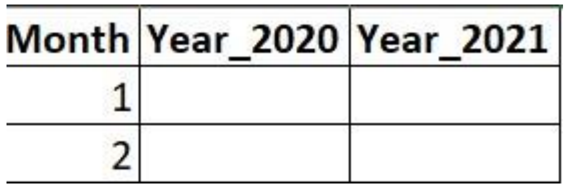
ASSIGNMENT - 5

Q1. You are an Analyst working for an E-commerce company. Your manager had a discussion with the leadership and asked you to work on the analysis below.

A. Create a YOY analysis for the count of customers enrolled with the company each month. The output should look like:



Month	Year_2020	Year_2021
1		
2		

B. Find out the top 3 best-selling products in each of the categories that are currently active on the Website

C. Find the out the least selling products in each of the categories that are currently active on the website

D. We are trying to find paired products that are often purchased together by the same user, such as chips and soft drinks, milk and curd etc.. Find the top paired products names.

E. We want to understand the impact of running a campaign during July’21-Oct’21 what was the total sales generated for the categories “Beauty & Hygiene” and “Bevarages” by

a. entire customer base

b. customers who enrolled with the company during the same period F. Create a Quarter-wise ranking in terms of revenue generated in each category in Year 2020

G. Find the top 3 Shipper companies in terms of

a. Average delivery time for each category for the latest year

b. Volume for latest year

H. Find the top 25 customers in terms of

a. Total no. of orders placed for Year 2021

b. Total Purchase Amount for the Year 2021

I. FInd out the difference between the last two order dates for each of the customers and categorize the customers in two categories such that if the difference is less than 5 days tag the customer as “Frequent Buyer” else tag it as “Infrequent”.

J. FInd the cumulative average order amount at a monthly level for year 2021

a. Each category

b. Each customer

K. Find the 3-day rolling average for the total purchase amount by each customer.

L. Create the below table where values for each Payment method should contain the total order amount by each customer resp.

CustomerName	Debit Card	POD	PayPal	Credit Card	Wallet	Net banking
A						
B						
.						
.						
.						

M. Create a Procedure to filter the orders from Orders table where total purchase amount in a each order id < @x and purchase of year is @Y where @x and @y are the inputs provided by the user.

Q2. What is the difference between Group by and Partition by?

Q3. What is the difference between a Temp Table and a View?

Q4. What is the difference between View and CTE?

Q5. What is the difference between Row Number, Rank and Dense Rank?

Q6. Consider the single-column table below.

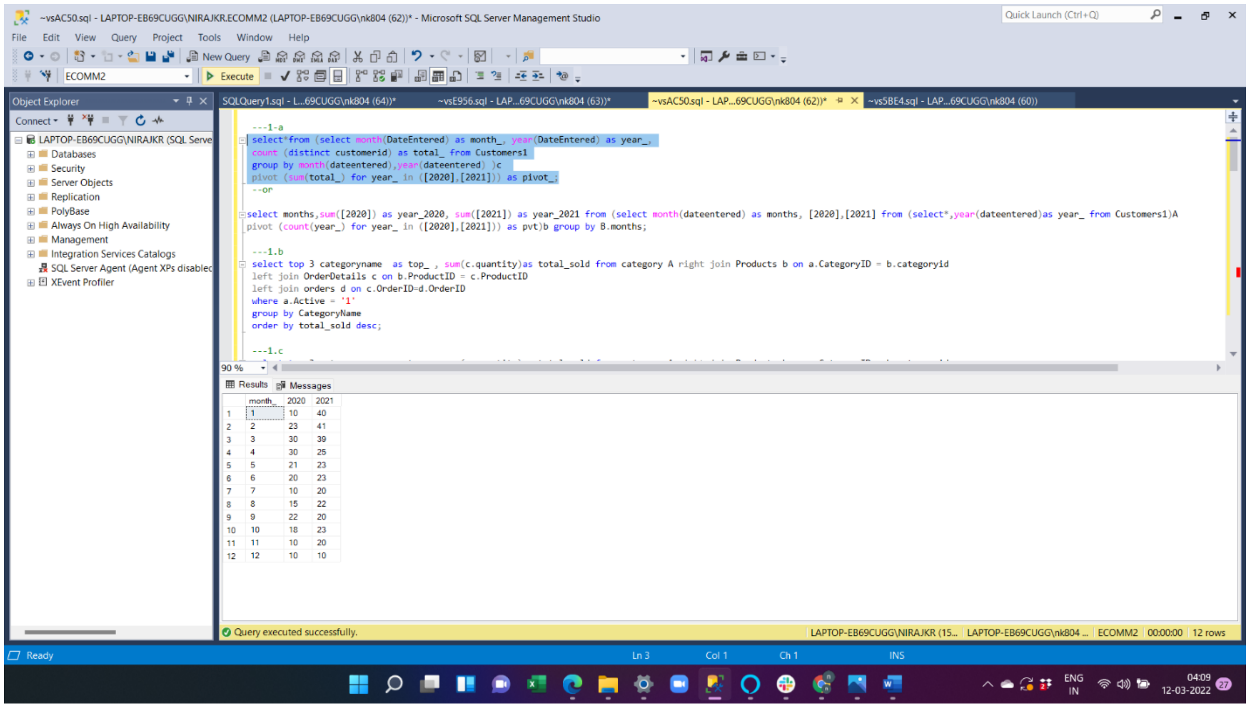
COL1

1
1.2
1.5
2.2
NA
NAN
NULL

Answer the following questions:

1. What is the possible data type of the column ‘COL1’?
2. What will the output of the following SQL statements
   a. ‘SELECT COUNT (*) AS ENTRIES FROM TABLE;’
   b. ‘SELECT COUNT(COL1) AS ENTRIES FROM TABLE;’
   c. ‘SELECT COUNT (DISTINCT COL1) AS ENTRIES FROM TABLE;’

```
---1-a
```

