

Lab 9: AWS/VSCode Setup

GW CS 2541W: Database Systems and Team Projects - 2024

Prof. Gabe Parmer, Sameen Ahmad, Kate Halushka, and Dania Abdalla



AWS RDS Setup

Create a MySQL Database Server in the Cloud



Log into AWS Academy and click **Modules** → **AWS Academy Learner Lab**

▼ AWS Academy Learner Lab



Launch AWS Academy Learner Lab

▼ AWS Academy Learner Lab Resources



Demo - How to Access Learner Lab



Demo - General Troubleshooting Tips



Demo - How to Launch Services through AWS Console



Learner Lab Activity - CodeWhisperer

Click **Start Lab**, wait, then click **AWS** once green

The screenshot displays the AWS Academy Instructure web application. The browser address bar shows the URL: <https://awsacademy.instructure.com/courses/16958/modules/items/1398292>. The page title is "Learner Lab - Foundational Services".

Key elements highlighted with a pink oval:

- The **AWS** logo in the top navigation bar.
- The **Start Lab** button in the top right corner.

The main content area features a large blue "V" logo with an orange circular arrow to its right. On the right side, there is a sidebar with the following content:

- Region: EN-US
- Learner Lab - Foundational Level**
- Environment Overview
- Environment Navigation
- [Access the AWS Management Console](#)
- [Region restriction](#)
- [Service usage and other restrictions](#)
- [Using the terminal in the browser](#)
- [Running AWS CLI commands](#)
- [Using the AWS SDK for Python](#)
- [Preserving your budget](#)
- [Accessing EC2 Instance\(s\)](#)
- [SSH Access to EC2 Instances](#)
- [SSH access from Windows](#)
- [SSH access from a Mac](#)
- Environment Overview**
- This Learner Lab provides a sandbox

At the bottom of the page, there is a purple banner with the following text:

You are currently logged into Student View

Resetting the test student will clear all history for this student, allowing you to view the course as a brand new student.

Buttons: **Reset Student** and **Leave Student View**

In AWS Management Console - Type "RDS" into the search bar, then click **Databases**

The screenshot shows the AWS Management Console search results for 'rds'. The search bar at the top left contains the text 'rds' and is circled in pink. Below the search bar, the search results are displayed in a dark-themed sidebar. The 'Services' section is highlighted with a pink circle, and the 'Databases' link is circled in pink. The 'Features' section is also visible, showing various AWS services and their features. The main content area on the right displays a 'Welcome to AWS' message and several links for getting started, training, and what's new with AWS. The footer of the console includes a 'Feedback' link, the language 'English (US)', and a 'Trusted Advisor' link.

Search results for 'rds'

Services [See all 8 results](#)

- RDS** [Star](#)
Relational Database Service
Top features: [Dashboard](#) [Databases](#) [Query Editor](#) [Performance Insights](#) [Snapshots](#)
- Kinesis** [Star](#)
Work with Real-Time Streaming Data
- Amazon OpenSearch Service** [Star](#)
Run and Scale OpenSearch and Elasticsearch Clusters (successor to Amazon Elasticse...

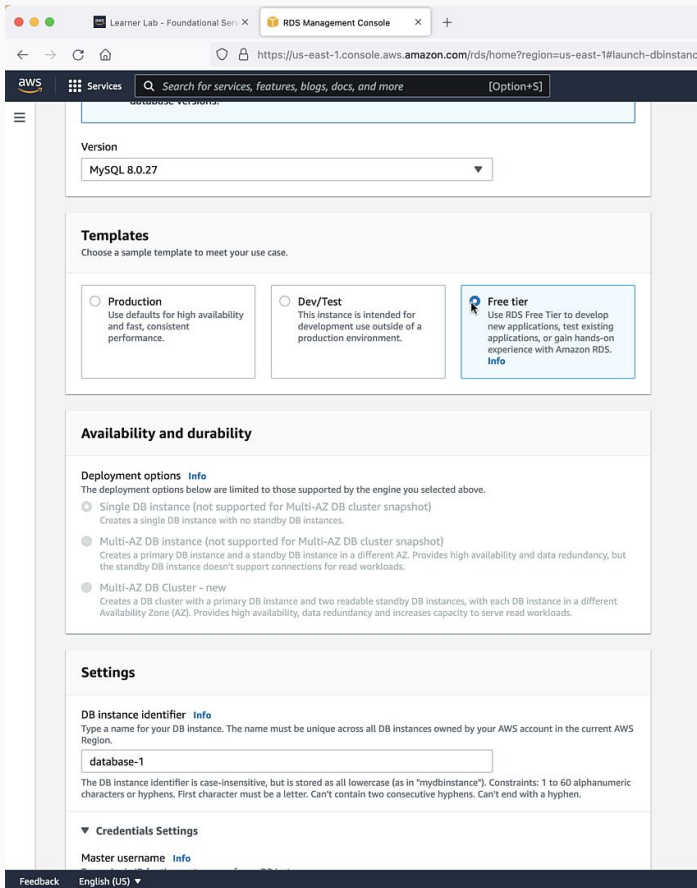
Features [See all 25 results](#)

