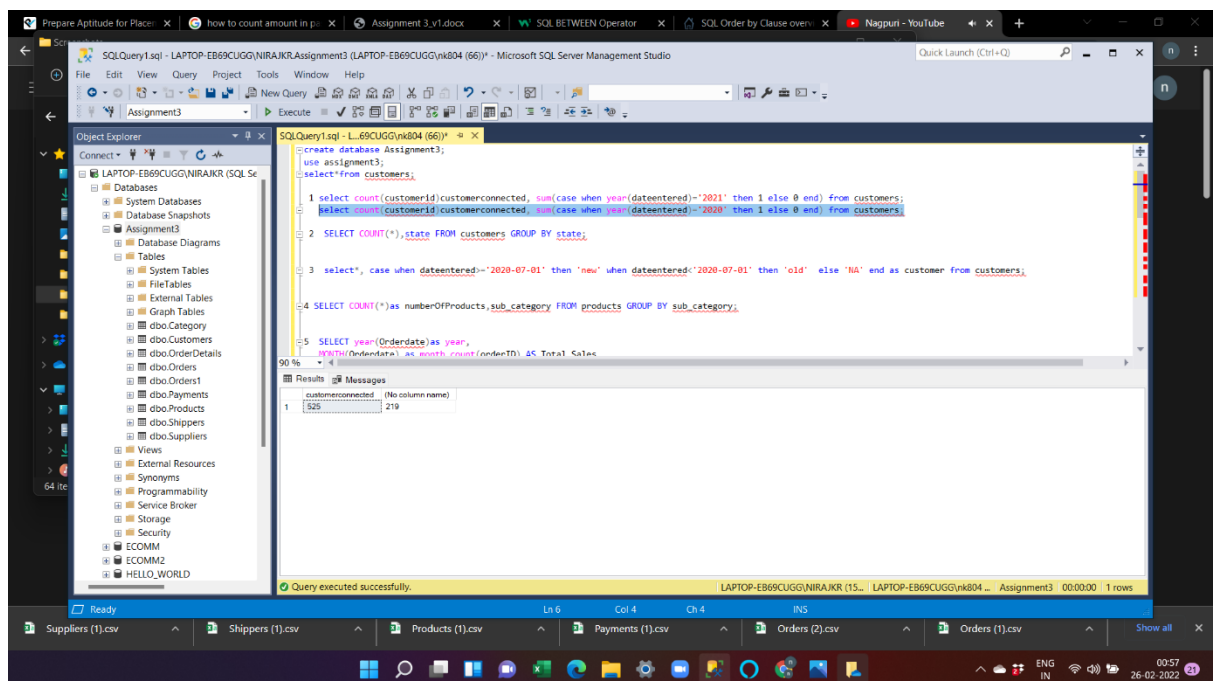
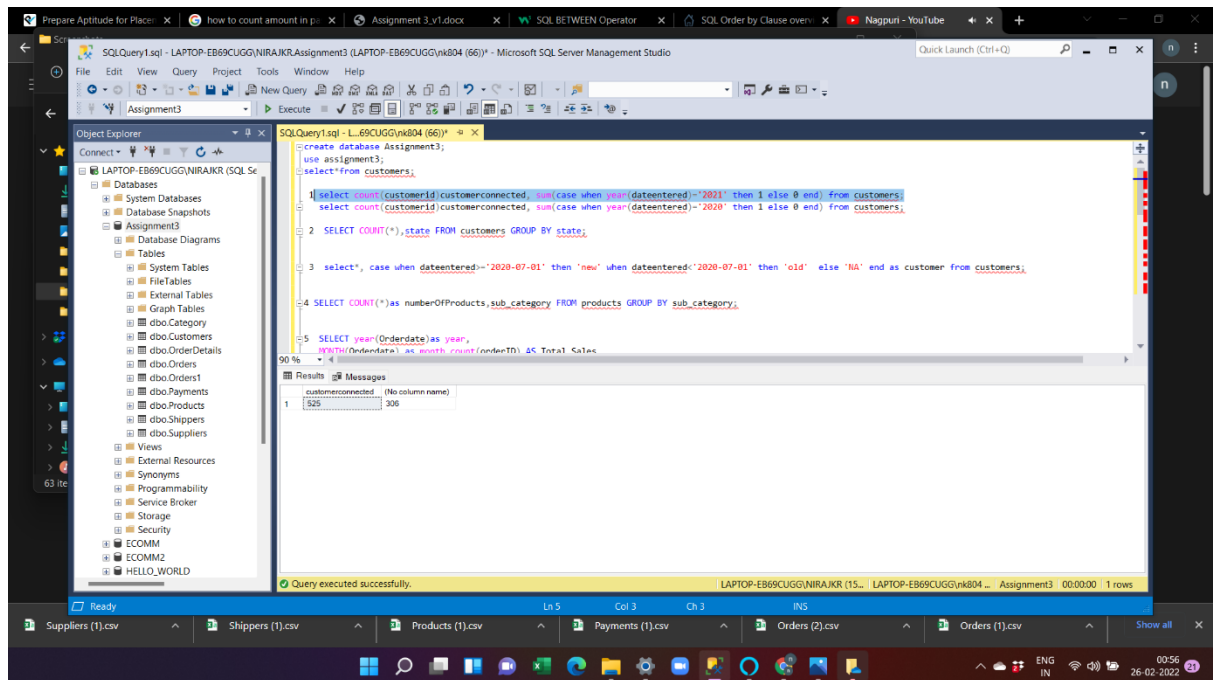