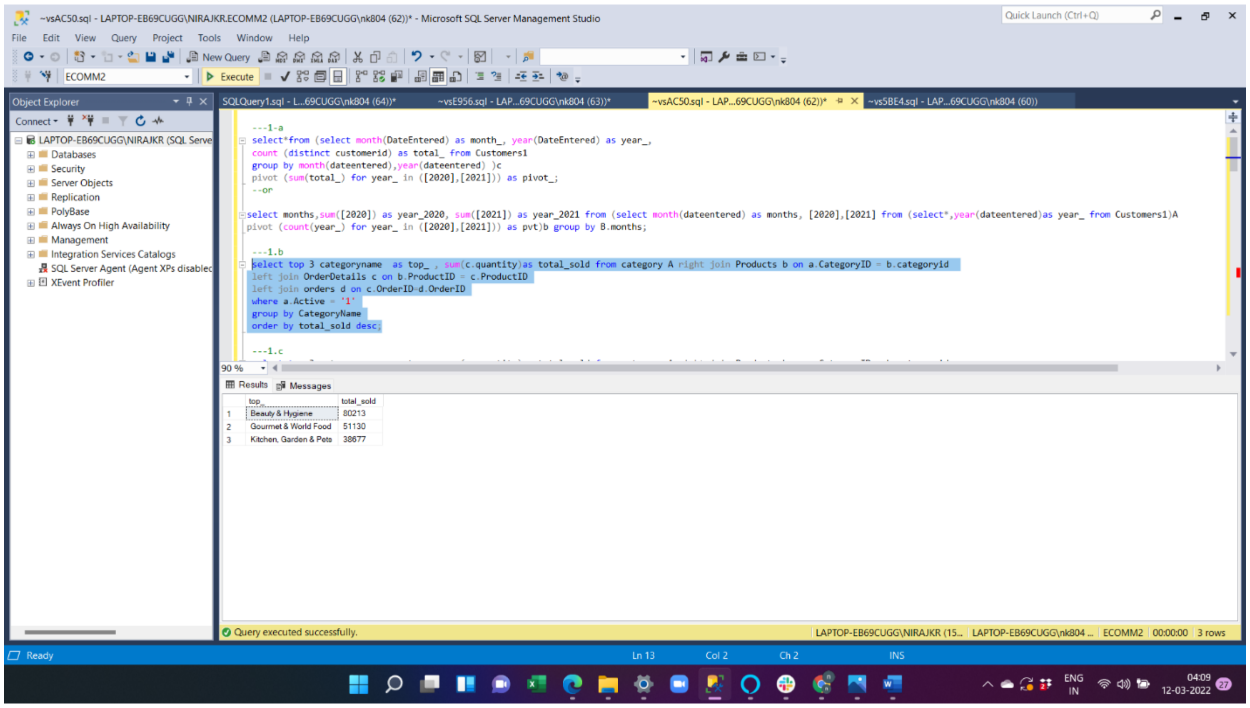


```
---1-a
select*from (select month(DateEntered) as month_, year(DateEntered) as year_,
count (distinct customerid) as total_ from Customers1
group by month(dateentered),year(dateentered) )c
pivot (sum(total_) for year_ in ([2020],[2021])) as pivot_;
--or

select months,sum([2020]) as year_2020, sum([2021]) as year_2021 from (select month(dateentered) as months, [2020],[2021] from (select*,year(dateentered)as year_ from Customers1)A
pivot (count(year_) for year_ in ([2020],[2021])) as pvt)b group by B.months;

---1.b
select top 3 categoryname  as top_ , sum(c.quantity)as total_sold from category A right join Products b on a.CategoryID = b.categoryid
left join OrderDetails c on b.ProductID = c.ProductID
left join orders d on c.OrderID=d.OrderID
where a.Active = '1'
group by CategoryName
order by total_sold desc;

---1.c
```

	month_	2020	2021
1	1	10	40
2	2	23	41
3	3	30	39
4	4	30	25
5	5	21	23
6	6	20	23
7	7	10	20
8	8	15	22
9	9	22	20
10	10	18	23
11	11	10	20
12	12	10	10

Query executed successfully.

```
select*from (select month(DateEntered) as month_, year(DateEntered) as year_,
count (distinct customerid) as total_ from Customers1
group by month(dateentered),year(dateentered) )c
pivot (sum(total_) for year_ in ([2020],[2021])) as pivot_;


--or

select months,sum([2020]) as year_2020, sum([2021]) as year_2021 from (select
month(dateentered) as months, [2020],[2021] from (select*,year(dateentered)as year_ from
Customers1)A
pivot (count(year_) for year_ in ([2020],[2021])) as pvt)b group by B.months;

---1.b
```

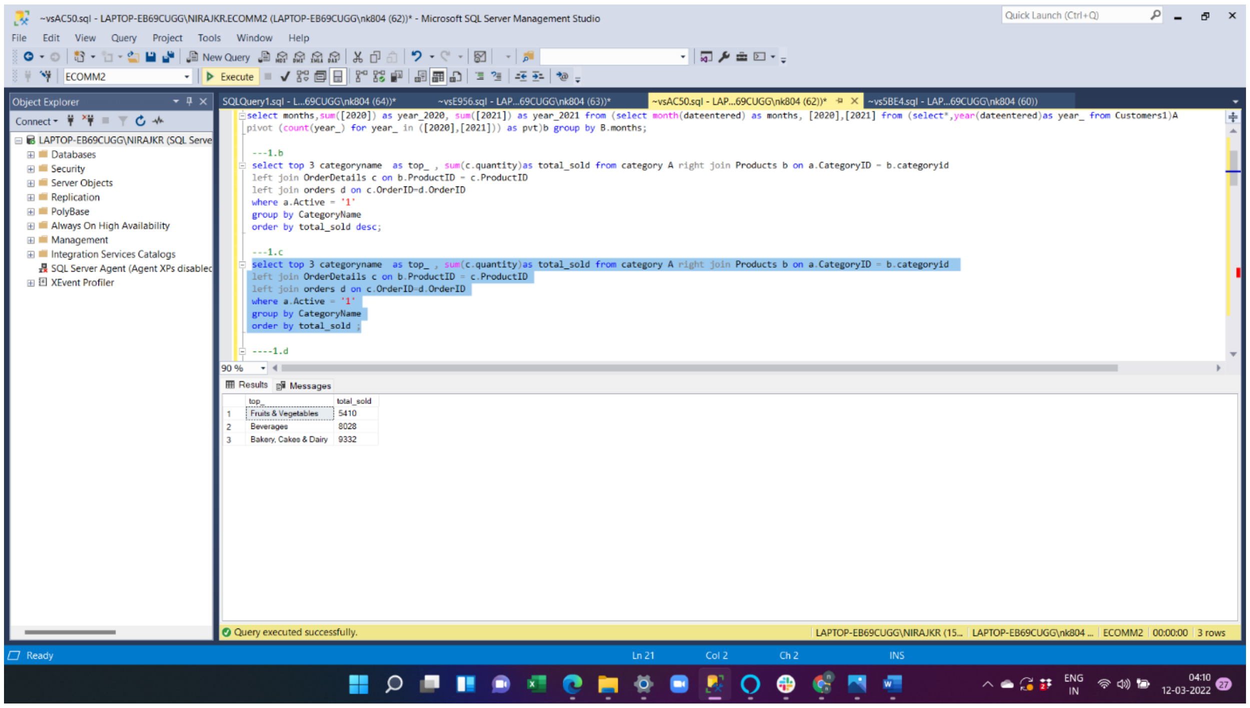


```
---1-a
select*from (select month(DateEntered) as month_, year(DateEntered) as year_,
count (distinct customerid) as total_ from Customers1
group by month(dateentered),year(dateentered) )c
pivot (sum(total_) for year_ in ([2020],[2021])) as pivot_;
--or

select months,sum([2020]) as year_2020, sum([2021]) as year_2021 from (select month(dateentered) as months, [2020],[2021] from (select*,year(dateentered)as year_ from Customers1)A
pivot (count(year_) for year_ in ([2020],[2021])) as pvt)b group by B.months;

---1.b
select top 3 categoryname  as top_ , sum(c.quantity)as total_sold from category A right join Products b on a.CategoryID = b.categoryid
left join OrderDetails c on b.ProductID = c.ProductID
left join orders d on c.OrderID=d.OrderID
where a.Active = '1'
group by CategoryName
order by total_sold desc;

---1.c
```

```
   top_                   total_sold
1  Beauty & Hygiene       80213
2  Gourmet & World Food   51130
3  Kitchen, Garden & Pets 38677
```

Query executed successfully.

```
select top 3 categoryname  as top_ , sum(c.quantity)as total_sold from category A right
join Products b on a.CategoryID = b.categoryid
 left join OrderDetails c on b.ProductID = c.ProductID
 left join orders d on c.OrderID=d.OrderID
 where a.Active = '1'
 group by CategoryName
 order by total_sold desc;

 ---1.c
```

