G64DBS EXERCISE 3: QUERYING THE DATABASE

INTRODUCTION

This exercise will cover a variety of queries you can accomplish using SQL's SELECT statement. If you have completed Exercises 1 and 2 then your database should contain tables for Artist and CD records. In order to add some variety for the queries, we should add a few more CDs. If you'd like more practice with INSERT and UPDATE operations then try to match your tables to these:

```
mysql> SELECT * FROM Artist;
+-------+---------------------------------+
| artID | artName                         |
+-------+---------------------------------+
|     6 | Animal Collective               |
|     3 | DeadMau5                        |
|     7 | Kings of Leon                   |
|     4 | Mark Ronson                     |
|     5 | Mark Ronson & The Business Intl |
|     8 | Maroon 5                        |
|     2 | Mr. Scruff                      |
|     1 | Muse                            |
+-------+---------------------------------+
8 rows in set (0.00 sec)

mysql> SELECT * FROM CD;
+------+-------+-----------------------------+---------+---------------+
| cdID | artID | cdTitle                     | cdPrice | cdGenre       |
+------+-------+-----------------------------+---------+---------------+
|    1 |     1 | Black Holes and Revelations |    9.99 | Rock          |
|    2 |     1 | The Resistance              |   11.99 | Rock          |
|    3 |     2 | Ninja Tuna                  |    9.99 | Electronica   |
|    4 |     3 | For Lack of a Better Name   |    9.99 | Electro House |
|    5 |     4 | Version                     |   11.99 | Rock          |
|    6 |     5 | Record Collection           |   12.99 | Pop           |
|    7 |     6 | Merriweather Post Pavilion  |   12.99 | Electronica   |
|    8 |     7 | Only By The Night           |    9.99 | Rock          |
|    9 |     7 | Come Around Sundown         |   12.99 | Rock          |
|   10 |     8 | Hands All Over              |   11.99 | Pop           |
+------+-------+-----------------------------+---------+---------------+
10 rows in set (0.00 sec)
```

You may have different ID values, do not worry about this. If you feel you don't need to practice any more insertions or updates, then you can recreate these tables exactly using the file setupex3.sql. This is a text file that contains a list of SQL commands. Feel free to open the file in a text editor and have a look. The file will:

1. `DROP` any existing tables named Artist and CD. Do not run this file if you wish to keep databases you have with these names. Also note that if you have foreign key constraints from a Track table to CD, you will need to remove these first.

2. `CREATE` tables for Artist and CD with appropriate `AUTO_INCREMENT` and `KEY CONSTRAINTS` using `ENGINE=InnoDB`.
3. `INSERT` the names of all artists.
4. `INSERT` the CDs for all artists.

Note: You might like to examine the file. The single insert artists command demonstrates how to add multiple rows simultaneously. The multiple insert CD commands show how to use nested SELECT statements. There will be more on that during this exercise.

To run this file, copy it into your home directory on your Linux server. Run `mysql`, and type:

mysql> source setupex3.sql

You should see multiple commands executed; your tables should now match those above.

For each exercise below, you can find the correct output at the end of this document. There is often more than one solution to a query problem; the most important thing is that your results are correct. The ID numbers will vary depending on how your database is set up, and the order of output is also unimportant unless you have used an `ORDER BY` clause.

SQL SELECT

SQL's SELECT statement has a number of different options, some of which we've already seen. The simplest form of a SELECT statement returns all of the information from a single table. The general format is:

```
SELECT * FROM tableName;
```

There are many more options to build up more complex queries. In this exercise we shall cover:

- SELECTing specific columns from a table
- Using WHERE clauses to select specific rows
- Using ORDER BY to present the results in a specified order
- SELECTing from multiple tables
- Using subqueries
- Using aggregate functions, with GROUP BY and HAVING to produce summary information based on a table

Exercise: Write a SELECT statement to output all of the information from the Artist table.