## Q.NO.1 ANSWERS:

1.



```
select count(customerid)customerconnected, sum(case when year(dateentered)='2021'
then 1 else 0 end) from customers;
select count(customerid)customerconnected, sum(case when year(dateentered)='2020'
then 1 else 0 end) from customers;
```

2.

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The SQL query editor contains the following code:

```

--create database Assignment3;
--use assignment3;
--select*from customers;

1 select count(customerid)customerconnected, sum(case when year(dateentered)='2021' then 1 else 0 end) from customers;
select count(customerid)customerconnected, sum(case when year(dateentered)='2020' then 1 else 0 end) from customers;
2 SELECT COUNT(*),state FROM customers GROUP BY state;

3 select, case when dateentered>='2020-07-01' then 'new' when dateentered<'2020-07-01' then 'old' else 'NA' end as customer from customers;

--4 SELECT COUNT(*)as numberOfProducts,sub_category FROM products GROUP BY sub_category;

--5 SELECT year(Orderdate)as year,
    sum(TotalSales) as month_count(month(TotalSales)) as Total Sales
    FROM orders
    GROUP BY year, month;
  
```

The Results pane shows the output of the second query:

(No column name)	state
1	Arizona
2	Alabama
3	Alaska
4	Australian Capital Territory
5	Auvergne-Rhône-Alpes
6	Basel
7	Basel-Stadt
8	Berlin
9	Bourgogne-Franche-Comté
10	Braga
11	Bremen
12	Brussels
13	Brussels-Capital
14	Bucharest

SELECT COUNT(\*),state FROM customers GROUP BY state;

3.

