

Hive Query Editor

Query: Database default Type text

```
3: SELECT * FROM gold_media_avaliacao_por_filme ORDER BY media_avaliacao DESC LIMIT 10;
2: SELECT * FROM gold_media_avaliacao_por_genero ORDER BY media_avaliacao DESC;
3:
```

WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.

Tables Statement 2/2

	gold_media_avaliacao_por_filme.movie_id	gold_media_avaliacao_por_filme.titulo_filme	gold_media_avaliacao_por_filme.media_avaliacao	gold_media_avaliacao_por_filme.total_avaliacoes
1	101	Bottle Rocket (1996)	5	1
2	160	Congo (1995)	5	1
3	131724	The Jinx: The Life and Deaths of Robert Durst (2015)	5	1
4	125	Flirting With Disaster (1996)	5	1
5	162	Crumb (1994)	5	1
6	22	Copycat (1995)	5	1
7	86	White Squall (1996)	5	1
8	176	Living in Oblivion (1995)	5	1
9	166528	Rogue One: A Star Wars Story (2016)	5	1
10	5	Father of the Bride Part II (1995)	5	1

Ativar o Windows
Acesse Configurações para ativar o Windows.

Hive Query Editor

Query: Database default Type text

```
2.13s Database default Type text
3: SELECT * FROM gold_media_avaliacao_por_filme ORDER BY media_avaliacao DESC LIMIT 10;
2: SELECT * FROM gold_media_avaliacao_por_genero ORDER BY media_avaliacao DESC;
3:
```

WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.

Tables Statement 2/2

	gold_media_avaliacao_por_genero.genero	gold_media_avaliacao_por_genero.media_avaliacao	gold_media_avaliacao_por_genero.total_avaliacoes
1	The (Mad Max 2) (1981)*	5	2
2	An (1986)*	5	1
3	White Cat (Orna macka	5	1
4	An (2009)*	5	1
5	Longer and Uncut (1999)*	5	1
6	The (1938)*	5	1
7	Vietnam (1987)*	5	1
8	The (1940)*	5	1
9	The (1952)*	5	1
10	The (1954)*	5	1
11	The (1968)*	5	1
12	The (1970)*	5	1
13	The (1985)*	5	1

Ativar o Windows
Acesse Configurações para ativar o Windows.

 Query ▾

  

◀  default

Tables (11) + ↴

Filter...

- bronze_links
 - movieid (string)
 - imdbid (string)
 - tmdbid (string)
- bronze_movies
 - movieid (string)
 - title (string)
 - genres (string)
- bronze_ratings
 - user_id (string)
 - movie_id (string)
 - rating (string)
 - date_timestamp (string)
- bronze_tags
 - userid (string)
 - movieid (string)
 - tag (string)
 - date_timestamp (string)
- gold_filme_genero_detalhado
- gold_media_avaliacao_por_filme
- gold_media_avaliacao_por_genero
- silver_links
 - movie_id (int)
 - imdb_id (string)
 - tmdb_id (string)
- silver_movies
 - movie_id (int)
 - title (string)
 - genres (string)
- silver_ratings
 - user_id (int)
 - movie_id (int)
 - rating (int)
 - date_timestamp (timestamp)

?

A

»

 Query ▾

  

◀  default

Tables (11) + ↻

Filter...

- bronze_ratings
 - user_id (string)
 - movie_id (string)
 - rating (string)
 - date_timestamp (string)
- bronze_tags
 - userid (string)
 - movieid (string)
 - tag (string)
 - date_timestamp (string)
- gold_filme_genero_detalhado
- gold_media_avaliacao_por_filme
- gold_media_avaliacao_por_genero
- silver_links
 - movie_id (int)
 - imdb_id (string)
 - tmdb_id (string)
- silver_movies
 - movie_id (int)
 - title (string)
 - genres (string)
- silver_ratings
 - user_id (int)
 - movie_id (int)
 - rating (int)
 - date_timestamp (timestamp)
 - year (int) ↗
 - month (int) ↗
- silver_tags
 - user_id (int)
 - movie_id (int)
 - tag (string)
 - date_timestamp (timestamp)
 - year (int) ↗
 - month (int) ↗