- Dashboards**
 - CloudWatch feature**
- Security standards**
 - Security Hub feature**
- Redis**
 - ElastiCache feature**
- Users**
 - IAM feature**

Blogs [See all 1,380 results](#)

© 2022, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

Click Create Database. Use these options



Method: Standard Create

Engine Type: MySQL

Template: Free Tier

Instance Identifier: something like REGS23-Wood

Master Password: something secure that you write down!

Storage Autoscaling: Disable

Public Access: YES

Additional Configuration

Initial DB name: university

Disable Automatic Backups

Disable Auto minor version upgrade

Then click **Create Database**

In Database Status window, click the Security Group

The screenshot shows the Amazon RDS console interface. The main content area displays the status of the database instance 'instructor-1'. The 'Connectivity & security' tab is active, showing details for the security group 'default-sg-03846ac637a32371f', which is circled in red. The console includes a navigation sidebar on the left, a top navigation bar with the AWS logo and search bar, and a bottom status bar with copyright information.

Summary

DB identifier instructor-1	CPU 0.00%	Status Creating	Class db.t2.micro
Role Instance	Current activity 0 Connections	Engine MySQL Community	Region & AZ -

Connectivity & security

Endpoint & port	Networking	Security groups
Endpoint -	Availability Zone -	VPC security groups default-sg-03846ac637a32371f Active
Port -	VPC vpc-0b699f540a8be85f4	Public Yes
	Subnet group default-vpc-0b699f540a8be85f4	Certificate authority rds-ca-2019
	Subnets subnet-0da09d5a8f868f76c subnet-05d9291f53956f38b subnet-03954f8ea006d729e subnet-08c87f4b16adaf686 subnet-0a8720349a826bbab subnet-0d25c733e03dee097	Certificate authority date August 22, 2024, 01:08 (UTC+1:08)

Add a new Inbound Rule - Port 3306 - Source IP 0.0.0.0/0

The screenshot shows the AWS Management Console interface for editing inbound rules. The breadcrumb navigation is **EC2 > Security Groups > sg-03846ac637a32371f - default > Edit inbound rules**. The page title is **Edit inbound rules** with an **Info** link. A sub-header states: **Inbound rules control the incoming traffic that's allowed to reach the instance.**

The **Inbound rules** section contains a table with the following columns: **Security group rule ID**, **Type**, **Protocol**, **Port range**, **Source**, and **Description - optional**. There are two existing rules and an **Add rule** button.

Security group rule ID	Type	Protocol	Port range	Source	Description - optional	
sg-089996e2a73bcdbe6	All traffic	All	All	Custom		Delete
-	Custom TCP	TCP	3306	Custom		Delete

The **Add rule** process is shown with a search dropdown for the **Source** field. The search input contains **sg-03846ac637a32371f**. A dropdown menu is open, showing **CIDR blocks** and **Security Groups**. Under **CIDR blocks**, the option **0.0.0.0/0** is selected. At the bottom right of the console, there are buttons for **Cancel**, **Preview changes**, and **Save rules**.

Check your DB creation status. When ready, copy host info

The screenshot shows the Amazon RDS console for an instance named 'instructor-1'. The instance is in an 'Available' status. The 'Connectivity & security' tab is selected, showing the endpoint 'instructor-1.ck8ualavedvt.us-east-1.rds.amazonaws.com' circled in red. Other details include the instance role, CPU usage (0.00%), and various networking and security configurations.

