# **Database Management**

**Design and Implementation of a Relational Database**

# **Submitted by :** Group 1

**Team Members :** Sapna Patel, Nidhi Chandramohan, Jennifer An, Prithiviraj Maniram, Neha Pinjala

1. **Select an application for which a database management system is needed. Describe the**

**application and state why it is an important application from a management perspective.**

Application Description:

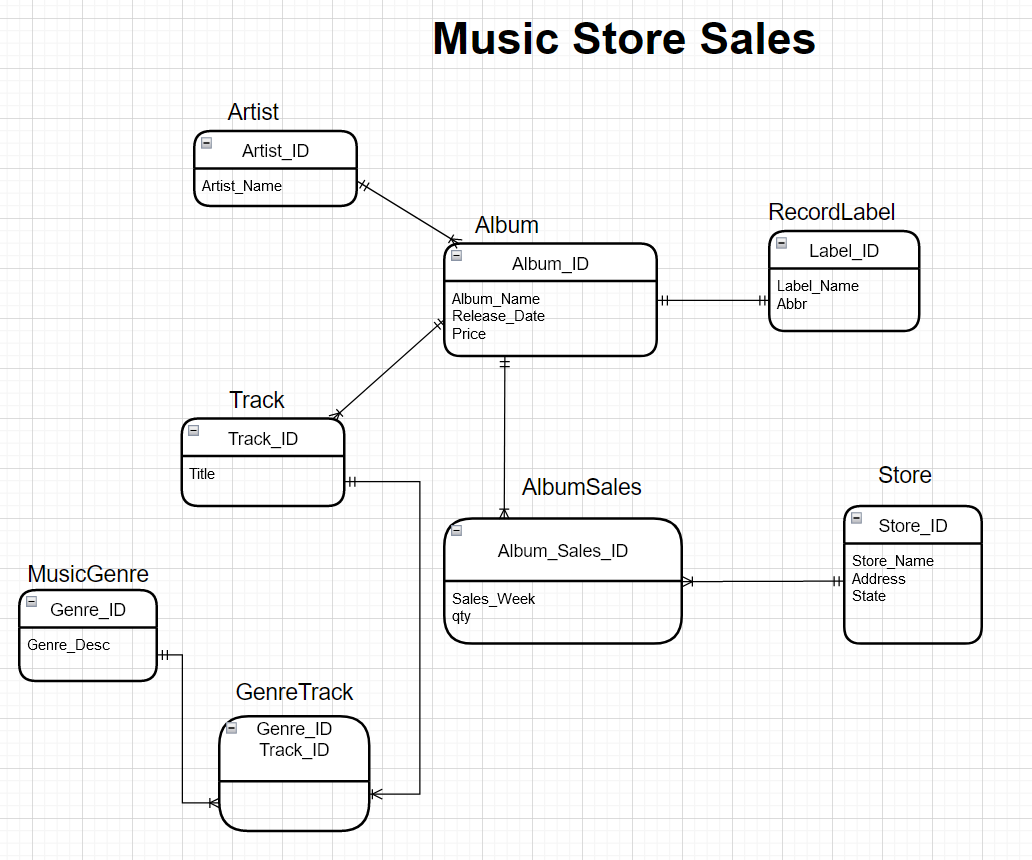
This is an application to track music sales from stores across the country and create charts based on popularity measured in terms of units sold. The system is used to measure the success of songs, albums or artists and is calculated using the information of sales distribution.

Why Database management System is needed:

1. This system requires data storage and management as and when sales data from stores are received.
2. DBMS is used to maintain the store data, sales data and other information including artistdata, album and track details, corresponding genre , sales and label information
3. Through DBMS we can easily retrieve the stored data and can easily perform operations like generating reports ,sales charts in this case

Why application is important from Management Perspective:

1. Analysis of music popularity can create revenue streams in the entertainment industry.
2. If the charts are popular , it will be used by many listeners
   1. Listeners can use the charts to browse through various songs and can also see the popularity of each album etc.
3. **Create a conceptual model for this application. Be sure to include proper names for entities, attributes, and relationships. Identify min/max cardinalities. You may use either the Chen or Crow’s Feet representation. Use a drawing tool to create the conceptual model. There should be 5-8 entities in the conceptual model.**



