Final Project: Databases, Andrew Johnson

Outline

I have created a database representing the relationship of people involved in making motion pictures. The entities included are actors, spouses, directors, and films. Relationships are marriages, actors acting in films, directors directing films, actors in religions, spouses in religions, and directors in religions.

The relationships are nothing technical and would be familiar to everyone. I thought it would be good to learn with a familiar example, although many people are acutely interested in celebrities.

Database Outline in Words

Actors have unique identifier ID, first name, last name, and birth date. Actors must have an ID, first name, and last name they cannot be null.

Spouses have unique identifier ID, first name, last name, and birth date. Spouses must have an ID, first name, and last name they cannot be null.

Directors have unique identifier ID, first name, last name, and birth date. Directors must have an ID, first name, and last name they cannot be null.

Films have unique film ID, title, starting budget, and projected release date. Films must have a title it cannot be null.

Religions have unique religion ID, name of religion, and date of founding. Religions must have a name, it cannot be null.

Actors will have 0 or 1 religion. They will also have 0 or 1 spouse as well. Actors can be in 1 or more films.

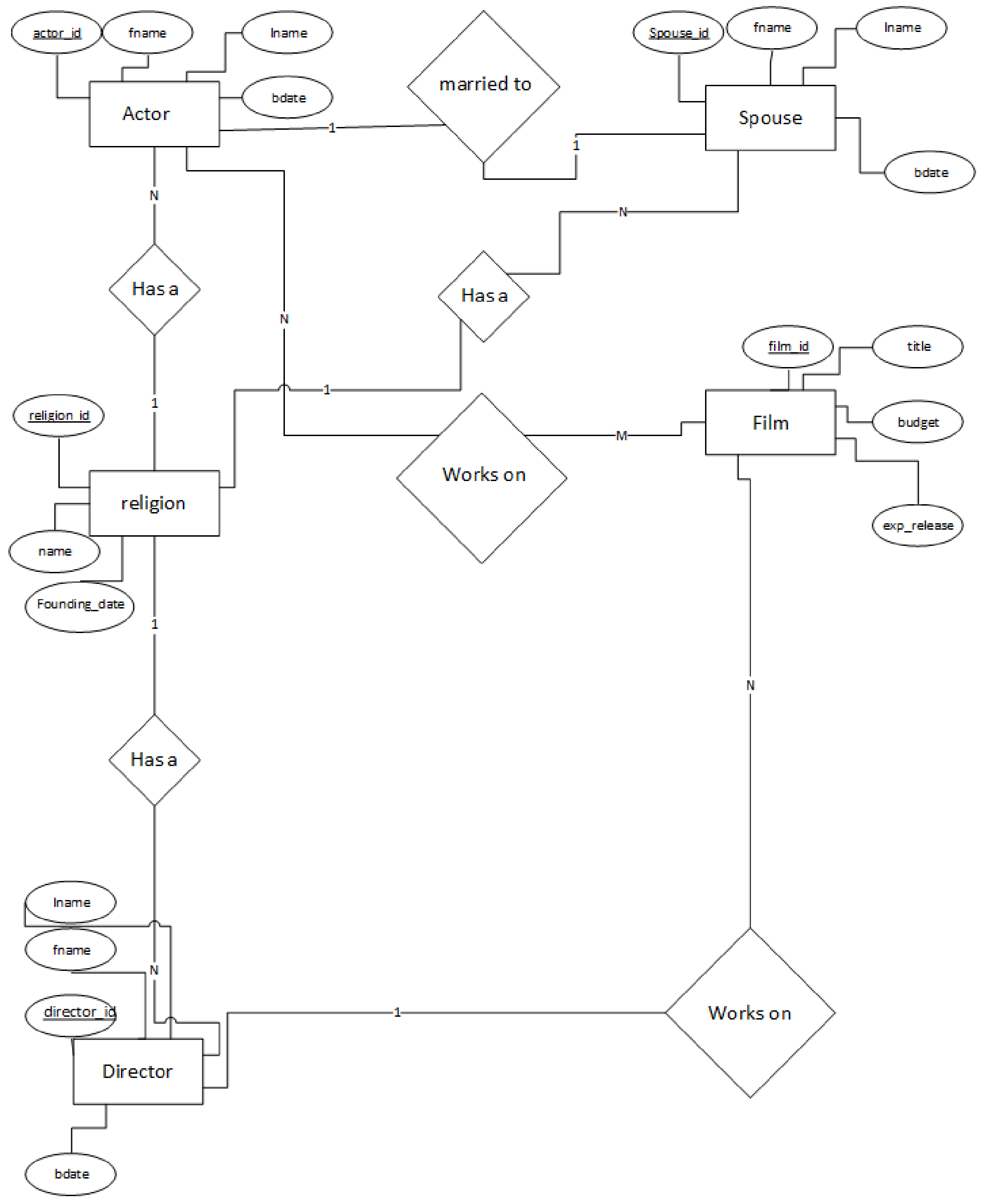
Directors will have 0 or more films that they are working on, directors can work on more than one film, but films can only have 0 or 1 directors. Directors will have 0 or 1 religion. Director spouses are not tracked.

