SQL Challenge - Solving Scenario Based Questions - Part 1

Prepared by - Vishnu T R

Question 1

(a) Write a SQL to get all the products that got sold on both the days and the number of times the product is sold.

(b) Write a SQL to get products that was ordered on 02-May-2099 but not on 01-May-2099

Order_Date	Order_Id	Product_Id	Quantity	Price
01-05-2099	ODR1	PROD1	5	5
01-05-2099	ODR2	PROD2	2	10
01-05-2099	ODR3	PROD3	10	25
01-05-2099	ODR4	PROD1	20	5
02-05-2099	ODR5	PROD3	5	25
02-05-2099	ODR6	PROD4	6	20
02-05-2099	ODR7	PROD1	2	5
02-05-2099	ODR8	PROD5	1	50
02-05-2099	ODR9	PROD6	2	50
02-05-2099	ODR10	PROD2	4	10

Solution(a) - Method 1

```
SELECT product_id,
       Count(*)       AS product_sold_count,
       Sum(quantity) AS Total_quantity_sold
FROM   orders
GROUP  BY product_id
HAVING Count(DISTINCT order_date) >= 2
```

Solution(a) - Method 2

```
SELECT product_id,
       Count(*)       AS product_sold_count,
       Sum(quantity) AS Total_quantity_sold
FROM   orders
WHERE  product_id IN (SELECT product_id
                      FROM   (SELECT product_id,
                                     order_date
                              FROM   orders
                              GROUP  BY product_id,
                                        order_date) a
                      GROUP  BY product_id
                      HAVING Count(order_date) >= 2)
GROUP  BY product_id
```

Solution(a) - Method 3

```
WITH dist_prod_date
     AS (SELECT product_id,
                order_date
         FROM   orders
         GROUP  BY product_id,
                   order_date),
     choose_prod
     AS (SELECT product_id,
                Count(order_date) AS sold_days
         FROM   dist_prod_date
         GROUP  BY product_id
         HAVING Count(order_date) > = 2)
SELECT product_id,
       Count(*)      AS product_sold_count,
       Sum(quantity) AS Total_quantity_sold
FROM   orders
WHERE  product_id IN (SELECT product_id
                      FROM   choose_prod)
GROUP  BY product_id
```

Result(a)

Product_Id	**Product_sold_count**	**Total_quantity_sold**
PROD1	3	27
PROD2	2	6
PROD3	2	15

Solution(b)

```
SELECT DISTINCT product_id
FROM   orders
WHERE  product_id NOT IN (SELECT DISTINCT product_id
                          FROM   orders
                          WHERE  order_date = '2099-05-01')
       AND order_date = '2099-05-02'
```

Result(b)

Product_Id
PROD4
PROD5
PROD6

Question 2

Write a SQL which will explode the above data into single unit level records.

order_id	product_id	quantity
ODR1	PRD1	5
ODR2	PRD2	1
ODR3	PRD3	3

Solution

```sql
WITH cte
     AS (SELECT order_id,
                product_id,
                quantity,
                1 AS new_quantity
         FROM   order_tab
         UNION ALL
         SELECT c.order_id,
                c.product_id,
                ( c.quantity - 1 ) AS quantity,
                new_quantity
         FROM   cte c
         WHERE  c.quantity > 1)
SELECT order_id,
       product_id,
       new_quantity AS quantity

FROM   cte
ORDER  BY order_id,
          product_id
```

Result

order_id	product_id	quantity
ODR1	PRD1	1
ODR1	PRD1	1
ODR1	PRD1	1
ODR1	PRD1	1
ODR1	PRD1	1
ODR2	PRD2	1
ODR3	PRD3	1
ODR3	PRD3	1
ODR3	PRD3	1

Question 3

Write a SQL to find all Employees who earn more than the average salary in their corresponding department.

emp_id	emp_name	salary	dept_id
1001	Mark	60000	2
1002	Antony	40000	2
1003	Andrew	15000	1
1004	Peter	35000	1
1005	John	55000	1
1006	Albert	25000	3
1007	Donald	35000	3

