URL to see DATABASE http://flip1.engr.oregonstate.edu:7485/

Project Group 16
David LaMartina
Deborah Kretzschmar

Project Step 5 Draft Version: UPDATE Operations

Feedback from Project Step 3 Final Version: HTML + DML + DDL
None from grader

Feedback from Project Step 4 Group Review

Feedback #1
Wow, this looks really good! Here is some feedback ideas:

For the powers/weaknesses and the friendships/rivalries pages, I recommend removing the id numbers of the relationships, since the user doesn't care about them. I would also replace the character/weakness and character/power ids with their names by using an INNER JOIN query. It's a little cumbersome to reference the number and then look at the associated table to find out which superhero has which powers. The same is true for the friendship/rivalry table as well.

Also, I can add new stuff by leaving blank info. (I have entered a blank hero into your table). Should this be allowed?

I tried using the UPDATE, DELETE, and SEARCH BY functions, but I couldn't get them to work. I'm going to guess that they are the "work in progress" :D. Good luck with the rest of the project!
Cheers,

Feedback #2
Deborah,
The website looks great, and your CREATE and SELECT functions appear to be operating as expected, with one exception:
You can submit into your tables with empty strings for some of the NOT NULL attributes, like names. To help with this my partner and I used "required" in the HTML for the appropriate <input> and <select> tags in our forms. This will prompt the user if there's nothing entered that should be.
Otherwise, I think it would be worth consideration to populate the tables with Character's names where appropriate through the use of INNER JOINs.
Best of luck with the rest of the website!
Brian

Feedback #3

Deborah,

The functions that I tested out to add new information to your database all seemed to work as intended.

I like the way your tables are organized with in the web pages. I think having your forms hidden until the user chooses to add to the tables is a smart choice. It helps the pages from becoming cluttered since you have multiple tables on each page.

I'm going to echo what Ben and Brian have said about using an inner join to use names (character, city, weakness, etc.) rather than just id numbers. It is possible to look back at your other tables to reference what these correspond to, but it does become cumbersome.

You may also want to find a way to display what the role is instead of a 0/1 on the Characters table. The information given on the index page, "The "role" boolean value also indicates whether the character is considered a hero or villain"; doesn't exactly explain which one is a hero and which one is a villain. I was able to go back and look at your project outline to figure it out, but there is a lack of clarification on the site itself.

Feedback #4

David and Deborah,

The Superhero project looks incredible and super appreciate the organization this early on in our project. I think the search functionality makes sense and it satisfies the single query requirement as mentioned in the guide.

A minor suggestion is to edit the "Friendships and Rivalries (Links to an external site.)Links to an external site." page where it lists numbers instead of actual names of the alter egos. It might be easier for grading if the tables had a clear connection to the Character contents database. If this is not the intention that you had, you can disregard- only thought of it because of the requirement to satisfy "One should be able to add and remove certifications for a person without deleting either bsg_people rows or bsg_certification rows."

Likewise, in the "Character Weaknesses" there are only numbers listed and not names. Again just trying to make it more intuitive for the TA's. Overall, blown away by the progress and it looks like you're nearly there with the database front-end requirements.

Actions based on the feedback from Step 4 Draft

1. Every reviewer recommended that in our tables that represented a many-to-many relationship, Power/Weaknesses and Friendship/Rivalries had id numbers instead of names. They suggested replacing them with the names by using inner joins to make it more user friendly. We wholeheartedly agree with this recommendation and implemented several left join statements so that now our tables display user friendly names instead of id numbers for equipment, cities, or characters involved in a relationship.

2. A reviewer recommended changing the Boolean for the role of a Character to what it represented i.e. hero or villain. This change was made.
3. Two reviewers noted that our add form for entities allowed for blank entry where the input was not supposed to be not null. The recommendation was to add the “required” attribute to the form. This was accomplished. In addition, a javascript file was created to validate the number for Powers and Weaknesses so that it was not allowed to enter a new Power or Weakness unless there was a number between 1 and 10.

