**Day 04**

**#to create database**

create database guvidtm13;

**#to use database**

use guvidtm13;

#leo => 3

**#to create table**

create table student(

name varchar(100),

age int,

city varchar(100),

salary float(8,2));

**#to select the table**

select \* from student;

**#to insert values**

insert into student values ("pooja",23,"chennai",35000.78);

select \* from student;

**#how to load excel file into mysql:**

Choose the database

=> right click the database

=> table data import wizard

=> browse the file

=> select the file from your pc

=> next

⇒ finish

**#SORTING/FILTERING**

#\* => all the column

select \* from car\_prices;

**#to rename the column**

alter table car\_prices rename column made to make;

select \* from car\_prices where make = "Chevrolet" and model = "Cruze";

select \* from car\_prices where make = "Chevrolet" or model = "Cruze";

**#condition => #Toyota + price <20000 or #Nissan + year > 2015**

select \* from car\_prices where (make = 'Toyota' and price < 20000) or

(make = "Nissan" and year > 2015);

#sorting

select \* from car\_prices order by price;

select \* from car\_prices order by price desc;

select \* from car\_prices order by make asc,price desc;

#and/or => condition operators

#between

**select \* from car\_prices where price between 20000 and 30000;**

**select \* from car\_prices where price >= 20000 and price <= 30000;**

**Practice question:**

<https://leetcode.com/problems/big-countries/?envType=study-plan-v2&envId=top-sql-50>

**select name,population,area from world where population >= 25000000 or area >= 3000000;**

**#to create table**

create table student1(

id int,

name varchar(25),

age int,

salary decimal(9,2),

city varchar(25));

insert into student1 values (101,"pooja",24,1000000.45,"chennai");

insert into student1 values (102,"mala",23,800000.34,"madurai"),

(103,"sheela",26,500000,"chennai");

**#Commands in SQL**

**#commands in sql**

**#DDL -> Data Definition Language**

**#we cant roll back the DDL commands\***

**#Create,alter,drop,truncate,rename**

#create -> table/database

**#alter is used for column modification**

#alter => **(add,modify,drop,rename)**

**#to add new column**

alter table student add column country varchar(20);

update student set country = "india";

**#to alter the datatype of the column**

alter table student modify column name varchar(50);

**#to delete the column**

alter table student drop column country;

**#to rename the column**

alter table student rename column name to stud\_name;

**#to delete values in the table**

truncate table student;

**#to delete the table completely from the database**

drop table student;

select \* from student;

describe student;

**#to change the table name**

**#rename**

rename table car\_prices to car;

**#DML -> Data Manipulation Language**

**#insert,delete,update**

**#we can roll back the DML commands\***

insert into student values(103,"raju",34,2300000,"mumbai");

**#to update/delete system req**

set sql\_safe\_updates = 0;

**#to delete the row with condition**

delete from student where city = 'mumbai';

**#to update the row with certain condition**

update student set city = "Madras" where city = "chennai";

**#DQL -> Data Querying Language**

#select

select \* from student;

**#to transfer records from one table to another table**

**INSERT INTO car\_prices1**

**SELECT \* FROM car\_prices;**

**select \* from car\_prices;**

create table car\_prices1(make varchar(20),model varchar(20)

,year int,mileage int,price int,color varchar(20),state varchar(20));

select \* from car\_prices1;

**#limit - Limiting the top rows**

**#offset - skipping the top rows**

select \* from car order by price desc limit 5;

select \* from car order by price desc limit 3 offset 2;

select \* from car order by price desc;

**#task 👍**

[**https://leetcode.com/studyplan/top-sql-50/**](https://leetcode.com/studyplan/top-sql-50/)

**Module name : select**

**Day-2**

**#IN/LIKE**

**#in Operator is used to select multiple values**

select \* from car where model in ("Cruze","Altima","Fusion");

**#like act as wild card in Mysql**

select \* from car where model like "c%";

select \* from car where model like "%e";

select \* from car where model like "%z\_";

select \* from car where model like "%india%";

use youtube;

select \* from uber where classification like "%fare%";

**#concat,left,right,round,ceil,floor**

select \*,concat(make,"----->",model)as new\_column,left(make,1),right(make,2),

concat(price/3," %"),round(price/3),floor(price/3),ceil(price/3)

from car\_prices;

**Task:**

[**https://www.hackerrank.com/challenges/weather-observation-station-12/problem?isFullScreen=true**](https://www.hackerrank.com/challenges/weather-observation-station-12/problem?isFullScreen=true)

**Answer1:**

select distinct city from station where left(city,1) not in ("a","e","i","o","u")

and right(city,1) not in ("a","e","i","o","u");

