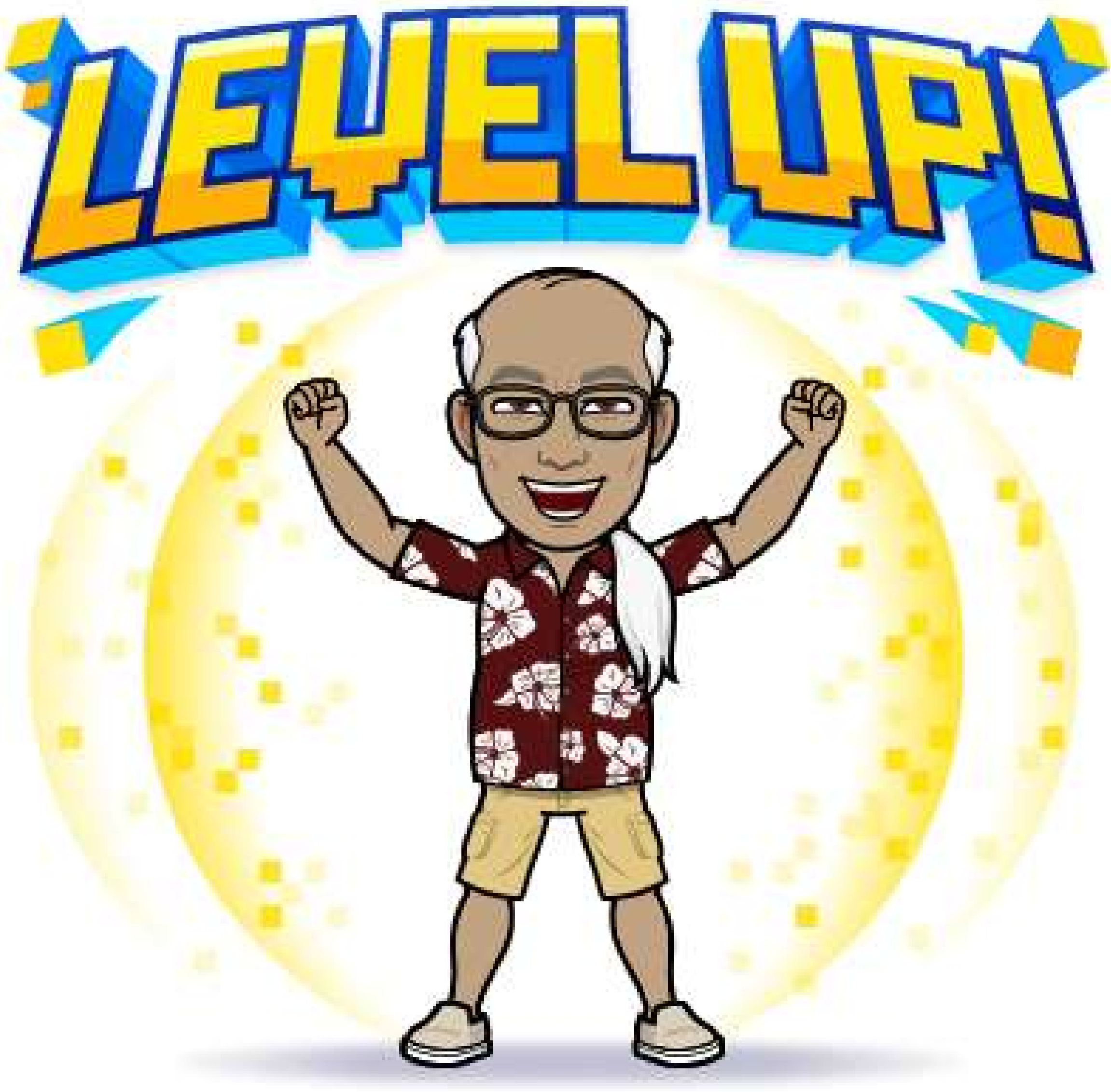
MODULE 2 DATABASE PROGRAMMING

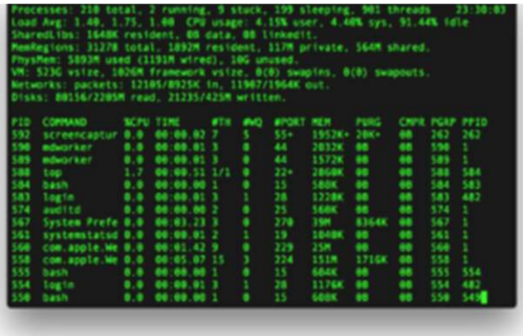
# Introduction to Databases and SELECT

TECH
ELEVATOR
A Stride Company

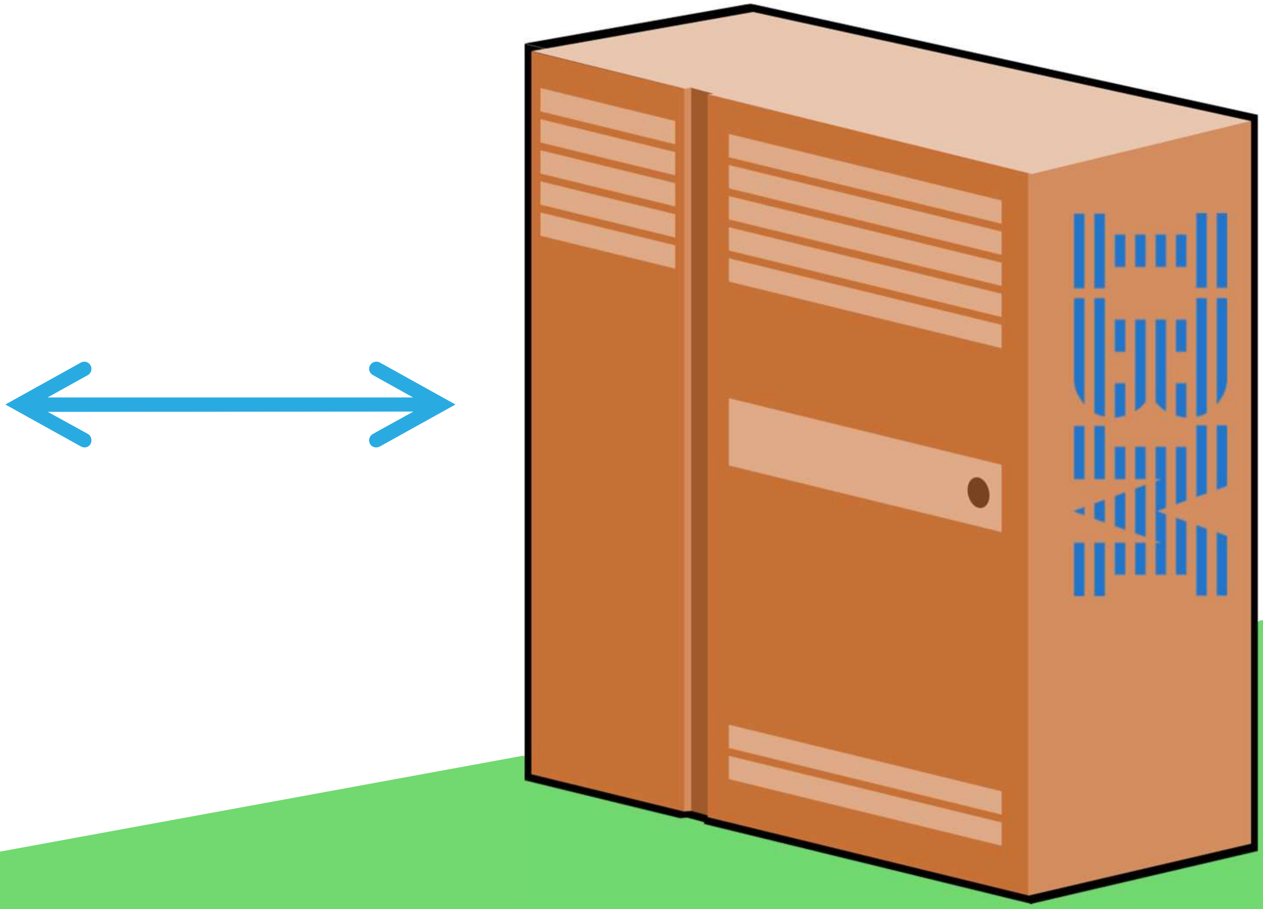


ELEVATE YOUR

# Before the PC Revolution



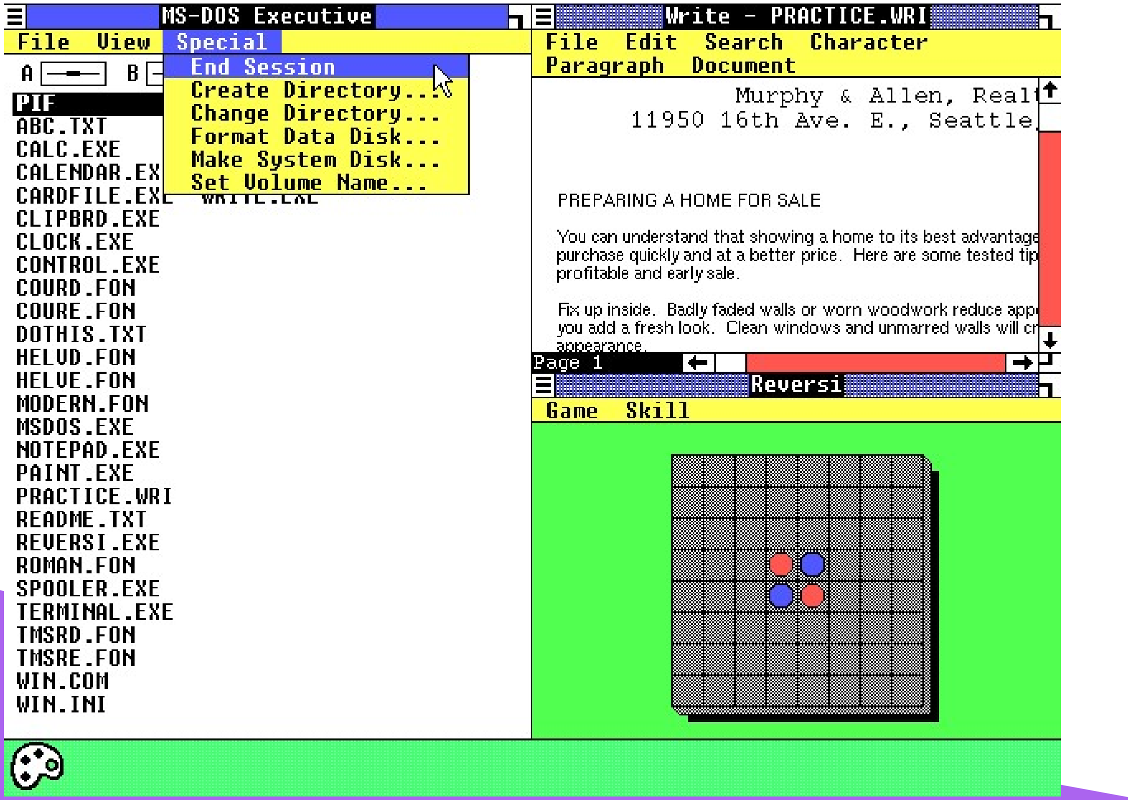
```
Processes: 210 total, 2 running, 9 stuck, 199 sleeping, 901 threads    23:30:03
Load Avg: 1.40, 1.75, 1.00  CPU usage: 4.15% user, 4.40% sys, 91.44% idle
SharedLibs: 1648K resident, 0B data, 0B linkedit.
MemRegions: 31278 total, 1092M resident, 117M private, 564M shared.
PhysMem: 5893M used (1191M wired), 10G unused.
VM: 523G vsize, 1026M framework vsize, 0(0) swapins, 0(0) swapouts.
Networks: packets: 12105/8925K in, 11907/1964K out.
Disks: 80156/2205M read, 21235/425M written.

PID  COMMAND      %CPU TIME     #TH  #WQ  #PORT MEM    PURG   CMPR PGRP PPID
