Practical No. 6

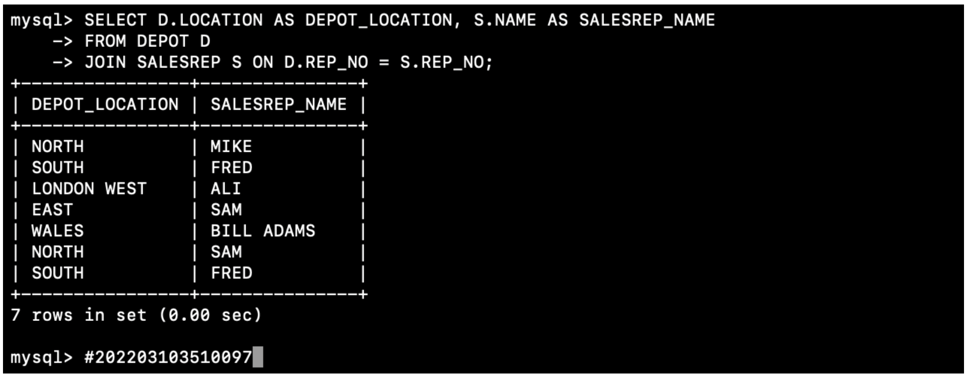
Aim: To implement Joins.

Theory:

Implementing joins in a relational database management system (RDBMS) is a fundamental aspect of query processing and data retrieval. Joins are used to combine data from multiple tables based on a common attribute or set of attributes. There are several theories and techniques to efficiently implement joins in a database system. The primary join types are INNER JOIN, LEFT JOIN (or LEFT OUTER JOIN), RIGHT JOIN (or RIGHT OUTER JOIN), and FULL JOIN (or FULL OUTER JOIN). Each of these joins combines rows from two tables based on a specific condition.

Queries:

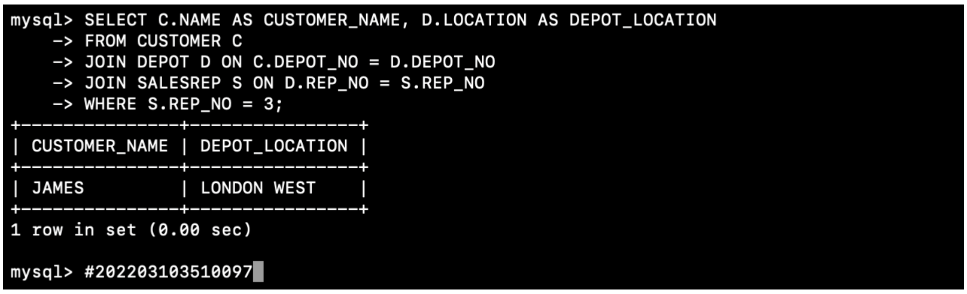
1) Give a list of depot locations paired with the name of the sales rep who covers that depot.



```
mysql> SELECT D.LOCATION AS DEPOT_LOCATION, S.NAME AS SALESREP_NAME
    -> FROM DEPOT D
    -> JOIN SALESREP S ON D.REP_NO = S.REP_NO;
+----------------+---------------+
| DEPOT_LOCATION | SALESREP_NAME |
+----------------+---------------+
| NORTH          | MIKE          |
| SOUTH          | FRED          |
| LONDON WEST    | ALI           |
| EAST           | SAM           |
| WALES          | BILL ADAMS    |
| NORTH          | SAM           |
| SOUTH          | FRED          |
+----------------+---------------+
7 rows in set (0.00 sec)

mysql> #202203103510097
```

