SQL-Assignment-2

create table salesman(

salesman\_id int primary key identity(1,1),

name varchar(40) not null,

city varchar(40) not null,

commission float,

);

insert into salesman(name,city,commission)

values('James Hoog','New York',0.15);

insert into salesman(name,city,commission)

values('Nail Knite','Paries',0.12);

insert into salesman(name,city,commission)

values('Pit Alex','London',0.11);

insert into salesman(name,city,commission)

values('Mc Lyon','Paris',0.25);

insert into salesman(name,city,commission)

values('Paul Adam','Rome',0.15);

insert into salesman(name,city,commission)

values('Lauson Hen','San Jose',0.17);

insert into salesman(name,city,commission)

values('Sam Karan','Tokiyo',0.15);

select \* from customer;

create table customer(

customer\_id int primary key identity(1,1),

cust\_name varchar(40) not null,

city varchar(40) not null,

grade int,

salesman\_id int foreign key references salesman(salesman\_id),

);

insert into customer(cust\_name,city,grade,salesman\_id)

values('Nick Rimando','New York',100,1);

insert into customer(cust\_name,city,grade,salesman\_id)

values('Brad Davis','New York',200,1);

insert into customer(cust\_name,city,grade,salesman\_id)

values('Graham Zusi','California',200,2);

insert into customer(cust\_name,city,grade,salesman\_id)

values('Julia Green','London',300,2);

insert into customer(cust\_name,city,grade,salesman\_id)

values('Fabian Johnson','Paris',200,4);

insert into customer(cust\_name,city,grade,salesman\_id)

values('Geoff Cameron','Berlin',100,6);

insert into customer(cust\_name,city,grade,salesman\_id)

values('Jozy Altidor','Moscow',200,5);

insert into customer(cust\_name,city,grade,salesman\_id)

values('Brad Guzan','London',300,3);

insert into customer(cust\_name,city,grade,salesman\_id)

values('Devid Warner','Perth',null,3);

create table orders(

ord\_no int primary key identity(1,1),

purch\_amt float not null,

ord\_date date not null,

customer\_id int foreign key references customer(customer\_id),

salesman\_id int foreign key references salesman(salesman\_id),

);

select \* from orders;

insert into orders(purch\_amt,ord\_date,customer\_id,salesman\_id)

values(150.5,'2023-10-05',2,1);

insert into orders(purch\_amt,ord\_date,customer\_id,salesman\_id)

values(270.65,'2023-09-06',1,1);

insert into orders(purch\_amt,ord\_date,customer\_id,salesman\_id)

values(65.26,'2023-10-25',5,4);

insert into orders(purch\_amt,ord\_date,customer\_id,salesman\_id)

values(110.5,'2023-09-18',8,3);

insert into orders(purch\_amt,ord\_date,customer\_id,salesman\_id)

values(2450.50,'2023-10-03',4,2);

insert into orders(purch\_amt,ord\_date,customer\_id,salesman\_id)

values(948.0,'2023-09-30',7,5);

insert into orders(purch\_amt,ord\_date,customer\_id,salesman\_id)

values(1983.43,'2023-09-20',3,2);

insert into orders(purch\_amt,ord\_date,customer\_id,salesman\_id)

values(250.45,'2023-09-22',1,1);

insert into orders(purch\_amt,ord\_date,customer\_id,salesman\_id)

values(1690.40,'2023-10-12',8,3);

insert into orders(purch\_amt,ord\_date,customer\_id,salesman\_id)

values(400.0,'2023-09-21',4,2);

insert into orders(purch\_amt,ord\_date,customer\_id,salesman\_id)

values(1999.90,'2023-10-10',7,5);

insert into orders(purch\_amt,ord\_date,customer\_id,salesman\_id)

values(1370.50,'2023-09-27',null,5);

insert into orders(purch\_amt,ord\_date,customer\_id,salesman\_id)

values(590.10,'2023-10-01',3,2);

