Thomas Luong
Kim McLeod

League of Legends Champion Database

DATABASE OUTLINE

We will be making a database representing the champions in the game League of Legends. This game has many champions that has different combat styles, roles, item synergy, and their homeworlds which will provide enough complexity for a database. It is convenient to map out for each champion, a summary about how they are played in the game.

Entities in the database are:

- Champions: Each champion is a character in the game League of Legends. Champions have a lane, 5-6 items, one homeworld, and 1-2 gameplay roles.

- Items: Items that are available in game. Each champion will have five to six items associated with them, this is their “recommended build” for many games.

- Lanes: The five lanes available to be taken up in dynamic queue.

- Gameplay: Each champion has one or two roles they fill in game, which determines how they play and what items they will buy.

- Homeworlds: Where each champion is born or resides in.

Relationships between the entities are:

- Champions synergize with what item - many-to-many relationships, as champions can use many items, and an item can belong to many champions.

- Champions do better in specific lanes - many-to-many relationship as champions can belong to multiple lanes, and a lane can be taken up by any number of champions.

- Champions and their type of gameplay/role - many-to-many relationship as champions can belong to a combination of roles and a role can be filled by many champions.

Database Outline in Words

The Champions table includes champions which have a champion_id, a homeworld_id, a name and a damage type. The primary key is the champion_id. champion_id will also be used as a foreign key for the Lanes, Items and Roles tables.

The homeworld_id is a foreign key to homeworld_id of the Homeworld table. There is a ON DELETE RESTRICT ON UPDATE CASCADE between the two. Meaning the entry cannot be deleted if there is at least one link between a homeworld and a champion.

In the Homeworld table, there is a homeworld_id, homeworld_name, and homeworld_description. The primary key is the id.

In the Lane table, there is the lane_id and lane_name. The primary key is the id.

The Lanes table connects the Lane and Champion tables together through foreign keys on both their respective champion_ids and lane_ids. Both of which have ON DELETE RESTRICT ON UPDATE CASCADE constraints.

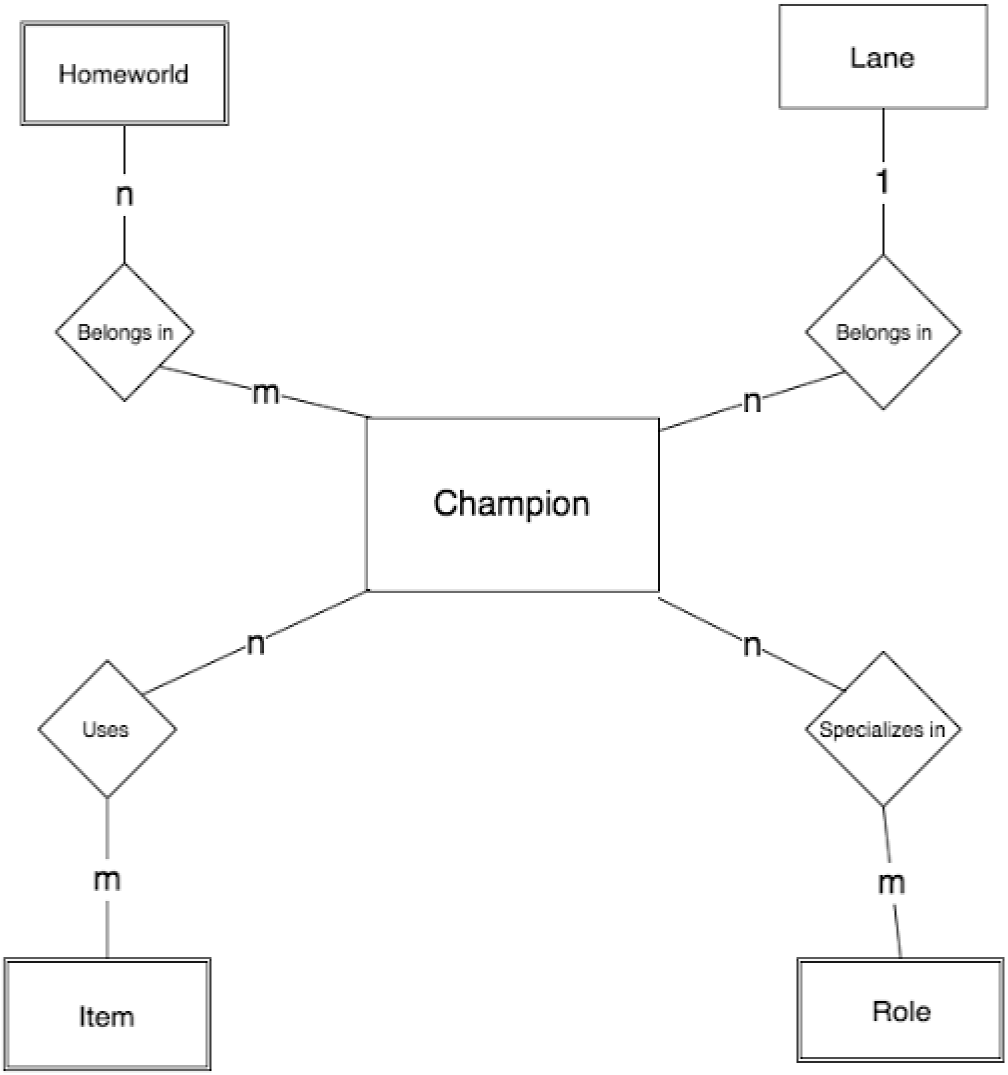
In the Role table, there is the role_id and role_name. The primary key is the id.