Other updates since Step 4 Draft

We have not had to make changes to our Outline, ERD, Schema, or DDQ. However, in order to implement the filter and search functions and also populate our tables with more meaningful data for the user instead of id numbers, we had to add several left join statements and include select statements with where clauses in our DMQ file.

Feedback by the reviews from Step 3 Draft

Feedback #1
After looking through all your components for this assignment, it seems like you two have everything covered. You included each of the necessary manipulation queries that are required for the project and your website is really awesome. I like how you have it step up and just from going through your queries and website I feel like I learned a ton, so thank you! The only minor thing I notice was when creating tables and dropping if existing you sometimes included backticks and other times did not. I’m not sure if that matters at all because I was able to successfully import your sql file. Sorry I don’t have much feedback, you two did such an awesome job!

Feedback #2

Great job as usual, everything is very thorough I just have one technical point to add:

I originally also used CHECK in mySQL to try to have numbers of certain ranges, unfortunately I found that the MariaDB version we’re using(10.1.22) does not actually enforce the CHECKS, see https://mariadb.com/kb/en/library/constraint/#check-constraints (Links to an external site.) Links to an external site.

From MariaDB 10.2.1, constraints are enforced. Before MaraDB 10.2.1, constraint expressions were accepted in the syntax but ignored.

I tested this on your Power table and it is indeed true, I inserted a row with power magnitude 20 despite the <11 CHECK:
MariaDB [cs340_newtoner]> INSERT INTO Power (power_type, power_magnitude) VALUES ('x-ray vision2', 20);
Query OK, 1 row affected (0.00 sec)

```
MariaDB [cs340_newtoner]> select * from Power;
+----------+----------------------+----------------+
| power_id | power_type | power_magnitude |
+----------+----------------------+----------------+
| 1 | x-ray vision | 10 |
| 2 | super strength | 5 |
| 3 | super strength | 10 |
| 4 | flight | 5 |
| 5 | genius-level intellect | 10 |
| 6 | x-ray vision2 | 20 |
+----------+----------------------+----------------+
6 rows in set (0.00 sec)

MariaDB [cs340_newtoner]>
```

I personally just gave up on this since the CHECKs weren't critical for my project but do let me know if you find a work around. Could be a good question for the instructor how we should do this with this MariaDB version.

Feedback #3

Deborah,
I have really enjoyed seeing the progress of your project. It's a fun idea to make a database out of.
I think the site that you have created to display the information from the database looks really good. Since there are so many different entities and relationships that you have the ability to add new information to, it is a good idea to have all of the forms hidden to avoid clutter. However, having the "Hide Form" button still there even when the form is hidden is a little confusing. Maybe consider hiding the button while the form is hidden, just to make it a little more clear.
The groupings that you've used to put the different tables together makes sense. Changing the heroes and villains into one characters entity made sense to do, but you could still add the ability to filter by hero or villain on the Characters, Equipment and Cities page. This would give the user a way to see the entities that you originally intended to create.
Great job so far!
Edited by Joshua Nicholson on Feb 13 at 6:07pm

Feedback #4

First of all your website is really well put together and definitely looks like effort was put into the aesthetics which I appreciate a lot! I like the navbar at the top and found that it was easy to make my way around the site.
I do agree with the other reviewer that you should look into a different option for the hide button as it definitely confused me at first, possibly just one button that toggles the add menu instead of two?
Looking through your .sql files and your project outline I couldn't find anything that stood out to me as particularly wrong. Though I would recommend throwing comments into you ddq file as it would just make everything a little clearer. Otherwise it seems like you are really killing this project!