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The SQL query editor contains the following code:

```

--create database Assignment3;
--use assignment3;
--select*from customers;

1 select count(customerid)customerconnected, sum(case when year(dateentered)='2021' then 1 else 0 end) from customers;
select count(customerid)customerconnected, sum(case when year(dateentered)='2020' then 1 else 0 end) from customers;
2 SELECT COUNT(*),state FROM customers GROUP BY state;

3 select, case when dateentered>='2020-07-01' then 'new' when dateentered<'2020-07-01' then 'old' else 'NA' end as customer from customers;

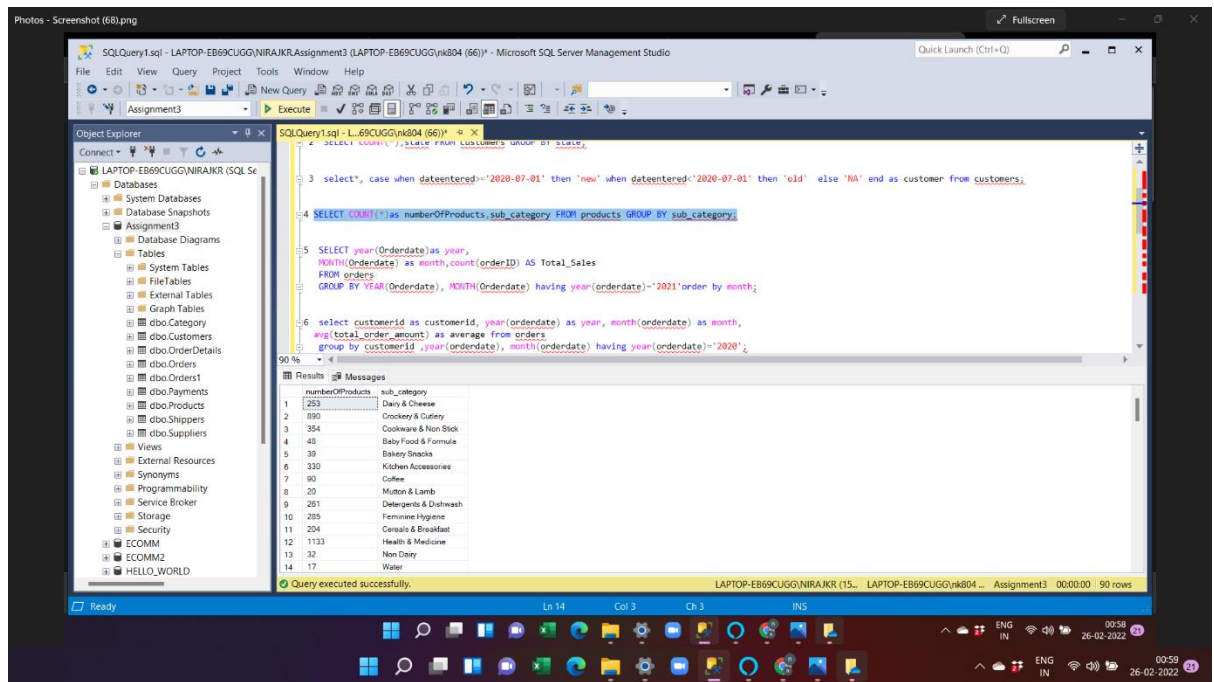
--4 SELECT COUNT(*)as numberOfProducts,sub_category FROM products GROUP BY sub_category;

--5 SELECT year(Orderdate)as year,
    sum(TotalSales) as month_count(month(TotalSales)) as Total Sales
    FROM orders
    GROUP BY year, month;
  
```

The Results pane shows the output of the third query:

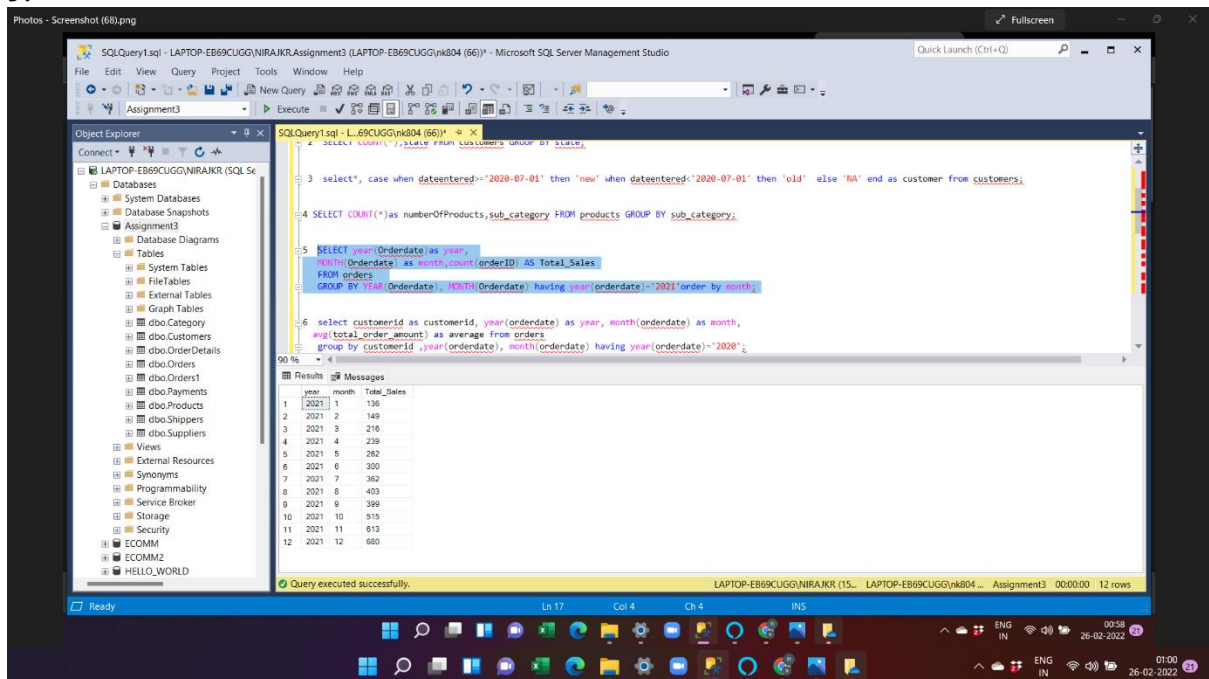
CustomerID	FirstName	LastName	City	State	Country	PostalCode	Phone	Email	DateEntered	customer
1	James	Smith	New York	New York	United States	20082	963848394	James.Smith@gmail.com	2020-01-02	old
2	Robert	Downey Jr	New York	New York	United States	37673	658220115	Robert.Downey.Jr@gmail.com	2020-01-06	old
3	John	Williams	Chicago	Illinois	United States	45529	7641021429	John.Williams@gmail.com	2020-01-11	old
4	Michael	Johnson	Brisbane	Queensland	Australia	260866	7354232181	Michael.Johnson@gmail.com	2020-01-16	old
5	Steve	Williams	Bremen	Bremen	Germany	74038	625552036	Steve.Williams@gmail.com	2020-01-17	old
6	David	Beckham	Vilach	Carinthia	Austria	421492	755761677	David.Beckham@gmail.com	2020-01-20	old
7	Richard	Brown	San Antonio	Texas	United States	110945	8420963031	Richard.Brown@gmail.com	2020-01-20	old
8	Joseph	James	Amsterdam	North Holland	Netherlands	108996	6691466381	Joseph.James@gmail.com	2020-01-25	old
9	Thomas	Jones	Dallas	Texas	United States	174080	8024463594	Thomas.Jones@gmail.com	2020-01-26	old
10	Charles	King	Warsaw	Mazowiec	Poland	589538	8495454902	Charles.King@gmail.com	2020-01-27	old
11	Christopher	Garcia	Austin	Texas	United States	772795	9615650459	Christopher.Garcia@gmail.com	2020-02-03	old
12	Daniel	Fleisch	Dublin	Dublin	Ireland	189723	7947012222	Daniel.Fleisch@gmail.com	2020-02-03	old
13	Mathew	Miller	Brisbane	Queensland	Australia	901170	7784122382	Mathew.Miller@gmail.com	2020-02-04	old
14	Anthony	James	Brussels	Brussels-Capital	Belgium	322713	9371469569	Anthony.James@gmail.com	2020-02-07	old

select\*, case when dateentered>='2020-07-01' then 'new' when dateentered<'2020-07-01' then 'old' else 'NA' end as customer from customers;



