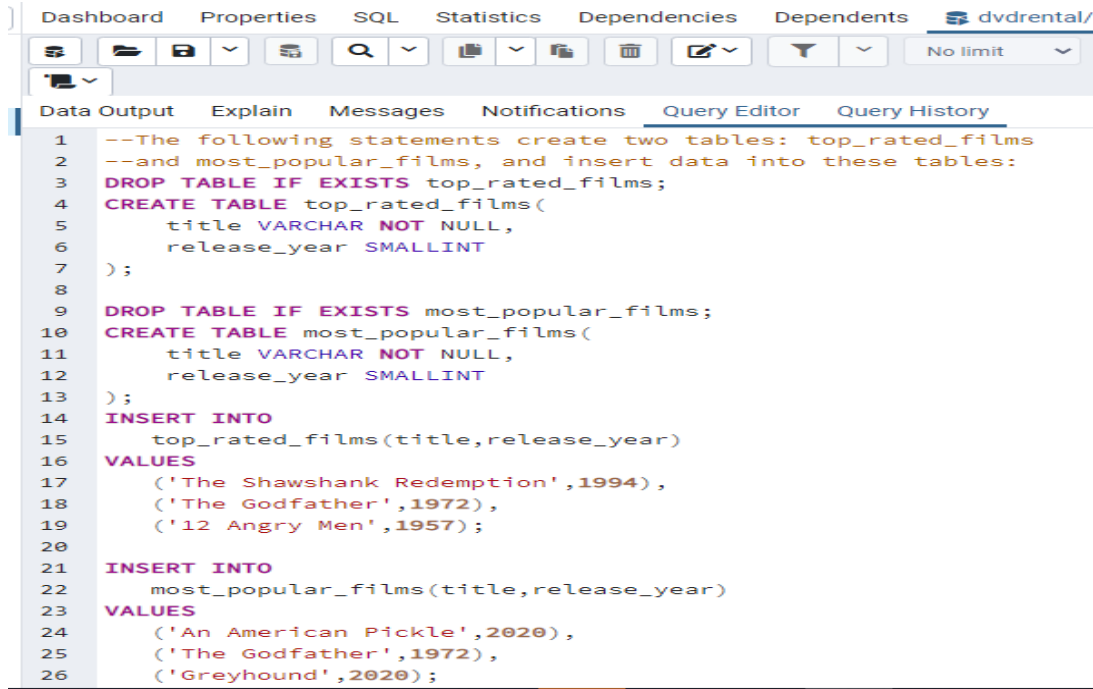


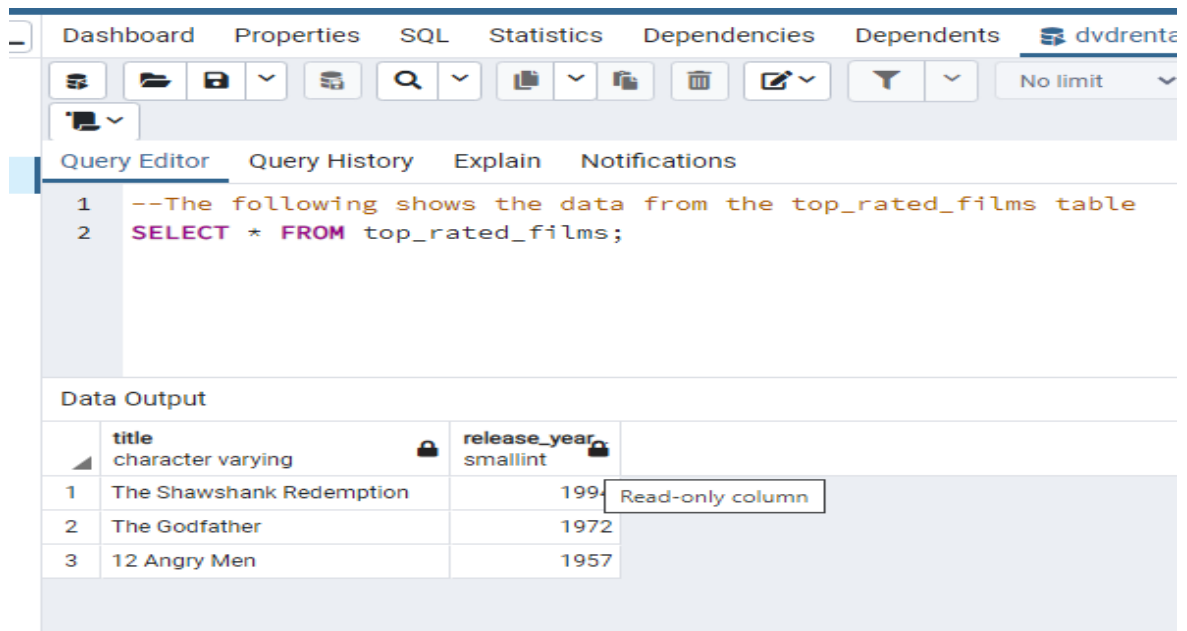
PostgreSQL Union

Setting up sample tables



The screenshot shows the PostgreSQL Query Editor interface. The top navigation bar includes 'Dashboard', 'Properties', 'SQL', 'Statistics', 'Dependencies', 'Dependents', and 'dvdrental/'. Below this is a toolbar with icons for various database actions. The 'Query Editor' tab is active, displaying the following SQL script:

```
1  --The following statements create two tables: topRatedFilms
2  --and mostPopularFilms, and insert data into these tables:
3  DROP TABLE IF EXISTS topRatedFilms;
4  CREATE TABLE topRatedFilms(
5      title VARCHAR NOT NULL,
6      release_year SMALLINT
7  );
8
9  DROP TABLE IF EXISTS mostPopularFilms;
10 CREATE TABLE mostPopularFilms(
11     title VARCHAR NOT NULL,
12     release_year SMALLINT
13 );
14 INSERT INTO
15     topRatedFilms(title,release_year)
16 VALUES
17     ('The Shawshank Redemption',1994),
18     ('The Godfather',1972),
19     ('12 Angry Men',1957);
20
21 INSERT INTO
22     mostPopularFilms(title,release_year)
23 VALUES
24     ('An American Pickle',2020),
25     ('The Godfather',1972),
26     ('Greyhound',2020);
```



The screenshot shows the PostgreSQL Query Editor interface with the 'Query Editor' tab active. The SQL script in the editor is:

```
1  --The following shows the data from the topRatedFilms table
2  SELECT * FROM topRatedFilms;
```

Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table format:

	title character varying	release_year smallint
1	The Shawshank Redemption	1994
2	The Godfather	1972
3	12 Angry Men	1957

A tooltip 'Read-only column' is visible over the 'release_year' column header.

Dashboard Properties SQL Statistics Dependencies Dependents [dvdrental/postgres@PostgreSQL](#)

Query Editor Query History Explain Notifications

