The Magic DB

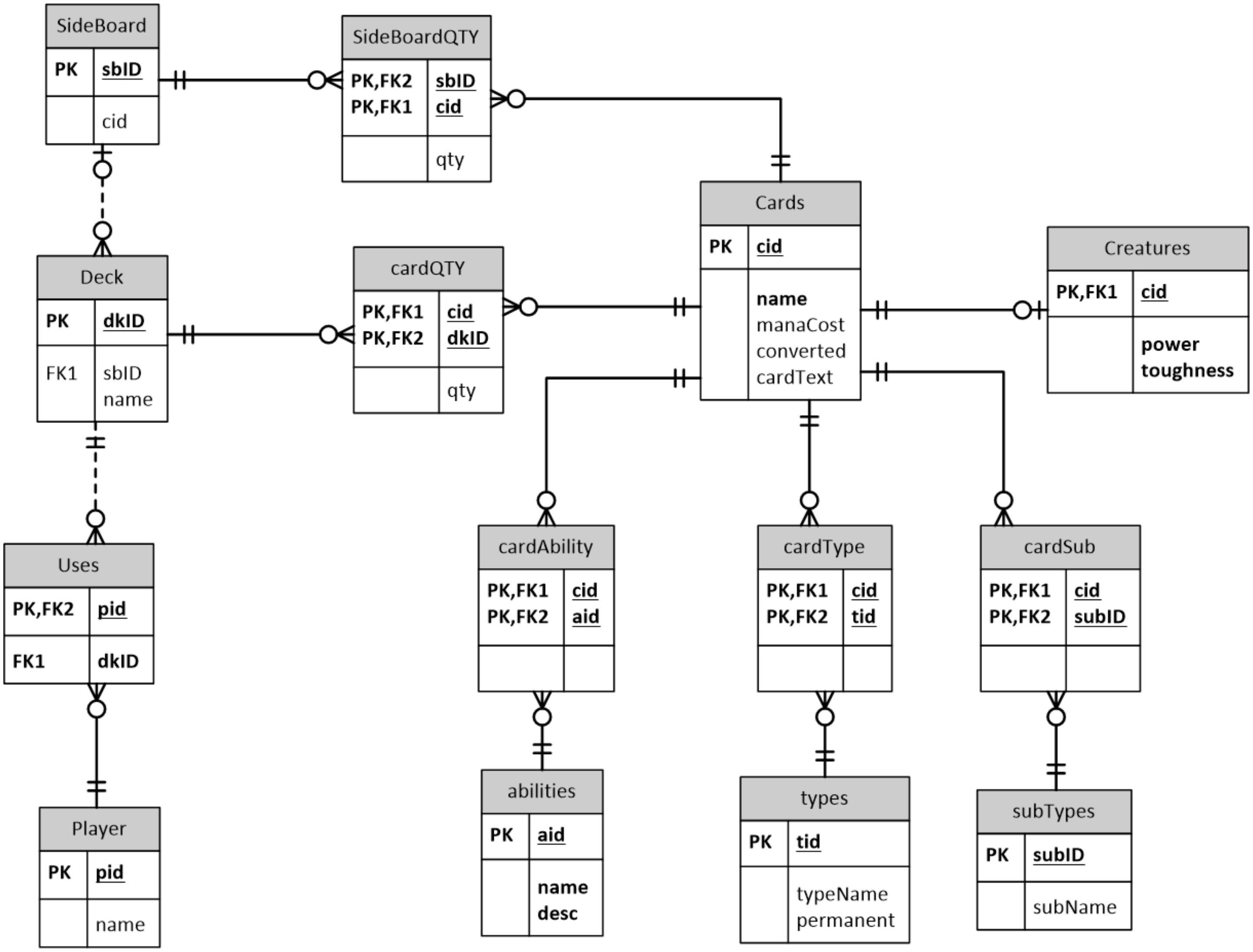
Ryan Owens | Database Systems | 12/9/16

Executive Summary

Magic The Gathering has been one of the longest lasting trading card games created. Since its creation in 1993, Magic has introduced thousands of new cards that open doors to new strategy combinations and alterations. Although there is a massive amount of cards available, it is difficult to find all cards that would work toward a specific strategy, while also maximizing probability of seeing certain cards. My objective is to help users find cards that will help maximize their Deck efficiency through The Magic DB.

The Magic DB, was designed to help users with deck strategy. While the minimum deck size is sixty cards, and the duplicate card maximum being four cards of the same name, it is hard to find cards that make a deck cohesive. Some cards have been reprinted with different names, yet still have the same ability, meaning a deck can contain sets of four for both cards. This structure for finding cards with the same ability along with analyzing card quantity per deck can be used together to give the user information on popular strategies, and maximizing probability.

