CCT College Dublin

Assessment Cover Page

Module Title:	Web Development
Assessment Title:	Data Manipulation and Validation
Lecturer Name:	Mikhail Timofeev
Student Full Name:	Halil UGUR
Student Number:	2022389
Assessment Due Date:	28/10/2022 @23:59
Date of Submission:	23.10.2022

Declaration

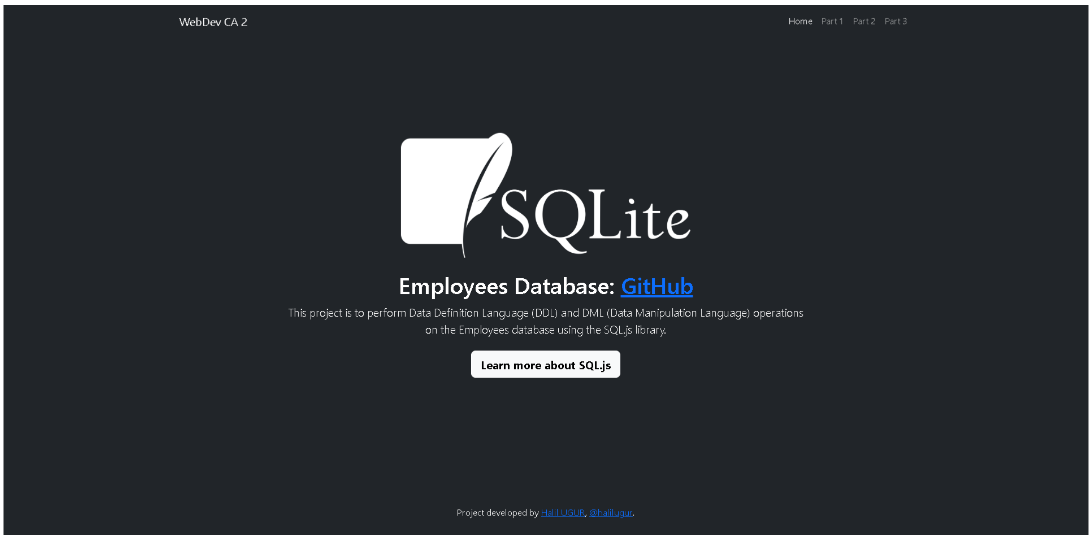
Contents

Abstract

This project is to perform Data Definition Language (DDL) and DML (Data Manipulation Language) operations on the employees database using the SQL.js library. It has been developed as user-friendly by using bootstrap, codemirror and code-prettify in the design of the interface. The project has been developed in 3 parts and the results are given with the visuals.

Introduction

Our project has 4 pages. First one is home page and we have been explained what this project is. Other 3 pages have been working on the employee's database. Part 1, 2 and 3 pages were developed using bootstrap to provide detailed information to the user.



