ECE 656 – Winter 2015
Assignment #2

Overview

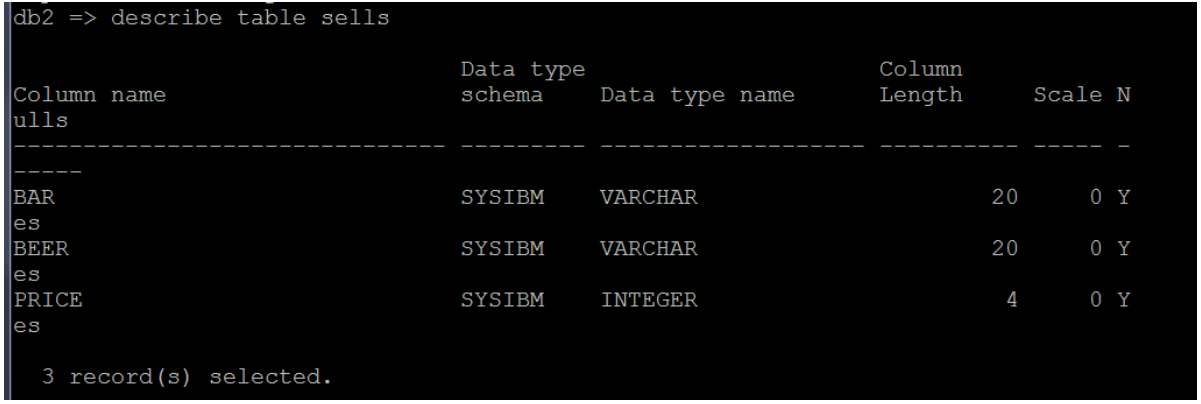
There are two parts to this assignment: SQL DDL and advanced SQL applications, and SQL API applications.

Part 1. SQL DDL and Advanced SQL Applications [25 marks]

For this part of the assignment, follow these steps in deriving your solution.

Step 1. Implement the ER model using SQL DDL in the DBMS of your choice. When selecting the DBMS, make sure that it enforces FOREIGN KEYs, and supports PROCEDURES and TRIGGERS.

Step 2. Use "DESCRIBE TABLE <table-name> " on each created table, and include the output from the DBMS (e.g., using the Print Screen key), as shown in the example below.

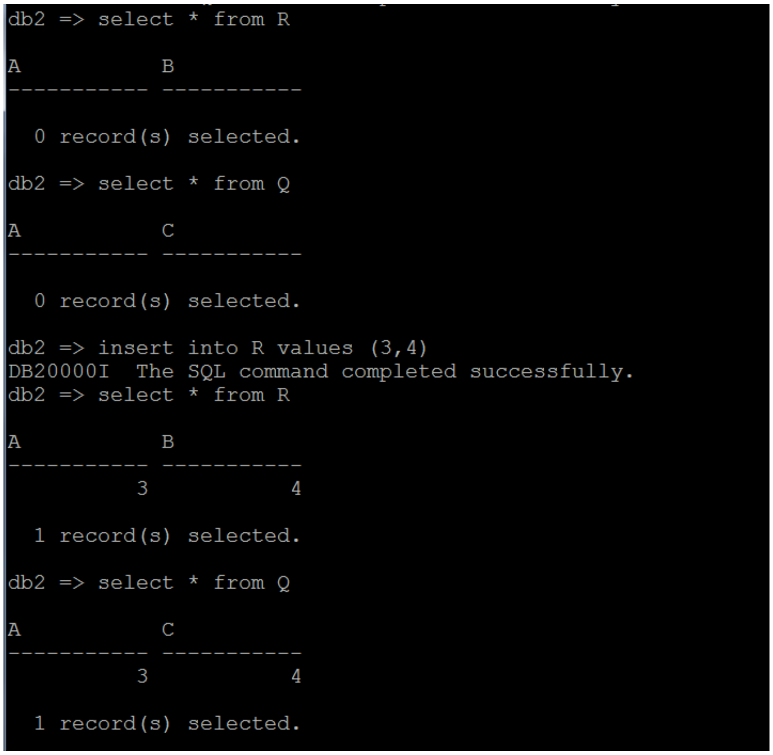


```
db2 => describe table sells

                                Data type                     Column
Column name                     schema    Data type name      Length     Scale N
ulls
------------------------------- --------- ------------------- ---------- ----- -
-----
BAR                             SYSIBM    VARCHAR                     20     0 Y
es
BEER                            SYSIBM    VARCHAR                     20     0 Y
es
PRICE                           SYSIBM    INTEGER                      4     0 Y
es

  3 record(s) selected.
```

Step 3. For each of the ensuing SQL PSM questions, construct the specified SQL PSM object (or its equivalent; for instance, in PL/SQL for Oracle/DB2, or T-SQL for the SQL Server).

