Jessica Calnan
Final Project
CS 340
06/07/2017

Outline:

I have decided to pivot from my original project proposal due to some difficulties that I ran into with entities/relationships with the previous topic I had chosen. In order to better fit the project requirements and to ensure that I have met all the criteria, I chose to create a database and website based on wines offered at various tasting rooms across the United States. (Although some of the information that I have used to prepopulate my database tables is accurate based on research, some of the information is fictional for the purpose of this project).

The site that I have created gives users the ability to interact with my database in many ways. Users can use the homepage to read a little bit about the site's purpose as well as to see a table (list) of the top 5 ranking wines stored in the database. From here, users can navigate to 5 different sections using the top navigation bar. They can search for a specific wine in the database based on type, color, and price, they can search a populated table of tasting rooms listed in the database, they can search for wine makers and their specs based on tasting room location, and they also have the option to add or delete wines or tasting rooms from the database.

Database Outline:

The entities that are included in my database are:

- Wines
- Locations
- Tasting Rooms
- Wine Makers
- Amenities

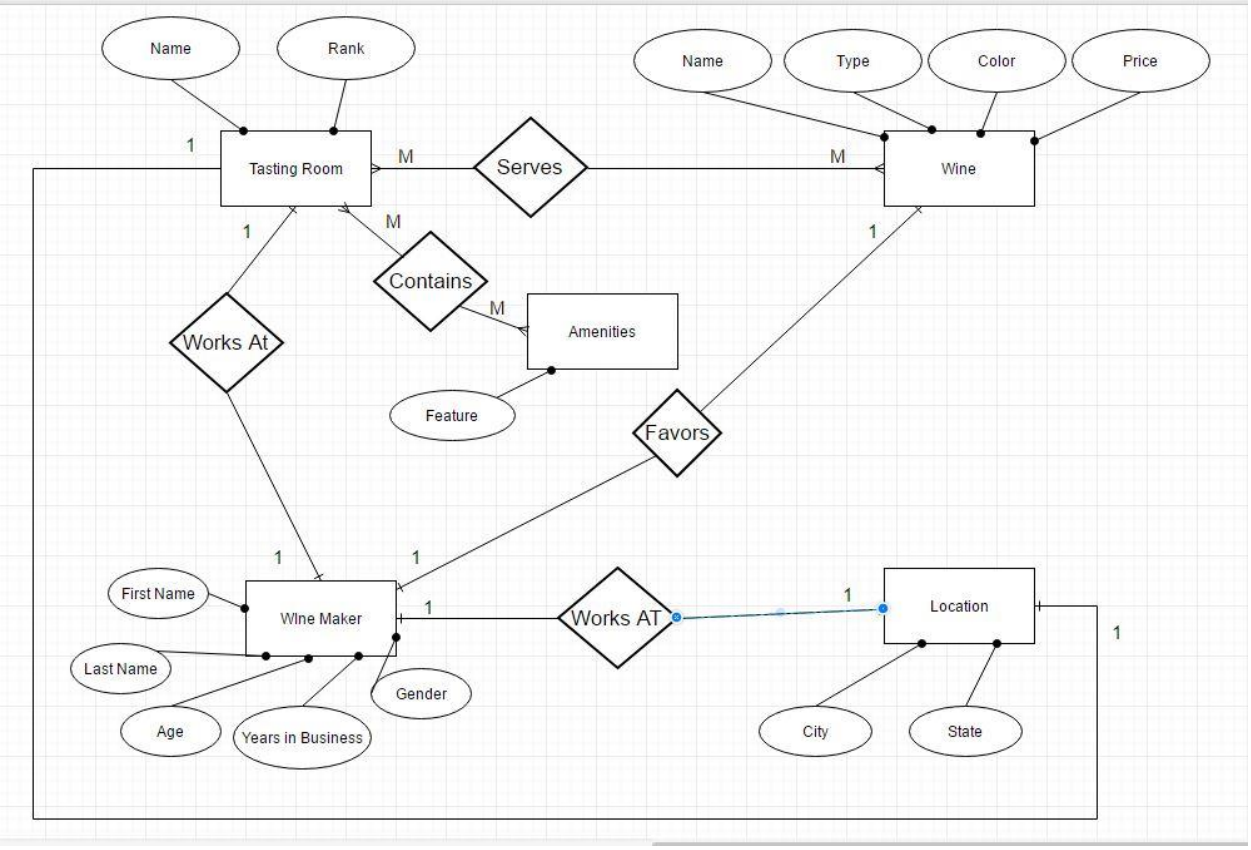
The properties that are included in my database are:

- Serves
- Contains
- Favors
- Works At
- Found At

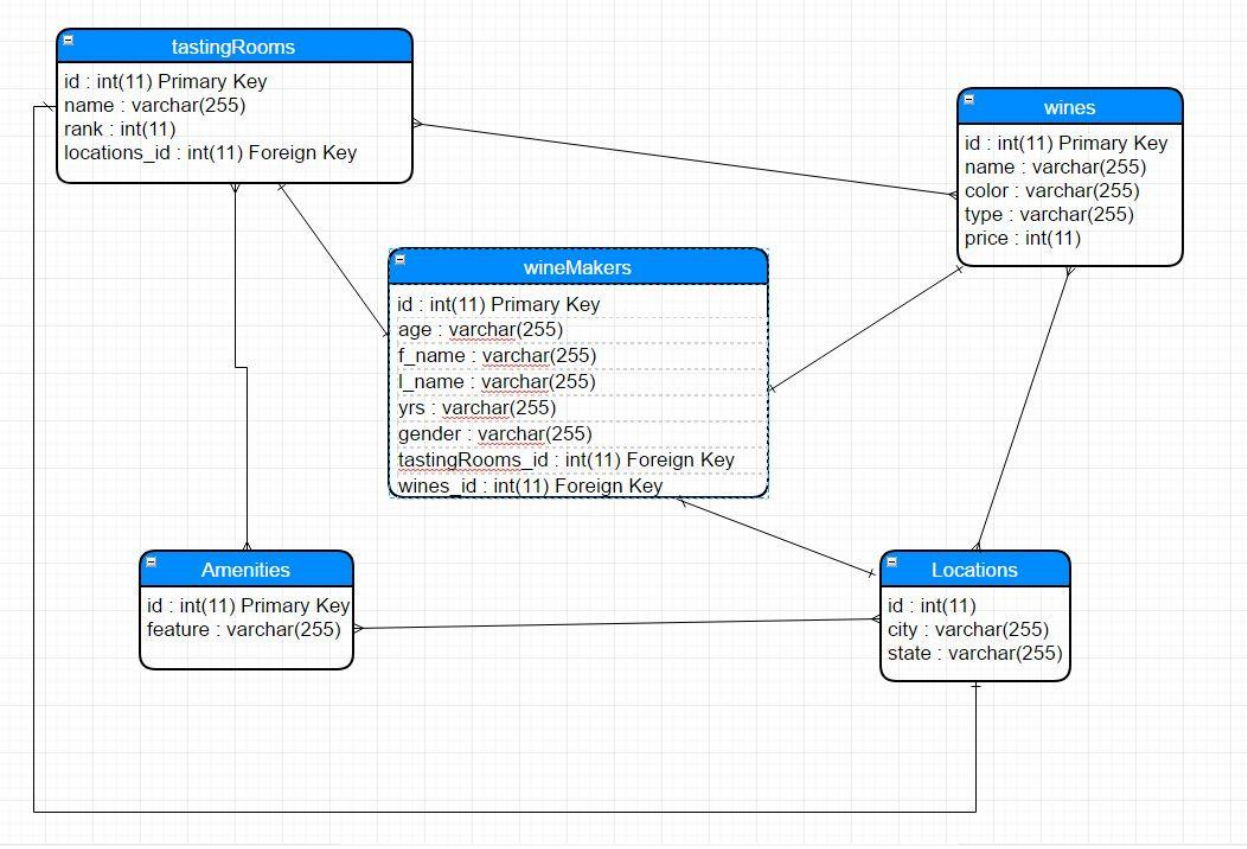
The relationships that are included in my database are:

- Wines are served at tasting rooms – each wine could be served at many tasting rooms and many tasting rooms could serve the same wines (Many to Many Relationship)
- Tasting Rooms each have one location (One to One Relationship)
- Wine Makers work at tasting rooms – tasting rooms can have one wine maker (One to One Relationship)
- Wine Makers work at one location (One to One Relationship)
- Wine Makers have a favorite wine (One to One Relationship)
- Tasting Rooms offer amenities – each tasting room could offer many different amenities, many different tasting rooms could offer the same amenities (Many to Many Relationship)

ER Diagram:


Name
Rank
Tasting Room
1
M
Serves
M
Name
Type
Color
Price
Wine
1
M
Contains
M
Amenities
Feature
Works At
1
1
Favors
1
1
First Name
Wine Maker
Last Name
Age
Years in Business
Gender
1
Works AT
1
Location
1
City
State

Schema:


tastingRooms
id : int(11) Primary Key
name : varchar(255)
rank : int(11)
locations_id : int(11) Foreign Key
wines
id : int(11) Primary Key
name : varchar(255)
color : varchar(255)
type : varchar(255)
price : int(11)
wineMakers
id : int(11) Primary Key
age : varchar(255)
f_name : varchar(255)
l_name : varchar(255)
yrs : varchar(255)
gender : varchar(255)
tastingRooms_id : int(11) Foreign Key
wines_id : int(11) Foreign Key
Amenities
id : int(11) Primary Key
feature : varchar(255)
Locations
id : int(11)
city : varchar(255)
state : varchar(255)

Data Definition Queries:

```
CREATE TABLE locations (
        id INT NOT NULL AUTO_INCREMENT,
        city VARCHAR(255),
        state VARCHAR(255) NOT NULL,
        UNIQUE (city, state),
        PRIMARY KEY (id)
) ENGINE = InnoDB DEFAULT CHARSET = utf8;

CREATE TABLE wines (
        id INT NOT NULL AUTO_INCREMENT,
        name VARCHAR(255) NOT NULL,
        color VARCHAR(255) NOT NULL,
        type VARCHAR(255) NOT NULL,
        price INT,
        PRIMARY KEY (id)
) ENGINE = InnoDB DEFAULT CHARSET = utf8;

CREATE TABLE amenities (
        id INT NOT NULL AUTO_INCREMENT,
        feature VARCHAR(255) NOT NULL,
        PRIMARY KEY (id)
) ENGINE = InnoDB DEFAULT CHARSET = utf8;

CREATE TABLE tastingRooms (
        id INT NOT NULL AUTO_INCREMENT,
        name VARCHAR(255) NOT NULL,
        rank INT,
        locations_id INT NOT NULL,
        PRIMARY KEY (id),
        FOREIGN KEY (locations_id) REFERENCES locations (id) ON DELETE CASCADE ON UPDATE
CASCADE
) ENGINE = InnoDB DEFAULT CHARSET = utf8;

CREATE TABLE wineMakers (
        id INT NOT NULL AUTO_INCREMENT,
        age INT NOT NULL,
        f_name VARCHAR(255) NOT NULL,
        l_name VARCHAR(255) NOT NULL,
        yrs INT,
        gender VARCHAR(255) NOT NULL,
```

```
        tastingRooms_id INT NOT NULL,
        wines_id INT NOT NULL,
        PRIMARY KEY (id),
        FOREIGN KEY (tastingRooms_id) REFERENCES tastingRooms (id) ON DELETE CASCADE,
        FOREIGN KEY (wines_id) REFERENCES wines (id) ON DELETE CASCADE
) ENGINE = InnoDB DEFAULT CHARSET = utf8;

CREATE TABLE contains (
        amenities_id INT NOT NULL,
        tastingRooms_id INT NOT NULL,
        PRIMARY KEY (amenities_id, tastingRooms_id),
        FOREIGN KEY (amenities_id) REFERENCES amenities (id) ON DELETE CASCADE,
        FOREIGN KEY (tastingRooms_id) REFERENCES tastingRooms (id) ON DELETE CASCADE
) ENGINE = InnoDB DEFAULT CHARSET = utf8;

CREATE TABLE serves (
        wines_id INT NOT NULL,
        tastingRooms_id INT NOT NULL,
        PRIMARY KEY (wines_id, tastingRooms_id),
        FOREIGN KEY (wines_id) REFERENCES wines (id) ON DELETE CASCADE,
        FOREIGN KEY (tastingRooms_id) REFERENCES tastingRooms (id) ON DELETE CASCADE
) ENGINE = InnoDB DEFAULT CHARSET = utf8;
```

Data Manipulation Queries:

//On Every Page
//PHP Code used to connect to the database

```
$mysqli = newmysqli("classmysql.engr.oregonstate.edu","cs340_calnanj",
"6902","cs340_calnanj");
      if(!$mysqli || $mysqli->connect_errno){
      echo "Connection error " . $mysqli->connect_errno . " " .
      $mysqli->connect_error;
      }
```

