Lab Manual for Introduction to Database Systems

Lab-04: SQL SELECT (FUNCTIONS, AGGREGATE FUNCTIONS AND ARITHMETIC OPERATIONS)

Table of Contents

Lab 4: SQL SELECT (FUNCTIONS, AGGREGATE FUNCTIONS AND ARITHMETIC OPERATIONS)

1. Introduction

This lab is a continuation of the last lab. So far, we have learned how to query subset of data from the database using SELECT statement, we have simply displayed the results of any query. MySQL can do more that this and has many built in functions that can transform data to meet our requirements. These include:

- *Date Functions* are used to manipulate the display format of a date as well as calculate time. i.e. Formatting the date using *DATE_FORMAT()* function, or getting year from date using *YEAR()* function.
- *String Functions* can format or manipulate a text string. i.e. Adding text to an existing value using *CONCAT()* function or getting a substring from text using *RIGHT*, *LEFT*, or *MID* functions.
- *Numeric Functions* can format or manipulate figures/numbers. i.e. getting floor or ceiling value using *FLOOR()* or *CEILING()* functions respectively.
- Aggregate/Summarizing Functions are used to get a summarized/aggregate result from a query. i.e. counting the number of rows in the *ResultSet* of a query using *COUNT()* function.

There are also *Control Functions* that can be used to give conditionality to queries. In this lab, you can learn about the select command using functions.

Relevant Lecture Material

a) Text Book: Java: Text Book: Database Systems, A practical approach to design, implementation and management by Thomas Connolly, Carolyn Begg, Addison Wesley, Fifth Edition,

1. Read URL:
   - https://www.w3schools.com/sql/sql_min_max.asp
   - https://www.w3schools.com/sql/sql_count_avg_sum.asp

2. Activity Time boxing

Table 1: Activity Time Boxing

Task No.	Activity Name	Activity time	Total Time
6.3	Walkthrough Tasks	30mins	60mins
7	Practice tasks	20 to 30mins for each task	50mins
8	Evaluation Task	40mins for all assigned task	40mins

3. Objective of the experiment

- To get basic understanding of MySQL functions i.e. DATE functions, String function, Numeric functions, and Aggregate functions.
- To get familiar with querying a database using SELECT statement with functions.

4. Concept Map

In this section, a brief overview of the concepts is presented, those will be used in this lab afterwards. MySQL has many built in functions that can transform data to meet our requirements. These include:

- **Date Functions**
- **String Functions**
- **Numeric Functions**
- **Aggregate/Summarizing Functions**

The following sections describe each category one by one.

4.1. Date/Time Functions

MySQL provides these built-in functions for getting the *current* date, time and datetime:

- NOW: returns the current date and time in the format of 'YYYY-MM-DD HH:MM:SS'.
- CURDATE (or CURRENT_DATE(), or CURRENT_DATE): returns the current date in the format of 'YYYY-MM-DD'.
- CURTIME (or CURRENT_TIME(), or CURRENT_TIME): returns the current time in the format of 'HH:MM:SS'.

For Example: We use the above built in function in a query like given below:

SELECT NOW (), CURDATE (), CURTIME ();

4.2. String Functions

String values are can be explained as 'bits of text' and much like the date functions, the string functions allow us to manipulate these values before they are displayed. Although there are once more many different functions, we are going to concentrate on the functions that fall into a few broad categories.

- Adding text to an existing value
- Changing Part of a String
- Finding a piece of text in a string

4.2.1. Adding text to an existing value

There are two simple ways to add more text to an existing value - either at the start or end of the text. Placing the text at either end is best achieved with the CONCAT() function.

Syntax:

```
CONCAT(string1,string2,...)
```

Thus we can take an existing value (say **string2**) and place a new value (**string1**) at the beginning to get **string1string2**.

For example: Concat first name and job id of employees table:

SELECT CONCAT(first_name, '-', job_id) as 'Employee Detail' FROM employees;

4.2.2. Changing Part of a String

As well as add text we can replace it or overwrite it completely. To replace an instance of text within a string we can use the REPLACE() function.

Syntax:

```
REPLACE(column name, original string ,replacement string)
```

For Example: Replace the name of Steven to Steven hock in employee table. For this we have used replace function which temporally replace the name.