Entity-Relationship Diagram

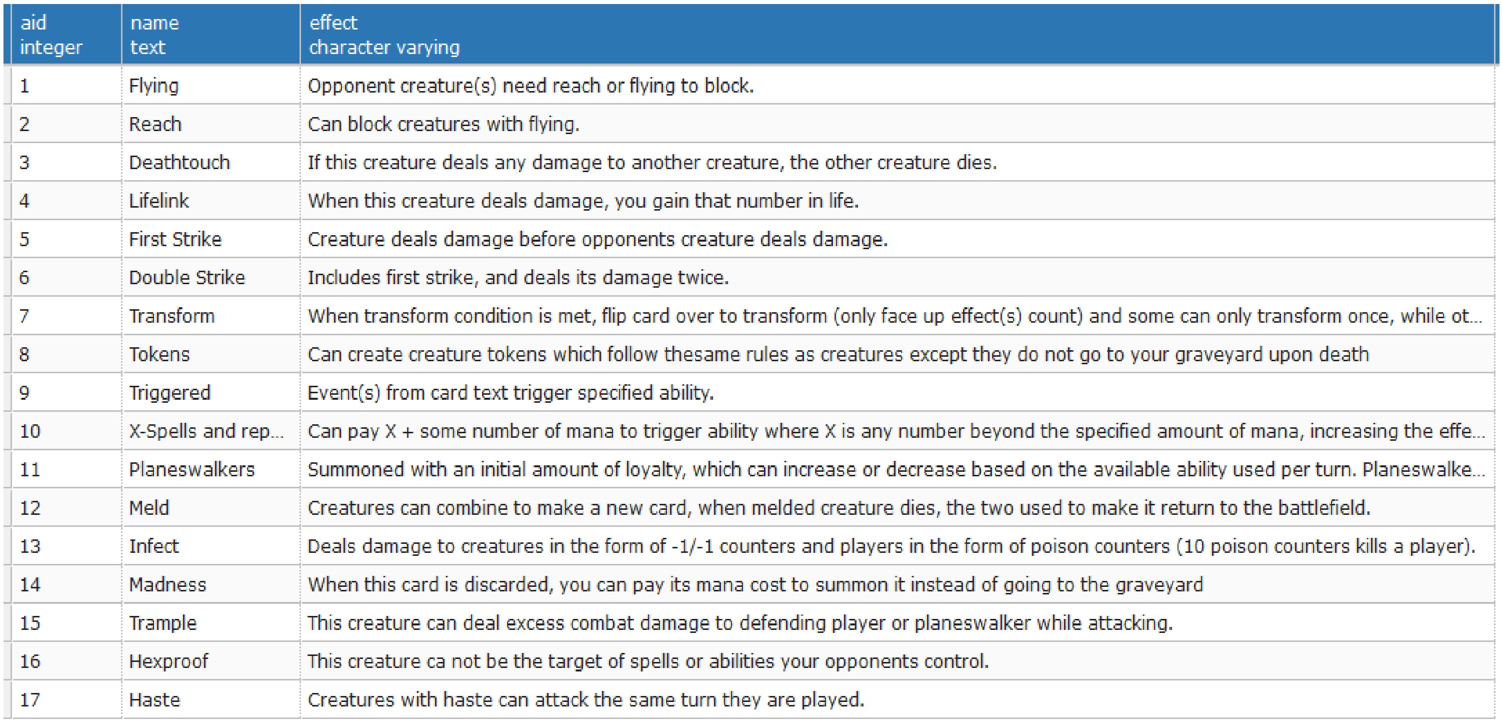


Abilities Table

The abilities table stores the possible abilities that cards might have or provide to other cards.

```
CREATE TABLE Abilities
(
  aid             int not null,
  name            text not null,
  effect                     varchar(250),
 primary key(aid)
);
```

Functional Dependencies: aid → name, effect



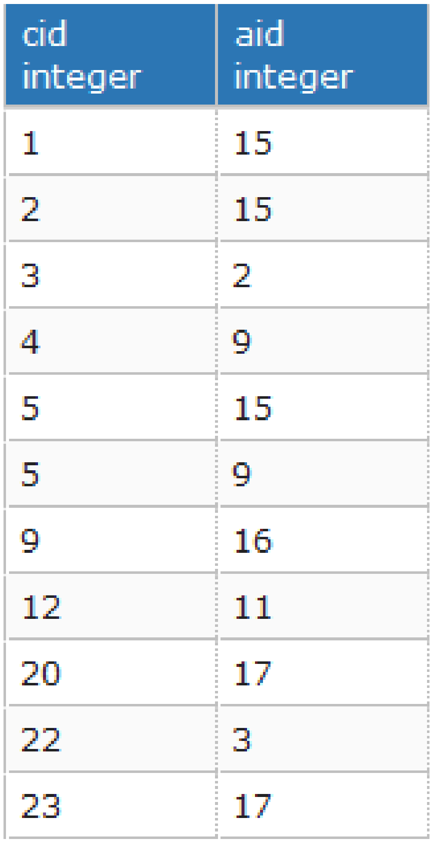
aid integer	name text	effect character varying
1	Flying	Opponent creature(s) need reach or flying to block.
2	Reach	Can block creatures with flying.
3	Deathtouch	If this creature deals any damage to another creature, the other creature dies.
4	Lifelink	When this creature deals damage, you gain that number in life.
5	First Strike	Creature deals damage before opponents creature deals damage.
6	Double Strike	Includes first strike, and deals its damage twice.
7	Transform	When transform condition is met, flip card over to transform (only face up effect(s) count) and some can only transform once, while ot...
8	Tokens	Can create creature tokens which follow thesame rules as creatures except they do not go to your graveyard upon death
9	Triggered	Event(s) from card text trigger specified ability.
10	X-Spells and rep...	Can pay X + some number of mana to trigger ability where X is any number beyond the specified amount of mana, increasing the effe...
11	Planeswalkers	Summoned with an initial amount of loyalty, which can increase or decrease based on the available ability used per turn. Planeswalke...
12	Meld	Creatures can combine to make a new card, when melded creature dies, the two used to make it return to the battlefield.
13	Infect	Deals damage to creatures in the form of -1/-1 counters and players in the form of poison counters (10 poison counters kills a player).
14	Madness	When this card is discarded, you can pay its mana cost to summon it instead of going to the graveyard
15	Trample	This creature can deal excess combat damage to defending player or planeswalker while attacking.
16	Hexproof	This creature ca not be the target of spells or abilities your opponents control.
17	Haste	Creatures with haste can attack the same turn they are played.

CardAbility Table

The card ability table stores cards that are mapped to their ability or abilities.

```
CREATE TABLE cardAbility
(
  cid            int not null,
  aid            int not null,
 primary key(cid)
);
```

Functional Dependencies: cid → aid



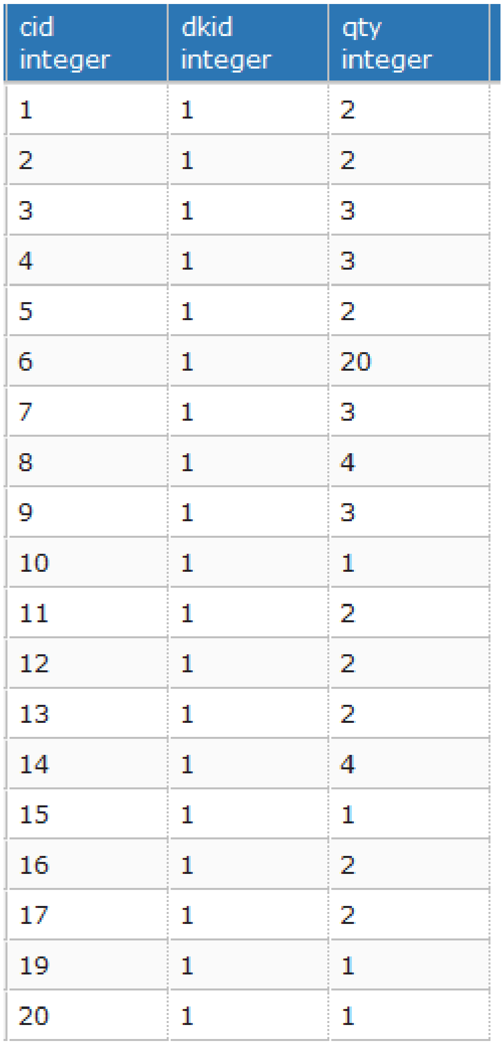
cid integer	aid integer
1	15
2	15
3	2
4	9
5	15
5	9
9	16
12	11
20	17
22	3
23	17

