DUNGEONS & DRAGONS

Character Guide

By: Kevin Scharr

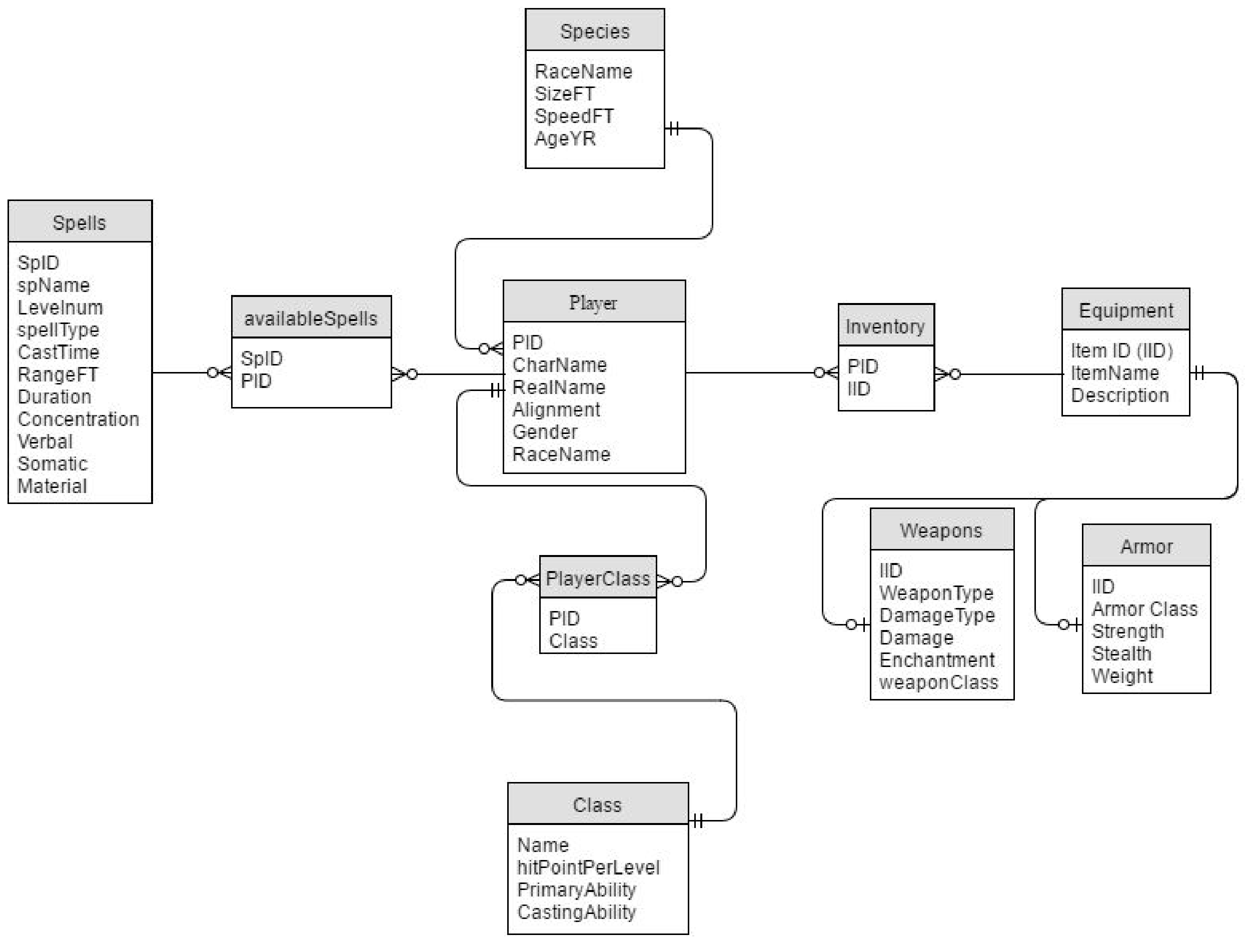
Table Of Contents

Executive Summary:

This paper intends to define the usefulness of a database designed around creating and maintaining a character for a dungeons and dragons campaign in the fifth edition (D&D 5e). This will detail information involved in every class. Players interested in playing D&D 5e will find this database useful to update and finding information as they go. This database will include information that is relevant at any time during the game, information more specific to each character will still be up to the player to maintain.

