2022-06-21	18032	17032	2022-08-01	244	23000
17023	Veera	3002	80	2022-06-22	18032	17032	2022-08-01	244	23000
17024	Devi	3005	70	2022-06-23	18032	17032	2022-08-01	244	23000
17025	Devan	3006	60	2022-06-24	18032	17032	2022-08-01	244	23000
17026	Keerthi	3001	60	2022-06-25	18032	17032	2022-08-01	244	23000
17027	Venkatesh	3003	80	2022-06-26	18032	17032	2022-08-01	244	23000

Result 37

Joins with sub queries

- **Case and end statement :**
- We can set Multiple Criteria for the different ranges in same time for multiple updates. **Query:** select*,casewhen date_of_join <='2022-06-01' then "seniors"else "junions"end as seniorityfrom employee_details;

EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join	seniority
17001	Geetha	3001	50	2022-05-10	seniors
17002	Guru	3002	50	2022-05-12	seniors
17003	Gokul	3003	50	2022-05-15	seniors
17005	Moorthy	3005	50	2022-05-23	seniors
17006	Amutha	3006	50	2022-06-05	juniors
17007	Jaga	3003	70	2022-06-06	juniors
17008	Pavithra	3007	60	2022-06-07	juniors
17009	Arthi	3005	50	2022-06-08	juniors

Result 59

- **Multiple when in one Case End:**
- Here We can use multiple when and then statement in one case and end statement.
- **Query :** select *,case when department_no = 50 then "sales department"when department_no = 60 then "finance department"when department_no = 70 then "production department"when department_no = 80 then "R&D department"else "other"end as Department_namefrom employee_details;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

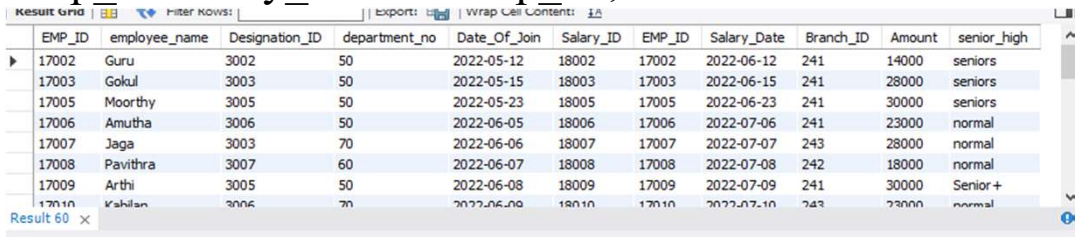
EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join	Department_name
17001	Geetha	3001	50	2022-05-10	sales department
17002	Guru	3002	50	2022-05-12	sales department
17003	Gokul	3003	50	2022-05-15	sales department
17005	Moorthy	3005	50	2022-05-23	sales department
17006	Amutha	3006	50	2022-06-05	sales department
17007	Jaga	3003	70	2022-06-06	production department
17008	Pavithra	3007	60	2022-06-07	finance department
17009	Arthi	3005	50	2022-06-08	sales department

Result 38

Double cases in single statement:

Use case statement into another case statement.

Query: select*,case when date_of_join <='2022-06-01' then "seniors" else case when amount >=30000 then "Senior+" else "normal" end end as senior_high from employee_details inner join salary_details1 on employee_details.emp_id=salary_details1.emp_id ;



EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join	Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount	senior_high
17002	Guru	3002	50	2022-05-12	18002	17002	2022-06-12	241	14000	seniors
17003	Gokul	3003	50	2022-05-15	18003	17003	2022-06-15	241	28000	seniors
17005	Moorthy	3005	50	2022-05-23	18005	17005	2022-06-23	241	30000	seniors
17006	Amutha	3006	50	2022-06-05	18006	17006	2022-07-06	241	23000	normal
17007	Jaga	3003	70	2022-06-06	18007	17007	2022-07-07	243	28000	normal
17008	Pavithra	3007	60	2022-06-07	18008	17008	2022-07-08	242	18000	normal
17009	Arthi	3005	50	2022-06-08	18009	17009	2022-07-09	241	30000	Senior+

