CSCI.620.01/03   Homework5   Suraj Sureshkumar

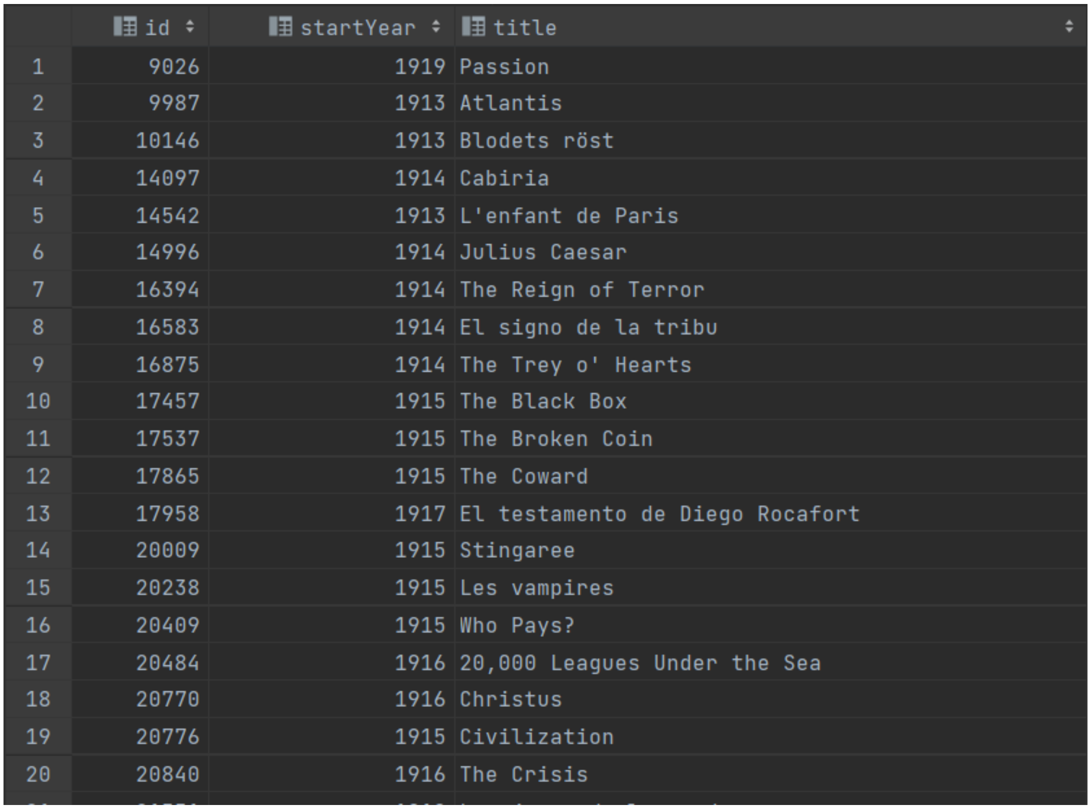
Task 1:
Non-materialized

<u>ComedyMovie</u>:

Created a view and selected the required fields where genre is Comedy and the runtime is greater than 75 and joined the id of movie and the genre and its respective id.

```
CREATE VIEW ComedyMovie AS
   SELECT t."id", t."startYear", t.title
   from hw_schema."Genre" g, hw_schema."Title" t, hw_schema."Title_Genre" tg
WHERE g.genre = 'Comedy'
  and t.runtime > 75
  and t.id = tg.title
  and tg.genre = g.id;
```

<u>Output</u>:



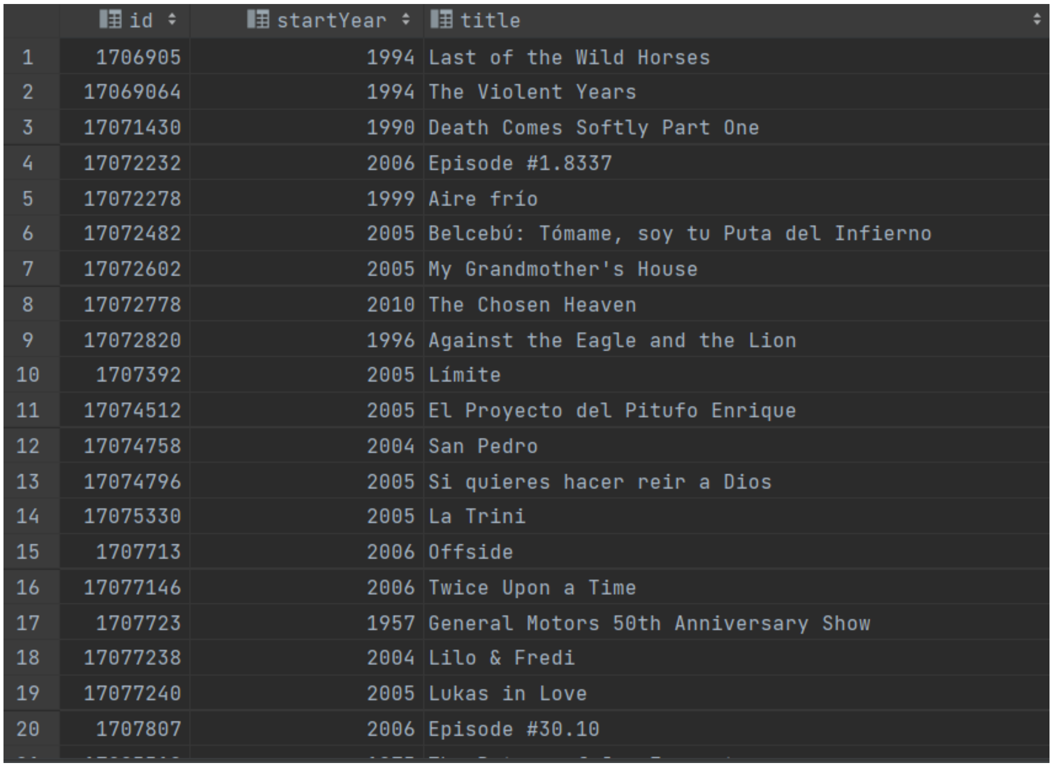
	id	startYear	title
1	9026	1919	Passion
2	9987	1913	Atlantis
3	10146	1913	Blodets röst
4	14097	1914	Cabiria
5	14542	1913	L'enfant de Paris
6	14996	1914	Julius Caesar
7	16394	1914	The Reign of Terror
8	16583	1914	El signo de la tribu
9	16875	1914	The Trey o' Hearts
10	17457	1915	The Black Box
11	17537	1915	The Broken Coin
12	17865	1915	The Coward
13	17958	1917	El testamento de Diego Rocafort
14	20009	1915	Stingaree
15	20238	1915	Les vampires
16	20409	1915	Who Pays?
17	20484	1916	20,000 Leagues Under the Sea
18	20770	1916	Christus
19	20776	1915	Civilization
20	20840	1916	The Crisis

<u>NonComedyMovie</u>:

Created a view and selected the required fields where genre is not Comedy and the runtime is greater than 75 and joined on the title and see where genre= 3 which is the comedy id.

```
CREATE VIEW NonComedyMovie AS
   SELECT t."id", t."startYear", t.title
   from hw_schema."Title" t
WHERE
  t.runtime > 75
 and t.id not in(select t.id from hw_schema."Title" t, hw_schema."Title_Genre" tg WHERE
tg.genre = 3 and t.id = tg.title);
```

<u>Output</u>:



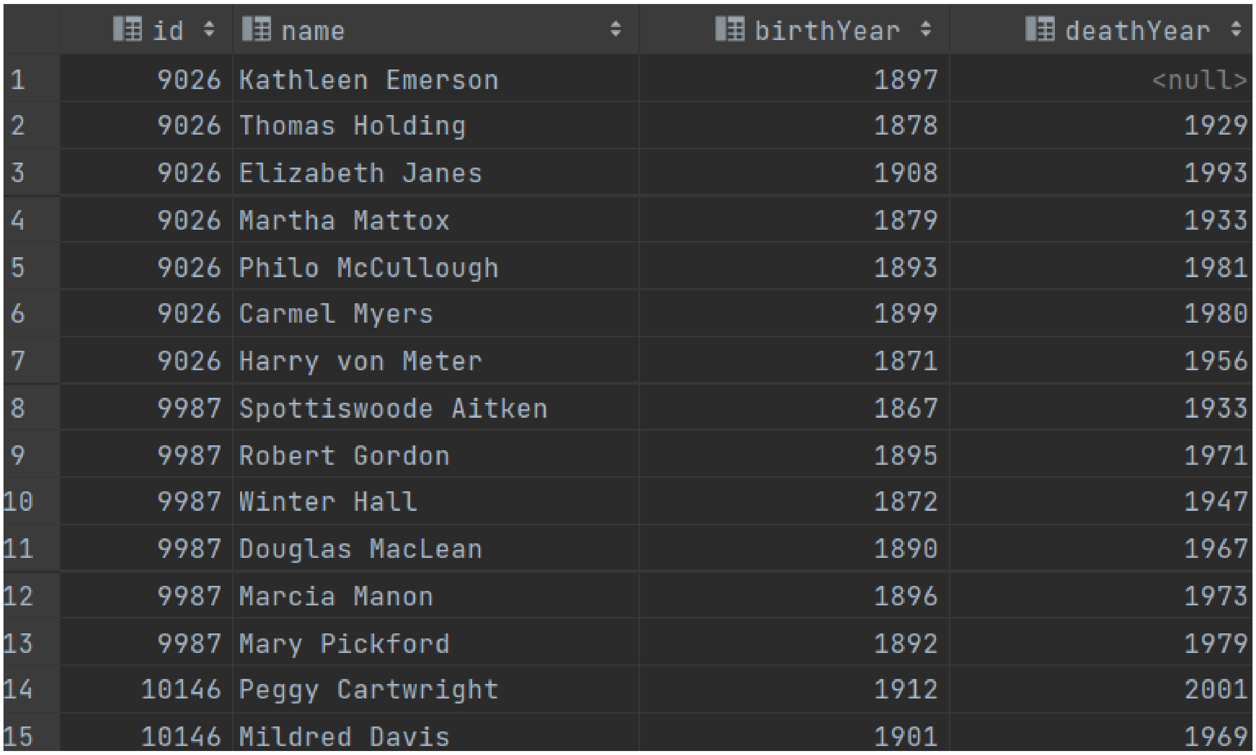
	id	startYear	title
1	1706905	1994	Last of the Wild Horses
2	17069064	1994	The Violent Years
3	17071430	1990	Death Comes Softly Part One
4	17072232	2006	Episode #1.8337
5	17072278	1999	Aire frío
6	17072482	2005	Belcebú: Tómame, soy tu Puta del Infierno
7	17072602	2005	My Grandmother's House
8	17072778	2010	The Chosen Heaven
9	17072820	1996	Against the Eagle and the Lion
10	1707392	2005	Límite
11	17074512	2005	El Proyecto del Pitufo Enrique
12	17074758	2004	San Pedro
13	17074796	2005	Si quieres hacer reir a Dios
14	17075330	2005	La Trini
15	1707713	2006	Offside
16	17077146	2006	Twice Upon a Time
17	1707723	1957	General Motors 50th Anniversary Show
18	17077238	2004	Lilo & Fredi
19	17077240	2005	Lukas in Love
20	1707807	2006	Episode #30.10