592  screencaptur 0.0  00:00.02 7    5    55+   1952K+ 20K+   0B   262  262
590  mdworker     0.0  00:00.01 3    0    44    2032K  0B     0B   590  1
589  mdworker     0.0  00:00.01 3    0    44    1572K  0B     0B   589  1
588  top          1.7  00:00.51 1/1  0    22+   2860K  0B     0B   588  584
584  bash         0.0  00:00.00 1    0    15    580K   0B     0B   584  583
583  login        0.0  00:00.01 3    1    28    1220K  0B     0B   583  482
574  auditd       0.0  00:00.00 2    0    25    560K   0B     0B   574  1
567  System Prefe 0.0  00:03.23 3    0    270   39M    8364K  0B   567  1
561  systemstatsd 0.0  00:00.01 2    1    19    1040K  0B     0B   561  1
560  com.apple.We 0.0  00:01.42 9    0    229   25M    0B     0B   560  1
558  com.apple.We 0.0  00:05.07 15   3    224   151M   1716K  0B   558  1
555  bash         0.0  00:00.00 1    0    15    604K   0B     0B   555  554
554  login        0.0  00:00.01 3    1    28    1176K  0B     0B   554  482
550  bash         0.0  00:00.00 1    0    15    600K   0B     0B   550  549
```



ELEVATE YOURSELF

# PC Revolution!

# GUI Revolution



```
PIF
ABC.TXT
CALC.EXE
CALENDAR.EX
CARDFILE.EXE
CLIPBRD.EXE
CLOCK.EXE
CONTROL.EXE
COURD.FON
COURE.FON
DOTHIS.TXT
HELVD.FON
HELVE.FON
MODERN.FON
MSDOS.EXE
NOTEPAD.EXE
PAINT.EXE
PRACTICE.WRI
README.TXT
REVERSI.EXE
ROMAN.FON
SPOOLER.EXE
TERMINAL.EXE
TMSRD.FON
TMSRE.FON
WIN.COM
WIN.INI
WRITE.EXE
```

Murphy & Allen, Realt
11950 16th Ave. E., Seattle

PREPARING A HOME FOR SALE

You can understand that showing a home to its best advantage purchase quickly and at a better price. Here are some tested tip profitable and early sale.

Fix up inside. Badly faded walls or worn woodwork reduce app you add a fresh look. Clean windows and unmarred walls will cr appearance.

