Kabeer Ahmed
SE-19028

Database Management System
(DBMS – 204)

Experiment # 05

Aggregating Data Using Group Functions

Student Name: Kabeer Ahmed

Roll Number: SE-19028

Maximum Marks	Performance = 05	Viva = 05	Total = 10
Marks Obtained			
Remarks (if any)			

Experiment evaluated by

Instructor Name: Engr. Adiba Jafar

Signature and Date:

Kabeer Ahmed
SE-19028

Outcome

After completing this lesson, you should be able to do the following:

1. Identify the available group functions
2. Describe the use of group functions
3. Group data using the GROUP BY clause
4. Include or exclude grouped rows by using the HAVING clause

What Are Group Functions?

Group functions operate on sets of rows to give one result per group.

Types of Group Functions

1. AVG
2. COUNT
3. MAX
4. MIN
5. SUM

Group Functions Syntax

```
SELECT [ column ,] group_function(column), ...
FROM table
[WHERE condition ]
[GROUP BY column ]
[ORDER BY column ];
```

Using the AVG and SUM Functions

You can use AVG and SUM for numeric data.

```
SELECT AVG(sal), MAX(sal),MIN(sal), SUM(sal) FROM   emp
WHERE  job LIKE '%MAN%';
```

Using the MIN and MAX Functions

You can use MIN and MAX for any data type.

1. SELECT MIN(hiredate), MAX(hiredate)
   FROM emp;

2. SELECT MIN(ename), MAX(ename)
   FROM   emp;

Note: AVG , SUM , VARIANCE , and STDDEV functions can be used only with numeric data types.

Using the COUNT Function COUNT(*)

1. COUNT(expr) returns the number of rows with non-null values for the expr.
2. Display the number of department values in the EMP table, excluding the null values.

```
SELECT COUNT(*) FROM emp WHERE  deptno = 30;

SELECT COUNT(comm)  FROM emp WHERE deptno = 30;

SELECT COUNT(deptno)  FROM   emp;
```

Using the DISTINCT Keyword

1. COUNT(DISTINCT expr) returns the number of distinct nonnull values of the expr .
2. Display the number of distinct department values in the EMPLOYEES table.

```
SELECT COUNT(DISTINCT deptno) FROM   emp;
```

Kabeer Ahmed
SE-19028

Group Functions and Null Values

Group functions ignore null values in the column.
SELECT AVG(comm) FROM emp;

Using the NVL Function with Group Functions

The NVL function forces group functions to include null values.
SELECT AVG(NVL(comm, 0)) FROM emp;

Creating Groups of Data

GROUP BY Clause Syntax

```
SELECT column , group_function(column)
FROM table
[WHERE condition ]
[GROUP BY group_by_expression ]
[ORDER BY column ];
```

Divide rows in a table into smaller groups by using the

Using the GROUP BY Clause

All columns in the SELECT list that are not in group functions must be in the GROUP BY clause.
Group By X means put all those with the same value for X in the one group**.**

```
SELECT deptno, AVG(sal) FROM emp
GROUP BY deptno;

SELECT deptno, AVG(sal) FROM emp GROUP BY deptno ORDER BY AVG(sal);
```

Using the GROUP BY Clause on Multiple Columns

Group By X, Y means put all those with the same values for both X and Y in the one group**.**

```
SELECT deptno, job,count(job),SUM(sal) FROM emp GROUP BY deptno,job;
```

Illegal Queries Using Group Functions

Any column or expression in the SELECT list that is not an aggregate function must be in the GROUP BY clause.

```
SELECT deptno, COUNT(ename) FROM emp;
SELECT deptno, COUNT(ename)
*
ERROR at line 1:
ORA-00937: not a single-group group function

SELECT deptno, count(ename) FROM emp GROUP BY deptno;
```

Any column or expression in the SELECT list that is not an aggregate function must be in the GROUP BY clause.

1. You cannot use the WHERE clause to restrict groups.
2. You use the HAVING clause to restrict groups.
3. You cannot use group functions in the WHERE clause.

```
SELECT deptno, AVG(sal) FROM emp WHERE AVG(sal) > 8000 GROUP BY deptno;

WHERE AVG(sal) > 8000
*
```

Kabeer Ahmed
SE-19028

ERROR at line 3:
ORA-00934: group function is not allowed here

SELECT deptno, AVG(sal) FROM emp HAVING AVG(sal)>2000 GROUP BY deptno;

Excluding Group Results: The HAVING Clause

Use the HAVING clause to restrict groups:

1. Rows are grouped.
2. The group function is applied.
3. Groups matching the

SELECT column , group_function FROM table
[WHERE condition]
[GROUP BY group_by_expression]
[HAVING group_condition]
[ORDER BY column];

Using the HAVING Clause

SELECT   deptno, MAX(sal) FROM   emp
GROUP BY deptno HAVING   MAX(sal)>3000;

SELECT   deptno, AVG(sal) FROM   emp GROUP BY deptno
HAVING   max(sal)>3000;

SELECT   job, SUM(sal) PAYROLL FROM   emp
WHERE   job NOT LIKE '%MAN%' GROUP BY job HAVING   SUM(sal) > 3000
ORDER BY SUM(sal);

Nesting Group Functions

Display the maximum average salary.
SELECT MAX(AVG(sal)) FROM   emp GROUP BY deptno;

Kabeer Ahmed
SE-19028

LAB # 05

Aggregation Data Using Group Functions

1. Group functions work across many rows to produce one result.

True

2. Group functions include nulls in calculations.

False.