SELECTING SPECIFIC COLUMNS

Often tables in a database have many columns, and we only require a few. A good example of this is the information_schema.tables table, which can list information on all of your tables. You can retrieve a list of columns in this table like this:

```
DESCRIBE information_schema.tables;
```
As you can see, there are a lot of columns, we can select useful ones and use aliases to make them more understandable:

```
SELECT table_name as Name, table_rows as Rows, table_schema, engine
FROM information_schema.tables;
```

You can now see the tables you have created where `table_schema` equals your username. The aliases give the columns names to make them easier to understand. Let's try doing the same with our CD table:

```
SELECT cdTitle AS Title, cdGenre AS Genre FROM CD;
```

Exercise: Write SQL Statements to do the following (Output can be found at the end of this document):

- List the names of the artists in the database [Output 1]
- List the genres of the CDs in the database [Output 2]
- List the titles and prices of the CDs in the database [Output 3]

You will find that the genre output contains multiple rows for the same genres. You can avoid this by using the `DISTINCT` keyword in your select statement:

```
SELECT DISTINCT cdGenre AS Genre FROM CD;
```

Exercise: Write a SELECT statement to find the artist IDs and genres from the CD table, without duplicate entries. [Output 4]

USING WHERE CLAUSES

Using a WHERE clause you can return rows from tables that match specific criteria. Some examples of simple where clauses using a single test are:

- `SELECT * FROM CD WHERE cdPrice > 10.00;`
- `SELECT * FROM CD WHERE cdGenre = 'Rock';`
- `SELECT * FROM CD WHERE cdGenre <> 'Rock';`
- `SELECT * FROM Artist WHERE artName LIKE 'Ma%';`

The LIKE keyword finds similar strings, and can use the '%' character as a wildcard to represent any number of characters. In the last example, the expression would return "Mark Ronson", "Mark Ronson & The Business Intl" and "Maroon 5.

Exercise: Write SQL statements to do the following:

- Find a list of all information on Electronica CDs [Output 5]
- Find a list of the titles of all CDs that cost less than 10.00 [Output 6]
- Find a list of CD titles and prices where the title contains a space [Output 7]

More complex conditions can be created using the operators AND, OR and NOT. Brackets can be used to control the order of evaluation. For example:

```
SELECT * FROM CD WHERE (cdPrice < 10.00) AND NOT (cdGenre='Rock');
```

Will return a list of CDs that cost less than 10.00, but are not Rock CDs.

Exercise: Write SQL queries to:

- Find a list of all information on CDs that have the genre 'Pop' or cost more than 10.00 [Output 8]
- Find a list of titles of CDs that are Rock albums costing between 10.00 and 12.00 [Output 9]
- Find a list of the titles of CDs that cost less than 10.00, and are either "Rock", or "Electronica". [Output 10]

USING ORDER BY

You can order results in the output should we wish the output to be in a more readable form. The ORDER BY clause allows us to specify columns whose data will order the rows, and whether the ordering is ascending or descending. For example, to order CDs by title alphabetically, you would use:

SELECT * FROM CD ORDER BY cdTitle;

By default, results are sorted in ascending order when an order by is used. To specify which we require, we use ASC or DESC:

SELECT * FROM CD ORDER BY cdTitle ASC;

SELECT * FROM CD ORDER BY cdTitle DESC;

You can order first by one column, then by another. This can be done for any number of columns. For example, to order first by genre, then by title we would use:

SELECT * FROM CD ORDER BY cdGenre, cdTitle;

Exercise: Write SQL queries to:

- List the artist names in alphabetical order [Output 11]
- List the titles and prices of CDs in order of price from highest to lowest [Output 12]
- List the titles and prices of CDs in order of price from lowest to highest [Output 13]
- List the titles, genres and prices CDs in alphabetical order by Genre, then by price from highest to lowest [Output 14]

SELECTING FROM MULTIPLE TABLES

A database will normally contain a number of tables. It is often useful to combine information from several tables in a single query. Usually we will use a where clause to clarify which results we want, that is not always the case however. The following example will select every combination of rows from Artist and CD:

SELECT * FROM Artist, CD;

This is a CROSS JOIN, sometimes referred to as a product. In this case, the select query will return 10 * 8 rows, or 80 rows. This is far too many to be useful in this case, we should use a where clause, or a different join to simplify results. The other types of joins we might use are:

INNER JOIN: Returns results based on a condition specified with a USING or ON clause.