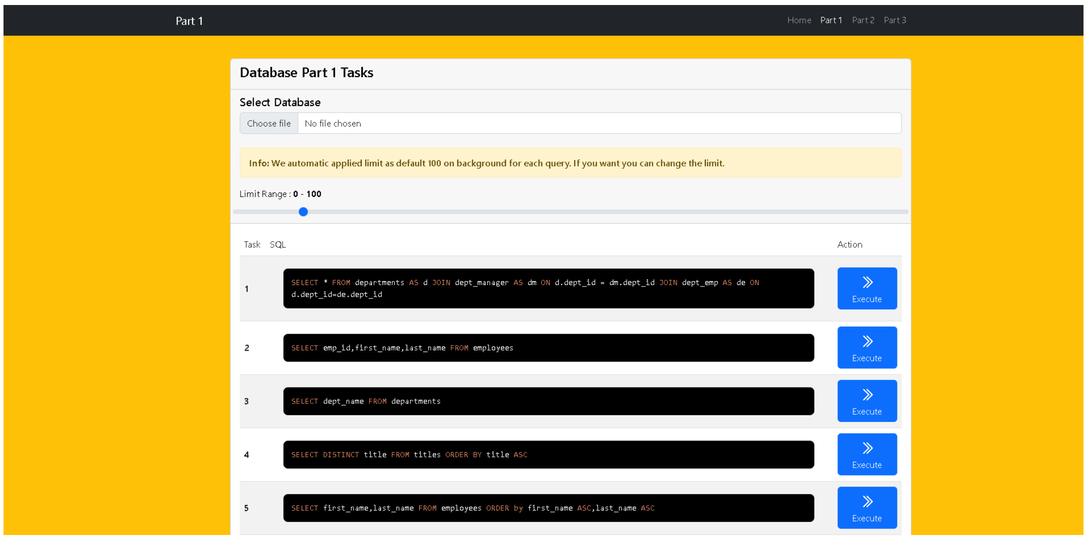
Employees Database: GitHub

This project is to perform Data Definition Language (DDL) and DML (Data Manipulation Language) operations on the Employees database using the SQL.js library.

Project developed by Halil UGUR, @halilugur.

Picture 1: Home page screen

This page as a cover that explained project. This page has structure that navbar, content and footer sections. Also, part pages allow us to see the SQL outputs instantly by writing the answers to the questions asked in the database project here. Since the employee's database is a large database, we need to set limits on each query. That's why we put a bar on each section page where we can increase the limit from 1 to 1000. Thus, the user will be able to dynamically select the desired range.



Database Part 1 Tasks

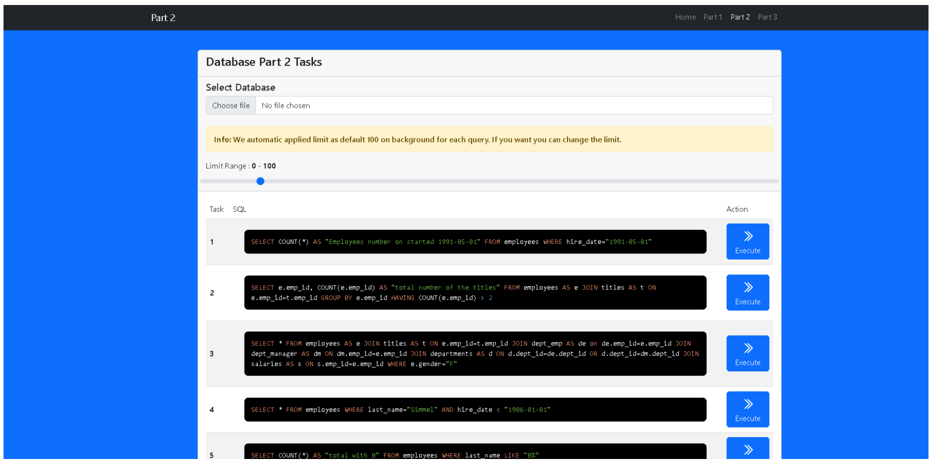
Select Database

Info: We automatic applied limit as default 100 on background for each query. If you want you can change the limit.

Limit Range : **0** - **100**

Task	SQL	Action
1	`SELECT * FROM departments AS d JOIN dept_manager AS dm ON d.dept_id = dm.dept_id JOIN dept_emp AS de ON d.dept_id=de.dept_id`	Execute
2	`SELECT emp_id,first_name,last_name FROM employees`	Execute
3	`SELECT dept_name FROM departments`	Execute
4	`SELECT DISTINCT title FROM titles ORDER BY title ASC`	Execute
5	`SELECT first_name,last_name FROM employees ORDER by first_name ASC,last_name ASC`	Execute

Picture 2: Part 1 page screen

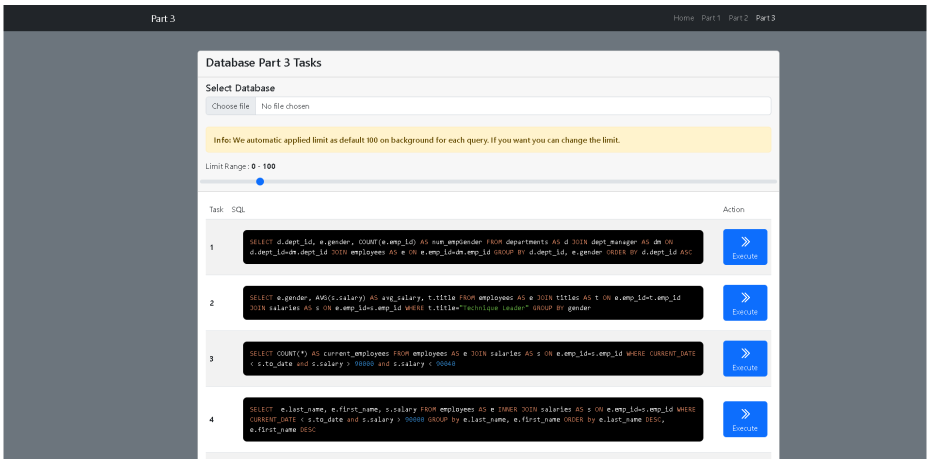


Info: We automatic applied limit as default 100 on background for each query. If you want you can change the limit.

