**Human Resources**

**CREATE TABLE Department (**

**DepartmentID INT PRIMARY KEY,**

**DepartmentName VARCHAR(100),**

**ManagerID INT,**

**Location VARCHAR(100),**

**FOREIGN KEY (ManagerID) REFERENCES Employee(EmployeeID)**

**);**

**CREATE TABLE Employee (**

**EmployeeID INT PRIMARY KEY,**

**FirstName VARCHAR(50),**

**LastName VARCHAR(50),**

**BirthDate DATE,**

**Gender CHAR(1),**

**DepartmentID INT,**

**HireDate DATE,**

**Salary DECIMAL(10, 2),**

**Email VARCHAR(100),**

**Phone VARCHAR(15),**

**FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)**

**);**

**CREATE TABLE Payroll (**

**PayrollID INT PRIMARY KEY,**

**EmployeeID INT,**

**PayDate DATE,**

**Salary DECIMAL(10, 2),**

**Deductions DECIMAL(10, 2),**

**NetSalary DECIMAL(10, 2),**

**FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)**

**);**

**CREATE TABLE JobApplicants (**

**ApplicantID INT PRIMARY KEY,**

**FirstName VARCHAR(50),**

**LastName VARCHAR(50),**

**Email VARCHAR(100),**

**Phone VARCHAR(15),**

**PositionApplied VARCHAR(100),**

**ApplicationDate DATE,**

**Resume BLOB,**

**CoverLetter TEXT**

**);**

**CREATE TABLE TrainingRecords (**

**RecordID INT PRIMARY KEY,**

**EmployeeID INT,**

**TrainingName VARCHAR(100),**

**TrainingDate DATE,**

**TrainerName VARCHAR(100),**

**DurationHours INT,**

**TrainingLocation VARCHAR(100),**

**CertificateIssued BOOLEAN,**

**FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)**

**);**

INSERT INTO Department (DepartmentID, DepartmentName, ManagerID, Location)

VALUES

(1, 'HR', 1, 'New York'),

(2, 'IT', 2, 'San Francisco'),

(3, 'Finance', 3, 'Chicago'),

(4, 'Marketing', 4, 'Los Angeles'),

(5, 'Operations', 5, 'Dallas'),

(6, 'Customer Service', 6, 'Miami'),

(7, 'Research and Development', 7, 'Boston'),

(8, 'Sales', 8, 'Seattle'),

(9, 'Quality Assurance', 9, 'Denver'),

(10, 'Legal', 10, 'Phoenix'),

(11, 'Supply Chain', 11, 'Atlanta'),

(12, 'Public Relations', 12, 'San Diego'),

(13, 'Product Management', 13, 'Austin'),

(14, 'Facilities', 14, 'Houston'),

(15, 'Training', 15, 'San Jose'),

(16, 'Warehouse', 16, 'Detroit'),

(17, 'Security', 17, 'Portland'),

(18, 'Information Security', 18, 'Minneapolis'),

(19, 'Logistics', 19, 'Tampa'),

(20, 'Procurement', 20, 'St. Louis');

INSERT INTO Employee (EmployeeID, FirstName, LastName, BirthDate, Gender, DepartmentID, HireDate, Salary, Email, Phone)

VALUES

(1, 'John', 'Doe', '1985-05-15', 'M', 1, '2010-03-20', 65000.00, 'john.doe@example.com', '555-123-4567'),

(2, 'Jane', 'Smith', '1990-11-25', 'F', 2, '2012-07-10', 55000.00, 'jane.smith@example.com', '555-987-6543'),

(3, 'David', 'Johnson', '1988-08-10', 'M', 1, '2011-02-15', 62000.00, 'david.j@example.com', '555-456-7890'),

(4, 'Sarah', 'Williams', '1992-03-05', 'F', 3, '2014-05-30', 58000.00, 'sarah.w@example.com', '555-789-0123'),

(5, 'Michael', 'Brown', '1987-12-20', 'M', 2, '2013-09-12', 60000.00, 'michael.b@example.com', '555-234-5678'),

(6, 'Emily', 'Davis', '1991-06-18', 'F', 1, '2015-11-05', 63000.00, 'emily.d@example.com', '555-345-6789'),

(7, 'Daniel', 'Lee', '1986-02-28', 'M', 3, '2016-04-22', 58000.00, 'daniel.l@example.com', '555-456-7890'),

(8, 'Olivia', 'Martinez', '1994-09-14', 'F', 2, '2018-03-17', 59000.00, 'olivia.m@example.com', '555-567-8901'),