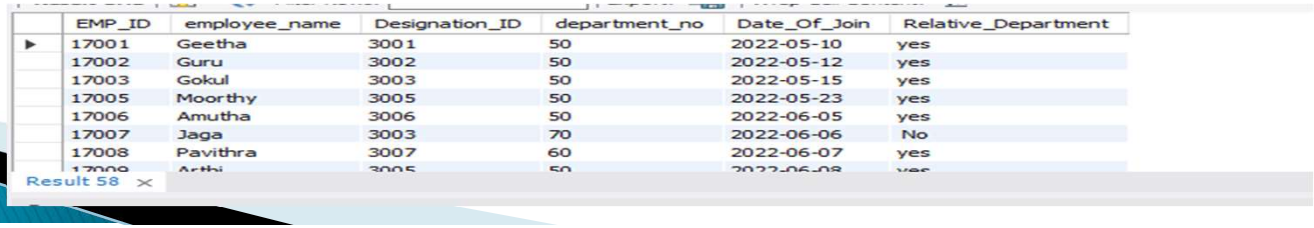
Result 60

- **Case with and: Query:** select employee_details.emp_id,employee_details.employee_name, employee_details.department_no,employee_details.date_of_join, salary_details1.amount, case when (employee_details.department_no =50 and salary_details1.amount>=20000 and employee_details.date_of_join <='2022-06-01') then "sales deparment +" else "sales department" end as sales_department_high from employee_details inner join salary_details1 on employee_details.emp_id=salary_details1.emp_id where employee_details.department_no=50 and salary_details1.amount>=20000;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

emp_id	employee_name	department_no	date_of_join	amount	sales_department_high
17003	Gokul	50	2022-05-15	28000	sales department +
17005	Moorthy	50	2022-05-23	30000	sales department +
17006	Amutha	50	2022-06-05	23000	sales department
17009	Arthi	50	2022-06-08	30000	sales department

- **Case with or**: **Query:** select*, case when (department_no =50 or department_no =60) then "yes" else " No" end as Relative_Department from employee_details;



EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join	Relative_Department
17001	Geetha	3001	50	2022-05-10	yes
17002	Guru	3002	50	2022-05-12	yes
17003	Gokul	3003	50	2022-05-15	yes
17005	Moorthy	3005	50	2022-05-23	yes
17006	Amutha	3006	50	2022-06-05	yes
17007	Jaga	3003	70	2022-06-06	No
17008	Pavithra	3007	60	2022-06-07	yes

Result 58

RDBMS with sub Queries:

- **Having clauses** **Basic join triggers** (create table)
- **Having clause:**
- We can fin the date from which is not create as the column of the table fields.
- **Queries**:select employee_details.emp_id,employee_details.employee_name, employee_details.department_no,employee_details.date_of_join, salary_details1.amount, case when (employee_details.department_no =50 and salary_details1.amount>=20000 and employee_details.date_of_join <='2022-06-01') then "sales department +" else "sales department" end as sales_department_high from employee_details inner join salary_details1 on employee_details.emp_id=salary_details1.emp_id where employee_details.department_no=50 and salary_details1.amount>=20000 having sales_department_high= "sales department +" ;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

emp_id	employee_name	department_no	date_of_join	amount	sales_department_high
17003	Gokul	50	2022-05-15	28000	sales department +
17005	Moorthy	50	2022-05-23	30000	sales department +

Basic join trigger (create table): we can create the new table of the query which include the sub query of the table .

