

Databases, SQL, and Django

Scott Coughlin, Computational Specialist
Some slides adapted from Professor Michael
Coughlin and Matthew Graham
July 19th 2022

Databases, SQL, and Django

Topics Covered

1. Servers, Databases, Schema, Relations, OH MY!
2. Doing Things the Structured Query Language (SQL) Way
3. How to use docker-compose

Servers, Databases, Schema,
Relations, OH MY!

What is the server?

It is the physical (or virtual) machine where the database (and the application managing the database lives).

- “The database runs at/lives at
xxxx.ciera.northwestern.edu”

What is a database?

- A structured collection of data residing on a computer system (server) that can be easily accessed, managed and updated.
- Data is organized according to a database model.
- A Database Management System (DBMS) is a software package designed to store and manage databases.

Some Popular Database Management System

- Different DBMS:
 - Microsoft/Sybase
 - MySQL
 - Oracle
 - **PostgreSQL**
 - Redis, Hadoop

Why use a Database Management System?

- Provides concurrent access
- data scalability, expandability and flexibility
- Security through managing access to what databases, tables, and even types of queries a individual user can make.
- efficient memory management and indexing of the data
- Integrity constraints

How do we model our data? With tables (or relations)

- Data is organized as **relations (tables)**, **attributes (columns)** and **domains (type)**
- A **relation** is a table with columns (attributes) and rows (tuples)
- The **domain** is the set of values that the attributes are allowed to take
- Within the relation, each row is unique, the column order does not matter, and each row contains a single value for each of its columns

Relational Example

Relational Model

Activity Code	Activity Name
23	Patching
24	Overlay
25	Crack Sealing

Key = 24

Activity Code	Date	Route No.
24	01/12/01	I-95
24	02/08/01	I-66

Date	Activity Code	Route No.
01/12/01	24	I-95
01/15/01	23	I-495
02/08/01	24	I-66

What is the “Schema”

- The **schema** is a named collection of tables.
- A schema can contain many additional pieces of information relevant to a collection of tables including views, indexes, sequences, data types, operators, and functions.

Say What Now?

The screenshot displays the pgAdmin web interface in a browser window. The browser's address bar shows 'localhost:15432/browser/'. The pgAdmin interface includes a top navigation bar with 'File', 'Object', 'Tools', and 'Help' menus. A left sidebar contains 'Browser' and 'Servers' sections. The main content area features a 'Welcome' message, the pgAdmin logo, and a description: 'pgAdmin is an Open Source administration and management tool to answer the needs of developers, DBAs and system administrators'. Below this are 'Quick Links' and 'Getting Started' sections, including a link to 'PostgreSQL Documentation'. A modal dialog titled 'Register - Server' is open, showing the 'Connection' tab. The dialog contains fields for 'Host name/address' (postgres), 'Port' (5432), 'Maintenance database' (postgres), 'Username' (imdb), 'Kerberos authentication?' (disabled), 'Password' (masked), 'Save password?' (disabled), 'Role', and 'Service'. At the bottom of the dialog are buttons for 'Close', 'Reset', and 'Save'.

Browser | localhost:15432/browser/ | what is a schema postgres | admin@pgadmin.com (internal)

pgAdmin | File | Object | Tools | Help | Dashboard | Properties | SQL | Statistics | Dependencies | Dependents

Browser | Servers

Welcome

pgAdmin
Management Tools for PostgreSQL

Feature rich | Maximises PostgreSQL productivity

pgAdmin is an Open Source administration and management tool to answer the needs of developers, DBAs and system administrators.

