Table creation schema:

IMDB_MOVIES:

```
CREATE TABLE IF NOT EXISTS public.imdb_movies
(
    imdb_title_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),
    title character varying COLLATE pg_catalog."default" NOT NULL,
    year integer,
    genre character varying COLLATE pg_catalog."default",
    duration integer,
    country character varying COLLATE pg_catalog."default",
    language character varying COLLATE pg_catalog."default",
    description text COLLATE pg_catalog."default",
    production_house character varying COLLATE pg_catalog."default",
    budget character varying COLLATE pg_catalog."default",
    director character varying COLLATE pg_catalog."default",
    writer character varying COLLATE pg_catalog."default",
    actor character varying COLLATE pg_catalog."default",
    rating double precision,
    total_votes integer,
    CONSTRAINT imdb_movies_pkey PRIMARY KEY (imdb_title_id),
    CONSTRAINT imdb_title_id UNIQUE (imdb_title_id)
        INCLUDE(imdb_title_id),
    CONSTRAINT title UNIQUE (title, imdb_title_id),
    CONSTRAINT actor FOREIGN KEY (actor)
        REFERENCES public.person_details (primary_name) MATCH SIMPLE
        ON UPDATE RESTRICT
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT director FOREIGN KEY (director)
        REFERENCES public.person_details (primary_name) MATCH SIMPLE
```

```
    ON UPDATE RESTRICT
    ON DELETE RESTRICT
    NOT VALID,
  CONSTRAINT writer FOREIGN KEY (writer)
    REFERENCES public.person_details (primary_name) MATCH SIMPLE
    ON UPDATE RESTRICT
    ON DELETE RESTRICT
    NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.imdb_movies
  OWNER to postgres;
```

IMDB_TMDB_CONNECTOR:

```
CREATE TABLE IF NOT EXISTS public.imdb_tmdb_connector
(
  movie_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),
  imdb_id integer,
  tmbd_id integer,
  CONSTRAINT link_pkey PRIMARY KEY (movie_id),
  CONSTRAINT "imdbId" FOREIGN KEY (imdb_id)
    REFERENCES public.imdb_movies (imdb_title_id) MATCH SIMPLE
    ON UPDATE RESTRICT
    ON DELETE RESTRICT
    NOT VALID
)
```

```
TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.imdb_tmdb_connector
    OWNER to postgres;
```

LISTED_IN:

```
CREATE TABLE IF NOT EXISTS public.listed_in
(
    imdb_title_id integer NOT NULL,
    show_id integer NOT NULL,
    listed_in_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),
    CONSTRAINT listed_in_id PRIMARY KEY (listed_in_id)
        INCLUDE(listed_in_id),
    CONSTRAINT imdb_title_id FOREIGN KEY (imdb_title_id)
        REFERENCES public.imdb_movies (imdb_title_id) MATCH SIMPLE
        ON UPDATE RESTRICT
        ON DELETE RESTRICT,
    CONSTRAINT show_id FOREIGN KEY (show_id)
        REFERENCES public.netflix (show_id) MATCH SIMPLE
        ON UPDATE RESTRICT
        ON DELETE RESTRICT
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.listed_in
    OWNER to postgres;
```

MOVIE_TAG_SCORES:

```
CREATE TABLE IF NOT EXISTS public.movie_tag_scores

(

    tag_score_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1
MAXVALUE 2147483647 CACHE 1 ),

    movie_id integer,

    tag_id integer,

    tag_score double precision,

    CONSTRAINT genome_scores_pkey PRIMARY KEY (tag_score_id),

    CONSTRAINT "movieId" FOREIGN KEY (movie_id)

        REFERENCES public.imdb_tmdb_connector (movie_id) MATCH SIMPLE

        ON UPDATE RESTRICT

        ON DELETE RESTRICT,

    CONSTRAINT tag_id FOREIGN KEY (tag_id)

        REFERENCES public.tag (tag_id) MATCH SIMPLE

        ON UPDATE RESTRICT

        ON DELETE RESTRICT

)


TABLESPACE pg_default;


ALTER TABLE IF EXISTS public.movie_tag_scores

    OWNER to postgres;
```

NETFLIX:

```
CREATE TABLE IF NOT EXISTS public.netflix

(

    show_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE
2147483647 CACHE 1 ),

    type character varying COLLATE pg_catalog."default",

    date_added date,
```

```
    sensor_rating character varying COLLATE pg_catalog."default",
    CONSTRAINT netflix_pkey PRIMARY KEY (show_id)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.netflix
    OWNER to postgres;
```

OSCAR_AWARDS:

```
CREATE TABLE IF NOT EXISTS public.oscar_awards
(
    ceremony_number integer,
    category character varying COLLATE pg_catalog."default",
    gender "char",
    name character varying COLLATE pg_catalog."default",
    winner boolean,
    oscar_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),
    imdb_title_id integer NOT NULL,
    CONSTRAINT oscar_awards_pkey PRIMARY KEY (oscar_id),
    CONSTRAINT imdb_title_id FOREIGN KEY (imdb_title_id)
        REFERENCES public.imdb_movies (imdb_title_id) MATCH SIMPLE
        ON UPDATE RESTRICT
        ON DELETE RESTRICT
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.oscar_awards
```

OWNER to postgres;

PERSON_DETAILS:

```
CREATE TABLE IF NOT EXISTS public.person_details
(
    person_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),
    primary_name character varying COLLATE pg_catalog."default",
    birth_year integer,
    death_year integer,
    primary_profession character varying COLLATE pg_catalog."default",
    CONSTRAINT person_details_pkey PRIMARY KEY (person_id),
    CONSTRAINT person_id UNIQUE (person_id)
        INCLUDE(person_id),
    CONSTRAINT primary_name UNIQUE (primary_name)
        INCLUDE(primary_name)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.person_details
    OWNER to postgres;
```

REVIEWS:

```
CREATE TABLE IF NOT EXISTS public.reviews
(
    imdb_title_id integer,
    review_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),
```

```
    reviewer_name character varying COLLATE pg_catalog."default",
    review_date date,
    review text COLLATE pg_catalog."default",
    CONSTRAINT reviews_pkey PRIMARY KEY (review_id),
    CONSTRAINT imdb_title_id FOREIGN KEY (imdb_title_id)
        REFERENCES public.imdb_movies (imdb_title_id) MATCH SIMPLE
        ON UPDATE RESTRICT
        ON DELETE RESTRICT
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.reviews
    OWNER to postgres;
```

TAG:

```
CREATE TABLE IF NOT EXISTS public.tag
(
    tag_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),
    tag character varying(100) COLLATE pg_catalog."default",
    CONSTRAINT genome_tags_pkey PRIMARY KEY (tag_id)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.tag
    OWNER to postgres;
```

USER_TAG:

```
CREATE TABLE IF NOT EXISTS public.user_tag
```

```
(
    user_tag_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1
MAXVALUE 2147483647 CACHE 1 ),
    user_id integer,
    movie_id integer,
    tag character varying COLLATE pg_catalog."default",
    CONSTRAINT tag_pkey PRIMARY KEY (user_tag_id),
    CONSTRAINT movie_id FOREIGN KEY (movie_id)
        REFERENCES public.imdb_tmdb_connector (movie_id) MATCH SIMPLE
        ON UPDATE RESTRICT
        ON DELETE RESTRICT
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.user_tag
    OWNER to postgres;
```