STAR WARS

# Database Design Project

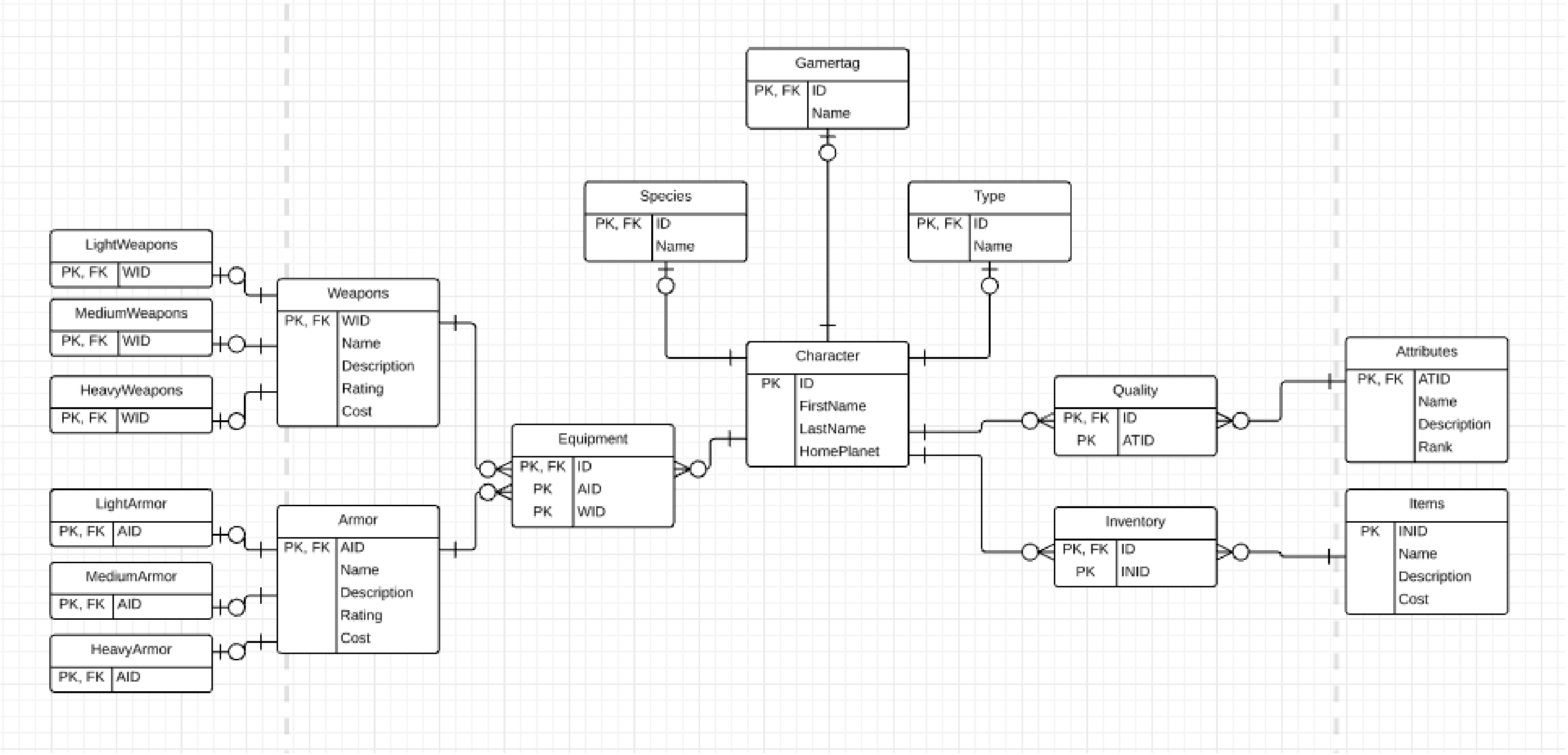
By: Matthew Rahtelli

# Executive Summary

This database design is intended for the relationships surrounding a Star Wars character based video game. This design will include all of the items, armor, and weapons that can be found in the game. The user is able to see what types of character that they can play, as well as what species they can be. This database also shows the relationships between the character and the items, armor, and weapons that they will find.

This design will go into detail about the relationships each table has in the database. It will also include future expectations and already known problems that exist with the design.

# <u>Entity Relationship Diagram</u>



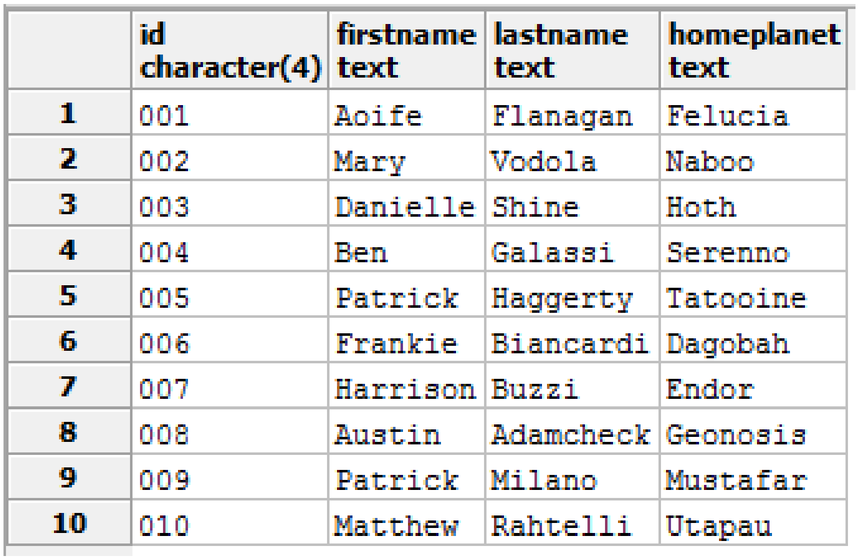
# Tables

Character Table:

```
CREATE TABLE Character (
  ID      char(4) not null,
  FirstName TEXT,
  LastName TEXT,
  HomePlanet TEXT,
 primary key(ID)
);
```

This table lists all of the characters that have been created or will store any of the characters that will be created in the future. This will list their ID, name, and planet of origin.

