Script SQL

Connexion au serveur :

```
psql "host=c-groupe2.jh4mqc5jxykkfg.postgres.cosmos.azure.com port=5432
dbname=netfloox user=citus password=floox94! sslmode=require"
```

Ajout de schémas :

```
DROP SCHEMA IF EXISTS test1 CASCADE;
CREATE SCHEMA test1;
SET search_path to test1;
-- DROP SCHEMA IF EXISTS test2 CASCADE;
-- CREATE SCHEMA test2;
-- SET search_path to test2;
-- DROP SCHEMA IF EXISTS test3 CASCADE;
-- CREATE SCHEMA test3;
-- SET search_path to test3;
```

Création des tables :

```
DROP TABLE IF EXISTS name_basics;
CREATE TABLE "name_basics"
(
  "nconst" text,
  "primaryName" text,
  "birthYear" text,
  "deathYear" text,
  "primaryProfession" text,
  "knownForTitles" text
);

DROP TABLE IF EXISTS title_basics;
CREATE TABLE title_basics
(
  "tconst" text NOT NULL,
  "titleType" text,
  "primaryTitle" text,
  "originalTitle" text,
  "isAdult" text,
  "startYear" text,
  "endYear" text,
  "runtimeMinutes" text,
  "genres" text
);

DROP TABLE IF EXISTS title_principals;
```

```
CREATE TABLE title_principals
(
  "tconst" text NOT NULL,
  "ordering" int2 NOT NULL,
  "nconst" text,
  "category" text,
  "job" text,
  "characters" text
);

DROP TABLE IF EXISTS title_crew;
CREATE TABLE title_crew
(
  "tconst" text,
  "directors" text,
  "writers" text
);

DROP TABLE IF EXISTS title_akas;
CREATE TABLE title_akas
(
  "titleId" text,
  "ordering" int,
  "title" text,
  "region" text,
  "language" text,
  "types" text,
  "attributes" text,
  "isOriginalTitle" text
);

DROP TABLE IF EXISTS title_episode;
CREATE TABLE title_episode
(
  "tconst" text,
  "parentTconst" text,
  "seasonNumber" text,
  "episodeNumber" text
);

DROP TABLE IF EXISTS title_ratings;
CREATE TABLE title_ratings
(
  "tconst" text,
  "averageRating" text,
  "numVotes" text
);
```

Importation des données :

```
\COPY name_basics FROM PROGRAM 'curl -s https://datasets.imdbws.com/
name.basics.tsv.gz | zcat | head -50000' with (format csv, delimiter E'\t', header
TRUE, quote E'\b');

\COPY title_basics FROM PROGRAM 'curl -s https://datasets.imdbws.com/
title.basics.tsv.gz | zcat | head -50000' with (format csv, delimiter E'\t', header
TRUE, quote E'\b');

\COPY title_principals FROM PROGRAM 'curl -s https://datasets.imdbws.com/
title.principals.tsv.gz | zcat | head -50000' with (format csv, delimiter E'\t',
header TRUE, quote E'\b');

\COPY title_crew FROM PROGRAM 'curl -s https://datasets.imdbws.com/
title.crew.tsv.gz | zcat | head -50000' with (format csv, delimiter E'\t', header
TRUE, quote E'\b');

\COPY title_akas FROM PROGRAM 'curl -s https://datasets.imdbws.com/
title.akas.tsv.gz | zcat |  with (format csv, delimiter E'\t', header TRUE, quote
E’\b’, NULL '\N');

\COPY title_episode FROM PROGRAM 'curl -s https://datasets.imdbws.com/
title.episode.tsv.gz | zcat | head -50000' with (format csv, delimiter E'\t', header
TRUE, quote E'\b');

\COPY name_basics FROM 'title_akas.tsv' WITH (FORMAT csv, DELIMITER E'\t',
HEADER TRUE, NULL '\N', QUOTE E'\b', ENCODING 'UTF8');
```

- \COPY title_akas FROM PROGRAM 'curl -s https://datasets.imdbws.com/title.akas.tsv.gz | zcat with (format csv, delimiter E'\t', Nheader TRUE, quote E'\b');

