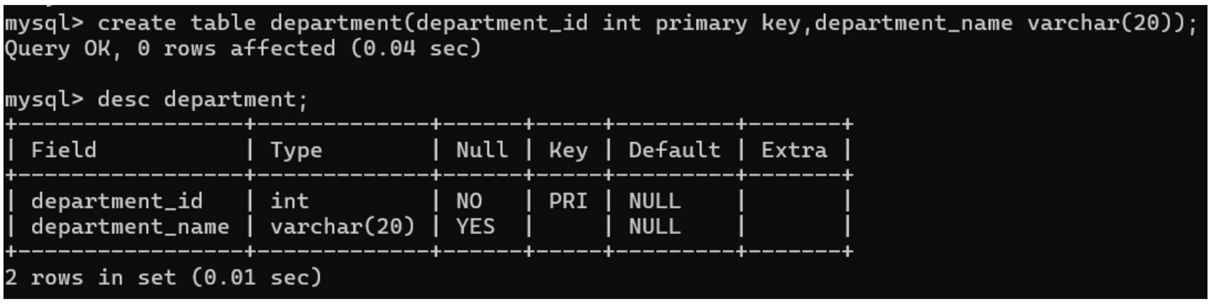
Creating Tables

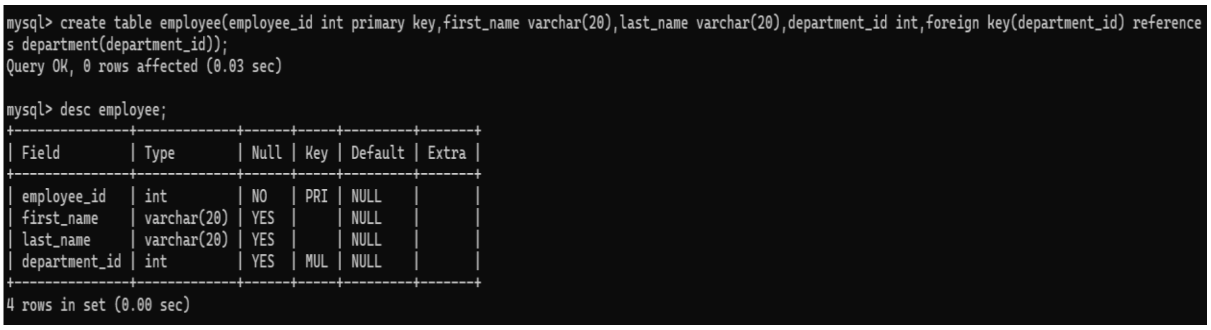
- **Department Table**



```
mysql> create table department(department_id int primary key,department_name varchar(20));
Query OK, 0 rows affected (0.04 sec)

mysql> desc department;
+-----------------+-------------+------+-----+---------+-------+
| Field           | Type        | Null | Key | Default | Extra |
+-----------------+-------------+------+-----+---------+-------+
| department_id   | int         | NO   | PRI | NULL    |       |
| department_name | varchar(20) | YES  |     | NULL    |       |
+-----------------+-------------+------+-----+---------+-------+
2 rows in set (0.01 sec)
```

- **Employee Table**

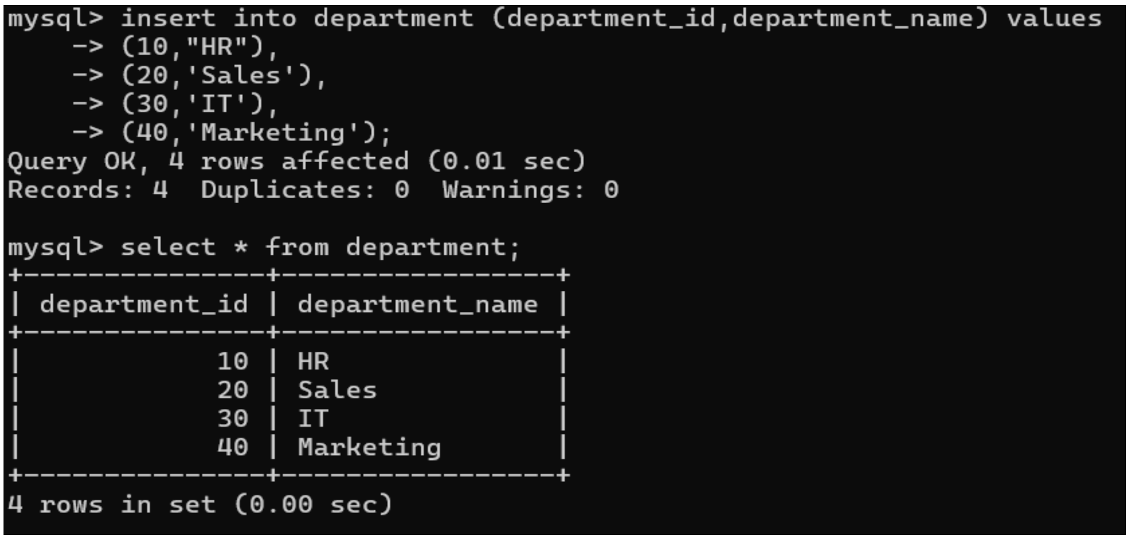


```
mysql> create table employee(employee_id int primary key,first_name varchar(20),last_name varchar(20),department_id int,foreign key(department_id) reference
s department(department_id));
Query OK, 0 rows affected (0.03 sec)

mysql> desc employee;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| employee_id   | int         | NO   | PRI | NULL    |       |
| first_name    | varchar(20) | YES  |     | NULL    |       |
| last_name     | varchar(20) | YES  |     | NULL    |       |
| department_id | int         | YES  | MUL | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