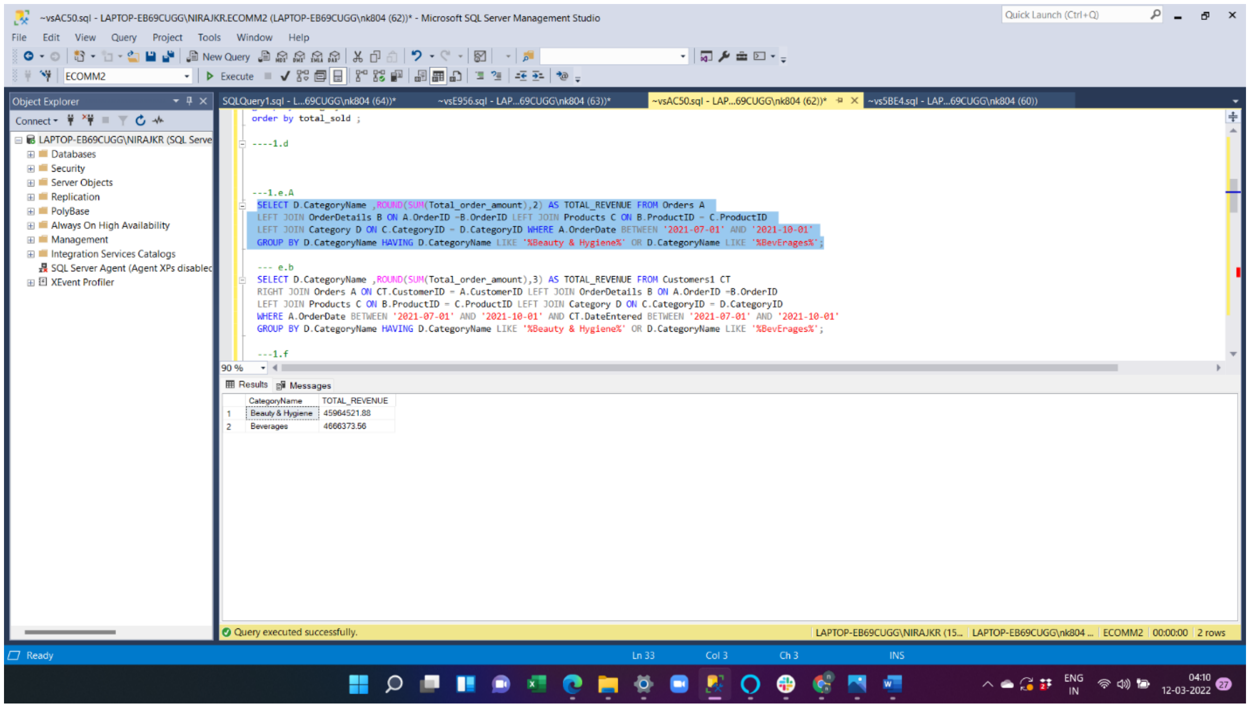


```
select months,sum([2020]) as year_2020, sum([2021]) as year_2021 from (select month(dateentered) as months, [2020],[2021] from (select*,year(dateentered)as year_ from Customers1)A
pivot (count(year_) for year_ in ([2020],[2021])) as pvt)b group by B.months;

---1.b
select top 3 categoryname  as top_ , sum(c.quantity)as total_sold from category A right join Products b on a.CategoryID = b.categoryid
left join OrderDetails c on b.ProductID = c.ProductID
left join orders d on c.OrderID=d.OrderID
where a.Active = '1'
group by CategoryName
order by total_sold desc;

---1.c
select top 3 categoryname  as top_ , sum(c.quantity)as total_sold from category A right join Products b on a.CategoryID = b.categoryid
left join OrderDetails c on b.ProductID = c.ProductID
left join orders d on c.OrderID=d.OrderID
where a.Active = '1'
group by CategoryName
order by total_sold ;

----1.d
```

```
   top_                   total_sold
1  Fruits & Vegetables    5410
2  Beverages              8028
3  Bakery, Cakes & Dairy  9332
```

Query executed successfully.

```
 select top 3 categoryname  as top_ , sum(c.quantity)as total_sold from category A right
join Products b on a.CategoryID = b.categoryid
 left join OrderDetails c on b.ProductID = c.ProductID
 left join orders d on c.OrderID=d.OrderID
 where a.Active = '1'
 group by CategoryName
 order by total_sold ;

 ----1.d



 ---1.e.A
```



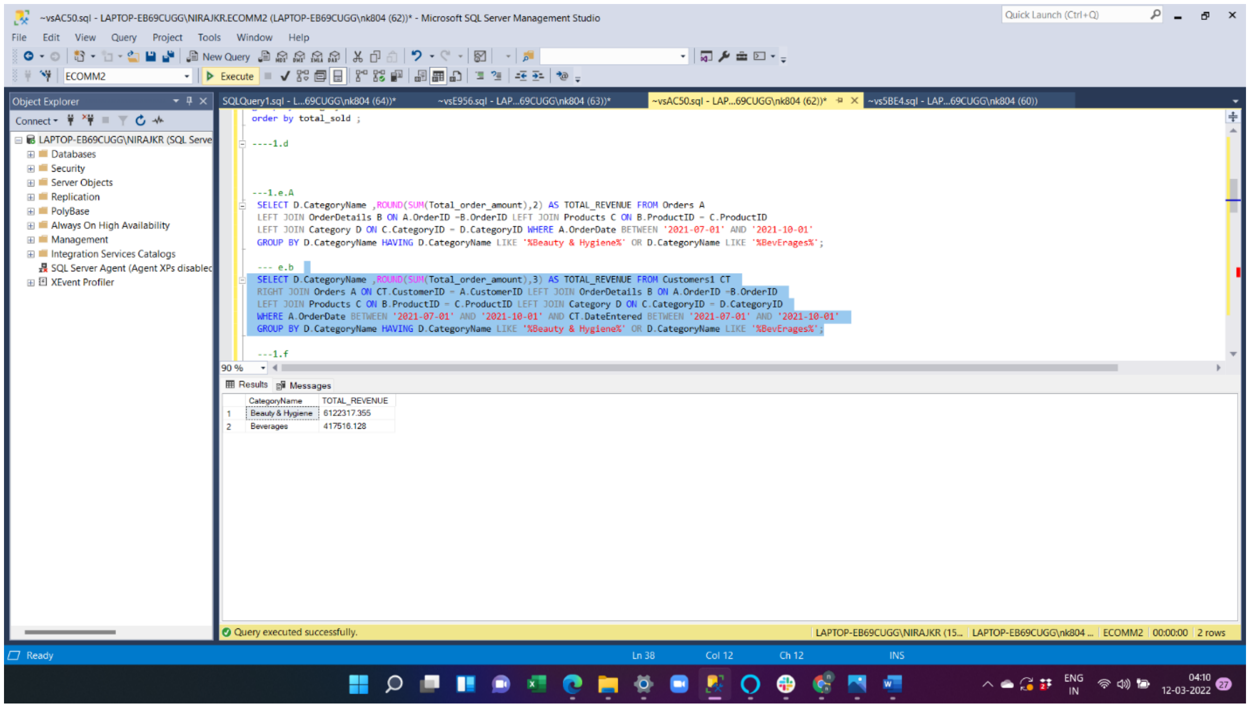
```
order by total_sold ;

----1.d



---1.e.A
 SELECT D.CategoryName ,ROUND(SUM(Total_order_amount),2) AS TOTAL_REVENUE FROM Orders A
 LEFT JOIN OrderDetails B ON A.OrderID =B.OrderID LEFT JOIN Products C ON B.ProductID = C.ProductID
 LEFT JOIN Category D ON C.CategoryID = D.CategoryID WHERE A.OrderDate BETWEEN '2021-07-01' AND '2021-10-01'
 GROUP BY D.CategoryName HAVING D.CategoryName LIKE '%Beauty & Hygiene%' OR D.CategoryName LIKE '%BevErages%';

 --- e.b
 SELECT D.CategoryName ,ROUND(SUM(Total_order_amount),3) AS TOTAL_REVENUE FROM Customers1 CT
 RIGHT JOIN Orders A ON CT.CustomerID = A.CustomerID LEFT JOIN OrderDetails B ON A.OrderID =B.OrderID
 LEFT JOIN Products C ON B.ProductID = C.ProductID LEFT JOIN Category D ON C.CategoryID = D.CategoryID
 WHERE A.OrderDate BETWEEN '2021-07-01' AND '2021-10-01' AND CT.DateEntered BETWEEN '2021-07-01' AND '2021-10-01'
 GROUP BY D.CategoryName HAVING D.CategoryName LIKE '%Beauty & Hygiene%' OR D.CategoryName LIKE '%BevErages%';

 ---1.f
```