1. **[20 points]**

**Transform the conceptual model into a relational model and show:**

**1) The transformation rules applied;**

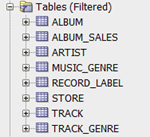
**and**

**2) The final relational model.**

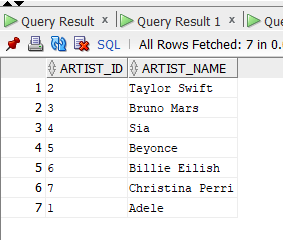
|  |  |  |  |
| --- | --- | --- | --- |
| **Transformation to Relational Model** | | | |
| **Entities ( Tables created for these entities )** | **Attributes** |  |  |
| Artist | Artist\_ID(PK), Artist\_Name |  |  |
| Album | Album\_ID(PK), Album\_Name, Release\_Date,Price,Label\_ID(FK),Artist\_ID(FK) | **one to many ( Artist to Album and label to Album ) :**  One artist can create many albums  One Album can be created by one artist ( not exactly real world)  One album produced by one label. one label can produce many albums | Album table will have **artist\_ID and Label\_ID as foreignkey** |
| Track | Track\_ID(PK), Title, Album\_ID(FK) | one to many( Album to Track):  Album contains many tracks  One track can be part of only one Album | Track table will have **Album\_ID as foreignkey for each track** |
| Store | Store\_ID(PK), Store\_Name, Address, State |  |  |
| Record\_Label | Label\_ID(PK), Label\_Name,ABBR |  |  |
| Music\_Genre | Genre\_ID(PK), Genre\_Desc |  |  |
| Track\_Genre | Track\_ID,Genre\_ID ( Composite PK ) | This table is created as there are many to many relationships between Track and Genre. | **Track\_ID and Genre\_ID** combined to form **composite primary key** |
| Album\_Sales | Album\_Sales\_ID(PK), Qty, Album\_ID(FK),Sales\_Order\_ID(FK),Sales\_Week,Store\_ID(FK) | Table created as there is many to many relationship between store and album. So sales requires separate table | **Album\_ID** and **Store\_ID** are foreign keys here |

1. **Implement the database using your Oracle account provided for this course and populate it with data. Note that you will need to create the data for your relations. Show the populated relations. You can do this by using the “select \* command.**

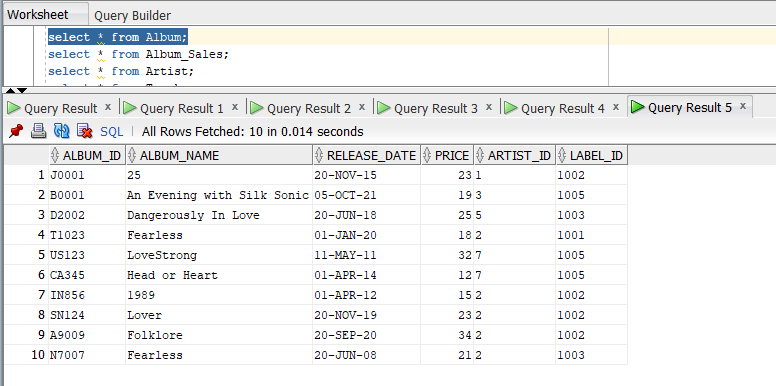
**There should be at least 5-10 entries for each relation. Make sure that the data in the relations are consistent with each other.**