Supprimer les lignes pas présentes : **DELETE FROM** principal.title_akas **WHERE** "titleId" **NOT IN** (**SELECT** "tconst" **FROM** principal.title_basics)

```
CREATE TABLE principal.name_basics (
    nconst varchar(11) NOT NULL,
    "primaryName" text NULL,
    "birthYear" int2 NULL,
    "deathYear" int2 NULL,
    "primaryProfession" text NULL,
```

```sql
	"knownForTitles" text NULL,
	CONSTRAINT name_basics_pk PRIMARY KEY (nconst)
);
CREATE INDEX name_basics_primaryname_idx ON principal.name_basics
USING btree ("primaryName");


CREATE TABLE principal.title_akas (
	"titleId" varchar(11) NOT NULL,
	"ordering" int4 NOT NULL,
	title text NULL,
	region text NULL,
	"language" text NULL,
	"types" text NULL,
	"attributes" text NULL,
	"isOriginalTitle" int4 NULL,
	CONSTRAINT title_akas_pk PRIMARY KEY ("titleId", ordering)
);

CREATE TABLE principal.title_basics (
	tconst varchar(11) NOT NULL,
	"titleType" varchar(11) NULL,
	"primaryTitle" text NULL,
	"originalTitle" text NULL,
	"isAdult" int2 NULL,
	"startYear" int2 NULL,
	"endYear" int2 NULL,
	"runtimeMinutes" int4 NULL,
	genres text NULL,
	CONSTRAINT title_basics_pk PRIMARY KEY (tconst)
);
CREATE INDEX title_basics_genres_idx ON principal.title_basics USING btree
(genres);
CREATE INDEX title_basics_originaltitle_idx ON principal.title_basics USING
btree ("originalTitle");
CREATE INDEX title_basics_primarytitle_idx ON principal.title_basics USING
btree ("primaryTitle");
CREATE INDEX title_basics_runtimeminutes_idx ON principal.title_basics
USING btree ("runtimeMinutes");
CREATE INDEX title_basics_tconst_idx ON principal.title_basics USING btree
(tconst);


CREATE TABLE principal.title_crew (
	tconst varchar(11) NOT NULL,
	directors text NULL,
	writers text NULL,
	CONSTRAINT title_crew_pk PRIMARY KEY (tconst)
);
CREATE INDEX title_crew_tconst_idx ON principal.title_crew USING btree
(tconst);

CREATE TABLE principal.title_episode (
	tconst varchar(11) NOT NULL,
	"parentTconst" varchar(11) NOT NULL,
	"seasonNumber" int4 NULL,
	"episodeNumber" int4 NULL,
	CONSTRAINT title_episode_pk PRIMARY KEY (tconst, "parentTconst")
);
CREATE INDEX title_episode_parenttconst_idx ON principal.title_episode
USING btree ("parentTconst");
CREATE INDEX title_episode_tconst_idx ON principal.title_episode USING
btree (tconst);

CREATE TABLE principal.title_principals (
	tconst varchar(11) NOT NULL,
	"ordering" int2 NOT NULL,
	nconst varchar(11) NULL,
	category text NULL,
	job text NULL,
	"characters" text NULL,
	CONSTRAINT title_principals_pk PRIMARY KEY (tconst, ordering)
);
CREATE INDEX title_principals_nconst_idx ON principal.title_principals USING
btree (nconst);
CREATE INDEX title_principals_tconst_idx ON principal.title_principals USING
btree (tconst);

CREATE TABLE principal.title_ratings (
	tconst varchar(11) NOT NULL,
	"averageRating" float8 NULL,
	"numVotes" int4 NULL,
	CONSTRAINT title_ratings_pk PRIMARY KEY (tconst),
	CONSTRAINT title_ratings_title_basics_fk FOREIGN KEY (tconst)
REFERENCES principal.title_basics(tconst)
);



-- principal.filmview source

CREATE MATERIALIZED VIEW principal.filmview
TABLESPACE pg_default
AS SELECT tb.tconst,
    tb."primaryTitle",
    tb."titleType",
    tb."isAdult",
    tb."startYear",
    tb."endYear",
    tb."runtimeMinutes",
    tb.genres,
    rt."averageRating",
    rt."numVotes",
    array_agg((tp.category || '_'::text) || replace(nb."primaryName", ' '::text,
'_'::text)) AS "Cate&names"
   FROM principal.title_basics tb