ComedyActor:

Created view and selected the required fields from the respective tables where genre is Comedy and the runtime is greater than 75 minutes and joined on the actor id and movieid and the genre and its respective id.

```
CREATE VIEW ComedyActor AS
    SELECT t."id",m.name,m."birthYear",m."deathYear"
    from hw_schema."Title" t, hw_schema."Member" m, hw_schema."Genre" g,
                                                       hw_schema."Title_Genre" tg,hw_schema."Title_Actor"
ta
WHERE g.genre = 'Comedy'
  and t.runtime > 75
  and t.id = tg.title
  and tg.genre = g.id
  and ta.actor = m.id
  and t.id = ta.title;
```

Output:



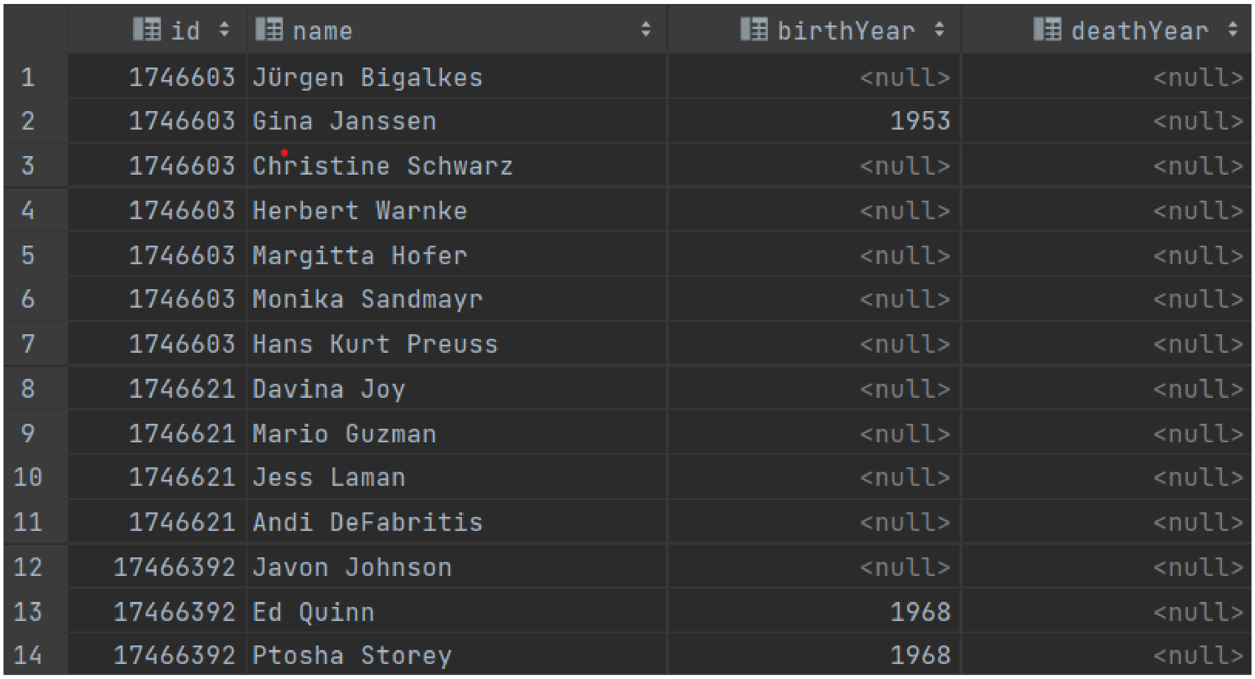
	id	name	birthYear	deathYear
1	9026	Kathleen Emerson	1897	<null>
2	9026	Thomas Holding	1878	1929
3	9026	Elizabeth Janes	1908	1993
4	9026	Martha Mattox	1879	1933
5	9026	Philo McCullough	1893	1981
6	9026	Carmel Myers	1899	1980
7	9026	Harry von Meter	1871	1956
8	9987	Spottiswoode Aitken	1867	1933
9	9987	Robert Gordon	1895	1971
10	9987	Winter Hall	1872	1947
11	9987	Douglas MacLean	1890	1967
12	9987	Marcia Manon	1896	1973
13	9987	Mary Pickford	1892	1979
14	10146	Peggy Cartwright	1912	2001
15	10146	Mildred Davis	1901	1969

<u>NonComedyActor</u>:

Created view and selected the required fields from the respective tables where genre is not Comedy and the runtime is greater than 75 minutes and joined on the actor id and movieid and checked with not in where the genre = 3 which is Comedy genre id.

```
CREATE VIEW NonComedyActor AS
   SELECT t."id",m.name,m."birthYear",m."deathYear"
   from hw_schema."Title" t, hw_schema."Member" m, hw_schema."Title_Actor" ta
WHERE t.runtime > 75
 and ta.actor = m.id
 and ta.title = t.id
 and t.id not in (SELECT t.id from hw_schema."Title" t , hw_schema."Title_Genre" tg
                        WHERE tg.genre = 3
                        and t.id = tg.title);
```

<u>Output</u>

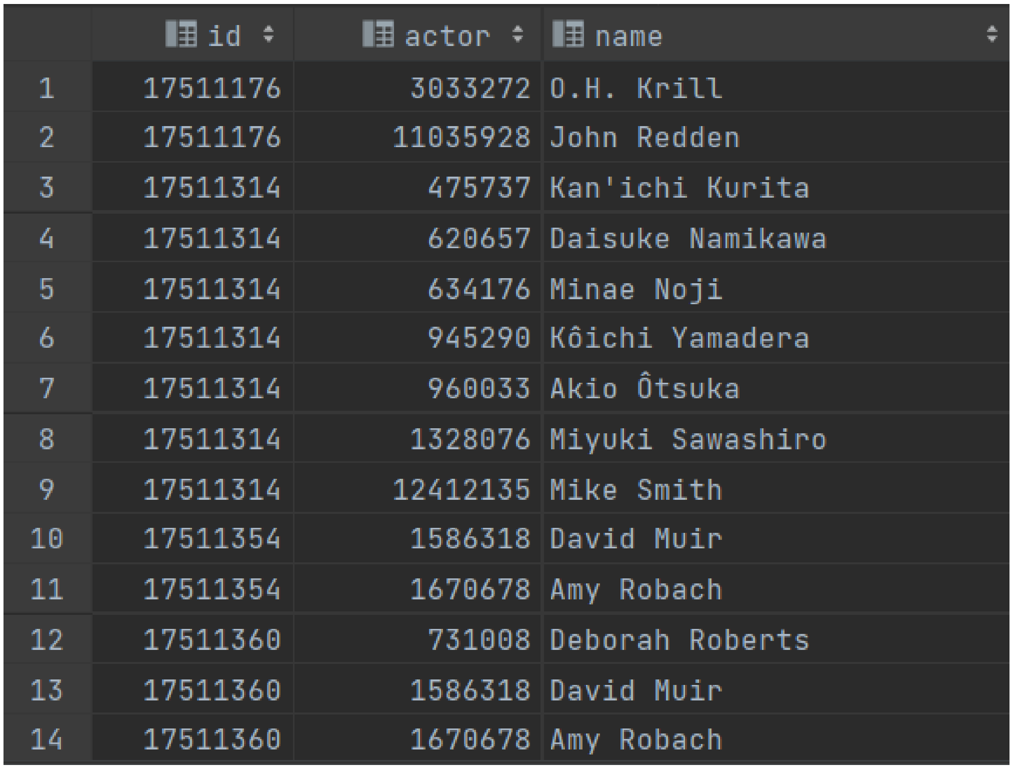


	id	name	birthYear	deathYear
1	1746603	Jürgen Bigalkes	<null>	<null>
2	1746603	Gina Janssen	1953	<null>
3	1746603	Christine Schwarz	<null>	<null>
4	1746603	Herbert Warnke	<null>	<null>
5	1746603	Margitta Hofer	<null>	<null>
6	1746603	Monika Sandmayr	<null>	<null>
7	1746603	Hans Kurt Preuss	<null>	<null>
8	1746621	Davina Joy	<null>	<null>
9	1746621	Mario Guzman	<null>	<null>
10	1746621	Jess Laman	<null>	<null>
11	1746621	Andi DeFabritis	<null>	<null>
12	17466392	Javon Johnson	<null>	<null>
13	17466392	Ed Quinn	1968	<null>
14	17466392	Ptosha Storey	1968	<null>

ActedIn:

Created a view with all actors and their movies and added the names of the actors.

```
CREATE  VIEW ActedIn AS
   SELECT t.id, ta.actor, m.name
   from hw_schema."Title" t, hw_schema."Title_Actor" ta,
hw_schema."Member" m
WHERE t.id = ta.title and m.id = ta.actor and t.runtime > 75 and t.type =
'movie';
```



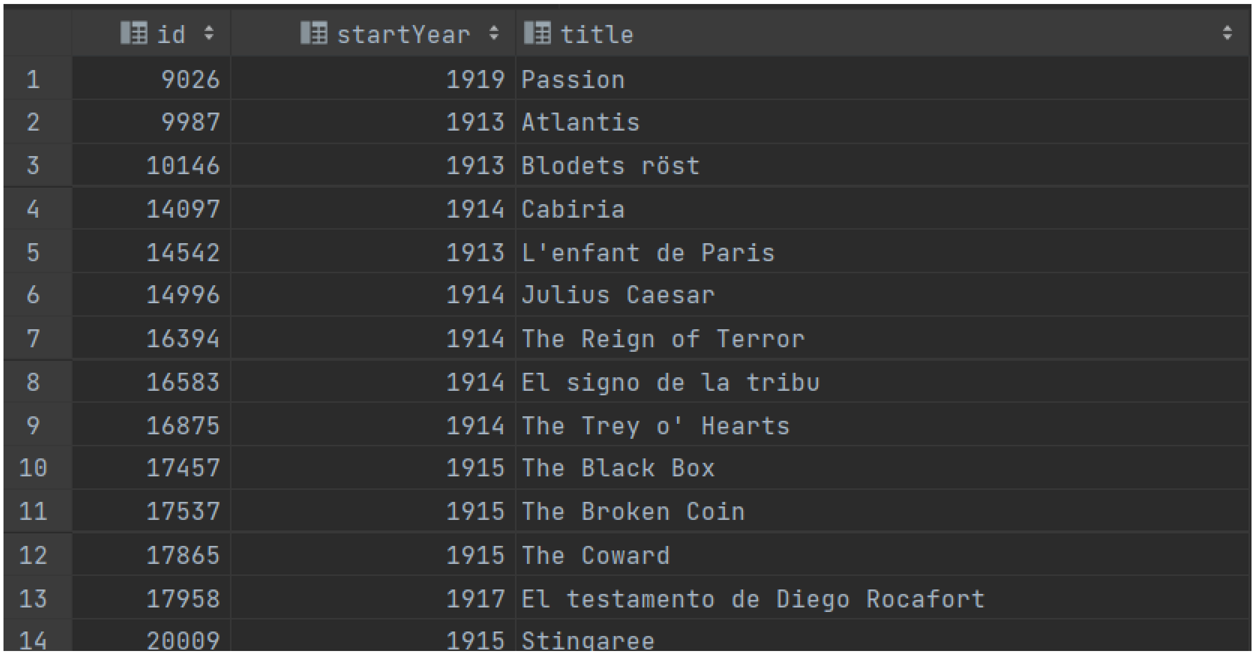
	id	actor	name
1	17511176	3033272	O.H. Krill
2	17511176	11035928	John Redden
3	17511314	475737	Kan'ichi Kurita
4	17511314	620657	Daisuke Namikawa
5	17511314	634176	Minae Noji
6	17511314	945290	Kôichi Yamadera
7	17511314	960033	Akio Ôtsuka
8	17511314	1328076	Miyuki Sawashiro
9	17511314	12412135	Mike Smith
10	17511354	1586318	David Muir
11	17511354	1670678	Amy Robach
12	17511360	731008	Deborah Roberts
13	17511360	1586318	David Muir
14	17511360	1670678	Amy Robach

