# Stored Procedure

1. Get the number of movies released in a year. Input year, output count of movies.

Stored Procedure:

DELIMITER %%

CREATE procedure CountOfMovies\_InYear(IN release\_year VARCHAR(25),OUT Number\_of\_movies INT)

BEGIN

DECLARE search\_pattern VARCHAR(255);

SET search\_pattern = CONCAT(release\_year,”-%-%”);

Select count(\*) into Number\_of\_movies from movie\_metadata where release\_date like search\_pattern;

END

%%

Call CountOfMovies\_InYear(1999, @number\_of\_movies);

Select @number\_of\_movies;

A screenshot of a computer

Description automatically generated

1. Get top N movies by average rating with a minimum number of ratings.

Stored Procedure 2:

DELIMITER $$

CREATE PROCEDURE GetTopRatedMovies(

IN `limit` INT,

IN min\_ratings INT

)

BEGIN

SELECT m.title, AVG(r.rating) AS average\_rating, COUNT(r.rating) AS rating\_count

FROM movie\_metadata m

INNER JOIN ratings r ON m.tmdb\_id = r.movie\_id

GROUP BY m.title

HAVING rating\_count >= min\_ratings

ORDER BY average\_rating DESC

LIMIT `limit`;

END

$$

call GetTopRatedMovies(2,5);

Screenshot 2:

A screenshot of a computer

Description automatically generated

1. Top 5 Collaborations of Director and Actors based on count

DELIMITER //

CREATE PROCEDURE display\_top\_collaborations(IN director\_name VARCHAR(255))

BEGIN

SELECT

casts.name AS 'Cast Name',

GROUP\_CONCAT(title) AS 'Collaborated Movies',

COUNT(\*) AS 'Total Number of Collaborations'

FROM

casts

JOIN

movie\_casts ON casts.credit\_id = movie\_casts.credit\_id

JOIN

movie\_crews ON movie\_casts.tmdb\_id = movie\_crews.tmdb\_id

JOIN

crews ON crews.credit\_id = movie\_crews.credit\_id

JOIN

movie\_metadata ON movie\_metadata.tmdb\_id = movie\_crews.tmdb\_id

WHERE

crews.name = director\_name

AND department = 'Directing'

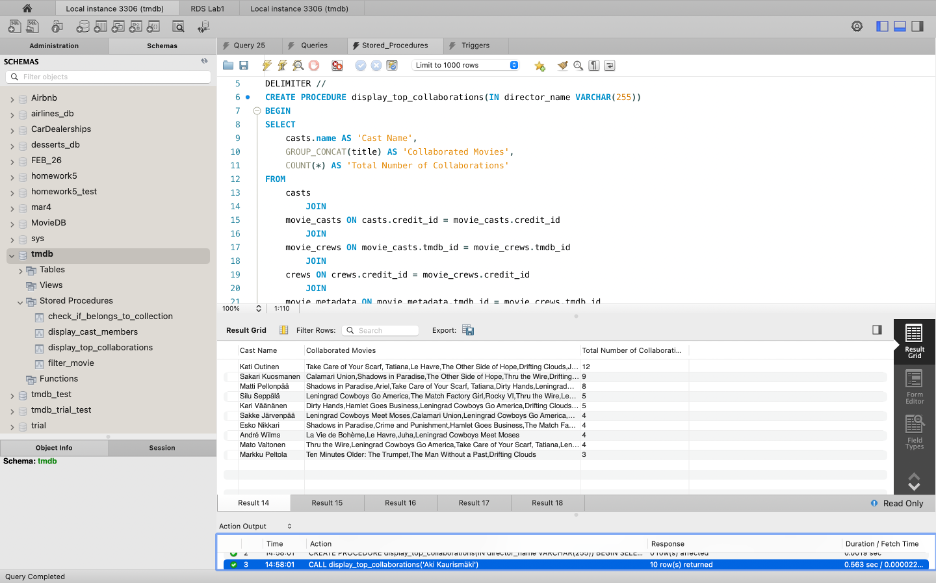
GROUP BY casts.name

ORDER BY COUNT(\*) DESC

LIMIT 10;

END //

DELIMITER ;



1. Filter the movies by genre, release year and language

DELIMITER //

CREATE PROCEDURE filter\_movie(IN genre\_name VARCHAR(255), IN language\_code VARCHAR(255), IN release\_year YEAR, IN lim int)

BEGIN

SELECT

original\_title AS 'Movie Title', release\_date AS 'Release Date', overview AS 'Synopsis', runtime AS 'Runtime'

FROM

movie\_metadata

JOIN

movie\_genres ON movie\_genres.tmdb\_id = movie\_metadata.tmdb\_id

JOIN

genres ON genres.genres\_id = movie\_genres.genres\_id

WHERE

genres.name = genre\_name

AND movie\_metadata.original\_language = language\_code

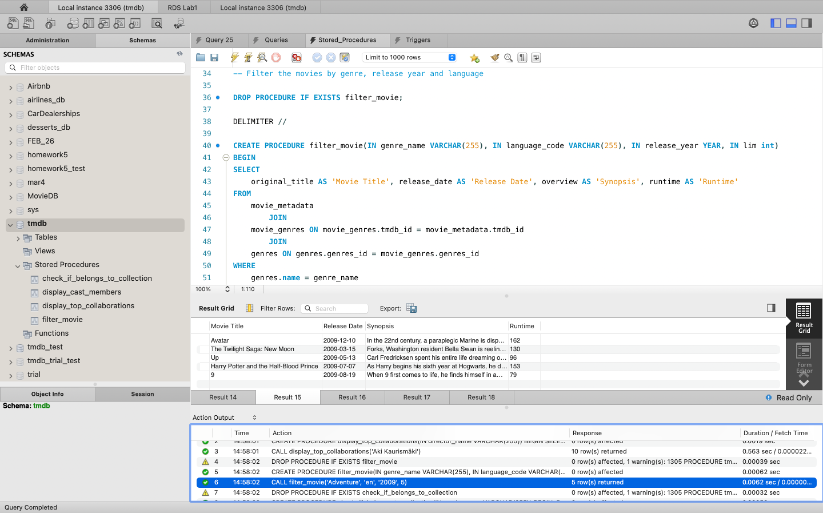
AND YEAR(movie\_metadata.release\_date) = release\_year

ORDER BY popularity desc

LIMIT lim;

END //

DELIMITER ;



1. Check if a movie belongs to any collection

DELIMITER //

CREATE PROCEDURE check\_if\_belongs\_to\_collection(IN movie\_name VARCHAR(255))

BEGIN

DECLARE collection\_name VARCHAR(255);

SELECT belongs\_to\_collection.name INTO collection\_name

FROM belongs\_to\_collection

INNER JOIN movie\_belongs\_to\_collection ON belongs\_to\_collection.belongs\_to\_collection\_id = movie\_belongs\_to\_collection.belongs\_to\_collection\_id

join movie\_metadata on movie\_belongs\_to\_collection.tmdb\_id = movie\_metadata.tmdb\_id

WHERE movie\_metadata.title = movie\_name;

IF collection\_name IS NOT NULL THEN

SELECT CONCAT(movie\_name, ' belongs to collection: ', collection\_name) AS result;

ELSE

SELECT CONCAT(movie\_name, ' does not belong to any collection') AS result;

END IF;

END //

DELIMITER ;

