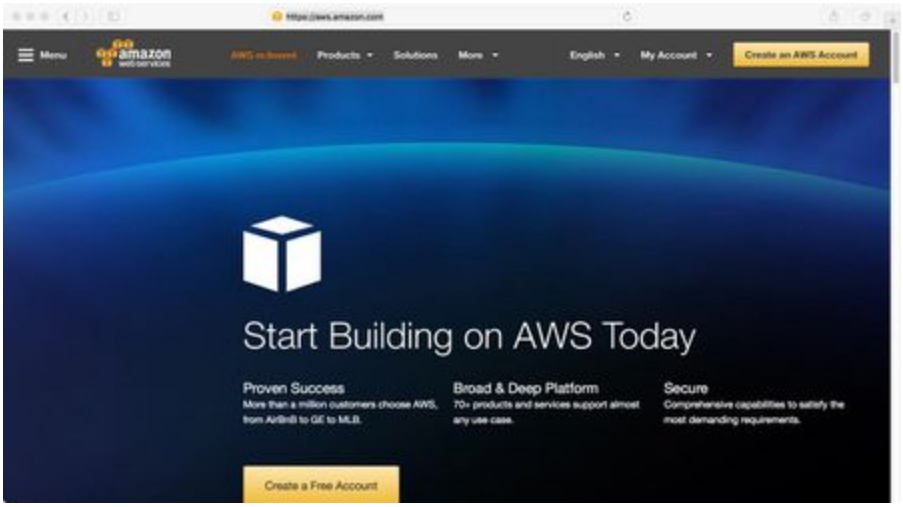
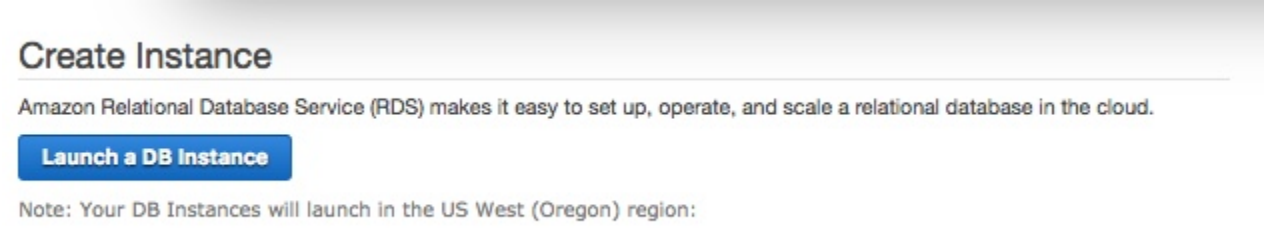
Setting up a Database and PgAdmin4

1. Go to https://aws.amazon.com/
2. Click on create an aws account.

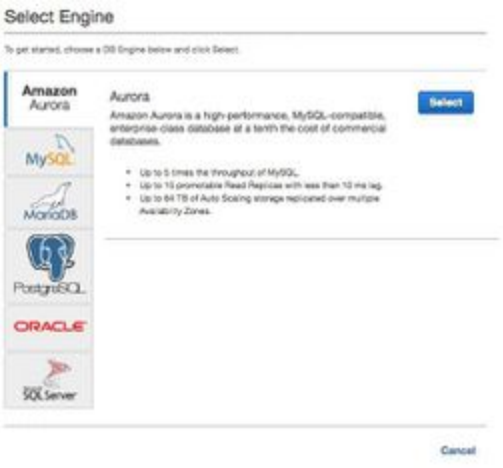


3. Fill in your personal information. You will have to enter a credit card. You will not be charged as long as you select free options. You could also used a prepaid card if you are concerned. You will have to verify your account by text or phone call.
4. Click on services, under Database click on RDS.

