Evie Phan
Database Managment i257
Date:12/10/12

Final Project: Personal Database Project Design

My project is a relational database about a person versus person (PVP) online game called League of Legends (LOL). LOL is a multiplayer online battle arena game where two teams play against each other. There are 3-5 people on each team (the number of people on a team depends on which arena map the player chooses). The objective of the game is to destroy a structure in the opposing team's base. During the game, each player controls one hero (champion) who has a variety of special abilities that can attack other champions and structures.

This game has become extremely popular since its launch in 2009 and currently has players all over the world. There are world wide LOL tournaments. Last year $3 million dollars in cash was given out as prizes.

There are a few websites that have helpful guidelines for players on specific champions, but these guidelines are scattered. My purpose for creating this database is to aggregate all this information and help answer game-related questions for players who want to become more proficient in the game. For example:

1. What champions or items are good at countering another champion?
2. What items should I buy for a champion?
3. How much does a certain build cost?
4. Which champions work well together?

While the game is in progress there are certain items that a player can buy for her champion. Depending on how the player wants to play, her in game items may lean towards one of the following play styles: Support (a support champion that helps other players and does not take credit for the kill), Jungler (a champion that roams in the jungle and kills off neutral monsters to obtain buffs that will aide in ambushing the opposing team), an Attack Damage Carry (a champion that has very high physical attack damage), a Mage (a champion that uses only magic spells to cause damage), etc. The set of items that a player purchases to play in a certain style is known as a Build.

In situations where one champion is over powering another (either from their special abilities or a Build), a “Counter” is necessary in order to avoid getting killed by the opposing champion. Some examples of Counters are:

- selecting a champion that can cancel out the other champion’s abilities (this is good for knowing which lane to take during early game to avoid deaths which could cause your team to be behind in gold and experience).
- knowing what items to buy to counter the damage dealt by the opposing champion.
- knowing how to get out of certain life or death situations.

Originally, I was planning to include information about the individual abilities, but it turned out that it’s hard to classify them: there are a lot of nuances and exceptions in how each ability works. So I decided to exclude abilities from the database.

On the other hand, as I was exploring the idea further, I also found more new information that can be added to the database. For example, some items combine into other items. I included the “recipes” that make items as a new relationship that connects the Result to the Component item.

All the tables are normalized.

The Champions table contains all the champions in the game. I use champion name as the primary key. The Champions table also contains all the specs for each champion. For example, some fields rate the champion from 1 to 10 in terms of their strength, etc.

The CounterChampions table provides information about which champion can be countered by another champion.

The Synergy table provides information about which champion works well with another champion.

The Items table provides information about all the items that are available in the game. The main field is the Item Name (which is the primary key), and the rest of the fields contains the stats of the items. For example, how much the item costs in game, and what types of boosts the item provides the champion (i.e. health regeneration, Mana or magic regeneration, attack damage, magic damage, etc.).

The CounterItems table provides information about which item is good to get for countering an enemy champion.

The CounterMechanics table provides information about what techniques to use to counter an enemy champion. For example: the champion “Caitlyn” is a damage dealer, to counter Caitlyn, a technique to use would be “long range harass.” (Long range harass means to shoot at her from afar.)

The ItemBuilds table provides information on what items to build for a champion depending on the play style. For example, if a champion wants to play as a magic user, then building “Ability Power” (AP) items would be recommended. In addition to the columns Item, Champion, and BuildType, this table has a column BuildId. I need this column, BuildId, because two different types of builds could be for AP but they are totally different builds.

The ItemRecipes table provides information about what items builds into what item. For example, if I wanted to build a “Frozen Heart,” which is an item that gives the champion extra magic and armor, it would require 2 Cloth Armors and a Glacial Shroud. The fields in this table will be: the ID, Result, and Component. Where Result is the item I want to build and Component are items that is used to build the Result item. The ID field is added to the table and is the primary key. The ID field was added because the table needed a primary key so that it would be easier in the future to link other tables to this table. Unlike other tables where I used the pair of foreign keys as primary keys, I did not use Result and Component as a primary key because there are multiple low end items that are needed to make a high end item so it is not unique.

The Lanes Table contains information about the different lanes champions can take during the game. Lanes are based on the map zones of the game. This map in the game is called “Summoner’s Rift.” There are 3 main lanes which are the top lane, the middle lane, and the bottom lane. Then there is the jungle. Champions can go in any of those lanes during the game, but some lanes are recommended for certain champions depending on the champions’ abilities and what types of items they build.