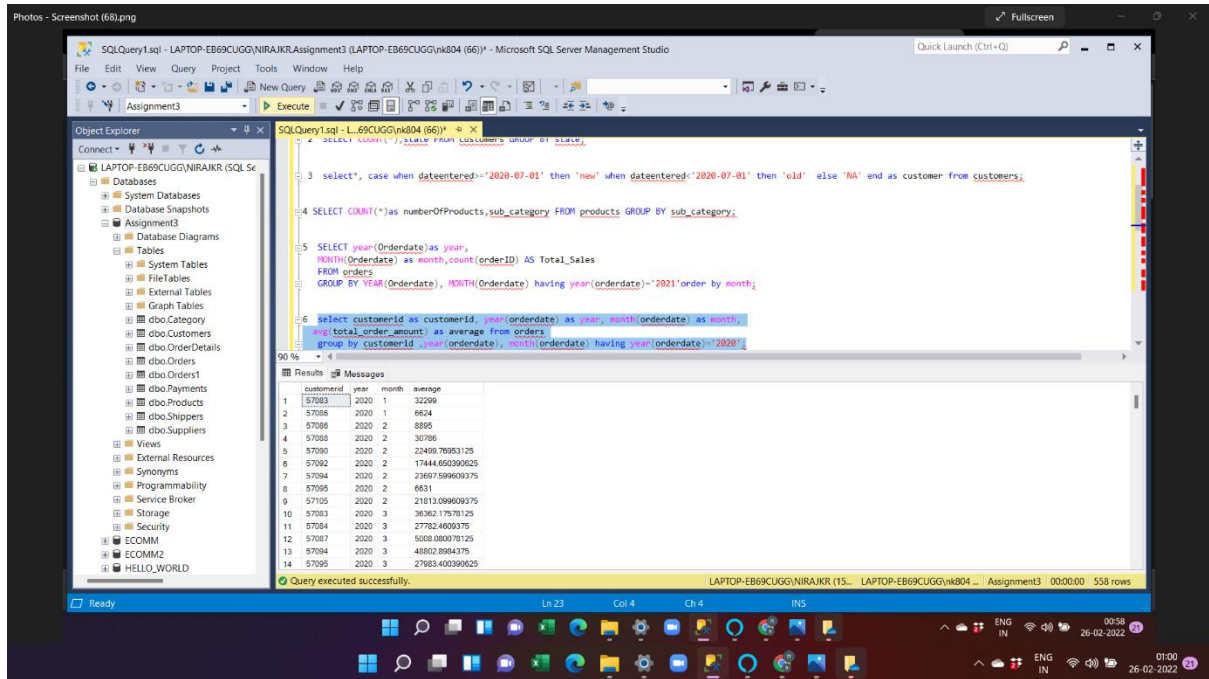
4. `SELECT COUNT(*) as numberOfProducts, sub_category FROM products GROUP BY sub_category;`

5.