**Answer2:**

**select distinct(city) from station**

**where (city not like "a%"**

**and city not like "e%"**

**and city not like "i%"**

**and city not like "o%"**

**and city not like "u%")and( city not like "%a"**

**and city not like "%e"**

**and city not like "%i"**

**and city not like "%o"**

**and city not like "%u"**

**)**

**#Aggregated functions**

**#sum,avg,min,max,count,count(\*)**

select sum(price),min(price),max(price),count(price),count(\*) from car\_prices;

**#count/count(\*)**

select \* from student;

select count(\*),count(id) from student;

insert into student values (103,"binoth",35,"chennai",45000.00);

insert into student values (null,null,null,null,null);

**#group by and Having**

**#Distinct -> used to remove duplicates**

select distinct model,year from car\_prices;

select distinct \* from car\_prices;

**#grouping -> aggreagate function**

select make,year,avg(price),count(make) from car\_prices group by make,year;

select make,year,count(\*) from car\_prices group by make,year;

**#count greater than 26**

select make,year,count(\*) from car\_prices

where year = 2010

group by make,year having count(\*)>26 and make = 'Toyota';

select make,year,count(\*) from car\_prices

where year = 2010 and make = 'Toyota'

group by make,year having count(\*)>26;

**#where**

1.Where will be used before group by

2.Where will not support aggregate functions

3.Where can be used without group by

**#Having**

1.Having will be used after group by

2.Having will support aggregate function

3.Having cannot be used without group by

**Task :**

<https://leetcode.com/studyplan/top-sql-50/>

1.Basic Aggregate Functions

2.Sorting and Grouping

<https://www.hackerrank.com/domains/sql?filters%5Bskills%5D%5B%5D=SQL%20%28Basic%29&filters%5Bdifficulty%5D%5B%5D=easy&filters%5Bsubdomains%5D%5B%5D=select&filters%5Bsubdomains%5D%5B%5D=advanced-select&filters%5Bsubdomains%5D%5B%5D=aggregation&badge_type=sql>

**Filter**: basic => easy => **basic select/ advance select/ aggregation**

Task:

[**https://leetcode.com/problems/not-boring-movies/description/?envType=study-plan-v2&envId=top-sql-50**](https://leetcode.com/problems/not-boring-movies/description/?envType=study-plan-v2&envId=top-sql-50)

**select \* from cinema where id%2!=0 and description != "boring"**

**order by rating desc**

**Task:**

[**https://www.hackerrank.com/challenges/revising-aggregations-the-average-function/problem?isFullScreen=true**](https://www.hackerrank.com/challenges/revising-aggregations-the-average-function/problem?isFullScreen=true)

**select avg(population) from city where district = "California";**

**#case statement:**

#Case statement

**select distinct year from car\_prices;**

**#2014 < low model**

**#2016 > high model**

**#medium model**

select \*,"india" as country from car\_prices;

**select \*,**

**case**

**when year < 2014 then "low"**

**when year > 2016 then "high"**

**else "medium"**

**end as status from car\_prices;**

**#condition statement => grouping the new column => fetching the count**

**select**

**case**

**when year < 2014 then "low"**

**when year > 2016 then "high"**

**else "medium"**

**end as status,count(\*),avg(price) from car\_prices**

**group by 1;**

**Question Link :**

<https://www.hackerrank.com/challenges/what-type-of-triangle/problem?isFullScreen=true>

select

case

when a+b <= c or b+c <= a or c+a <= b then "Not A Triangle"

when a = b and b = c then "Equilateral"

when a=b or b = c or c=a then "Isosceles"

else "Scalene"

end

from triangles

**Day - 3**

**#substr/instr**

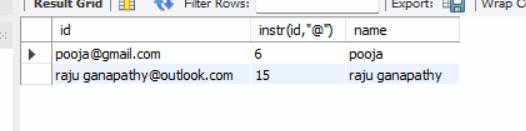
Substr = > it is used to slice the characters substr(column\_name,start\_index,no\_of\_char)