Feedback #5

Hi Deborah,
The website looks great! I like how your group included explanations about what each page will display to the user. Having the sample data filled in also helps to better visualize how you'll use the data from your database. I see that you have a search functionality available on the Characters, Equipment, and Cities page, but one suggestion might be to have a page dedicated to allowing the user to search for a single superhero or villain and then present all of the associated info about that particular person. (i.e. info from all tables).
In your DDL file, everything is laid out as expected and I didn't see any syntax errors. I only saw discrepancies in the use of backticks for the table names of Character and Power. Obviously that's not egregious, but for uniformity's sake maybe just pick one and stick to it. Also, with using the name Power for a table, I see that it is a protected keyword (Links to an external site.)Links to an external site., so just be aware that might cause unintentional errors in certain cases. My partner and I ran into a similar issue with our "user" table and renamed it to "db_user".
In your DML file I would caution against using SELECT * queries. I know Wolford mentions this in some of the lecture videos, but this (Links to an external site.)Links to an external site. stack overflow page shows why it could be detrimental to use.Granted it's not likely our databases will be so big or that we continue using these databases beyond this class, but if you do, then just give consideration to what you're losing with using *.
I noted the part in your feedback response about the arrows in the schema. My mistake, I'll also be sure to make the correction in my own project outline =). The rest of your responses to feedback made sense. I can't wait to see your finished product!

Actions based on the feedback from Step 3 Draft

1. Reviewer #1 and #5 have mentioned the back ticks we use because Character is a reserved word and Power is a protected keyword. We carefully considered their comments and chose not to change the names at this point. However, in order to provide consistency for using backticks for table names, we are now using a mysqldump file for our Data Definition File which resolves the issue of backticks by automatically inserting them around all table names.
2. Reviewer #2 brought up a problem with our current version of MariaDB ignoring CHECK clauses. The MariaDB even ignores CHECK clauses when a file is exported. We asked about this issue on Piazza @ 134. Professor Hedaoo responded: “So for this step draft and final versions, it’s okay if you include CHECK constraints in your SQL file. When we come to the relevant steps in coming weeks, you can implement that using the back end programming language”. So since our instructor is aware that CHECK constraints will be ignored by the current version of MariaDB on our server, we will follow his advice and implement the necessary constraints with our back end programming language per Piazza @134.
3. Reviewer #3 and #4 , felt that having the "Hide Form" button still there even when the form is hidden is confusing. They recommended hiding the button while the form is hidden, just to make it a little more clear. We agreed and implemented a change. The hide form button was removed. Instead, there is only one button that either says what the function does, or when it is clicked, the text is changed to

"hide form". So we implemented a toggle to address the issue of the confusion two reviewers found with our buttons.

4. Reviewer #4 recommended inserting comments into our files. Our data manipulation file has comments and by using an export for our data definition file, it automatically inserts comments.
5. Reviewer #3 and #5 made suggestions on different types of filters that we can do. We certainly appreciate their suggestions and may use them when when implement our front-end JS. However, for now our filter/search capabilities meet requirements.
6. Reviewer #4 recommended inserting comments into our files. Our data manipulation file has comments and by using an export for our data definition file, it automatically inserts comments.
7. Reviewer #5 cautioned against the use of select * as being a bad practice. We can see the reviewer's point; however, given the size of our database, our goal is to display everything (at first), and then let the user filter as desired. Should our tables become so large as to be unwieldy, we will consider the following solution. We won't display anything until the user clicks some "display" button, which takes into account whatever filters they have or have not put in place. This button could be a kind of on / off switch, which allows the user to get the data back OFF their screen if they're overwhelmed. Again, with our small database, we do not anticipate a problem with efficiency and indexing.

Upgrades to the Draft Version

We did not make any additional changes other than those noted above.

Feedback from Grader from Step 2 Final

None

Feedback by the peer review from Step 2 Draft

Feedback #1

Superheroes and Villains Database ERD

Real_city and real_name - these categories sound similar, should real_first_name and real_last_name also have a boolean? What indicates that the names are real or alter egos?

Schema

I understand the relationships between rivals and friends, but how does this differ from the mentors/helpers relationship? Would it be useful to have a helper_id and make it a relationship instead?

Overall, I think it's a really detailed database and you both put so much work into it. I think the final project will really impress our instructor based on this initial outline. Great job!

