1. Execute arena.sql

The tables created have the following schemas (with the primary keys underlined):

Activities (ActivityID, ActivityName)

Performers (PerformerID, PerformerName, Street, City, State, Zip, ActivityID)

Arenas (ArenaID, ArenaName, City, ArenaCapacity)

Concerts (PerformerID, ArenaID, ConcertDate, TicketPrice)

1. Write SQL queries to accomplish the following tasks.
   1. Write a SQL query to list all the performers’ name and the name of the activity the performer is involved in.
   2. Write a SQL statement to create a view called Act\_perf based on the query in Exercise 1.
   3. Unlike a query, a virtual table or view can be used as if it is a table in the database. Now write a SQL query against the view as if it was a table: select everything from the view.

*NOTE: The view does not store any data. The data is stored just in the tables used in the definition of view. When a query that uses Act\_perf is executed, the system first evaluates the query that defines Act\_perf, and then performs further evaluation of the query that uses Act\_perf.*

1. To demonstrate this, do the following:
   1. Insert a row in Performers table (use INSERT INTO … VALUES …).
   2. Execute SELECT \* FROM Act\_perf again.

Did the results change? Why?

1. Updates on views
   1. Run the following query:

UPDATE Act\_perf

SET PerformerName = 'Harry Chapinn'

WHERE PerformerName = 'Harry Chapin'

* 1. Does the query succeed? Why?
  2. Execute “SELECT \* FROM Performers” and check whether the name of Harry Chapin was changed to Harry Chapinn.

*NOTE: One reason to use the views: Hide complexity and simplify queries when frequently using the results of a complex query.*

1. Create a view called Perf\_Income to show the name of the performer, the arena name, the date of the concert, and total\_income. total\_income represents the total payment received by a performer for a concert and is based on the fraction of ticket price that goes to the performer, and the total number of tickets sold. We assume that the artist receives 10% of the face value of each ticket and the number of tickets sold is always considered to be 80% of the arena capacity.

SQL has a keyword “LIMIT n” that allows you to display only the first n rows in the result of a select query. For example “SELECT \* FROM Performers LIMIT 5” will display the first 5 rows from Performers table.

1. Write the SQL query on the Perf\_Income view to show the top 8 concerts based on total income in order of highest income to lowest income.
2. Write the SQL query to achieve the same result as above, but without using the Perf\_Income view (you can still use TOP n). This query should look more complicated than the query in the previous exercise.
3. Write the SQL query to find the total money made by all of the artists during all the concerts (use the view).
4. Write the SQL statement (use the view) to update the total\_income for ‘Jimmy Buffett’ to be $100,000. Run the query.

Does the query succeed? Why?