Function Dependencies: ID → FirstName, LastName, HomePlanet



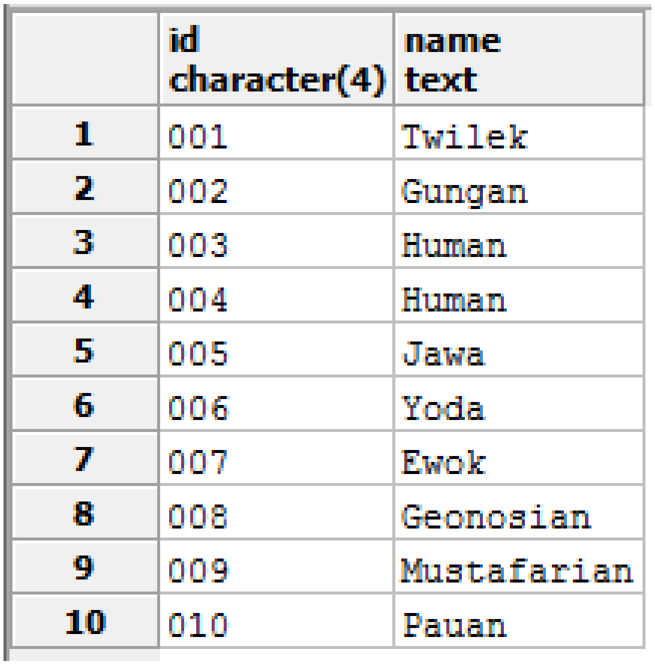
|  | id<br>character(4) | firstname<br>text | lastname<br>text | homeplanet<br>text |
|---|---|---|---|---|
| 1 | 001 | Aoife | Flanagan | Felucia |
| 2 | 002 | Mary | Vodola | Naboo |
| 3 | 003 | Danielle | Shine | Hoth |
| 4 | 004 | Ben | Galassi | Serenno |
| 5 | 005 | Patrick | Haggerty | Tatooine |
| 6 | 006 | Frankie | Biancardi | Dagobah |
| 7 | 007 | Harrison | Buzzi | Endor |
| 8 | 008 | Austin | Adamcheck | Geonosis |
| 9 | 009 | Patrick | Milano | Mustafar |
| 10 | 010 | Matthew | Rahtelli | Utapau |

Species Table:

```
CREATE TABLE Species (
  ID      char(4) not null REFERENCES Character(ID),
  Name    TEXT,
 primary key(ID)
);
```

This table lists all of the species that a character can be in the video game. If any more species are created or available to be chosen, it will be listed in this table.

Functional Dependencies: ID → Name



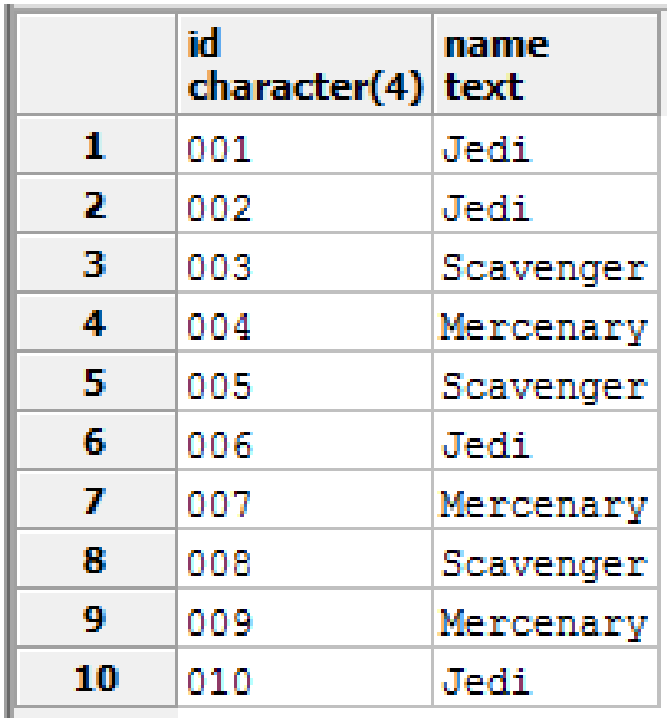
| | id<br>character(4) | name<br>text |
|---|---|---|
| 1 | 001 | Twilek |
| 2 | 002 | Gungan |
| 3 | 003 | Human |
| 4 | 004 | Human |
| 5 | 005 | Jawa |
| 6 | 006 | Yoda |
| 7 | 007 | Ewok |
| 8 | 008 | Geonosian |
| 9 | 009 | Mustafarian |
| 10 | 010 | Pauan |

## Type Table

```
CREATE TABLE Type (
  ID      char(4) not null REFERENCeS Character(ID),
  Name    TEXT,
 primary key(ID)
);
```

This table lists all of the types that the player can be during the game. This will affect in the future what types of items that they can wield, or developing different types of abilities.

Functional Dependencies: ID → Name



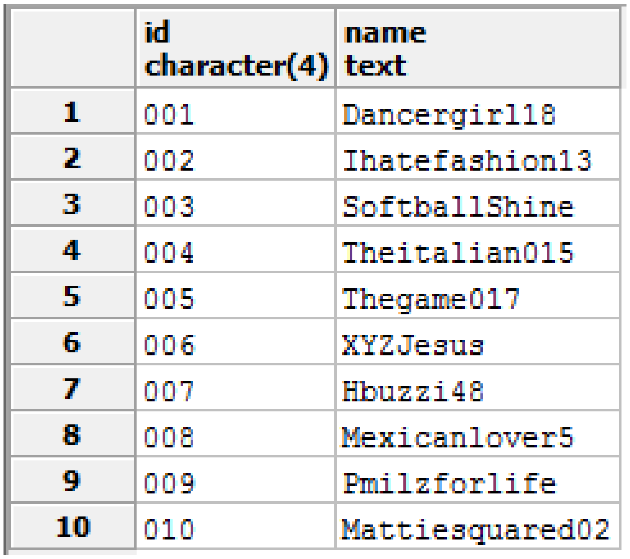
| | id<br>character(4) | name<br>text |
|---|---|---|
| 1 | 001 | Jedi |
| 2 | 002 | Jedi |
| 3 | 003 | Scavenger |
| 4 | 004 | Mercenary |
| 5 | 005 | Scavenger |
| 6 | 006 | Jedi |
| 7 | 007 | Mercenary |
| 8 | 008 | Scavenger |
| 9 | 009 | Mercenary |
| 10 | 010 | Jedi |

GamerTag Table

```
CREATE TABLE Gamertag (
  ID      char(4) not null REFERENCeS Character(ID),
  Name    TEXT,
 primary key(ID)
);
```

This table lists all of the gamertags of the players who have created characters in the game. This links the character directly with the players account name so users may be able to play together in the future.