Inst => it is used to identify the index position of the given character

**use youtube;**

**select \*,substr(make,1,3),substr(make,-3) from car\_prices;**

**select \*,instr(make,"o") from car\_prices;**

****

**select id,instr(id,"@"),substr(id,1,instr(id,"@")-1) as name from email;**

**#joins => RDBMS**

**JOIN => joining two or more tables is calles joins**

**Types of Joins 👍**

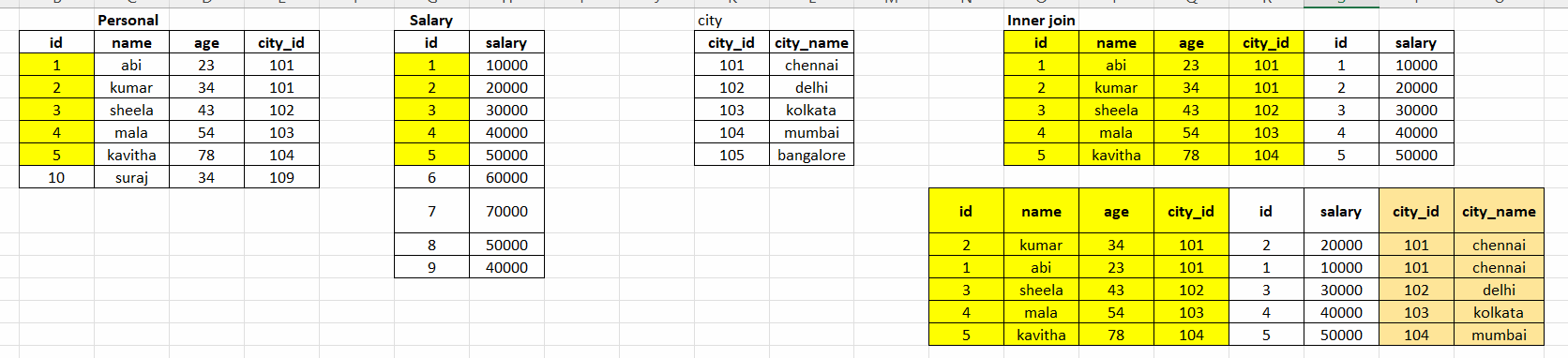
**#Joins**

#**inner join / join** => Exact match between 2 table

**select \* from personal as a inner join salary as b on a.id=b.id;**

**select \* from personal as a inner join salary as b on a.id=b.id**

**inner join city as c on a.city\_id = c.city\_id;**

#left **join/left outer**

#left join it will fix the left table and fetch the

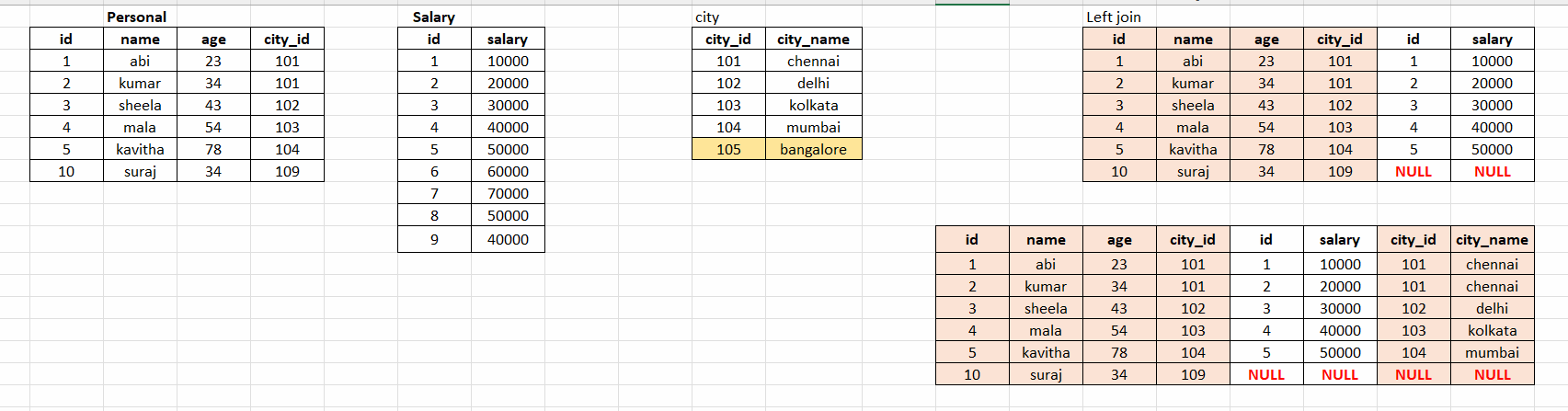
match from right table

#if no match it will return null as output

select \* from personal as a

Left join salary as b

on a.id = b.id;



