- SELECT COUNT(CITY)
  FROM STAFF
  WHERE CITY = "BELGAUM"
  GROUP BY CITY;

- SELECT NAME
  FROM
  ~~GI~~ WHERE NAME LIKE "B%";

- SELECT City, COUNT(DEPT)
  FROM STAFF
  GROUP BY CITY;

- JOINS
- Natural :- Same name
- Equi~~valent~~ :- DIFF name
- self
- Conditional
- Inner
- Outer - left, Right,

Join = Cross-product + condition.

Emp

Eno	Ename	
1	Ram	~~Delhi~~ Delhi
2	Varun	~~Chandigarh~~ Chd.
3	Ravi	Chd
4	Amrit	Delhi

Dept

Deptno	Name	Eno
D_1	HR	1
D_2	IT	2
D_3	MRKT.	4

```
SELECT Address
FROM Emp
Where Ename = "Varun" ;
```

Retrive the address & Dept- of the emp whose name is varun.

```
SELECT Address , Name
FROM Emp , Dept .
WHERE Ename = "Varun" and Eno = Eno ;
```

```
SELECT   Address , Name.
FROM     Emp E , Dept D.      # Emp as E , Dept a
WHERE    Ename = 'Varun' and E.Eno = D.Eno;
```

- Find emp names. who is working in a department

→
```
SELECT   Ename , Name
FROM     Emp E , Dept D
WHERE    E.Eno = D.Eno ;
```

—

ALTER TABLE <TABLE-NAME> ADD PRIMARY KEY (<COLUMN_NAME>)

ALTER TABLE <CHILD TABLE_NAME> ADD FOREIGN KEY (<CHILD COLUMN_NAME>) REFERENCES <PARENT TABLE_NAME> (<PARENT COLUMN_NAME>)

Person

D_id	name	add
1	aaa	Bgm
2	bbb	Pwd.

Car

regno	model	year
11	aa	2000
12	bb	2005

Accident

regno	acc_date	loc
111	12-DEC-2018	Bgm
123	13-SEP-2020	Bgl

Owns

driver_id	reg_no
1	11
1	11
2	13

Participated

Did regno rep_num damage_amt

- CREATE TABLE PERSON
  (d_id int,
  name varchar(20),
  ~~ddd~~ add varchar(20),
  PRIMARY KEY (d_id)
  );

- CREATE TABLE CAR
  (regno int, model varchar(20), year int
  PRIMARY KEY (regno));
- CREATE TABLE ACCident
  (

Find the driver name and the model of the car which own by them.

→
```
SELECT P.Name, C.Model
FROM Person P, Car C, Owns O
WHERE P.D-id = O.D-id AND O.Reg-No =
      C.regno;
```

Update the damage-amt for the car with specific regno in the accident with repno = 12 to 25000

```
UPDATE Participated
SET Damage-amt = 25000
WHERE Rep-no = 12 and regno = '5';
```

Find the total no of people who owned cars that were involved in accidents in the year xxxx

→
```
SELECT COUNT (DISTINCT O.D-id) AS People
FROM OWNS O, Participated P, Accident A
WHERE A.ACCD-Date LiKe '%08' AND
      O.Reg no = P.Regno AND P.rep-no = A.
      repno;
```

- Find the number

```
SELECT  COUNT(*)
FROM.   CAR C, Participated P
WHERE   C.Regno = P.regno AND C.MODEL = 'ALTO';
```

STUDENT

RNO	Name	Branch	marks
1	Ravi	CSE	89
2	Amrit	CSE	96
3	Amit	ECE	86
4	Asha	EISE	92
5	Rajesh	ISE	65

[i] Count the number of students branch wise where marks are greater than 50.

→
```
SELECT BRANCH, COUNT(*)
FROM STUDENTS
WHERE MARKS > 50
GROUP BY BRANCH;
HAVING MARKS > 50;
```

[ii] Count the number of students in CSE whose marks are greater than 90

→
```
SELECT Name, COUNT(*)
FROM STUDENTS
WHERE marks > 90 and Branch = 'CSE';
```

OR

```
SELECT . Name, COUNT(*)
FROM STUDENT  → WHERE  marks > 90
GROUP BY BRANCH
HAVING BRANCH IN 'CSE';
```

(iii) Count the number of students in ISE & CSE whose marks are greater than 50

→
```
SELECT Branch, COUNT(*)
FROM STUDENT
WHERE marks > 50
GROUP BY Branch
HAVING Branch IN ('CSE', 'ISE');
```

(iv) Find the total number of student in each department.

