**Application Description**

1. **Introduction:**

The IMDb (Internet Movie Database) management system is a sophisticated and comprehensive application that serves as a centralized platform for storing, organizing, and managing a vast collection of movie-related information. It ensures data accuracy and completeness through validation techniques and user contributions, while providing users with a user-friendly interface, advanced search capabilities, and analytical tools. This powerful system revolutionizes the exploration and understanding of movies, allowing users to effortlessly navigate, search, and analyze movie data, while fostering a vibrant community for engagement and discussion among movie enthusiasts.

1. **All the minimum required entities; hierarchies, is-a, contains, related-to and appropriate use of surrogate keys:**

* Movie: Represents movies and their attributes like movie ID, start year, average rating, and movie name.

Release: Represents movie releases in different regions and languages, linked to the Movie entity.

Director: Represents directors and their attributes such as director ID, name, and birth year.

Actor: Represents actors and their attributes like actor ID, name, and birth year.

Person: Represents people associated with movies, including both directors and actors, with attributes like person ID and name.

Genres: Represents genres of movies, with attributes like genre ID and name.

Movie\_has\_Person: Represents the association between movies and people, capturing their involvement in different roles.

Movie\_has\_Genres: Represents the association between movies and genres, indicating the genres to which a movie belongs.

* Is-a relationship: Table Person is the parent of table Director and table Actor. The "is-a" relationship allows Director and Actor to be treated as specialized types of persons for different movies, inheriting and extending the properties of the Person table.
* Contain relationship: The "contain" relationship exists between the Movie table and the Person table. Each movie can contain multiple people, and a person can be associated with multiple movies. The Movie\_has\_Person table acts as a junction table that establishes the relationship between Movie and Person.
* Related-to relationships: The schema includes various related-to relationships, such as:

Movie related to Release: Captures the relationship between a movie and its releases in different regions and languages.

Person related to Movie\_has\_Person: Represents the involvement of a person in a movie in different roles.

Movie related to Movie\_has\_Genres: Indicates the genres to which a movie belongs.

* Appropriate use of surrogate keys: The schema employs surrogate keys (integer-based primary keys) for each entity, such as movie\_id, release\_id, director\_id, actor\_id, person\_id, genres\_id, etc. These surrogate keys ensure uniqueness and facilitate efficient data retrieval and relationships between entities.

1. **Relationships match business rules with correct cardinality:**

* Release and movie relationship: A movie can have multiple releases (1 or more) in different regions and languages.
* Person and director/actor relationship: A person can be associated with either a director or an actor role.
* Movie and person relationship: A movie can have multiple people (0 or more) associated with it, such as directors or actors, and a person (0 or more) can be associated with multiple movies.
* Movie and genres relationship: A movie can belong to multiple genres (1 or more), and a genre (1 or more) can be associated with multiple movies.

1. **Physical model (from SQL server diagram and My SQL):**

A screenshot of a computer

Description automatically generated with medium confidence