The Laning Table provides recommendations for which lanes champions should take. Since the game is based on strategy, all recommendations are based on my personal experiences and experiences of others who have shared information with me. The fields in this table are: Champion and Lane. The table lists all the recommended lanes that a champion should take.

Having played this game on a regular basis, I will be using my own experience as a source for creating the database as well as the following websites: Leagueoflegends.com, mobafire.com/league-of-legends/build/, leagueoflegends.wikia.com/wiki/Crowd_control, and lolcounter.com/list.php.

Problems that came up during my database development was the gathering of all the information. Since there are a few website sources I have to aggregate, it took a lot of time to gather all of the

information. Another difficulty was creating meaningful queries that can add value to the existing data. Since the game gets new patches every couple of weeks, maintaining fresh data will also be an issue.

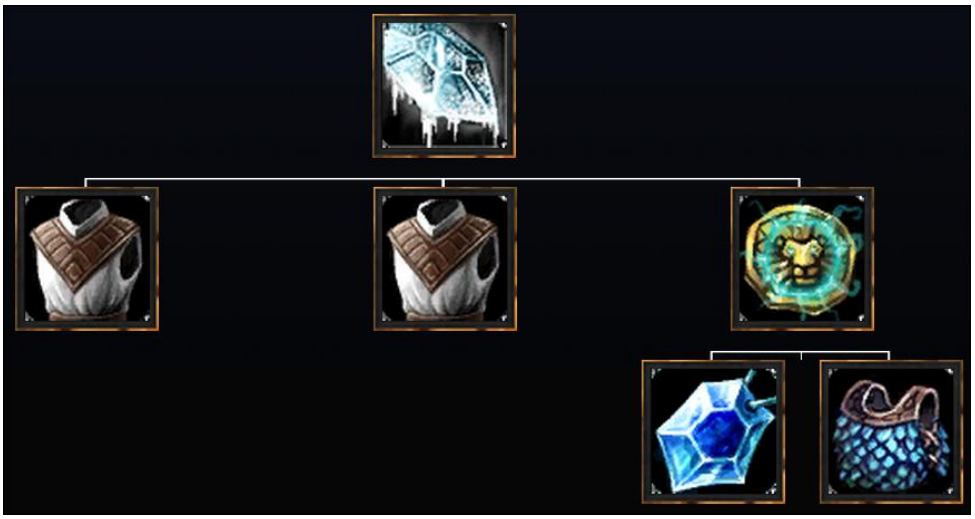
Issues that I encountered while building the database were different spellings that did not match between tables, and different variations of the same concept in the same table. Since there is a lot of data, I used certain queries to see what types of mis-spellings I had from table to table. For example, to see the differences between the names from two tables I used the following query to look for any irregular rows that came up as null and vice versa:

```
SELECT DISTINCT Champions.ChampionName, CounterChampions.Enemy
FROM Champions RIGHT JOIN CounterChampions
ON Champions.ChampionName=CounterChampions.Enemy
WHERE Champions.ChampionName is NULL
```

I then manually corrected mis-spellings that I found using this query. I verified that everything is correct by adding the FOREIGN KEY constraints.

I also had some issues with the design of a few tables.

For example, I could not use Result and Component as Primary Key in ItemRecipes because one item may be needed several times to build a complex item:



Option 1

Result	Component	Quantity
Frozen Heart	Glacial Shroud	1
Frozen Heart	Cloth Armor	2

Option 2

Result	Component	BuildID
Frozen Heart	Glacial Shroud	1
Frozen Heart	Cloth Armor	2
Frozen Heart	Cloth Armor	3

I picked Option 2 for ItemRecipes because later on it is easier to calculate the sum of the cost of the components. I encountered the same problem for ItemBuilds: one champion may use several identical items.

Another example. In the Synergy table, I had to choose between repeating each relationship twice or once:

Synergy Table

Option 1

Champion1	Champion2
Ahri	Jax
Jax	Ahri

- Updates are tedious
- But queries are simple:

```
Select Champion2
From Synergy
Where Synergy.Champion1='Ahri'
```

Option 2

Champion1	Champion2
Ahri	Jax

- Updates are simpler
- But queries are more tedious:

```
Select Champion2
From Synergy
Where Synergy.Champion1='Ahri'
Union
Select Champion1
From Synergy
Where Synergy.Champion2='Ahri'
```

I chose option 2 to keep updates simpler, but I am still not sure which is the better approach.