Inserting Values into Tables

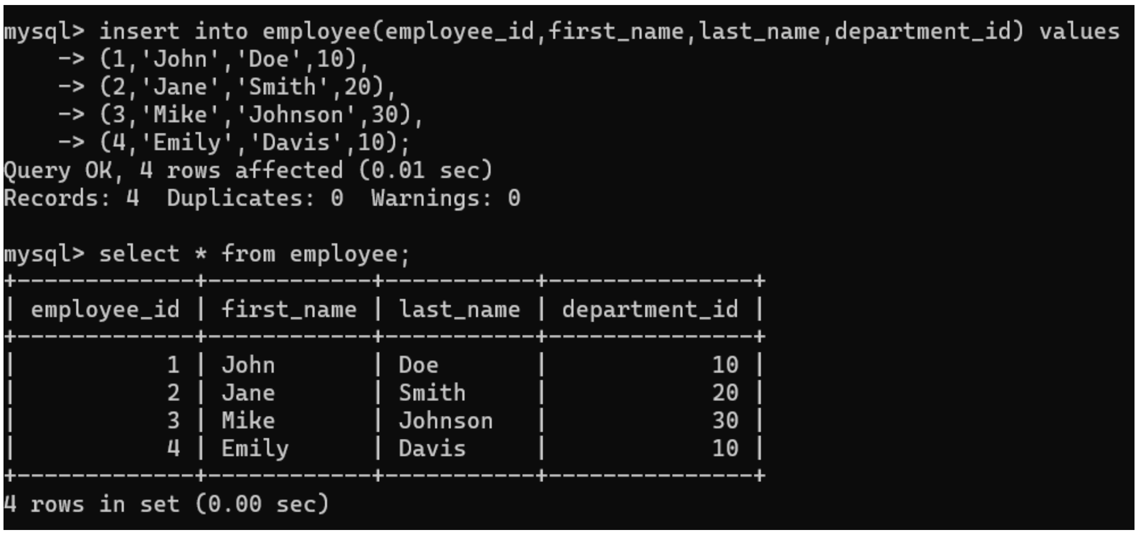
- **Department Table**



```
mysql> insert into department (department_id,department_name) values
    -> (10,"HR"),
    -> (20,'Sales'),
    -> (30,'IT'),
    -> (40,'Marketing');
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> select * from department;
+---------------+-----------------+
| department_id | department_name |
+---------------+-----------------+
|            10 | HR              |
|            20 | Sales           |
|            30 | IT              |
|            40 | Marketing       |
+---------------+-----------------+
4 rows in set (0.00 sec)
```

- **Employee Table**

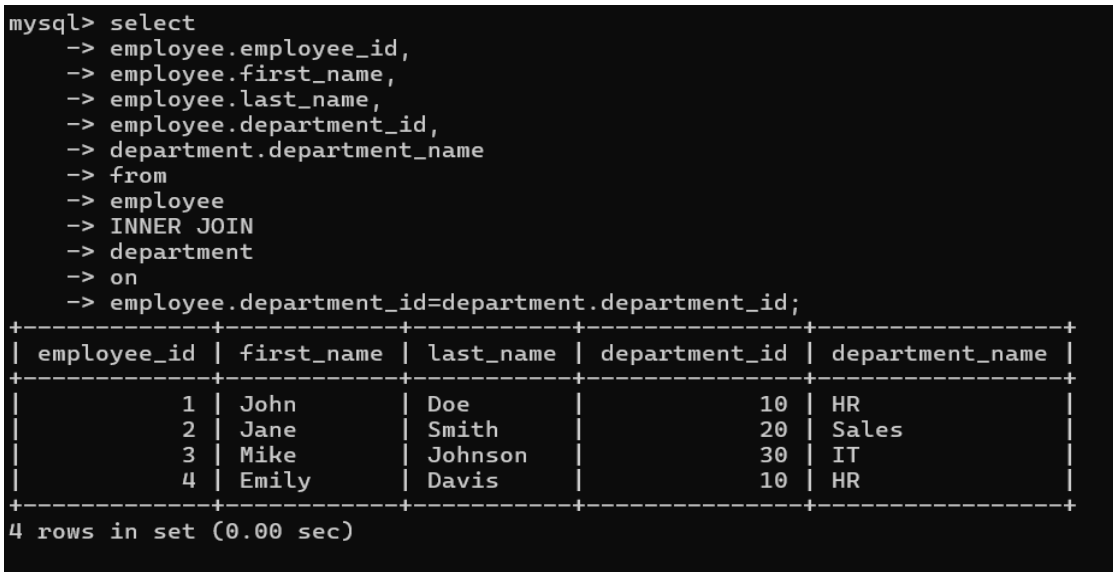


```
mysql> insert into employee(employee_id,first_name,last_name,department_id) values
    -> (1,'John','Doe',10),
    -> (2,'Jane','Smith',20),
    -> (3,'Mike','Johnson',30),
    -> (4,'Emily','Davis',10);
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> select * from employee;
+-------------+------------+-----------+---------------+
| employee_id | first_name | last_name | department_id |
+-------------+------------+-----------+---------------+
|           1 | John       | Doe       |            10 |
|           2 | Jane       | Smith     |            20 |
|           3 | Mike       | Johnson   |            30 |
|           4 | Emily      | Davis     |            10 |
