SQL (aka sequel)
STRUCTURED QUERY LANGUAGE

Create, populate + manipulate databases

1 Create a data model to represent the objects + relationships in a database

2 Create schemas, tables and databases for relational data

3 Retrieve data using advanced queries

SQL is designed to efficiently handle large amounts of data, which is a highly valued capability

POSTGRES AND pgADMIN

pgAdmin is a management tool for postgres

PostgreSQL is an object-relational database

To the left of screen under Servers
default is Postgres

Create a database
Right click on PostgreSQL 12
Create → Database
give it a name - click on SQL tab

CREATE DATABASE "animal_DB"
hit save + it creates a database

when you click on animal_db there are many more things to set under it

there are initially no tables

In an entire company you may have different tables for different departments

SQL

create schema

```
CREATE SCHEMA
    AUTHORIZATION
```

Now we want to create a TABLE

want to create tables for a query
right click animal_DB and gives you a query tool - opens a query window for you

```
create table people (
    name varchar(30) NOT NULL,
    has_pet boolean default false,
    pet_type varchar(10) NOT NULL,
    pet_name varchar(30),
    pet_age int
);
```

cannot be NULL
colname varchar
; ← ends the statement

now execute this code hit the play button
now you can look in your table + see it

```
SELECT * FROM PEOPLE;          -- everything

SELECT PET_NAME, PET_TYPE FROM PEOPLE;

INSERTING VALUES

INSERT INTO PEOPLE
  (NAME, HAS_PET, PET_TYPE, PET_NAME,
   PET_AGE)
  VALUES ('Ann', true, 'Dog', 'Bailey', 10)
         'James', true, 'Cat', 'Queso', 1);

-- you can add as many rows as you like
-- end statement

SELECT PET_NAME, PET_TYPE FROM PEOPLE;

SELECT PET_NAME, PET_TYPE
FROM PEOPLE
WHERE PET_TYPE = 'Dog'          -- THIS IS case-sensitive
```

What if you want to limit it to age of pet

```
SELECT
FROM
WHERE
  and pet_age < 5
;
```

```
CREATE TABLE cities (
    city    VARCHAR(30) NOT NULL,
    state   VARCHAR(30) NOT NULL,
    Population INT NOT NULL
);

INSERT INTO cities (city, state, Population)
VALUES
    ('Alameda', 'California', 79177),
    ('Mesa', 'Anzona', 496401),
    ('Boene', 'Texas');

SELECT city FROM cities;

SELECT city FROM cities WHERE state = 'Arizona';
WHERE population < 100000;

WHERE state = 'California'
AND OR population < 100000;
```

V

DROP

```
DROP TABLE people;
```

dropping entire structure

PRIMARY KEYS

```
CREATE TABLE people (
    id SERIAL PRIMARY KEY,
    name varchar(30) NOT NULL)
```

→ this will generate a sequential id for you

DELETE

```
DELETE FROM PEOPLE
WHERE ID = 3
```

UPDATE

```
UPDATE people
SET has_pet = true, pet_name = 'Rocket',
    pet_age = 8
where id = 6
```

This will change values in the columns you specified.

The keyword serial will generate a new value

Importing from CSV

create a new DB

new query tool

create a DB TABLE that has same no of cols as csv file

empty table structure to input

names don't have to be same but structure has to be

```
CREATE TABLE birdsong (
    englishname VARCHAR,
    country VARCHAR,
    longitude DEC,
    latitude DEC
);
```

now load csv file - select/highlight the table

go to import/export prompt

change export to import

find the csv file

header is yes

select , as delimeter

SELECT * from publicobirdsong

OID is index (object ID) then check that

you can also click on columns tab to check headers in csv

you can select columns from csv that you want in that tab

ORDER BY author

SELECT * FROM tablename
WHERE author >= 1
AND author <= 10
ORDER BY AUTHOR, WORD1, WORD2

AVG

SELECT AVG(column name)
FROM TABLE NAME
WHERE CONDITION

When looking for strings in sql use ' '

Create — Insert INTO table
Read — SELECT * FROM
Update — UPDATE table SET column1 = value WHERE cond
Delete — DELETE FROM table WHERE idx5

WILDCARDS - substitute 0, 1 or multiple characters in a string
The word LIKE indicates use of a wildcard

```
select * from
  actor
  where lastname LIKE 'Will%';
```

% - any match begin Will

```
select *
from actor
where firstname LIKE '_AN'
```

only one character
(Dan, Yan, Nan, etc)

VIRTUAL TABLE
It does not exist
You need to create
a VIEW if you want
to keep it

UPPER

```
Upper(word1)
```

INNER JOIN
Join 2 tables based
on a common column

```
SELECT players.first_name, players.last_name, players.hand,
       matches, loser_rank
FROM matches
INNER JOIN players ON
  players.players_id = matches.loser_id;
```