SELECT REPLACE(employees.first_name, 'Steven', 'Steven hock') from employees where employee_id=100

4.2.3. Finding a piece of text in a string.

In some of the string functions we have seen so far it has been necessary to provide a starting position as part of the function This position can be found using the LOCATE() function specifying the text to find (**substring**) as well as the **string** to search in.

Syntax:

```
LOCATE(substring, string)
```

For Example: locate the string king in employee last name attribute so

SELECT LOCATE ('King', employees.last_name) from employees;

4.3. Numeric Functions

Before talking about the specific numeric functions, it is probably worth mentioning that MySQL can perform simple math functions using mathematical operators.

Operator	Function
+	Add
-	Subtract
*	Multiply
/	Divide

Examples:

SELECT 6/2, 6*8, 6+3, 6-5;

4.4. Aggregate/Summarizing Functions

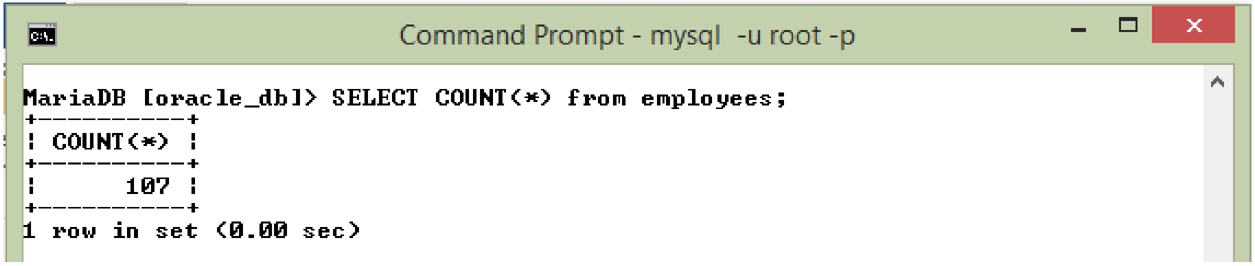
Aggregate function performs a calculation on a set of values and returns a single value. MySQL provides many aggregate functions that include AVG, COUNT, SUM, MIN, MAX, etc. An aggregate function ignores NULL values when it performs calculation except for the COUNT function.COUNT()

This counts the number of times a row (or field) is returned.

Syntax:

```
COUNT(field)
```

For Example: If count the all rows of employee table then use count function



```
MariaDB [oracle_db]> SELECT COUNT(*) from employees;
+----------+
| COUNT(*) |
+----------+
|      107 |
+----------+
1 row in set (0.00 sec)
```

Figure 1: Count Function

4.5.1. AVG()

The next function we are going to look at is the AVG() which unsurprisingly is the average function.

Syntax:

```
AVG(field)
```

For Example: Show the average salary paid by department whose number is 90.

SELECT AVG(salary) from employees WHERE department_id=90;

4.5.2. MIN() and MAX()

These functions are very similar and select the lowest and highest figure respectively from a result set.

Syntax:

```
MIN(field)
MAX(field)
```

For Example: Show the Min salary paid by department whose number is 90.

SELECT MIN(salary) from employees WHERE department_id=90;

For Example: Show the MAX salary paid by department whose number is 90.

SELECT MAX(salary) from employees WHERE department_id=90;

4.5.3. SUM()

The final summary function that we will look at is the SUM() function which adds rows of one field in the results set together.

Syntax:

```
SUM(field)
```

For Example: Show the Total salary paid by department whose number is 90.

SELECT SUM(salary) from employees WHERE department_id=90;

4. Homework before Lab

You must solve the following problems at home before the lab.

5.1.Problem Solution Modeling

After reading the reference material mentioned in the introduction, now you are ready to perform homework assigned to you.

5.1.1. Problem description:

Describe the date functions, string functions, numeric functions, aggregate/summarizing functions and its purpose. You must create at least 10 examples of group by and having clause other than your practice tasks, submit to lab teacher in hard form.

5. Procedure& Tools

In this section, you will study how to make and run a customized exception.

6.1.Tools

In this section tools installation and setup is defined.