Materialized

ComedyMovie_mv:

Created materialized view and selected the required fields where genre is Comedy and the runtime is greater than 75 and joined the id of movie and the genre and its respective id.

```
CREATE MATERIALIZED VIEW ComedyMovie_mv AS
    SELECT t."id", t."startYear", t.title
    from hw_schema."Genre" g, hw_schema."Title" t,
hw_schema."Title_Genre" tg
WHERE g.genre = 'Comedy'
  and t.runtime > 75
  and t.id = tg.title
  and tg.genre = g.id;
```

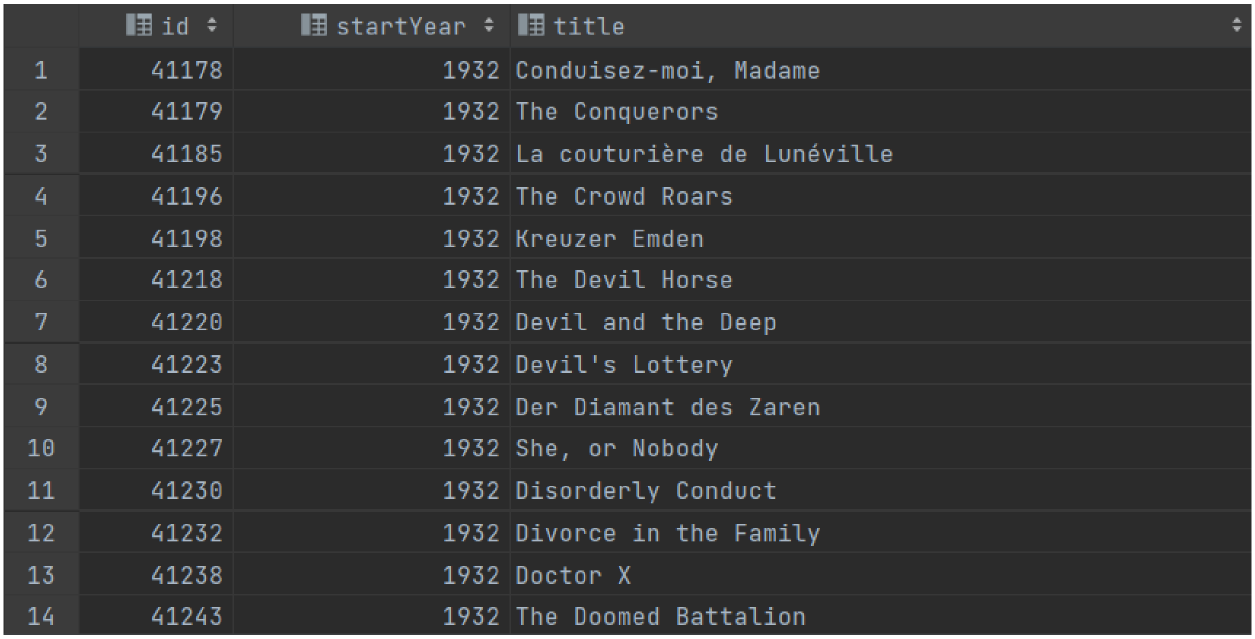


	id	startYear	title
1	9026	1919	Passion
2	9987	1913	Atlantis
3	10146	1913	Blodets röst
4	14097	1914	Cabiria
5	14542	1913	L'enfant de Paris
6	14996	1914	Julius Caesar
7	16394	1914	The Reign of Terror
8	16583	1914	El signo de la tribu
9	16875	1914	The Trey o' Hearts
10	17457	1915	The Black Box
11	17537	1915	The Broken Coin
12	17865	1915	The Coward
13	17958	1917	El testamento de Diego Rocafort
14	20009	1915	Stingaree

NonComedyMovie_mv:

Created a view and selected the required fields where genre is not Comedy and the runtime is greater than 75 and joined on the title and see where genre= 3 which is the comedy id.

```
CREATE MATERIALIZED VIEW NonComedyMovie_mv AS
    SELECT t."id", t."startYear", t.title
    from hw_schema."Title" t
WHERE
   t.runtime > 75
  and t.id not in(select t.id from hw_schema."Title" t,
hw_schema."Title_Genre" tg WHERE tg.genre = 3 and t.id = tg.title);
```

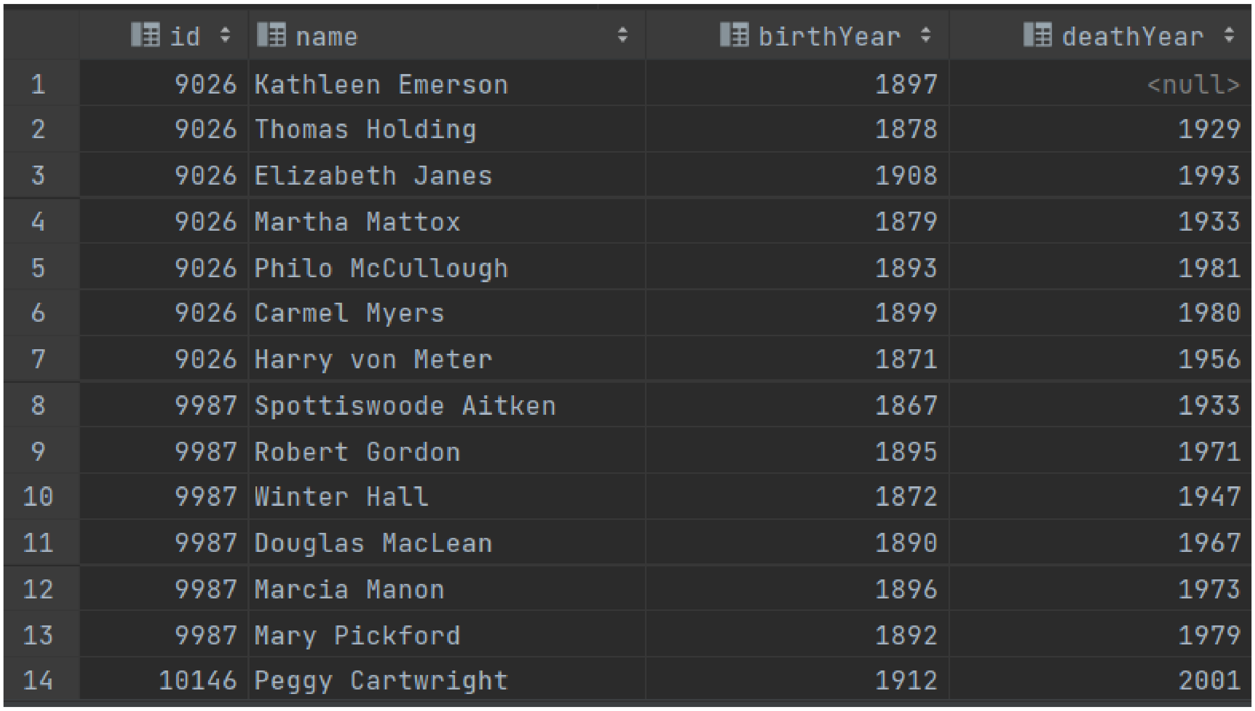


	id	startYear	title
1	41178	1932	Conduisez-moi, Madame
2	41179	1932	The Conquerors
3	41185	1932	La couturière de Lunéville
4	41196	1932	The Crowd Roars
5	41198	1932	Kreuzer Emden
6	41218	1932	The Devil Horse
7	41220	1932	Devil and the Deep
8	41223	1932	Devil's Lottery
9	41225	1932	Der Diamant des Zaren
10	41227	1932	She, or Nobody
11	41230	1932	Disorderly Conduct
12	41232	1932	Divorce in the Family
13	41238	1932	Doctor X
14	41243	1932	The Doomed Battalion

ComedyActor_mv:

Created view and selected the required fields from the respective tables where genre is Comedy and the runtime is greater than 75 minutes and joined on the actor id and movieid and the genre and its respective id.

```
CREATE MATERIALIZED VIEW ComedyActor_mv AS
   SELECT t."id",m.name,m."birthYear",m."deathYear"
   from hw_schema."Title" t, hw_schema."Member" m, hw_schema."Genre" g,
                                                  hw_schema."Title_Genre"
tg,hw_schema."Title_Actor" ta
WHERE g.genre = 'Comedy'
  and t.runtime > 75
  and t.id = tg.title
  and tg.genre = g.id
  and ta.actor = m.id
  and t.id = ta.title;
```

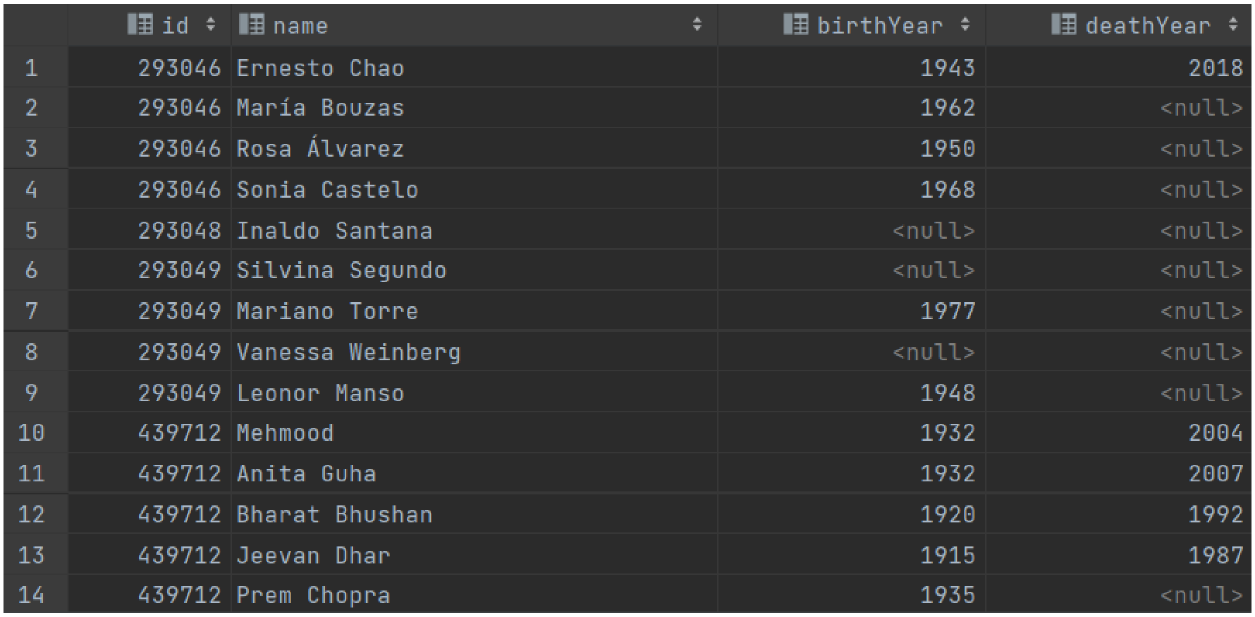


	id	name	birthYear	deathYear
1	9026	Kathleen Emerson	1897	<null>
2	9026	Thomas Holding	1878	1929
3	9026	Elizabeth Janes	1908	1993
4	9026	Martha Mattox	1879	1933
5	9026	Philo McCullough	1893	1981
6	9026	Carmel Myers	1899	1980
7	9026	Harry von Meter	1871	1956
8	9987	Spottiswoode Aitken	1867	1933
9	9987	Robert Gordon	1895	1971
10	9987	Winter Hall	1872	1947
11	9987	Douglas MacLean	1890	1967
12	9987	Marcia Manon	1896	1973
13	9987	Mary Pickford	1892	1979
14	10146	Peggy Cartwright	1912	2001