5. Next, you will click on Launch a DB Instance under Create Instance.



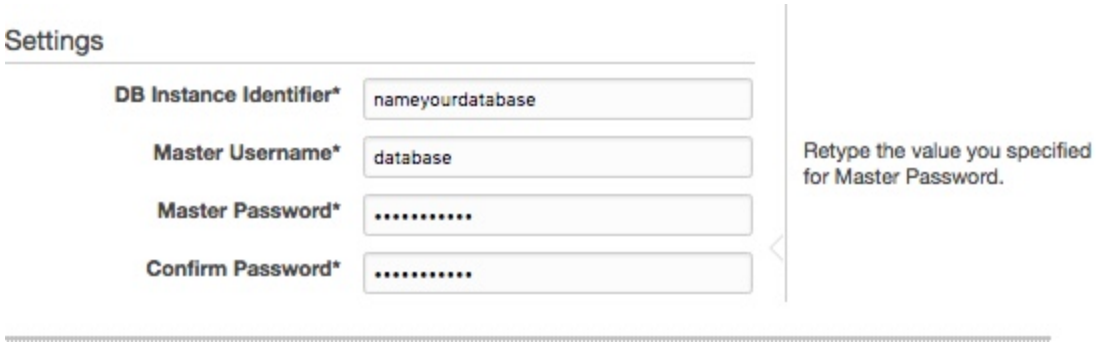
6. You will then click on PostgreSQL in the left hand side of the page and click select.



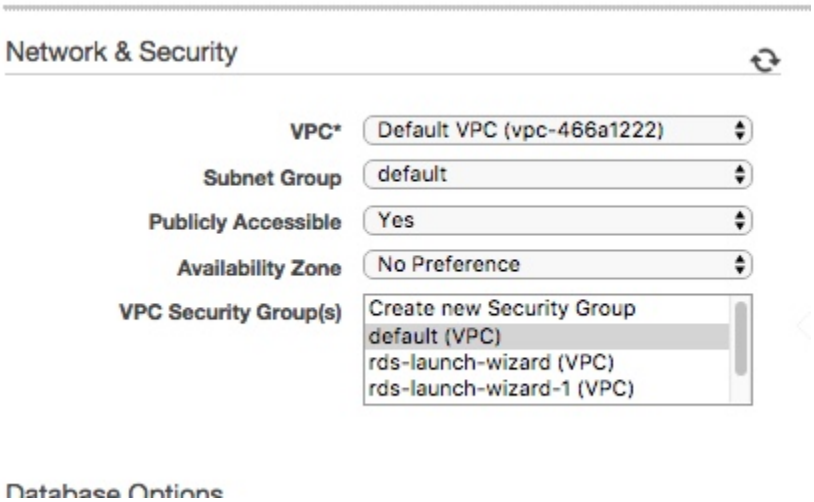
7. Next, choose the option under DEV/TEST, click next step.

8. Click the box next to Only show options that are eligible for RDS Free Tier.

9. Next, you will choose a name for your database under the settings in the DB Instance Identifier field. Name it something that is related to the project you are working on.

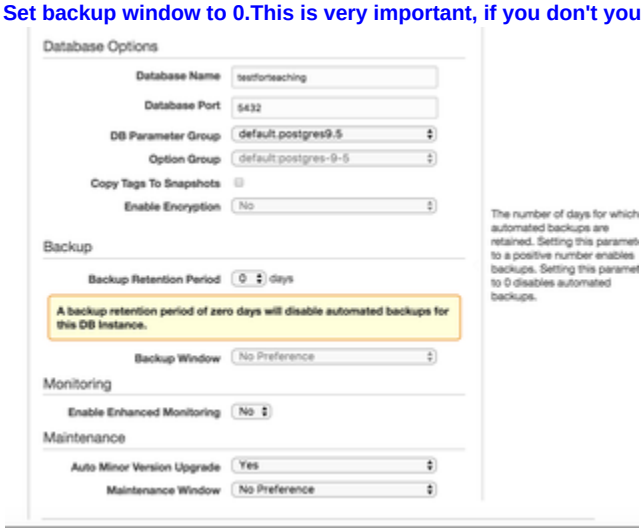


10. Set a username and password for the database. Make sure you write it down so you can access the database.
11. On the next page you will put the database name in the box under Database options. Then select default under security group.