	CategoryName	TOTAL_REVENUE
1	Beauty & Hygiene	45964521.88
2	Beverages	4666373.56

Query executed successfully.

```
  SELECT D.CategoryName ,ROUND(SUM(Total_order_amount),2) AS TOTAL_REVENUE FROM Orders A
  LEFT JOIN OrderDetails B ON A.OrderID =B.OrderID LEFT JOIN Products C ON B.ProductID =
C.ProductID
  LEFT JOIN Category D ON C.CategoryID = D.CategoryID WHERE A.OrderDate BETWEEN '2021-07-
01' AND '2021-10-01'
  GROUP BY D.CategoryName HAVING D.CategoryName LIKE '%Beauty & Hygiene%' OR
D.CategoryName LIKE '%BevErages%';
```

```
---1.e.b
```



~vsAC50.sql - LAPTOP-EB69CUGG\NIRAJKR.ECOMM2 (LAPTOP-EB69CUGG\nk804 (62))* - Microsoft SQL Server Management Studio

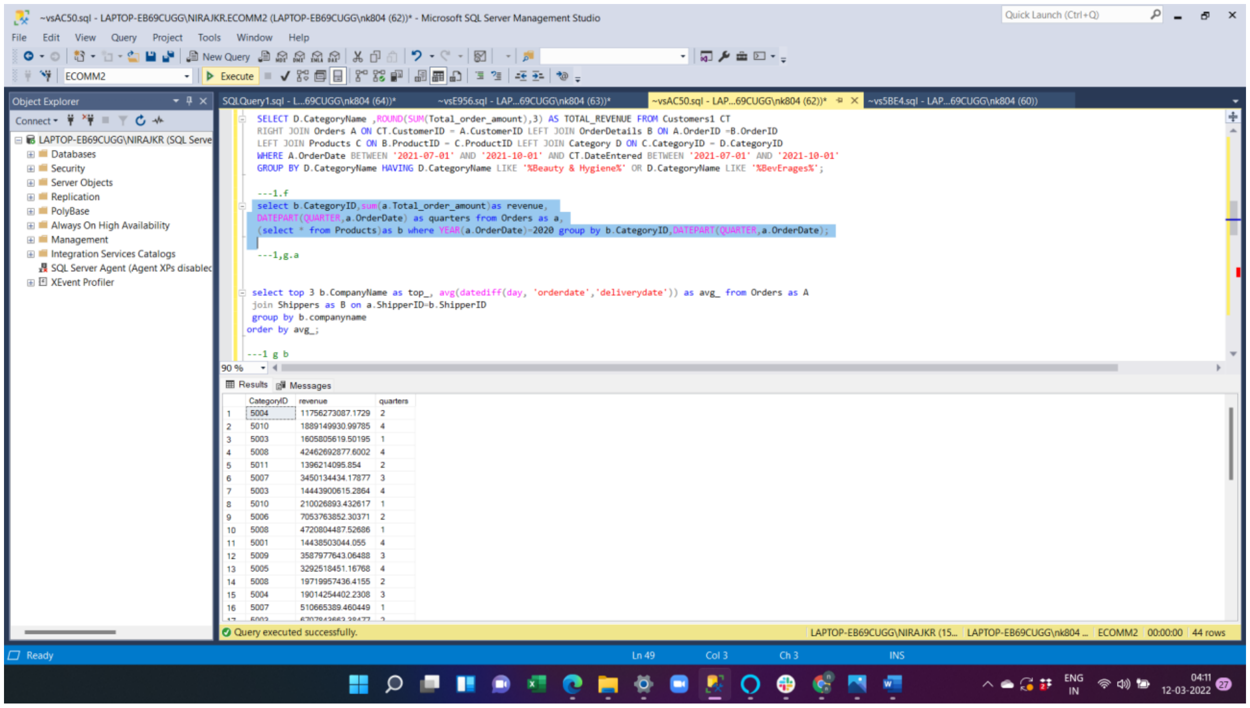
```
order by total_sold ;

----1.d



---1.e.A
 SELECT D.CategoryName ,ROUND(SUM(Total_order_amount),2) AS TOTAL_REVENUE FROM Orders A
 LEFT JOIN OrderDetails B ON A.OrderID =B.OrderID LEFT JOIN Products C ON B.ProductID = C.ProductID
 LEFT JOIN Category D ON C.CategoryID = D.CategoryID WHERE A.OrderDate BETWEEN '2021-07-01' AND '2021-10-01'
 GROUP BY D.CategoryName HAVING D.CategoryName LIKE '%Beauty & Hygiene%' OR D.CategoryName LIKE '%BevErages%';

 --- e.b
 SELECT D.CategoryName ,ROUND(SUM(Total_order_amount),3) AS TOTAL_REVENUE FROM Customers1 CT
 RIGHT JOIN Orders A ON CT.CustomerID = A.CustomerID LEFT JOIN OrderDetails B ON A.OrderID =B.OrderID
 LEFT JOIN Products C ON B.ProductID = C.ProductID LEFT JOIN Category D ON C.CategoryID = D.CategoryID
 WHERE A.OrderDate BETWEEN '2021-07-01' AND '2021-10-01' AND CT.DateEntered BETWEEN '2021-07-01' AND '2021-10-01'
 GROUP BY D.CategoryName HAVING D.CategoryName LIKE '%Beauty & Hygiene%' OR D.CategoryName LIKE '%BevErages%';

 ---1.f
```

	CategoryName	TOTAL_REVENUE
1	Beauty & Hygiene	6122317.355
2	Beverages	417516.128

Query executed successfully. LAPTOP-EB69CUGG\NIRAJKR (15... LAPTOP-EB69CUGG\nk804 ... ECOMM2 00:00:00 2 rows

```
  SELECT D.CategoryName ,ROUND(SUM(Total_order_amount),3) AS TOTAL_REVENUE FROM
Customers1 CT
  RIGHT JOIN Orders A ON CT.CustomerID = A.CustomerID LEFT JOIN OrderDetails B ON
A.OrderID =B.OrderID
  LEFT JOIN Products C ON B.ProductID = C.ProductID LEFT JOIN Category D ON C.CategoryID
= D.CategoryID
  WHERE A.OrderDate BETWEEN '2021-07-01' AND '2021-10-01' AND CT.DateEntered BETWEEN
'2021-07-01' AND '2021-10-01'
  GROUP BY D.CategoryName HAVING D.CategoryName LIKE '%Beauty & Hygiene%' OR
D.CategoryName LIKE '%BevErages%';
```

```
  ---1.f
```



