

# Assignment 1

Q-1 Write a query to get a Product list (id, name, unit price) where current products cost less than \$20.

The screenshot shows a SQL query window with the following query: `select ProductID, ProductName, UnitPrice from Products where UnitPrice <= 20;`. The results pane displays a table with 32 rows of data. The status bar at the bottom indicates the query was executed successfully.

ProductID	ProductName	UnitPrice
1	Chai	18.00
2	Chang	19.00
3	Aniseed Syrup	10.00
4	Korbu	6.00
5	Genen Shouyu	15.50
6	Pavlova	17.45
7	Teatime Chocolate Biscuits	9.20
8	Sir Rodney's Scones	10.00
9	Tunnbröd	9.00
10	Guaraná Fantástica	4.50
11	NuNuCa Nuß-Nougat-Creme	14.00
12	Gorgonzola Telino	12.50
13	Gelbst	2.50
14	Sasquatch Ale	14.00
15	Steeleye Stout	18.00
16	Inlagd Sill	19.00
17	Chartreuse verte	18.00
18	Boston Crab Meat	18.40
19	Jack's New England Clam	9.65
20	Singaporean Hokkien Fried...	14.00
21	Gula Malacca	19.45
22	Rogede sild	9.50
23	Spegesild	12.00
24	Zaanse koeken	9.50
25	Chocolade	12.75
26	Maxilaku	20.00
27	Valkoinen suklaa	16.25
28	Filo Mix	7.00
29	Tourtière	7.45
30	Ravioli Angelo	19.50
31	Escargots de Bourgogne	13.25
32	Louisiana Hot Spiced Okra	17.00

Q-2 Write a query to get Product list (id, name, unit price) where products cost between \$15 and \$25

The screenshot shows a SQL query window with the following query: `select ProductID, ProductName, UnitPrice from Products where UnitPrice between 15 and 25;`. The results pane displays a table with 25 rows of data. The status bar at the bottom indicates the query was executed successfully.

ProductID	ProductName	UnitPrice
1	Chai	18.00
2	Chang	19.00
3	Chef Anton's Cajun Seasoning	22.00
4	Chef Anton's Gumbo Mix	21.35
5	Grandma's Boysenberry Spread	25.00
6	Queso Cabrales	21.00
7	Tofu	23.25
8	Genen Shouyu	15.50
9	Pavlova	17.45
10	Gustaf's Knäckebröd	21.00
11	Steeleye Stout	18.00
12	Inlagd Sill	19.00
13	Chartreuse verte	18.00
14	Boston Crab Meat	18.40
15	Gula Malacca	19.45
16	Maxilaku	20.00
17	Valkoinen suklaa	16.25
18	Pâté chinois	24.00
19	Ravioli Angelo	19.50
20	Louisiana Fiery Hot Pepper Sauce	21.05
21	Louisiana Hot Spiced Okra	17.00
22	Outback Lager	15.00
23	Flotemysost	21.50
24	Röd Kaviar	15.00
25	Lakkalikööri	18.00

Q-3 Write a query to get Product list (name, unit price) of above average price.

The screenshot shows a SQL Server Enterprise Manager window with a query executed successfully. The query is: `select ProductName, UnitPrice from Products where UnitPrice > (select avg(UnitPrice) from Products);`

The results are displayed in a table with 25 rows. The columns are ProductName and UnitPrice. The products listed are:

ProductID	ProductName	UnitPrice
1	Uncle Bob's Organic Dried Pears	30.00
2	Northwoods Cranberry Sauce	40.00
3	Mishi Kobe Niku	97.00
4	Ikura	31.00
5	Queso Manchego La Pastora	38.00
6	Alice Mutton	39.00
7	Camaronvion Tigers	62.50
8	Sir Rodney's Marmalade	81.00
9	Gumbär GummiBärchen	31.23
10	Schoggi Schokolade	43.90
11	Rössle Sauerkraut	45.60
12	Thüringer Rostbratwurst	123.79
13	Mascarpone Fabioli	32.00
14	Côte de Blaye	263.50
15	Ipoh Coffee	46.00
16	Manjimup Dried Apples	53.00
17	Perth Pasties	32.80
18	Gnocchi di nonna Alice	38.00
19	Raclette Courdavault	55.00
20	Camembert Pierrot	34.00
21	Tarte au sucre	49.30
22	Vegie-spread	43.90
23	Wimmers gute Semmelknödel	33.25
24	Gudbrandsdalsost	36.00
25	Mozzarella di Giovanni	34.80

The status bar at the bottom indicates: Query executed successfully. LAPTOP-H765FCE8 (16.0 RTM) | LAPTOP-H765FCE8\vipas ... Northwind 00:00:00 25 rows

Q-4 Write a query to get Product list (name, unit price) of ten most expensive products

The screenshot shows a SQL Server Enterprise Manager window with a query executed successfully. The query is: `select top(10) ProductName, UnitPrice from Products order by UnitPrice desc;`

The results are displayed in a table with 10 rows. The columns are ProductName and UnitPrice. The products listed are:

ProductID	ProductName	UnitPrice
1	Côte de Blaye	263.50
2	Thüringer Rostbratwurst	123.79
3	Mishi Kobe Niku	97.00
4	Sir Rodney's Marmalade	81.00
5	Camaronvion Tigers	62.50
6	Raclette Courdavault	55.00
7	Manjimup Dried Apples	53.00
8	Tarte au sucre	49.30
9	Ipoh Coffee	46.00
10	Rössle Sauerkraut	45.60

The status bar at the bottom indicates: Query executed successfully. LAPTOP-H765FCE8 (16.0 RTM) | LAPTOP-H765FCE8\vipas ... Northwind 00:00:00 10 rows

Q-5 Write a query to count current and discontinued products

The screenshot shows a SQL Server Enterprise Manager window with a query editor and a results pane. The query editor contains the following SQL code:

```
SELECT Count(ProductName)
FROM Products
GROUP BY Discontinued;
```

The results pane shows two rows of data:

	Count
1	69
2	8

The status bar at the bottom indicates "Query executed successfully." and "2 rows".

