SQL Project:

Application Description

I decided to model my database around the playable characters in a game
I used to play called SMITE. These playable characters are known as Gods and there are over 100 of them available to choose from but I'll only analyse a select few of them. Each God is made up of a number of attributes. The main aspects I'll be including as Relationship Tables are: the God pool, the Pantheons, the Classes, the Positions, The Playable Positions and the Ability Types.

God pool:

This is the collection of Gods that a player can choose from when they load into a game. The attributes associated with God pool are: Id, name, pantheon, class and their abilities 1-4

Pantheon:

Pantheon is the religion, mythology or culture the character belongs to. There are 15 pantheons in the game and the attributes a pantheon has include name and region

Classes:

This is the collection of classes a God can be. The attributes associated with a class are: name, damage type and typical attack

Positions:

The main map in SMITE is a 5v5 game mode. On that map, there are 5 positions a God can play in. Each Position has the attributes name, start location, main buff camp and role.

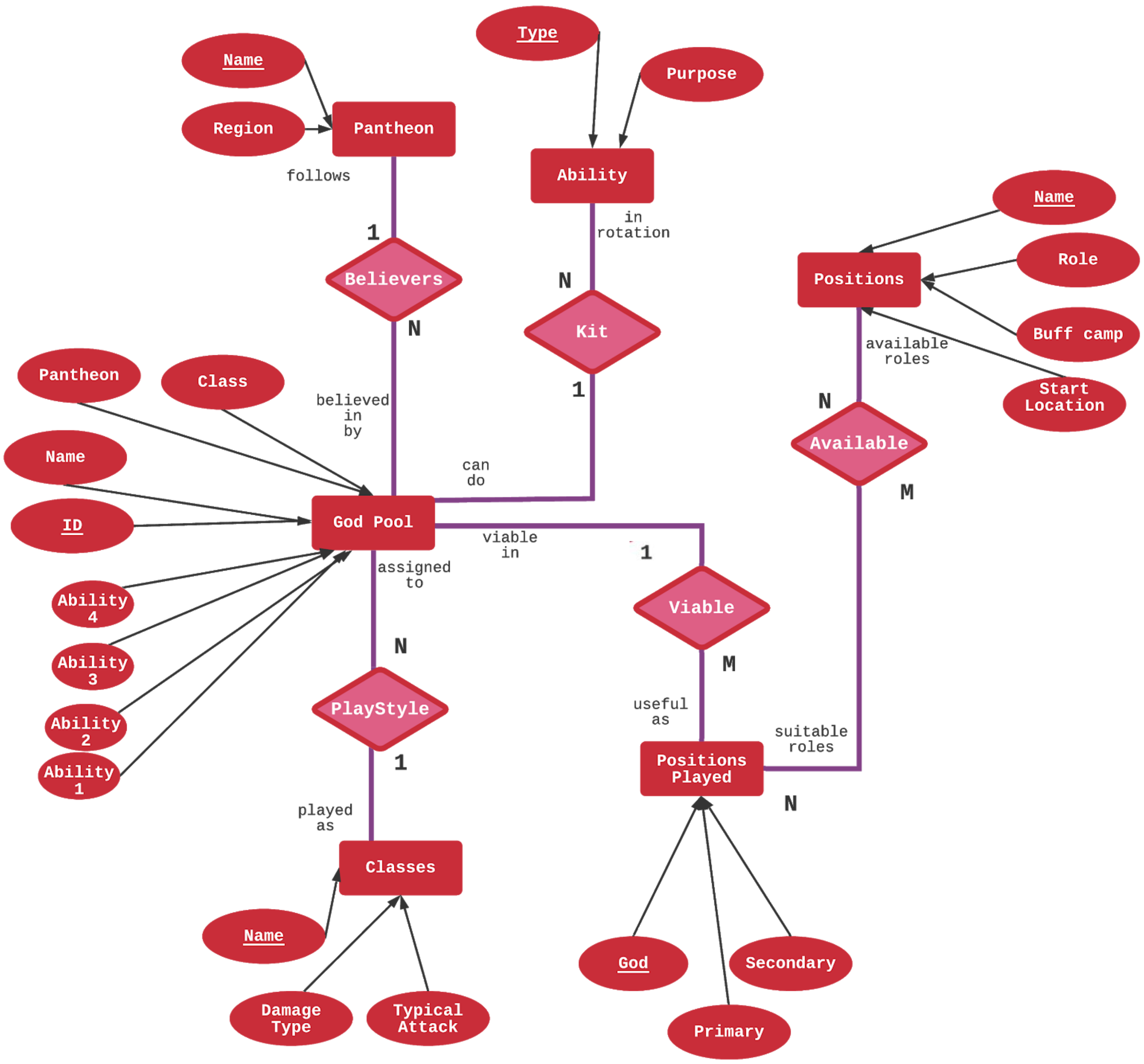
Positions Played:

This is a collection of the viable Positions God can play. Each god has a primary position and may also have a secondary position that they might be viable in.

