COSC 416: Topics in Databases (DBaaS)

TOPIC 3: AMAZON RDS SETUP

SCHEDULE

1. *Amazon RDS*
    1. *Setting up Amazon RDS*
    2. *Security and Server Configuration*
    3. *Connecting to your Database*
    4. *Snapshots and Backups*

SETTING UP AMAZON RDS

AMAZON RDS ACCOUNT

- The first thing that you'll need to do is set up an RDS (AWS) account, if you don't have one
- You can use this link:

https://aws.amazon.com/free

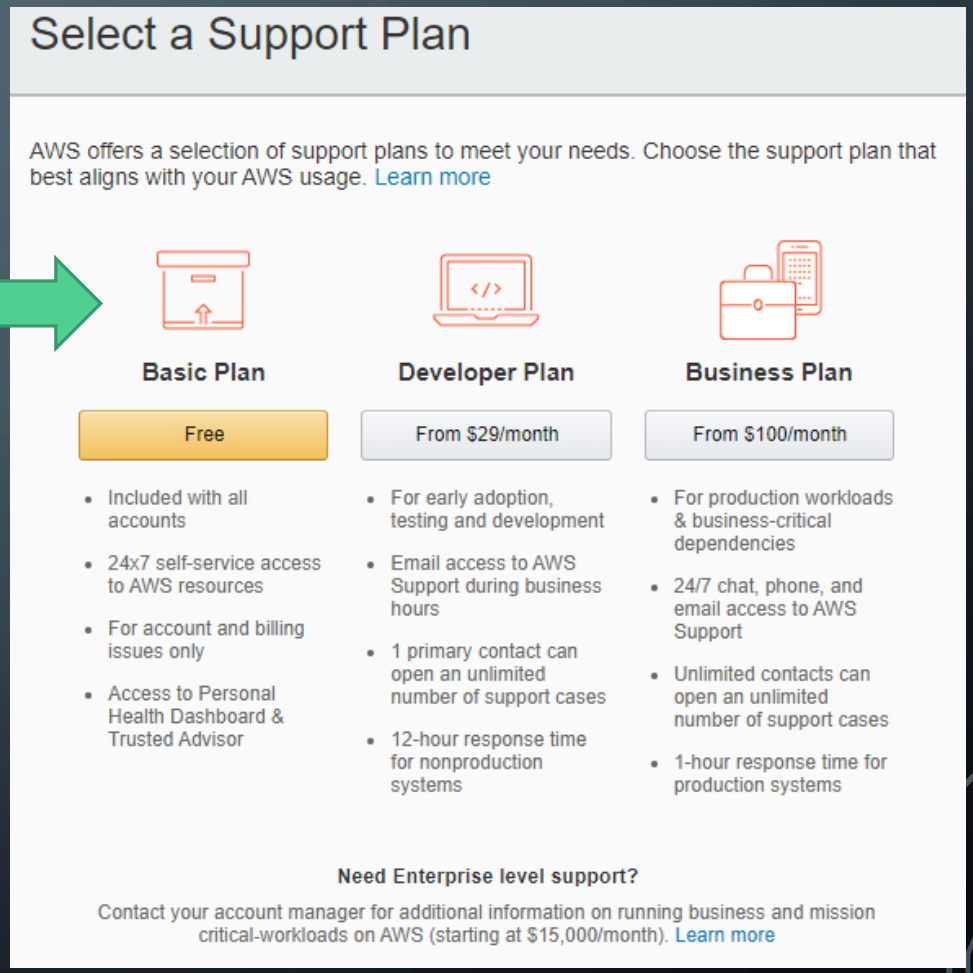
- You'll need to fill out some basic information – name, address, and credit card information

HOW MUCH WILL RDS COST?

- Luckily for us, Amazon offers 12 months of free access to the RDS service, with the following limitations:
  - Db.t2.micro instance size only (smallest instance)
  - Up to 750 hours per month
  - Maximum 20GB storage
- Amazon won't charge your credit card, as long as you stay within these bounds

