

Домашнее задание Джоины и агрегации

- Создание структуры:

```
CREATE DATABASE imdb;
```

```
CREATE TABLE imdb.actors
```

```
(  
    id      UInt32,  
    first_name String,  
    last_name String,  
    gender   FixedString(1)  
) ENGINE = MergeTree ORDER BY (id, first_name, last_name, gender);
```

```
CREATE TABLE imdb.genres
```

```
(  
    movie_id UInt32,  
    genre    String  
) ENGINE = MergeTree ORDER BY (movie_id, genre);
```

```
CREATE TABLE imdb.movies
```

```
(  
    id UInt32,  
    name String,  
    year UInt32,  
    rank Float32 DEFAULT 0  
) ENGINE = MergeTree ORDER BY (id, name, year);
```

```
CREATE TABLE imdb.roles
```

```
(  
    actor_id UInt32,  
    movie_id UInt32,  
    role     String,  
    created_at DateTime DEFAULT now()
```

) **ENGINE** = MergeTree **ORDER BY** (actor_id, movie_id);

- Наполнение данными:

INSERT INTO imdb.actors

SELECT *

FROM s3('https://datasets-documentation.s3.eu-west-3.amazonaws.com/imdb/imdb_ijs_actors.tsv.gz',
'TSVWithNames');

INSERT INTO imdb.genres

SELECT *

FROM s3('https://datasets-documentation.s3.eu-west-3.amazonaws.com/imdb/imdb_ijs_movies_genres.tsv.gz',
'TSVWithNames');

INSERT INTO imdb.movies

SELECT *

FROM s3('https://datasets-documentation.s3.eu-west-3.amazonaws.com/imdb/imdb_ijs_movies.tsv.gz',
'TSVWithNames');

INSERT INTO imdb.roles(actor_id, movie_id, **role**)

SELECT actor_id, movie_id, role

FROM s3('https://datasets-documentation.s3.eu-west-3.amazonaws.com/imdb/imdb_ijs_roles.tsv.gz',
'TSVWithNames');

- Запросы:

--Найти жанры для каждого фильма

```
SELECT  
  
    m.id AS movie_id,  
  
    m.name AS movie_name,  
  
    g.genre  
  
FROM imdb.movies AS m  
  
INNER JOIN imdb.genres AS g ON m.id = g.movie_id  
  
LIMIT 10;
```

The screenshot shows a SQL IDE interface. The top pane contains a SQL query with line numbers 52 to 60. The query is: `--Найти жанры для каждого фильма`, `SELECT`, `m.id AS movie_id,`, `m.name AS movie_name,`, `g.genre`, `FROM imdb.movies AS m`, `INNER JOIN imdb.genres AS g ON m.id = g.movie_id`, `LIMIT 10;`. The bottom pane shows the results of the query, titled 'Результат 1'. It displays a table with 6 rows and 4 columns: 'movie_id', 'movie_name', 'genre', and 'Значение'. The data is as follows:

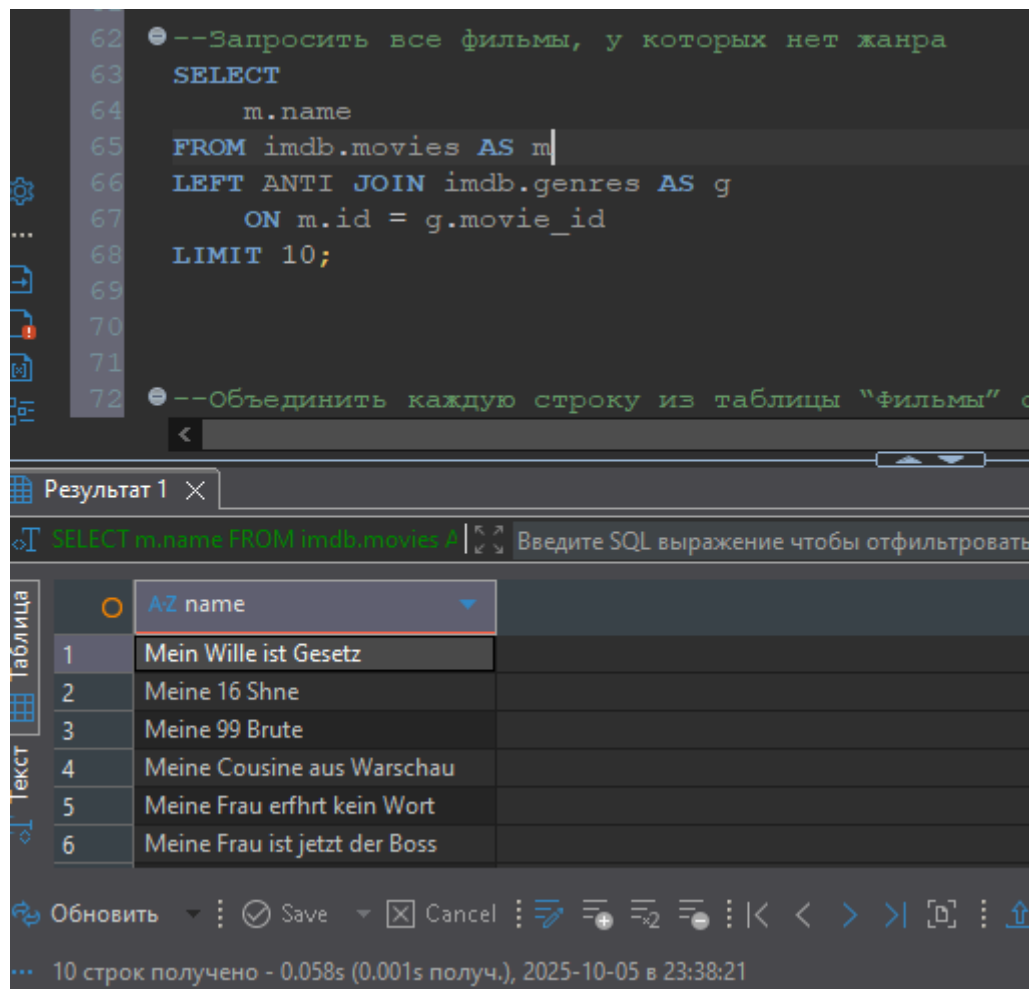
	movie_id	movie_name	genre	Значение
1	154 604	I slik en natt	Thriller	154604
2	154 604	I slik en natt	War	
3	154 605	I snova Aniskin	Crime	
4	154 605	I snova Aniskin	Mystery	
5	154 606	I snova utro	Drama	
6	154 608	I Speak French Like Tarzan	Documentary	

Below the table, there is a status bar showing '10 строк получено - 0.081s, 2025-10-05 в 23:34:58'. At the bottom, there are tabs for 'MSK', 'ru_RU', 'Запись', and 'Инт. вставка'.

--Запросить все фильмы, у которых нет жанра

```
SELECT  
  
    m.id,  
  
    m.name,  
  
    m.year  
  
FROM imdb.movies AS m  
  
LEFT JOIN imdb.genres AS g ON m.id = g.movie_id  
  
WHERE g.movie_id IS NULL
```

LIMIT 10;



--Объединить каждую строку из таблицы "Фильмы" с каждой строкой из таблицы "Жанры"