(9, 'William', 'Garcia', '1989-07-07', 'M', 1, '2019-08-29', 64000.00, 'william.g@example.com', '555-678-9012'),

(10, 'Sophia', 'Harris', '1993-04-03', 'F', 2, '2020-12-10', 55000.00, 'sophia.h@example.com', '555-789-0123'),

(11, 'Alexander', 'Walker', '1990-01-12', 'M', 3, '2017-06-07', 60000.00, 'alex.w@example.com', '555-890-1234'),

(12, 'Ava', 'Thompson', '1988-10-26', 'F', 1, '2014-08-14', 61000.00, 'ava.t@example.com', '555-901-2345'),

(13, 'James', 'Lopez', '1987-03-17', 'M', 2, '2012-09-25', 59000.00, 'james.l@example.com', '555-012-3456'),

(14, 'Mia', 'Wilson', '1995-07-30', 'F', 1, '2016-01-30', 57000.00, 'mia.w@example.com', '555-123-4567'),

(15, 'Benjamin', 'Anderson', '1986-11-08', 'M', 3, '2013-05-18', 63000.00, 'benjamin.a@example.com', '555-234-5678'),

(16, 'Ella', 'Thomas', '1991-04-22', 'F', 2, '2019-02-01', 56000.00, 'ella.t@example.com', '555-345-6789'),

(17, 'Joseph', 'Hill', '1989-09-03', 'M', 1, '2017-10-12', 61000.00, 'joseph.h@example.com', '555-456-7890'),

(18, 'Sofia', 'Clark', '1993-02-19', 'F', 2, '2015-12-05', 58000.00, 'sofia.c@example.com', '555-567-8901'),

(19, 'Samuel', 'Young', '1988-06-14', 'M', 3, '2018-04-20', 59000.00, 'samuel.y@example.com', '555-678-9012'),

(20, 'Chloe', 'Adams', '1994-01-07', 'F', 1, '2019-09-15', 62000.00, 'chloe.a@example.com', '555-789-0123');

INSERT INTO Payroll (PayrollID, EmployeeID, PayDate, Salary, Deductions, NetSalary)

VALUES

(1, 1, '2023-01-31', 6500.00, 500.00, 6000.00),

(2, 2, '2023-01-31', 5500.00, 400.00, 5100.00),

(3, 3, '2023-01-31', 6200.00, 600.00, 5600.00),

(4, 4, '2023-01-31', 5800.00, 450.00, 5350.00),

(5, 5, '2023-01-31', 6000.00, 550.00, 5450.00),

(6, 6, '2023-01-31', 6300.00, 500.00, 5800.00),

(7, 7, '2023-01-31', 5800.00, 400.00, 5400.00),

(8, 8, '2023-01-31', 5900.00, 450.00, 5450.00),

(9, 9, '2023-01-31', 6400.00, 550.00, 5850.00),

(10, 10, '2023-01-31', 5500.00, 400.00, 5100.00),

(11, 11, '2023-01-31', 6000.00, 500.00, 5500.00),

(12, 12, '2023-01-31', 6100.00, 450.00, 5650.00),

(13, 13, '2023-01-31', 5900.00, 400.00, 5500.00),

(14, 14, '2023-01-31', 5700.00, 350.00, 5350.00),

(15, 15, '2023-01-31', 6300.00, 550.00, 5750.00),

(16, 16, '2023-01-31', 5600.00, 400.00, 5200.00),

(17, 17, '2023-01-31', 6100.00, 450.00, 5650.00),

(18, 18, '2023-01-31', 5800.00, 400.00, 5400.00),

(19, 19, '2023-01-31', 5900.00, 450.00, 5450.00),

(20, 20, '2023-01-31', 6200.00, 500.00, 5700.00);

INSERT INTO JobApplicants (ApplicantID, FirstName, LastName, Email, Phone, PositionApplied, ApplicationDate, Resume, CoverLetter)

VALUES

(1, 'Mark', 'Johnson', 'mark.johnson@example.com', '555-222-3333', 'Software Engineer', '2023-08-15', NULL, 'I am excited to join your team.'),

(2, 'Emily', 'Davis', 'emily.davis@example.com', '555-444-5555', 'Data Analyst', '2023-08-20', NULL, 'I have a strong analytical background.'),

(3, 'Daniel', 'Brown', 'daniel.brown@example.com', '555-666-7777', 'HR Specialist', '2023-08-25', NULL, 'I am passionate about HR.'),

(4, 'Olivia', 'Miller', 'olivia.miller@example.com', '555-888-9999', 'Marketing Manager', '2023-08-30', NULL, 'I have a proven track record in marketing.'),

