



# United Federation of Planets



By Joseph Gust



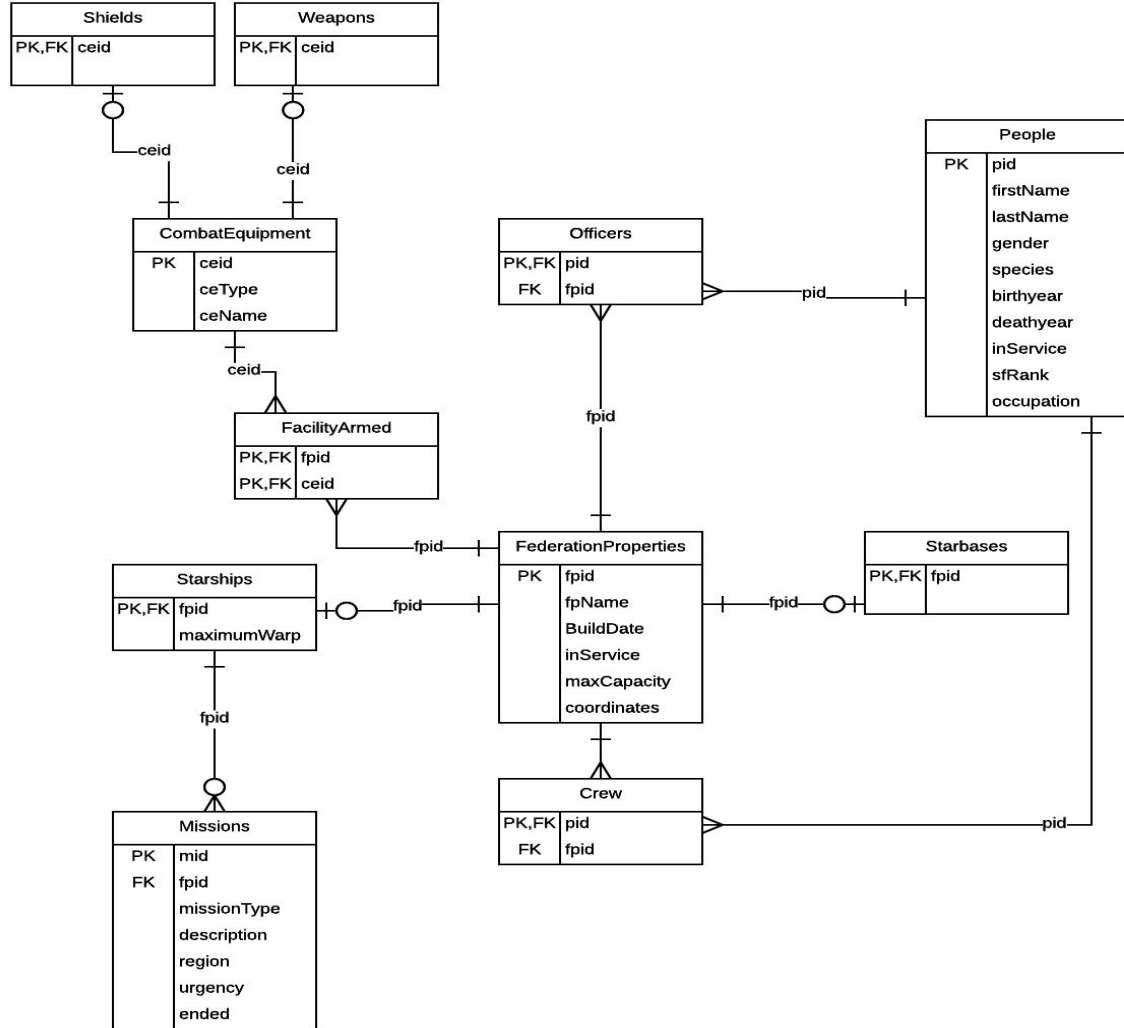
# Table of Contents

Executive Summary .....	3	Officers Table .....	10
ER Diagram .....	4	Create Views .....	11
People Table .....	5	Stored Procedure .....	13
Federation Prop Table .....	7	Trigger .....	15
Starships Table .....	8	Security .....	16
Starbases Table .....	9	Future Enhancements .....	19

# Executive Summary

---

The United Federation of Planets needs a way to keep tracks of their many ships, starbases, and people. This database keeps track of everyone working for the federation, as well as what ship or starbase they are assigned to. It also keeps track of what weaponry each ships is equipped with as well as what missions that have.