Limit Range : 0 - 100

Task	SQL	Action
1	`SELECT COUNT(*) AS "Employees number on started 1991-05-01" FROM employees WHERE hire_date="1991-05-01"`	Execute
2	`SELECT e.emp_id, COUNT(e.emp_id) AS "total number of the titles" FROM employees AS e JOIN titles AS t ON e.emp_id=t.emp_id GROUP BY e.emp_id HAVING COUNT(e.emp_id) > 2`	Execute
3	`SELECT * FROM employees AS e JOIN titles AS t ON e.emp_id=t.emp_id JOIN dept_emp AS de on de.emp_id=e.emp_id JOIN dept_manager AS dm ON dm.emp_id=e.emp_id JOIN departments AS d ON d.dept_id=de.dept_id OR d.dept_id=dm.dept_id JOIN salaries AS s ON s.emp_id=e.emp_id WHERE e.gender="F"`	Execute
4	`SELECT * FROM employees WHERE last_name="Simmel" AND hire_date < "1986-01-01"`	Execute
5	`SELECT COUNT(*) AS "total with B" FROM employees WHERE last_name LIKE "B%"`	

Picture 3: Part 2 page screen



Info: We automatic applied limit as default 100 on background for each query. If you want you can change the limit.

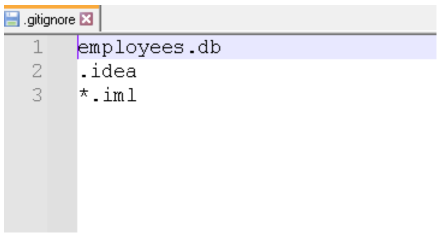
Limit Range : 0 - 100

Task	SQL	Action
1	`SELECT d.dept_id, e.gender, COUNT(e.emp_id) AS num_empGender FROM departments AS d JOIN dept_manager AS dm ON d.dept_id=dm.dept_id JOIN employees AS e ON e.emp_id=dm.emp_id GROUP BY d.dept_id, e.gender ORDER BY d.dept_id ASC`	Execute
2	`SELECT e.gender, AVG(s.salary) AS avg_salary, t.title FROM employees AS e JOIN titles AS t ON e.emp_id=t.emp_id JOIN salaries AS s ON e.emp_id=s.emp_id WHERE t.title="Technique Leader" GROUP BY gender`	Execute
3	`SELECT COUNT(*) AS current_employees FROM employees AS e JOIN salaries AS s ON e.emp_id=s.emp_id WHERE CURRENT_DATE < s.to_date and s.salary > 90000 and s.salary < 90040`	Execute
4	`SELECT  e.last_name, e.first_name, s.salary FROM employees AS e INNER JOIN salaries AS s ON e.emp_id=s.emp_id WHERE CURRENT_DATE < s.to_date and s.salary > 90000 GROUP by e.last_name, e.first_name ORDER by e.last_name DESC, e.first_name DESC`	Execute

Picture 4: Part 3 page screen

.gitignore File

Git is a tool that helps us save different versions of files. We can make sure certain files don't get saved by using a ".gitignore" file. This will keep private files like passwords, large file, and API keys safe, and keep our git from getting cluttered. It's not a good idea to put the employee's database in the git repository because it is too big.



```
employees.db
.idea
*.iml
```

