Homework 3 Spatial Database

In this homework, you are going to work with **spatial data** - you will create some data, visualize it, do queries on it, and visualize the query results. The exercise will give you a taste of working with spatial data, use of a spatial file format and spatial query functions, all of which are quite useful from a real-world (or job interview) perspective.

## Part 1 Setting up PostgreSQL on Amazon AWS

This tutorial is mainly following the User Guide on http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide//CHAP_SettingUp.html and http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide//CHAP_GettingStarted.Creating Connecting.PostgreSQL.html. The only difference is that I attached some screenshot to help you do the setup process

Task 1 Sign Up for AWS

Go to https://aws.amazon.com/free/ and "create a free account". I think the USC email address is fine, even though I only tested with my person email address. It will give you a 12-month free trial.

Please Choose "Personal Account", the registration process would require you to input credit card info and phone verification. Once you finish registration, we can start to login by clicking the following button.
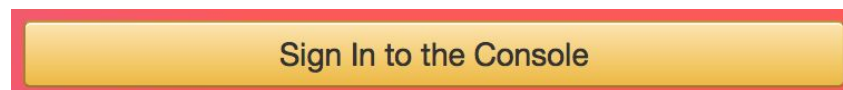
Task 2 Create an IAM User

1. Sign in to the IAM console at https://console.aws.amazon.com/iam/.

2. In the navigation pane, choose **Users**, and then choose **Add user**
3. For **User name**, type a user name, such as `admin`.
4. Then do the following setting as suggested by the webpage, including password.

Add user

1 **Details**　2 Permissions　3 Review　4 Complete

Set user details

You can add multiple users at once with the same access type and permissions. Learn more

User name*　admin

➕ **Add another user**

Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. Learn more

Access type*　☐ **Programmatic access**
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.
☑ **AWS Management Console access**
Enables a **password** that allows users to sign-in to the AWS Management Console.

Console password*　○ Autogenerated password
● Custom password

•••••　🔑⌄
☐ Show password

5. Choose **Next: Permissions**.

6. On the **Set permissions for user** page, choose **Add user to group**.

1 Details　2 **Permissions**　3 Review　4 Complete

Set permissions for admin

Add user to group　Copy permissions from existing user　Attach existing policies directly

ℹ️ Get started with groups
You haven't created any groups yet. Using groups is a best-practice way to manage users' permissions by job functions, AWS service access, or your custom permissions. Get started by creating a group. Learn more

Create group

7. Choose **Create group**.

8. In the **Create group** dialog box, type the name for the new group. The name can consist of letters, digits, and the following characters: plus (+), equal (=), comma (,), period (.), at (@), underscore (_), and hyphen (-). The name is not case sensitive and can be a maximum of 128 characters in length.

9. For **Filter**, choose **Job function**.

10. In the policy list, select the check box for **AdministratorAccess**. Then choose **Create group**.

Create group   ✕

Create a group and select the policies to be attached to the group. Using groups is a best-practice way to manage users' permissions by job functions, AWS service access, or your custom permissions. Learn more

Group name   adminGroup

Create policy   ⟳ Refresh

Filter: Job function ⌄   🔍 Search   Showing 10 results

		Policy name ▾	Type	Attachments ▾	Description
☑	▸	🟧 AdministratorAccess	Job function	0	Provides full access to AWS services and resources.
☐	▸	🟧 Billing	Job function	0	Grants permissions for billing and cost management. This in…
☐	▸	🟧 DatabaseAdministrator	Job function	0	Grants full access permissions to AWS services and actions…
☐	▸	🟧 DataScientist	Job function	0	Grants permissions to AWS data analytics services.
☐	▸	🟧 NetworkAdministrator	Job function	0	Grants full access permissions to AWS services and actions…

11. Back in the list of groups, select the check box for your new group. Choose **Refresh** if necessary to see the group in the list.

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. Learn more

Create group   ⟳ Refresh

🔍 Search   Showing 1 result

	Group ▾	Attached policies
☑	adminGroup	AdministratorAccess

12. Choose **Next: Review** to see the list of group memberships to be added to the new user. When you are ready to proceed, choose **Create user**.

Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

User details

User name	admin
AWS access type	AWS Management Console access - with a password
Console password type	Custom
Require password reset	No

Permissions summary

The user shown above will be added to the following groups.

Type	Name
Group	adminGroup

Cancel   Previous   **Create user**

If you create the user successfully, you will see the following like this:

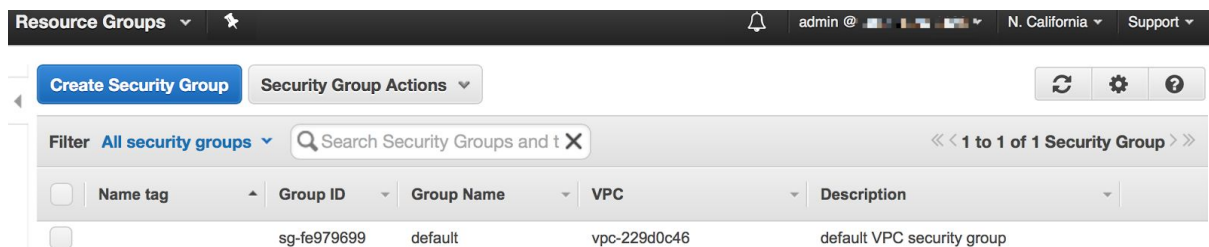✅ **Success**
You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: https://██.██.██.signin.aws.amazon.com/console

Task 3 Provide Access to the DB Instance in the VPC by Creating a Security Group

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc. Using the user and password you just created.

2. In the top right corner of the AWS Management Console, select the region in which you want to create the VPC security group and the DB instance. In the list of Amazon VPC resources for that region, it should show that you have at least one VPC and several Subnets. If it does not, you do not have a default VPC in that region.

3. In the navigation pane, click **Security Groups**.

4. Click **Create Security Group.**

Resource Groups ∨ ★	🔔 admin @ ██ ██ ∨	N. California ∨	Support ∨

Create Security Group	Security Group Actions ∨			↻ ⚙ ❓

Filter **All security groups** ∨ | 🔍 Search Security Groups and t ✕ | « ‹ 1 to 1 of 1 Security Group › »

	Name tag ▲	Group ID ▼	Group Name ▼	VPC ▼	Description ▼
☐		sg-fe979699	default	vpc-229d0c46	default VPC security group

5. In the **Create Security Group** window, type the **Name tag**, **Group name**, and **Description** of your security group. Select the **VPC** that you want to create your DB instance in. Click **Yes, Create**.

6. The VPC security group you created should still be selected. The details pane at the bottom of the console window displays the details for the security group, and tabs for working with inbound and outbound rules. Click the **Inbound Rules** tab.

7. On the **Inbound Rules** tab, click **Edit**. Select **Custom TCP Rule** from the **Type** list. Type the port value you will use for your DB instance in the **PortRange** text box, and then type the IP address range (CIDR value) from where you will be accessing the instance, or select a security group name in the **Source** text box. (At least, set an inbound rule for you to connect the server)

8. The following the instruction, you would finishing this task.

Task 4 Creating a PostgreSQL DB Instance and Connecting to a Database on a PostgreSQL DB Instance

1. Sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/. Using the user you created.
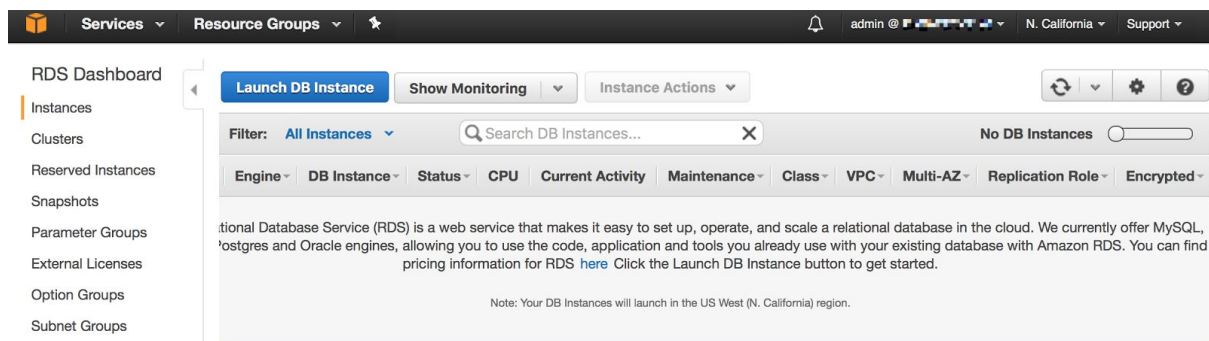
2. In the top right corner of the AWS Management Console, choose the region in which you want to create the DB instance.

3. In the navigation pane, choose **Instances**.

4. Choose **Launch DB Instance** to start the **Launch DB Instance Wizard**. Make sure you check the "Free tier eligible only"

5. Set the DB according to the suggestion roughly as follows:

Step 2:   Specify DB Details

Step 3:   Configure Advanced Settings

ⓘ Your current selection is eligible for the free tier.

Learn More.

ⓘ Estimate your monthly costs for the DB Instance using the RDS Instance Cost Calculator.

Free Tier

The Amazon RDS Free Tier provides a single db.t2.micro instance as well as up to 20 GB of storage, allowing new AWS customers to gain hands-on experience with Amazon RDS. Learn more about the RDS Free Tier and the instance restrictions here.

☑ Only show options that are eligible for RDS Free Tier

Instance Specifications

DB Engine	postgres
License Model	postgresql-license
DB Engine Version	PostgreSQL 9.6.2-R1
DB Instance Class	db.t2.micro — 1 vCPU, 1 GiB RAM
Multi-AZ Deployment	No
Storage Type	General Purpose (SSD)
Allocated Storage*	5 GB

Settings

DB Instance Identifier*	postgreSQL-test
Master Username*	postgres
Master Password*	••••••••
Confirm Password*	••••••••

Step 2:   Specify DB Details

Step 3:   Configure Advanced Settings

Network & Security

VPC*	Default VPC (vpc-229d0c46)
Subnet Group	default
Publicly Accessible	Yes
Availability Zone	No Preference
VPC Security Group(s)	Create new Security Group dbAdmin (VPC) default (VPC)

Database Options

Database Name	postgresCS585
Database Port	5432
DB Parameter Group	default.postgres9.6
Option Group	default:postgres-9-6
Copy Tags To Snapshots	☐
Enable Encryption	No

7

Backup

Backup Retention Period	7 ◇ days
Backup Window	No Preference ◇

Monitoring

Enable Enhanced Monitoring	No ◇

Maintenance

Auto Minor Version Upgrade	Yes ◇
Maintenance Window	No Preference ◇

---

* Required   Cancel   [Previous]   **Launch DB Instance**

6. On the final page of the wizard, choose **View Your DB Instances**.

> ✅ **Your DB Instance is being created.**
>
> Note: Your instance may take a few minutes to launch.

Connecting to your DB Instance

Once Amazon RDS finishes provisioning your DB instance, you can use a SQL client application or utility to connect to the instance.

Learn about connecting to your DB instance

View Your DB Instances

7. You will see the database is being created.

Launch DB Instance	Show Monitoring ⌄	Instance Actions ⌄				⟳ ⌄ ⚙ ❓

Filter:	All Instances ⌄		🔍 Search DB Instances...	✕			Viewing 1 of 1 DB Instances	

	Engine	⌄	DB Instance	⌄	Status	⌄	CPU	Current Activity	Maintenance ⌄	Class	⌄	VPC	⌄	Multi-AZ ⌄	Replicati
▸	PostgreSQL		postgresql-test		creating				None	db.t2.micro		vpc-229d0c46		No	