CardQty Table

The card quantity table stores which cards are in each deck, along with the quantity of each card with a max quantity of four per card.

```
CREATE TABLE cardQTY
(
  cid            int not null,
  dkID           int not null,
  qty            int not null,
 primary key (cid, dkID)
);
```

Functional Dependencies: cid, dkID → qty



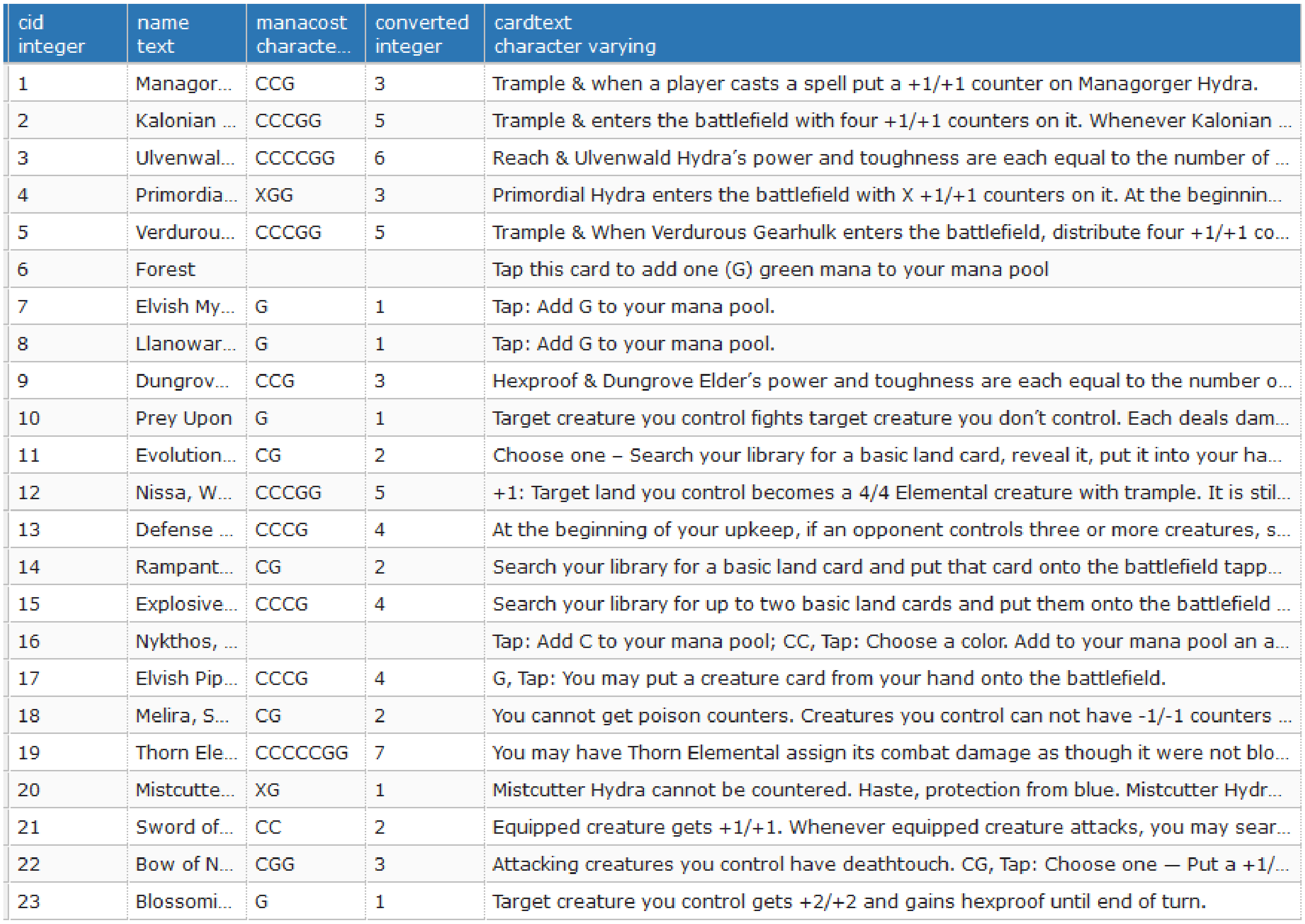
cid integer	dkid integer	qty integer
1	1	2
2	1	2
3	1	3
4	1	3
5	1	2
6	1	20
7	1	3
8	1	4
9	1	3
10	1	1
11	1	2
12	1	2
13	1	2
14	1	4
15	1	1
16	1	2
17	1	2
19	1	1
20	1	1

Cards Table

The cards table stores the information that is generic to every card, except for manaCost and converted(mana cost sum).

```
CREATE TABLE Cards
(
  cid           int not null,
  name          text not null,
  manaCost      varchar(20),
  converted     int,
  cardText      varchar(750),
 primary key(cid)
);
```