NonComedyActor_mv:

Created view and selected the required fields from the respective tables where genre is not Comedy and the runtime is greater than 75 minutes and joined on the actor id and movieid and checked with not in where the genre = 3 which is Comedy genre id.

```
CREATE MATERIALIZED VIEW NonComedyActor_mv AS
   SELECT t."id",m.name,m."birthYear",m."deathYear"
   from hw_schema."Title" t, hw_schema."Member" m,
hw_schema."Title_Actor" ta
WHERE t.runtime > 75
  and ta.actor = m.id
  and ta.title = t.id
  and t.id not in (SELECT t.id from hw_schema."Title" t ,
hw_schema."Title_Genre" tg
                              WHERE tg.genre = 3
                              and t.id = tg.title);
```

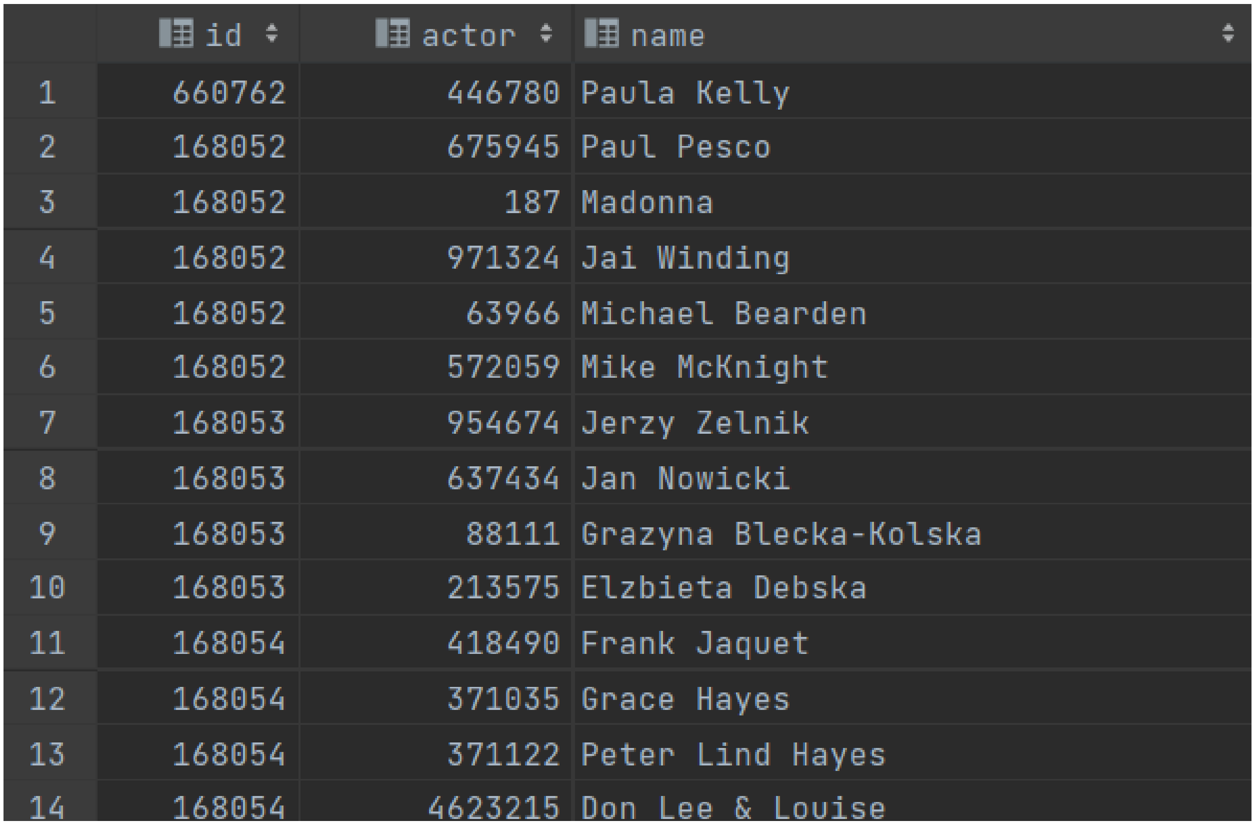


	id	name	birthYear	deathYear
1	293046	Ernesto Chao	1943	2018
2	293046	María Bouzas	1962	<null>
3	293046	Rosa Álvarez	1950	<null>
4	293046	Sonia Castelo	1968	<null>
5	293048	Inaldo Santana	<null>	<null>
6	293049	Silvina Segundo	<null>	<null>
7	293049	Mariano Torre	1977	<null>
8	293049	Vanessa Weinberg	<null>	<null>
9	293049	Leonor Manso	1948	<null>
10	439712	Mehmood	1932	2004
11	439712	Anita Guha	1932	2007
12	439712	Bharat Bhushan	1920	1992
13	439712	Jeevan Dhar	1915	1987
14	439712	Prem Chopra	1935	<null>

ActedIn_mv:

Created a view with all actors and their movies and added the names of the actors.

```
CREATE  MATERIALIZED VIEW ActedIn_mv AS
   SELECT t.id, ta.actor, m.name
   from hw_schema."Title" t, hw_schema."Title_Actor" ta,
hw_schema."Member" m
WHERE t.id = ta.title and m.id = ta.actor and t.runtime > 75 and t.type
= 'movie';
```



	id	actor	name
1	660762	446780	Paula Kelly
2	168052	675945	Paul Pesco
3	168052	187	Madonna
4	168052	971324	Jai Winding
5	168052	63966	Michael Bearden
6	168052	572059	Mike McKnight
7	168053	954674	Jerzy Zelnik
8	168053	637434	Jan Nowicki
9	168053	88111	Grazyna Blecka-Kolska
10	168053	213575	Elzbieta Debska
11	168054	418490	Frank Jaquet
12	168054	371035	Grace Hayes
13	168054	371122	Peter Lind Hayes
14	168054	4623215	Don Lee & Louise

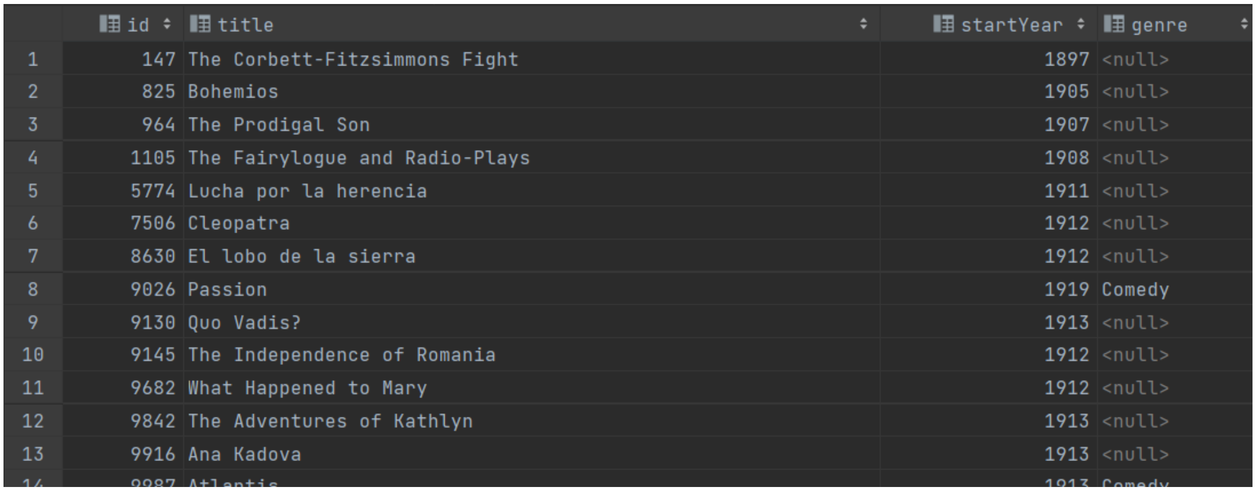
Task 2:

GAV Non-materialized

All_Movie:

Created the global view of all movie using two select statements by using union on the sources created where genre is set to null to include all the movies.

```
CREATE VIEW gav_all_movie AS
    SELECT cm.id, cm.title, cm."startYear", 'Comedy' as genre
    FROM public.comedymovie cm UNION
                              SELECT ncm.id, ncm.title, ncm."startYear", Null as
genre FROM public.noncomedymovie ncm;
```