- **Query:** create table relative_table as select*, case when (department_no =50 or department_no =60) then "yes" else " No" end as Relative_Department from employee_details; show tables;

Result Grid | Filter Rows: | Expo

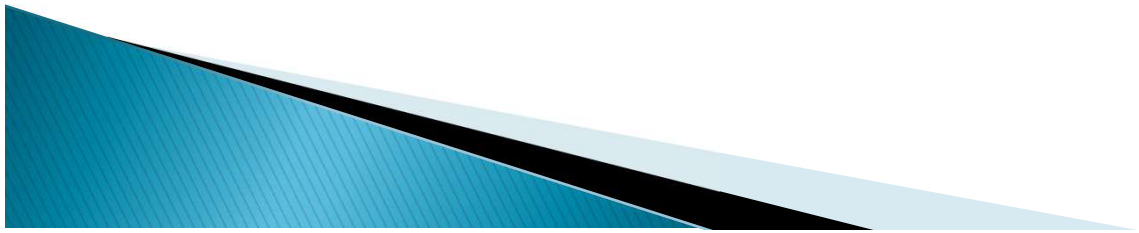
Tables_in_employement_process3
dep_det
destination_det
employee_details
relative_table
salary_details1

Stored Procedure in SQL:

- Stored producer is one of the best way to stored set of code of and execute there in anywhere in the code by call the store procedure .

- **Advantage:**
- Store sum important formulas and method for often usage of that in SQL.

- **Delimiter creation :**
- We have to create the delimiter to because to avoid statement wise run of the code of the that stored procedure.
- Change the Delimiter before Create the Store Procedure also change the default delimiter that(;) for usual run of the code to avoid repeat the delimiter which is created by us before the stored procedure creations.
-

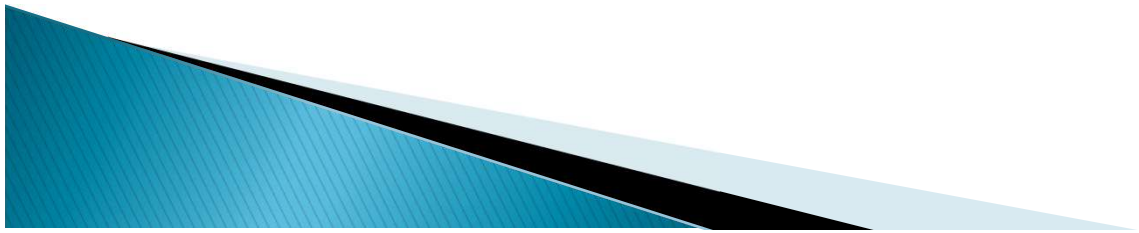


- **Creation:**
- First we have to create the Stored procedure.
- Ex: create procedure X;

- **Begin end:**
- Use begin end in the next step
- And write the set of the code in between
- That begin end

- **Call the procedure:**
- Then we can call that stored procedure in any of the code.

- **Alter Stored Procedures:**
- We an alter that the stored procedure by go to alter option
- And change any data as we want in that altering data.
- And click apply.
- We can see the changes the review.



Stored Procedure:

Query:delimiter //create procedure Y()begin select*from salary_details1 where amount= (select max(amount) as high_salary_of_company from salary_details1);select*, case when date_of_join<='2022-06-01' then "company_seniors" end as senior_staff_of_company from employee_details having senior_staff_of_company ="company_seniors";select *,case when employee_details.department_no=80 and salary_details1.amount >=20000 then "RD_above_20000" end as RD_department_above_2000 from employee_details inner join salary_details1on employee_details.emp_id=salary_details1.emp_id having RD_department_above_2000= "RD_above_20000";end // delimiter ;

Call procedure:

Call Y();