Q-6 Write a query to get Product list (name, units on order, units in stock) of stock is less than the quantity on order

The screenshot shows a SQL Server Enterprise Manager window with a query editor and a results pane. The query editor contains the following SQL code:

```
SELECT ProductName, UnitsOnOrder, UnitsInStock
FROM Products
WHERE ((UnitsInStock)<(UnitsOnOrder));
```

The results pane shows 14 rows of data:

	ProductName	UnitsOnOrder	UnitsInStock
1	Chang	40	17
2	Aniseed Syrup	70	13
3	Queso Cabrales	30	22
4	Sir Rodney's Scones	40	3
5	Gorgonzola Telino	70	0
6	Mascarpone Fabioli	40	9
7	Gravad lax	50	11
8	Rogede sild	70	5
9	Chocolade	70	15
10	Maxilaku	60	10
11	Wimmers gute Semmelknödel	80	22
12	Louisiana Hot Spiced Okra	100	4
13	Scottish Longbreads	10	6
14	Longlife Tofu	20	4

The status bar at the bottom indicates "Query executed successfully." and "14 rows".

## Assignment 2

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

The screenshot shows the SQL Developer interface with a query window titled 'SQLQuery8.sql - LA\_765FCE8(vipas (52))'. The query is:

```
select salesman.name as "Salesman",  
customer.cust_name, customer.city  
from salesman, customer  
where salesman.city=customer.city
```

The Results pane shows 6 rows of data:

Salesman	cust_name	city
Pit Alex	Brad Guzan	London
James Hoog	Nick Rimando	New York
Nail Knite	Fabian Johnson	Paris
Mc Lyon	Fabian Johnson	Paris
James Hoog	Brad Davis	New York
Pit Alex	Julian Green	London

The status bar at the bottom indicates 'Query executed successfully.' and '6 rows'.

Q-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

The screenshot shows the SQL Developer interface with a query window titled 'SQLQuery9.sql - LA\_765FCE8(vipas (53))'. The query is:

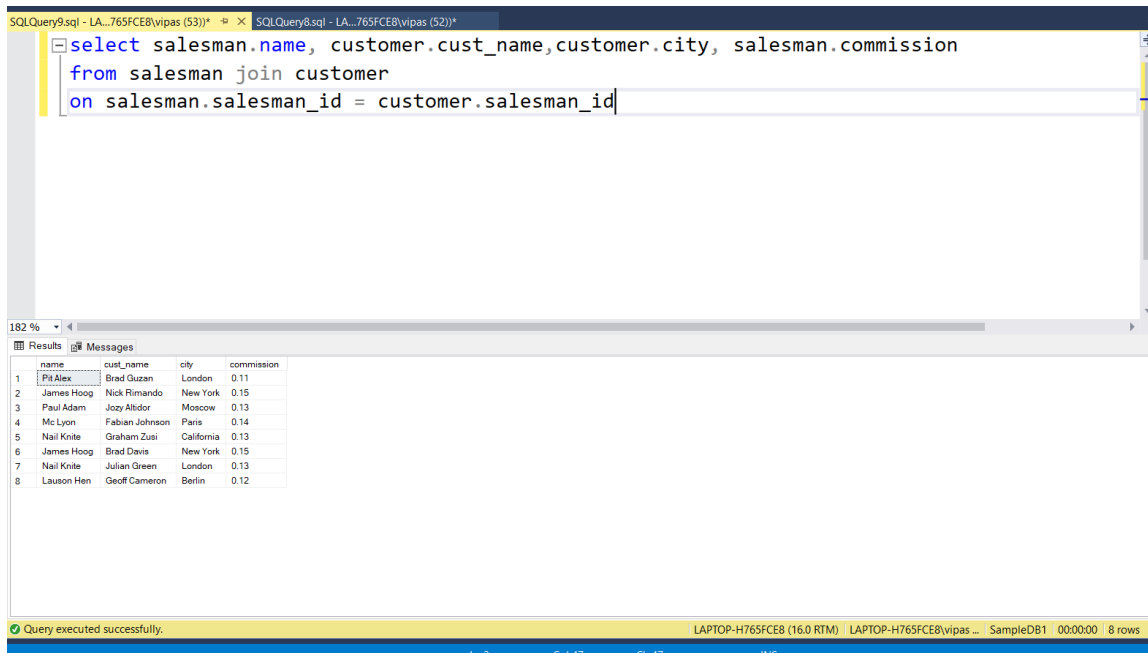
```
select orders.ord_no, orders.purch_amt, customer.cust_name, customer.city  
from orders, customer  
where orders.customer_id=customer.customer_id and  
orders.purch_amt between 500 and 2000
```

The Results pane shows 2 rows of data:

ord_no	purch_amt	cust_name	city
70007	948.5	Graham Zusi	California
70010	1983.43	Fabian Johnson	Paris

The status bar at the bottom indicates 'Query executed successfully.' and '2 rows'.

