12. SQL Challenge: Employee Tracker

Start Assignment

Due Sunday by 11:59pm **Points** 100 **Submitting** a text entry box or a website url

Your Task

Developers frequently have to create interfaces that make it easy for non-developers to view and interact with information stored in databases. These interfaces are called **content management systems (CMS)**. Your challenge this week is to build a command-line application to manage a company's employee database, using Node.js, Inquirer, and MySQL.

Because this application won't be deployed, you'll also need to create a walkthrough video that demonstrates its functionality and all of the following acceptance criteria being met. You'll need to submit a link to the video and add it to the README of your project.

User Story

```
AS A business owner
I WANT to be able to view and manage the departments, roles, and emplo
SO THAT I can organize and plan my business
```

Acceptance Criteria

```
GIVEN a command-line application that accepts user input
WHEN I start the application
THEN I am presented with the following options: view all departments,
WHEN I choose to view all departments
THEN I am presented with a formatted table showing department names ar
WHEN I choose to view all roles
THEN I am presented with the job title, role id, the department that r

WHEN I choose to view all employees
THEN I am presented with a formatted table showing employee data, incl
WHEN I choose to add a department
THEN I am prompted to enter the name of the department and that depart
WHEN I choose to add a role
THEN I am prompted to enter the name, salary, and department for the r
WHEN I choose to add an employee
THEN I am prompted to enter the employee's first name, last name, role
WHEN I choose to update an employee role
THEN I am prompted to select an employee to update and their new role
```

Mock-Up

The following animation shows an example of the application being used from the command line:

Getting Started

You'll need to use the **MySQL2 package (https://www.npmjs.com/package/mysql2)** to connect to your MySQL database and perform queries, the **Inquirer package (https://www.npmjs.com/package/inquirer)** to interact with the user via the command-line, and the **console.table package (https://www.npmjs.com/package/console.table)** to print MySQL rows to the console.