6.2.Setting-up and Setting up XAMPP (MySQL, Apache) [Expected time = 5mins]

Refer to Lab 1 sec 6.2.

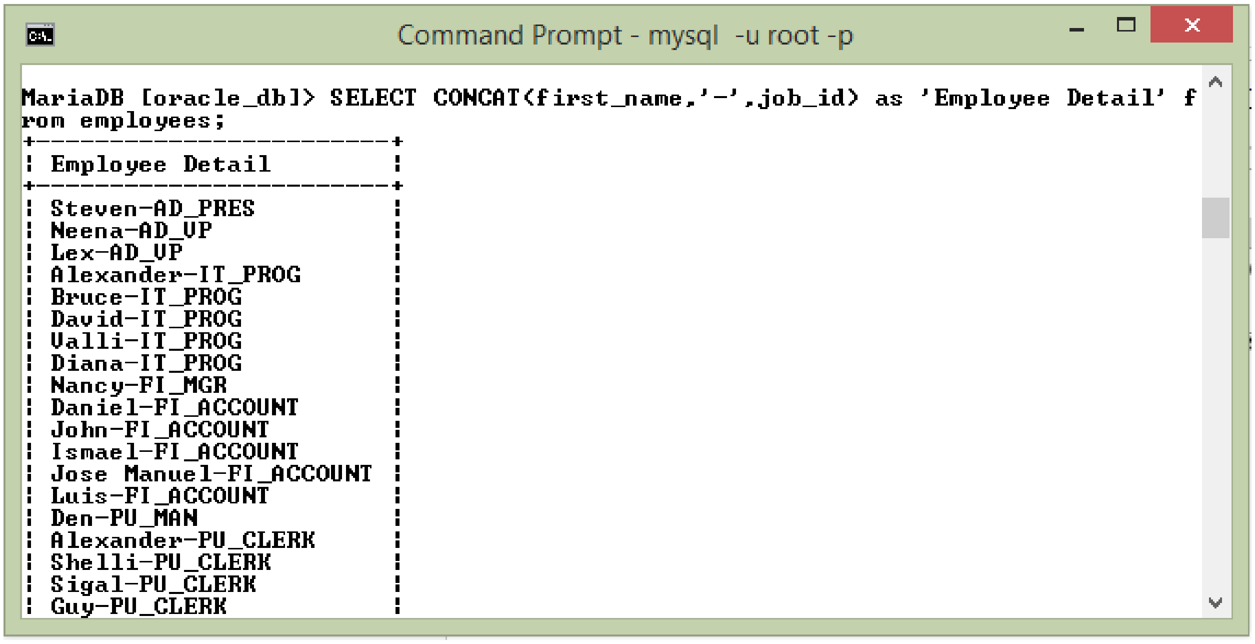
6.3.Walkthrough Task [Expected time = 30mins]

6.3.1. String Functions

String Functions can format or manipulate a text string. You can concatenate a few columns as one using function CONCAT ().

For example: Concat first name and job id of employees table:

SELECT CONCAT(first_name, ‘-’, job_id) as ‘Employee Detail’ FROM employees;

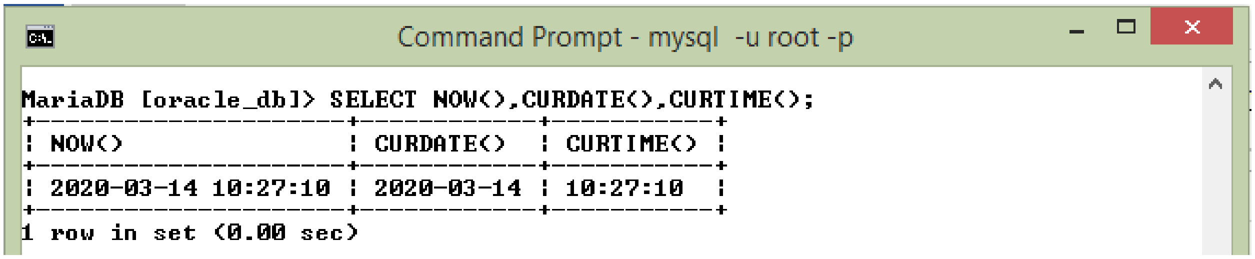


```
MariaDB [oracle_db]> SELECT CONCAT(first_name,'-',job_id) as 'Employee Detail' f
rom employees;
+--------------------------+
| Employee Detail          |
+--------------------------+
| Steven-AD_PRES           |
| Neena-AD_VP              |
| Lex-AD_VP                |
| Alexander-IT_PROG        |
| Bruce-IT_PROG            |
| David-IT_PROG            |
| Valli-IT_PROG            |
| Diana-IT_PROG            |
| Nancy-FI_MGR             |
| Daniel-FI_ACCOUNT        |
| John-FI_ACCOUNT          |
| Ismael-FI_ACCOUNT        |
| Jose Manuel-FI_ACCOUNT   |
| Luis-FI_ACCOUNT          |
| Den-PU_MAN               |
| Alexander-PU_CLERK       |
| Shelli-PU_CLERK          |
| Sigal-PU_CLERK           |
| Guy-PU_CLERK             |
```

