Institute of Computer Technology

B. Tech Computer Science and Engineering

Sub: Database Management System (2CSE301)

Practical: 5 Perform Queries using Group by and Having clause.

The SQL **GROUP BY** clause is used in collaboration with the SELECT statement to arrange identical data into groups. This GROUP BY clause follows the WHERE clause in a SELECT statement and precedes the ORDER BY clause.

Syntax

The basic syntax of a GROUP BY clause is shown in the following code block. The GROUP BY clause must follow the conditions in the WHERE clause and must precede the ORDER BY clause if one is used.

```
SELECT column1, column2
FROM table_name
WHERE [ conditions ]
GROUP BY column1, column2
ORDER BY column1, column2
```

```
SELECT column1, column2
FROM table_name
GROUP BY column1, column2
Having [CONDITION]
```

Example

Consider the CUSTOMERS table is having the following records –

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```

If you want to know the total amount of the salary on each customer, then the GROUP BY query would be as follows.

```
SQL> SELECT NAME, SUM(SALARY) FROM CUSTOMERS
  GROUP BY NAME;
```

This would produce the following result –

```
+----------+-------------+
| NAME     | SUM(SALARY) |
+----------+-------------+
| Chaitali |     6500.00 |
| Hardik   |     8500.00 |
| kaushik  |     2000.00 |
| Khilan   |     1500.00 |
| Komal    |     4500.00 |
| Muffy    |    10000.00 |
| Ramesh   |     2000.00 |
+----------+-------------+
```

Now, let us look at a table where the CUSTOMERS table has the following records with duplicate names –

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Ramesh   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | kaushik  |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```

Now again, if you want to know the total amount of salary on each customer, then the GROUP BY query would be as follows –

```
SQL> SELECT NAME, SUM(SALARY) FROM CUSTOMERS
  GROUP BY NAME;
```

This would produce the following result –

```
+---------+-------------+
| NAME    | SUM(SALARY) |
+---------+-------------+
| Hardik  |     8500.00 |
| kaushik |     8500.00 |
| Komal   |     4500.00 |
| Muffy   |    10000.00 |
| Ramesh  |     3500.00 |
+---------+-------------+
```

The SQL HAVING Clause

The HAVING clause was added to SQL because the WHERE keyword cannot be used with aggregate functions.

Queries:

1) How many employees are there in each department?
2) Find out total number of job role assigned in each department.
3) Find out employee's names and salary whose having salary more than 2000. (Duplication in employee name should be removed)
4) Find out number of employees hired after 03rd April 1991.
5) lists the number of employees in each job role, sorted high to low.
6) lists the number of employees in each department. Only include department with more than 3 employees in each.
7) Display the total amount of the salary on each department.
8) Count total number of employees assigned in each department whose name end with "n".
9) Find out total number of employees having "a" as a character in their name in each department.
10) Find out total number of employees having salary more than average salary of all the employee in each department.
11) Display total number of employees in each department whose department having more than 2 employees also display department id in descending order.
12) Display department wise average salary of employee.
13) Display department id of the employee along with salary whose salary is maximum in respective department.
14) Display department id of the employee along with salary whose salary is minimum in respective department.