**How To Launch Postgres RDS With AWS Command Line Interface**

Now that we can launch a bootstrapped EC2 ([**How To Launch EC2 With Bootstrapping in AWS**](https://www.mydatahack.com/how-to-launch-ec2-with-bootstrap-in-aws)) with AWS CLI, let’s explore how to launch Postgres RDS with AWS CLI. As in the previous post, we can basically code infrastructure and resources in AWS. This is a step toward coding up your entire infrastructure.

This entry is a bonus step from [**How To Create Your Personal Data Science Computing Environment In AWS**](https://www.mydatahack.com/how-to-create-your-personal-data-science-computing-environment-in-aws). In the post, we have a step to launch Postgres RDS from the management console ([How To Launch an RDS Instance In a Specific Subnet](https://www.mydatahack.com/how-to-launch-a-rds-instance-in-a-specific-subnet)). Here, we can use a line of code to do it.

First of all, you need to install AWS CLI and configure it (read the previous post here).

We are going to use two rds commands: [create-db-instance](https://docs.aws.amazon.com/cli/latest/reference/rds/create-db-instance.html) (for creating a rds instance) and [delete-db-instance](https://docs.aws.amazon.com/cli/latest/reference/rds/delete-db-instance.html) (for deleting it).

**create-db-instance**

The key point here is to specify the VPC security group and DB Subnet group appropriately and set avaliablitiy zone to the second one according to our plan [here](https://www.mydatahack.com/how-to-create-your-personal-data-science-computing-environment-in-aws).

Make sure to include the publicly-accessible value so that you can get the end point to connect to DB from your local machine. As default, Multi AZ is turned off. Postgres engine is the latest one unless you specify the version.

The db-instance-identifier is required. DB instance identifier is an unique id for each RDS.

[cc lang="text" tab\_size="4" lines="-1" theme="mac-classic" line\_numbers="false"]

aws rds create-db-instance

--db-name mydatahack ^

--db-instance-identifier mydatahackpgres ^

--allocated-storage 20 ^

--db-instance-class db.t2.micro ^

--engine postgres ^

--master-username mydatahack ^

--master-user-password <password> ^

--publicly-accessible ^

--vpc-security-group-ids <vpc security group for rds> ^

--db-subnet-group <subnet group e.g. default-yourvpcid> ^

--availability-zone <zone 2 according to our plan> ^

--port 5432

[/cc]

**delete-db-instance**

Now, let’s delete the RDS instance with AWS CLI. Use DB instance identifier that you specified when you launched the RDS for delete.

You can either include final snapshot or skip it. In the example below, I am skipping the final snapshot. It is usually a good idea to do a snapshot and launch the new RDS from it if you need it later.

[cc lang="text" tab\_size="4" lines="-1" theme="mac-classic" line\_numbers="false"]

aws rds delete-db-instance

--db-instance-identifier mydatahackpgres ^

--skip-final-snapshot

[/cc]

Fun!