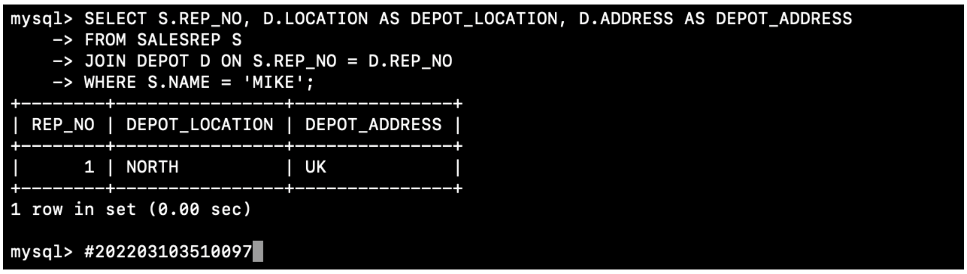
2) List the customer name and the depot location for the depot delivering to that customer for all customers who receive deliveries from depots looked after by sales rep number (rep_no)3.



```
mysql> SELECT C.NAME AS CUSTOMER_NAME, D.LOCATION AS DEPOT_LOCATION
    -> FROM CUSTOMER C
    -> JOIN DEPOT D ON C.DEPOT_NO = D.DEPOT_NO
    -> JOIN SALESREP S ON D.REP_NO = S.REP_NO
    -> WHERE S.REP_NO = 3;
+---------------+----------------+
| CUSTOMER_NAME | DEPOT_LOCATION |
+---------------+----------------+
| JAMES         | LONDON WEST    |
+---------------+----------------+
1 row in set (0.00 sec)

mysql> #202203103510097
```

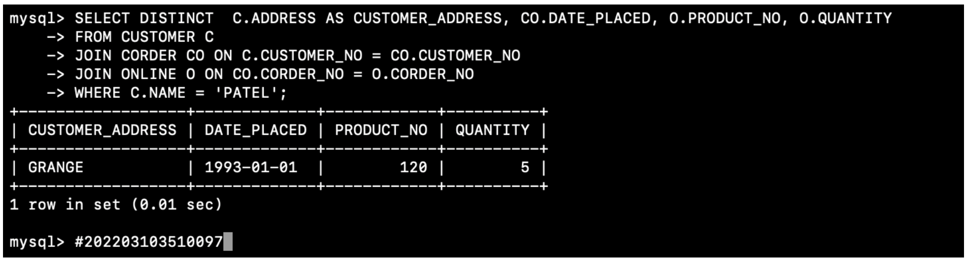
3) List the sales rep number (rep_no) and depot location and address for depots looked after by the sales rep whose name is mike.



```
mysql> SELECT S.REP_NO, D.LOCATION AS DEPOT_LOCATION, D.ADDRESS AS DEPOT_ADDRESS
    -> FROM SALESREP S
    -> JOIN DEPOT D ON S.REP_NO = D.REP_NO
    -> WHERE S.NAME = 'MIKE';
+--------+----------------+---------------+
| REP_NO | DEPOT_LOCATION | DEPOT_ADDRESS |
+--------+----------------+---------------+
|      1 | NORTH          | UK            |
+--------+----------------+---------------+
1 row in set (0.00 sec)

mysql> #202203103510097
```

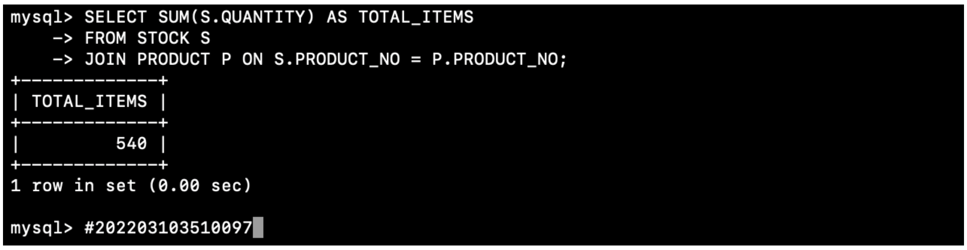
4) For all order lines (oline) for all orders (corder) for customers whose name is patel, list the customer address, the date_placed, the product_no and the quantity.



```
mysql> SELECT DISTINCT  C.ADDRESS AS CUSTOMER_ADDRESS, CO.DATE_PLACED, O.PRODUCT_NO, O.QUANTITY
    -> FROM CUSTOMER C
    -> JOIN CORDER CO ON C.CUSTOMER_NO = CO.CUSTOMER_NO
    -> JOIN ONLINE O ON CO.CORDER_NO = O.CORDER_NO
    -> WHERE C.NAME = 'PATEL';
+------------------+-------------+------------+----------+
| CUSTOMER_ADDRESS | DATE_PLACED | PRODUCT_NO | QUANTITY |
+------------------+-------------+------------+----------+
| GRANGE           | 1993-01-01  |        120 |        5 |
+------------------+-------------+------------+----------+
1 row in set (0.01 sec)

mysql> #202203103510097
```