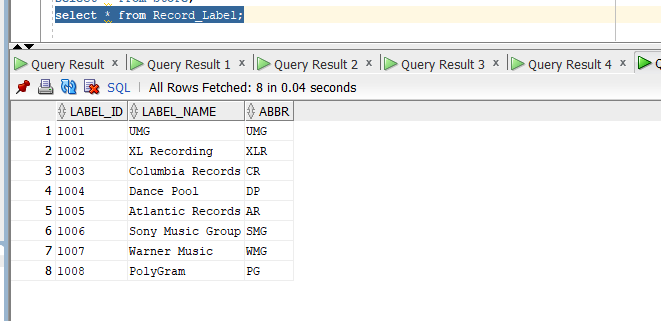
**Artist table:**



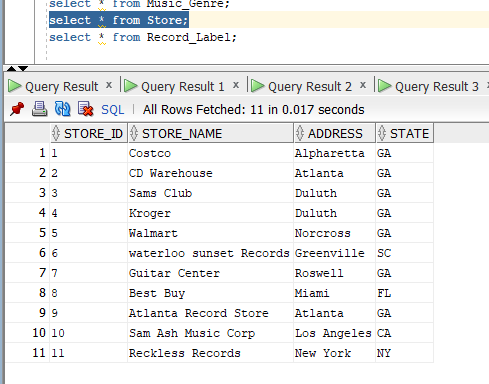
**Album table:**



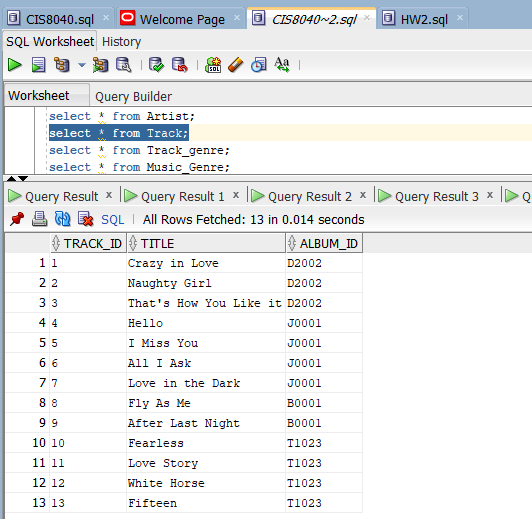
**Record\_Label table:**



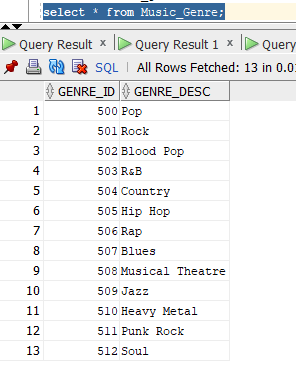
**Store:**



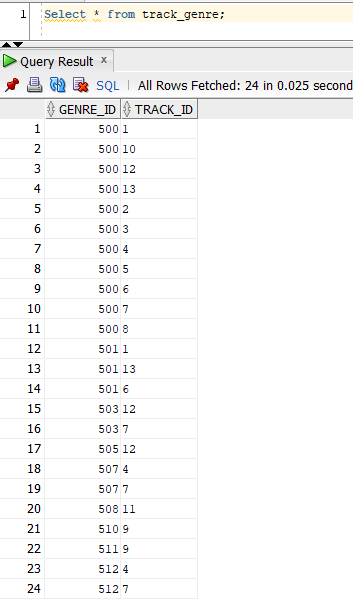
**Track:**



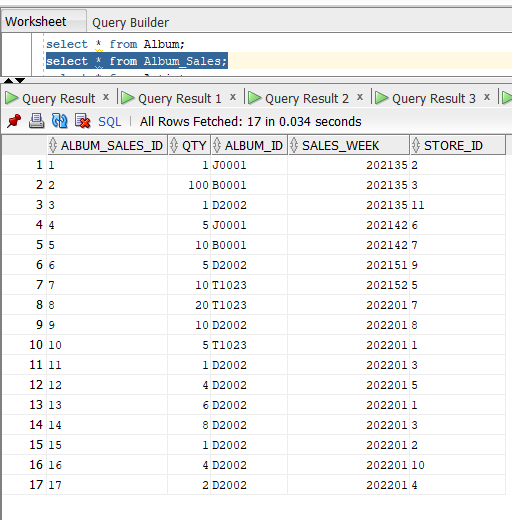
**Music\_Genre table:**



**Track\_Genre: ( Shows the genre mapping to tracks )**



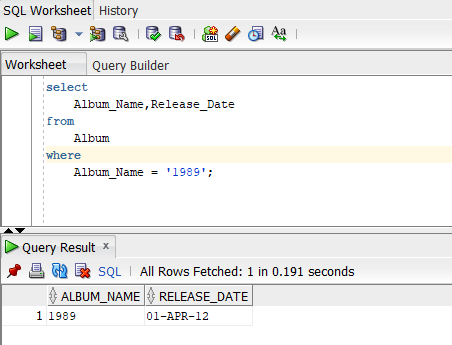
**Album\_Sales:**



1. **Identify and run 7 representative, and non-trivial, queries. Show the results. Queries of the form “select \*” are considered trivial queries and should not be included. Queries of the form “select name from customer;” are also considered to be trivial queries. At least 4 of the queries must involve two or more relations, so you will need to use join commands. Write the queries in English. Then, use screen shots to show the SQL form of the queries and the results obtained from running the queries on the data you created.**