NATURAL JOIN: Returns results where information in identically named columns from both tables

Remember, exactly how you implement your query is not as important as whether it is correct! The next exercises, for example, have many solutions.

Exercise: Write an SQL query to:

- Return all the information from Artist and CD where the artID records are the same for both tables [Output 15]
- Find a list of the titles of all CDs by Muse [Output 16]
- Find a list of the names of all artists, without duplication, who have produced a rock album [Output 17]
- Find a list of the artist names, titles and genres for all CDs that cost less than 12.00, sorted first by name, then by CD title. [Output 18]

SUBQUERIES

Combining multiple tables can lead to extremely complex queries very quickly. It is sometimes easier to use subqueries to pass values into other queries. For example, suppose we want to find out a list of CDs that cost the same as 'Ninja Tuna':

```
SELECT cdTitle FROM CD
      WHERE cdPrice =
      (SELECT cdPrice FROM CD WHERE cdTitle = 'Ninja Tuna');
```

It is important to remember that there are various ways of comparing values with the results of a subquery. What you do is dependent not only on what results you want, but also how many values you can expect the subquery to return. In general:

- If the subquery returns a single value, you can use the normal conditional operators (=,<,>,<>,etc.)
- If the subquery returns multiple values, you can use IN, ANY, ALL and NOT to compare a single value to a set
- You can use (NOT) EXISTS to see if a set is empty or not.

For example, to find a list of all CDs of the same genre as any that the artist "Mark Ronson%" has produced, we can use the following:

```
SELECT cdTitle FROM CD
     WHERE cdGenre IN (SELECT cdGenre FROM CD, Artist
                          WHERE Artist.artID = CD.cdID
                          AND artName LIKE 'Mark Ronson%");
```

Exercise: Write the following queries:

- Use a subquery to find a list of CDs that have the same genre as 'The Resistance' [Output 19]
- Use IN to find a list of the titles of albums that are the same price as any 'Pop' album [Output 20]
- Use ANY to find the titles of CDs that cost more than at least one other CD [Output 21]
- Use ALL to find a list of CD titles that cost more or the same as all other CDs [Output 22]

USING AGGREGATE FUNCTIONS, GROUP BY AND HAVING

The last feature of SQL SELECT taught in this course is the use of aggregate functions. Used with GROUP BY and HAVING clauses, these can produce summary information from a table, or set of tables. The common functions you will need to know for this course are:

- `COUNT([columns])` returns the number of rows in that column. You can use `COUNT(*)` to count the number of rows returned by a select statement. You can also use `COUNT(DISTINCT column)` to count the number of distinct values for a specific column.
- `SUM (column)` returns the sum of all values in a column
- `MAX (column)` returns the maximum value
- `MIN (column)` returns the minimum value
- `AVG (column)` returns the average value

The mathematical functions above require the data is of a numeric type like REAL or INT.

