Introduction to Python Development

CHAPTER 10.2
Setting Up External Dependencies

Before we dig too far into our pgbackup project we'll need to make sure that we have access to AWS S3 buckets and also have a PostgreSQL database to interact with. In this lesson, we'll use the Linux Academy Cloud Playground to deploy the database and set up an AWS user with access to S3.

Note: You'll need to have an AWS account and be able to create a new user in IAM.

Documentation For This Video
db_setup.sh (https://raw.githubusercontent.com/linuxacademy/content-intro-to-python-development/master/helpers/db_setup.sh)
PostgreSQL RPM (https://download.postgresql.org/pub/repos/yum/9.6/redhat/rhel-7-x86_64/pgdg-centos96-9.6-3.noarch.rpm)
The AWS CLI (https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-install.html)

Setting up PostgreSQL Cloud Playground
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 Playground to create and run our database. Create a "CentOS 7" Cloud Playground and run the following on it:

```
$ curl -o db_setup.sh https://raw.githubusercontent.com/linuxacademy/content-python-for-sys-admins/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.

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

```
$ sudo yum install -y https://download.postgresql.org/pub/repos/yum/reporpms/EL-7-x86_64/pgdg-redhat-repo-latest.noarch.rpm
$ sudo yum update -y
$ sudo yum autoremove -y postgresql
$ sudo yum install -y postgresql96-server
$ sudo /usr/pgsql-9.6/bin/postgresql96-setup initdb
$ sudo systemctl enable postgresql-9.6
$ sudo systemctl start postgresql-9.6
```

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;"
```

Setting Up Our AWS User
The last external dependency that we have is Amazon S3, so we're going to need an AWS account for this step.

NOTE: Linux Academy Cloud Sandboxes do not permit IAM user creation, so you must use your own account. AWS Free Tier accounts are available [here](https://aws.amazon.com/free/).

Once we're logged in we need to do the following:

Open IAM console
Select "Users" from the sidebar
Click "Add User"
Set the user name, and give programmatic access
Select "Attach existing policies directly"
Add the following permissions to the new user:
AmazonS3FullAccess
Continue through tags and review.
Now we have a user, copy the "Access Key ID" and the "Secret access key" so that we can use them in a moment.

Installing and Configuring the AWS CLI
The AWS CLI is written in Python and we can install it globally by exiting our virtualenv for a moment and running this command:

```
(pgbackup) $ exit
$ pip3.7 install --user awscli
```

Next, we'll take the access key ID and secret access key that we copied before to configure out AWS client:

```
$ aws configure
AWS Access Key ID [None]: XXXXXXXXXXXXXXXXXXX
AWS Secret Access Key [None]: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
Default region name [None]:
Default output format [None]:
```

That will create a ~/.aws directory that includes our default configuration and credentials.

Creating Our S3 Bucket
From within the AWS Console, we need to create a new S3 bucket that we can use with our project. To do this, we need to search for "S3" in the services. Next, click "Create Bucket", give it a name, and use the default settings.

Once the bucket is created, we'll select it by name and go to the "Permissions" tab so that we can change a few things. From the "Permissions" tab, select the "Access Control List" sub-tab and then for "Public Access" allow "List objects".

Now we should be able to add items to our S3 bucket and make them publically readable to make things a little easier for us later.