**JAVA FOUNDATION TRAINING**

**BATCH 8**

**MySQL CODING CHALLENGE**

**Name:** Rajalakshmi Ganesh              **Case Study:** Career Hub

**TASKS**

**1. Provide a SQL script that initializes the database for the Job Board scenario CareerHub.**

🡪 CREATE DATABASE IF NOT EXISTS CareerHubSystem;

🡪 USE CareerHubSystem;

**2. Create tables for Companies, Jobs, Applicants and Applications.**

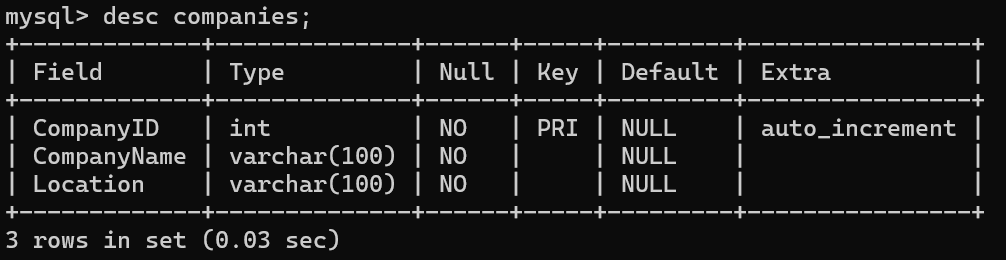
**1. Companies:**

🡪 CREATE TABLE IF NOT EXISTS Companies (

CompanyID INT AUTO\_INCREMENT PRIMARY KEY,

CompanyName VARCHAR(100) NOT NULL,

Location VARCHAR(100) NOT NULL);



**2.Jobs:**

**🡪** CREATE TABLE IF NOT EXISTS Jobs (

JobID INT AUTO\_INCREMENT PRIMARY KEY,

CompanyID INT,

JobTitle VARCHAR(100) NOT NULL,

JobDescription TEXT NOT NULL,

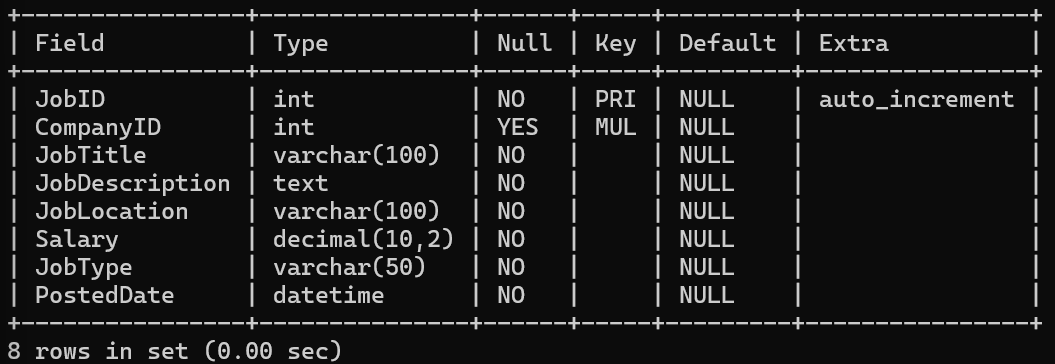
JobLocation VARCHAR(100) NOT NULL,

Salary DECIMAL(10,2) NOT NULL,

JobType VARCHAR(50) NOT NULL,

PostedDate DATETIME NOT NULL,

FOREIGN KEY (CompanyID) REFERENCES Companies(CompanyID) );