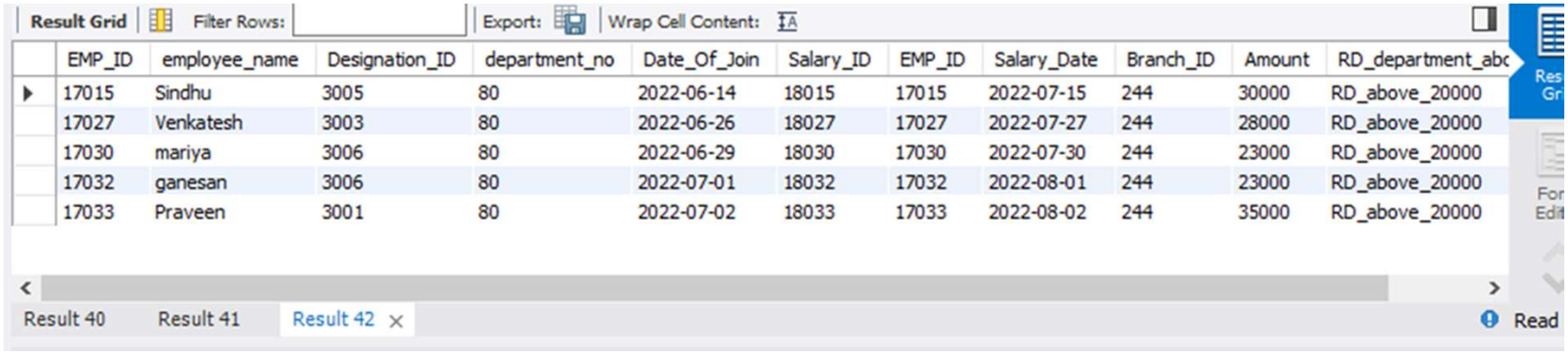
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount
18001	17001	2022-06-10	241	35000
18011	17011	2022-07-11	243	35000
18026	17026	2022-07-26	242	35000
18033	17033	2022-08-02	244	35000

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join	senior_staff_of_company
17001	Geetha	3001	50	2022-05-10	company_seniors
17002	Guru	3002	50	2022-05-12	company_seniors
17003	Gokul	3003	50	2022-05-15	company_seniors
17004	Mani	3004	60	2022-05-20	company_seniors
17005	Moorthy	3005	50	2022-05-23	company_seniors