(5, 'James', 'Smith', 'james.smith@example.com', '555-111-2222', 'IT Analyst', '2023-09-05', NULL, 'I am skilled in IT operations.'),

(6, 'Sophia', 'Garcia', 'sophia.garcia@example.com', '555-333-4444', 'Financial Analyst', '2023-09-10', NULL, 'I have a finance background.'),

(7, 'William', 'Johnson', 'william.johnson@example.com', '555-555-6666', 'Software Engineer', '2023-09-15', NULL, 'I have coding experience.'),

(8, 'Mia', 'Harris', 'mia.harris@example.com', '555-777-8888', 'Data Scientist', '2023-09-20', NULL, 'I am a data analysis expert.'),

(9, 'Benjamin', 'Thompson', 'benjamin.thompson@example.com', '555-999-0000', 'HR Specialist', '2023-09-25', NULL, 'I have HR experience.'),

(10, 'Ella', 'Davis', 'ella.davis@example.com', '555-222-3333', 'Marketing Coordinator', '2023-09-30', NULL, 'I am creative in marketing.'),

(11, 'Joseph', 'Smith', 'joseph.smith@example.com', '555-444-5555', 'IT Specialist', '2023-10-05', NULL, 'I am a problem solver in IT.'),

(12, 'Sofia', 'Lopez', 'sofia.lopez@example.com', '555-666-7777', 'Finance Manager', '2023-10-10', NULL, 'I am experienced in finance.'),

(13, 'Samuel', 'Williams', 'samuel.williams@example.com', '555-888-9999', 'Software Developer', '2023-10-15', NULL, 'I am passionate about coding.'),

(14, 'Chloe', 'Martin', 'chloe.martin@example.com', '555-111-2222', 'Data Analyst', '2023-10-20', NULL, 'I am detail-oriented in data analysis.'),

(15, 'David', 'Johnson', 'david.johnson@example.com', '555-333-4444', 'HR Specialist', '2023-10-25', NULL, 'I am dedicated to HR.'),

(16, 'Oliver', 'Brown', 'oliver.brown@example.com', '555-555-6666', 'Marketing Coordinator', '2023-10-30', NULL, 'I am a marketing enthusiast.'),

(17, 'Sophia', 'Walker', 'sophia.walker@example.com', '555-777-8888', 'IT Specialist', '2023-11-05', NULL, 'I am an IT expert.'),

(18, 'Ella', 'Hill', 'ella.hill@example.com', '555-999-0000', 'Financial Analyst', '2023-11-10', NULL, 'I am a financial expert.'),

(19, 'Liam', 'Clark', 'liam.clark@example.com', '555-222-3333', 'Software Engineer', '2023-11-15', NULL, 'I am skilled in software development.'),

(20, 'Ava', 'Garcia', 'ava.garcia@example.com', '555-444-5555', 'Data Scientist', '2023-11-20', NULL, 'I am a data science professional.');

INSERT INTO TrainingRecords (RecordID, EmployeeID, TrainingName, TrainingDate, TrainerName, DurationHours, TrainingLocation, CertificateIssued)

VALUES

(1, 1, 'Java Programming', '2023-03-10', 'John Trainer', 40, 'New York', TRUE),

(2, 2, 'SQL Fundamentals', '2023-04-05', 'Emily Trainer', 32, 'San Francisco', TRUE),

(3, 3, 'HR Policies', '2023-04-15', 'David Trainer', 24, 'Chicago', TRUE),

(4, 4, 'Marketing Strategies', '2023-05-02', 'Sarah Trainer', 36, 'Los Angeles', TRUE),

(5, 5, 'IT Security', '2023-05-20', 'Michael Trainer', 28, 'San Francisco', TRUE),

(6, 6, 'Financial Analysis', '2023-06-08', 'Emily Trainer', 40, 'New York', TRUE),

(7, 7, 'Leadership Skills', '2023-06-25', 'Daniel Trainer', 32, 'Chicago', TRUE),

(8, 8, 'Digital Marketing', '2023-07-12', 'Olivia Trainer', 36, 'Los Angeles', TRUE),

(9, 9, 'Conflict Resolution', '2023-07-30', 'William Trainer', 24, 'San Francisco', TRUE),

(10, 10, 'Database Administration', '2023-08-10', 'Sophia Trainer', 40, 'New York', TRUE),

(11, 11, 'Agile Development', '2023-08-28', 'Alexander Trainer', 32, 'Chicago', TRUE),

(12, 12, 'Performance Evaluation', '2023-09-15', 'Ava Trainer', 36, 'Los Angeles', TRUE),