**3.Applicants:**

**🡪** CREATE TABLE IF NOT EXISTS Jobs (

JobID INT AUTO\_INCREMENT PRIMARY KEY,

CompanyID INT,

JobTitle VARCHAR(100) NOT NULL,

JobDescription TEXT NOT NULL,

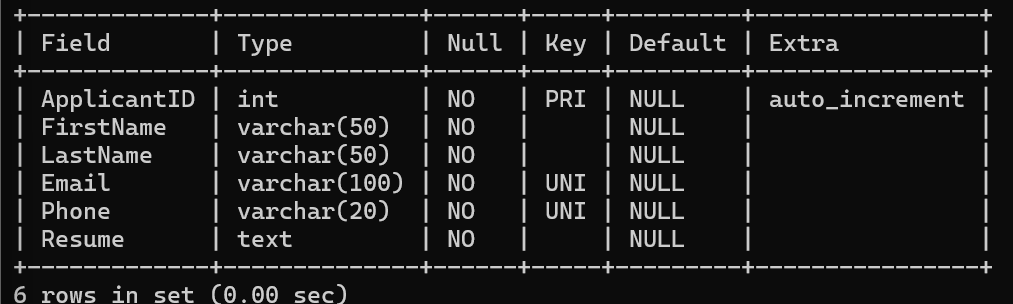
JobLocation VARCHAR(100) NOT NULL,

Salary DECIMAL(10,2) NOT NULL,

JobType VARCHAR(50) NOT NULL,

PostedDate DATETIME NOT NULL,

FOREIGN KEY (CompanyID) REFERENCES Companies(CompanyID) );



**4.Applications:**

🡪 CREATE TABLE IF NOT EXISTS Applications (

ApplicationID INT AUTO\_INCREMENT PRIMARY KEY,

JobID INT,

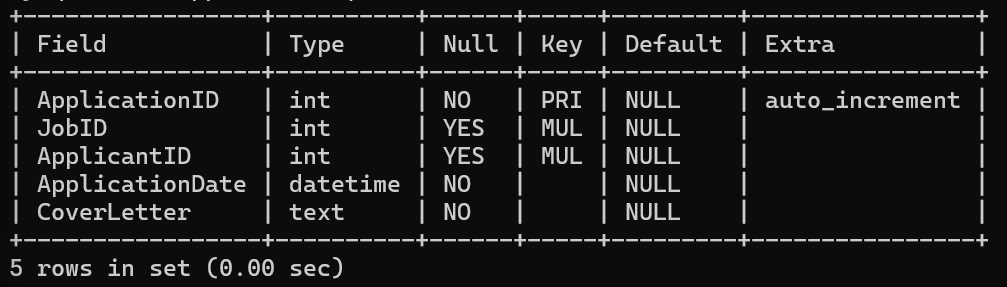
ApplicantID INT,

ApplicationDate DATETIME NOT NULL,

CoverLetter TEXT,

FOREIGN KEY (JobID) REFERENCES Jobs(JobID),

FOREIGN KEY (ApplicantID) REFERENCES Applicants(ApplicantID) );



3. Define appropriate primary keys, foreign keys, and constraints.

4. Ensure the script handles potential errors, such as if the database or tables already exist.

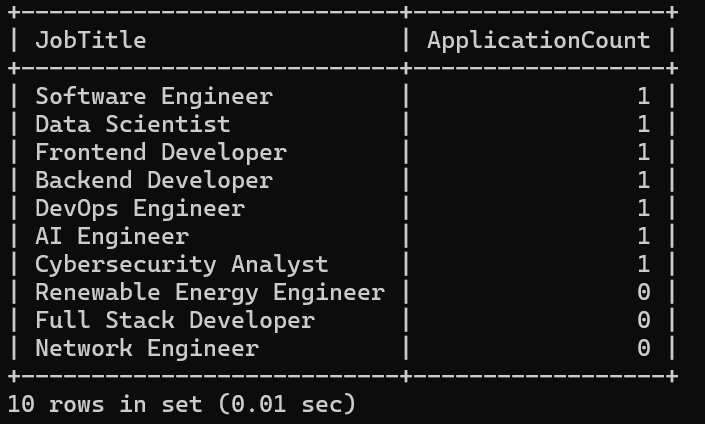
**5. Write an SQL query to count the number of applications received for each job listing in the "Jobs" table. Display the job title and the corresponding application count. Ensure that it lists all jobs, even if they have no applications.**