Quick Links

Getting Started

PostgreSQL Documentation

Register - Server

General | Connection | SSL | SSH Tunnel | Advanced

Host name/address: postgres

Port: 5432

Maintenance database: postgres

Username: imdb

Kerberos authentication? ☐

Password:

Save password? ☐

Role:

Service:

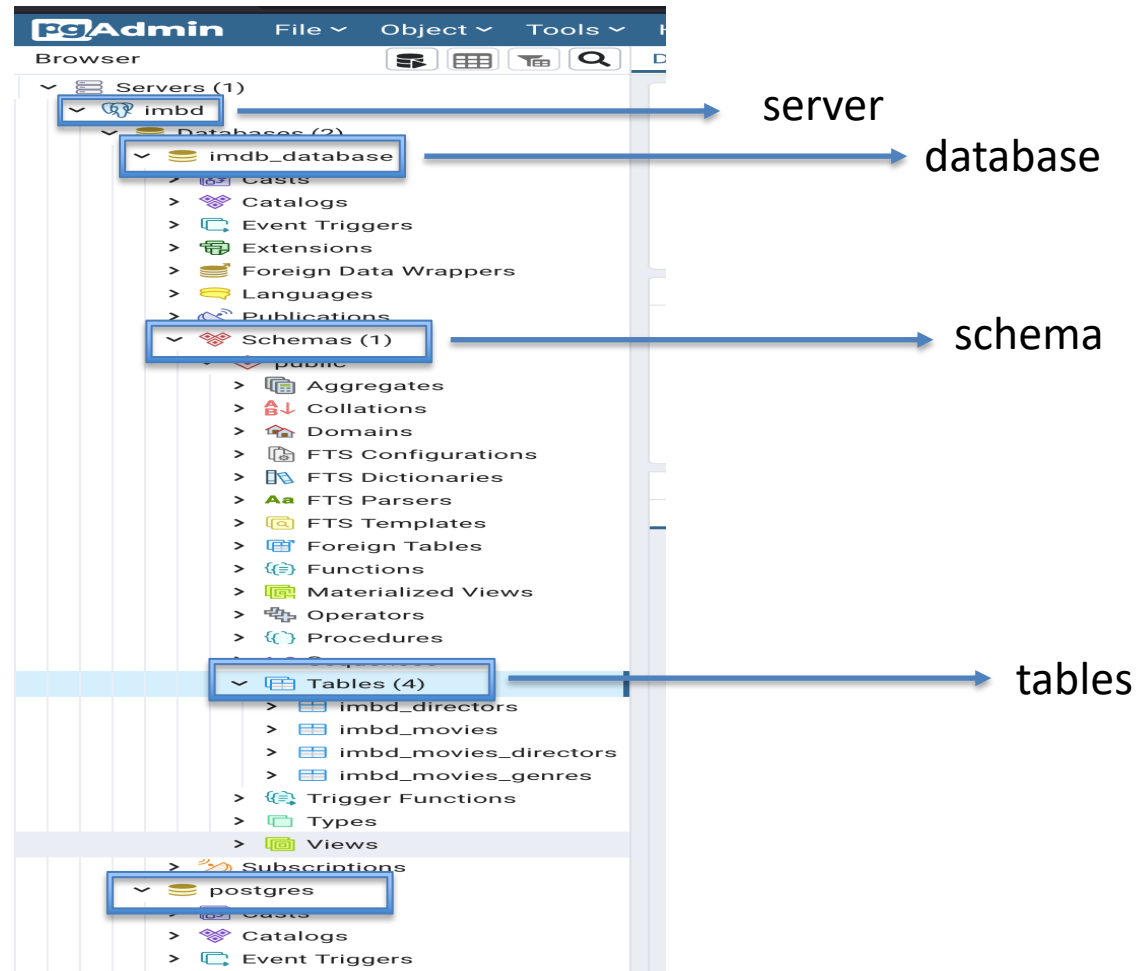
Close | Reset | Save

Configure pgAdmin

Planet PostgreSQL

Community Support

Say What Now?