+-------------+------------+-----------+---------------+
4 rows in set (0.00 sec)
```

Queries

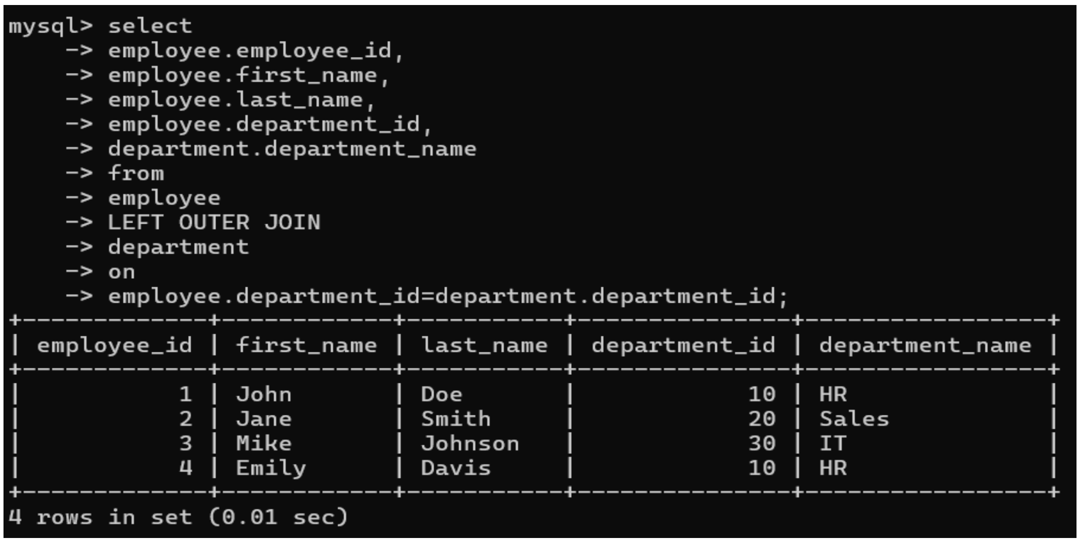
- **Inner Join**



```
mysql> select
    -> employee.employee_id,
    -> employee.first_name,
    -> employee.last_name,
    -> employee.department_id,
    -> department.department_name
    -> from
    -> employee
    -> INNER JOIN
    -> department
    -> on
    -> employee.department_id=department.department_id;
+-------------+------------+-----------+---------------+-----------------+
| employee_id | first_name | last_name | department_id | department_name |
+-------------+------------+-----------+---------------+-----------------+
|           1 | John       | Doe       |            10 | HR              |
|           2 | Jane       | Smith     |            20 | Sales           |
|           3 | Mike       | Johnson   |            30 | IT              |
|           4 | Emily      | Davis     |            10 | HR              |
+-------------+------------+-----------+---------------+-----------------+
4 rows in set (0.00 sec)
```

- This query will return only those rows where there is a similar department_id in both the Employee and Department tables. If an employee's department_id does not have a corresponding department_id in the Department table, that employee's data will not be included in the result set. Similarly, departments without any employees will not appear in the result set.

