



# YESTERDAY...

What is normalization?

How do we create a table in the database?

How do we drop a table in the database?



#### Gallery Customer History Form

Customer Name

Jackson, Elizabeth 123 – 4<sup>th</sup> Avenue Fonthill, ON L3J 4S4

#### Purchases Made

Artist	Title	Purchase Date	Sales Price
03 - Carol Channing	Laugh with Teeth	09/17/2000	7000.00
15 - Dennis Frings	South toward Emerald Sea	05/11/2000	1800.00
03 - Carol Channing	At the Movies	02/14/2002	5550.00
15 - Dennis Frings	South toward Emerald Sea	07/15/2003	2200.00

The Gill Art Gallery wishes to maintain data on their customers, artists and paintings. They may have several paintings by each artist in the gallery at one time. Paintings may be bought and sold several times. In other words, the gallery may sell a painting, then buy it back at a later date and sell it to another customer.



#### Persistent State

What is state?



# Connecting to the database

What have we used to connect to the database?

A client =>

```
Admin File V Object V Tools V Help V
                     ■ Tm Q >_ Dashboard Properties SQL Statistics
       > # Foreign Data Wrappers
                                        1 -- Table: public.city
       > 🤤 Languages
       > 💕 Publications
                                        3 -- DROP TABLE IF EXISTS public.city;
       Schemas (1)
                                        5 CREATE TABLE IF NOT EXISTS public.city
         public
           > Aggregates
                                              city_id integer NOT NULL DEFAULT nextval('city_city_id_seq'::regclass),
           > A Collations
                                       8 city_name character varying(50) COLLATE pg_catalog."default" NOT NULL,
           > & Domains
                                       9 state_abbreviation character(2) COLLATE pg_catalog."default" NOT NULL,
           > 🖟 FTS Configurations
                                       10 population integer NOT NULL DEFAULT 0,
           > TS Dictionaries
                                             area numeric(5,1) NOT NULL,
                                             CONSTRAINT pk_city PRIMARY KEY (city_id),
           > Aa FTS Parsers
                                       13 CONSTRAINT fk_city_state FOREIGN KEY (state_abbreviation)
           >  FTS Templates
                                                REFERENCES public.state (state_abbreviation) MATCH SIMPLE
           > # Foreign Tables
                                       15
                                                  ON UPDATE NO ACTION
           > (ii) Functions
                                       16
                                                  ON DELETE NO ACTION
           > R Materialized Views
                                       17 )
           > 4 Operators
                                       18 WITH (
           > ( ) Procedures
                                       19 OIDS = FALSE
           > 1.3 Sequences
                                       21 TABLESPACE pg_default;

→ Tables (4)

                                       23 ALTER TABLE IF EXISTS public.city
               > 🗎 Columns
                                             OWNER to postgres;
               > > Constraints
               > 🤼 Indexes
               > 🔓 RLS Policies
```



# Solving the database conundrum

- How many databases are there?
  - Microsoft SQL
  - MySQL
  - Oracle
  - PostGRE SQL
  - etc.



#### Needs for database access

- Database connection information
  - DataSource object;
- Client object
  - JdbcTemplate(dataSource);
- Set the command text
  - String sql = <command text variable>;
- Query the Database
  - queryForRowSet(sql,<parameter>);
- Get the data
  - Results.next()



# Reading the data

```
    Get the data
    SqlRowSet results = jdbcTemplate.queryForRowSet(sql, cityId);
```

Read the data

```
while (results.next())
  {
    some code;
  }
```



#### Other Methods

- queryForObject(String,class,parameters)
  - Returns data of type <class>.
- update(String,parameters)
  - Updates or inserts into the database
  - No data is returned



#### **DAO Pattern**

• **Data Access Object (DAO)** design pattern encapsulates the details of persistent storage inside of classes whose only role is to store and retrieve data.

