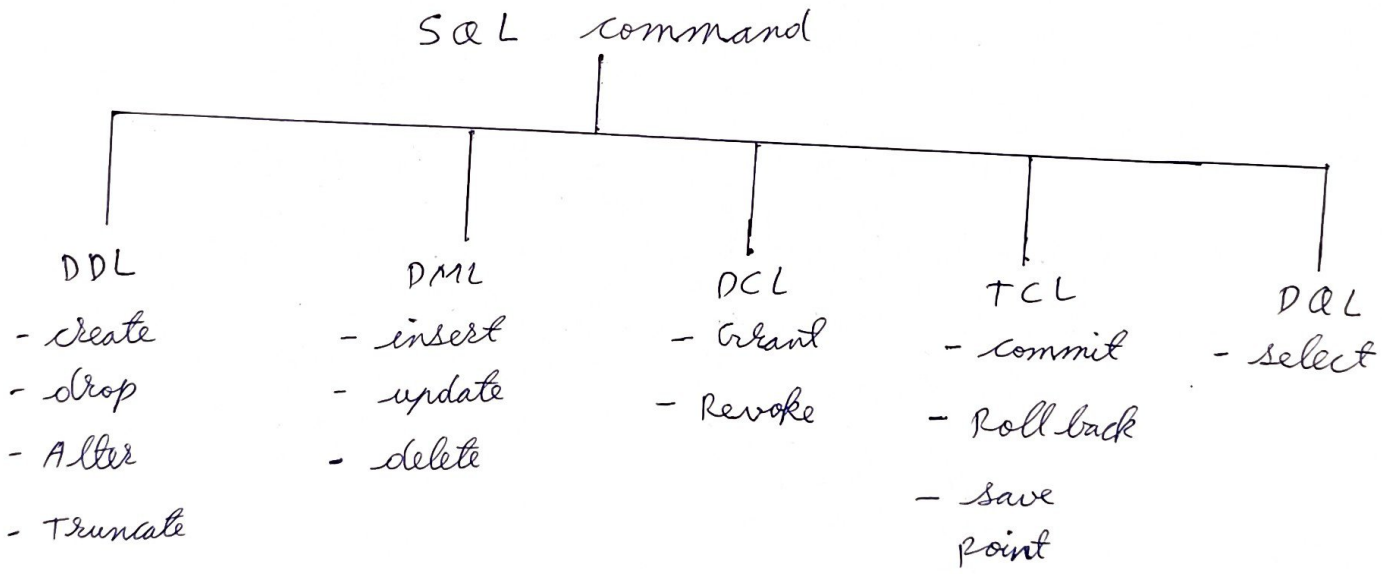
Basic concepts of SQL

Introduction to SQL:

SQL stands for "structured query language". It has become a standard universal language used by most of the relational database management system.
DBMS is a computerized system that enables user to create and maintain a database. The DBMS is general purpose software system that facilitates the processes of defining, constructing, manipulating and sharing databases among various users and applications.
SQL commands are instructions. It is used to communicate with the database and perform specific tasks, functions and queries of data.
SQL can perform various tasks like create, add data, drop, modify the table, set permission for users.



1) Data Definition language (DDL):- These SQL commands are used for creating, modifying and dropping the structure of database objects. The commands are create, Alter, Drop, Rename, Truncate

2) Data Manipulation Language (DML):- These SQL commands are used for storing, retreiving, modifying and deleting data. These commands are select, insert, update and delete.

3) Transaction control language (TCL):- These SQL commands are used for managing changes affecting the data. These commands are commit, Rollback and save point.

4) Data control language (DCL):- These SQL commands are used for providing security to database objects. These commands are Grant and Revoke.

5) Data Query language (DQL):- These SQL commands are used for performing queries on the data within schema objects. These commands are select.

1) create :-

(a) create table : This is used to create a new table

syntax : create table table_name (column1 datatype (size, column2 datatype (size), ...);

Ex : create table student (sno int (3), sname char (10), class char (5));

2) Alter :-

(a) Alter table ... Add : This is used to add some extra fields into existing relation.

syntax : Alter table table_name ADD (new field1 data type (size), new field2 data type (size), ...);

Ex : Alter table student ADD (address char (10));

(b) Alter table ... Modify : This is used to change the width & data type of column of existing table.

syntax : Alter table table_name Modify (column_name new datatype (size), column_name new data type (size)...);

Ex : Alter table student modify (sname varchar (10), class varchar (10));

c) Alter table ... drop : This is used to remove any field of existing relation.

Syntax : Alter table table_name drop column (column_name);

Ex : Alter table student drop column (sname);

---

d) Alter table ... Rename : This is used to change the name of columns in existing relation.

