Joseph Sullivan
CS 340
Final Project
3/18/2018

Marvel Film Database & Website

TOPIC OUTLINE

My database stores data regarding Marvel Comics characters and films. Below I will give an overview of what each entity and its attributes represent.

- **Characters**: These are fictional characters from Marvel comics and films.
    - first name: The character's first name.
    - last name: The character's last name.
    - alias: An alternate name that the character goes by, typically a superhero or villain name.
    - species: The species of the character. Many characters are human, but some belong to alien races from other planets or dimensions.
- **Actor**: Film actors
    - first name: The actor's first name.
    - last name: The actor's last name.
- **Superpowers**: The superhuman or supernatural abilities that comic heroes and villains may have.
    - name: The name that describes the superpower.
- **Films**: Films that have marvel characters in them.
    - title: The title of the film.
    - release date: The date that the film was released.
    - opening weekend: Gross ticket sales for the opening weekend that the film was in theaters, rounded to the nearest whole dollar.
- **Directors**: Film directors
    - first name: director's first name.
    - last name: director's last name.

Website details

My website uses express.js with handlebars, thus I have submitted my files in the same folder structure that is typical of handlebars templated websites. I excluded the node_modules folder from my submission as it is very large and does not contain any code that I have written, thus it did not seem necessary for grading purposes. A dropdown menu allows the user to choose a table to view. Each table has functionality for viewing, entering, deleting, and updating entries. The Characters' table also has a dropdown menu for filtering the characters displayed by superpower.

Joseph Sullivan
CS 340
Final Project
3/18/2018

DATABASE OUTLINE

This section describes each entity and relationship and their attributes in terms of how they are defined in the ER diagram, schema, and data definition sql queries.

Entities

- **Characters**
  - **ID**: An int. can't be NULL. Auto incremented. The primary key.
  - **firstName**: A varchar. This cannot be NULL because a character must have at least one name. If a character does not have a last name or alias, the single name that they go by will be entered in the firstName attribute.
  - **lastName**: A varchar. lastName is optional and may be NULL. If a character does have a last name, their firstName, lastName combination must be unique.
  - **alias**: A varchar. alias is optional and may be NULL
  - **species**: A varchar. Cannot be NULL. If the species of a character is unknown, then enter unknown as the species.
- **Actors**
  - **ID**: An int. can't be NULL. Auto incremented. The primary key.
  - **firstName**: A varchar. Cannot be NULL.
  - **lastName**: A varchar. Cannot be NULL.
    - The firstName, lastName combination must be unique.
  - **characterID**: An int, foreign key referencing a Character ID that represents the relationship of an actor playing a character.
    - If a character is deleted, any actors that play that character have this value set to NULL.
- **Superowers**
  - **ID:** An int. can't be NULL. Auto incremented. The primary key.
  - **name:** A varchar that cannot be NULL and must be unique.
- **Directors**
  - **ID:** An int. can't be NULL. Auto incremented. The primary key.
  - **firstName:** A varchar that cannot be NULL.
  - **lastName:** A varchar that cannot be NULL.
    - firstName, lastName combinations must be unique.
- **Films**
  - **ID:** An int. can't be NULL. Auto incremented. The primary key.
  - **title:** A varchar that cannot be NULL.
  - **releaseDate:** A date that cannot be NULL.
    - Some older films share names with newer films, thus films can have the same title, but must have a unique title, releaseDate combination.

        - For films that are not released yet, the releaseDate is the represents the expected release date.
    - **openingWeekend:** An int. Can be NULL because films that have not come out yet do not have a gross ticket sales for an opening weekend.
    - **directorID:** An int, foreign key that references a Director ID. Represents the relationship of a film being directed by a director.
        - Is set to NULL if the referenced director is deleted.

