Lab 5
Summary Queries Using Aggregation Functions

1. Learning Objectives

After the successful completion of this lab, you will be able

- To explain what summary queries and aggregation functions are
- To determine the outputs of SQL statements that summarize data using aggregation functions
- To write SQL statements that summarize data using aggregation functions
- To write SQL statements that get information from one or more tables

2. Tasks to Complete

Complete the questions about data summaries on the **MGS Database** included in the later part of this document. These queries use the tables in **user mgs**.

Each query will use one or more aggregation functions.

Each query gets desired summary information from one or multiple tables using the concepts below:

- Summary query, Aggregation functions
- GROUP BY clause, HAVING clause

NOTE 1:

The goal of this lab is for you to practice summary queries and aggregation functions.

So if your answer to a question is not a summary query that uses aggregation functions, you will get a 0 for this question.

NOTE 2:

The links to online Oracle SQL Language references are available in the Canvas Page: Links to Oracle SQL Language References.

RELATED KNOWLEDGE POINTS

- **A summary query returns the summary information about data stored in the relational database.**
- **A summary query includes at least one aggregation function.**
- **An aggregation function operates on a group of values and returns one value.**
- **Aggregation functions covered in this course include AVG, SUM, MAX, MIN, COUNT.**
- **GROUP BY clause is used to generate a data summary for EACH GROUP of rows.**
- **HAVING clause is used to ELIMINATE data summaries of UNDESIRED GROUPs of rows.**
- **WHERE clause can NOT contain any aggregation function.**
- **When a query contains a GROUP BY clause, SELECT clause can ONLY contain the following**
  - **Aggregation functions**
  - **Expressions in GROUP BY clause or Expressions involving GROUP BY columns**
  - **Constant values or Expressions resulting in a constant value**

3. Submission Requirements

WHEN

- **Please see the canvas page for details**.

WHAT

- A text file with the extension **.sql, including all your SQL statements.**
- The file is in the following format**.**

Mark each query based on the question number. Write your FULL name on the first page.
Sample:

```
--Lab5
--Your full name

--Q1
Your solution.
--Q2
Your solution.
…
```

HOW

- Submit your SQL script file by attaching it to the link **Lab 5** in folder **Assignments\Labs** on Canvas.

4. Grading

This assignment is graded based on CORRECTNESS.

There are 8 questions. Each question counts 12.5 points.

If your answer to a question is NOT A SUMMARY QUERY, you will get 0 points for that question.

If your answer is not 100% correct, you will get partial credits based on the extent of the error.

NOTE:

- **IF YOU DON'T DO THE WORK IN THIS LAB, YOUR MIDTERM TEST GRADE WILL MOST LIKELY BE AFFECTED.**

Summary Queries on MGS Database

1. Print the total number of orders and their total tax amount.
   Use the headings ORDER_COUNT, TOTAL_TAX in the query result.

   HINT:
   - **You need to use two aggregation functions and select from one table.**
   - **You need to use column aliases.**

   OUTPUT

	ORDER_COUNT	TOTAL_TAX
1	9	122.24

2. How many orders are paid by Visa cards, and what is the total tax amount of these orders?
   Use a single query to get both answers.
   Use the headings VISA_ORDER_COUNT, VISA_TOTAL_TAX in the query result.

   HINT:
   - **You need to use two aggregation functions and select from one table.**
   - **You need to use column aliases.**
   - **You need a WHERE clause to eliminate undesired rows.**

OUTPUT

	VISA_ORDER_COUNT	VISA_TOTAL_TAX
1	6	122.24

3. For each type of credit cards, print the card type, the total number of orders paid by credit cards of this type, and the total tax amount of such orders for this card type.
   Use the headings CARD_TYPE, CARD_ORDER_COUNT, CARD_TOTAL_TAX in the query result.

   HINT:
   - **You need to use two aggregation functions and select from one table.**
   - **You need to use column aliases.**
   - **You need a GROUP BY clause to generate summaries for different groups of rows.**

   OUTPUT

	CARD_TYPE	CARD_ORDER_COUNT	CARD_TOTAL_TAX
1	MasterCard	1	0
2	Visa	6	122.24
3	Discover	1	0
4	American Express	1	0

4. For each product category, print the category name, the number of products in this category, the highest listing price of products in this category, and the lowest listing price of products in this category.
   Use the headings CATEGORY_NAME, PRODUCT_COUNT, HIGHEST_LISTING_PRICE, and LOWEST_LISTING_PRICE in the query result.