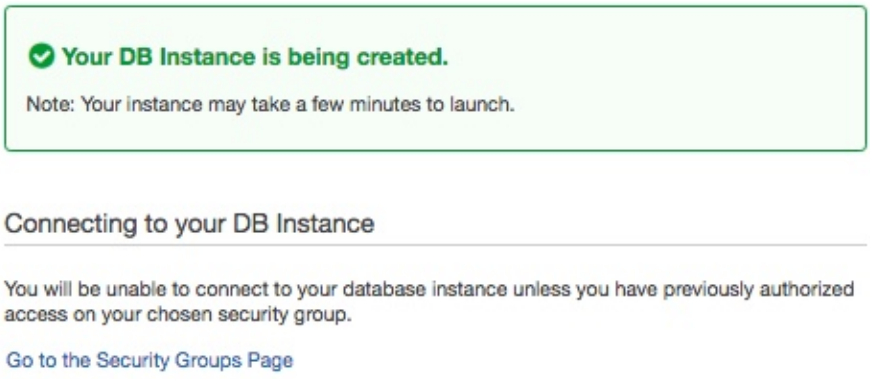


Database Options

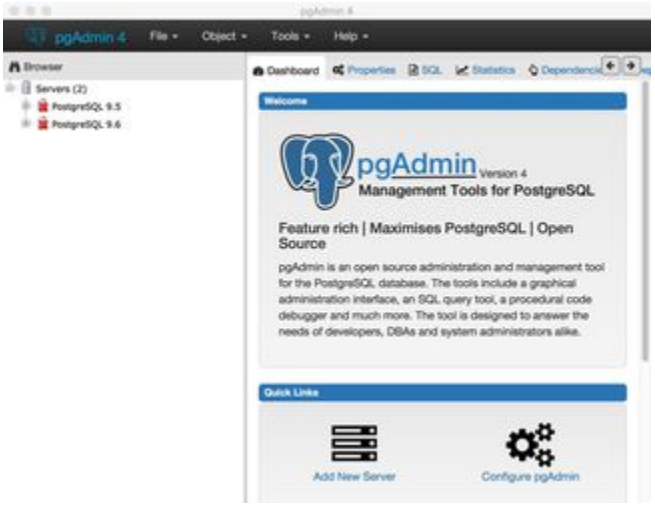
Set backup window to 0.This is very important, if you don't you will be charged a fee each month.



Click Launch DB Instance.
It will take several minutes for the DB to create.



12. While waiting for DB to create you will want to go to https://www.pgadmin.org/ and download pgadmin4 for your operating system and install. This is what you will use to view your database, make changes, and set up your tables.
13. You will also need to install the PG gem in your terminal by doing the following (gem install pg) minus the parenthesis. *For mac users you will need to have x-code installed. https://developer.apple.com/xcode/. You have to open x-code once it installs and accept the agreement before you can use it. Then you can (`brew install postgresql`) minus parenthesis, and then (`gem install pg`) minus parenthesis. *If you are having issues installing the pg gem with windows make sure you have devkit installed `http://rubyinstaller.org/downloads/`. Searching google can also help you solve any errors you may be getting on installation of gems.
14. Once you have installed pgadmin4 and your DB is finished creating you will need to set up pgadmin4 to be able to see the DB. You do this by clicking on the the new server link under quick links.



You will need some info from the database you have just created.