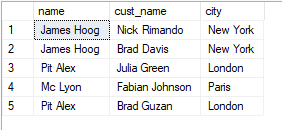
insert into orders(purch\_amt,ord\_date,customer\_id,salesman\_id)

values(1000.50,'2023-09-10',null,5);

--Queries

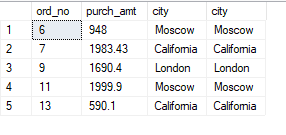
--1)write a SQL query to find the salesperson and customer who reside in the same city. Return Salesman, cust\_name and city

select s.name,c.cust\_name,c.city from salesman s inner join customer c on c.city=s.city;



--2)write a SQL query to find those orders where the order amount exists between 500 and 2000. Return ord\_no, purch\_amt, cust\_name, city

select o.ord\_no,o.purch\_amt,c.city,c.city from orders o inner join customer c on c.customer\_id=o.customer\_id where o.purch\_amt between 500 and 2000;



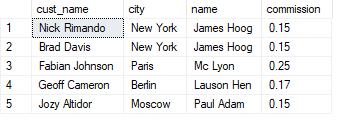
--3)write a SQL query to find the salesperson(s) and the customer(s) he represents. Return Customer Name, city, Salesman, commission

select c.cust\_name,c.city,s.name,s.commission from customer c inner join salesman s on s.salesman\_id=c.salesman\_id;



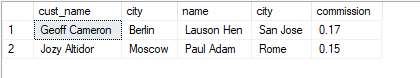
--4)write a SQL query to find salespeople who received commissions of more than 12 percent from the company. Return Customer Name, customer city, Salesman, commission.

select c.cust\_name,c.city,s.name,s.commission from customer c inner join salesman s on s.salesman\_id=c.salesman\_id where s.commission>0.12;



--5)write a SQL query to locate those salespeople who do not live in the same city where their customers live and have received a commission of more than 12% from the company. Return Customer Name, customer city, Salesman, salesman city, commission

select c.cust\_name,c.city,s.name,s.city,s.commission from customer c inner join salesman s on s.salesman\_id=c.salesman\_id where s.commission>0.12 and c.city != s.city;

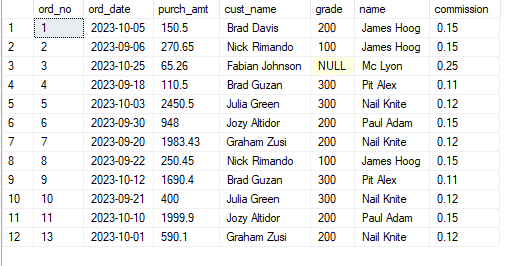


--6)write a SQL query to find the details of an order. Return ord\_no, ord\_date, purch\_amt, Customer Name, grade, Salesman, commission

Select o.ord\_no,o.ord\_date,o.purch\_amt,c.cust\_name,c.grade,s.name,s.commission

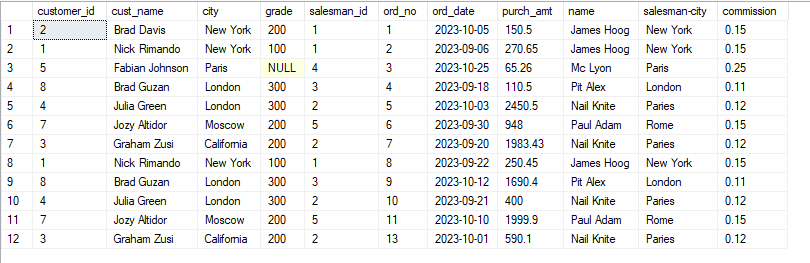
from orders o inner join salesman s on s.salesman\_id=o.salesman\_id

inner join customer c on c.customer\_id=o.customer\_id;



