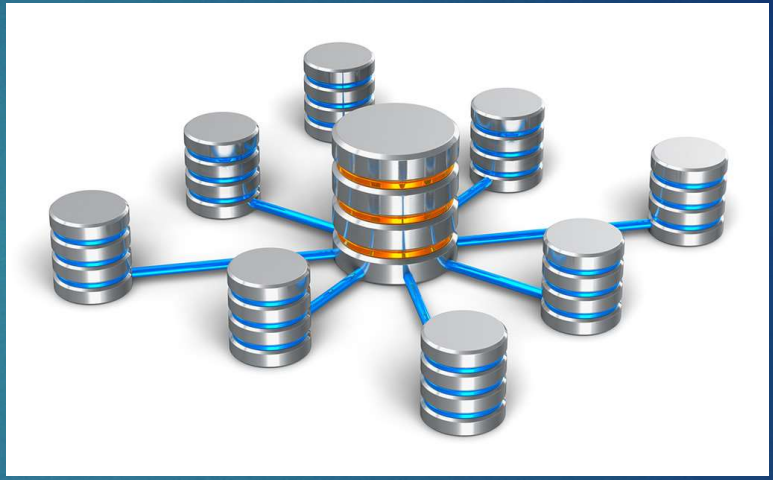
Logols Learning

WEEKEND WEB DEVELOPMENT BOOT CAMP

TRAINING: SQL

MySql

- Database
  - Tables
    - Columns
    - Rows



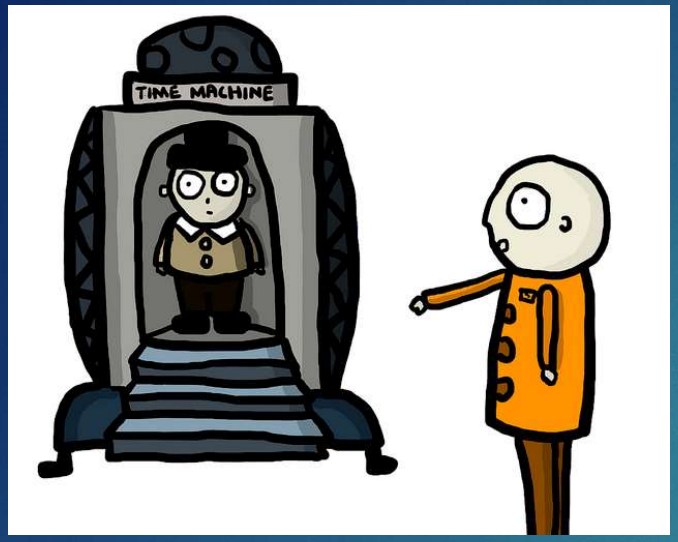
MySql Workbench

Select Statements

- Components in Logical Order
  - SELECT
  - FROM
  - WHERE
  - GROUP BY
  - HAVING
  - ORDER BY

- Example:

```
SELECT FirstName,
LastName
FROM Person
WHERE personId = 2
```

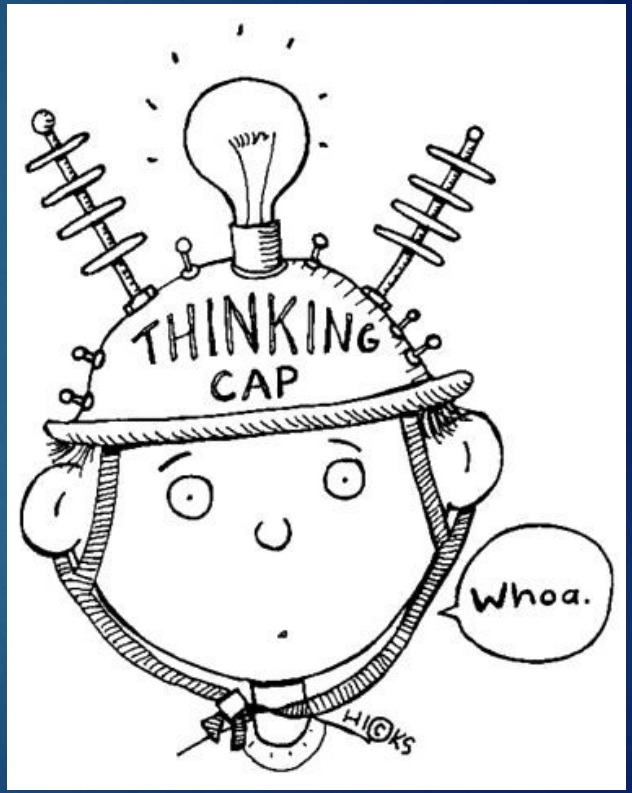


EXAMPLE

SELECT QUERIES

ASSESSMENT

SELECT QUERIES



Insert Statements

- Components
  - INSERT
  - INTO
  - VALUES
- Example:

  INSERT INTO Person (FirstName, LastName)

  VALUES ('Joe', 'Mackie')

Update Statements

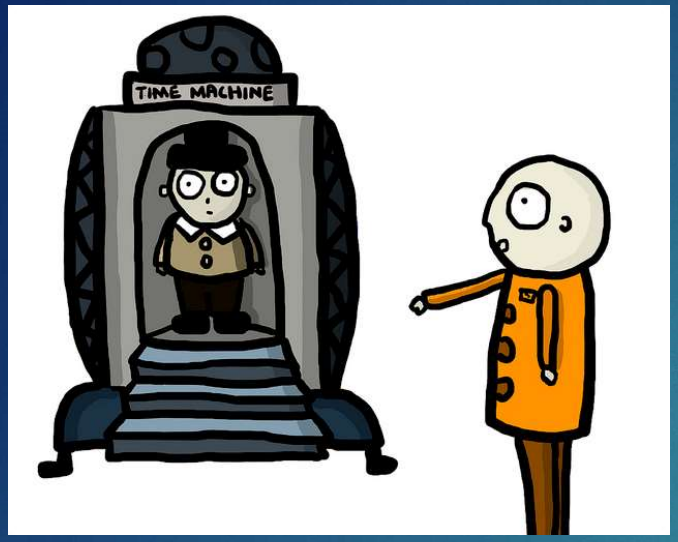
- Components
  - UPDATE
  - SET
  - WHERE
- Example:

```
UPDATE Person
  SET FirstName = 'Joe'
WHERE LastName =
'Mackie'
```

Delete Statements

- Components
  - DELETE
  - FROM
  - WHERE
- Example:

```
DELETE
FROM Person
WHERE LastName =
'Mackie'
```

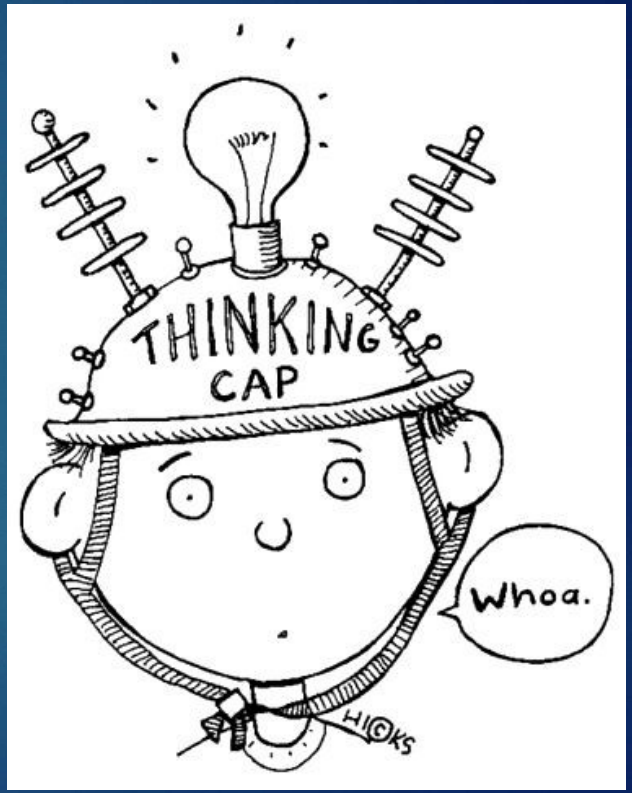


EXAMPLE

INSERT, UPDATE, AND DELETE QUERIES

ASSESSMENT

INSERT, UPDATE, AND DELETE QUERIES



MySql Data Types

- int, smallint, tinyint, bigint
- bit
- numeric, decimal
- date, datetime
- char, text, varchar

Null

- Null is the lack of a value.
- When doing a comparison you use the is operator to see if a value is null

Ex.

```
SELECT FirstName, LastName
FROM Person
WHERE personId IS NULL
```

Comparison Operators

- < Less Than
- > Greater Than
- <= Less Than or Equal To
- >= Greater Than or Equal To
- = Equal To
- <> Not Equal To
- IS NULL
- IS NOT NULL

