Practical No. 4

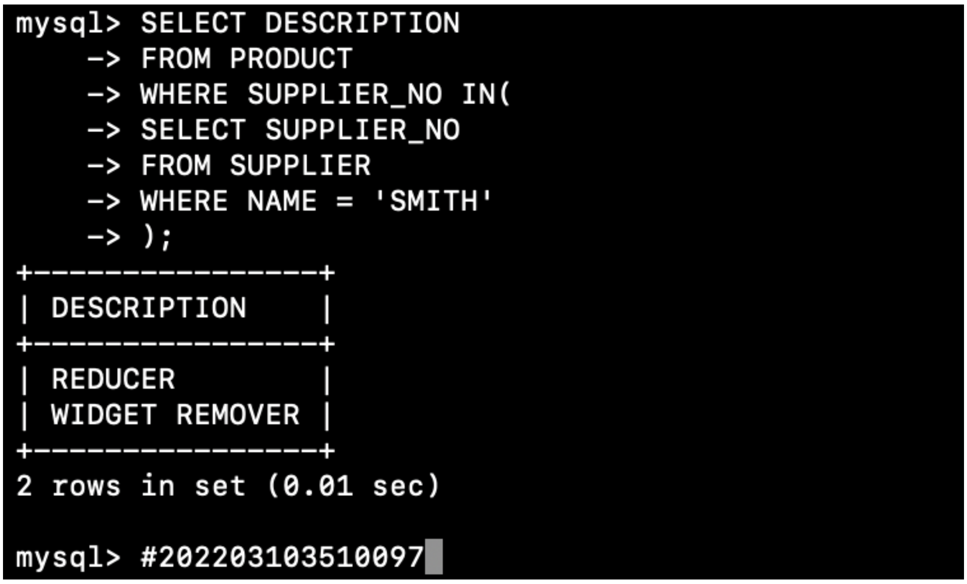
Aim: To implement Integrity Constraints. Queries (along with sub Queries)

Theory:

Integrity constraints are rules or conditions that govern the validity and consistency of data in a database. They are essential for maintaining data quality and ensuring that the database accurately represents the real-world domain it models. While integrity constraints are often defined during the database schema design phase, their enforcement within queries and subqueries is equally vital.

Queries:

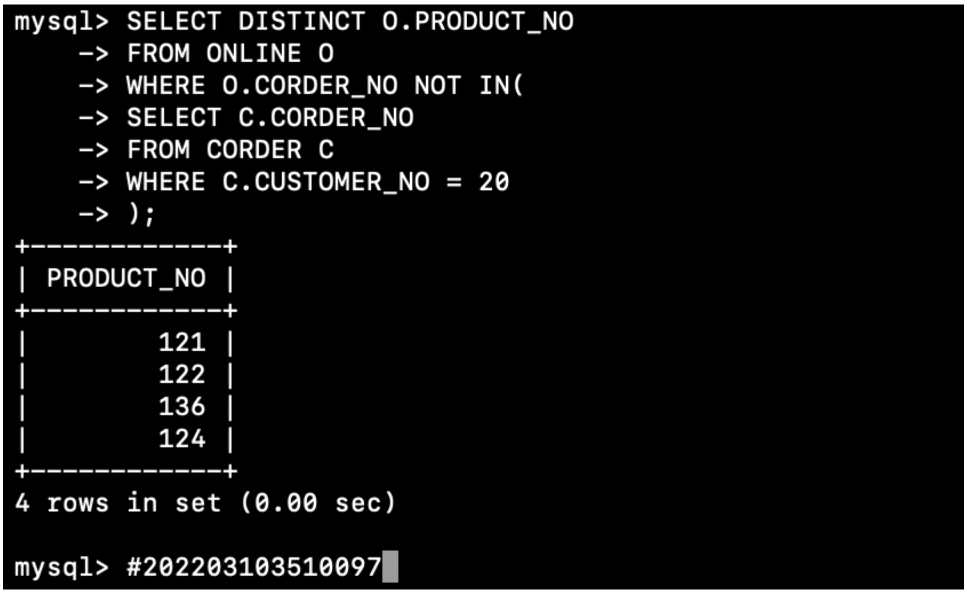
1) List the description of product which are supplied by supplier SMITH using IN.



```
mysql> SELECT DESCRIPTION
    -> FROM PRODUCT
    -> WHERE SUPPLIER_NO IN(
    -> SELECT SUPPLIER_NO
    -> FROM SUPPLIER
    -> WHERE NAME = 'SMITH'
    -> );
+----------------+
| DESCRIPTION    |
+----------------+
| REDUCER        |
| WIDGET REMOVER |
+----------------+
2 rows in set (0.01 sec)

mysql> #202203103510097
```

