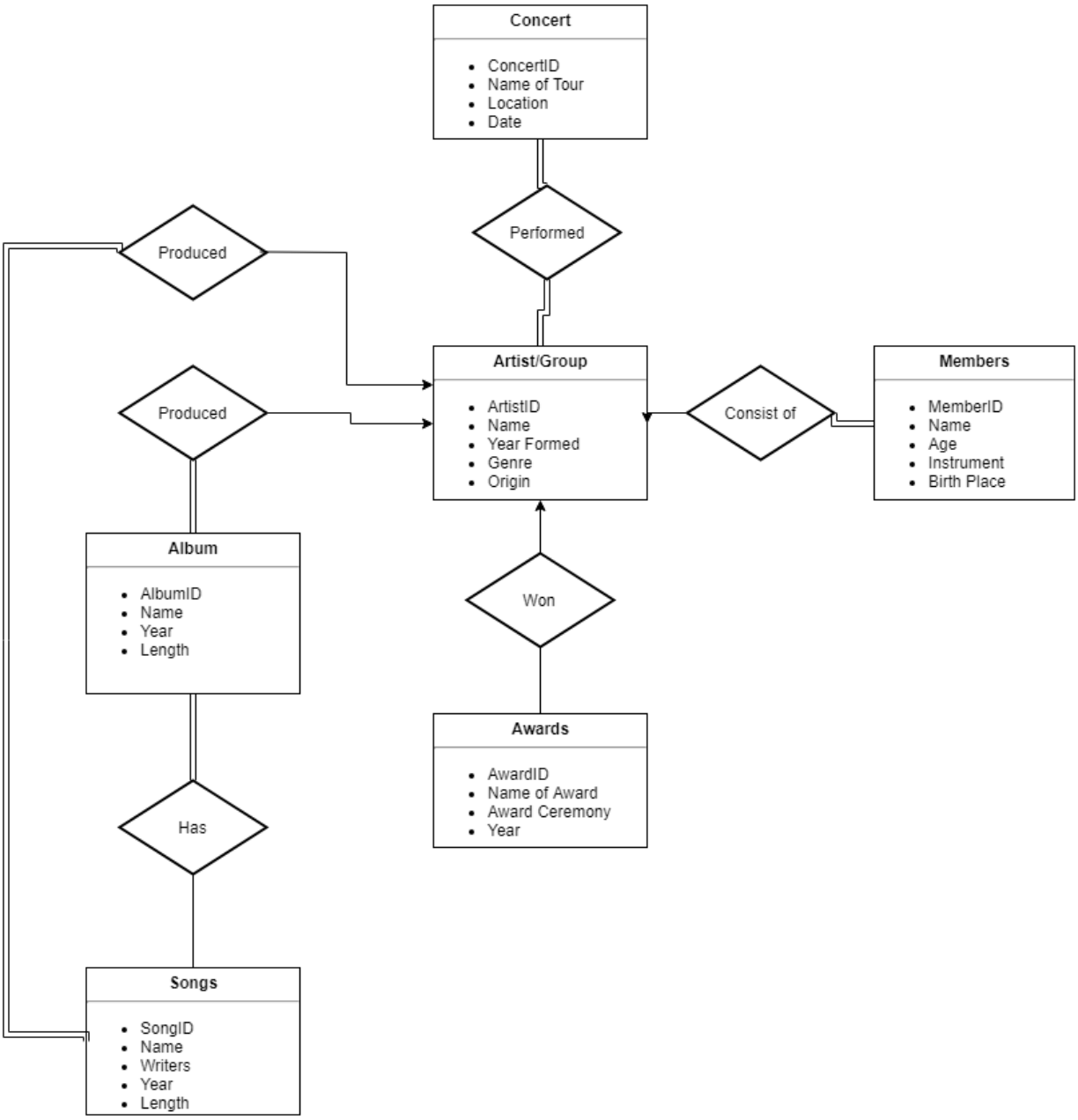
Nicholas Chan, 500652811

COE848 Lab 6

ER Diagram



Above is an image of the ER diagram that was used as a model for my music library. In order to implement XML foreign keys were added to Members, Awards, Album, and Songs which contained the Artist ID (Primary Key of Artist). Performed was also turned into a table containing the primary keys of Concert and Artist, since Concert and Artist were both many to many entities.