3. The WHERE clause restricts rows prior to inclusion in a group calculation.

True

4. Display 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 whole number. Place your SQL statement in a text file named lab5_6.sql.

```
1  SELECT ROUND(MAX(SAL),0) "Maximum",
2  ROUND(MIN(SAL),0) "Minimum",
3  ROUND(SUM(SAL),0) "Sum",
4  ROUND(AVG(SAL),0) "Average"
5  FROM EMP;
6
```

Maximum	Minimum	Sum	Average
5000	800	29025	2073

Kabeer Ahmed
SE-19028

5. Modify the query in lab5_4.sql to display the minimum, maximum, sum, and average salary for each job type. Resave lab5_4.sql To lab5_5.sql. Run the statement in lab5_5.sql.

```
1  SELECT JOB,
2  ROUND(MAX(SAL),0)  "Maximum",
3  ROUND(MIN(SAL),0)  "Minimum",
4  ROUND(SUM(SAL),0)  "Sum",
5  ROUND(AVG(SAL),0)  "Average"
6  FROM EMP
7  GROUP BY JOB;
8
```

JOB	Maximum	Minimum	Sum	Average
ANALYST	3000	3000	6000	3000
CLERK	1300	800	4150	1038
SALESMAN	1600	1250	5600	1400
MANAGER	2975	2450	8275	2758
PRESIDENT	5000	5000	5000	5000

6. Write a query to display the number of people with the same job.

```
1  SELECT JOB, COUNT(*)
2  FROM EMP
3  GROUP BY JOB;
4
```

JOB	COUNT(*)
ANALYST	2
CLERK	4
SALESMAN	4
MANAGER	3
PRESIDENT	1

Kabeer Ahmed
SE-19028

7. Determine the number of managers without listing them. Label the column Number of Managers. Hint: Use the MANAGER_ID column to determine the number of managers.

```
1  SELECT COUNT(DISTINCT MGR) "Number of Managers"
2  FROM EMP;
3
```

Number of Managers
6

8. Write a query that displays the difference between the highest and lowest salaries. Label the column DIFFERENCE.

```
1  SELECT MAX(sal) - MIN(sal) DIFFERENCE
2  FROM EMP;
3
```

DIFFERENCE
4200

Kabeer Ahmed
SE-19028

9. 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 less than $6,000. Sort the output in descending order of salary.

```
select empno as "Manager_id", MIN (sal)
from emp
where job = 'MANAGER' and empno IS NOT NULL
group by empno
having MIN(sal) > 6000
order by MIN (sal) DESC;
```

Script Output | Query Result

Task completed in 0.016 seconds

Manager_id	MIN(SAL)
7566	6500
7994	6500
7782	6500

10. Write a query to display each department's name, location, number of employees, and the average salary for all employees in that department. Label the columns Name, Location, Number of people, and Salary, respectively. Round the average salary to two decimal places.

```
SELECT d.DNAME "Name", d.LOC "Location",COUNT(*) "Number of People",ROUND(AVG(sal),2) "Salary"
FROM emp e, dept d
WHERE e.deptno = d.deptno
GROUP BY d.DNAME, d.LOC;
```

Script Output | Query Result

Task completed in 0.016 seconds

Name	Location	Number of People	Salary
RESEARCH	DALLAS	4	1968.75
SALES	CHICAGO	5	5620
ACCOUNTING	NEWYORK	4	2375

Kabeer Ahmed
SE-19028

11. Create a query that will display the total number of employees and, of that total, the number of employees hired in 1995, 1996, 1997, and 1998. Create appropriate column headings.

```
SELECT COUNT(*) total,
SUM(DECODE(TO_CHAR(hiredate, 'YYYY'),1995,1,0))"1995",
SUM(DECODE(TO_CHAR(hiredate, 'YYYY'),1996,1,0))"1996",
SUM(DECODE(TO_CHAR(hiredate, 'YYYY'),1997,1,0))"1997",
SUM(DECODE(TO_CHAR(hiredate, 'YYYY'),1998,1,0))"1998"
FROM emp;
```

Script Output | Query Result

Task completed in 0.016 seconds

TOTAL	1995	1996	1997	1998
13	0	0	0	0

12. Create a matrix query to display the job, the salary for that job based on department number, and the total salary for that job, for departments 20, 50, 80, and 90, giving each column an appropriate heading.

```
SELECT job "Job",
SUM(DECODE(deptno , 20, sal)) "Deptno 20",
SUM(DECODE(deptno , 50, sal)) "Deptno 50",
SUM(DECODE(deptno , 80, sal)) "Deptno 80",
SUM(DECODE(deptno , 90, sal)) "Deptno 90",
SUM(sal) "Total"
FROM emp GROUP BY job;
```

Script Output | Query Result

Task completed in 0.016 seconds

Job	Deptno 20	Deptno 50	Deptno 80	Deptno 90	Total
SALESMAN	3775				10375
CLERK	1100				3150
PRESIDENT					7000
MANAGER					19500
ANALYST	3000				5450