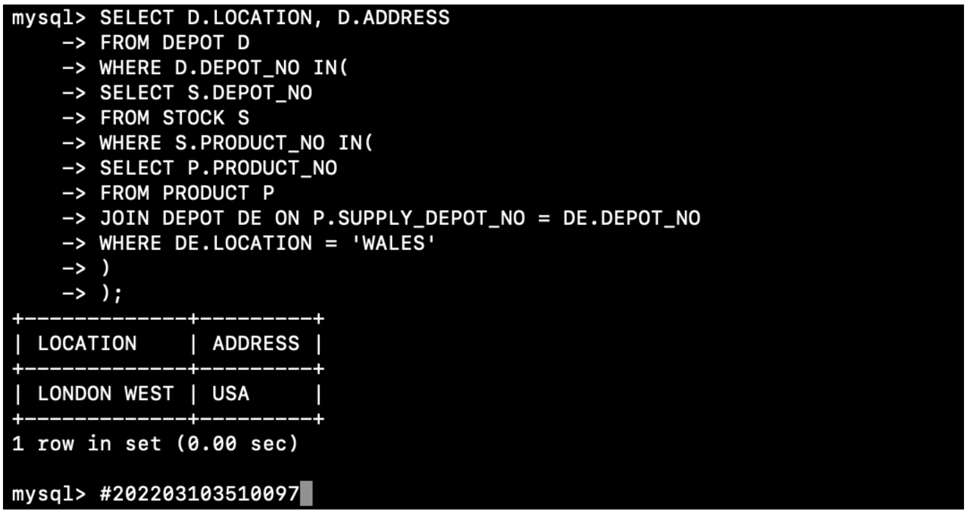
2) List all product no which are not ordered by the customer having same CORDER_NO as the CUSTOMER_NO 20.



```
mysql> SELECT DISTINCT O.PRODUCT_NO
    -> FROM ONLINE O
    -> WHERE O.CORDER_NO NOT IN(
    -> SELECT C.CORDER_NO
    -> FROM CORDER C
    -> WHERE C.CUSTOMER_NO = 20
    -> );
+------------+
| PRODUCT_NO |
+------------+
|        121 |
|        122 |
|        136 |
|        124 |
+------------+
4 rows in set (0.00 sec)

mysql> #202203103510097
```