Amazon RDS ×

RDS > Databases > instructor-1

instructor-1

[Modify](#) [Actions](#) ▾

Summary

DB identifier instructor-1	CPU 0.00%	Status Available	Class db.t2.micro
Role Instance	Current activity 0 Connections	Engine MySQL Community	Region & AZ us-east-1c

[Connectivity & security](#) | [Monitoring](#) | [Logs & events](#) | [Configuration](#) | [Maintenance & backups](#) | [Tags](#)

Connectivity & security

Endpoint Endpoint instructor-1.ck8ualavedvt.us-east-1.rds.amazonaws.com 3306	Networking Availability Zone us-east-1c VPC vpc-0b699f540a8be85f4 Subnet group default-vpc-0b699f540a8be85f4 Subnets subnet-0da09d5a8f868f76c subnet-05d9291f53956f38b subnet-03954f8ea006d729e subnet-08c87fab15ada6f86 subnet-0a8720349a826bbab subnet-0d25c733e03dee097	Security VPC security groups default (sg-03846ac637a32371f) Active Public accessibility Yes Certificate authority rds-ca-2019 Certificate authority date August 22, 2024, 01:08 (UTC-1:08)
--	--	--

Security group rules (3)

Filter by security group rules

Feedback English (US) © 2022, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Test from AWS Academy Lab Console (other tab)

The screenshot shows the AWS Academy Lab Console interface. The browser tab is titled "Learner Lab - Foundational Services" and the URL is "https://awsacademy.instructure.com/courses/16958/modules/items/1398292". The left sidebar contains navigation options: Account, Dashboard, Courses, Calendar, Inbox, History, and Help. The main content area shows a terminal window with the following text:

```
ddd_v1_w_9M5_1123374@runweb51466:~$ ls
ddd_v1_w_9M5_1123374@runweb51466:~$ mysql -h instructor-1.ck8ualavedvt.us-east-1.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 16
Server version: 8.0.27 Source distribution

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| university |
+-----+
5 rows in set (0.07 sec)

mysql> use university;
Database changed
mysql> show tables;
Empty set (0.07 sec)

mysql>
```

Commands:
**mysql -h [endpoint] -u
admin -p**

Enter password

**mysql> show
databases;**

mysql> use university;

mysql> show tables;

MySQL vs SQLite

Code / System Differences



SQLite vs MySQL

SQLite: embedded database

- Simple, single file based DB integrated in your software
- Great for mobile apps, or single user applications

MySQL: full featured database server

- Independent server application
- Can be deployed across a cluster of servers
- Supports larger scale and stronger reliability guarantees