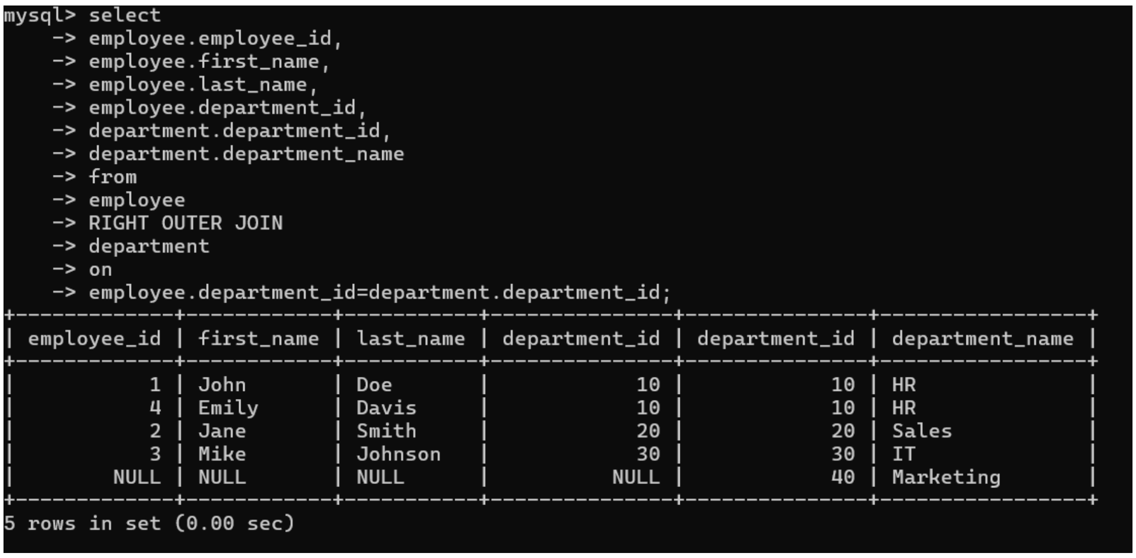
- **Left Outer Join**



```
mysql> select
    -> employee.employee_id,
    -> employee.first_name,
    -> employee.last_name,
    -> employee.department_id,
    -> department.department_name
    -> from
    -> employee
    -> LEFT OUTER JOIN
    -> department
    -> on
    -> employee.department_id=department.department_id;
+-------------+------------+-----------+---------------+-----------------+
| employee_id | first_name | last_name | department_id | department_name |
+-------------+------------+-----------+---------------+-----------------+
|           1 | John       | Doe       |            10 | HR              |
|           2 | Jane       | Smith     |            20 | Sales           |
|           3 | Mike       | Johnson   |            30 | IT              |
|           4 | Emily      | Davis     |            10 | HR              |
+-------------+------------+-----------+---------------+-----------------+
4 rows in set (0.01 sec)
```

- LEFT (OUTER) JOIN returns all records from the left table, and the matched records from the right table. This query will return all records from the Employee table, along with their corresponding department information from the Department table if available.