Syntax : Alter table table_name Rename column (old column_name) to (new column_name);

Ex : Alter table student Rename column sname to stud_name);

---

3) Drop :-

(a) Drop table : This is used to delete the structure of a relation. It permently deletes the records in the table.

syntax : Drop table table_name;

Ex : Drop table student ;

Experiment No : 1

1) Implementation of DDL commands of SQL with examples
   - create table
   - Altee table
   - Drop table

1) create a table employee with following schema
   (Emp_no, Emp_name, Emp_address, Emp_ph_no, Dept_no, Dept_name, Job_id, salary)
2) Add a new column HIredate to existing relation
3) change the datatype of Job_ID from char to varchar
4) change the name of column Emp_no to E_no
5) Modify the column width of Job_id of emp table

---

schema diagram :

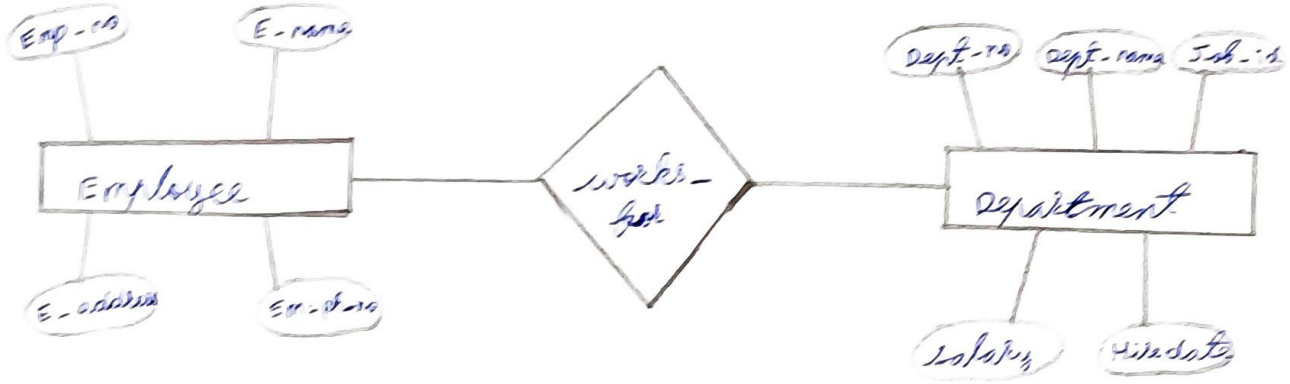
Employee

Emp_no	E_name	E_address	E_pho_no

Department

Dept_no	Dept_name	Job_id	salary	Hiredate

ER-Diagram :



i) create table employee (Emp_no int (10), E-name varchar (20), E-address varchar (20), E-ph-no int (10), dept-no int (10), dept-name varchar (20), Job-id char (5), salary int (10));

Desc employee;

Field	Type	NULL	Key	default	extra
Emp-no	int	Yes		NULL	
E-name	varchar (20)	Yes		NULL	
E-address	varchar (20)	Yes		NULL	
E-ph-no	int	Yes		NULL	
dept-no	int	Yes		NULL	
dept-name	varchar (20)	Yes		NULL	
Job-id	char (5)	Yes		NULL	
salary	int	Yes		NULL	

insert into employee values (1, "sumeet", "abc", 12345, 10, "CSD", 100, 20000), (2, "satvik", "xyz", 56789, 20, "CA", 200, 25000);

select * from employee;

Emp-no	E-name	E-address	E-ph-no	Dept-no	Dept-name	Job-id	salary
1	sumeet	abc	12345	10	CSD	100	20000
2	satvik	xyz	56789	20	CA	200	25000

2) alter table employee add Hiredate date;

select * from employee;

Emp-no	E-name	E-address	E-ph-no	Dept-no	Dept-name	Job-id
1	sumeet	abc	12345	10	CSD	100
2	satvik	xyz	56789	20	CA	200

salary	Hiredate
20000	Null
25000	Null

3) Alter table employee modify column
Job_id varchar(5);

Desc employee;

Field	Type
Emp_no	int
E_name	varchar(20)
E_address	varchar(20)
E_ph_no	int
Dept_no	int
Dept_name	varchar(20)
Job_id	varchar(5)
Salary	int
Hiredate	date

4) Alter table employee rename column
Emp_no to E_no;

select E_no from employee;

E_no
1
2

5) ~~Modify~~ Alter table employee modify
Job_id varchar(10);

1) Insert into :- This is used to add records into a relation.

