CS340 Project – Marvel Database

URL: http://flip1.engr.oregonstate.edu:9401/

Feedback

Step 1

Feedback by peer reviewer: On a personal note, I appreciate the fact that you're doing a Marvel Database. I'm a fan of the Marvel movies, but I don't read the comic books, so I frequently find myself turning to online resources to research, say, the limits of the Hulk's strength or the history of Wakanda. General Outline Feedback (The numbers reference the questions from the "How to peer review Project Step 1" page) 1) Yes—the outline clearly establishes the universe that the database is set in (the MCU). 2) No—what's presented is sufficient. Database Outline Feedback (The numbers reference the questions from the "How to peer review Project Step 1" page) 1) Yes—4 entities are presented. However, 3 of the entities (Teams, Ability and Planet) do not have any attributes listed. 2) Yes—there are 4 relationships, and two of them are many-to-many. 3) Yes—I think the entities are well chosen and cannot just be attributes of other entities. 4) Yes—the relationships do make sense to me. The fourth relationship (Characters are the nemesis' of other characters) references superheroes and villains. It wasn't entirely clear to me if the distinction between superheroes and villains references the affiliation attribute under character, so perhaps this could be made explicit. If it doesn't, then perhaps a Superhero/Villain Boolean type attribute could be added to Character. 5) Yes—there is sufficient information about each attribute. 6) No—data types are not mentioned. 7) N/A—No data types are given. 8) Yes—constraints are described for the attributes and they do make sense. 9) As I mentioned above, attributes should be listed for the remaining 3 entities. Further, data types should be listed for all of the attributes. As side note (and this is just a suggestion, not actionable feedback): it would be cool to capture, somehow, the fact that a given superhero role (Captain Marvel, for example) has been filled by multiple people over time. This would require a pretty significant overhaul to your design, though.

Actions based on feedback and upgrades to draft version: I did not receive feedback from the grader, I did however receive feedback from two of my classmates. Based on the peer feedback and my own tinkering, I made some minor changes to my attributes and added a new villain entity.

Step 2

Feedback by peer reviewer (1)*:* It reads as characters must have teams, but I am assuming it should be teams must have characters to exist. I am unsure about the character/ability relationship too, but I would think characters must have an ability which would mean total not partial participation. Many-to-many relationships should be represented with m and n. There are no ; at the end of the create table statements. There are additional commas and missing in the create table statements. Some datatypes are misspelled. ` ` are missing in the insert statements.

Feedback by peer reviewer (2)*:* The ER Diagram should have attributes shown as circles connected to each respective entity. The cardinality and participation look correct and are consistent with the outline. Schema: The schema is missing. I believe there are two ways you can do this if you check under symbols and keys example in one of the modules on canvas. Should be tables for each entity and relationship. Should have relationships with an attribute(s) that points to another tables key (i.e. a foreign key). I think you can add another entity or relationship Nemesis that could have attributes super_hero and nemesis and these each could have foreign keys defined that reference a specific character in the marvel_character table (also you might want to keep what you call entities and attributes the same for your outline, ERD, Schema, and SQL queries for your final draft). Characters as members of teams should also be another table in the schema, with possible foreign keys referring to marvel_character 'id' and marvel_teams 'id'. The sql queries when creating tables with foreign keys would look something like this (disregard if you already know, there's also good explanation in Piazza): Ex: Nemesis_Relationship table CREATE TABLE `Nemesis_Relationship` (`id` int(11) NOT NULL AUTO_INCREMENT, `super_hero` int(11) NOT NULL, `nemesis` int(11) NOT NULL, PRIMARY KEY (`id`), FOREIGN KEY (`super_hero`) REFERENCES `marvel_character` (`id`), FOREIGN KEY (`nemsis`) REFERENCES `marvel_character` (`id`)) ENGINE=InnoDB; DDQ: You can put some DROP TABLE IF EXISTS statements first thing before you create a table. In case you want to easily fix something, might be easier to drop it first and recreate it. It might make sense to drop all tables at once, especially if there are foreign keys. Ex: DROP TABLE IF EXISTS `marvel_character`; Also, when creating tables you want to make sure they are in the right order, tables with foreign keys should be after the table that's being referred too. I think it's similar when dropping tables. Also, when inserting into tables, you don't have to do each attribute one by one. Ex: Insert Tony Stark INSERT INTO `marvel_character`(fname, lname, species, age, alias, affiliation, power-level) VALUES ('Tony','Stark','human',43,'Iron Man',1,90); This would be easier to quickly copy and paste each, unless you are doing it programmatically? Also watch out for putting quotes for something you called an int. I would also double check this for binary value 1 or 0. I saw a couple of errors like these in your queries. Overall, I love your database idea! I'm a big fan of the marvel universe. I think the front-end component of this will be awesome! My main advice would be to see if you can add a couple more of those relationship tables and foreign keys.