In this paper we will go into detail on each table involved in this database, including their SQL code. Included will also be a demonstration of the possible uses this database has in form of functions and queries.

Entity Relationship Diagram



<u>Tables:</u>

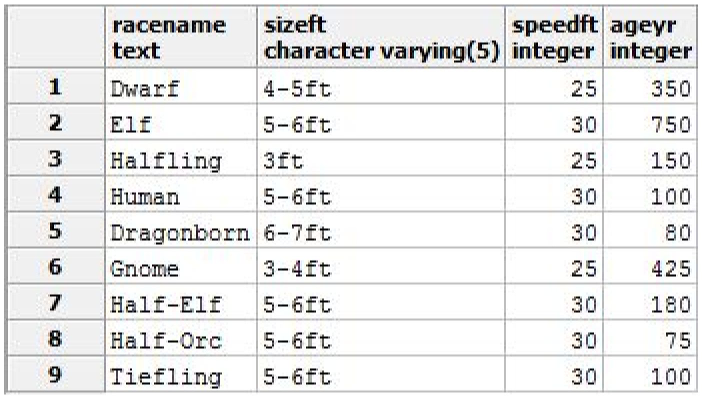
Race Table

--Race--

```
create table Race (
  RaceName      text,
  SizeFT      varchar(5),
  SpeedFT     integer,
  AgeYR       integer,
primary key(RaceName)
);
```

The race table lists all races that a player can be when creating their character, as well as details specific to that race such as average height and player speed.

Functional Dependencies: sizeft, speedft and ageyr are all functionally dependant on Racename



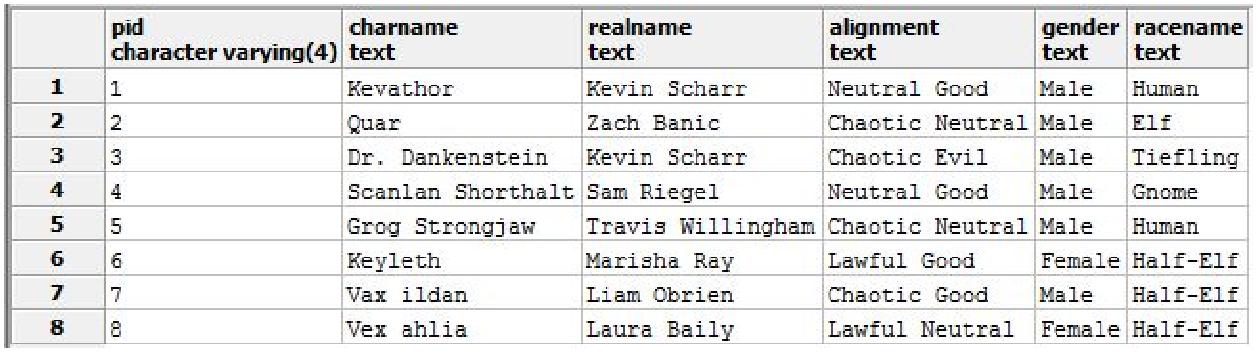
	racename text	sizeft character varying(5)	speedft integer	ageyr integer
1	Dwarf	4-5ft	25	350
2	Elf	5-6ft	30	750
3	Halfling	3ft	25	150
4	Human	5-6ft	30	100
5	Dragonborn	6-7ft	30	80
6	Gnome	3-4ft	25	425
7	Half-Elf	5-6ft	30	180
8	Half-Orc	5-6ft	30	75
9	Tiefling	5-6ft	30	100

Player

```
-- Player --
CREATE TABLE Player (
  PID        Varchar(4) not null,
  CharName   text,
  RealName   Text,
  Alignment  text,
  Gender     Text,
  RaceName   Text not null references Race(RaceName),
  primary key(PID)
);
```

The player table lists all players the user or users wish to keep track of. This table records their character name as well as the name of the person playing that character. For this table i didn't use the player name as the primary key because i'm open to the possibility that characters can have the same name, for whatever the reason.

Functional Dependencies: CharName, RealName, Alignment, Gender and RaceName are all dependent on PID.



