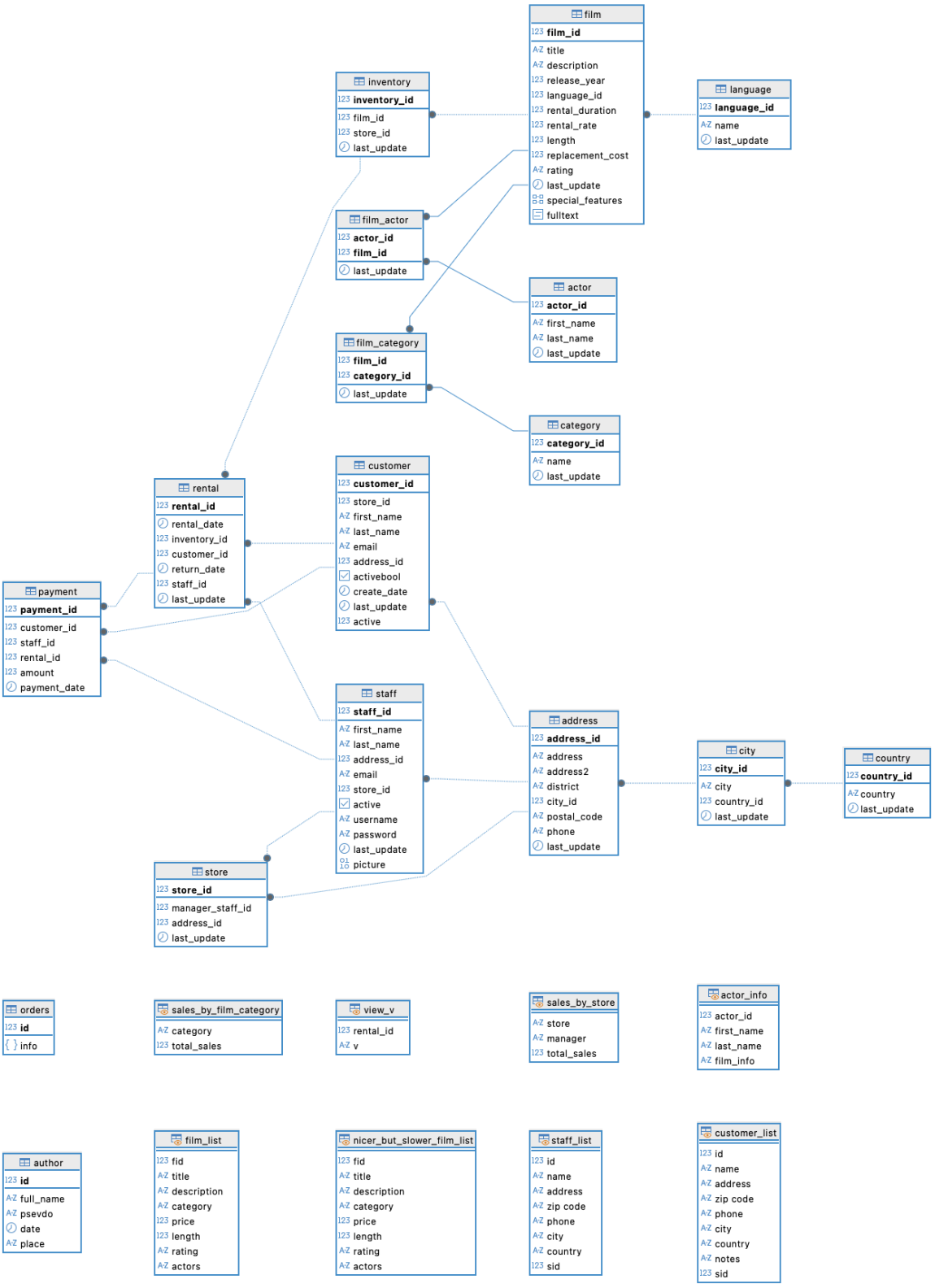
Домашнее задание 3

Подготовка программного обеспечения и базы данных

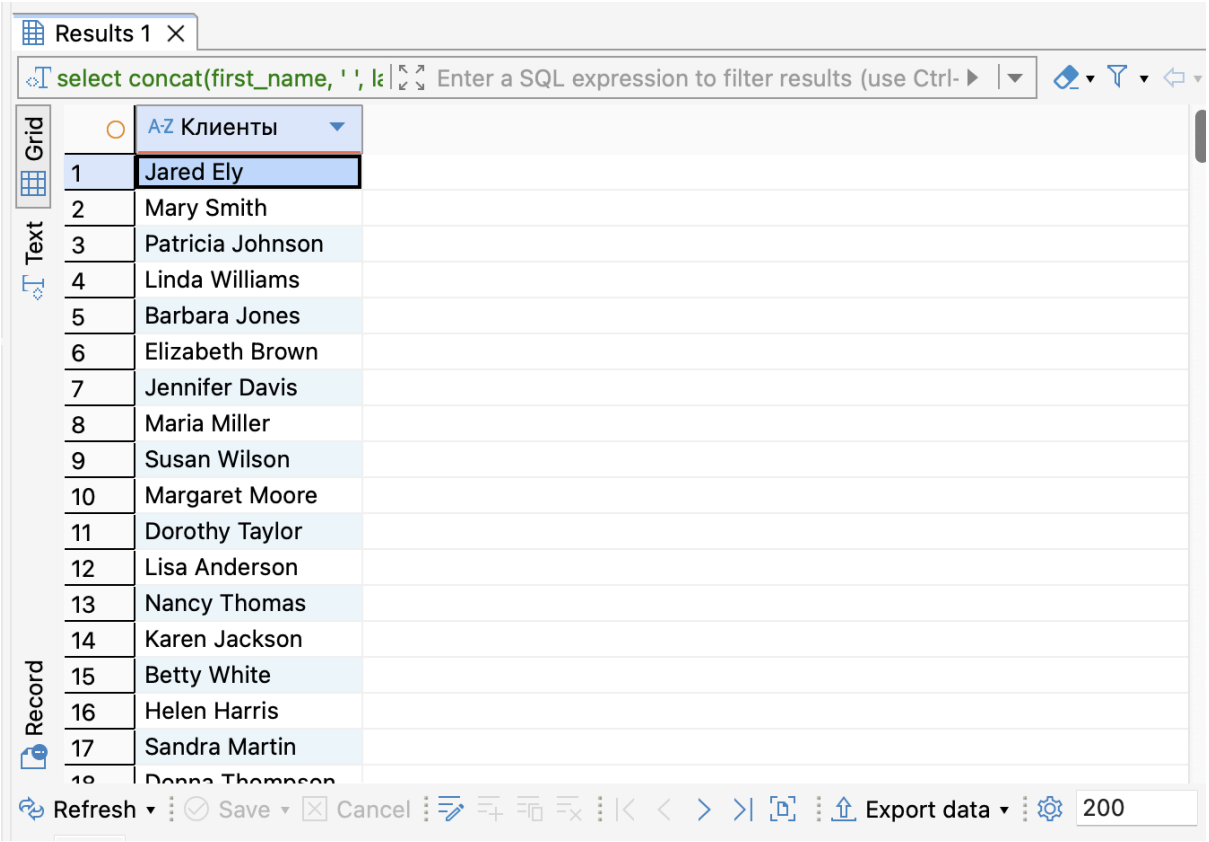
Открыть ER-диаграмму базы данных dvd_rental в DBeaver и сделать скриншот