Functional Dependencies: ID → Name



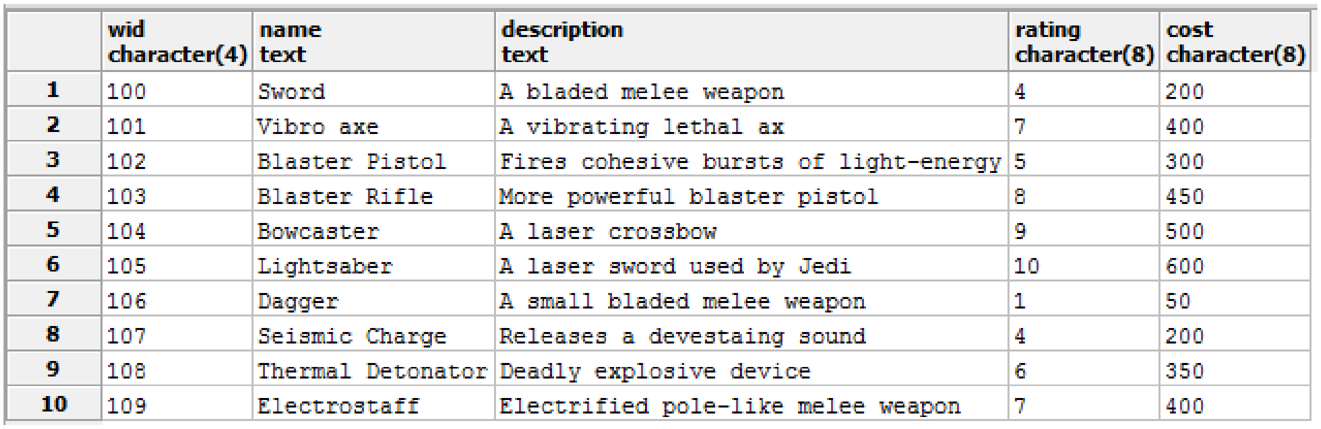
| | id<br>character(4) | name<br>text |
|---|---|---|
| 1 | 001 | Dancergirl18 |
| 2 | 002 | Ihatefashion13 |
| 3 | 003 | SoftballShine |
| 4 | 004 | Theitalian015 |
| 5 | 005 | Thegame017 |
| 6 | 006 | XYZJesus |
| 7 | 007 | Hbuzzi48 |
| 8 | 008 | Mexicanlover5 |
| 9 | 009 | Pmilzforlife |
| 10 | 010 | Mattiesquared02 |

Weapons Table

```
CREATE TABLE Weapons (
  WID       char(4) not null,
  NAME      TEXT,
  Description TEXT,
  Rating    INTEGER  NOT NULL,
  Cost      CHAR(8) NOT NULL,
 primary key(WID)
);
```

This table lists all of the weapons that are in the game. It gives a description of what each weapon is. It also shows the rating of the weapon which will affect the skill level of the player. It also determines the cost of the weapon that the player may chose to sell.

Functional Dependencies: WID → Name, Description, Rating, Cost

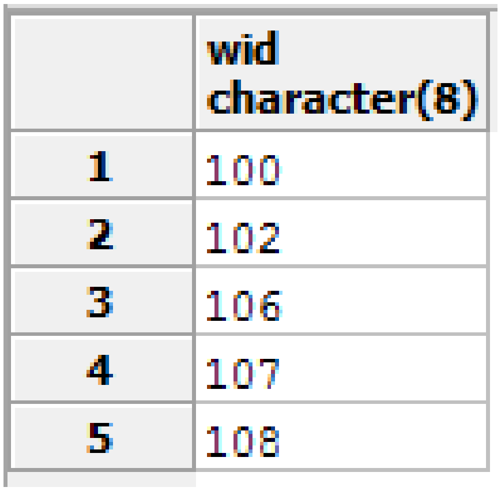


| | wid<br>character(4) | name<br>text | description<br>text | rating<br>character(8) | cost<br>character(8) |
|---|---|---|---|---|---|
| 1 | 100 | Sword | A bladed melee weapon | 4 | 200 |
| 2 | 101 | Vibro axe | A vibrating lethal ax | 7 | 400 |
| 3 | 102 | Blaster Pistol | Fires cohesive bursts of light-energy | 5 | 300 |
| 4 | 103 | Blaster Rifle | More powerful blaster pistol | 8 | 450 |
| 5 | 104 | Bowcaster | A laser crossbow | 9 | 500 |
| 6 | 105 | Lightsaber | A laser sword used by Jedi | 10 | 600 |
| 7 | 106 | Dagger | A small bladed melee weapon | 1 | 50 |
| 8 | 107 | Seismic Charge | Releases a devestaing sound | 4 | 200 |
| 9 | 108 | Thermal Detonator | Deadly explosive device | 6 | 350 |
| 10 | 109 | Electrostaff | Electrified pole-like melee weapon | 7 | 400 |

<u>LightWeapons Table</u>

```
CREATE TABLE LightWeapons (
  WID     char(8) not NULL REFERENCES Weapons(WID),
 primary key(WID)
);
```

This table lists all of the weapons that are classified as “light”. In the future, some of the types of players or certain species might not able to wield anything but light weapons.

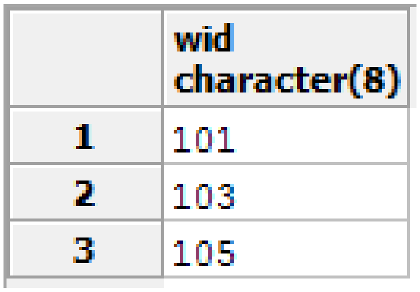


| | wid<br>character(8) |
|---|---|
| 1 | 100 |
| 2 | 102 |
| 3 | 106 |
| 4 | 107 |
| 5 | 108 |

MediumWeapons Table

```
CREATE TABLE MediumWeapons (
  WID     char(8) not NULL REFERENCES Weapons(WID),
 primary key(WID)
);
```

This table lists all of the weapons that are classified as “medium”. In the future, a character might not be able to wield a medium type weapon.

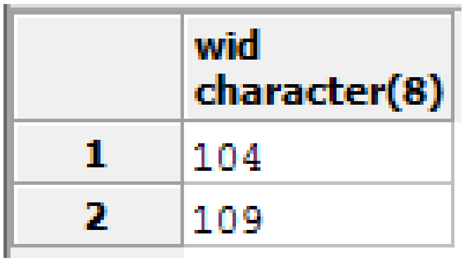


| | wid character(8) |
|---|---|
| 1 | 101 |
| 2 | 103 |
| 3 | 105 |

HeavyWeapons Table

```
CREATE TABLE HeavyWeapons (
  WID     char(8) not NULL REFERENCES Weapons(WID),
 primary key(WID)
);
```

This table lists all of the weapons that are classified as “heavy”. In the future, some types or species will be able to wield this type of weapon rather other species or types may not.