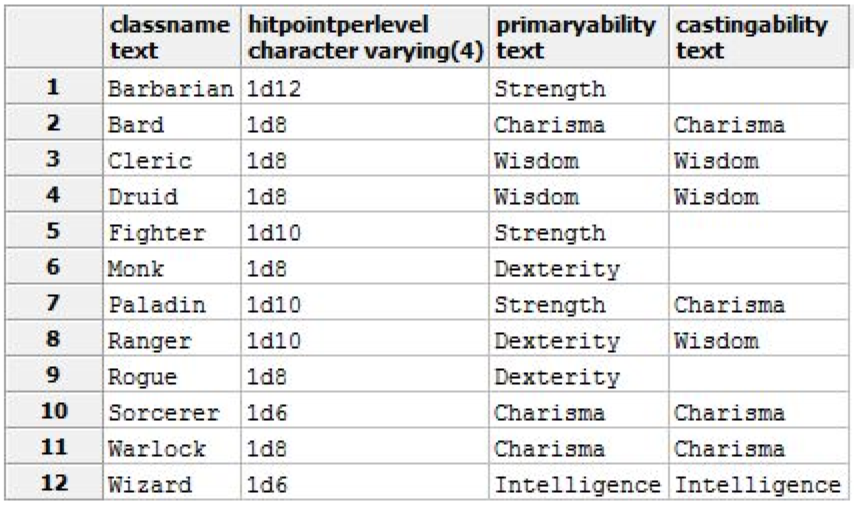
	pid character varying(4)	charname text	realname text	alignment text	gender text	racename text
1	1	Kevathor	Kevin Scharr	Neutral Good	Male	Human
2	2	Quar	Zach Banic	Chaotic Neutral	Male	Elf
3	3	Dr. Dankenstein	Kevin Scharr	Chaotic Evil	Male	Tiefling
4	4	Scanlan Shorthalt	Sam Riegel	Neutral Good	Male	Gnome
5	5	Grog Strongjaw	Travis Willingham	Chaotic Neutral	Male	Human
6	6	Keyleth	Marisha Ray	Lawful Good	Female	Half-Elf
7	7	Vax ildan	Liam Obrien	Chaotic Good	Male	Half-Elf
8	8	Vex ahlia	Laura Baily	Lawful Neutral	Female	Half-Elf

Classes

```
-- Classes --
CREATE TABLE Classes(
  ClassName        Text not null,
  hitPointPerLevel  varchar(4) not null,
  PrimaryAbility    Text not null,
  CastingAbility    Text,
  primary key(ClassName)
);
```

The classes table lists all classes a player can choose at character creation, as well as hitpoints per level, which is what dice a player rolls to see how much health they gain per level. Primary ability, which is the ability score a player should maximize for that class, and casting ability. If a player's class can cast spells, the casting ability is the ability score that class uses to deterime a wide range of factors.

Functional dependencies: hitPointPerLevel, PrimaryAbility, CastingAbility are all functionally dependant on ClassName.



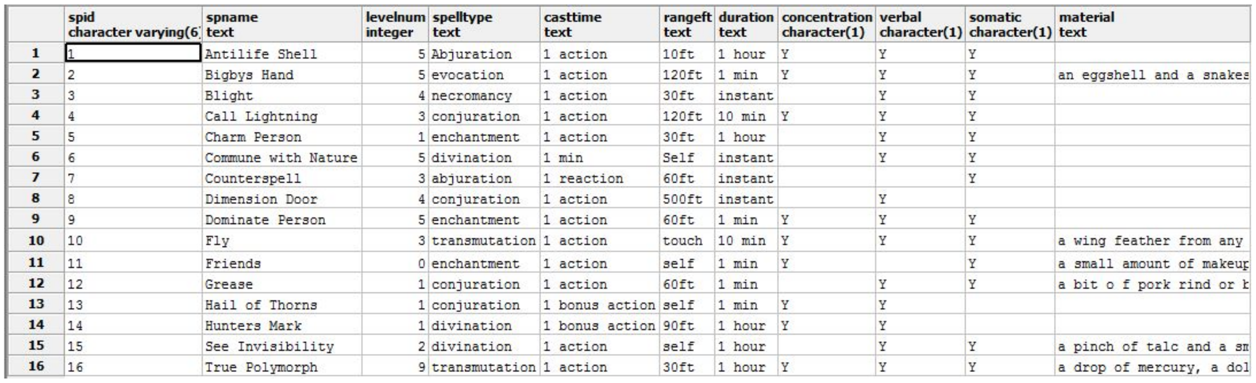
	classname text	hitpointperlevel character varying(4)	primaryability text	castingability text
1	Barbarian	1d12	Strength	
2	Bard	1d8	Charisma	Charisma
3	Cleric	1d8	Wisdom	Wisdom
4	Druid	1d8	Wisdom	Wisdom
5	Fighter	1d10	Strength	
6	Monk	1d8	Dexterity	
7	Paladin	1d10	Strength	Charisma
8	Ranger	1d10	Dexterity	Wisdom
9	Rogue	1d8	Dexterity	
10	Sorcerer	1d6	Charisma	Charisma
11	Warlock	1d8	Charisma	Charisma
12	Wizard	1d6	Intelligence	Intelligence

