**Movies**

Provided to you is a file called movies.db, a SQLite database that stores data from IMDb about movies, the people who directed and starred in them, and their ratings.

In a terminal window, run sqlite3 movies.db so that you can begin executing queries on the database.

First, when sqlite3 prompts you to provide a query, type .schema and press enter. This will output the CREATE TABLE statements that were used to generate each of the tables in the database. By examining those statements, you can identify the columns present in each table.

* Notice that the movies table has an id column that uniquely identifies each movie, as well as columns for the title of a movie and the year in which the movie was released.
* The people table also has an id column, and also has columns for each person’s name and birth year.
* Movie ratings, meanwhile, are stored in the ratings table. The first column in the table is movie\_id: a foreign key that references the id of the movies table. The rest of the row contains data about the rating for each movie and the number of votes the movie has received on IMDb.
* Finally, the stars and directors tables match people to the movies in which they acted or directed. (Only principal stars and directors are included.) Each table has just two columns: movie\_id and person\_id, which reference a specific movie and person, respectively.

The challenge ahead of you is to write SQL queries to answer a variety of different questions by selecting data from one or more of these tables.

# Testing

* To test your queries on CS50 IDE, you can query the database by running

$ cat filename.sql | sqlite3 movies.db

* where filename.sql is the file containing your SQL query.
* Or you can paste them into DB Browser for SQLite’s Execute SQL tab and click ▶.