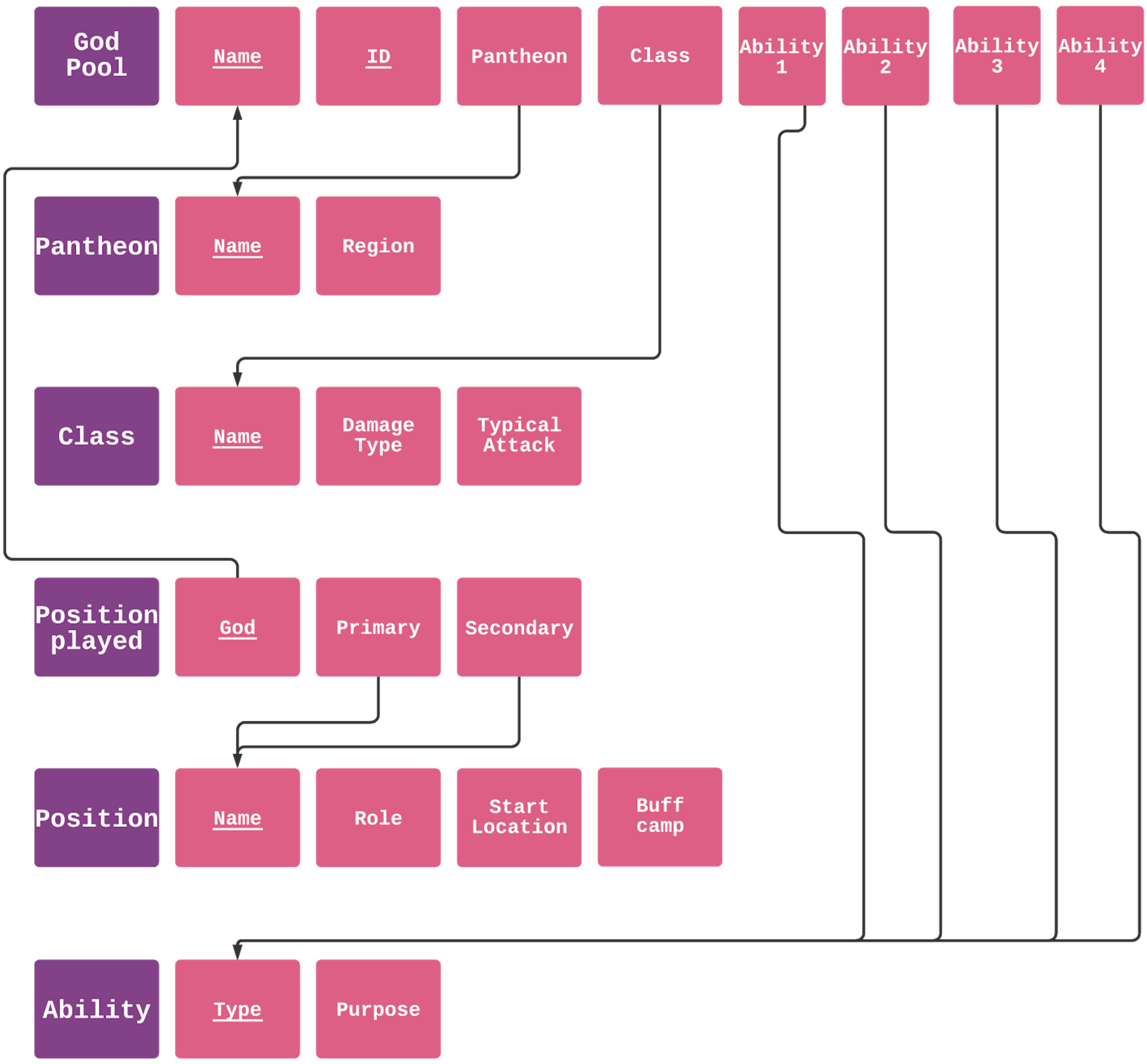
Ability:

