CSC 174 Fall 2014
Homework 2
(Total 100 points)

This homework allows students to practice the development of a database using MySQL, as well as the design of views, procedures, functions, and triggers.

Section 1

Using SQL, create tables according to the given schema in Figure 1. The EER is shown in Figure 2. You must use the exact same attribute names in your tables as the relational schema (figure1). Cutomer_ID must be the type of Integer. Customer Name must be VARCHAR(30). You can define other attributes with reasonable types.

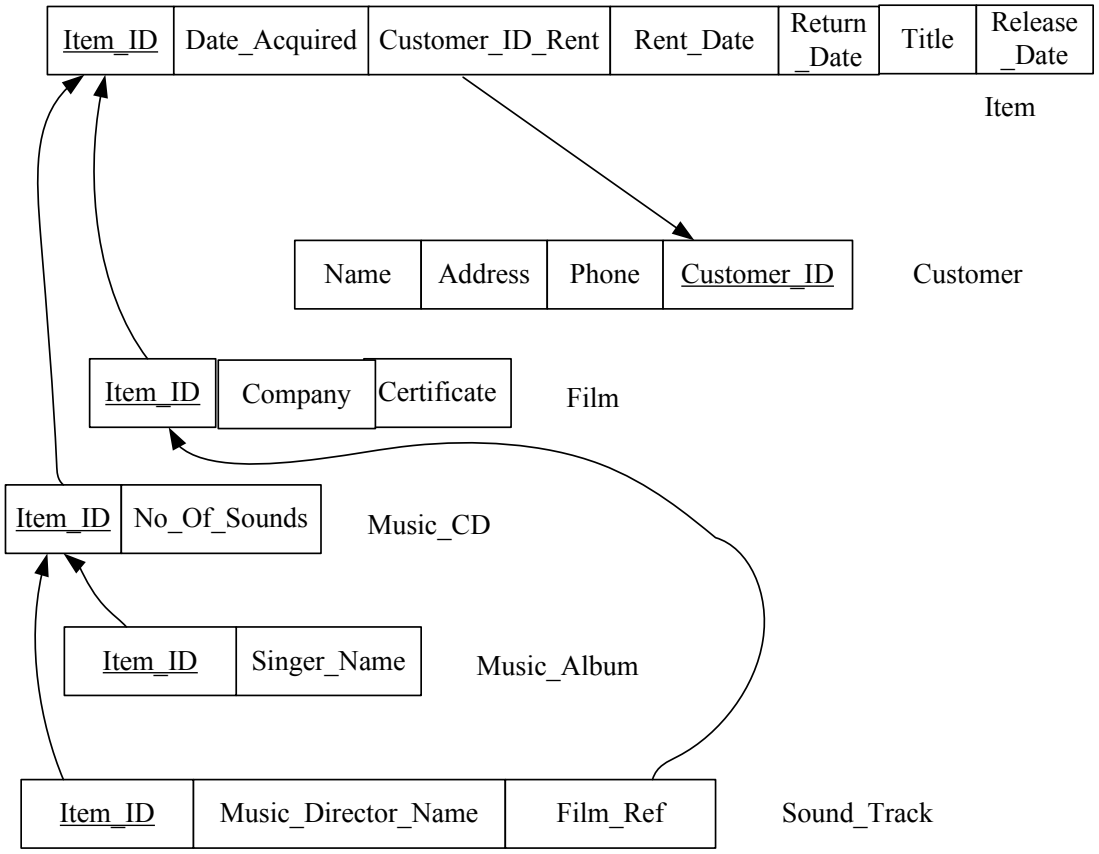

Figure 1. Relational Schema

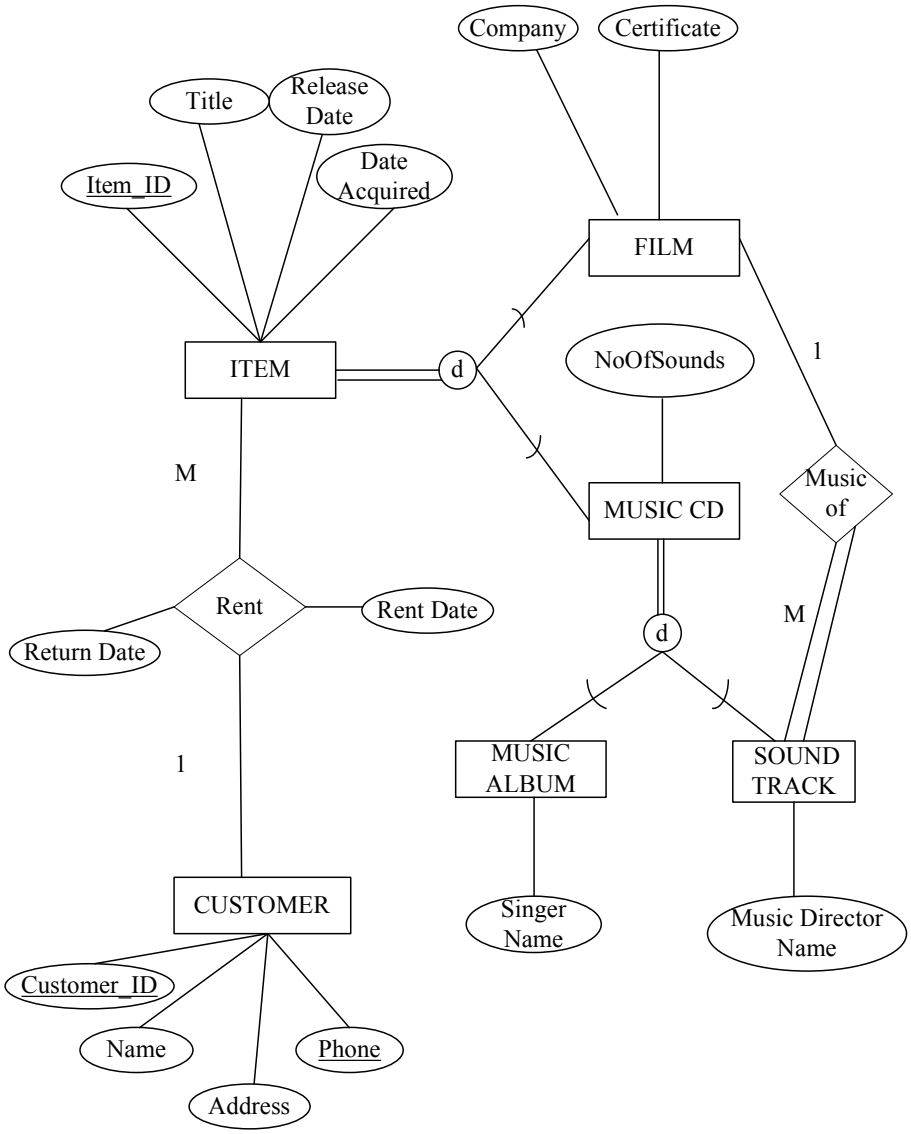

Figure 2. EER

Views related to the specialization are defined as follows.

CREATE VIEW Film_vw As

Select I.Item_ID, I.Date_Acquired, I.Customer_ID_Rent, I.Rent_Date, I.Return_Date, I.Title, I.Released_Date, F.Certificate, F.Company,