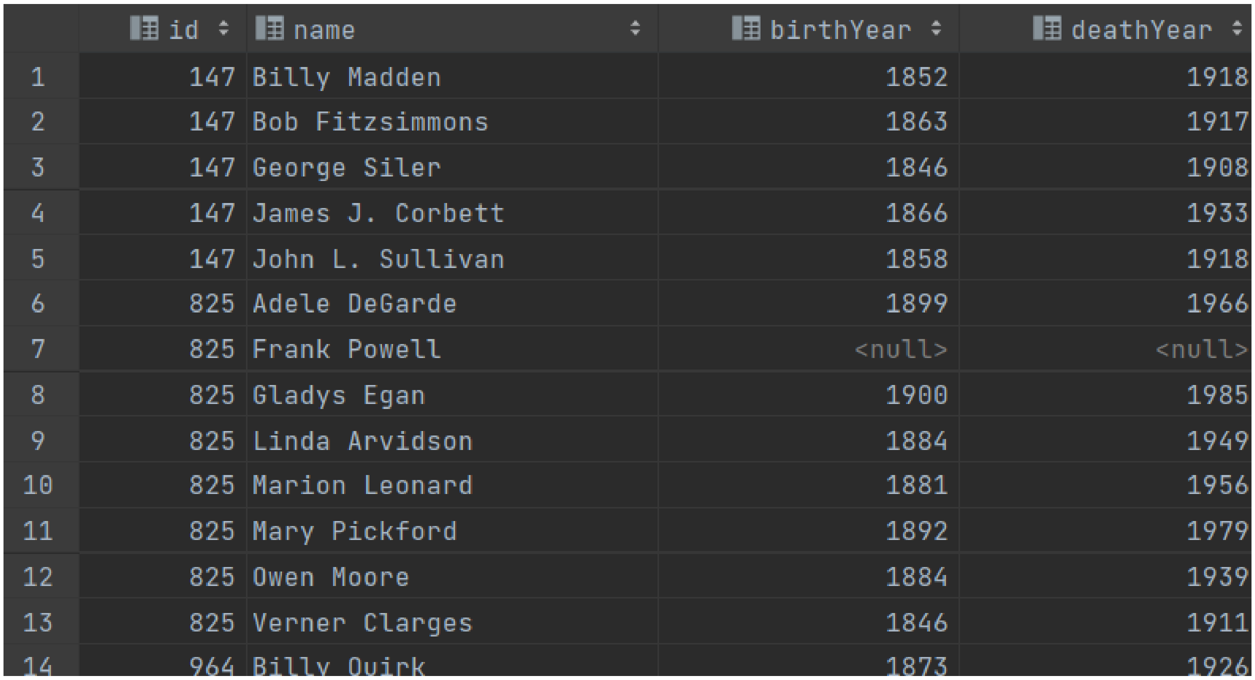


	id	title	startYear	genre
1	147	The Corbett-Fitzsimmons Fight	1897	<null>
2	825	Bohemios	1905	<null>
3	964	The Prodigal Son	1907	<null>
4	1105	The Fairylogue and Radio-Plays	1908	<null>
5	5774	Lucha por la herencia	1911	<null>
6	7506	Cleopatra	1912	<null>
7	8630	El lobo de la sierra	1912	<null>
8	9026	Passion	1919	Comedy
9	9130	Quo Vadis?	1913	<null>
10	9145	The Independence of Romania	1912	<null>
11	9682	What Happened to Mary	1912	<null>
12	9842	The Adventures of Kathlyn	1913	<null>
13	9916	Ana Kadova	1913	<null>
14	9987	Atlantis	1913	Comedy

All_Actor:

Created the global view of all actor using two select statements by using union on the sources created.

```
CREATE VIEW gav_all_actor AS
    SELECT ca.id, ca.name, ca."birthYear", ca."deathYear"
    FROM public.comedyactor ca UNION
                         SELECT nca.id, nca.name, nca."birthYear", nca."deathYear" FROM
public.noncomedyactor nca;
```

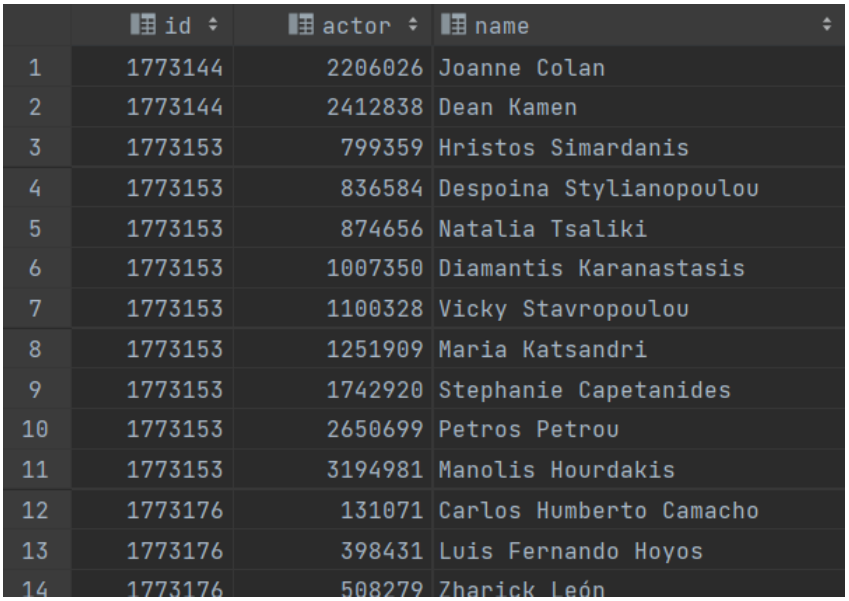


	id	name	birthYear	deathYear
1	147	Billy Madden	1852	1918
2	147	Bob Fitzsimmons	1863	1917
3	147	George Siler	1846	1908
4	147	James J. Corbett	1866	1933
5	147	John L. Sullivan	1858	1918
6	825	Adele DeGarde	1899	1966
7	825	Frank Powell	<null>	<null>
8	825	Gladys Egan	1900	1985
9	825	Linda Arvidson	1884	1949
10	825	Marion Leonard	1881	1956
11	825	Mary Pickford	1892	1979
12	825	Owen Moore	1884	1939
13	825	Verner Clarges	1846	1911
14	964	Billy Quirk	1873	1926

All_Movie_Actor:

Created the global view of all movie and actor using two select statements by using union on the sources created.

```
CREATE VIEW gav_all_movie_actor AS
   SELECT ain.id, ain.actor, ain.name FROM public.actedin ain;
```



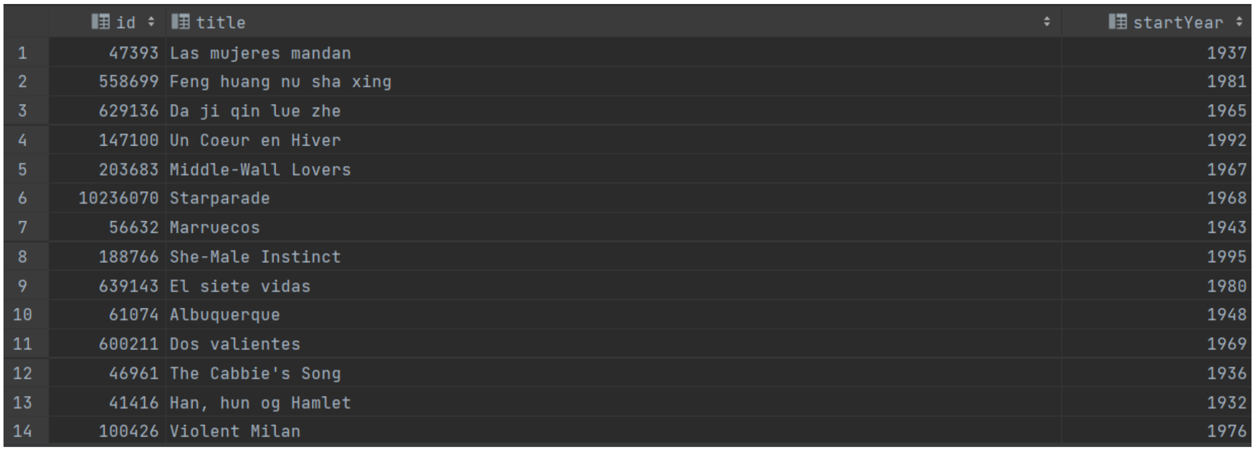
	id	actor	name
1	1773144	2206026	Joanne Colan
2	1773144	2412838	Dean Kamen
3	1773153	799359	Hristos Simardanis
4	1773153	836584	Despoina Stylianopoulou
5	1773153	874656	Natalia Tsaliki
6	1773153	1007350	Diamantis Karanastasis
7	1773153	1100328	Vicky Stavropoulou
8	1773153	1251909	Maria Katsandri
9	1773153	1742920	Stephanie Capetanides
10	1773153	2650699	Petros Petrou
11	1773153	3194981	Manolis Hourdakis
12	1773176	131071	Carlos Humberto Camacho
13	1773176	398431	Luis Fernando Hoyos
14	1773176	508279	Zharick León

GAV Materialized

All_Movie_mv:

Created the global view of all movie using two select statements by using union on the sources created where genre is set to null to include all the movies.

```
CREATE MATERIALIZED VIEW gav_all_movie_mv AS
   SELECT cm.id, cm.title, cm."startYear", 'Comedy' as genre
   FROM public.comedymovie cm UNION
                       SELECT ncm.id, ncm.title, ncm."startYear", Null as genre FROM
public.noncomedymovie ncm;
```

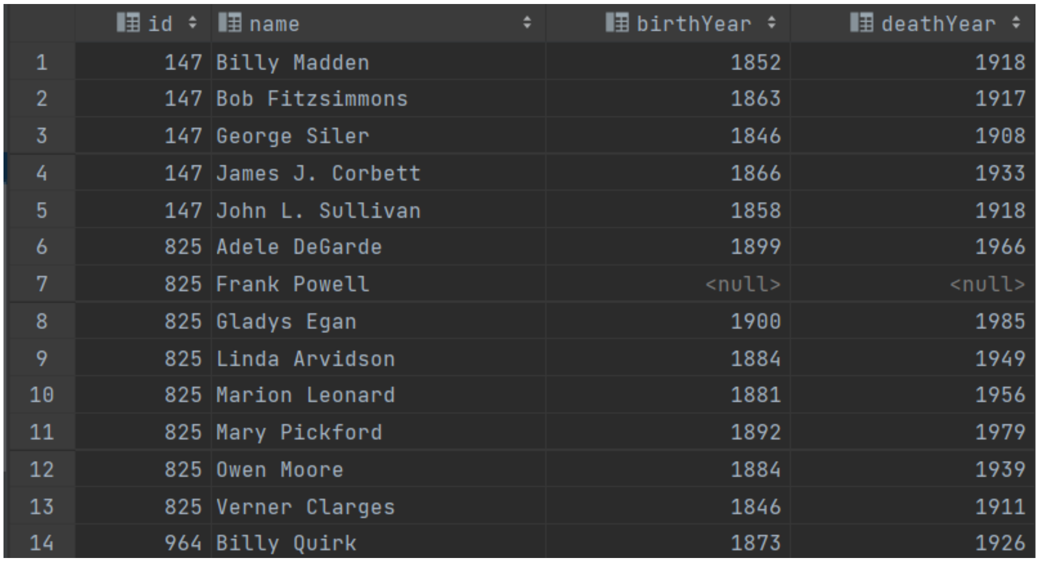


	id	title	startYear
1	47393	Las mujeres mandan	1937
2	558699	Feng huang nu sha xing	1981
3	629136	Da ji qin lue zhe	1965
4	147100	Un Coeur en Hiver	1992
5	203683	Middle-Wall Lovers	1967
6	10236070	Starparade	1968
7	56632	Marruecos	1943
8	188766	She-Male Instinct	1995
9	639143	El siete vidas	1980
10	61074	Albuquerque	1948
11	600211	Dos valientes	1969
12	46961	The Cabbie's Song	1936
13	41416	Han, hun og Hamlet	1932
14	100426	Violent Milan	1976

All_Actor_mv:

Created the global view of all actor using two select statements by using union on the sources created.

```
CREATE MATERIALIZED VIEW  gav_all_actor_mv AS
   SELECT ca.id, ca.name, ca."birthYear", ca."deathYear"
   FROM public.comedyactor ca UNION
                       SELECT nca.id, nca.name, nca."birthYear", nca."deathYear" FROM
public.noncomedyactor nca;
```

