```
--QUESTIONS
--1: Write a SQL query to find the most profitable category for each customer.
--this SQL query want to determine the category with most profit for each customer

WITH sub as (SELECT s.customer_id, p.category, SUM(s.profit) as SumProfit
                     from dbo.sales s
                     JOIN dbo.product p
                     ON s.product_id = p.product_id
                     JOIN dbo.customer c
                     ON c.customer_id= s.customer_id
                     GROUP BY s.customer_id,  p.category
                     )

SELECT sub.customer_id, sub.category, sub.SumProfit
FROM
sub
JOIN
(SELECT customer_id, MAX(SumProfit) MaxSum
FROM
sub
GROUP BY customer_id) as subMax ON sub.customer_id = subMax.customer_id AND sub.SumProfit
= subMax. MaxSum




--OR the code below ; but this method will not eactly give you the profitable category
for each customer, because it gives
-- almost all the categories.
SELECT s.customer_id, p.category, MAX(profit)
from dbo.sales s
                     JOIN dbo.product p
                     ON s.product_id = p.product_id
                     JOIN dbo.customer c
                     ON c.customer_id= s.customer_id
                     GROUP BY s.customer_id,  p.category






--2:Total number of customers in each segment
SELECT segment, COUNT(customer_id) AS Segment_Count
FROM dbo.customer
GROUP BY segment
ORDER BY 2 DESC;


--Top 5 most profitable products along with their total profits
SELECT TOP 5  product_name, SUM(profit) as Total_Profit
FROM dbo.sales s
JOIN dbo.product  p
ON s.product_id = p.product_id
GROUP BY product_name
ORDER BY 2 DESC;
```

```sql
--Total sales for each category of products
SELECT category, SUM(sales) TotalSales
FROM dbo.sales s
JOIN dbo.product p
ON s.product_id = p.product_id
GROUP BY category;

--Average age of customers in each region

SELECT region ,AVG(Age) Avg_Age
FROM dbo.customer
GROUP BY region



--Product names and their corresponding categories for products that have been ordered at
least once
SELECT DISTINCT      p.product_name,  p.category
FROM dbo.sales s
JOIN dbo.product p
ON s.product_id= p.product_id
WHERE quantity > 0 AND p.product_id IS NOT NULL;



--Total profit earned for each year
SELECT DATEPART(year, order_date) as Order_Year, SUM(profit) as SumProfit
FROM dbo.sales
GROUP BY  DATEPART(year, order_date)
ORDER BY  2 DESC;

--Customer names and their total number of orders for customers who have placed more than
5 orders

SELECT c.customer_name, c.customer_id, count(order_id) TotalOrder
FROM dbo.sales s
JOIN dbo.product p
ON s.product_id= p.product_id
JOIN dbo.customer c on c.customer_id = s.customer_id
GROUP BY c.customer_name, c.customer_id
HAVING count(order_id) > 5
ORDER BY 3 DESC, 1 DESC;

--Products with a discount greater than 20%
SELECT DISTINCT p.product_name, discount
FROM dbo.product p
JOIN dbo.sales s
ON s.product_id= p.product_id
WHERE discount > 0.20


--Total sales for each sub-category in the 'Office Supplies' category
SELECT category, sub_category, SUM(sales)as TotalSales
FROM dbo.product p
JOIN dbo.sales s
ON s.product_id= p.product_id
```

```sql
WHERE category = 'Office Supplies'
GROUP BY category, sub_category


--Customer with the highest total profit
SELECT TOP 1 c.customer_id, c.customer_name, SUM(profit)
FROM dbo.sales s
JOIN dbo.customer c
ON s.customer_id= c.customer_id
GROUP BY c.customer_id, c.customer_name
ORDER BY 3 DESC;


--Write a SQL query to find the customers who have made purchases in every category.