From Item as I, Film as F

Where I.Item_ID = F.Item_ID;

```
CREATE VIEW Music_CD_vw As
    Select I.Item_ID, I.Date_Acquired, I.customer_ID_rent, I.Rent_date,
I.Return_Date, I.Title, I.Release_Date, M.No_Of_Sounds
    From   Item as I, Music_CD  as M
    Where I.Item_ID = M.Item_ID;

CREATE VIEW Music_Album_vw As
    Select I.Item_ID, I.Date_Acquired, I.Customer_ID_rent, I.Rent_Date,
I.Return_Date, I.Title, I.Release_Date, I. No_Of_Sounds, M.Singer_Name
    From   Music_CD_vw as I, Music_Album as M
    Where I.Item_ID = M.Item_ID;

CREATE VIEW Sound_Track_vw As
    Select I.Item_ID, I.Date_Acquired, I.Customer_ID_rent, I.Rent_Date,
I.Return_date, I.Title, I.Release_Date, I. No_Of_Sounds, S.Music_Director_Name,
S.Film_Ref
    From   Music_CD_vw as I, Sound_Track as S
    Where I.Item_ID = S.Item_ID;
```

Section 2

Populate the database. Please check sections 3, 4, and 5 for details.

Section 3

Create the following **view**. Display the results of selecting all tuples from the view. When you populate the database, insert data such that at least one tuple will be display as the result of querying the view (select all tuples from the views).

A view constructed by a list of sound tracks that belongs to a film.
Name of the view: Film_Sound_Track

Attributes of the view: film title, item ID of a sound track, the music director name of the sound track. Accordingly, the name of the attributes of the view should be: flim_title, item_ID_st, music_dir.

Section 4

Create a **functio**n to implement the following requirements:

Retrieve the number of films rented by a given customer

Function name: Num_Of_Film_Rented

Input parameter: Customer_ID
Return: Integer number of number of film rented

If the customerID does not exist, return 0.

Section 5

Design a procedure to implement the following requirements:

A stored procedure to output the list of music album's Item_IDs and the singer's names that a given customer (identified by a customer_ID and customer name) rents

Name of the procedure: List_Music_Albums

Input parameter: Customer_ID, Name (of the customer)

Output: Item_ID, Singer_Name

When you populate the database, insert data such that at least one result will be display as the result of calling your store procedure.

Section 6

Create two triggers to enforce the consistency among Item, Music CD, Music Album when deleting a music album.

Trigger 1:

Name: del_album

Description: When delete a music album, the corresponding record in the Music CD table should be deleted.

Trigger 2:

Name: del_CD

Description: When delete a music CD, the corresponding record in the Item table should be deleted.

Section 7

Specify the statements to drop all the tables, views, functions, procedures, and triggers. Pay attention to the order of the drop statements in order to drop everything successfully.

Submission

1) What you need to do:

- Section 1: Create table statements
- Section 2: Insert statements to populate database
- Section 3: View definitions (don't include views that are provided by the instructor). Print the results of selecting all tuples from the views.
- Section 4: definition of the function. Print how to call the function to generate a result, as well as the result of calling the function.
- Section 5: Procedure definition. Print how to call the procedures. Print the result of calling the procedures that can generate a non-empty result.
- Section 6: Triggers definitions. The Testing results to show that you triggers work.
- Section 7: Statements to drop tables, views, functions, triggers, and procedures.

2) What you need to submit:

Submit the following files to SacCT.

a. Create table statements, as well as all the views defined by the instructor. (1_create_table.sql)
b. Insert statements to populate database (2_populate_db.sql)
c. Create view statement (View your created) (3_create_view.sql)
d. Definition of the function. (4_func.sql)

e. Definition of the function (same as d.). Print how to call the function to generate a result, as well as the result of calling the function. (4_func_all.doc or 4_func_all.pdf)
f. Definition of the procedure. (5_proc.sql)
g. Procedure definition (same as f.). As well as how to call the procedure. Include the results of calling the procedure that can generate a non-empty result. (5_proc_all.doc or 5_proc_all.pdf)
h. Trigger definition (6_trigger.sql) including both triggers in one file.
i. Definition of the trigger (same as h). As well as your test results to show that the triggers work (6_trigger_all.doc or 6_trigger_all.pdf).
j. Statements to drop all tables, views, functions, and procedures (6_drop_all.sql)