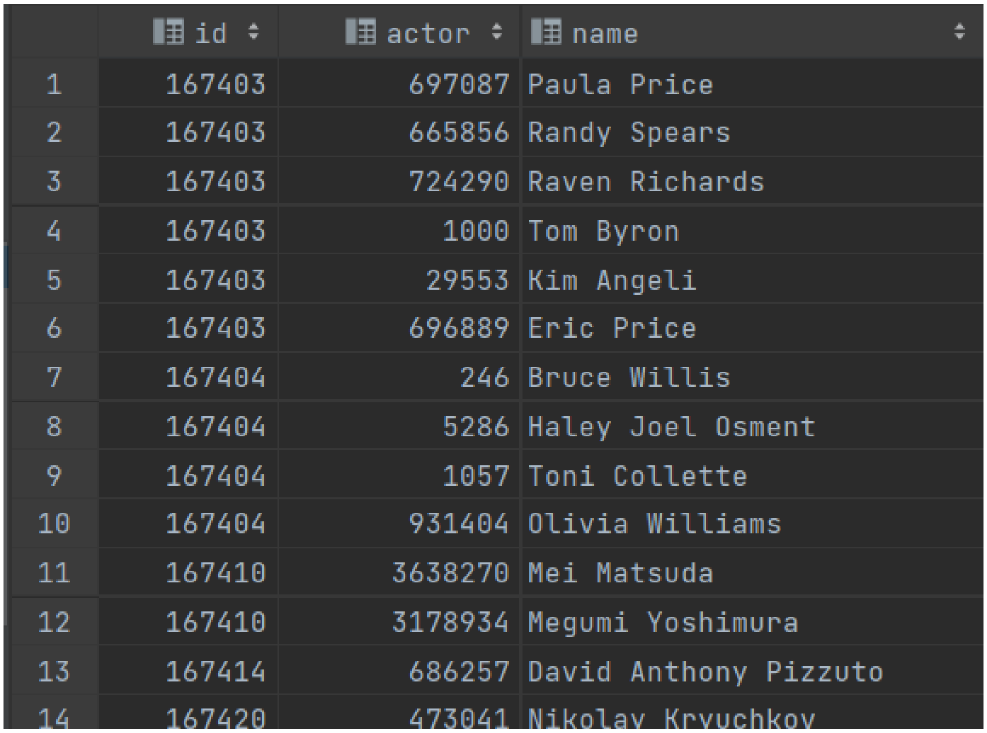


	id	name	birthYear	deathYear
1	147	Billy Madden	1852	1918
2	147	Bob Fitzsimmons	1863	1917
3	147	George Siler	1846	1908
4	147	James J. Corbett	1866	1933
5	147	John L. Sullivan	1858	1918
6	825	Adele DeGarde	1899	1966
7	825	Frank Powell	<null>	<null>
8	825	Gladys Egan	1900	1985
9	825	Linda Arvidson	1884	1949
10	825	Marion Leonard	1881	1956
11	825	Mary Pickford	1892	1979
12	825	Owen Moore	1884	1939
13	825	Verner Clarges	1846	1911
14	964	Billy Quirk	1873	1926

All_Movie_Actor_mv:

Created the global view of all movies and actors using two select statements by using union on the sources created.

CREATE MATERIALIZED VIEW gav_all_movie_actor_mv AS
SELECT ain.id, ain.actor, ain.name FROM public.actedin ain;



	id	actor	name
1	167403	697087	Paula Price
2	167403	665856	Randy Spears
3	167403	724290	Raven Richards
4	167403	1000	Tom Byron
5	167403	29553	Kim Angeli
6	167403	696889	Eric Price
7	167404	246	Bruce Willis
8	167404	5286	Haley Joel Osment
9	167404	1057	Toni Collette
10	167404	931404	Olivia Williams
11	167410	3638270	Mei Matsuda
12	167410	3178934	Megumi Yoshimura
13	167414	686257	David Anthony Pizzuto
14	167420	473041	Nikolay Kryuchkov

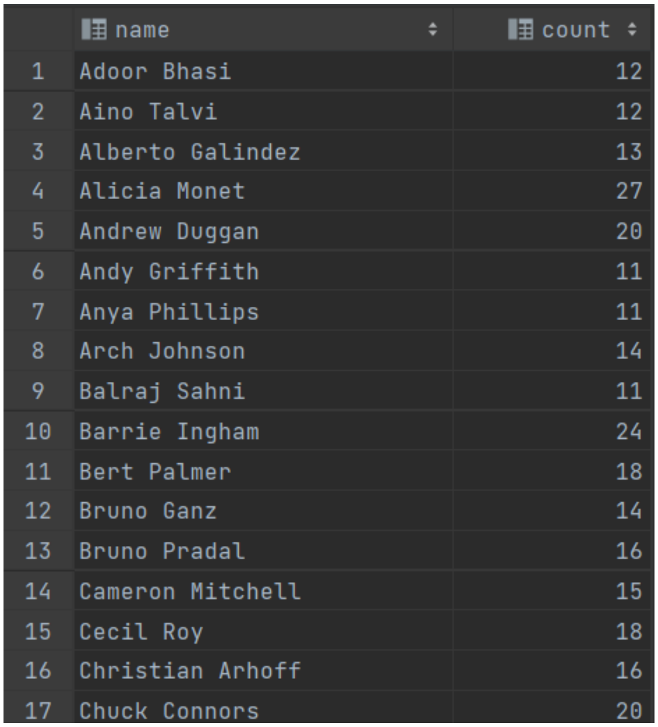
Task 3:

3.1

Selecting the actor name and their count where the actor is alive and the movies between 2000 and 2005 and grouby name and performing a count aggregate to get the actors who have worked in more than 10 movies.

```
SELECT gaa.name, count(gam.id)
FROM public.gav_all_actor gaa,
     public.gav_all_movie gam,
     public.gav_all_movie_actor gama
WHERE gaa."deathYear" is not NULL
  and gam."startYear" > 2000
  and gam."startYear" <= 2005
  AND gam.id = gama.id
  and gaa.id = gama.actor
GROUP BY gaa.name
HAVING count(gam.id) > 10;
```

Output:



	name	count
1	Adoor Bhasi	12
2	Aino Talvi	12
3	Alberto Galindez	13
4	Alicia Monet	27
5	Andrew Duggan	20
6	Andy Griffith	11
7	Anya Phillips	11
8	Arch Johnson	14
9	Balraj Sahni	11
10	Barrie Ingham	24
11	Bert Palmer	18
12	Bruno Ganz	14
13	Bruno Pradal	16
14	Cameron Mitchell	15
15	Cecil Roy	18
16	Christian Arhoff	16
17	Chuck Connors	20

Query runtime was **6sec 322ms**

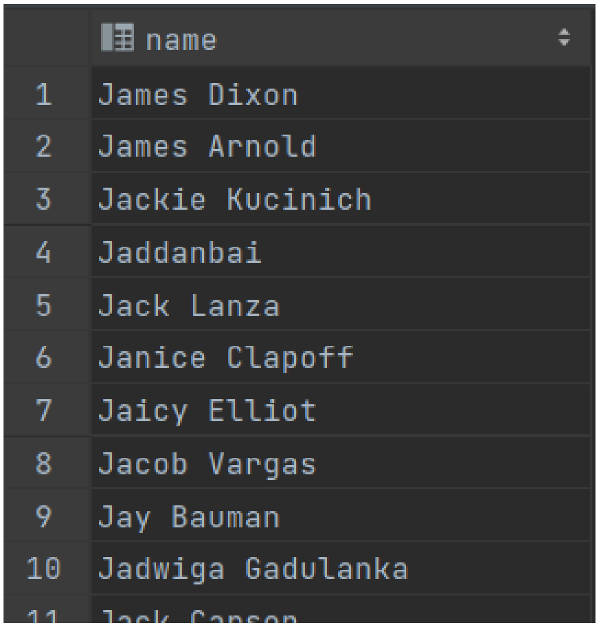
```
5   Andrew Duggan    20
6   Andy Griffith    11
```

3.2

Selecting all distinct actors whose name starts with Ja and the genre is null to get all the actors who have acted in movies apart from the Comedy movie and joining on actor id.

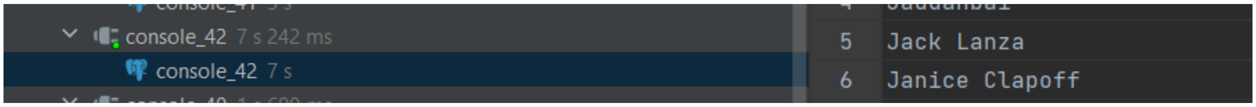
```
SELECT DISTINCT gaa.name
FROM public.gav_all_actor gaa,
     public.gav_all_movie gam,
     public.gav_all_movie_actor gama
WHERE gaa.name LIKE 'Ja%'
  and gam.genre is NULL and gam.id = gama.id
and gaa.id = gama.id;
```

Output:



```
    name
1   James Dixon
2   James Arnold
3   Jackie Kucinich
4   Jaddanbai
5   Jack Lanza
6   Janice Clapoff
7   Jaicy Elliot
8   Jacob Vargas
9   Jay Bauman
10  Jadwiga Gadulanka
11  Jack Carson
```

Query runtime was **7sec 242ms**



```
5   Jack Lanza
6   Janice Clapoff
```

Task 4:

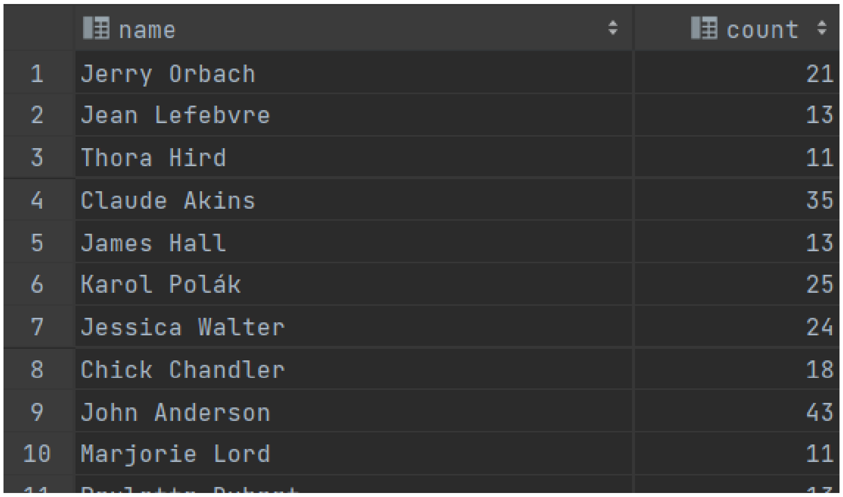
Q3.1 after expanding using GAV mapping from task2

Selecting the name and count of the movies by using the gav mapping queries by substituting the views into the queries where we are selecting actors who have acted more than 10 movies between the year 2000 and 2005 and actors who are alive.