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

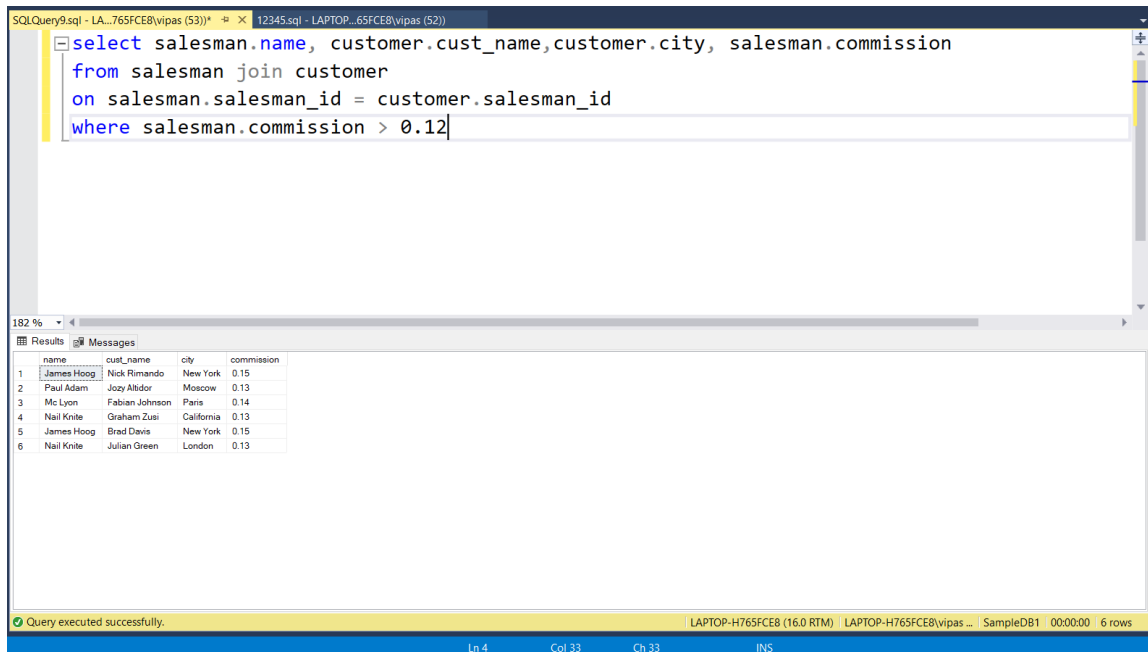


```
select salesman.name, customer.cust_name, customer.city, salesman.commission
from salesman join customer
on salesman.salesman_id = customer.salesman_id
```

	name	cust_name	city	commission
1	Pit Alex	Brad Guzan	London	0.11
2	James Hoog	Nick Rimando	New York	0.15
3	Paul Adam	Jozy Altidor	Moscow	0.13
4	Mc Lyon	Fabian Johnson	Paris	0.14
5	Nail Knite	Graham Zusi	California	0.13
6	James Hoog	Brad Davis	New York	0.15
7	Nail Knite	Julian Green	London	0.13
8	Lauson Hen	Geoff Cameron	Berlin	0.12

Query executed successfully. LAPTOP-H765FCE8 (16.0 RTM) LAPTOP-H765FCE8(vipas ... SampleDB1 00:00:00 8 rows

Q-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.

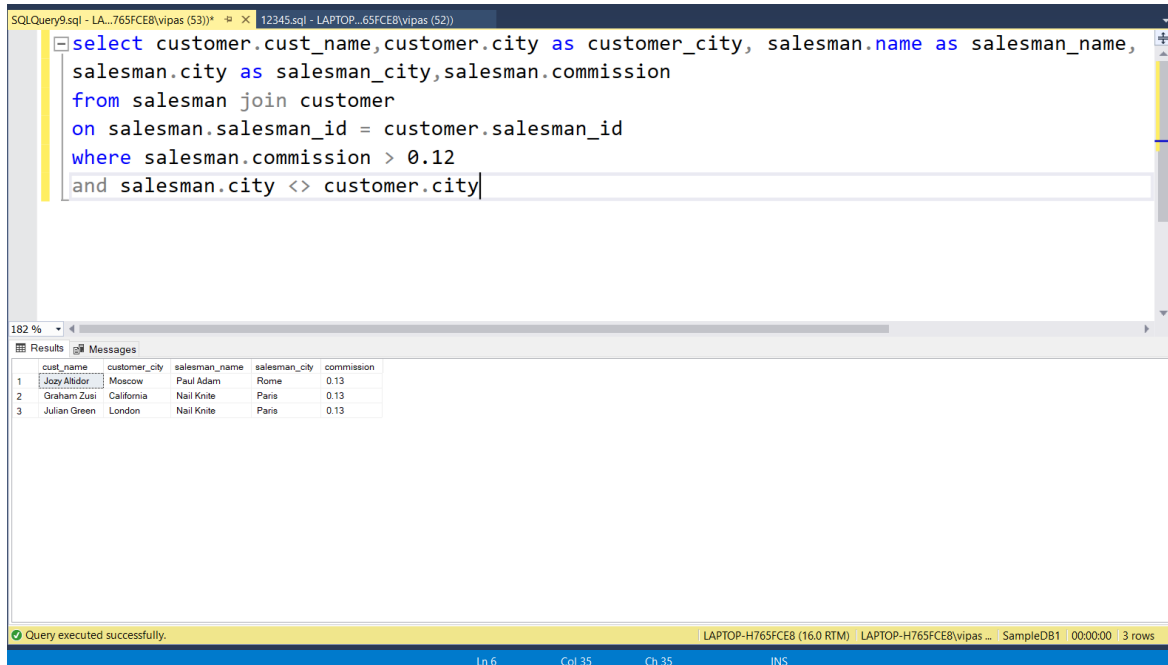


```
select salesman.name, customer.cust_name, customer.city, salesman.commission
from salesman join customer
on salesman.salesman_id = customer.salesman_id
where salesman.commission > 0.12
```

	name	cust_name	city	commission
1	James Hoog	Nick Rimando	New York	0.15
2	Paul Adam	Jozy Altidor	Moscow	0.13
3	Mc Lyon	Fabian Johnson	Paris	0.14
4	Nail Knite	Graham Zusi	California	0.13
5	James Hoog	Brad Davis	New York	0.15
6	Nail Knite	Julian Green	London	0.13