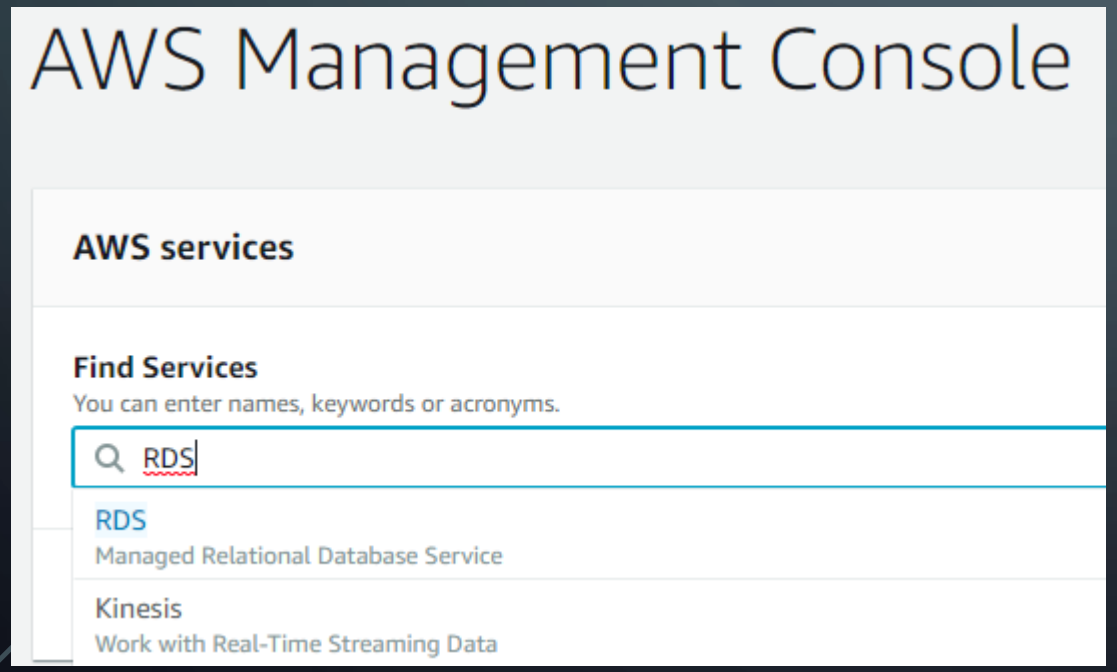
AVOIDING ADDITIONAL COSTS ON RDS

- When you're creating your account, make sure you choose the Basic Plan
- Note that the developer and business plans provide more *management*, for a price



NAVIGATING IN AMAZON AWS/RDS

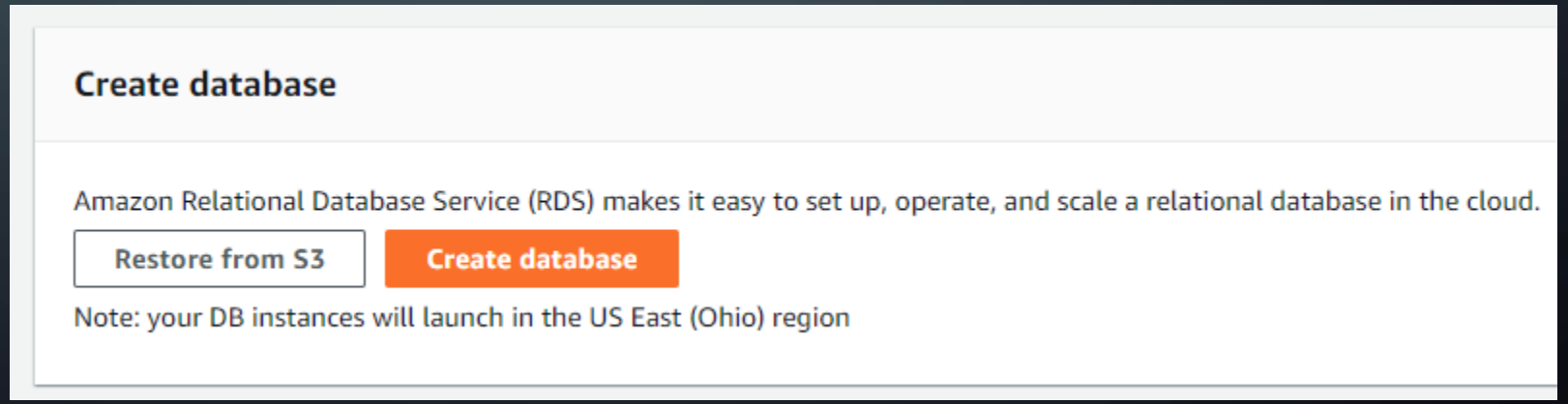
- Amazon uses a combined web application for it's computing services (EC2, AWS, RDS, etc)



- Just search for RDS in the "Find Services" bar, or use the dropdown "Services" menu at the top and navigate to Databases > RDS

CREATING YOUR RDS DATABASE

- Once you've navigated into the RDS dashboard, you should see a button that says "Create database"
- Clicking it will take you to the database creation wizard



AVAILABLE RDS DATABASES

- One of the first options you'll see will be the choice of database engine
- RDS supports the following engines:
  - Aurora
  - MySQL/MariaDB
  - PostgreSQL
  - Oracle
  - MSSQL
- For our first database, pick MySQL as your database type

ADVANCED ENGINE OPTIONS

- Note that you can choose which version of your database engine that you want to use (default selection is fine)
- For open-source/free engines like MySQL, MariaDB, and PostgreSQL, all versions are free of charge
- For Oracle and MSSQL, choice of edition will impact price of the database instance (cost of licensing the software temporarily)