```

1 --The following statement returns the data from the most_popular_films table
2 SELECT * FROM most_popular_films;

```

Data Output

	title character varying	release_year smallint
1	An American Pickle	2020
2	The Godfather	1972
3	Greyhound	2020

Simple PostgreSQL UNION example

Dashboard Properties SQL Statistics Dependencies Dependents [dvdrental/postgres](#)

Query Editor Query History Explain Notifications

```

1 --statement uses the UNION operator to combine data from both tables
2 SELECT * FROM top_rated_films
3 UNION
4 SELECT * FROM most_popular_films;

```

Data Output

	title character varying	release_year smallint
1	An American Pickle	2020
2	Greyhound	2020
3	The Shawshank Redemption	1994
4	The Godfather	1972
5	12 Angry Men	1957

PostgreSQL UNION ALL

Dashboard Properties SQL Statistics Dependencies Dependents dvdrental/postgres@Postgre!

Query Editor Query History Explain Notifications

```
1 --The following statement uses the UNION ALL operator to combine result sets
2 --from the top_rated_films and most_popular_films tables
3 SELECT * FROM top_rated_films
4 UNION ALL
5 SELECT * FROM most_popular_films;
```

Data Output

	title character varying	release_year smallint
1	The Shawshank Redemption	1994
2	The Godfather	1972
3	12 Angry Men	1957
4	An American Pickle	2020
5	The Godfather	1972
6	Greyhound	2020

PostgreSQL UNION ALL with ORDER BY clause

Dashboard Properties SQL Statistics Dependencies Dependents

Query Editor Query History Explain Notifications

```
1 -- PostgreSQL UNION ALL with ORDER BY clause
2 SELECT * FROM top_rated_films
3 UNION ALL
4 SELECT * FROM most_popular_films
5 ORDER BY title;
```

Data Output

	title character varying	release_year smallint
1	12 Angry Men	1957
2	An American Pickle	2020
3	Greyhound	2020
4	The Godfather	1972
5	The Godfather	1972
6	The Shawshank Redemption	1994