**#right join/right outer**

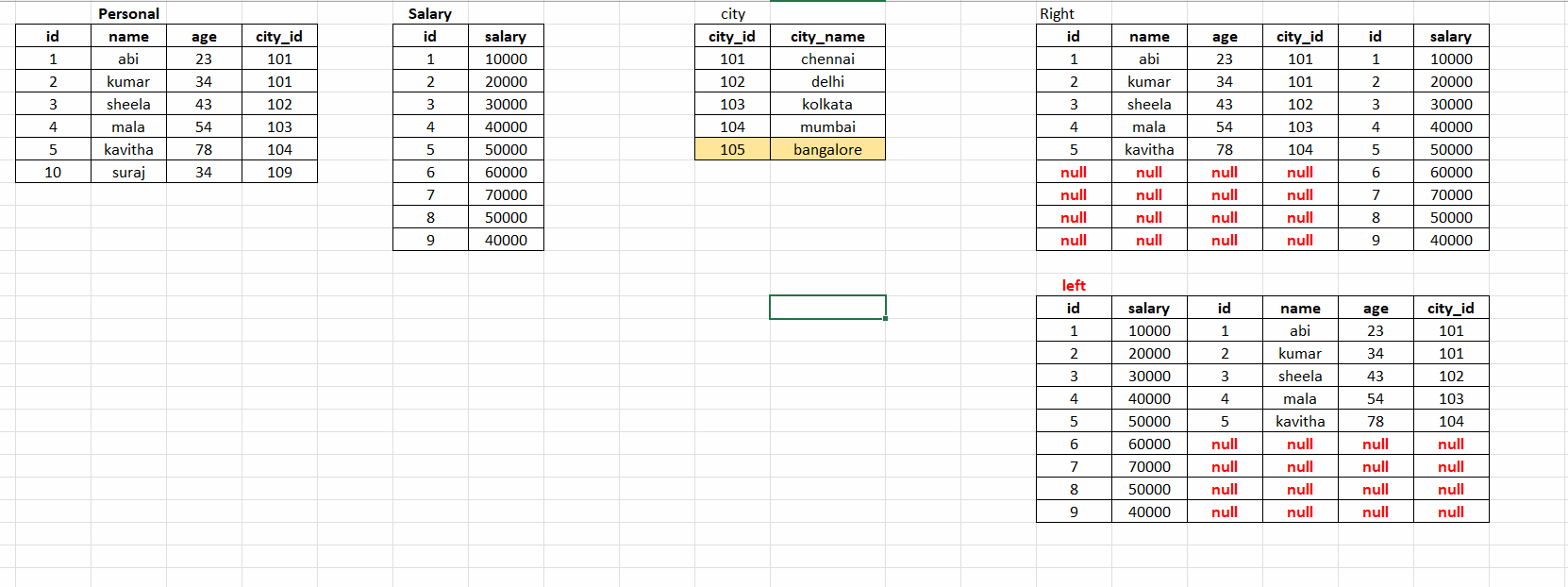
**#it will fix the right table and return the matches**

**#we can alternately use the left join instead of right join by reversing the table names**

**select \* from personal as a**

**right join salary as b**

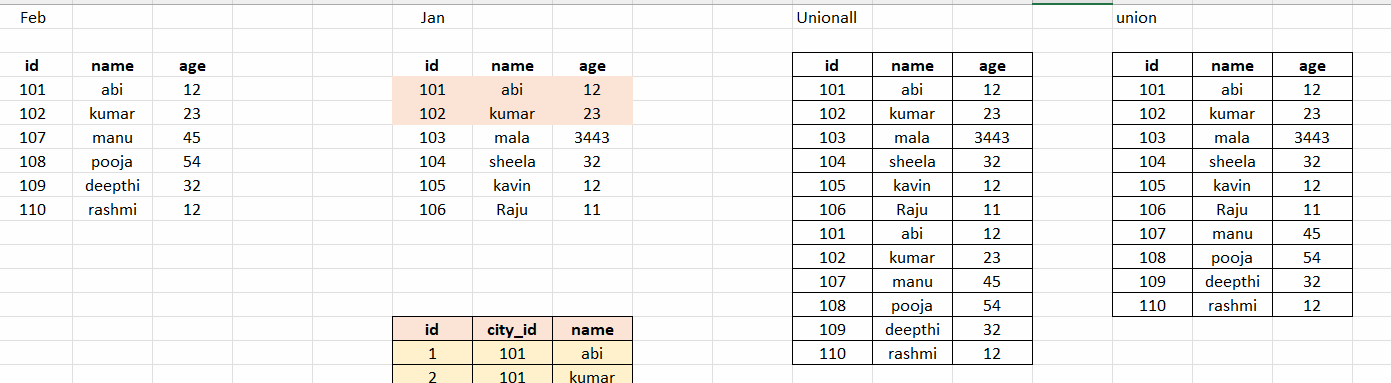
**on a.id = b.id;**



**#union/union all**

Union => it is used to append 2 tables (it will remove duplicates)

Union all =>it is used to append 2 tables (it will not remove duplicates)