TEMPLATES

- Amazon RDS offers some templates to help shape your instance – Production, Dev, and Free tier – make sure you pick the free tier, otherwise your machine may be over-provisioned and cost you $$$

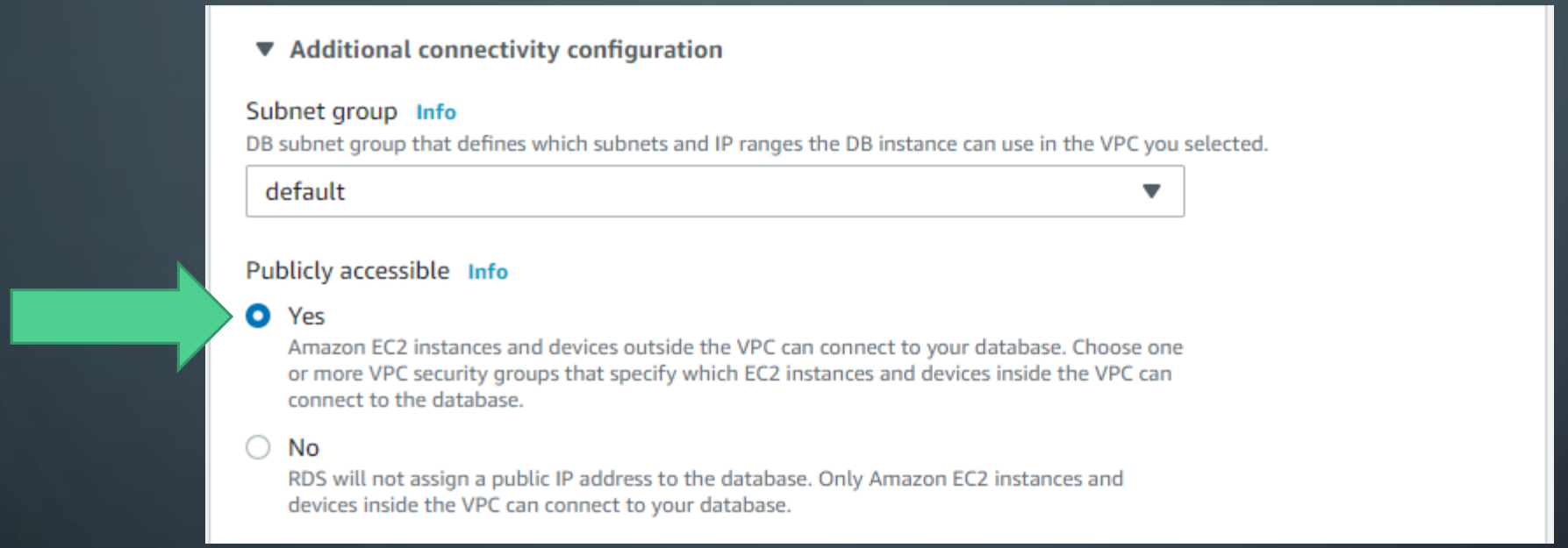
INSTANCE IDENTIFIER AND CREDENTIALS

- The *instance identifier* is the unique name for this particular database instance. Make sure you use descriptive names so you can keep track of your database instances
- The master username and password will be used to log into your machine via SQL – make sure you choose something secure that you won't forget

DB INSTANCE SIZES

- Since we're using the free tier, we're stuck using the default db.t2.micro instance size
  - 1 virtual CPU, 1 GB of RAM
- Amazon offers three types of instances: Standard, Memory-Optimized, and Burstable
  - The t2.micro instance is a Burstable type, meaning it can actually scale up the available CPU to meet burst loads

CONNECTIVITY



- Under the “Connectivity” section, expand to show all the options, and make sure “Publicly accessible” is checked:

FINISHING IT UP

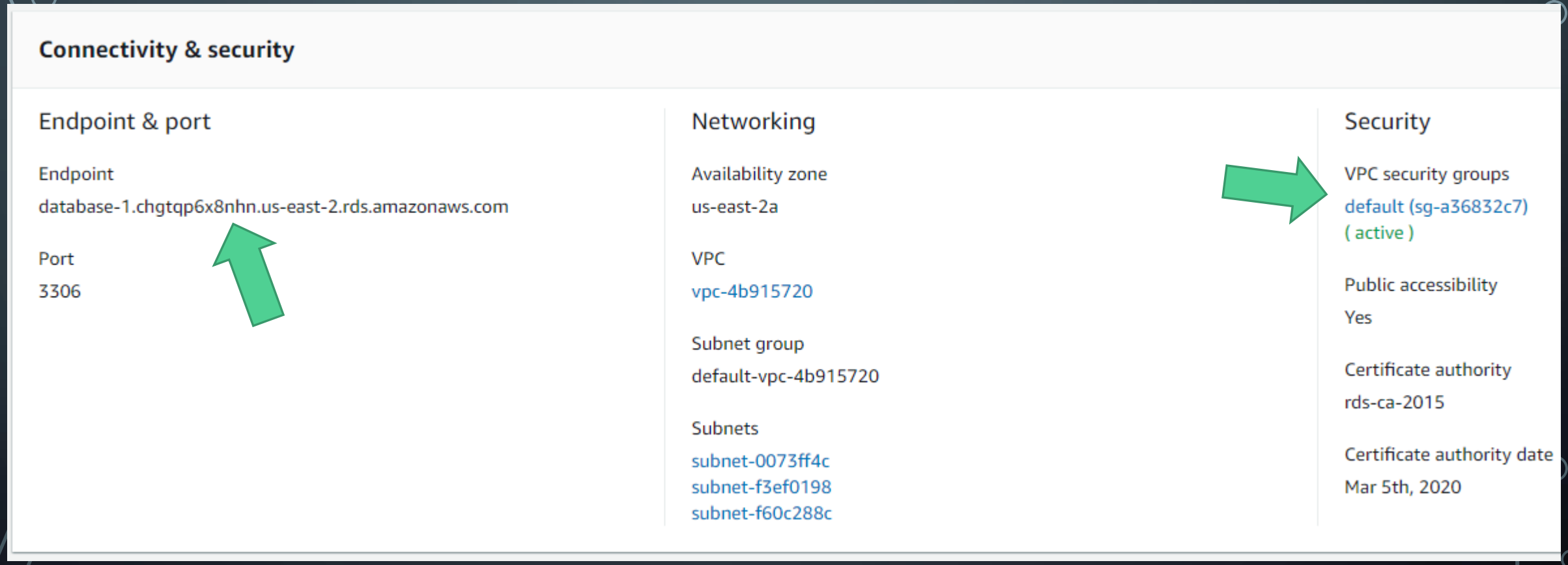
- There are some other options available, but they aren't strictly necessary
  - You may want to enable Enhanced Monitoring, if you want to see what your database is doing on a very granular timescale
- Hit the "Create database" button at the bottom. If it says you're missing information, then fill out the required sections and try again
- Congratulations – you've created an RDS Database Instances

SECURITY AND SERVER CONFIGURATION

AMAZON WEB SERVICE SECURITY

- Even though we made our database instance Publicly accessible, we won't be able to communicate with our machine yet
- We will first need to manage some firewall/security settings so we can actually create a remote connection

WHAT YOU SHOULD SEE NOW



- Note the endpoint, and VPC security groups

AMAZON SECURITY GROUPS

- Security groups are ways of managing routing and firewall policies for your instances
- Each instance will have a security group (or the default security group) associated with it, which defines what inbound and outbound traffic is allowed from the server
- Clicking on the security group link should take you to the security group dashboard

INBOUND AND OUTBOUND TRAFFIC

- The security group manages what traffic, on which ports, using which protocol, will be accepted by our instances
- Despite our machine being publicly accessible, it's security group by default will only allow inbound traffic from other instances within the security group
- You can adjust the inbound and outbound rules to only have ports open for the specific type of traffic you're handling (TCP/IP on port 3306 to allow MySQL, for example)

ENABLING REMOTE CONNECTIONS

- For easiness, we're just going to open up our instance to all traffic on all ports
- In case you were wondering, this is generally a *terrible* idea and you shouldn't do this on a production server, ever
- For our lab/dev servers, this will be fine, and make it a bit easier on us configuration-wise

