Assignment for Module 3

The graded problems in Module 3 provide experience with the CREATE TABLE statement. You can use Oracle or MySQL for this assignment. If you use Oracle, you will need to use the Oracle SQL Developer to connect to an Oracle server.

1. Basic CREATE TABLE Statement Requirements

You should use the table descriptions in the Intercollegiate Database background document. You must use the same table and column names as specified in the background document. Here is some advice about data type selections.

- You should use standard SQL data types specified in the notes except for using VARCHAR2 (an Oracle data type) instead of VARCHAR for columns containing varying length character strings. For MySQL, you should use VARCHAR for variable length strings.
- For primary key fields (*CustNo*, *LocNo*, *EventNo*, *PlanNo*, *EmpNo*, *ResNo*, and *FacNo*), use the VARCHAR (or VARCHAR2 in Oracle) data type with length 8. For consistency, corresponding foreign keys (such as *EventRequest.CustNo*) should also be the same data type and length.
- For Oracle, you should use the DATE data type for the columns involving dates or times. The *EventPlanLine.TimeStart* and *EventPlanLine.TimeEnd* columns will store both date and time data so you should use the DATE data type. In MySQL use the DATE data type for columns with just date details (date columns in the *EventRequest* and *EventPlan* tables) and DATETIME for columns with date and time details (time columns in the *EventPlanLine* table).

- Use CHAR(1) for the *Customer.Internal* column as Oracle does not provide a BOOLEAN data type. MySQL has the Boolean data type, but I suggest that you use CHAR(1) instead.

2. Constraints

After writing the basic CREATE TABLE statements, you should modify the statements with constraints. The CONSTRAINT clauses can be either inline in a column definition or separate after column definitions except where noted. You should specify a meaningful name for each CONSTRAINT clause.

- For each primary key, you should specify a PRIMARY KEY constraint clause. For single column primary keys (*CustNo*, *LocNo*, *EventNo*, *PlanNo*, *EmpNo*, *ResNo*, and *FacNo*), the constraint clause can be inline or external. For multiple column primary keys (combination of *PlanNo* and *LineNo*), the CONSTRAINT clause must be external.
- For each foreign key, you should specify a FOREIGN KEY constraint clause. The constraint clauses can be inline or separate.
- Define NOT NULL constraints for all columns except *eventplan.empno*, *EventRequest.DateAuth*, *EventRequest.BudNo*, and *EventPlan.Notes*. Make sure that you define NOT NULL constraints for the PK of each table. Because of MySQL syntax limitations for NOT NULL constraints (inline with no constraint name and no CONSTRAINT keyword), you should define inline NOT NULL constraints.
- Define a named CHECK constraint to restrict the *eventrequest.status* column to have a value of “Pending”, “Denied”, or “Approved”. You can use the IN operator in this constraint. In MySQL, the syntax does not allow the CONSTRAINT keyword and a constraint name for CHECK constraints. You should use the CHECK keyword followed the condition enclosed in parentheses.

- Define named CHECK constraints to ensure that the *resource.rate* and *eventrequest.estaudience* are greater than 0. In MySQL, you cannot use a constraint name and the CONSTRAINT keyword for CHECK constraints. In MySQL, the syntax does not allow the CONSTRAINT keyword and a constraint name for CHECK constraints. You should use the CHECK keyword followed the condition enclosed in parentheses.
- Define a named CHECK constraint involving *EventPlanLine.TimeStart* and *EventPlanLineTimeEnd*. The start time should be smaller (chronologically before) than the end time. This CHECK constraint must be external because it involves two columns. In MySQL, the syntax does not allow the CONSTRAINT keyword and a constraint name for CHECK constraints. You should use the CHECK keyword followed the condition enclosed in parentheses.

3. Populating Tables

The course website contains a text file containing SQL INSERT statements to populate the tables. You need to create the tables before inserting rows in each table. You need to insert rows in parent tables before child tables that reference parent tables. The INSERT statements in the file are in a proper order for populating the tables.

4. Initial CREATE TABLE Statements

To facilitate your work, you can use the CREATE TABLE statements you created in the practice assignment in module 03 for the *Customer*, *Facility*, and *Location* tables. Thus, you only need to write CREATE TABLE statements for the remaining five tables (*ResourceTbl*, *Employee*, *EventRequest*, *EventPlan*, and *EventPlanLine*). You still need to execute the CREATE TABLE statements to create all of the tables and the INSERT statements to populate all tables.

5. Submission

The submission requirements involve CREATE TABLE statements and evidence that you executed the statements and created the tables in Oracle or MySQL. You will submit 5 documents with each document containing a CREATE TABLE statement and screen snapshot to indicate that you created the table in Oracle or MySQL. In each document, you should neatly format your CREATE TABLE statement so that it can be easily graded. You should use the same table and column names as specified in the ICA database background document. You should not put any identifying details about yourself in your submitted document. For the screen snapshot, you need to capture a screen showing most columns and rows of the populated table. You can use a feature of the Oracle or MySQL client to show the rows in a table. Alternatively, you can execute an SQL statement (for example, SELECT * FROM ResourceTbl) to show the columns and rows.

You should submit the documents in this order.

- Employee
- ResourceTbl
- EventRequest
- EventPlan
- EventPlanLine