//To Add a Tasting Room

```
SELECT id FROM tastingRooms WHERE tastingRooms.name =
[tastingRoomName];

SELECT id FROM locations WHERE locations.city = [city] AND
locations.state = [state];

INSERT INTO tastingRooms (name, rank, locations_id) VALUES
([tastingRoomName], [city], [state]);

SELECT id FROM amenities WHERE amenities.feature = [feature];

INSERT INTO contains (amenities_id, tastingRooms_id) VALUES
([feature], [tastingRoomName]);
```

```
INSERT INTO locations (city, state) VALUES ([city], [state]);

INSERT INTO tastingRooms (name, rank, locations_id) VALUES
([tastingRoomName], [rank#], [location]);

SELECT id FROM amenities WHERE amenities.feature = [feature];

INSERT INTO contains (amenities_id, tastingRooms_id) VALUES
([feature], [tastingRoomName]);

SELECT feature FROM amenities;

SELECT name FROM tastingRooms;
```

//To Add a Wine

```
SELECT id FROM wines WHERE wines.name = [wineName];
INSERT INTO wines (name, color, type, price) VALUES ([wineName],
[color], [type], [price]);

SELECT id FROM tastingRooms WHERE tastingRooms.name =
[tastingRoomName];

INSERT INTO serves (wines_id, tastingRooms_id) VALUES ([wineName],
[tastingRoomName]);
```

//To Delete a Tasting Room

```
DELETE FROM tastingRooms WHERE tastingRooms.name = [tastingRoomName];
```

//To Delete a Wine

```
DELETE FROM wines WHERE wines.name = [wineName];

SELECT name FROM tastingRooms;

SELECT name FROM wines;
```

//To Display Top 5 Ranking Wines on Homepage

```
SELECT name, rank FROM tastingRooms
       ORDER BY rank DESC limit 5;
```

//To Display Database Info for Tasting Rooms

```
SELECT tastingRooms.name, tastingRooms.rank, locations.city,
     locations.state, GROUP_CONCAT(amenities.feature) AS feature
     FROM tastingRooms INNER JOIN locations
     ON tastingRooms.locations_id = locations.id
     INNER JOIN contains ON tastingRooms.id = contains.tastingRooms_id
     INNER JOIN amenities ON amenities.id = contains.amenities_id
     GROUP BY tastingRooms.name;
```

//To Search for Database Info for Wines

```
//User performs these queries using drop down menu options with
(database data options)
SELECT DISTINCT type FROM wines;

SELECT DISTINCT color FROM wines;

SELECT DISTINCT price FROM wines;
```

//To Search for a Wine in the Database

```
SELECT wines.name, wines.type, wines.color, wines.price,
     GROUP_CONCAT(tastingRooms.name) AS found_at
     FROM tastingRooms INNER JOIN serves
     ON tastingRooms.id = serves.tastingRooms_id
     INNER JOIN wines ON wines.id = serves.wines_id
     WHERE wines.type = [type] AND wines.color = [color] AND
     wines.price > [price lo] AND wines.price < [price hi] GROUP BY
     wines.name;

SELECT wines.name, wines.type, wines.color, wines.price,
     GROUP_CONCAT(tastingRooms.name) AS found_at
     FROM tastingRooms INNER JOIN serves
     ON tastingRooms.id = serves.tastingRooms_id
     INNER JOIN wines ON wines.id = serves.wines_id
     GROUP BY wines.name;
```

//To Search for Wine Maker by Tasting Room Location

```
//User does this with drop down menu of database "locations" options
SELECT city, state FROM locations;


SELECT tastingRooms.name AS TR_name, wineMakers.f_name,
     wineMakers.l_name, wineMakers.age, wineMakers.gender,
     wineMakers.yrs, wines.name AS fav_wine FROM wineMakers
     INNER JOIN wines ON wineMakers.wines_id = wines.id
     INNER JOIN tastingRooms ON
     wineMakers.tastingRooms_id = tastingRooms.id
     INNER JOIN locations ON tastingRooms.locations_id = locations.id
     WHERE locations.city = [city] AND locations.state = [state];

SELECT tastingRooms.name AS TR_name, wineMakers.f_name,
     wineMakers.l_name, wineMakers.age, wineMakers.gender,
     wineMakers.yrs, wines.name AS fav_wine FROM wineMakers
     INNER JOIN wines ON wineMakers.wines_id = wines.id
     INNER JOIN tastingRooms ON
     wineMakers.tastingRooms_id = tastingRooms.id;
```

Website Functionality:

http://web.engr.oregonstate.edu/~calnanj/finalProjPHP/home.php