CM500292 –Databases Coursework

**Developing a Database-Driven Application**

|  |
| --- |
| **Percentage of overall unit mark**: **50%** |
| **Submission Location: engage**  **Submission Format: .pdf or .docx file and .txt file**  **Submission Files: DOC or PDF named "FlightManagement\_Document”, .txt file named "FlightManagement\_RepoLink.txt"** |
| **Anonymous Marking: yes**  **Generative AI Assessment Categorisation- Type A [guidance can be found**[**here**](https://www.bath.ac.uk/announcements/academic-integrity-in-the-generative-artificial-intelligence-genai-context/)**.]** |

Overview

This coursework addresses the following learning outcome for this unit:

* Practically demonstrate how programs and users can interact with databases through query languages.

This assignment aims to assess your understanding of relational databases by designing, creating, and interacting with a database through SQL queries. You will develop a simple database-driven application that demonstrates CRUD operations, allowing users to interact with a relational database through a basic program interface.

Scenario

Suppose you have been appointed as the **Database Manager** of an airline company that is updating its IT system. The airline requires a database to manage essential information about its **flights**, **pilots**, and **destinations**. You are tasked with designing and implementing a **Flight Management Database** that holds relevant data and enables staff members to easily interact with it. The program should allow the airline staff to manage flight schedules and update records as needed.

To meet these requirements, you need to create a database application that includes a user-friendly menu system, allowing staff to add, retrieve, update, and delete information about flights, pilots, and destinations. This application should demonstrate how programs and users interact with databases through SQL queries.

Tasks

You must complete the following tasks based on the scenario and provide the requested evidence of each. The three primary tasks are as follows:

1. **ER Diagram and Relational Schema** - Include a detailed ER diagram and the SQL code for table creation.
2. **Source Code** - Submit the Python source code for your application, including all SQL queries.
3. **Reflective Documentation** - A report explaining your approach, SQL query explanations, and reflection on the assignment.

More details of these tasks are on the next page.

1. ER Diagram and Relational Schema

* Design a relational schema to represent the airline’s data based on above scenario that would incorporate database interaction mentioned in the source code section below.
* Create an **Entity-Relationship (ER) Diagram** for the flight management system.
* Define primary and foreign keys to establish relationships between entities.

2. Source Code

* Database Setup in SQLite

Create and populate the database in **SQLite**. You can use the SQLite3 tool which is included with Python, to work with SQLite databases.

Populate each table with **sample data** (10–15 records per table) to facilitate testing and demonstration.

* SQL Queries and Database Interaction
* Write SQL queries that enable the following interactions:
* **Flight Retrieval**: Retrieve flights based on multiple criteria, such as destination, status, or departure date.
* **Schedule Modification**: Update flight schedules (e.g., change departure time or status).
* **Pilot Assignment**: Assign pilots to flights and retrieve information about pilot schedules.
* **Destination Management**: View and update destination information as required.
* Include additional queries that summarise data, such as the number of flights to each destination or the number of flights assigned to a pilot.
* Application Development in Python (using SQLite3)
* Develop a **command-line interface (CLI)** in Python using the sqlite3 library to interact with your SQLite database.
* The application should present a menu with options such as:
  + **Add a New Flight**
  + **View Flights by Criteria**
  + **Update Flight Information**
  + **Assign Pilot to Flight**
  + **View Pilot Schedule**
  + **View/Update Destination Information**
* Ensure the interface displays results clearly and allows users to make changes based on input.

1. Reflective Documentation

Document your approach, including the purpose and function of each SQL query.

Reflect on your experience, noting any challenges faced and solutions implemented.

Include screenshots demonstrating interactions between the interface and the database, such as adding a new flight or updating a schedule.

