ASSIGNMENT 5 - DATA INTEGRATION

Q1
Creating a table named Movies which contains all the movies with runtime greater than 75

```
CREATE TABLE Movies(
        id integer PRIMARY KEY,
        type varchar(25),
        title text,
        originalTitle text,
        startYear integer,
        endYear integer,
        runtime integer,
        avgRating float,
        numVotes integer);

insert into Movies
select * from title where runtime > 75
and type='movie';
```

Creating Source S1

● S1: ComedyMovie(id, title, year), which stores comedy movies.

Creating materialized and non-materialized views for the source s1

Creating non-materialized view

```
create view ComedyMovie as
select mo.id, mo.title, mo.startYear as year from Movies as mo
join
Title_Genre as tg on mo.id = tg.title
join Genre as ge on ge.id = tg.genre
where ge.genre = 'Comedy';
```

--Query returned successfully in 45 msec.

Creating materialized view

```
create materialized view ComedyMovieM as
select mo.id, mo.title, mo.startYear as year from Movies as mo
join
Title_Genre as tg on mo.id = tg.title
join Genre as ge on ge.id = tg.genre
where ge.genre = 'Comedy';
```

--Query returned successfully in 984 msec.

Creating Source S2

● S2: NonComedyMovie(id, title, year), which stores movies that are not comedies (there is no comedy genre related to them)

Creating materialized and non-materialized views for the source s2

Creating non-materialized view

```
create view NonComedyMovie as
select mo.id, mo.title, mo.startYear as year from Movies as mo
join
Title_Genre as tg on mo.id = tg.title
join
Genre as ge on ge.id = tg.genre
where ge.genre <> 'Comedy' and mo.id not in (select id from ComedyMovie);
```

--Query returned successfully in 44 msec.

Creating materialized view

```
create materialized view NonComedyMovieM as
select mo.id, mo.title, mo.startYear as year from Movies as mo
join
Title_Genre as tg on mo.id = tg.title
join
Genre as ge on ge.id = tg.genre
where ge.genre <> 'Comedy' and mo.id not in (select id from ComedyMovie);
```

--Query returned successfully in 4 secs 216 msec.

Creating Source S3

● S3: ComedyActor(id, name, birthYear, deathYear), which stores actors who have participated in at least a comedy movie

Creating materialized and non-materialized views for the source s3

Creating non-materialized view

```
create view ComedyActor as
select me.id, me.name, me.birthYear, me.deathYear from Member as me where me.id in
(select actor from Title_Actor as ta where exists(select mo.id from Movies as mo
join
Title_Genre as tg on mo.id = tg.title
join Genre as ge on ge.id = tg.genre
where ge.genre = 'Comedy'));
```

--Query returned successfully in 39 msec.

Creating materialized view

```
create materialized view ComedyActorM as
select me.id, me.name, me.birthYear, me.deathYear from Member as me where me.id in
(select actor from Title_Actor as ta where exists(select mo.id from Movies as mo
join
Title_Genre as tg on mo.id = tg.title
join Genre as ge on ge.id = tg.genre
where ge.genre = 'Comedy'));
```

--Query returned successfully in 13 secs 630 msec.

Creating Source S4

● S4: NonComedyActor(id, name, birthYear, deathYear), which stores actors who have never participated in any comedy movie

Creating materialized and non-materialized views for the source s4

Creating non-materialized view

```
create view NonComedyActor as
select me.id, me.name, me.birthYear, me.deathYear from Member as me where me.id in
(select actor from Title_Actor as ta where title in (select mo.id from Movies as mo
join
Title_Genre as tg on mo.id = tg.title
join
Genre as ge on ge.id = tg.genre
where ge.genre <> 'Comedy'));
```

--Query returned successfully in 40 msec.

Creating materialized view

```
create materialized view NonComedyActorM as
select me.id, me.name, me.birthYear, me.deathYear from Member as me where me.id in
(select actor from Title_Actor as ta where title in (select mo.id from Movies as mo
join
Title_Genre as tg on mo.id = tg.title
join
Genre as ge on ge.id = tg.genre
where ge.genre <> 'Comedy'));
```

--Query returned successfully in 13 secs 618 msec.

Creating Source S5

● S5: ActedIn(actor, movie), which stores all actors participation in movies

Creating materialized and non-materialized views for the source s5

Creating non-materialized view

```
create view ActedIn as
select ta.actor, ta.title as movie
from Title_Actor as ta
where ta.title in (select id from Movies as mo where ta.title = mo.id);
```

--Query returned successfully in 43 msec.

Creating materialized view

```
create materialized view ActedInM as
select ta.actor, title as movie
from Title_Actor as ta
where ta.title in (select id from Movies as mo where ta.title = mo.id);
```

--Query returned successfully in 9 secs 349 msec.

Creating the global schemas:

Non-Materialized

```
create view All_Movie as
select id,title,year,'Comedy' as genre from ComedyMovie
Union
select id,title,year,'Non-Comedy' as genre from NonComedyMovie;

Create view All_Actor as
select id,name,birthyear,deathyear from ComedyActor
Union
select id,name,birthyear,deathyear from NonComedyActor;

Create view All_Movie_Actor as
select actor,movie from ActedIn;
```

Materialized

```
create materialized view All_MovieM as
select id,title,year,'Comedy' as genre from ComedyMovieM
Union
select id,title,year,'Non-Comedy' as genre from NonComedyMovieM;

Create materialized view All_ActorM as
select id,name,birthyear,deathyear from ComedyActorM
```

```
Union
select id,name,birthyear,deathyear from NonComedyActorM;

Create materialized view All_Movie_ActorM as
select actor,movie from ActedInM;
```

The All_MovieM global schema was created by the union of ComedyMovieM and NonComedyMovieM where the genre was set as ‘Comedy’ and ‘Non-Comedy’.The All_ActorM global schema was created by the union of ComedyActorM and NonComedyActorM.The All_Movie_ActorM was created by selecting actor and movie data from ActedInM.