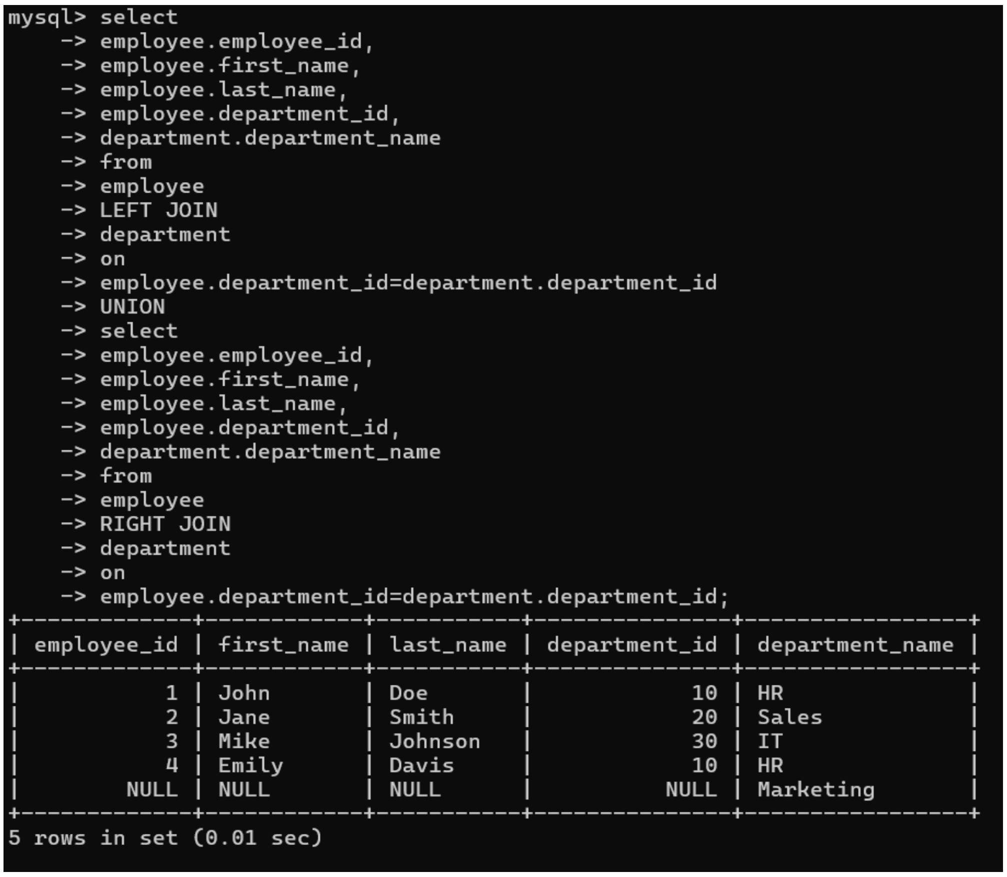
- **Right Outer Join**



```
mysql> select
    -> employee.employee_id,
    -> employee.first_name,
    -> employee.last_name,
    -> employee.department_id,
    -> department.department_id,
    -> department.department_name
    -> from
    -> employee
    -> RIGHT OUTER JOIN
    -> department
    -> on
    -> employee.department_id=department.department_id;
+-------------+------------+-----------+---------------+---------------+-----------------+
| employee_id | first_name | last_name | department_id | department_id | department_name |
+-------------+------------+-----------+---------------+---------------+-----------------+
|           1 | John       | Doe       |            10 |            10 | HR              |
|           4 | Emily      | Davis     |            10 |            10 | HR              |
|           2 | Jane       | Smith     |            20 |            20 | Sales           |
|           3 | Mike       | Johnson   |            30 |            30 | IT              |
|        NULL | NULL       | NULL      |          NULL |            40 | Marketing       |
+-------------+------------+-----------+---------------+---------------+-----------------+
5 rows in set (0.00 sec)
```

- The Department table is on the left side. RIGHT OUTER JOIN specifies that all rows from the right table (Employee table) will be included in the result set, along with matched rows from the left table (Department table).If there's no match in the left table (Department), NULL values will be included for columns selected from the left table.