Result 40 | Result 41 | Result 42

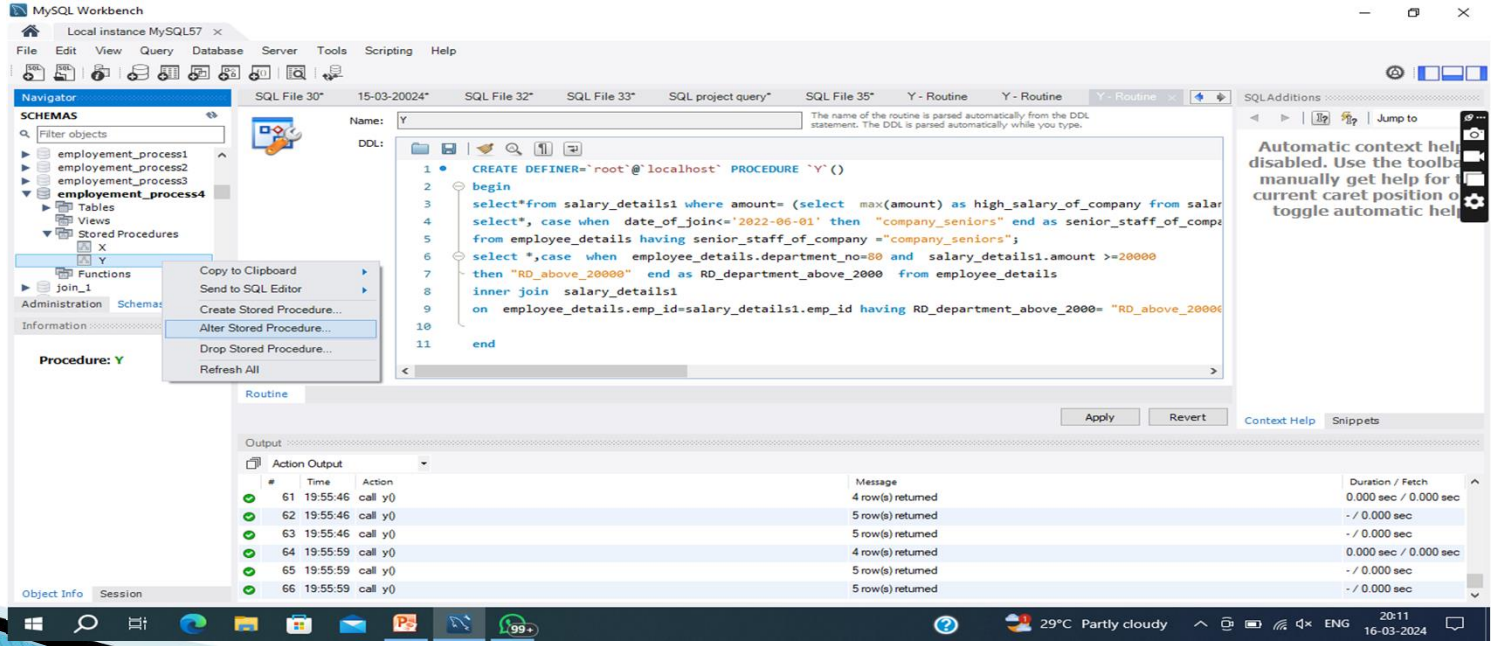


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

EMP_ID	employee_name	Designation_ID	department_no	Date_Of_Join	Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount	RD_department_abc
17015	Sindhu	3005	80	2022-06-14	18015	17015	2022-07-15	244	30000	RD_above_20000
17027	Venkatesh	3003	80	2022-06-26	18027	17027	2022-07-27	244	28000	RD_above_20000
17030	mariya	3006	80	2022-06-29	18030	17030	2022-07-30	244	23000	RD_above_20000
17032	ganesan	3006	80	2022-07-01	18032	17032	2022-08-01	244	23000	RD_above_20000
17033	Praveen	3001	80	2022-07-02	18033	17033	2022-08-02	244	35000	RD_above_20000

Result 40 Result 41 Result 42 Read

Alter Stored Procedure:

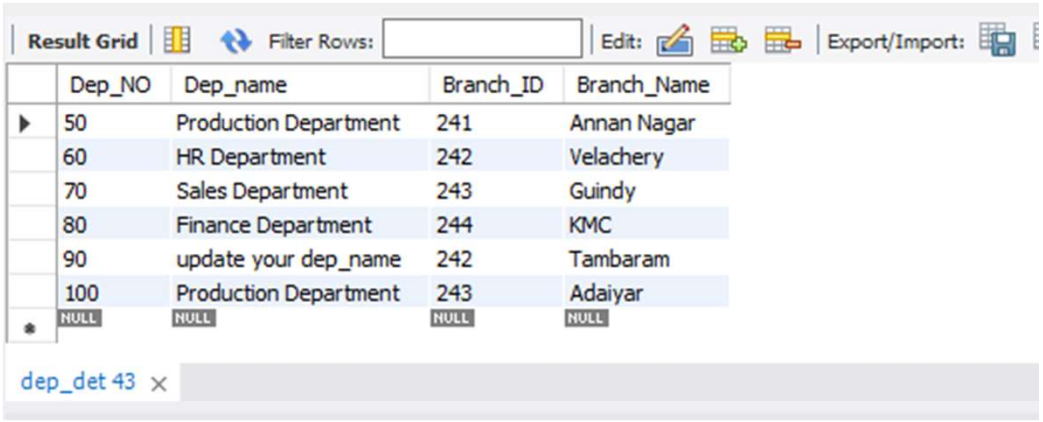


Triggers in SQL

- **Triggers creations:**
- What is trigger A database trigger is a stored program which is automatically fired or executed when some events occur.
-
- **Types of Trigger :**
- **-Row level Trigger** -A event is triggered at row level for each row updated, inserted or deleted.
- **-Statement Level trigger** -An event is triggered at table Level for each sql statement executed

