Task 1. Retrieve employee device data

In this task, you need to obtain information on employee devices because your team needs to update them. The information you need is in the `machines` table in the `organization` database.

First, you need to retrieve all the information about the employee devices.

1. Run the following query to select all device information from the `machines` table:

```
SELECT *
FROM machines;
```

```
MariaDB [organization]> SELECT *
    ->
    -> FROM machines;
+--------------+------------------+----------------+---------------+-------------+
| device_id    | operating_system | email_client   | OS_patch_date | employee_id |
+--------------+------------------+----------------+---------------+-------------+
| a184b775c707 | OS 1             | Email Client 1 | 2021-09-01    |        1156 |
| a192b174c940 | OS 2             | Email Client 1 | 2021-06-01    |        1052 |
| a305b818c708 | OS 3             | Email Client 2 | 2021-06-01    |        1182 |
| a317b635c465 | OS 1             | Email Client 2 | 2021-03-01    |        1130 |
| a320b137c219 | OS 2             | Email Client 2 | 2021-03-01    |        1000 |
| a398b471c573 | OS 3             | Email Client 2 | 2021-12-01    |           0 |
```

Next, you want to focus on the email client running on various devices.

2. Run the following query to select only the `device_id` and `email_client` columns from the `machines` table. Replace `X` with `device_id` and `Y` with `email_client`:

SELECT device_id, email_client FROM machines;

```
MariaDB [organization]> SELECT device_id, email_client FROM machines;
+--------------+----------------+
| device_id    | email_client   |
+--------------+----------------+
| a184b775c707 | Email Client 1 |
| a192b174c940 | Email Client 1 |
| a305b818c708 | Email Client 2 |
| a317b635c465 | Email Client 2 |
| a320b137c219 | Email Client 2 |
| a398b471c573 | Email Client 2 |
| a667b270c984 | Email Client 1 |
```

3. Complete the query to return only the `device_id`, `operating_system`, and `OS_patch_date` columns from the `machines` table. Replace `X`, `Y`, and `Z` with the columns that you need to return:

SELECT device_id, operating_system, OS_patch_date FROM machines;

```
MariaDB [organization]> SELECT device_id, operating_system, OS_patch_date
 FROM machines;
+--------------+------------------+---------------+
| device_id    | operating_system | OS_patch_date |
+--------------+------------------+---------------+
| a184b775c707 | OS 1             | 2021-09-01    |
| a192b174c940 | OS 2             | 2021-06-01    |
| a305b818c708 | OS 3             | 2021-06-01    |
| a317b635c465 | OS 1             | 2021-03-01    |
| a320b137c219 | OS 2             | 2021-03-01    |
| a398b471c573 | OS 3             | 2021-12-01    |
```

Task 2. Investigate login activity

In this task, you need to analyze the information from the `log_in_attempts` table to determine if any unusual activity has occurred.

First, you need to investigate the locations where login attempts were made to ensure that they're in expected areas (the United States, Canada, or Mexico).

1. Write a SQL query to select the `event_id` and `country` columns from the `log_in_attempts` table.

SELECT event_id, country FROM log_in_attempts;

```
MariaDB [organization]> SELECT event_id, country  FROM log_in_attempts;
+----------+---------+
| event_id | country |
+----------+---------+
|        1 | CAN     |
|        2 | CAN     |
|        3 | USA     |
|        4 | USA     |
|        5 | CANADA  |
|        6 | MEXICO  |
```

2. Write a SQL query that selects the `username`, `login_date`, and `login_time` columns from the `log_in_attempts` table.

SELECT username, login_date, login_time FROM log_in_attempts;

```
MariaDB [organization]> SELECT username, login_date, login_time columns FROM log_in_attempts;
+----------+------------+----------+
| username | login_date | columns  |
+----------+------------+----------+
| jrafael  | 2022-05-09 | 04:56:27 |
| apatel   | 2022-05-10 | 20:27:27 |
| dkot     | 2022-05-09 | 06:47:41 |
| dkot     | 2022-05-08 | 02:00:39 |
| jrafael  | 2022-05-11 | 03:05:59 |
| arutley  | 2022-05-12 | 17:00:59 |
```

3. Write a SQL query that selects all columns from the `log_in_attempts` table, using a single symbol after the `SELECT` keyword.