Functional Dependencies: cid → name, manaCost, converted, cardText



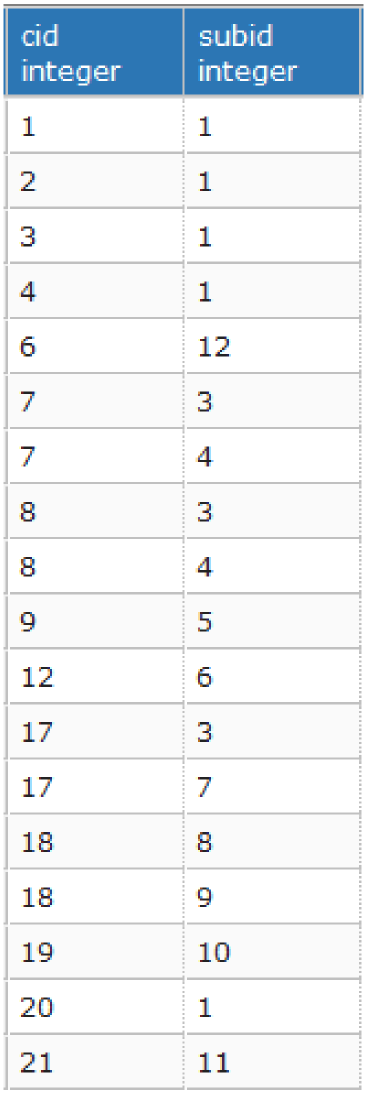
cid integer	name text	manacost characte...	converted integer	cardtext character varying
1	Managor...	CCG	3	Trample & when a player casts a spell put a +1/+1 counter on Managorger Hydra.
2	Kalonian ...	CCCGG	5	Trample & enters the battlefield with four +1/+1 counters on it. Whenever Kalonian ...
3	Ulvenwal...	CCCCGG	6	Reach & Ulvenwald Hydra's power and toughness are each equal to the number of ...
4	Primordia...	XGG	3	Primordial Hydra enters the battlefield with X +1/+1 counters on it. At the beginnin...
5	Verdurou...	CCCGG	5	Trample & When Verdurous Gearhulk enters the battlefield, distribute four +1/+1 co...
6	Forest			Tap this card to add one (G) green mana to your mana pool
7	Elvish My...	G	1	Tap: Add G to your mana pool.
8	Llanowar...	G	1	Tap: Add G to your mana pool.
9	Dungrov...	CCG	3	Hexproof & Dungrove Elder's power and toughness are each equal to the number o...
10	Prey Upon	G	1	Target creature you control fights target creature you don't control. Each deals dam...
11	Evolution...	CG	2	Choose one – Search your library for a basic land card, reveal it, put it into your ha...
12	Nissa, W...	CCCGG	5	+1: Target land you control becomes a 4/4 Elemental creature with trample. It is stil...
13	Defense ...	CCCG	4	At the beginning of your upkeep, if an opponent controls three or more creatures, s...
14	Rampant...	CG	2	Search your library for a basic land card and put that card onto the battlefield tapp...
15	Explosive...	CCCG	4	Search your library for up to two basic land cards and put them onto the battlefield ...
16	Nykthos, ...			Tap: Add C to your mana pool; CC, Tap: Choose a color. Add to your mana pool an a...
17	Elvish Pip...	CCCG	4	G, Tap: You may put a creature card from your hand onto the battlefield.
18	Melira, S...	CG	2	You cannot get poison counters. Creatures you control can not have -1/-1 counters ...
19	Thorn Ele...	CCCCCGG	7	You may have Thorn Elemental assign its combat damage as though it were not blo...
20	Mistcutte...	XG	1	Mistcutter Hydra cannot be countered. Haste, protection from blue. Mistcutter Hydr...
21	Sword of...	CC	2	Equipped creature gets +1/+1. Whenever equipped creature attacks, you may sear...
22	Bow of N...	CGG	3	Attacking creatures you control have deathtouch. CG, Tap: Choose one — Put a +1/...
23	Blossomi...	G	1	Target creature you control gets +2/+2 and gains hexproof until end of turn.

CardSub Table

The card subtype table stores cards and their corresponding subtypes.

```
CREATE TABLE cardSub
(
  cid              int not null,
  subID            int not null,
  primary key(cid)
);
```

Functional Dependencies: cid → subID



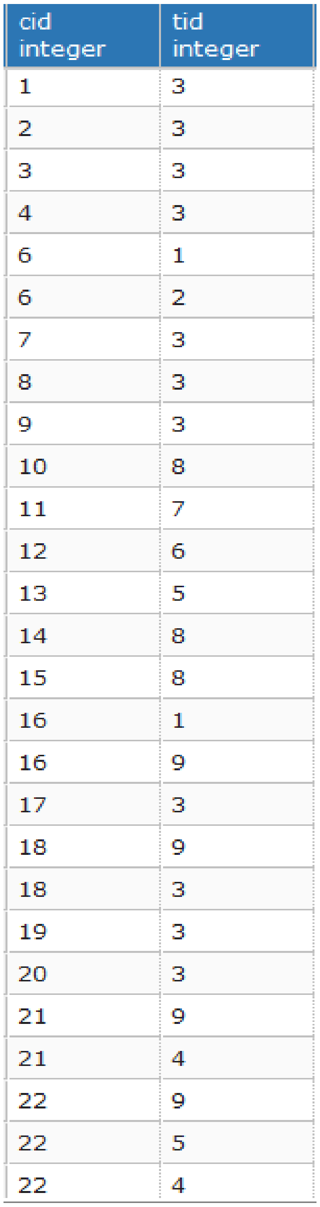
cid integer	subid integer
1	1
2	1
3	1
4	1
6	12
7	3
7	4
8	3
8	4
9	5
12	6
17	3
17	7
18	8
18	9
19	10
20	1
21	11

CardType Table

The card type table stores cards and their corresponding types.

```
CREATE TABLE cardType
(
  cid             int not null,
  tid     int not null,
  primary key(cid)
);
```

Functional Dependencies: cid → tid



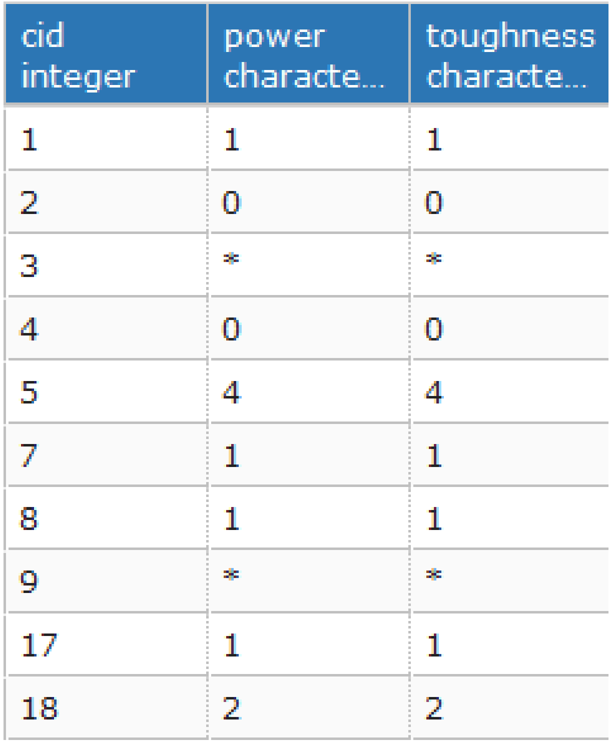
cid	tid
integer	integer
1	3
2	3
3	3
4	3
6	1
6	2
7	3
8	3
9	3
10	8
11	7
12	6
13	5
14	8
15	8
16	1
16	9
17	3
18	9
18	3
19	3
20	3
21	9
21	4
22	9
22	5
22	4

Creatures Table

The creatures table stores cards with some form of power and toughness.

```
CREATE TABLE Creatures
(
  cid            int not null,
  power          varchar(10),
  toughness      varchar(10),
 primary key(cid)
);
```