- **Triggers Timings:**
- - before insert
- - after insert
- - before update
- - after update
- - before delete
- - after delete

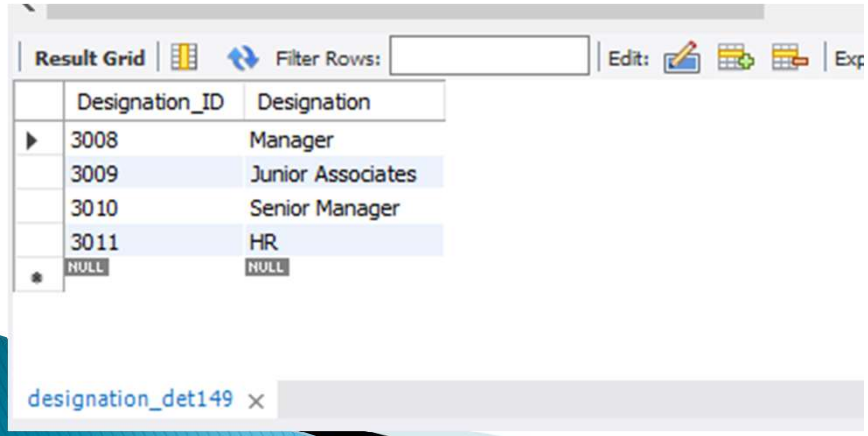
- **Trigger before insert:**
- **Create trigger:**
- **Query:** delimiter // create trigger dep_update before insert on dep_det for each row begin if new.dep_name is null then set new.dep_name ="update your dep_name"; end if ; end // delimiter ;
- **Insert values:**
- **Query:** insert into dep_det values (90,null ,242,'Tambaram'),(100,'Production Department',243,'Adaiyar');
- **Run trigger:**
- **Query:**select*from dep_det;



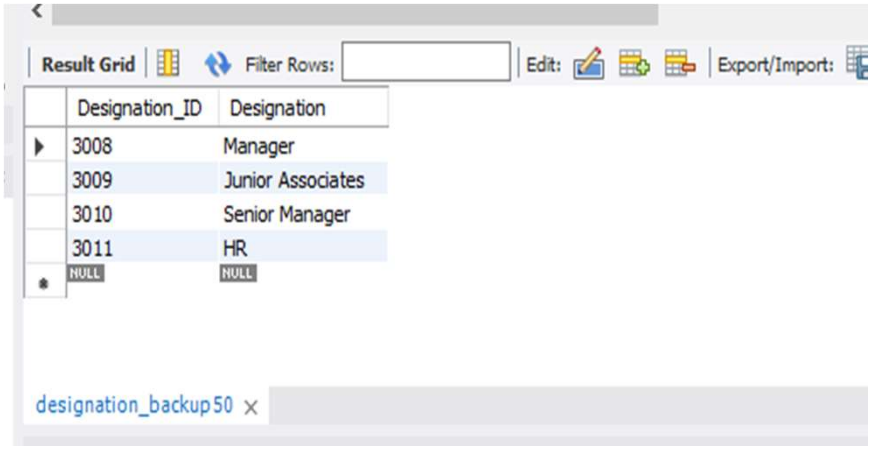
Dep_NO	Dep_name	Branch_ID	Branch_Name
50	Production Department	241	Annan Nagar
60	HR Department	242	Velachery
70	Sales Department	243	Guindy
80	Finance Department	244	KMC
90	update your dep_name	242	Tambaram
100	Production Department	243	Adaiyar
NULL	NULL	NULL	NULL

- **Trigger after insert:**
- **Create trigger:**
- **Query:**delimiter //create trigger designation_update after insert on designation_det1 for each row begininsert into designation_backup(designation_id,designation) values (new.Designation_ID,new.Designation); end //delimiter ;
- Insert values:
- **Query:**insert into designation_det1 values (3008, 'Manager'),(3009, 'Junior Associates'),(3010, 'Senior Manager'),(3011, 'HR');
- **Run trigger:**

