LaSalle College
Montréal

Project 2: 15%

Course Identification

Name of program – Code:	
Course Identification	**DATABASE II**
Course number:	420-BD2-AS
Group:	07438
Teacher’s name:	Jean-François Parent
Duration:	Extended
Semester:	Fall 2023

Student Identification

Name: ____ Student number: _______________

Date: ____________________ Result: _______________

☐ I declare that this is an original work, and that I credited all content sources of which I am not the author (online and printed, images, graphics, films, etc.), in the required quotation and citation style for this work.

Standard of the Evaluated Competency
Statement of the evaluated competency – Code
Use a database management system – 00Q7
Evaluated elements of the competency 00Q7
1. Create the database. *2. Formulate read requests, insertion requests, modification requests and deletion requests.* *3. Ensure data confidentiality and consistency.* *4. Program automated data processing operations.* *5. Save and restore the database.*

Instructions

- Create a database (schema) according to the requirements
- It is the teacher’s responsibility to identify language errors. If such errors are found, teachers may apply a penalty of up to 20% of the grade (IPEL – Article 5.7).
- Plagiarism, attempts at plagiarism or complicity in plagiarism during a summative evaluation results in a mark of zero (0). In the case of recidivism, in the same course or in another course, the student will be given a grade of '0' for the course in question. (IPEL – Article 5.16).
- Submit dates on Omnivox must be respected.

- Please consult the correction grid at the end of document for more details.

TOTAL: 100 POINTS

Because this work must be done at home, if we think that the answers are not yours, the department reserves the right to complete your evaluation with a virtual meeting to verify that you attained the required competencies.

Project 2

Directives

Note: This is **NOT** a team work

1. For objects names, only use the recommended characters (A-Z (objects names must begin by a letter), 0-9 and the underscore (_).

2. Each field must have the correct data type. The size must be big enough to store all required data, but not too big to use too much space on the disk.

3. Unless specified otherwise, each field must contain data and must not accept NULL values and/or empty strings ('').

4. Specify a default value where you can do it.

5. Specify constraints where you can do it to prevent users to enter unwanted data.

6. Each table must have a primary key which identifies each row with a unique value (GUID).

7. You must create indexes for each field used in a search, for each foreign key, and for each column which must be unique.

8. Each SQL query must use good indentation (for example, no query on a single line).

9. Write comments to describe what the SQL code does.

10. For each table and object, you must write the revision history. It must contains the date, your name, your student number and the modifications made. For tables, enter this information in the comments of the primary key. For the other objects (stored procedures, functions, etc.) you must inscribe these informations in the comments. For example:
Table:

2023-10-29	Bill Torvalds (1244556)	Created the customers table.
2023-10-31	Bill Torvalds (1244556)	Added the firstname column.

Stored procedure:

```
--Revision history:
--2023-11-01       Bill Torvalds (1244556)              Created the procedure customers_select.
--2023-11-03       Bill Torvalds (1244556)              Fixed the ORDER BY bug.
```

11. All the newly created tables must include the 2 fields for the date and time of creation/modification (like in project 1).

Projet description

This project must re-use the database created in project 1. Marks will be awarded only for the new elements. It is thus not mandatory to fix the errors made in project 1, unless these elements are required in project 2.

(00Q7.1 / 00Q7.4 / 00Q7.4) (23 pt)

Create or modify the tables to manage the following elements. Enter data (at least 5 rows) in each of the tables:

- **Teachers**
  Add a field for the social insurance number, which is made of 9 digits in Canada (ex: 123 456 789 or 123456789). Choose the most appropriate data type to avoid invalid entries made by the users.

  Add a field to know, for each worked hour, which amount the teacher must pay in insurances (for example: 1.25$ per hour). Use NULL instead of 0$ when there is no insurance to pay.

- **Students**
  Add a field to count the number of tries for all the exams (successfully passed or not) for each student. The default value must be NULL.

- **Reports**
  Add a field for the date when the different exams were done.

- **Timesheets**
  This table must be used by all the teachers to fill-in their timesheets. It must contain the date for the beginning of a working week week, a foreign key to point to an existing teacher, and the number of hours worked during the week (only one digit authorized after the decimal).

(00Q7.2 / 00Q7.3 / 00Q7.4) (18 pt)

- Create a trigger to keep the history of all the modifications (UPDATE) made in the reports table. Create a table to store these modifications in **different fields**. Each row must include the following fields: the name of the modified field, the date and time of the modification, the old value and the new value. For example:

Field name	Modification date	Old value	New value
RESULT	2023-10-01	57	60
STUDENT_ID	2023-10-02	ACDEF-01234567890	987654-3210FEDCBA
TEACHER_ID	2023-10-03	112233-445566778899	FFEEDD-CCBBAA00

- Create another trigger to add one try for a student each time a new mark is added to the reports. The number of tries must be assigned to the correct student.

  Make sure that the triggers do NOT lock the tables.

(00Q7.2 / 00Q7.3 / 00Q7.4) (54 pt)

To access the data, create stored procedures and/or functions to perform the following operations with the **timesheets**:

- INSERT
- UPDATE
- SELECT one timesheet with its primary key. Display the table data without fetching the data from the foreign table.
- SELECT all timesheets for one specific teacher. The results must include the lastname and firstname of the teacher, and all the other fields from the teachers table. Sort from the most recent timesheet to the oldest one.
- DELETE one row with the teacher ID and the starting date for the week (do not use the primary key for deleting).

Create stored procedures and/or functions to perform the following operations on the **reports**:

- INSERT data (including the foreign keys)
- UPDATE
- SELECT all the rows for one specific student. The returned resultset must contains the fields from the foreign tables (lastnames, firstnames, etc.). Sort from the oldest date to the most recent one.
- DELETE one row with the student ID and the course ID (do not use the primary key for deleting)

(00Q7.5) (5pt)
When your database is ready, export it into a text file (.sql). The filename must contain at least your lastname and the date when the backup was made.

This exported file only creates the objects and not the database. You must thus add manually the SQL code to create and use a database (schema) which has the same name than your database. Use the password “123”. The script must create all the objects (database/schema, tables, procedures, etc.) so test it properly.

Your script must NOT delete any existing database/schema if it already contains object(s).

Your script must be error-free.

Submit this unique file on Omnivox.