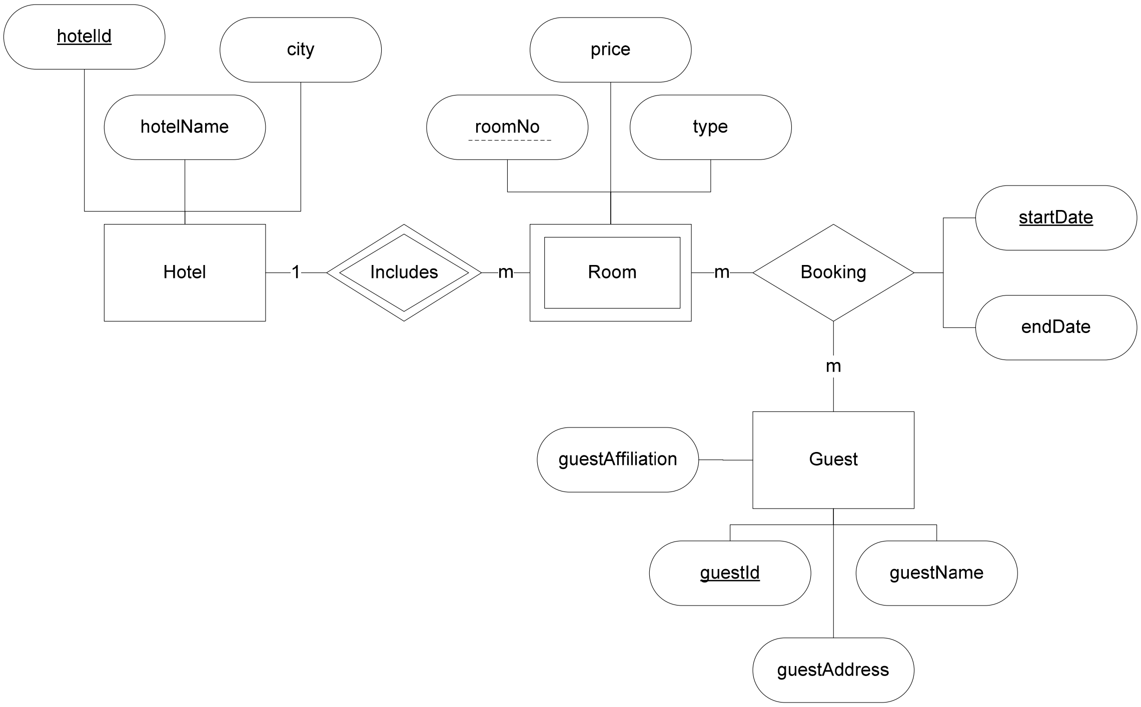
For the PSM procedures, show the effects of executing the procedure (e.g., new tuples inserted, existing tuples updated). For the SQL triggers, run a set of SQL statements against the related tables to show that the corresponding rules are enforced.

Include the output from the DBMS in your report, as shown in the example below in which an AFTER TRIGGER (not shown) is used to perform an INSERT on Q for each INSERT on R.



```
db2 => select * from R

A           B
----------- -----------

  0 record(s) selected.

db2 => select * from Q

A           C
----------- -----------

  0 record(s) selected.

db2 => insert into R values (3,4)
DB20000I  The SQL command completed successfully.
db2 => select * from R

A           B
----------- -----------
          3           4

  1 record(s) selected.

db2 => select * from Q

A           C
----------- -----------
          3           4

  1 record(s) selected.
```

Hotel Database



SQL DDL Statements:

1. Create TABLE Hotel, where

- hotelID is a fixed-length string of 4 characters and is the PRIMARY KEY,
- hotelName is a variable-length string of 30 characters, and
- city is a fixed-length string that allows values: ‘Guelph’, ‘Kitchener’, and ‘Waterloo’.

2. Create TABLE Room, where

- hotelID is a fixed-length string of 4 characters, is part of the PRIMARY KEY, and it REFERENCES Hotel(hotelID) with the DEFAULT policy for UPDATE and DELETE actions,

- roomNo is a fixed-length string of 4 characters and is part of the PRIMARY KEY,
- price is of decimal data type with two-digit decimal precision and allows values between 50.00 and 250.00, and
- type is a fixed-length string that allows values: ‘Single’, ‘Double’, ‘Queen’, and ‘King’.

3. Create TABLE Booking where

- hotelID, roomNo, and guestID are all fixed-length strings of 4 characters, are all part of the PRIMARY KEY, and all REFERENCE the corresponding columns in the Hotel, Room, and Guest tables respectively, with the DEFAULT policy for UPDATE and DELETE actions; and
- startDate and endDate are of type DATE, are of format YYYY-MM-DD, and startDate is part of the PRIMARY KEY.

4. Create TABLE Guest where

- guestID is a fixed-length string of 4 characters and is the PRIMARY KEY,
- guestName is a variable-length string of 30 characters that stores the full name,
- guestAddress is a variable-length string of 50 characters that stores the full address, and
- guestAffiliation is a variable-length string of 30 characters that stores the affiliation (e.g., CAA).