~vsAC50.sql - LAPTOP-EB69CUGG\NIRAJKR.ECOMM2 (LAPTOP-EB69CUGG\nk804 (62))* - Microsoft SQL Server Management Studio

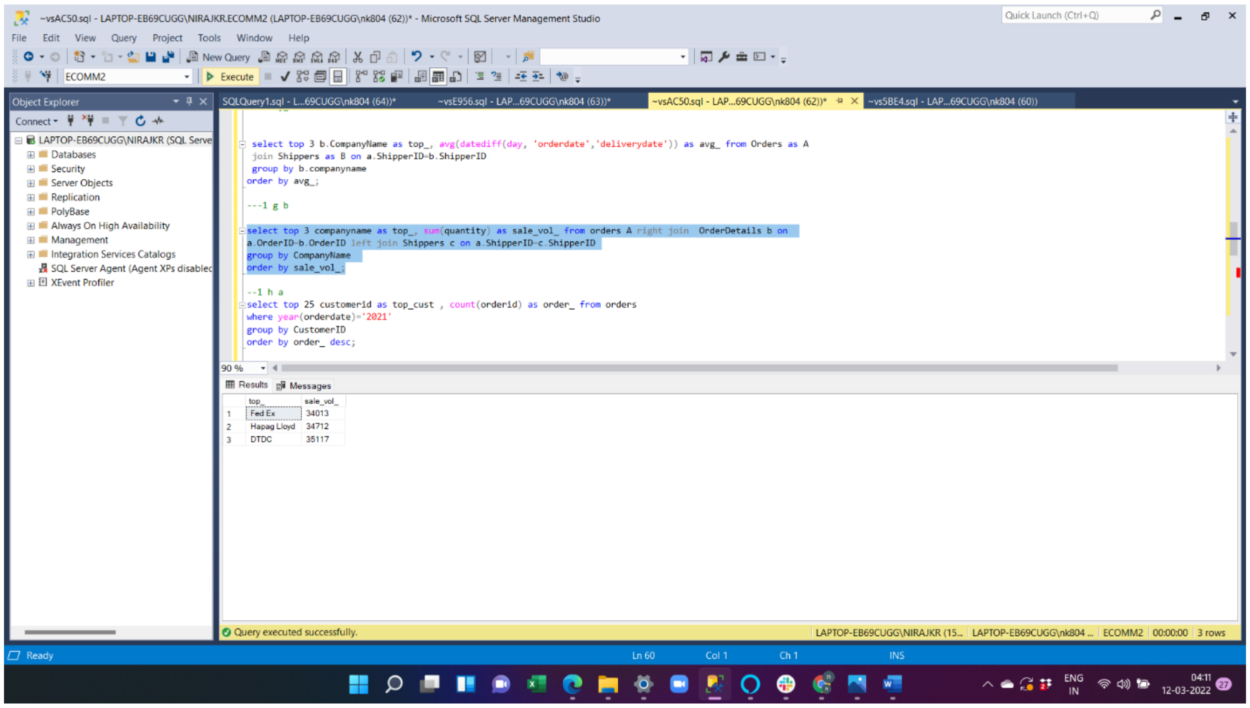
```
SELECT D.CategoryName ,ROUND(SUM(Total_order_amount),3) AS TOTAL_REVENUE FROM Customers1 CT
RIGHT JOIN Orders A ON CT.CustomerID = A.CustomerID LEFT JOIN OrderDetails B ON A.OrderID =B.OrderID
LEFT JOIN Products C ON B.ProductID = C.ProductID LEFT JOIN Category D ON C.CategoryID = D.CategoryID
WHERE A.OrderDate BETWEEN '2021-07-01' AND '2021-10-01' AND CT.DateEntered BETWEEN '2021-07-01' AND '2021-10-01'
GROUP BY D.CategoryName HAVING D.CategoryName LIKE '%Beauty & Hygiene%' OR D.CategoryName LIKE '%BevErages%';

---1.f
select b.CategoryID,sum(a.Total_order_amount)as revenue,
DATEPART(QUARTER,a.OrderDate) as quarters from Orders as a,
(select * from Products)as b where YEAR(a.OrderDate)=2020 group by b.CategoryID,DATEPART(QUARTER,a.OrderDate);

---1,g.a


select top 3 b.CompanyName as top_, avg(datediff(day, 'orderdate','deliverydate')) as avg_ from Orders as A
join Shippers as B on a.ShipperID=b.ShipperID
group by b.companyname
order by avg_;

---1 g b
```

	CategoryID	revenue	quarters
1	5004	11756273087.1729	2
2	5010	1889149930.99785	4
3	5003	1605805619.50195	1
4	5008	42462692877.6002	4
5	5011	1396214095.854	2
6	5007	3450134434.17877	3
7	5003	14443900615.2864	4
8	5010	210026893.432617	1
9	5006	7053763852.30371	2
10	5008	4720804487.52686	1
11	5001	14438503044.055	4
12	5009	3587977643.06488	3
13	5005	3292518451.16768	4
14	5008	19719957436.4155	2
15	5004	19014254402.2308	3
16	5007	510665389.460449	1

Query executed successfully.

LAPTOP-EB69CUGG\NIRAJKR (15... LAPTOP-EB69CUGG\nk804 ... ECOMM2 00:00:00 44 rows

```
  select b.CategoryID,sum(a.Total_order_amount)as revenue,
  DATEPART(QUARTER,a.OrderDate) as quarters from Orders as a,
  (select * from Products)as b where YEAR(a.OrderDate)=2020 group by
b.CategoryID,DATEPART(QUARTER,a.OrderDate);

  ---1,g.a


 select top 3 b.CompanyName as top_, avg(datediff(day, 'orderdate','deliverydate')) as
avg_ from Orders as A
 join Shippers as B on a.ShipperID=b.ShipperID
 group by b.companyname
order by avg_;

---1 g b
```



