CSCI 4140 Advanced Database Systems

# Assignment 2

## Preamble

This is a relatively simple assignment that is asking you to write an application that accesses a DB.

## Specifications

Recall that in your first assignment you were to create an ER model to support creation and management of purchase orders. This assignment is asking you to write an application that supports request from users to submit purchase orders and to query the status of a purchase order. The specs for DB tables to support creation and management of the purchase orders are given first, which is followed by functional specifications.

### DB Tables to Support Purchase Orders, Parts for Sale, and Clients/Customers

Recall that each identifier in your tables (table names and names of attributes in your tables) needs to end with the last three digits of your student ID. Thus, you need to replace "007", which are the last three digits of my Banner ID, in identifier names with the last three digits of your Banner (student) ID.

***Table Parts007***

The company keeps track of parts for same using the table Parts007 that has the following attributes:

- *partNo007* … unique (PK) - *partName007* - *currentPrice007*
- *qoh007*... quantity on hand … quantity of parts in stock (also referred to as Quantity On Hand (QOH))

***Table Clients007***

Information that needs to be known about clients/customers is stored in the table Clients007 that has attributes:

- *clientId007* … ID of the client comp (PK) - *clienName007*
- *clientCity007* - *clientPassword007*
- *moneyOwed007*

***Table POs007***

Information about purchase orders (POs), but not on lines containing the parts ordered in a purchase order, is stored in the table POs007 that has attributes:

- *poNo007*… unique PO number (PK) - *clientCompID007*… ID of the client comp.
- *dateOfPO007*… date of the purchase order - *status007* … PO status

***Table Lines007***

Each PO has a number of lines, each one denoting an order for a number of units of one part. Thus, for each part on a PO, the part number, quantity, and price need to be stored. Each line on a purchase order is represented by a tuple in the table Lines007. The table has the following attributes:

- *poNo007*… (FK and part of PK) - *partNo007* ... (part of PK)
- *qty007* … quantity ordered - *priceOrdered007* ... price at which the part was ordered

Thus, you need to create a DB containing the above tables and populated it with few tuples in each table.

### Functional Requirements

Assume that the application you are developing is for a company that offers parts for sale. Your task is to develop an application that supports the following functionality so that clients are able to (i) list parts, (ii) prepare a purchase order, (iii) submit a purchase order for a number of parts, and (iii) query the status of a submitted PO. In essence, you need to develop a method for each of the functionality:

- List parts for sale … returns/provides the list of part stored in the Parts007 table, but not the quantity on hand.
- Find information about a part, given the part number.
- List information about POs (but not lines).
- List lines for a given PO number.
- Prepare a PO … this may be done in two parts, e.g., two methods:
  - First, the user (client) provides/inputs information about a purchase order, but not its lines.
  - Then, the user provides information about lines for the purchase order.

  Once information about a purchase order is prepared, it is submitted for processing (storage in a DB) by invocation of a method described below (receiving and storing a purchase order).

- Submit a PO by invocation of a method that first checks that:
  - PO number is unique.
  - Client ID on a PO is valid.
  - Part numbers on the lines of the submitted purchase order are valid.

  If the validation finds any errors, the purchase order is rejected.

## Additional Requirements

- FCS provides a DB server. However, for the purposes of this assignment, you may use any DB server as long as you document what it is (e.g., you may wish to use a DB server that is on your local system).
- Naming Conventions
  - Your DB must be such that the names of your tables and of the columns must end with the last three digits of your student ID.
  - In your software, the names of your methods/functions/procedures must end with the last three digits of your student ID and the same applies about any parameter of a method/procedure/function.

## Software Options

Preferably, you should use REST services to support the above functionality, or you may write the software as a Java program.

### REST Services

If you are developing REST services, each of the above requirements may be fulfilled by a REST service. You may develop services individually, while testing using POSTMAN software, or similar, to invoke a service. However, you also need to develop a simple UI that will invoke your REST services.

Students who use REST services successfully for this assignment will be rewarded with bonus marks (10%).

### Java Program

You may also submit a single Java program.

You will need to have code/methods that interact with the user (a client/customer) who issues the requests.

As indicated in class, you shall not be evaluated on your UI design …

## Submission requirements

Your submission should have the usual packaging (front page, TOC (Table of Contents), etc.) and contain the following information:

- Listing of DB server and software/frameworks you used for development and for which parts (e.g., Angular for UI).
- Listing of DB schema.
- Sample content of the tables (e.g., initial content of the tables before your testing).
- Description of methods supporting the required functionality ( i.e., for each of your methods describe the method's signature and a line or few lines describing what it does.
- Information how your marker can access your Git repository that stores your software. Please ensure that the markers, your TA, and the course instructor have access to your git. (The user IDs for this will be provided.)
- Section that identifies under which situation/condition your program does not work (known bugs … if any)
- Your Git repository should have all code, including SQL scripts that you used to "seed" your data.
- Judiciously selected and annotated screenshots that will demonstrate that your software works. Your annotation should assist the marker to easily determine that your software works.

In case of doubts about the functionality of your software, your TA/marker may contact you to arrange time when to meet so that you can demonstrate your software. This will be done within 2 weeks of the due date.

For an appeal of your grade, see the course syllabus.

CSCI 4140 Advanced Database Systems

# Assignment 1 – Rubric

**Requirements that**

- " … software, the names of your methods/functions/procedures must end with the last three digits of your student ID and the same applies about any parameter of a method/procedure/function" and that
- " … DB must be such that the names of your tables and of the columns must end with the last three digits of your student ID". However, tuples/data in tables need not end with the last three digits of you student ID.

must be satisfied or else the assignment receives a failure grade and is brought to my attention for a follow-up.

## Rubric Categories

- Software (e.g., middleware and DB access) … 70%
    - Functional requirements satisfied?
    - Software organization/structure
- Submission … 20%
    - Format (e.g., format (e.g., title page, section headings, use of white space))
    - Submission organization (structure of the submission, logical flow of material)
- Evaluator impression … 10%
    - Used when the evaluator determines that your assignment is above or below average submissions. Most of the time, the evaluator impression is about the average of the above two categories.

## Rubric

The rubric is presented by first describing requirements to obtain a C grade range (satisfactory), meaning that the requirements were not quite met. Then, B range is described, following by the A (Excellent) range.

In general, if your submission satisfies the requirements and the submission is good to in all evaluation categories, you will receive a very good grade (A- to B+).

### Satisfactory: C- to C+ range

The submission has sufficient information to convince the marker that the main objectives of the assignment were more or almost satisfied, but there are some issues with the software functionality. Documentation organization, presentation and completeness will affect the variations within the range …

### Good: B- to range B+ range

As above but the functional requirements are satisfied, and all evaluation categories must be classified as good with some variation (range of "mostly good" -> "good" -> "very good" determine which of B-, B, and B+ applies).

### Excellent: A- to A+ range

As above and software works correctly with submission quality being classified be as excellent. Your marker likes your submission.

## Appeal process

To appeal your grade on the assignment, please follow the steps described in the course syllabus.