

Module Code: CMP-4010B/CMP-7025B
Coursework: Assessment Sequence {001}
Set by: Farhana Liza <F.Liza@uea.ac.uk>
Date set: 27th February 2025
Value: 50%
Date due: 8th May 2025, 3 PM
Returned by: Assessment Period Week 2
Submission: Blackboard
Checked by: Kazhan Misri < K.Misri@uea.ac.uk >

Learning Outcomes

Experience in the following:

- problem solving techniques using SQL, PostgreSQL; (SQL, Transferable skills)
- interpreting user requirement and defining solutions; (SQL, DBMS understanding)
- creating table definitions using SQL; (SQL, Relational Model)
- creating ER diagrams (SQL, DBMS understanding)
- manipulating table data using SQL; (SQL)
- SQL programming in PostgreSQL; (SQL)
- SQL transactions in PostgreSQL; (SQL, Transactions)
- professionally presenting project works and managing time based on workload, deadlines, and distribution of effort. (Transferable skills)

Specification

Overview

You are required to develop a database application by first completing the table structures (e.g., adding integrity constraints), writing interactive SQL statements, and documenting your work in a professional assessment template. Finally, you will demonstrate the application through a graphical user interface (GUI) that allows users to interact with the database seamlessly.

Description

The Computing and Mathematics Professionals Society (CMPS) is a professional body that promotes the interests of computer scientists and mathematicians. There are two types of membership: *student* member and *accredited* member. To advance from student member to accredited member it is necessary to be a graduate of a recognised university, and to pass several examinations organised by the society. CMPS organises examinations once each year. The examinations are held at a

variety of locations throughout the country. Student members enter examinations by contacting CMPS's office directly or through their website. After the examinations have taken place, CMPS examiners award grades for the students' examination entries and these grades are posted on CMPS's website. If a student withdraws from the society, all his/her examination entries are cancelled. Details of examinations, student members and examination entries are held in a database. This information is accessed over the Internet and by using workstations available to CMPS office staff. The CMPS IT team refer to these various inserts, updates, deletes and queries as *transactions of interest*.

For this coursework exercise you are required to design a database for the CMPS examinations and write SQL statements for the *transactions of interest*. Your role is to prototype and test the functionality (*transactions of interest*) required for the system. Naturally the exercise is greatly simplified compared to a real application. A description of the tables and required functionality has been provided. A detailed specification of the work to be undertaken and the deliverables to be produced for assessment is given below.

The first stage in this process is to analyse the requirements and write SQL statements to perform these tasks. These statements can be tested using an interactive SQL interface to ensure correct functionality.

You may, of course, use your own facilities to develop your program but the final version **must use PostgreSQL and run in the CMP labs**.

System Functionality

The database consists of the following tables, each containing specific fields as listed below:

exam (excode, extitle, exlocation, exdate, extime)
student (sno, sname, semail)
entry (eno, excode, sno, egrade)
cancel (eno, excode, sno, cdate, cuser)

Notes:

- The exam table holds details of each examination scheduled for the coming year.
- The student table holds details of student members of the society.
- The entry table holds details of the examination entries made by students for the coming year.
- The cancel table is used to record details of all entries that have been cancelled.

- excode is a four-character code identifying an examination, e.g. DB01 for the Database Exam 1.
- extitle is a unique descriptive title of the examination.
- exlocation is a place where the examination is held.
- exdate is a date on which the examination is held. All currently planned examinations are scheduled for the month of November 2025.
- extime is a start time of the examination. No examinations start before 09:00 hours or after 18:00 hours.
- sno is a membership number given to a student member of the society.
- sname is a name of a student.
- semail is an email address of the student.
- eno is a reference number for an entry made for an examination by a student. Entries are unique reference numbers to aid anonymous marking of student's scripts.
- egrade is the grade given to a student by the examiners. When an entry is first made this field is empty. If the student attends the examination and is awarded a grade, it is recorded in this field. The range of grades is 0 to 100. If a student does not attend the examination then this field remains empty.
- cdate is a timestamp showing when the cancellation of an entry takes place.
- cuser is a user id of a person causing a cancellation of an entry to be recorded in the cancel table, for simplicity you can use system user or any string (e.g., admin) for the user id.

