Dan Blossom

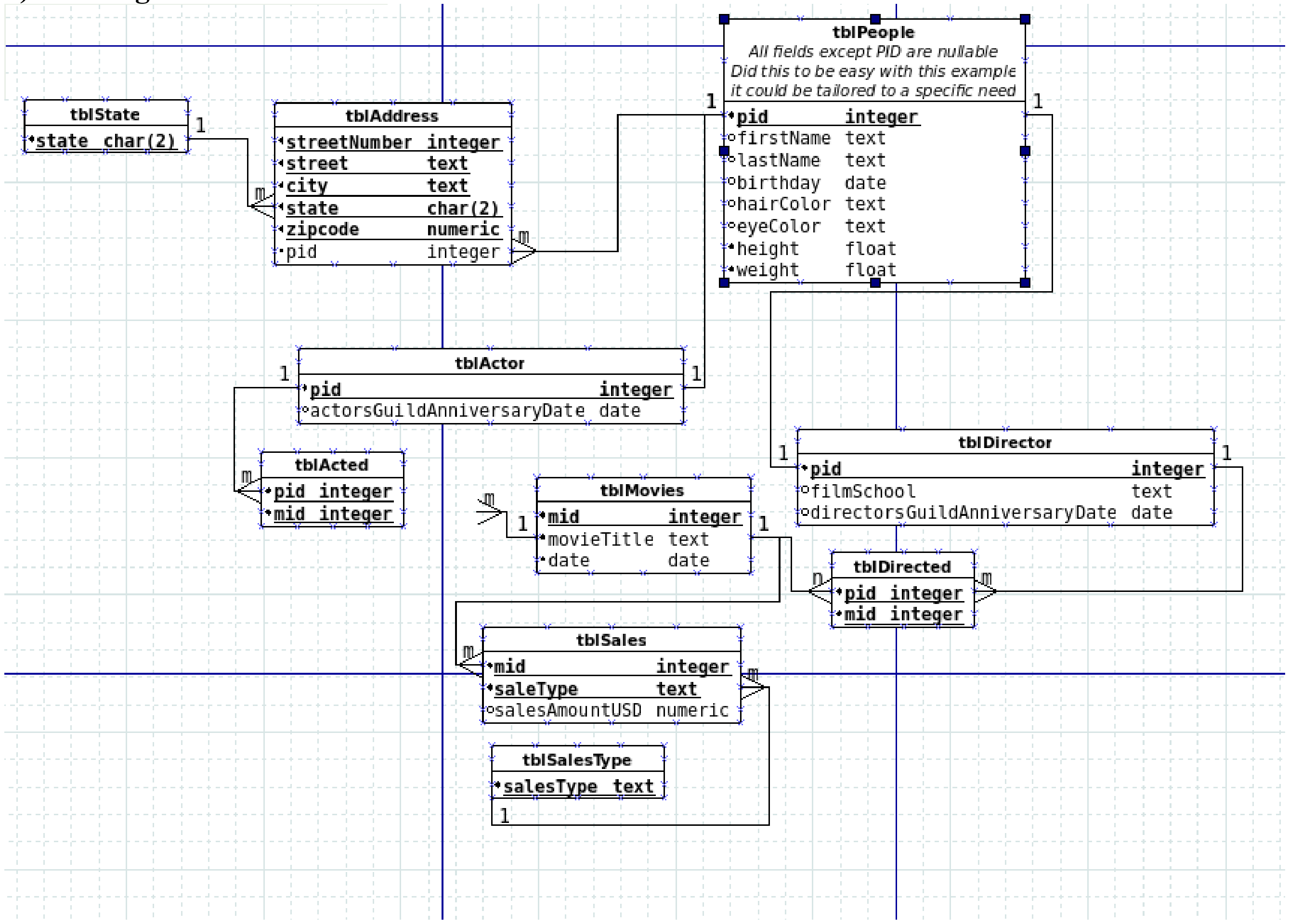
Professor Labouseur

CMPT 308 Database

November 4th, 2013

Normalization Homework 2

1) ER Diagram:



2) SQL Create Statements

-state table

```
drop table if exists states
--create statement
create table states(
  state     char(2) not null,
  primary key(state)
);
```

--address table
```
drop table if exists addresses
create table addresses(
  streetNumber  integer not null,
  street        text not null,
  city          text not null,
  state         char(2) not null references states(state),
  zipcode       numeric not null,
  pid           integer not null references people(pid),
  primary key(streetNumber, street, city, state, zipcode)
);
```