Every god has 4 abilities. Every ability is unique but shares common attributes. The attributes an ability has is Type and Purpose

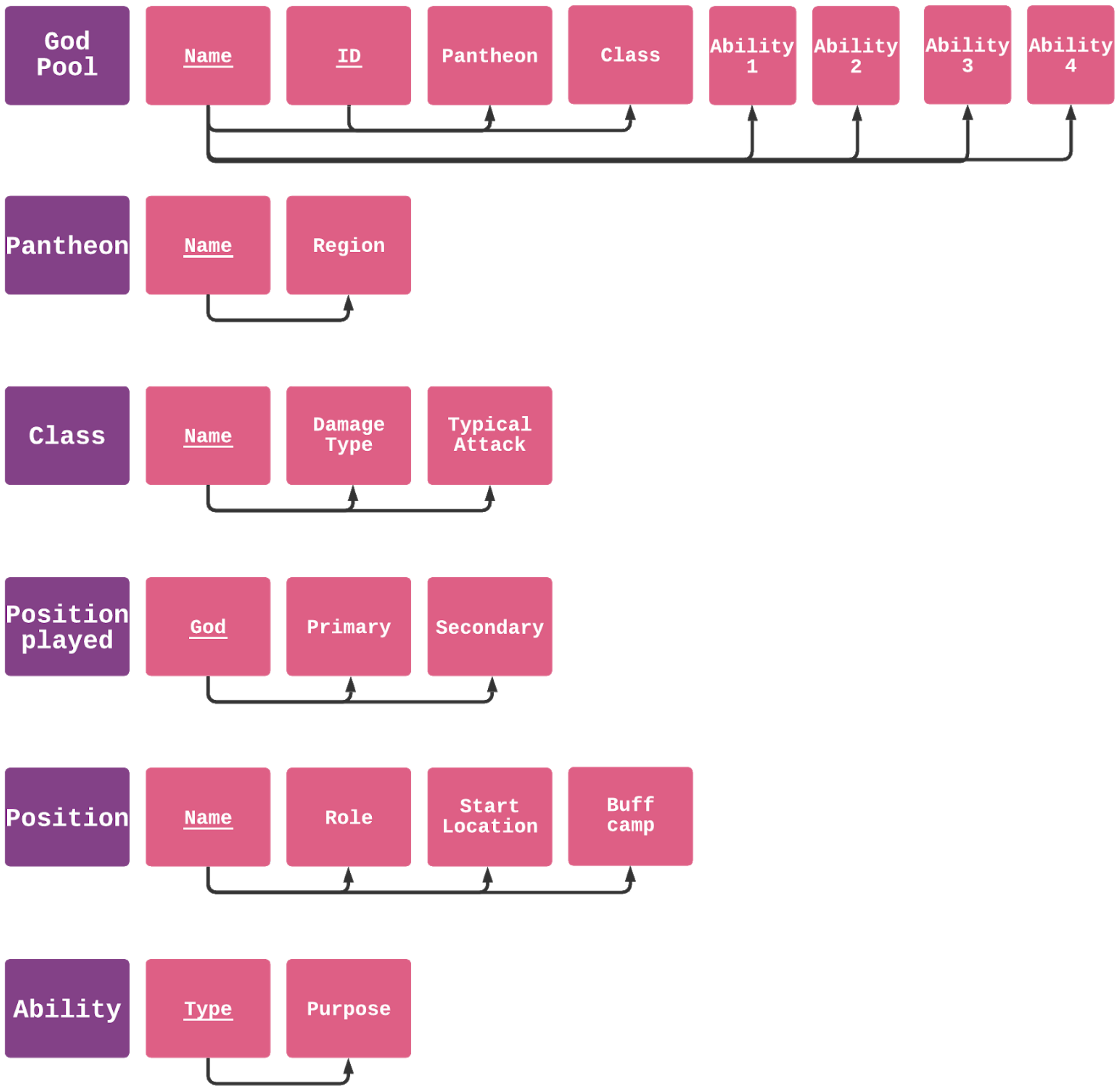
Entity Relationship Diagram



Relational Schema



Functional Dependency Diagram



Constraints

There are a handful of constraints that I have added to my database.