Doing things the Structured Query Language (SQL) Way

Structured Query Language

- Different flavors:
 - **DBMS – Flavor of SQL**
 - Microsoft/Sybase - Transact-SQL
 - MySQL - MySQL
 - Oracle - PL/SQL
 - PostgreSQL - PL/pgSQL

SELECT

SELECT column1, column2 FROM table WHERE condition (LIMIT #ofrows) ORDER BY sort_expression [ASC | DESC];

```
SELECT name, constellation FROM star WHERE dec > 0 ORDER  
BY vmag;
```

```
SELECT * FROM star WHERE ra BETWEEN 0 AND 90;
```

```
SELECT DISTINCT constellation FROM star;
```

```
SELECT name FROM star LIMIT 5 ORDER BY vmag;
```

JOIN

- Inner join: combining related rows

```
SELECT * FROM imdb_movies as s INNER JOIN imdb_movies_genres as t ON  
s.movie_id = t.movie_id;
```

```
SELECT * FROM star s, stellarTypes t WHERE s.stellarType = t.id;
```

- Outer join: each row does not need a matching row

```
SELECT * from star s LEFT OUTER JOIN stellarTypes t ON s.stellarType = t.id;
```

```
SELECT * from star s RIGHT OUTER JOIN stellarTypes t ON s.stellarType = t.id;
```

```
SELECT * from star s FULL OUTER JOIN stellarTypes t ON s.stellarType = t.id;
```


Aggregate Functions

COUNT, AVG, MIN, MAX, SUM

```
SELECT COUNT(*) FROM star;
```

```
SELECT AVG(vmag) FROM star;
```

```
SELECT stellarType, MIN(vmag), MAX(vmag) FROM star GROUP BY  
stellarType;
```

```
SELECT stellarType, AVG(vmag), COUNT(id) FROM star GROUP BY  
stellarType HAVING vmag > 14;
```

Create

- `CREATE DATABASE databaseName;`
- `CREATE TABLE tableName (name1 type1, name2 type2, ...);`

```
CREATE TABLE star (name varchar(20), ra float, dec float, vmag float);
```

- Data types:
 - Boolean, bit, tinyint, smallint, int, bigint;
 - real/float, double, decimal;
 - char, varchar, text, binary, blob, longblob;
 - date, time, datetime, timestamp

```
CREATE TABLE star (name varchar(20) not null, ra float default 0, ...);
```

KEYS

A **primary key** is a unique identifier for a row and is automatically not null

```
CREATE TABLE star (name varchar(20), ra float, dec float,  
vmag float, CONSTRAINT PRIMARY KEY (name));
```

A **foreign key** is a referential constraint between two tables identifying a column in one table that refers to a column in another table.

```
CREATE TABLE star (name varchar(20), ...,  
stellarType varchar(8), CONSTRAINT stellarType_fk  
FOREIGN KEY (stellarType) REFERENCES  
stellarTypes(id));
```

Show and Describe

SHOW ...

SHOW INDEXES IN star;

SHOW WARNINGS;

DESCRIBE...

DESCRIBE star;

INSERT

INSERT INTO *table* VALUES(val1, val2, ...);

```
INSERT INTO star VALUES('Sirius', 101.287, -16.716,  
-1.47);
```

```
INSERT INTO star(name, vmag) VALUES('Canopus', -  
0.72);
```

```
INSERT INTO star SELECT ...;
```

DELETE

DELETE FROM *table* WHERE *condition*;
DROP TABLE *table*;

```
DELETE FROM star WHERE name = 'Canopus';  
DELETE FROM star WHERE name LIKE 'C_n%';
```

UPDATE

UPDATE *table* SET *column* = val1 WHERE condition;

```
UPDATE star SET vmag = vmag + 0.5;
```

```
UPDATE star SET vmag = -1.47 WHERE name LIKE 'Sirius';
```

```
UPDATE star INNER JOIN temp on star.id = temp.id SET star.vmag =  
temp.mag;
```

