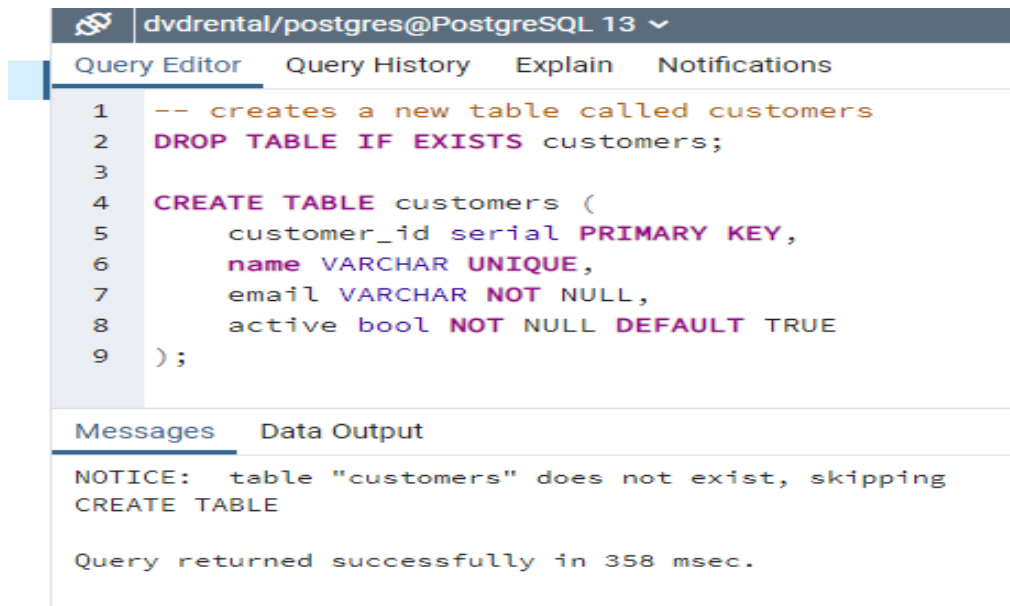


POSTGRESQL UPSERT



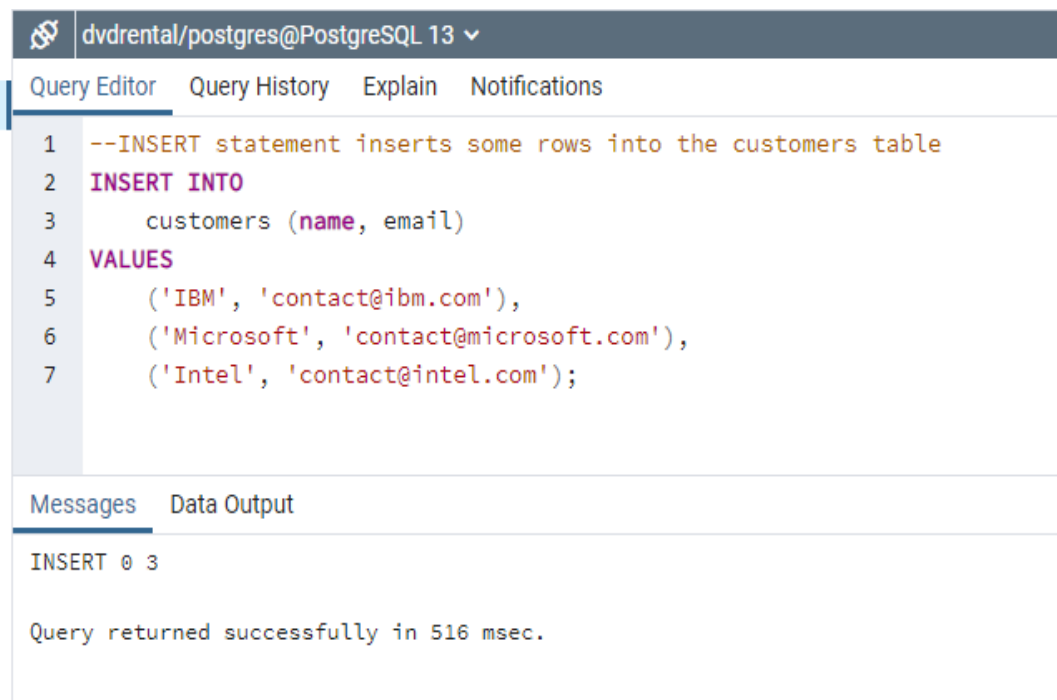
The image shows a PostgreSQL Query Editor window. The title bar indicates the connection is 'dvdrental/postgres@PostgreSQL 13'. The 'Query Editor' tab is active. The SQL code in the editor consists of a comment followed by a DROP statement and a CREATE TABLE statement for a table named 'customers'. The 'Messages' tab is also visible, showing a notice that the table 'customers' does not exist and was skipped, and a success message indicating the query completed in 358 msec.

```
1  -- creates a new table called customers
2  DROP TABLE IF EXISTS customers;
3
4  CREATE TABLE customers (
5      customer_id serial PRIMARY KEY,
6      name VARCHAR UNIQUE,
7      email VARCHAR NOT NULL,
8      active bool NOT NULL DEFAULT TRUE
9  );
```