The implicit constraints in my database include:

Primary keys are NOT NULL, Foreign keys are NOT NULL except for the Secondary Column

The first type of Semantic constraint is a type constraint. The ID of a God in God Pool must be less than the Number of Gods in the Game which is 112. The second type of constraints is the ones on what values are allowed in certain attributes. For example,

In God Pool,

- ALL Attributes are NOT NULL

In Class,

- **DamageType** can only be the string values (Physical, Magical) and NOT NULL
- **TypicalAttack** can only be the string values (Ranged, Melee) and NOT NULL

In Positions,

- **Role** can only be the string values (Offence, Hybrid, Defence) and NOT NULL
- **StartLocation** can only be the string values (Duel Lane, Mid Lane, Solo Lane, Jungle) and NOT NULL

In Positions Played,

- **PrimaryPos** is NOT NULL
- **SecondaryPos** can be NULL

In Ability,

- **Purpose** can only be the string values (Damage, Mobility, Support) and NOT NULL

Code For Constraints

*Also in Appendix

```
ALTER TABLE godpool ADD CHECK   ( id <112 );
ALTER TABLE class ADD CHECK     ( DamageType    IN ( 'physical', 'magical' ) );
ALTER TABLE class ADD CHECK     ( TypicalAttack IN ( 'ranged', 'melee') );
ALTER TABLE position ADD CHECK  ( Role          IN ( 'offence', 'melee', 'hybrid') );
ALTER TABLE position ADD CHECK  ( StartLocation IN ( 'duel lane', 'mid lane', 'jungle', 
'solo lane') );
ALTER TABLE ability ADD CHECK   ( Purpose       IN ( 'mobility', 'damage', 'support') );
```

Security

Security in relation to databases is concerned with the deliberate corruption of the database. Access control is essential in making sure that there is no unauthorised access to the data. This is necessary to prevent deliberate corruption of the database by unauthorised personnel or unauthorised access to the data. In my database, I will be restricting access to the database by establishing access control for each user. The benefit of handling security in this way is I can vary the level of security clearance and privileges for the different roles I want to implement.

The roles I will be granting permissions to will be LeadDeveloper, Developer, ContentCreator and Smite. I am assuming The database administrator already has all privileges.

```
CREATE ROLE LeadDeveloper;
CREATE ROLE Developer;
CREATE ROLE ContentCreator;
CREATE ROLE Smite;

GRANT LeadDeveloper TO PonPon;
GRANT ALL PRIVILEGES ON Smite.* TO LeadDeveloper;
GRANT SELECT, INSERT ON Smite.* TO Developer;
GRANT SELECT, INSERT ON Smite.* TO ContentCreator;
GRANT SELECT ON Smite.* TO Smite;
```

Triggers

The only Trigger that I felt was necessary for the database was a trigger that would update the positions played of a new God. When a new God is added to the God Pool, the user base don't know what position id most viable for them so there will be a delay in updating the PositionPlayed Table. To help resolve this, my Trigger GodAdded will automatically update the position played of the new God with a default position depending on its class.

```
DELIMITER $$
CREATE TRIGGER GodAdded
AFTER INSERT
ON godpool FOR EACH ROW
BEGIN
    IF NEW.class = 'Mage' THEN
        INSERT INTO positionplayed
        VALUES(NEW.name, 'Mid', NULL) ;
    ELSEIF NEW.class = 'Hunter' THEN
        INSERT INTO positionplayed(Type, PrimaryPos, SecondaryPos)
        VALUES(NEW.name, 'ADC', NULL) ;
    ELSEIF NEW.class = 'Warrior' THEN
        INSERT INTO positionplayed(Type, PrimaryPos, SecondaryPos)
        VALUES(NEW.name, 'Solo', NULL);
    ELSEIF NEW.class = 'Assassin' THEN
        INSERT INTO positionplayed(Type, PrimaryPos, SecondaryPos)
        VALUES(NEW.name, 'Jungle', NULL) ;
    ELSE
        INSERT INTO positionplayed(Type, PrimaryPos, SecondaryPos)
        VALUES(NEW.name, 'Support', NULL);
    END IF;
END$$
DELIMITER ;
```