The Roles table connects the Role and Champion tables together through foreign keys on both their respective champion_ids and role_ids. Both of which have ON DELETE RESTRICT ON UPDATE CASCADE constraints.

In the Item table, there is the item_id and item_name. The primary key is the id.

The Items table connects the Item and Champion tables together through foreign keys on both their respective champion_ids and item_ids. Both of which have ON DELETE RESTRICT ON UPDATE CASCADE constraints.

ER DIAGRAM


Homeworld
n
Belongs in
m
Lane
1
Belongs in
n
Champion
n
Uses
m
Item
n
Specializes in
m
Role

SCHEMA

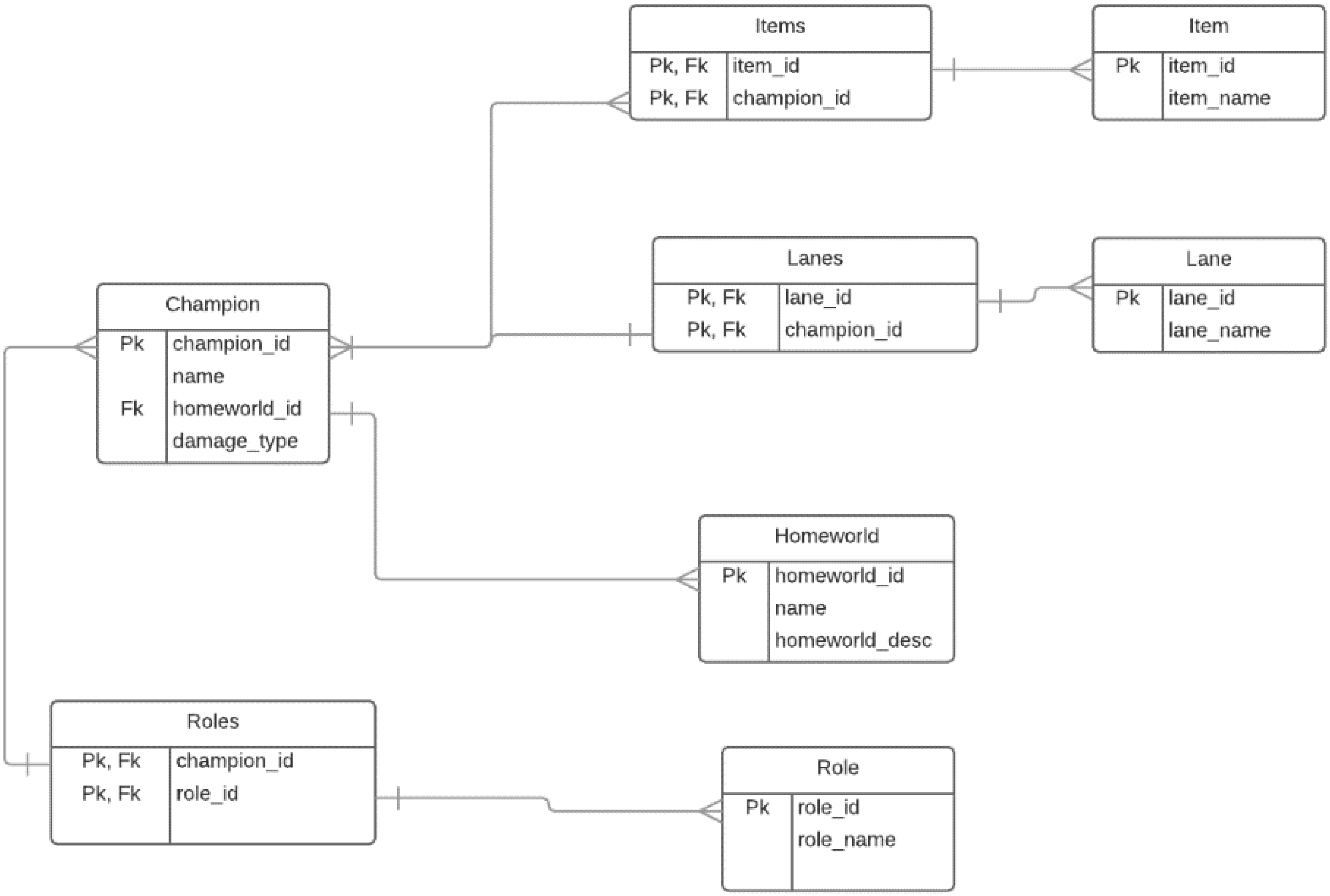

Items
Pk, Fk item_id
Pk, Fk champion_id
Item
Pk item_id
item_name
Lanes
Pk, Fk lane_id
Pk, Fk champion_id
Lane
Pk lane_id
lane_name
Champion
Pk champion_id
name
Fk homeworld_id
damage_type
Homeworld
Pk homeworld_id
name
homeworld_desc
Roles
Pk, Fk champion_id
Pk, Fk role_id
Role
Pk role_id
role_name

TABLE CREATION QUERIES

```
-- Sets foreign key checks off so we can add whole tables
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL';



DROP TABLE IF EXISTS Homeworld;
DROP TABLE IF EXISTS Champion;
DROP TABLE IF EXISTS Lane;
DROP TABLE IF EXISTS Lanes;
DROP TABLE IF EXISTS Item;
DROP TABLE IF EXISTS Items;
DROP TABLE IF EXISTS Role;
DROP TABLE IF EXISTS Roles;


--
-- Table structure for table `Homeworld`
--

CREATE TABLE Homeworld (
  homeworld_id TINYINT UNSIGNED NOT NULL AUTO_INCREMENT,
  homeworld_name VARCHAR(50) NOT NULL,
  homeworld_desc TEXT,
  PRIMARY KEY  (homeworld_id)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;


--
--  Table structure for table `champion
--

CREATE TABLE Champion (
  champion_id INT(11) UNSIGNED NOT NULL AUTO_INCREMENT,
  name VARCHAR(45) NOT NULL,
  homeworld_id TINYINT UNSIGNED NOT NULL,
  damage_type VARCHAR(45) NOT NULL,   -- AP, AD, MIXED

  CONSTRAINT FK_homeworld_id FOREIGN KEY (homeworld_id)
    REFERENCES Homeworld (homeworld_id) ON DELETE RESTRICT ON UPDATE
CASCADE,

  PRIMARY KEY  (champion_id),
  KEY idx_champion_name (name)

)ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```sql
--
-- Table structure for table `Lane`