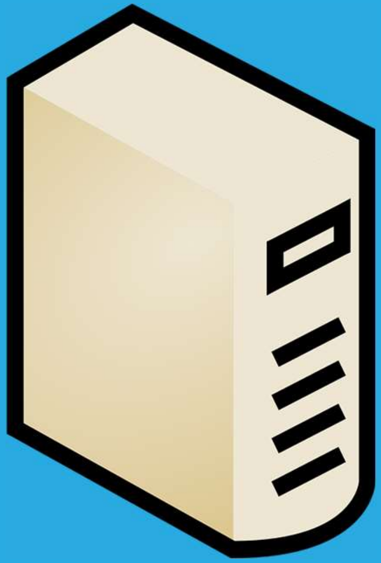
# Internet Revolution

# Separation of Concerns

User interaction
UI, UX

Module 3

Business
Logic



Persistent
Storage

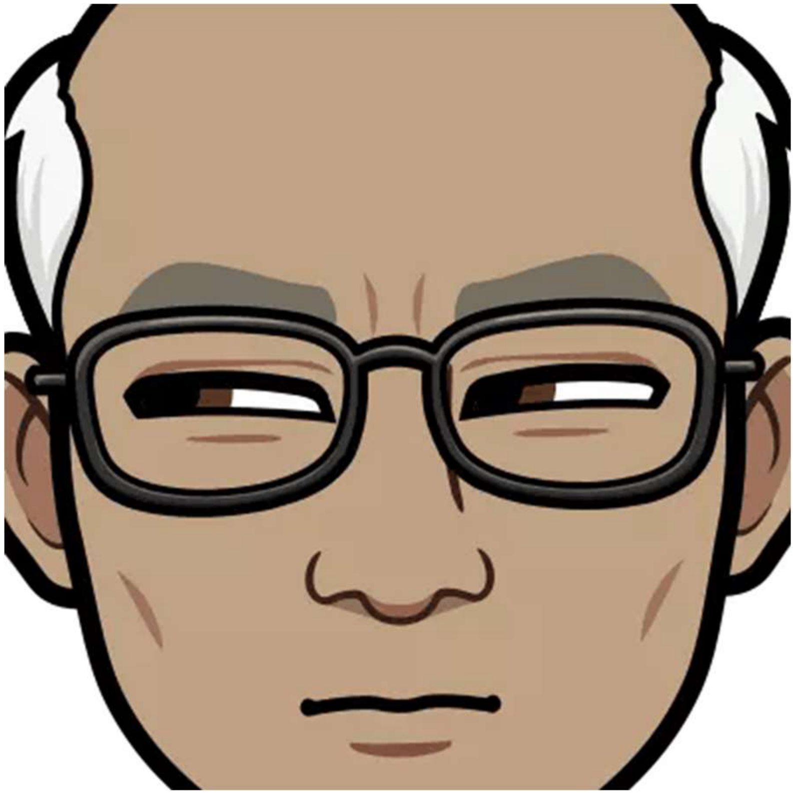
Module 2

Remember those awesome programs from Module 1?

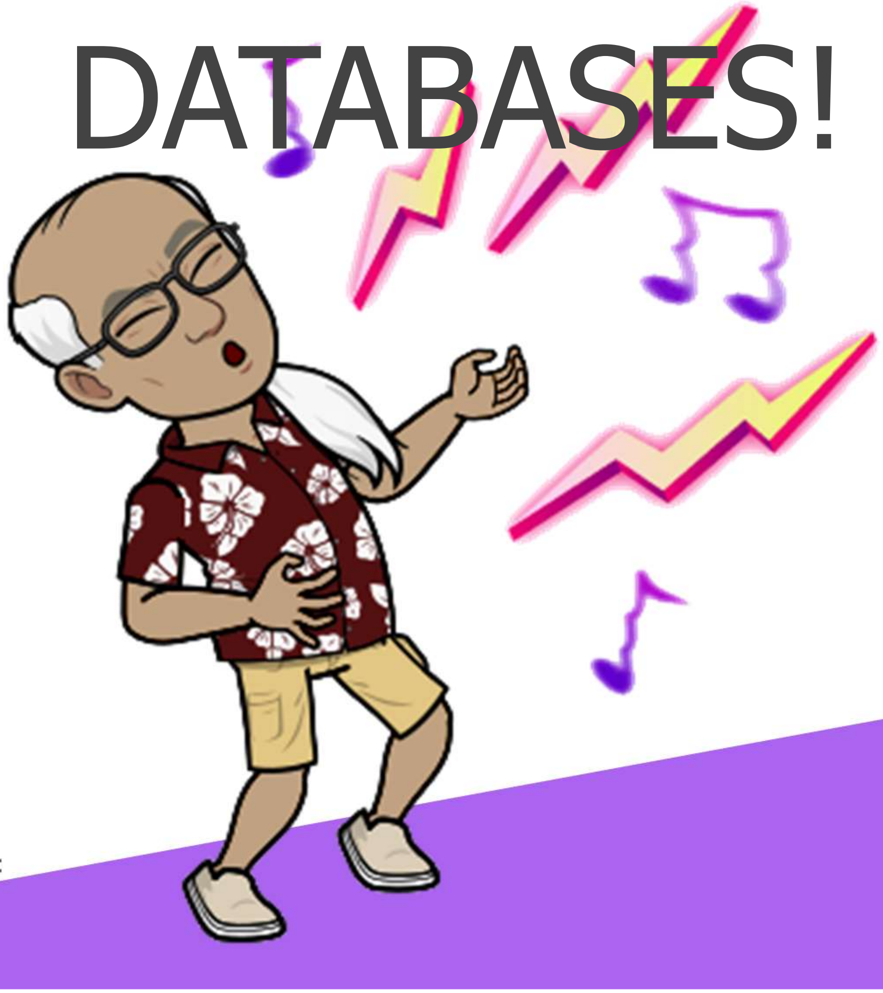
# Typing fun?

```
Dictionary<string, string> animals = new Dictionary<string, string>()
        {
           {"rhino", "Crash" },
           {"giraffe", "Tower" },
           {"elephant", "Herd" },
           {"lion", "Pride" },
           {"crow", "Murder" },
           {"pigeon", "Kit" },
           {"flamingo", "Pat" },
           {"deer", "Herd" },
           {"dog", "Pack" },
           {"crocodile", "Float" },
        };
```