Exercise: Write queries to do the following:

- Find the lowest price of any CD [Output 23]

- Find the number of CDs costing 11.99 [Output 24]
- Find the title of the most expensive rock CD(s) [Output 25]
- Find the number of different Genres in the CD table [Output 26]
- List the information on the cheapest CDs [Output 27]

As above, aggregate functions will normally work on an entire table, unless you use a GROUP BY clause. This is done by selecting some normal columns, and aggregate functions, then grouping the results. For example, to find the number of CDs in each genre:

```
SELECT cdGenre AS Genre, COUNT(*) FROM CD GROUP BY cdGenre;
```

We can then use a HAVING clause, which is essentially the same as a where. The difference is that a WHERE is applied before GROUP BY, the HAVING clause is applied after rows have been grouped, and aggregate functions have been calculated. To find the number of CDs in each Genre, but only where that count is greater than 3:

```
SELECT cdGenre AS Genre, COUNT(*) AS Count FROM CD
      GROUP BY cdGenre HAVING Count > 3;
```

Notice that it is sometimes convenient to give an aggregate function an alias.

Exercise: Write queries to do the following:

- Find a list of artist names, the number of CDs they have produced, and the average price for their CDs. Only return results for artists with more than one CD. [Output 28]
- Find a list of artist names, the number of CDs by that artist and the average price for their CDs but not including 'Electronica' albums (you might like to use a WHERE in this one too) [Output 29]

FURTHER EXERCISES

The above exercises should give you a good grounding in all you may need to do for the SELECT components of the coursework and exam. For those that would like an even harder challenge:

Exercise: Write an SQL select statement to:

- Find the difference between the average price of an album by the artist 'Muse', and the average price of all albums in the database. (ABS() will produce the absolute value of a calculation, rather than positive or negative) [Output 30]
- Find the most expensive genre of music by calculating the maximum of the averages of all genres [Output 31]
- Find the artist name(s) with the most expensive CDs by average. [Output 32]

CORRECT OUTPUT

Here you can see the expected output of all the queries in this exercise. Remember, some IDs or column names might be different if you've created them differently. In general I have used aliases to make the column names a little neater and more informative. The important thing is that your statements produce the same information as the results here.

OUTPUT 1

```
+---------------------------------+
| Name                            |
+---------------------------------+
| Animal Collective               |
| DeadMau5                        |
| Kings of Leon                   |
| Mark Ronson                     |
| Mark Ronson & The Business Intl |
| Maroon 5                        |
| Mr. Scruff                      |
| Muse                            |
+---------------------------------+
```

OUTPUT 2

```
+---------------+
| Genre         |
+---------------+
| Rock          |
| Rock          |
| Electronica   |
| Electro House |
| Rock          |
| Pop           |
| Electronica   |
| Rock          |
| Rock          |
| Pop           |
+---------------+
```

OUTPUT 3

```
+-----------------------------+-------+
| Title                       | Price |
+-----------------------------+-------+
| Black Holes and Revelations |  9.99 |
| The Resistance              | 11.99 |
| Ninja Tuna                  |  9.99 |
| For Lack of a Better Name   |  9.99 |
| Version                     | 11.99 |
| Record Collection           | 12.99 |
| Merriweather Post Pavilion  | 12.99 |
| Only By The Night           |  9.99 |
| Come Around Sundown         | 12.99 |
| Hands All Over              | 11.99 |
+-----------------------------+-------+
```

OUTPUT 4

ID	cdGenre
1	Rock
2	Electronica
3	Electro House
4	Rock
5	Pop
6	Electronica
7	Rock
8	Pop

OUTPUT 5

cdID	artID	cdTitle	cdPrice	cdGenre
3	2	Ninja Tuna	9.99	Electronica
7	6	Merriweather Post Pavilion	12.99	Electronica

OUTPUT 6

Title
Black Holes and Revelations
Ninja Tuna
For Lack of a Better Name
Only By The Night

OUTPUT 7

Title	Price
Black Holes and Revelations	9.99
The Resistance	11.99
Ninja Tuna	9.99
For Lack of a Better Name	9.99
Record Collection	12.99
Merriweather Post Pavilion	12.99
Only By The Night	9.99
Come Around Sundown	12.99
Hands All Over	11.99