--


CREATE TABLE Lane (
  lane_id TINYINT UNSIGNED NOT NULL AUTO_INCREMENT,
  lane_name VARCHAR(25) NOT NULL,
  PRIMARY KEY  (lane_id)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;



--
-- Table structure for table `Lanes`
--

CREATE TABLE Lanes (
  lane_id TINYINT UNSIGNED NOT NULL,
  champion_id INT(11) UNSIGNED NOT NULL,
  PRIMARY KEY  (lane_id, champion_id),

CONSTRAINT FK_Lanes_champion_id FOREIGN KEY (champion_id)
    REFERENCES Champion (champion_id) ON DELETE RESTRICT ON UPDATE
CASCADE,

CONSTRAINT FK_Lanes_lane_id FOREIGN KEY (lane_id)
    REFERENCES Lane (lane_id) ON DELETE RESTRICT ON UPDATE CASCADE

)ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Table structure for table `Role`
--

CREATE TABLE Role (
  role_id TINYINT UNSIGNED NOT NULL AUTO_INCREMENT,
  role_name VARCHAR(25) NOT NULL,
  PRIMARY KEY  (role_id)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
--
-- Table structure for table `Roles`
--

CREATE TABLE Roles (
  role_id TINYINT UNSIGNED NOT NULL,
  champion_id INT(11) UNSIGNED NOT NULL,
  PRIMARY KEY  (role_id, champion_id),

  CONSTRAINT FK_Roles_champion_id FOREIGN KEY (champion_id)
    REFERENCES Champion (champion_id) ON DELETE RESTRICT ON UPDATE
CASCADE,

  CONSTRAINT FK_Roles_role_id FOREIGN KEY (role_id)
    REFERENCES Role (role_id) ON DELETE RESTRICT ON UPDATE CASCADE

)ENGINE=InnoDB DEFAULT CHARSET=utf8;




--
-- Table structure for table `Item`
--


CREATE TABLE Item (
  item_id TINYINT UNSIGNED NOT NULL AUTO_INCREMENT,
  item_name VARCHAR(25) NOT NULL,
  PRIMARY KEY  (item_id)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;




--
-- Table structure for table `Items`
--

CREATE TABLE Items (
  item_id TINYINT UNSIGNED NOT NULL,
  champion_id INT(11) UNSIGNED NOT NULL,
  PRIMARY KEY  (item_id, champion_id),

  CONSTRAINT FK_Items_item_id FOREIGN KEY (item_id)
    REFERENCES Item (item_id) ON DELETE RESTRICT ON UPDATE CASCADE,

  CONSTRAINT FK_Items_champion_id FOREIGN KEY (champion_id)
    REFERENCES Champion (champion_id) ON DELETE RESTRICT ON UPDATE
CASCADE

)ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

GENERAL USE QUERIES

