Activities (ActivityID, ActivityName)

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

Arenas (ArenaID, ArenaName, City, ArenaCapacity)

Concerts (PerformerID, ArenaID, ConcertDate, TicketPrice)

1. Create a trigger called *deletePerformer* that prevents the last performer for a particular activity to be deleted from the database. The trigger should be associated with a delete operation on the Peformers table: if the performer to be deleted is the last one for his/her particular ActivityID, then the performer cannot be deleted. Otherwise, the performer should be deleted.
2. Create a stored procedure called *spInsertConcert* that inserts a new concert into the database, under some conditions. The stored procedure has as input parameters performer id, arena id, concert date, and ticket price. The stored procedure should insert a row in the Concerts table only if the following conditions are true:
3. there is no other concert by the same performer on the same date (in that case, the performer is already busy) AND
4. there is no other concert in the same arena for that date (arena is busy).

If the above conditions are not true, the procedure should print an explicative message and terminate.