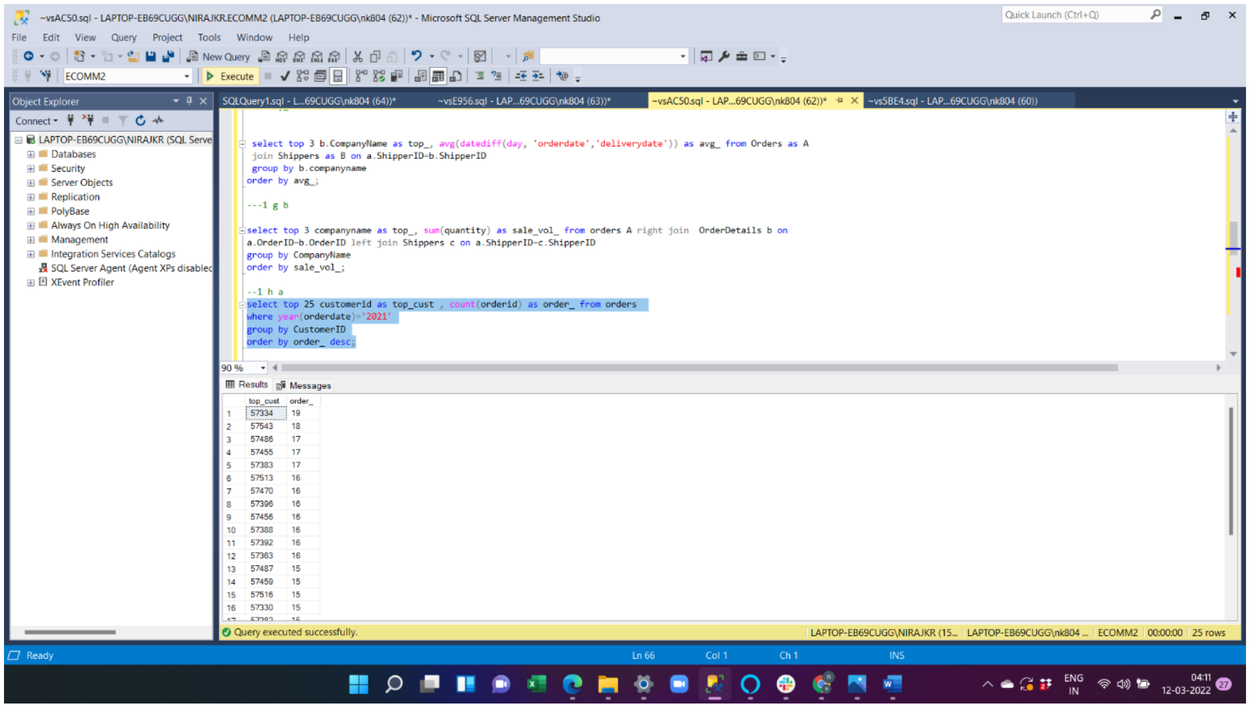
~vsAC50.sql - LAPTOP-EB69CUGG\NIRAJKR.ECOMM2 (LAPTOP-EB69CUGG\nk804 (62))* - Microsoft SQL Server Management Studio

```
select top 3 b.CompanyName as top_, avg(datediff(day, 'orderdate','deliverydate')) as avg_ from Orders as A
 join Shippers as B on a.ShipperID=b.ShipperID
 group by b.companyname
order by avg_;

---1 g b

select top 3 companyname as top_, sum(quantity) as sale_vol_ from orders A right join  OrderDetails b on
a.OrderID=b.OrderID left join Shippers c on a.ShipperID=c.ShipperID
group by CompanyName
order by sale_vol_;

--1 h a
select top 25 customerid as top_cust , count(orderid) as order_ from orders
where year(orderdate)='2021'
group by CustomerID
order by order_ desc;
```

	top_	sale_vol_
1	Fed Ex	34013
2	Hapag Lloyd	34712
3	DTDC	35117

Query executed successfully. LAPTOP-EB69CUGG\NIRAJKR (15... LAPTOP-EB69CUGG\nk804 ... ECOMM2 00:00:00 3 rows

```
select top 3 companyname as top_, sum(quantity) as sale_vol_ from orders A right join
OrderDetails b on
a.OrderID=b.OrderID left join Shippers c on a.ShipperID=c.ShipperID
group by CompanyName
order by sale_vol_;

--1 h a
```



~vsAC50.sql - LAPTOP-EB69CUGG\NIRAJKR.ECOMM2 (LAPTOP-EB69CUGG\nk804 (62))* - Microsoft SQL Server Management Studio

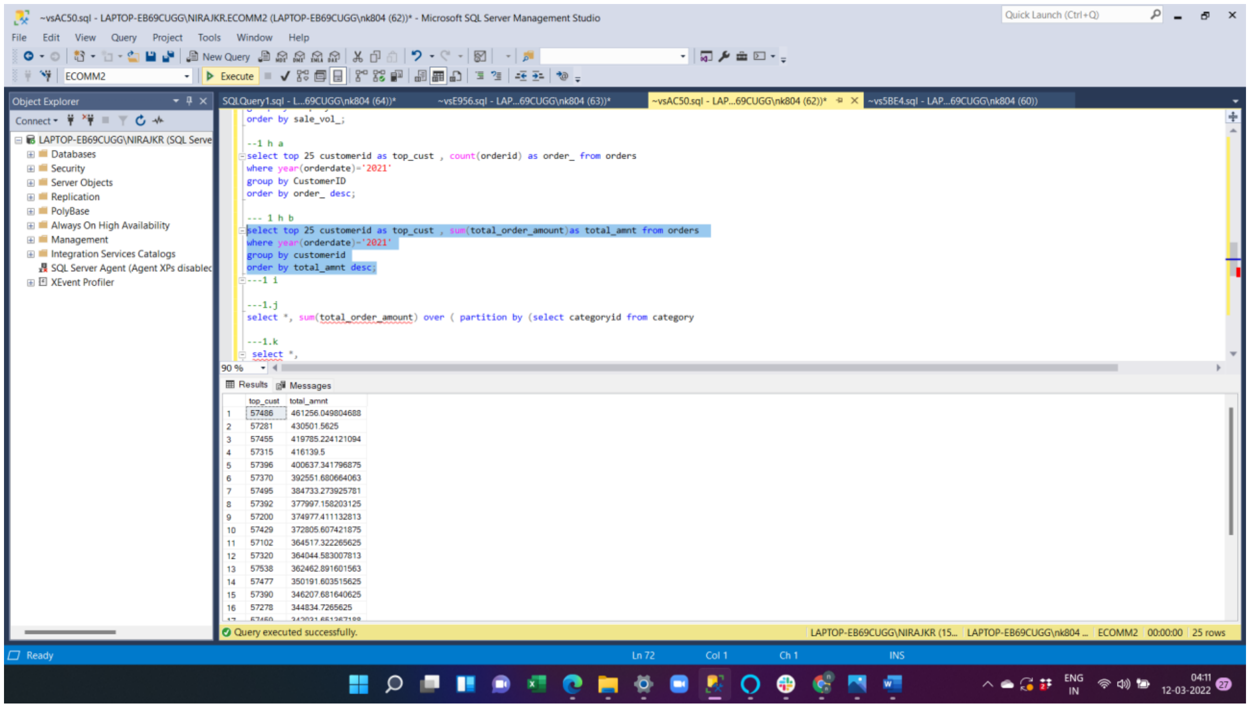
```
select top 3 b.CompanyName as top_, avg(datediff(day, 'orderdate','deliverydate')) as avg_ from Orders as A
 join Shippers as B on a.ShipperID=b.ShipperID
 group by b.companyname
order by avg_;

---1 g b

select top 3 companyname as top_, sum(quantity) as sale_vol_ from orders A right join  OrderDetails b on
a.OrderID=b.OrderID left join Shippers c on a.ShipperID=c.ShipperID
group by CompanyName
order by sale_vol_;

--1 h a
select top 25 customerid as top_cust , count(orderid) as order_ from orders
where year(orderdate)='2021'
group by CustomerID
order by order_ desc;
```

	top_cust	order_
1	57334	19
2	57543	18
3	57486	17
4	57455	17
5	57383	17
6	57513	16
7	57470	16
8	57396	16
9	57456	16
10	57388	16
11	57392	16
12	57363	16
13	57487	15
14	57459	15
15	57516	15
16	57330	15

Query executed successfully. LAPTOP-EB69CUGG\NIRAJKR (15... LAPTOP-EB69CUGG\nk804 ... ECOMM2 00:00:00 25 rows

```
select top 25 customerid as top_cust , count(orderid) as order_ from orders
where year(orderdate)='2021'
```

```
group by CustomerID
order by order_ desc;

--- 1 h b
```



