# Module 12 Challenge

**Due** Jan 12 by 11:59pm **Points** 100 **Submitting** a text entry box

### **SQL Challenge: Employee Tracker**

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

#### **IMPORTANT**

In order to install (inquirer), please use (npm i inquirer@8.2.4)

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.

Refer to the <u>Video Submission Guide</u> (https://coding-boot-camp.github.io/full-stack/computer-literacy/video-submission-guide) on the Full-Stack Blog for additional guidance on creating a video.

### **User Story**

AS A business owner

I WANT to be able to view and manage the departments, roles, and employees in my company 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, view all roles, view all employees, add a WHEN I choose to view all departments

THEN I am presented with a formatted table showing department names and department ids

WHEN I choose to view all roles

THEN I am presented with the job title, role id, the department that role belongs to, and the salary for that roles. WHEN I choose to view all employees

THEN I am presented with a formatted table showing employee data, including employee ids, first names, last name. WHEN I choose to add a department

THEN I am prompted to enter the name of the department and that department is added to the database. WHEN I choose to add a role

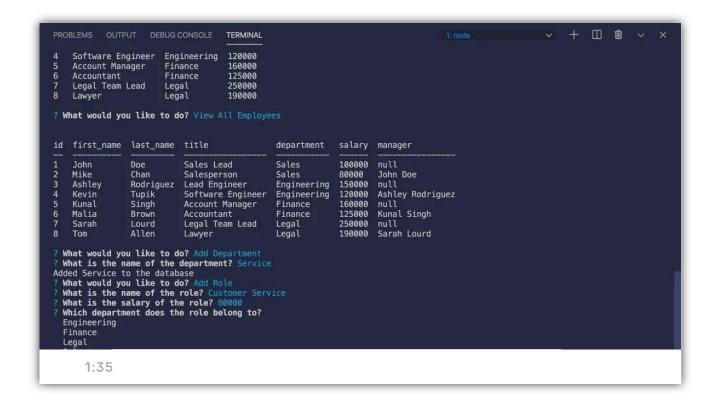
THEN I am prompted to enter the name, salary, and department for the role and that role is added to the database. WHEN I choose to add an employee

THEN I am prompted to enter the employee's first name, last name, role, and manager, and that employee is added when I choose to update an employee role

THEN I am prompted to select an employee to update and their new role and this information is updated in the database.
```

### **Mock-Up**

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



### **Getting Started**

You'll need to use the <a href="MySQL2 package">MySQL2 package</a> (<a href="https://www.npmjs.com/package/mysql2">https://www.npmjs.com/package/mysql2</a>) to connect to your MySQL database and perform queries, the <a href="https://www.npmjs.com/package/inquirer/v/8.2.4">Inquirer package</a> (<a href="https://www.npmjs.com/package/inquirer/v/8.2.4">https://www.npmjs.com/package/inquirer/v/8.2.4</a>) to interact with the user via the command line, and the <a href="mailto:console.table-package">console.table-package</a> (<a href="https://www.npmjs.com/package/console.table">https://www.npmjs.com/package/console.table</a>) to print MySQL rows to the console.

#### **IMPORTANT**

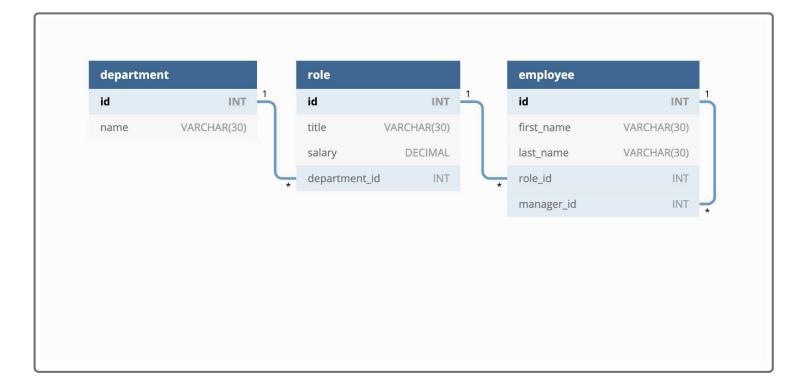
In order to install (inquirer), please use (npm i inquirer@8.2.4)

#### **IMPORTANT**

You will be committing a file that contains your database credentials. Make sure that 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. To learn more and make your queries asynchronous, refer to the npm documentation on MySQL2 (https://www.npmjs.com/package/mysql2).

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 the manager of the current employee (null) if the employee has no manager)

You might want to use a separate file that contains 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, making the development of individual features much easier.

#### **Bonus**

Try to 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**

#### **NOTE**

If a Challenge assignment submission is marked as "0", it is considered incomplete and will not count towards your graduation requirements. Examples of incomplete submissions include the following:

- · A repository that has no code
- A repository that includes a unique name but nothing else
- A repository that includes only a README file but nothing else
- A repository that only includes starter code

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 <u>Inquirer package</u> ⇒ (https://www.npmjs.com/package/inquirer/v/8.2.4)
  - Uses the MySQL2 package (https://www.npmjs.com/package/mysql2) to connect to a MySQL database.
  - Uses the <u>console.table package</u> (https://www.npmjs.com/package/console.table) to print MySQL rows to the console.
- Follows the table schema outlined in the homework 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.

#### **Bonus**

Fulfilling any of the following can add up to 20 points to your grade. Note that the highest grade you can achieve is still 100:

- Application allows users to update employee managers (2 points).
- Application allows users to view employees by manager (2 points).
- Application allows users to view employees by department (2 points).
- Application allows users to delete departments, roles, and employees (2 points for each).
- Application allows users to view the total utilized budget of a department—in other words, the combined salaries of all employees in that department (8 points).

### **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 Next, and move on to the next Module.

Comments are disabled for graded submissions in BootCamp Spot. If you have questions about your feedback, please notify your instructional staff or the Student Success Manager. If you would like to resubmit your work for an improved grade, you can use the Resubmit Assignment button to upload new links. You may resubmit up to three times for a total of four submissions.

Text Entry	
Copy and p	aste or type your submission right here.
Edit Vie	w Insert Format Tools Table
12pt ∨	Paragraph $\vee$ $\mid$
•	
р	<b>(iii)</b> 0 words
P	
Comments	
Cancel	Submit Assignment

© 2023 edX Boot Camps LLC