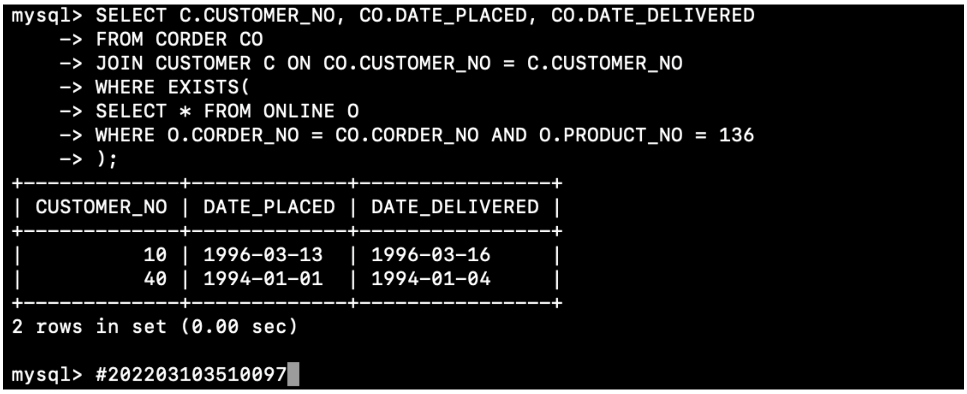
3) List the locations and addresses of all depots which stock any product which is supplied to the depot whose location is wales.



```
mysql> SELECT D.LOCATION, D.ADDRESS
    -> FROM DEPOT D
    -> WHERE D.DEPOT_NO IN(
    -> SELECT S.DEPOT_NO
    -> FROM STOCK S
    -> WHERE S.PRODUCT_NO IN(
    -> SELECT P.PRODUCT_NO
    -> FROM PRODUCT P
    -> JOIN DEPOT DE ON P.SUPPLY_DEPOT_NO = DE.DEPOT_NO
    -> WHERE DE.LOCATION = 'WALES'
    -> )
    -> );
+-------------+---------+
| LOCATION    | ADDRESS |
+-------------+---------+
| LONDON WEST | USA     |
+-------------+---------+
1 row in set (0.00 sec)

mysql> #202203103510097
```

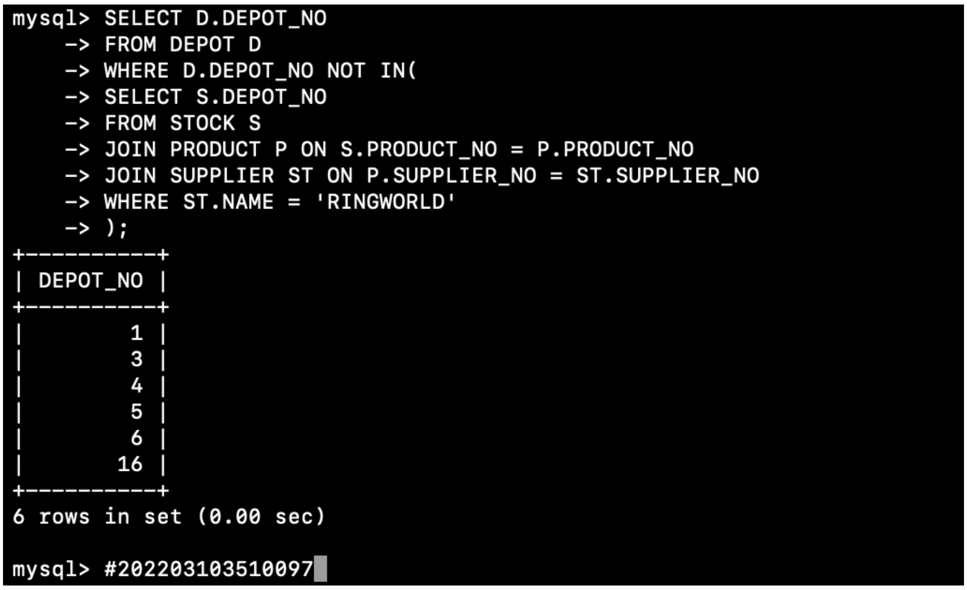
4) List the customer_no, date_placed and date_delivered for all orders which contain order lines for the product with product_no 137 using existential quantification (ie the where exists condition).



```
mysql> SELECT C.CUSTOMER_NO, CO.DATE_PLACED, CO.DATE_DELIVERED
    -> FROM CORDER CO
    -> JOIN CUSTOMER C ON CO.CUSTOMER_NO = C.CUSTOMER_NO
    -> WHERE EXISTS(
    -> SELECT * FROM ONLINE O
    -> WHERE O.CORDER_NO = CO.CORDER_NO AND O.PRODUCT_NO = 136
    -> );
+-------------+-------------+----------------+
| CUSTOMER_NO | DATE_PLACED | DATE_DELIVERED |
+-------------+-------------+----------------+
|          10 | 1996-03-13  | 1996-03-16     |
|          40 | 1994-01-01  | 1994-01-04     |
+-------------+-------------+----------------+
2 rows in set (0.00 sec)

mysql> #202203103510097
```