→
```
SELECT Branch *, COUNT(*)
FROM STUDENTS
GROUP BY BRANCH WHERE ~~COUNT~~;
~~HAVING COU BRANCH~~
```

(v) Find the total marks of students in each branch

```
SELECT Branch, SUM(Marks)
FROM students
GROUP BY Branch;
```

(v) Find the avg marks of students of each branch except ECE and sort results in the decreasing order

→
```
SELECT Branch, AVG(MARKS)
FROM Students
GROUP BY Branch
Having Branch != 'ECE'                    // <>
ORDER BY AVG(marks) DESC;
```

(vi) Count the number of students from each branch where name of the student does not start with A

→
```
SELECT Branch, COUNT(*)
FROM Students
WHERE Name not like 'A%'
GROUP BY Branch;
```

(vii) Display the sum of all marks except ECE

→
```
SELECT Branch, SUM(Marks)
FROM Students
WHERE Branch != 'ECE';
```

~~Find Students~~ -

~~Student~~ Enrolled

Sid	Cid	Since
S1	C1	2016.
S2	C2	2017
S1	C2	2017

[i] Finds the students who has enrolled in atleast two course.

→ SELECT ~~Name~~ E1.Sid // Self joint
FROM Enrolled as E1, Enrolled as E2
~~GROUP BY~~ Sid
Having WHERE E1.Sid = E2.Sid AND E1.Cid != E2.Cid;

S1 C1 S1 C1 ×
S1 C1 S2 C2 ×
S1 C1 S1 C2 ✓
S2 C2 S1 C1 ×
S2 C2 S2 C2 ×
S2 C2 S1 C2 ×
S1 C2 S1 C1 ✓
S1 C2 S2 C2 ×
S1 C2 S1 C2 ×

[ii]

→ Select Name
FROM Employee
WHERE Loc LIKE

Employee

Eno	Ename	Add
1	Ram	Delhi
2	Varun	chd
3	Ravi	chd
4	Amit	Delhi

Dept

Eno	D.NO	Location
1	D_1	Delhi
2	D_2	Pune
4	D_3	Bang

(i) Find employee name who worked in dept having location same as their address

→ SELECT E.Name (Equi-Join)
FROM Employee E, Dept D
WHERE E.Eno = D.Eno and E.Add = D.Location;

COLLEGE DATA

(i) Retrive the name and percentage of all students for the course 16CS45

Student (USN, Name, Branch, Percentage)
Faculty (Fid Fname, Dept, Desi, sal)
Course (Cid, Cname, Fid)
Enroll (Cid, USN, grade)

```
Select S.Name, S.Percentage
From Student S, Course C, Enroll E
WHERE S.USN = E.USN and C.Cid = E.Cid and
      C.Cname = '16CS45';
```

(ii) List the Dept having an average salary of the faculties above Rs 30 000.

→
```
Select Dept, avg(sal)
FROM Faculty
Group By Dept
having avg(sal) > 30000;
```

(iii) List name of the course having students grade 'A' maximum

→
```
Select E.Cid, C.Cname,
From Enroll E, Course C.
```

Group By C.Cid = C.Cid
having E.grade = 'A'

OR

```
Select CName, Max(grade)
From Course C, Enroll E
Where C.Cid = E.Cid
Group By CName;
```

NESTED QUERY / SUB QUERY / INNER QUERY

```
Select column 1, ... 2
From table_name
where condition operators (Select column 1... 2
                           From table name
                           where condition);
```

Order by clause cannot be used in inner query.

Eid	Ename	Did	Dname	Gid
1	A	D1	HR	1
2	B	D2	MRKT	2
3	C	D3	IT	4
4	D			
5	E			

classmate

Date
Page

IN = OR

IN Comparing multiple value.
= Comparing single value

→
```
SELECT *
FROM EMP E, DEPT D
WHERE E.Eid = D.Eid ;
```

```
Select *
FROM EMP                {1, 2, 4}
WHERE Eid IN (Select Eid From dept);
```

Write a SQL query to display the maximum salary and also display the ~~max~~ name of the employee who is taking that maximum salary.

→ Emp

Eid	Ename	Dept	Salary
1	Ram	HR	10,000
2	Amrit	MRKT	20,000
3	Ravi	HR	30,000
4	Nibin	MRKT	40,000
5	Varun	IT	50,000

```
Select * Max(Salary)
From Emp
Select Ename, Salary.
From Emp
Where Salary = (Select max(Salary) From Emp);
```

Write the SQL query to display second highest salary from Emp table

→ Select Ename, max(salary)
From Emp
Where salary != (Select max(salary) From Emp);

Write the SQL query to display employee name who is taking second highest salary.

