Christopher Merrill
CS340
Final Project

OUTLINE

I chose to make a database about various drummers and the equipment they use. It will contain information on drummers, drum kits, stick types and equipment brands. Each of these categories will have their own entity table, making it possible to easily find out which drummers use which type of equipment.

To easily cross-examine drum kits and the people who played them, a many-to-many relationship table called “plays” was established (example: Neil Peart *plays* a 14-piece DW set with Sabian Cymbals). I didn’t want each drummer to be associated with only one drum kit, since most drummers play many different kits over the course of their career. Likewise, I didn’t want each drum kit to be associated with only one drummer, since many drummers may use the same type of drum kit. Thus a many-to-many relationship between drum kits and drummers seemed like the logical choice.

In case you are unfamiliar with drum terminology, the following two definitions should be all you need to understand the database:

Cymbals:

Cymbals are the shiny metal round things shown above!

Drum Kit:

Drum kit refers to everything you see above, including the drums, cymbals and hardware. The number of “pieces” in a drum kit refers to the number of drums, so a 5-piece set (shown above) has five drums.

DATABASE OUTLINE

Entities and their Attributes:

This database contains the following four entities:

1. **Drummer**: Contains biographical information about a particular drummer. Attributes include:
   - Drummer ID: Primary key.
   - Stick ID: Foreign key that links to the stick entity.
   - Name: The first and last name of the drummer (must be unique).
   - Hometown: The city the drummer was born in.
   - Band: The band the drummer is most famous for.

2. **Drum Kit:** Contains details about a particular drum set. Attributes include:
   - Kit ID: Primary Key
   - Brand ID: Foreign key that links to the brand entity.
   - Pieces: The number of drums used in a drum set (i.e. a 4-piece kit has four drums)
   - Cymbal Type: The type of cymbals used in the kit.

   Note: The combination of Brand ID, Pieces and Cymbal Type must be unique.

3. **Stick:** Contains details about particular drum sticks. Attributes include:
   - Stick ID: Primary Key
   - Brand ID: Foreign key that links to the brand entity.
   - Type: The type of drumstick (wood type, size and/or name of the stick)

   Note: The combination of Brand ID and type must be unique.

4. **Brand:** Contains details about particular manufacturer of drum kits or drum sticks. Attributes include:
   - Brand ID: Primary Key (must be unique).
   - Name: The brand name.
   - Year Established: The year the company was founded.
   - Country of Origin: The country the company was founded in.
   - Founder: The name of the person who created the company.

DATABASE OUTLINE (continued)

Relationships:

The following four relationships exist in this database:

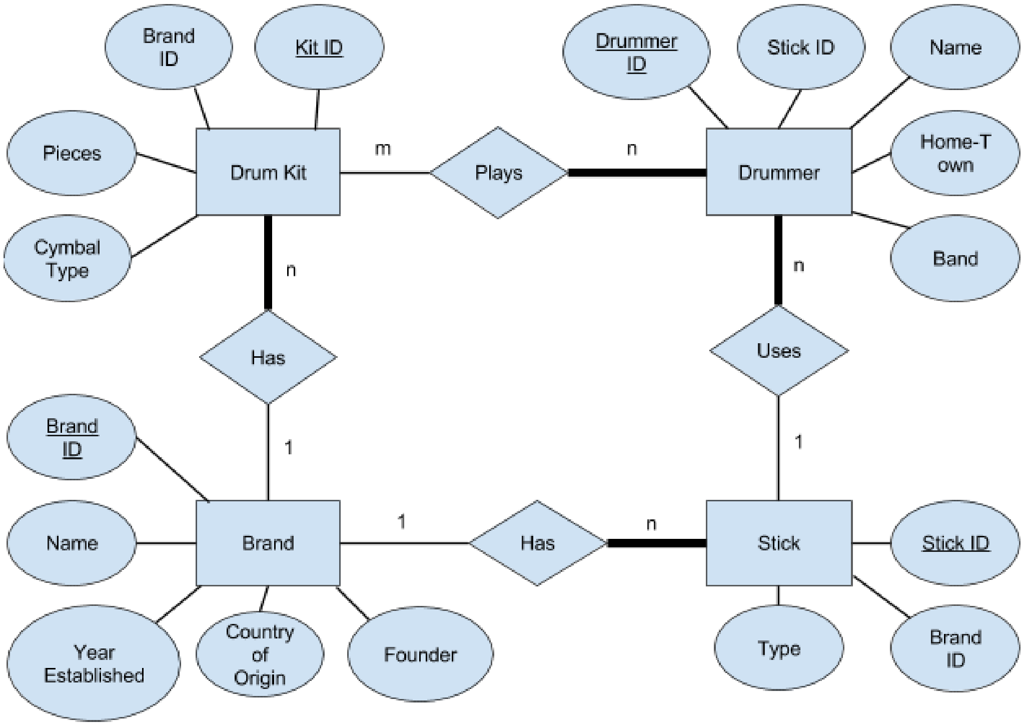
1. A many-to-many relationship between the Drummer and Drum Kit entities.
    - A drummer plays at least one drum set (they can play many different types of drum sets over the course of their career, but they can't play 0 drum sets or else they wouldn't be a drummer).
    - A drum kit can have 0 or more drummers (a drum kit could be created without having ever been played, and many different drummers could use the same type of drum kit).