**select \* from jan\_admission**

**union**

**select \* from feb\_admission;**

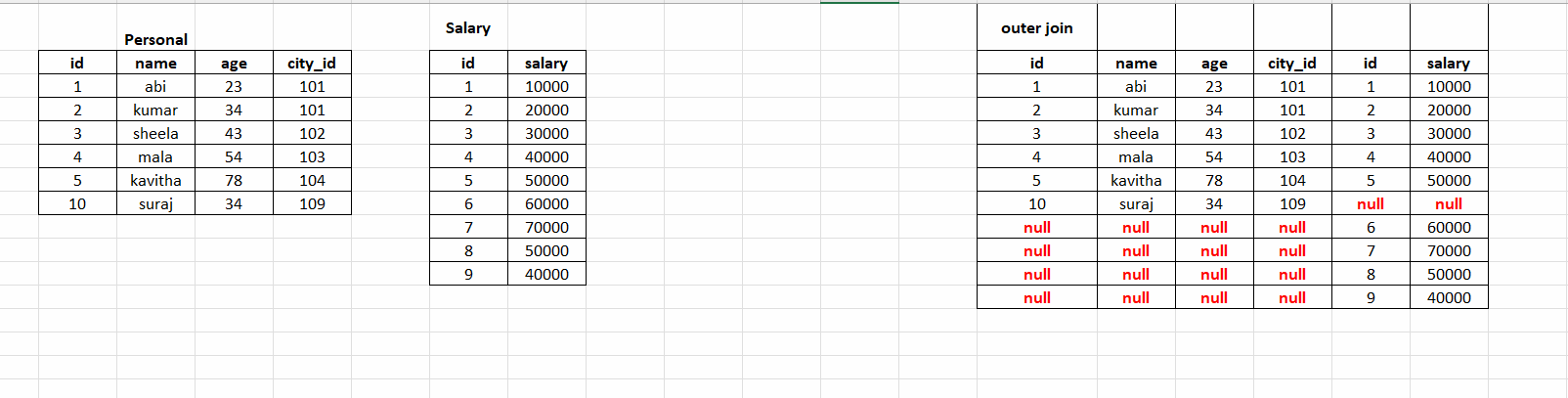
**select \* from city**

**union all**

**select \* from jan\_admission;**

**#outer join /full outer join**

**Union of left join and right join**

****

select \* from salary as a

left join personal as b

on a.id = b.id

union

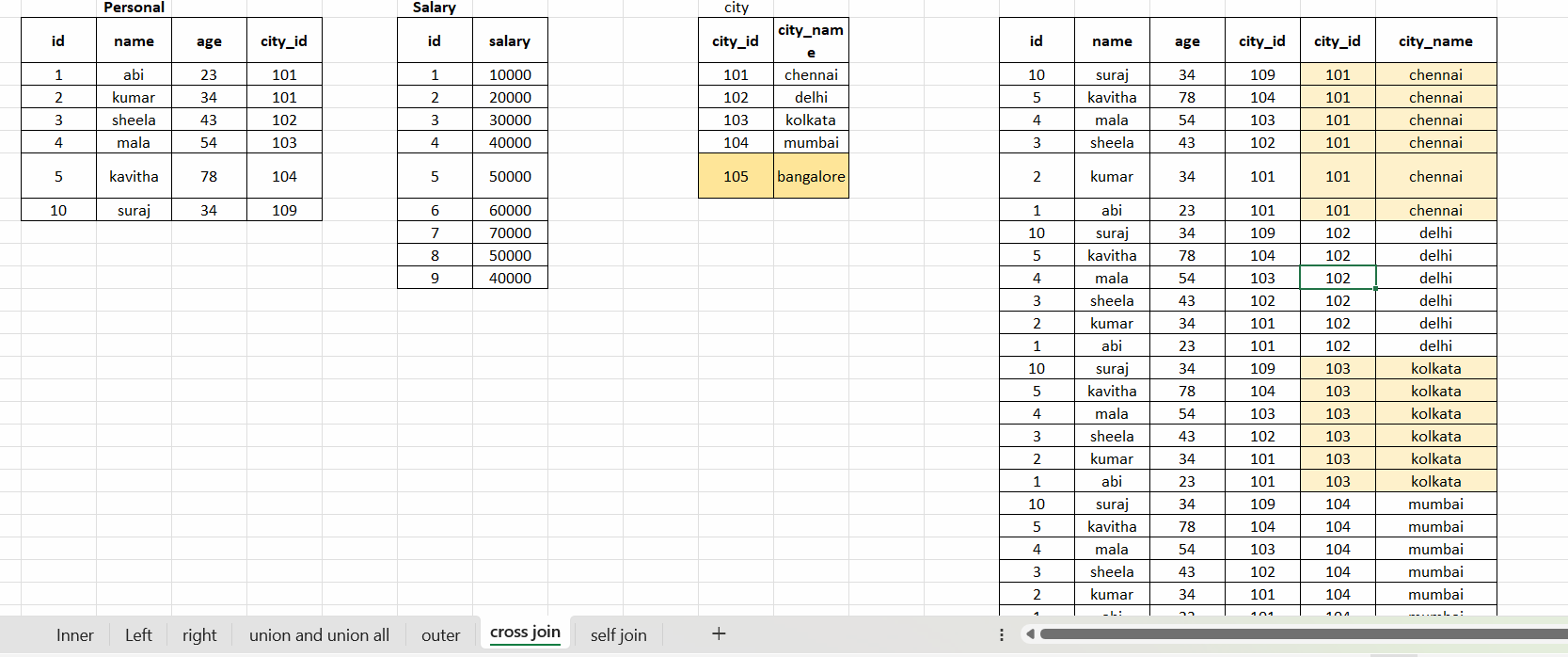
select \* from salary as a

right join personal as b

on a.id = b.id;

**#cross join => it gives cartesian product between tables**

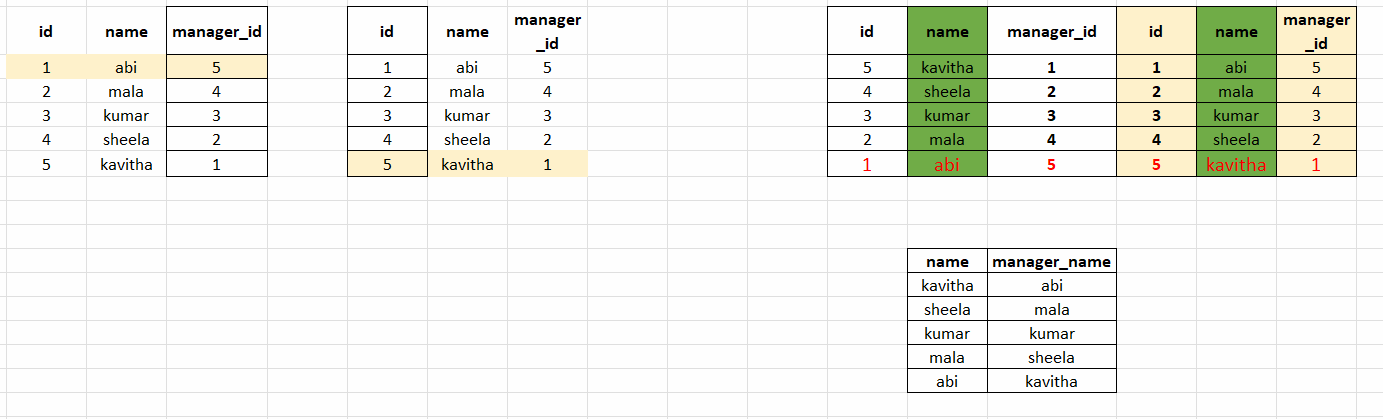
select \* from personal cross join city;



**#self join => joining its own table**

**select a.id,a.name,b.name as manager\_name from manager as a**

**inner join manager as b on a.manager\_id = b.id;**



**Group by link :** [**https://leetcode.com/problems/number-of-unique-subjects-taught-by-each-teacher/?envType=study-plan-v2&envId=top-sql-50**](https://leetcode.com/problems/number-of-unique-subjects-taught-by-each-teacher/?envType=study-plan-v2&envId=top-sql-50)

**select distinct(teacher\_id),count(distinct(subject\_id)) as cnt from teacher group by 1;**

**#practice (inner/left)**

**#inner join**

1. <https://leetcode.com/problems/product-sales-analysis-i/?envType=study-plan-v2&envId=top-sql-50>

**select product\_name,year,price from sales as a**

**inner join product as b**

**on**

**a.product\_id = b.product\_id;**

**#left join**

1. <https://leetcode.com/problems/customer-who-visited-but-did-not-make-any-transactions/?envType=study-plan-v2&envId=top-sql-50>

**select customer\_id,count(\*) as count\_no\_trans from visits as a**

**left join transactions as b on**

**a.visit\_id = b.visit\_id**

**where b.transaction\_id is null**

**group by 1;**

**Right:**

1. <https://leetcode.com/problems/replace-employee-id-with-the-unique-identifier/?envType=study-plan-v2&envId=top-sql-50>

**select unique\_id,name from EmployeeUNI as a right join Employees as b on a.id=b.id;**

**#using left join**

**select unique\_id,name from employees as a**

**left join employeeuni as b**

**on a.id = b.id;**

**Self join:**

1. <https://leetcode.com/problems/managers-with-at-least-5-direct-reports/description/?envType=study-plan-v2&envId=top-sql-50>