🡪 SELECT j.JobTitle,COUNT(a.ApplicationID) AS ApplicationCount

FROM Jobs j

JOIN Applications a ON j.JobID = a.JobID

GROUP BY j.JobTitle;

****

**6. Develop an SQL query that retrieves job listings from the "Jobs" table within a specified salary range. Allow parameters for the minimum and maximum salary values. Display the job title, company name, location, and salary for each matching job.**

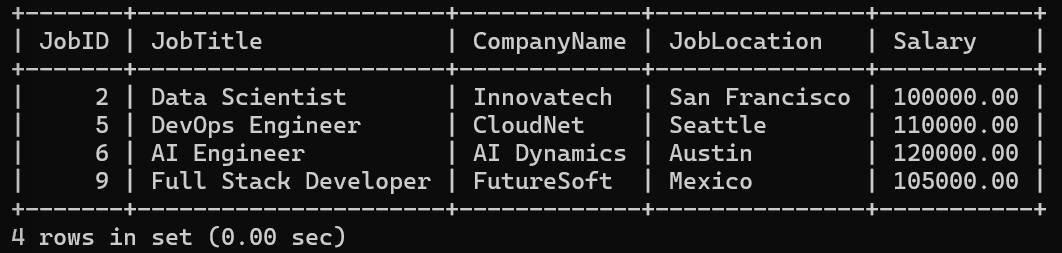
**🡪** SELECT j.JobID, j.JobTitle, c.CompanyName,j.JobLocation, j.Salary

FROM Jobs j

JOIN Companies c ON j.CompanyID = c.CompanyID

WHERE j.Salary BETWEEN 100000.00 AND 200000.00;

(WHERE j.Salary BETWEEN **Min\_Salary** AND **Max\_Salary**)



**7. Write an SQL query that retrieves the job application history for a specific applicant. Allow a parameter for the ApplicantID, and return a result set with the job titles, company names, and application dates for all the jobs the applicant has applied to.**

**🡪** SELECT j.JobTitle, c.CompanyName, a.ApplicationDate

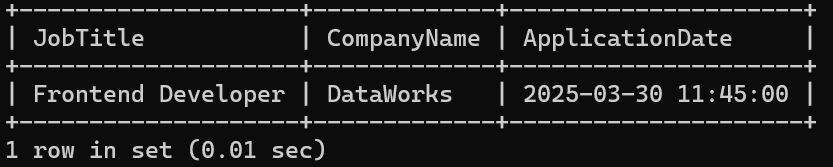
FROM Applications a

JOIN Jobs j ON a.JobID = j.JobID

JOIN Companies c ON j.CompanyID = c.CompanyID

WHERE a.ApplicantID = 3;

(WHERE a.ApplicantID = **@ApplicantID**)

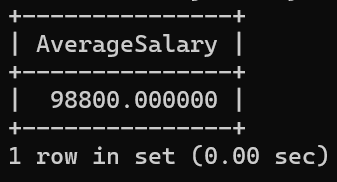
****

**8. Create an SQL query that calculates and displays the average salary offered by all companies for job listings in the "Jobs" table. Ensure that the query filters out jobs with a salary of zero.**

**🡪** SELECT AVG(j.Salary) AS AverageSalary

FROM Jobs j

WHERE j.Salary > 0;

****

**9. Write an SQL query to identify the company that has posted the most job listings. Display the company name along with the count of job listings they have posted. Handle ties if multiple companies have the same maximum count.**