SELECT * FROM log_in_attempts;

```
MariaDB [organization]> SELECT * FROM log_in_attempts;
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|        1 | jrafael  | 2022-05-09 | 04:56:27   | CAN     | 192.168.243.140 |       1 |
|        2 | apatel   | 2022-05-10 | 20:27:27   | CAN     | 192.168.205.12  |       0 |
|        3 | dkot     | 2022-05-09 | 06:47:41   | USA     | 192.168.151.162 |       1 |
|        4 | dkot     | 2022-05-08 | 02:00:39   | USA     | 192.168.178.71  |       0 |
|        5 | jrafael  | 2022-05-11 | 03:05:59   | CANADA  | 192.168.86.232  |       0 |
|        6 | arutley  | 2022-05-12 | 17:00:59   | MEXICO  | 192.168.3.24    |       0 |
```

Task 3. Order login attempts data

In this task, you need to use the `ORDER BY` keyword. You'll sequence the data that your query returns according to the login date and time.

First, you need to sort the information by date.

1. Run the following query, which orders `log_in_attempts` data by `login_date`:

```
SELECT *
FROM log_in_attempts
ORDER BY login_date;
```

```
MariaDB [organization]> SELECT *
    ->
    -> FROM log_in_attempts
    ->
    -> ORDER BY login_date;
+----------+----------+------------+------------+---------+----------------+---------+
| event_id | username | login_date | login_time | country | ip_address     | success |
+----------+----------+------------+------------+---------+----------------+---------+
|      145 | ivelasco | 2022-05-08 | 09:06:02   | CANADA  | 192.168.39.196 |       1 |
|      163 | tmitchel | 2022-05-08 | 09:21:16   | MEX     | 192.168.119.29 |       0 |
|       36 | asundara | 2022-05-08 | 09:00:42   | US      | 192.168.78.151 |       1 |
|      165 | jreckley | 2022-05-08 | 15:28:43   | MEXICO  | 192.168.34.193 |       0 |
|      168 | jlansky  | 2022-05-08 | 13:25:42   | USA     | 192.168.210.94 |       1 |
|      169 | alevitsk | 2022-05-08 | 08:10:43   | CANADA  | 192.168.210.228 |       0 |
```

Now, you need to further organize the previous results by ordering them by login_time.

2. Modify the query from the previous step by adding the login time to the ORDER BY clause. You must replace X with the appropriate column name:

```
SELECT *

FROM log_in_attempts

ORDER BY login_date, X;
```

```
SELECT *

FROM log_in_attempts

ORDER BY login_date, login_time;
```

```
MariaDB [organization]> SELECT *
    ->
    -> FROM log_in_attempts
    ->
    -> ORDER BY login_date, login_time;
+----------+----------+------------+------------+---------+-----------------+---------+
| event_id | username | login_date | login_time | country | ip_address      | success |
+----------+----------+------------+------------+---------+-----------------+---------+
|      117 | bsand    | 2022-05-08 | 00:19:11   | USA     | 192.168.197.187 |       0 |
|       92 | pwashing | 2022-05-08 | 00:36:12   | US      | 192.168.247.219 |       0 |
|        8 | bisles   | 2022-05-08 | 01:30:17   | US      | 192.168.119.173 |       0 |
|        4 | dkot     | 2022-05-08 | 02:00:39   | USA     | 192.168.178.71  |       0 |
|       80 | cjackson | 2022-05-08 | 02:18:10   | CANADA  | 192.168.33.140  |       1 |
|       43 | mcouliba | 2022-05-08 | 02:35:34   | CANADA  | 192.168.16.208  |       0 |
```

Conclusion

I have completed this activity, and I now have practical experience in running basic SQL queries to

- select specific columns from a table,
- select all columns from a table by using an asterisk (*), and
- sort query results using the ORDER BY keyword.

These basic queries form the foundation for running more advanced queries and applying filters later.