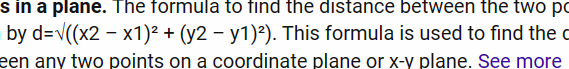
**select b.name from employee as a**

**inner join employee as b on a.managerId = b.id**

**group by b.name having count(\*) >= 5;**

**Cross join:**

**#find the shortest distance between 2 points in the table cross1**

****

**select \*,sqrt(pow(b.x-a.x,2) + pow(b.y - a.y,2)) as distance from cross1 as a cross join cross1 as b**

**where a.point\_name != b.point\_name order by distance asc;**

**Day - 4**

**#Subqueries**

**#query inside the query**

select \*,(select sum(price) from car\_prices) from car\_prices;

**#print the list of cars whose price is greater than average of price**

select \* from car\_prices where price > (select avg(price) from car\_prices);

select avg(price) from car\_prices;

**#having => subquery**

select make,model,avg(price) from car\_prices group by make,model

having avg(price) > (select avg(price) from car\_prices);

**#from #derived table #alias name**

select \* from (select make,model,price/3 from car\_prices) as a;

**#single row subquery:**

select \* from painting;

select \* from sales\_agent;

**#query the paiting information whose price > avg(price)**

select \* from painting where price > (select avg(price) from painting);

**#multiple row subquery**

#select painting information whose agency fees > 3000

**#join**

select a.\* from painting as a inner join sales\_agent as b on a.gallery\_id = b.gallery\_id

where b.agency\_fee > 3000;

**#subquery**

select \* from painting

where gallery\_id in

(

select gallery\_id from sales\_agent where agency\_fee > 3000

);

**#Problem**

<https://www.hackerrank.com/challenges/binary-search-tree-1/problem>

select N,

case

when p is null then "Root"

when n in (select distinct p from bst WHERE P IS NOT NULL) then "Inner"

else "Leaf"

end as status

from bst

order by N;

**#MULTIPLE COLUMN SUBQUERIES**

select \* from painting where (gallery\_id,price)

in

(

SELECT gallery\_id,max(price) from painting group by gallery\_id

);

**#correlated subquery:**

select city,count(b.id) from painting as a

inner join gallery as b on a.gallery\_id = b.id group by city;

**select \*,(select count(\*) from painting as p where g.id = p.gallery\_id)**

**from gallery as g;**

[**https://www.hackerrank.com/challenges/the-blunder/problem?isFullScreen=true**](https://www.hackerrank.com/challenges/the-blunder/problem?isFullScreen=true)

**select ceil(avg(salary)- avg(replace(salary,0,""))) from employees;**

**#windows function**

**#aggregate windows function(sum,min,max,count)**

select \*,avg(price) over() from car\_prices;

select \*,avg(price) over(rows between unbounded preceding and

unbounded following) from car\_prices;

select \*,avg(price) over(rows between 1 preceding and

1 following) from car\_prices;

select \*,avg(price) over(partition by year,model) from car\_prices;

select \*,count(\*) over(partition by model order by price desc) from car\_prices;

select \*,sum(price) over(partition by model order by price desc)

from car\_prices;