--people table
```
drop table if exists people
create table people(
  pid          integer not null,
  firstName    text,
  lastName     text,
  birthday     date,
  hairColor    text,
  eyeColor     text,
  height       float,
  weight       float,
  primary key(pid)
);
```

--actor table
```
drop table if exists actors
create table actors(
  pid                         integer not null references people(pid),
  actorsGuildAnniversaryDate  date,
  primary key(pid)
);
```

--director table
```
drop table if exists directors
create table directors(
  pid                            integer not null references people(pid),
  directorsGuildAnniversaryDate date,
  primary key(pid)
);
```
--movies table
```
drop table if exists movies
create table movies(
  mid          integer not null,
  movieTitle   text not null,
  dateReleased date not null,
  primary key(mid)
);

--acted table
drop table if exists acted
create table acted(
  pid    integer not null references actors(pid),
  mid    integer not null references movies(mid),
  primary key(pid, mid)
);

--directed table
drop table if exists directed
create table directed(
  pid integer not null references directors(pid),
  mid integer not null references movies(mid),
  primary key(pid, mid)
);

--sales type table
drop table if exists sales_types
create table sales_types(
  salesType text not null,
  primary key(salesType)
);

--sales table
drop table if exists sales
create table sales(
  mid            integer not null references movies(mid),
  salesType      text not null references sales_types(salesType),
  salesAmountUSD numeric,
  primary key(mid, salesType)
);
```

3) SQL insert statements:

```
--insert states table
insert into states(state)
      values('NY');
insert into states(state)
      values('NJ');
insert into states(state)
      values('CA');
insert into states(state)
      values('VA');
insert into states(state)
      values('PA');
insert into states(state)
      values('OH');
```

```sql
insert into states(state)
    values('FL');
insert into states(state)
    values('GA');
insert into states(state)
    values('NC');

--insert address table
insert into addresses(streetNumber, street, city, state, zipcode, pid)
    values(5, 'happy', 'poughkeepsie','NY',12602, 1);
insert into addresses(streetNumber, street, city, state, zipcode, pid)
    values(23, 'main', 'san diego','CA',90123, 2);
insert into addresses(streetNumber, street, city, state, zipcode, pid)
    values(223, 'lucky', 'jersey','NJ',09876, 3);
insert into addresses(streetNumber, street, city, state, zipcode, pid)
    values(9155, 'lorna', 'atlanta','GA',99999, 4);
insert into addresses(streetNumber, street, city, state, zipcode, pid)
    values(88, 'trexell', 'forest','VA',42420, 5);

--insert people table
insert into people(pid, firstName, lastName, birthday, hairColor, eyeColor, height, weight)
    values(1, 'sean', 'connery', '08-25-1930', 'gray', 'brown', 70, 155);
insert into people(pid, firstName, lastName, birthday, hairColor, eyeColor, height, weight)
    values(2, 'tom', 'hanks', '07-09-1956', 'black', 'brown', 74, 188);
insert into people(pid, firstName, lastName, birthday, hairColor, eyeColor, height, weight)
    values(3, 'michael', 'fox', '06-09-1961', 'black', 'blue', 66, 166);
insert into people(pid, firstName, lastName, birthday, hairColor, eyeColor, height, weight)
    values(4, 'frank', 'darabont', '01-28-1959', 'gray', 'green', 85, 220);
insert into people(pid, firstName, lastName, birthday, hairColor, eyeColor, height, weight)
    values(5, 'morgan', 'freeman', '6-1-1937', 'brown', 'brown', 72, 275);
insert into people(pid, firstName, lastName, birthday, hairColor, eyeColor, height, weight)
    values(6, 'robert', 'zemeckis', '05-14-1951', 'brown', 'blue', 72, 275);
insert into people(pid, firstName, lastName, birthday, hairColor, eyeColor, height, weight)
    values(7, 'terence', 'young', '06-20-1915', 'brown', 'blue', 65, 123);

--insert actor table
insert into actors(pid, actorsGuildAnniversaryDate)
    values(5, '03-03-1999');
insert into actors(pid, actorsGuildAnniversaryDate)
    values(2, '02-12-2001');
insert into actors(pid, actorsGuildAnniversaryDate)
    values(3, '10-21-2015');
insert into actors(pid, actorsGuildAnniversaryDate)
    values(1, '1-15-1944');

--insert director table
insert into directors(pid, directorsGuildAnniversaryDate)
    values(2, '02-11-2001');
insert into directors(pid, directorsGuildAnniversaryDate)
```

```
      values(4, '8-21-1978');
insert into directors(pid, filmSchool, directorsGuildAnniversaryDate)
      values(3, 'Marist', '10-21-2015');
insert into directors(pid, filmSchool, directorsGuildAnniversaryDate)
      values(6, 'NYU', '10-21-2015');
insert into directors(pid, filmSchool, directorsGuildAnniversaryDate)
      values(7, 'NYU', '1-21-1950');

-- insert movies table
insert into movies(mid, movieTitle, dateReleased)
      values(1,'Back to the future', '07-03-1985');
insert into movies(mid, movieTitle, dateReleased)
      values(2,'From Russia With Love', '05-27-1964');
insert into movies(mid, movieTitle, dateReleased)
      values(3,'green mile', '12-10-1999');
insert into movies(mid, movieTitle, dateReleased)
      values(4,'shawsank', '07-03-1985');

--insert acted table
insert into acted(pid, mid)
      values(3, 1);
insert into acted(pid, mid)
      values(1, 2);
insert into acted(pid, mid)
      values(2, 3);
insert into acted(pid, mid)
      values(5, 4);

--directed table
insert into directed(pid, mid)
      values(7, 2);
insert into directed(pid, mid)
      values(6, 1);
insert into directed(pid, mid)
      values(4, 4);
insert into directed(pid, mid)
      values(4, 3);

--insert sales type table
insert into sales_types(salesType)
      values('domestic box office sales');
insert into sales_types(salesType)
      values('foreign box office sales');
insert into sales_types(salesType)
      values('dvd sales');
insert into sales_types(salesType)
      values('blu-ray sales');
```

