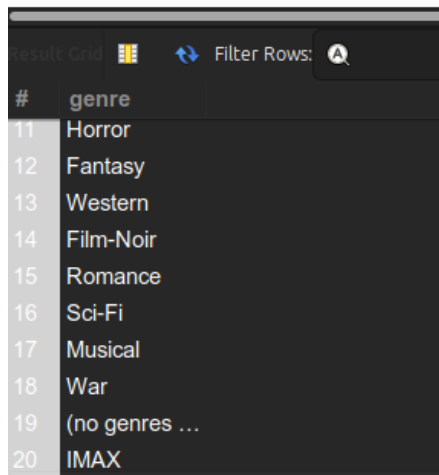


1. Вывести все уникальные жанры, которые встречаются в фильмах. Отдельно найти все фильмы у которых жанр не указан.

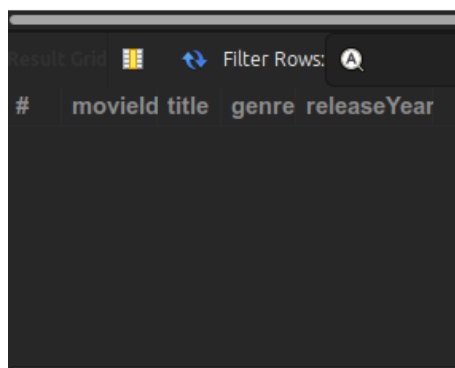
```
1 • SELECT distinct genre
2   from movie;
```



The screenshot shows a database query result grid with a dark theme. At the top, there are icons for 'result Grid', a table icon, and a 'Filter Rows' button with a magnifying glass icon. The table has two columns: '#' and 'genre'. The rows are numbered 11 to 20. The genres listed are Horror, Fantasy, Western, Film-Noir, Romance, Sci-Fi, Musical, War, (no genres ...), and IMAX.

#	genre
11	Horror
12	Fantasy
13	Western
14	Film-Noir
15	Romance
16	Sci-Fi
17	Musical
18	War
19	(no genres ...)
20	IMAX

```
1 • SELECT *
2   from movie
3  where genre is null;
```



The screenshot shows a database query result grid with a dark theme. At the top, there are icons for 'result Grid', a table icon, and a 'Filter Rows' button with a magnifying glass icon. The table has four columns: '#', 'movieId', 'title', and 'genre'. The rows are numbered 1 to 20, but only the first row is visible, showing movieId 1 and title 'The Godfather Part II'.

#	movieId	title	genre
1	1	The Godfather Part II	

2. Создать таблицы и загрузить в них данные из [Data](#) ratings.csv и tags.csv.

```
1 • drop table if exists ta
2 • create table tags(
3     userId int,
4     movieId int,
5     tag text,
6     timestamp double
7 );
```

```
1 • set names utf8;
2 • set global local_infile=true;
3
4 • truncate table tags;
5
6 • load data local infile '/home/user/Documents/BSU_labs/6_sem/database/lab6/Data/tags.csv'
7   into table tags
8     fields terminated by ','
9     optionally enclosed by '"'
10    lines terminated by '\r\n'
11    ignore 1 rows;
```

```
1 • drop table if exists ratings;
2 • create table ratings(
3     userId int,
4     movieId int,
5     rating int,
6     timestamp double
7 );
```

```
1 • set names utf8;
2 • set global local_infile=true;
3
4 • truncate table ratings;
5
6 • load data local infile '/home/user/Documents/BSU_labs/6_sem/database/lab6/Data/ratings.csv'
7   into table ratings
8     fields terminated by ','
9     optionally enclosed by '"'
10    lines terminated by '\r\n'
11    ignore 1 rows;
```

3. Найти фильмы, которые имеют средний рейтинг ≥ 4.0 . Посчитать кол-во оценок, которые были получены этими фильмами.

