****

**Green University of Bangladesh**

**Department of Computer Science and Engineering (CSE)**

**Faculty of Sciences and Engineering**

**Semester: (Summer, Year:2025), B.Sc. in CSE (Day)**

**LAB REPORT NO: 3**

**Course Title:** Database Lab

**Course Code:** CSE 210 **Section:** 232-D1

**Lab Experiment Name**: Subquery-Related Problems in MySQL

**Student Details**

|  |  |  |
| --- | --- | --- |
| **Name** | | **ID** |
| **1.** | Rukonuzzaman Topu | 232002280 |

**Submission Date : 08-06-2025**

**Course Teacher’s Name :** Farhana Akter Sunny

**[For Teachers use only: Don’t Write Anything inside this box]**

|  |
| --- |
| **Lab Report Status**  **Marks: ………………………………… Signature:.....................**  **Comments:.............................................. Date:..............................** |

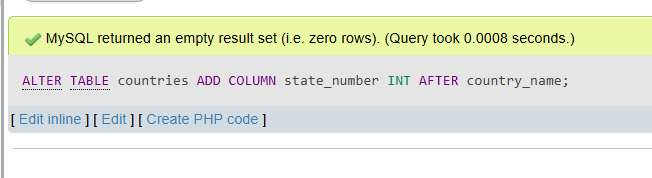
1. **TITLE OF THE LAB EXPERIMENT:**

Subquery-Related Problems in MySQL.

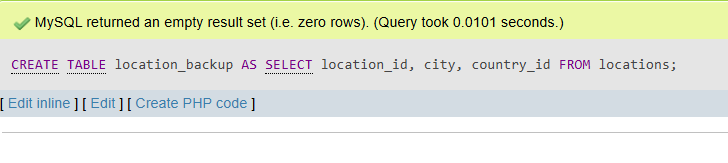
**2.OBJECTIVES:**

* Change the table safely – add, rename, and delete columns without losing data.
* Make a backup – copy key columns from one table into a smaller backup table.
* Use subqueries – write short “query-inside-a-query” SQL to answer real HR questions, such as:
* How many employees were hired after 21 Sep 1989?
* Which employees left before 31 Dec 1998?
* Who earns the second-highest salary?
* Which staff share the same job as employee 108?
* Which countries have a city that starts with “B”?
* Compare salaries – find employees who earn more than the average or the lowest pay in each department.
* Check departments – spot teams whose average salary is over 14 000.

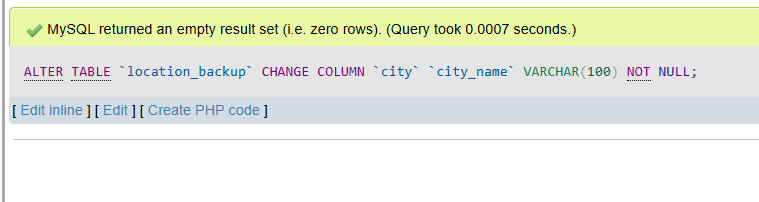
**PROCEDURE:**

**1. Create a new column state\_number after the country name in the countries table.**

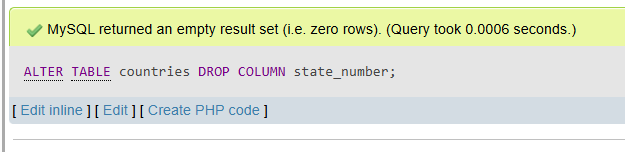
**2. Create a backup table for location with only location\_id, city, country\_id.**