Select*from designation_det1;

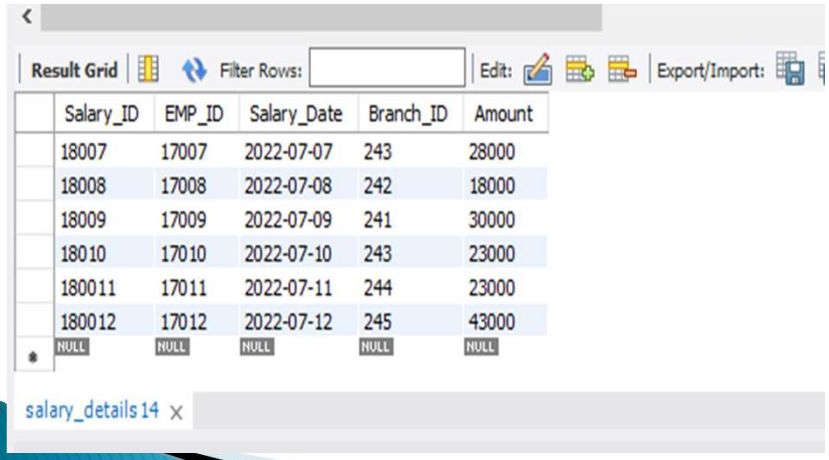


select*from designation_backup;

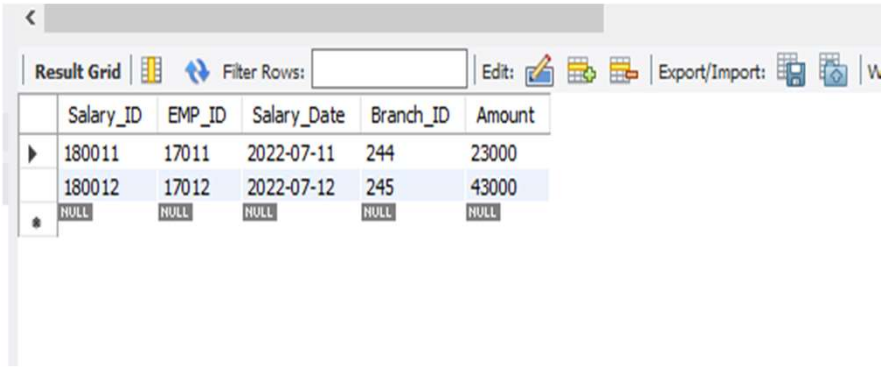


- **Trigger before update:**
- **Create trigger:**
- **Query**:delimiter //create trigger sal_d before update on salary_details for each row begininsert into salary_details_update(salary_id,emp_id,salary_date,branch_id,amount) values (new.salary_id,new.emp_id,new.salary_date,new.branch_id,new.amount); end //delimiter ;
- **Run trigger:**
- update salary_details set salary_id=180011,emp_id=17011,salary_date='2022-07-11',branch_id=244,amount=23000 where salary_id=18001;
- update salary_details set salary_id=180012,emp_id=17012,salary_date='2022-07-12',branch_id=245,amount=43000 where salary_id=18002;

- **Select*from salary_details;**



Select*from salaary_modify;



Trigger after update:

- **Create trigger:**
- delimiter //create trigger SAAA after update on salary_details for each row beginif new.amount >=40000 then insert into salary_modify (salary_id,amount) values (new.salary_id,"High salary");elseif new.amount >=35000 then insert into salary_modify (salary_id,amount) values (new.salary_id,"good salary");elseif new.amount >=15000 then insert into salary_moidfy (salary_id,amount) values (new.salary_id,"average salary");elseif new.amount >=0 then insert into salary_modify (salary_id,amount) values (new.salary_id,"low salary");end if ;end //delimiter ;
- **Run trigger:**
- update salary_details set amount=46000 where salary_id=18002;
- **Select*from salary_modify;**