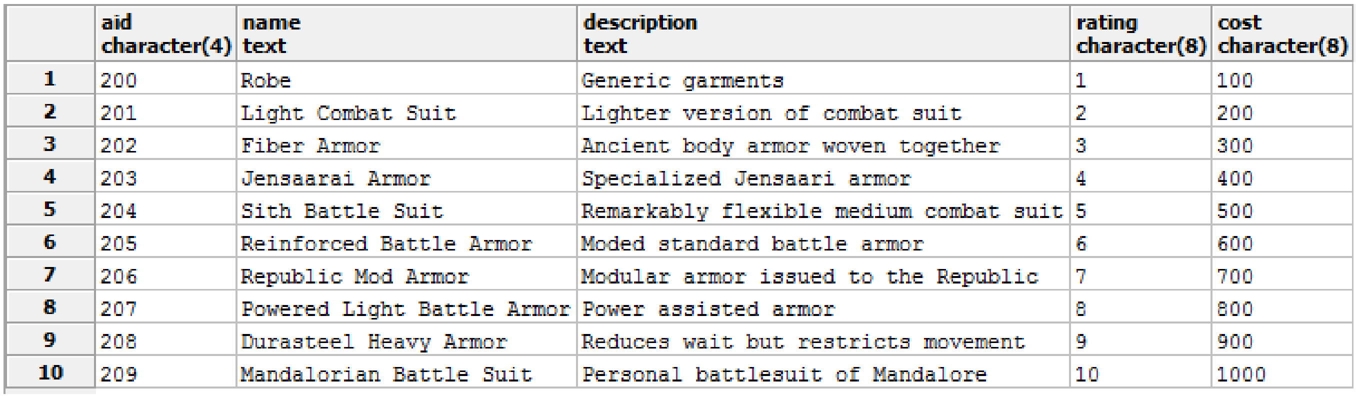
| | wid character(8) |
|---|---|
| 1 | 104 |
| 2 | 109 |

Armor Table

```
CREATE TABLE Armor (
  AID       char(4) not null,
  NAME      TEXT,
  Description TEXT,
  Rating   INTEGER  NOT NULL,
  Cost     CHAR(8) NOT NULL,
 primary key(AID)
);
```

This table lists all of the armor that are in the game. It gives a description of what the armor is. It also shows the rating of the armor and the affect it will have on the player's defense. It also determines the cost of the armor that the player may chose to sell.

Functional Dependencies: AID → Name, Description, Rating, Cost

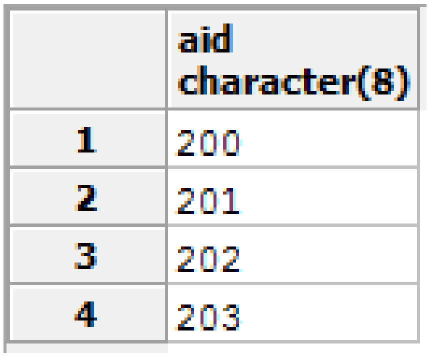


| | aid | name | description | rating | cost |
|---|---|---|---|---|---|
| | character(4) | text | text | character(8) | character(8) |
| 1 | 200 | Robe | Generic garments | 1 | 100 |
| 2 | 201 | Light Combat Suit | Lighter version of combat suit | 2 | 200 |
| 3 | 202 | Fiber Armor | Ancient body armor woven together | 3 | 300 |
| 4 | 203 | Jensaarai Armor | Specialized Jensaari armor | 4 | 400 |
| 5 | 204 | Sith Battle Suit | Remarkably flexible medium combat suit | 5 | 500 |
| 6 | 205 | Reinforced Battle Armor | Moded standard battle armor | 6 | 600 |
| 7 | 206 | Republic Mod Armor | Modular armor issued to the Republic | 7 | 700 |
| 8 | 207 | Powered Light Battle Armor | Power assisted armor | 8 | 800 |
| 9 | 208 | Durasteel Heavy Armor | Reduces wait but restricts movement | 9 | 900 |
| 10 | 209 | Mandalorian Battle Suit | Personal battlesuit of Mandalore | 10 | 1000 |

LightArmor Table

```
CREATE TABLE LightArmor (
  AID     char(8) not NULL REFERENCES Armor(AID),
 primary key(AID)
);
```

This table lists all of the armor that is classified as "light armor". In the future, some types of characters may only be allowed to wear light armor.

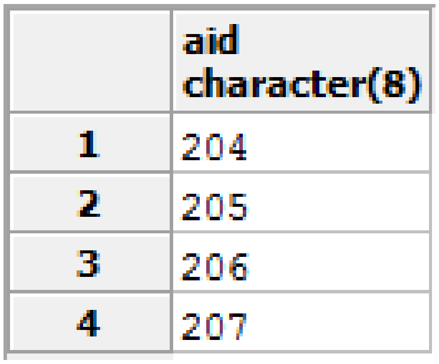


| | aid<br>character(8) |
|---|---|
| 1 | 200 |
| 2 | 201 |
| 3 | 202 |
| 4 | 203 |

MediumArmor Table

```
CREATE TABLE MediumArmor (
  AID     char(8) not NULL REFERENCES Armor(AID),
 primary key(AID)
);
```

This table lists all of the armor that is classified as “medium armor”. In the future, some types of characters may not be permitted or allowed to wear medium armor.

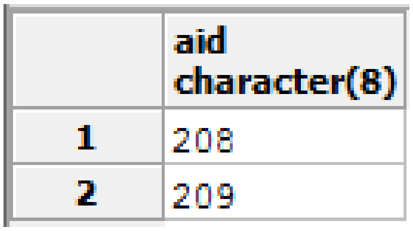


| | aid<br>character(8) |
|---|---|
| 1 | 204 |
| 2 | 205 |
| 3 | 206 |
| 4 | 207 |

HeavyArmor Table

```
CREATE TABLE HeavyArmor (
  AID     char(8) not NULL REFERENCES Armor(AID),
 primary key(AID)
);
```

This table lists all of the armor that is classified as “heavy armor”. In the future, some types of characters may not be permitted or allowed to wear heavy armor.



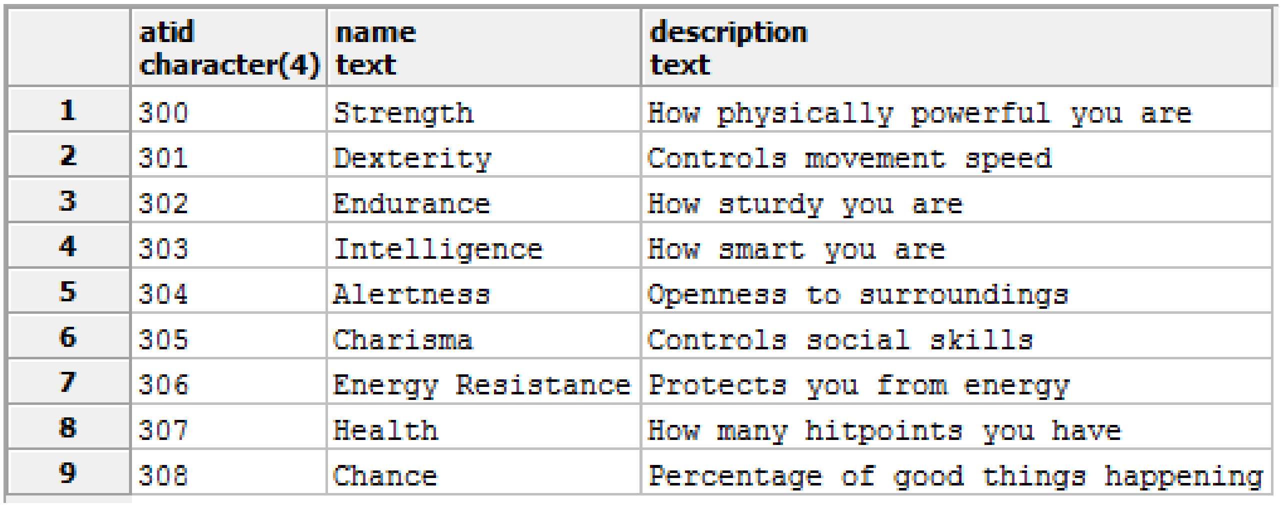
| | aid<br>character(8) |
|---|---|
| 1 | 208 |
| 2 | 209 |

Attributes Table

```
CREATE TABLE Attributes (
  ATID      char(4) not null,
  NAME     TEXT,
  Description TEXT,
  Rank   Char(8) NOT NULL,
 primary key(ATID)
);
```

This table lists all of the attributes that apply to the character in the game. These attributes will affect the player’s skill level. In the future, only certain types will be able to access certain attributes.