ALTER

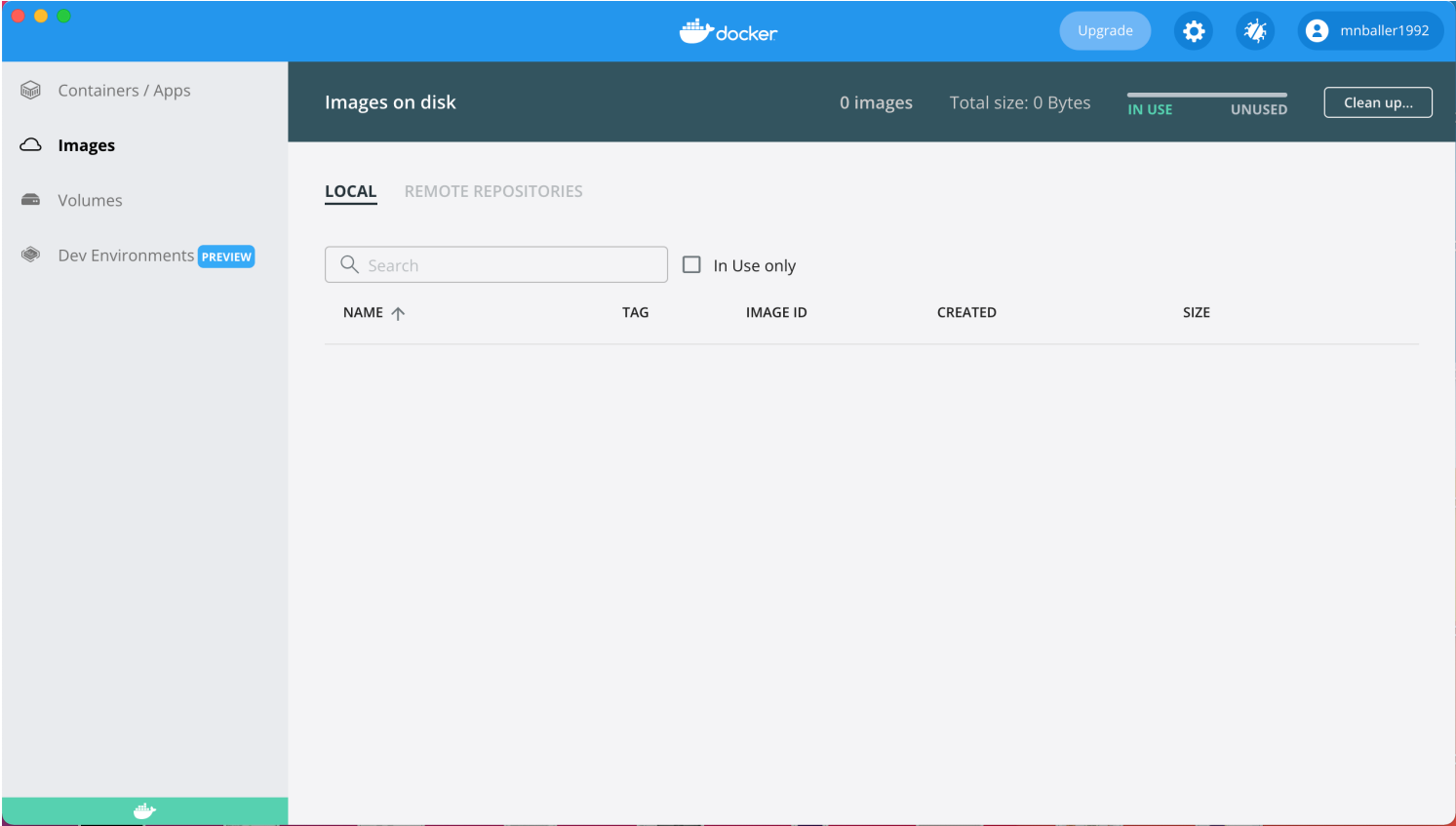
ALTER TABLE *table* ...;

```
ALTER TABLE star ADD COLUMN bmag double AFTER vmag;
```

```
ALTER TABLE star DROP COLUMN bmag;
```


How to use docker-compose

Launch DockerHub



Open Terminal or PowerShell

```
(base) scottcoughlin@EDMRCSLC02G23LAMD6R ~ % pwd
/Users/scottcoughlin
(base) scottcoughlin@EDMRCSLC02G23LAMD6R ~ % cd ~/Documents/GitHub/LSSTC-DSFP-
Sessions/Sessions/Session15/Day3/docker
(base) scottcoughlin@EDMRCSLC02G23LAMD6R docker % ls
database  docker-compose.yml  jupyter
(base) scottcoughlin@EDMRCSLC02G23LAMD6R docker %
```

docker-compose

```
# Mac or Linux
```

```
$ cd LSSTC-DSFP-Sessions/Sessions/Session15/Day3/docker
```

```
# Power Shell
```

```
$ cd LSSTC-DSFP-Sessions\Sessions\Session15\Day3\docker
```

