POSTGRESQL - HAVING CLAUSE

The HAVING clause allows us to pick out particular rows where the function's result meets some condition.

The WHERE clause places conditions on the selected columns, whereas the HAVING clause places conditions on groups created by the GROUP BY clause.

Syntax:

The following is the position of the HAVING clause in a SELECT query:

```
SELECT
FROM
WHERE
GROUP BY
HAVING
ORDER BY
```

The HAVING clause must follow the GROUP BY clause in a query and must also precede the ORDER BY clause if used. The following is the syntax of the SELECT statement, including the HAVING clause:

```
SELECT column1, column2
FROM table1, table2
WHERE [ conditions ]
GROUP BY column1, column2
HAVING [ conditions ]
ORDER BY column1, column2
```

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)
```

Following is the example, which would display record for which name count is less than 2:

```
testdb-# SELECT NAME FROM COMPANY GROUP BY name HAVING count(name) < 2;
```

This would produce the following result:

```
  name
 -------
  Teddy
  Paul
  Mark
  David
  Allen
  Kim
  James
(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)
```

Following is the example, which would display record for which name count is greater than 1:

```
testdb-# SELECT NAME FROM COMPANY GROUP BY name HAVING count(name) > 1;
```

This would produce the following result:

```
 name
-------
 Paul
 James
(2 rows)
```