**CRUD** operations on domain objects.

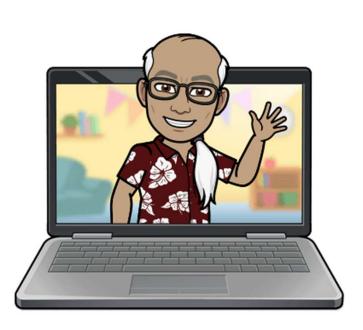
- Create
- Read
- Update
- Delete

DAO pattern makes code loosely coupled

- Handle structure changes
- Change database technology

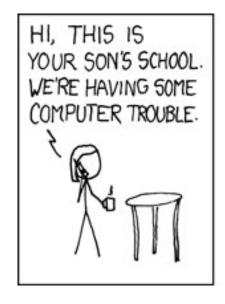


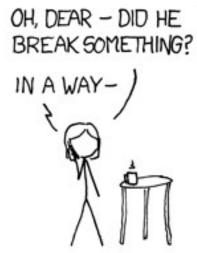
# LET'S CODE!

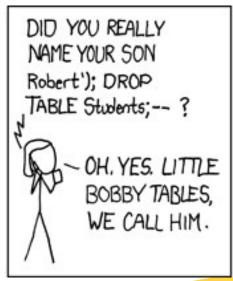


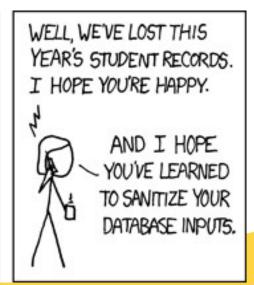


#### Stop the bad guys.











## SQL Injection

- 1. We ask the user for their input:
  - 1. Please enter your name:
  - 2. Henry Edwards
- 2. We use this input to query our database
  - String nameEntered = scanner.nextLine();
  - String sql = "SELECT \* FROM message WHERE private = FALSE AND sender\_name = " + nameEntered + " 'ORDER BY create\_date DESC"
- 3. We think that should be:
  - SELECT \* FROM message WHERE private = FALSE AND sender\_name = 'Henry Edwards' ORDER BY create\_date DESC
- 4. Riiight?



## SQL Injection

- 1. We ask the user for their input:
  - 1. Please enter your name:
  - 2. Robert'; Drop TABLE Students;--
- 2. We use this input to query our database
  - String nameEntered = scanner.nextLine();
  - 2. String sql = "SELECT \* FROM message WHERE private = FALSE AND sender\_name = " + nameEntered + " 'ORDER BY create\_date DESC"
- 3. We think that should be:
  - SELECT \* FROM message WHERE private = FALSE AND sender\_name ='Robert'; Drop TABLE Students;-- ORDER BY create\_date DESC
- 4. Ooops.



## SQL Injection

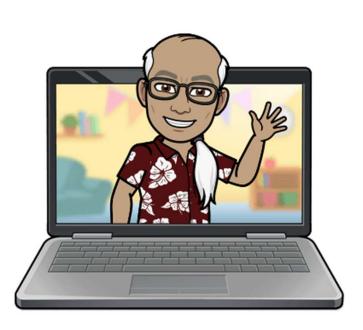
- 1. We ask the user for their input:
  - 1. Please enter your name:
  - 2. 'OR'1'='1
- 2. We use this input to query our database
  - String nameEntered = scanner.nextLine();
  - 2. String sql = "SELECT \* FROM message WHERE private = FALSE AND sender\_name = " + nameEntered + " 'ORDER BY create\_date DESC"
- 3. We think that should be:
  - SELECT \* FROM message WHERE private = FALSE AND sender\_name = "OR '1'='1' ORDER BY create\_date DESC
- 4. Ooops.



## **Practice Safe Computing**

- Parameterized Queries: If this is done consistently, SQL injection will not be possible
- Input Validation: Only allow certain values to be accepted. (Many of you did this in your Vending Machine)
- Limit Database User Privileges: A web application should always use a database user to connect to the database that has as few permissions as necessary. Never a good idea to use an admin's account

# LET'S CODE!





# WHAT QUESTIONS DO YOU HAVE?





# Reading for Tonight:

# DAO Part 2