Sample queries and results (at least 3 sample queries)

1. Who has synergy with Ashe?

```
SELECT DISTINCT Synergy.Champion1
FROM Synergy
WHERE Synergy.Champion2='Ashe'
UNION
SELECT DISTINCT Synergy.Champion2
FROM Synergy
WHERE Synergy.Champion1='Ashe'
```

Champion1
Janna
Lulu
Lux
Nautilus
Sejuani
Sona
Soraka

2. Find the name and the cost of each item that counters Annie.

```
SELECT Items.Item, Items.TotalCost
FROM Items, CounterItems
WHERE Items.Item=CounterItems.CounterItem AND CounterItems.Enemy='Annie'
```

Item	TotalCost
Banshee's Veil	2715
Hexdrinker	1400

3. Find all champions that can go to the top lane and counter Cho'Gath

```
SELECT CounterChampions.Counterpick
FROM CounterChampions, Laning
WHERE CounterChampions.Enemy='Cho\'Gath' AND CounterChampions.Counterpick=Laning.Champion
AND Laning.Lane='Top'
```

Counterpick
Garen
Irelia
Jayce
Olaf
Renekton
Rumble
Teemo
Warwick

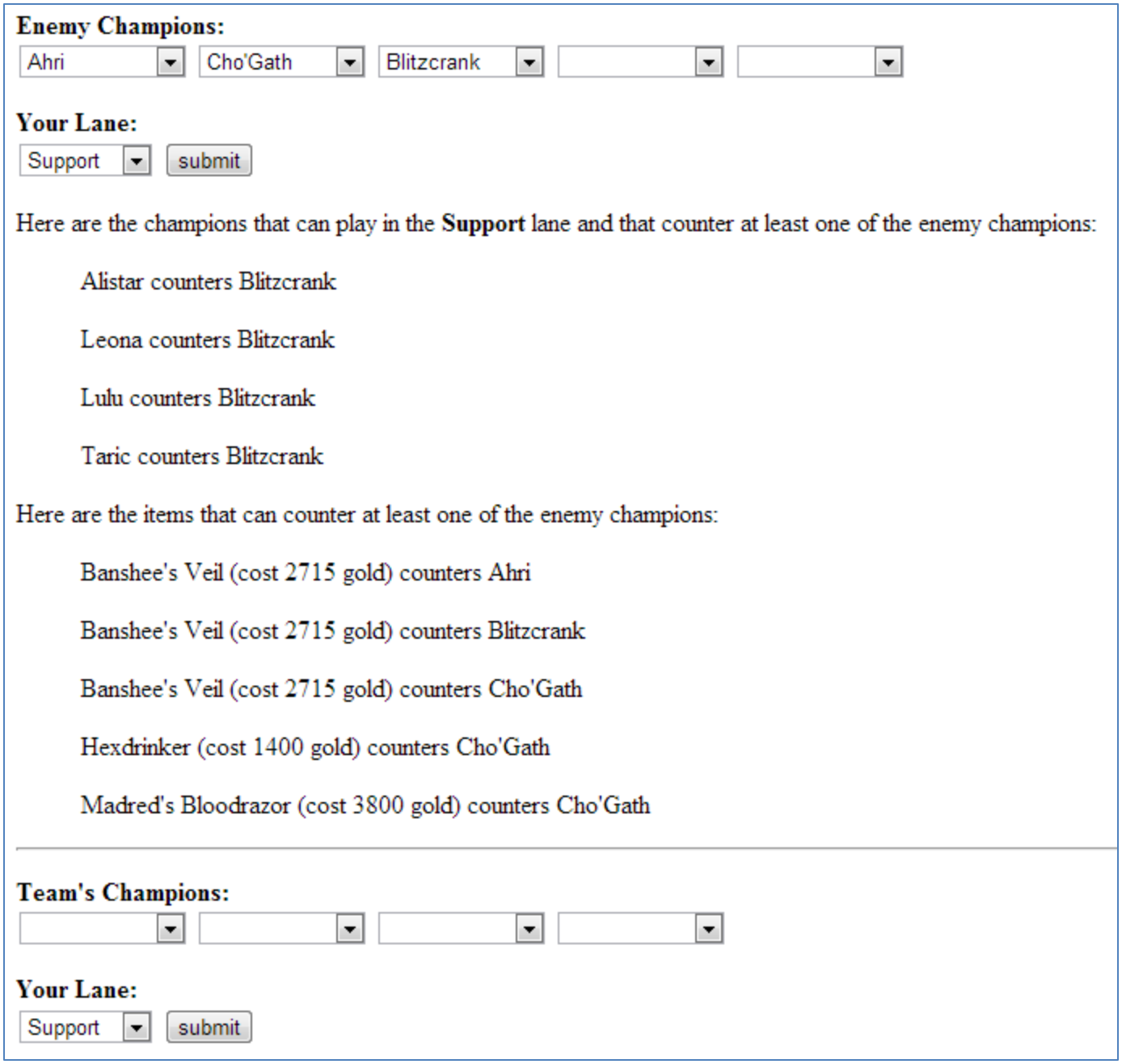
Interactive forms

I put all the forms on one web page at:
http://people.ischool.berkeley.edu/~evphan/lol.php

This is how the page looks like initially:

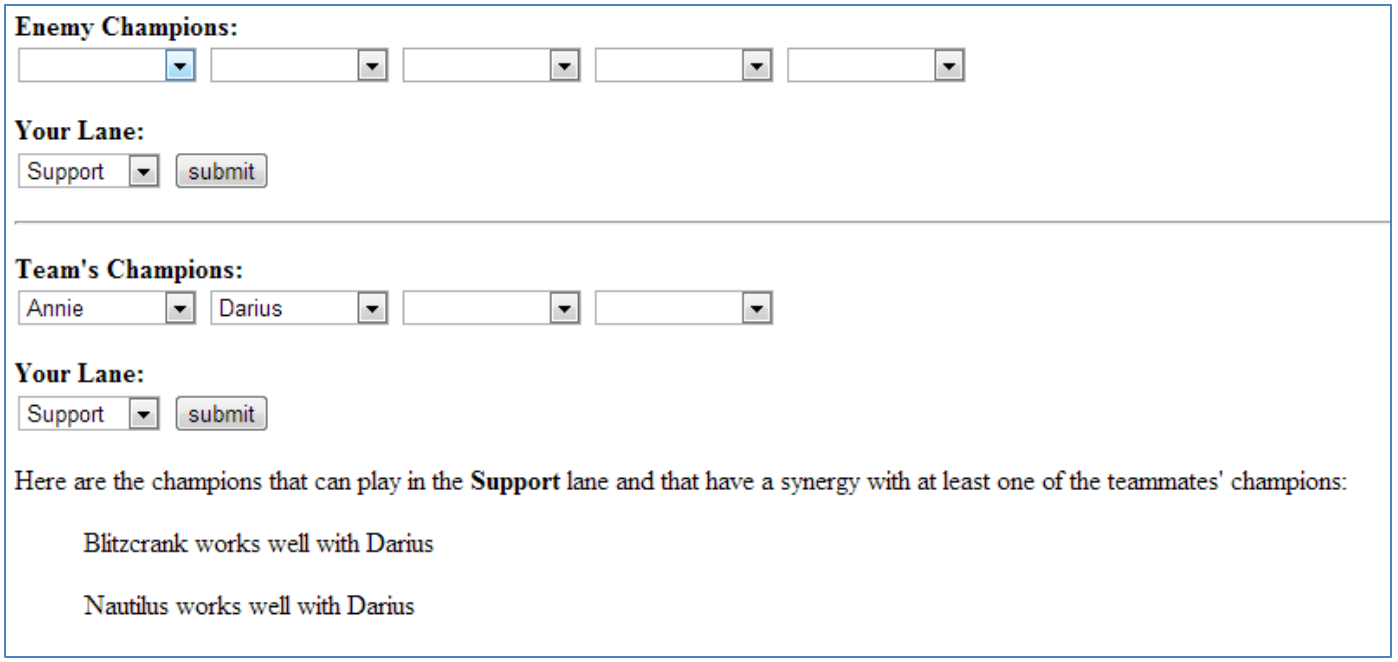
Here, the user can choose to enter either the enemy champions or her own team's champions. In both cases, she will need to choose the lane she's going to play in. If she hasn't decided on the lane yet, and wants to see results unrestricted by lane, she needs to enter "Any Lane".