- Write the SQL query to display all the details of the emp where number of employees are less than 2

→ SELECT Dept, COUNT(Ename)
FROM Emp
GROUP BY Dept
HAVING COUNT(*) < 2 ;

SELECT *
FROM Emp
WHERE Dept IN (Select Dept FROM Emp
GROUP BY Dept
HAVING COUNT(*) < 2);

- Write the query to display highest salary dept wise and name of the employee who is drawing that salary

```
SELECT  Ename , Dept , Salary
FROM  Emp
WHERE  Ename IN (SELECT Ename FROM Emp
                 GROUP BY Dept
                 HAVING MAX(salary));
```

- List all the players whose injury severity level is high and who's name starts with "H"

→
```
SELECT * P.Pname
FROM player P , Injury , I
WHERE P.Pid = I.Pid AND severity = 'High'
      AND P.Pname LIKE 'H%' ;
```

- Count the Host teams, those who have participated more than one game

→
```
SELECT HTeam
FROM Game
GROUP BY HTeam
WHERE COUNT(*) > 1 ;
```

- List the name of the Coach in ascending order

→
```
SELECT COACH
FROM TEAM
OREDER BY COACH ;
```

- List the names of the Players who started their career in Hockey league from year 2000 to 2010

→ SELECT Name
FROM PLAYER
WHERE FROMDATE BETWEEN '2000' AND '2010'

OR

WHERE FROMDATE LIKE '2000' '------'
'200-'

OR
~~AND~~ FROMDATE LIKE '------2010'

- List all the defenders

```
select *
FROM PLAYER
WHERE POSITION = 'Defender';
```

- Count the total number of games ~~sport~~ scores

→ SELECT COUNT (~~SCORE~~ *)
FROM GAME
GROUP BY SCORE;

ALL & ANY / SOME

ALL 40000 > [80000, 60000, 20000] → False ANY, True

SELECT ColumnName FROM TableName
WHERE ColumnName Relational operator ALL (set of values / ANY or sub-query);

Emp

Eid	Ename	Dno	Salary
1	A	D_1	80000
2	B	D_2	75000
3	C	D_1	40000
4	D	D_3	50000
5	E	D_5	60000
6	F	D_5	20000
7	G	D_3	15000

Ex

```
SELECT Ename . FROM Emp .
WHERE Salary > ALL (20000, 30000, 40000)
```

Output → A B D E

ANY ⟶ A B C D E

Find the Ename of all the emp whose salary is greater than all the emp. who are working ~~in~~ Dno=5

→

```
SELECT Ename
FROM Emp                                   60000, 20000
WHERE Salary > ALL ( Stelect Salary FROM Emp
                     WHERE Dno = 'D5' );
```

- Cinema (Theator, Address, Capacity);

Query

```
SELECT  Address
FROM    Cinema
```

Such that it always finds the addresses of theator with maximum capacity.

1 `WHERE Capacity >= (SELECT MAX(Capacity) FROM Cinema);`

2 `WHERE Capacity > (SELECT MAX(Capacity) FROM Cinema);`

3 `WHERE Capacity <= ( " " " " );`

✓ 4 `WHERE Capacity = (" " " " );`

- The number of rows returned by the below SQL query is,

```
SELECT RollNo, SUM(Marks) FROM Test
WHERE Marks > 20
GROUP BY RollNO ;
```

Test

RollNo	CourseCode	Marks	WHERE	Group
~~1 SE~~	SE	2~~0~~		
2	DBMS	25 ✓	]	48
1	OS	23 ✓		
2	DBMS	23 ✓	—	23
2	SE	18		
2	OS	13		
3	SE	10		
3	DBMS	22 ✓	—	22
3	OS	19.		

RNO	SUM(Marks)
1	48
2	23
3	22

∴ 3 rows

SELECT RollNO, SUM(Marks) FROM Test
WHERE Marks > ~~20~~ ALL (SELECT Marks FROM ~~Scalar~~ Test
WHERE Marks <= 20)

GROUP BY RollNO;

Consider the two tables Cat~~er~~ and prod the number or rows ~~ub~~ updated by below query is 3

```
UPDATE prod
SET price = (price + price * 0.1)
WHERE Cid IN (SELECT P.Cid From prod P, Cat C
              WHERE P.Cid = C.Cid AND Cname =
              'Biscuit');
```

→ Cat~~gory~~

Cid	Cname
1	Biscuit
2	Wafers
3	Food Grains

Prod.

Pid	Pname	Price	Cid
11	Oreo	30	1
22	Parle-G	40	1
33	Tiger	45	1
44	Rice	35	3

⟹

~~Pid~~ d	Pname	Price	Cid
~~Pid~~ 11	Oreo	33	1
11 22	Parle-G	44	1
~~22~~ 33	Tiger	49.5	1
~~33~~ 44	Rice	35	3

~~44~~

Allcad → Exists.

Emp

Eid	name	Address
1	A	Delhi
2	B	Pune
3	A	Chd
4	B	Delhi
5	C	Pune
6	D	Mumbai

Dept

Did	name	Eid
D1	HR	1
D2	TR	2
D3	MRKT	3
D4	Testing	4

Corealated Query.
Exists / Not Exist [Top down approach]

① Find all the employee details who work in a atleast one dept.