Spells

```
-- Spells --
CREATE TABLE Spells(
  SpID         varchar(6),
  spName        Text,
  Levelnum      integer,
  spellType     text,
  CastTime      text,
  RangeFT       text,
  Duration      text,
  Concentration Char(1),
  Verbal        char(1),
  Somatic       Char(1),
  Material      text,
  primary key(SpID)
);
```

The spells table lists a sample number of spells. This table will be the most useful table in terms of in game use. A player with the abilityto casts spells can swap out their prepared spells every short rest, which makes a method of quickly searching spells via a number of details is very useful. This includes a number of boolean values, listing "Y" is a spell requires one of the component columns. While no spells share a name, I thought it best to still use a designated number as the primary key, for user discernment considering some spells have similar names.

Functional Dependencies: spName, Levelnum, spellType, CastTime, RangeFT, duration, concentration, verbal, somatic and material are all dependant on the SpID.



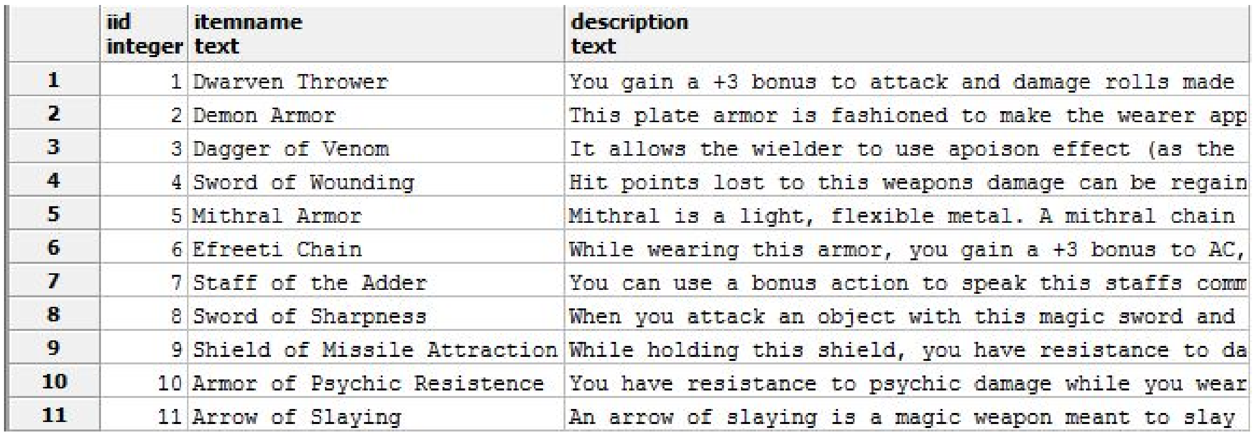
	spid character varying(6	spname text	levelnum integer	spelltype text	casttime text	rangeft text	duration text	concentration character(1)	verbal character(1)	somatic character(1)	material text
1	1	Antilife Shell	5	Abjuration	1 action	10ft	1 hour	Y	Y	Y	
2	2	Bigbys Hand	5	evocation	1 action	120ft	1 min	Y	Y	Y	an eggshell and a snakes
3	3	Blight	4	necromancy	1 action	30ft	instant		Y	Y	
4	4	Call Lightning	3	conjuration	1 action	120ft	10 min	Y	Y	Y	
5	5	Charm Person	1	enchantment	1 action	30ft	1 hour		Y	Y	
6	6	Commune with Nature	5	divination	1 min	Self	instant		Y	Y	
7	7	Counterspell	3	abjuration	1 reaction	60ft	instant			Y	
8	8	Dimension Door	4	conjuration	1 action	500ft	instant		Y		
9	9	Dominate Person	5	enchantment	1 action	60ft	1 min	Y	Y	Y	
10	10	Fly	3	transmutation	1 action	touch	10 min	Y	Y	Y	a wing feather from any
11	11	Friends	0	enchantment	1 action	self	1 min	Y		Y	a small amount of makeup
12	12	Grease	1	conjuration	1 action	60ft	1 min		Y	Y	a bit o f pork rind or b
13	13	Hail of Thorns	1	conjuration	1 bonus action	self	1 min	Y	Y		
14	14	Hunters Mark	1	divination	1 bonus action	90ft	1 hour	Y	Y		
15	15	See Invisibility	2	divination	1 action	self	1 hour		Y	Y	a pinch of talc and a sm
16	16	True Polymorph	9	transmutation	1 action	30ft	1 hour	Y	Y	Y	a drop of mercury, a dol