2. A 1-to-many relationship between the Drummer and Stick entities.
    - For the sake of this database, we will assume that a drummer uses exactly one type of drumstick.
    - A type of drumstick can be used by many different drummers or no drummers.

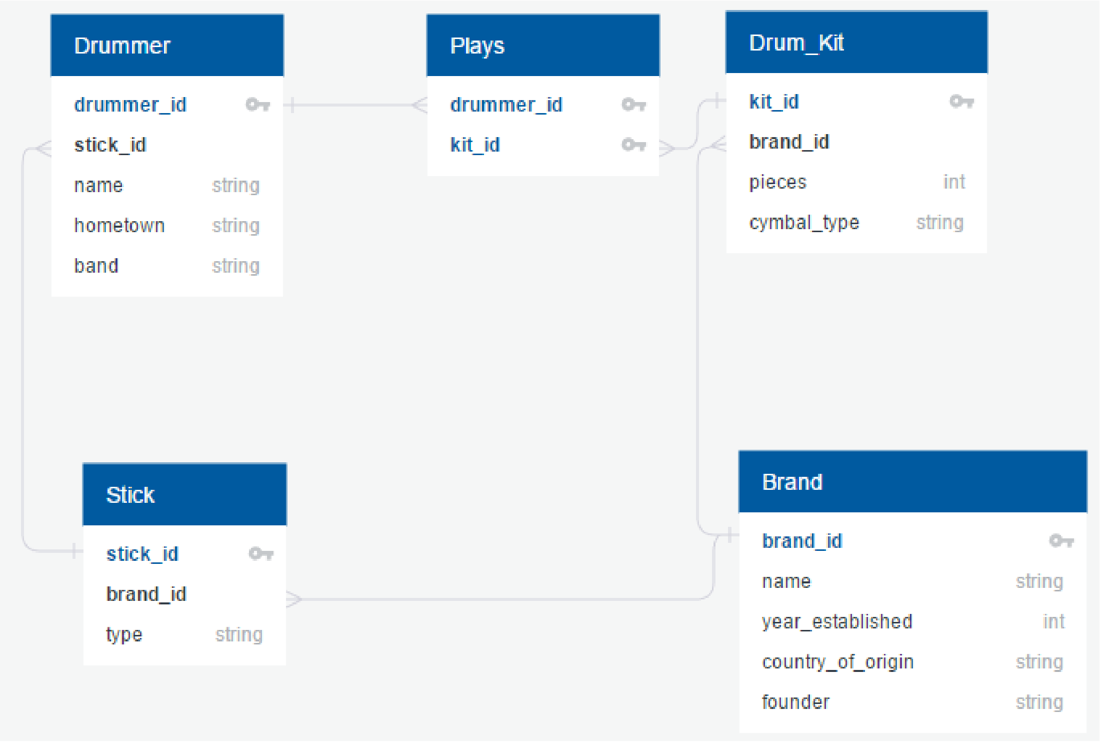
3. A 1-to-many relationship between the Brand and Drum Kit entities.
    - A drum kit has exactly one brand.
    - A brand can manufacture many different types of drum kits (or none if they are a stick manufacturer).

4. A 1-to-many relationship between the Brand and Stick entities.
    - A stick has exactly one brand.
    - A brand can manufacture many different types of sticks (or none if they are a drum-kit manufacturer).