(13, 13, 'Networking Fundamentals', '2023-09-30', 'James Trainer', 28, 'San Francisco', TRUE),

(14, 14, 'Data Visualization', '2023-10-15', 'Mia Trainer', 40, 'New York', TRUE),

(15, 15, 'Project Management', '2023-10-28', 'Benjamin Trainer', 32, 'Chicago', TRUE),

(16, 16, 'Web Development', '2023-11-10', 'Ella Trainer', 36, 'Los Angeles', TRUE),

(17, 17, 'Financial Modeling', '2023-11-25', 'Joseph Trainer', 24, 'San Francisco', TRUE),

(18, 18, 'Software Testing', '2023-12-10', 'Sofia Trainer', 40, 'New York', TRUE),

(19, 19, 'Data Mining', '2023-12-22', 'Samuel Trainer', 32, 'Chicago', TRUE),

(20, 20, 'Cloud Computing', '2024-01-05', 'Chloe Trainer', 36, 'Los Angeles', TRUE);

**Exercise 1: Establish a JDBC Connection** Write a Java program to establish a JDBC connection to your HR database. Use the appropriate JDBC driver and connection URL.

**Answer 1:**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class JDBCTest {

public static void main(String[] args) {

String jdbcURL = "jdbc:mysql://localhost:3306/hrdb";

String username = "your\_username";

String password = "your\_password";

try {

Connection connection = DriverManager.getConnection(jdbcURL, username, password);

if (connection != null) {

System.out.println("Connected to the database!");

connection.close();

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 2: Retrieve Employee Data** Write a Java program that connects to the HR database, retrieves all employee records from the Employee table, and displays them in the console.

**Answer 2:**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.Statement;

import java.sql.SQLException;

public class RetrieveEmployees {

public static void main(String[] args) {

String jdbcURL = "jdbc:mysql://localhost:3306/hrdb";

String username = "your\_username";

String password = "your\_password";

try {

Connection connection = DriverManager.getConnection(jdbcURL, username, password);

Statement statement = connection.createStatement();

String sql = "SELECT \* FROM Employee";

ResultSet resultSet = statement.executeQuery(sql);

while (resultSet.next()) {

int employeeID = resultSet.getInt("EmployeeID");

String firstName = resultSet.getString("FirstName");

String lastName = resultSet.getString("LastName");

// Retrieve and display other fields

System.out.println("Employee ID: " + employeeID);

System.out.println("Name: " + firstName + " " + lastName);

// Display other fields as needed

}

resultSet.close();

statement.close();

connection.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 3: Insert Employee Data** Write a Java program to insert a new employee record into the Employee table of the HR database. Prompt the user for employee details.

**Answer 3:**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.util.Scanner;

public class InsertEmployee {

public static void main(String[] args) {

String jdbcURL = "jdbc:mysql://localhost:3306/hrdb";

String username = "your\_username";

String password = "your\_password";

try {

Connection connection = DriverManager.getConnection(jdbcURL, username, password);

Scanner scanner = new Scanner(System.in);

System.out.println("Enter employee details:");

System.out.print("First Name: ");

String firstName = scanner.nextLine();

System.out.print("Last Name: ");

String lastName = scanner.nextLine();

// Prompt for other fields and retrieve input

String sql = "INSERT INTO Employee (FirstName, LastName) VALUES (?, ?)";

PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.setString(1, firstName);

preparedStatement.setString(2, lastName);

// Set other parameters as needed

int rowsAffected = preparedStatement.executeUpdate();

if (rowsAffected > 0) {

System.out.println("Employee record inserted successfully.");

}

preparedStatement.close();

connection.close();

scanner.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 4: Update Employee Data** Write a Java program that updates the salary of an employee in the Employee table. Prompt the user for the employee's ID and the new salary.

**Answer 4:**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.util.Scanner;

public class UpdateSalary {

public static void main(String[] args) {

String jdbcURL = "jdbc:mysql://localhost:3306/hrdb";

String username = "your\_username";

String password = "your\_password";

try {

Connection connection = DriverManager.getConnection(jdbcURL, username, password);

Scanner scanner = new Scanner(System.in);

System.out.print("Enter Employee ID: ");

int employeeID = scanner.nextInt();

System.out.print("Enter New Salary: ");

double newSalary = scanner.nextDouble();

String sql = "UPDATE Employee SET Salary = ? WHERE EmployeeID = ?";

PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.setDouble(1, newSalary);

preparedStatement.setInt(2, employeeID);

int rowsAffected = preparedStatement.executeUpdate();

if (rowsAffected > 0) {

System.out.println("Salary updated successfully.");

} else {

System.out.println("Employee not found or no updates were made.");

}

preparedStatement.close();

connection.close();

scanner.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 5: Delete Employee Data** Write a Java program that deletes an employee record from the Employee table based on the employee's ID. Prompt the user for the ID of the employee to be deleted.