****

**3. Change the name of the column city to city\_name in the locations backup table.**

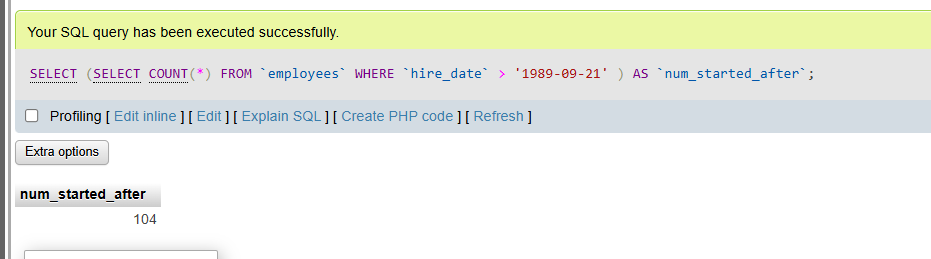


**4. Delete column state\_number  from the countries table.**

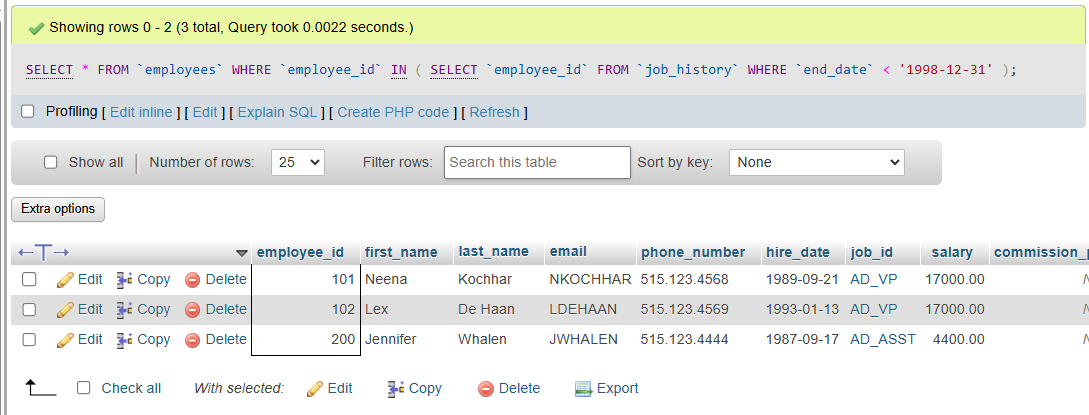
****

**Subqueries**

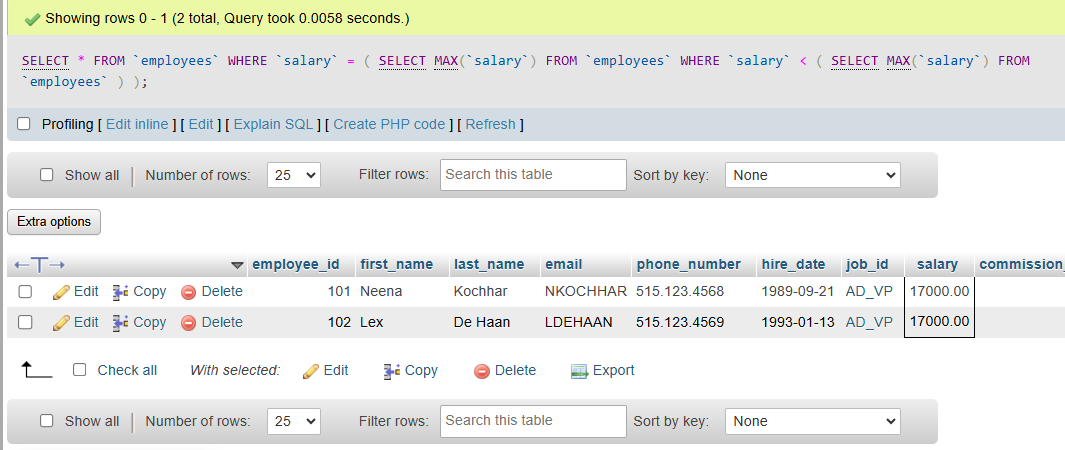
**5. Find the number of employees who started their job after ‘1989-09-21’.**



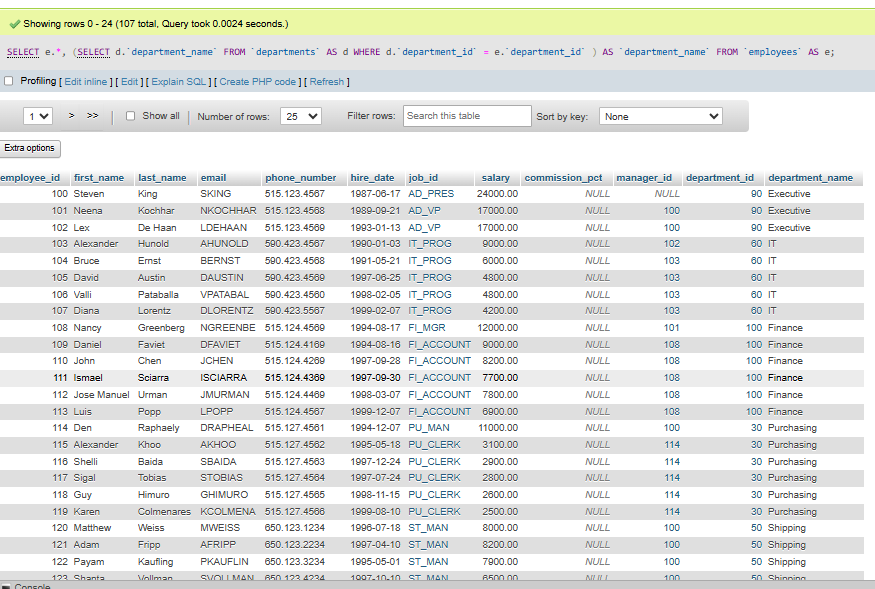
**6. Find the employees' information whose job ended before 1998-12-31.**