OUTPUT 8

cdID	artID	cdTitle	cdPrice	cdGenre
2	1	The Resistance	11.99	Rock
5	4	Version	11.99	Rock
6	5	Record Collection	12.99	Pop
7	6	Merriweather Post Pavilion	12.99	Electronica
9	7	Come Around Sundown	12.99	Rock
10	8	Hands All Over	11.99	Pop

OUTPUT 9

```
+----------------+
| Title          |
+----------------+
| The Resistance |
| Version        |
+----------------+
```

OUTPUT 10

```
+-----------------------------+
| Title                       |
+-----------------------------+
| Black Holes and Revelations |
| Ninja Tuna                  |
| Only By The Night           |
+-----------------------------+
```

OUTPUT 11

```
+---------------------------------+
| Name                            |
+---------------------------------+
| Animal Collective               |
| DeadMau5                        |
| Kings of Leon                   |
| Mark Ronson                     |
| Mark Ronson & The Business Intl |
| Maroon 5                        |
| Mr. Scruff                      |
| Muse                            |
+---------------------------------+
```

OUTPUT 12

```
+-----------------------------+-------+
| Title                       | Price |
+-----------------------------+-------+
| Record Collection           | 12.99 |
| Come Around Sundown         | 12.99 |
| Merriweather Post Pavilion  | 12.99 |
| Hands All Over              | 11.99 |
| Version                     | 11.99 |
| The Resistance              | 11.99 |
| For Lack of a Better Name   |  9.99 |
| Ninja Tuna                  |  9.99 |
| Only By The Night           |  9.99 |
| Black Holes and Revelations |  9.99 |
+-----------------------------+-------+
```

It’s fine if the titles are in different orders for CDs of the same price.

OUTPUT 13

Title	Price
Black Holes and Revelations	9.99
Only By The Night	9.99
Ninja Tuna	9.99
For Lack of a Better Name	9.99
Hands All Over	11.99
Version	11.99
The Resistance	11.99
Merriweather Post Pavilion	12.99
Come Around Sundown	12.99
Record Collection	12.99

It's fine if the titles are in different orders for CDs of the same price.

OUTPUT 14

Title	Genre	Price
For Lack of a Better Name	Electro House	9.99
Merriweather Post Pavilion	Electronica	12.99
Ninja Tuna	Electronica	9.99
Record Collection	Pop	12.99
Hands All Over	Pop	11.99
Come Around Sundown	Rock	12.99
The Resistance	Rock	11.99
Version	Rock	11.99
Black Holes and Revelations	Rock	9.99
Only By The Night	Rock	9.99

OUTPUT 15

artID	artName	cdID	cdTitle	cdPrice	cdGenre
6	Animal Collective	7	Merriweather Post Pavilion	12.99	Electronica
3	DeadMau5	4	For Lack of a Better Name	9.99	Electro House
7	Kings of Leon	8	Only By The Night	9.99	Rock
7	Kings of Leon	9	Come Around Sundown	12.99	Rock
4	Mark Ronson	5	Version	11.99	Rock
5	Mark Ronson & The Business Intl	6	Record Collection	12.99	Pop
8	Maroon 5	10	Hands All Over	11.99	Pop
2	Mr. Scruff	3	Ninja Tuna	9.99	Electronica
1	Muse	1	Black Holes and Revelations	9.99	Rock
1	Muse	2	The Resistance	11.99	Rock

OUTPUT 16

Title
Black Holes and Revelations
The Resistance

OUTPUT 17

Name
Kings of Leon
Mark Ronson
Muse

OUTPUT 18

Name	Title	Genre
DeadMau5	For Lack of a Better Name	Electro House
Kings of Leon	Only By The Night	Rock
Mark Ronson	Version	Rock
Maroon 5	Hands All Over	Pop
Mr. Scruff	Ninja Tuna	Electronica
Muse	Black Holes and Revelations	Rock
Muse	The Resistance	Rock

OUTPUT 19

Title
Black Holes and Revelations
The Resistance
Version
Only By The Night
Come Around Sundown