SQL Code Differences

```
create.sql -- flask-sample-mysql

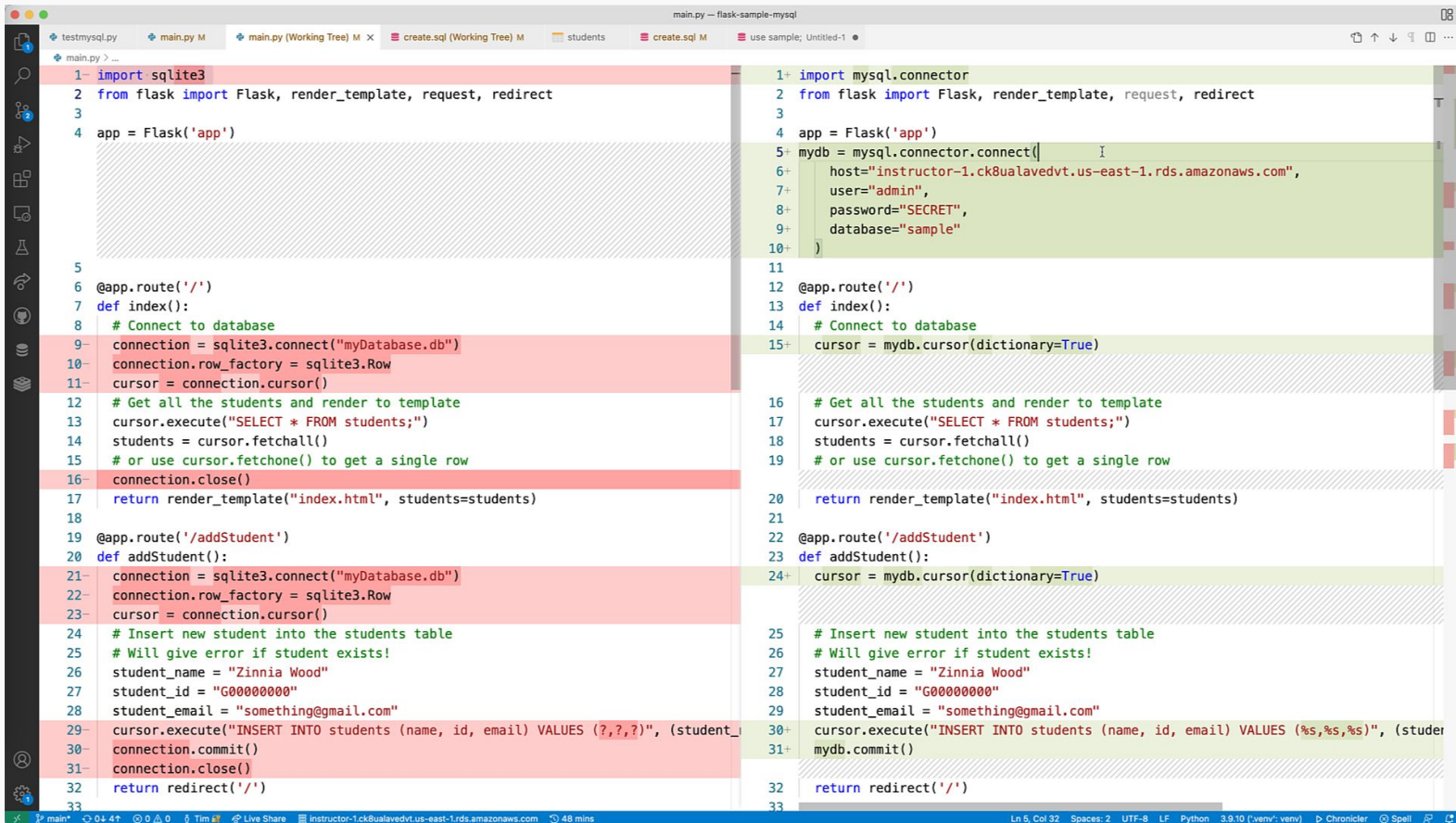
testmysql.py main.py M main.py (Working Tree) M create.sql (Working Tree) M x students create.sql M use sample; Untitled-1 •

create.sql > ...
1- PRAGMA foreign_keys=off;
2
3 DROP TABLE IF EXISTS students;
4 CREATE TABLE students (
5   id      varchar(32) not null PRIMARY KEY,
6   name    varchar(50) not null,
7   email   varchar(50) not null
8 );
9-
10- PRAGMA foreign_keys=on;
11
12 INSERT INTO students VALUES ('G12345678', 'Jett Jacobs', 'jacobsemail@fakeemail.com');
13 INSERT INTO students VALUES ('G22489071', 'Alex Coleman', 'alexcoleman@fakeemail.com');
14 INSERT INTO students VALUES ('G82915273', 'Ethan Baron', 'ethan@fakeemail.com');
15 INSERT INTO students VALUES ('G22004676', 'Cat Meadows', 'cat@fakeemail.com');
16
17 UPDATE students SET name = 'Catherine Meadows' WHERE id = 'G22004676';

1- CREATE DATABASE sample
2-   DEFAULT CHARACTER SET = 'utf8mb4';
3- use sample;
4-
5- SET FOREIGN_KEY_CHECKS=0;
6
7 DROP TABLE IF EXISTS students;
8 CREATE TABLE students (
9   id      varchar(32) not null PRIMARY KEY,
10  name    varchar(50) not null,
11  email   varchar(50) not null
12 );
13- SET FOREIGN_KEY_CHECKS=1;
14
15 INSERT INTO students VALUES ('G12345678', 'Jett Jacobs', 'jacobsemail@fakeemail.com');
16 INSERT INTO students VALUES ('G22489071', 'Alex Coleman', 'alexcoleman@fakeemail.com');
17 INSERT INTO students VALUES ('G82915273', 'Ethan Baron', 'ethan@fakeemail.com');
18 INSERT INTO students VALUES ('G22004676', 'Cat Meadows', 'cat@fakeemail.com');
19
20 UPDATE students SET name = 'Catherine Meadows' WHERE id = 'G22004676';

Ln 9, Col 24 (11 selected) Spaces: 2 UTF-8 LF SQL Chronieler Spell
```