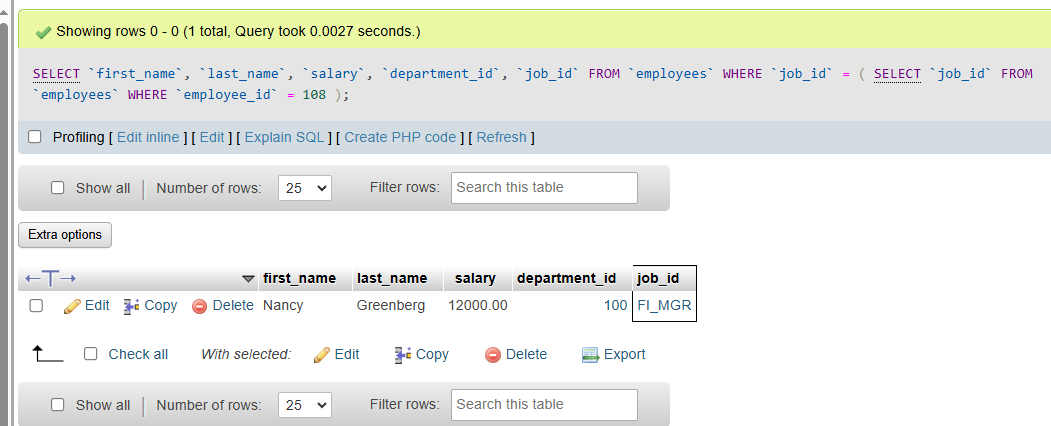
**7.** **Find the employee who has the second highest salary.**

****

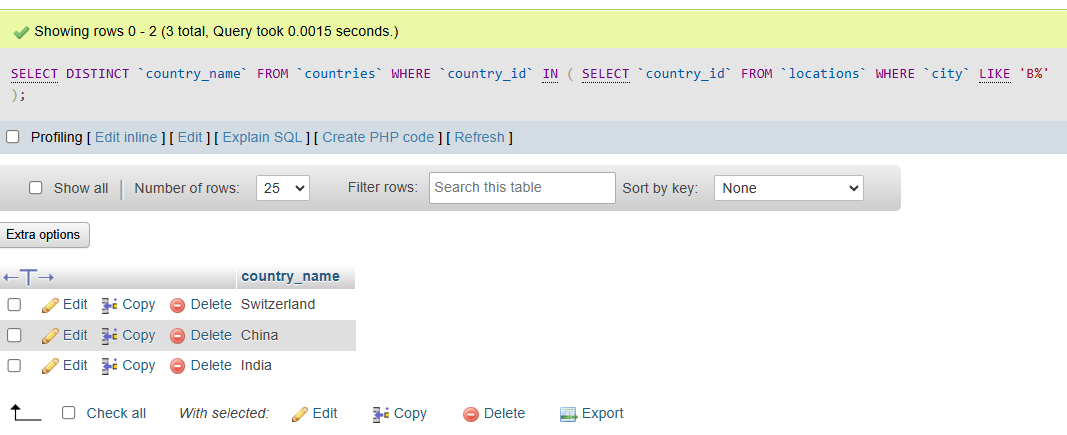
8. Find the employee’s information with their department name**.**



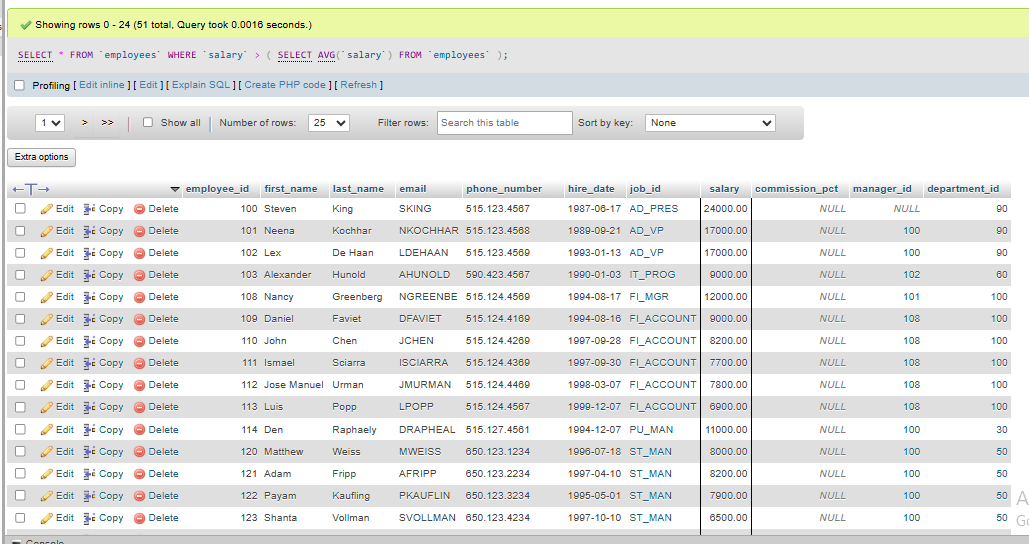
9. Display the name, salary, department\_id, job\_id for those employees who work in the same designation as the employees whose id is 108**.**