Query executed successfully. LAPTOP-H765FCE8 (16.0 RTM) LAPTOP-H765FCE8(vipas ... SampleDB1 00:00:00 6 rows

Q-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



```

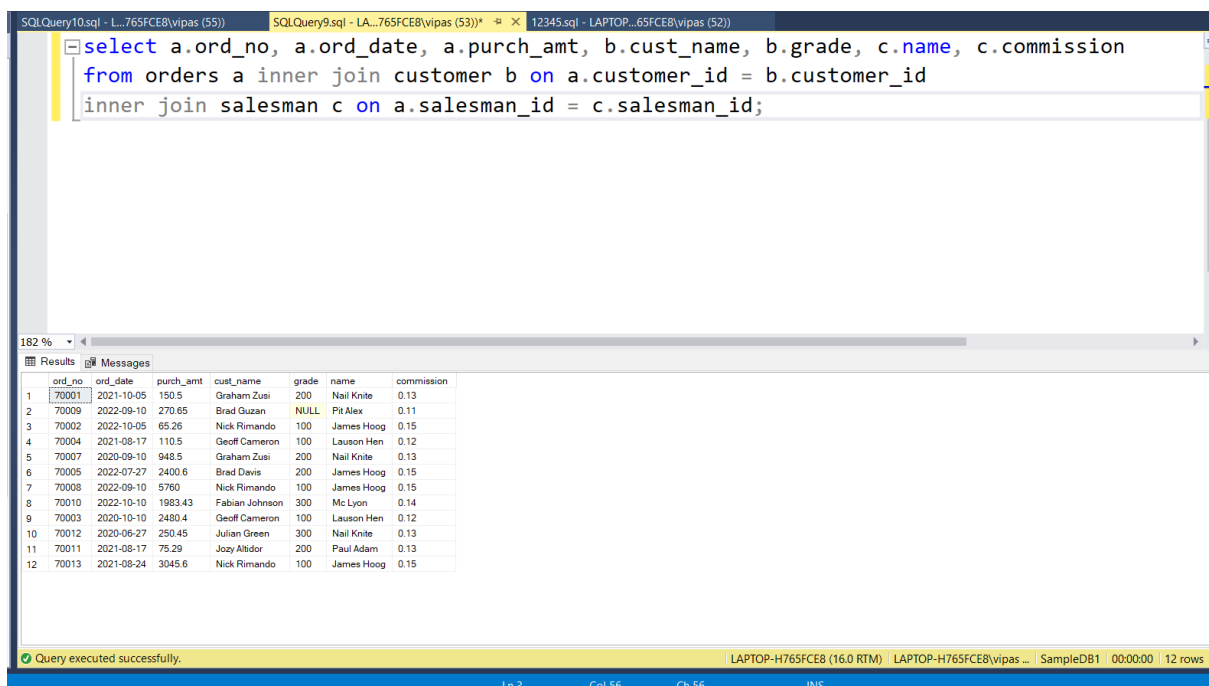
select customer.cust_name, customer.city as customer_city, salesman.name as salesman_name,
salesman.city as salesman_city, salesman.commission
from salesman join customer
on salesman.salesman_id = customer.salesman_id
where salesman.commission > 0.12
and salesman.city <> customer.city

```

	cust_name	customer_city	salesman_name	salesman_city	commission
1	Jozy Altidor	Moscow	Paul Adam	Rome	0.13
2	Graham Zusi	California	Nail Knite	Paris	0.13
3	Julian Green	London	Nail Knite	Paris	0.13

Query executed successfully.

Q-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 a.ord_no, a.ord_date, a.purch_amt, b.cust_name, b.grade, c.name, c.commission
from orders a inner join customer b on a.customer_id = b.customer_id
inner join salesman c on a.salesman_id = c.salesman_id;

```

	ord_no	ord_date	purch_amt	cust_name	grade	name	commission
1	70001	2021-10-05	150.5	Graham Zusi	200	Nail Knite	0.13
2	70009	2022-09-10	270.65	Brad Guzan	NULL	Pit Alex	0.11
3	70002	2022-10-05	65.26	Nick Rimando	100	James Hoog	0.15
4	70004	2021-08-17	110.5	Geoff Cameron	100	Lauson Hen	0.12
5	70007	2020-09-10	948.5	Graham Zusi	200	Nail Knite	0.13
6	70005	2022-07-27	2400.6	Brad Davis	200	James Hoog	0.15
7	70008	2022-09-10	5760	Nick Rimando	100	James Hoog	0.15
8	70010	2022-10-10	1983.43	Fabian Johnson	300	Mc Lyon	0.14
9	70003	2020-10-10	2480.4	Geoff Cameron	100	Lauson Hen	0.12
10	70012	2020-06-27	250.45	Julian Green	300	Nail Knite	0.13
11	70011	2021-08-17	75.29	Jozy Altidor	200	Paul Adam	0.13
12	70013	2021-08-24	3045.6	Nick Rimando	100	James Hoog	0.15

Query executed successfully.

Q-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 a.customer_id, a.cust_name, a.city, a.grade, b.salesman_id, b.name, b.commission, c.ord_no, c.purch_amt, c.ord_date
from customer a, salesman b, orders c
where a.customer_id = c.customer_id and c.salesman_id = b.salesman_id and a.city = b.city