# People Table

```
CREATE TABLE People (
```

```
  pid          char(5) NOT NULL,
```

```
  firstName    TEXT NOT NULL,
```

```
  lastName     TEXT,
```

```
  gender       char(1),
```

```
  species      TEXT NOT NULL,
```

```
  birthyear    integer,
```

# People Continued

```
deathyear    integer,  
inService    boolean NOT NULL,  
sfRank       TEXT NOT NULL,  
occupation   TEXT,  
PRIMARY KEY (pid)  
);
```

**Functional Dependencies:** Every attribute besides pid is dependent on pid.

# Federation Prop Table

```
CREATE TABLE FederationProperties (
```

```
  fpid          char(4) NOT NULL,
```

```
  fpName       TEXT NOT NULL,
```

```
  buildDate    integer NOT NULL,
```

```
  inService    boolean NOT NULL,
```

```
  maxCapacity  integer NOT NULL,
```

```
  coordinates  TEXT NOT NULL,
```

```
  PRIMARY KEY (fpid)
```

```
);
```

**Functional Dependencies:** all non-fpid attributes dependent on fpid

# Starships Table

---

```
CREATE TABLE Starships (  
  fpid          char(4) NOT NULL REFERENCES FederationProperties(fpid),  
  maximumWarp   integer NOT NULL,  
  PRIMARY KEY (fpid)  
);
```

**Functional Dependencies:** maximumWarp dependent on fpid.



# Starbases Table

```
CREATE TABLE Starbases (  
    fpid          char(4) NOT NULL REFERENCES FederationProperties(fpid),  
    PRIMARY KEY   (fpid)  
);
```

**Functional Dependencies:** none.

# Officers Table

---

```
CREATE TABLE Officers (
```

```
  pid          char(5) NOT NULL REFERENCES People(pid),
```

```
  fpid         char(4) NOT NULL REFERENCES FederationProperties(fpid),
```

```
  PRIMARY KEY  (pid,fpid)
```

```
);
```

**Functional Dependencies:** none.

# Create Views

---

-- View shows on which ship or starbase each officer is stationed

CREATE view

FPOfficers(fpid,fpName,pid,firstName,lastName,sfRank,occupation) as

select fp.fpid,fp.fpName,p.pid,p.firstName,p.lastName,p.sfRank,p.occupation

from officers o inner join FederationProperties fp on o.fpid = fp.fpid

inner join People p on o.pid = p.pid

;

# Create Views

-- View shows on which ship or starbase each (non-officer) crew member is stationed

```
CREATE view FPCrew(fpid,fpName,pid,firstName,lastName,sfRank,occupation) as  
  
  select    fp.fpid,fp.fpName,p.pid,p.firstName,p.lastName,p.sfRank,p.occupation  
  
  from      crew c   inner join FederationProperties fp on c.fpid = fp.fpid  
  
              inner join People p on c.pid = p.pid  
  
;
```

# Stored Procedure

-- stored procedure: if type w -> insert ceid into weapons | if type s-> insert into shields

```
CREATE OR REPLACE FUNCTION addCE()
```

```
RETURNS TRIGGER AS $$
```

```
BEGIN
```

```
    IF NEW.ceType = 'w' THEN
```

```
        INSERT INTO Weapons(ceid)
```

```
        values(NEW.ceid);
```

```
    END IF;
```

# Stored Procedure Continued

---

```
IF NEW.ceType = 's' THEN  
    INSERT INTO Shields(ceid)  
        values(NEW.ceid);  
  
END IF;  
  
RETURN NEW;  
  
END;  
  
$$ language plpgsql;
```

# Trigger

---

-- If something is inserted in combatEquipment, run addCE()

CREATE TRIGGER addCE

AFTER INSERT OR UPDATE ON combatEquipment

FOR EACH ROW

EXECUTE PROCEDURE addCE();

# Security

---

-- An administrator can alter or even delete data

create role Administrator;

grant select, insert, update, delete

on all tables in schema public

to Administrator;



# Security

---

-- Officers can add updates to the data, but they cannot delete data

create role Officer;

grant select, insert, update

on all tables in schema public

to Officer;

# Security

---

- other (non-officer) crew members can view people and federation
- properties, as well as where crew members are stationed
- but they cannot view where the officers are stationed

create role CrewMember;

grant select

on People, FederationProperties, Crew, Starships, Starbases

to CrewMember;

# Future Enhancements

---

I would like to extend the database to include a Planets table for planets that are in the federation. I would like the planets to be a child of fpid just like starships and starbases so people could be stationed on planets.