Views

An interesting view I decided to add to my database is the ClassPositions view. This view allows me to see what classes are seen in which positions most often. Gods in the game can be played very differently even if they are in the same class. This view lets us see what these different roles are for the different classes.

```sql
CREATE VIEW ClassPositions AS
SELECT g.Class, p.PrimaryPos, p.SecondaryPos
FROM Godpool g, PositionPlayed p
WHERE g.name = p.name
```

Appendix

```sql
CREATE TABLE GodPool (
    Name varchar(255) NOT NULL,
    ID int NOT NULL,
    Pantheon varchar(255) NOT NULL,
    Class varchar(255) NOT NULL,
    Ability1 varchar(255) NOT NULL,
    Ability2 varchar(255) NOT NULL,
    Ability3 varchar(255) NOT NULL,
    Ability4 varchar(255) NOT NULL,
    Primary Key(Name),
    Foreign Key(Pantheon) References Pantheon(Name),
    Foreign Key(Class) References Class(Name),
    Foreign Key(Ability1) References Ability(Type),
    Foreign Key(Ability2) References Ability(Type),
    Foreign Key(Ability3) References Ability(Type),
    Foreign Key(Ability4) References Ability(Type)
);
```

```sql
CREATE TABLE Pantheon (
    Name varchar(255) NOT NULL,
    Region varchar(255) NOT NULL,
    Primary Key(Name)
);

CREATE TABLE Class (
    Name varchar(255) NOT NULL,
    DamageType varchar(255) NOT NULL,
    TypicalAttack varchar(255) NOT NULL,
    Primary Key(Name)
);

CREATE TABLE Position (
    Name varchar(255) NOT NULL,
    Role varchar(255) NOT NULL,
    StartLocation varchar(255) NOT NULL,
    BuffCamp varchar(255),
    Primary Key(Name)
);

CREATE TABLE PositionPlayed (
    Name varchar(255) NOT NULL,
    PrimaryPos varchar(255) NOT NULL,
    SecondaryPos varchar(255),
    Primary Key(Name),
    Foreign Key(PrimaryPos) References smite.Position(Name),
    Foreign Key(SecondaryPos) References smite.Position(Name)
);

CREATE TABLE Ability (
    Type varchar(255) NOT NULL,
    Purpose varchar(255) NOT NULL,
    Primary Key(Type)
);
```

```
ALTER TABLE godpool ADD CHECK   ( id <112 );
ALTER TABLE class ADD CHECK     ( DamageType    IN ( 'Physical', 'Magical' )
);
ALTER TABLE class ADD CHECK     ( TypicalAttack IN ( 'Ranged', 'Melee') );
ALTER TABLE position ADD CHECK  ( Role          IN ( 'Offence', 'Defence',
'Hybrid') );
ALTER TABLE position ADD CHECK  ( StartLocation IN ( 'duel lane', 'mid lane',
'jungle', 'solo lane') );
ALTER TABLE ability ADD CHECK   ( Purpose       IN ( 'mobility', 'damage',
'support') );



INSERT INTO pantheon VALUES('Arthurian','British Isles');
INSERT INTO pantheon VALUES('Celtic','British Isles');
INSERT INTO pantheon VALUES('Chinese','China');
INSERT INTO pantheon VALUES('Egyptian','Egypt');
INSERT INTO pantheon VALUES('Great Old Ones','Fiction');
INSERT INTO pantheon VALUES('Greek','Greece');
INSERT INTO pantheon VALUES('Hindu','Earth');
INSERT INTO pantheon VALUES('Japanese','Japan');
INSERT INTO pantheon VALUES('Mayan','America');
INSERT INTO pantheon VALUES('Norse','Scandanavia');
INSERT INTO pantheon VALUES('Polynesian','Polynesia');
INSERT INTO pantheon VALUES('Roman','Rome');
INSERT INTO pantheon VALUES('Slavic','Europe');
INSERT INTO pantheon VALUES('Voodoo','Earth');
INSERT INTO pantheon VALUES('Yoruba','Africa');


INSERT INTO class VALUES('Hunter','Physical', 'Ranged');
INSERT INTO class VALUES('Guardian','Magical', 'Melee');