1. Launch the *pgAdmin* application on your client computer. You can install *pgAdmin* from http://www.pgadmin.org/.

2. Choose **Add Server** from the **File** menu.

3. In the **New Server Registration** dialog box, enter the DB instance endpoint (for example, mypostgresql.c6c8dntfzzhgv0.us-west-2.rds.amazonaws.com) in the **Host** box. Do not include the colon or port number as shown on the Amazon RDS console (mypostgresql.c6c8dntfzzhgv0.us-west-2.rds.amazonaws.com:*5432*).
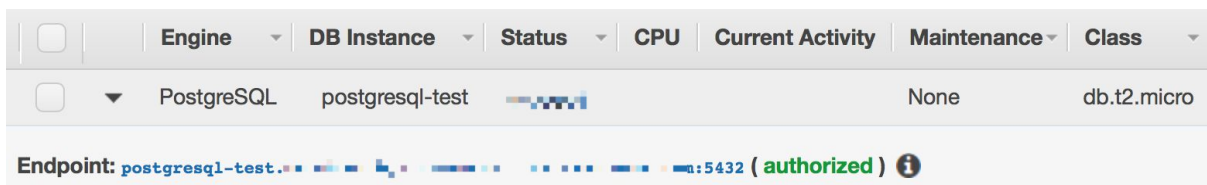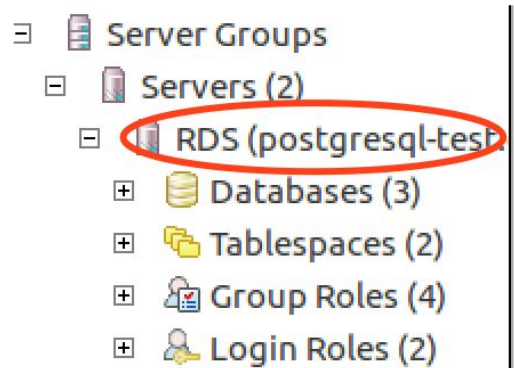

4. Enter the port you assigned to the DB instance into the **Port** box. Enter the user name and user password you entered when you created the DB instance into the **Username** and **Password** boxes, respectively.
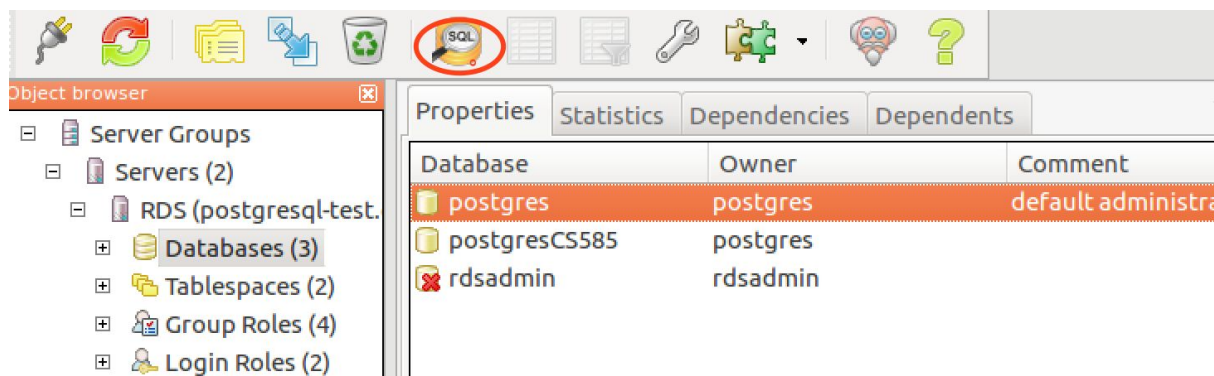

5. Choose **OK**.

6. In the **Object browser**, expand the **Server Groups**. Choose the Server (the DB instance) you created, and then choose the database name.