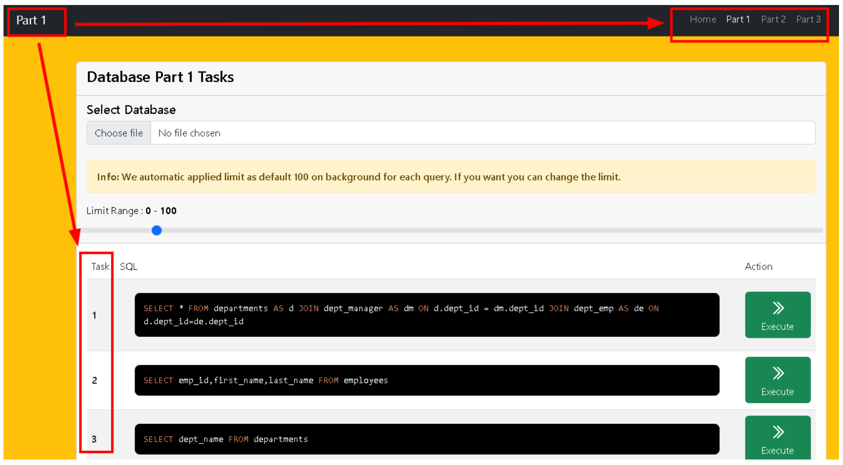
Picture 5: .gitignore file content

Bootstrap

Bootstrap is a tool that helps web developers build websites faster by providing them with templates and commands. It is made up of HTML, CSS, and JavaScript, and makes it easier to design websites that look great on any device. In this project, we developed our interface using bootstrap.

Page Sub-Parts

Each page has sections where the user can execute queries. Also, Part 1 page is bolded.



Database Part 1 Tasks

Select Database

No file chosen

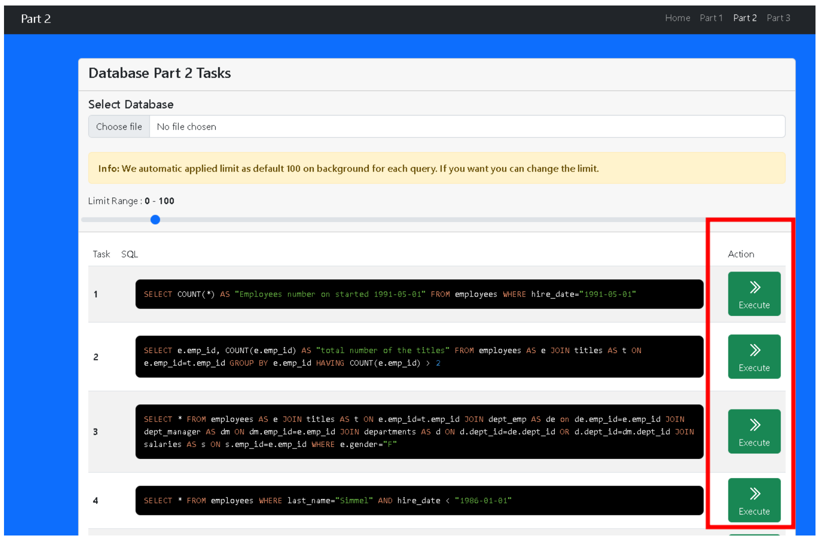
Info: We automatic applied limit as default 100 on background for each query. If you want you can change the limit.

Limit Range : **0** - **100**

Task	SQL	Action
1	`SELECT * FROM departments AS d JOIN dept_manager AS dm ON d.dept_id = dm.dept_id JOIN dept_emp AS de ON d.dept_id=de.dept_id`	Execute
2	`SELECT emp_id,first_name,last_name FROM employees`	Execute
3	`SELECT dept_name FROM departments`	Execute