# Let’s use files!



# DATABASES!



# Official nerdy definition

- A **database** is an organized collection of data that can be accessed, managed, and updated

# Sample Entity

Record

Attribute

**ANSI SQL Datatypes**

char

varchar

nvarchar

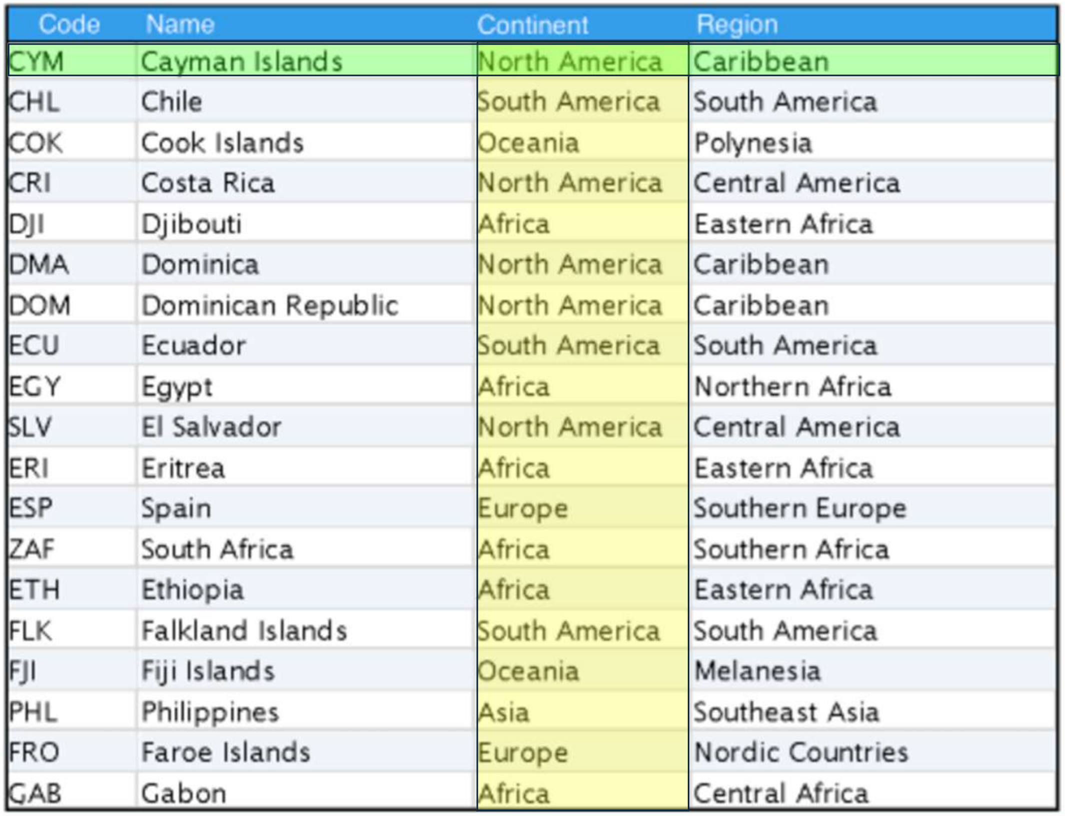
int

decimal

bigint

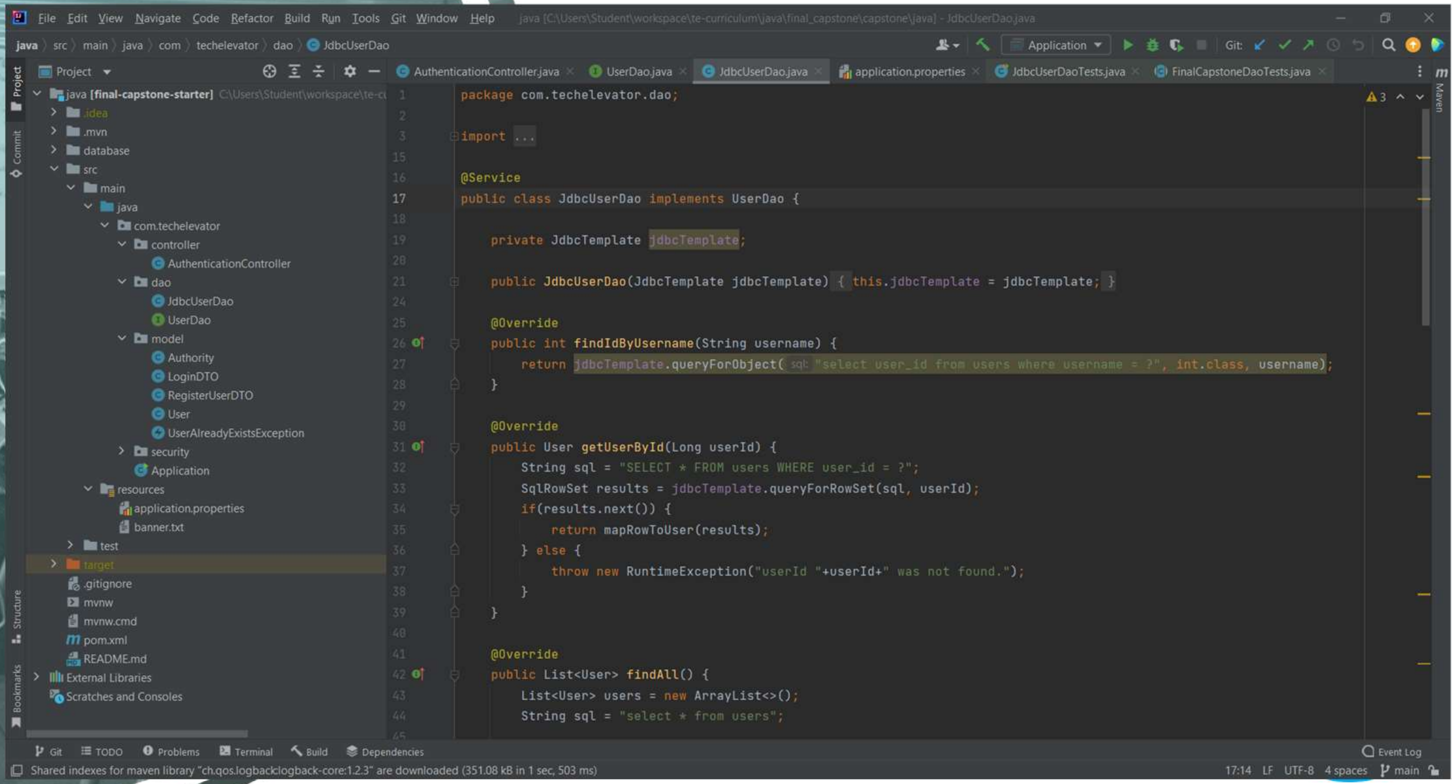
boolean/bit

datetime



| Code | Name | Continent | Region |
|---|---|---|---|
| CYM | Cayman Islands | North America | Caribbean |
| CHL | Chile | South America | South America |
| COK | Cook Islands | Oceania | Polynesia |
| CRI | Costa Rica | North America | Central America |
| DJI | Djibouti | Africa | Eastern Africa |
| DMA | Dominica | North America | Caribbean |
| DOM | Dominican Republic | North America | Caribbean |
| ECU | Ecuador | South America | South America |
| EGY | Egypt | Africa | Northern Africa |
| SLV | El Salvador | North America | Central America |
| ERI | Eritrea | Africa | Eastern Africa |
| ESP | Spain | Europe | Southern Europe |
| ZAF | South Africa | Africa | Southern Africa |
| ETH | Ethiopia | Africa | Eastern Africa |
| FLK | Falkland Islands | South America | South America |