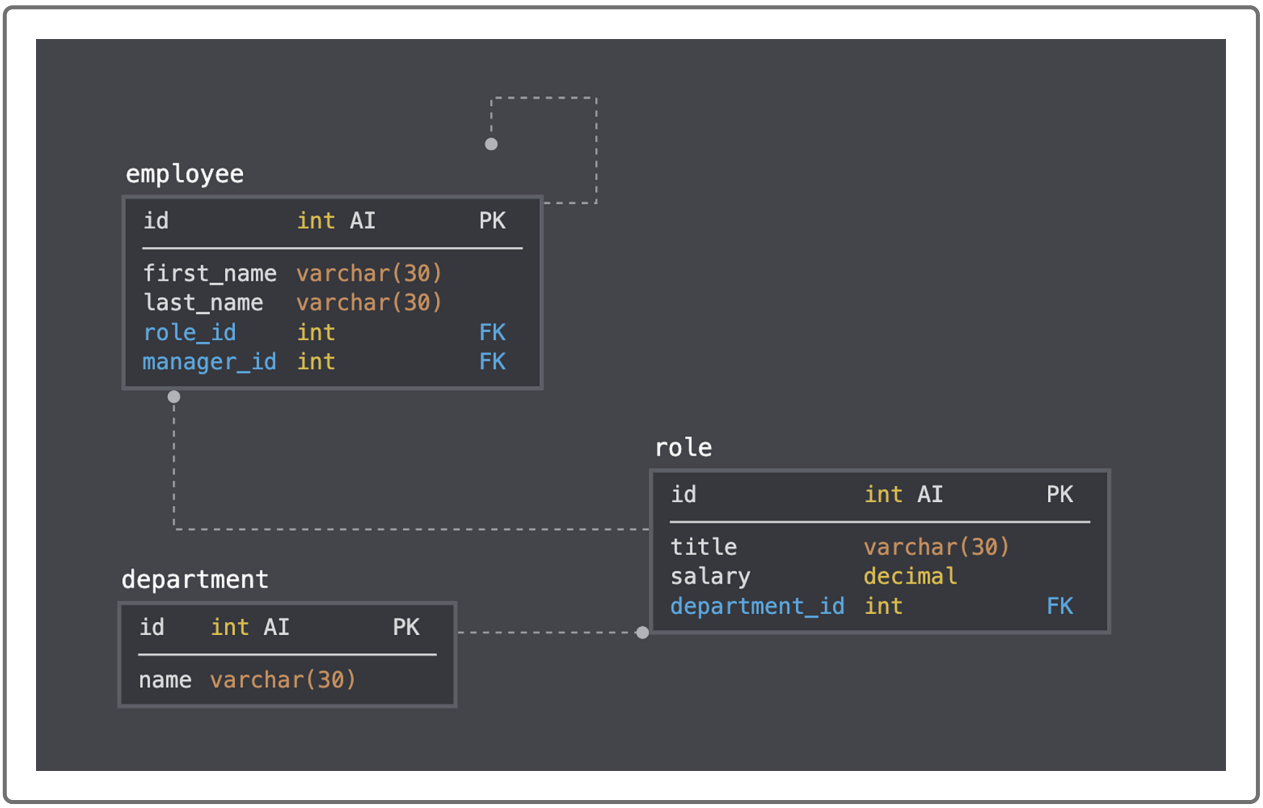
IMPORTANT

> You will be committing a file that contains your database credentials. Make sure your MySQL password is not used for any other personal accounts, because it will be visible on GitHub. In upcoming lessons, you

> will learn how to better secure this password, or you can start researching npm packages now that could help you.

You might also want to make your queries asynchronous. MySQL2 exposes a `.promise()` function on Connections to "upgrade" an existing non-Promise connection to use Promises. Look into **MySQL2's documentation (https://www.npmjs.com/package/mysql2)** to make your queries asynchronous.

Design the database schema as shown in the following image:



As the image illustrates, your schema should contain the following three tables:

- Department

    - `id`: INT PRIMARY KEY
    - `name`: VARCHAR(30) to hold department name

- Role
    - `id`: INT PRIMARY KEY
    - `title`: VARCHAR(30) to hold role title
    - `salary`: DECIMAL to hold role salary
    - `department_id`: INT to hold reference to department role belongs to

- Employee
    - `id`: INT PRIMARY KEY
    - `first_name`: VARCHAR(30) to hold employee first name
    - `last_name`: VARCHAR(30) to hold employee last name
    - `role_id`: INT to hold reference to employee role
    - `manager_id`: INT to hold reference to another employee that is manager of the current employee. This field might be null if the employee has no manager.

You might want to use a separate file containing functions for performing specific SQL queries you'll need to use. A constructor function or class could be helpful for organizing these. You might also want to include a `seeds.sql` file to pre-populate your database. This will make the development of individual features much easier.

Bonus

See if you can add some additional functionality to your application, such as the ability to do the following:

- Update employee managers.
- View employees by manager.
- View employees by department.
- Delete departments, roles, and employees.
- View the total utilized budget of a department—in other words, the combined salaries of all employees in that department.

Grading Requirements

This Challenge is graded based on the following criteria:

Deliverables: 10%

- Your GitHub repository containing your application code.

Walkthrough Video: 27%

- A walkthrough video that demonstrates the functionality of the Employee Tracker must be submitted, and a link to the video should be included in your README file.
- The walkthrough video must show all of the technical acceptance criteria being met.

- The walkthrough video must demonstrate how a user would invoke the application from the command line.

- The walkthrough video must demonstrate a functional menu with the options outlined in the acceptance criteria.

Technical Acceptance Criteria: 40%

- Satisfies all of the preceding acceptance criteria plus the following:

    - Uses the **Inquirer package (https://www.npmjs.com/package/inquirer)** .

    - Uses the **MySQL2 package (https://www.npmjs.com/package/mysql2)** to connect to a MySQL database.

    - Uses the **console.table package (https://www.npmjs.com/package/console.table)** to print MySQL rows to the console.

- Follows the table schema outlined in the Challenge instructions.

Repository Quality: 13%

- Repository has a unique name.

- Repository follows best practices for file structure and naming conventions.

- Repository follows best practices for class/id naming conventions, indentation, quality comments, etc.

- Repository contains multiple descriptive commit messages.

- Repository contains a high-quality README with description and a link to a walkthrough video.

Application Quality 10%

- The application user experience is intuitive and easy to navigate.

How to Submit the Challenge

You are required to submit BOTH of the following for review:

- A walkthrough video demonstrating the functionality of the application.

- The URL of the GitHub repository, with a unique name and a README describing the project.

NOTE

> You are allowed to miss up to two Challenge assignments and still earn your certificate. If you complete all Challenge assignments, your lowest two grades will be dropped. If you wish to skip this assignment, click Submit, then indicate you are skipping by typing “I choose to skip this assignment” in the text box.