A CLOUD GURU

Lesson Description - Examining the Problem & Prep Work

In this last segment, we're tackling a single, large problem over multiple videos. We'll dig into development practices that we can utilize to ensure the success of our projects.

Our approach will include:

1. Project Planning
2. Documentation
3. Test Driven Development (TDD)

Through Test Driven Development, we'll run into a wide variety of errors and establish a familiarity with the stack trace that will make debugging projects in the future easier.

Links For This Video

- db_setup.sh
- PostgreSQL RPM

The Project

We have many database servers that we manage, and we want to create a single tool that we can use to easily back up the databases to either AWS S3 or locally. We would like to be able to:

1. Specify the database URL to backup.
2. Specify a "driver" (`local` or `s3`)
3. Specify the backup "destination". This will be a file path for `local` and a bucket name for `s3`.
4. Depending on the "driver", create a local backup of the database or upload the backup to an S3 bucket.

Setting up PostgreSQL Lab Server

Before we begin, we're going to need to need a PostgreSQL database to work with. The code repository for this course contains a `db_setup.sh` script that we'll use on a CentOS 7 cloud server to create and run our database. Create a "CentOS 7" cloud server and run the following on it:

```
$ curl -o db_setup.sh https://raw.githubusercontent.com/linuxacademy
/content-python3-sysadmin/master/helpers/db_setup.sh
$ chmod +x db_setup.sh
$ ./db_setup.sh
```

You will be prompted for your sudo password and for the username and password you'd like to use to access the database.

Installing The Postgres 9.6 Client

On our development machines, we'll need to make sure that we have the Postgres client installed. The version needs to be 9.6.6.

On Red-hat systems we'll use the following:

```
$ wget https://download.postgresql.org/pub/repos/yum/9.6/redhat/
rhel-7-x86_64/pgdg-redhat-repo-latest.noarch.rpm
$ sudo yum install -y pgdg-redhat-repo-latest.noarch.rpm
$ sudo yum update -y
$ sudo yum autoremove -y postgresql
$ sudo yum install -y postgresql96
```

On debian systems, the equivalent would be:

```
$ sudo apt-get install postgres-client-9.6
```

Test connection from Workstation

Let's make sure that we can connect to the PostgreSQL server from our development machine by running the following command:

*Note: You'll need to substitute in your database user's values for `[USERNAME]`, `[PASSWORD]`, and `[SERVER_IP]`.

```
$ psql postgres://[USERNAME]:[PASSWORD]@[SERVER_IP]:80/sample -c
"SELECT count(id) FROM employees;"
```

With this prep work finished, we're ready to start planning the project itself.