**#paypal interview question**

**select \*,sum(**

**case**

**when type = "deposited" then amount**

**else amount\*-1**

**end) over(partition by id order by date) as net\_transaction**

**from paypal;**

**#paypal problem**

**#ranking windows function(rank,dense\_rank,row\_number)**

**#rank => rank it will skip continuity**

**#dense\_rank => rank it will not skip anf give rank in order**

**#row\_number**

**select \*,rank() over(partition by make order by price desc),**

**dense\_rank() over(partition by make order by price desc),**

**row\_number() over(partition by make order by price desc)**

**from car\_prices;**

#car prices

**#give me 4th and 5th costliest car on each year**

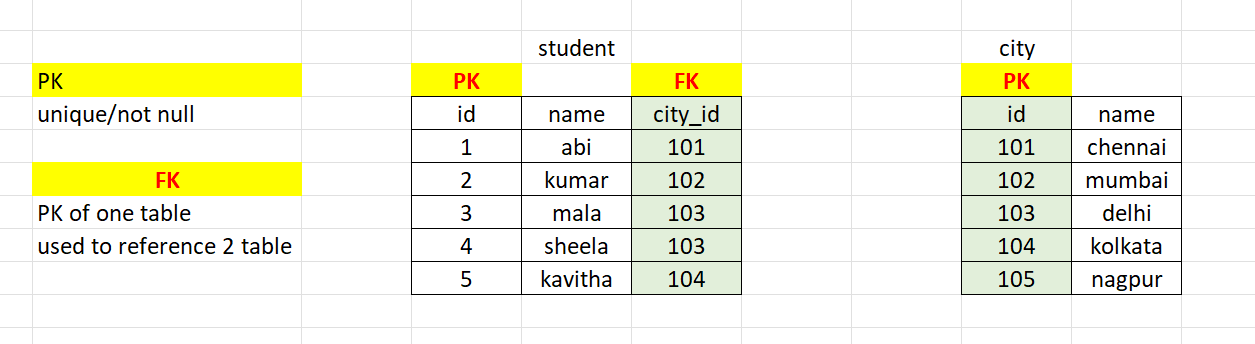
**select \* from**

**(**

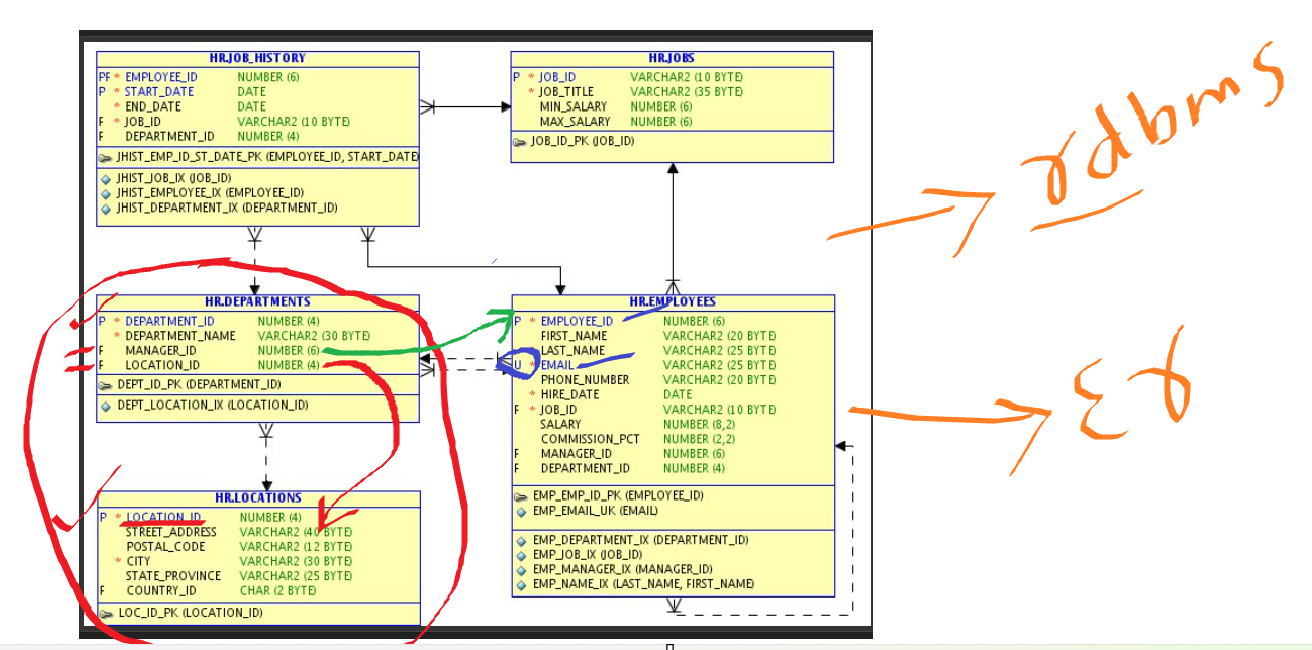
**select \*,dense\_rank() over(partition by year order by price desc) as rk from car\_prices) as a**

**where rk in (4,5);**

**Primary Key and Foreign Key :**

****

**HR Schema:**

****

**#python\_sql connect**

**#Date function**

select date,year(date),month(date),dayname(date),week(date),monthname(date),day(date),

current\_date(),datediff(current\_date(),date),

date\_add(date,interval 10 day),

date\_sub(date,interval 10 day),

date\_add(date,interval 10 month),

date\_add(date,interval 10 year),

date\_add(date,interval 10 week)

from uber;

**Last 15 days data:**

select \* from uber where datediff(current\_date(),date) <= 15;

select dayname(date),round(sum(amount),2) from uber group by 1 order by 2 desc;

select \* from uber where date between '2022-05-01' and '2022-06-31';

**#pivot table**

select year,

sum(case when model = "cruze" then price end) as Cruze,

sum(case when model = "Altima" then price end) as Altima,

sum(case when model = "Fusion" then price end) as Fusion,

sum(case when model = "Accord" then price end) as Accord,

sum(case when model = "Camry" then price end) as camry

from car\_prices where year = 2011 group by 1;