Equipment

```
-- Equipment --
CREATE TABLE Equipment(
    IID          Integer,
    ItemName       text,
    Description   text,
  primary key(II D)
);
```

The equipment table is the list of all armor and weapons available to the players. Since the details of armor and weapons are very different they will be represented on a different table. Once again, despite the fact that no 2 items share a name, i decided to use IID to make sure that items woth similar names would still be unique.

Functional Dependencies: itemname and description are both functionally dependant on item name.

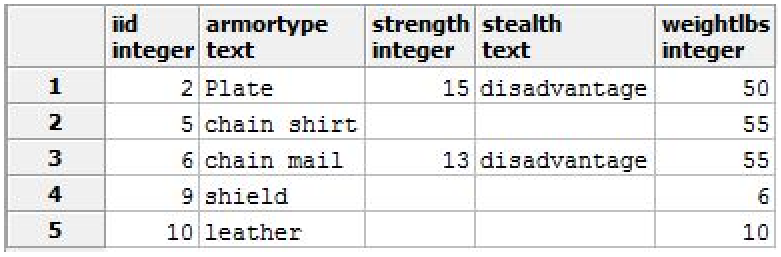


	iid integer	itemname text	description text
1	1	Dwarven Thrower	You gain a +3 bonus to attack and damage rolls made
2	2	Demon Armor	This plate armor is fashioned to make the wearer app
3	3	Dagger of Venom	It allows the wielder to use apoison effect (as the
4	4	Sword of Wounding	Hit points lost to this weapons damage can be regain
5	5	Mithral Armor	Mithral is a light, flexible metal. A mithral chain
6	6	Efreeti Chain	While wearing this armor, you gain a +3 bonus to AC,
7	7	Staff of the Adder	You can use a bonus action to speak this staffs comm
8	8	Sword of Sharpness	When you attack an object with this magic sword and
9	9	Shield of Missile Attraction	While holding this shield, you have resistance to da
10	10	Armor of Psychic Resistence	You have resistance to psychic damage while you wear
11	11	Arrow of Slaying	An arrow of slaying is a magic weapon meant to slay

Armor

```
-- Armor --
CREATE TABLE Armor(
    IID            Integer not null references equipment(IID),
    ArmorType      text not null,
    Strength       integer,
    Stealth        text,
    weightlbs      integer,
  primary key(IID)
);
```
Displays all armor on the equipment table.