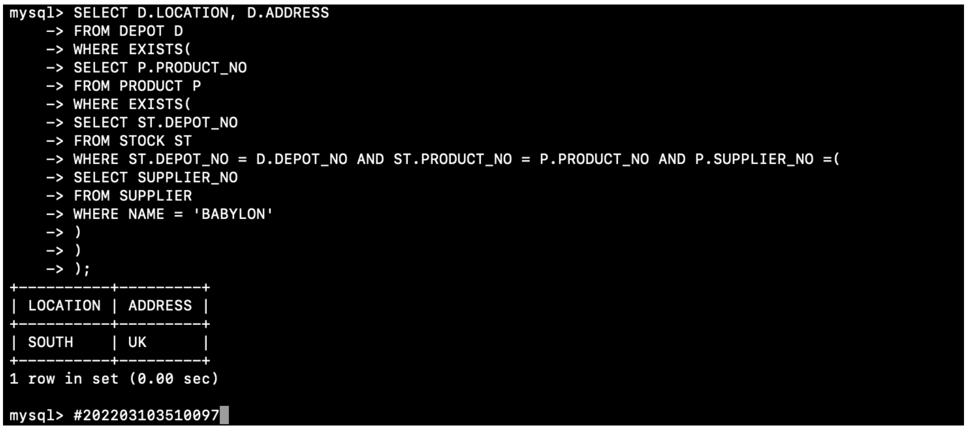
5) List the depots which do not stock any product supplied by the supplier whose name is ringworld.



```
mysql> SELECT D.DEPOT_NO
    -> FROM DEPOT D
    -> WHERE D.DEPOT_NO NOT IN(
    -> SELECT S.DEPOT_NO
    -> FROM STOCK S
    -> JOIN PRODUCT P ON S.PRODUCT_NO = P.PRODUCT_NO
    -> JOIN SUPPLIER ST ON P.SUPPLIER_NO = ST.SUPPLIER_NO
    -> WHERE ST.NAME = 'RINGWORLD'
    -> );
+----------+
| DEPOT_NO |
+----------+
|        1 |
|        3 |
|        4 |
|        5 |
|        6 |
|       16 |
+----------+
6 rows in set (0.00 sec)

mysql> #202203103510097
```

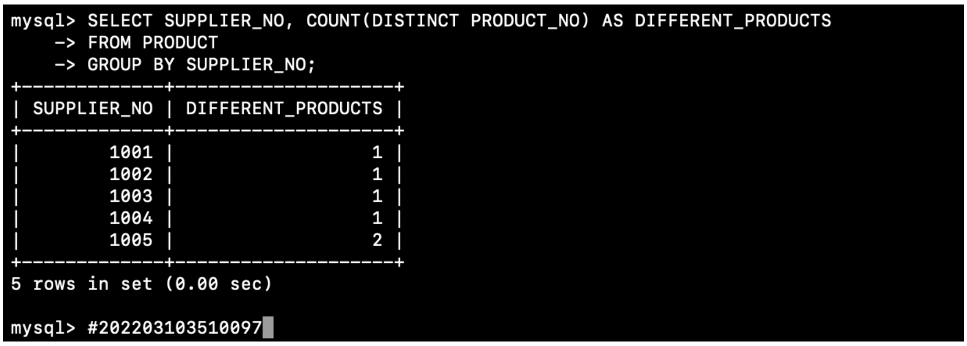
6)List the locations and addresses of all depots which stock all products supplied by the supplier babylon 5.



```
mysql> SELECT D.LOCATION, D.ADDRESS
    -> FROM DEPOT D
    -> WHERE EXISTS(
    -> SELECT P.PRODUCT_NO
    -> FROM PRODUCT P
    -> WHERE EXISTS(
    -> SELECT ST.DEPOT_NO
    -> FROM STOCK ST
    -> WHERE ST.DEPOT_NO = D.DEPOT_NO AND ST.PRODUCT_NO = P.PRODUCT_NO AND P.SUPPLIER_NO =(
    -> SELECT SUPPLIER_NO
    -> FROM SUPPLIER
    -> WHERE NAME = 'BABYLON'
    -> )
    -> )
    -> );
+----------+---------+
| LOCATION | ADDRESS |
+----------+---------+
| SOUTH    | UK      |
+----------+---------+
1 row in set (0.00 sec)

mysql> #202203103510097
```