Actions based on feedback and upgrades to draft version: I did not receive feedback from the grader, I did however receive feedback from two of my classmates. Based on the peer feedback and my own tinkering with the project, I made several somewhat significant changes to my overall design. I added an entity for Villains to help with relationships with the hero. In the ERD, I added m/n and 1's to the relationship lines. I also fixed syntax errors. I added the schema which was missing from my draft and fixed my sql queries and uploaded them to myphpadmin.

Step 3

Feedback by peer reviewer: You could consider changing the heroes-villains-enemies relationship to a many to many relationship. A villain can have many hero enemies and and a hero can have many villain enemies. This would involve a little rework of moving the enemy attribute off of villain and create a new relationship table, but I think it would be a worthwhile

addition. The other option might be just having an arch nemesis for each hero and villain--but are they always the same? Maybe you could add a first appeared in attribute to heroes and villains. You could also do the same for teams, and can villains be on teams? I wonder if the love interest is another hero or villain does the attribute reference that entity or are you trying to keep that part simple? I can see how that could get very complicated quickly. Planets entity incorrectly labeled as Abilities in the outline, simple typo. Maybe you could add an intensity attribute to abilities so that two characters with the same ability could be compared, which I see you have an attribute named plevel in your ER diagram and that's probably what that is, but it is also missing from your schema diagram. It would be cool if you added an equivalent to the villain entity. I'm going to look at your sql now! Data definition query loaded with no problems, and the sql is very neatly formatted. The data manipulation are appropriate. You're just missing some update and delete queries that I'm sure you're working on. Be sure to include a query to reflect a search or filter function as well. Your index page looks complete and ready to get populated with some awesome super hero data!

Actions based on feedback and upgrades to draft version: Added delete queries and a filter query. Fixed typo.

Step 4

No feedback

Actions based on feedback and upgrades to draft version: None.

Step 5

Feedback by peer reviewer:

READ functionalities

Q - Are rows being listed for all entities?
A - Yes rows are being displayed in rows. However number indicators could be helpful.

Q - Are rows being listed for all relationships, as described in the Specs?
A - Unable to utilize relationship functionality. No rows displayed.

Q - Is the Search OR filter functionality present and working?
A - Search functionality is currently not working. Cannot see any statistics associated with the search field. Search functionality is not a separate page and does not display requested information. an improvement can be made by separating this functionality from others.

Q - Is there a better way that data could be displayed on these pages? OR Could the style of the webpage be improved?
A - yes, the page is limited by aesthetic implementation. Isolation of features could be a nice feature to add. Linkage to other pages in the overall layout would be a welcomed feature. Navigating is accomplished by the forward and back buttons. Better layouts to each field would also be a welcome addition (indentation of fields), better alignment, and/or spacing between fields and data entry.

DELETE functionalities
Q - Are rows being listed for all entities?
A - Yes rows can be seen. Delete functionality of the delete button does not work.

Q - Are rows being listed for all relationships, as described in the Specs?
A - Same as read functionality above. Cannot see delete functionality for new heroes added since additions are not present.

Q - Is there a better way that data could be displayed on these pages? OR Could the style of the webpage be improved?
A - See comments in read functionality

Actions based on feedback and upgrades to draft version: Added linkage to other pages and updated DELETE functionalities properly. Fixed search functionality (minus CSS implementation).

Step 6

Feedback by peer reviewer (1): It seems like the update works! I updated a hero and it reflected on the other pages. After testing each one, all the updates seem to work and the changes are reflected on other web pages. I also like that when you click update, the boxes are prefilled with what the previous values were. Saves a lot of typing! I am also able to update and delete, and make new relationships with updated entities. Overall it seems to already be working 100%!

Feedback by peer reviewer (2): Hi Cory, the website is looking really cool! I like the index page carousel. The page functionalities are also working very well. Creating and deleting rows is easy and effective for each entity and updating, similarly, is working very well. One issue is that creating a new hero goes into the database backend ok but the page hangs/doesn't render the new row without manually refreshing the page--I'm sure this could be an easy fix for you. Overall, very nice work this week! Fun, cool project!