If the user enters one or more (up to 5) enemy champions, she will see the following screen:



On this screen, she sees some ideas for champions to play and items to buy that counter at least one of the existing enemy selections.

The user can then enter the champions on her own team, and see the following result:



Here, the champion recommendations are provided that work well with the existing teammates' selections.

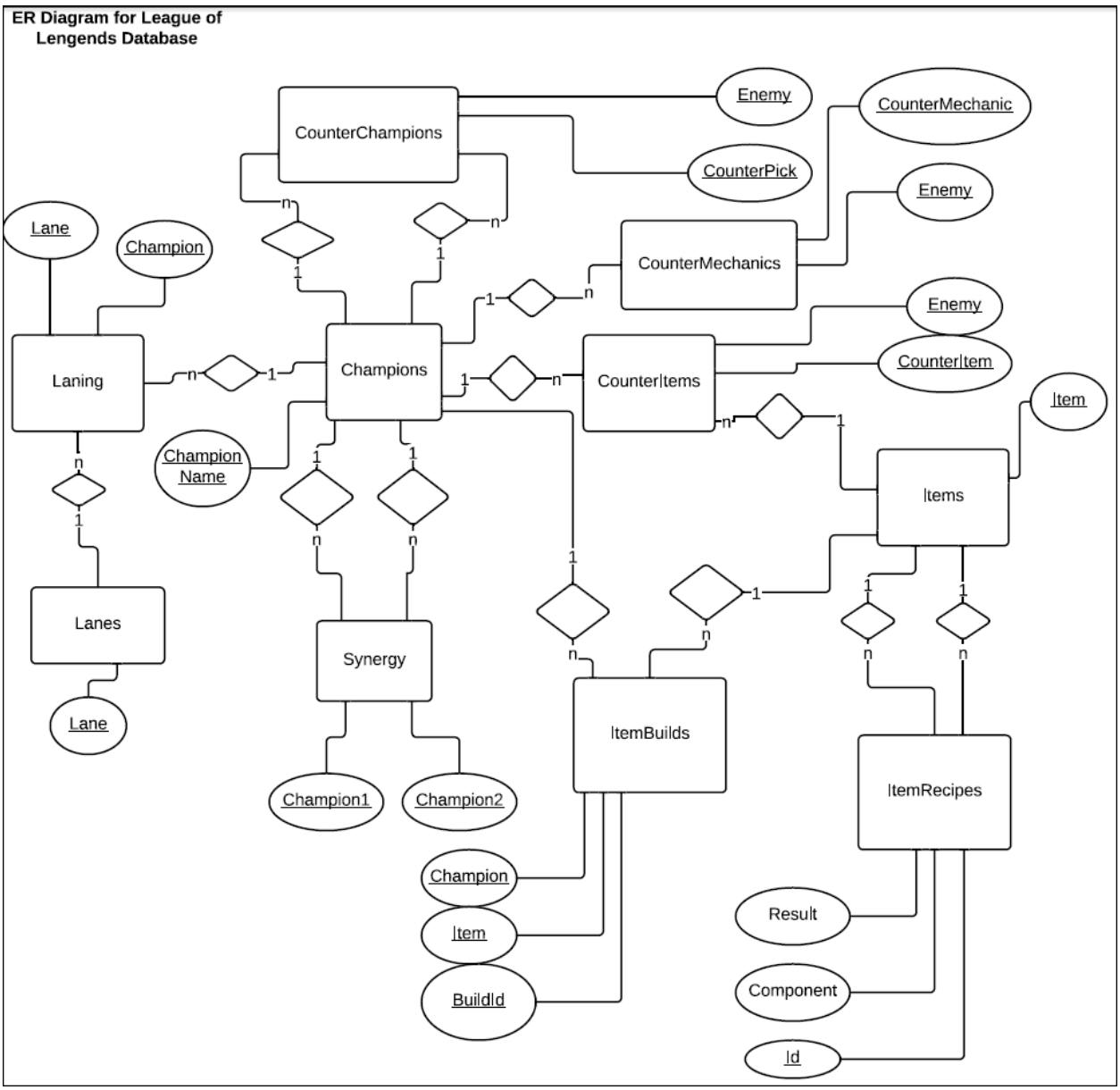
Sample reports

SQL Script

```
\T report1
SELECT COUNT(Champions.ChampionName) as 'Number of Champions:'
FROM Champions
\g
\t

\T report2
SELECT SUM(Items.TotalCost) as 'Cost of All Items:'
FROM Items
\g
\t

\T report3
SELECT COUNT(Items.Item) as 'Number of Items:'
FROM Items
\g
SELECT COUNT(Items.Item) as 'Number of Simple Items:'
FROM Items LEFT JOIN ItemRecipes ON Items.Item = ItemRecipes.Result