Functional Dependencies: ArmorType, Strength, Stealth and weightlbs are all functionally dependant on IID



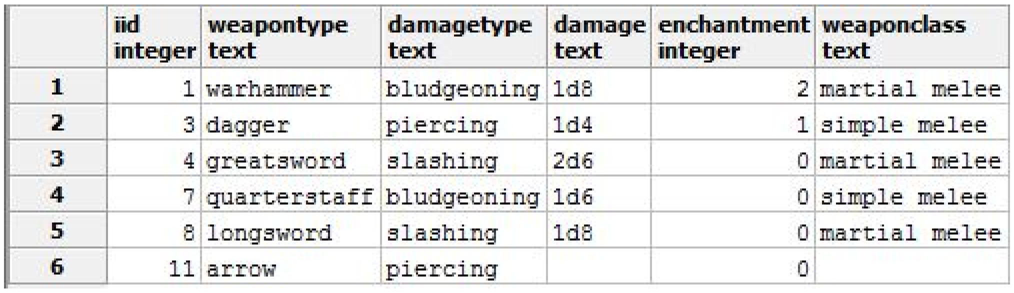
	iid integer	armortype text	strength integer	stealth text	weightlbs integer
1	2	Plate	15	disadvantage	50
2	5	chain shirt			55
3	6	chain mail	13	disadvantage	55
4	9	shield			6
5	10	leather			10

Weapons

```
-- Weapons --
CREATE TABLE weapons(
    IID          Integer not null references equipment(IID),
    weaponType     text,
    damageType     text,
    damage         text,
    enchantment    integer,
    weaponClass    text,
  primary key(IID)
);
```

Display all weapons on the equipment table.

Functional Dependencies: weaponType damageType, damage, enchantment, weaponClass are all functionally dependant on IID.

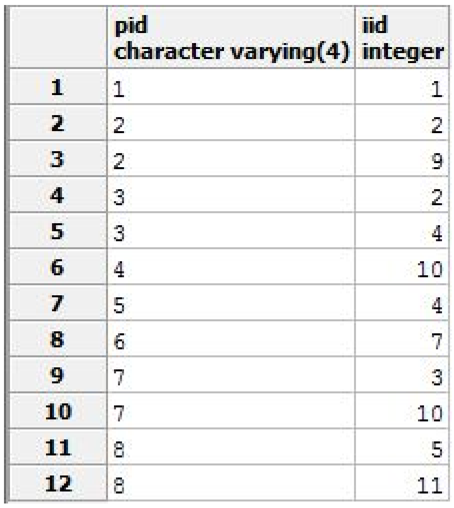


	iid integer	weapontype text	damagetype text	damage text	enchantment integer	weaponclass text
1	1	warhammer	bludgeoning	1d8	2	martial melee
2	3	dagger	piercing	1d4	1	simple melee
3	4	greatsword	slashing	2d6	0	martial melee
4	7	quarterstaff	bludgeoning	1d6	0	simple melee
5	8	longsword	slashing	1d8	0	martial melee
6	11	arrow	piercing		0	

Inventory

```
-- Inventory --
CREATE TABLE inventory(
    PID    varchar(4) not null references player(PID),
  IID    Integer not null references equipment(IID),
 primary key(IID, PID)
);
```

Inventory is the relation table between equipment and players. It shows Which players have which items since many players can have many items.

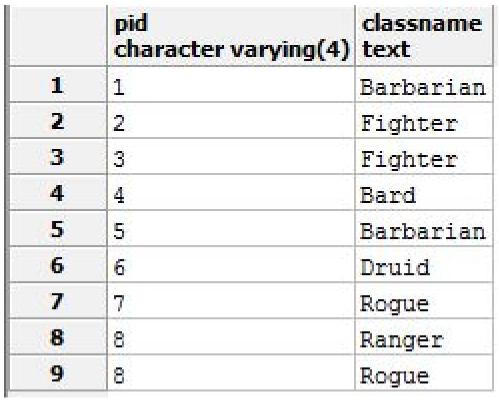


	pid character varying(4)	iid integer
1	1	1
2	2	2
3	2	9
4	3	2
5	3	4
6	4	10
7	5	4
8	6	7
9	7	3
10	7	10
11	8	5
12	8	11

PlayerClass

```
-- PlayerClass --
CREATE TABLE PlayerClass(
  PID    varchar(4) not null references player(PID),
  ClassName    text not null references classes(ClassName),
 primary key(ClassName, PID)
);
```

This tables shows the relation of players to classes. Since Players can multiclass, and have multiple classes.