Feedback #2

Hi David and Deborah, wonderfully detailed project. I would however second the confusion above with the different mentor/helper/friend/rival relationship. I think there is probably some way to simplify here, I haven't read much about recursive relationships yet but I think in many cases it may be best to eliminate them (http://www.users.csbsju.edu/~lziegler/CS331/Logical4.html (Links to an external site.)Links to an external site.). But even if not, there's definitely a lot of overlap in categories here, for instance, isn't a helper basically a friend? There's also some relationships I'm not seeing in the schema such as that there may be 0 or more helpers for each character, there doesn't seem to be a helper field beyond the mentor_id for the character so I'm not sure how this 0 to many relationship could be implemented.

Other than simplifying/clarifying those relationships, great work!

Feedback #3

Hi Deborah,

Overall I think your group has a great idea for a database.

- The explanations for each entity, relationship, and attribute made sense and were made with adequate detail.

- The ER diagram accurately reflects the relationships as you explained them and the notes for the recursive relationships also help clarify what's going on.

- The schema format makes sense, however I think the direction of the arrows pointing toward the character_id attribute should be reversed. This may be a nitpick, but to me it reads like the character_id attribute is derived from those entities or relationships, which we know is not the case. Everything else about it make sense.

- I'm on the fence about your decision to get rid of the sidekick/henchman attributes/relationships. To me it makes sense to have those attributes specifically defined in addition to the mentor attribute. Admittedly my experience and ability with databases are extremely limited but I struggle with how you would define who's a mentor to whom and who's a helper.

Overall I think you have a great idea and plan!

Feedback #4

I enjoyed reading the plan for your database.

- I like the decision to turn Superhero and Villain entities into one Character entity. Since these characters have the same attributes, it makes to combine them.

- I like the color coding in you schema. It makes the difference between your entities and relationships that much easier to recognize. It looks like it follows entity-relationship diagram properly.

- The only question that I have relates to the equipment entity. Your example " Thor has a hammer and

Wonder Woman has a lasso" makes it seem like the values for the entity will be generic equipment types. If that is the case, it doesn't seem like the one to many relationship is appropriate. For example, a generic equipment type could be a bow. Two heroes that would both connect to a bow are Hawkeye and Green Arrow.

Overall, I think it is a very good plan.

Actions based on the feedback

1. We chose not to change the Character attributes real_first_name and real_last_name to Booleans because we felt that the name and data type of a varchar indicates that we are specifying what the actual first and last names are of the Character. For example, Superman's real first name is Clark and his real last name is Kent. Cities used in comic books do not have alternate names. Thus all the boolean for real_city signifies is whether or not the city is fictional. The definitions for the attributes for the attributes are defined in our outline.
2. We did not add a helper_id to the Character entity because the helper relationship between a mentor and another character is a recursive one-to-many relationship. A character can be a helper to at most one mentor but a mentor can have many characters that they mentor. Adding a helper_id to the Character entity would not change that this is a recursive relationship between two characters.
3. We did indeed simplify the recursive friends and rivals many-to-many relationships by creating relationship tables in the schema.
4. Another reviewer was concerned that by not having a helper_id we would not be able to enforce the relationship that a character can be a helper to at most 1 mentor. The reason for not having a helper_id was explained in #2 and the cardinality and participation of the relationship will be enforced by the definition of the Character mentor_id which is included in our outline. It is as follows: "this represents the id of the character who is the mentor for this character. It may be blank and there is no default. If this character is a helper, then the mentor must have the same role as this character. For example, a villian can only be mentored by a villain. Also the mentor_id cannot be equal to the character_id."
5. With regards to the direction that the arrows are pointing, we did not change them because it is our understanding that the arrow should point from the foreign key to the entity that it is referencing.
6. One reviewer had confusion about our Equipment entity and that it seemed to describing generic equipment instead of being specialized for each individual superhero or villain. We decided to add clarity by adding a name attribute and changed the name of the attribute equipment_type to description.
7. Since one reviewer was concerned about the overlap between helpers and friend_relationships and also had some difficulty understanding the relationship, we changed the name from helper to assistant. The idea is that a more descriptive name will help the reviewers visualize that an assistant has a subordinate position to their mentor or leader. An example would be Batman and Robin where Batman is the mentor and Robin is the assistant. We also added text that explains that an assistant may be a friend with their mentor i.e. the two relationships may coexist.