```

customer_id	cust_name	city	grade	salesman_id	name	commission	ord_no	purch_amt	ord_date
3001	Brad Guzan	London	NULL	5005	Pit Alex	0.11	70009	270.65	2022-09-10
3002	Nick Rimando	New York	100	5001	James Hoog	0.15	70002	65.26	2022-10-05
3007	Brad Davis	New York	200	5001	James Hoog	0.15	70005	2400.6	2022-07-27
3002	Nick Rimando	New York	100	5001	James Hoog	0.15	70008	5760	2022-09-10
3004	Fabian Johnson	Paris	300	5006	Mc Lyon	0.14	70010	1983.43	2022-10-10
3002	Nick Rimando	New York	100	5001	James Hoog	0.15	70013	3045.6	2021-08-24

Query executed successfully. LAPTOP-H765FCE8 (16.0 RTM) LAPTOP-H765FCE8(vipas ... SampleDB1 00:00:00 6 rows

Q-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 a.cust_name, a.city as customer_city, a.grade, b.name, b.city as salesman_city
from customer a join salesman b
on a.salesman_id = b.salesman_id
order by a.customer_id

```

cust_name	customer_city	grade	name	salesman_city
Brad Guzan	London	NULL	Pit Alex	London
Nick Rimando	New York	100	James Hoog	New York
Jozy Altidor	Moscow	200	Paul Adam	Rome
Fabian Johnson	Paris	300	Mc Lyon	Paris
Graham Zusi	California	200	Nail Knite	Paris
Brad Davis	New York	200	James Hoog	New York
Julian Green	London	300	Nail Knite	Paris
Geoff Cameron	Berlin	100	Lauson Hen	San Jose

Query executed successfully. LAPTOP-H765FCE8 (16.0 RTM) LAPTOP-H765FCE8(vipas ... SampleDB1 00:00:00 8 rows

Q-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 a.cust_name,a.city,a.grade, b.name, b.city
FROM customer a JOIN salesman b
ON a.salesman_id=b.salesman_id
WHERE a.grade < 300
ORDER BY a.customer_id;

```

	cust_name	city	grade	name	city
1	Nick Rimando	New York	100	James Hoog	New York
2	Jozzy Altidor	Moscow	200	Paul Adam	Rome
3	Graham Zusi	California	200	Nail Knite	Paris
4	Brad Davis	New York	200	James Hoog	New York
5	Geoff Cameron	Berlin	100	Lauson Hen	San Jose

Query executed successfully. LAPTOP-H765FCE8 (16.0 RTM) LAPTOP-H765FCE8\vipas ... SampleDB1 00:00:00 5 rows

Q-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 a.cust_name,a.city, b.ord_no, b.ord_date,b.purch_amt AS "Order Amount"
FROM customer a JOIN orders b
ON a.customer_id = b.customer_id
order by b.ord_date;

```

	cust_name	city	ord_no	ord_date	Order Amount
1	Julian Green	London	70012	2020-06-27	250.45
2	Graham Zusi	California	70007	2020-09-10	948.5
3	Geoff Cameron	Berlin	70003	2020-10-10	2480.4
4	Geoff Cameron	Berlin	70004	2021-08-17	110.5
5	Jozzy Altidor	Moscow	70011	2021-08-17	75.29
6	Nick Rimando	New York	70013	2021-08-24	3045.6
7	Graham Zusi	California	70001	2021-10-05	150.5
8	Brad Davis	New York	70005	2022-07-27	2400.6
9	Nick Rimando	New York	70008	2022-09-10	5760
10	Brad Guzan	London	70009	2022-09-10	270.65
11	Nick Rimando	New York	70002	2022-10-05	65.26
12	Fabian Johnson	Paris	70010	2022-10-10	1983.43

Query executed successfully. LAPTOP-H765FCE8 (16.0 RTM) LAPTOP-H765FCE8\vipas ... SampleDB1 00:00:00 12 rows



Q-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 a.cust_name,a.city, b.ord_no, b.ord_date,b.purch_amt AS "Order Amount", c.name,c.commission
FROM customer a LEFT OUTER JOIN orders b ON a.customer_id = b.customer_id
LEFT OUTER JOIN salesman c ON c.salesman_id = b.salesman_id;

```

	cust_name	city	ord_no	ord_date	Order Amount	name	commission
1	Brad Guzan	London	70009	2022-09-10	270.65	Pit Alex	0.11
2	Nick Rimando	New York	70002	2022-10-05	65.26	James Hoog	0.15
3	Nick Rimando	New York	70008	2022-09-10	5760	James Hoog	0.15
4	Nick Rimando	New York	70013	2021-08-24	3045.6	James Hoog	0.15
5	Jozy Altidor	Moscow	70011	2021-08-17	75.29	Paul Adam	0.13
6	Fabian Johnson	Paris	70010	2022-10-10	1983.43	Mc Lyon	0.14
7	Graham Zusi	California	70001	2021-10-05	150.5	Nail Knite	0.13
8	Graham Zusi	California	70007	2020-09-10	948.5	Nail Knite	0.13
9	Brad Davis	New York	70005	2022-07-27	2400.6	James Hoog	0.15
10	Julian Green	London	70012	2020-06-27	250.45	Nail Knite	0.13
11	Geoff Cameron	Berlin	70004	2021-08-17	110.5	Lauson Hen	0.12
12	Geoff Cameron	Berlin	70003	2020-10-10	2480.4	Lauson Hen	0.12