Feedback by peer reviewer (3): Hey! Looks like your update functionality is working! It didn't affect a relationship, but this is as intended I believe. Also, adding or deleting also doesn't affect the relationship and the updated entity, as intended. Overall looks great! There's not much to say.

Feedback by peer reviewer (4): First off, very cool project!
1. Is the UPDATE functionality properly implemented for at least one entity? Yes, works great!
2. What is the effect of this UPDATE on the relationships that the entity is participating in? Everything seems to be in good order where UPDATING is concerned!
3. What is the effect of CREATE and DELETE operations on the other entities that are participating in the relationship with this entity? CREATE doesn't seem to have any advers effects. However, if I delete a planet, then it will delete all the Heroes on that planet. Maybe include a warning message if this is intentional.

4. Anything else that you think is important for the UPDATE functionality? Nothing to change here I think, but I like how you have updates available for many of your entities. Good work!

Actions based on feedback and upgrades to draft version: none.

Project Outline

Summary: I will be making a database representing supernatural humans, humanoids, aliens, and other creatures from the Marvel Comic Universe. This is a universe with superheroes and villains. The database will contain beloved characters and lesser known ones. Anyone who is familiar with movies like Iron Man or Spider-Man will understand this universe well.

Database Outline

Entities

Heroes – This entity holds the heroes in the database, is involved in a relation with every other entity, and has the following attributes:

hero_id: This attribute holds a number that is automatically assigned to each hero when they are recorded in the database. This is an auto-incrementing number and is the primary key of the entity. This cannot be NULL. This is an integer data type and holds up to 9 characters.

firstName: This attribute holds the first name of the hero. This cannot be NULL. This is a varchar data type and holds up to 20 characters.

lastName: This attribute holds the first name of the hero. The default value for this attribute is NULL, as some characters do not have a last name. This is a varchar data type and holds up to 20 characters.

species: This attribute holds the species of the hero. This cannot be NULL and the default value is human. This a varchar data type and holds up to 50 characters.

age: This attribute holds the age of the hero. There is no max age and the default value is NULL as some ages are unknown. This is a bigint data type and can hold up to 20 characters.

alias: This attribute holds the alias often used by the hero to protect their real identity. The default value is NULL as some heroes opt to not conceal their identity. This is a varchar data type and can hold up to 20 characters.

loveInterest: This attribute holds the (non-supernatural) name of the love interest of the hero (first and last together). The default value is NULL as some heroes do not have a love interest. This is a varchar data type and can hold up to 50 characters.

homePlanet: This attribute holds the home planet of the hero. This is a foreign key and refers to the Planet entity. The default value is NULL as it is unknown where some heroes are from. This is a varchar data type and can hold up to 50 characters.

Villains – This entity holds the heroes in the database, is involved in a relation with every other entity, and has the following attributes:

villain_id: This attribute holds a number that is automatically assigned to each villain when they are recorded in the database. This is an auto-incrementing number and is the primary key of the entity. This cannot be NULL. This is an integer data type and holds up to 9 characters.

firstName: This attribute holds the first name of the villain. This cannot be NULL. This is a varchar data type and holds up to 20 characters.

lastName: This attribute holds the first name of the villain. The default value for this attribute is NULL, as some characters do not have a last name. This is a varchar data type and holds up to 20 characters.

species: This attribute holds the species of the villain. This cannot be NULL and the default value is human. This a varchar data type and holds up to 50 characters.

age: This attribute holds the age of the villain. The default value is NULL as some ages are unknown. This is a bigint data type and can hold up to 20 characters.

alias: This attribute holds the alias often used by the villain to protect their real identity. The default value is NULL as some villains opt to not conceal their identity. This is a varchar data type and can hold up to 20 characters.

enemy: This attribute holds the name of the enemy of the villain (a hero). This value cannot be NULL as all villains have an enemy. This is a varchar data type and can hold up to 50 characters.

Teams – This entity holds the teams in the database, is involved in a relation with the heroes entity, and has the following attributes:

team_id: This attribute holds a number that is automatically assigned to each team when it is are recorded in the database. This is an auto-incrementing number and is the primary key of the entity. This cannot be NULL. This is an integer data type and holds up to 9 characters.

name: This attribute holds the name of the team. This cannot be NULL. This is a varchar data type and holds up to 50 characters.

size: This attribute holds the size of the team. This cannot be NULL. This is a bigint data type and holds up to 100 characters.