Result Grid | Filter Rows: | Edit:

salary_id	amount
18002	High salary
NULL	NULL

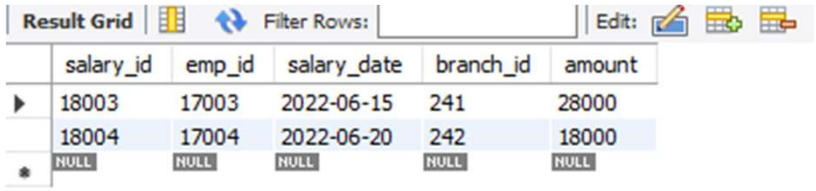
Select *from salary_details;

Result Grid | Filter Rows: | Edit: | Export/Import:

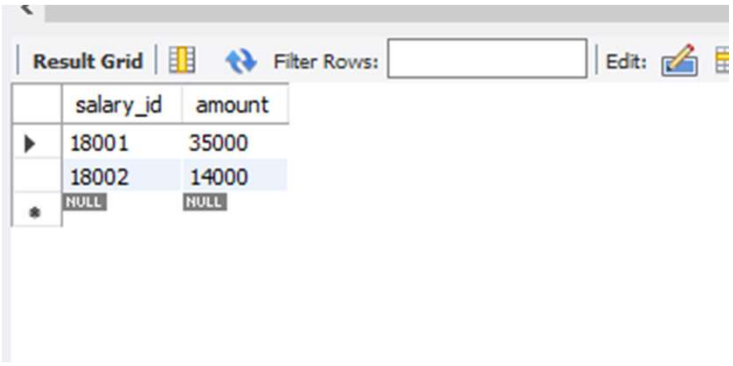
Salary_ID	EMP_ID	Salary_Date	Branch_ID	Amount
18001	17001	2022-06-10	241	35000
18002	17002	2022-06-12	241	46000
18003	17003	2022-06-15	241	28000
18004	17004	2022-06-20	242	18000
NULL	NULL	NULL	NULL	NULL

Trigger Before Delete

- **Create trigger:**
- delimiter //create trigger MNL before delete on salary_details for each row begin insert into salary_delete (salary_id,amount) values (old.salary_id,old.amount); end //delimiter ;
- **Run trigger:**
- **delete from salary_details where salary_id=18001;delete from salary_details where salary_id=18002;**
- **Select*from salary_details;**



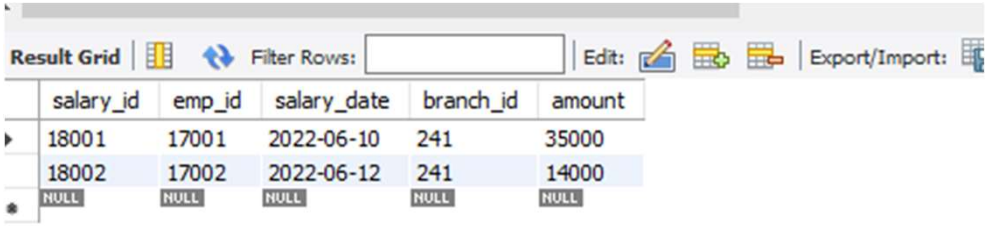
Select*from salary_delete;



Trigger After Delete:

- **Create trigger:**
- delimiter //create trigger MNLOP after delete on salary_details for each row begin if old.amount<=30000 then insert into salary_after_delete (salary_id,amount) values (old.salary_id,old.amount); end if;end //delimiter ;
- **Run trigger:**
- delete from salary_details where salary_id=18003;
- delete from salary_details where salary_id=18004;

- **Select*from salary_details;**



salary_id	emp_id	salary_date	branch_id	amount
18001	17001	2022-06-10	241	35000
18002	17002	2022-06-12	241	14000
NULL	NULL	NULL	NULL	NULL

Select*from salary_after_delete;

salary_id	amount
18003	28000
18004	18000
NULL	NULL


Thanking you....