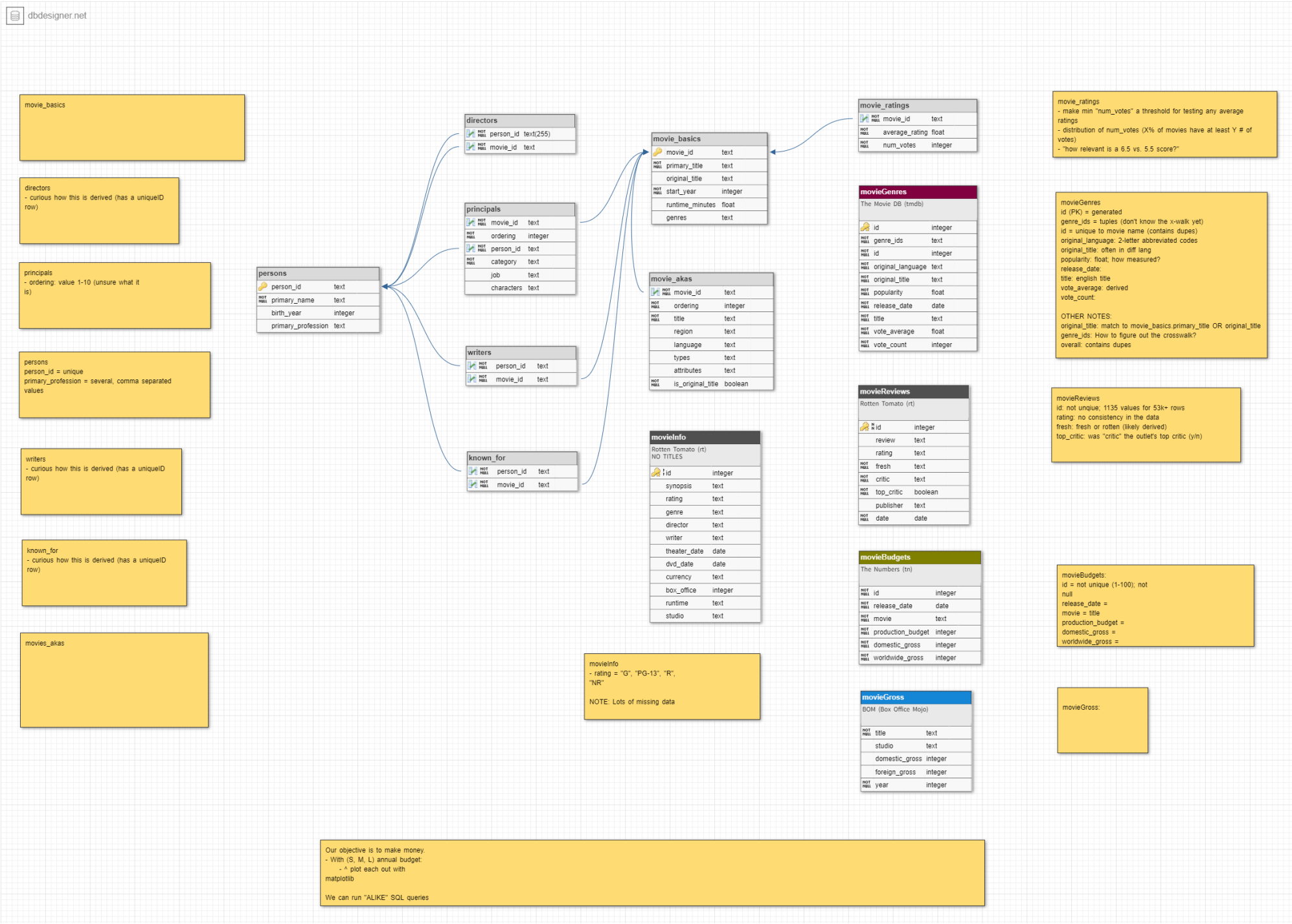
dbdesigner.net
movie_basics
directors
- curious how this is derived (has a uniqueID row)
principals
- ordering: value 1-10 (unsure what it is)
persons
person_id = unique
primary_profession = several, comma separated values
writers
- curious how this is derived (has a uniqueID row)
known_for
- curious how this is derived (has a uniqueID row)
movies_akas
directors
person_id text(255)
movie_id text
principals
movie_id text
ordering integer
person_id text
category text
job text
characters text
persons
person_id text
primary_name text
birth_year integer
primary_profession text
writers
person_id text
movie_id text
known_for
person_id text
movie_id text
movie_basics
movie_id text
primary_title text
original_title text
start_year integer
runtime_minutes float
genres text
movie_akas
movie_id text
ordering integer
title text
region text
language text
types text
attributes text
is_original_title boolean
movieInfo
Rotten Tomato (rt)
NO TITLES
id integer
synopsis text
rating text
genre text
director text
writer text
theater_date date
dvd_date date
currency text
box_office integer
runtime text
studio text
movieInfo
- rating = "G", "PG-13", "R", "NR"
NOTE: Lots of missing data
movie_ratings
movie_id text
average_rating float
num_votes integer
movie_ratings
- make min "num_votes" a threshold for testing any average ratings
- distribution of num_votes (X% of movies have at least Y # of votes)
- "how relevant is a 6.5 vs. 5.5 score?"
movieGenres
The Movie DB (tmdb)
id integer
genre_ids text
id integer
original_language text
original_title text
popularity float
release_date date
title text
vote_average float
vote_count integer
movieGenres
id (PK) = generated
genre_ids = tuples (don't know the x-walk yet)
id = unique to movie name (contains dupes)
original_language: 2-letter abbreviated codes
original_title: often in diff lang
popularity: float; how measured?
release_date:
title: english title
vote_average: derived
vote_count:
OTHER NOTES:
original_title: match to movie_basics.primary_title OR original_title
genre_ids: How to figure out the crosswalk?
overall: contains dupes
movieReviews
Rotten Tomato (rt)
id integer
review text
rating text
fresh text
critic text
top_critic boolean
publisher text
date date
movieReviews
id: not unqiue; 1135 values for 53k+ rows
rating: no consistency in the data
fresh: fresh or rotten (likely derived)
top_critic: was "critic" the outlet's top critic (y/n)
movieBudgets
The Numbers (tn)
id integer
release_date date
movie text
production_budget integer
domestic_gross integer
worldwide_gross integer
movieBudgets:
id = not unique (1-100); not null
release_date =
movie = title
production_budget =
domestic_gross =
worldwide_gross =
movieGross
BOM (Box Office Mojo)
title text
studio text
domestic_gross integer
foreign_gross integer
year integer
movieGross:
Our objective is to make money.
- With (S, M, L) annual budget:
- ^ plot each out with matplotlib
We can run "ALIKE" SQL queries