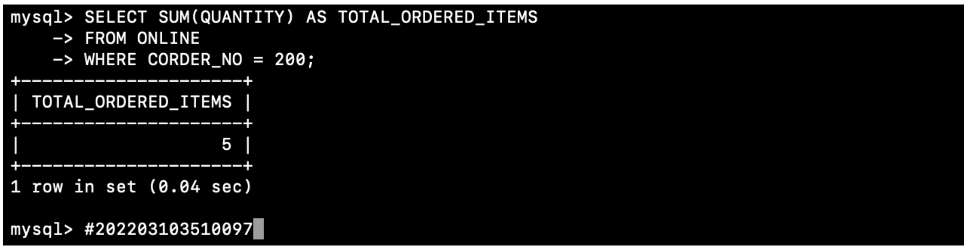
5) Give the total number of items (quantity) in stock in all depots.



```
mysql> SELECT SUM(S.QUANTITY) AS TOTAL_ITEMS
    -> FROM STOCK S
    -> JOIN PRODUCT P ON S.PRODUCT_NO = P.PRODUCT_NO;
+-------------+
| TOTAL_ITEMS |
+-------------+
|         540 |
+-------------+
1 row in set (0.00 sec)

mysql> #202203103510097
```

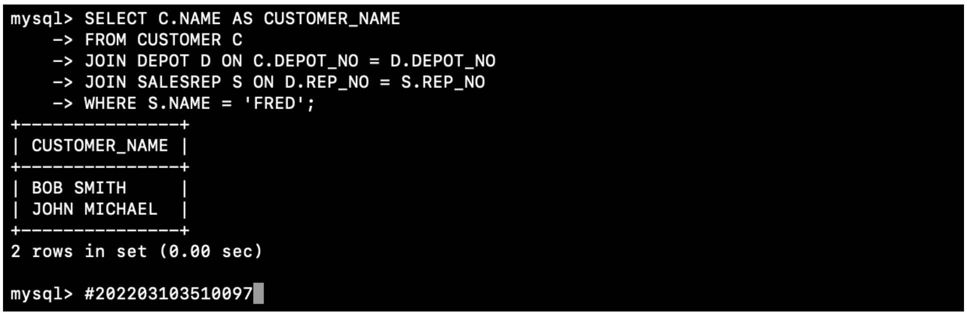
6) Give the total number of items (order line quantity) which have been ordered on the order with corder_no 200.



```
mysql> SELECT SUM(QUANTITY) AS TOTAL_ORDERED_ITEMS
    -> FROM ONLINE
    -> WHERE CORDER_NO = 200;
+---------------------+
| TOTAL_ORDERED_ITEMS |
+---------------------+
|                   5 |
+---------------------+
1 row in set (0.04 sec)

mysql> #202203103510097
```

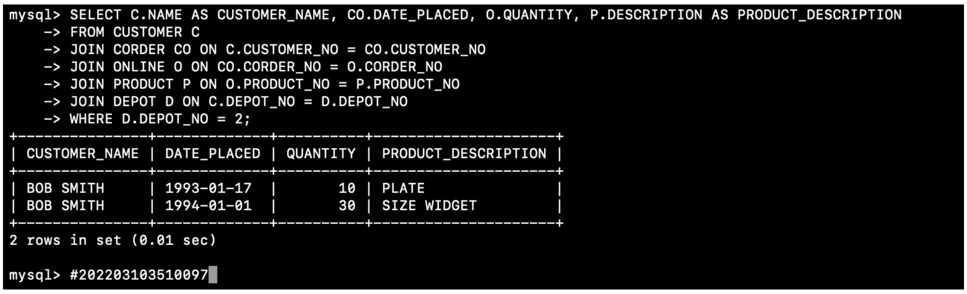
7) List the names of all customers who receive deliveries from depots which are looked after by the sales rep whose name is fred.



```
mysql> SELECT C.NAME AS CUSTOMER_NAME
    -> FROM CUSTOMER C
    -> JOIN DEPOT D ON C.DEPOT_NO = D.DEPOT_NO
    -> JOIN SALESREP S ON D.REP_NO = S.REP_NO
    -> WHERE S.NAME = 'FRED';
+---------------+
| CUSTOMER_NAME |
+---------------+
| BOB SMITH     |
| JOHN MICHAEL  |
+---------------+
2 rows in set (0.00 sec)

mysql> #202203103510097
```