Picture 6: Queries screen

Green Buttons



Database Part 2 Tasks

Select Database

No file chosen

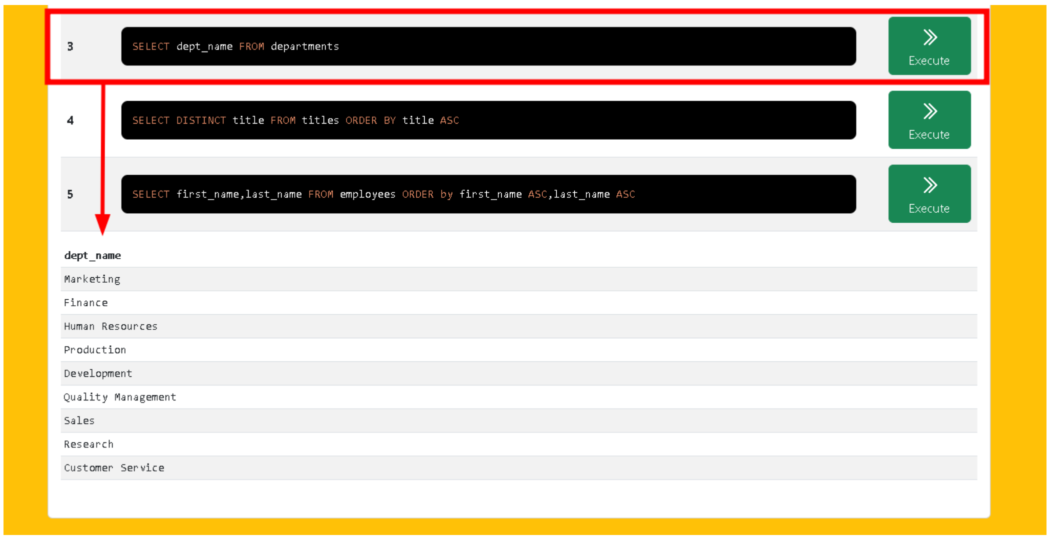
Info: We automatic applied limit as default 100 on background for each query. If you want you can change the limit.

Limit Range : **0** - **100**

Task	SQL	Action
1	`SELECT COUNT(*) AS "Employees number on started 1991-05-01" FROM employees WHERE hire_date="1991-05-01"`	Execute
2	`SELECT e.emp_id, COUNT(e.emp_id) AS "total number of the titles" FROM employees AS e JOIN titles AS t ON e.emp_id=t.emp_id GROUP BY e.emp_id HAVING COUNT(e.emp_id) > 2`	Execute
3	`SELECT * FROM employees AS e JOIN titles AS t ON e.emp_id=t.emp_id JOIN dept_emp AS de on de.emp_id=e.emp_id JOIN dept_manager AS dm ON dm.emp_id=e.emp_id JOIN departments AS d ON d.dept_id=de.dept_id OR d.dept_id=dm.dept_id JOIN salaries AS s ON s.emp_id=e.emp_id WHERE e.gender="F"`	Execute
4	`SELECT * FROM employees WHERE last_name="Simmel" AND hire_date < "1986-01-01"`	Execute

Picture 7: Execute has been changed to a green button.

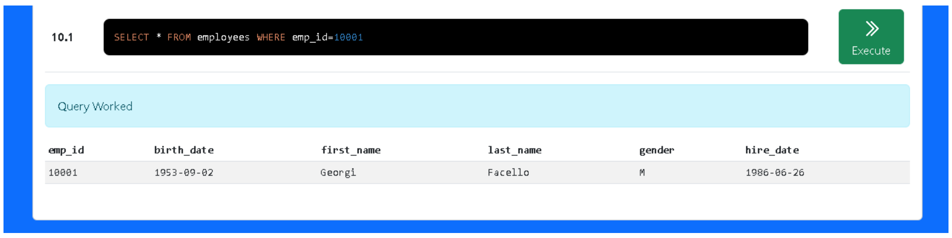
SQL Result Screens



```
3  SELECT dept_name FROM departments                                       Execute
4  SELECT DISTINCT title FROM titles ORDER BY title ASC                    Execute
5  SELECT first_name,last_name FROM employees ORDER by first_name ASC,last_name ASC   Execute
```

dept_name
Marketing
Finance
Human Resources
Production
Development
Quality Management
Sales
Research
Customer Service

Picture 8: Page 1 example result

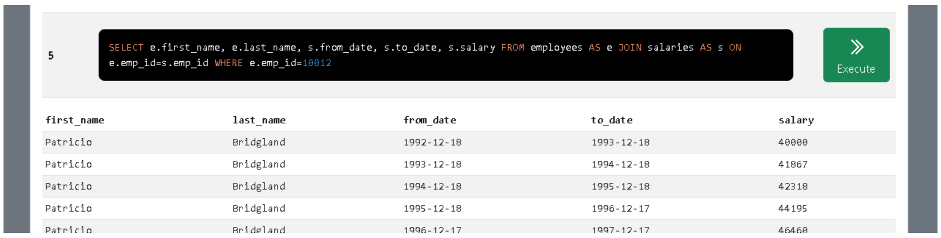


```
10.1  SELECT * FROM employees WHERE emp_id=10001     Execute
```