XML

In order to create the XML files for each of the entities each attribute was turned into an element that contained a path that contained the proper table it belonged to followed by the primary key of that table. For example:

```
<xs:key name = "artistID">
        <xs:selector xpath = "Artist"/>
        <xs:field xpath = "artistID"/>
</xs:key>
```

The artistID is created within the <xs:key> element with the selector xpath and field xpath nested within the element. This applied to each attribute in the model except foreign keys. The foreign keys were instead referenced like this:

```
<xs:keyref name = "Album_Artist_ID" refer = "artistID">
        <xs:selector xpath = "Album"/>
        <xs:field xpath = "artistID"/>
</xs:keyref>
```

This way each foreign key had a reference to the primary key of the table in which was being referenced. Nested within the <xs:keyref> element was the table that is using the reference as well as the name of the foreign key it is referencing. After each attribute was created the tables were than formed:

```
<xs:element name = "Artist">
    <xs:complexType>
        <xs:sequence>
            <xs:element name = "artistID" type = "xs:int"/>
            <xs:element name = "ArtistName" type = "xs:string"/>
            <xs:element name = "Formed" type = "xs:int"/>
            <xs:element name = "Genre" type = "xs:string"/>
            <xs:element name = "Origin" type = "xs:string"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
```

In the code above it can be seen that once all the attributes were created they were placed in a table named Artist using the code above.

Afterwards the instances for each table were created using the below code:

```
<Artist>
        <artistID>2</artistID>
        <ArtistName>Bring Me the Horizon</ArtistName>
        <Formed>2004</Formed>
        <Genre>Rock</Genre>
        <Origin>United Kingdom</Origin>

</Artist>
```

As it can be seen, the columns in each table will be filled with the information between each of the element tag names, which are nested within the outside tags containing the table name. In order to make the queries the following code was used:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<body>
  <h2> Artist and Album </h2>
  <table border="1">
    <tr bgcolor="#9acd32">
      <th style="text-align:left">Artist</th>
      <th style="text-align:left">Album</th>
    </tr>
    <xsl:for-each select="MusicData/Artist">
    <tr>
      <xsl:variable name="t" select="artistID"/>
      <xsl:variable name="p" select="//MusicData/Album[Album_Artist_ID=$t]"/>
      <td><xsl:value-of select="ArtistName"/></td>
      <td><xsl:value-of select="$p/AlbumName"/></td>
    </tr>
    </xsl:for-each>
  </table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

The <xsl:for-each select="…"> tag is used to select the main table that will be referenced. In order to display the information within the table(s) the <td><xsl:value-of select="…"/></td> tag can be used. If

you need to reference another table variables may be used containing the path to the variable that is being referenced: <xsl:variable name="p" select="//MusicData/path[variable=$t]"/>. If there is a condition, an “if” statement can be inserted surrounding the <td><xsl:value-of select="…"/></td> tag.

Below is a list of the queries and bried explanation of what each query does. The code can be found in the query.txt file that is attached to this project.

Query 1

Artist and Songs

Artist	Genre
Rise Against	Rock
Bring Me the Horizon	Rock
Kid Cudi	Rap
Rush	Rock
Porter Robinson	EDM
Beyonce	RnB
ABBA	Pop
Lady Gaga	Pop
The Weeknd	RnB
The Beatles	Rock

The first query lists all the bands within the database followed by the Genre in which they belong to.

Query 2

Artist and Album

Artist	Album
Rise Against	Siren Song of the Counter Culture
Bring Me the Horizon	Sempiternal
Kid Cudi	Man on the Moon: End of Day
Rush	Moving Pictures
Porter Robinson	Worlds
Beyonce	Dangerously in Love
ABBA	ABBA
Lady Gaga	Born This Way
The Weeknd	Starboy
The Beatles	A Hard Day's Night

This query finds all the artists listed in the database and the albums that correspond to the proper artist.

Query 3

Members with Age Above 40

Member	Age
Joe Principe	43
Neil Peart	65
Alexandar Zivojinovich	64
Agnetha Faltskog	67
Paul McCartney	75

This query searches the members entity and looks for all indiviuals who are above the age of 40.

Query 4

Artist and Songs

Artist	Songs
Rise Against	State of the Union
Rise Against	The First Drop

This query searches looks for all the songs that belong to the artist “Rise Against.”

Query 5

Artist and Album

Artist	Award
The Beatles	Academy Awards
The Beatles	BPI Awards
Beyonce	Billboard Awards
Bring Me the Horizon	Kerrang! Awards
Lady Gaga	American Music Awards
Rush	Juno Awards
Beyonce	Billboard Awards
ABBA	Bravo Otto Awards
The Weeknd	American Music Awards
The Weeknd	Juno Awards

This query uses the Artist and Award entity to search for all artists who have won an award followed by the name of the artist and the award they won.