| FJI | Fiji Islands | Oceania | Melanesia |
| PHL | Philippines | Asia | Southeast Asia |
| FRO | Faroe Islands | Europe | Nordic Countries |
| GAB | Gabon | Africa | Central Africa |

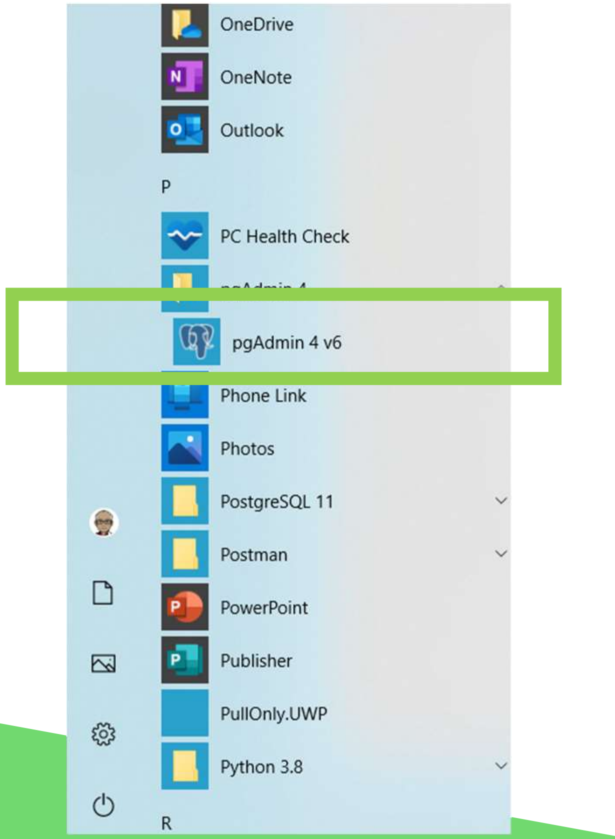
# WRITE CODE?



```java
package com.techelevator.dao;

import ...

@Service
public class JdbcUserDao implements UserDao {

    private JdbcTemplate jdbcTemplate;

    public JdbcUserDao(JdbcTemplate jdbcTemplate) { this.jdbcTemplate = jdbcTemplate; }

    @Override
    public int findIdByUsername(String username) {
        return jdbcTemplate.queryForObject( sql: "select user_id from users where username = ?", int.class, username);
    }

    @Override
    public User getUserById(Long userId) {
        String sql = "SELECT * FROM users WHERE user_id = ?";
        SqlRowSet results = jdbcTemplate.queryForRowSet(sql, userId);
        if(results.next()) {
            return mapRowToUser(results);
        } else {
            throw new RuntimeException("userId "+userId+" was not found.");
        }
    }

    @Override
    public List<User> findAll() {
        List<User> users = new ArrayList<>();
        String sql = "select * from users";
```