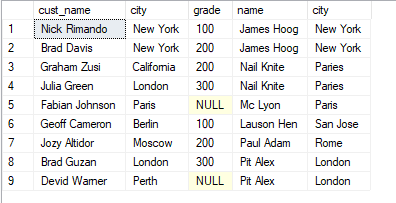
--7)Write a SQL statement to join the tables salesman, customer and orders so that the same column of each table appears once and only the relational rows are returned.

select c.\*,o.ord\_no,o.ord\_date,o.purch\_amt,s.name,s.city as "salesman-city",s.commission from customer c inner join orders o on c.customer\_id=o.customer\_id inner join salesman s on c.salesman\_id=s.salesman\_id;



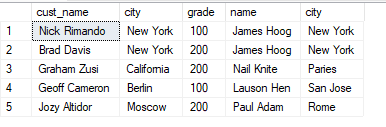
--8)write a SQL query to display the customer name, customer city, grade, salesman, salesman city. The results should be sorted by ascending customer\_id

select c.cust\_name,c.city,c.grade,s.name,s.city from customer c inner join salesman s on s.salesman\_id=c.salesman\_id order by customer\_id asc;



--9)write a SQL query to find those customers with a grade less than 300. Return cust\_name, customer city, grade, Salesman, salesmancity. The result should be ordered by ascending customer\_id.

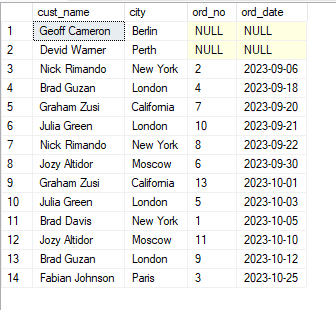
select c.cust\_name,c.city,c.grade,s.name,s.city from customer c inner join salesman s on s.salesman\_id=c.salesman\_id where c.grade<300 order by c.customer\_id asc;



--10)Write a SQL statement to make a report with customer name, city, order number, order date, and order amount in ascending order according to the order date to determine whether any of the existing customers have placed an order or not

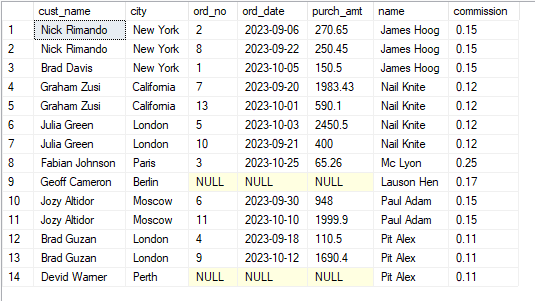
select c.cust\_name,c.city,o.ord\_no,o.ord\_date from customer c left join orders o

on c.customer\_id=o.customer\_id order by o.ord\_date asc;



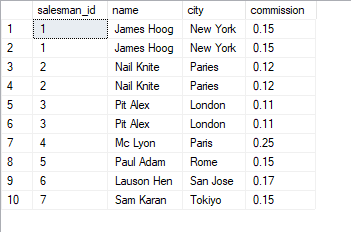
--11)Write a SQL statement to generate a report with customer name, city, order number, order date, order amount, salesperson name, and commission to determine if any of the existing customers have not placed orders or if they have placed orders through their salesman or by themselves

select c.cust\_name,c.city,o.ord\_no,o.ord\_date,o.purch\_amt,s.name,s.commission from customer c left join orders o on c.customer\_id=o.customer\_id left join salesman s on c.salesman\_id=s.salesman\_id;



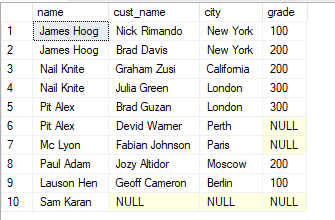
--12)Write a SQL statement to generate a list in ascending order of salespersons who work either for one or more customers or have not yet joined any of the customers

select s.\* from salesman s left join customer c on s.salesman\_id=c.salesman\_id order by s.salesman\_id;