Functional Dependencies: cid → power, toughness



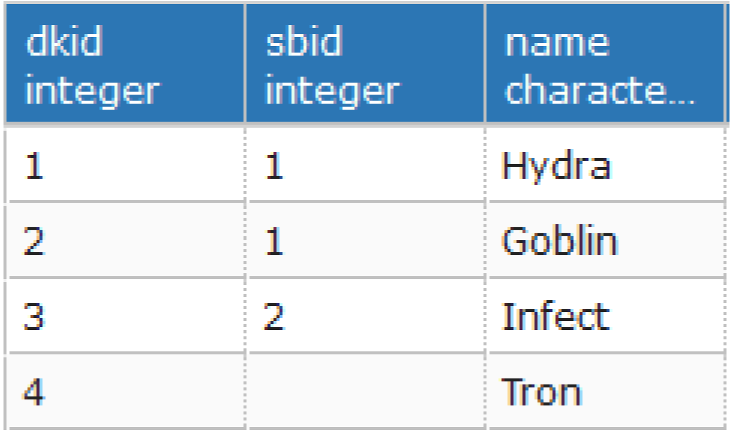
cid integer	power characte...	toughness characte...
1	1	1
2	0	0
3	*	*
4	0	0
5	4	4
7	1	1
8	1	1
9	*	*
17	1	1
18	2	2

Deck Table

The deck table stores the associations between deck names and the sideboard used by that deck.

```
CREATE TABLE Deck
(
  dkID           int not null,
  sbID           int,
  name           varchar(50),
primary key(aid)
);
```

Functional Dependencies: dkID → sbID, name, playedPro



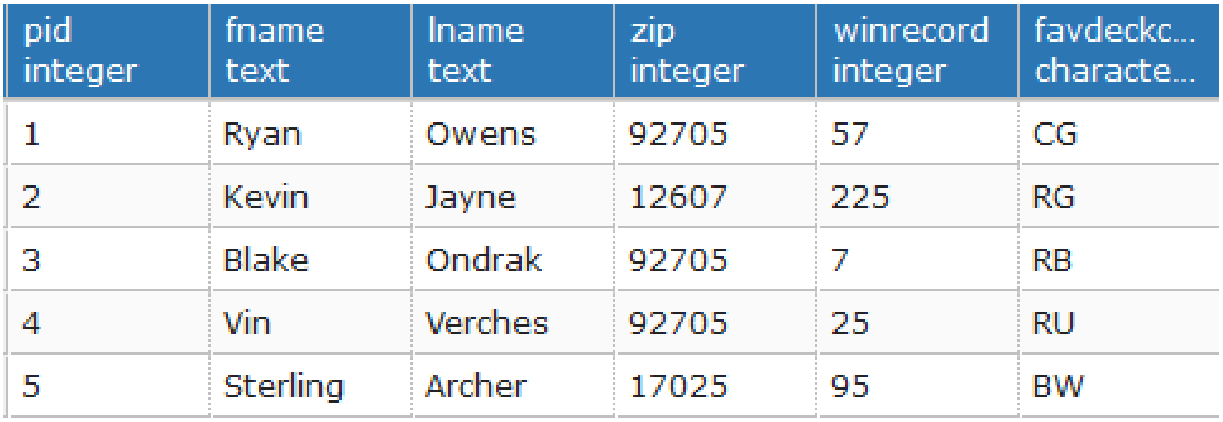
dkid integer	sbid integer	name characte...
1	1	Hydra
2	1	Goblin
3	2	Infect
4		Tron

Player Table

The player table stores the player data so that we can display the types of decks each player owns.

```
CREATE TABLE Player
(
  pid              int not null,
  fName            text not null,
  lName            text not null,
  zip              int,
  winRecord    int,
  favDeckColor varchar(10),
 primary key(pid)
);
```

Functional Dependencies: pid → fName, lName, zip, winRecord, favDeckColor



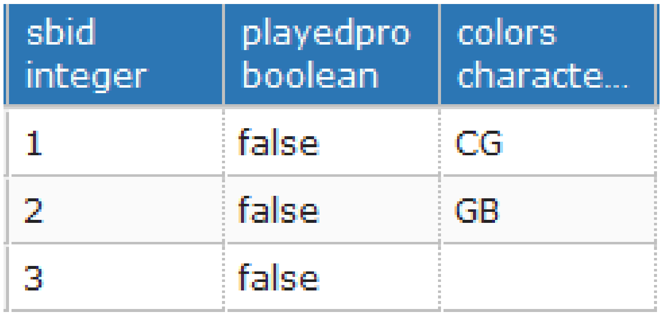
pid integer	fname text	lname text	zip integer	winrecord integer	favdeckc... characte...
1	Ryan	Owens	92705	57	CG
2	Kevin	Jayne	12607	225	RG
3	Blake	Ondrak	92705	7	RB
4	Vin	Verches	92705	25	RU
5	Sterling	Archer	17025	95	BW

SideBoard Table

The side board table lists some existing side board structures, including the mana colors it uses.

```
CREATE TABLE SideBoard
(
  sbID          int not null,
  playedPro     bool,
  colors                varchar(10),
 primary key(sbID)
);
```

Functional Dependencies: sbID → playedPro, colors



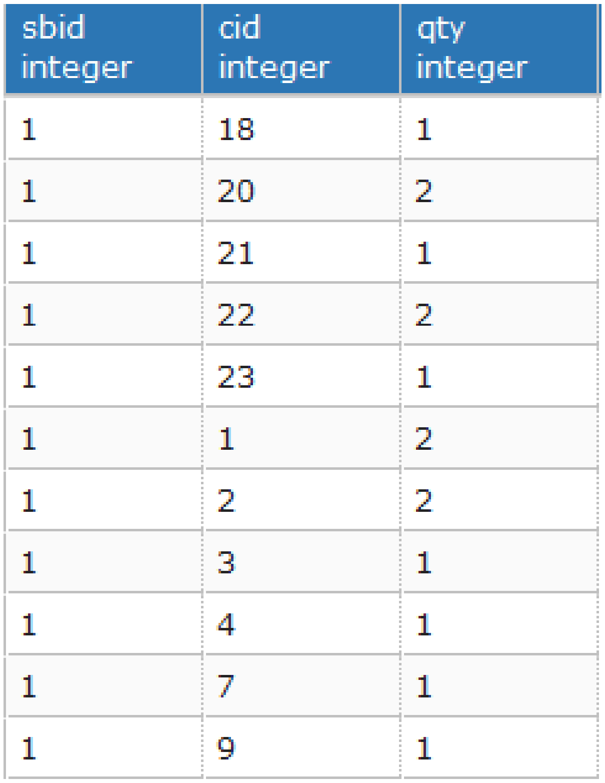
sbid integer	playedpro boolean	colors characte...
1	false	CG
2	false	GB
3	false	

SideBoardQty Table

The side board quantity table stores which cards are in a side board, along with the quantity of each card with a max quantity of four per card & fifteen max total.

```
CREATE TABLE SideBoardQTY
(
  sbID          int not null,
  cid           int not null,
  qty           int not null,
 primary key(sbID, cid)
);
```

