

Course	: BIC 21404 Database
Session	: II 2024/2025
Lab task	: 6
Lab Topic	: Reporting Aggregated Data Using the Group Functions
Name	:
Matric No	:

Instructions: answer all questions

Write a summary (300 words) on the lesson learned, difficulties arise or any new knowledge obtained throughout the Lab 6 Exercise.

Lesson learned in Lab 6:

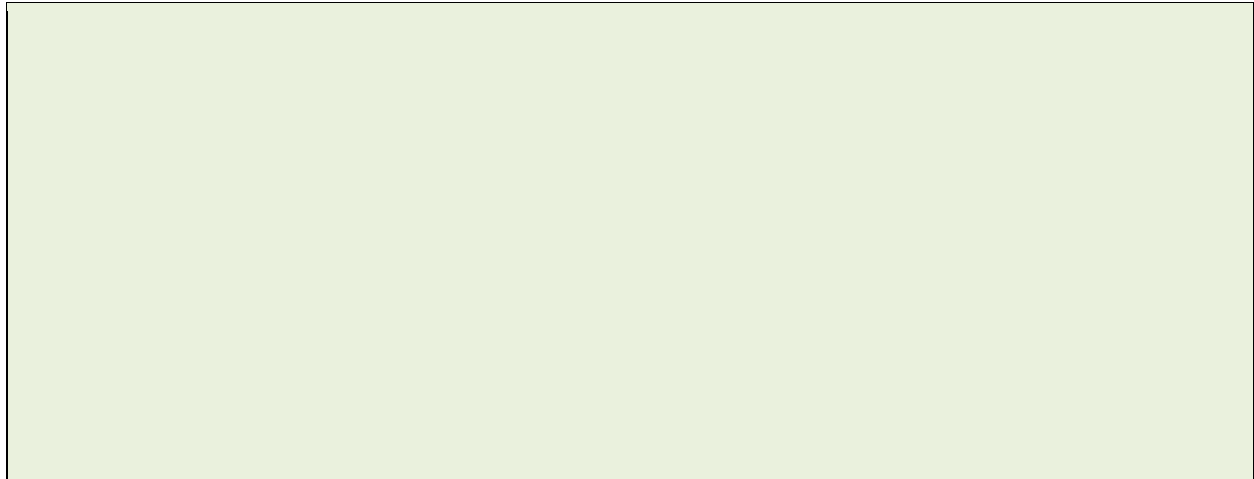
In Lab 6 I really felt what it's like to turn a pile of rows into clear, high-level insights with just a few lines of SQL. Playing with the classic group functions like SUM, AVG, MIN and MAX showed me I don't need to drag data into Excel to see totals or averages. I could label each result right inside the query, even round those numbers so they looked neat for anyone reading the output.

Counting turned out to have its own little quirks. Using COUNT(*) to tally every row was straightforward, but I also learned how COUNT(column) skips nulls and how COUNT(DISTINCT column) finds unique values. It was a good reminder that nulls don't always behave the way you expect unless you handle them—so using IFNULL (or NVL in Oracle) became second nature for me.

Next came GROUP BY, which at first felt like a rule to simply avoid errors. But once I saw how grouping by job type or manager lets me roll up many rows into one summary per group, it clicked. And adding a HAVING clause to filter those groups such as only showing departments whose total salaries exceed a certain amount felt like getting a post-group quality check. It's like telling the database, "Okay, now that you've summarized everything, only show me the summaries that really matter."

The real "aha" moment arrived when I nested queries: calculating an average per department, then feeding those averages into another query to find the maximum. It was a neat demonstration of how you can chain summaries together without ever leaving SQL.

Sure, I glanced at the docs a few times to remember exact syntax should it be ROUND(AVG(salary), 2) or AVG(ROUND(salary, 2)) but each lookup made the commands more familiar. By the end of Lab 6 I no longer saw SQL as just row-fetching; I saw it as a full-blown reporting engine capable of delivering polished, professional results straight from the database.