--13)write a SQL query to list all salespersons along with customer name, city, grade, order number, date, and amount.

select s.name,c.cust\_name,c.city,c.grade from salesman s left join customer c on s.salesman\_id=c.salesman\_id;



--14)Write a SQL statement to make a list for the salesmen who either work for one or more customers or yet to join any of the customers. The customer may have placed, either one or more orders on or above order amount 2000 and must have a grade, or he may not have placed any order to the associated supplier.

select s.\* from customer c right join salesman s on c.salesman\_id=s.salesman\_id left join orders o on c.customer\_id=o.customer\_id where o.purch\_amt>2000 and c.grade is not null;



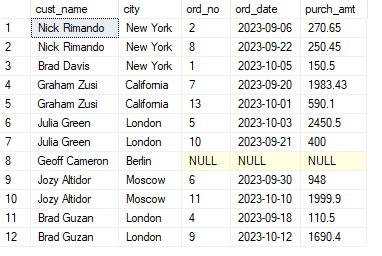
--15)Write a SQL statement to generate a list of all the salesmen who either work for one or more customers or have yet to join any of them. The customer may have placed one or more orders at or above order amount 2000, and must have a grade, or he may not have placed any orders to the associated supplier.

select s.\* from customer c right join salesman s on c.salesman\_id=s.salesman\_id left join orders o on c.customer\_id=o.customer\_id where o.purch\_amt>2000 and c.grade is not null;



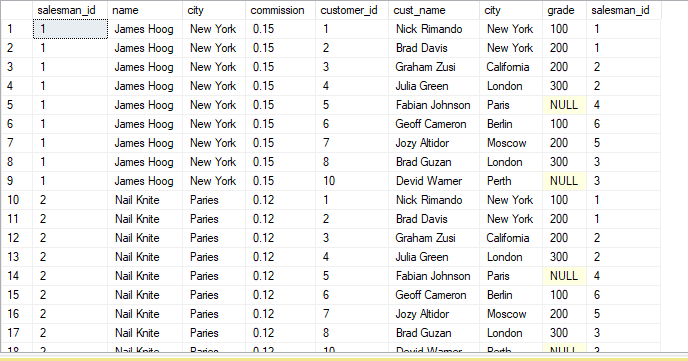
--16)Write a SQL statement to generate a report with the customer name, city, order no. order date, purchase amount for only those customers on the list who must have a grade and placed one or more orders or which order(s) have been placed by the customer who neither is on the list nor has a grade.

select c.cust\_name,c.city,o.ord\_no,o.ord\_date,o.purch\_amt from customer c full outer join orders o on c.customer\_id=o.customer\_id where c.grade is not null;



--17)Write a SQL query to combine each row of the salesman table with each row of the customer table

select \* from salesman s cross join customer c ;



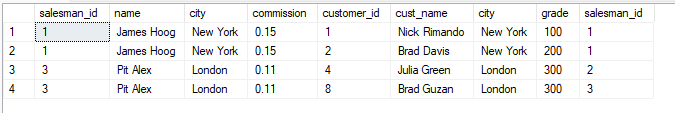
--18)Write a SQL statement to create a Cartesian product between salesperson and customer, i.e. each salesperson will appear for all customers and vice versa for that salesperson who belongs to that city

select \* from salesman s cross join customer c where s.city=c.city;



--19)Write a SQL statement to create a Cartesian product between salesperson and customer, i.e. each salesperson will appear for every customer and vice versa for those salesmen who belong to a city and customers who require a grade

select \* from salesman s cross join customer c where s.city=c.city and c.grade is not null;



--20)Write a SQL statement to make a Cartesian product between salesman and customer i.e. each salesman will appear for all customers and vice versa for those salesmen who must belong to a city which is not the same as his customer and the customers should have their own grade

select \* from salesman s cross join customer c where s.city!=c.city and c.grade is not null;