SELECT

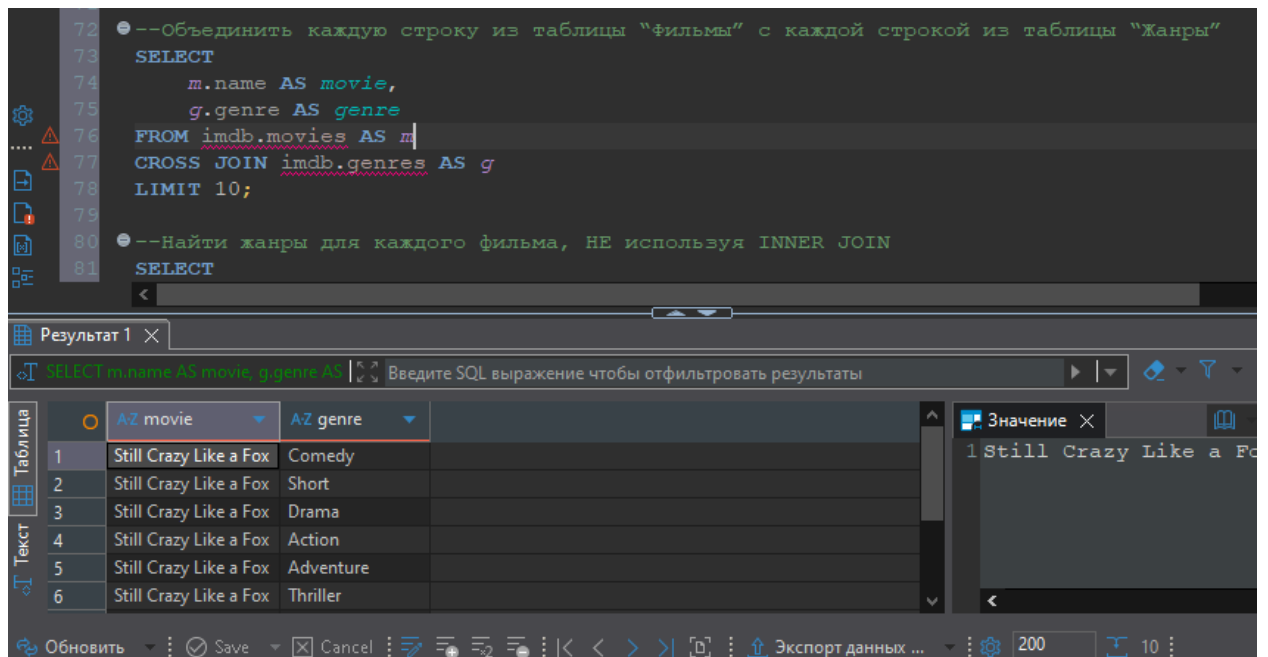
m.name AS movie,

g.genre AS genre

FROM imdb.movies AS *m*

CROSS JOIN imdb.genres AS *g*

LIMIT 10;



--Найти жанры для каждого фильма, НЕ используя INNER JOIN

SELECT

m.id,

m.name,

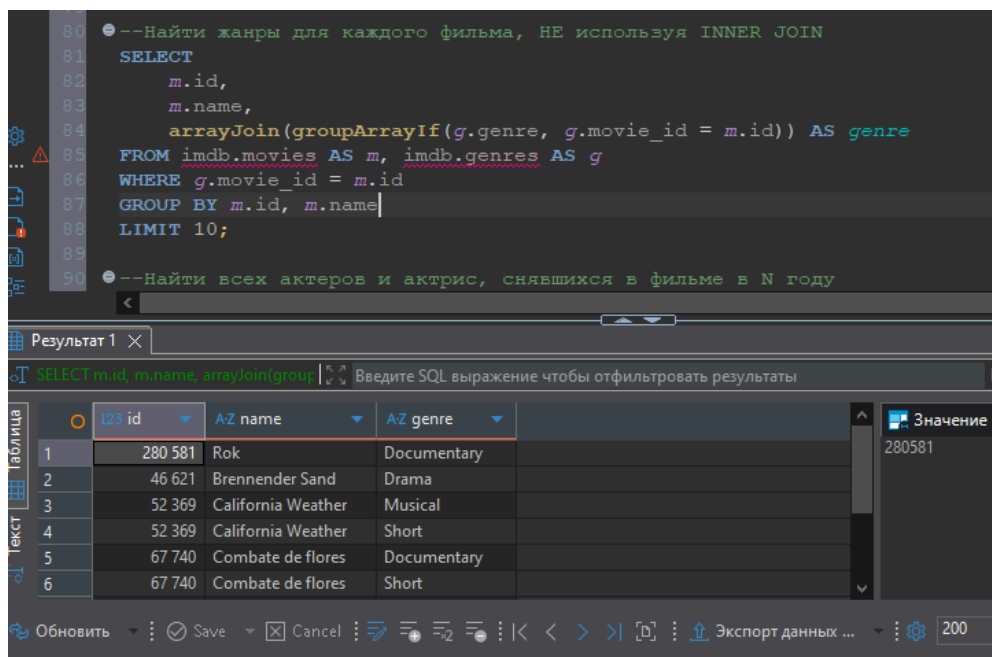
arrayJoin(groupArrayIf(g.genre, g.movie_id = m.id)) AS genre