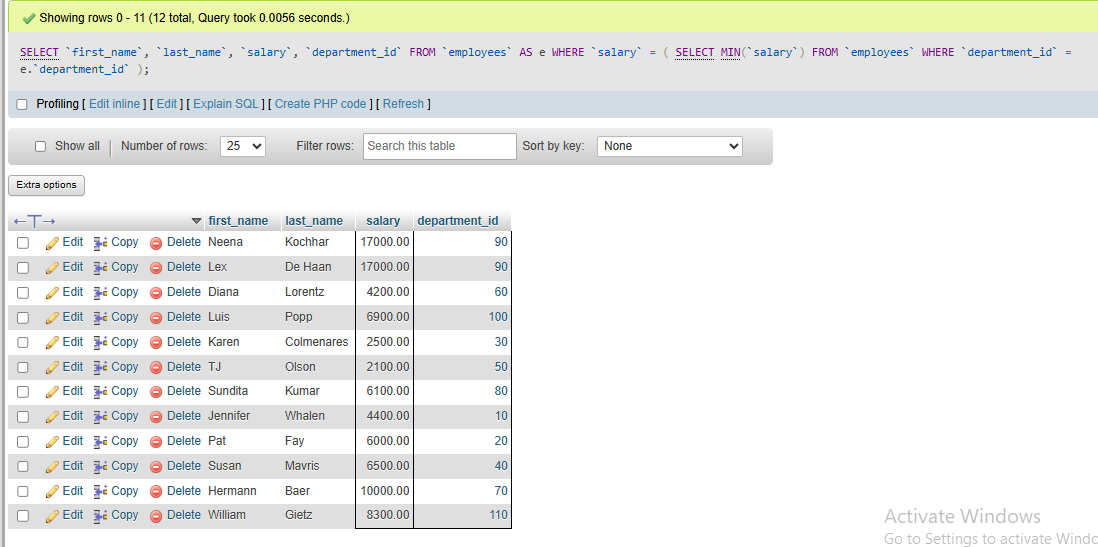
10. Find the name of the country with a city name starting with ‘B’.



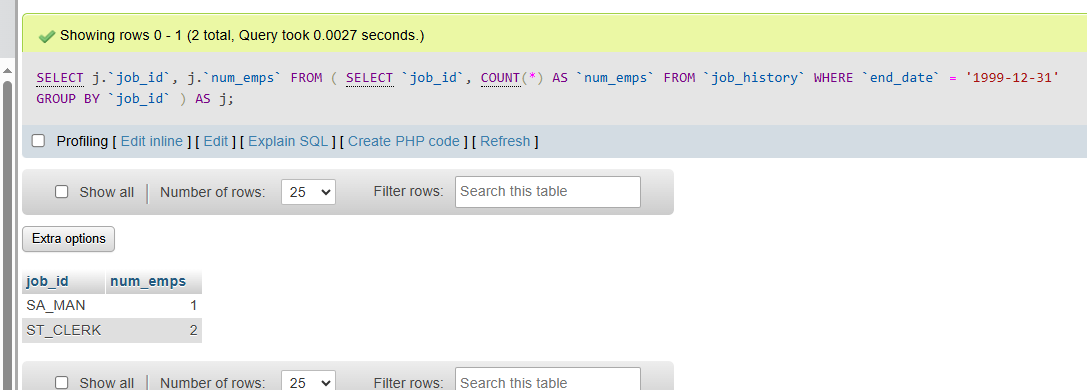
11. Find the employee’s information whose salary is greater than average salary.