```
-- Adds new Champion to the Champion table
INSERT INTO Champion (homeworld_id, name, damage_type)
VALUES ((SELECT homeworld_id FROM Homeworld WHERE
Homeworld.homeworld_name LIKE [homeworld]), [name], [damage_type]);
```

```
-- Adds new Role to existing Champion, updates Roles table
INSERT INTO Roles (champion_id, role_id)
VALUES ((SELECT champion_id FROM Champion WHERE name LIKE [champion]),
(SELECT role_id FROM Role WHERE role_name LIKE [role]));
```

```
-- Adds new Item to existing Champion, updates Items table
INSERT INTO Items (item_id, champion_id)
VALUES ((SELECT item_id FROM Item WHERE item_name LIKE [item]),
(SELECT champion_id FROM Champion WHERE name LIKE [champion]));
```

```
-- Adds new Lane to existing Champion, updates Lanes table
INSERT INTO Lanes (lane_id, champion_id)
VALUES ((SELECT lane_id FROM Lane WHERE lane_name LIKE [lane]),
(SELECT champion_id FROM Champion WHERE name LIKE [champion]));
```

```
-- Adds new Role to Role table
INSERT INTO Role (role_name) VALUES ([role]);
```

```
-- Adds new Item to Item table
INSERT INTO Item (item_name) VALUES ([name]);
```

```
-- Adds new Lane to Lane table
INSERT INTO Lane (lane_name) VALUES ([name]);
```

```sql
-- Adds new Homeworld to Homeworld table
INSERT INTO Homeworld (homeworld_name, homeworld_desc)
VALUES ([name], [description]);
```

```sql
-- Searches database and returns Champion's name, roles, homeworld,
items, and damage type
SELECT DISTINCT name, damage_type, role_name, homeworld_name,
lane_name, item_name
FROM Champion
INNER JOIN Roles ON Champion.champion_id = Roles.champion_id
INNER JOIN Role ON Roles.role_id = Role.role_id
INNER JOIN Homeworld ON Champion.homeworld_id = Homeworld.homeworld_id
INNER JOIN Lanes ON Champion.champion_id = Lanes.champion_id
INNER JOIN Lane ON Lanes.lane_id = Lane.lane_id
INNER JOIN Items ON Champion.champion_id = Items.champion_id
INNER JOIN Item ON Items.item_id = Item.item_id
WHERE Champion.name LIKE [name];
```

```sql
-- Updates Champion's name in Champion table
UPDATE Champion
SET name = [new_name]
WHERE champion_id = [champion_id]
```