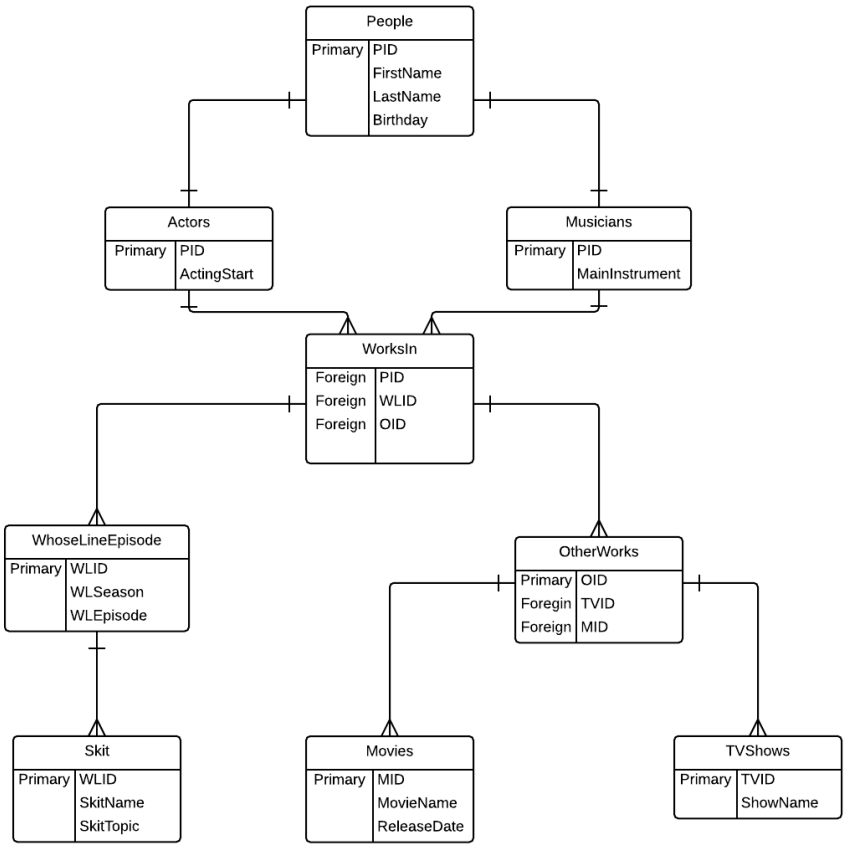
Executive Summary

The Whose Line is it Anyway Database can mainly be used to see what other production actors that have made an appearance in a Whose Line is it Anyway episode have worked in. It consists of Whose Line Episodes and skits that were associated with that episodes, and also T.V.'s and Movies they have appeared in as well. The database also has the dates that the actors started working. This database is a fun way to make connections as obscure as linking a specific skit in a Whose Line episode with a movie that was made a few decades prior to the airing of the skit. It is relatively incomplete, but has a lot of room for more information to be put into it, such as more actors, and even other crew members from Whose Line and what other productions they've worked on.

Entity Diagram



Tables

People

Create Statement

```
CREATE TABLE People(
PID SERIAL NOT NULL,
FirstName TEXT NOT NULL,
LastName TEXT NOT NULL,
Birthday DATE NOT NULL,
PRIMARY KEY (PID)
);
```

Sample Data

```
INSERT INTO People (FirstName, LastName, Birthday)
VALUES  ('Drew', 'Carey', '1958-05-23'),
        ('Ryan', 'Styles', '1959-04-22'),
        ('Robin', 'Williams', '1951-07-21'),
        ('Stephen', 'Colbert', '1964-05-13');
```

Functional Dependencies:

FirstName, LastName, Birthday → PID

	pid integer	firstname text	lastname text	birthday date
1	1	Drew	Carey	1958-05
2	2	Ryan	Styles	1959-04
3	3	Robin	Williams	1951-07
4	4	Stephen	Colbert	1964-05

Actors

Create Statement

```
CREATE TABLE Actors(
PID SERIAL REFERENCES People,
ActingStart INT NOT NULL
);
```

Sample Data

```
INSERT INTO Actors (ActingStart)
VALUES    ('1991'),
          ('1989'),
          ('1984'),
          ('1976');
```