Solution - Method 1

```
SELECT *
FROM   employee e
WHERE  salary > (SELECT Avg(salary)
                 FROM   employee f
                 WHERE  f.dept_id = e.dept_id
                 GROUP  BY dept_id)
```

Solution - Method 2

```
WITH avg_dept_salary
     AS (SELECT dept_id,
                Avg(salary) AS avg_salary
         FROM   employee
         GROUP  BY dept_id)
SELECT e.emp_id,
       e.emp_name,
       e.salary,
       e.dept_id
FROM   employee e
       LEFT JOIN avg_dept_salary aavg
              ON e.dept_id = aavg.dept_id
WHERE  e.salary > aavg.avg_salary
```

Result

emp_id	emp_name	salary	dept_id
1005	John	55000	1
1001	Mark	60000	2
1007	Donald	35000	3

Question 4

Write SQL to get the most recent / latest balance, and TransactionID for each AccountNumber.

AccountNumber	TransactionTime	TransactionID	Balance
550	12-05-2099 05:29:44'	1001	2000
550	15-05-2099 10:29:26'	1002	8000
460	15-03-2099 11:29:24'	1003	9000
460	30-04-2099 11:29:57'	1004	7000
460	30-04-2099 12:32:44'	1005	5000
640	18-02-2099 06:29:34'	1006	5000
640	18-02-2099 06:29:37'	1007	9000

Solution - Method 1

```
WITH latest_trans
     AS (SELECT accountnumber,
                Max(transactiontime) AS Latest_Trans_Time
         FROM   transaction_table
         GROUP  BY accountnumber)
SELECT lt.accountnumber,
       lt.latest_trans_time AS Transaction_Time,
       tt.balance           AS Account_Balance,
       tt.transactionid
FROM   transaction_table tt
       RIGHT JOIN latest_trans lt
               ON tt.accountnumber = lt.accountnumber
                  AND tt.transactiontime = lt.latest_trans_time
ORDER  BY tt.transactionid ASC
```

Solution - Method 2

```
WITH ranked_latest_trans
     AS (SELECT *,
                ROW_NUMBER()
                  OVER(
                    partition BY accountnumber
                    ORDER BY transactiontime DESC) AS Latest_Flag
         FROM   transaction_table)
SELECT rlt.accountnumber,
       rlt.transactiontime,
       rlt.balance AS Account_Balance,
       rlt.transactionid
FROM   ranked_latest_trans rlt
WHERE  latest_flag = 1
ORDER  BY rlt.transactionid ASC
```

Solution - Method 3

```sql
WITH latest_trans_time
     AS (SELECT *,
                Max(transactiontime)
                  OVER(
                    partition BY accountnumber) AS Latest_Trans_Time
         FROM   transaction_table)
SELECT accountnumber,
       transactionid,
       balance           AS Account_Balance,
       latest_trans_time AS Transaction_Time
FROM   latest_trans_time
WHERE  latest_trans_time = transactiontime
ORDER  BY transactionid
```

Result

AccountNumber	TransactionID	Account_Balance	Transaction_Time
550	1002	8000	15-05-2099 10:29:26'
460	1005	5000	30-04-2099 12:32:44'
640	1007	9000	18-02-2099 06:29:37'

Question 5

Write an SQL query to generate a pivot table displaying the total sales for all products in the years 2097, 2098, and 2099.

id	product	sales_year	quantity_sold
1	Laptop	2097	2500
2	Laptop	2098	3600
3	Laptop	2099	4200
4	Keyboard	2097	2300
5	Keyboard	2098	4800
6	Keyboard	2099	5000
7	Mouse	2097	6000
8	Mouse	2098	3400
9	Mouse	2099	4600

Solution

```
SELECT *
FROM   (SELECT product,
               sales_year,
               quantity_sold
        FROM   sales) base_table
       PIVOT( Sum(quantity_sold)
            FOR sales_year IN ([2097],
                               [2098],
                               [2099]) ) AS result
```

Result

Product	2097	2098	2099
Keyboard	2300	4800	5000
Laptop	2500	3600	4200
Mouse	6000	3400	4600