Actors:
Guest- anyone who is using the system without being logged in.
Registered User- anyone who is logged into the system with an account in the *Users* table.
Admin- Those with the 'admin' role in the *Users* table
(Note: Registered users inherit the abilities of a Guest, and an admin inherits the abilities of a Registered User)

Guest Cases

Use Case Name: Create a New Account
Actor: Guest
Steps:
1. The system asks the guest for a new password and unique username.
2. If the guest has provided valid input, they confirm the information and a new account is added to the database.

Alternate:
2a. If the user provides a taken username or an invalid password, they are asked to re enter valid information.

SQL Statements:
INSERT INTO users VALUES('sampleusername', 'samplepassword', 'basic')

Use Case Name: Search For/View a Card
Actor: Guest
Steps:
1. The user picks search criteria from name, color, mana cost, artist name, etc.
2. The user provides input for each criteria they selected. (ex: what name? what mana cost?)
3. Search returns matching cards from the DB in a sorted list.
4. User clicks on a card to open the detailed card view.

SQL Statements:
(Search for a card by name)
SELECT * FROM cards c
WHERE (name = 'armistice')

(Search for a split/flip card with some rule-based constraints)
SELECT * FROM card c
INNER JOIN splitflipcard sf ON sf.cardid = c.cardid
INNER JOIN card_format b ON b.cardid = c.cardid
INNER JOIN rulings r ON r.cardid = c.cardid
WHERE (b.formatname = 'standard' AND b.bantype = 'restricted' AND r.textruling LIKE 'Here is some text you may find in a ruling')

(Search for a card that has a ‘blue’ color identity)

```
SELECT * FROM cards c
INNER JOIN card_coloridentity cci ON cci.cardid = cards.cardid
INNER JOIN coloridentity ci ON ci.coloridentityid = cci.coloridentityid
WHERE(ci.color = ‘blue’)
```

(When viewing a split/flip card, the system needs to know how many cards are linked together)

```
SELECT COUNT(*)
FROM splitflipcard
WHERE cardid = 8
```

(Show how many winning tournament decks the card is included in)

```
SELECT COUNT(*)
FROM (SELECT DISTINCT deckid, cardid
FROM card_deck cd
OUTER LEFT JOIN tournamentdeck td ON cd.deckid = td.deckid
WHERE cd.cardid = 156)
```

(Show how many card are banned for a certain format)

```
SELECT COUNT(*)
FROM format_card
WHERE (formatname = ‘sampleformatname’)
```

(Show how many types a particular card has)

```
SELECT COUNT(*)
FROM card_type
WHERE (cardid = 1802)
```

(Show how many cards belong to a certain color identity in a particular set)

```
SELECT COUNT(*)
FROM coloridentity ci
INNER JOIN card_coloridentity cci
INNER JOIN card c ON c.cardid = cci.cardid
INNER JOIN set s ON set.setname = card.setname
WHERE (s.setname = ‘some set name’ AND ci.color = ‘green’)
```

(Count number of cards named ‘forest’)

```
SELECT COUNT(*)
FROM card
WHERE cardname = ‘forest’
```

Use Case Name: Search for a Deck

Actor: Guest

Steps:

1. The user picks search criteria from deck name, name of the user who created the deck, and deck format name.
2. The user provides input for each criteria they selected.
3. Search returns matching decks from the DB in a sorted list.
4. The user can click on a deck to open the detailed deck view.

SQL Statements:

(Search for a deck from the pool of all decks)

```
SELECT d.deckname, d.format, u.username, t.playername
FROM deck d
LEFT OUTER JOIN userdeck u on u.deckid = d.deckid
LEFT OUTER JOIN tournamentreck t on t.deckid = t.deckid
WHERE (d.deckname = ‘decknamegoeshere’ AND t.playername = ‘John Smith’)
```

(Show number of visible decks created by users)

```
SELECT COUNT(*)
FROM userdeck
WHERE visible = TRUE
```

Use Case Name: View a Deck

Actor: guest/registered user/admin

Steps:

1. The user requests to see a deck.
2. The system serves the detailed deck page.

SQL Statements:

(showing the number of cards per mana in the deck view)

```
SELECT COUNT(*)
FROM deck_card dc
INNER JOIN color co ON co.cardid = dc.cardid
WHERE (dc.deckid = 5 AND co.color = ‘green’)
```

Registered User Cases

Use Case Name: Create New Deck

Actor: Registered User

Steps:

1. The system asks for a deck name (unique per user), deck format, and admins select whether the deck is a user deck or a player deck.
2. The user is brought to the deck view for their new deck.

Alternate:

2a. If the user provides a duplicate deck name, no action is taken and they are asked to re-enter a valid deck name.

SQL Statements:

(creating a new invisible user deck)
DECLARE @id INT
INSERT INTO decks (deckname, format) VALUES (‘sampledeckname’, ‘standard’)
SET @id = SCOPE_IDENTITY()
INSERT INTO userdeck(username, visible, deckid)
VALUES (‘samplename’, FALSE, @id)

Use Case Name: Add Cards to Deck
Actor: Registered User
Steps:

1. From the deck view, registered user selects to modify their deck.
2. A dialogue is presented for the registered user to quickly search for cards within the deck’s format and add them to the deck.

Alternate:

1a. If the registered user is viewing a card page, they may add the card directly to one of their own decks

SQL Statements:
(showing how many cards are in the deck being modified)
SELECT COUNT(*)
FROM decks
WHERE deckid = 8

(adding a card to a user’s deck)
INSERT INTO deck_card (deckid, cardid, mainboardqty, sideboardqty) VALUES (1, 523, 1, 0)

(show how many sets are contained in the newest 4 blocks. this calculation will be used to figure out which sets are allowed in the ‘standard’ format)
SELECT COUNT(setname)
FROM (SELECT DISTINCT TOP 4 block FROM sets ORDER BY releasedate DESC)

Admin Cases

Use Case Name: Search for/View Users
Actor: Admin
Steps:

1. The admin requests to see a list of registered users with some or none search criteria.
2. The system provides a view of registered users, each row showing username, regdate, and role.

SQL Statements:
(show admins)
SELECT (username, role, regdate) FROM users WHERE (role = ‘admin’)

(display number of registered users)

```
SELECT COUNT(*) FROM users
```

Use Case Name: Admin Promotion
Actor: Admin
Steps:

1. The system asks the admin for the unique name of the user they'd like to promote.
2. The admin is asked to input their password confirm that they want to make this change to the user in question.

SQL Statements:
(Promote existing user to admin)

```
UPDATE users
SET role = 'admin'
WHERE username = 'searchedusername'
```

Use Case Name: Import Card Data
Actor: Admin
Steps:

1. The admin pulls up the data import page.
2. The admin selects which data they would like to import (entire DB, cards only, rulings, formats, etc).
3. The admin is prompted to select a json file with the updated information..
4. The relevant tables are created from the data in the json.

SQL Statements:
(Create new table, populate it with some data)

```
CREATE TABLE sets(
setName VARCHAR(50),
code VARCHAR(10),
setType VARCHAR(10),
releaseDate VARCHAR(20),
Block VARCHAR(20),
PRIMARY KEY(setName)
)
INSERT INTO sets VALUES ('setname', 'codename', 'typename', 2017-03-20, 'blockname')
```