	gold_filme_genero_detalhado
	movie_id (int)
	title (string)
	genre (string)
	imdb_id (string)
	tmdb_id (string)
	gold_media_avaliacao_por_filme
	movie_id (int)
	titulo_filme (string)
	media_avaliacao (double)
	total_avaliacoes (bigint)
	gold_media_avaliacao_por_genero
	genero (string)

```

8 horas atrás   !    INSERT OVERWRITE TABLE silver_ratings PARTITION (year, month) SELECT CAST(TRIM(user_id) AS INT), CAST(TRIM(movie_id) AS INT),
CAST(TRIM(rating) AS INT), FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT)), YEAR(FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT))),
MONTH(FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT))) FROM bronze_ratings LIMIT 2000;
8 horas atrás   !    SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict;
8 horas atrás   !    SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict;
8 horas atrás   !    CREATE TABLE IF NOT EXISTS silver_ratings ( user_id INT, movie_id INT, rating INT, date_timestamp TIMESTAMP )
PARTITIONED BY (year INT, month INT) STORED AS PARQUET;
8 horas atrás   !    SELECT DISTINCT YEAR(FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT))) AS year FROM bronze_ratings;
SELECT DISTINCT MONTH(FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT))) AS month FROM bronze_ratings;
8 horas atrás   !    SELECT user_id, movie_id, rating, date_timestamp, CAST(TRIM(user_id) AS INT), CAST(TRIM(movie_id) AS INT), CAST(TRIM(rating) AS INT),
FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT)) AS rating_date FROM bronze_ratings LIMIT 10;
8 horas atrás   !    DESCRIBE bronze_ratings; SELECT * FROM bronze_ratings LIMIT 10; DESCRIBE bronze_movies; SELECT * FROM bronze_movies LIMIT 10;
DESCRIBE bronze_links; SELECT * FROM bronze_links LIMIT 10; DESCRIBE bronze_tags; SELECT * FROM bronze_tags LIMIT 10;
8 horas atrás   !    DROP TABLE IF EXISTS bronze_ratings; CREATE EXTERNAL TABLE IF NOT EXISTS bronze_ratings ( user_id STRING, movie_id STRING, rating STRING,
date_timestamp STRING ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
LOCATION '/user/admin/datasets/movieLens/bronze/ratings' TBLPROPERTIES ("skip.header.line.count"="1");
8 horas atrás   !    DROP TABLE IF EXISTS bronze_ratings; CREATE EXTERNAL TABLE IF NOT EXISTS bronze_ratings ( user_id STRING, movie_id STRING, rating STRING,
date_timestamp STRING ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
LOCATION '/user/admin/datasets/movieLens/bronze/ratings' TBLPROPERTIES ("skip.header.line.count"="1");
8 horas atrás   !    CREATE EXTERNAL TABLE IF NOT EXISTS bronze_tags ( userId STRING, movieId STRING, tag STRING, date_timestamp STRING ) ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/admin/datasets/movieLens/bronze/tags'
TBLPROPERTIES ("skip.header.line.count"="1");
8 horas atrás   !    CREATE EXTERNAL TABLE IF NOT EXISTS bronze_links ( movieId STRING, imbdId STRING, tmbdId STRING ) ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/admin/datasets/movieLens/bronze/links'
TBLPROPERTIES ("skip.header.line.count"="1");
8 horas atrás   !    CREATE EXTERNAL TABLE IF NOT EXISTS bronze_movies ( movieId STRING, title STRING, genres STRING ) ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/admin/datasets/movieLens/bronze/movies'
TBLPROPERTIES ("skip.header.line.count"="1");
8 horas atrás   !    CREATE EXTERNAL TABLE IF NOT EXISTS bronze_ratings ( movieId STRING, title STRING, genres STRING ) ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/admin/datasets/movieLens/bronze/movies'
TBLPROPERTIES ("skip.header.line.count"="1");
8 horas atrás   !    DROP TABLE IF EXISTS bronze_ratings; CREATE EXTERNAL TABLE IF NOT EXISTS bronze_ratings ( user_id STRING, movie_id STRING, rating STRING,
date_timestamp STRING ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
LOCATION '/user/admin/datasets/movieLens/bronze/ratings' TBLPROPERTIES ("skip.header.line.count"="1");
8 horas atrás   !    CREATE DATABASE IF NOT EXISTS movielens; USE movielens;
8 horas atrás   !    CREATE DATABASE IF NOT EXISTS movielens; USE movielens;
8 horas atrás   !    CREATE DATABASE IF NOT EXISTS movielens; USE movielens;
8 horas atrás   !    CREATE DATABASE IF NOT EXISTS movielens; USE movielens;