Films will have 0 or 1 directors and 0 or more actors (we can have a film without actors here because we are not interested in voice actors, so animated films will have no actors, also films might not have a director in the planning stages).

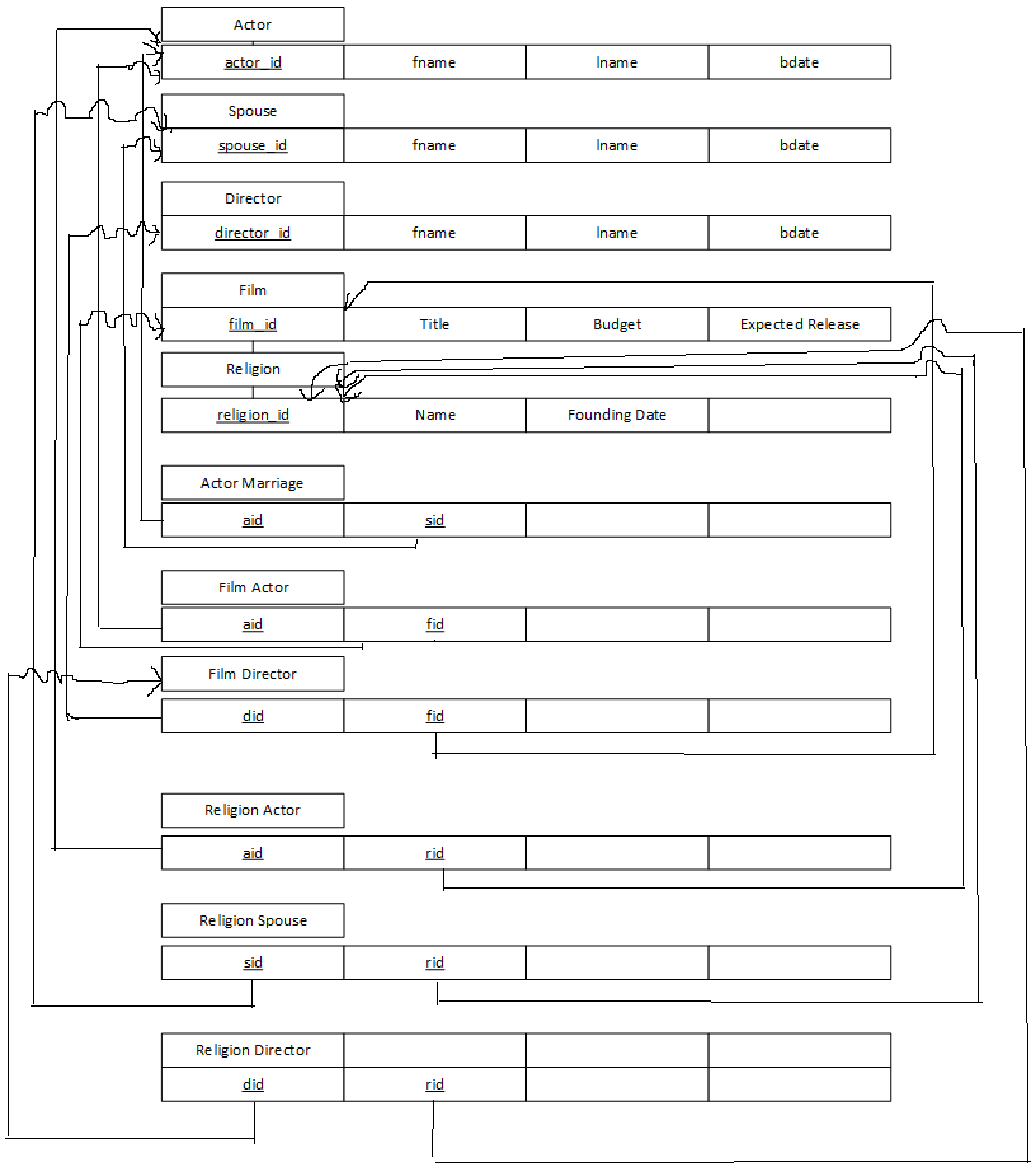
Spouses will have 1 actor that they are married to. No more no less. Spouses cannot be actors, this database tracks only non-acting spouses so there is no overlap. Spouses can have 0 or 1 religion.

Religions can have 0 or more members.