Figure 2: CONCAT Function

6.3.2. Date/Time Functions

For Example: We use the above built in function in a query like given below:
SELECT NOW (), CURDATE (), CURTIME ();

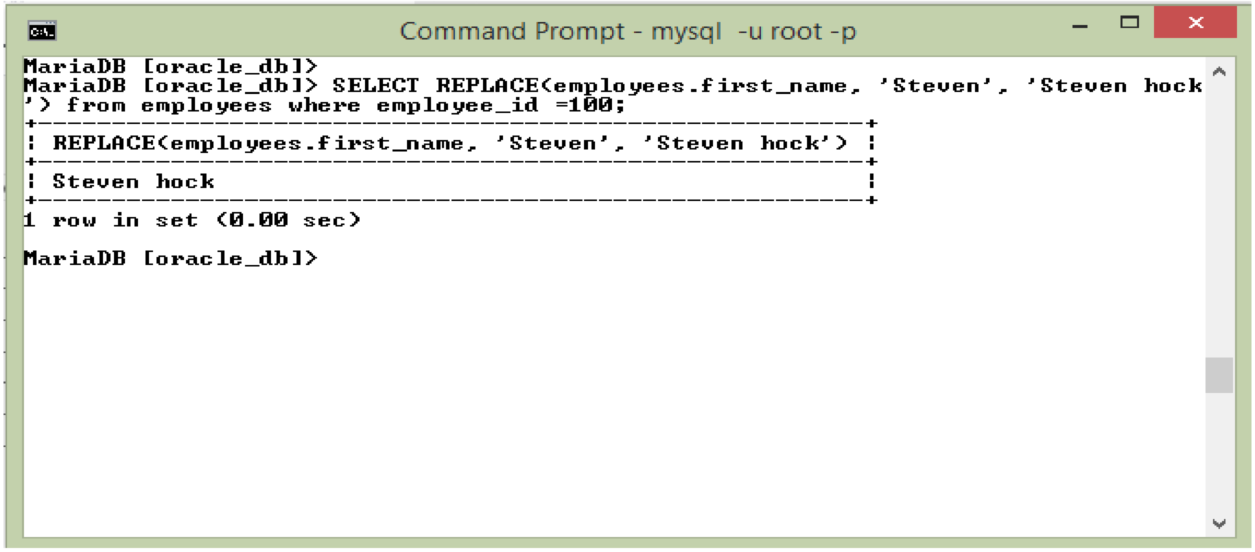


```
MariaDB [oracle_db]> SELECT NOW(),CURDATE(),CURTIME();
+---------------------+------------+-----------+
| NOW()               | CURDATE()  | CURTIME() |
+---------------------+------------+-----------+
| 2020-03-14 10:27:10 | 2020-03-14 | 10:27:10  |
+---------------------+------------+-----------+
1 row in set (0.00 sec)
```

Figure 3: Date Time Function

6.3.3. Replace Function:

For Example: Replace the name of Steven to Steven hock in employee table. For this we have used replace function which temporally replace the name.
SELECT REPLACE(employees.first_name, ‘Steven’, ‘Steven hock’) from employees where employee_id=100