```

Ativar o Wi
Acesso Convidado

```

    FROM silver_movies m LATERAL VIEW explode(split(m.genres, '\\"|\\\"')) g AS genre ) exploded LEFT JOIN silver_links l
    ON exploded.movie_id = l.movie_id;

6 horas atrás ! SELECT exploded.movieid, exploded.title, exploded.genre, l.imdbid, l.tmdbid FROM ( SELECT m.movieid, m.title, g.genre FROM silver_movies m
LATERAL VIEW explode(split(m.genres, '\\"|\\\"')) g AS genre ) exploded LEFT JOIN silver_links l ON exploded.movieid = l.movieid;

7 horas atrás ! TRUNCATE TABLE silver_movies; SELECT * From bronze_movies WHERE movieid =11;

7 horas atrás ! TRUNCATE TABLE silver_movies; SELECT * From bronze_links WHERE movieid =11;

7 horas atrás ! SELECT DISTINCT YEAR(date_timestamp) AS ano FROM silver_tags ORDER BY ano;

7 horas atrás ! SELECT COUNT(*) FROM silver_tags; SELECT DISTINCT year, month FROM silver_tags ORDER BY year, month;

7 horas atrás ! INSERT OVERWRITE TABLE silver_tags PARTITION (year, month) SELECT CAST(TRIM(userid) AS INT), CAST(TRIM(movieid) AS INT), tag,
FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT)), YEAR(FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT))), MONTH(FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT))) FROM bronze_tags LIMIT 2000;

7 horas atrás ! SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict;

7 horas atrás ! SET hive.exec.dynamic.partition=true; SET hive.exec.dynamic.partition.mode=nonstrict;

7 horas atrás ! CREATE TABLE IF NOT EXISTS silver_tags ( user_id INT, movie_id INT, tag STRING, date_timestamp TIMESTAMP )
PARTITIONED BY (year INT, month INT) STORED AS PARQUET;

7 horas atrás ! DROP TABLE IF EXISTS silver_tags;

7 horas atrás ! SELECT DISTINCT YEAR(date_timestamp) AS ano FROM silver_tags ORDER BY ano;

7 horas atrás ! CREATE TABLE IF NOT EXISTS silver_links ( movie_id INT, imdb_id STRING, tmbd_id STRING ) STORED AS PARQUET;
INSERT OVERWRITE TABLE silver_links SELECT CAST(TRIM(movieid) AS INT), imdbId, tmdbId FROM bronze_links;

7 horas atrás ! CREATE TABLE IF NOT EXISTS silver_links ( movie_id INT, imdb_id STRING, tmbd_id STRING ) STORED AS PARQUET;
INSERT OVERWRITE TABLE silver_links SELECT CAST(TRIM(movieid) AS INT), imdbId, tmdbId FROM bronze_links;

7 horas atrás ! SELECT * FROM silver_movies LIMIT 10;

7 horas atrás ! INSERT OVERWRITE TABLE silver_movies SELECT CAST(TRIM(movieId) AS INT), title, genres FROM bronze_movies;

7 horas atrás ! CREATE TABLE IF NOT EXISTS silver_movies ( movie_id INT, title STRING, genres STRING ) STORED AS PARQUET;

7 horas atrás ! CREATE TABLE IF NOT EXISTS silver_movies ( movie_id INT, title STRING, genres STRING ) STORED AS PARQUET;
INSERT OVERWRITE TABLE silver_movies SELECT CAST(TRIM(movieId) AS INT), title, genres FROM bronze_movies;

7 horas atrás ! INSERT OVERWRITE TABLE silver_ratings PARTITION (year, month) SELECT CAST(TRIM(userid) AS INT), CAST(TRIM(movie_id) AS INT),
CAST(TRIM(rating) AS INT), FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT)), YEAR(FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT))), MONTH(FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT))) FROM bronze_ratings LIMIT 2000; SELECT COUNT(*) FROM silver_ratings;
SELECT DISTINCT year, month FROM silver_ratings ORDER BY year, month;

7 horas atrás ! INSERT OVERWRITE TABLE silver_ratings PARTITION (year, month) SELECT CAST(TRIM(userid) AS INT), CAST(TRIM(movie_id) AS INT),
CAST(TRIM(rating) AS INT), FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT)), YEAR(FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT))), MONTH(FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT))) FROM bronze_ratings LIMIT 2000; SELECT COUNT(*) FROM silver_ratings;
SELECT DISTINCT year, month FROM silver_ratings ORDER BY year, month;

8 horas atrás ! INSERT OVERWRITE TABLE silver_ratings PARTITION (year, month) SELECT CAST(TRIM(userid) AS INT), CAST(TRIM(movie_id) AS INT),
CAST(TRIM(rating) AS INT), FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT)), YEAR(FROM_UNIXTIME(CAST(TRIM(date_timestamp) AS BIGINT))),
```