Q-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 a.cust_name,a.city,a.grade, b.name as "Salesman", b.city
from customer a right join salesman b
on b.salesman_id = a.salesman_id
order by b.salesman_id;

```

	cust_name	city	grade	Salesman	city
1	Nick Rimando	New York	100	James Hoog	New York
2	Brad Davis	New York	200	James Hoog	New York
3	Graham Zusi	California	200	Nail Knite	Paris
4	Julian Green	London	300	Nail Knite	Paris
5	Geoff Cameron	Berlin	100	Lauson Hen	San Jose
6	Brad Guzan	London	NULL	Pit Alex	London
7	Fabian Johnson	Paris	300	Mc Lyon	Paris
8	Jozy Altidor	Moscow	200	Paul Adam	Rome

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

```
SQLQuery10.sql - L:\765FCEB\vipas (55)* 678910.sql - LAPTO...65FCEB\vipas (53) 12345.sql - LAPTOP...65FCEB\vipas (52)
select a.cust_name,a.city,a.grade,b.name as "Salesman", c.ord_no, c.ord_date, c.purch_amt
from customer a right join salesman b on b.salesman_id = a.salesman_id
right join orders c
on c.customer_id = a.customer_id;
```

165 % Results Messages

	cust_name	city	grade	Salesman	ord_no	ord_date	purch_amt
1	Graham Zusi	California	200	Nail Kite	70001	2021-10-05	150.5
2	Brad Guzan	London	NULL	Pit Alex	70009	2022-09-10	270.65
3	Nick Rimando	New York	100	James Hoog	70002	2022-10-05	65.26
4	Geoff Cameron	Berlin	100	Lauson Hen	70004	2021-08-17	110.5
5	Graham Zusi	California	200	Nail Kite	70007	2020-09-10	948.5
6	Brad Davis	New York	200	James Hoog	70005	2022-07-27	2400.6
7	Nick Rimando	New York	100	James Hoog	70008	2022-09-10	5760
8	Fabian Johnson	Paris	300	Mc Lyon	70010	2022-10-10	1983.43
9	Geoff Cameron	Berlin	100	Lauson Hen	70003	2020-10-10	2480.4
10	Julian Green	London	300	Nail Kite	70012	2020-06-27	250.45
11	Josy Altidor	Moscow	200	Paul Adam	70011	2021-08-17	75.29
12	Nick Rimando	New York	100	James Hoog	70013	2021-08-24	3045.6

Query executed successfully. LAPTOP-H765FCEB (16.0 RTM) LAPTOP-H765FCEB\vipas ... SampleDB1 00:00:00 12 rows

Ln 4 Col 34 Ch 34 INS

Q-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.

```
SQLQuery10.sql - L:\765FCEB\vipas (55)* 678910.sql - LAPTO...65FCEB\vipas (53) 12345.sql - LAPTOP...65FCEB\vipas (52)
select a.cust_name,a.city,a.grade, b.name AS "Salesman", c.ord_no, c.ord_date, c.purch_amt
from customer a right join salesman b on b.salesman_id = a.salesman_id
left join orders c
on c.customer_id = a.customer_id
where c.purch_amt >= 2000
and a.grade is not null;
```

165 % Results Messages

	cust_name	city	grade	Salesman	ord_no	ord_date	purch_amt
1	Brad Davis	New York	200	James Hoog	70005	2022-07-27	2400.6
2	Nick Rimando	New York	100	James Hoog	70008	2022-09-10	5760
3	Geoff Cameron	Berlin	100	Lauson Hen	70003	2020-10-10	2480.4
4	Nick Rimando	New York	100	James Hoog	70013	2021-08-24	3045.6

Query executed successfully. LAPTOP-H765FCEB (16.0 RTM) LAPTOP-H765FCEB\vipas ... SampleDB1 00:00:00 4 rows

Ln 6 Col 25 Ch 25 INS

Q-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 a.cust_name, a.city, b.ord_no, b.ord_date, b.purch_amt
from customer a left join orders b
on a.customer_id = b.customer_id;

```

cust_name	city	ord_no	ord_date	purch_amt
Brad Guzan	London	70009	2022-09-10	270.65
Nick Rimando	New York	70002	2022-10-05	65.26
Nick Rimando	New York	70008	2022-09-10	5760
Nick Rimando	New York	70013	2021-08-24	3045.6
Jozey Altidor	Moscow	70011	2021-08-17	75.29
Fabian Johnson	Paris	70010	2022-10-10	1983.43
Graham Zusi	California	70001	2021-10-05	150.5
Graham Zusi	California	70007	2020-09-10	948.5
Brad Davis	New York	70005	2022-07-27	2400.6
Julian Green	London	70012	2020-06-27	250.45
Geoff Cameron	Berlin	70004	2021-08-17	110.5
Geoff Cameron	Berlin	70003	2020-10-10	2480.4