Query Worked

emp_id	birth_date	first_name	last_name	gender	hire_date
10001	1953-09-02	Georgi	Facello	M	1986-06-26

Picture 9: Part 2 example result

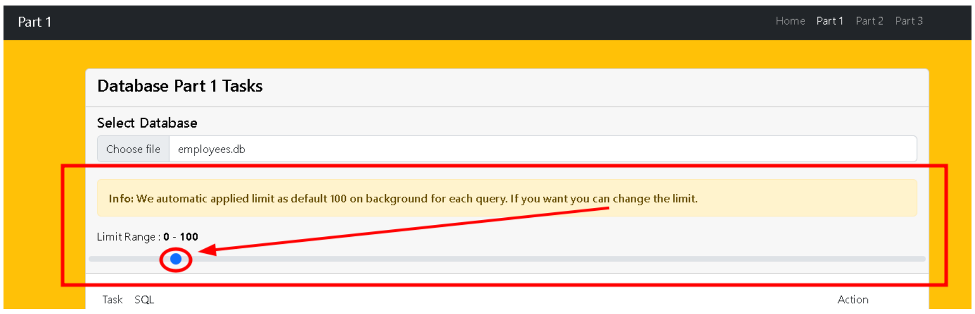


```
5  SELECT e.first_name, e.last_name, s.from_date, s.to_date, s.salary FROM employees AS e JOIN salaries AS s ON
   e.emp_id=s.emp_id WHERE e.emp_id=10012                                                      Execute
```

first_name	last_name	from_date	to_date	salary
Patricio	Bridgland	1992-12-18	1993-12-18	40000
Patricio	Bridgland	1993-12-18	1994-12-18	41867
Patricio	Bridgland	1994-12-18	1995-12-18	42318
Patricio	Bridgland	1995-12-18	1996-12-17	44195
Patricio	Bridgland	1996-12-17	1997-12-17	46460

Picture 10: Page 3 example result

Limited SQL Queries



Database Part 1 Tasks

Select Database

Choose file employees.db

Info: We automatic applied limit as default 100 on background for each query. If you want you can change the limit.