Relationships

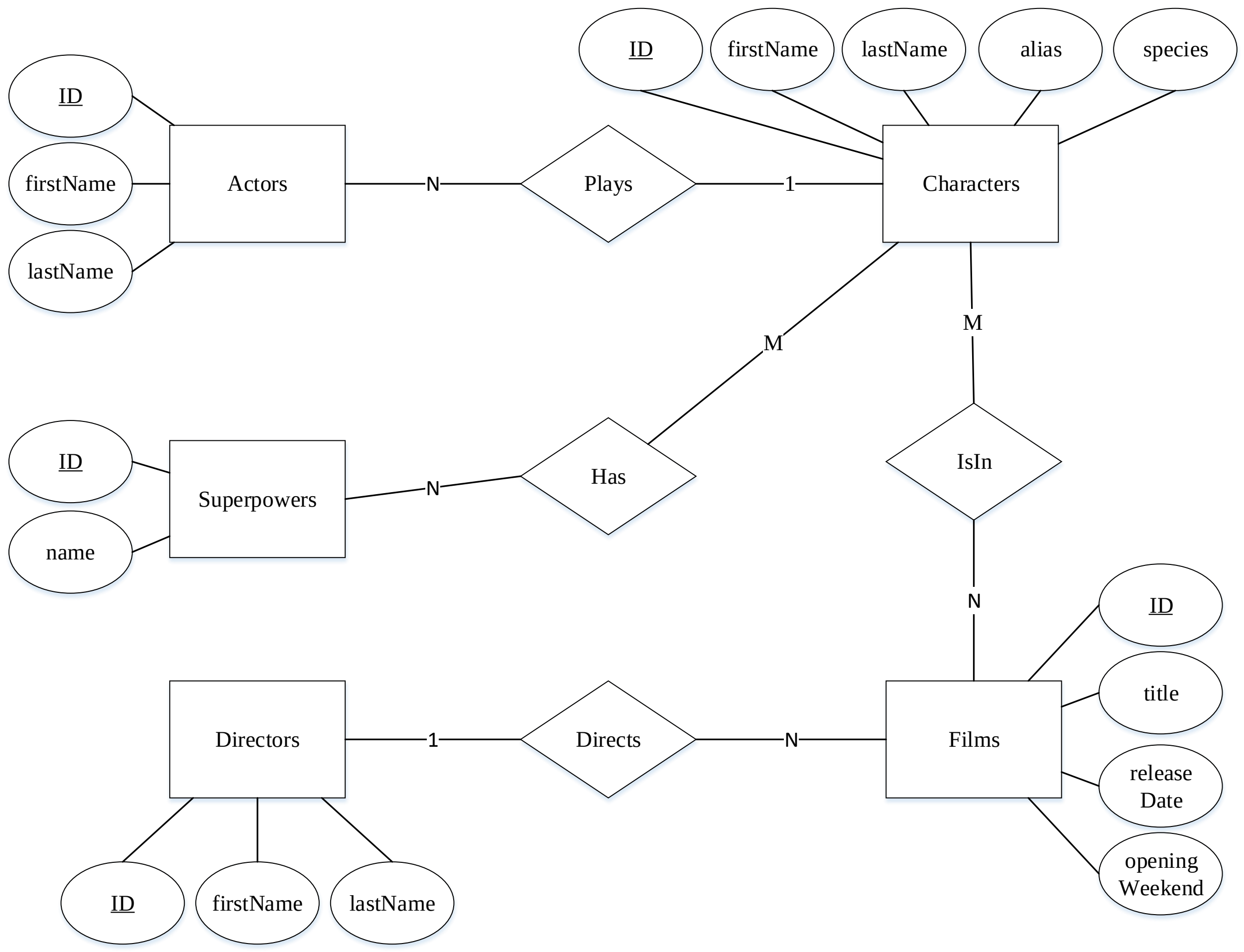
- **Plays:** Actors entities play Characters entities.
    - Represented by the characterID attribute of the Actors entity.
    - Actors have **partial** participation in this relationship and cannot play more than **one** character for the purposes of this database.
    - Characters can be played by **many** actors because several films have been made over the years about the same characters and these characters are sometimes played by different actors from one film to the next. Characters also have **partial** participation in this relationship.
- **Directs:** Directors entities direct Films entities.
    - Represented by the directorID attribute of the Films entity.
    - Films have **partial** participation in this relationship and cannot be directed by more than director for the purposes of this database.
    - Directors can direct **many** films and also have partial participation in this relationship.
- **Has:** A Characters entity has a Superpowers entity. Represented by a relationship table.
    - **superpowerID:** An int. Cannot be NULL.
        - Part of the primary key.
        - A foreign key reference to a Superpower ID.
    - **characterID:** An int. Cannot be NULL.
        - Part of the primary key.
        - A foreign key reference to a Character ID.
    - If either foreign key is deleted, this relationship is deleted.
    - Characters can have **many** superpowers and a superpower can be had by **many** characters.
    - Both entities have partial participation in this relationship.