     JOIN principal.title_ratings rt ON tb.tconst::text = rt.tconst::text
     JOIN principal.title_principals tp ON tb.tconst::text = tp.tconst::text
     JOIN principal.name_basics nb ON tp.nconst::text = nb.nconst::text
  GROUP BY tb.tconst, rt."averageRating", rt."numVotes"
WITH DATA;

-- principal.castview source

CREATE MATERIALIZED VIEW principal.castview
TABLESPACE pg_default
AS SELECT tb.tconst,
    tb."primaryTitle",
    tb."titleType",
    tb."isAdult",
    tb."startYear",
    tb."endYear",
    tb."runtimeMinutes",
    tb.genres,
    avg(rt."averageRating") AS "averageRating",
    avg(rt."numVotes") AS "numVotes",
    string_agg(DISTINCT cw.directors, ','::text) AS directors,
    string_agg(DISTINCT cw.writers, ','::text) AS writers,
    string_agg(DISTINCT replace(nbt."primaryName", ' '::text, '_'::text),
','::text) FILTER (WHERE tp.category ~~ 'act%'::text) AS actor,
    string_agg(DISTINCT replace(nbt."primaryName", ' '::text, '_'::text),
','::text) FILTER (WHERE tp.category = 'director'::text) AS director,
    string_agg(DISTINCT replace(nbt."primaryName", ' '::text, '_'::text),
','::text) FILTER (WHERE tp.category = 'writer'::text) AS writer,
    string_agg(DISTINCT replace(nbt."primaryName", ' '::text, '_'::text),
','::text) FILTER (WHERE tp.category = 'producer'::text) AS producer,
    string_agg(DISTINCT replace(nbt."primaryName", ' '::text, '_'::text),
','::text) FILTER (WHERE tp.category = 'cinematographer'::text) AS
cinematographer,
    string_agg(DISTINCT replace(nbt."primaryName", ' '::text, '_'::text),
','::text) FILTER (WHERE tp.category = 'composer'::text) AS composer,
   string_agg(DISTINCT replace(nbt."primaryName", ' '::text, '_'::text),
','::text) FILTER (WHERE tp.category = 'editor'::text) AS editor,
   string_agg(DISTINCT replace(nbt."primaryName", ' '::text, '_'::text),
','::text) FILTER (WHERE tp.category = 'production_designer'::text) AS
production_designer,
   string_agg(DISTINCT replace(nbt."primaryName", ' '::text, '_'::text),
','::text) FILTER (WHERE tp.category = 'self'::text) AS self,
   string_agg(DISTINCT replace(nbt."primaryName", ' '::text, '_'::text),
','::text) FILTER (WHERE tp.category = 'archive_footage'::text) AS
archive_footage,
   string_agg(DISTINCT replace(nbt."primaryName", ' '::text, '_'::text),
','::text) FILTER (WHERE tp.category = 'archive_sound'::text) AS
archive_sound
  FROM principal.title_basics tb
    LEFT JOIN principal.title_ratings rt ON tb.tconst::text = rt.tconst::text
    LEFT JOIN principal.crewview cw ON tb.tconst::text = cw.tconst::text
    LEFT JOIN principal.title_principals tp ON tb.tconst::text = tp.tconst::text
    LEFT JOIN principal.name_basics nbt ON tp.nconst::text = nbt.nconst::text
 GROUP BY tb.tconst
WITH DATA;

-- View indexes:
CREATE INDEX castview_tconst_idx ON principal.castview USING btree
(tconst);

CREATE MATERIALIZED VIEW principal.crewview
TABLESPACE pg_default
AS SELECT tb.tconst,
   string_agg(DISTINCT replace(nc_directors."primaryName", ' '::text,
'_'::text), ','::text) AS directors,
   string_agg(DISTINCT replace(nc_writers."primaryName", ' '::text,
'_'::text), ','::text) AS writers
  FROM principal.title_basics tb
    JOIN principal.title_crew tc ON tb.tconst::text = tc.tconst::text
    LEFT JOIN principal.name_basics nc_directors ON
nc_directors.nconst::text = ANY (string_to_array(tc.directors, ','::text))
    LEFT JOIN principal.name_basics nc_writers ON nc_writers.nconst::text =
ANY (string_to_array(tc.writers, ','::text))
 GROUP BY tb.tconst
WITH DATA;
```