Functional Dependencies

ActingStart → PID

	pid integer	actingstart integer
1	1	1991
2	2	1989
3	3	1984
4	4	1976

Musicians

Create Statement

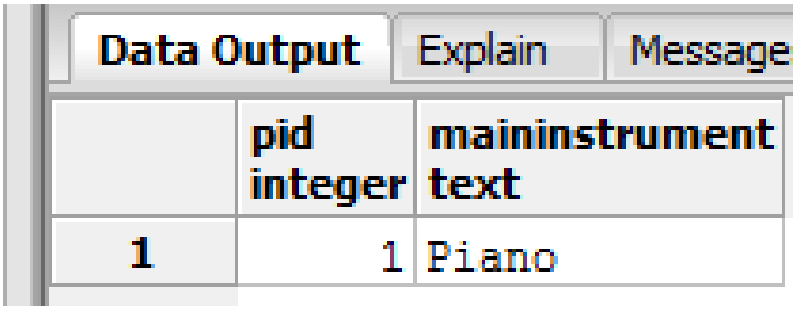
```
CREATE TABLE Musicians(
PID SERIAL REFERENCES People,
MainInstrument TEXT NOT NULL
);
```

Sample Data

```
INSERT INTO Musicians (MainInstrument)
VALUES    ('Piano');
```

Functional Dependencies

MainInstrument → PID



Data Output | Explain | Message

	pid integer	maininstrument text
1	1	Piano

WorksIn

Create Statement

```
CREATE TABLE WorksIn(
PID INT REFERENCES People (PID),
WLID INT REFERENCES WhoseLineEpisodes (WLID),
OID INT REFERENCES OtherWorks (OID)
);
```

Sample Data

```
INSERT INTO WorksIn (PID, WLID)
VALUES    ('1', '1'), ('1', '2'), ('1', '3'), ('1', '4'),
          ('2', '1'), ('2', '2'), ('2', '3'), ('2', '4'),
          ('3', '1'), ('3', '2'), ('3', '3'), (‘3’, ‘4’),
          ('4', '1'), ('4', '2'), ('4', '3'), ('4', '4'),
          ('5', '1'), ('5', '2'), ('5', '3'), ('5', '4');

INSERT INTO WorksIn (PID, OID)
VALUES    ('1', '1'), ('1', '2'), ('1', '9'), ('1', '10'),
          ('2', '3'), ('2', '4'), ('2', '11'), ('2', '12'),
          ('3', '5'), ('3', '6'), ('3', '13'), ('3', '14'),
          ('4', '7'), ('4', '8'), ('4', '15'), ('4', '16');
```

Functional Dependencies

OID, WLID → PID

	pid integer	wlid integer	oid integer
13	4	1	
14	4	2	
15	4	3	
16	4	4	
17	5	1	
18	5	2	
19	5	3	
20	5	4	
21	1		1
22	1		2
23	1		9
24	1		10
25	2		3
26	2		4
27	2		11
28	2		12
29	3		5

WhoseLineEpisodes

Create Statement

```
CREATE TABLE WhoseLineEpisodes(
WLID SERIAL,
WLSeason INT NOT NULL,
WLEpisode INT NOT NULL,
PRIMARY KEY (WLID)
);
```

Sample Data

```
INSERT INTO WhoseLineEpisodes (WLSeason, WLEpisode)
VALUES    ('1', '10'),
          ('1', '17'),
          ('3', '9'),
          ('3', '1');
```

Functional Dependencies

WLSeason, WLEpisode → WLID

	wlid integer	wlseason integer	wlepisode integer
1	1	1	10
2	2	1	17
3	3	3	9
4	4	3	1

OtherWorks

Create Statement

```
CREATE TABLE OtherWorks(
OID INT NOT NULL,
TVID INT,
MID INT,
PRIMARY KEY (OID)
);
```

Sample Data

```
INSERT INTO OtherWorks (OID, MID)
VALUES    ('1', '1'), ('2', '2'), ('3', '3'), ('4', '4'),
          ('5', '5'), ('6', '6'), ('7', '7'), ('8', '8');

INSERT INTO OtherWorks (OID, TVID)
VALUES    ('9', '1'), ('10', '2'), ('11', '3'), ('12', '4'),
          ('13', '5'), ('14', '6'), ('15', '7'), ('16', '8');
```

Functional Dependencies

MID, TVID → PID

	oid integer	tvid integer	mid integer
1	1		1
2	2		2
3	3		3
4	4		4
5	5		5
6	6		6
7	7		7
8	8		8
9	9	1	
10	10	2	
11	11	3	
12	12	4	
13	13	5	
14	14	6	
15	15	7	
16	16	8	