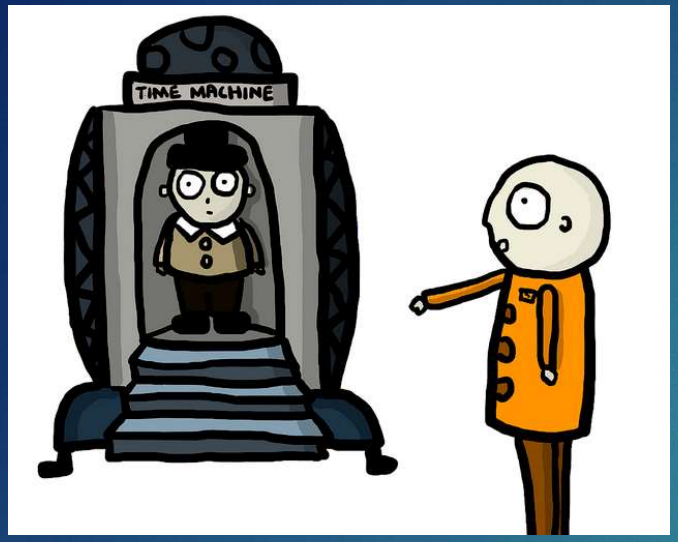
Create Table

- Components
  - CREATE TABLE
  - (
    - Column1 data type,
    - Column2 data type
  -);

```
CREATE TABLE People
(
    PersonID int,
    LastName varchar(255),
    FirstName varchar(255),
    Address varchar(255),
    City varchar(255)
);
```

Primary Key & Auto Increment

```
CREATE TABLE People
(
    PersonID int not null AUTO_INCREMENT,
    LastName varchar(255),
    FirstName varchar(255),
    Address varchar(255),
    City varchar(255),
    PRIMARY KEY (PersonID)
);
```

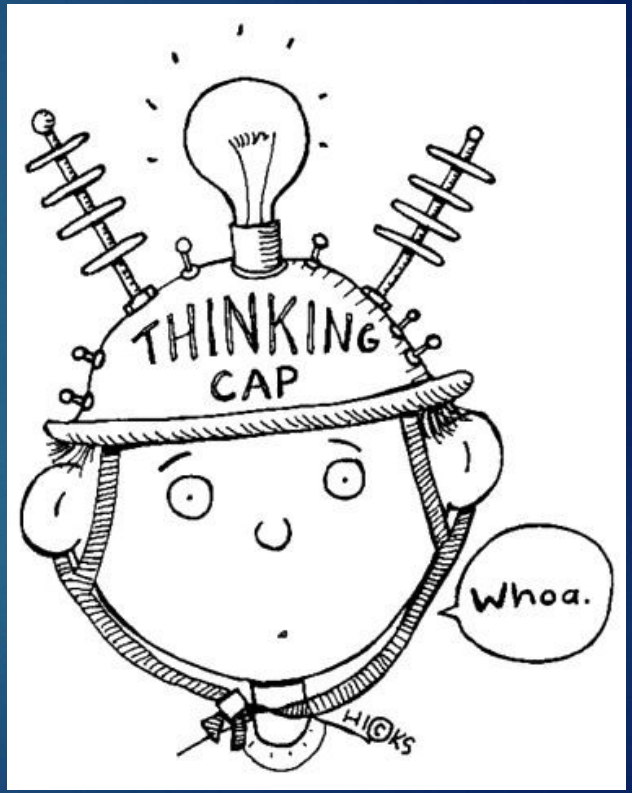


EXAMPLE

DATABASE AND TABLES

ASSESSMENT

CREATE TABLES



Assignment

- Create a table named Person with the following columns: PersonId, FirstName, LastName, and PersonStatusId.
- Create a table named PersonStatus with the following columns: PersonStatusId and StatusDescription.

Assignment

- Insert the following statuses into the PersonStatus table:
  - 1: Alive, 2: Zombie, 3: Dead, 4: Unknown
- Insert people into the Person table.

Assignment

- Perform a select of all people.
- Select all people that have an unknown status.
- Select all people that are alive or have an unknown status.
- Select all people that are alive and have first name Tom.

Assignment

- Update the status to Zombie for a given person based upon their name that currently has a status of Alive.
- Delete every person that is dead.