Submission Requirements

You should submit two files as follows:

1. **ER Diagram and Reflective Documentation**
   * **File format**: PDF or Word document (.pdf or .docx).
   * **Contents**:
     + **ER Diagram**: A visual representation of the relational schema (can be created using any diagram tool or by hand, but should be inserted into the document as an image if done separately). <https://draw.io> is a freely available tool for drawing software diagrams.
     + **Relational Schema**: A description of each table, including primary and foreign keys.
     + **Reflective Documentation**: A brief report that includes:
       - Explanation of the database structure and the purpose of each table.
       - A description of each SQL query, including how it works and why it’s used.
       - A reflection on the experience, challenges faced, and solutions.
       - Screenshots of the application interacting with the database (e.g., adding a flight, retrieving pilot schedules).
2. **Text file with a link to a GitHub repository containing the Python Source Code and database file(s).**

* **File format**: a text file containing a link to a GitHub repository, which contains the Python source code and database files. The code will be tested in GitHub CodeSpaces. Ensure that your repository is public to ensure that the assessors can access the code.
* **Repository Contents**: The full code for the command-line interface application, using sqlite3 to connect and interact with the SQLite database. The code should include comments explaining the main parts of the program (e.g., functions for each menu option, database connection, SQL queries).

File Naming Convention

Files should be named as follows:

* Document with ER diagram and reflections: FlightManagement\_Document.pdf
* Repository link to Python source code: FlightManagement\_RepoLink.txt

****Assessment Criteria****

|  |  |
| --- | --- |
| **Criteria** | **Expected Standards** |
| **Database Design (20%)** | Well-structured ER diagram, normalised schema, proper use of primary and foreign keys. |
| **SQL Queries (30%)** | Accurate SQL queries performing required CRUD operations and aggregations, efficient use of joins and conditions. |
| **Application Functionality (30%)** | Functional menu-based interface, correctly handling user input and interactions with the database. |
| **Documentation & Reflection (20%)** | Clear explanations, thoughtful reflection on the process and challenges, and relevant screenshots of the application. |

Feedback

Formative feedback is available in the dedicated lab sessions.

You will receive **summative feedback** on your work within 3 semester weeks of the submission deadline. The feedback will discuss your performance based on the criteria for marking, including what you did well and how specific components/sections could have been improved.

Academic Integrity

Your work will be checked to ensure that you have not plagiarised. For more information about the plagiarism policy at the University, see: <https://library.bath.ac.uk/referencing/plagiarism>

The Generative AI assessment categorisation for this assignment is **Type A**. This means that you are **NOT** allowed to use generative AI tools to complete this assignment. More information on this can be found [here.](https://www.bath.ac.uk/announcements/academic-integrity-in-the-generative-artificial-intelligence-genai-context/)

Remember that published work that you refer to in your report should be clearly referenced in your text and listed in a bibliography section given at the end of your report. For more information, see, <https://library.bath.ac.uk/referencing/new-to-referencing>

Extension requests

Requests for extensions should be made to the Director of Studies. Lecturers and tutors cannot approve extensions. Please make sure you are familiar with the department’s coursework deadline extension policy which can be found on Moodle PGT zone.

It is your responsibility to check that correctly submit your work. Once you have submitted to Moodle you should download your submission and check it.

FAQ

What should I submit?

You should submit two files. a .doc or .pdf file with ER Diagram, Relational Schema, and the Reflective Documentation**.** You should also submit a .txt file with a link to a public GitHub repository containing your Python source code and database file for your application. The Python source code should include all SQL queries. This code will be evaluated in GitHub CodeSpaces. A guide to creating a GitHub repository can be found [here](https://github.blog/developer-skills/github/beginners-guide-to-github-repositories-how-to-create-your-first-repo/). A short primer on GitHub CodeSpaces can be found [here](https://www.geeksforgeeks.org/github-codespaces/).

What should I use to draw my ERD?

You can use any diagram software you wish; however, the recommendation is to use ‘draw.io’ (<https://draw.io>) as this is freely available. You can hand draw a diagram and include an image of it in your submission.

What style of notation do I use for ERD?

While drawing ER diagrams, you could use either Crow's foot, Chen or UML notation. However, make sure you use one style consistently throughout the diagram.

Can I make assumptions?

Yes, you can make assumptions. Please ensure any assumptions are mentioned in the documentation.

Can I use AI to complete the assignment?

You should **NOT** use AI to complete the assignment. The only exception to this is to populate the database. You may use AI to create data to populate the database.