Functional Dependencies: ATID → Name, Description, Rank



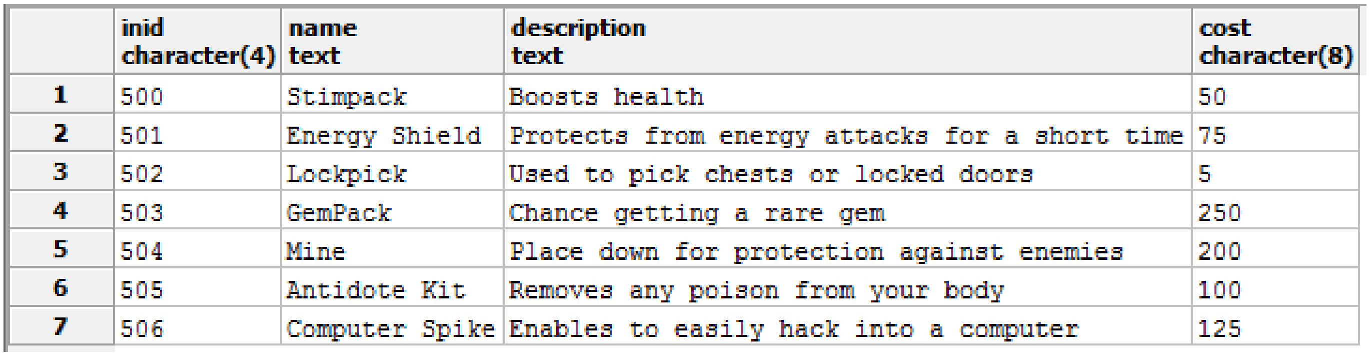
| | atid | name | description |
|---|---|---|---|
| | character(4) | text | text |
| 1 | 300 | Strength | How physically powerful you are |
| 2 | 301 | Dexterity | Controls movement speed |
| 3 | 302 | Endurance | How sturdy you are |
| 4 | 303 | Intelligence | How smart you are |
| 5 | 304 | Alertness | Openness to surroundings |
| 6 | 305 | Charisma | Controls social skills |
| 7 | 306 | Energy Resistance | Protects you from energy |
| 8 | 307 | Health | How many hitpoints you have |
| 9 | 308 | Chance | Percentage of good things happening |

Items Table

```
CREATE TABLE Items (
  INID        char(4) not null,
  NAME        TEXT,
  Description TEXT,
  Cost        INTEGER  NOT NULL,
 primary key(INID)
);
```

This table lists all of the items that can be found in the game. It shows what the item does, what the name of it is, and how much the player can sell it for if the user so chooses. In the future, it will hopefully be able to combine items to weapons or armor to improve them.

Functional Dependencies: INID → Name, Description, Cost

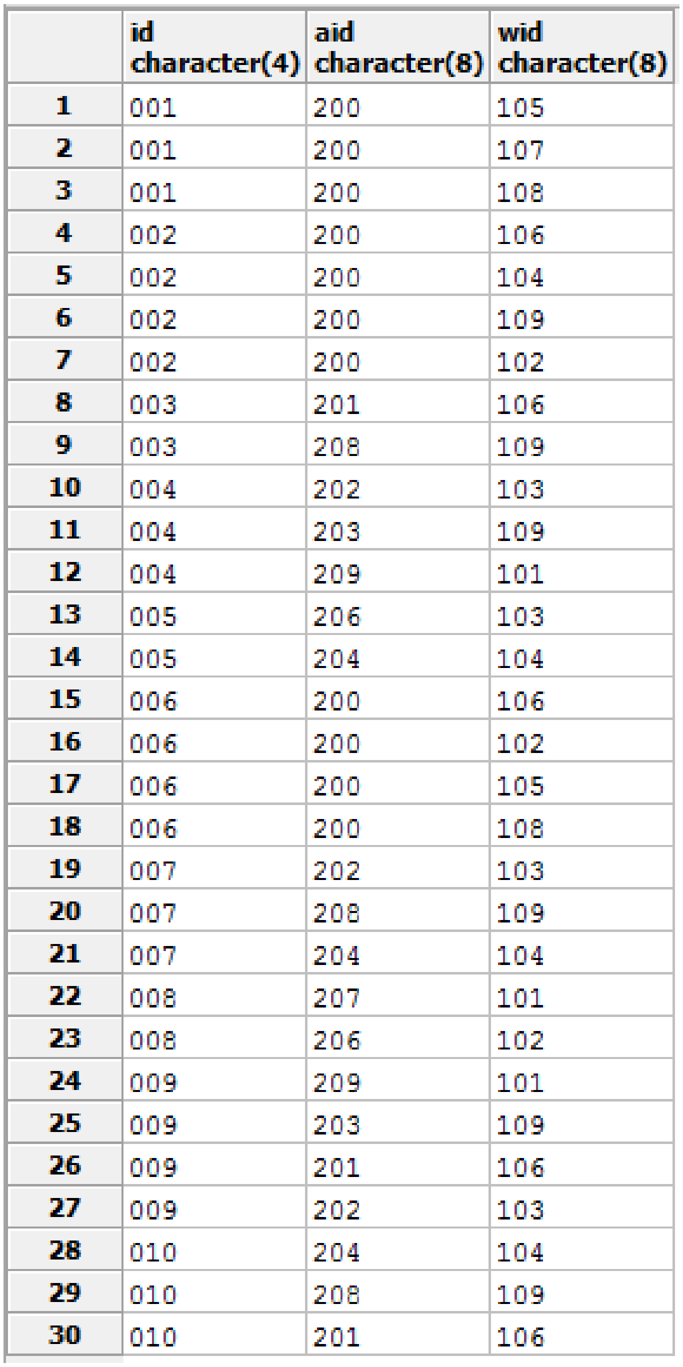


| | inid | name | description | cost |
|---|---|---|---|---|
| | character(4) | text | text | character(8) |
| 1 | 500 | Stimpack | Boosts health | 50 |
| 2 | 501 | Energy Shield | Protects from energy attacks for a short time | 75 |
| 3 | 502 | Lockpick | Used to pick chests or locked doors | 5 |
| 4 | 503 | GemPack | Chance getting a rare gem | 250 |
| 5 | 504 | Mine | Place down for protection against enemies | 200 |
| 6 | 505 | Antidote Kit | Removes any poison from your body | 100 |
| 7 | 506 | Computer Spike | Enables to easily hack into a computer | 125 |

Equipment Table

```
CREATE TABLE Equipment (
  ID      char(4) not null REFERENCeS Character(ID),
  AID     char(8) not NULL REFERENCES Armor(AID),
  WID     char(8) not NULL REFERENCES Weapons(WID),
 primary key(ID, AID, WID)
);
```

This table shows the relationship between the weapons and armor tables to the character table. It shows which weapons and armor are available for the character to find and/or use.