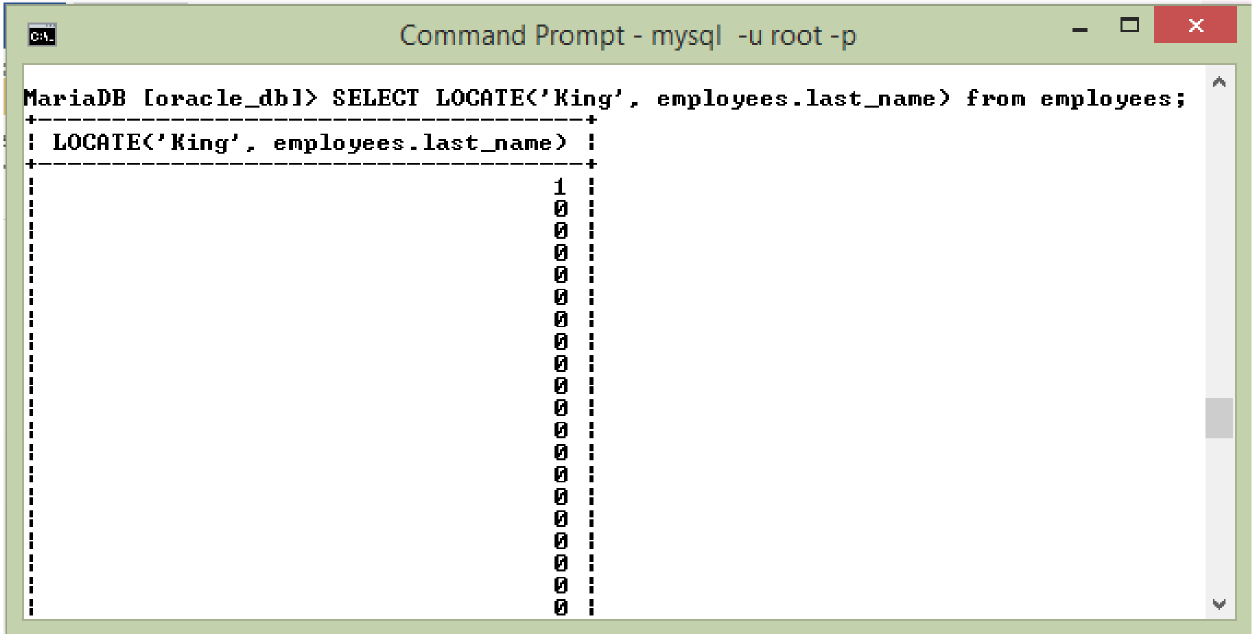
```
MariaDB [oracle_db]>
MariaDB [oracle_db]> SELECT REPLACE(employees.first_name, 'Steven', 'Steven hock
') from employees where employee_id =100;
+------------------------------------------------------+
| REPLACE(employees.first_name, 'Steven', 'Steven hock') |
+------------------------------------------------------+
| Steven hock                                          |
+------------------------------------------------------+
1 row in set (0.00 sec)

MariaDB [oracle_db]>
```

Figure 4: Replace String

6.3.4. Locate Function

For Example: locate the string king in employee last name attribute so

SELECT LOCATE ('King', employees.last_name) from employees;



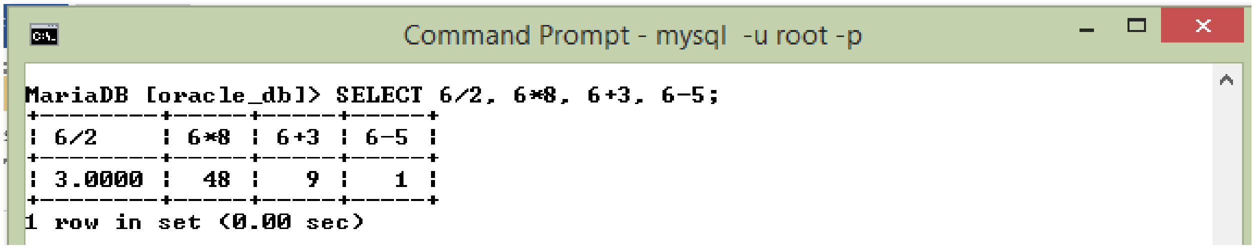
```
MariaDB [oracle_db]> SELECT LOCATE('King', employees.last_name) from employees;
+-------------------------------------+
| LOCATE('King', employees.last_name) |
+-------------------------------------+
|                                   1 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
|                                   0 |
```