The **transactions of interest** for CMPS IT are as below:

A. Insert a new student member of the society.

B. Insert a new examination for the coming year.

C. Delete a student. This happens if a student withdraws from the society. All the examination entries for the student must be cancelled. The cancelled entries must retain their student reference number even though there is no longer a matching row in the student table.

D. Delete an examination. Examinations that have no entries may be deleted from the database. The examination must not have any current (not cancelled) entries.

E. Insert an examination entry. A student can only enter a specific examination once in a year. The student cannot take more than one examination on the same day.

F. Update an entry. This records the grade awarded by the examiners to an entry made by a student for an examination. The entry is specified by entry reference number.

G. Produce a table showing the examination timetable for a given student. The student is specified by his/her student membership number. The timetable should contain the student's name and location, code, title, day and time of each examination for which the student has entered.

H. Produce a table showing the result obtained by each student for each examination. The table should be sorted by examination code and then by student name. If the student is awarded a grade of 70% or more then the result is to be shown as 'Distinction', a grade of at least 50% but less than 70% is to be shown as 'Pass' and grades below 50% are to be shown as 'Fail'. If the student has not taken the examination then the result is shown as 'Not taken'. The table should display the exam code, exam title, student name and exam result (e.g., 'Distinction', 'Pass', 'Fail', 'Not taken').

I. As H above but for a given examination. The examination is specified by examination code.

Professional Presentation:

Design Python based Graphical User Interface for the above transactions of interest functionalities and record a MP4 video demonstration.

Tasks:

Part 1: DDL and DML (approx. 70% of marks)

For consistency, use the following minimal database definition: A copy of this text can be found in the file Cw_Schema.txt in Blackboard under the Coursework folder.

Minimal database definition

```
CREATE TABLE exam (
    excode      CHAR(4),
    extitle     VARCHAR(200),
    exlocation  VARCHAR(200),
    exdate      DATE,
    exptime     TIME);

CREATE TABLE student (
    sno         INTEGER,
    sname       VARCHAR(200),
    semail      VARCHAR(200));

CREATE TABLE entry (
    eno         INTEGER,
    excode      CHAR(4),
    sno         INTEGER,
    egrade      DECIMAL(5,2));

CREATE TABLE cancel (
    eno         INTEGER,
    excode      CHAR(4),
    sno         INTEGER,
    cdate       TIMESTAMP,
    cuser       VARCHAR(200));
```

Add additional SQL clauses and/or statements to complete the definition of the database by specifying DDL statements including data integrity and constraints (e.g., primary keys, domain constraints, entity and referential integrity constraints), views, functions, triggers, comments. Note that **you should NOT modify the name and type of the attributes in the minimal database definition**. Save all your Data Definition Language (DDL) statements in a .sql file.

At this stage the tables are empty. Load (e.g., using insert statements) a reasonable volume of data into the tables for testing the Transactions of Interest. The data should be reasonable to test the Transactions of Interest with their expected output and should provide a suitable environment in which to test normal operation as well as abnormal conditions (e.g., if you constraint an attribute by defining a primary key on the attribute, adding multiple rows with same value for that attribute would consider as an abnormal condition).

Prepare and test interactive SQL statements for the Transactions of Interest. Test these statements using the SQL Query Tool editor in pgAdmin. The purpose is to test your SQL statements before real world deployment in the production (i.e., Prototyping Phase). You may need more than one execution of some of the Transactions of Interest to verify the correctness of your work (e.g., test primary keys, referential integrity, correct and incorrect execution). Please be prepared to add the SQL statements for an assessment template which will be released to you 48 hours prior to the submission deadline with the assessment test data. The template will look like the exemplar template in the Appendix (see section Template for Submission with a Mock Question and Answer). The assessment test data will consist of insert statements for the four database tables.

Part 2. Presentation (approx. 25% of marks)

Prepare a MP4 video recording of a 5-minute demonstration presentation to evidence your design and development of graphical user interface for the **transactions of interest** prototyped in Part 1. The demonstration of Graphical user Interface can be done with your own data only.

Quality of Submission and Showcase Your Technical Skills (approx. 5% of marks)

Clarity and professionalism of the template and demonstration presentation.