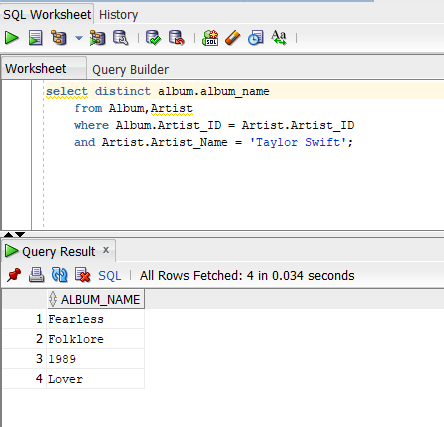
1. List release date of album ‘1989’ and display it with Album Name

Note: Single table query



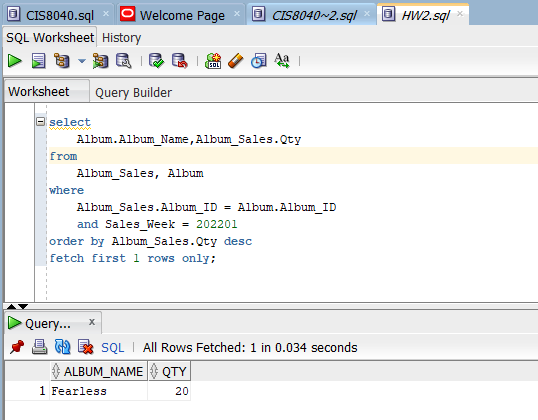
2. List albums of artist "Taylor Swift"

Note : Multi table-Implicit join query



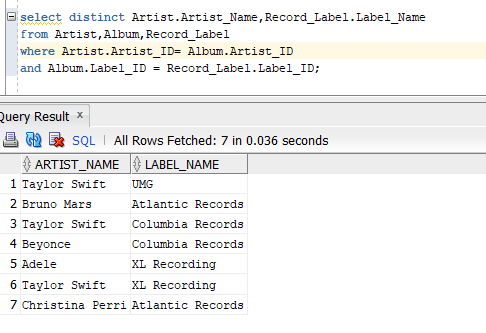
3. List the album that has the most sales in week 202201. List name and quantity of album

Note : Multi table-Implicit join query



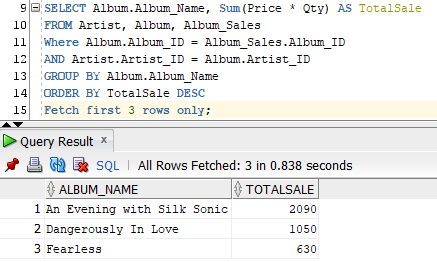
4. List artists and the labels they are associated with

Note : Multi table-Implicit join query



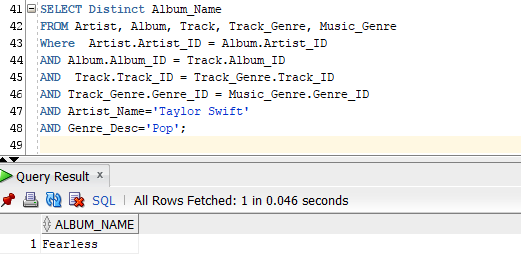
5. List top 3 albums based on total sales

Note : Multi table-Implicit join query



6. List albums of artist 'Taylor Swift' which has ‘pop’ tracks in it ( multitable )

Note : Multi table-Implicit join query



7. List number of albums sold by each state

Note : Multi table-Implicit join query