Upgrades to the Draft version

We made no additional changes other than what is described above

Fixes based on Feedback from Step 1

We did not receive feedback from our grading TA.

Based upon our own internal review, we made several changes to streamline our database. By consolidating some of our entities, we feel that the entities and their relationships will be more effectively displayed. Below are the changes.

1. Originally, we had a Superhero and Villain entity. We removed these two entities and combined them into one entity called Character. We then added an attribute called role which is a Boolean. This Boolean will be true if the character is a superhero and false if the character is a villain.
2. Originally, we had a Sidekick and Henchman entity to show that a superhero or villain could be a sidekick or henchman, or they could be a mentor to another superhero or villain. To better display this recursive relationship, we removed both the Sidekick and Henchman entities and added another attribute to the Character entity. The new attribute is called mentor_id. It represents the id of the character who is the mentor for the character.
3. With the changes to the entities, we rewrote the original relationships for Superheroes and Villains. These relationships were combined to reflect the relationships Characters have with the entities Power, Weakness, Equipment, and City.
4. Since there is no longer a separate Sidekick and Henchman, we removed the separate relationships between a Superhero and Sidekick and a Villain and Henchman. We replaced this with a helper relationship. The attribute mentor_id in the Character entity is now a foreign key to the character_id. This helper relationship is a recursive one-to-many relationship.
5. We replaced the Superheroes can be friends with other superheroes and villains can be friends with other villains relationships with a recursive many-to-many relationship.
6. We replaced the Superheroes and Villains are rivals relationship with a recursive many-to-many relationship.
7. With the changes, we now have 5 entities, 2 one-to-many, 2 many-to-many, and 3 recursive relationships as described above.

Project Outline and Database Outline - Updated Version

Superheroes and Villains Database

Project Outline

This database will represent the Marvel and DC Comic Book Characters and includes their roles as superheroes or villains. It specifically refers to characters, not teams that may be formed such as the *Avengers* or *Fantastic Four*. Representing these characters, their special powers, weaknesses, weapons, and relationships to other characters will add complexity to make an excellent database project.

Database Outline, in Words

The entities in the database are:

- **Character** -- Characters is the most important entity. It has the following attributes:
    - **character_id**: Automatically assigned to each character when they are entered into the database. It is an auto-incrementing integer that is the primary key.
    - **character_name**: Name of the character is a string of maximum 100 characters. It cannot be blank and there is no default. The name is taken as a whole and is not further broken down. For example, *Ghost Rider* or *Luke Cage* are considered as a whole string.
    - **real_first_name**: Characters have alter egos or real names. This is the first name. It is a string of maximum 100 characters. It can be blank and there is no default.
    - **real_last_name**: Real last name of the character. It is a string of maximum 100 characters. It can be blank and there is no default.
    - **city:** Name of the city where the character lives most of the time. It can be blank and there is no default. If present, the value is the city_id from the City entity.
    - **role:** This is a Boolean that represents whether the Character is a superhero or a villain. True means that the Character is a superhero and false means that the Character is a villain. The role cannot be blank and there is no default.
    - **mentor_id**: this represents the id of the character who is the mentor for this character. It may be blank and there is no default. If this character is an assistant, then the mentor must have the same role as this character. For example, a villian can only be mentored by a villain. Also the mentor_id cannot be equal to the character_id.
- **Power** -- An ability that is beyond the capability of normal humans. The different powers that a superhero or villain can have are recorded here. It has the following attributes:
    - **power_id:** Automatically assigned to each power when it is entered into the database. It is an auto-incrementing integer and is the primary key.
    - **power_type**: type of power the character has. Examples are x-ray vision, super strength. It is a string of maximum 100 characters. It cannot be blank and there is no default. It is evaluated as a whole even if the string uses more than one word.
    - **power_magnitude**: A relative rating of the power that uses an integer scale rating of 1 to 10 with 1 being the weakest and 10 being the strongest. It can be blank and there is no default.
- **Weakness** -- This is an element that can weaken the powers of either a superhero or villain. It has the following attributes:
    - **weakness_id**: Automatically assigned to each weakness when it is entered into the database. It is an auto-incrementing integer and is the primary key.
    - **weakness_type:** type of weakness the character may possess. For example, Dare Devil's weakness is noise pollution. It is a string of maximum 100 characters. It cannot be blank and there is no default. It is evaluated as a whole even if the string contains more than one word.