Non-Materialized:

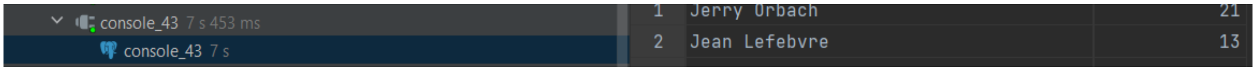
```
SELECT ca.name,count(c.id) FROM (SELECT cm.id, cm.title, cm."startYear", 'Comedy' as
genre
   FROM public.comedymovie cm UNION
                        SELECT ncm.id, ncm.title, ncm."startYear", Null as genre FROM
public.noncomedymovie ncm) AS c,
(SELECT ca.id, ca.name, ca."birthYear", ca."deathYear"
   FROM public.comedyactor ca UNION
                        SELECT nca.id, nca.name, nca."birthYear", nca."deathYear" FROM
public.noncomedyactor nca) AS ca,
            (SELECT ain.id, ain.actor FROM public.actedin ain) ain
WHERE ca."deathYear" is not NULL
  and c."startYear" > 2000
  and c."startYear" <= 2005
  AND c.id = ca.id
  and c.id = ain.actor
GROUP BY ca.name
HAVING count(c.id) > 10;
```

Output:



	name	count
1	Jerry Orbach	21
2	Jean Lefebvre	13
3	Thora Hird	11
4	Claude Akins	35
5	James Hall	13
6	Karol Polák	25
7	Jessica Walter	24
8	Chick Chandler	18
9	John Anderson	43
10	Marjorie Lord	11

Time taken to run the query without materialized view is **7s 453ms**



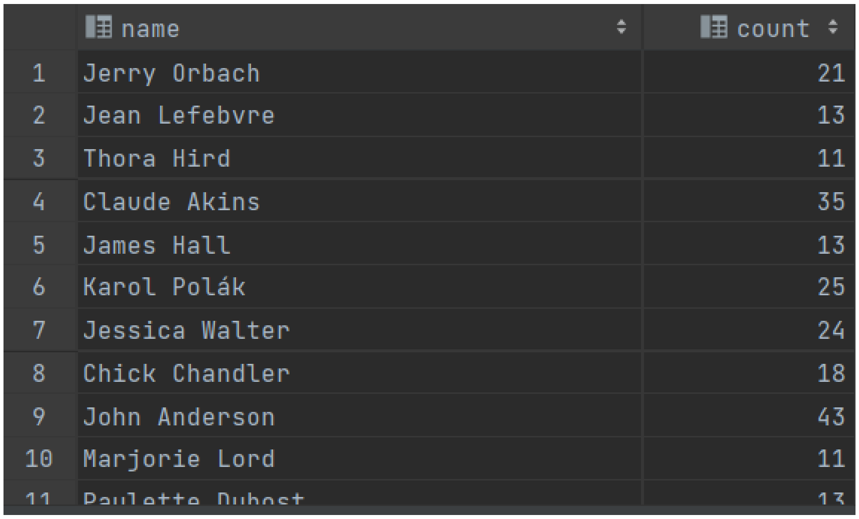
```
console_43  7 s 453 ms
    console_43  7 s
```

1	Jerry Orbach	21
2	Jean Lefebvre	13

Materialized:

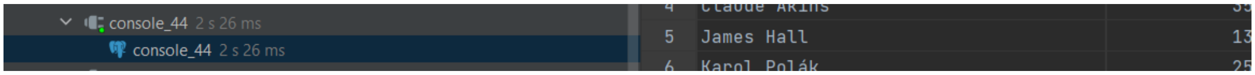
```
SELECT ca.name,count(c.id) FROM (SELECT cm.id, cm.title, cm."startYear", 'Comedy' as genre
   FROM public.comedymovie_mv cm UNION
                          SELECT ncm.id, ncm.title, ncm."startYear", Null as genre FROM public.noncomedymovie_mv ncm) AS c,
(SELECT ca.id, ca.name, ca."birthYear", ca."deathYear"
   FROM public.comedyactor_mv ca UNION
                          SELECT nca.id, nca.name, nca."birthYear", nca."deathYear" FROM public.noncomedyactor_mv nca) AS ca,
             (SELECT ain.id, ain.actor FROM public.actedin_mv ain) ain
WHERE ca."deathYear" is not NULL
  and c."startYear" > 2000
  and c."startYear" <= 2005
  AND c.id = ca.id
  and c.id = ain.actor
GROUP BY ca.name
HAVING count(c.id) > 10;
```

Output:



	name	count
1	Jerry Orbach	21
2	Jean Lefebvre	13
3	Thora Hird	11
4	Claude Akins	35
5	James Hall	13
6	Karol Polák	25
7	Jessica Walter	24
8	Chick Chandler	18
9	John Anderson	43
10	Marjorie Lord	11
11	Paulette Dubost	13

Time taken to run the query with materialized view is **2s 26ms**



```
console_44 2 s 26 ms
    console_44 2 s 26 ms
```

4	Claude Akins	35
5	James Hall	13
6	Karol Polák	25

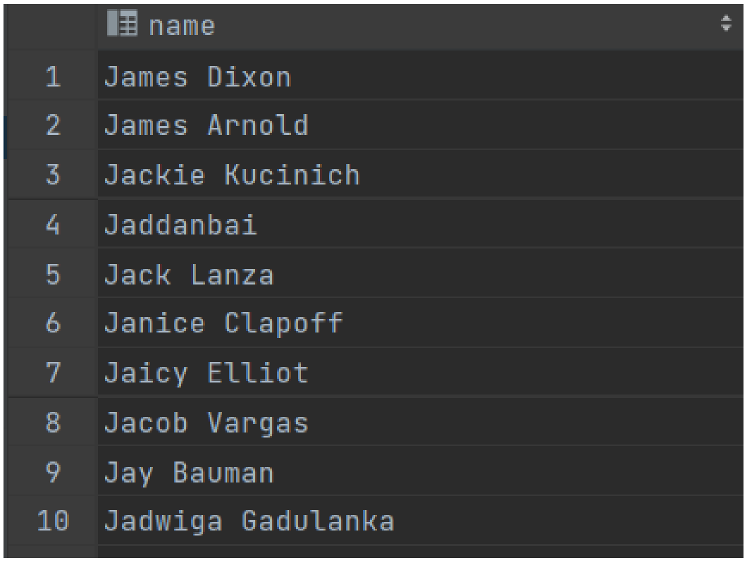
Q3.2 after expanding using GAV mapping from task2

Non-Materialized:

Selecting the name of the actor whose name starts with Ja and who have not participated in any comedy where setting the genre as Null selects only the movies which don't have genre in it and joining on the the actor id in comedy_actor and acted_in

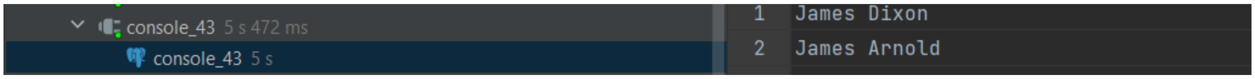
```
SELECT DISTINCT ca.name FROM (SELECT cm.id, cm.title, cm."startYear", 'Comedy' as
genre
   FROM public.comedymovie cm UNION
                          SELECT ncm.id, ncm.title, ncm."startYear", Null as genre FROM
public.noncomedymovie ncm) AS c,
(SELECT ca.id, ca.name, ca."birthYear", ca."deathYear"
   FROM public.comedyactor ca UNION
                          SELECT nca.id, nca.name, nca."birthYear", nca."deathYear" FROM
public.noncomedyactor nca) AS ca,
             (SELECT ain.id, ain.actor FROM public.actedin ain) ain
WHERE ca.name LIKE 'Ja%'
  and c.genre is NULL and ca.id = ain.id
and c.id = ain.id;
```

Output:



	name
1	James Dixon
2	James Arnold
3	Jackie Kucinich
4	Jaddanbai
5	Jack Lanza
6	Janice Clapoff
7	Jaicy Elliot
8	Jacob Vargas
9	Jay Bauman
10	Jadwiga Gadulanka

Time taken to run the query without materialized view is **5s 472ms**



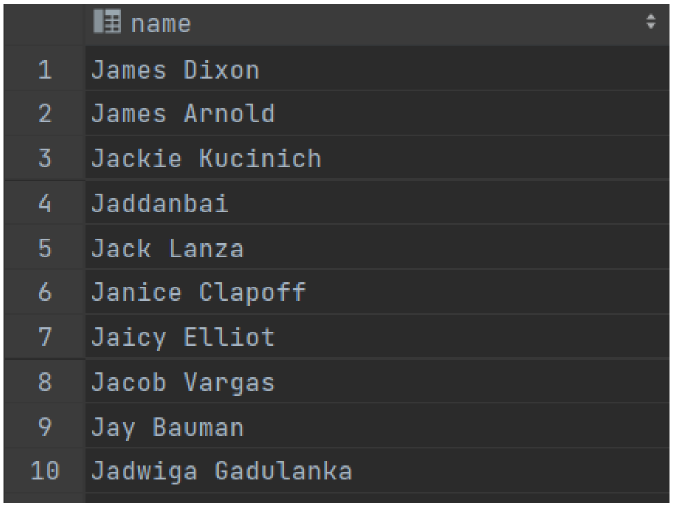
```
console_43 5 s 472 ms
    console_43 5 s
```

```
1   James Dixon
2   James Arnold
```

Materialized:

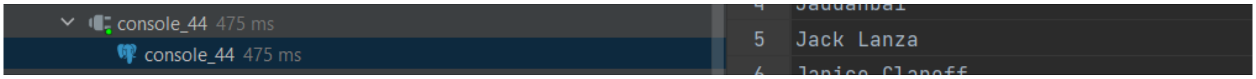
```
SELECT DISTINCT ca.name FROM (SELECT cm.id, cm.title, cm."startYear", 'Comedy' as
genre
   FROM public.comedymovie_mv cm UNION
                         SELECT ncm.id, ncm.title, ncm."startYear", Null as genre FROM
public.noncomedymovie_mv ncm) AS c,
(SELECT ca.id, ca.name, ca."birthYear", ca."deathYear"
   FROM public.comedyactor_mv ca UNION
                         SELECT nca.id, nca.name, nca."birthYear", nca."deathYear" FROM
public.noncomedyactor_mv nca) AS ca,
             (SELECT ain.id, ain.actor FROM public.actedin_mv ain) ain
WHERE ca.name LIKE 'Ja%'
  and c.genre is NULL and ca.id = ain.id
and c.id = ain.id;
```

Output:



	name
1	James Dixon
2	James Arnold
3	Jackie Kucinich
4	Jaddanbai
5	Jack Lanza
6	Janice Clapoff
7	Jaicy Elliot
8	Jacob Vargas
9	Jay Bauman
10	Jadwiga Gadulanka

Time taken to run the query with materialized view is **475ms**



```
console_44 475 ms
    console_44 475 ms
```

```
4  Jaddanbai
5  Jack Lanza
6  Janice Clapoff
```

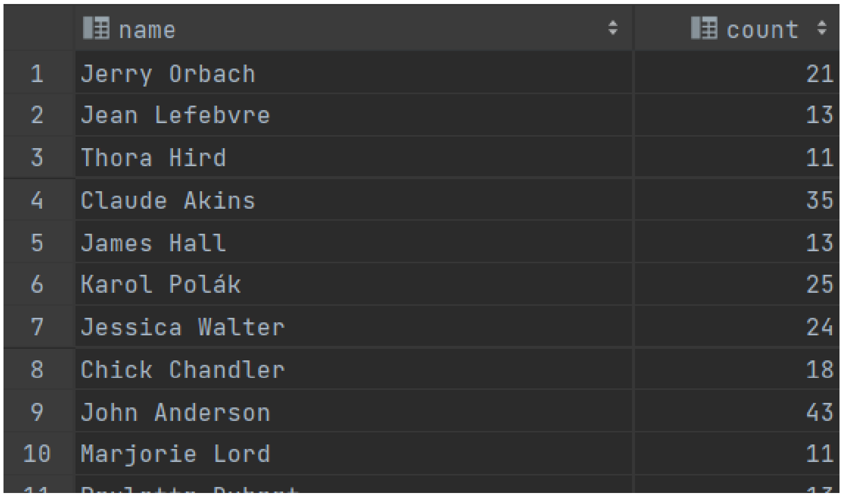
Task 5:

Non-materialized timing

There will not be any reduction in the joining and the reduction of joins can only be achieved by having all the fields in a single source or a separate new global schema where all the fields are in that schema.