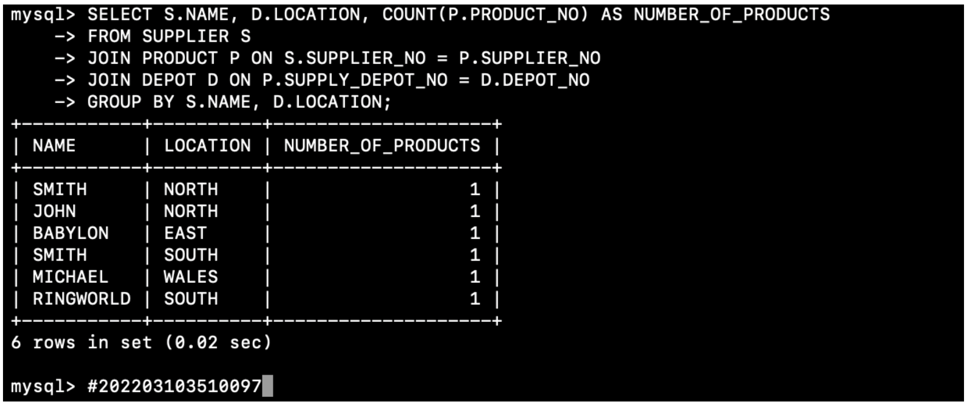
7) List the number of different products supplied by each supplier_no.



```
mysql> SELECT SUPPLIER_NO, COUNT(DISTINCT PRODUCT_NO) AS DIFFERENT_PRODUCTS
    -> FROM PRODUCT
    -> GROUP BY SUPPLIER_NO;
+-------------+--------------------+
| SUPPLIER_NO | DIFFERENT_PRODUCTS |
+-------------+--------------------+
|        1001 |                  1 |
|        1002 |                  1 |
|        1003 |                  1 |
|        1004 |                  1 |
|        1005 |                  2 |
+-------------+--------------------+
5 rows in set (0.00 sec)

mysql> #202203103510097
```

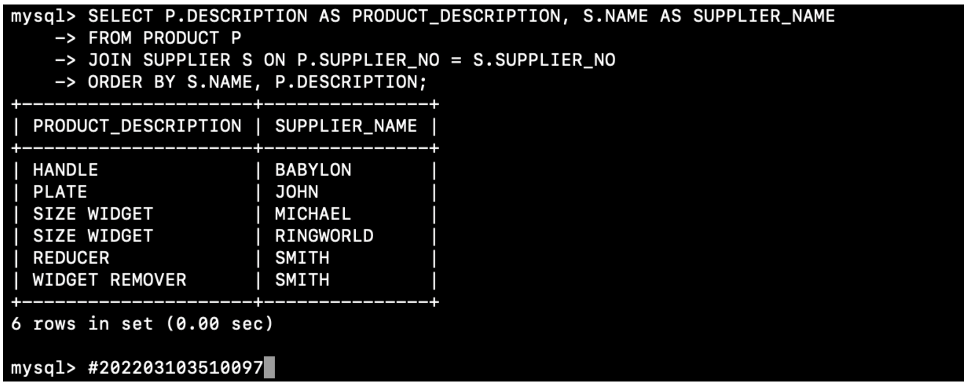
8) List the name of each supplier with the location of each depot and the number of products supplied by that supplier and stocked at that depot.



```
mysql> SELECT S.NAME, D.LOCATION, COUNT(P.PRODUCT_NO) AS NUMBER_OF_PRODUCTS
    -> FROM SUPPLIER S
    -> JOIN PRODUCT P ON S.SUPPLIER_NO = P.SUPPLIER_NO
    -> JOIN DEPOT D ON P.SUPPLY_DEPOT_NO = D.DEPOT_NO
    -> GROUP BY S.NAME, D.LOCATION;
+-----------+----------+--------------------+
| NAME      | LOCATION | NUMBER_OF_PRODUCTS |
+-----------+----------+--------------------+
| SMITH     | NORTH    |                  1 |
| JOHN      | NORTH    |                  1 |
| BABYLON   | EAST     |                  1 |
| SMITH     | SOUTH    |                  1 |
| MICHAEL   | WALES    |                  1 |
| RINGWORLD | SOUTH    |                  1 |
+-----------+----------+--------------------+
6 rows in set (0.02 sec)

mysql> #202203103510097
```