7. Now you can start to execute the SQL query by click the icon.

Part 2 Assignment Details

1. You need to create (generate) [latitude,longitude] spatial coordinates for 11 locations.
   a. One of those needs to be where your home/apartment/dorm room is.
   b. The other 10 would have to be spread out - spatially distinct, at least 100 feet between adjacent locations (and at most 'several hundred feet' - we don't want to cover a huge region overall!). You can do either way of the following:
      ■ If you are on campus, you can obtain the coords of its four corners (Exposition/Vermont, Vermont/Jefferson, Jefferson/Figueroa, Figueroa/Exposition), and get coordinates for 6 spots inside the campus (classrooms, labs, offices, restaurants, landmarks..).
      ■ If you are a DEN student, get your coordinates from your place of work or neighborhood (again, make sure they are not too close to each other or too far apart).

   How would you obtain spatial coordinates at a location?

You can get the coordinate by using Google Maps easily. Select any point and right click, then choose "What's here?". You will be able to see the coordinate.
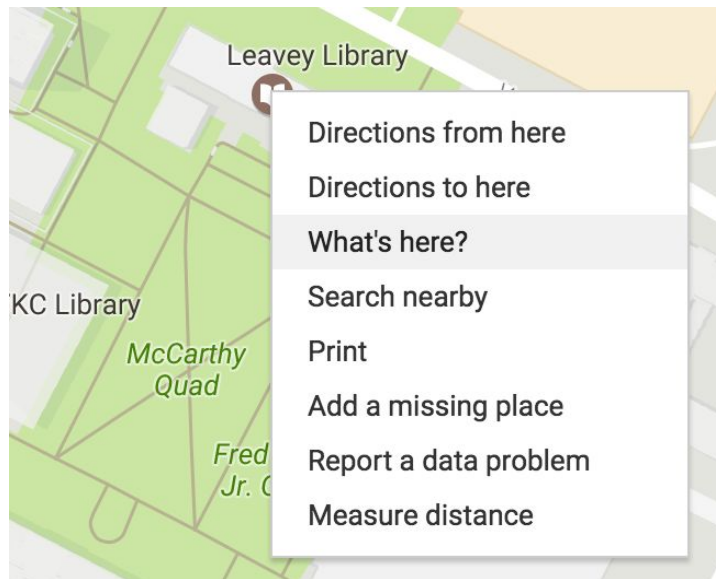
2. Now that you have **11** coordinates and their label strings (ie. text descriptions such as "Tommy Trojan", "SAL", "Chipotle"..), you are going to create a KML file (.kml format) out of them using a text editor. KML is a map-oriented file format, with XML tags. Specifically, each location you surveyed will be a 'placemark' in your .kml file (specified using coords and labels). Here is more detail. The .kml file with the 11 placemarks is going to be your starter file, for doing visualizations and queries. Here is a .kml skeleton to get you started (just download, rename and edit it to put in your coords and labels).

   NOTE - keep each label to be 15 characters or less (including spaces) - otherwise they might not be displayed properly. KML file doesn't allow spaces between coordinate seperator (ie, comma)!

3. Download Google Earth on your laptop, install it, bring it up. Load your .kml file into it - that should show you your 10 sampled locations, on Google Earth's globe :) Take a snapshot (screengrab) of this, for submitting.
4. Setup Postgres + PostGIS (i.e. Part 1).
5. You need to use the above software to execute the following two spatial queries that you will need to write:
   a. 1) **compute the convex hull for your 11 points** [a convex hull for a set of 2D points is the smallest convex polygon that contains the point set]. If you use Postgres, read this. 2) Use the query's result polygon's coords, to create a polygon in your .kml file (edit the .kml file, add relevant XML to specify the KML polygon's coords). Load this into Google Earth, **visually verify that your 11 points are inside the convex hull, then take a screenshot**. (Note that even your data points happen to have a concave perimeter and/or

happen to be self-intersecting, the convex hull, by definition, would be a tight, enclosing boundary (hull) that is a simple convex polygon. The convex hull is a very useful object - eg. see this discussion.. If you want to explore geometry algorithms (of which convex hull computation is one) in more detail, this is a great resource [thanks to Mark Debord (one of the students in our course) for the link) .