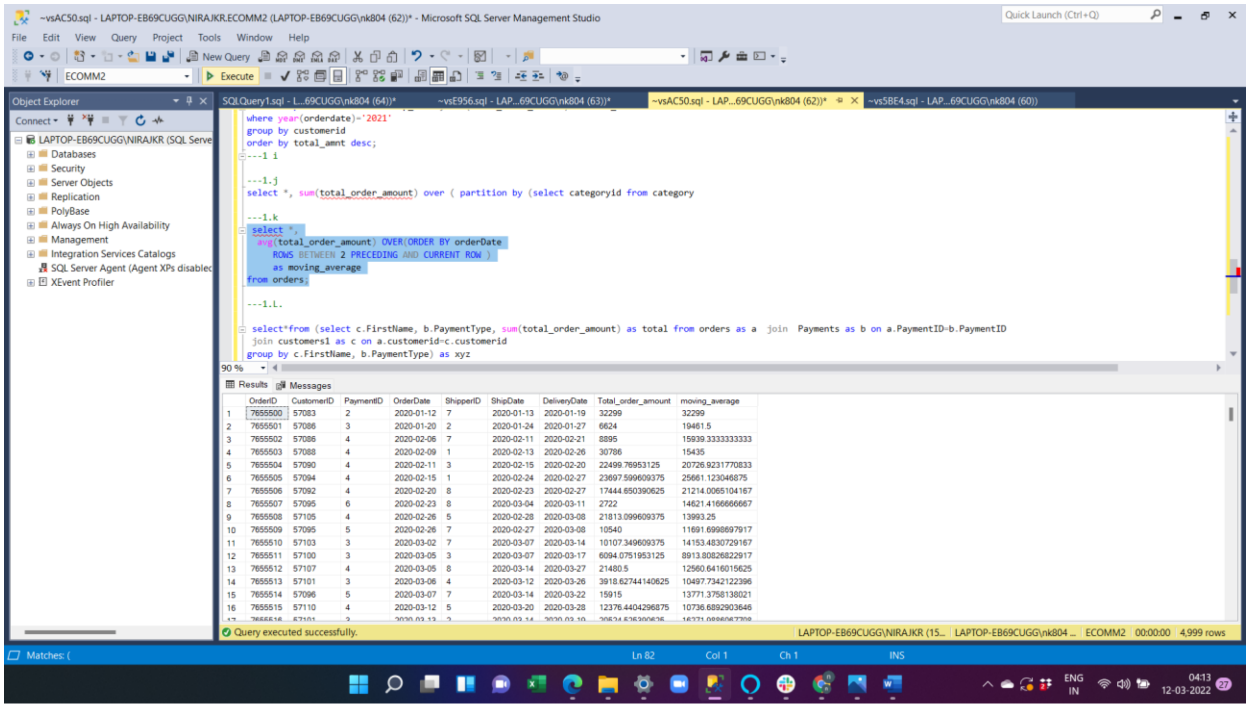
```
order by sale_vol_;

--1 h a
select top 25 customerid as top_cust , count(orderid) as order_ from orders
where year(orderdate)='2021'
group by CustomerID
order by order_ desc;

--- 1 h b
select top 25 customerid as top_cust , sum(total_order_amount)as total_amnt from orders
where year(orderdate)='2021'
group by customerid
order by total_amnt desc;
---1 i

---1.j
select *, sum(total_order_amount) over ( partition by (select categoryid from category

---1.k
 select *,
```

	top_cust	total_amnt
1	57486	461256.049804688
2	57281	430501.5625
3	57455	419785.224121094
4	57315	416139.5
5	57396	400637.341796875
6	57370	392551.680664063
7	57495	384733.273925781
8	57392	377997.158203125
9	57200	374977.411132813
10	57429	372805.607421875
11	57102	364517.322265625
12	57320	364044.583007813
13	57538	362462.891601563
14	57477	350191.603515625
15	57390	346207.681640625
16	57278	344834.7265625

Query executed successfully.

```
select top 25 customerid as top_cust , sum(total_order_amount)as total_amnt from orders
where year(orderdate)='2021'
group by customerid
order by total_amnt desc;
```

```
---1.k
```



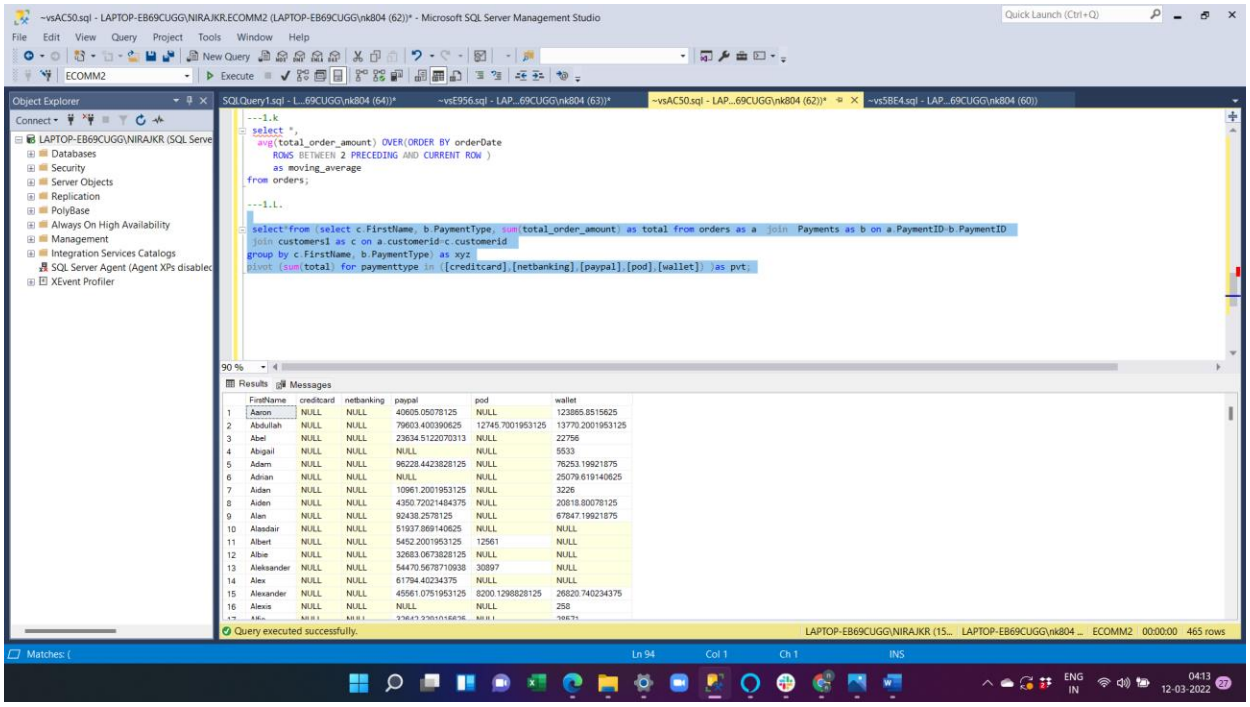
~vsAC50.sql - LAPTOP-EB69CUGG\NIRAJKR.ECOMM2 (LAPTOP-EB69CUGG\nk804 (62))* - Microsoft SQL Server Management Studio

```
where year(orderdate)='2021'
group by customerid
order by total_amnt desc;
---1 i

---1.j
select *, sum(total_order_amount) over ( partition by (select categoryid from category

---1.k
 select *,
  avg(total_order_amount) OVER(ORDER BY orderDate
     ROWS BETWEEN 2 PRECEDING AND CURRENT ROW )
     as moving_average
from orders;

---1.L.

 select*from (select c.FirstName, b.PaymentType, sum(total_order_amount) as total from orders as a  join  Payments as b on a.PaymentID=b.PaymentID
 join customers1 as c on a.customerid=c.customerid
group by c.FirstName, b.PaymentType) as xyz
```