SELECT customer_name,c.customer_id, COUNT( category) as CountCategory
FROM dbo.customer c
JOIN dbo.sales s
ON c.customer_id = s.customer_id
JOIN dbo.product p
ON p.product_id= s.product_id
WHERE quantity > 0
GROUP BY customer_name,c.customer_id, category
HAVING COUNT(category) =
(SELECT COUNT (DISTINCT category)
FROM  dbo.product);

--USING CASE STATEMENT TO IDENTIFY

SELECT customer_name,c.customer_id, (CASE
                                  WHEN COUNT (DISTINCT category) = 3 THEN 'Purchased All'
                                                            END) AS Category_Purchase,
                                                            COUNT (DISTINCT category)as
Category_count
FROM dbo.customer c
JOIN dbo.sales s
ON c.customer_id = s.customer_id
JOIN dbo.product p
ON p.product_id= s.product_id
GROUP BY customer_name,c.customer_id;



--Write a SQL query to calculate the rolling average profit for each customer over the
past 3 orders.

WITH Sales1 as
              ( SELECT customer_id, order_date, profit, ROW_NUMBER() OVER (PARTITION BY
customer_id ORDER BY order_date
                DESC)as row_num
                FROM dbo.sales)

SELECT  customer_id, order_date, profit,
AVG(profit) OVER (PARTITION BY customer_id ORDER BY order_date  ROWS BETWEEN 2 PRECEDING
 AND CURRENT ROW) as rn
FROM Sales1
WHERE row_num <=3;
```

```
--selecting the row number is to group by  the customer id  and order it by the order
date from the past orders,
--so for each customer id, it creates a third column (row number) to order it from the
first day off sales to the last
--so whenyouu calculate the avg it can be able to make use of the row number for the
calculation for current rows and two
--preceding



--Write a SQL query to find the customers who have made purchases in at least 3 different
states.

select c.customer_id, count( c.state) CountState
FROM dbo.sales s
JOIN dbo.customer c
ON c.customer_id = s.customer_id
GROUP BY c.customer_id
HAVING COUNT ( DISTINCT c.state) >= 3


--Write a SQL query to calculate the year-over-year growth in sales for each category.
FIRST get the sum of the sales

WITH YearlySales AS
                                (SELECT category, SUM(sales) as metrics , DATEPART(year,
order_date) as SalesYear
                                   FROM dbo.sales  s
                                   JOIN dbo.product p
                                   ON s.product_id= p.product_id
                                   GROUP BY category, DATEPART(year, order_date))

SELECT a.category, a.SalesYear, CAST (((a.metrics - b.metrics) / b.metrics )*100 AS
INT)as YoYGrowth
FROM YearlySales a
LEFT JOIN YearlySales b
ON a.category = b.category AND  a.SalesYear=b.SalesYear +1

--USING THE  formula for year by year growth, value of currwnt year - value of previous
year / value of previous year
-- multiply by 100, first find the metric you want to use either sum of sales, avg of
profit or count of quantity
--, then thats your value. join the table by itself where category of a is same as b, and
year a = year b


--Write a SQL query to find the customers who have placed orders on consecutive days.

SELECT DISTINCT o1.customer_id, customer_name
FROM dbo.sales o1
JOIN dbo.customer c
ON o1.customer_id = c.customer_id
JOIN dbo.sales o2 on o1.customer_id=o2.customer_id
AND o2.order_date = DATEADD(day, 1 , o1.order_date);

--OR
WITH Consec_day AS
(SELECT c.customer_id, c.customer_name, order_date,
LAG(order_date) OVER(PARTITION BY c.customer_id ORDER BY order_date) as prev_day
FROM dbo.sales s
JOIN dbo.customer c
ON s.customer_id= c.customer_id)

SELECT customer_id, customer_name
FROM Consec_day
WHERE DATEDIFF(day, prev_day, order_date)=1



--LAG is used to get the previous date like a date and the next date of order  for each
custoer, so when w,
--after we sekect it
--we also need to check  if the date diff between the two dates is equal to 1, like 1 jan
and 2n Jan , then it s one, but
--1st JAN AND 3rd JAN is 2 days so its not consecutive


--Write a SQL query to find the products that have never been ordered.