**Answer 5:**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.util.Scanner;

public class DeleteEmployee {

public static void main(String[] args) {

String jdbcURL = "jdbc:mysql://localhost:3306/hrdb";

String username = "your\_username";

String password = "your\_password";

try {

Connection connection = DriverManager.getConnection(jdbcURL, username, password);

Scanner scanner = new Scanner(System.in);

System.out.print("Enter Employee ID to Delete: ");

int employeeID = scanner.nextInt();

String sql = "DELETE FROM Employee WHERE EmployeeID = ?";

PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.setInt(1, employeeID);

int rowsAffected = preparedStatement.executeUpdate();

if (rowsAffected > 0) {

System.out.println("Employee record deleted successfully.");

} else {

System.out.println("Employee not found or no deletions were made.");

}

preparedStatement.close();

connection.close();

scanner.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 6: Retrieve Employee Data by Department** Write a Java program that connects to the HR database and retrieves all employee records from the "Employee" table for a specific department. Prompt the user to enter the department name, and then display the employee records for that department.

**Answer 6:**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.Scanner;

public class RetrieveEmployeesByDepartment {

public static void main(String[] args) {

String jdbcURL = "jdbc:mysql://localhost:3306/hrdb";

String username = "your\_username";

String password = "your\_password";

try {

Connection connection = DriverManager.getConnection(jdbcURL, username, password);

Scanner scanner = new Scanner(System.in);

System.out.print("Enter Department Name: ");

String departmentName = scanner.nextLine();

String sql = "SELECT \* FROM Employee WHERE DepartmentID IN (SELECT DepartmentID FROM Department WHERE DepartmentName = ?)";

PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.setString(1, departmentName);

ResultSet resultSet = preparedStatement.executeQuery();

while (resultSet.next()) {

int employeeID = resultSet.getInt("EmployeeID");

String firstName = resultSet.getString("FirstName");

String lastName = resultSet.getString("LastName");

// Retrieve and display other fields

System.out.println("Employee ID: " + employeeID);

System.out.println("Name: " + firstName + " " + lastName);

// Display other fields as needed

}

resultSet.close();

preparedStatement.close();

connection.close();

scanner.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 7: Calculate Total Salary Expense** Write a Java program that connects to the HR database and calculates the total salary expense for a given department. Prompt the user to enter the department name, and then display the total salary expense.

**Answer 7:**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.Scanner;

public class CalculateTotalSalaryExpense {

public static void main(String[] args) {

String jdbcURL = "jdbc:mysql://localhost:3306/hrdb";

String username = "your\_username";

String password = "your\_password";

try {

Connection connection = DriverManager.getConnection(jdbcURL, username, password);

Scanner scanner = new Scanner(System.in);

System.out.print("Enter Department Name: ");

String departmentName = scanner.nextLine();

String sql = "SELECT SUM(Salary) AS TotalSalary FROM Employee WHERE DepartmentID IN (SELECT DepartmentID FROM Department WHERE DepartmentName = ?)";

PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.setString(1, departmentName);

ResultSet resultSet = preparedStatement.executeQuery();

if (resultSet.next()) {

double totalSalary = resultSet.getDouble("TotalSalary");

System.out.println("Total Salary Expense for " + departmentName + ": $" + totalSalary);

} else {

System.out.println("Department not found or no employees in the department.");

}

resultSet.close();

preparedStatement.close();

connection.close();

scanner.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 8: Retrieve Job Applicants** Write a Java program that connects to the HR database and retrieves all job applicants from the "JobApplicants" table. Display the applicant details.

**Answer 8:**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

public class RetrieveJobApplicants {

public static void main(String[] args) {

String jdbcURL = "jdbc:mysql://localhost:3306/hrdb";

String username = "your\_username";

String password = "your\_password";

try {

Connection connection = DriverManager.getConnection(jdbcURL, username, password);

String sql = "SELECT \* FROM JobApplicants";

PreparedStatement preparedStatement = connection.prepareStatement(sql);

ResultSet resultSet = preparedStatement.executeQuery();

while (resultSet.next()) {

int applicantID = resultSet.getInt("ApplicantID");

String firstName = resultSet.getString("FirstName");

String lastName = resultSet.getString("LastName");

String email = resultSet.getString("Email");

String phone = resultSet.getString("Phone");

// Retrieve and display other fields

System.out.println("Applicant ID: " + applicantID);

System.out.println("Name: " + firstName + " " + lastName);

System.out.println("Email: " + email);

System.out.println("Phone: " + phone);

// Display other fields as needed

}

resultSet.close();

preparedStatement.close();

connection.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 9: Insert Job Applicant** Write a Java program that inserts a new job applicant record into the "JobApplicants" table. Prompt the user for applicant details.