ER DIAGRAM



SCHEMA



Drummer	
drummer_id	🔑
stick_id	
name	string
hometown	string
band	string

Plays	
drummer_id	🔑
kit_id	🔑

Drum_Kit	
kit_id	🔑
brand_id	
pieces	int
cymbal_type	string

Stick	
stick_id	🔑
brand_id	
type	string

Brand	
brand_id	🔑
name	string
year_established	int
country_of_origin	string
founder	string

DATA DEFINITION QUERIES

The following queries were used to create the four entity tables (drummer, brand, stick, drum kit) as well as the relationship table (plays):

```
CREATE TABLE `drummer` (
	`drummer_id` INT(11) AUTO_INCREMENT,
	`name` VARCHAR(255) NOT NULL,
	`hometown` VARCHAR(255) NOT NULL,
	`band` VARCHAR(255) NOT NULL,
	`stick_id` INT(11) NOT NULL,
	PRIMARY KEY(`drummer_id`),
	FOREIGN KEY (`stick_id`) REFERENCES `stick`(`stick_id`),
	UNIQUE (`name`)
) ENGINE=InnoDB;


CREATE TABLE `brand` (
	`brand_id` INT(11) AUTO_INCREMENT,
	`name` VARCHAR(255) NOT NULL,
	`founder` VARCHAR(255) NOT NULL,
	`country_of_origin` VARCHAR(255) NOT NULL,
	`year_established` INT(11) NOT NULL,
	PRIMARY KEY(`brand_id`),
	UNIQUE (`name`)
) ENGINE=InnoDB;


CREATE TABLE `stick` (
	`stick_id` INT(11) AUTO_INCREMENT,
	`type` VARCHAR(255) NOT NULL,
	`brand_id` INT(11) NOT NULL,
	PRIMARY KEY(`stick_id`),
	FOREIGN KEY (`brand_id`) REFERENCES `brand`(`brand_id`),
	UNIQUE (`type`, `brand_id`)
) ENGINE=InnoDB;
```

DATA DEFINITION QUERIES (continued)

```
CREATE TABLE `drum_kit` (
	`kit_id` INT(11) AUTO_INCREMENT,
	`brand_id` INT(11) NOT NULL,
	`pieces` INT(11) NOT NULL,
	`cymbal_type` VARCHAR(255) NOT NULL,
	PRIMARY KEY(`kit_id`),
	FOREIGN KEY (`brand_id`) REFERENCES `brand`(`brand_id`),
	UNIQUE (`brand_id`, `pieces`, `cymbal_type`)
) ENGINE=InnoDB;


CREATE TABLE `plays` (
	`drummer_id` INT(11) NOT NULL,
	`kit_id` INT(11) NOT NULL,
	FOREIGN KEY (`drummer_id`) REFERENCES `drummer`(`drummer_id`),
	FOREIGN KEY (`kit_id`) REFERENCES `drum_kit`(`kit_id`),
	PRIMARY KEY(`drummer_id`, `kit_id`),
	UNIQUE (`drummer_id`, `kit_id`)
) ENGINE=InnoDB;
```

***Notes:**

- All primary keys are auto incremented upon creation.
- Every table besides “brand” contains at least one foreign key (as shown in the schema) which allows for the 1-to-many relationships outlined above.
- Certain attributes were marked unique to prevent the user from adding duplicate drummer names, brand names, drum kit configurations or stick types.

DATA MANIPULATION QUERIES

Selects:

The following queries populate the data tables on the web page, allowing the user to see up-to-date versions of the data (basic select * functions):

```
SELECT * FROM brand

SELECT * FROM drum_kit

SELECT * FROM stick

SELECT * FROM drummer

SELECT * FROM plays
```

These select queries were utilized in the search functionalities:

Search Drum Kit by Drummer Name Query:

Entered: Drummer Name
Returned: The brand name, number of pieces and cymbal type of all of the kits that drummer has played

```
SELECT brand.name, drum_kit.pieces, drum_kit.cymbal_type FROM drummer
INNER JOIN plays ON drummer.drummer_id = plays.drummer_id
INNER JOIN drum_kit ON plays.kit_id = drum_kit.kit_id
INNER JOIN brand ON drum_kit.brand_id = brand.brand_id
WHERE drummer.name = [name];
```

Search Drummer by Drum Kit Query:

Entered: Drummer Kit
Returned: The name, band and hometown of all drummers who have played that type of drum kit

```
SELECT drummer.name, drummer.band, drummer.hometown FROM drummer
INNER JOIN plays ON drummer.drummer_id = plays.drummer_id
INNER JOIN drum_kit ON plays.kit_id = drum_kit.kit_id
WHERE drum_kit.kit_id = [kit_id];
```

DATA MANIPULATION QUERIES (continued)

Search Brand by Drummer Query:
This query consists of two, separate SELECT queries joined together using UNION. One SELECT query pulls the brand of sticks the drummer uses and the other pulls the brand of drum kit.