ER Diagram


actor_id
fname
lname
bdate
Actor
married to
Spouse_id
fname
lname
Spouse
bdate
1
1
N
N
Has a
Has a
1
N
1
film_id
title
Film
budget
exp_release
religion_id
religion
name
Founding_date
Works on
M
1
N
Has a
lname
fname
N
director_id
Director
bdate
1
Works on

Schema



I apologize for the crudity of this drawing, I was extremely careful but it is very detailed.

Table Creation Queries

```
CREATE TABLE actor(
	actor_id int(11) unsigned NOT NULL AUTO_INCREMENT,
	fname varchar(45) NOT NULL,
	lname varchar(45) NOT NULL,
	bdate DATE,
	PRIMARY KEY (actor_id)
)ENGINE=InnoDB

CREATE TABLE spouse(
	spouse_id int(11) unsigned NOT NULL AUTO_INCREMENT,
	fname varchar(45) NOT NULL,
	lname varchar(45) NOT NULL,
	bdate DATE,
	PRIMARY KEY (spouse_id)
)ENGINE=InnoDB

CREATE TABLE director(
	director_id int(11) unsigned NOT NULL AUTO_INCREMENT,
	fname varchar(45) NOT NULL,
	lname varchar(45) NOT NULL,
	bdate DATE,
	PRIMARY KEY (director_id)
)ENGINE=InnoDB

CREATE TABLE film(
	film_id int(11) unsigned NOT NULL AUTO_INCREMENT,
	title varchar(45) NOT NULL,
	budget numeric(13,2),
	exp_release DATE,
	PRIMARY KEY (film_id)
)ENGINE=InnoDB

CREATE TABLE religion(
	religion_id int(11) unsigned NOT NULL AUTO_INCREMENT,
	name varchar(45) NOT NULL,
	PRIMARY KEY (religion_id),
	foundingDate DATE
)ENGINE=InnoDB

CREATE TABLE actor_marriage(
	aid int(11) unsigned UNIQUE NOT NULL,
	sid int(11) unsigned UNIQUE NOT NULL,
	PRIMARY KEY (aid,sid),
	CONSTRAINT FOREIGN KEY (aid) REFERENCES actor (actor_id) ON DELETE CASCADE,
	CONSTRAINT FOREIGN KEY (sid) REFERENCES spouse (spouse_id) ON DELETE CASCADE
)ENGINE=InnoDB

CREATE TABLE film_actor(
	aid int(11) unsigned NOT NULL,
	fid int(11) unsigned NOT NULL,
	PRIMARY KEY (aid,fid),
	CONSTRAINT FOREIGN KEY (aid) REFERENCES actor (actor_id) ON DELETE CASCADE,
	CONSTRAINT FOREIGN KEY (fid) REFERENCES film (film_id) ON DELETE CASCADE
)ENGINE InnoDB

CREATE TABLE film_director(
	did int(11) unsigned  NOT NULL,
	fid int(11) unsigned UNIQUE NOT NULL,
	PRIMARY KEY (did,fid),
	CONSTRAINT FOREIGN KEY (did) REFERENCES director (director_id) ON DELETE CASCADE,
	CONSTRAINT FOREIGN KEY (fid) REFERENCES film (film_id) ON DELETE CASCADE
)ENGINE InnoDB

CREATE TABLE religion_actor(
	rid int(11) unsigned  NOT NULL,
	aid int(11) unsigned UNIQUE NOT NULL,
	PRIMARY KEY (rid,aid),
	CONSTRAINT FOREIGN KEY (rid) REFERENCES religion (religion_id) ON DELETE CASCADE,
	CONSTRAINT FOREIGN KEY (aid) REFERENCES actor (actor_id) ON DELETE CASCADE
)ENGINE InnoDB

CREATE TABLE religion_spouse(
	rid int(11) unsigned  NOT NULL,
	sid int(11) unsigned UNIQUE NOT NULL,
	PRIMARY KEY (rid,sid),
	CONSTRAINT FOREIGN KEY (rid) REFERENCES religion (religion_id) ON DELETE CASCADE,
	CONSTRAINT FOREIGN KEY (sid) REFERENCES spouse (spouse_id) ON DELETE CASCADE
)ENGINE InnoDB

CREATE TABLE religion_director(
	rid int(11) unsigned  NOT NULL,
	did int(11) unsigned UNIQUE NOT NULL,
	PRIMARY KEY (rid,did),
	CONSTRAINT FOREIGN KEY (rid) REFERENCES religion (religion_id) ON DELETE CASCADE,
	CONSTRAINT FOREIGN KEY (did) REFERENCES director (director_id) ON DELETE CASCADE
)ENGINE InnoDB
```

General Use Queries

Used many times to show what table currently contains so user can run other operations.