Skits

Create Statement

```
CREATE TABLE Skits(
WLID SERIAL REFERENCES WhoseLineEpisodes (WLID),
SkitName TEXT,
SkitTopic TEXT
);
```

Sample Data

```
INSERT INTO Skits (SkitName)
VALUES    ('Scenes From a Hat'),
          ('HoeDown'),
          ('Props'),
          ('Three Headed Broadway Star');
```

Functional Dependencies

SkitName, SkitTopic → WLID

	wlid integer	skitname text	skittopic text
1	1	Scenes From a Hat	
2	2	HoeDown	
3	3	Props	
4	4	Three Headed Broadway Star	

Movies

Create Statement

```
CREATE TABLE Movies(
MID INT NOT NULL,
Name TEXT NOT NULL,
ReleaseDate DATE NOT NULL,
PRIMARY KEY (MID)
);
```

Sample Data

```
INSERT INTO Movies (MID, Name, ReleaseDate)
VALUES      ('1', 'Jack and Jill', '2011-11-11'),
            ('2', 'Robots', '2005-03-11'),
            ('3', 'Courting Courtney', '1997-10-30'),
            ('4', 'Astro Boy', '2009-10-23'),
            ('5', 'The Hobbit: Desolation of Smaug', '2013-12-13'),
            ('6', 'Mr. Peabody & Sherman', '2014-03-07'),
            ('7', 'Night at the Museum', '2006-12-22'),
            ('8', 'Good Will Hunting', '1998-01-09');
```

Functional Dependencies

MovieName, ReleaseDate → MID

	mid integer	moviename text	releasedate date
1	1	Jack and J	2011-11-11
2	2	Robots	2005-03-11
3	3	Courting C	1997-10-30
4	4	Astro Boy	2009-10-23
5	5	The Hobbit	2013-12-13
6	6	Mr. Peabod	2014-03-07
7	7	Night at t	2006-12-22
8	8	Good Will	1998-01-09

TVShows

Create Statement

```
CREATE TABLE TVShows(
TVID INT NOT NULL,
Name TEXT NOT NULL,
PRIMARY KEY (TVID)
);
```

Sample Data

```
INSERT INTO TVShows (TVID, Name)
VALUES    ('1', 'The Drew Carey Show'),
          ('2', 'The Price Is Right'),
          ('3', 'Reno 911!'),
          ('4', 'Rugrats'),
          ('5', 'The Colbert Report'),
          ('6', 'The O''Reilly Factor'),
          ('7', 'Friends'),
          ('8', 'Mork & Mindy');
```

Functional Dependencies

ShowName → TVID

	mid integer	moviename text	releasedate date
1	1	Jack and J	2011-11-11
2	2	Robots	2005-03-11
3	3	Courting C	1997-10-30
4	4	Astro Boy	2009-10-23
5	5	The Hobbit	2013-12-13
6	6	Mr. Peabod	2014-03-07
7	7	Night at t	2006-12-22
8	8	Good Will	1998-01-09

Views and Sample Output

The purpose of this view is to select all of the other work that Drew Carey has done besides “Whose Line is it Anyway?”

```
DROP VIEW IF EXISTS Drew_Carey_Other;
CREATE VIEW Drew_Carey_Other AS
SELECT TVShows.ShowName, Movies.MovieName
FROM TVShows
FULL JOIN OtherWorks
ON TVShows.TVID=OtherWorks.TVID
FULL JOIN Movies
ON Movies.MID=OtherWorks.MID
WHERE OID IN
     (SELECT OID FROM WorksIn
     WHERE PID IN
          (SELECT PID FROM People
          WHERE FirstName = 'Drew'));

SELECT * FROM Drew_Carey_Other;
```

	showname text	moviename text
1		Jack and Jill
2	The Price Is Right	
3	The Drew Carey Show	
4		Robots

Reported Query

The purpose of this query is to show that the database is functional by selecting all of the Movies that Ryan Stiles has acted in.

```
SELECT * FROM Movies
WHERE MID IN
     (SELECT MID FROM OtherWorks
     WHERE OID IN
          (SELECT OID FROM WorksIn
          WHERE PID IN
               (SELECT PID FROM People
               WHERE FirstName = 'Ryan')));
```

	mid integer	moviename text	releasedate date
1	3	Courting Courtney	1997-10-30
2	4	Astro Boy	2009-10-23