15. Click in the left hand side of the screen it says security group. Click on EC2 console in the blue area of the screen. Then click default, then go to the bottom of page and click on inbound tab. Click Edit, then source and change it to anywhere. Click save.
16. Click on the database on the AWS site. If you click on the little icon that looks like paper with a magnifying glass you will see a section called Writer Endpoint. Copy the link. Now in the window that opened when you clicked the plug icon on pgadmin4 you will paste the link in the Host field.
17. You will also enter a name (can be the same as you called the database), the username and password you gave the database will also be entered now.Click OK.
18. Now you will see the database name in the list on the left hand side of the window. If you double click you will see Databases, double click on the one you named. It will bring down more categories, click Schemas, public, Tables. Right click on Tables and select new table.
19. Now you will create your first table of data. Start by giving the table a name. (ex. Products, users, etc) You want it to represent what data will in the table.
20. Once you have given a name you will want to click on the column tab in the top of the box. Click add. Now you will name your column.(ex. Product, description, size, color, price ,or if you are doing a user table you might add columns like name, address, city,state,zip,phone.)You will also need to set the data type of each column you create. This is done under the name field. The biggest part of the time you will use text or integer. There are times you will use other and a simple Google search will help you decide which one could be of use. You will then click ok.
21. To add data you will write a sequel statement. This will go in your app.rb file for the project you are working on. A statement can be used to add or update your table. An example would be db.exec("INSERT INTO users(fname,lname,address,city,state,zipcode) VALUES

('#{firstname}','#{lastname}','#{street}','#{city}','#{state}','#{zip}')") . Here you are inserting first name, last name, address, city, state, and zip code into the columns of the same values. This will populate your database table.

22. You will need to put your database information in your app.rb. You will require the pg gem like so(require 'pg')minus the parenthesis.

```
require 'sinatra'
require 'pony'
require 'pg'
require 'bundler/setup'
require 'recaptcha'
load "./local_env.rb" if File.exists?("./local_env.rb")
```

23. Next, you will add the following lines to your app.rb under where you require the gems. The code will resemble the following

```
require 'sinatra'
require 'pony'
require 'pg'
require 'bundler/setup'
require 'recaptcha'
load "./local_env.rb" if File.exists?("./local_env.rb")


db_params = {
  host: ENV['host'],
  port: ENV['port'],
  dbname: ENV['dbname'],
  user: ENV['user'],
  password: ENV['password']
}

db= PG::Connection.new(db_params)
```

As you can see we used local environmental variables so we are not sharing your private database information with everyone once you push to github. Refer to the how to guide on environmental variables to learn more.

24. When you create your tables you will be asked about making a primary key. You will use this as a way of making your table unique. The primary key can be an id, customer number, etc. There can only be one primary key per table. The key lets the column be used as a unique identifier. You can't have duplicate information in the primary key field. This can be helpful when dealing with a table of customers. If you set a primary key to the customer id field you will know each row represents one customer. If you want to be able to edit your table directly from pgadmin4 you must have a primary key. Primary keys cannot contain null values
25. The next constraint we will discuss is foreign keys. This allows two tables to communicate between each other. It sets a constraint on one table based off of the primary key from another table. Foreign keys can contain null values. The foreign key creates a link between two tables.
26. To add data from your app/site to the database you will have code that resembles the following.

post '/subscribe' do
email = params[:email]
*check_email = db.exec("SELECT * FROM mailing_list WHERE email = '#{email}'")*

if
check_email.num_tuples.zero? == false
erb :mailing_list, :locals => {:message => "You have already joined our mailing list"}
else
subscribe = db.exec("insert into mailing_list(email)VALUES('#{email}')")
erb :mailing_list, :locals => {:message => "Thanks, for joining our mailing list."}
end
end
This is the code that will go in the app.rb. It is for a mailing list signup. The code on the page of your site where the user actually enters their email will look like the following.

<h4>Join our newsletter</h4>

<form class="subscribe" action="subscribe" method="POST">

```
<div class="input-group">

<input type="text" class="form-control"name="email" placeholder="mail@example.com">

<span class="input-group-btn">

<button class="btn btn-default" type="submit"><i class="fa fa-send"></i></button>

</span>

</div>
<!-- /input-group -->
</form>
```

Here is what the box looks like on the web page.

27. There is a lot to learn about databases. There are many online resources that can help. One of the best resources is Treehouse. There is a monthly cost to use the site, but they also offer a free trial. It is well worth the cost. http://www.tutorialspoint.com/postgresql/ offers a free tutorial to help you learn Postgresql. http://www.w3resource.com/PostgreSQL/tutorial.php is also a great resource to bookmark for future reference.