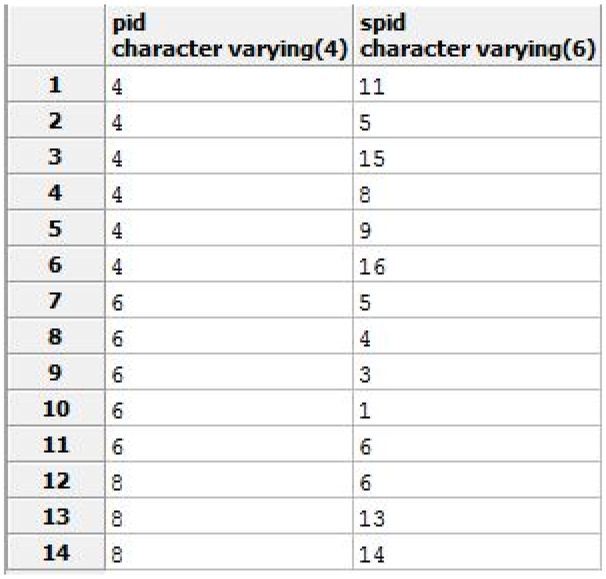


	pid character varying(4)	classname text
1	1	Barbarian
2	2	Fighter
3	3	Fighter
4	4	Bard
5	5	Barbarian
6	6	Druid
7	7	Rogue
8	8	Ranger
9	8	Rogue

Available Spells

```
-- AvailableSpells --
CREATE TABLE availableSpells(
  PID    varchar(4) not null references player(PID),
    SpID    varchar(6) not null references Spells(SpID),
  primary key(SpID, PID)
);
```

This table shows the relation of players and spells. A player that can cast spells has a wide variety at his disposal. This table shows which spells are available to which players, via their ID numbers.



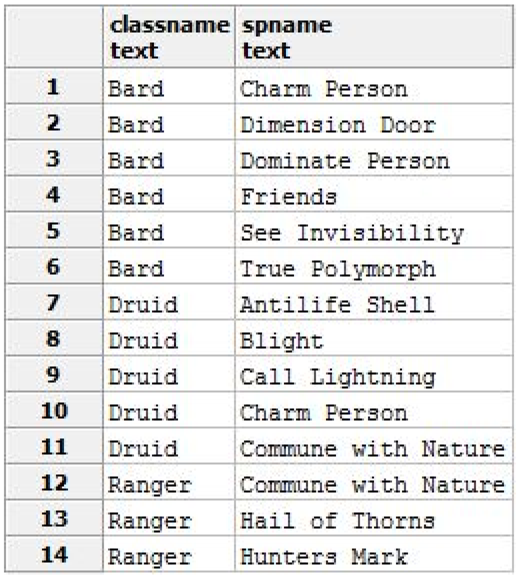
	pid character varying(4)	spid character varying(6)
1	4	11
2	4	5
3	4	15
4	4	8
5	4	9
6	4	16
7	6	5
8	6	4
9	6	3
10	6	1
11	6	6
12	8	6
13	8	13
14	8	14

Views

```
create view ClassSpellList
as
```

```
select distinct classes.classname, spname
from spells, player, availablespells, playerclass, classes
where spells.spid = availableSpells.spid
and player.pid = availablespells.pid
and player.pid = playerclass.pid
and playerclass.classname = classes.classname
and classes.castingability is not null
order by classes.classname asc
```

This table will provide a specific spell list for each class. The Spells table lists all spells without regard to class. The available spells list, lists the spells a player has prepared. This view, will show all spells listed by class, depicting the full spells list for each class.