Figure 5: Locate Function

6.3.5. Numeric Functions

Examples:

SELECT 6/2, 6*8, 6+3, 6-5;

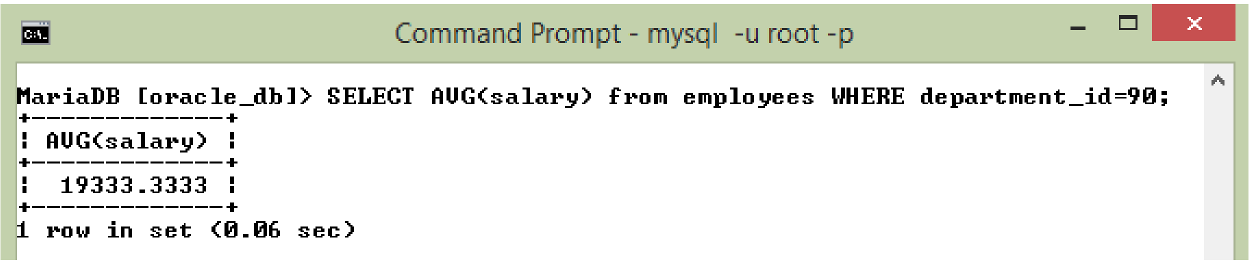


```
MariaDB [oracle_db]> SELECT 6/2, 6*8, 6+3, 6-5;
+--------+-----+-----+-----+
| 6/2    | 6*8 | 6+3 | 6-5 |
+--------+-----+-----+-----+
| 3.0000 |  48 |   9 |   1 |
+--------+-----+-----+-----+
1 row in set (0.00 sec)
```

Figure 6: Numeric Function

6.3.6. Aggregate/Average Functions

For Example: Show the average salary paid by department whose number is 90.
SELECT AVG(salary) from employees WHERE department_id=90;

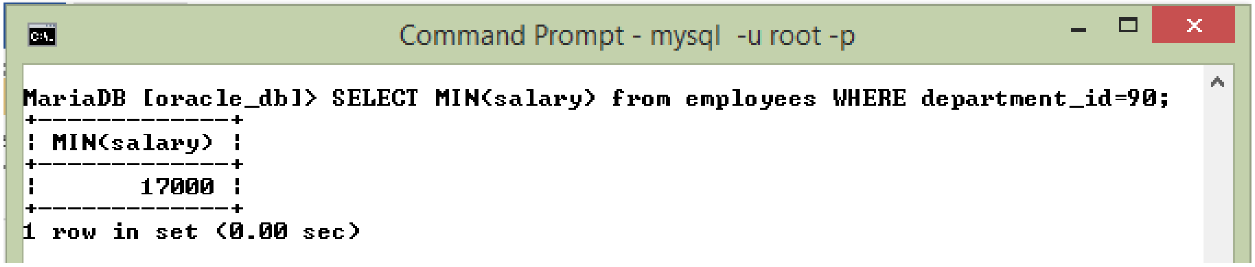


```
MariaDB [oracle_db]> SELECT AVG(salary) from employees WHERE department_id=90;
+-------------+
| AVG(salary) |
+-------------+
|  19333.3333 |
+-------------+
1 row in set (0.06 sec)
```