Messages Data Output

NOTICE: table "customers" does not exist, skipping
CREATE TABLE

Query returned successfully in 358 msec.



The image shows a second PostgreSQL Query Editor window with the same connection. The 'Query Editor' tab is active. The SQL code is an INSERT INTO statement for the 'customers' table, inserting three rows with company names and email addresses. The 'Messages' tab shows the result 'INSERT 0 3', indicating that 3 rows were inserted. A success message at the bottom states the query returned successfully in 516 msec.

```
1  --INSERT statement inserts some rows into the customers table
2  INSERT INTO
3      customers (name, email)
4  VALUES
5      ('IBM', 'contact@ibm.com'),
6      ('Microsoft', 'contact@microsoft.com'),
7      ('Intel', 'contact@intel.com');
```

Messages Data Output

INSERT 0 3

Query returned successfully in 516 msec.

 dvdrental/postgres@PostgreSQL 13 ▾

Query Editor

Query History

Explain

Notifications


```
1 -- Microsoft changes the contact email from contact@microsoft.com to hotline@microsoft.com, we can update
2 -- using the UPDATE statement. However, to demonstrate the upsert feature,
3 -- use the following INSERT ON CONFLICT statement
4 INSERT INTO customers (NAME, email)
5 VALUES('Microsoft','hotline@microsoft.com')
6 ON CONFLICT ON CONSTRAINT customers_name_key
7 DO NOTHING;
```

Messages

Data Output

INSERT 0 0

Query returned successfully in 476 msec.

 dvdrental/postgres@PostgreSQL 13 ▾

Query Editor

Query History

Explain

Notifications

```
1 -- statement is equivalent to the above statement but it uses the name column instead of
2 --- unique constraint name as the target of the INSERT statement.
3 INSERT INTO customers (name, email)
4 VALUES('Microsoft','hotline@microsoft.com')
5 ON CONFLICT (name)
6 DO NOTHING;
```

Messages

Data Output

INSERT 0 0

Query returned successfully in 483 msec.



dvdrental/postgres@PostgreSQL 13 ▾

Query Editor Query History Explain Notifications

```
1  -- use the UPDATE clause as the action of the INSERT statement as follows
2  INSERT INTO customers (name, email)
3  VALUES('Microsoft','hotline@microsoft.com')
4  ON CONFLICT (name)
5  DO
6      UPDATE SET email = EXCLUDED.email || ';' || customers.email;
```

Messages Data Output

INSERT 0 1

Query returned successfully in 298 msec.