Functional Dependencies: sbID, cid → qty



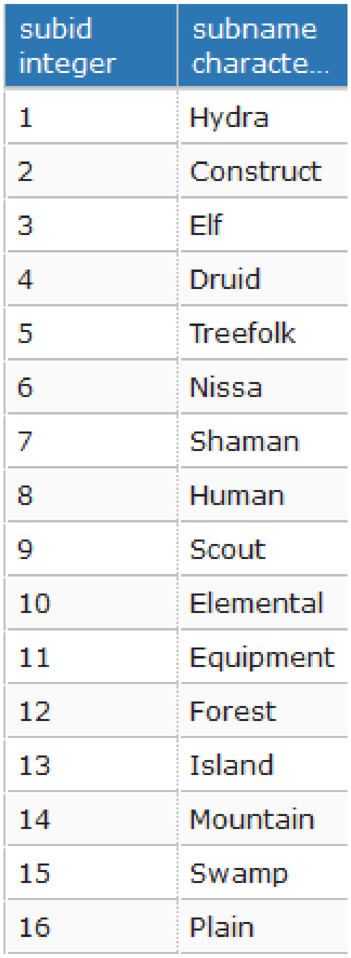
sbid integer	cid integer	qty integer
1	18	1
1	20	2
1	21	1
1	22	2
1	23	1
1	1	2
1	2	2
1	3	1
1	4	1
1	7	1
1	9	1

SubTypes Table

The subtypes table stores the list of card subtypes.

```
CREATE TABLE subTypes
(
  subID          int not null,
  subName        varchar not null,

 primary key(subID)
);
```

Functional Dependencies: subID → subName



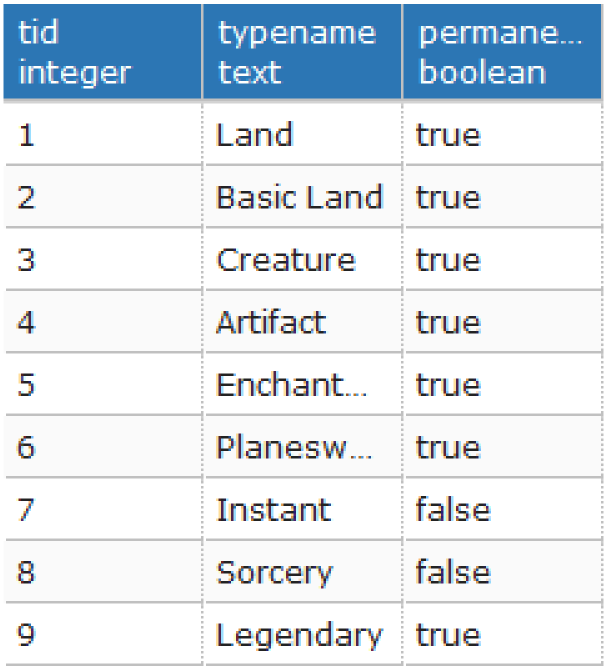
subid integer	subname characte...
1	Hydra
2	Construct
3	Elf
4	Druid
5	Treefolk
6	Nissa
7	Shaman
8	Human
9	Scout
10	Elemental
11	Equipment
12	Forest
13	Island
14	Mountain
15	Swamp
16	Plain

Types Table

The types table stores a list of card primary types.

```
CREATE TABLE types
(
  tid              int not null,
  typeName                   text not null,
  permanent    Boolean,
 primary key(tid)
);
```

Functional Dependencies: tid → typeName, permanent



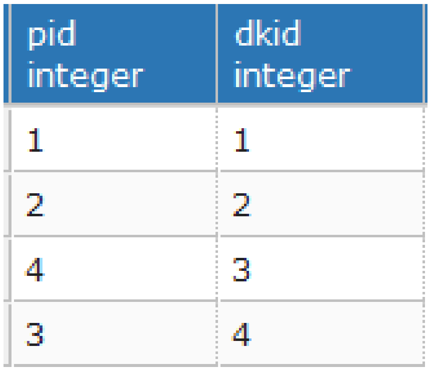
tid integer	typename text	permane... boolean
1	Land	true
2	Basic Land	true
3	Creature	true
4	Artifact	true
5	Enchant...	true
6	Planesw...	true
7	Instant	false
8	Sorcery	false
9	Legendary	true

Uses Table

The uses table stores the deck(s) each player uses.

```
CREATE TABLE Uses
(
  pid              int not null,
  dkID             int not null,
 primary key(pid)
);
```

Functional Dependencies: pid → dkID

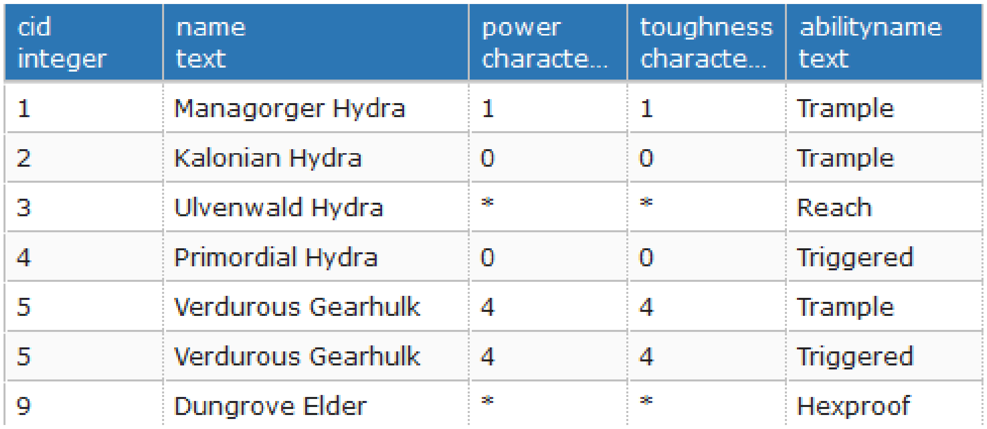


pid integer	dkid integer
1	1
2	2
4	3
3	4

Views

Creature Abilities View

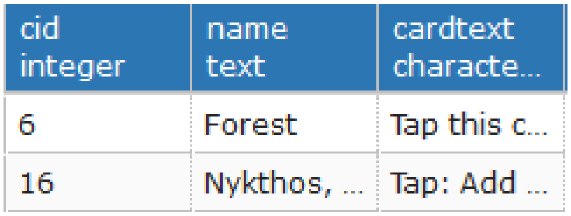
```
CREATE VIEW CreatureAbilities AS
SELECT c.cid, c.name, cr.power, cr.toughness, a.abilityname
FROM cards c JOIN creatures cr ON c.cid = cr.cid
                          JOIN cardAbility ca ON c.cid = ca.cid
                        JOIN abilities a ON ca.aid = a.aid
WHERE c.cid = ca.cid;
```



cid integer	name text	power characte...	toughness characte...	abilityname text
1	Managorger Hydra	1	1	Trample
2	Kalonian Hydra	0	0	Trample
3	Ulvenwald Hydra	*	*	Reach
4	Primordial Hydra	0	0	Triggered
5	Verdurous Gearhulk	4	4	Trample
5	Verdurous Gearhulk	4	4	Triggered
9	Dungrove Elder	*	*	Hexproof