**Answer 9:**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.util.Scanner;

public class InsertJobApplicant {

public static void main(String[] args) {

String jdbcURL = "jdbc:mysql://localhost:3306/hrdb";

String username = "your\_username";

String password = "your\_password";

try {

Connection connection = DriverManager.getConnection(jdbcURL, username, password);

Scanner scanner = new Scanner(System.in);

System.out.println("Enter job applicant details:");

System.out.print("First Name: ");

String firstName = scanner.nextLine();

System.out.print("Last Name: ");

String lastName = scanner.nextLine();

System.out.print("Email: ");

String email = scanner.nextLine();

System.out.print("Phone: ");

String phone = scanner.nextLine();

// Prompt for other fields and retrieve input

String sql = "INSERT INTO JobApplicants (FirstName, LastName, Email, Phone) VALUES (?, ?, ?, ?)";

PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.setString(1, firstName);

preparedStatement.setString(2, lastName);

preparedStatement.setString(3, email);

preparedStatement.setString(4, phone);

// Set other parameters as needed

int rowsAffected = preparedStatement.executeUpdate();

if (rowsAffected > 0) {

System.out.println("Job applicant record inserted successfully.");

}

preparedStatement.close();

connection.close();

scanner.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 10: Retrieve Training Records** Write a Java program that connects to the HR database and retrieves all training records from the "TrainingRecords" table. Display the training details.

**Answer 10:**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

public class RetrieveTrainingRecords {

public static void main(String[] args) {

String jdbcURL = "jdbc:mysql://localhost:3306/hrdb";

String username = "your\_username";

String password = "your\_password";

try {

Connection connection = DriverManager.getConnection(jdbcURL, username, password);

String sql = "SELECT \* FROM TrainingRecords";

PreparedStatement preparedStatement = connection.prepareStatement(sql);

ResultSet resultSet = preparedStatement.executeQuery();

while (resultSet.next()) {

int recordID = resultSet.getInt("RecordID");

int employeeID = resultSet.getInt("EmployeeID");

String trainingName = resultSet.getString("TrainingName");

String trainingDate = resultSet.getString("TrainingDate");

String trainerName = resultSet.getString("TrainerName");

int durationHours = resultSet.getInt("DurationHours");

String trainingLocation = resultSet.getString("TrainingLocation");

boolean certificateIssued = resultSet.getBoolean("CertificateIssued");

// Retrieve and display other fields

System.out.println("Record ID: " + recordID);

System.out.println("Employee ID: " + employeeID);

System.out.println("Training Name: " + trainingName);

System.out.println("Training Date: " + trainingDate);

System.out.println("Trainer Name: " + trainerName);

System.out.println("Duration (Hours): " + durationHours);

System.out.println("Location: " + trainingLocation);

System.out.println("Certificate Issued: " + certificateIssued);

// Display other fields as needed

}

resultSet.close();

preparedStatement.close();

connection.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 11: Retrieve Employee with Highest Salary**

**Question: Write a Java program that connects to the HR database and retrieves the employee with the highest salary. Display the details of that employee.**

import java.sql.\*;

public class Exercise11 {

public static void main(String[] args) {

String url = "jdbc:mysql://localhost:3306/jdbc\_exercises";

String user = "your\_username";

String password = "your\_password";

try (Connection connection = DriverManager.getConnection(url, user, password)) {

Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery("SELECT \* FROM Employee ORDER BY Salary DESC LIMIT 1");

if (resultSet.next()) {

System.out.println("Employee with Highest Salary:");

System.out.println("ID: " + resultSet.getInt("EmployeeID"));

System.out.println("Name: " + resultSet.getString("FirstName") + " " + resultSet.getString("LastName"));

System.out.println("Salary: " + resultSet.getBigDecimal("Salary"));

} else {

System.out.println("No employees found.");

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 12: Average Salary by Department**

**Question: Write a Java program that connects to the HR database and calculates the average salary for each department. Display the department name and its average salary.**

import java.sql.\*;

public class Exercise12 {

public static void main(String[] args) {

String url = "jdbc:mysql://localhost:3306/jdbc\_exercises";

String user = "your\_username";

