**HVA Mini Hackathon 1 Queries on Imbd Dataset**

-- Selecting all records from the dataset

select \* from `jkdb.imdb`;

**#1st analysis:-**

-- Top 10 directors in the dataset according to gross earnings

select `Director`, avg(`Gross`) as AverageGross

from `jkdb.imdb`

group by `Director`

order by AverageGross desc

limit 10;

-- Top 3 directors in the dataset according to gross earnings

select `Director`, avg(`Gross`) as AverageGross

from `jkdb.imdb`

group by `Director`

order by AverageGross desc

limit 3;

-- Number of movies released by directors

select `Director`, count(`Movie\_ID`) as NumberOfMovies, avg(`Gross`) as AverageGross

from `jkdb.imdb`

group by `Director`

order by AverageGross desc

limit 10;

-- Max number of movies released by a director in a year

with moviecount as (

  select `director`, `released\_year`, count(\*) as movies\_count

  from `jkdb.imdb`

  group by `director`, `released\_year`

),

maxmovies as (

  select `director`, `released\_year`, movies\_count

  from moviecount

  where movies\_count = (

    select max(movies\_count)

    from moviecount

    where `director` = moviecount.`director` and `released\_year` = moviecount.`released\_year`

  )

)

select `director`, `released\_year`, movies\_count

from maxmovies

order by movies\_count desc;

**#2nd analysis:-**

#number of movies released per genre

select genre,count(\*) as numberOfMovies from jkdb.imdb group by genre order by numberOfMovies desc;

#most popular query according to the Rating

select genre,AVG(IMDB\_Rating) as MostRated from jkdb.imdb group by genre order by MostRated;

#most released genre in every year of 10 range form 1920 to 2020 according to number of movies

with decade\_genre\_counts as (

  select

    case

      when released\_year between 1920 and 1929 then 1920

      when released\_year between 1930 and 1939 then 1930

      when released\_year between 1940 and 1949 then 1940

      when released\_year between 1950 and 1959 then 1950

      when released\_year between 1960 and 1969 then 1960

      when released\_year between 1970 and 1979 then 1970

      when released\_year between 1980 and 1989 then 1980

      when released\_year between 1990 and 1999 then 1990

      when released\_year between 2000 and 2009 then 2000

      when released\_year between 2010 and 2019 then 2010

    end as decade,

    genre,

    count(\*) as number\_of\_movies

  from `jkdb.imdb`

  where released\_year between 1920 and 2020

  group by decade, genre

),

most\_popular\_genre as (

  select

    decade,

    genre,

    number\_of\_movies,

    rank() over (partition by decade order by number\_of\_movies desc) as rn

  from decade\_genre\_counts

)

select

  decade,

  genre as PopularGenereByMovies,

  number\_of\_movies

from most\_popular\_genre

where rn = 1

order by decade;

#Most populat genre in every decade between 1920 to 2020 according to average rating

with decade\_genre\_avg as (

  select

    case

      when released\_year between 1919 and 1929 then 1920

      when released\_year between 1930 and 1939 then 1930

      when released\_year between 1940 and 1949 then 1940

      when released\_year between 1950 and 1959 then 1950

      when released\_year between 1960 and 1969 then 1960

      when released\_year between 1970 and 1979 then 1970

      when released\_year between 1980 and 1989 then 1980

      when released\_year between 1990 and 1999 then 1990

      when released\_year between 2000 and 2009 then 2000

      when released\_year between 2010 and 2021 then 2010

    end as decade,

    genre,

    round(avg(IMDB\_Rating), 2) as rating

  from `jkdb.imdb`

  group by decade, genre

  order by decade

),

most\_popular\_genre as (

  select

    decade,

    genre,

    rating,

    rank() over (partition by decade order by rating desc) as rn

  from decade\_genre\_avg

)

select

  decade,

  genre as PopularGenereByMovies,

  rating

from most\_popular\_genre

where rn = 1

order by decade;

**#3 analysis:-**

#earning varies with ratings and boxoffice earnings

with ratings\_range as(

  select

    case

      when IMDB\_Rating between 5.0 and 5.9 then 5

      when IMDB\_Rating between 6.0 and 6.9 then 6

      when imdb\_rating between 7.0 and 7.9 then 7

      when imdb\_rating between 8.0 and 8.9 then 8

      when imdb\_rating between 9.0 and 9.9 then 9

    end as rating,

    genre,

    round(avg(gross),2) as AverageEarning

  from `jkdb.imdb`

  group by rating,genre

  order by rating

),

max\_rated\_genre as(

  select genre,AverageEarning,rating,

  rank() over(partition by rating order by AverageEarning) as rn

  from ratings\_range

)

select

  genre,

  rating,

  AverageEarning

from

  max\_rated\_genre

where rn=1

order by AverageEarning;

#corelation analysis of imdb ratings and earings

with required\_data as (

  select imdb\_rating, gross

  from `jkdb.imdb`

),

AveragesInData as (

  select

    round(avg(imdb\_rating), 2) as average\_rating,

    round(avg(gross), 2) as average\_earning

  from required\_data

),

VariancesInData as (

  select

    round(variance(imdb\_rating), 2) as var\_rating,

    round(variance(gross), 2) as var\_earning

  from required\_data

),

StandardDeviations as (

  select

    round(stddev(imdb\_rating), 2) as std\_rating,

    round(stddev(gross), 2) as std\_earning

  from required\_data

)

select

  'Rating Avg' as metric,

  avg.average\_rating as ratings,

  avg.average\_earning as gross

from AveragesInData avg

union all

select

  'Rating Var' as metric,

  var.var\_rating as ratings,

  var.var\_earning as gross

from VariancesInData var

union all

select

  'Rating Std' as metric,

  std.std\_rating as ratings,

  std.std\_earning as gross

from StandardDeviations std;

**#4th Analysis:-**

#Duration of the movie vs rating

select runtime,imdb\_rating from `jkdb.imdb` order by Runtime;

#Duration of the movie vs Gross earning

select runtime,gross from `jkdb.imdb` order by Runtime;

#Runtime vs avegage rating and runtime vs average earnings.

select

  runtime,

  round(avg(imdb\_rating),2),

  round(avg(gross),2)

from `jkdb.imdb`

group by

  Runtime

order by

  runtime;

**#5th analysis:-**

#influence of rating and their gross earning on a movie for all stars who worked for the movie

with actor\_ratings\_earnings as (

  select

    star1 as actor,

    imdb\_rating,

    gross

  from `jkdb.imdb`

  union all

  select

    star2 as actor,

    imdb\_rating,

    gross

  from `jkdb.imdb`

  union all

  select

    star3 as actor,

    imdb\_rating,

    gross

  from `jkdb.imdb`

  union all

  select

    star4 as actor,

    imdb\_rating,

    gross

  from `jkdb.imdb`

)

select

  actor,

  round(avg(imdb\_rating), 2) as average\_rating,

  round(avg(gross), 2) as average\_earning,

  count(\*) as movie\_count

from actor\_ratings\_earnings

group by actor

order by average\_rating desc, average\_earning desc;