INSERT INTO class VALUES('Mage','Magical', 'Ranged');
INSERT INTO class VALUES('Assassin','Physical', 'Melee');
INSERT INTO class VALUES('Warrior','Physical', 'Melee');
```

```sql
INSERT INTO position VALUES('ADC','Offence', 'Duel Lane', 'Purple');
INSERT INTO position VALUES('Support','Defence', 'Duel Lane', NULL);
INSERT INTO position VALUES('Mid','Offence', 'Mid Lane', 'Red');
INSERT INTO position VALUES('Jungle','Offence', 'Jungle', 'Yellow');
INSERT INTO position VALUES('Solo','Hybrid', 'Solo Lane', 'Blue');



INSERT INTO positionPlayed VALUES('Ra','Mid', 'Support');
INSERT INTO positionPlayed VALUES('Xbalanque','ADC', NULL);
INSERT INTO positionPlayed VALUES('Ymir','Support', 'Solo');
INSERT INTO positionPlayed VALUES('Hercules','Solo', 'Support');
INSERT INTO positionPlayed VALUES('Bakasura','Jungle', 'Solo');
INSERT INTO positionPlayed VALUES('Merlin','Mid', 'ADC');



INSERT INTO Ability VALUES('Leap','Mobility');
INSERT INTO Ability VALUES('Dash','Mobility');
INSERT INTO Ability VALUES('Line','Damage');
INSERT INTO Ability VALUES('Cone','Damage');
INSERT INTO Ability VALUES('AOE','Damage');
INSERT INTO Ability VALUES('Heal','Support');
INSERT INTO Ability VALUES('Buff','Support');
INSERT INTO Ability VALUES('Debuff','Support');
INSERT INTO Ability VALUES('Wall','Support');
INSERT INTO Ability VALUES('Stance Switch', 'Support');
INSERT INTO godpool VALUES('Ra',        1, 'Egyptian','Mage',
'Line','AOE','Heal','Line');
INSERT INTO godpool VALUES('Xbalanque', 2, 'Mayan','Hunter',
'Buff','Cone','Leap','Debuff');
INSERT INTO godpool VALUES('Ymir',      3, 'Norse','Guardian',
'Wall','Line','Cone','AOE');
INSERT INTO godpool VALUES('Hercules',  4, 'Roman','Warrior',
'Dash','Line','Heal','Line');
INSERT INTO godpool VALUES('Bakasura',  5, 'Hindu','Assassin',
'Leap','Heal','Buff','Buff');
INSERT INTO godpool VALUES('Merlin',    6, 'Arthurian','Mage',
'Line','AOE','Leap','Stance Switch');



CREATE VIEW ClassPositions AS
SELECT g.Class, p.PrimaryPos, p.SecondaryPos
FROM Godpool g, PositionPlayed p
WHERE g.name = p.name
```

```
DELIMITER $$
CREATE TRIGGER GodAdded
AFTER INSERT
ON godpool FOR EACH ROW
BEGIN
    IF NEW.class = 'Mage' THEN
        INSERT INTO positionplayed
        VALUES(NEW.name, 'Mid', NULL) ;
    ELSEIF NEW.class = 'Hunter' THEN
        INSERT INTO positionplayed(Type, PrimaryPos, SecondaryPos)
        VALUES(NEW.name, 'ADC', NULL) ;
    ELSEIF NEW.class = 'Warrior' THEN
        INSERT INTO positionplayed(Type, PrimaryPos, SecondaryPos)
        VALUES(NEW.name, 'Solo', NULL);
    ELSEIF NEW.class = 'Assassin' THEN
        INSERT INTO positionplayed(Type, PrimaryPos, SecondaryPos)
        VALUES(NEW.name, 'Jungle', NULL) ;
    ELSE
        INSERT INTO positionplayed(Type, PrimaryPos, SecondaryPos)
        VALUES(NEW.name, 'Support', NULL);
    END IF;
END$$
DELIMITER ;


CREATE ROLE LeadDeveloper;
CREATE ROLE Developer;
CREATE ROLE ContentCreator;
CREATE ROLE Smite;

GRANT LeadDeveloper TO PonPon;
GRANT ALL PRIVILEGES ON Smite.* TO LeadDeveloper;
GRANT SELECT,INSERT ON Smite.* TO Developer;
GRANT SELECT,INSERT ON Smite.* TO ContentCreator;
GRANT SELECT ON Smite.* TO Smite;

-- //// Fixes to code

-- alter table position drop constraint position_chk_1 ;
-- ALTER TABLE position ADD CHECK   ( Role IN ( 'Offence', 'Defence', 'Hybrid'));
- -DELETE FROM Ability WHERE type = 'leap';
-- ////
```