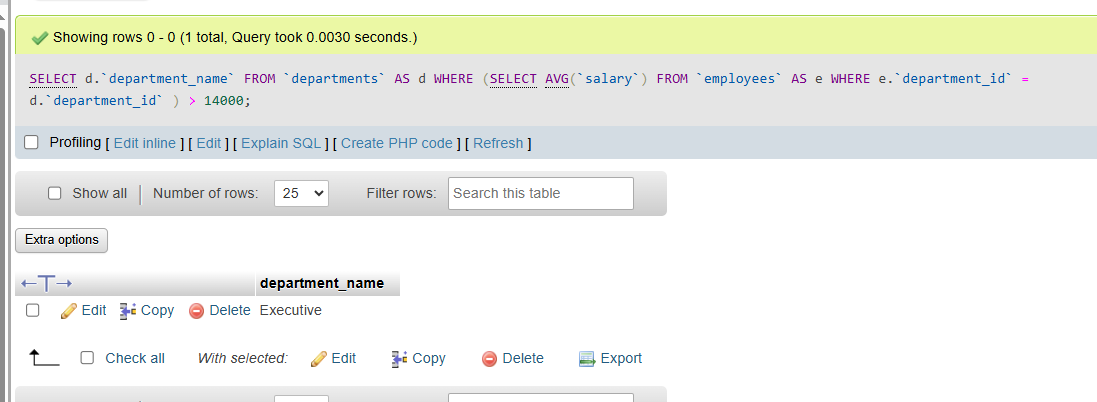


12. Display the name, salary, department\_id for those employees whose salary is the smallest salary of any of the departments**.**

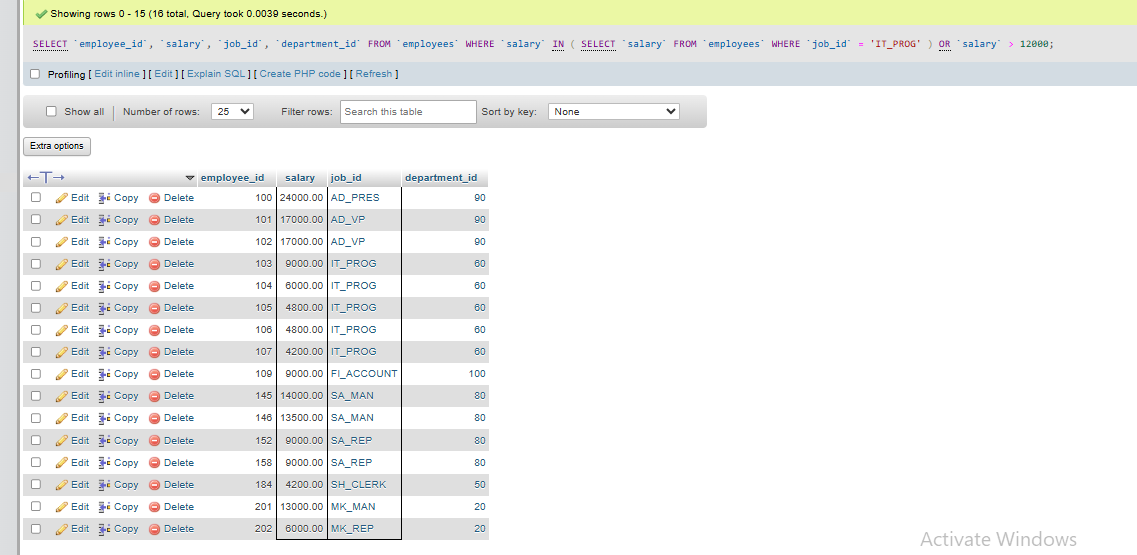


13. Find the number of employees in each designation who have ended their job on 1999-12-31.



14. Find those departments where the average salary of the instructors is more than 14,000.  


15. Find the employee\_id, salary, job\_id, department\_id for those employees whose salary will be like anyone from IT\_prog or greater than 12000



16. Find the average salary of each job whose salary is greater than 25000



**Discussion:**  
In this report, I use the dummy database provided in the lab to tackle a series of employment-related subquery problems. I begin by making safe tweaks to the tables—adding and later removing a state\_number column, creating a slim backup of the locations table, and renaming city to city\_name so the data reads more clearly.After the table changes, I dive into subqueries. These “queries inside a query” let me answer real HR questions in one go: counting who was hired after 21 Sept 1989, spotting employees who left before 31 Dec 1998, finding the second-highest salary, and pulling everyone who shares the same job as employee 108. I also search for countries with cities that start with “B,” list staff earning above the average, pick out the lowest-paid worker in every department, and flag departments whose average pay tops 14 000.By combining careful table edits with well-aimed subqueries, I show how to extract useful insights quickly and safely from a workplace database—without risking data loss or running dozens of separate commands.