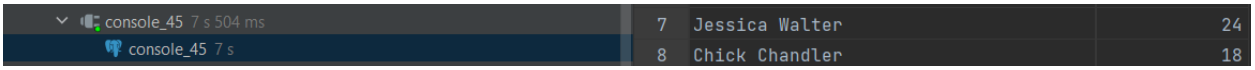
```
SELECT ca.name,count(c.id) FROM (SELECT cm.id, cm.title, cm."startYear", 'Comedy' as genre
    FROM public.comedymovie cm UNION
                              SELECT ncm.id, ncm.title, ncm."startYear", Null as genre FROM public.noncomedymovie ncm) AS c,
(SELECT ca.id, ca.name, ca."birthYear", ca."deathYear"
    FROM public.comedyactor ca UNION
                              SELECT nca.id, nca.name, nca."birthYear", nca."deathYear" FROM public.noncomedyactor nca) AS ca,
                (SELECT ain.id, ain.actor FROM public.actedin ain) ain
WHERE ca."deathYear" is not NULL
  and c."startYear" > 2000
  and c."startYear" <= 2005
  AND c.id = ca.id
  and c.id = ain.actor
GROUP BY ca.name
HAVING count(c.id) > 10;
```

Output:



	name	count
1	Jerry Orbach	21
2	Jean Lefebvre	13
3	Thora Hird	11
4	Claude Akins	35
5	James Hall	13
6	Karol Polák	25
7	Jessica Walter	24
8	Chick Chandler	18
9	John Anderson	43
10	Marjorie Lord	11

Time taken to run the query is **7s 504ms**



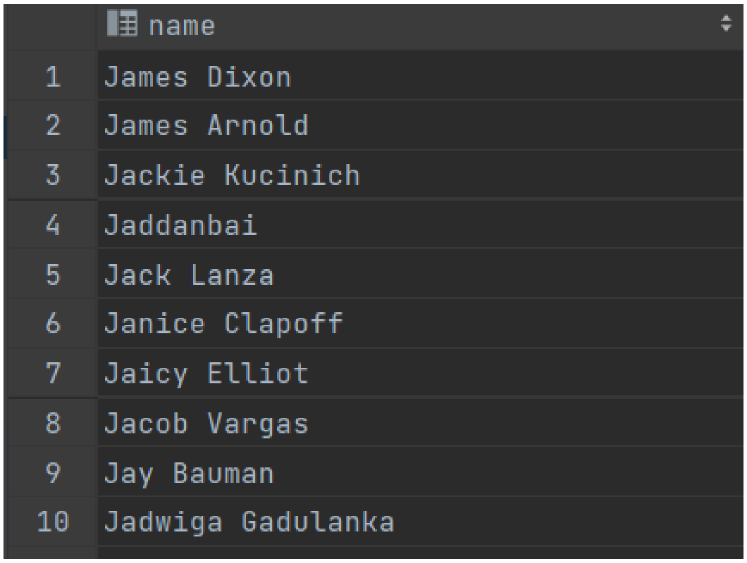
```
console_45 7 s 504 ms        7  Jessica Walter    24
  console_45 7 s             8  Chick Chandler    18
```

For Q3.2

As you can see there are not many joins, in fact no joins were used as you can get the Name starts with Ja from noncomedyactor

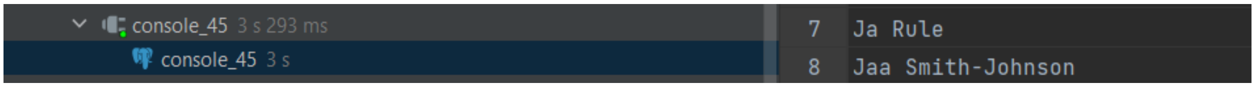
SELECT DISTINCT nca.name
FROM public.noncomedyactor nca
WHERE nca.name LIKE 'Ja%';

Output:



	name
1	James Dixon
2	James Arnold
3	Jackie Kucinich
4	Jaddanbai
5	Jack Lanza
6	Janice Clapoff
7	Jaicy Elliot
8	Jacob Vargas
9	Jay Bauman
10	Jadwiga Gadulanka

The time taken to run the query **is 3s 293ms**



```
console_45 3 s 293 ms        7  Ja Rule
  console_45 3 s             8  Jaa Smith-Johnson
```

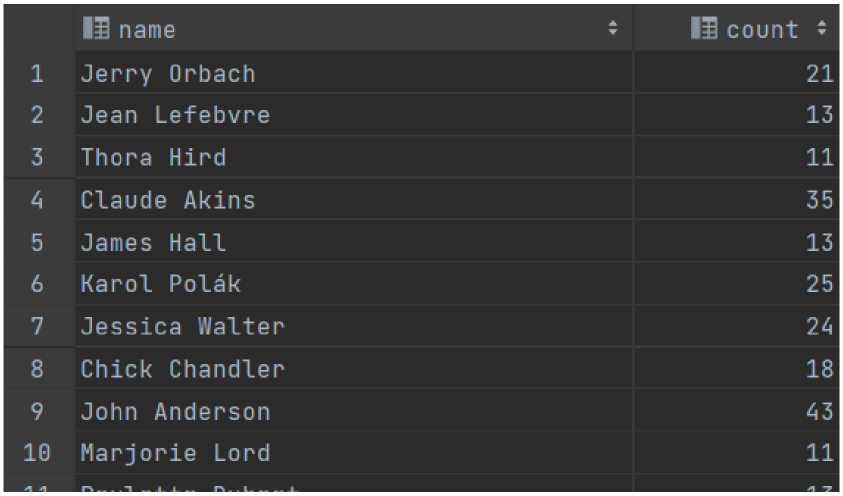
Materialized timing

For Q3.1

There will not be any reduction in the joining and the reduction of joins can only be achieved by having all the fields in a single source or a separate new global schema where all the fields are in that schema.

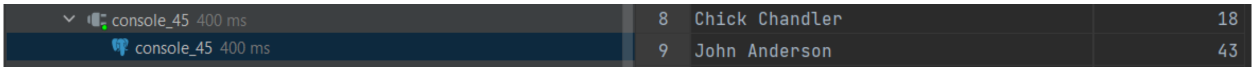
```
SELECT ca.name,count(c.id) FROM (SELECT cm.id, cm.title, cm."startYear", 'Comedy' as
genre
   FROM public.comedymovie_mv cm UNION
                        SELECT ncm.id, ncm.title, ncm."startYear", Null as genre FROM
public.noncomedymovie_mv ncm) AS c,
(SELECT ca.id, ca.name, ca."birthYear", ca."deathYear"
   FROM public.comedyactor_mv ca UNION
                        SELECT nca.id, nca.name, nca."birthYear", nca."deathYear" FROM
public.noncomedyactor_mv nca) AS ca,
            (SELECT ain.id, ain.actor FROM public.actedin_mv ain) ain
WHERE ca."deathYear" is not NULL
  and c."startYear" > 2000
  and c."startYear" <= 2005
  AND c.id = ca.id
  and c.id = ain.actor
GROUP BY ca.name
HAVING count(c.id) > 10;
```

Output:



	name	count
1	Jerry Orbach	21
2	Jean Lefebvre	13
3	Thora Hird	11
4	Claude Akins	35
5	James Hall	13
6	Karol Polák	25
7	Jessica Walter	24
8	Chick Chandler	18
9	John Anderson	43
10	Marjorie Lord	11

Time taken to run the query is **400ms**



```
console_45 400 ms
  console_45 400 ms
```

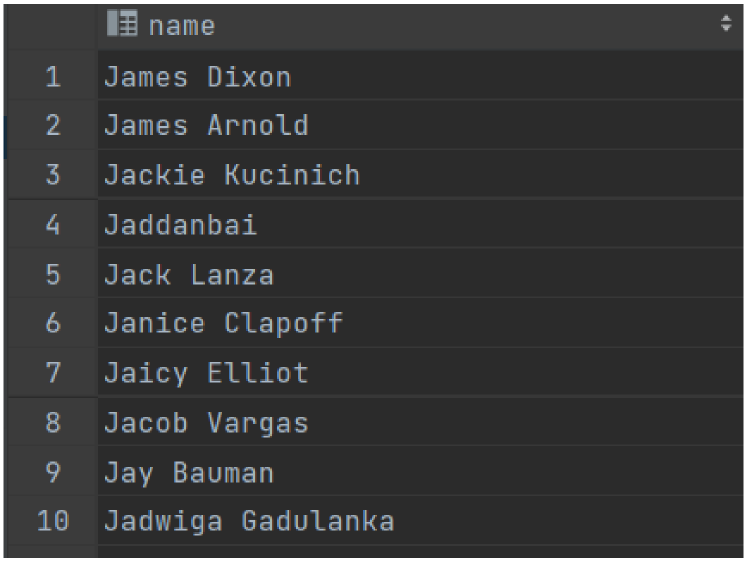
8	Chick Chandler	18
9	John Anderson	43

For Q3.2

As you can see there are not many joins, in fact no joins were used as you can get the Name starts with Ja from noncomedyactor_mv

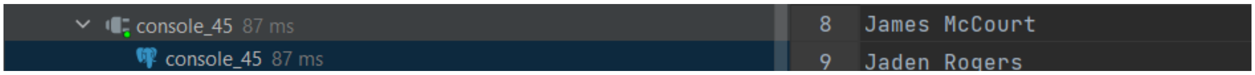
```
SELECT DISTINCT nca.name
FROM public.noncomedyactor_mv nca
WHERE nca.name LIKE 'Ja%';
```

Output



	name
1	James Dixon
2	James Arnold
3	Jackie Kucinich
4	Jaddanbai
5	Jack Lanza
6	Janice Clapoff
7	Jaicy Elliot
8	Jacob Vargas
9	Jay Bauman
10	Jadwiga Gadulanka

Time taken to run the query is **87ms**



```
console_45 87 ms
   console_45 87 ms
```

```
8  James McCourt
9  Jaden Rogers
```