Project #4: Working with Data

1 Add/Edit/Delete Data (.5 points each x 10 = 5 points)

Write SQL to accomplish the following tasks using the **World** sample database provided with MySQL. *Be aware that some of the commands you issue will intentionally or unintentionally change or destroy data.* Make sure you know how to recreate the sample databases, so you can get back into a known state. Also, before you do attempt destructive operations, run a SELECT query first so you know exactly what you're about to do (i.e., see what your WHERE clause filter will affect).

1.1 Try These

Attempt to these operations. When they fail[1], add a comment just after your query telling why SQL rejected the request. *Don't just regurgitate or paste the error message*; say in your own words what's going on and why it's appropriate and expected that MySQL would reject the request.

1. Add a row to the City table with ID = 1, Name = Foo, CountryCode = ABW, District = Whee, and Population = 100000.
2. Add a row to the CountryLanguage table with CountryCode = ZZZ, Language = Zebraese, IsOfficial = T, and Percentage = 34.2.
3. In the Country table, there's a row with Code = ABW. Delete that row.
4. In the Country table, there's a row with Code = ABW. Change that row's continent to Minionland.
5. Add a row to the City table, specifying no column names. Use these values, in this order: DEFAULT, 'Whoville', 'ABW', 'DistrictX', and NULL.

1.2 Do These

Write SQL to accomplish the following tasks:

6. You discovered the smallest incorporated city in Texas is Impact. It has a population of 43. Add a row for this city. Use the form of the command that supplies a column list; specify the fewest columns possible to fill in the specified data.
7. Write SQL to delete the row you created just above (be mindful of Safe Update Mode).
8. Now rewrite the SQL that adds Impact, TX, but write it *without* a column list. *In the same, single SQL statement* add four more rows with sample data of your choosing. Do not specify a specific value for the auto-incremented PK column; ask SQL to supply that.
9. Impact, Texas, had a new resident move into town. Write SQL that sets its population to 44.
10. A baby was born in Impact, Texas. Write SQL that increases its population by 1 *without giving a specific new population* (i.e., ask SQL to add one to what is already there).

[1] You'll need the latest schema and data to ensure these all fail as intended; get the World sample from the class site and run the query. Also, for some reason sometimes this setting has been disabled, so please enable it:

```
SET FOREIGN_KEY_CHECKS=1;
```

2 Data Types (.5 points each x 10 = 5 points)

2.1 Vinnie's Vehicles: Pick a Data Type and Size

For each of the scenarios below, select an appropriate column data type and size. Choose the most efficient (smallest) data type that suits the description (i.e., don't waste space needlessly) and use UNSIGNED wisely. For types where it makes sense (most types), follow the type with parentheses and the appropriate number(s), e.g., DECIMAL(8,2).

For each answer, write a SQL comment with the question you're answering and your response, e.g., #34, VIN Number = YEAR(4) (but of course, with a correct answer!). *Note: don't list multiple options and expect me to choose one to grade; YOU pick the best option and show it. If you want to discuss it, include that in a comment as well.*

11. **VIN Number**: each vehicle has a unique identifier associated with it. This is always 17 in length and consists of number and letters.
12. **MSRP**: the manufacturer's suggested price for the vehicle. At Vinnie's this won't be higher than $99,999. There are no "cents" associated here; dollars only.
13. **Estimated MPG, City**: the number of miles per gallon you may get when driving in the city. This is a number with no more than two digits, e.g., 34 or 51.
14. **Consumer Rating**: the rating given to this vehicle by consumers. This is a small number with one digit after the decimal place, e.g., 3.5 or 4.9. The maximum rating is 5.0, the minimum 0.0.
15. **Sold**: shows whether this vehicle has been sold.
16. **Make**: the "brand" or car, e.g., Ford or Mitsubishi.
17. **Color**: without getting specific to the manufacturer's crazy colors (e.g., "Champagne Pearl"), Vinnie's lists one of these colors for each car: White, Silver, Black, Yellow, Red, Green, or Blue.
18. **Options**: cars come with one or more of these options: Stereo Upgrade, Roof Rack, Mud Guards, Wheel Upgrade, Nav System.
19. **Sales Ratio**: gives the percent of sales this model car represents, e.g., 13.1 would indicate that this model car represents 13.1% of all of Vinnie's car sales. Vinnie cares about one digit after the decimal place. This number is used for averages and gets recalculated periodically.
20. **Purchase Date**: records the date and time when Vinnie acquired the vehicle.

3 Test Your Work

Where you're working with real databases and tables, test your work in MySQL and ensure it meets the requirements. Remember to think and double-check; just because the SQL runs and produces output doesn't mean it's the *right* output.

4 Submitting Your Work

Place comment headers over each SQL command, indicating what question it is answering. Submit the SQL file containing your work via Canvas.

5 Grading

You'll be graded on form and correctness, i.e., does your work produce the correct output and does it produce it in a straightforward and efficient way?