**🡪** SELECT c.CompanyName, COUNT(j.JobID) AS JobCount

FROM Companies c

JOIN Jobs j ON c.CompanyID = j.CompanyID

GROUP BY c.CompanyID, c.CompanyName

HAVING JobCount = (

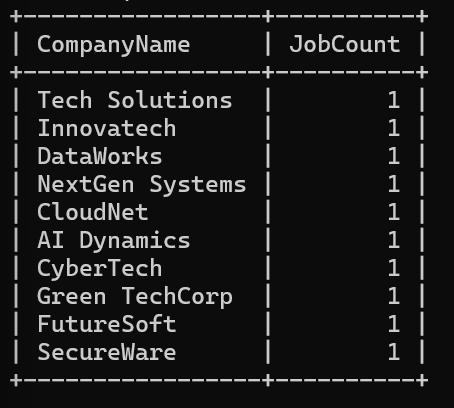
SELECT MAX(JobCount)

FROM (

SELECT COUNT(j.JobID) AS JobCount

FROM Jobs j

GROUP BY j.CompanyID ) AS JobCounts );



**10. Find the applicants who have applied for positions in companies located in 'CityX' and have at least 3 years of experience.**

**🡪** SELECT DISTINCT a.ApplicantID, a.FirstName, a.LastName, a.Email, a.Phone, a.ExperienceYears

FROM Applicants a

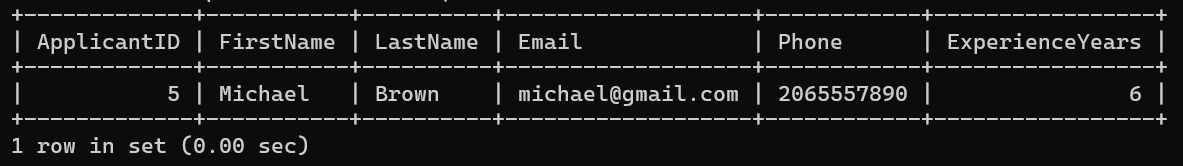
JOIN Applications app ON a.ApplicantID = app.ApplicantID

JOIN Jobs j ON app.JobID = j.JobID

JOIN Companies c ON j.CompanyID = c.CompanyID

WHERE c.Location = 'Seattle'

AND a.ExperienceYears >= 3;

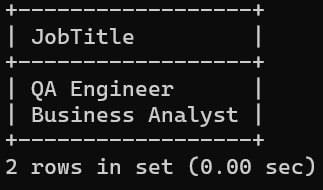
****

**11. Retrieve a list of distinct job titles with salaries between $60,000 and $80,000.**

**🡪** SELECT DISTINCT JobTitle

FROM Jobs

WHERE Salary BETWEEN 60000 AND 80000; **(Inserted new values in jobs)**

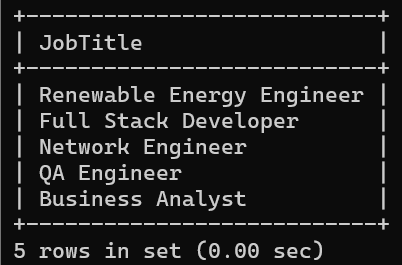


**12. Find the jobs that have not received any applications.**

**🡪** SELECT JobTitle

FROM Jobs

WHERE JobID NOT IN (SELECT DISTINCT JobID FROM Applications);



**13. Retrieve a list of job applicants along with the companies they have applied to and the positions they have applied for.**

**🡪** SELECT a.ApplicantID, a.FirstName, a.LastName, c.CompanyName, j.JobTitle

FROM Applications app

JOIN Applicants a ON app.ApplicantID = a.ApplicantID

JOIN Jobs j ON app.JobID = j.JobID

JOIN Companies c ON j.CompanyID = c.CompanyID;



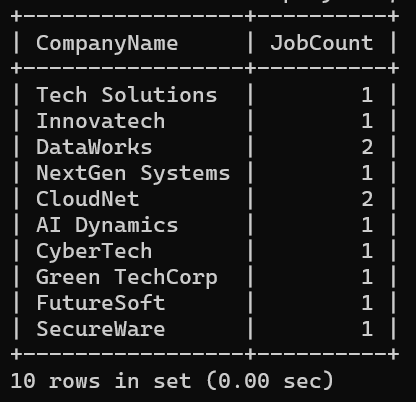
**14. Retrieve a list of companies along with the count of jobs they have posted, even if they have not received any applications.**