3.1. Alive actors who have participated in more than 10 movies between 2000 and 2005.

```
select act.name
from All_Movie as am
Join All_Movie_Actor as ama
on ama.movie=am.id
join All_Actor as act
on act.id=ama.actor
where am.year between 2000 and 2005
group by actor,act.name
having count(movie)>10;
```
--Total query runtime: 485 msec.

3.2. Actors whose name starts with “Ja” and who have never participated in any comedy movie

```
select act.name
from All_Movie as am
Join All_Movie_Actor as ama
on ama.movie=am.id
join All_Actor as act
on act.id=ama.actor
where act.name like 'Ja%'
and am.genre='Non-Comedy';
```
--Total query runtime: 249 msec.

Q4.

4.1 Non-Materialized

```
select act.name
from
(select id,title,year,'Comedy' as genre from ComedyMovie
Union
select id,title,year,'Non-Comedy' as genre from NonComedyMovie) as am
```

```
Join
(select actor,movie from ActedIn) as aim
on aim.movie=am.id
join
(select id,name,birthyear,deathyear from ComedyActor
Union
select id,name,birthyear,deathyear from NonComedyActor) as act
on act.id=aim.actor
where am.year between 2000 and 2005
group by actor,act.name
having count(movie)>10;
```
--Total query runtime: 43 secs 378 msec.

4.1 Materialized

```
select act.name
from
(select id,title,year,'Comedy' as genre from ComedyMovieM
Union
select id,title,year,'Non-Comedy' as genre from NonComedyMovieM) as am
Join
(select actor,movie from ActedInM) as aim
on aim.movie=am.id
join
(select id,name,birthyear,deathyear from ComedyActorM
Union
select id,name,birthyear,deathyear from NonComedyActorM) as act
on act.id=aim.actor
where am.year between 2000 and 2005
group by actor,act.name
having count(movie)>10;
```
--Total query runtime: 1 secs 283 msec.

4.2 Non-Materialized

```
select act.name
from
(select id,title,year,'Comedy' as genre from ComedyMovie
Union
select id,title,year,'Non-Comedy' as genre from NonComedyMovie) as am
Join
(select actor,movie from ActedIn) as aim
on aim.movie=am.id
join
(select id,name,birthyear,deathyear from ComedyActor
Union
```

```
select id,name,birthyear,deathyear from NonComedyActor) as act
on act.id=aim.actor
where act.name like 'Ja%'
and am.genre='Non-Comedy';
```

--Total query runtime: 21 secs 600 msec.

4.2 Materialized

```
select act.name
from
(select id,title,year,'Comedy' as genre from ComedyMovieM
Union
select id,title,year,'Non-Comedy' as genre from NonComedyMovieM) as am
Join
(select actor,movie from ActedInM) as aim
on aim.movie=am.id
join
(select id,name,birthyear,deathyear from ComedyActorM
Union
select id,name,birthyear,deathyear from NonComedyActorM) as act
on act.id=aim.actor
where act.name like 'Ja%'
and am.genre='Non-Comedy';
```

--Total query runtime: 1 secs 172 msec.

Q5.

5.1

4.1 Materialized

```
select act.name
from
(select id,title,year,'Comedy' as genre from ComedyMovieM
Union
select id,title,year,'Non-Comedy' as genre from NonComedyMovieM) as am
Join
(select actor,movie from ActedInM) as aim
on aim.movie=am.id
join
(select id,name,birthyear,deathyear from ComedyActorM
Union
select id,name,birthyear,deathyear from NonComedyActorM) as act
on act.id=aim.actor
where am.year between 2000 and 2005
group by actor,act.name
```

having count(movie)>10;
--Total query runtime: 1 secs 283 msec.

4.2 Non-Materialized

```
select act.name
from
(select id,title,year,'Comedy' as genre from ComedyMovie
Union
select id,title,year,'Non-Comedy' as genre from NonComedyMovie) as am
Join
(select actor,movie from ActedIn) as aim
on aim.movie=am.id
join
(select id,name,birthyear,deathyear from ComedyActor
Union
select id,name,birthyear,deathyear from NonComedyActor) as act
on act.id=aim.actor
where act.name like 'Ja%'
and am.genre='Non-Comedy';
```

--Total query runtime: 21 secs 600 msec.

The query 4.1 cannot be further optimized and there are no redundant joins in it which can be eliminated further.

5.2
Optimizing query 4.2 by eliminating redundant joins.

Non-materialized view

```
select distinct act.name
from
(select id,title,year from NonComedyMovie) as am
inner join
(select actor,movie from ActedIn) as aim
on aim.movie=am.id
inner join
(select id,name,birthyear,deathyear from NonComedyActor) as act
on act.id=aim.actor
where act.name like 'Ja%';
```

--Total query runtime: 17 secs 477 msec.

Materialized view

```
select distinct act.name
from
(select id,title,year from NonComedyMovieM) as am
inner join
(select actor,movie from ActedInM) as aim
on aim.movie=am.id
inner join
(select id,name,birthyear,deathyear from NonComedyActorM) as act
on act.id=aim.actor
where act.name like 'Ja%';
```

--Total query runtime: 210 msec.

After eliminating the redundant joins the materialized view runs faster than the non-materialized view. The materialized view took 210 msec to give the output whereas the non-materialized view took 17 sec 477 msec to give the output.