Python Code Differences



```
main.py - flask-sample-mysql
testmysql.py main.py M main.py (Working Tree) M x create.sql (Working Tree) M students create.sql M use sample; Untitled-1
main.py > ...
1- import sqlite3
2 from flask import Flask, render_template, request, redirect
3
4 app = Flask('app')
5
6 @app.route('/')
7 def index():
8     # Connect to database
9     connection = sqlite3.connect("myDatabase.db")
10    connection.row_factory = sqlite3.Row
11    cursor = connection.cursor()
12    # Get all the students and render to template
13    cursor.execute("SELECT * FROM students;")
14    students = cursor.fetchall()
15    # or use cursor.fetchone() to get a single row
16    connection.close()
17    return render_template("index.html", students=students)
18
19 @app.route('/addStudent')
20 def addStudent():
21    connection = sqlite3.connect("myDatabase.db")
22    connection.row_factory = sqlite3.Row
23    cursor = connection.cursor()
24    # Insert new student into the students table
25    # Will give error if student exists!
26    student_name = "Zinnia Wood"
27    student_id = "G00000000"
28    student_email = "something@gmail.com"
29    cursor.execute("INSERT INTO students (name, id, email) VALUES (?, ?, ?)", (student_name, student_id, student_email))
30    connection.commit()
31    connection.close()
32    return redirect('/')
33
1+ import mysql.connector
2 from flask import Flask, render_template, request, redirect
3
4 app = Flask('app')
5+ mydb = mysql.connector.connect({
6+     host="instructor-1.ck8ualavedvt.us-east-1.rds.amazonaws.com",
7+     user="admin",
8+     password="SECRET",
9+     database="sample"
10+ })
11
12 @app.route('/')
13 def index():
14     # Connect to database
15+    cursor = mydb.cursor(dictionary=True)
16
17 # Get all the students and render to template
18 cursor.execute("SELECT * FROM students;")
19 students = cursor.fetchall()
20 # or use cursor.fetchone() to get a single row
21
22 return render_template("index.html", students=students)
23
24 @app.route('/addStudent')
25 def addStudent():
26+    cursor = mydb.cursor(dictionary=True)
27
28 # Insert new student into the students table
29 # Will give error if student exists!
30 student_name = "Zinnia Wood"
31 student_id = "G00000000"
32 student_email = "something@gmail.com"
33+    cursor.execute("INSERT INTO students (name, id, email) VALUES (%s,%s,%s)", (student_name, student_id, student_email))
34+    mydb.commit()
35
36 return redirect('/')
Ln 5, Col 32 Spaces: 2 UTF-8 LF Python 3.9.10 ('venv': venv) > Chronidler Spell
```

MySQL vs SQLite Summary

MySQL allows you to run a database on a separate server than your application

—This helps with performance, scalability, and reliability

SQL query syntax is mostly the same

—Remember to "**use**" a database

Python code is mostly the same

—Create one MySQL connection in global scope, then create cursors for each route

—use `%s` to fill in query parameters instead of ?

Python Virtual Environments

And installing Flask Library



Virtual Environment

Create environment

```
python3 -m venv .venv
```

Tell VS Code to use the environment when it shows a popup!