Lands View

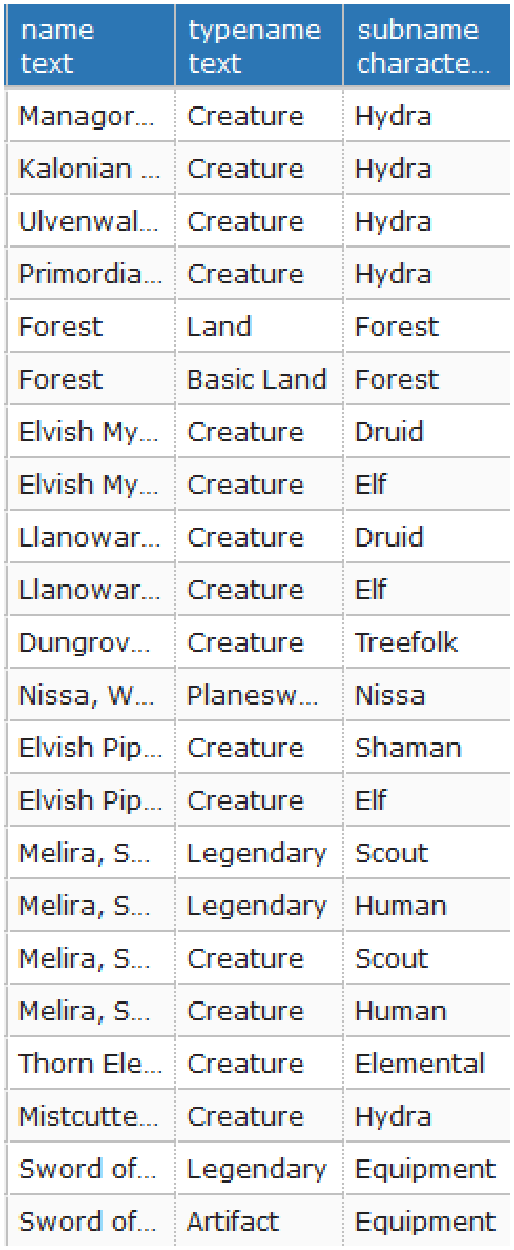
```
CREATE VIEW lands AS
SELECT c.cid, c.name, c.cardtext
FROM cards c JOIN cardType ct ON c.cid = ct.cid
                              JOIN types t ON ct.tid = t.tid
WHERE t.typename = 'Land';
```



cid integer	name text	cardtext characte...
6	Forest	Tap this c...
16	Nykthos, ...	Tap: Add ...

Card Types & Subtypes View

```
CREATE VIEW Card_Types_Subtypes AS

SELECT c.name, t.typename, s.subname

FROM cards c JOIN cardType ct ON c.cid = ct.cid

                    JOIN types t ON ct.tid = t.tid

                    JOIN cardSub cs ON c.cid = cs.cid

                    JOIN subTypes s ON cs.subid = s.subid

WHERE ct.cid = cs.cid;
```

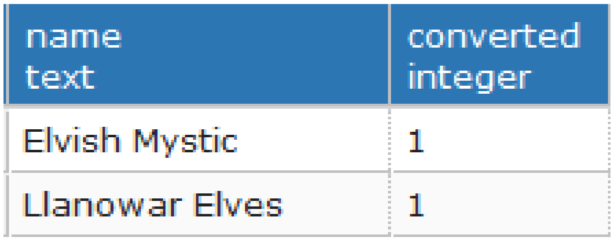


name	typename	subname
text	text	characte...
Managor...	Creature	Hydra
Kalonian ...	Creature	Hydra
Ulvenwal...	Creature	Hydra
Primordia...	Creature	Hydra
Forest	Land	Forest
Forest	Basic Land	Forest
Elvish My...	Creature	Druid
Elvish My...	Creature	Elf
Llanowar...	Creature	Druid
Llanowar...	Creature	Elf
Dungrov...	Creature	Treefolk
Nissa, W...	Planesw...	Nissa
Elvish Pip...	Creature	Shaman
Elvish Pip...	Creature	Elf
Melira, S...	Legendary	Scout
Melira, S...	Legendary	Human
Melira, S...	Creature	Scout
Melira, S...	Creature	Human
Thorn Ele...	Creature	Elemental
Mistcutte...	Creature	Hydra
Sword of...	Legendary	Equipment
Sword of...	Artifact	Equipment

Report

Sample report for card text - see what cards have the same abilities to bypass the four card limit in a deck.

```
SELECT c.name, c.converted
FROM cards c, cardqty cq, cards c1
WHERE cq.cid = c.cid
AND c.cardtext = c1.cardtext
AND c.cid != c1.cid
GROUP BY c.name, c.converted;
```