b. assuming the points (your collected locations) are called #1,#2,#3....#11, create a polygon using your points #1,#2,#3,#9,#10,#11 (in that order), and another polygon with the remaining points in order (#4,#5,#6,#7,#8). Then **write a query to find out if the two polygons disjoint** - the result would be (Boolean) true or false, depending on your coordinates. See this. **Add these two polygons to your .kml file, and visually verify (in Google Earth) the overlap as being true or false. Take a screenshot.** (feel free to REARRANGE the points #1,#2,#3,#9,#10,#11 to get a non self-crossing polygon (if you get a self-crossing polygon when you don't reorder and that bothers you!); likewise feel free to reorder #4 through #8. Doing so might give you a *different* result for the disjointing [compared to the result from the polygons where the points are all in ascending order], which is fine.

What to submit

- Your .kml file contents from step 5a & 5b above - with the placemarks, convex hull and two region polygons. Please also explain what these points map to, and why you choose them. (2 points)

- Your two queries from step 5 - table creation commands (if you use Postgres and directly specify points in your queries, you won't have table creation commands, in which case you wouldn't need to worry about this part), and the queries themselves. (6 points)

- Screenshots from step 5. (1 point)
- List the issues you met in this homework, and your solutions. (1 point)

Extra credit: 1 point

Using OHE 136 as the center, compute (don't use GPS!) a set (sequence) of lat-long (ie. spatial) coordinates that lie along a pretty Epicycloid curve :) Create a new KML file with these points, visualize it on Google Earth, submit these three items: **your point generation code** (see below), the **resulting .kml file content** and **a screenshot**.

Parametric Equations: $(a = 7, \ b = 3)$

$x(t) = (a + b) \cos t - b \cos((a/b + 1)t)$

$y(t) = (a + b)\sin t - b \sin((a/b + 1)t)$

Using the above equations, loop through t from 0.00 to n*Pi (eg. 2*Pi; note that 'n' might need to be more than 2, for the curve to close on itself; and, t is in radians, not degrees), in steps of 0.01. That will give you the sequence of (x,y) points that make up the Epicycloid curve, which would/should look like the curve in the right side of the screengrab below, when a = 7, b = 3 (my JavaScript code for the point generation loop is on the left):

```
var a=7, b=3;
var x0=a, y0=0;
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.moveTo(150+10*x0, 150+10*y0);

var cos=Math.cos, sin=Math.sin, pi=Math.PI, nRev=
for(var t=0.0;t<(pi*nRev);t+=0.01) {
  var x=(a+b)*cos(t) - b*cos((a/b+1)*t);
  var y=(a+b)*sin(t) - b*sin((a/b+1)*t);
  ctx.lineTo(150+10*x,150+10*y);
}

ctx.stroke();
```
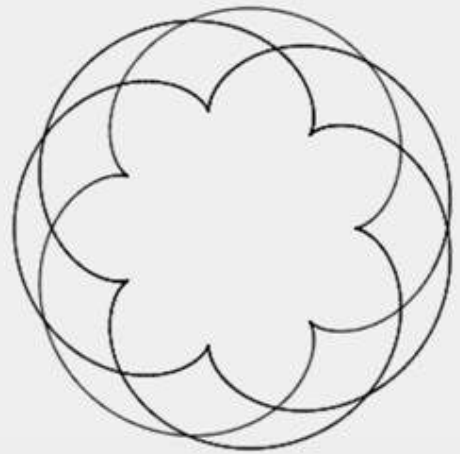
You need to ADD each (x,y) curve point to the (lat,long) of the center, ie. to that of OHE 136 - that will give you valid Epicycloid-based spatial coords for use in your .kml file. You can use any coding language you want, to generate (and visualize) the curve's coords: JavaScript, C/C++, Java, Python, MATLAB, Scala, Haskell, Ruby, R. You can also use Excel, SAS, SPSS, JMP etc., for computing and plotting the points.

Submission Guideline

- *The submission* MUST *be a pdf file named [Student First Name]_[Student Last Name]_HW3.pdf*
- *The deadline is Tuesday, June 20, 2017 11:59 PM. No submissions will be accepted past the deadline.*
- *To avoid to be charged, stop your database once you are not using it.*