Relationship to Formative Assessment

All SQL Labs build towards this exercise, with each lab helping you to perform Transactions of Interest that contribute to building this solution. Labs include expected query results so you can ensure you get correct answers and perform a form of self-assessment.

Deliverables

Your solution must be submitted via blackboard.

- Part 1: Submit two .sql file containing SQL data definition statements and your own data; a word document (.docx) containing the completed assessment template, and python source code of the Graphical User Interface. The assessment template will contain your SQL statements for each of the **Transactions of Interest** together with evidence of testing for each statement on assessment data. The Assessment Template along with the assessment data will be issued 48 hours prior to submission deadline.
- Part 2: Submit your MP4 video recording of Graphical User Interface demonstration (max 5 minutes).

Blackboard submission point upload

- After entering the submission point, submit your part 1 solution by selecting 'Upload Files'.
- For the part 2 solution, submit your MP4 video recording to eStream by selecting 'Create submission' and following the instructions in this link: <https://my.uea.ac.uk/departments/learning-technology/students/video-and-audio-assignments>

The submission point will be released 1 week before the submission date and will show in the Summative Assessment tab on Blackboard. Name the submission documents should be with your student ID (e.g., student_id_DDL.sql, student_id_own_data.sql, student_id_assessment_template.docx, student_id_GUI_source_code.py, student_id_GUI_demo.mp4). A mock sample submission file structure is available on the Blackboard under the 'CMP-4010B/CMP-7025B Assessment Sequence {001} – Coursework' thread.

The primary process for marking this assignment is the assessment template and demonstration recording. Your marks will be determined by the accuracy of the SQL statements in accordance with the assessment data and quality of the demonstration provided within the Assessment Templates and of the GUI demonstration.

Important notes

- The document you submit should be complete and neatly formatted to ease reading.
- This is an individual piece of coursework NOT a group project. Collusion/plagiarism checks may be carried out.
- As you will be given the assessment data based on the tables in the minimal database definition, it is vital that you do not change the table names, field names, field types.

Resources

The necessary materials are in the lecture notes/lab materials. Links to relevant PostgreSQL help pages are in the lecture slides. Recommended books in reading list are also useful.

Marking scheme

Marking Details	Marks
Part 1 – DDL and DML A. Data integrity and constraints, Views, functions, triggers, comments, your own data for testing. (Approx. 20%) B. Data Insertion, Update, and Deletion: Correctness and data integrity. (Approx. 20%) Data Retrieval: Correctness. (Approx. 30%)	Approx. 70%
Part 2 – Presentation Recording Prepare a MP4 video recording of a 5-minutes PowerPoint presentation to evidence your design and development with the Graphical User Interface.	Approx. 25%
Overall Presentation - Clarity and professionalism of the template and demonstration presentation.	Approx. 5%

Assessment criteria

- Good use of SQL data definition language to complete the table definitions;
- Good use of SQL data manipulation language to write interactive queries;
- Ability to interpret project specification correctly and accurately. This may include a little bit of independent thinking on what may make a table/report look good;
- Correct functionality and output as required by each requirement;
- Neatly presented work with correct program output in the assessment template; Sufficiency and completeness of Submitted code and tests data.
- Professional demonstrate the application through a graphical user interface (GUI) that allows users to interact with the database seamlessly.

Plagiarism, Collusion, and Contract Cheating

The University takes academic integrity very seriously. You must not commit plagiarism, collusion, or contract cheating in your submitted work. Our Policy on Plagiarism, Collusion, and Contract Cheating explains:

- what is meant by the terms 'plagiarism', 'collusion', and 'contract cheating'
- how to avoid plagiarism, collusion, and contract cheating
- using a proof reader
- what will happen if we suspect that you have breached the policy.

It is essential that you read this policy and you undertake (or refresh your memory of) our school's training on this. You can find the policy and related guidance here:

<https://my.uea.ac.uk/departments/learning-and-teaching/students/academic-cycle/regulations-and-discipline/plagiarism-awareness>

The policy allows us to make some rules specific to this assessment. Note that:

In this assessment, working with others is *not* permitted. All aspects of your submission, including but not limited to: research, design, development and writing, must be your own work according to your own understanding of topics. Please pay careful attention to the definitions of contract cheating, plagiarism and collusion in the policy and ask your module organiser if you are unsure about anything.