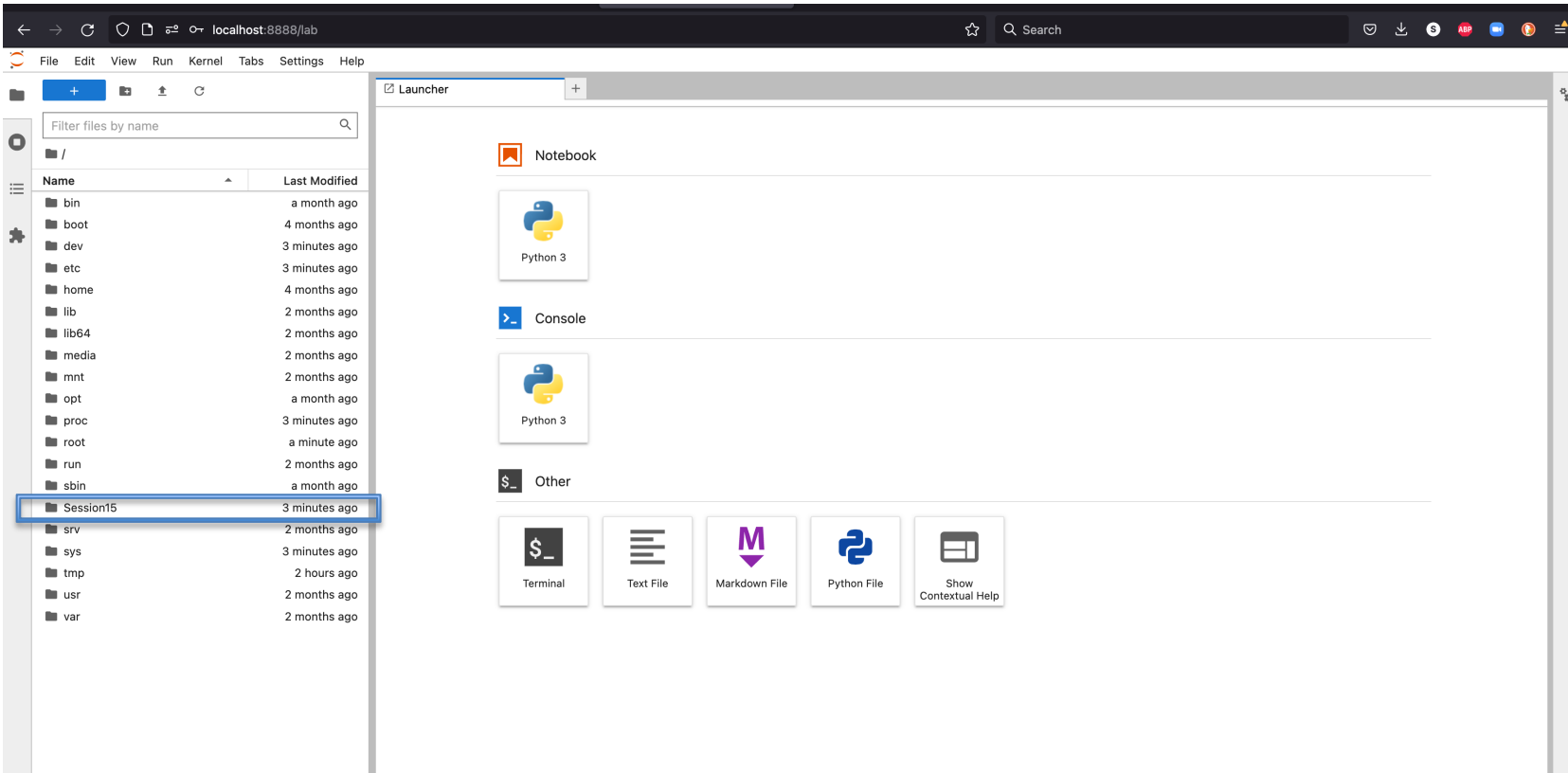
```
$ docker-compose up
```

- Three things are happening once you run this command!
 - A container with PostgreSQL installed is being downloaded, a database is being created and a table is being made for each “sheet” of IMBD data.
 - A useful web application, pgAdmin, is also being downloaded so that you can interact with PostGres via your browser.
 - A container with JupyterLab and Django is being made.

JupyterLab

- Wait about 3 minutes and then...
- In your browser go to localhost:8888
- Password: Session15

JupyterLab



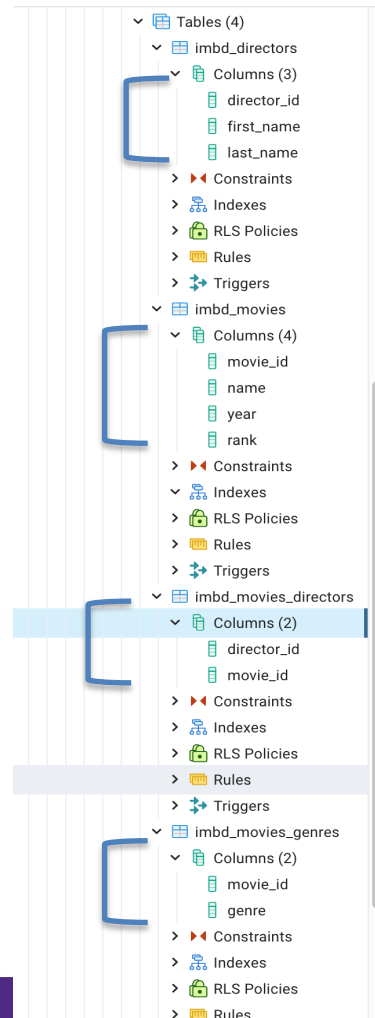
JupyterLab

The screenshot displays the JupyterLab web interface in a browser window. The address bar shows the URL `localhost:8888/lab/tree/Session15/Day3/IntroductionToSQL.ipynb`. The interface is divided into three main sections:

- File Browser (Left):** A sidebar showing a file tree for `/ Session15 / Day3 /`. It lists files like `docker`, `jumpballodds`, `IMDB-directors.csv`, `IMDB-movies_directors.csv`, `IMDB-movies_genres.csv`, `IMDB-movies.csv`, `IntroductionToDjango.ipynb`, `IntroductionToSQL.ipynb` (selected), `IntroductionToSQLMichael.ipynb`, and `README.md`.
- Code Editor (Center):** A window titled `IntroductionToSQL.ipynb` showing Python code. The first cell contains imports for `numpy`, `pandas`, `matplotlib.pyplot`, `sqlalchemy`, and `psycopg2`. Subsequent cells show database connection attempts using `load_ext sql` and `sql postgresql://imdb:imdb_admin@postgres:5432/imdb_database`. An error message is displayed: `(psycopg2.OperationalError) connection to server at "postgres" (172.28.0.2), port 5432 failed: FATAL: password authentication failed for user "imdb"`. A note below the error suggests checking the connection string format.
- Document Viewer (Right):** A window titled `Introduction to SQL (Structured Query Language)` showing a document. The document includes a version number (0.1), author information (Scott Coughlin), and a discussion of data handling efficiency. It lists several ways to improve performance: building faster algorithms, spreading calculations across multiple computers, and finding compact storage solutions. It concludes by stating that the session will start with simple queries and discusses creating tables using `pandas`.

The bottom status bar indicates the current mode is `Command`, the cursor is at `Ln 1, Col 1`, and the active file is `IntroductionToSQL.ipynb`.

JupyterLab



PGADMIN

In your browser go to localhost:15432

Username: admin@pgadmin.com

Password: password

