0. Prior to the Laboratory
   1. Review the previous lab handout.
   2. Review this handout.
   3. Review the following materials (red items are important):

<table>
<thead>
<tr>
<th>Topic</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating Tables</td>
<td><a href="http://www.w3schools.com/sql/sql_create_table.asp">http://www.w3schools.com/sql/sql_create_table.asp</a></td>
</tr>
<tr>
<td>Types</td>
<td><a href="http://www.w3schools.com/sql/sql_datatypes.asp">http://www.w3schools.com/sql/sql_datatypes.asp</a></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Primary Keys &amp; Auto Incrementing</td>
<td><a href="http://www.w3schools.com/sql/sql_primarykey.asp">http://www.w3schools.com/sql/sql_primarykey.asp</a></td>
</tr>
<tr>
<td></td>
<td><a href="http://www.w3schools.com/sql/sql_autoincrement.asp">http://www.w3schools.com/sql/sql_autoincrement.asp</a></td>
</tr>
<tr>
<td>Foreign Keys</td>
<td><a href="http://www.w3schools.com/sql/sql_foreignkey.asp">http://www.w3schools.com/sql/sql_foreignkey.asp</a></td>
</tr>
<tr>
<td>Remove Table From DB</td>
<td><a href="http://www.w3schools.com/sql/sql_drop.asp">http://www.w3schools.com/sql/sql_drop.asp</a></td>
</tr>
<tr>
<td>Modify Structure of Existing Tables</td>
<td><a href="http://www.w3schools.com/sql/sql">http://www.w3schools.com/sql/sql</a> ALTER.asp</td>
</tr>
<tr>
<td>Other Constraints</td>
<td><a href="http://www.w3schools.com/sql/sql_notnull.asp">http://www.w3schools.com/sql/sql_notnull.asp</a></td>
</tr>
<tr>
<td></td>
<td><a href="http://www.w3schools.com/sql/sql_unique.asp">http://www.w3schools.com/sql/sql_unique.asp</a></td>
</tr>
<tr>
<td>Advanced Activity</td>
<td><a href="http://www.w3schools.com/sql/sql_top.asp">http://www.w3schools.com/sql/sql_top.asp</a></td>
</tr>
</tbody>
</table>

1. Lab Objectives & Topics
   Upon completion of this lab you should:
   - Gain experience with simple database design strategy
   - Understand the motivation for entity relations and concepts
   - Learn how to create a database from your design

2. Problem Statement
   In this lab we will continue to work with the Album database. For reference, the database schema is presented in Figure 1 as an Entity-Relation diagram.
A careful examination of the DDL file provided (albums.sql) indicates how these tables were built and related to each other. New requirements may mean that the underlying data model must be modified to support new pieces of data. For example, if we wanted to keep track of the emails of each Musician we could modify the Musicians table to include an email address. SQL allows us to alter tables:

```
ALTER TABLE Musicians ADD EmailAddress VARCHAR(50);
```

A better solution would add support for multiple emails in which case we would need to add an entirely new table.

```
CREATE TABLE email (
    EmailID INT NOT NULL AUTO_INCREMENT,
    MusicianID INT NOT NULL,
    Address VARCHAR(100) NOT NULL,
    PRIMARY KEY(email_id),
    FOREIGN KEY `fk_email_to_musician` (MusicianID) REFERENCES Musician(MusicianID)
); 
```

In this lab, you will build on the Albums database to add support for modeling venues (concert halls) at which bands are under contract to play select album songs. You will add tables and keys to this database to support this functionality.

**Activity 1: Solution outline**

In this activity you will design entities and relation(s) to extend the Albums database such that it supports the following concert information:
1. The band playing at the concert
2. The band’s select album songs played at the concert
3. The date the concert was held
4. The name of the hall where the concert was held
5. The number of seats in the concert hall
6. The number of concert tickets sold

List 1: The information which needs to be supported by the new database.

The entities and relations illustrated in Figure 2 serves as a basis for the solution. You will add fields and another entity by following the first section of the worksheet.

The Songs table is part of the original Albums database. Relationships between two entities are indicated by a line between the two entities and in general are either a one-to-one relationship or a one-to-many relationship. Figure 3 shows the standard way to draw the two types of relations.

Activity 2: SQL Script Modification

Now that you have properly designed the database modifications, you will realize them by writing SQL scripts (or modifying the original DDL file, albums.sql). Follow the directions in the worksheet in order to add the entities and relations you have designed in the previous activity to the tables and relations in the original Albums database.