Q-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 a.cust_name, a.city, b.ord_no, b.ord_date, b.purch_amt as "Order Amount"
from customer a full outer join orders b
on a.customer_id = b.customer_id
where a.grade is not null;

```

cust_name	city	ord_no	ord_date	Order Amount
Nick Rimando	New York	70002	2022-10-05	65.26
Nick Rimando	New York	70008	2022-09-10	5760
Nick Rimando	New York	70013	2021-08-24	3045.6
Jozey Altidor	Moscow	70011	2021-08-17	75.29
Fabian Johnson	Paris	70010	2022-10-10	1983.43
Graham Zusi	California	70001	2021-10-05	150.5
Graham Zusi	California	70007	2020-09-10	948.5
Brad Davis	New York	70005	2022-07-27	2400.6
Julian Green	London	70012	2020-06-27	250.45
Geoff Cameron	Berlin	70004	2021-08-17	110.5
Geoff Cameron	Berlin	70003	2020-10-10	2480.4

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

SQLQuery10.sql - L\_765FCEB(vipas (55))\* X 678910.sql - LAPTD\_65FCEB(vipas (53))\* 12345.sql - LAPTOP\_65FCEB(vipas (52))

```
select * from salesman cross join customer;
```

165 %

Results Messages

	salesman_id	name	city	commission	customer_id	cust_name	city	grade	salesman_id
1	5001	James Hoog	New York	0.15	3001	Brad Guzan	London	NULL	5005
2	5001	James Hoog	New York	0.15	3002	Nick Rimando	New York	100	5001
3	5001	James Hoog	New York	0.15	3003	Josy Altidor	Moscow	200	5007
4	5001	James Hoog	New York	0.15	3004	Fabian Johnson	Paris	300	5006
5	5001	James Hoog	New York	0.15	3005	Graham Zusi	California	200	5002
6	5001	James Hoog	New York	0.15	3007	Brad Davis	New York	200	5001
7	5001	James Hoog	New York	0.15	3008	Julian Green	London	300	5002
8	5001	James Hoog	New York	0.15	3009	Geoff Cameron	Berlin	100	5003
9	5002	Nail Knite	Paris	0.13	3001	Brad Guzan	London	NULL	5005
10	5002	Nail Knite	Paris	0.13	3002	Nick Rimando	New York	100	5001
11	5002	Nail Knite	Paris	0.13	3003	Josy Altidor	Moscow	200	5007
12	5002	Nail Knite	Paris	0.13	3004	Fabian Johnson	Paris	300	5006
13	5002	Nail Knite	Paris	0.13	3005	Graham Zusi	California	200	5002
14	5002	Nail Knite	Paris	0.13	3007	Brad Davis	New York	200	5001
15	5002	Nail Knite	Paris	0.13	3008	Julian Green	London	300	5002
16	5002	Nail Knite	Paris	0.13	3009	Geoff Cameron	Berlin	100	5003
17	5003	Lauson Hen	San Jose	0.12	3001	Brad Guzan	London	NULL	5005
18	5003	Lauson Hen	San Jose	0.12	3002	Nick Rimando	New York	100	5001
19	5003	Lauson Hen	San Jose	0.12	3003	Josy Altidor	Moscow	200	5007
20	5003	Lauson Hen	San Jose	0.12	3004	Fabian Johnson	Paris	300	5006
21	5003	Lauson Hen	San Jose	0.12	3005	Graham Zusi	California	200	5002

Query executed successfully. LAPTOP-H765FCEB (16.0 RTM) LAPTOP-H765FCEB(vipas ... SampleDB1 00:00:00 48 rows

Ln 1 Col 44 Ch 44 INS

Q-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.

SQLQuery10.sql - L\_765FCEB(vipas (55))\* X 678910.sql - LAPTD\_65FCEB(vipas (53))\* 12345.sql - LAPTOP\_65FCEB(vipas (52))

```
select * from salesman cross join customer where salesman.city is not null;
```

165 %

Results Messages

	salesman_id	name	city	commission	customer_id	cust_name	city	grade	salesman_id
1	5001	James Hoog	New York	0.15	3001	Brad Guzan	London	NULL	5005
2	5001	James Hoog	New York	0.15	3002	Nick Rimando	New York	100	5001
3	5001	James Hoog	New York	0.15	3003	Josy Altidor	Moscow	200	5007
4	5001	James Hoog	New York	0.15	3004	Fabian Johnson	Paris	300	5006
5	5001	James Hoog	New York	0.15	3005	Graham Zusi	California	200	5002
6	5001	James Hoog	New York	0.15	3007	Brad Davis	New York	200	5001
7	5001	James Hoog	New York	0.15	3008	Julian Green	London	300	5002
8	5001	James Hoog	New York	0.15	3009	Geoff Cameron	Berlin	100	5003
9	5002	Nail Knite	Paris	0.13	3001	Brad Guzan	London	NULL	5005
10	5002	Nail Knite	Paris	0.13	3002	Nick Rimando	New York	100	5001
11	5002	Nail Knite	Paris	0.13	3003	Josy Altidor	Moscow	200	5007
12	5002	Nail Knite	Paris	0.13	3004	Fabian Johnson	Paris	300	5006
13	5002	Nail Knite	Paris	0.13	3005	Graham Zusi	California	200	5002
14	5002	Nail Knite	Paris	0.13	3007	Brad Davis	New York	200	5001
15	5002	Nail Knite	Paris	0.13	3008	Julian Green	London	300	5002
16	5002	Nail Knite	Paris	0.13	3009	Geoff Cameron	Berlin	100	5003
17	5003	Lauson Hen	San Jose	0.12	3001	Brad Guzan	London	NULL	5005
18	5003	Lauson Hen	San Jose	0.12	3002	Nick Rimando	New York	100	5001
19	5003	Lauson Hen	San Jose	0.12	3003	Josy Altidor	Moscow	200	5007
20	5003	Lauson Hen	San Jose	0.12	3004	Fabian Johnson	Paris	300	5006
21	5003	Lauson Hen	San Jose	0.12	3005	Graham Zusi	California	200	5002

Query executed successfully. LAPTOP-H765FCEB (16.0 RTM) LAPTOP-H765FCEB(vipas ... SampleDB1 00:00:00 48 rows

Ln 1 Col 76 Ch 76 INS

Q-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.

SQLQuery10.sql - L..765FCE8(vipas (55))\* X 678910.sql - LAPTO...65FCE8(vipas (53))\* 12345.sql - LAPTOP...65FCE8(vipas (52))