    - **weakness_magnitude**: A relative rating of the weakness that uses an integer scale rating of 1 to 10 with 1 being the weakest and 10 being the strongest. It can be blank and there is no default.
- **Equipment** – A special piece of equipment or weapon that augments the abilities of a superhero or villain. For example, Hawkeye has specialized arrows and Wonder Woman has a truth lasso. This equipment belongs to only one character.
    - **equipment_id:** Automatically assigned to each piece of equipment when it is entered into the database. It is an auto-incrementing integer and is the primary key.
    - **name:** Some equipment has specific names. For example, “Mjollnir” is the name of Thor’s hammer or Batman’s car is called the “Batmobile”. It is a string of maximum 100 characters. It can be blank and there is no default. It is evaluated as a whole even if the string uses more than one word.
    - **description:** Description of the specialized equipment. It is a string of maximum 100 characters. It cannot be blank and there is no default. It is evaluated as a whole even if the string uses more than one word.
    - **materia**l: The dominant material the equipment is made of. It is a string of maximum 100 characters. It can be blank and there is no default.
    - **character_id**: the id from either the superhero or villain that has this equipment. It cannot be blank as the specialized equipment would not exist without the superhero or villain. There is no default.
- **City** -- This is the city where the superhero or villain resides. It can be real or make-believe
    - **city_id**: Automatically assigned to each city when it is entered into the database. It is an auto-incrementing integer that is the primary key.
    - **city_name**: Name of the city. It is a string of maximum 100 characters. It cannot be blank and there is no default. It is evaluated as a whole even if the string contains more than one word.
    - **real_city:** This is a Boolean that is represented by an integer of either 1 or 0. True means the city is real . False means the city is make-believe. This cannot be blank, and the default is true .