Activate environment

```
source .venv/bin/activate (mac)
```

```
.venv\Lib\activate (windows)
```

(LATER, after installing all modules)

Save required package list

```
pip3 freeze > requirements.txt
```

Install from required package list (on another machine)

```
pip install -r requirements.txt
```

Use this to save a list of packages you have installed. Then add the file to your git repository

On another computer you could use this to load all the libraries needed by the repository

Check

Will say "(.venv)"
above every command
prompt in terminal if
working correctly



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS zsh + - [ ] [ ] ... ^ x
...
• sameenahmad@Sameens-MacBook-Pro ~ % python3 -m venv .venv
• sameenahmad@Sameens-MacBook-Pro ~ % source .venv/bin/activate
○ (.venv) sameenahmad@Sameens-MacBook-Pro ~ % █
```

The screenshot shows a terminal window with a sidebar on the left containing icons for database, layers, sharing, user, and settings. The terminal title bar includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS, along with window controls and a 'zsh' label. The terminal content shows three lines of output: a blue dot followed by the command 'python3 -m venv .venv', another blue dot followed by 'source .venv/bin/activate', and a grey circle followed by the prompt change to '(.venv) sameenahmad@Sameens-MacBook-Pro ~ %'. A red arrow points to this prompt change. The bottom status bar shows window management icons, 'Live Share', and environment details: 'Ln 1, Col 1 Spaces: 4 UTF-8 LF Python 3.11.6 64-bit'.

Flask Module Installation

Inside your virtual environment terminal:

```
pip install flask
```

MySQL Local Setup

VS Code Extension and Python Library



MySQL Extension for VS Code

mysql

- MySQL** 838K 3.5
MySQL management tool
Jun Han [Install](#)
- MySQL** 203ms
Database manager for MySQL/M...
cweijan [Install](#)
- MySQL Syntax** 335K 3
MySQL syntax highlighting supp...
Jake Bathman [Install](#)
- SQLTools MySQL/M...** 291K 5
SQLTools MySQL/MariaDB
Matheus Teixeira [Install](#)
- mysql-inline-decora...** 90K 3
Add color coding to inline MYSQ...
odubuc [Install](#)
- MySQL Statement S...** 36K 5
Easy mysql statement running wi...
Jared Black [Install](#)
- MySQL Autocomplete** 7K
MySQL Syntax Autocomplete for ...
nospinozacr [Install](#)
- ES7 JavaScript/Nod...** 23K 5
Simple extension for Node, javas...
abrahamwilliam007 [Install](#)

Get the one by cweijan!

MySQL v4.8.6
cweijan | 446,300 | ★★★★★ (138)
Database manager for MySQL/MariaDB, PostgreSQL, SQLite, Redis and ElasticSearch.

[Disable](#) [Uninstall](#) [Refresh](#) [Settings](#)

This extension is enabled globally.

[Details](#) [Feature Contributions](#) [Changelog](#) [Runtime Status](#)

Database Client for Visual Studio Code

vscod marketplace v4.8.6 installs 447.38K stars 1.7k rating 4.4/5 (139) license MIT

This project is a database client for VSCode, supports manager **MySQL/MariaDB**, **PostgreSQL**, **SQLite**, **Redis**, **ClickHouse**, **达梦**, and **ElasticSearch**, and works as an **SSH** client, boost your maximum productivity!

Project site: [vscod-database-client](#), [中文文档](#)

Database Client

Click DB icon in left menu, Add Database, fill in your RDS connection info

The screenshot shows the 'Connect Server' dialog in a database client. The 'Host' field is set to '127.0.0.1' and the 'Port' field is set to '3306'. The 'Username' field is set to 'root' and the 'Password' field is set to 'admin'. The 'Server Type' is set to 'MySQL'. The 'Database' field is set to 'mysql,information_schema'. The 'Connect Timeout' is set to '5000'. The 'Socket Path' is set to 'Unix Socket Path'. The 'Time Zone' is set to '+00:00'. The 'SSH Tunnel' is disabled. The 'Use SSL' is disabled. The 'Hide System Schema' is enabled. The 'Use Connection String' is disabled. The 'Connect' and 'Close' buttons are visible at the bottom.

Annotations in the image include a pink circle around the 'Add Database' icon in the left sidebar and another pink circle around the 'Host' and 'Port' fields in the 'Connect Server' dialog. The text 'RDS Endpoint' and 'admin' are overlaid on the 'Host' and 'Password' fields respectively.

MySQL Python Module

Inside your virtual environment terminal:

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

Now save the packages

Save required package list

```
pip3 freeze > requirements.txt
```

Install from required package list (on another machine)

```
pip install -r requirements.txt
```

Use this to save a list of packages you have installed. Then add the file to your git repository

On another computer you could use this to load all the libraries needed by the repository