Figure 7: Average Function

6.3.7. Aggregate/Min, Max Functions

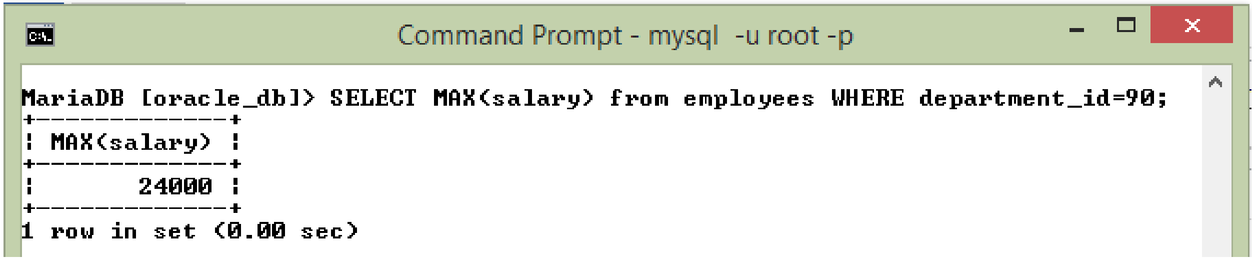
For Example: Show the Min salary paid by department whose number is 90.
SELECT MIN(salary) from employees WHERE department_id=90;



```
MariaDB [oracle_db]> SELECT MIN(salary) from employees WHERE department_id=90;
+-------------+
| MIN(salary) |
+-------------+
|       17000 |
+-------------+
1 row in set (0.00 sec)
```

Figure 8: Min Function

For Example: Show the MAX salary paid by department whose number is 90.
SELECT MAX(salary) from employees WHERE department_id=90;

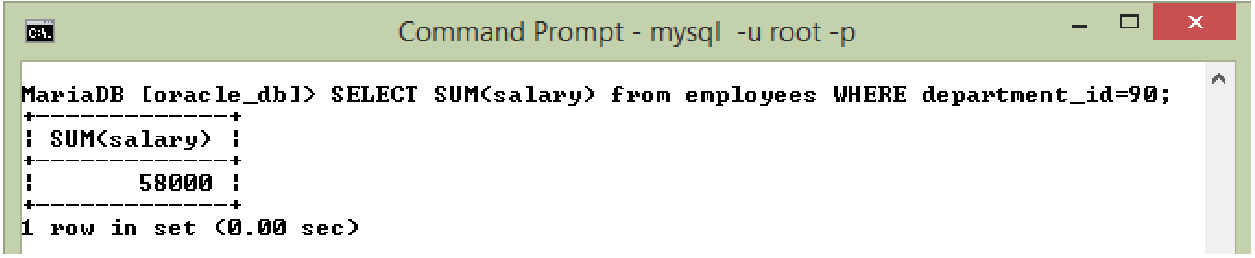


```
MariaDB [oracle_db]> SELECT MAX(salary) from employees WHERE department_id=90;
+-------------+
| MAX(salary) |
+-------------+
|       24000 |
+-------------+
1 row in set (0.00 sec)
```

Figure 9: MAX Function

6.3.8. Aggregate/SUM Functions

For Example: Show the Total salary paid by department whose number is 90.



```
MariaDB [oracle_db]> SELECT SUM(salary) from employees WHERE department_id=90;
+-------------+
| SUM(salary) |
+-------------+
|       58000 |
+-------------+
1 row in set (0.00 sec)
```

Figure 10: Sum Function

6. Practice Tasks

This section will provide more practice exercises which you need to finish during the lab. You need to finish the tasks in the required time. When you finish them, put these tasks in the following folder:
\\fs\assignments$