Joseph Sullivan
CS 340
Final Project
3/18/2018

- **IsIn**: A Characters entity is in a Films entity. Represented by a relationship table.
    - **characterID:** An int. Cannot be NULL.
        - Part of the primary key.
        - A foreign key reference to a Character ID.
    - **filmID**: An int. Cannot be NULL.
        - Part of the primary key.
        - A foreign key reference to a Film ID.
    - If either foreign key is deleted, this relationship is deleted.
    - Characters can be in **many** films, and films can have **many** characters in them.
    - Both entities have partial participation in this relationship.

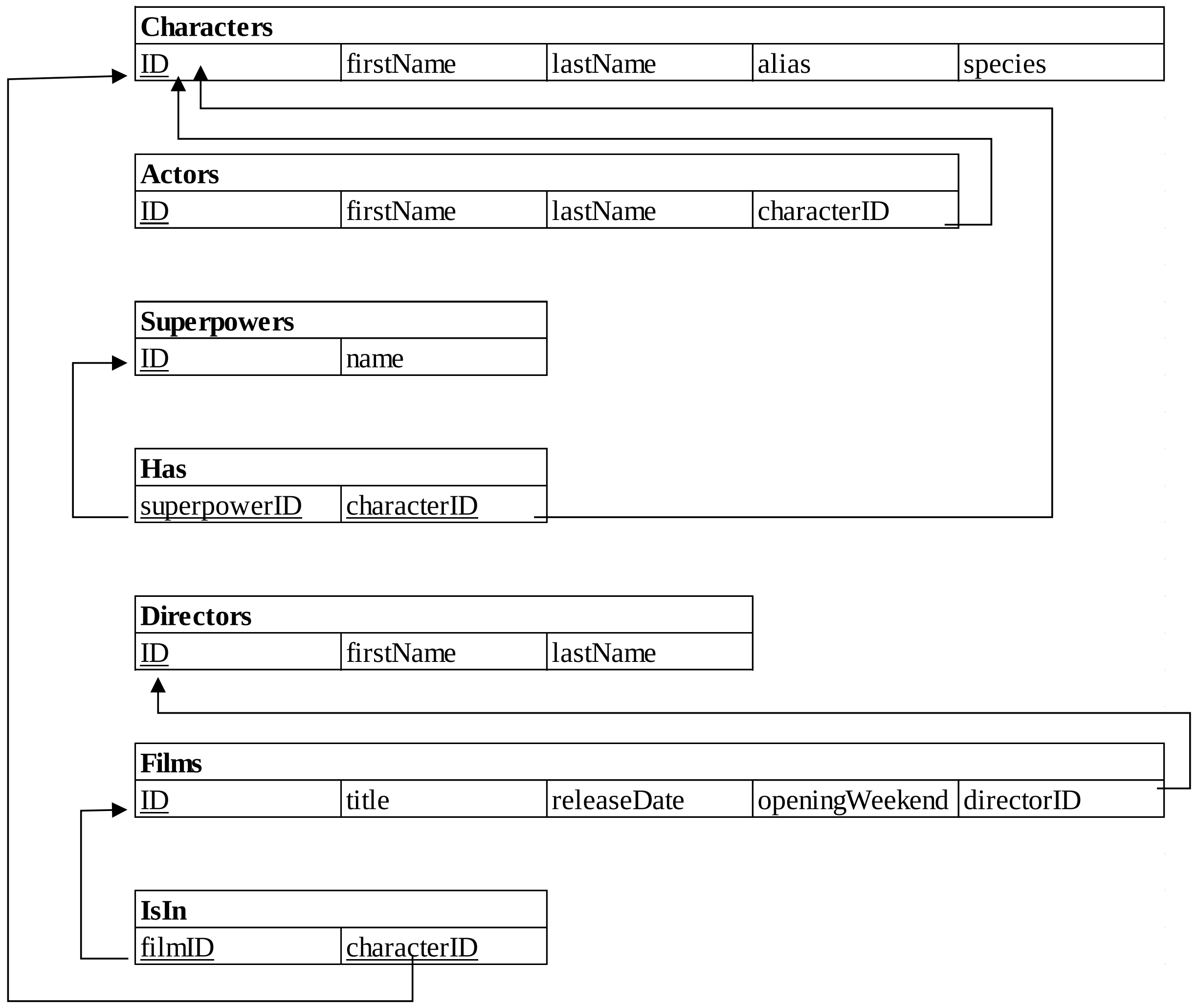
Joseph Sullivan
CS 340
Final Project
3/18/2018