Planets – This entity holds the abilities in the database, is involved in a relation with the heroes entity, and has the following attributes:

planet_id: This attribute holds a number that is automatically assigned to each ability when it is are recorded in the database. This is an auto-incrementing number and is the primary key. This cannot be NULL. This is an integer data type and holds up to 9 characters.

name: This attribute holds the name of the ability. This cannot be NULL. This is a varchar data type and holds up to 50 characters.

populationSize: This attribute holds the population size of the planet. This cannot be NULL. This is a bigint data type and holds up to 20 characters.

Abilities – This entity holds the planets in the database, is involved in a relation with the heroes entity, and has the following attributes:

ability_id: This attribute holds a number that is automatically assigned to each planet when it is are recorded in the database. This is an auto-incrementing number and is the primary key of the entity. This cannot be NULL. This is an integer data type and holds up to 9 characters.

name: This attribute holds the name of the planet. This cannot be NULL. This is a varchar data type and holds up to 50 characters.

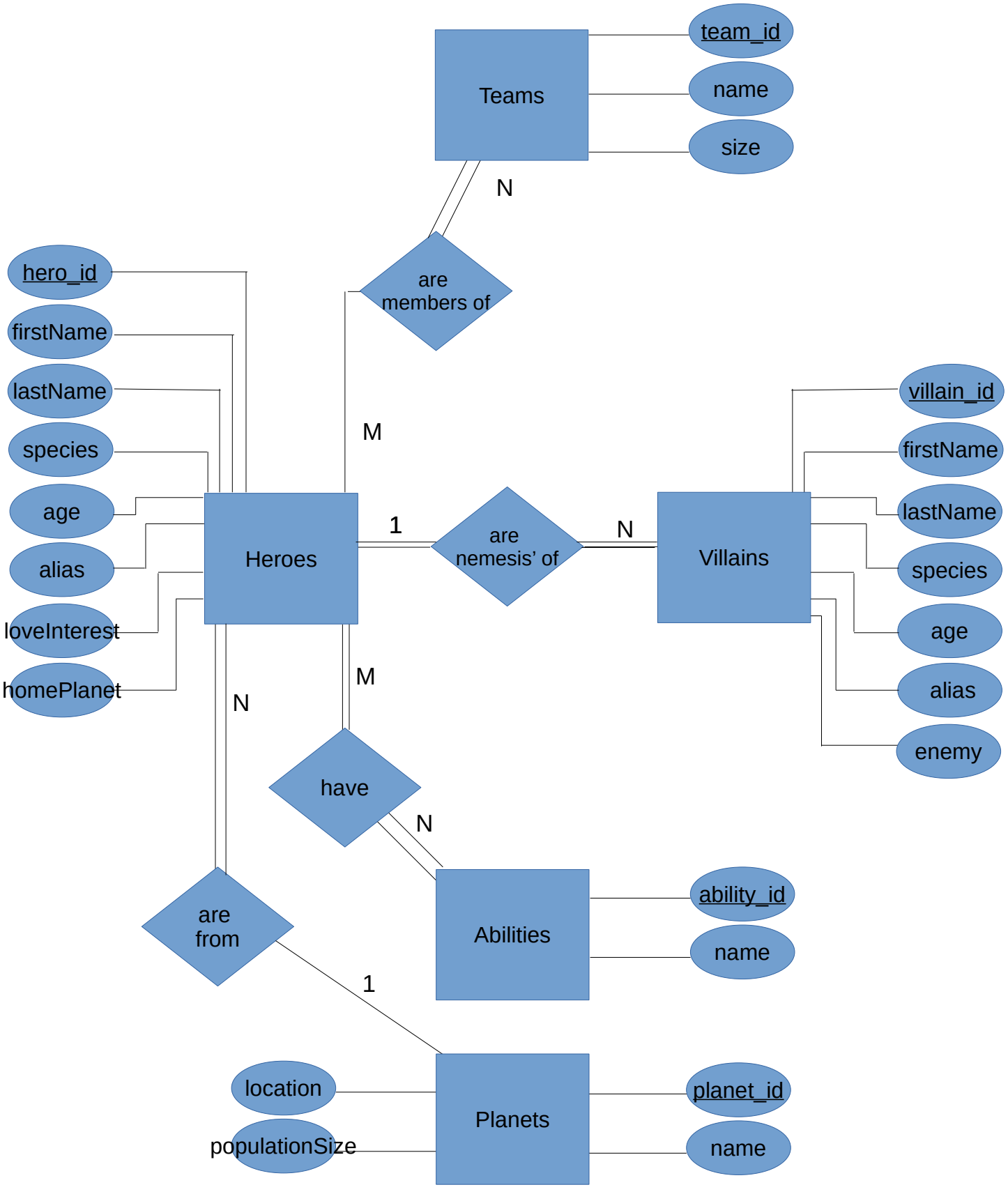
Relationships

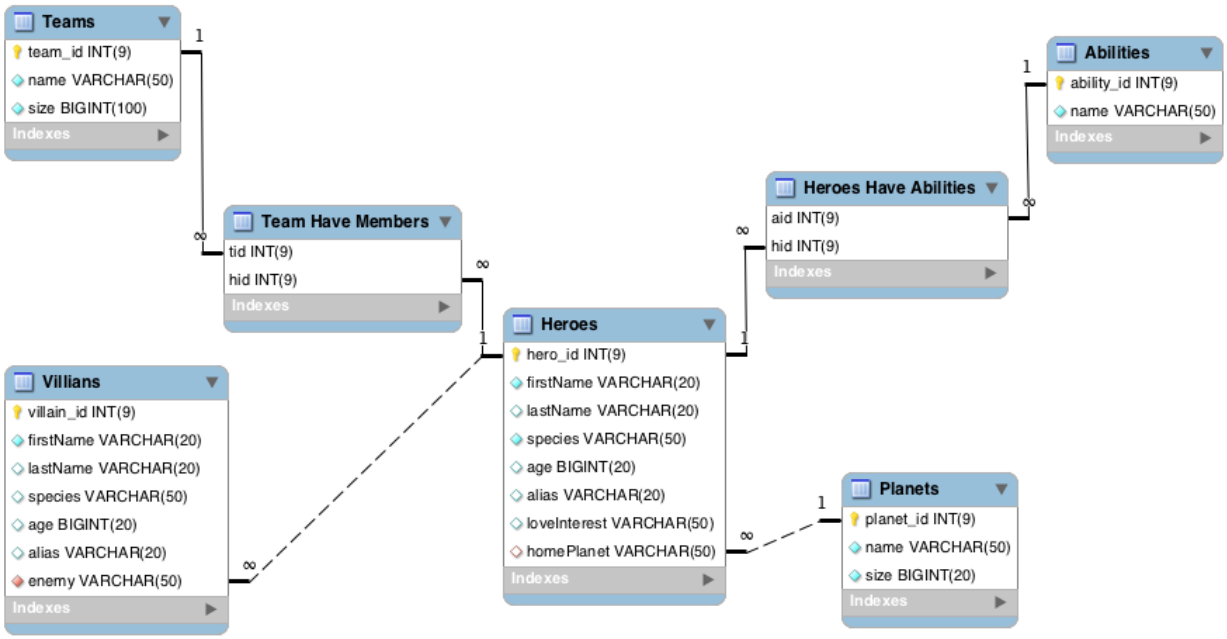