--insert sales table

```
insert into sales(mid, salesType, salesAmountUSD)
      values(1, 'domestic box office sales', 3000000);
insert into sales(mid, salesType, salesAmountUSD)
      values(1, 'foreign box office sales', 4000000);
insert into sales(mid, salesType, salesAmountUSD)
      values(1, 'dvd sales', 123456);
insert into sales(mid, salesType, salesAmountUSD)
      values(1, 'blu-ray sales', 997676722);
insert into sales(mid, salesType, salesAmountUSD)
      values(2, 'domestic box office sales', 3000000);
insert into sales(mid, salesType, salesAmountUSD)
      values(3, 'foreign box office sales', 5000000000);
insert into sales(mid, salesType, salesAmountUSD)
      values(2, 'dvd sales', 6000);
insert into sales(mid, salesType, salesAmountUSD)
      values(3, 'blu-ray sales', 564333);
insert into sales(mid, salesType, salesAmountUSD)
      values(4, 'domestic box office sales', 300000000000);
```

-- THIS IS IF YOU WANT TO SEE IF CONNERY DIRECTED HIMSELF
-- first he needs to become a director

```
insert into directors(pid, directorsGuildAnniversaryDate)
      values(1, '4-21-1940');
```

-- now he needs to become a director of a film

```
insert into directed(pid, mid)
      values(1, 2);
```

4) Functional dependencies for each table

- **State:** state →
- **Address:** (streetNumber, street, city, state, zipcode) → pid
- **People:** pid → firstName, lastName, birthday, hairColor, eyeColor, height, weight
- **Actor:** pid → actorsGuildAnniversaryDate
- **Director:** pid → filmSchool, directorsGuildAnniversaryDate
- **Movies:** mid → movieTitle, dateReleased
- **Acted:** (pid, mid) →
- **Directed:** (pid, mid) →
- **SalesType:** salesType →
- **Sales:** (mid, salesType) → salesAmountUSD

5) SQL Query to return all directors which whom actor Sean Connery has worked.
Using the test data provided above, this will return either: Terence Young or Terence Young AND Sean Connery to show that he directed himself.

```
SELECT p_directed.lastName, p_directed.firstName
FROM  movies m,
           directed d,
           people p_acted,
           people p_directed,
           acted a
WHERE m.mid = d.mid
     AND d.pid = p_directed.pid
     AND a.pid = p_acted.pid
     AND a.mid = m.mid
     AND (p_acted.lastName = 'connery'AND p_acted.firstName = 'sean');
```