OUTPUT 20

Title
The Resistance
Version
Record Collection
Merriweather Post Pavilion
Come Around Sundown
Hands All Over

OUTPUT 21

Title
The Resistance
Version
Record Collection
Merriweather Post Pavilion
Come Around Sundown
Hands All Over

OUTPUT 22

Title
Record Collection
Merriweather Post Pavilion
Come Around Sundown

OUTPUT 23

```
+---------+
| Minimum |
+---------+
|    9.99 |
+---------+
```

OUTPUT 24

```
+-------+
| Total |
+-------+
|     3 |
+-------+
```

OUTPUT 25

```
+---------------------+
| Title               |
+---------------------+
| Come Around Sundown |
+---------------------+
```

OUTPUT 26

```
+-------+
| Total |
+-------+
|     4 |
+-------+
```

OUTPUT 27

```
+------+-------+-----------------------------+---------+---------------+
| cdID | artID | cdTitle                     | cdPrice | cdGenre       |
+------+-------+-----------------------------+---------+---------------+
|    1 |     1 | Black Holes and Revelations |    9.99 | Rock          |
|    3 |     2 | Ninja Tuna                  |    9.99 | Electronica   |
|    4 |     3 | For Lack of a Better Name   |    9.99 | Electro House |
|    8 |     7 | Only By The Night           |    9.99 | Rock          |
+------+-------+-----------------------------+---------+---------------+
```

OUTPUT 28

```
+---------------+---------------+---------------+
| Name          | Number of CDs | Average Price |
+---------------+---------------+---------------+
| Muse          |             2 |         10.99 |
| Kings of Leon |             2 |         11.49 |
+---------------+---------------+---------------+
```

OUTPUT 29

```
+---------------------------------+---------------+---------------+
| Name                            | Number of CDs | Average Price |
+---------------------------------+---------------+---------------+
| Muse                            |             2 |         10.99 |
| DeadMau5                        |             1 |          9.99 |
| Mark Ronson                     |             1 |         11.99 |
| Mark Ronson & The Business Intl |             1 |         12.99 |
| Kings of Leon                   |             2 |         11.49 |
| Maroon 5                        |             1 |         11.99 |
+---------------------------------+---------------+---------------+
```

OUTPUT 30

```
+-------------------+
| Difference        |
+-------------------+
| 0.499999999999998 |
+-------------------+
```

OUTPUT 31

```
+-------+---------+
| Genre | Average |
+-------+---------+
| Pop   |   12.49 |
+-------+---------+
```

OUTPUT 32

```
+--------------------------------+
| Name                           |
+--------------------------------+
| Animal Collective              |
| Mark Ronson & The Business Intl |
+--------------------------------+
```

USEFUL COMMANDS

Here are some SQL/MySQL commands you may find useful:

```
mysql> describe tableName;
```

This command will provide an overview of your table, including columns, names, data types and keys.

```
mysql> show tables;
```

This command shows a list of names all of your tables. Other information can be seen by querying the `information_schema.tables` table.

```
mysql> SHOW CREATE TABLE tableName;
```

This command will show you the create table command for a given table. It will list all columns and constraints. This is useful if you can't remember what constraints you have on that table, or what they are called.

```
mysql> system <command>
```

The system command allows you to enter unix commands without leaving `mysql`. For example, `system clear` will clear the screen.

```
mysql> ALTER TABLE tableName AUTO_INCREMENT=0;
```

If you are using a table with an auto increment column, this will reset the current value to 0. This is useful if you've deleted all of the rows in your table, and want the automatically generated values to start at 0, or some other specific number.

```
mysql> source <filename>
```

This will execute any SQL code you provide it in a text file. You might find this an easier way of working than entering line by line. You must execute mysql from the directory that the text file is stored in. The common extension for these files is .sql. If you use notepad++ (installed on the school servers) then it will do syntax highlighting.