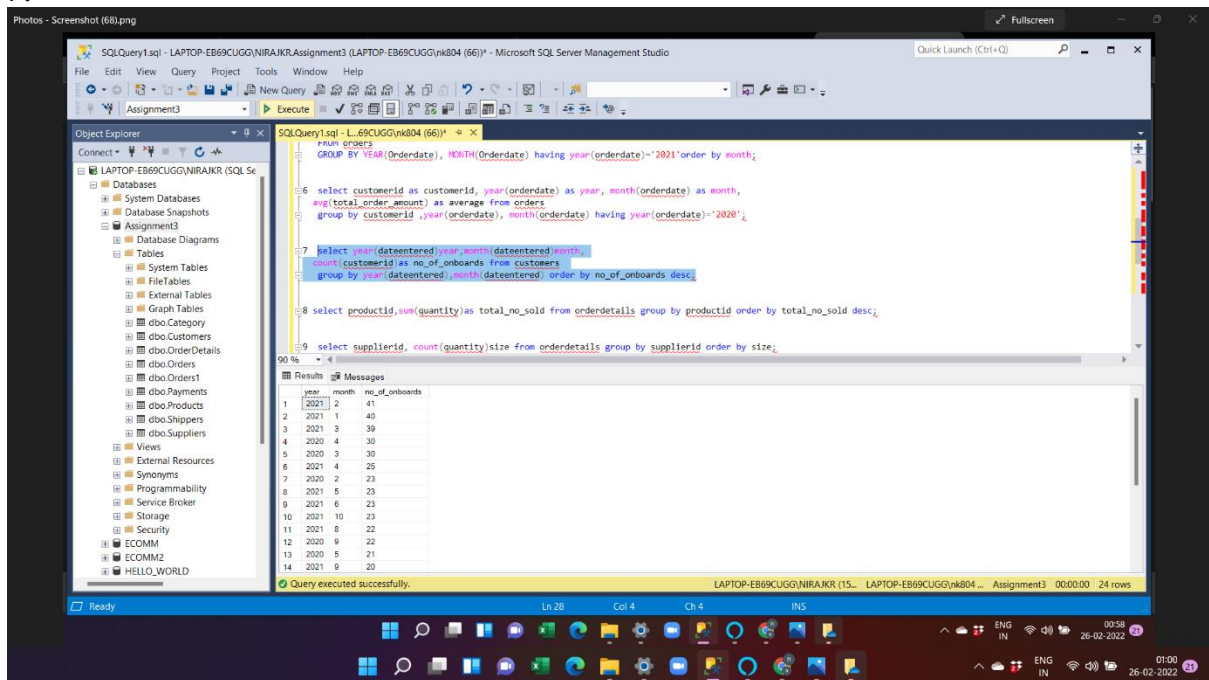
`SELECT year(Orderdate) as year,  
MONTH(Orderdate) as month, count(orderID) AS Total_Sales  
FROM orders  
GROUP BY YEAR(Orderdate), MONTH(Orderdate) having year(orderdate)='2021' order by month;`

6.



```
select customerid as customerid, year(orderdate) as year, month(orderdate) as month,
avg(total_order_amount) as average from orders
group by customerid ,year(orderdate), month(orderdate) having
year(orderdate)='2020';
```

7.



```
select year(dateentered)year,month(dateentered)month,
count(customerid)as no_of_onboards from customers
group by year(dateentered),month(dateentered) order by no_of_onboards desc;
```



8.

The screenshot shows the Microsoft SQL Server Management Studio interface. The query window displays the following SQL code:

```

-- Run orders
GROUP BY YEAR(Orderdate), MONTH(Orderdate) having year(Orderdate)='2021' order by month;

6 select customerid as customerid, year(Orderdate) as year, month(Orderdate) as month,
avg(total_order_amount) as average from orders
group by customerid, year(Orderdate), month(Orderdate) having year(Orderdate)='2020';

7 select year(dateentered) as year, month(dateentered) as month,
count(customerid) as no_of_onboards from customers
group by year(dateentered), month(dateentered) order by no_of_onboards desc;

8 select productid, sum(quantity) as total_no_sold from orderdetails group by productid order by total_no_sold desc;

9 select supplierid, count(quantity) as size from orderdetails group by supplierid order by size;
15day *****

```

The results pane shows the output for query 8:

productid	total_no_sold
1	2274
2	22325
3	5122
4	15388
5	2294
6	19435
7	9955
8	26729
9	19712
10	13797
11	14171
12	25261
13	4277
14	8375
15	23760
16	4882
17	71

select productid, sum(quantity) as total\_no\_sold from orderdetails group by productid order by total\_no\_sold desc;

9.

The screenshot shows the Microsoft SQL Server Management Studio interface. The query window displays the following SQL code:

```

-- Run orders
GROUP BY YEAR(Orderdate), MONTH(Orderdate) having year(Orderdate)='2021' order by month;

6 select customerid as customerid, year(Orderdate) as year, month(Orderdate) as month,
avg(total_order_amount) as average from orders
group by customerid, year(Orderdate), month(Orderdate) having year(Orderdate)='2020';

7 select year(dateentered) as year, month(dateentered) as month,
count(customerid) as no_of_onboards from customers
group by year(dateentered), month(dateentered) order by no_of_onboards desc;

8 select productid, sum(quantity) as total_no_sold from orderdetails group by productid order by total_no_sold desc;

9 select supplierid, count(quantity) as size from orderdetails group by supplierid order by size;
15day *****

```

The results pane shows the output for query 9:

supplierid	size
1	4566
2	4574
3	4578
4	4589
5	4607
6	4613

select supplierid, count(quantity) as size from orderdetails group by supplierid order by size;

11.