SELECT  p.product_name,            p.product_id , order_id
       FROM dbo.sales s
       RIGHT JOIN  dbo.product p
       ON  p.product_id = s.product_id
       WHERE order_id  IS NULL



--Write a SQL query to calculate the average profit margin for each sub-category
SELECT AVG(CASE WHEN sales <> 0 THEN profit/ sales ELSE 0 END) * 100 as AvgProfit_Margin
       FROM dbo.sales

--the profit margin is profit divided by sales times 100, using the when case statement
gurrantees that NO ZERO



--Write a SQL query to find the customers who have placed orders on all weekdays

SELECT customer_id, product_id
       FROM dbo.sales
       WHERE DATEPART(dw, order_date) BETWEEN 2 AND 6



--Write a SQL query to find the customers who have placed the highest number of orders in
each region.

WITH sub  AS
                    (SELECT DISTINCT c.customer_id, region, COUNT(order_id)CountOrder,
                           ROW_NUMBER() OVER (PARTITION BY region ORDER BY
COUNT(order_id) DESC) as rm
                           FROM dbo.sales s
                           JOIN dbo.customer c on  s.customer_id = c.customer_id
                           GROUP BY c.customer_id, region)

SELECT customer_id, region, CountOrder
FROM sub
```

```
WHERE rm= 1



--OR

WITH sub1 AS
                     (SELECT c.customer_id, region, COUNT(order_id)count_order
                     FROM dbo.sales s
                     JOIN dbo.customer c on  s.customer_id = c.customer_id
                     GROUP BY c.customer_id, region),

sub2 AS
                     (SELECT region , MAX(count_order)MAX_order
                      FROM sub1
                      GROUP  BY region)

SELECT sub1.region, sub1.customer_id
       FROM sub1
       JOIN sub2 ON sub2.region= sub1.region
       AND sub1.count_order=sub2.MAX_order;

--Write a SQL query to find the products that were ordered more than once on the same
day.


SELECT product_id, DATETRUNC(day, order_date) as day,COUNT(product_id)product_count
       FROM dbo.sales
       GROUP BY product_id,DATETRUNC(day, order_date)
       HAVING COUNT(product_id) >1;



--Write a SQL query to calculate the total profit for each quarter.
SELECT SUM(profit)Total_Profit , DATEPART(QUARTER, order_date)as Quarter
       FROM dbo.sales
       GROUP BY DATEPART(QUARTER, order_date)
       ORDER BY 2 ASC;


--Write a SQL query to find the customers who have never placed an order in the
'Technology' category.

SELECT c.customer_id, customer_name,category,order_id
       FROM dbo.customer c
       LEFT JOIN dbo.sales s
       ON s.customer_id     = c.customer_id
       LEFT JOIN dbo.product p
       ON  p.product_id = s.product_id
       WHERE category NOT IN
                                          (
                                          SELECT category
                                          FROM dbo.product
                                          WHERE category = 'Technology')
```

```sql
--Write a SQL query to find the average age of customers in each segment.

SELECT segment, AVG(age) Avg_Age
       FROM dbo.customer
       GROUP BY segment;


--Write a SQL query to find the products that have had a decrease in sales compared to
the previous year.


WITH yearly_sales AS
                                   (SELECT product_id, DATEPART(YEAR, order_date) AS YEAR,
SUM(sales) AS Total_Sales
                                   FROM dbo.sales
                                   GROUP BY product_id,  DATEPART(YEAR, order_date))

SELECT c.product_id, c.YEAR, c.Total_Sales, p.YEAR as previous_year , (c.Total_Sales-
p.Total_Sales) as Sales_diff
FROM yearly_sales as c
LEFT JOIN yearly_sales as p
ON c.product_id = p.product_id AND c.YEAR =p.YEAR +1
WHERE c.Total_Sales < p.Total_Sales;

--first sum the sales so you can aggregate it with per product id and yearr, then ttry to
find to find a situation where
--current sale is lower than previous sales and that we use left join to join he table
at additional year per product id,
--so it doesnt repeat.



--Write a SQL query to calculate the cumulative profit for each customer, ordered by
their total profit in descending order.


SELECT c.customer_id, customer_name, SUM(profit) as Cumulative_Profit
       FROM dbo.customer c
       JOIN  dbo.sales  s
       ON c.customer_id = s.customer_id
       GROUP BY c.customer_id, customer_name
       ORDER BY 3 DESC;


--WE can see that over 100 customers are on loss up to -6625


--Write a SQL query to identify the products with the highest and lowest profit margins
within each category.