→
```
Select *
From Emp E
Where E.Eid EXISTS(Select * from Dept D.
                   Where E.Eid = D.Eid);
```

② Find the Emp id and name of the professor who is an advisor for atleast one male student.

Professor (Eid PK, Name)
Student (RollNo, Name, Gender, Eid FK)

→
```
Select P.Eid, P.name From Professor P
Where P.Eid Exists (Seled * from Student S
                    Where S.Eid = P.Eid AND Gender = male)
```

Emp (Eid, Fname, Add)
Project (Eid, Pid, Pname, location).

① Find the Details of Emp who is working on atleast one project

→ Select *
From Emp E
Where E.Eid EXISIS (Select * From Project P
Where E.Eid = P.Eid);

② Find the details of Emp who is working on atleast one project and whose Emp add is ~~equ~~ same as project location

→ Select * From Emp E
Where E.Eid EXISIS (Select * From Project P
Where E.Eid = P.Eid AND
E.Add = P.location);

⊙ Sailors (Sids, sname, Rating, Age)
Boats (Bid, Bname, Color);
Reserves (Sid, Bid, Day);

① Find the names and ages of all sailors

→ Select Sname, Age From Sailors;

(2) Find all sailors with rating > 7

→ Select * from Sailor
where Rating > 7;

(3) Find the names of sailors who have reserved boat number 3.

→ Select S.Sname
From Sailors S, Boats B, Reserves R
Where S.Sid = R.Sid AND B.Bid = ~~5~~ 3 R.Bid AND
B.Bid = 3;

(4) Find the sids of sailors who have reserved a red boat.

→ Select S.s~~name~~id
From Sailors S, Boat B, Reserves R
Where S.Sid = R.Sid AND B.Bid = R.Bid AND
B.Colour = "Red";

(5) Find the colors of boats reserved by JOHN

→ select B.Color
From Sailors S, Boats B, Reserves R
Where S.Sid = R.Sid AND B.Bid = R.Bid AND
S.Sname = 'JOHN';

⑥ Find the name of sailors who have reserved at least one boat.

→ Select S.Sname
From Sailors S, Boat B, Reserves S
Where 'S.Sid = R.Sid AND B.Bid = R.Bid';

Select S.Sname
From Sailors S
Where Sid Exists (select R.Sid From Reserves
Where S.Sid = R.Sid);

⑦ Find the names of sailors who have reserved ~~at~~ a red or a green boat.

→ Select S.Sname
From Sailor S, Boat B, Reserve R
Where S.Sid = R.Sid AND B.Bid = R.Bid AND
B.Colour = 'Red' OR B.Colour = 'Green';

8 Find the names of sailor who have reserved both red and a green

⑨ Find the name of sailor who have not reserved any boat.

→ Select S.Sname

From Sailor S
Where Sid NOT EXISTS (Select R.Sid ~~Fro~~
From Reserve R, Boat B
WHere R.Sid=S.Sid AND
B.Bid= R.Bid ;) ;

OR

Select S.Sname
from Sailor S
where S.Sid NOT IN (Select R.Sid From Reserve R
where R.Sid= S.Sid);

(10) Find the name of sailor who have reserved all the boats.

→ Select Sname
From Sailors S
where Sid IN (Select R.Sid
From Reserve R
where R.Bid = ALL (Select B.Bid
From Boat B
Wherere
B.Bid=R.Bid));

(11) Find the name of sailors who are older than oldest sailor rating of 10.
Select Sname

From Sailor S
Where Rating = 10 AND S.Age ~~not in~~ != (Select Max(age) from Sailor)

⊙ Branch (Branchid, Bname, HOD)
Student (USN, Name, Add, Bid, Sem)
Book (BookId, Bookname, Authorid, Publisher, Bid)
Author (Aid, Aname, Country, age)
Borrow (USN, BookId, B_Date)

① list the details of student who are studying in 4th sem CSE dept.

→ Select * From Student Where Sem=4 ;

② list the student who have not borrowed any book

→

③ list the details of student who borrowed the book which are all published by the same publisher

→ Select * From Student S, Book B, Borrow B1
Where B1.USN = S.USN AND B1.Bookid = B.Bookid
AND (Select publisher From Book)

Select * From Student
Where USN IN (select USN
From Borrow EXISTS (Select Publisher
from Book));

- Display the student details who borrowed books of more than one author

Select USN
From Student S, Book B, Author A, Borrow BR
Where S.USN = BR.USN AND BR.Bookid = B.Bookid
AND B.Authorid = A.Aid.

Group By USN
having Count(*) > 1