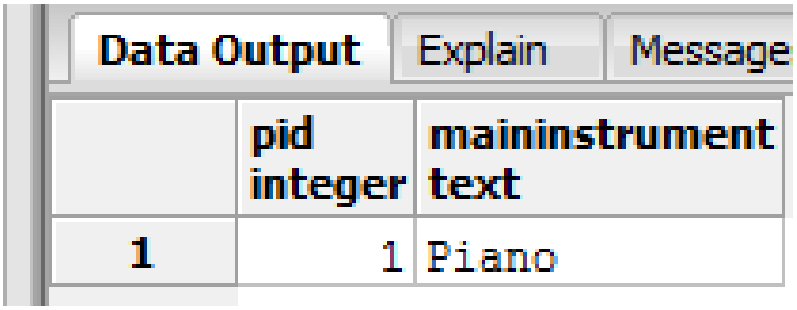
Stored Procedure

The purpose of this stored procedure is to default a “null” value in a musicians main instrument to

“piano” so there is not a null in the MainInstrument column.

```
CREATE FUNCTION DefaultInstrument()
RETURNS TABLE (MainInstrument TEXT) AS $$
BEGIN
     UPDATE Musicians
     SET MainInstrument="Piano"
     WHERE MainInstrument = null;
END
$$ LANGUAGE PLPGSQL

select DefaultInstrument();
Fetch all from results;
```



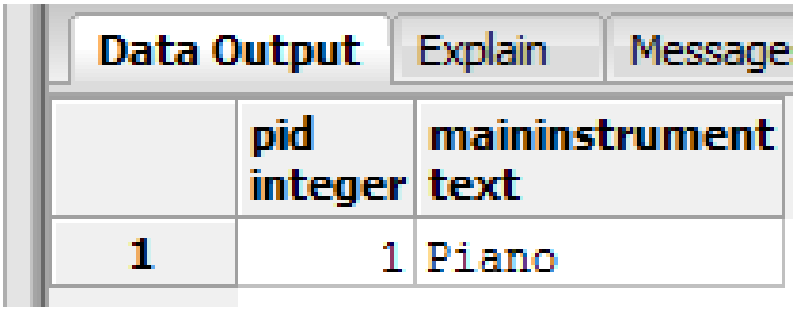
Data Output | Explain | Message

	pid integer	maininstrument text
1	1	Piano

Trigger

The purpose of this trigger is to run the stored procedure during an insert on the table “Musicians” so there is not a null for the column “MainInstrument”

```
CREATE TRIGGER DefaultInstrument
AFTER INSERT ON Musicians
FOR EACH ROW
EXECUTE PROCEDURE DefaultInstrument();
```



Data Output | Explain | Message

	pid integer	maininstrument text
1	1	Piano

Security

```
CREATE ROLE admin WITH LOGIN PASSWORD 'alpaca';
GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES TO admin;

CREATE ROLE Drew_Carey WITH LOGIN PASSWORD 'Points';
GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES TO
Drew_Carey;

CREATE ROLE Ryan_Stiles WITH LOGIN PASSWORD 'ILoveDrewCarey'
REVOKE SELECT, INSERT, UPDATE, DELETE, ON ALL TABLES TO
Ryan_Stiles;
```

Implementation Notes

Since there is no pre-existing database, there will be no issue with transferring files. It can be implemented quickly and easily with little time or effort. The code that is written also comes with user names and passwords so it is a secure database.

Known Errors

The database, due to many foreign keys, has a lot of nulls in a few different tables. Although there could be a better design, for now the database is functional and easy to read. Other errors include that musicians can have multiple instruments, so the column MainInstrument can be misleading/uninformative. Also, none of the skit names have topics due to the inability to find working episodes of Whose Line is it Anyway on the internet, so it is impossible to figure out which skit from a certain episode had a specific topic.

Future Implementations

The Actors and Musicians as a subentity of people allows for more types of people to be added to the database such as directors, producers, writers, etc. Because a director IS A person, a writer IS A person, and a producer IS A person. Also, it will be very easy to add new movies and TV shows as a person takes more and more roles. Since Whose Line is it Anyway is still running, it will also be able to accept more inserts of whose line episodes and skits. Also, there can be a future implementation for a separate entity of works that are not movies or TV shows, such as voice acting roles.