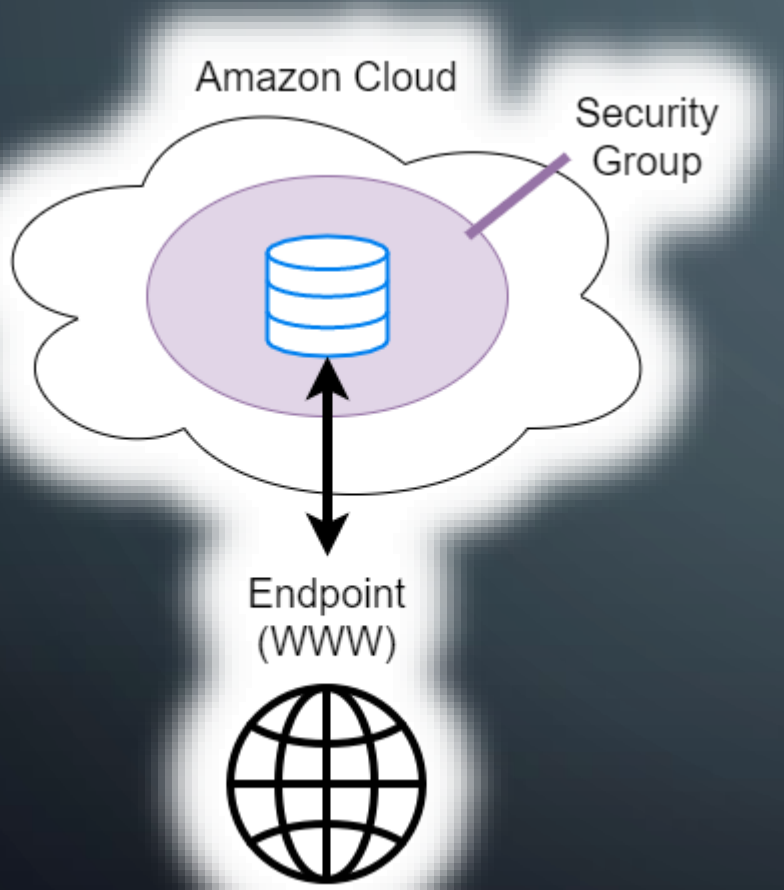
ENABLING REMOTE CONNECTIONS

- Bottom rule is default, top rule is new
- 0.0.0.0/0 will allow traffic from all source IP addresses

USING SECURITY GROUPS EFFECTIVELY

- To actually get more benefit from security groups, we'd need to use other Amazon products like AWS
- Idea: Have your RDS and AWS instances in the same security group, allow traffic between them so they can communicate, but disallow outside (risky) traffic

WHAT WE HAVE NOW



- A database instance (MySQL), in the cloud, encapsulated within a security group
- The database instance has an endpoint, which is the access point between the instance and the wider network (WWW)

CONNECTING TO YOUR RDS INSTANCE

ENDPOINTS

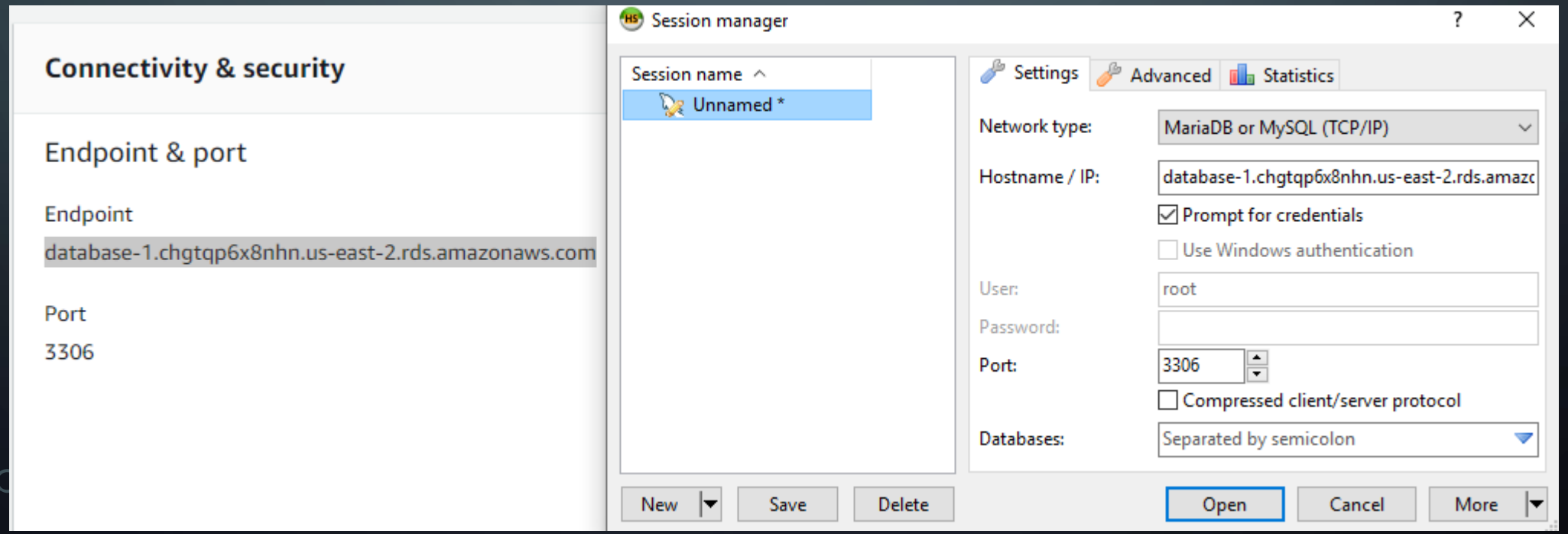
- Your instance will have an endpoint, available via the main dashboard page for your database instance
- This long URL serves as the remote address for the instance, rather than using an IP address
- For example, my database instance has the endpoint:

  database-1.chgtqp6x8nhn.us-east-2.rds.amazonaws.com

- Note that the instance name, region, and service are part of the endpoint URL

CONNECTING USING SQL SOFTWARE

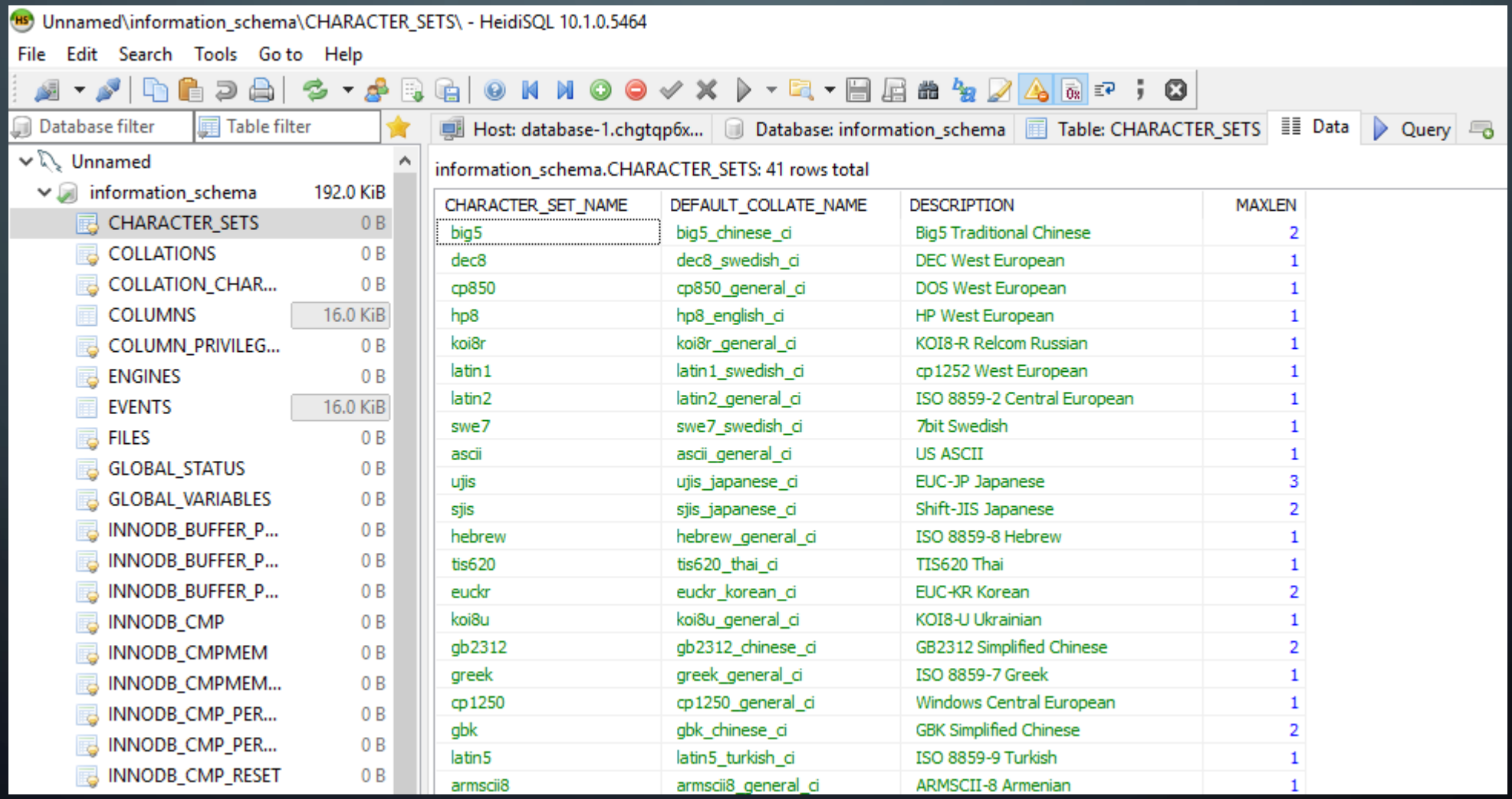
- You can connect to your RDS instance using standard SQL clients like SQuirreL or HeidiSQL



CONNECTION INFORMATION

- Besides the endpoint, you'll also need the username and password that you set when you were configuring your database server
- Note that you can use TCP/IP via Port 3306, which is the standard for MySQL and MySQL variants like MariaDB
- While you can also connect via the MySQL command-line utility (Mac, Linux), I don't recommend it – a GUI will probably make things a lot easier

