Crunchy Containers

Standalone Examples

We provide some examples of running the Crunchy containers in the examples/standalone directory. Those examples are explained below.

Example 1 - Running crunchy-ose-pg without using Openshift

Run the container with this command:

```
./run-pg-master.sh
```

This script will do the following:

- start up a container named master
- create a data directory for the database in /tmp/master-data
- initialize the database using the passed in environment variables, building a user, database, schema, and table based on the environment variable values.
- maps the PostgreSQL port of 5432 in the container to your local host port of 12000.

The container creates a default database called 'testdb', a default user called 'testuser' with a default password of 'testpsw', you can use this to connect from your local host as follows:

```
psql -h localhost -p 12000 -U testuser -W testdb
```

To shut down the instance, run the following commands:

```
docker stop master
```

To start the instance, run the following commands:

```
docker start master
```

Example 2 - Performing a backup

In order to run this backup script, you first need to edit run-backup.sh to specify your host IP address you are running on. The script assumes you are going to backup the container created in Example 1.

Run the backup with this command:

```
./run-backup.sh
```

This script will do the following:

- start up a backup container named masterbackup
- run pg_basebackup on the container named master
- store the backup in /tmp/backups/masterbackup directory
- exit after the backup

Example 3 - Running PostgreSQL in Master-Slave Configuration

The container can be passed environment variables that will cause it to assume a PostgreSQL replication configuration with a master instance streaming to a slave replica instance.

The following env variables are specified for this configuration option:

- PG_MASTER_USER - The username used for master-slave replication value=master
- PG_MASTER_PASSWORD - The password for the PG master user
- PG_USER - The username that clients will use to connect to PG server value=user
- PG_PASSWORD - The password for the PG master user
- PG_DATABASE - The name of the database that will be created value=userdb
- PG_ROOT_PASSWORD - The password for the PG admin

This examples assumes you have run Example 1, and that the master container is running.

For running the master-slave configuration , you can run the following scripts:

```
run-pg-slave.sh
```

You can verify that replication is working by the following commands:

```
psql -h 0.0.0.0 -p 12001 -U postgres postgres
```

Example 4 - pgpool (non-Openshift)

A pgpool example is provided that will run a pgpool container that is configured to be used with the master and slave example provided in the run-pg-master.sh and run-pg-replica.sh scripts. After running those commands to create a master and replica, you can create a pgpool container by running the following example command:

```
sudo ./run-pgpool.sh
```

Enter the following command to connect to the pgpool that is mapped to your local port 12002, in this case the host is named jeffded.crunchy.lab, fill in your hostname instead:

```
psql -h jeffded.crunchy.lab -U testuser -p 12002 testdb
```

You will enter the password of testpsw when prompted. At this point you can execute both INSERT and SELECT statements on the pgpool connection. Pgpool will direct INSERT statements to the master and SELECT statements will be sent round-robin to both master and replica.