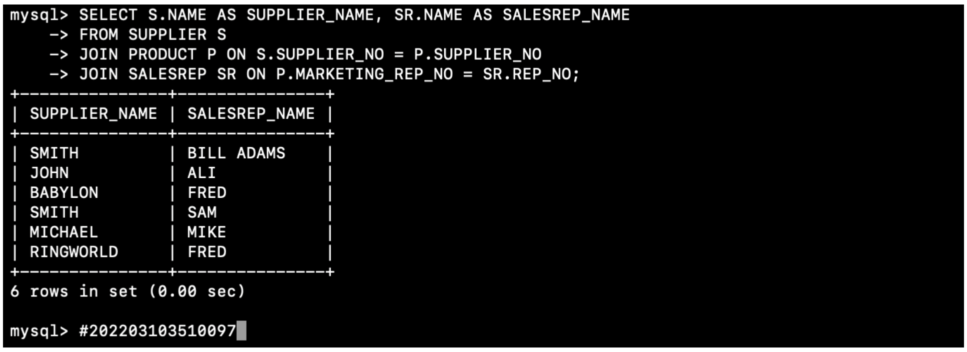
8) List the customer name, order date_placed, order line quantity and product description for each order line (with its linked, order, customer and product rows) for customers who receive deliveries from depot number 2.



```
mysql> SELECT C.NAME AS CUSTOMER_NAME, CO.DATE_PLACED, O.QUANTITY, P.DESCRIPTION AS PRODUCT_DESCRIPTION
    -> FROM CUSTOMER C
    -> JOIN CORDER CO ON C.CUSTOMER_NO = CO.CUSTOMER_NO
    -> JOIN ONLINE O ON CO.CORDER_NO = O.CORDER_NO
    -> JOIN PRODUCT P ON O.PRODUCT_NO = P.PRODUCT_NO
    -> JOIN DEPOT D ON C.DEPOT_NO = D.DEPOT_NO
    -> WHERE D.DEPOT_NO = 2;
+---------------+-------------+----------+---------------------+
| CUSTOMER_NAME | DATE_PLACED | QUANTITY | PRODUCT_DESCRIPTION |
+---------------+-------------+----------+---------------------+
| BOB SMITH     | 1993-01-17  |       10 | PLATE               |
| BOB SMITH     | 1994-01-01  |       30 | SIZE WIDGET         |
+---------------+-------------+----------+---------------------+
2 rows in set (0.01 sec)

mysql> #202203103510097
```

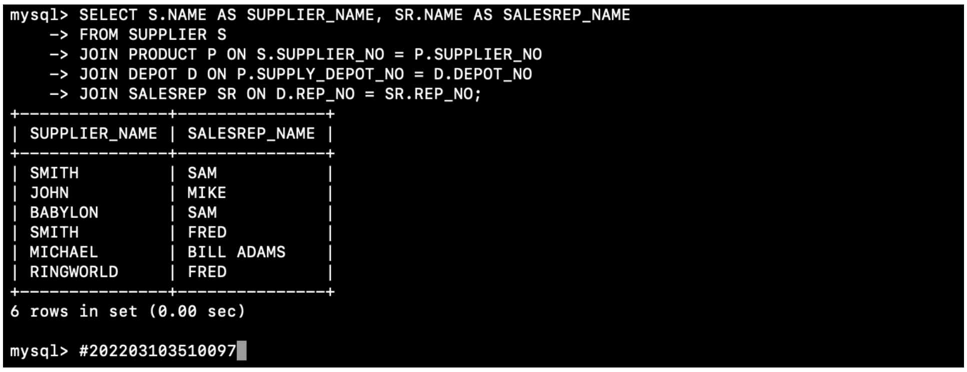
9) List supplier names paired with the names of the sales reps who market products supplied by that supplier.



```
mysql> SELECT S.NAME AS SUPPLIER_NAME, SR.NAME AS SALESREP_NAME
    -> FROM SUPPLIER S
    -> JOIN PRODUCT P ON S.SUPPLIER_NO = P.SUPPLIER_NO
    -> JOIN SALESREP SR ON P.MARKETING_REP_NO = SR.REP_NO;
+---------------+---------------+
| SUPPLIER_NAME | SALESREP_NAME |
+---------------+---------------+
| SMITH         | BILL ADAMS    |
| JOHN          | ALI           |
| BABYLON       | FRED          |
| SMITH         | SAM           |
| MICHAEL       | MIKE          |
| RINGWORLD     | FRED          |
+---------------+---------------+
6 rows in set (0.00 sec)

mysql> #202203103510097
```

