POSTGRESQL - GROUP BY

The PostgreSQL **GROUP BY** clause is used in collaboration with the SELECT statement to group together those rows in a table that have identical data. This is done to eliminate redundancy in the output and/or compute aggregates that apply to these groups.

The GROUP BY clause follows the WHERE clause in a SELECT statement and precedes the ORDER BY clause.

Syntax:

The basic syntax of GROUP BY clause is given below. The GROUP BY clause must follow the conditions in the WHERE clause and must precede the ORDER BY clause if one is used.

```
SELECT column-list
FROM table_name
WHERE [ conditions ]
GROUP BY column1, column2....columnN
ORDER BY column1, column2....columnN
```

You can use more than one column in the GROUP BY clause. Make sure whatever column you are using to group, that column should be available in column-list.

Example:

Consider the table COMPANY having records as follows:

```
# select * from COMPANY;
 id | name  | age | address   | salary
----+-------+-----+-----------+--------
  1 | Paul  |  32 | California|  20000
  2 | Allen |  25 | Texas     |  15000
  3 | Teddy |  23 | Norway    |  20000
  4 | Mark  |  25 | Rich-Mond |  65000
  5 | David |  27 | Texas     |  85000
  6 | Kim   |  22 | South-Hall|  45000
  7 | James |  24 | Houston   |  10000
(7 rows)
```

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

```
testdb=# SELECT NAME, SUM(SALARY) FROM COMPANY GROUP BY NAME;
```

This would produce the following result:

```
  name  |  sum
 -------+-------
  Teddy | 20000
  Paul  | 20000
  Mark  | 65000
  David | 85000
  Allen | 15000
  Kim   | 45000
  James | 10000
(7 rows)
```

Now, let us create three more records in COMPANY table using the following INSERT statements:

```
INSERT INTO COMPANY VALUES (8, 'Paul', 24, 'Houston', 20000.00);
INSERT INTO COMPANY VALUES (9, 'James', 44, 'Norway', 5000.00);
INSERT INTO COMPANY VALUES (10, 'James', 45, 'Texas', 5000.00);
```

Now, our table has the following records with duplicate names:

```
 id | name  | age | address      | salary
----+-------+-----+--------------+--------
  1 | Paul  |  32 | California   |  20000
  2 | Allen |  25 | Texas        |  15000
  3 | Teddy |  23 | Norway       |  20000
  4 | Mark  |  25 | Rich-Mond    |  65000
  5 | David |  27 | Texas        |  85000
  6 | Kim   |  22 | South-Hall   |  45000
  7 | James |  24 | Houston      |  10000
  8 | Paul  |  24 | Houston      |  20000
  9 | James |  44 | Norway       |   5000
 10 | James |  45 | Texas        |   5000
(10 rows)
```

Again, let us use the same statement to group-by all the records using NAME column as follows:

```
testdb=# SELECT NAME, SUM(SALARY) FROM COMPANY GROUP BY NAME ORDER BY NAME;
```

This would produce the following result:

```
 name  |  sum
-------+-------
 Allen | 15000
 David | 85000
 James | 20000
 Kim   | 45000
 Mark  | 65000
 Paul  | 40000
 Teddy | 20000
(7 rows)
```

Let us use ORDER BY clause along with GROUP BY clause as follows:

```
testdb=#  SELECT NAME, SUM(SALARY)
          FROM COMPANY GROUP BY NAME ORDER BY NAME DESC;
```

This would produce the following result:

```
 name  |  sum
-------+-------
 Teddy | 20000
 Paul  | 40000
 Mark  | 65000
 Kim   | 45000
 James | 20000
 David | 85000
 Allen | 15000
(7 rows)
```