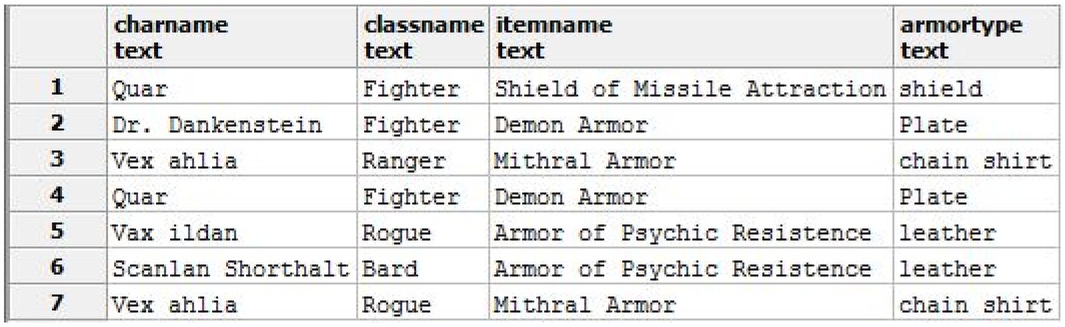
	classname text	spname text
1	Bard	Charm Person
2	Bard	Dimension Door
3	Bard	Dominate Person
4	Bard	Friends
5	Bard	See Invisibility
6	Bard	True Polymorph
7	Druid	Antilife Shell
8	Druid	Blight
9	Druid	Call Lightning
10	Druid	Charm Person
11	Druid	Commune with Nature
12	Ranger	Commune with Nature
13	Ranger	Hail of Thorns
14	Ranger	Hunters Mark

Armor proficiencies view

```
create view armorprof
as
```

```
select distinct charname, classes.classname, itemname, armortype
from player, playerclass, classes, inventory, equipment
inner join armor
on equipment.iid = armor.iid
where classes.classname = playerclass.classname
and playerclass.pid = player.pid
and player.pid = inventory.pid
and inventory.iid = equipment.iid
```

This table would be primarily used to see what each character is wearing.



	charname text	classname text	itemname text	armortype text
1	Quar	Fighter	Shield of Missile Attraction	shield
2	Dr. Dankenstein	Fighter	Demon Armor	Plate
3	Vex ahlia	Ranger	Mithral Armor	chain shirt
4	Quar	Fighter	Demon Armor	Plate
5	Vax ildan	Rogue	Armor of Psychic Resistence	leather
6	Scanlan Shorthalt	Bard	Armor of Psychic Resistence	leather
7	Vex ahlia	Rogue	Mithral Armor	chain shirt

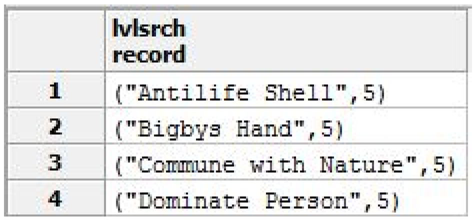
Function

```
create or replace function lvlsrch(splvl integer)
returns table(speName text, lvlNum integer)
```

```
as
$$
begin
return query select spName, Levelnum
from spells
where levelnum = splvl;
end;
$$ language plpgsql;
```

This is the function i was most interested in, it will search spells by specifically, their level number, allowing players to prepare their daily spells more easily, considering they are limited in their spells by number, but only have an alphabetical list of spells.

```
select lvlsrch(5);
```



	lvlsrch record
1	("Antilife Shell",5)
2	("Bigbys Hand",5)
3	("Commune with Nature",5)
4	("Dominate Person",5)

Security:

There would be two levels of security for this database,

- **Admin** would be allowed to insert new content such as player information, and perhaps home brewed spells, equipment and classes.
- **User** would be allowed to make only searches, granted only select, and would query the database for in game uses, making no changes.

<u>Implementation/Known Problems</u>

The database was able to be implemented fairly easily. I think the best use this database has is with the spell list. This is a factor that users will constantly be looking up for rulings and helpful hints. I think the main problem with this database is details. DnD is a very in depth game and to cover all its complexity I would need much more time and many more tables to ensure that this database was maximized in its efficiency. This would also allow me to provide a great many more functions, because i feel the need or availability for functions at this point is very limited. However i do believe that this database is functional and works well to show its uses.

<u>Future Enhancements</u>

To improve this database I think I would rearrange the tables to focus on the spells table. As I said this database will be used primarily for its data on spells. I would create the class, player and race tables, as well as a new attributes table to all help makes the spells table more useful and simple in its searches. This would simplify the function I created to list spells by level, and would also provide a more readily available table for spells by class.

Besides that I think the equipment table needs a lot of work. I’m not completely satisfied with the final version, and I believe is could be made more useful and creative, but i cannot decipher how.