Table Alias

- Short name that can be given to a table
- Example:

```
SELECT c.ClassId, c.ClassName, t.TeacherId,
t.FirstName, t.LastName
FROM Class c
INNER JOIN Teacher t
ON c.TeacherId = t.TeacherId
```

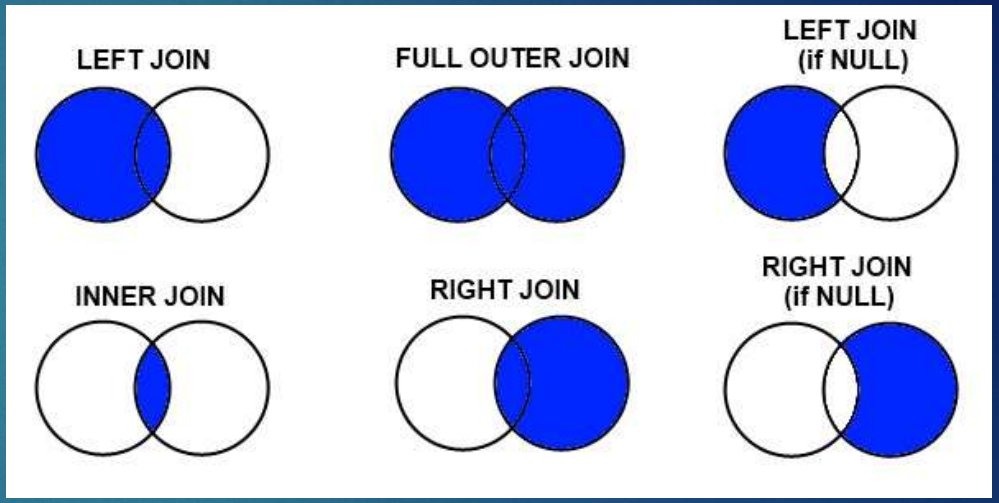
Column Alias

- Short name that can be given to a column
- Example:

```
SELECT c.ClassId, c.ClassName, t.TeacherId,
t.FirstName AS TeacherFirstName, t.LastName AS
TeacherLastName
FROM Class c
INNER JOIN Teacher t
ON c.TeacherId = t.TeacherId
```

Join Types

- Inner Join
- Left Outer Join
- Right Outer Join



Inner Join

- Every row from the first table will be matched with every row from the second table based upon the on conditions specified.

```
SELECT c.ClassId, c.ClassName, t.TeacherId,
t.FirstName, t.LastName
FROM Class c
INNER JOIN Teacher t
ON c.TeacherId = t.TeacherId
```

Left Outer Join

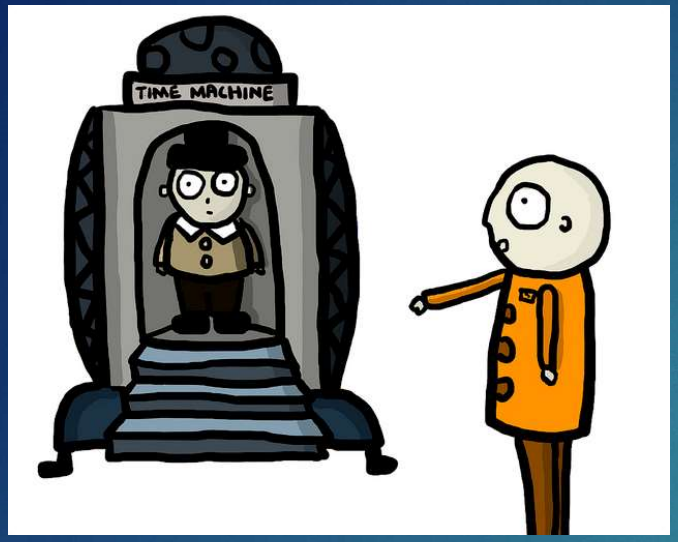
- Every row from the first table will be returned and results from the second table will be included only if the condition matches.

```
SELECT c.ClassId, c.ClassName, t.TeacherId,
t.FirstName, t.LastName
FROM Class c
LEFT OUTER JOIN Teacher t
ON c.TeacherId = t.TeacherId
```

Right Outer Join

- Every row from the second table will be returned and results from the first table will be included only if the condition matches.

```
SELECT c.ClassId, c.ClassName, t.TeacherId,
t.FirstName, t.LastName
FROM Class c
RIGHT OUTER JOIN Teacher t
ON c.TeacherId = t.TeacherId
```