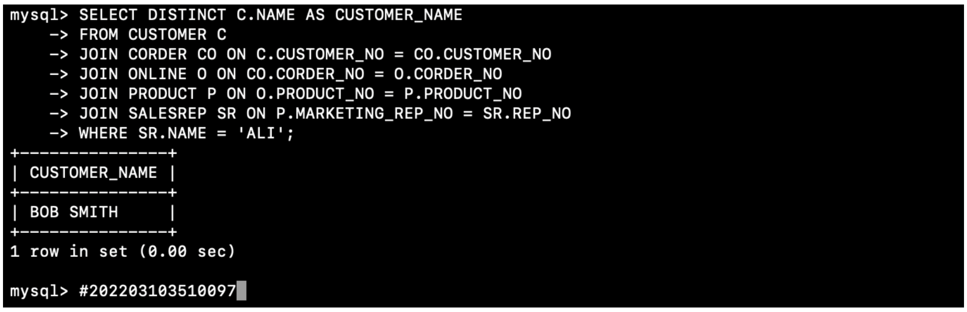
10) List supplier names paired with the names of the sales reps who look after the depots where products from that supplier are delivered.



```
mysql> SELECT S.NAME AS SUPPLIER_NAME, SR.NAME AS SALESREP_NAME
    -> FROM SUPPLIER S
    -> JOIN PRODUCT P ON S.SUPPLIER_NO = P.SUPPLIER_NO
    -> JOIN DEPOT D ON P.SUPPLY_DEPOT_NO = D.DEPOT_NO
    -> JOIN SALESREP SR ON D.REP_NO = SR.REP_NO;
+---------------+---------------+
| SUPPLIER_NAME | SALESREP_NAME |
+---------------+---------------+
| SMITH         | SAM           |
| JOHN          | MIKE          |
| BABYLON       | SAM           |
| SMITH         | FRED          |
| MICHAEL       | BILL ADAMS    |
| RINGWORLD     | FRED          |
+---------------+---------------+
6 rows in set (0.00 sec)

mysql> #202203103510097
```

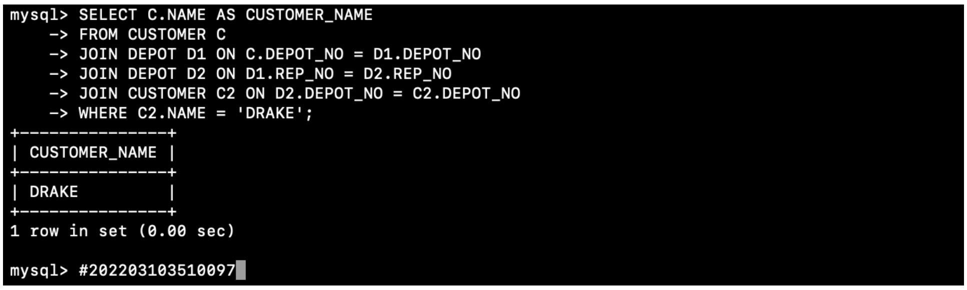
11) List the names of all customers who have ordered products which are marketed by the sales rep whose name is ali.



```
mysql> SELECT DISTINCT C.NAME AS CUSTOMER_NAME
    -> FROM CUSTOMER C
    -> JOIN CORDER CO ON C.CUSTOMER_NO = CO.CUSTOMER_NO
    -> JOIN ONLINE O ON CO.CORDER_NO = O.CORDER_NO
    -> JOIN PRODUCT P ON O.PRODUCT_NO = P.PRODUCT_NO
    -> JOIN SALESREP SR ON P.MARKETING_REP_NO = SR.REP_NO
    -> WHERE SR.NAME = 'ALI';
+---------------+
| CUSTOMER_NAME |
+---------------+
| BOB SMITH     |
+---------------+
1 row in set (0.00 sec)

mysql> #202203103510097
```

