LESSON 6 LAB

Subqueries in WHERE and FROM Clauses

Ihor Liutak

Content

1. Understand and implement subqueries in the WHERE and FROM clauses
2. Explore practical scenarios using the db.loc database
3. Provide hands-on practice with real-world queries

Short description

Familiarize students with the MySQL console and basic commands. Teach how to create databases and tables with appropriate data types. Cover essential database management commands (import/export, user management).

Kurzbeschreibung

Machen Sie die Studierenden mit der MySQL-Konsole und grundlegenden Befehlen vertraut. Bringen Sie ihnen bei, wie man Datenbanken und Tabellen mit entsprechenden Datentypen erstellt. Behandeln Sie wichtige Befehle zur Datenbankverwaltung (Import/Export, Benutzerverwaltung).

Subqueries in the WHERE Clause

General Syntax:

```
SELECT columns
FROM table
WHERE column = (SELECT column FROM table WHERE condition);
```

Key Components:

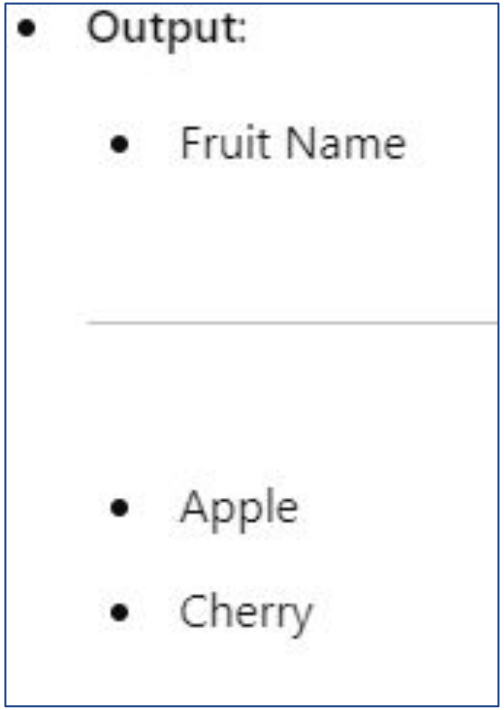
- Subqueries in the WHERE clause allow filtering based on results from another query.

Example 1 - WHERE Clause

Scenario: Find all fruits originating from countries in Europe.

Query:

```
SELECT name
FROM fruits
WHERE origin IN (
    SELECT id
    FROM countries
    WHERE continent = 'Europe'
);
```



- Output:
    - Fruit Name

    - Apple
    - Cherry

Subqueries in the FROM Clause

General Syntax:

Example Use Case:

Perform calculations or aggregations before filtering or displaying results

```
SELECT columns
FROM (SELECT column FROM table WHERE condition) AS alias;
```

Key Components:

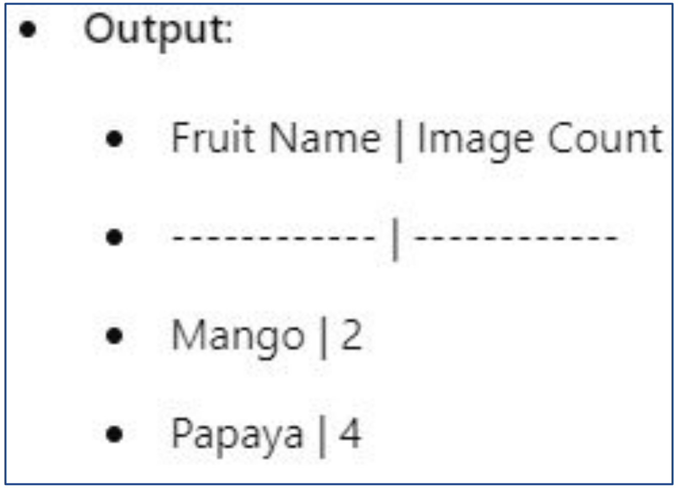
- Subqueries in the FROM clause create a temporary table for the main query to use.

Example 2 - FROM Clause

Scenario: List all fruits along with the total number of images per fruit.

Query:

```
SELECT fruits.name, images.image_count
FROM fruits
JOIN (
    SELECT fruit_id, COUNT(*)
                  AS image_count
    FROM fruit_images
    GROUP BY fruit_id
) AS images
ON fruits.id = images.fruit_id;
```



```
• Output:

    • Fruit Name | Image Count

    • ----------- | -----------

    • Mango | 2

    • Papaya | 4
```

Hands-On Task 1

Task: Find fruits with storage period the same as "Mangosteen,".

Query:

```
SELECT DISTINCT name
FROM fruits
WHERE storage_period IN (
    SELECT DISTINCT storage_period
    FROM fruits
    WHERE name = 'Mangosteen'
);
```

Objective: Use a subquery in the WHERE clause to filter results.

Hands-On Task 2

Task: Display each continent and the number of fruits originating from it.

Query:

```
SELECT continent, COUNT(*) AS fruit_count
FROM (
    SELECT countries.continent, fruits.name
    FROM fruits
    JOIN countries ON fruits.origin = countries.id
) AS subquery
GROUP BY continent;
```

Objective: Use a subquery in the FROM clause to aggregate data.

Recap and Q&A

Recap:

- Subqueries in WHERE and FROM clauses enhance SQL query flexibility.

Best Practices:

- Avoid overly complex subqueries; prefer joins if possible
- Use Aliases: Always name subquery results clearly for better readability
- Indexing: Ensure columns used in subqueries are indexed for faster execution