Найти фильмы, которые имеют средний рейтинг ≤ 2.0 . Посчитать кол-во оценок, которые были получены этими фильмами. Вывести title, releaseYear, avgRate, countMark.

```

1 • select m.title, m.releaseYear, AVG(r.rating) as avgRate, count(r.rating) as countMark
2   from movie m
3  join ratings r on m.movieId = r.movieId
4  group by m.movieId, m.title, m.releaseYear
5  having avgRate >= 4.0;

```

#	title	releaseYear	avgRate	countMark
1	Forrest Gump	1994	4.164133738601824	1316
2	Pulp Fiction	1994	4.197068403908795	1228
3	Fight Club	1999	4.272935779816514	872
4	Silence of the Lambs, The	1991	4.161290322580645	837
5	Matrix, The	1999	4.192446043165468	834
6	Star Wars: Episode IV - A New Hope	1977	4.231075697211155	753
7	Fargo	1996	4.116022099447513	724
8	Braveheart	1995	4.031645569620253	711
9	Princess Bride, The	1987	4.232394366197183	710

```

1 • select m.title, m.releaseYear, AVG(r.rating) as avgRate, count(r.rating) as countMark
2   from movie m
3  join ratings r on m.movieId = r.movieId
4  group by m.movieId, m.title, m.releaseYear
5  having avgRate <= 2.0;

```

#	title	releaseYear	avgRate	countMark
1	Psycho	1998	1.8888888888888888	27
2	NeverEnding Story III, The	1994	2	21
3	Visit, The	2000	2	1
4	Mad Love	1995	1.5	4
5	Stuart Saves His Family	1995	1.4444444444444444	9
6	Super Mario Bros.	1993	2	126
7	Carpool	1996	1.75	4
8	Making Mr. Right	1987	1.8333333333333333	9
9	Godzilla	1998	1.9545454545454546	99

4. Найти фильмы, у которых имеется тег **family**. Узнать средний рейтинг для фильмов с таким тегом.

Вывести title, releaseYear, avgRate.

Найти фильмы, которые содержат в теге слово **bad** или **stupid**.

Вывести title, releaseYear.

```

1 • select m.title, m.releaseYear, AVG(r.rating) as avgRate
2   from movie m
3   inner join tags t on m.movieId = t.movieId
4   inner join ratings r on m.movieId = r.movieId
5   where t.tag = 'family'
6   group by m.movieId, m.title, m.releaseYear;

```

#	title	releaseYear	avgRate
1	Addams Family Values	1993	3.1011904761904763
2	Home Alone 2: Lost in New York	1992	2.5161290322580645
3	Hannah and Her Sisters	1986	3.769230769230769
4	Addams Family, The	1991	3.289473684210526
5	Spanglish	2004	3.3636363636363638
6	Kind Hearts and Coronets	1949	3.3333333333333335
7	You Can Count on Me	2000	4.166666666666667
8	Little Foxes, The	1941	4

5. Найти пользователя(ей), который поставил больше всего оценок фильмам, посчитать среднее значение его оценок.

Вывести userId, countMark, avgMark.

1	2	3	4	5	6	7
1 •	SELECT	userId,	COUNT(*) AS	countMark,	AVG(rating) AS	avgMark
2	FROM	ratings				
3	GROUP BY	userId				
4	ORDER BY	countMark	DESC			
5	LIMIT	1;				
6						
7						

	userId	countMark	avgMark
▶	414	2698	3.4755

6. Написать хранимую процедуру, которая будет принимать в качестве параметров **startYear int**, **finishYear int**, **titleTemplate text**, **genresTemplate** и находить фильмы согласно этим параметрам.

Query 1 x usp_parseMovies

Limit to 1000 rows

```
1 • CALL usp_SearchMovies(2000, 2020, 'action', 'action|thriller');
2 |
3
4
5
6
7
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	movieId	title	releaseYear	genre
▶	5666	Rules of Attraction, The	2002	Comedy
	6946	Looney Tunes: Back in Action	2003	Action
	7450	Laws of Attraction	2004	Comedy
	5666	Rules of Attraction, The	2002	Drama
	6946	Looney Tunes: Back in Action	2003	Animation
	7450	Laws of Attraction	2004	Romance
	5666	Rules of Attraction, The	2002	Romance
	6946	Looney Tunes: Back in Action	2003	Children
	5666	Rules of Attraction, The	2002	Thriller
	6946	Looney Tunes: Back in Action	2003	Fantasy

Написать хранимую процедуру, которая принимает в качестве параметра **title** и считает рейтинг для этого фильма.

Query 1 x usp_parseMovies usp_CalculateMovieRating - Ro...

Limit to 1000 rows

```
1 • call usp_CalculateMovieRating ('Stuck in Love')
2 |
3
4
5
6
7
```

Result Grid | Filter Rows: | Export: | Wrap Cell Con

	movieId	movieTitle	rating
▶	103606	Stuck in Love	4.00
	103606	Stuck in Love	4.00
	103606	Stuck in Love	4.00