ER DIAGRAM



Joseph Sullivan
CS 340
Final Project
3/18/2018

SCHEMA


Characters
ID | firstName | lastName | alias | species
Actors
ID | firstName | lastName | characterID
Superpowers
ID | name
Has
superpowerID | characterID
Directors
ID | firstName | lastName
Films
ID | title | releaseDate | openingWeekend | directorID
IsIn
filmID | characterID

Joseph Sullivan
CS 340
Final Project
3/18/2018

DATA DEFINITION QUERIES

```
DROP TABLE IF EXISTS `IsIn`;
DROP TABLE IF EXISTS `Films`;
DROP TABLE IF EXISTS `Directors`;
DROP TABLE IF EXISTS `Has`;
DROP TABLE IF EXISTS `Superpowers`;
DROP TABLE IF EXISTS `Actors`;
DROP TABLE IF EXISTS `Characters`;

CREATE TABLE `Characters` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `firstName` varchar(255) NOT NULL,
  `lastName` varchar(255),
  `alias` varchar(255),
  `species` varchar(255) NOT NULL,
  PRIMARY KEY (`ID`),
  UNIQUE (`firstName`, `lastName`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `Actors` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `firstName` varchar(255) NOT NULL,
  `lastName` varchar(255) NOT NULL,
  `characterID` int(11),
  PRIMARY KEY (`ID`),
  UNIQUE (`firstName`, `lastName`),
  FOREIGN KEY (`characterID`) REFERENCES `Characters` (`ID`)
  ON DELETE SET NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `Superpowers` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) NOT NULL UNIQUE,
  PRIMARY KEY (`ID`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Joseph Sullivan
CS 340
Final Project
3/18/2018

```
CREATE TABLE `Has` (
  `SuperpowerID` int(11) NOT NULL DEFAULT '0',
  `characterID` int(11) NOT NULL DEFAULT '0',
  PRIMARY KEY (`superpowerID`, `characterID`),
  FOREIGN KEY (`superpowerID`) REFERENCES `Superpowers` (`ID`)
  ON DELETE CASCADE,
  FOREIGN KEY (`characterID`) REFERENCES `Characters` (`ID`)
  ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `Directors` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `firstName` varchar(255) NOT NULL,
  `lastName` varchar(255) NOT NULL,
  PRIMARY KEY (`ID`),
  UNIQUE (`firstName`, `lastName`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `Films` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `title` varchar(255) NOT NULL,
  `releaseDate` date,
  `openingWeekend` int(11),
  `directorID` int(11),
  PRIMARY KEY (`ID`),
  UNIQUE (`title`, `releaseDate`),
  FOREIGN KEY (`directorID`) REFERENCES `Directors` (`ID`)
  ON DELETE SET NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `IsIn` (
  `filmID` int(11) NOT NULL DEFAULT '0',
  `characterID` int(11) NOT NULL DEFAULT '0',
  PRIMARY KEY (`filmID`, `characterID`),
  FOREIGN KEY (`filmID`) REFERENCES `Films` (`ID`)
  ON DELETE CASCADE,
  FOREIGN KEY (`characterID`) REFERENCES `Characters` (`ID`)
  ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Joseph Sullivan
CS 340
Final Project
3/18/2018

DATA MANIPULATION QUERIES

```
--Some queries appear in multiple .js files but are only listed here once.

--characters.js
SELECT table_name FROM information_schema.tables
WHERE table_schema='cs340_sullijos';

SELECT ID, firstName, lastName, alias, species FROM Characters;

SELECT ID, firstName, lastName, alias, species FROM Characters
LEFT JOIN Has ON Characters.ID=Has.characterID
WHERE Has.superpowerID=[superpowerSelection];

SELECT ID, firstName, lastName, alias, species FROM Characters
WHERE ID=[characterSelection];

INSERT INTO Characters (firstName, lastName, alias, species)
VALUES ([fNameEntry], [lNameEntry], [aliasEntry], [speciesEntry]);

UPDATE Characters SET
    firstName=[fNameEntry],
    lastName=[lNameEntry],
    alias=[aliasEntry],
    species=[speciesEntry]
WHERE ID=[characterSelection];

DELETE FROM Characters WHERE ID=[characterSelection];

--actors.js
SELECT
    Actors.ID,
    Actors.characterID,
    Actors.firstName,
    Actors.lastName,
    Characters.firstName AS cFirstName,
    Characters.lastName AS cLastName,
    Characters.alias
FROM Actors
LEFT JOIN Characters ON Actors.characterID=Characters.ID
ORDER BY Characters.firstName;

SELECT ID, firstName, lastName, alias FROM Characters;
```