WHERE ItemRecipes.Result is NULL
\g
\t

mysql> \. report1
Logging to file 'report1.out'
+----------------------+
| Number of Champions: |
+----------------------+
|                  106 |
+----------------------+
1 row in set (0.00 sec)

+--------------------+
```

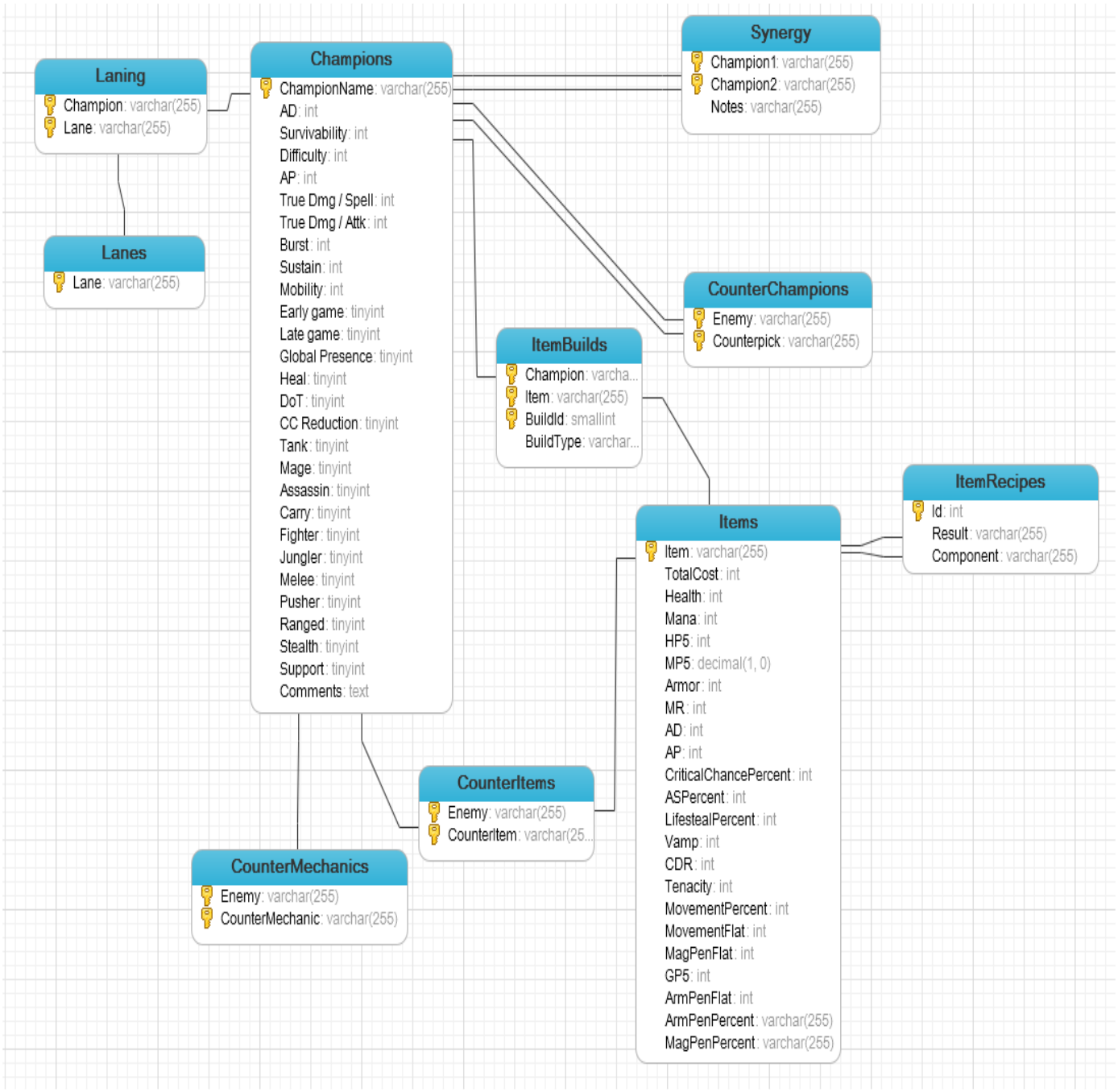
```
mysql> \. report2
Logging to file 'report2.out'
| Cost of All Items: |
+--------------------+
|             175803 |
+--------------------+
1 row in set (0.00 sec)

mysql> \. report3
Logging to file 'report3.out'

+------------------+
| Number of Items: |
+------------------+
|              110 |
+------------------+
1 row in set (0.00 sec)

+-------------------------+
| Number of Simple Items: |
+-------------------------+
|                      34 |
+-------------------------+
1 row in set (0.00 sec)
```