(a) Inserting a single record:

Syntax : insert into table_Name (column 1, column 2..)
values (data 1, data 2, ...);

Ex : insert into Student (sno, sname, class) values
(1, "Ravi", "CSD");

(or)

Syntax : insert into table_name values
(data 1, data 2, ...);

Ex : insert into Students values (1, "Ravi", "CSD);

(b) inserting all records from another relation

Syntax : insert into table_name 1 select column 1,
column 2, ... from table_name 2 where
column_x = data;

Ex : insert into stud select sno, sname from
student where name = "Ramu";

c) Inserting multiple records:

Syntax: insert into table_name (column1, column2...)
values (data1), (data), ... ;

Ex: insert into student (sno, sname, class)
values (1, "sumeet", "CSD"), (2, "Yeshwant",
"CSD"), (3, "satvik", "CA");

2) update :- This is used to update the content of a record in a table.

Syntax: update table_name set column1 = data,
column2 = data where column_name = data;

Ex: update student set sname = "Kumar"
where sno = 1;

3) Delete :- This is used to delete all the records of a relation but it will retain the structure of that table.

(a) Delete - from: This is used to delete all the records of relation.

Syntax: delete from table_name;

Ex: delete from stud;

(b) Delete - from - where : This is used to delete a selected record from a table.

syntax : delete from table_name where condition;

ex : delete from student where sno = 2;

4) Truncate :- This command will remove the data permanently but structure will not be removed.

syntax : Truncate table table_name;

ex : Truncate table student;

5) select :- To display all fields for all records.

syntax : select * from relation_name;

ex : select * from dept;

(a) select - from : To display a set of columns for all records of table

syntax : select column_name1, column_name_2 . from table_name;

ex : select deptno, deptname from dept;

(b) select - from - where : This is used to display a selected set of columns for selected set of records of table

ex : select * from dept where deptno <= 20;

Experiment No: 2

2) Implementation of DDL commands of SQL with examples.

- insert table
- update table
- delete table

1) create a table Employee with following

(Emp_no, E_name, E_address, E_ph_no, dept_no, dept_name, Job_id, salary)

2) ~~Dropped~~ Display all the information of EMP table

3) Display the record of each employee who works in d10.

4) update the city of Emp_no-12 with current city as Nagpur.

5) Display the details of Employee who works in dept MECH.

6) Delete the ~~email_id~~ E_ph_no of employee James.

7) Display the complete record of employees working in sales dept.

schema diagram:

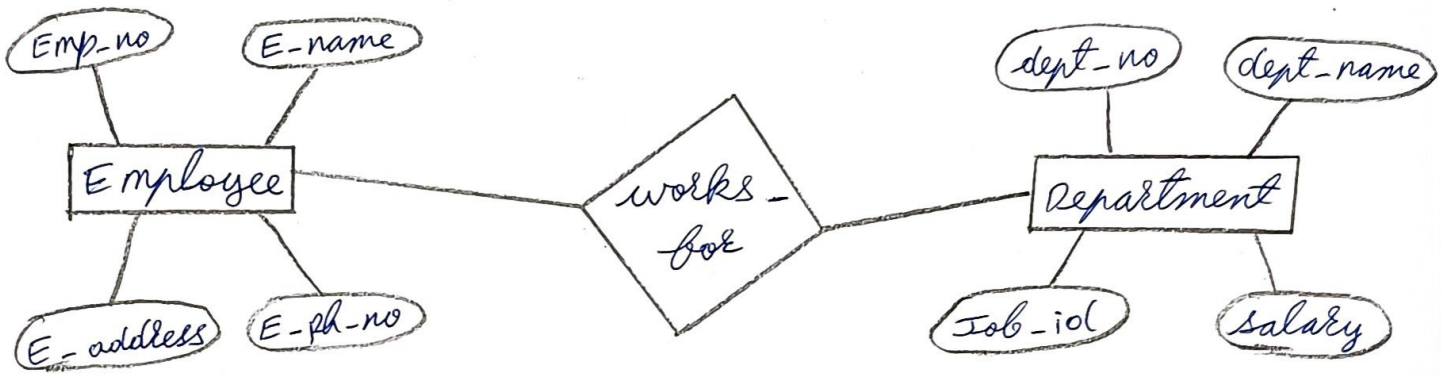
Employee

Emp_no	E_name	E_address	E_ph_no

Department

dept_no	dept_name	Job_id	salary

ER - diagram:



1) create table Employee (Emp_no int (5), E_name varchar(20), E_address varchar (20), E_ph_no int (10), dept_no int (5), dept_name varchar (15), Job_id varchar (10), salary int (10));

insert into Employee (Emp_no, E_name, E_address, E_ph_no, dept_no, dept_name, Job_id, salary)

values (11, "John", "123 main", 1234, 10, "HR", "HR1", 50000),

(12, "James", "456 pine", 4567, 20, "sales", "sal1", 60000),

(13, "sumeet", "P&T", 2345, 10, "HR", "HR2", 80000),

(14, "satvik", "main", 1234, 30, "Mech", "MC1", 70000);

2) select * from employee;

Emp_no	E_name	E_address	E_ph_no
11	John	123 main	1234
12	James	456 pine	4567
13	sumeet	P&T	2345
14	satvik	Main	1234

dept_no	dept_name	Job_id	salary
10	HR	HR1	50000
20	Sales	Sal1	60000
10	HR	HR2	80000
30	MECH	MC1	70000

3) select * from employee where dept_no = "10";

Emp_no	E-name	E-address	E-ph_no
11	John	123 main	1234
13	Sumeet	P&T	2345

dept_no	dept_name	Job_id	salary
10	HR	HR1	50000
10	HR	HR2	80000

4) update employee set E_address = "Nagpur"
where emp_no = 12;

select emp_no, E_address from employee;

emp_no	E_address
12	Nagpur

5) select * from employee where dept_name = "Mech";

Emp_no	E-name	E-address	E-ph-no	dept_no
14	satvik	Main	1234	30

dept_name	Jod_id	salary
MECH	MCI	70000

6) update employee set E-ph-no = NULL where E_name = "James";

select E_name, E_ph_no from employee;

E-name	E-ph-no
James	NULL

7) select * from employee where dept_name = "sales";

Emp_no	E-name	E_address	E_ph_no
12	James	Nagpur	NULL

Dept_no	Dept-name	Job_id	salary
20	sales	sal I	60000

Experiment No: 3

3) Implementation of different types of functions with examples

- Number Function
- Aggregate function
- character function
- conversion function
- Date function

Number function :-

1) Abs(n) : select abs(-15);

   → 15

2) Exp(n) : select exp(4);

   → 54.5981

3) Power(m,n) : select power(4,2);

   → 16

4) Mod(m,n) : select mod(10,3);

   → 0

5) Round(m,n) : select round(100.256, 2);

   → 100.26

6) Truncate(m,n) : select truncate(10.256, 2);

   → 100.25

7) sqrt(m,n) : select sqrt(4);

   → 2

8) ceiling(m,n) : select ceiling(25.75);

   → 26

Aggregate function :-

1) count : If distinct keyword is used then it will return only the count of unique tuple in the column.

Syntax : select count (column_name) from relation;

ex : select count (salary) from emp;

2) Sum : It returns the sum of all the values in that column.

Syntax : select sum (column_name) from relation;

ex : select sum (salary) from emp;

3) Avg : It returns the average value of that column values.

Syntax : select AVG (n1, n2...) ~~from relation~~;

ex : select AVG (10, 15, 30) ~~from~~;

4) Max : It returns maximum value of that column.

Syntax : select max (column_name) from relation;

ex : select max (salary) from emp;

5) Min : It returns minimum value of that column.

Syntax : select Min (column_name) from Ellation;

ex : select min (salary) from emp;

character function :-

1) lower (char): select lower ("hello");

→ hello

2) upper (char): select upper ("Hello");

→ HELLO

3) ltrim (char, [set]): select ltrim ("cseit", "cse");

→ cse

4) rtrim (char [set]): select rtrim ("cseit", "it");

→ it

5) Replace (char, search): select replace ("Jack and Jue",
"J", "bl");

→ black and blue

string function :-

1) concat : select concat ('sum', 'eet');

→ sumeet

2) Lpad : select Lpad ('sumeet', 10, '*');

→ ****sumeet

3) Rpad : select Rpad ('sumeet', 10, '*');

→ sumeet****

4) length : select length ('DBMS');

→ 4

5) substr : select substr ('abcdef', 3, 4);

→ cdef

Date function :-

1) Sysdate : select sysdate();

-> 2024-01-09 12:30:40

2) date_diff : select date_diff ("2024"-01-09",
"2003-10-02");

-> 7404

3) date_add : select date_add ("2024-01-09", interval
10 day);

-> 2024-01-19

4) select date_format ("2024-01-09", "%d %M %Y");

-> 14 Jan 2024

5) Dayname : select dayname ("2024-01-09");

-> Tuesday

6) extract : select extract (month from "2024-01-14");

-> 1

7) Last_day : select last_day ("2024-01-14");

-> 31

8) Make date : select makedate (2024, 14);

->2024-01-14

9) select period_add (201703, 5);

-> 201708

10) select Year_week ("2024-01-15");

-> 202402

* write sql query for the following

1) create a table Employee with following schema

(Emp_no , E_name, E_address, E_ph_no, dept_no,
dept_name, Job_id, designation, salary,
Joining_Year)

2) List the Emp_no, E_name, salary of all employees working for manager

3) Display all the details of employee whose salary is more than salary of IT profb.