   HINT:
   - **You need to use three aggregation functions and join two tables.**
   - **You need to use column aliases.**
   - **You need a GROUP BY clause to generate summaries for different groups of rows.**

   OUTPUT

	CATEGORY_NAME	PRODUCT_COUNT	HIGHEST_LISTING_PRICE	LOWEST_LISTING_PRICE
1	Drums	2	799.99	699.99
2	Basses	2	799.99	499.99
3	Guitars	6	2517	299

5. Rewrite the query in Question 4 such that only products with listing prices more than $400 are included in the query result.
   In another word, for each product category, print the category name, the number of products in this category with listing prices more than $400, the highest listing price of such products in this category, and the lowest listing price of such products in this category.
   Use the headings CATEGORY_NAME, PRODUCT_COUNT_OVER$400, HIGHEST_LISTING_PRICE_OVER$400, and LOWEST_LISTING_PRICE_OVER$400 in the query result.

   HINT:

- **You need to use three aggregation functions and join two tables.**
- **You need to use column aliases.**
- **You need a WHERE clause to eliminate undesired rows before the grouping.**
- **You need a GROUP BY clause to generate summaries for different groups of rows.**

OUTPUT

	CATEGORY_NAME	PRODUCT_COUNT_OVER$400	HIGHEST_LISTING_PRICE_OVER$400	LOWEST_LISTING_PRICE_OVER$400
1	Drums	2	799.99	699.99
2	Basses	2	799.99	499.99
3	Guitars	5	2517	415

6. Rewrite the query in Question 3 such that only credit card types with at least 2 orders are printed.
In another word, for each type of credit cards with at least 2 orders, print the card type, the total number of orders paid by credit cards of this type, and the total tax amount of such orders for this card type.
Use the headings CARD_TYPE, HOT_CARD_ORDER_COUNT, and HOT_CARD_TOTAL_TAX in the query result.

HINT:

- **You need to use two aggregation functions and select from one table.**
- **You need to use column aliases.**
- **You need a GROUP BY clause to generate summaries for different groups of rows.**
- **You need a HAVING clause to eliminate undesired groups of rows.**

OUTPUT

	CARD_TYPE	HOT_CARD_ORDER_COUNT	HOT_CARD_TOTAL_TAX
1	Visa	6	122.24

7. Rewrite the query in Question 4 such that only categories with at least 3 products are included in the query result.
In another word, for each category with at least 3 products, print the category name, the number of products in this category, the highest listing price of products in this category, and the lowest listing price of products in this category.
Use the headings CATEGORY_NAME, HOT_CAT_PRODUCT_COUNT, HOT_CAT_HIGHEST_LISTING_PRICE, and HOT_CAT_LOWEST_LISTING_PRICE in the query result.

HINT:

- **You need to use three aggregation functions and join two tables.**
- **You need to use column aliases.**
- **You need a GROUP BY clause to generate summaries for different groups of rows.**
- **You need to eliminate undesired groups of rows.**
    - **Think about which clause to add: a WHERE clause or HAVING clause?**

OUTPUT

	CATEGORY_NAME	HOT_CAT_PRODUCT_COUNT	HOT_CAT_HIGHEST_LISTING	HOT_CAT_LOWEST_LISTING
1	Guitars	6	2517	299

8. Rewrite the query in Question 3 such that only orders dated after March 30 2012 are included in the query result.
   In another word, for each type of credit cards with at least 2 orders dated after March 30 2012 and paid by credit cards of this card type, print the card type, the count of such orders, and the total tax amount of such orders for this credit card type.
   Use headings CARD_TYPE, HOT_CARD_ORDER_COUNT, HOT_CARD_ TOTAL_TAX in the query result.

   HINT:
   - **You need to use two aggregation functions and select from one table.**
   - **You need to use column aliases.**
   - **You need a GROUP BY clause to generate summaries for different groups of rows.**
   - **You need both WHERE clause and HAVING clause in the query.**

   OUTPUT

	CARD_TYPE	HOT_CARD_ORDER_COUNT	HOT_CARD_TOTAL_TAX
1	Visa	3	0