SQLQuery1.sql - LAPTOP-EB69CUGG\NIRAIKR.Assignment3 (LAPTOP-EB69CUGG\pk804 (66)) - Microsoft SQL Server Management Studio

```

9 select supplierid, count(quantity) size from orderdetails group by supplierid order by size;
15day *****
11 select customerid, orderdate, sum(total_order_amount) as totalamount from orders
   group by customerid, orderdate having orderdate >= '2021-10-01' and sum(total_order_amount) >= '7000' order by customerid;
*****
12 select*into order_2020 from orders where year(cast(orderdate as date))='2020';
   select*into order_2021 from orders where year(cast(orderdate as date))='2021';
   select*from(select shipperid from order_2021 except select shipperid from order_2020)c;

```

	customerid	orderdate	totalamount
1	57081	2021-11-30	55920
2	57082	2021-11-24	16030.08078125
3	57084	2021-12-28	37827
4	57085	2021-12-12	19396.44921875
5	57087	2021-10-18	9624
6	57087	2021-11-10	21390.99909375
7	57089	2021-11-25	12820.7001953125
8	57089	2021-11-26	16978.75
9	57092	2021-10-19	16775.263671875
10	57092	2021-12-27	8872.7998046875
11	57093	2021-12-08	7614
12	57096	2021-12-18	19052.599375
13	57096	2021-12-28	16915.19921875
14	57097	2021-12-06	16944.5703125
15	57097	2021-12-14	10091.8003515625
16	57099	2021-11-21	10729.25

Query executed successfully.

```

select customerid ,orderdate, sum(total_order_amount)as totalamount from orders
   group by customerid,orderdate having orderdate>='2021-10-01' and
sum(total_order_amount)>='7000' order by customerid;
*****

```

12.

SQLQuery1.sql - LAPTOP-EB69CUGG\NIRAIKR.Assignment3 (LAPTOP-EB69CUGG\pk804 (66)) - Microsoft SQL Server Management Studio

```

9 select supplierid, count(quantity) size from orderdetails group by supplierid order by size;
15day *****
11 select customerid, orderdate, sum(total_order_amount) as totalamount from orders
   group by customerid, orderdate having orderdate >= '2021-10-01' and sum(total_order_amount) >= '7000' order by customerid;
*****
12 select*into order_2020 from orders where year(cast(orderdate as date))='2020';
   select*into order_2021 from orders where year(cast(orderdate as date))='2021';
   select*from(select shipperid from order_2021 except select shipperid from order_2020)c;

```

	shipperid
1	57081

Query executed successfully.

```

select*into order_2020 from orders where year(cast(orderdate as date))='2020';
   select*into order_2021 from orders where year(cast(orderdate as date))='2021';
select*from(select shipperid from order_2021 except select shipperid from
order_2020)c;

```

**Q.5 ANS:** The UNION operation combines the results of two subqueries into a single result that comprises the rows that are returned by both queries.  
The INTERSECT operation combines the results of two queries into a single result that comprises all the rows common to both queries

**Q.6 ANS:** The INTERSECT operation combines the results of two queries into a single result that comprises all the rows common to both queries EXCEPT/MINUS operation finds the difference between the two queries and the result comprises the rows that belong only to the first query

**Q.7 ANS :** CONCAT\_WS() function can do the concatenation along with a separator between strings, whereas in CONCAT() function there is no concept of the separator

Eg: concat() = RAMISBOY and CONCAT\_WS()= RAM IS BOY

**Q.8 ANS:** a. 'esruoC LQS'

B. SQL

C. 'SQL Course'