4) List the employees in ascending order of designation of those joined after 1981 2020

5) List the employees who are either clerk. or Analyst

6) List the employee who joined on 1980, 1981

7) List the employees who are working for dept_no .101 to 103

8) List the E_names those name start with S

9) List all emp's except except president & MGR in asc order of salaries

schema diagram :

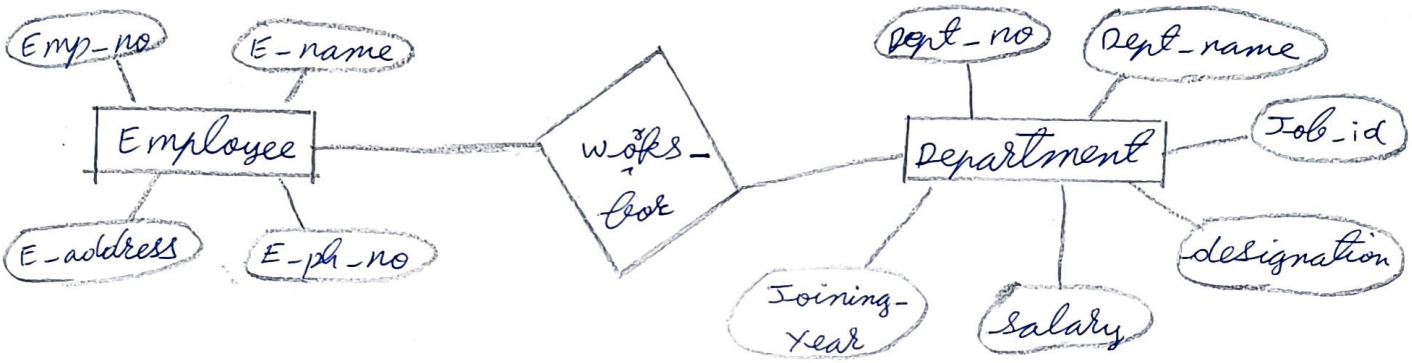
Employee

Emp_no	E_name	E-address	E-ph_no

Department

Dept_no	Dept_name	Job_id	Designation	salary	Joining-Year

ER - diagram :



1) create table Employee (Emp_no int(5), E_name varchar(20), E-address varchar(20), E-ph_no int(10), dept_no int(10), Dept_name varchar(20), Job_id int(10), Designation varchar(20), Salary int(10), Joining-Year Year);

insert into Employee (Emp_no, E_name, E_address, E_ph_no, Dept_no, Dept_name, Job_id, designation, salary, Joining_Year) values

(1, "Satvik", "abc", 91136, 101, "CA", 201, "clerk", 60000, 2020),

(2, "Sumeet", "XYZ", 84311, 102, "security", 202, "Manager", 80000, 2025),

(3, "Yash", "MNO", 1234, 103, "security", 203, "Manager", 70000, 2026),

(4, "Ramesh", "PQR", 1122, 104, "security", 204, "Analyst", 60000, 2015),

(5, "Abhi", "EFG", 2345, 105, "PDA", 205, "IT profff", 75000, 2005),

(6, "John", "XYZ", 9876, 106, "company", 206, "President", 100000, 1980),

(7, "James", "abc", 5678, 107, "company", 207, "MGR", 90000, 1981);

select * from Employee;

Emp_no	E_name	E_address	E_ph_no	Dept_no
1	satvik	abc	91136	101
2	Sumeet	XYZ	84311	102
3	Yash	MNO	1234	103
4	Ramesh	PQR	1122	104
5	Abhi	EFG	2345	105
6	John	XYZ	9876	106
7	James	abc	5678	107

Dept_name	Job_id	Designation	salary	Joining_Year
CA	201	clerk	60000	2020
security	202	Manager	80000	2025
security	203	Manager	70000	2026
security	204	Analyst	60000	2015
PDA	205	IT profff	75000	2005
company	206	President	100000	1980
company	207	MGR	90000	1981

2) select Emp_no, E_name, salary from Employee where designation = 'manager';

Emp_no	E_name	Salary
2	Sumeet	80000
3	Yash	70000

3) select * from Employee where salary > (select Max(salary) from employee where designation = 'IT proff');