Shared indexes for maven library "ch.qos.logback:logback-core:1.2.3" are downloaded (351.08 kB in 1 sec, 503 ms)

# LAUNCH PG Admin



# Don’t we need some data?

ELEVATE YOURSELF

# Load the world!

# THE LANGUAGE OF DATABASES

**SQL (Structured Query Language)**

- SQL consists of:
  - data definition language to define the data structures
  - data manipulation language to query and modify the data in a database
  - data control language to define access to a particular database

SQL is a declarative programming language

# THE LANGUAGE OF DATABASES

**SELECT** is used to return data

```
SELECT [column], [column-n] FROM [table];
```

# THE LANGUAGE OF DATABASES

- **WHERE clause** is used to filter the result set using one or more criteria rules

- Conditional clauses in the WHERE clause can include:
  - =, <>, !=, >, >=, <, <=
  - IN(values), NOT IN(values)
  - BETWEEN value AND value
  - IS NULL, IS NOT NULL
  - LIKE (with wildcard characters)
  - ILIKE
- Multiple conditions use AND and OR

# THE LANGUAGE OF DATABASES

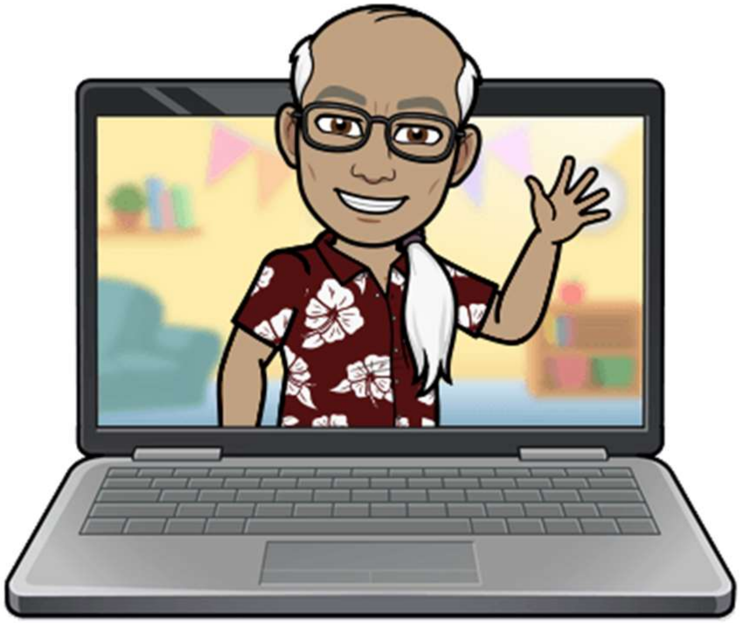
- **DISTINCT** to remove duplicates

- SELECT **DISTINCT** [column], [column-n] FROM [table];

- **AS** is an alias

- SELECT name **as title** from city

# LET’S CODE!



# ORDERING results

**SYNTAX:** ORDER BY col1 [ASC | DESC], col2 [ASC | DESC]

SELECT [state_name],[sales_tax],[state_abbreviation] FROM [UnitedStates].[dbo].[state] **order by sales_tax desc, state_name**

SELECT [state_name],[sales_tax],[state_abbreviation] FROM [UnitedStates].[dbo].[state] **where sales_tax > 6.0** order by sales_tax desc, state_name