Ativar o Win
Acessar configura

Query History	Saved Queries	Query Builder	Results (100+)
2 minutos atrás ✓			SELECT * FROM gold_media_avaliacao_por_genero ORDER BY media_avaliacao DESC
2 minutos atrás ✓			SELECT * FROM gold_media_avaliacao_por_filme ORDER BY media_avaliacao DESC LIMIT 10
6 horas atrás ✓			SELECT * FROM gold_media_avaliacao_por_filme ORDER BY media_avaliacao DESC LIMIT 10; SELECT * FROM gold_media_avaliacao_por_genero ORDER BY media_avaliacao DESC;
6 horas atrás ✓			SELECT * FROM gold_media_avaliacao_por_filme ORDER BY media_avaliacao DESC LIMIT 10; SELECT * FROM gold_media_avaliacao_por_genero ORDER BY media_avaliacao DESC;
6 horas atrás ✓			SELECT * FROM gold_media_avaliacao_por_filme ORDER BY media_avaliacao DESC LIMIT 10; SELECT * FROM gold_media_avaliacao_por_genero ORDER BY media_avaliacao DESC;
6 horas atrás ✓			CREATE TABLE IF NOT EXISTS gold_media_avaliacao_por_genero AS SELECT genero, ROUND(AVG(r.rating), 2) AS media_avaliacao, COUNT(*) AS total_avaliacoes FROM (SELECT m.movie_id, TRIM(g.genero) AS genero FROM silver_movies m LATERAL VIEW explode(split(m.genres, '\\" \\\"')) g AS genero) times_genero JOIN silver_ratings r ON times_genero.movie_id = r.movie_id GROUP BY genero;
6 horas atrás ✓			CREATE TABLE IF NOT EXISTS gold_media_avaliacao_por_filme AS SELECT m.movie_id, m.title AS titulo_filme, ROUND(AVG(r.rating), 2) AS media_avaliacao, COUNT(*) AS total_avaliacoes FROM silver_ratings r JOIN silver_movies m ON r.movie_id = m.movie_id GROUP BY m.movie_id, m.title;
6 horas atrás ✓			DESCRIBE gold_filme_genero_detalhado; SELECT * FROM gold_filme_genero_detalhado;
6 horas atrás ✓			DESCRIBE gold_filme_genero_detalhado; SELECT * FROM gold_filme_genero_detalhado;
6 horas atrás ✓			CREATE TABLE IF NOT EXISTS gold_filme_genero_detalhado AS SELECT exploded.movie_id, exploded.title, exploded.genre, l.imdb_id, l.tmbd_id FROM (SELECT m.movie_id, m.title, g.genre FROM silver_movies m LATERAL VIEW explode(split(m.genres, '\\" \\\"')) g AS genre) exploded LEFT JOIN silver_links l ON exploded.movie_id = l.movie_id; SHOW TABLES;
6 horas atrás ✓			CREATE TABLE IF NOT EXISTS gold_filme_genero_detalhado AS SELECT exploded.movie_id, exploded.title, exploded.genre, l.imdb_id, l.tmbd_id FROM (SELECT m.movie_id, m.title, g.genre FROM silver_movies m LATERAL VIEW explode(split(m.genres, '\\" \\\"')) g AS genre) exploded LEFT JOIN silver_links l ON exploded.movie_id = l.movie_id; SHOW TABLES;
6 horas atrás !			CREATE TABLE IF NOT EXISTS gold_filme_genero_detalhado AS SELECT exploded.movie_id, exploded.title, exploded.genre, l.imdb_id, l.tmbd_id FROM (SELECT m.movie_id, m.title, g.genre FROM silver_movies m LATERAL VIEW explode(split(m.genres, '\\" \\\"')) g AS genre) exploded LEFT JOIN silver_links l ON exploded.movie_id = l.movie_id;
6 horas atrás !			CREATE TABLE IF NOT EXISTS gold_filme_genero_detalhado AS SELECT exploded.movieid, exploded.title, exploded.genre, l.imdbid, l.tmbdid FROM (SELECT m.movieid, m.title, g.genre FROM silver_movies m LATERAL VIEW explode(split(m.genres, '\\" \\\"')) g AS genre) exploded LEFT JOIN silver_links l ON exploded.movieid = l.movieid; DESCRIBE gold_filme_genero_detalhado; SELECT * FROM gold_filme_genero_detalhado;
6 horas atrás !			CREATE TABLE IF NOT EXISTS gold_filme_genero_detalhado AS SELECT exploded.movie_id, exploded.title, exploded.genre, l.imdb_id, l.tmbd_id LEFT JOIN silver_links l ON exploded.movie_id = l.movie_id;
6 horas atrás !			SELECT exploded.movie_id, exploded.title, exploded.genre, l.imdb_id, l.tmbd_id FROM (SELECT m.movie_id, m.title, g.genre FROM silver_movies m LATERAL VIEW explode(split(m.genres, '\\" \\\"')) g AS genre) exploded LEFT JOIN silver_links l ON exploded.movie_id = l.movie_id;
6 horas atrás !			SELECT exploded.movieid, exploded.title, exploded.genre, l.imdbid, l.tmbdid FROM (SELECT m.movieid, m.title, g.genre FROM silver_movies m LATERAL VIEW explode(split(m.genres, '\\" \\\"')) g AS genre) exploded LEFT JOIN silver_links l