Emp_no	E_name	E_address	E_ph_no	dept_no
2	sumeet	XYZ	84311	102
6	John	XYZ	9876	106
7	James	abc	5678	107

dept_name	Job_id	Designation	salary	Joining_Year
security	202	Manager	80000	2025
company	206	President	100000	1980
company	207	MGR	90000	1981

4) select * from Employee where Joining_Year > '2020' order by Designation ASC;

Emp_no	E_name	E_address	E_ph_no	Dept_no
2	sumeet	XYZ	84311	102
3	Yash	MNO	1234	103

Dept_name	Job_id	Designation	salary	Joining_Year
security	202	Manager	80000	2025
security	203	Manager	70000	2026

4) ~~list the employees who are either clerk or analyst~~

5) select * from Employee where designation in (`clerk', `analyst');

Emp_no	E-name	E_address	E-ph-no	Dept_no
1	satvik	abc	91136	101
4	Ramesh	PQR	1122	104

Dept_name	Job_id	Designation	salary	Joining_Year
CA	201	clerk	60000	2020
security	204	analyst	60000	2015

6) select * from Employee where Joining_da/Year in (`1980', `1980');

Emp_no	E_name	E_address	E-ph_no	Dept_no
6	John	XYZ	9876	106
7	James	abc	5678	107

Dept_name	Job_id	Designation	salary	Joining_Year
company	206	President	10000	1980
company	207	MGR	90000	1981

7) select * from Employee where dept_no in (101, 102, 103);

Emp_no	E_name	E_address	E_ph_no	Dept_no
1	satvik	abc	91136	101
2	Sumeet	XYz	84311	102
3	Yash	MNo	1234	103

Dept_name	Job_id	Designation	Salary	Joining_Year
CA	201	clerk	60000	2020
security	202	Manager	80000	2025
security	203	Manager	70000	2026

8) select E_name from employee where E_name like 'S%';

E_name
satvik
Sumeet

9) select * from Employee where designation not in ('President', 'MGR') order by salary ASC;

Emp_no	E_name	E_address	E_ph_no	Dept_no
1	Satvik	abc	91136	101
4	Ramesh	PQR	1122	104
3	Yash	MNO	1234	103
5	Abhi	EFG	2345	105
2	Suneet	XYZ	84311	104

Dept_name	Job_id	Designation	salary	Joining_Year
CA	201	clerk	60000	2020
security	204	Analyst	60000	2015
security	203	Manager	70000	2026
PDA IT proff	205	IT proff	75000	2005
security	202	Manager	80000	2025

Experiment No - 4

4) Implementation of different types of operators in SQL

- Arithmetic operator
- Logical operator
- comparison operator
- special operator
- set operator

Arithmetic operator :

1) Addition (+) : syntax : select column1 + column2 from relation;

ex : select salary + bonus as Total_salary
from employee;

2) subtraction (-) :

syntax : select column1 - column2 from relation;

ex : select salary - 500 as update_salary
from employee;

3) Multiplication (*) :

syntax : select column1 * column2 from relation;

ex : select bonus * panelty as extra_money
from employee;

4) Division (/) :

syntax : select column1 / column2 from relation;

5) Power (^) :

syntax : selct column1^ column2 from relation;

6) Modulus (%) :

syntax : select column1 % column2 from relation;

Logical operators:

1) AND : It is used in existence of multiple condition in where clause.

2) OR : used to combine multiple condition in where clause.

3) NOT : It reverses the meaning of logical operator with which it is used.

comparison operators:

1) = : checks if values of two operands are equal or not, if yes then condition becomes true.

2) != : checks if values of two operands, if values are not equal then condition becomes true.

3) > : checks if the values of left operand is greater than values of right operand.

4) < : checks if the values of left operand is less than value of right operand.

5) >= : checks if the value of left operand is greater than or equal than the value of right operand.

6) <= : checks if the value of left operand is less than or equal than the value of right operand.

Special operator:

1) Between : It is used between two values to search for max & minimum value.

2) Is NULL: It is used to compare a value with NULL attribute value.

3) All : It is used to compare a value to all values in another set.

4) Any : used to compare a value to any applicable value in the list according to condition.