SHILLING FOR HEIDISQL



CHARACTER_SET_NAME	DEFAULT_COLLATE_NAME	DESCRIPTION	MAXLEN
big5	big5_chinese_ci	Big5 Traditional Chinese	2
dec8	dec8_swedish_ci	DEC West European	1
cp850	cp850_general_ci	DOS West European	1
hp8	hp8_english_ci	HP West European	1
koi8r	koi8r_general_ci	KOI8-R Relcom Russian	1
latin1	latin1_swedish_ci	cp1252 West European	1
latin2	latin2_general_ci	ISO 8859-2 Central European	1
swe7	swe7_swedish_ci	7bit Swedish	1
ascii	ascii_general_ci	US ASCII	1
ujis	ujis_japanese_ci	EUC-JP Japanese	3
sjis	sjis_japanese_ci	Shift-JIS Japanese	2
hebrew	hebrew_general_ci	ISO 8859-8 Hebrew	1
tis620	tis620_thai_ci	TIS620 Thai	1
euckr	euckr_korean_ci	EUC-KR Korean	2
koi8u	koi8u_general_ci	KOI8-U Ukrainian	1
gb2312	gb2312_chinese_ci	GB2312 Simplified Chinese	2
greek	greek_general_ci	ISO 8859-7 Greek	1
cp1250	cp1250_general_ci	Windows Central European	1
gbk	gbk_chinese_ci	GBK Simplified Chinese	2
latin5	latin5_turkish_ci	ISO 8859-9 Turkish	1
armscii8	armscii8_general_ci	ARMSCII-8 Armenian	1

CONNECTING FROM YOUR APPLICATION

- You can connect to your RDS server from an application just like you would any other MySQL or SQL database
- You'll need a MySQL driver for your application, such as the SQL Connector for Python
- You'll still need to pass the endpoint, username, and password as you did previously

PYTHON EXAMPLE

- In Python, the mysql-connector driver is the usual ‘gold standard’ for connecting to a MySQL database
- Can be installed using the Pip utility with one easy line:

```
pip install mysql-connector-python
```

- Once installed, we’re ready to try out our connection

PYTHON EXAMPLE

- We create a connection with the endpoint, username, and password

```
1  import mysql.connector
2
3  mydb = mysql.connector.connect(
4    host="database-1.chgtqp6x8nhn.us-east-2.rds.amazonaws.com",
5    user="admin",
6    passwd="****************"
7  )
8
9  mycursor = mydb.cursor()
10
11 mycursor.execute("CREATE DATABASE mynewdatabase")
12
13 mydb.commit()
14
```

- Then we can run some SQL: In this case, creating a new database. Don't forget to commit() your changes!

WORKING WITH THE RDS INSTANCE

- Once you've made a connection to your RDS instance, you can effectively treat it as you would any other database
- You can perform updates, create new databases, create user roles, and perform any other MySQL function typically available from the MySQL prompt
- Try connecting both ways – use one to verify the other is working correctly

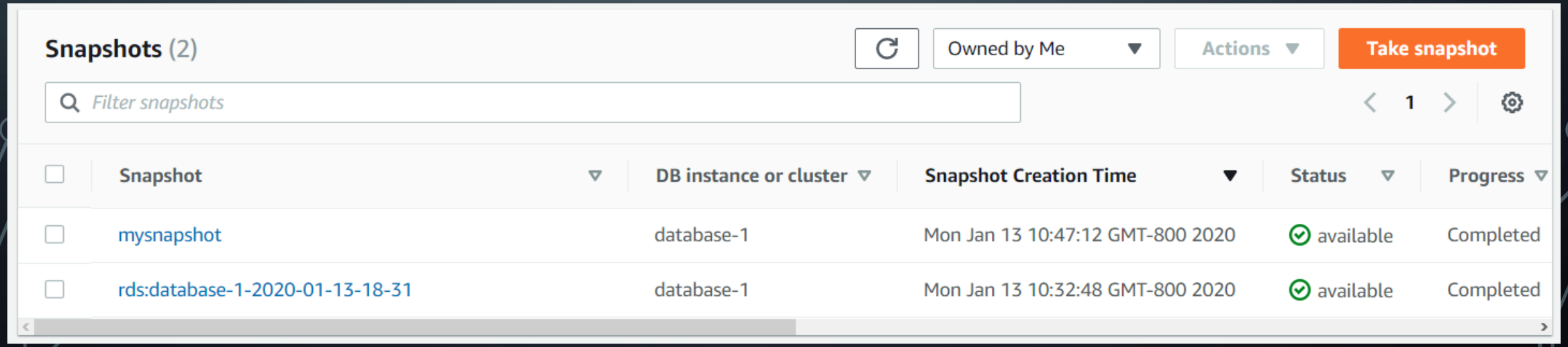
RDS SNAPSHOTS & BACKUPS

DATABASE SNAPSHOTS

- Amazon RDS allows us to quickly create “snapshots” of the current database state
- The RDS Free-tier access gives you up to 20GB a month of snapshot backups
- Good not just for backups, but for cloning databases across multiple servers as well

DATABASE SNAPSHOTS

- You can see your available snapshots in the ‘Snapshots’ sub-menu in the RDS service
- Amazon will also automatically create snapshots at creation time