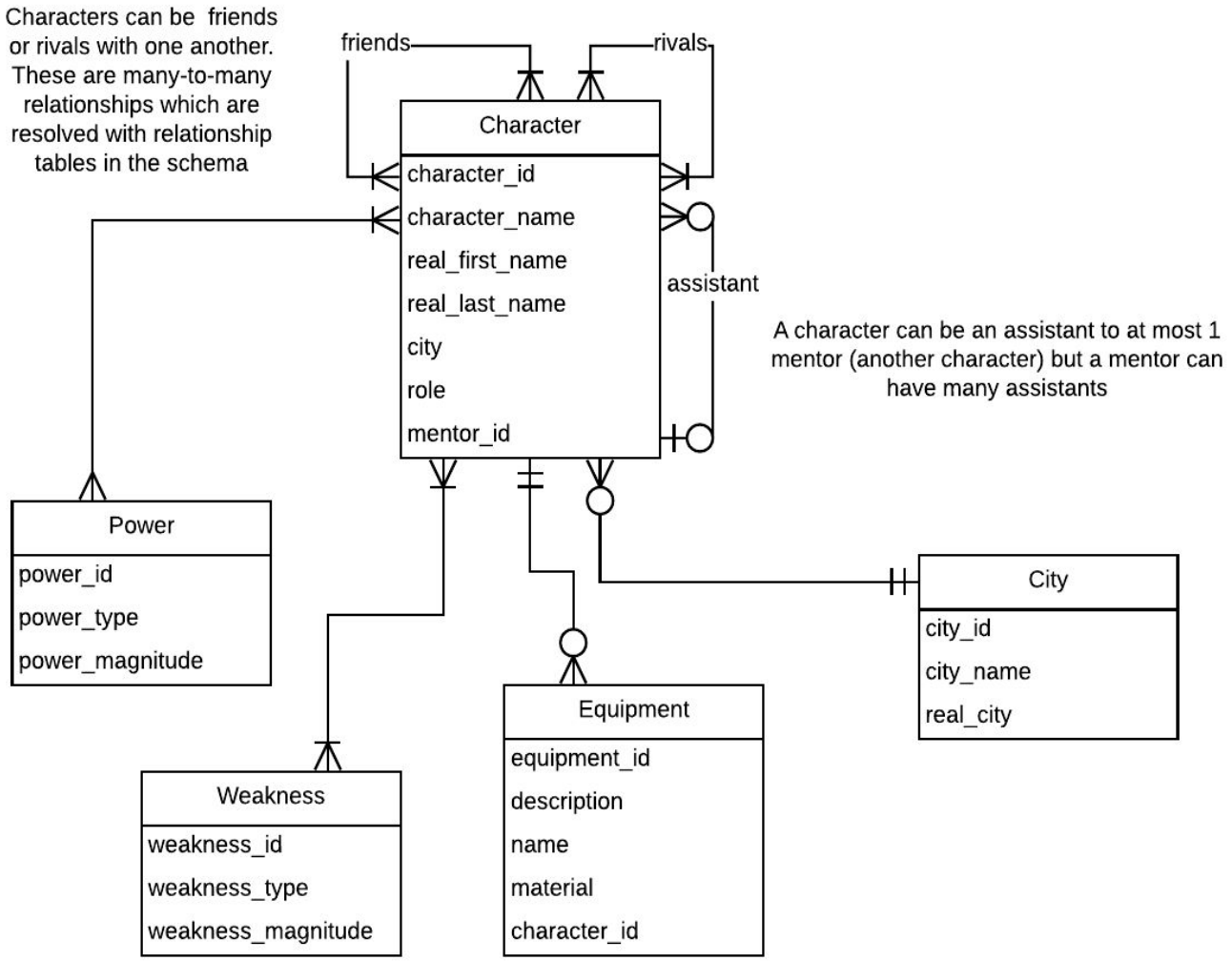
<u>Relationships in Database</u>

The relationships in the database are:

- **Characters have powers** - A character can have one or more powers and powers can belong to many Character. The Character and Power entities are in a many-to-many relationship.
- **Characters have weaknesses** - A character can have many weaknesses and weaknesses can belong to many Character. The Character and Weaknesses entities are in a many-to-many relationship.
- **Characters have equipment** - A Character may have many special types of equipment, but a special type of equipment must belong to one and only one character. This is a one-to-many relationship.

- **Characters are from cities** - A character lives in only one city, but a city can have many superheroes. So, the City and Superhero entities are in a one-to-many relationship.
- **Characters have assistants** – A character can have many assistants, but a character can have at most one mentor. An assistant is subordinate to their mentor i.e. the mentor functions as the leader. An example would be Batman and Robin where Batman is the mentor and Robin is the assistant. This is a recursive one-to-many relationship because both the mentor and assistant are characters.
- **Characters can be friends with other characters** – This is a recursive many-to-many relationship because each of the two characters involved in the relationship are from the same entity. Each character can be in many different friendships. This includes that an assistant may be friends with their mentor. This many-to-many relationship will be resolved in the schema with a relationship table.
- **Characters can be rivals with other characters** - This is a recursive many-to-many relationship because each of the two characters involved in the relationship are from the same entity. Each character can be in many different rivalries. This includes that an assistant may be rivals with their mentor. This many-to-many relationship will be resolved in the schema with a relationship table.

Entity-Relationship Diagram is on next page



Schema is on next page

SCHEMA FOR SUPERHEROES AND VILLAINS DATABASE

Note: blue-banded tables represent entities and green-banded represent relationships