5) Like: used to compare a value to similar values using wild card operators. It allows % and _ to match a given string pattern

6) In : It is used to compare a value to a list of literal values that have been specified

7) Exist : It is used to search for presence of a row in a specified table.

set operators:

1) union : Returns all distinct rows selected by both the queries

Syntax : select * from table_name1 union
select * from table_name2;

Ex : select * from Employee union
select * from dept;

2) union all : Returns all rows selected by either query including duplicates.

syntax : select * from table_name1 union all select * from table_name2;

ex : select * from student union all select * from dept;

3) Intersect : Returns row selected that are common to both queries

syntax : select * from table_name1 intersect select * from table_name2;

ex : select * from Employee intersect select * from Dept;

4) Minus : Returns all distinct rows selected by first query and not by second

syntax : select * from table_name1 minus select * from table_name2;

ex : select * from Emp1 minus select * from Emp2;

* write the SQL query for the following

1) create a Employee table and dept table with following schema

(Emp_no, Emp_name, dept_no)

(dept_no, dept_name)

2) Display all the dept_no's available with dept and employee tables avoiding dulicates

3) Display all the dept_no's available in Employee and not in dept tables and vice verse

4) Display all dept_no's available with dept and employee tables

schema diagram :

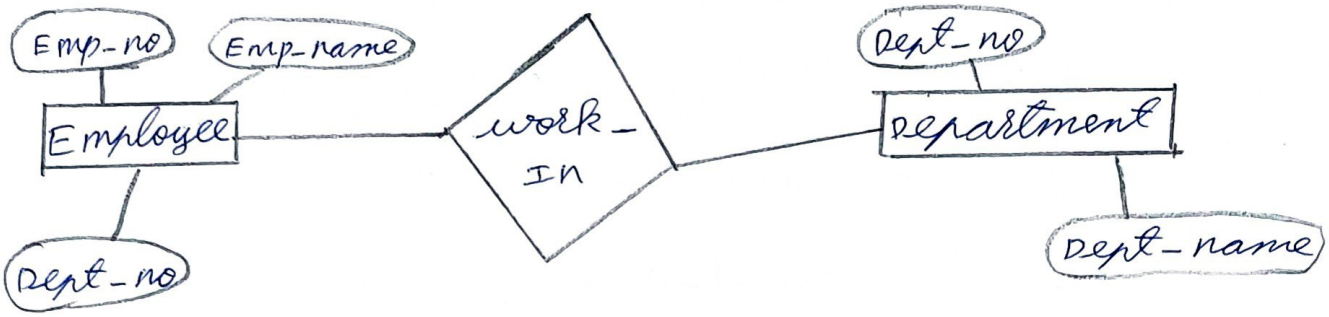
Employee

Emp_no	Emp_name	Dept_no

Department

Dept_no	Dept_name

ER - diagram:



1) create table Employee (Emp_no int (10),
Emp_name varchar (20), Dept_no int (10));

create table dept (dept_no int (10),
dept_name varchar (20));

insert into Employee values (1, "Sumeet", 10),
(2, "Satvik", 20), (3, "Yash", 30),
(4, "Abhi", 40);

insert into dept values (10, "CSD"), (11, "CSE"),
(20, "EC"), (21, "EE"), (30, "MECH"),
(31, "IS");

select * from Eemployee;

Emp_no	Emp_name	Dept_no
1	Sumeet	10
2	Satvik	20
3	Yash	30
4	Abhi	40

select * from Dept;

Dept_no	Dept_name
10	CSD
11	CSE
20	EC
21	EE
30	MECH
31	IS

2) select dept_no from dept union
select dept_no from Employee;

Dept_no
10
11
20
21
30
31
40

3) select distinct dept_no from employee where dept_no not in (select dept_no from dept);

dept_no
40

4) select distinct dept_no from dept where dept_no not in (select dept_no from employee);

dept_no
11
12
13

4) select dept_no from dept union all select dept_no from employee;

dept_no
10
11
20
21
30
31
10
20
30
40

5) Implementation of different types of Joins

- Inner Join
- outer Join
- Natural Join
- Full Join

Inner Join:- It selects allows from both tables as long as the condition is satisfied. This will combine all rows from both tables where condition satisfies.

Syntax:

```
select table1.column1, table1.column2,
       table2.column1, ....
From table1
inner Join table2
on table1.matching_column = table2.matching_column;
```

Ex:

```
select studentCourse.course_ID, student.Name,
       student.Age from student
inner Join studentcourse
on student.ROLL_No = studentCourse.ROLL_No;
```

Left Join :- This returns all rows of table on left side of Join and matches rows for the table on right side of Join.

Syntax:

```
select table1.column1, table1.column2, table2.column2,..
 from table1
 left Join table2
on table1.matching-column = table2.matching_column;
```

Right Join:- This returns all rows of table on right side of Join and matching rows for table on left side of Join.

Syntax:

```
select table1.column, table1.column2,...
 from table1
 Right Join table2
on table1.matching_column = table2.matching_column;
```

Ex:

```
select student.Name, student_course.courseID
 from student
 Right Join studentcourse
 on studentcourse.Roll_No = student.Roll_no;
```

Full Join :- It creates the result set by combining results of both left and Right Join. It will contain all rows from both tables.