Ativar o Win
Acessar configura

Exemplos de Queries

bronze_ratings

Query History		Saved Queries		Query Builder	Results (100+)
					bronze_ratings.rating
		bronze_ratings.user_id	bronze_ratings.movie_id		
1	1		1		4.0
2	1		3		4.0
3	1		6		4.0
4	1		47		5.0
5	1		50		5.0
6	1		70		3.0
7	1		101		5.0
8	1		110		4.0

bronze_movies

Query History		Saved Queries		Query Builder	Results (100+)
					bronze_movies.title
		bronze_movies.movieid	bronze_movies.title		
1	1		Toy Story (1995)		
2	2		Jumanji (1995)		
3	3		Grumpier Old Men (1995)		
4	4		Waiting to Exhale (1995)		
5	5		Father of the Bride Part II (1995)		
6	6		Heat (1995)		
7	7		Sabrina (1995)		

bronze_links

Query History		Saved Queries		Query Builder	Results (100+)
					bronze_links.tmdbid
		bronze_links.movieid	bronze_links.imdbid		
1	1		0114709		862
2	2		0113497		8844
3	3		0113228		15602
4	4		0114885		31357
5	5		0113041		11862
6	6		0113277		949
7	7		0114319		11860

bronze_tags

	bronze_tags.userid	bronze_tags.movieid	bronze_tags.tag	brc
grid	1	2	60756	funny
grid	2	2	60756	Highly quotable
grid	3	2	60756	will ferrell
grid	4	2	89774	Boxing story
grid	5	2	89774	MMA
grid	6	2	89774	Tom Hardy
grid	7	2	106782	drugs
grid	8	2	106782	Leonardo DiCaprio

silver_ratings

	ano	Results (1)
grid	1	2000
grid	2	
grid	3	

	ano	Results (17)
grid	1	1996
grid	2	1998
grid	3	1999
grid	4	2000
grid	5	2001
grid	6	2003

silver_tags

Query History Saved Queries Query Builder Results (13)

ano	
1	2006
2	2007
3	2008
4	2009
5	2010

silver_links

Query History Saved Queries Query Builder Results (100+)

	silver_links.movieid	silver_links.imdbid	silver_links.tmdbid
1	2660	44121	10785
2	2659	84156	30168
3	2657	73629	36685
4	2656	48696	9077
5	2655	80000	80784

silver_movies

Query History Saved Queries Query Builder Results (100+)

	silver_movies.title	silver_movies.genres
60	10 Things I Hate About You (1999)	Comedy Romance
23	After Life (Wandafuru raifu) (1998)	Drama Fantasy
99	Alligator (1980)	Action Horror Sci-Fi
86	Analyze This (1999)	Comedy
90	Avalanche (1978)	Action

gold_filme_genero_detalhado

[Query History](#)[Saved Queries](#)[Query Builder](#)[Results \(100+\)](#)

	gold_filme_genero_detalhado.title	gold_filme_g
1	Thing from Another World, The (1951)	Horror
2	Thing from Another World, The (1951)	Sci-Fi
3	It Came from Hollywood (1982)	Comedy
4	It Came from Hollywood (1982)	Documentary
5	Rocky Horror Picture Show, The (1975)	Comedy
6	Rocky Horror Picture Show, The (1975)	Horror