FROM imdb.movies AS m, imdb.genres AS g

WHERE g.movie_id = m.id

GROUP BY m.id, m.name

LIMIT 10;



--Найти всех актеров и актрис, снявшихся в фильме в N году

SELECT

DISTINCT *a*.first_name,
a.last_name,
a.gender,
m.name **AS** *movie_name*,
m.year

FROM *imdb.roles* **AS** *r*

INNER JOIN *imdb.actors* **AS** *a* **ON** *r*.actor_id = *a*.id

INNER JOIN *imdb.movies* **AS** *m* **ON** *r*.movie_id = *m*.id

WHERE *m*.year = 1994

LIMIT 10;

The screenshot shows a SQL IDE with a query editor at the top and a results pane at the bottom. The query editor contains the following SQL code:

```
--Найти всех актеров и актрис, снявшихся в фильме в N году
SELECT
    DISTINCT a.first_name,
    a.last_name,
    a.gender,
    m.name AS movie_name,
    m.year
FROM imdb.roles AS r
INNER JOIN imdb.actors AS a ON r.actor_id = a.id
INNER JOIN imdb.movies AS m ON r.movie_id = m.id
WHERE m.year = 1994
LIMIT 10;
```

The results pane shows a table with 6 columns: first_name, last_name, gender, movie_name, and year. The first 5 rows of results are displayed:

	first_name	last_name	gender	movie_name	year
1	Herbert	Knaup	M	Unschuldsgengel	1994
2	Mattias	Knave	M	Polismrdaren	1994
3	Mattias	Knave	M	""Solo""	1994
4	Hans Dieter	Knebel	M	Schatten des Schreibers, Der	1994
5	Günther	Knecht	M	Traumstreuner	1994

The bottom of the IDE shows a toolbar with buttons for 'Обновить', 'Save', 'Cancel', and 'Экспорт данных ...'.

--Запросить все фильмы, у которых нет жанра, через ANTI JOIN

SELECT

m.id,

m.name,

m.year

FROM imdb.movies **AS** m

LEFT ANTI JOIN imdb.genres **AS** g **ON** m.id = g.movie_id

LIMIT 10;

The screenshot shows a SQL client interface with a query editor at the top and a results pane below. The query editor contains the following SQL code:

```
103 --Запросить все фильмы, у которых нет жанра, через ANTI JOIN
104 SELECT
105     m.id,
106     m.name,
107     m.year
108 FROM imdb.movies AS m
109 LEFT ANTI JOIN imdb.genres AS g ON m.id = g.movie_id
110 LIMIT 10;
111
```

The results pane shows the results of the query. It has a tab labeled "Результат 1" and a search bar with the text "Введите SQL выражение чтобы отфильтровать результаты". The results are displayed in a table with the following columns: "id", "name", and "year".

	id	name	year
1	35 582	Beware of the Aliens	1 998
2	35 584	Beware of the Dog	1 923
3	35 585	Beware of the Law	1 922
4	35 587	Beware of Widows	1 927
5	35 599	Bewegung der Zeit, Eine	1 987
6	35 600	Bewerbungen	1 996

At the bottom of the interface, there is a status bar that reads: "10 строк получено - 0.043s (0.001s получ.), 2025-10-05 в 23:40:36".