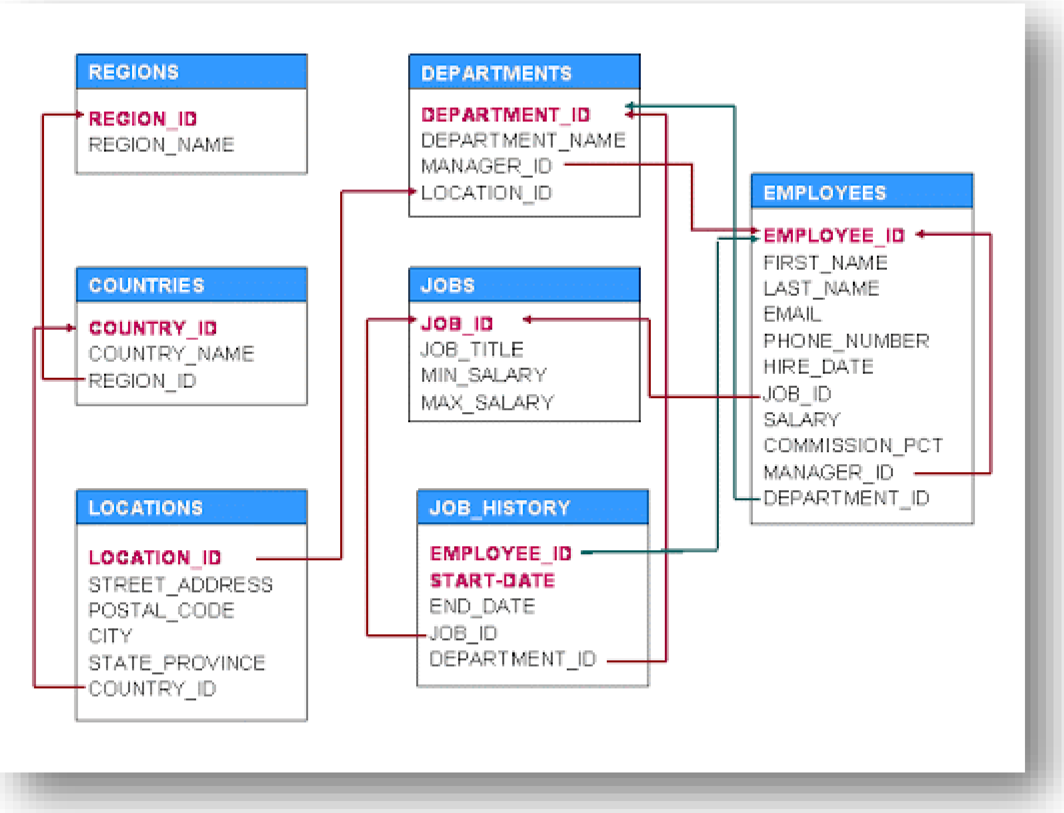


Figure 10: Database Design Diagram for "Oracle_db" Schema

7.1.Practice Task 1 [Expected time = 40mins]

Consider the above given schema, and write down the following SQL queries.

1. Show all the employees information ordered by first name (Ascending order).
2. Show all the employees information ordered by last name whose salary is in the range of 10,000 to 50,000.
3. What is the minimum salary given by department whose id is 90?
4. Which departments have location id greater than 1,000?
5. List the countries name located in Europe and concatenation with country id.
6. List all the street addresses of Italy.
7. Display the highest, lowest, sum and average salary of all employees. Label the columns Maximum, Minimum, Sum and Average respectively.
8. Write a query that displays the difference between the highest and lowest salaries. Label the column difference.
9. Show all jobs title whose difference between minimum and maximum salaries is greater than average salary of all jobs.
10. Show name, and job title of all employees who reports directly to Steven.

7. Evaluation Task (Unseen) [Expected time = 55mins for two tasks]

The lab instructor will give you unseen task depending upon the progress of the class.

8. Evaluation criteria

The evaluation criteria for this lab will be based on the completion of the following tasks. Each task is assigned the marks percentage which will be evaluated by the instructor in the lab whether the student has finished the complete/partial task(s).

Table 3: Evaluation of the Lab

Sr. No.	Task No	Description	Marks
1	6	Procedures and Tools	05
2	7	Practice tasks and Testing	15
3	8	Evaluation Tasks (Unseen)	80

9. Further Reading

This section provides the references to further polish your skills.

10. Books

a) Text Book:

- Database Systems, A practical approach to design, implementation and management by by Thomas Connolly, Carolyn Begg, Addison Wesley , Fifth Edition,

b) Slides

- The slides and reading material can be accessed from the folder of the class instructor available at \\fs\lectures$

11. REFERENCES:

SQL-99 Complete, Really, by Peter Gulutzan & Trudy Pelzer.

- More examples for the SELECT command:
  http://dev.mysql.com/doc/mysql/en/select.html
- MySQL operators:
  http://dev.mysql.com/doc/mysql/en/non-typed_operators.html
- Built-in functions:
  http://dev.mysql.com/doc/mysql/en/functions.html
- Joining tables:
  http://www.melonfire.com/community/columns/trog/article.php?id=148
- Using subqeries:
  http://www.melonfire.com/community/columns/trog/article.php?id=204