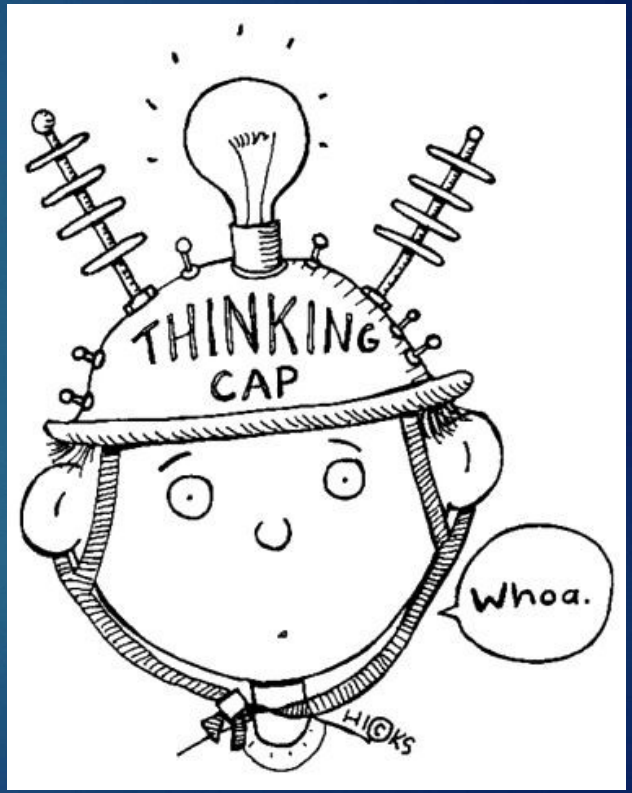


EXAMPLE

JOINS

ASSESSMENT

JOINS



Assignment

- Select all statuses joined with people to get a full list of everyone and their status.
- Change the select so that it shows all statuses regardless of whether the status is currently associated with any person.

What is an ORM?

- Object Relational Mapper
- Mismatch between Object Model and Relational Model
- ORM converts between the two
- Provides Query and Persistence Capability

Entity Framework

- Implementation of an ORM
- Created by Microsoft
- Linq Syntax used for Queries
- Data Model First vs. Code First

Dapper

- Micro ORM
- Performs only mapping and nothing else
- Fast
- Why are we using it?
  - Allows practice with database queries
  - Visibility into database and application interaction

Connection String Configuration

- Instead of hard coding
- appSettings.json file
- Read configurations from file
- Nuget Package:
  - Microsoft.Extensions.Configuration.Json

Base Repository

```
private string connectionString;
public Repository()
{
    var builder = new ConfigurationBuilder()
        .SetBasePath(Directory.GetCurrentDirectory())
        .AddJsonFile("appsettings.json");
    var connectionStringConfig = builder.Build();
    connectionString = connectionStringConfig.GetConnectionString("DefaultConnection");
}
public IDbConnection Connection
{
    get { return new MySqlConnection(connectionString); }
}
```

Repository

```
public Subject Get(int subjectID)
{
   using (IDbConnection dbConnection = Connection)
   {
      dbConnection.Open();
       string sql = "Select SubjectId, Name, Description From Subject Where SubjectId = @SubjectId“;
      return dbConnection.Query<Subject>(sql, new { SubjectId = subjectID }, commandType: CommandType.Text).FirstOrDefault();
   }
}
```

Using

- Creates instance, allows use inside block, and Disposes it
- Dispose of external resources like database connections.

```
public Subject Get(int subjectID)
{
  using (IDbConnection dbConnection = Connection)
  {
    statements…
  }
}
```

Opening the Connection

```
public Subject Get(int subjectID)
{
  using (IDbConnection dbConnection = Connection)
  {
    dbConnection.Open();
    …
  }
}
```

Sql Statement

```
public Subject Get(int subjectID)
{
   using (IDbConnection dbConnection = Connection)
   {
      …
      string sql = "Select SubjectId, Name, Description From Subject
Where SubjectId = @SubjectId‘;
      …
   }
}
```

Repository

```
public List<Subject> Get(int subjectID)
{
   using (IDbConnection dbConnection = Connection)
   {
      …
      return dbConnection.Query<Subject>(sql, new { SubjectId = subjectID }, commandType: CommandType.Text);
   }
}
```