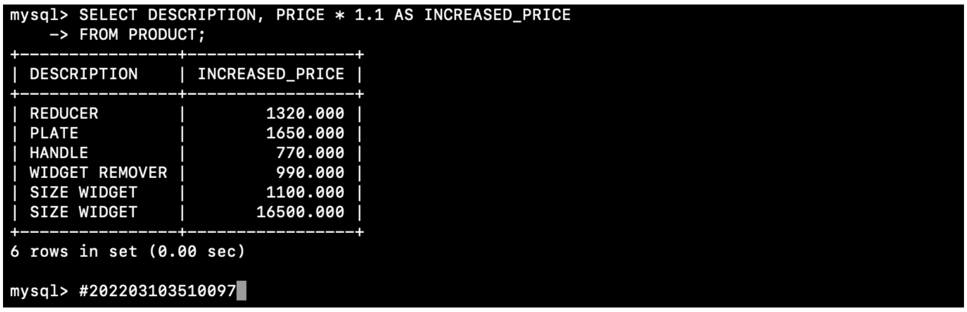
12) List the names of all customers who are delivered to by the depot which delivers to the customer whose name is drake.



```
mysql> SELECT C.NAME AS CUSTOMER_NAME
    -> FROM CUSTOMER C
    -> JOIN DEPOT D1 ON C.DEPOT_NO = D1.DEPOT_NO
    -> JOIN DEPOT D2 ON D1.REP_NO = D2.REP_NO
    -> JOIN CUSTOMER C2 ON D2.DEPOT_NO = C2.DEPOT_NO
    -> WHERE C2.NAME = 'DRAKE';
+---------------+
| CUSTOMER_NAME |
+---------------+
| DRAKE         |
+---------------+
1 row in set (0.00 sec)

mysql> #202203103510097
```

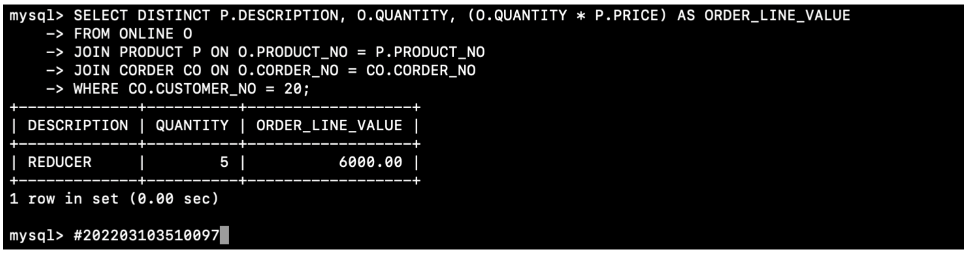
13) List each product description and its price increased by 10%.



```
mysql> SELECT DESCRIPTION, PRICE * 1.1 AS INCREASED_PRICE
    -> FROM PRODUCT;
+----------------+-----------------+
| DESCRIPTION    | INCREASED_PRICE |
+----------------+-----------------+
| REDUCER        |        1320.000 |
| PLATE          |        1650.000 |
| HANDLE         |         770.000 |
| WIDGET REMOVER |         990.000 |
| SIZE WIDGET    |        1100.000 |
| SIZE WIDGET    |       16500.000 |
+----------------+-----------------+
6 rows in set (0.00 sec)

mysql> #202203103510097
```

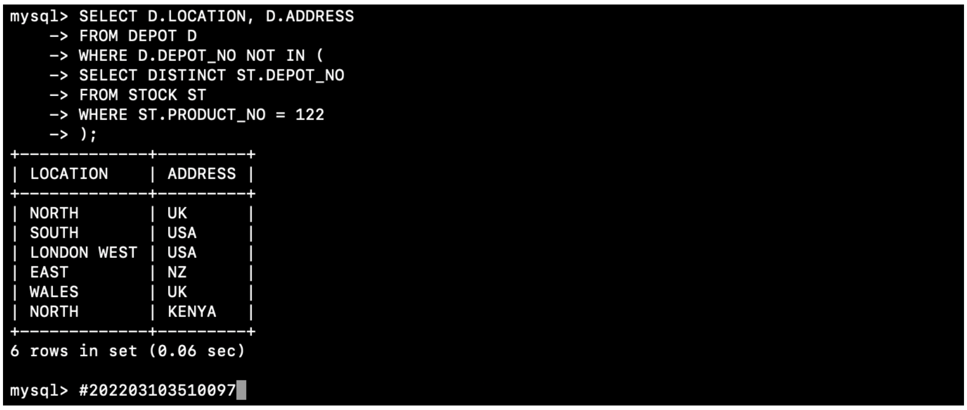
14) List all order lines for the customer with customer_no 20 giving the product description, the order line quantity and the value of the order line. (i.e. the order line quantity * the price from the linked product row)



```
mysql> SELECT DISTINCT P.DESCRIPTION, O.QUANTITY, (O.QUANTITY * P.PRICE) AS ORDER_LINE_VALUE
    -> FROM ONLINE O
    -> JOIN PRODUCT P ON O.PRODUCT_NO = P.PRODUCT_NO
    -> JOIN CORDER CO ON O.CORDER_NO = CO.CORDER_NO
    -> WHERE CO.CUSTOMER_NO = 20;
+-------------+----------+------------------+
| DESCRIPTION | QUANTITY | ORDER_LINE_VALUE |
+-------------+----------+------------------+
| REDUCER     |        5 |          6000.00 |
+-------------+----------+------------------+
1 row in set (0.00 sec)

mysql> #202203103510097
```