SELECT actor.fname, actor.lname, actor.bdate FROM actor

SELECT spouse.fname, spouse.lname, spouse.bdate FROM spouse

SELECT director.fname, director.lname, director.bdate FROM director

SELECT film.title, film.budget, film.exp_release FROM film

SELECT religion.name, religion.foundingDate FROM religion

SELECT a.fname, a.lname, f.title FROM film_actor fa INNER JOIN film f ON f.film_id = fa.fid INNER JOIN actor a ON fa.aid=a.actor_id

SELECT d.fname, d.lname, f.title FROM film_director fd INNER JOIN film f ON f.film_id = fd.fid INNER JOIN director d ON fd.did=d.director_id

SELECT a.fname, a.lname, s.fname, s.lname FROM actor_marriage am INNER JOIN actor a ON a.actor_id = am.aid INNER JOIN spouse s ON am.sid=s.spouse_id

SELECT d.fname, d.lname, r.name FROM religion_director rd INNER JOIN director d ON d.director_id = rd.did INNER JOIN religion r ON rd.rid=r.religion_id

SELECT s.fname, s.lname, r.name FROM religion_spouse rs INNER JOIN spouse s ON s.spouse_id = rs.sid INNER JOIN religion r ON rs.rid=r.religion_id

Used many times to create spouse drop down for selection

SELECT actor_id, fname, lname FROM actor

SELECT spouse_id, fname, lname FROM spouse

SELECT director_id, fname, lname FROM director

SELECT film_id, title FROM film

SELECT religion_id, name FROM religion

Queries used in Filters, adds, deletes etc.

INSERT INTO actor (fname, lname, bdate) VALUES ([firstname input],[lastname input],[birtdate input])

```sql
INSERT INTO director (fname, lname, bdate) VALUES ([firstname input],[lastname input],[birtdate input])

INSERT INTO film (title, budget, exp_release) VALUES ([title input],[budget input],[releaseDate input])

INSERT INTO spouse (fname, lname, bdate) VALUES ([firstname input],[lastname input],[birtdate input])

DELETE FROM actor WHERE actor_id[actor id input]

DELETE FROM spouse WHERE spouse_id=[spouse id input]

DELETE FROM director WHERE director_id=[director id input]

DELETE FROM film WHERE film_id=[film id input]

INSERT INTO religion (name, foundingDate) VALUES ([name input, founding date input?)

DELETE FROM religion WHERE religion_id=[religion ID input]

INSERT INTO film_actor (aid,fid) VALUES ([actor id input],[film id input])

INSERT INTO actor_marriage (aid,sid) VALUES ([actor id input], [film id input

SELECT a.fname, a.lname, s.fname, s.lname FROM actor a INNER JOIN actor_marriage am ON
a.actor_id=am.aid INNER JOIN spouse s ON s.spouse_id=am.sid WHERE a.actor_id=[actor id input]

SELECT a.fname, a.lname, f.title FROM actor a INNER JOIN film_actor fa ON a.actor_id = fa.aid INNER
JOIN film f ON fa.fid=f.film_id WHERE a.actor_id = [actor id input]

SELECT a.fname, a.lname, f.title FROM actor a INNER JOIN film_actor fa ON fa.aid=a.actor_id INNER JOIN
film f ON f.film_id=fa.fid WHERE f.film_id  = [film id input]

INSERT INTO film_director (did,fid) VALUES ([director id input], [film id input])

INSERT INTO religion_director (rid,did) VALUES ([religion id input], [director id input])

SELECT d.fname, d.lname, f.title FROM director d INNER JOIN film_director fd ON fd.did=d.director_id
INNER JOIN film f ON f.film_id=fd.fid WHERE f.film_id = [film id input]

INSERT INTO religion_actor (rid,aid) VALUES ([religion id input],[actor id input])

INSERT INTO religion_spouse (rid,sid) VALUES ([religion id input], [spouse id input])

DELETE FROM film_director WHERE did = [director id input]

DELETE FROM actor_marriage WHERE aid = [actor id input]

DELETE FROM religion_actor WHERE aid = [actor id input]
DELETE FROM religion_director WHERE did = [director id input]
```

```
DELETE FROM religion_spouse WHERE sid = [spouse id input]
```

```
SELECT  r.name, a.fname, a.lname, s.fname, s.lname, d.fname, d.lname FROM religion_actor ra
INNER JOIN religion_director rd ON ra.rid=rd.rid INNER JOIN religion_spouse rs ON rd.rid=rs.rid INNER
JOIN actor a ON ra.aid=a.actor_id INNER JOIN spouse s ON s.spouse_id=rs.sid INNER JOIN director d ON
rd.did=d.director_id INNER JOIN religion r ON r.religion_id=rd.rid WHERE r.religion_id = [religion id
input]
```

```
DELETE FROM film_actor WHERE aid=[actor id input]&&fid=[film id input]
```