```

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

```

165 % Results Messages

	salesman_id	name	city	commission	customer_id	cust_name	city	grade	salesman_id
1	5001	James Hoog	New York	0.15	3002	Nick Rimando	New York	100	5001
2	5001	James Hoog	New York	0.15	3003	Jozy Altidor	Moscow	200	5007
3	5001	James Hoog	New York	0.15	3004	Fabian Johnson	Paris	300	5006
4	5001	James Hoog	New York	0.15	3005	Graham Zusi	California	200	5002
5	5001	James Hoog	New York	0.15	3007	Brad Davis	New York	200	5001
6	5001	James Hoog	New York	0.15	3008	Julian Green	London	300	5002
7	5001	James Hoog	New York	0.15	3009	Geoff Cameron	Berlin	100	5003
8	5002	Nail Kite	Paris	0.13	3002	Nick Rimando	New York	100	5001
9	5002	Nail Kite	Paris	0.13	3003	Jozy Altidor	Moscow	200	5007
10	5002	Nail Kite	Paris	0.13	3004	Fabian Johnson	Paris	300	5006
11	5002	Nail Kite	Paris	0.13	3005	Graham Zusi	California	200	5002
12	5002	Nail Kite	Paris	0.13	3007	Brad Davis	New York	200	5001
13	5002	Nail Kite	Paris	0.13	3008	Julian Green	London	300	5002
14	5002	Nail Kite	Paris	0.13	3009	Geoff Cameron	Berlin	100	5003
15	5003	Lauson Hen	San Jose	0.12	3002	Nick Rimando	New York	100	5001
16	5003	Lauson Hen	San Jose	0.12	3003	Jozy Altidor	Moscow	200	5007
17	5003	Lauson Hen	San Jose	0.12	3004	Fabian Johnson	Paris	300	5006
18	5003	Lauson Hen	San Jose	0.12	3005	Graham Zusi	California	200	5002
19	5003	Lauson Hen	San Jose	0.12	3007	Brad Davis	New York	200	5001
20	5003	Lauson Hen	San Jose	0.12	3008	Julian Green	London	300	5002
21	5003	Lauson Hen	San Jose	0.12	3009	Geoff Cameron	Berlin	100	5003

Query executed successfully. LAPTOP-H765FCE8 (16.0 RTM) LAPTOP-H765FCE8(vipas ... SampleDB1 00:00:00 42 rows

Ln 2 Col 64 Ch 64 INS

Q-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.

SQLQuery10.sql - L..765FCE8(vipas (55))\* X 678910.sql - LAPTO...65FCE8(vipas (53))\* 12345.sql - LAPTOP...65FCE8(vipas (52))

```

select * from salesman cross join customer
where salesman.city is not null and customer.grade is not null
and salesman.city <> customer.city;

```

165 % Results Messages

	salesman_id	name	city	commission	customer_id	cust_name	city	grade	salesman_id
1	5001	James Hoog	New York	0.15	3003	Jozy Altidor	Moscow	200	5007
2	5001	James Hoog	New York	0.15	3004	Fabian Johnson	Paris	300	5006
3	5001	James Hoog	New York	0.15	3005	Graham Zusi	California	200	5002
4	5001	James Hoog	New York	0.15	3008	Julian Green	London	300	5002
5	5001	James Hoog	New York	0.15	3009	Geoff Cameron	Berlin	100	5003
6	5002	Nail Kite	Paris	0.13	3002	Nick Rimando	New York	100	5001
7	5002	Nail Kite	Paris	0.13	3003	Jozy Altidor	Moscow	200	5007
8	5002	Nail Kite	Paris	0.13	3005	Graham Zusi	California	200	5002
9	5002	Nail Kite	Paris	0.13	3007	Brad Davis	New York	200	5001
10	5002	Nail Kite	Paris	0.13	3008	Julian Green	London	300	5002
11	5002	Nail Kite	Paris	0.13	3009	Geoff Cameron	Berlin	100	5003
12	5003	Lauson Hen	San Jose	0.12	3002	Nick Rimando	New York	100	5001
13	5003	Lauson Hen	San Jose	0.12	3003	Jozy Altidor	Moscow	200	5007
14	5003	Lauson Hen	San Jose	0.12	3004	Fabian Johnson	Paris	300	5006
15	5003	Lauson Hen	San Jose	0.12	3005	Graham Zusi	California	200	5002
16	5003	Lauson Hen	San Jose	0.12	3007	Brad Davis	New York	200	5001
17	5003	Lauson Hen	San Jose	0.12	3008	Julian Green	London	300	5002
18	5003	Lauson Hen	San Jose	0.12	3009	Geoff Cameron	Berlin	100	5003
19	5005	Pit Alex	London	0.11	3002	Nick Rimando	New York	100	5001
20	5005	Pit Alex	London	0.11	3003	Jozy Altidor	Moscow	200	5007
21	5005	Pit Alex	London	0.11	3004	Fabian Johnson	Paris	300	5006

Query executed successfully. LAPTOP-H765FCE8 (16.0 RTM) LAPTOP-H765FCE8(vipas ... SampleDB1 00:00:00 37 rows

Ln 3 Col 36 Ch 36 INS