ER Diagram:



The main entities in this diagram are Champions, Items, and Lanes. The other entities represent relationships between them. For example, Synergy represents a relationship between two Champions; each row in that table says that two champions work well together. Another example: ItemBuilds recommends several items to buy for a particular champion. ItemRecipes shows how some items are built from simpler components.

Database Schema:



Data Dictionary

Champions Table:

```
CREATE TABLE `Champions` (
  `ChampionName` varchar(255) NOT NULL,
  `AD` int(11) DEFAULT NULL,
  `Survivability` int(11) DEFAULT NULL,
  `Difficulty` int(11) DEFAULT NULL,
  `AP` int(11) DEFAULT NULL,
  `True Dmg / Spell` int(11) DEFAULT NULL,
  `True Dmg / Attk` int(11) DEFAULT NULL,
  `Burst` int(11) DEFAULT NULL,
  `Sustain` int(11) DEFAULT NULL,
  `Mobility` int(11) DEFAULT NULL,
  `Early game` tinyint(4) DEFAULT NULL,
  `Late game` tinyint(4) DEFAULT NULL,
  `Global Presence` tinyint(4) DEFAULT NULL,
  `Heal` tinyint(4) DEFAULT NULL,
  `DoT` tinyint(4) DEFAULT NULL,
  `CC Reduction` tinyint(4) DEFAULT NULL,
  `Tank` tinyint(4) DEFAULT NULL,
  `Mage` tinyint(4) DEFAULT NULL,
  `Assassin` tinyint(4) DEFAULT NULL,
  `Carry` tinyint(4) DEFAULT NULL,
  `Fighter` tinyint(4) DEFAULT NULL,
  `Jungler` tinyint(4) DEFAULT NULL,
  `Melee` tinyint(4) DEFAULT NULL,
  `Pusher` tinyint(4) DEFAULT NULL,
  `Ranged` tinyint(4) DEFAULT NULL,
  `Stealth` tinyint(4) DEFAULT NULL,
  `Support` tinyint(4) DEFAULT NULL,
  `Comments` text,
  PRIMARY KEY (`ChampionName`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

Laning Table:

```
CREATE TABLE `Laning` (
  `Champion` varchar(255) NOT NULL,
  `Lane` varchar(255) NOT NULL,
  PRIMARY KEY (`Champion`,`Lane`),
  KEY `Lane` (`Lane`),
  CONSTRAINT `Laning_ibfk_1` FOREIGN KEY (`Lane`) REFERENCES `Lanes` (`Lane`),
  CONSTRAINT `Laning_ibfk_2` FOREIGN KEY (`Champion`) REFERENCES `Champions` (`ChampionName`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

Lanes Table:

```
CREATE TABLE `Lanes` (
  `Lane` varchar(255) NOT NULL,
  PRIMARY KEY (`Lane`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

CounterMechanics Table:

```
CREATE TABLE `CounterMechanics` (
  `Enemy` varchar(255) NOT NULL,
  `CounterMechanic` varchar(255) NOT NULL,
  PRIMARY KEY (`Enemy`,`CounterMechanic`),
  CONSTRAINT `CounterMechanics_ibfk_1` FOREIGN KEY (`Enemy`) REFERENCES `Champions` (`ChampionName`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