**🡪** SELECT c.CompanyName, COUNT(j.JobID) AS JobCount

FROM Companies c

LEFT JOIN Jobs j ON c.CompanyID = j.CompanyID

GROUP BY c.CompanyName;

****

**15. List all applicants along with the companies and positions they have applied for, including those who have not applied.**

**🡪** SELECT a.FirstName, a.LastName, COALESCE(c.CompanyName, 'No Application') AS Company, COALESCE(j.JobTitle, 'No Application') AS JobTitle

FROM Applicants a

LEFT JOIN Applications app ON a.ApplicantID = app.ApplicantID

LEFT JOIN Jobs j ON app.JobID = j.JobID

LEFT JOIN Companies c ON j.CompanyID = c.CompanyID;

****

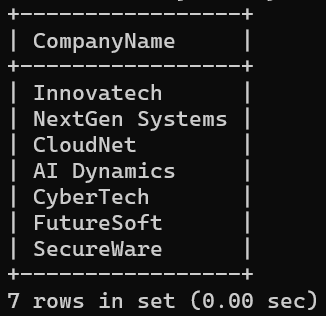
**16. Find companies that have posted jobs with a salary higher than the average salary of all jobs.**

**🡪** SELECT DISTINCT c.CompanyName

FROM Jobs j

JOIN Companies c ON j.CompanyID = c.CompanyID

WHERE j.Salary > (SELECT AVG(Salary) FROM Jobs);

****

**17. Display a list of applicants with their names and a concatenated string of their city and state.**

**🡪** SELECT ApplicantID, FirstName, LastName,

CONCAT(City, ', ', State) AS Location

FROM Applicants;



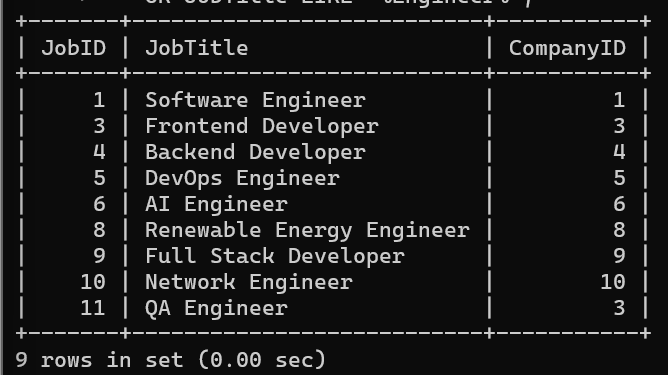
**18. Retrieve a list of jobs with titles containing either 'Developer' or 'Engineer'.**

**🡪** SELECT JobID, JobTitle, CompanyID

FROM Jobs

WHERE JobTitle LIKE '%Developer%'

OR JobTitle LIKE '%Engineer%';



**19. Retrieve a list of applicants and the jobs they have applied for, including those who have not applied and jobs without applicants.**

**🡪** SELECT a.FirstName, a.LastName, COALESCE(j.JobTitle, 'No Application') AS JobTitle

FROM Applicants a

LEFT JOIN Applications app ON a.ApplicantID = app.ApplicantID

LEFT JOIN Jobs j ON app.JobID = j.JobID

UNION

SELECT COALESCE(a.FirstName, 'No Applicant'), COALESCE(a.LastName, 'No Applicant'), j.JobTitle

FROM Jobs j

LEFT JOIN Applications app ON j.JobID = app.JobID

LEFT JOIN Applicants a ON app.ApplicantID = a.ApplicantID;



**20. List all combinations of applicants and companies where the company is in a specific city and the applicant has more than 2 years of experience. For example: city=Chennai**

**🡪** SELECT a.FirstName, a.LastName, c.CompanyName, c.Location

FROM Applicants a, Companies c

WHERE c.Location = 'Mexico'

AND a.ExperienceYears > 2;