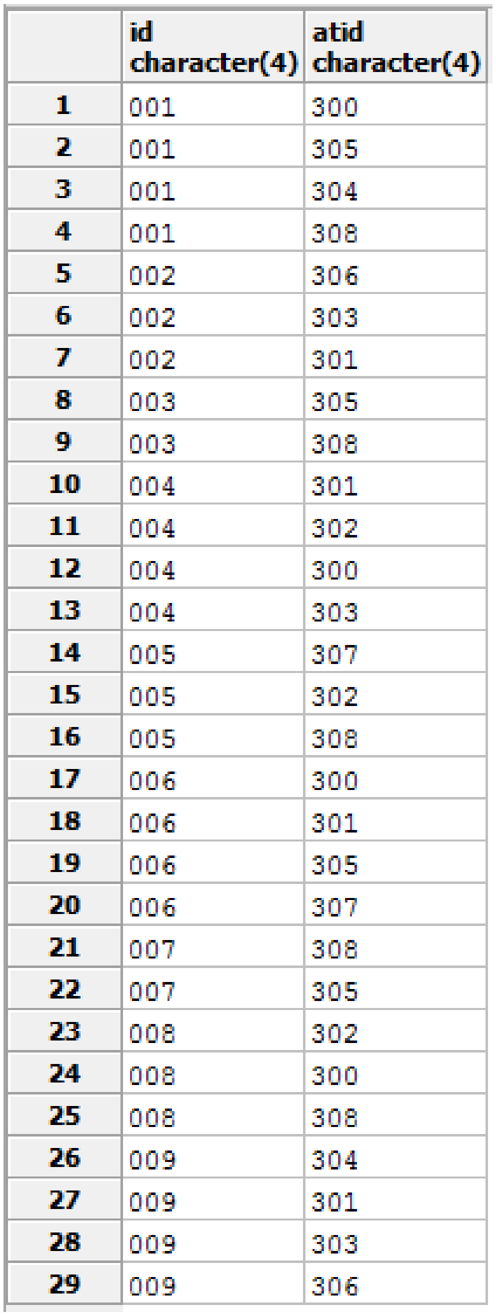
|  | id<br>character(4) | aid<br>character(8) | wid<br>character(8) |
|---|---|---|---|
| 1 | 001 | 200 | 105 |
| 2 | 001 | 200 | 107 |
| 3 | 001 | 200 | 108 |
| 4 | 002 | 200 | 106 |
| 5 | 002 | 200 | 104 |
| 6 | 002 | 200 | 109 |
| 7 | 002 | 200 | 102 |
| 8 | 003 | 201 | 106 |
| 9 | 003 | 208 | 109 |
| 10 | 004 | 202 | 103 |
| 11 | 004 | 203 | 109 |
| 12 | 004 | 209 | 101 |
| 13 | 005 | 206 | 103 |
| 14 | 005 | 204 | 104 |
| 15 | 006 | 200 | 106 |
| 16 | 006 | 200 | 102 |
| 17 | 006 | 200 | 105 |
| 18 | 006 | 200 | 108 |
| 19 | 007 | 202 | 103 |
| 20 | 007 | 208 | 109 |
| 21 | 007 | 204 | 104 |
| 22 | 008 | 207 | 101 |
| 23 | 008 | 206 | 102 |
| 24 | 009 | 209 | 101 |
| 25 | 009 | 203 | 109 |
| 26 | 009 | 201 | 106 |
| 27 | 009 | 202 | 103 |
| 28 | 010 | 204 | 104 |
| 29 | 010 | 208 | 109 |
| 30 | 010 | 201 | 106 |

Quality Table

CREATE TABLE Quality (

```
  ID      char(4) not null REFERENCEs Character(ID),
  ATID    CHAR(4) NOT NULL REFERENCES Attributes(ATID),
 primary KEY(ID, ATID)
);
```

This table shows the relationship between the character table and the attributes table. This is because of the many to many relationship between each other.

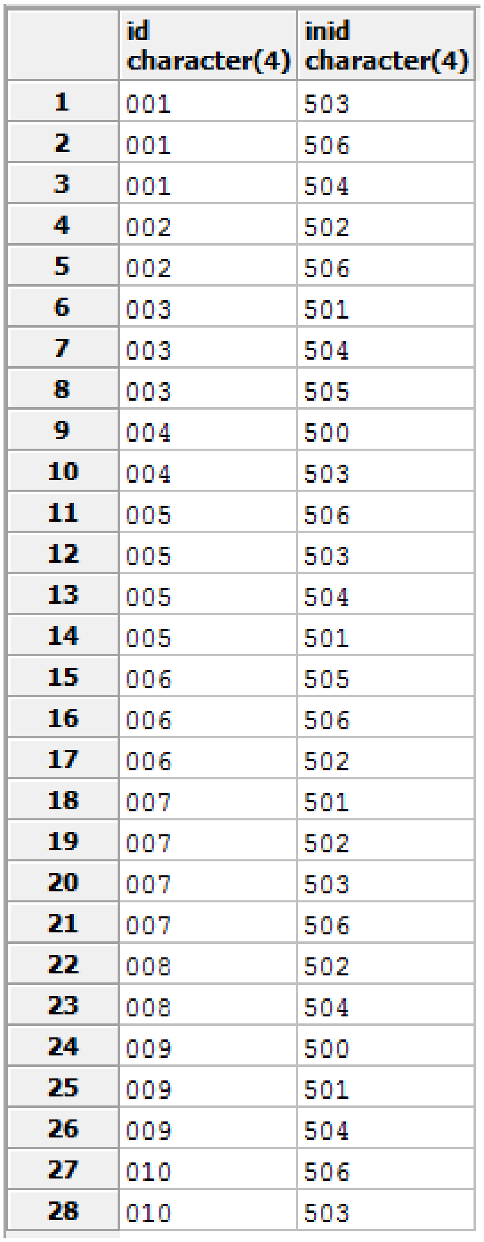


| | id<br>character(4) | atid<br>character(4) |
|---|---|---|
| 1 | 001 | 300 |
| 2 | 001 | 305 |
| 3 | 001 | 304 |
| 4 | 001 | 308 |
| 5 | 002 | 306 |
| 6 | 002 | 303 |
| 7 | 002 | 301 |
| 8 | 003 | 305 |
| 9 | 003 | 308 |
| 10 | 004 | 301 |
| 11 | 004 | 302 |
| 12 | 004 | 300 |
| 13 | 004 | 303 |
| 14 | 005 | 307 |
| 15 | 005 | 302 |
| 16 | 005 | 308 |
| 17 | 006 | 300 |
| 18 | 006 | 301 |
| 19 | 006 | 305 |
| 20 | 006 | 307 |
| 21 | 007 | 308 |
| 22 | 007 | 305 |
| 23 | 008 | 302 |
| 24 | 008 | 300 |
| 25 | 008 | 308 |
| 26 | 009 | 304 |
| 27 | 009 | 301 |
| 28 | 009 | 303 |
| 29 | 009 | 306 |