- **Full outer Join**

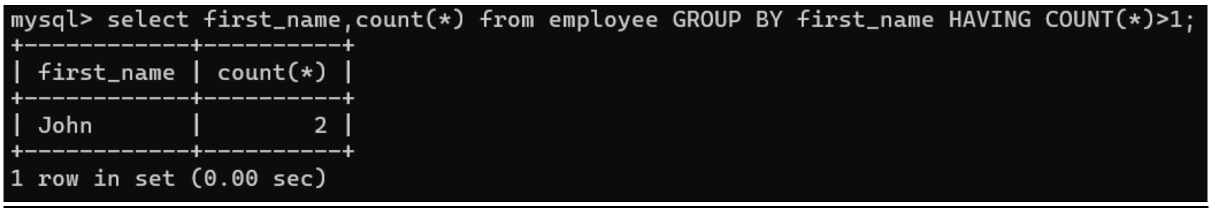


```
mysql> select
    -> employee.employee_id,
    -> employee.first_name,
    -> employee.last_name,
    -> employee.department_id,
    -> department.department_name
    -> from
    -> employee
    -> LEFT JOIN
    -> department
    -> on
    -> employee.department_id=department.department_id
    -> UNION
    -> select
    -> employee.employee_id,
    -> employee.first_name,
    -> employee.last_name,
    -> employee.department_id,
    -> department.department_name
    -> from
    -> employee
    -> RIGHT JOIN
    -> department
    -> on
    -> employee.department_id=department.department_id;
+-------------+------------+-----------+---------------+-----------------+
| employee_id | first_name | last_name | department_id | department_name |
+-------------+------------+-----------+---------------+-----------------+
|           1 | John       | Doe       |            10 | HR              |
|           2 | Jane       | Smith     |            20 | Sales           |
|           3 | Mike       | Johnson   |            30 | IT              |
|           4 | Emily      | Davis     |            10 | HR              |
|        NULL | NULL       | NULL      |          NULL | Marketing       |
+-------------+------------+-----------+---------------+-----------------+
5 rows in set (0.01 sec)
```

- An full outer join is a method of combining tables so that the result includes unmatched rows of both tables. If you are joining two tables and want the result set to include unmatched rows from both tables, use a FULL OUTER JOIN clause.