Entered: Drummer name
Returned: The brands that this drummer uses (both stick and drum kit brands)

```
(SELECT brand.name, brand.founder, brand.country_of_origin, brand.year_established FROM
brand
INNER JOIN drum_kit ON brand.brand_id = drum_kit.brand_id
INNER JOIN plays ON drum_kit.kit_id = plays.kit_id
INNER JOIN drummer ON drummer.drummer_id = plays.drummer_id
WHERE drummer.name = [name])
UNION
(SELECT brand.name, brand.founder, brand.country_of_origin, brand.year_established FROM
brand
INNER JOIN stick ON brand.brand_id = stick.brand_id
INNER JOIN drummer ON drummer.drummer_id = stick.drummer_id
WHERE drummer.name = [name]);
```

Search Drummer by Brand Query:
This query consists of two, separate SELECT queries joined together using UNION. One SELECT query pulls the drummers that use that brand as a kit, and the other pulls the drummers who use that brand for sticks. This allows for brands that produce both sticks and drum kits.

Entered: Brand name
Returned: The drummers that use that brand

```
(SELECT drummer.name, drummer.band, drummer.hometown FROM drummer
INNER JOIN plays ON drummer.drummer_id = plays.drummer_id
INNER JOIN drum_kit ON plays.kit_id = drum_kit.kit_id
INNER JOIN brand ON drum_kit.brand_id = brand.brand_id
WHERE brand.brand_id = [id])
UNION
(SELECT drummer.name, drummer.band, drummer.hometown FROM drummer
INNER JOIN stick ON drummer.stick_id = stick.stick_id
INNER JOIN brand ON stick.brand_id = brand.brand_id
WHERE brand.brand_id = [id]);
```

DATA MANIPULATION QUERIES (continued)

Search Stick Type by Drummer Name Query:

Entered: Drummer Name
Returned: The brand name and stick type of the sticks that drummer uses.

```
SELECT brand.name, stick.type FROM stick
INNER JOIN drummer ON drummer.stick_id = stick.stick_id
INNER JOIN brand ON stick.brand_id = brand.brand_id
WHERE drummer.name = [name];
```

Search Drummer by Stick Type Name Query:

Entered: Drummer Name
Returned: The brand name and stick type of the sticks that drummer uses.

```
SELECT drummer.name, drummer.band, drummer.hometown FROM drummer
INNER JOIN stick ON drummer.stick_id = stick.stick_id
WHERE stick.stick_id = [id];
```

Inserts:

The following queries were used to allow the user to add values to any of the entity tables:

```
INSERT INTO brand (`name`, `founder`, `country_of_origin`, `year_established`) VALUES
([name], [founder], [country_of_origin], [year_establised])

INSERT INTO drum_kit (`brand_id`, `pieces`, `cymbal_type`) VALUES ([brand_id], [pieces],
[cymbal_type])

INSERT INTO stick (`type`, `brand_id`) VALUES ([type], [brand_id])

INSERT INTO drummer (`name`, `hometown`, `band`, `stick_id`) VALUES ([name], [hometown],
[band], [stick_id])

INSERT INTO plays (`drummer_id`, `kit_id`) VALUES ([drummer_id], [kit_id])
```

DATA MANIPULATION QUERIES (continued)

Deletes:

The following queries delete a user specified row:

```
DELETE FROM brand WHERE brand_id=[brand_id]
```

```
DELETE FROM drum_kit WHERE kit_id=[kit_id]
```

```
DELETE FROM stick WHERE stick_id=[stick_id]
```

```
DELETE FROM drummer WHERE drummer_id=[drummer_id]
```

```
DELETE FROM plays WHERE plays_id=[plays_id]
```

Updates:

The following query updates a drummer value:

```
UPDATE drummer SET name=[name], hometown=[hometown], band=[band], stick_id=[stick_id]
WHERE drummer_id=[drummer_id]
```