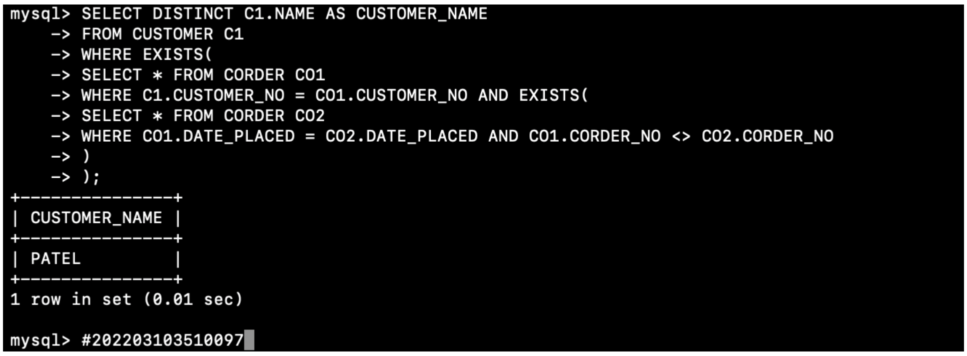
9) List all product descriptions with the product’s supplie name, sorted by product description within supplier name(i.e. all products for a supplier listed together in alphabetic order).



```
mysql> SELECT P.DESCRIPTION AS PRODUCT_DESCRIPTION, S.NAME AS SUPPLIER_NAME
    -> FROM PRODUCT P
    -> JOIN SUPPLIER S ON P.SUPPLIER_NO = S.SUPPLIER_NO
    -> ORDER BY S.NAME, P.DESCRIPTION;
+---------------------+---------------+
| PRODUCT_DESCRIPTION | SUPPLIER_NAME |
+---------------------+---------------+
| HANDLE              | BABYLON       |
| PLATE               | JOHN          |
| SIZE WIDGET         | MICHAEL       |
| SIZE WIDGET         | RINGWORLD     |
| REDUCER             | SMITH         |
| WIDGET REMOVER      | SMITH         |
+---------------------+---------------+
6 rows in set (0.00 sec)

mysql> #202203103510097
```

10) Display customer name who has ordered on same date.



```
mysql> SELECT DISTINCT C1.NAME AS CUSTOMER_NAME
    -> FROM CUSTOMER C1
    -> WHERE EXISTS(
    -> SELECT * FROM CORDER CO1
    -> WHERE C1.CUSTOMER_NO = CO1.CUSTOMER_NO AND EXISTS(
    -> SELECT * FROM CORDER CO2
    -> WHERE CO1.DATE_PLACED = CO2.DATE_PLACED AND CO1.CORDER_NO <> CO2.CORDER_NO
    -> )
    -> );
+---------------+
| CUSTOMER_NAME |
+---------------+
| PATEL         |
+---------------+
1 row in set (0.01 sec)

mysql> #202203103510097
```

Conclusion: Implementing integrity constraints in queries, including subqueries, is a fundamental practice in database management. It ensures that data remains valid, consistent, and aligned with the defined rules and constraints, promoting data quality and the reliability of database operations.