Appendix

Template For Submission with a Mock Question and Answer:

EXAMPLE OF HOW TO FILL THE SUBMISSION DOCUMENT (using different indicative Transactions of Interest) WITH YOUR OWN (TRAIN) DATA.

Testing task 1

1. Given flight details create new flight record

Create a new flight with values: flight ID = **120**, origin = **'STN'**, destination = **'OVD'**, flight date = **'30/7/2020'**, maximum capacity = **5**, and price per seat = **100**.

INSERT YOUR SQL QUERY AND OUTPUT HERE



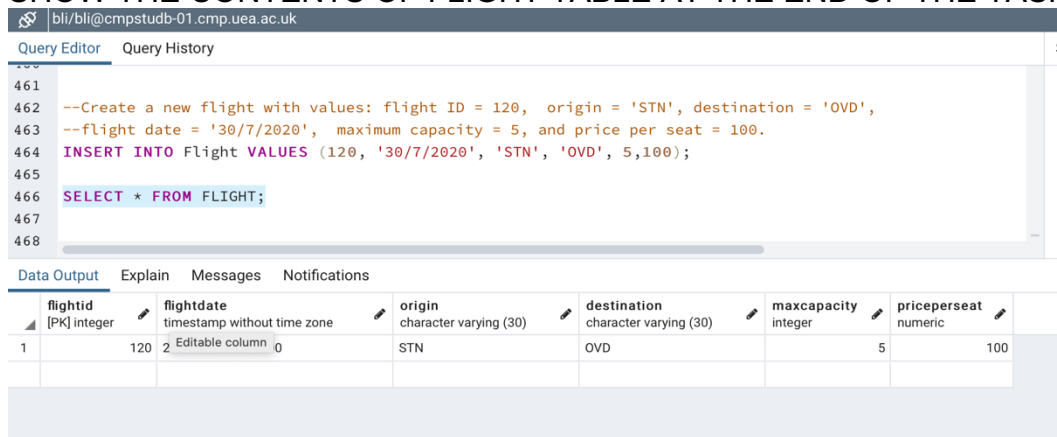
The screenshot shows a SQL query editor interface. The top bar indicates the user is logged in as 'bli/bli@cmpstadb-01.cmp.uea.ac.uk'. Below the bar, there are tabs for 'Query Editor' and 'Query History'. The 'Query Editor' tab is active, displaying the following SQL code:

```
461  
462 --Create a new flight with values: flight ID = 120, origin = 'STN', destination = 'OVD',  
463 --flight date = '30/7/2020', maximum capacity = 5, and price per seat = 100.  
464 INSERT INTO Flight VALUES (120, '30/7/2020', 'STN', 'OVD', 5,100);  
465  
466  
467  
468
```

Below the query editor, there are tabs for 'Data Output', 'Explain', 'Messages', and 'Notifications'. The 'Messages' tab is active, showing the following output:

```
INSERT 0 1  
  
Query returned successfully in 170 msec.
```

SHOW THE CONTENTS OF FLIGHT TABLE AT THE END OF THE TASK



The screenshot shows the same SQL query editor interface. The 'Query Editor' tab is active, displaying the following SQL code:

```
461  
462 --Create a new flight with values: flight ID = 120, origin = 'STN', destination = 'OVD',  
463 --flight date = '30/7/2020', maximum capacity = 5, and price per seat = 100.  
464 INSERT INTO Flight VALUES (120, '30/7/2020', 'STN', 'OVD', 5,100);  
465  
466 SELECT * FROM FLIGHT;  
467  
468
```

Below the query editor, there are tabs for 'Data Output', 'Explain', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing the following table:

	flightid [PK] integer	flightdate timestamp without time zone	origin character varying (30)	destination character varying (30)	maxcapacity integer	priceperseat numeric
1	120	2 Editable column 0	STN	OVD	5	100

2. Produce a query that counts the number of tuples in each table.

INSERT YOUR SQL QUERY AND OUTPUT HERE