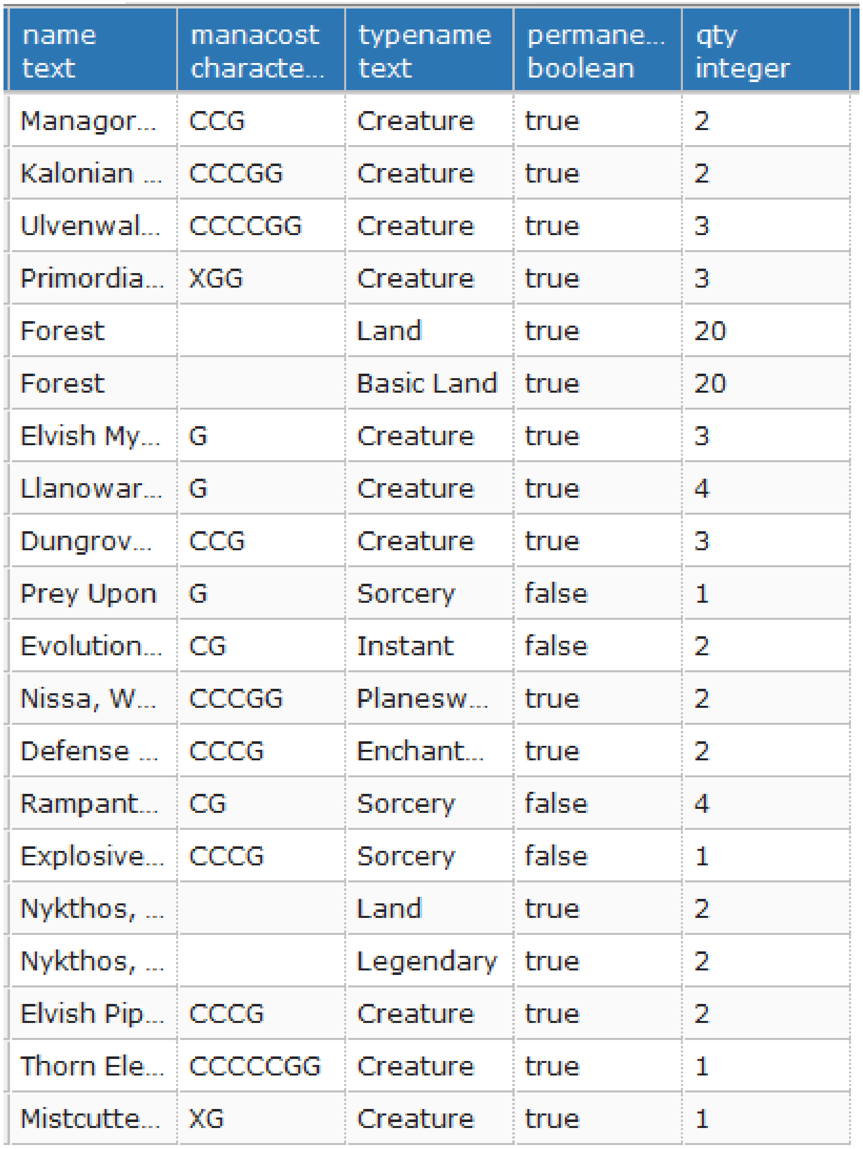


name text	converted integer
Elvish Mystic	1
Llanowar Elves	1

Deck_Cards Stored Procedure

```
CREATE OR REPLACE FUNCTION Deck_Cards(deckID integer) RETURNS

TABLE(name TEXT, manaCost VARCHAR, typename TEXT, permanent BOOLEAN, qty INTEGER) AS

$$

DECLARE

BEGIN

RETURN QUERY

SELECT c.name, c.manaCost, t.typename, t.permanent, cq.qty

FROM Cards c JOIN Cardqty cq ON c.cid = cq.cid

                         JOIN cardType ct ON c.cid = ct.cid

                         JOIN types t ON ct.tid = t.tid

WHERE cq.dkid = deckID;

END;

$$ LANGUAGE plpgsql;
```

Sample output for Deck_Cards:



name text	manacost characte...	typename text	permane... boolean	qty integer
Managor...	CCG	Creature	true	2
Kalonian ...	CCCGG	Creature	true	2
Ulvenwal...	CCCCGG	Creature	true	3
Primordia...	XGG	Creature	true	3
Forest		Land	true	20
Forest		Basic Land	true	20
Elvish My...	G	Creature	true	3
Llanowar...	G	Creature	true	4
Dungrov...	CCG	Creature	true	3
Prey Upon	G	Sorcery	false	1
Evolution...	CG	Instant	false	2
Nissa, W...	CCCGG	Planesw...	true	2
Defense ...	CCCG	Enchant...	true	2
Rampant...	CG	Sorcery	false	4
Explosive...	CCCG	Sorcery	false	1
Nykthos, ...		Land	true	2
Nykthos, ...		Legendary	true	2
Elvish Pip...	CCCG	Creature	true	2
Thorn Ele...	CCCCCGG	Creature	true	1
Mistcutte...	XG	Creature	true	1

Deck size Trigger

```
CREATE OR REPLACE FUNCTION Deck_size() RETURNS trigger AS
$$
DECLARE
        dID integer
        total integer := 0;
        currRec record;
BEGIN
     dID = NEW.dkID
     FOR currRec IN
        SELECT CardQty.qty
        FROM CardQty
        WHERE NEW.dkID = CardQty.dkID
    LOOP
        total:= total + currRec.qty;
    END LOOP;
  IF total > 60 THEN
        RAISE NOTICE 'The deck has % too many cards', total;
        RETURN NULL;
  ELSE
        RETURN NEW;
  END IF;
END;
$$ LANGUAGE plpgsql;
```

Side board size Trigger

```
CREATE OR REPLACE FUNCTION Side_board_size() RETURNS trigger AS
$$
DECLARE
      sbID integer;
      total integer := 0;
      currRec record;
BEGIN
    sbID = NEW.sbID
    FOR currRec IN
      SELECT sideboardQty.qty
      FROM sideboardQty
      WHERE NEW.sbID = sideboardQty.sbID
LOOP
      total:= total + currRec.qty;
END LOOP;
  IF total > 15 THEN
      RAISE NOTICE 'The side board has % too many cards', total;
      RETURN NULL;
  ELSE
      RETURN NEW;
  END IF;
END;
$$ LANGUAGE plpgsql;
```

Security

Roles – currently only Admin and User are supported

```
CREATE ROLE Admin;
```

```
CREATE ROLE User;
```

Admin:

```
GRANT SELECT, INSERT, UPDATE, DELETE ON Abilities TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON CardAbility TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON CardQTY TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON CardSub TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON Cards TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON CardType TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON Creatures TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON Deck TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON Player TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON Sideboard TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON Sideboardqty TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON SubTypes TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON Types  TO Admin;
GRANT SELECT, INSERT, UPDATE, DELETE ON Uses TO Admin;
```

User:

```
REVOKE ALL PRIVILEGES ON Abilities TO User;
REVOKE ALL PRIVILEGES ON CardAbility TO User;
REVOKE ALL PRIVILEGES ON CardQTY TO User;
REVOKE ALL PRIVILEGES ON CardSub TO User;
REVOKE ALL PRIVILEGES ON Cards TO User;
REVOKE ALL PRIVILEGES ON CardType TO User;
REVOKE ALL PRIVILEGES ON Creatures TO User;
REVOKE ALL PRIVILEGES ON Deck TO User;
REVOKE ALL PRIVILEGES ON Sideboard TO User;
REVOKE ALL PRIVILEGES ON Sideboardqty TO User;
REVOKE ALL PRIVILEGES ON SubTypes TO User;
REVOKE ALL PRIVILEGES ON Types  TO User;
REVOKE ALL PRIVILEGES ON Uses TO User;

GRANT SELECT, INSERT ON Player TO User;
```

Implementation Notes

- To simplify deck quantity requirements, I made the deck size maximum 60 to maximize probability.

Known Problems

- Inclusion of card probability based on a decks card quantity

- Creation of a view that groups cards together based on strategic playstyle.

Future Enhancements

- Deck Comparison for decks with similar strategic styles, and probabilities

- Possible card replacement suggestions for cards that work well with specific strategies.

- Suggesting a quantity of land cards by color, based on specified strategy or current deck composition.