String password = "your\_password";

try (Connection connection = DriverManager.getConnection(url, user, password)) {

Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery(

"SELECT Department.DepartmentName, AVG(Employee.Salary) AS AvgSalary " +

"FROM Employee " +

"JOIN Department ON Employee.DepartmentID = Department.DepartmentID " +

"GROUP BY Department.DepartmentName");

while (resultSet.next()) {

System.out.println("Department: " + resultSet.getString("DepartmentName"));

System.out.println("Average Salary: " + resultSet.getBigDecimal("AvgSalary"));

System.out.println("-----------");

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 13: Update Employee Email**

**Question: Write a Java program that connects to the HR database and updates the email of an employee. Prompt the user for the employee's ID and the new email.**

import java.sql.\*;

import java.util.Scanner;

public class Exercise13 {

public static void main(String[] args) {

String url = "jdbc:mysql://localhost:3306/jdbc\_exercises";

String user = "your\_username";

String password = "your\_password";

try (Connection connection = DriverManager.getConnection(url, user, password)) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter Employee ID: ");

int employeeID = scanner.nextInt();

scanner.nextLine(); // Consume the newline character

System.out.print("Enter New Email: ");

String newEmail = scanner.nextLine();

String updateQuery = "UPDATE Employee SET Email = ? WHERE EmployeeID = ?";

try (PreparedStatement preparedStatement = connection.prepareStatement(updateQuery)) {

preparedStatement.setString(1, newEmail);

preparedStatement.setInt(2, employeeID);

int rowsAffected = preparedStatement.executeUpdate();

if (rowsAffected > 0) {

System.out.println("Email updated successfully.");

} else {

System.out.println("Employee not found or email not updated.");

}

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 14: Delete Training Record**

**Question: Write a Java program that connects to the HR database and deletes a training record based on the training record ID. Prompt the user for the ID of the training record to be deleted.**

import java.sql.\*;

import java.util.Scanner;

public class Exercise14 {

public static void main(String[] args) {

String url = "jdbc:mysql://localhost:3306/jdbc\_exercises";

String user = "your\_username";

String password = "your\_password";

try (Connection connection = DriverManager.getConnection(url, user, password)) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter Training Record ID to Delete: ");

int recordID = scanner.nextInt();

String deleteQuery = "DELETE FROM TrainingRecords WHERE RecordID = ?";

try (PreparedStatement preparedStatement = connection.prepareStatement(deleteQuery)) {

preparedStatement.setInt(1, recordID);

int rowsAffected = preparedStatement.executeUpdate();

if (rowsAffected > 0) {

System.out.println("Training record deleted successfully.");

} else {

System.out.println("Training record not found or not deleted.");

}

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 15: Retrieve Employees without Training Certificate**

**Question: Write a Java program that connects to the HR database and retrieves employees who do not have a training certificate. Display their details.**

import java.sql.\*;

public class Exercise15 {

public static void main(String[] args) {

String url = "jdbc:mysql://localhost:3306/jdbc\_exercises";

String user = "your\_username";

String password = "your\_password";

try (Connection connection = DriverManager.getConnection(url, user, password)) {

Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery(

"SELECT \* FROM Employee " +

"WHERE EmployeeID NOT IN (SELECT EmployeeID FROM TrainingRecords WHERE CertificateIssued = TRUE)");

while (resultSet.next()) {

System.out.println("Employee ID: " + resultSet.getInt("EmployeeID"));

System.out.println("Name: " + resultSet.getString("FirstName") + " " + resultSet.getString("LastName"));

System.out.println("-----------");

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 16: Total Deductions by Employee**

**Question: Write a Java program that connects to the HR database and calculates the total deductions for a given employee. Prompt the user for the employee's ID and display the total deductions.**

import java.sql.\*;

import java.util.Scanner;

public class Exercise16 {

public static void main(String[] args) {

String url = "jdbc:mysql://localhost:3306/jdbc\_exercises";

String user = "your\_username";

String password = "your\_password";

try (Connection connection = DriverManager.getConnection(url, user, password)) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter Employee ID: ");

int employeeID = scanner.nextInt();

String totalDeductionsQuery = "SELECT SUM(Deductions) AS TotalDeductions FROM Payroll WHERE EmployeeID = ?";

try (PreparedStatement preparedStatement = connection.prepareStatement(totalDeductionsQuery)) {

preparedStatement.setInt(1, employeeID);

ResultSet resultSet = preparedStatement.executeQuery();

if (resultSet.next()) {

System.out.println("Total Deductions for Employee ID " + employeeID + ": " +

resultSet.getBigDecimal("TotalDeductions"));

} else {

System.out.println("Employee not found or no deductions.");