Inventory Table

```
CREATE TABLE Inventory (
  ID      char(4) not null REFERENCEs Character(ID),
  INID    CHAR(4) NOT NULL REFERENCES Items(INID),
 primary KEY(ID, INID)
);
```

This table shows the relationship between the character table and the items table. This is because of the many to many relationship that they have.



| | id<br>character(4) | inid<br>character(4) |
|---|---|---|
| 1 | 001 | 503 |
| 2 | 001 | 506 |
| 3 | 001 | 504 |
| 4 | 002 | 502 |
| 5 | 002 | 506 |
| 6 | 003 | 501 |
| 7 | 003 | 504 |
| 8 | 003 | 505 |
| 9 | 004 | 500 |
| 10 | 004 | 503 |
| 11 | 005 | 506 |
| 12 | 005 | 503 |
| 13 | 005 | 504 |
| 14 | 005 | 501 |
| 15 | 006 | 505 |
| 16 | 006 | 506 |
| 17 | 006 | 502 |
| 18 | 007 | 501 |
| 19 | 007 | 502 |
| 20 | 007 | 503 |
| 21 | 007 | 506 |
| 22 | 008 | 502 |
| 23 | 008 | 504 |
| 24 | 009 | 500 |
| 25 | 009 | 501 |
| 26 | 009 | 504 |
| 27 | 010 | 506 |
| 28 | 010 | 503 |

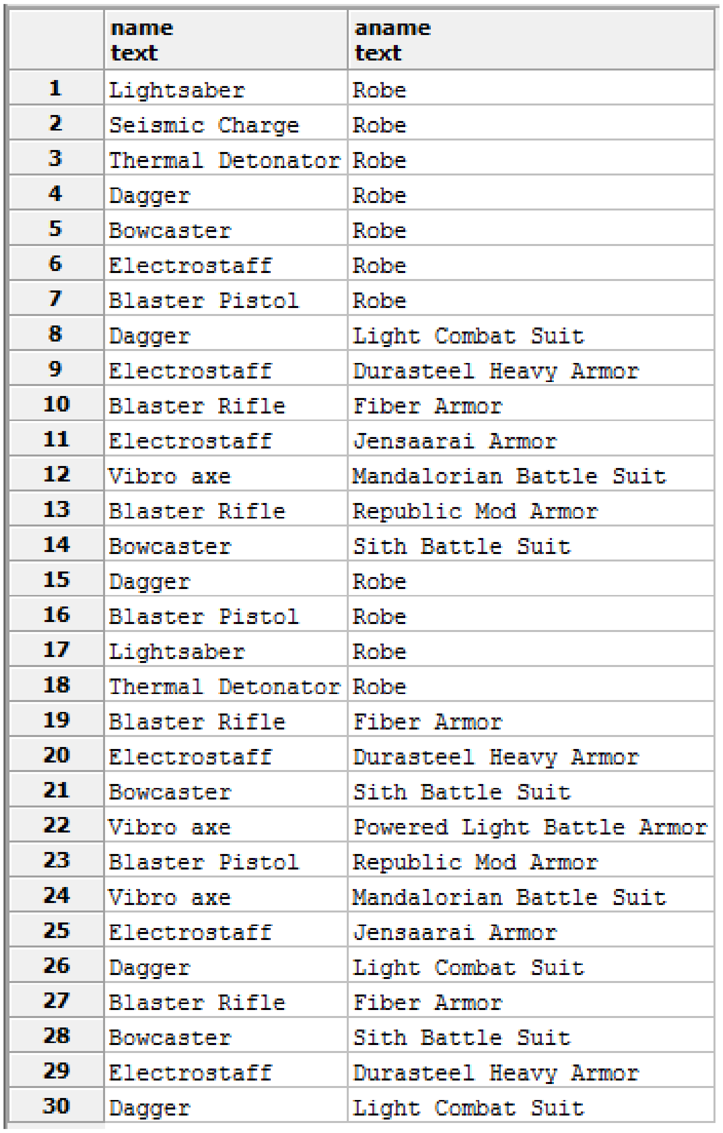
# VIEWS

AllEquipment View

```
DROP VIEW IF EXISTS AllEquipment
CREATE VIEW AllEquipment AS
SELECT w.name AS name, a.name AS Aname
FROM Weapons w, Armor a, Equipment e
WHERE w.wid = e.wid
AND   a.aid = e.aid
```

```
Select *
From AllEquipment;
```

This view creates a separate table in the database to see all of the equipment as a whole, including armor and weapons.

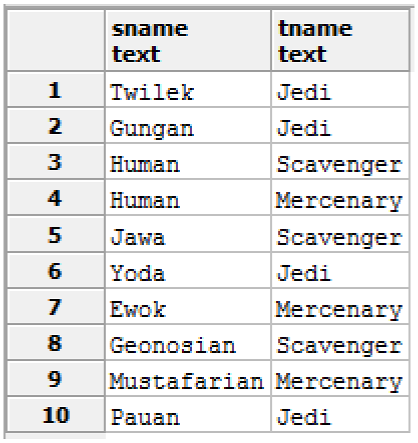


| | name | aname |
|---|---|---|
| | text | text |
| 1 | Lightsaber | Robe |
| 2 | Seismic Charge | Robe |
| 3 | Thermal Detonator | Robe |
| 4 | Dagger | Robe |
| 5 | Bowcaster | Robe |
| 6 | Electrostaff | Robe |
| 7 | Blaster Pistol | Robe |
| 8 | Dagger | Light Combat Suit |
| 9 | Electrostaff | Durasteel Heavy Armor |
| 10 | Blaster Rifle | Fiber Armor |
| 11 | Electrostaff | Jensaarai Armor |
| 12 | Vibro axe | Mandalorian Battle Suit |
| 13 | Blaster Rifle | Republic Mod Armor |
| 14 | Bowcaster | Sith Battle Suit |
| 15 | Dagger | Robe |
| 16 | Blaster Pistol | Robe |
| 17 | Lightsaber | Robe |
| 18 | Thermal Detonator | Robe |
| 19 | Blaster Rifle | Fiber Armor |
| 20 | Electrostaff | Durasteel Heavy Armor |
| 21 | Bowcaster | Sith Battle Suit |
| 22 | Vibro axe | Powered Light Battle Armor |
| 23 | Blaster Pistol | Republic Mod Armor |
| 24 | Vibro axe | Mandalorian Battle Suit |
| 25 | Electrostaff | Jensaarai Armor |
| 26 | Dagger | Light Combat Suit |
| 27 | Blaster Rifle | Fiber Armor |
| 28 | Bowcaster | Sith Battle Suit |
| 29 | Electrostaff | Durasteel Heavy Armor |
| 30 | Dagger | Light Combat Suit |