Repository

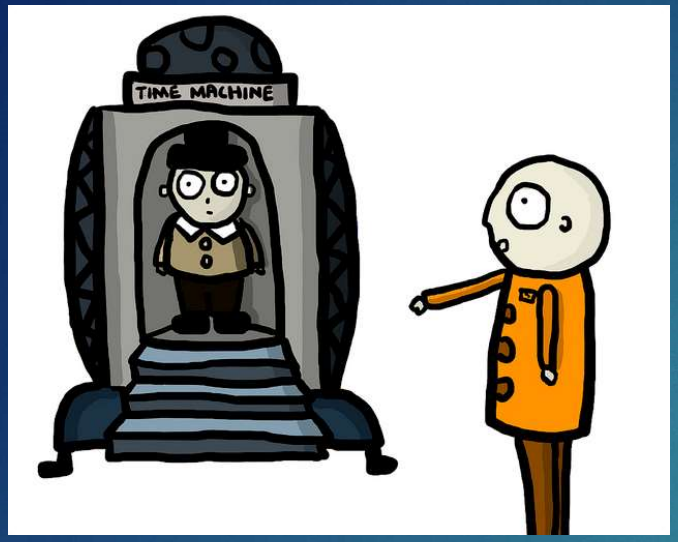
```
public Subject Get(int subjectID)
{
   using (IDbConnection dbConnection = Connection)
   {
      …
      return dbConnection.Query<Subject>(sql, new { SubjectId = subjectID }, commandType: CommandType.Text).FirstOrDefault();
   }
}
```

Repository

```
public void Insert(TimeTraveler timeTraveler)
{
    using (IDbConnection dbConnection = Connection)
    {
        …
        dbConnection.Execute(sql
            , new { FirstName = timeTraveler.FirstName,
                LastName = timeTraveler.LastName }
            ,commandType: CommandType.Text);
    }
}
```

CLI Commands

- mkdir – Create Directory
- cd – Change Directory
- Add project:
  - dotnet new classlib
  - dotnet new webapi
- Add reference:
  - dotnet add reference [path]/[name.csproj]
  - dotnet add package Dapper
  - dotnet add package MySql.Data
  - dotnet add package Microsoft.Extensions.Configuration.Json

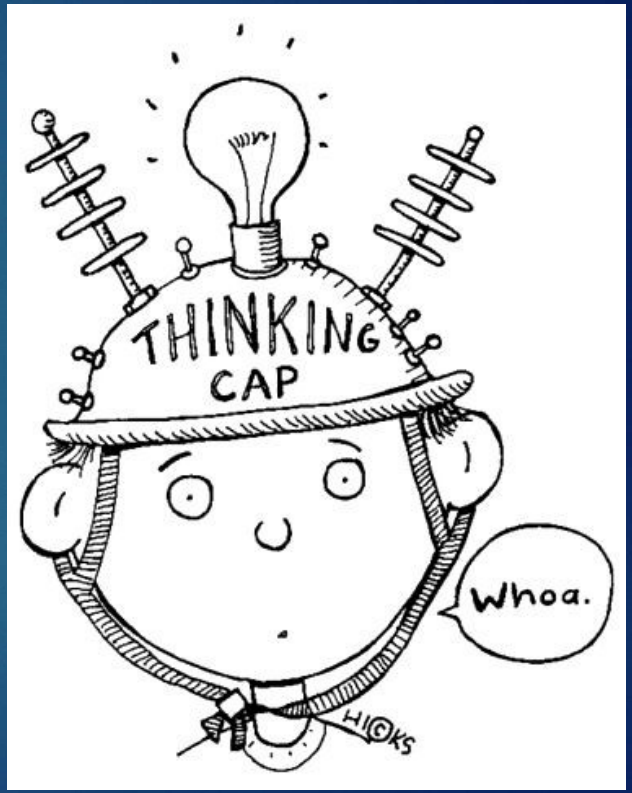


EXAMPLE

DAPPER

ASSESSMENT

DAPPER



Assignment

- A status report is needed of all government employees. Statuses are:
  - 1: Alive, 2: Zombie, 3: Dead, 4: Unknown
- Retrieve the data from the tables we created with a join.
- Loop through each record to display the status of each person in the database.

QUICK REVIEW

SQL

Not really a sign you'd want to see whilst driving through an eerily quiet neighbourhood...

Additional Resources

- Code Katas
  - https://www.codewars.com/
- UDacity
  - https://www.udacity.com/course/intro-to-relational-databases--ud197
- MySql
  - https://www.mysql.com/
- Sql Bolt
  - https://sqlbolt.com

Keep Practicing!

- Try creating more tables.
- Try different selects, inserts, deletes, and updates.
- Try different joins.
- Try connecting your new tables to an application using Dapper.