Select Distinct – provides you a clause that removes duplicate rows in the result set.

PostgreSQL `SELECT DISTINCT` statement syntax

The `DISTINCT` clause is used in the `SELECT` statement to remove duplicate rows from the result set. The `DISTINCT` clause keeps one row for each group of duplicates. The `DISTINCT`clause can be used on one or more columns of a table. The following illustrates the syntax of the `DISTINCT` clause:

```
SELECT  DISTINCT column_1  FROM table_name;
```

In this statement, the values in the `column_1` column are used to evaluate the duplicate.

If you specify multiple columns, the `DISTINCT` clause will evaluate the duplicate based on   the combination of values of these columns.

```
SELECT  DISTINCT column_1,column_2  FROM tbl_name;
```

In this case, the combination of both `column_1` and `column_2` will be used for evaluating duplicate.

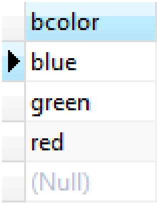
PostgreSQL also provides the `DISTINCT ON (expression)` to keep the “first” row of each group of duplicates using the following syntax:

```
SELECT  DISTINCT ON (column_1) column_1_alias,  column_2  FROM tbl_name  ORDER BY
column_1,column_2;
```

Select Distinct – provides you a clause that removes duplicate rows in the result set.

PostgreSQL DISTINCT on one column example

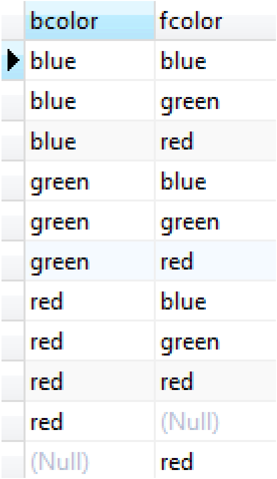
```
SELECT DISTINCT bcolor FROM t1 ORDER BY bcolor;
```



bcolor
blue
green
red
(Null)

PostgreSQL DISTINCT on multiple columns

```
SELECT DISTINCT bcolor, fcolor FROM t1 ORDER BY bcolor, fcolor;
```

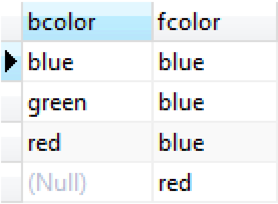


bcolor	fcolor
blue	blue
blue	green
blue	red
green	blue
green	green
green	red
red	blue
red	green
red	red
red	(Null)
(Null)	red

PostgreSQL DISTINCT ON ORDER BY example

The following statement sorts the result set by the `bcolor` and `fcolor`, and then for each group of duplicates, it keeps the first row in the returned result set.

```
SELECT DISTINCT ON (bcolor) bcolor, fcolor FROM t1 ORDER BY bcolor, fcolor;
```



bcolor	fcolor
blue	blue
green	blue
red	blue
(Null)	red

Select Distinct – provides you a clause that removes duplicate rows in the result set.

Try Below Query For Testing:

Q-1]

```
CREATE TABLE t1 (id serial NOT NULL PRIMARY KEY, bcolor VARCHAR, fcolor VARCHAR);
```

Q-2]

```
INSERT INTO t1 (bcolor, fcolor) VALUES
('red', 'red'),
('red', 'red'),
('red', NULL),
(NULL, 'red'),
('red', 'green'),
('red', 'blue'),
('green', 'red'),
('green', 'blue'),
('green', 'green'),
('blue', 'red'),
('blue', 'green'),
('blue', 'blue');
```