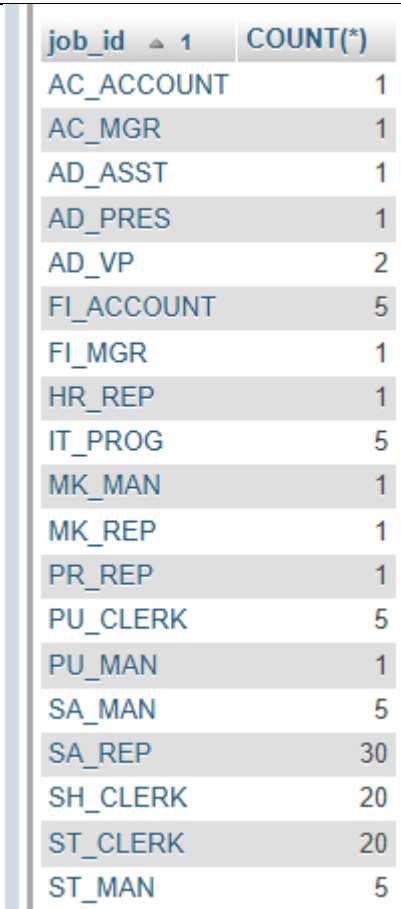


- Write a query to find the highest, lowest, sum, and average salary of all employees. Label the columns *Maximum*, *Minimum*, *Sum* and *Average*, respectively. Round your results to the nearest possible whole number.

Solution:									
SQL statement	<pre>SELECT ROUND(MAX(salary)) AS Maximum, ROUND(MIN(salary)) AS Minimum, ROUND(SUM(salary)) AS Sum, ROUND(AVG(salary)) AS Average FROM employees;</pre>								
Output display	<table><tr><th>Maximum</th><th>Minimum</th><th>Sum</th><th>Average</th></tr><tr><td>24000</td><td>2100</td><td>691400</td><td>6462</td></tr></table>	Maximum	Minimum	Sum	Average	24000	2100	691400	6462
Maximum	Minimum	Sum	Average						
24000	2100	691400	6462						

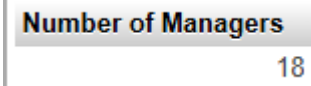
- Modify the query in Q1 to display the job id, minimum, maximum, sum and average salary for each job type.

Solution:	
SQL statement	<pre>SELECT job_id, ROUND(MIN(salary)) AS Minimum,</pre>

Output display	 <pre> job_id 1 COUNT(*) AC_ACCOUNT 1 AC_MGR 1 AD_ASST 1 AD_PRES 1 AD_VP 2 FI_ACCOUNT 5 FI_MGR 1 HR_REP 1 IT_PROG 5 MK_MAN 1 MK_REP 1 PR_REP 1 PU_CLERK 5 PU_MAN 1 SA_MAN 5 SA_REP 30 SH_CLERK 20 ST_CLERK 20 ST_MAN 5 </pre>
----------------	---

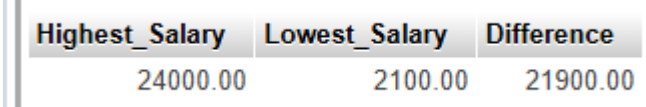
4. Determine the number of managers in the company without listing them. Label the column as Number of Managers.

Hint: Use the MANAGER_ID column to determine the number of managers.

Solution:	
SQL statement	<pre> SELECT COUNT(DISTINCT manager_id) AS `Number of Managers` FROM employees WHERE manager_id IS NOT NULL; </pre>
Output display	 <pre> Number of Managers 18 </pre>

--	--





















5. Find the difference between the highest and lowest salaries. Show the highest, lowest and the difference. Label the column appropriately

Solution:	
SQL statement	<pre>SELECT MAX(salary) AS Highest_Salary, MIN(salary) AS Lowest_Salary, (MAX(salary) - MIN(salary)) AS Difference FROM employees;</pre>
Output display	

6. Produce a report to display the manager number and the salary of the lowest-paid employee for that manager. Exclude anyone whose manager is not known. Exclude any groups where the minimum salary is \$6,000 or less. Sort the output in descending order of salary.

Solution:	
SQL statement	<pre>SELECT manager_id AS Manager_Number, MIN(salary) AS Lowest_Salary FROM employees WHERE manager_id IS NOT NULL GROUP BY manager_id HAVING MIN(salary) > 6000 ORDER BY Lowest_Salary DESC;</pre>

Output display

		Manager_Number	Lowest_Salary	1	
<input type="checkbox"/>	Click the drop-down arrow to toggle column's visibility.		102	9000.00	
<input type="checkbox"/>	 Edit	 Copy	 Delete	205	8300.00
<input type="checkbox"/>	 Edit	 Copy	 Delete	145	7000.00
<input type="checkbox"/>	 Edit	 Copy	 Delete	146	7000.00
<input type="checkbox"/>	 Edit	 Copy	 Delete	108	6900.00
<input type="checkbox"/>	 Edit	 Copy	 Delete	147	6200.00
<input type="checkbox"/>	 Edit	 Copy	 Delete	149	6200.00
<input type="checkbox"/>	 Edit	 Copy	 Delete	148	6100.00