Joseph Sullivan
CS 340
Final Project
3/18/2018

```sql
SELECT ID, firstName, lastName, characterID FROM Actors
WHERE ID=[actorSelection];

INSERT INTO Actors (firstName, lastName, characterID)
VALUES ([fNameEntry], [lNameEntry], [characterSelection]);

UPDATE Actors SET
    firstName=[fNameEntry],
    lastName=[lNameEntry],
    characterID=[characterSelection]
WHERE ID=[actorSelection];

DELETE FROM Actors WHERE ID=[actorSelection];

--superpowers.js
SELECT ID, name FROM Superpowers;

SELECT ID, name FROM Superpowers WHERE ID=[superpowerSelection];

INSERT INTO Superpowers (name) VALUES [nameEntry];

UPDATE Superpowers SET name=[nameEntry] WHERE ID=[superpowerSelection];

DELETE FROM Superpowers WHERE ID=[superpowerSelection];

--directors.js
SELECT ID, firstName, lastName FROM Directors;

SELECT ID, firstName, lastName FROM Directors
WHERE ID=[directorSelection];

INSERT INTO Directors (firstName, lastName)
VALUES ([fNameEntry], [lNameEntry]);

UPDATE Directors SET
    firstName=[fNameEntry],
    lastName=[lNameEntry]
WHERE ID=[directorSelection];

DELETE FROM Directors WHERE ID=[directorSelection];
```

Joseph Sullivan
CS 340
Final Project
3/18/2018

```
--films.js
SELECT
    Films.ID,
    Films.directorID,
    Films.title,
    DATE_FORMAT(Films.releaseDate, '%Y-%m-%d') AS releaseDate,
    Films.openingWeekend,
    Directors.firstName,
    Directors.lastName
FROM Films
LEFT JOIN Directors ON Films.directorID=Directors.ID
ORDER BY Films.title;

SELECT
    ID,
    title,
    DATE_FORMAT(releaseDate, '%Y-%m-%d') as releaseDate,
    openingWeekend,
    directorID
FROM Films WHERE ID=[filmSelection];

INSERT INTO Films (title, releaseDate, openingWeekend, directorID)
VALUES ([titleEntry],[rDateEntry],[oWeekendEntry],[directorSelection]);

UPDATE Films SET
    title=[titleEntry],
    releaseDate=[rDateEntry],
    openingWeekend=[oWeekendEntry],
    directorID=[directorSelection]
WHERE ID=[filmSelection];

DELETE FROM Films WHERE ID=[filmSelection];
```

Joseph Sullivan
CS 340
Final Project
3/18/2018

```
--has.js
SELECT
    Has.superpowerID,
    Has.characterID,
    Characters.firstName,
    Characters.lastName,
    Characters.alias,
    Superpowers.name
FROM Characters
JOIN Has ON Characters.ID=Has.characterID
JOIN Superpowers ON Has.superpowerID=Superpowers.ID
ORDER BY Superpowers.name;

INSERT INTO Has (characterID, superpowerID)
VALUES ([characterSelection],[superpowerSelection]);

UPDATE Has SET
    characterID=[characterSelection],
    superpowerID=[superpowerSelection]
WHERE characterID=[hasCharacterID]
AND superpowerID=[hasSuperpowerID];

DELETE FROM Has
WHERE characterID=[hasCharacterID]
AND superpowerID=[hasSuperpowerID];

--isin.js
SELECT
    IsIn.characterID,
    IsIn.filmID,
    Characters.firstName,
    Characters.lastName,
    Characters.alias,
    Films.title
FROM Characters
JOIN IsIn ON Characters.ID=IsIn.characterID
JOIN Films ON IsIn.filmID=Films.ID
ORDER BY Films.title;

SELECT ID, title FROM Films;

SELECT ID, firstName, lastName, alias FROM Characters
WHERE ID=[characterSelection];
```

Joseph Sullivan
CS 340
Final Project
3/18/2018

```
SELECT ID, title FROM Films WHERE ID=[filmSelection];

INSERT INTO IsIn (filmID, characterID)
VALUES ([filmSelection],[characterSelection]);

UPDATE IsIn SET
    characterID=[characterSelection],
    filmID=[filmSelection]
WHERE characterID=[isinCharacterID]
AND filmID=[isinFilmID];

DELETE FROM IsIn
WHERE characterID=[isinCharacterID]
AND superpowerID=[isinFilmID];
```