CounterItems Table:

```
CREATE TABLE `CounterItems` (
  `Enemy` varchar(255) NOT NULL,
  `CounterItem` varchar(255) NOT NULL,
  PRIMARY KEY (`Enemy`,`CounterItem`),
  KEY `CounterItem` (`CounterItem`),
  CONSTRAINT `CounterItems_ibfk_1` FOREIGN KEY (`CounterItem`) REFERENCES `Items` (`Item`),
  CONSTRAINT `CounterItems_ibfk_2` FOREIGN KEY (`Enemy`) REFERENCES `Champions` (`ChampionName`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

Items Table:

```
CREATE TABLE `Items` (
  `Item` varchar(255) NOT NULL,
  `TotalCost` int(11) DEFAULT NULL,
  `Health` int(11) DEFAULT NULL,
  `Mana` int(11) DEFAULT NULL,
  `HP5` int(11) DEFAULT NULL,
  `MP5` decimal(1,0) DEFAULT NULL,
  `Armor` int(11) DEFAULT NULL,
  `MR` int(11) DEFAULT NULL,
  `AD` int(11) DEFAULT NULL,
  `AP` int(11) DEFAULT NULL,
  `CriticalChancePercent` int(11) DEFAULT NULL,
  `ASPercent` int(11) DEFAULT NULL,
  `LifestealPercent` int(11) DEFAULT NULL,
  `Vamp` int(11) DEFAULT NULL,
  `CDR` int(11) DEFAULT NULL,
  `Tenacity` int(11) DEFAULT NULL,
  `MovementPercent` int(11) DEFAULT NULL,
  `MovementFlat` int(11) DEFAULT NULL,
```

```
  `MagPenFlat` int(11) DEFAULT NULL,
  `GP5` int(11) DEFAULT NULL,
  `ArmPenFlat` int(11) DEFAULT NULL,
  `ArmPenPercent` varchar(255) DEFAULT NULL,
  `MagPenPercent` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`Item`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

ItemRecipes Table:

```
CREATE TABLE `ItemRecipes` (
  `Id` int(11) NOT NULL AUTO_INCREMENT,
  `Result` varchar(255) DEFAULT NULL,
  `Component` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`Id`),
  KEY `Result` (`Result`),
  KEY `Component` (`Component`),
  CONSTRAINT `ItemRecipes_ibfk_1` FOREIGN KEY (`Result`) REFERENCES `Items` (`Item`),
  CONSTRAINT `ItemRecipes_ibfk_2` FOREIGN KEY (`Component`) REFERENCES `Items` (`Item`)
) ENGINE=InnoDB AUTO_INCREMENT=158 DEFAULT CHARSET=latin1
```

ItemBuilds Table:

```
CREATE TABLE `ItemBuilds` (
  `Champion` varchar(255) NOT NULL,
  `Item` varchar(255) NOT NULL,
  `BuildId` smallint(6) NOT NULL,
  `BuildType` varchar(255) NOT NULL,
  PRIMARY KEY (`Champion`,`Item`,`BuildId`),
  KEY `Item` (`Item`),
  CONSTRAINT `ItemBuilds_ibfk_2` FOREIGN KEY (`Item`) REFERENCES `Items` (`Item`),
  CONSTRAINT `ItemBuilds_ibfk_1` FOREIGN KEY (`Champion`) REFERENCES `Champions` (`ChampionName`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

CounterChampions Table:

```
CREATE TABLE `CounterChampions` (
  `Enemy` varchar(255) NOT NULL,
  `Counterpick` varchar(255) NOT NULL,
  PRIMARY KEY (`Enemy`,`Counterpick`),
  KEY `Counterpick` (`Counterpick`),
  CONSTRAINT `CounterChampions_ibfk_1` FOREIGN KEY (`Enemy`) REFERENCES `Champions` (`ChampionName`),
  CONSTRAINT `CounterChampions_ibfk_2` FOREIGN KEY (`Counterpick`) REFERENCES `Champions` (`ChampionName`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

Synergy Table:

```
CREATE TABLE `Synergy` (
  `Champion1` varchar(255) NOT NULL,
```

```
  `Champion2` varchar(255) NOT NULL,
  `Notes` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`Champion1`,`Champion2`),
  KEY `Champion2` (`Champion2`),
  CONSTRAINT `Synergy_ibfk_1` FOREIGN KEY (`Champion1`) REFERENCES `Champions` (`ChampionName`),
  CONSTRAINT `Synergy_ibfk_2` FOREIGN KEY (`Champion2`) REFERENCES `Champions` (`ChampionName`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```