SQL PSM Objects:

1. Create a **PSM PROCEDURE** that will help you populate the database with the information for the individual rooms in the area hotels. The procedure should take as input the hotelID, the number of floors in the hotel, and the number of rooms per each floor (assume constant for all floors). In the body of the procedure, insert tuples for the corresponding number of rooms with the room number is assigned as XY, where X is the floor number (assume X can range from 01 to 99) and Y is the room number (assume Y can range from 01 to 99), the room price is set to 50.00, and the room type is set to ‘Double’ by default.
   Signature: PopulateHotelRooms(IN hID CHAR(4), IN noFloors INT, IN noFRooms INT)

2. Create one or more **BEFORE TRIGGERs** on the Booking table that will reject (e.g., using SIGNAL SQLSTATE) a new booking or change to an existing booking if such a change will create a conflict with another booking for the same room. That is, the (startDate, endDate) range for the new or modified booking will not conflict with the (startDate, endDate) range for any other booking for the same room. You can assume that if the endDate of the earlier booking falls exactly on the startDate of a later one that there will be no conflict (e.g., there is a three-hour maintenance window between bookings).

[The following is a mandatory question for groups that include five or six members. Otherwise, for groups that include four members or less, the following is an optional, bonus question worth 5 bonus marks.]

3. Create a maintenance TABLE called BookingLog with the same base attributes as the Booking table and two additional attributes: userID of type VARCHAR and changeTime of type DATETIME (or its equivalent for the selected DBMS). Then, create an **AFTER TRIGGER** that will log all of the values inserted for each new booking, and record the userID of the user making the change and the current date and time when the change was being made.

Part 2. SQL API Applications [25 marks]

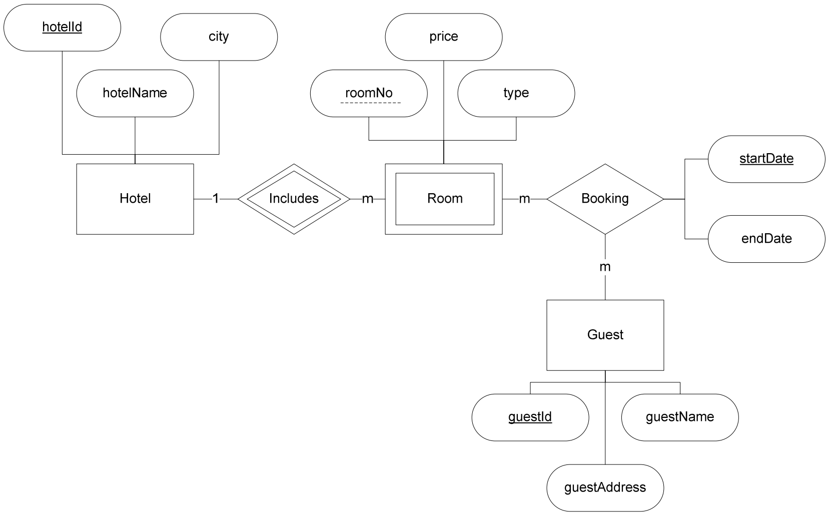
For this part of the assignment, follow these steps in deriving your solution.

Step 1. Use your Hotel Database implementation from above (i.e., the same SQL DDL statement) to implement the given ER model in the DBMS of your choice. The chosen DBMS could be the same as above or could be a different one, but it has to support the API of your choice (e.g., JDBC), and it has to have the corresponding connection drivers available.

Step 2. Implement the specified functionality in the host language using the API of your choice (e.g., implement the functionality in Java using JDBC as the connection API). Include the source code and test cases with your submission. You do not have to provide a user interface with menu options, and you can just include a demonstrative scenario instead.

Step 3. For each module in your report, include the output from your program (e.g., using the Print Screen key) showing a typical execution scenario, where important steps are clearly marked and illustrated.

Hotel Database Revisited



Module 1: Guest Registration

1. A booking agent registers a new guest and enters their information, including the name and address. The guestID is auto-generated, either through the API or by the DBMS itself.

Module 2: Booking Query

2. Once the guest information has been entered, the agent can then query hotels for available rooms on specified dates. That is, the agent enters one or all of the following: startDate, endDate, hotelName, city, room price, and room type. For any entry that is left blank, the corresponding condition is not applied (e.g., if city is left blank, all cities are considered).
3. The booking query returns all of the rooms that are available based on the information entered, and the query displays the hotelID, hotelName, city, roomNo, price, and type.

Module 3: Booking Registration

4. Once an available room has been found, the agent then registers a new booking by entering the hotelID, roomNo, guestID, startDate, and endDate. If the booking was successful, the system returns the bookingID that the agent can give to the customer as confirmation. The bookingID is auto-generated, either through the API or by the DBMS itself.
5. The booking registration should also ensure that bookings for the same room do not overlap.

Deliverables

- A PDF document that includes solutions for each of the assignment questions specified above submitted to the assignment dropbox on LEARN.

- You are permitted to work in groups on the assignment. The recommended group size is four students, and the maximum group size is six students.

- Include the names and IDs of all group members at the start of your document. Only one submission is required from each group.

- **The assignment is due on Mon Mar 23rd by 23:55pm. No late submissions will be accepted.**