CREATING A SNAPSHOT

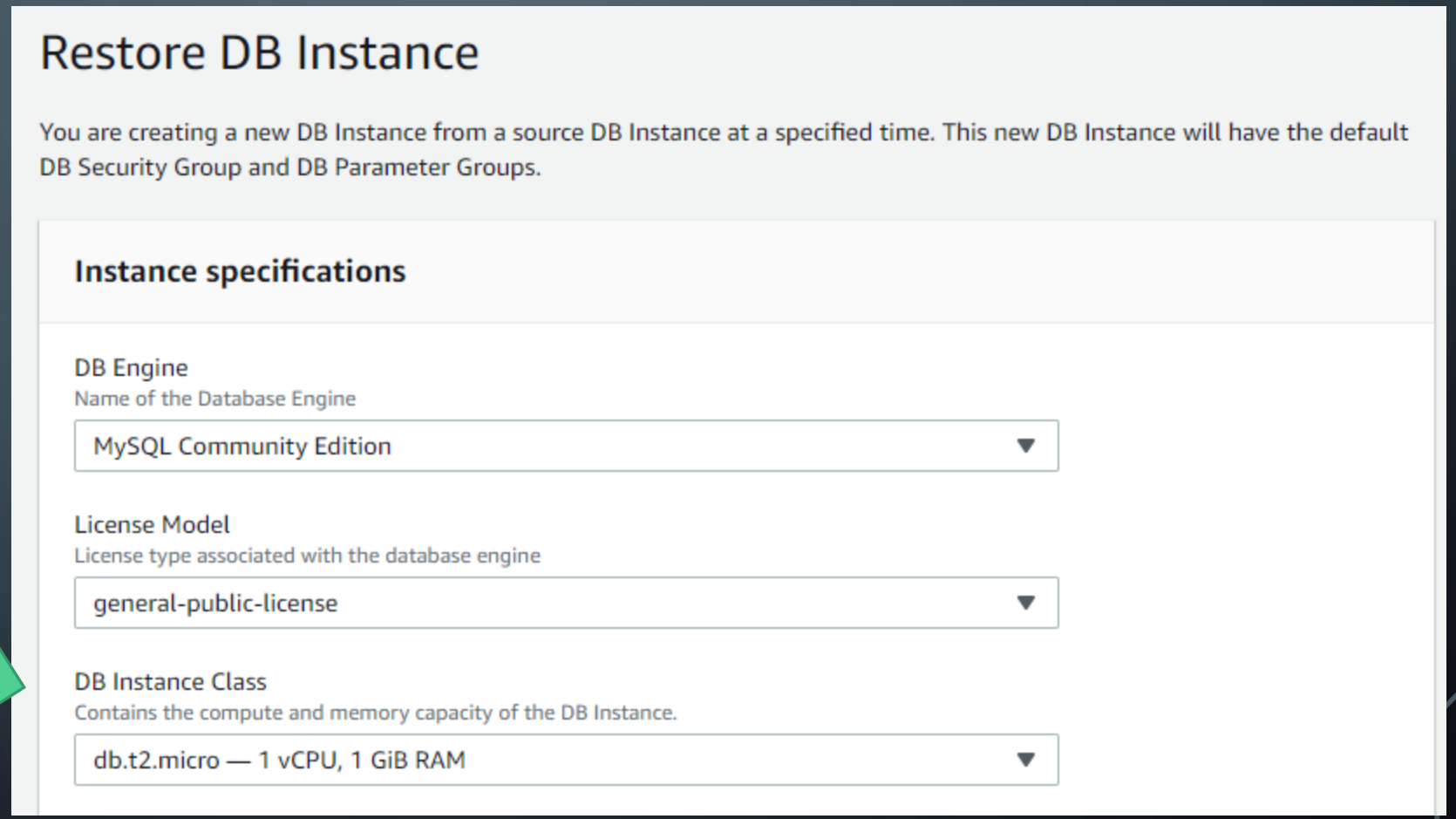
- Creating a snapshot is straightforward – simply click the “Take Snapshot” button
- It will ask you to select the instance you are trying to take a snapshot of, and a name for the snapshot
- It may take a couple minutes, but you should see your new snapshot in the snapshots menu when you’re done

RESTORING A SNAPSHOT

- If you select a snapshot, you can then select “Restore” from the drop-down actions menu
- This will take you into a new instance creation menu – your snapshot will be restored as a whole new, standalone database instance
- Be careful – it will try to default to a much larger (non-free!) machine when it creates the restored version!

RESTORING A SNAPSHOT

- Make sure you change the instance class!



AUTOMATED BACKUPS

- By default, Amazon will take automatic snapshots on a daily basis, and hold onto them for a period of 7 days
- You can configure to hold them for longer, or change when snapshots occur (say, midnight instead of noon)
- Keep in mind, holding onto snapshots longer will mean more snapshot storage space = $

SCHEDULE

1. *Amazon RDS*
    1. *Setting up Amazon RDS*
    2. *Security and Server Configuration*
    3. *Connecting to your Database*
    4. *Snapshots and Backups*

SO LONG, FOLKS!