SQL Exercise 4

1. **Display the minimum Status in the Supplier table.**

```
mysql> SELECT MIN(status) AS MinStatus
    -> FROM s;
+-----------+
| MinStatus |
+-----------+
|        10 |
+-----------+
1 row in set (0.00 sec)
```

2. **Display the maximum Weight in the Parts table.**

```
mysql> SELECT MAX(weight) AS MaxWeight
    -> FROM p;
+-----------+
| MaxWeight |
+-----------+
|        14 |
+-----------+
1 row in set (0.00 sec)
```

3. **Display the average Weight of the Parts.**

```
mysql> SELECT AVG(weight) AS AvgWeight
    -> FROM p;
+-----------+
| AvgWeight |
+-----------+
|     12.25 |
+-----------+
1 row in set (0.00 sec)
```

4. **Display the total Quantity sold for part ‘P1’.**

```
mysql> select sum(CAST(QTY AS UNSIGNED)) AS Total_Quantity_p1
    -> FROM spj
    -> WHERE 'P#' = 'P1';
+-------------------+
| Total_Quantity_p1 |
+-------------------+
|              NULL |
+-------------------+
1 row in set (0.01 sec)
```

5. **Display the total Quantity sold for each part.**

```
mysql> SELECT 'P#' AS Part_Number,
    -> SUM(CAST(QTY AS UNSIGNED)) AS Total_Quantity
    -> FROM spj
    -> GROUP BY 'P#';
+-------------+----------------+
| Part_Number | Total_Quantity |
+-------------+----------------+
| P#          |           1050 |
+-------------+----------------+
1 row in set (0.00 sec)
```

6. **Display the average Quantity sold for each part.**

```
mysql> SELECT `P#`, AVG(CAST(QTY AS DECIMAL)) AS Average_Quantity_Sold
    -> FROM spj
    ->  GROUP BY `P#`;
+------+----------------------+
| P#   | Average_Quantity_Sold |
+------+----------------------+
| P1   |             300.0000 |
| P2   |             200.0000 |
| P3   |             150.0000 |
| P4   |             400.0000 |
+------+----------------------+
4 rows in set (0.01 sec)
```

7. **Display the maximum Quantity sold for each part, provided the maximum Quantity is greater than 800.**

```
mysql> SELECT `P#`, MAX(CAST(`QTY` AS DECIMAL)) AS Max_Quantity_Sold
    -> FROM spj
    ->  GROUP BY `P#`
    -> HAVING MAX(CAST(`QTY` AS DECIMAL)) > 800;
Empty set (0.00 sec)
```

8. **Display the Status and the count of Suppliers with that Status.**

```
mysql> SELECT `STATUS`, COUNT(*) AS Supplier_Count
    -> FROM s
    ->  GROUP BY `STATUS`;
+--------+----------------+
| STATUS | Supplier_Count |
+--------+----------------+
|     20 |              2 |
|     10 |              2 |
|     30 |              1 |
+--------+----------------+
3 rows in set (0.00 sec)
```

9. **Display the count of Projects going on in different cities.**

```
mysql> SELECT CITY, COUNT(*) AS Project_Count
    ->  FROM j
    -> GROUP BY CITY;
+--------+---------------+
| CITY   | Project_Count |
+--------+---------------+
| London |             1 |
| Paris  |             1 |
| Athens |             1 |
| Rome   |             1 |
+--------+---------------+
4 rows in set (0.02 sec)
```

10. What is the difference between COUNT(Status) and COUNT(*) ?

COUNT(*)

- Counts all rows in the table, regardless of NULL values.
- It includes every row — even if some columns are NULL.

COUNT(Status)

- Counts only non-NULL values in the Status column.
- It ignores rows where Status is NULL**.**

11. Display the Status and the Count of Suppliers with that Status in the following format as shown below:-

Status	Count
Ten	1
Twenty	2
Thirty	3

```
mysql> SELECT
    -> CASE
    -> WHEN STATUS = '10' THEN 'Ten'
    -> WHEN STATUS = '20' THEN 'Twenty'
    -> WHEN STATUS = '30' THEN 'Thirty'
    -> ELSE 'Other'
    -> END AS Status,
    -> COUNT(*) AS Count
    -> FROM s
    -> GROUP BY Status;
+--------+-------+
| Status | Count |
+--------+-------+
| Twenty |     2 |
| Ten    |     2 |
| Thirty |     1 |
+--------+-------+
3 rows in set, 1 warning (0.00 sec)
```