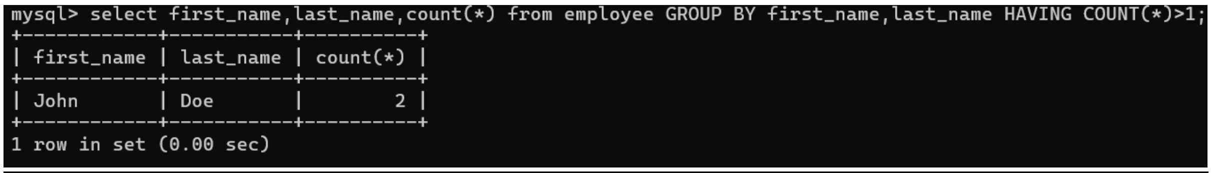
<u>FINDING DUPLICATE RECORDS</u>

- **Based on first name**



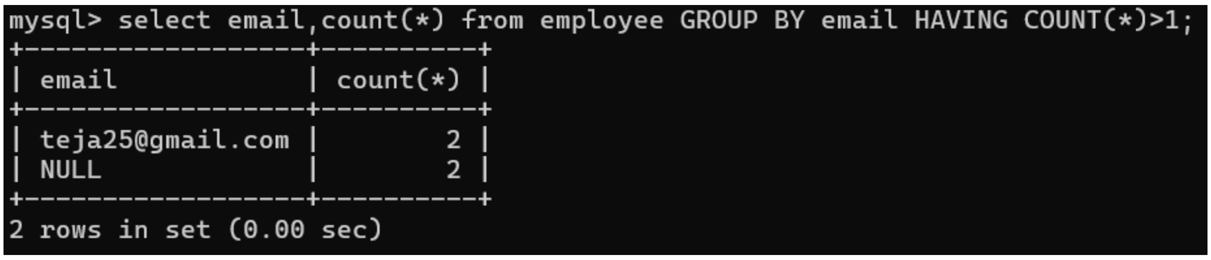
```
mysql> select first_name,count(*) from employee GROUP BY first_name HAVING COUNT(*)>1;
+------------+----------+
| first_name | count(*) |
+------------+----------+
| John       |        2 |
+------------+----------+
1 row in set (0.00 sec)
```

- **Based on first name and last name**



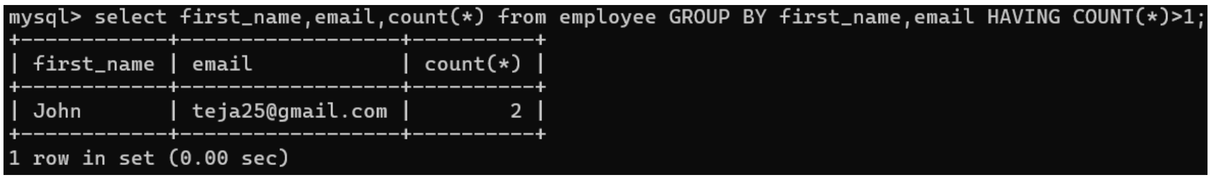
```
mysql> select first_name,last_name,count(*) from employee GROUP BY first_name,last_name HAVING COUNT(*)>1;
+------------+-----------+----------+
| first_name | last_name | count(*) |
+------------+-----------+----------+
| John       | Doe       |        2 |
+------------+-----------+----------+
1 row in set (0.00 sec)
```

- **Based on email**



```
mysql> select email,count(*) from employee GROUP BY email HAVING COUNT(*)>1;
+-------------------+----------+
| email             | count(*) |
+-------------------+----------+
| teja25@gmail.com  |        2 |
| NULL              |        2 |
+-------------------+----------+
2 rows in set (0.00 sec)
```

- **Based on first name and email**



```
mysql> select first_name,email,count(*) from employee GROUP BY first_name,email HAVING COUNT(*)>1;
+------------+------------------+----------+
| first_name | email            | count(*) |
+------------+------------------+----------+
| John       | teja25@gmail.com |        2 |
+------------+------------------+----------+
1 row in set (0.00 sec)
```

<u>Convert bookstore.xml into json</u>

<bookstore>

<book>

<title>Harry Potter</title>

<author>J.K. Rowling</author>

<price>29.99</price>

<available>true</available>

</book>

<book>

<title>The Hobbit</title>

<author>J.R.R. Tolkien</author>

<price>19.99</price>

<available>false</available>

</book>

</bookstore>

- **JSON**

```
{
"bookstore": {
"book": [
{
"title": "Harry Potter",
"author": "J.K. Rowling",
"price": 29.99,
"available": true
},
{
```