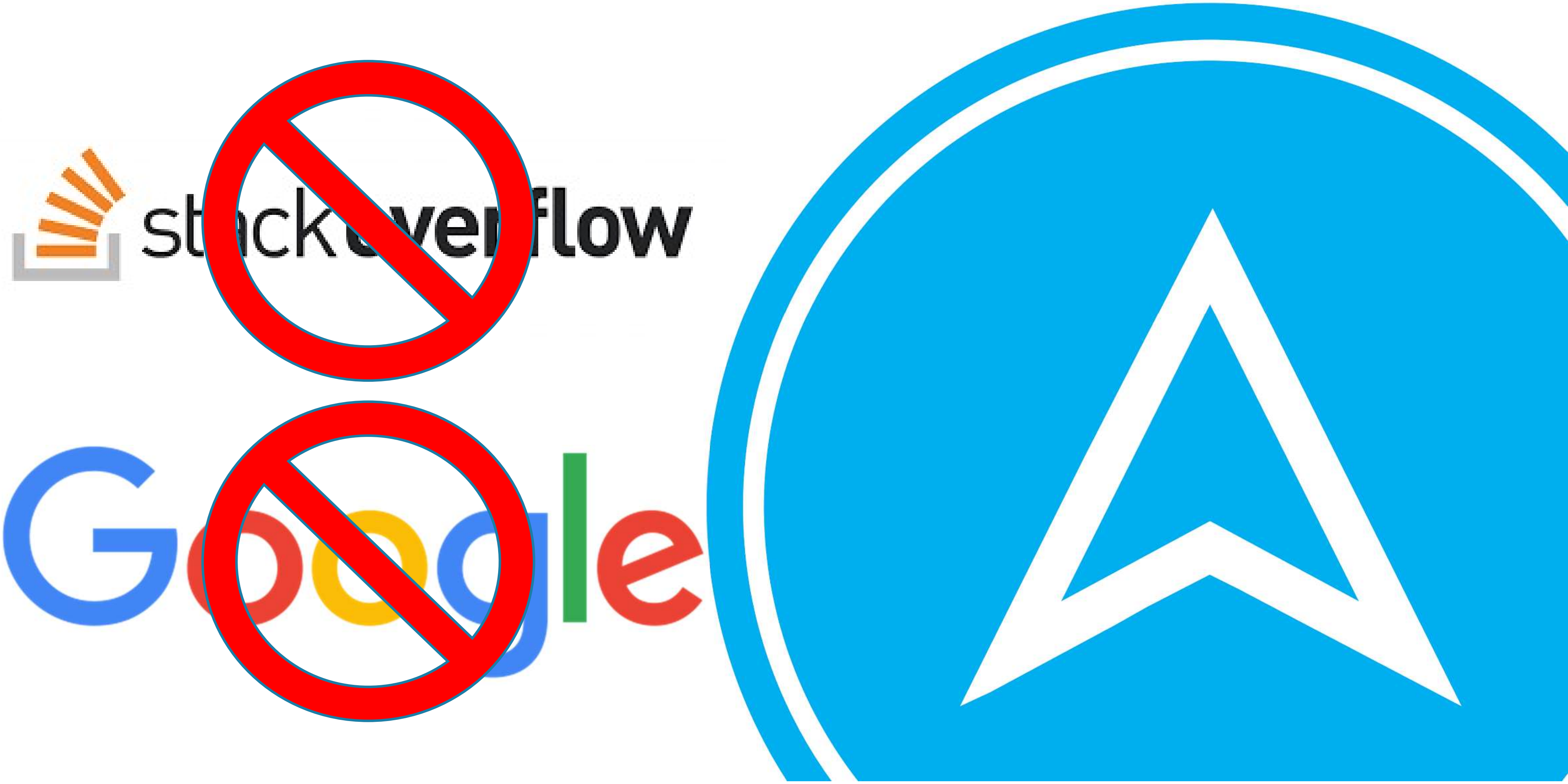
# LIMITING results

**SYNTAX:** LIMIT *n*

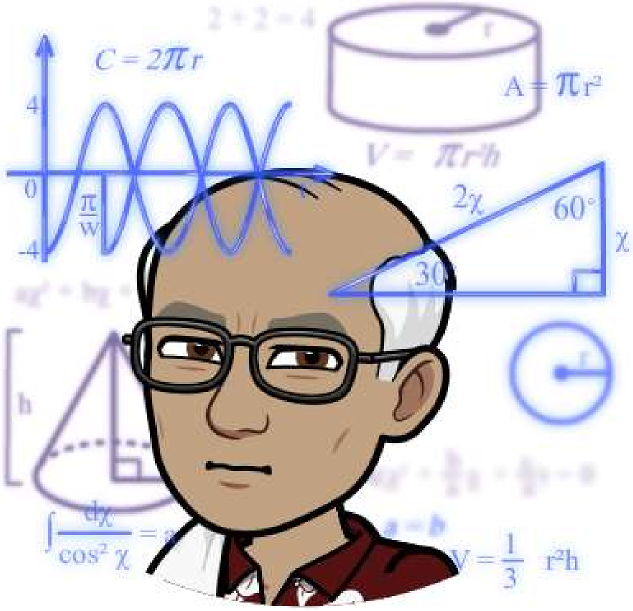
SELECT [state_name],[sales_tax],[state_abbreviation] FROM [UnitedStates].[dbo].[state] where sales_tax > 6.0 order by sales_tax desc, state_name **LIMIT 5**

# WHAT QUESTIONS DO YOU HAVE?

# For Module 1



# Capstone Code Reviews
Today 1:00 – 3:30



# Reading for tonight:
# **Subqueries and Aggregates**