### SpeciesTypes View

```
DROP VIEW IF EXISTS SpeciesTypes
CREATE VIEW SpeciesTypes AS
Select s.name AS Sname, t.name AS tname
FROM Species s, Type t, character c
WHERE s.id = c.id
AND t.id = c.id

SELECT *
FROM SpeciesTypes;
```

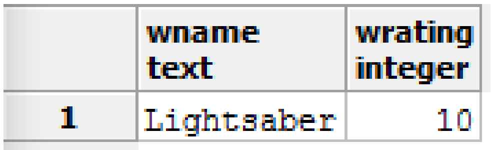


| | sname | tname |
|---|---|---|
| | text | text |
| 1 | Twilek | Jedi |
| 2 | Gungan | Jedi |
| 3 | Human | Scavenger |
| 4 | Human | Mercenary |
| 5 | Jawa | Scavenger |
| 6 | Yoda | Jedi |
| 7 | Ewok | Mercenary |
| 8 | Geonosian | Scavenger |
| 9 | Mustafarian | Mercenary |
| 10 | Pauan | Jedi |

# Reports

```
SELECT w.name AS wname, w.rating as wrating
FROM weapons w
ORDER BY Rating DESC
LIMIT 1;
```

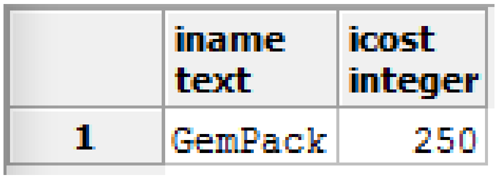
This query returns the name and value of the weapon with the highest rating and only returns one weapon.



| | wname<br>text | wrating<br>integer |
|---|---|---|
| 1 | Lightsaber | 10 |

```
SELECT i.name AS iname, i.cost as icost
FROM items i
ORDER BY cost DESC
limit 1;
```

This query returns the name and value of the item that costs the most money and only returns one item.



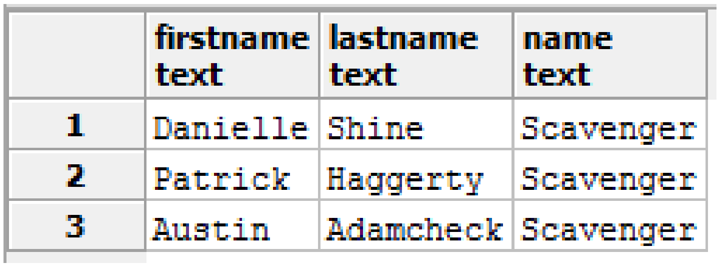
| | iname<br>text | icost<br>integer |
|---|---|---|
| 1 | GemPack | 250 |

# Stored Procedure

```
CREATE or REPLACE FUNCTION CharacterType(TEXT, REFCURSOR)
RETURNS REFCURSOR AS $$
DECLARE
  Player TEXT              :=$1;
  ResultSet REFCURSOR   :=$2;

BEGIN
  OPEN ResultSet FOR
  SELECT c.firstname, c.lastname, t.name
  FROM type t, character c
  WHERE t.id = c.id
  AND name = Player;
RETURN ResultSet;
END;
$$
LANGUAGE plpgsql;


SELECT CharacterType('Scavenger', 'results');
FETCH ALL FROM results;
```

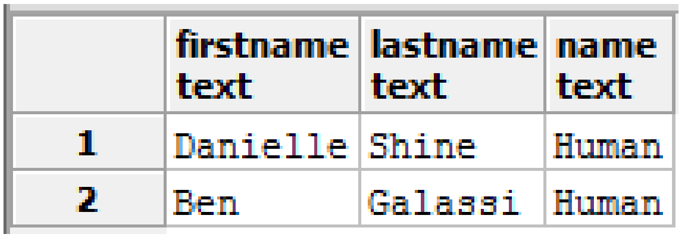


| | firstname | lastname | name |
|---|---|---|---|
| | text | text | text |
| 1 | Danielle | Shine | Scavenger |
| 2 | Patrick | Haggerty | Scavenger |
| 3 | Austin | Adamcheck | Scavenger |

```
CREATE or REPLACE FUNCTION SpeciesType(TEXT, REFCURSOR)
RETURNS REFCURSOR AS $$
DECLARE
  Race TEXT                    :=$1;
  ResultSet REFCURSOR   :=$2;

BEGIN
  OPEN ResultSet FOR
  SELECT c.firstname, c.lastname, s.name
  FROM Species s, character c
  WHERE s.id = c.id
  AND name = Race;
RETURN ResultSet;
END;
$$
LANGUAGE plpgsql;


SELECT SpeciesType('Human', 'results');
FETCH ALL FROM results;
```



| | firstname<br>text | lastname<br>text | name<br>text |
|---|---|---|---|
| 1 | Danielle | Shine | Human |
| 2 | Ben | Galassi | Human |

# Security

Administrator: The administrator would be allowed to change content within the database such as altering current content or adding new content.

```
CREATE ROLE admin;

GRANT SELECT, DELETE, UPDATE, INSERT
ON ALL TABLES IN SCHEMA public
TO admin;
```

User: A user would only be able to view the database. The user would not be given permissions to change or add new content.

```
CREATE ROLE OTHER;

GRANT SELECT
ON ALL TABLES IN SCHEMA public
TO OTHER;
```

## Implementation Notes-Known Problems

The implementation went well but I definitely see a lot of room for improvement. The setup is pretty basic for a regular star wars video game, but there are a lot of possibilities to tackle. One problem to tackle is that each type of character can access all of the armor even if it doesn't make sense for them to be able to. I think for the design to be so general that there are not too many problems, but to add more details and more enhancements would cause more problems to occur.

## Future Enhancements

This database could be improved by creating a way for attributes, armor level, and weapon level to increase the skill of the user. This way, the user would be allowed to level up the player and take on harder challenges in the game. It could also be improved by each item in the game having a standard cost and sell price but would change depending on the level of the user to make it hard to buy certain items.

The main update of the design would to only allow certain species to gain access to certain types of weapons and armor. For example, only a jedi should be able to wield a lightsaber or a mercenary to wear heavy armor.