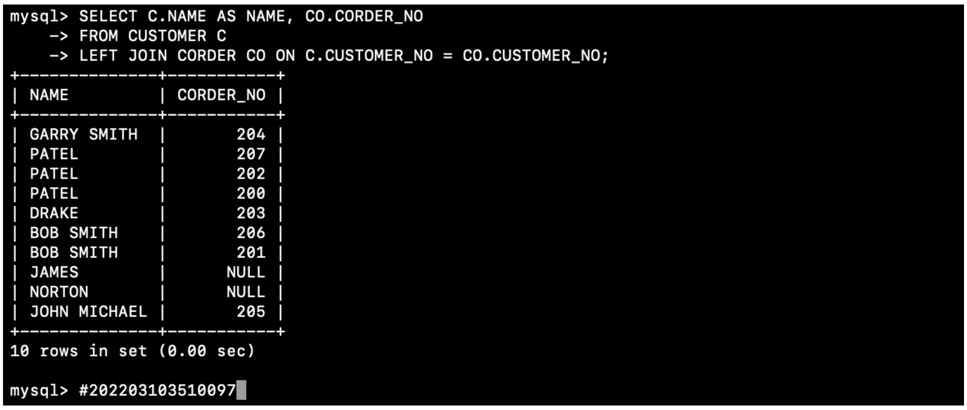
15) List the locations and addresses of all depots which do not stock product number 122. (ie where there is no stock row for that product for the depot)



```
mysql> SELECT D.LOCATION, D.ADDRESS
    -> FROM DEPOT D
    -> WHERE D.DEPOT_NO NOT IN (
    -> SELECT DISTINCT ST.DEPOT_NO
    -> FROM STOCK ST
    -> WHERE ST.PRODUCT_NO = 122
    -> );
+-------------+---------+
| LOCATION    | ADDRESS |
+-------------+---------+
| NORTH       | UK      |
| SOUTH       | USA     |
| LONDON WEST | USA     |
| EAST        | NZ      |
| WALES       | UK      |
| NORTH       | KENYA   |
+-------------+---------+
6 rows in set (0.06 sec)

mysql> #202203103510097
```

16) Set up a query which lists the names of all customers who have placed an order with the order number (corder_no) of the order merged with the names of all customers who have never placed an order (shown once, with the order number attribute null) i.e. an outer join.



```
mysql> SELECT C.NAME AS NAME, CO.CORDER_NO
    -> FROM CUSTOMER C
    -> LEFT JOIN CORDER CO ON C.CUSTOMER_NO = CO.CUSTOMER_NO;
+--------------+-----------+
| NAME         | CORDER_NO |
+--------------+-----------+
| GARRY SMITH  |       204 |
| PATEL        |       207 |
| PATEL        |       202 |
| PATEL        |       200 |
| DRAKE        |       203 |
| BOB SMITH    |       206 |
| BOB SMITH    |       201 |
| JAMES        |      NULL |
| NORTON       |      NULL |
| JOHN MICHAEL |       205 |
+--------------+-----------+
10 rows in set (0.00 sec)

mysql> #202203103510097
```

Conclusion: Implementing joins in a database system involves understanding the types of joins, parsing and optimizing SQL queries, selecting appropriate join algorithms, handling NULL values, considering parallelism, and continuously monitoring and tuning the system for optimal performance. The choice of implementation details can vary depending on the specific database system and use cases.