Syntax :

```
select table1.column1, table2.column2,...
  from table1  Full Join table2
  on table1.matching_column = table2.matching_column;
```

Ex: select student.Name, studentcourse.course_ID
  from student full Join studentcourse
  on studentcourse.Roll_no = student.Roll_no;

Natural Join :- It can Join tables based on common columns in tables being Joined. It returns all rows by matching values and data type of a column.

syntax :

```
from table1  Natural Join table2;
```

Ex :

```
select * from employee  Natural Join
      department;
```

Equi Join:- It creates a Join for matching column values of relative tables. It also creates Join by providing names of columns using equal(=) sign.

Syntax:

```
select column_list
from table1, table2,....
where table.column_name = table2.column_name;
```

Ex:

```
select student.name, record.class, record.city
 from student, record
where student.city = record.city;
```

Non-Equi Join:- It performs a Join using comparsion operator other than (=) sign like >, <, >=, <= with condition.

Syntax:

```
select * from table1, table2
where table1.column > table2.column;
```

Ex:

```
select student.name, record.id,
 record.city from student, record
 where student.id < record.id;
```

* Write the query for the following

1) consider the following schema:

Sailors (sid, sname, rating, age)

Boats (bid, bname, color)

Reserves (sid, bid, day (date))

2) Find all information of sailors who have reserved boat number 101

3) Find the name of boat reserved by Abhi

4) Find the names of sailors who have reserved a red boat and list in the order of age.

5) Find the names of sailors who have reserved at least one boat

6) Find the id's and names of sailors who have reserved two different boats on same day.

7) Find the iol's of sailors who have reserved a red boat or green boat

8) Find the name and age of the youngest sailor

9) count the number of different sailor names

10) Find the average age of sailors for each rating level

11) Find the average age of sailors for each rating level that has at least two sailors.

1) create table sailors (siol int(5), sname varchar(20),
rating int(10), age int(5));
create table boats (biol int(5), bname varchar(20),
color varchar(20));
create table reserves (siol int(5), bid int(5),
day date);

insert into sailors (siol, sname, rating, age)
values (1, "Sumeet", 9, 25), (2, "Satvik", 8, 30),
(3, "Abhi", 7, 20), (4, "Yeshwant", 7, 22);

insert into boats (biol, bname, color) values
(101, "going - merry", "green"), (102, "sunny", "blue"),
(103, "speedy", "red"), (104, "voyager", "yellow");

insert into Reserves (siol, bid, day) values
(1, 101, "2025-08-02"), (2, 102, "2024-02-07"),
(3, 103, "2024-02-09"), (4, 104, "2024-05-18"),
(1, 102, "2026-09-12"), (2, 103, "2028-06-25");

select * from sailors;

sid	sname	Rating	Age
1	Sumeet	9	25
2	Satvik	8	30
3	Abhi	7	20
4	Yeshwant	7	22

select * from Boats;

bid	bname	color
101	going-merry	green
102	Sunny	blue
103	Speedy	Red
104	voyagar	yellow

sllect * from Reserves;

sid	bid	day
1	101	2025-08-02
2	102	2024-02-07
3	103	2024-02-09
4	104	2024-05-18
1	102	2026-09-12
2	103	2028-06-25

2) select * from sailors where sid in
(select sid from reserves where bid = 101);

sid	sname	rating	age
1	Sumeet	9	25

3) select b.bname from boats b join reserves r
on b.bid = r.bid Join sailors s on
s.sid = s.sid where s.sname = "Abhi";

bname
speedy

4) select s.sname from sailors s Join reserves r
on s.sid = r.sid Join boats b on r.bid = b.bid
where b.color = "red" order by s.age;

sname
Abhi
satvik

5) select distinct s.sname from sailors s
Join Reserves r on s.sid = r.sid;

sname
Sumeet
satvik
Abhi
Yeshwant

6) select S.Sid, S.Sname from Sailors S Join Reserves r1 on S.Sid = r1.Sid Join reserves r2 on S.Sid = r2.Sid where r1.day = r2.day and r1.bid <> r2.bid;

Empty set

7) select distinct S.Sid from Sailors S Join reserves r on S.Sid = r.Sid Join boats b on r.bid = b.bid where b color in ("red", "green");

Sid
1
2
3

8) select Sname, age from Sailors order by age limit 1;

Sname	Age
Abhi	20

9) select count(distinct sname) as num_sailors from sailors;

num_sailors
4

10) select rating, avg(age) as avg_age from sailors group by rating;

rating	avg_age
9	25
8	30
7	21

11) select rating, avg(age) as avg_age from sailors group by rating having count(sid)>=2;

Rating	avg_age
7	21