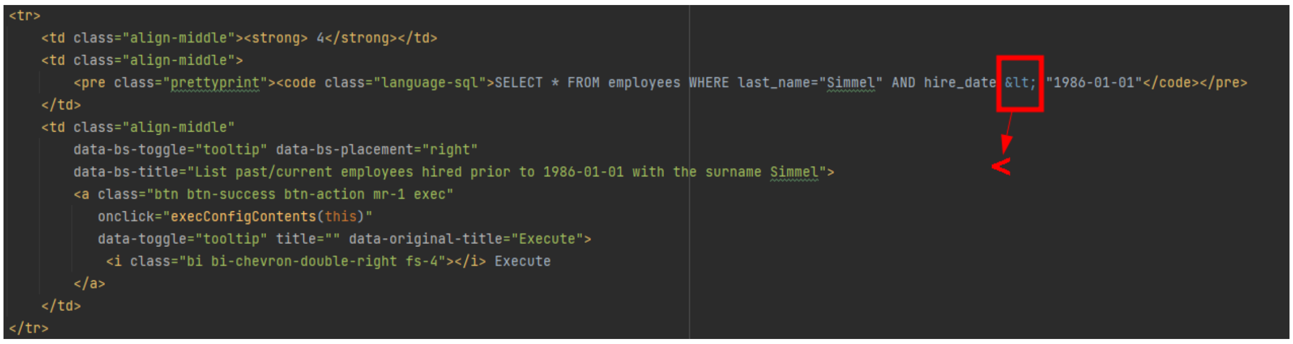
Limit Range : 0 - 100

Task SQL Action

Picture 11: Query limitation screen

Data SQL field

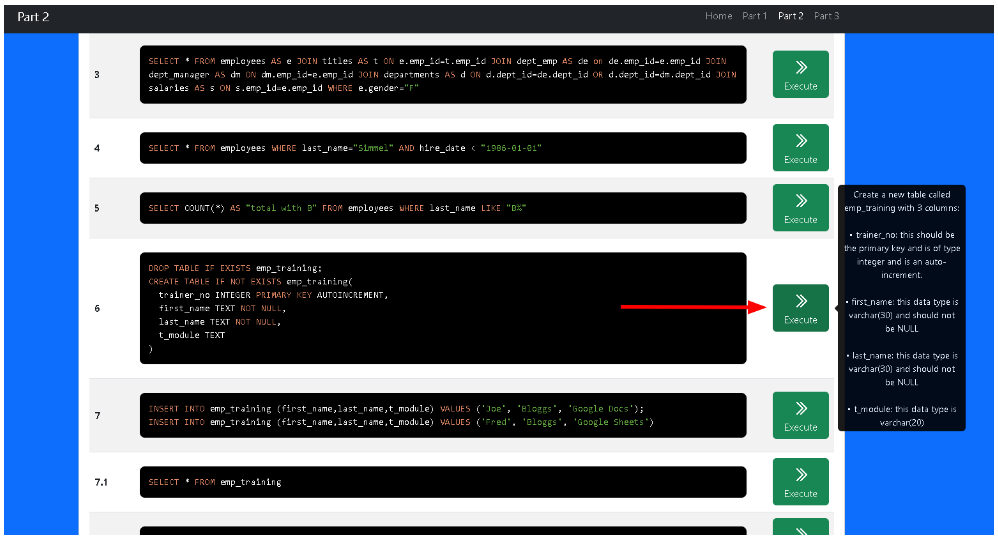
HTML Escape Characters are used in queries.



```
<tr>
    <td class="align-middle"><strong> 4</strong></td>
    <td class="align-middle">
        <pre class="prettyprint"><code class="language-sql">SELECT * FROM employees WHERE last_name="Simmel" AND hire_date &lt; "1986-01-01"</code></pre>
    </td>
    <td class="align-middle"
        data-bs-toggle="tooltip" data-bs-placement="right"
        data-bs-title="List past/current employees hired prior to 1986-01-01 with the surname Simmel">
        <a class="btn btn-success btn-action mr-1 exec"
           onclick="execConfigContents(this)"
           data-toggle="tooltip" title="" data-original-title="Execute">
            <i class="bi bi-chevron-double-right fs-4"></i> Execute
        </a>
    </td>
</tr>
```

Extra

When you go over the Run buttons, there is an English equivalent of the expression that SQL data wants to run.



3

```
SELECT * FROM employees AS e JOIN titles AS t ON e.emp_id=t.emp_id JOIN dept_emp AS de on de.emp_id=e.emp_id JOIN
dept_manager AS dm ON dm.emp_id=e.emp_id JOIN departments AS d ON d.dept_id=de.dept_id OR d.dept_id=dm.dept_id JOIN
salaries AS s ON s.emp_id=e.emp_id WHERE e.gender="F"
```

4

```
SELECT * FROM employees WHERE last_name="Simmel" AND hire_date < "1986-01-01"
```

5

```
SELECT COUNT(*) AS "total with B" FROM employees WHERE last_name LIKE "B%"
```

6

```
DROP TABLE IF EXISTS emp_training;
CREATE TABLE IF NOT EXISTS emp_training(
  trainer_no INTEGER PRIMARY KEY AUTOINCREMENT,
  first_name TEXT NOT NULL,
  last_name TEXT NOT NULL,
  t_module TEXT
)
```

Create a new table called emp_training with 3 columns:

- trainer_no: this should be the primary key and is of type integer and is an auto-increment.
- first_name: this data type is varchar(30) and should not be NULL
- last_name: this data type is varchar(30) and should not be NULL
- t_module: this data type is varchar(20)

7

```
INSERT INTO emp_training (first_name,last_name,t_module) VALUES ('Joe', 'Bloggs', 'Google Docs');
INSERT INTO emp_training (first_name,last_name,t_module) VALUES ('Fred', 'Bloggs', 'Google Sheets')
```

7.1

```
SELECT * FROM emp_training
```

GitHub Repository

All the materials we use can be accessed from this repo: GitHub Repo

References

1. https://github.com/googlearchive/code-prettify
2. https://codemirror.net/
3. https://getbootstrap.com/
4. https://www.geeksforgeeks.org/bootstrap/
5. https://www.w3schools.com/bootstrap/bootstrap_ver.asp
6. https://sql.js.org/#/
7. https://sql.js.org/examples/GUI/
8. https://stackoverflow.com/