Characters are members of a team (many-to-many) – A character can be a member of zero or many different teams, and many characters can be on any one team.

Characters are from planets (one-to-many) – A character can only be from zero or one planets but a planet can be home to many characters.

Characters have abilities (many-to-many) – Characters must have one or more abilities and many characters can have the same abilities.

Heroes have enemies (one-to-many) – Heroes must have one or more enemies (villains), but villains are only associated with one hero.


Teams
team_id
name
size
N
are members of
M
hero_id
firstName
lastName
species
age
alias
loveInterest
homePlanet
Heroes
1
are nemesis' of
N
Villains
villain_id
firstName
lastName
species
age
alias
enemy
M
N
have
N
are from
1
Abilities
ability_id
name
Planets
location
populationSize
planet_id
name


Teams
team_id INT(9)
name VARCHAR(50)
size BIGINT(100)
Indexes
Team Have Members
tid INT(9)
hid INT(9)
Indexes
Villians
villain_id INT(9)
firstName VARCHAR(20)
lastName VARCHAR(20)
species VARCHAR(50)
age BIGINT(20)
alias VARCHAR(20)
enemy VARCHAR(50)
Indexes
Heroes
hero_id INT(9)
firstName VARCHAR(20)
lastName VARCHAR(20)
species VARCHAR(50)
age BIGINT(20)
alias VARCHAR(20)
loveInterest VARCHAR(50)
homePlanet VARCHAR(50)
Indexes
Heroes Have Abilities
aid INT(9)
hid INT(9)
Indexes
Abilities
ability_id INT(9)
name VARCHAR(50)
Indexes
Planets
planet_id INT(9)
name VARCHAR(50)
size BIGINT(20)
Indexes