WITH p_margin AS

              (SELECT DISTINCT p.product_id, category, profit, s.sales,
                                   (CASE  WHEN sales <> 0 THEN profit / sales ELSE 0
END)*100 AS profit_margin,
                                   RANK () OVER (PARTITION BY category ORDER BY
                                                                                  CASE
WHEN sales <> 0 THEN (profit / NULLIF(s.sales, 0)) * 100

ELSE 0
                                                                                                    END
DESC) AS highest_profit_margin,
                                    RANK () OVER (PARTITION BY category ORDER BY
                                                                                                    CASE

WHEN sales <> 0 THEN (profit / NULLIF(s.sales, 0)) * 100

ELSE 0


       END ASC) AS lowest_profit_margin
       FROM dbo.sales s
       JOIN dbo.product p
       ON s.product_id= p.product_id)

SELECT category,
       MAX(CASE WHEN lowest_profit_margin =1 THEN product_id END) AS Lowest_p_margin,
       MAX (CASE WHEN  highest_profit_margin = 1 THEN product_id END) AS highest_p_margin
       FROM p_margin
       GROUP BY category

;

--rank was used to rank the product with the order of their profit margin which is in
desc and asc, after
-- which you will then use an if statement to find when highest margin goes with the
codition then you get the product if


--Write a SQL query to calculate the median order value for each sub-category.

SELECT sub_category ,
CAST( AVG(sales) AS INT) AS Median_Order
FROM
                              (SELECT sales, sub_category,
                              ROW_NUMBER() OVER (PARTITION BY sub_category ORDER BY sales)
as row_num,
                              COUNT(*) OVER (PARTITION BY sub_category) AS TotalSales
                              FROM dbo.sales s
                              JOIN dbo.product p
                              ON s.product_id = p.product_id
                              ) AS inner_sub

WHERE row_num IN ((TotalSales +1 )/2, (TotalSales +2)/2)
GROUP BY sub_category;


--Write a SQL query to categorize customers based on their total purchase amount into
three groups:
--'High Spenders', 'Medium Spenders', and 'Low Spenders'. Consider the following
criteria:
--'High Spenders' have a total purchase amount greater than $1000.
--'Medium Spenders' have a total purchase amount between $500 and $1000 (inclusive).
--'Low Spenders' have a total purchase amount less than $500.
```

```
SELECT  c.customer_id, customer_name, SUM(sales* quantity) as Total_Purchase_Amount,
                                                              (CASE
                                  WHEN SUM( sales* quantity) > 1000 THEN 'High Spenders'
              WHEN SUM(sales * quantity) BETWEEN 500 AND 1000 THEN 'Medium Spenders'
                                                              ELSE 'Low Spenders'
                                                              END) AS Customer_Category
       FROM dbo.customer c
       JOIN dbo.sales s
       ON c.customer_id = s.customer_id
       GROUP BY c.customer_id, customer_name
       ORDER BY 4;



--Write a SQL query to analyze customer purchasing behavior over time. Categorize
customers into
--'Steady Buyers', 'Occasional Buyers', and 'One-Time Buyers' based on the following
criteria:
--'Steady Buyers' are customers who have placed orders in at least three different
months.
--'Occasional Buyers' are customers who have placed orders in two different months.
--'One-Time Buyers' are customers who have placed orders in only one month.

SELECT  c.customer_id, customer_name,
                                                       (CASE
       WHEN COUNT (DISTINCT DATEPART(MONTH, order_date)) >= 3 THEN 'Steady Buyers'
     WHEN COUNT (DISTINCT DATEPART(MONTH, order_date)) = 2 THEN 'Occasional Buyers'
       WHEN COUNT (DISTINCT DATEPART(MONTH, order_date)) = 1 THEN 'One Buyers'

END ) AS Purchase_Behavior, COUNT (DISTINCT DATEPART(MONTH, order_date)) AS Month_Count


       FROM  dbo.customer c
       JOIN dbo.sales s
       ON s.customer_id = c.customer_id
       GROUP BY  c.customer_id, customer_name
```