Logical Replication in Postgres

Introduction to Logical Replication

Logical replication in PostgreSQL is designed to replicate data changes at the logical level, enabling you to replicate specific tables or entire databases across PostgreSQL instances. Unlike physical replication, which replicates the entire database cluster, logical replication allows for more granular control and is particularly useful for scenarios such as data migration, high availability, or multi-master replication setups.

Prerequisites

- Two PostgreSQL instances running on different servers or the same server for simplicity.
- Basic familiarity with SQL commands and PostgreSQL configuration.

Setup Overview

In this example, we'll configure logical replication between two PostgreSQL databases: `db1` (the primary database) and `db2` (the subscriber database). Our goal is to replicate a simple table from `db1` to `db2`.

1. Preparing the Primary Database (`db1`)

First, we need to set up the primary database to publish changes to a table. Follow these steps:

Create the Table and Insert Initial Data

Connect to `db1` and execute the following commands to create a table and insert some initial data:

```
CREATE TABLE table1 (ad VARCHAR(15),soyad VARCHAR(15) );

INSERT INTO table1 (ad, soyad) VALUES ('Kemal', 'oz');
```

Create a Publication

A publication is a named collection of changes (inserts, updates, and deletes) that you want to replicate. Create a publication for all tables:

```
CREATE PUBLICATION alltables12 FOR ALL TABLES;
```

2. Preparing the Subscriber Database (`db2`)

Next, configure the subscriber database to receive data from the publication created on `db1`.

Create the Table

Ensure that `db2` has the same table structure as `db1`:

```
CREATE TABLE table1 (ad VARCHAR(15),soyad VARCHAR(15) );
```

Create a Subscription

A subscription connects to a publication and applies changes from it. Execute the following command to create a subscription in `db2`:

```
CREATE SUBSCRIPTION mysub1 CONNECTION 'host=localhost port=5432 user=postgres dbname=test9'
PUBLICATION alltables;
```

Replace the connection parameters (`host`, `port`, `user`, and `dbname`) with those appropriate for your setup.

3. Testing the Replication

To verify that replication is working, insert new data into the table on `db1` and check if it appears in `db2`.

Insert Data into `db1`

Execute the following command on `db1`:

```
INSERT INTO table1 (ad, soyad) VALUES ('alı', 'oz');
```

Verify Data in `db2`

Connect to `db2` and check the table contents:

```
SELECT * FROM table1;
```

You should see both the initial data and the new data inserted into `db1`:

```
ad    | soyad
-------+-------
Kemal | oz
alı   | oz
```

Conclusion

Logical replication in PostgreSQL provides a flexible and powerful method to synchronize data between databases. By following these steps, you have set up a basic replication scenario between two PostgreSQL instances. You can extend this setup to more complex scenarios, such as replicating specific subsets of data or implementing multi-master replication.