}

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 17: Employee Birthdays This Month**

**Question: Write a Java program that connects to the HR database and retrieves employees whose birthdays fall in the current month. Display their details.**

import java.sql.\*;

import java.time.LocalDate;

import java.time.Month;

public class Exercise17 {

public static void main(String[] args) {

String url = "jdbc:mysql://localhost:3306/jdbc\_exercises";

String user = "your\_username";

String password = "your\_password";

try (Connection connection = DriverManager.getConnection(url, user, password)) {

LocalDate currentDate = LocalDate.now();

int currentMonth = currentDate.getMonthValue();

String birthdaysQuery = "SELECT \* FROM Employee " +

"WHERE MONTH(BirthDate) = ?";

try (PreparedStatement preparedStatement = connection.prepareStatement(birthdaysQuery)) {

preparedStatement.setInt(1, currentMonth);

ResultSet resultSet = preparedStatement.executeQuery();

while (resultSet.next()) {

System.out.println("Employee ID: " + resultSet.getInt("EmployeeID"));

System.out.println("Name: " + resultSet.getString("FirstName") + " " + resultSet.getString("LastName"));

System.out.println("Birth Date: " + resultSet.getDate("BirthDate"));

System.out.println("-----------");

}

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 18: Employees with Most Training Certificates**

**Question: Write a Java program that connects to the HR database and retrieves employees with the most training certificates. Display their details.**

import java.sql.\*;

public class Exercise18 {

public static void main(String[] args) {

String url = "jdbc:mysql://localhost:3306/jdbc\_exercises";

String user = "your\_username";

String password = "your\_password";

try (Connection connection = DriverManager.getConnection(url, user, password)) {

Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery(

"SELECT Employee.EmployeeID, Employee.FirstName, Employee.LastName, COUNT(TrainingRecords.RecordID) AS CertificatesCount " +

"FROM Employee " +

"JOIN TrainingRecords ON Employee.EmployeeID = TrainingRecords.EmployeeID " +

"GROUP BY Employee.EmployeeID " +

"ORDER BY CertificatesCount DESC LIMIT 1");

if (resultSet.next()) {

System.out.println("Employee with Most Training Certificates:");

System.out.println("ID: " + resultSet.getInt("EmployeeID"));

System.out.println("Name: " + resultSet.getString("FirstName") + " " + resultSet.getString("LastName"));

System.out.println("Certificates Count: " + resultSet.getInt("CertificatesCount"));

} else {

System.out.println("No employees found with training certificates.");

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 19: Retrieve Employees by Gender**

**Question: Write a Java program that connects to the HR database and retrieves employees based on their gender. Prompt the user for the gender and display the employee details.**

import java.sql.\*;

import java.util.Scanner;

public class Exercise19 {

public static void main(String[] args) {

String url = "jdbc:mysql://localhost:3306/jdbc\_exercises";

String user = "your\_username";

String password = "your\_password";

try (Connection connection = DriverManager.getConnection(url, user, password)) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter Gender (M/F): ");

String gender = scanner.nextLine().toUpperCase();

String employeesByGenderQuery = "SELECT \* FROM Employee WHERE Gender = ?";

try (PreparedStatement preparedStatement = connection.prepareStatement(employeesByGenderQuery)) {

preparedStatement.setString(1, gender);

ResultSet resultSet = preparedStatement.executeQuery();

while (resultSet.next()) {

System.out.println("Employee ID: " + resultSet.getInt("EmployeeID"));

System.out.println("Name: " + resultSet.getString("FirstName") + " " + resultSet.getString("LastName"));

System.out.println("Gender: " + resultSet.getString("Gender"));

System.out.println("-----------");

}

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Exercise 20: Retrieve Employees with No Training**

**Question: Write a Java program that connects to the HR database and retrieves employees who have not undergone any training. Display their details.**

import java.sql.\*;

public class Exercise20 {

public static void main(String[] args) {

String url = "jdbc:mysql://localhost:3306/jdbc\_exercises";

String user = "your\_username";

String password = "your\_password";

try (Connection connection = DriverManager.getConnection(url, user, password)) {

Statement statement = connection.createStatement();

ResultSet resultSet = statement.executeQuery(

"SELECT \* FROM Employee " +

"WHERE EmployeeID NOT IN (SELECT EmployeeID FROM TrainingRecords)");

while (resultSet.next()) {

System.out.println("Employee ID: " + resultSet.getInt("EmployeeID"));

System.out.println("Name: " + resultSet.getString("FirstName") + " " + resultSet.getString("LastName"));

System.out.println("-----------");

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}