	OrderID	CustomerID	PaymentID	OrderDate	ShipperID	ShipDate	DeliveryDate	Total_order_amount	moving_average
1	7655500	57083	2	2020-01-12	7	2020-01-13	2020-01-19	32299	32299
2	7655501	57086	3	2020-01-20	2	2020-01-24	2020-01-27	6624	19461.5
3	7655502	57086	4	2020-02-06	7	2020-02-11	2020-02-21	8895	15939.3333333333
4	7655503	57088	4	2020-02-09	1	2020-02-13	2020-02-26	30786	15435
5	7655504	57090	4	2020-02-11	3	2020-02-15	2020-02-20	22499.76953125	20726.9231770833
6	7655505	57094	4	2020-02-15	1	2020-02-24	2020-02-27	23697.599609375	25661.123046875
7	7655506	57092	4	2020-02-20	8	2020-02-23	2020-02-27	17444.650390625	21214.0065104167
8	7655507	57095	6	2020-02-23	8	2020-03-04	2020-03-11	2722	14621.4166666667
9	7655508	57105	4	2020-02-26	5	2020-02-28	2020-03-08	21813.099609375	13993.25
10	7655509	57095	5	2020-02-26	7	2020-02-27	2020-03-08	10540	11691.6998697917
11	7655510	57103	3	2020-03-02	7	2020-03-07	2020-03-14	10107.349609375	14153.4830729167
12	7655511	57100	3	2020-03-05	3	2020-03-07	2020-03-17	6094.0751953125	8913.80826822917
13	7655512	57107	4	2020-03-05	8	2020-03-14	2020-03-27	21480.5	12560.6416015625
14	7655513	57101	3	2020-03-06	4	2020-03-12	2020-03-26	3918.62744140625	10497.7342122396
15	7655514	57096	5	2020-03-07	7	2020-03-14	2020-03-22	15915	13771.3758138021
16	7655515	57110	4	2020-03-12	5	2020-03-20	2020-03-28	12376.4404296875	10736.6892903646

Query executed successfully. LAPTOP-EB69CUGG\NIRAJKR (15... LAPTOP-EB69CUGG\nk804 ... ECOMM2 00:00:00 4,999 rows

```
 select *,
  avg(total_order_amount) OVER(ORDER BY orderDate
     ROWS BETWEEN 2 PRECEDING AND CURRENT ROW )
     as moving_average
from orders;

---1.L.
```



~vsAC50.sql - LAPTOP-EB69CUGG\NIRAJKR.ECOMM2 (LAPTOP-EB69CUGG\nk804 (62))* - Microsoft SQL Server Management Studio

```
---1.k
 select *,
  avg(total_order_amount) OVER(ORDER BY orderDate
     ROWS BETWEEN 2 PRECEDING AND CURRENT ROW )
     as moving_average
from orders;

---1.L.

 select*from (select c.FirstName, b.PaymentType, sum(total_order_amount) as total from orders as a  join  Payments as b on a.PaymentID=b.PaymentID
 join customers1 as c on a.customerid=c.customerid
group by c.FirstName, b.PaymentType) as xyz
pivot (sum(total) for paymenttype in ([creditcard],[netbanking],[paypal],[pod],[wallet]) )as pvt;
```

	FirstName	creditcard	netbanking	paypal	pod	wallet
1	Aaron	NULL	NULL	40605.05078125	NULL	123865.8515625
2	Abdullah	NULL	NULL	79603.400390625	12745.7001953125	13770.2001953125
3	Abel	NULL	NULL	23634.5122070313	NULL	22756
4	Abigail	NULL	NULL	NULL	NULL	5533
5	Adam	NULL	NULL	96228.4423828125	NULL	76253.19921875
6	Adrian	NULL	NULL	NULL	NULL	25079.619140625
7	Aidan	NULL	NULL	10961.2001953125	NULL	3226
8	Aiden	NULL	NULL	4350.72021484375	NULL	20818.80078125
9	Alan	NULL	NULL	92438.2578125	NULL	67847.19921875
10	Alasdair	NULL	NULL	51937.869140625	NULL	NULL
11	Albert	NULL	NULL	5452.2001953125	12561	NULL
12	Albie	NULL	NULL	32683.0673828125	NULL	NULL
13	Aleksander	NULL	NULL	54470.5678710938	30897	NULL
14	Alex	NULL	NULL	61794.40234375	NULL	NULL
15	Alexander	NULL	NULL	45561.0751953125	8200.1298828125	26820.740234375
16	Alexis	NULL	NULL	NULL	NULL	258

Query executed successfully. LAPTOP-EB69CUGG\NIRAJKR (15... LAPTOP-EB69CUGG\nk804 ... ECOMM2 00:00:00 465 rows

```
 select*from (select c.FirstName, b.PaymentType, sum(total_order_amount) as total from
orders as a  join  Payments as b on a.PaymentID=b.PaymentID
 join customers1 as c on a.customerid=c.customerid
group by c.FirstName, b.PaymentType) as xyz
pivot (sum(total) for paymenttype in
([creditcard],[netbanking],[paypal],[pod],[wallet]) )as pvt;
```

Q.no.2 answer:-

A GROUP BY normally reduces the number of rows returned by rolling them up and calculating averages or sums for each row.

PARTITION BY does not affect the number of rows returned, but it changes how a window function's result is calculated.

Q.no.3 answer:-

Temporary tables are just the tables in tempdb. Views are stored queries for existing data in existing tables.

Temporary table needs to be populated first with data, and population is the main preformance-concerned issue.

So the data in views already exists and so views are faster than temporary table.

Q.NO 4 answer :-

Views being a physical object on database (but does not store data physically) and can be used on multiple queries, thus provide flexibility and centralized approach.

CTE, on the other hand are temporary and will be created when they are used; that's why they are called as inline view

Q.No 5 answer :-

The row_number gives continuous numbers, while rank and dense_rank give the same rank for duplicates, but the next number in rank is as per continuous order so you will see a jump but in dense_rank doesn't have any gap in rankings.

Eg	rank	: 1,2,2,4,5,5,5,8,9
	Dense_r	: 1,2,2,3,4,4,5,6,7,7,7,8,9
	Row	: 1,2,3,4,5,6,7,8,9

Q.no.06 answer

1. ans – A column can hold a specific datatype but in col1 it has int and string data type which is not possible in a single column.

So, that this kind of table can't be exist.

2. a. 'SELECT COUNT (*) AS ENTRIES FROM TABLE;' :- count the records for each column present in the table including commons and output of count of col1 as entries

b. 'SELECT COUNT(COL1) AS ENTRIES FROM TABLE;' :- count the specific column col1 including commons from the table and output as entries.

c. 'SELECT COUNT (DISTINCT COL1) AS ENTRIES FROM TABLE;':- will count col1 but will not includes the common records from the table;