SQL и получение данных

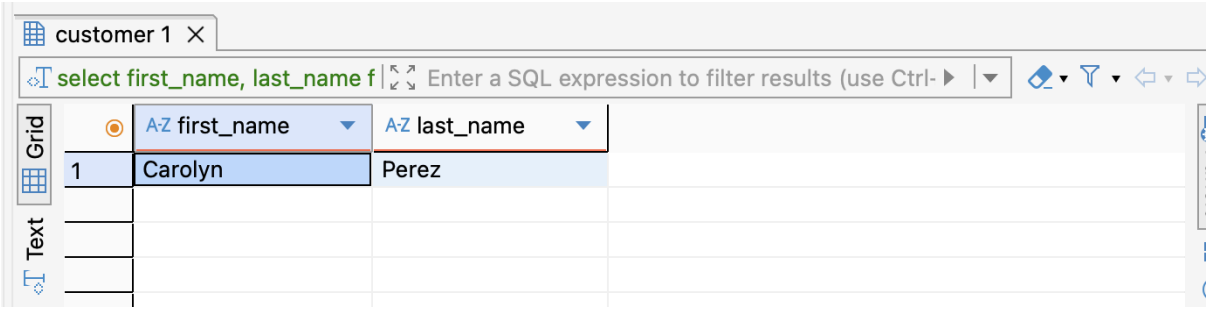
1. Вывести список всех клиентов (таблица customer)

```
select
        concat(first_name, ' ', last_name) Клиенты
from
        customer;
```



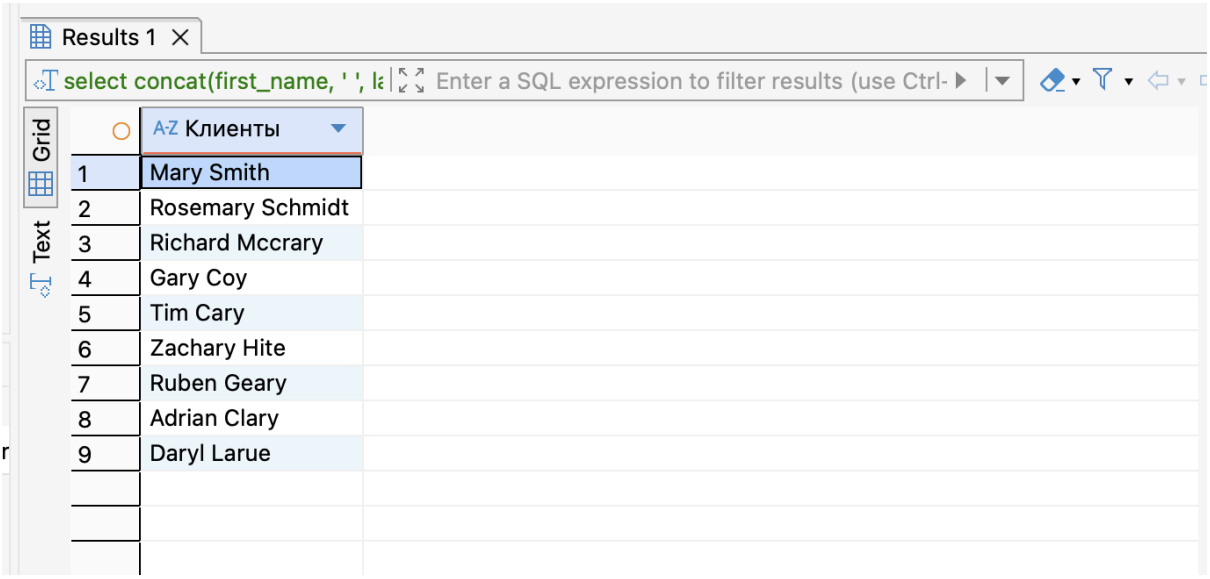
2. Вывести имена и фамилии клиентов с именем Carolyn.

```
select
        first_name,
        last_name
from
        customer
where
        first_name = 'Carolyn';
```



3. Вывести полные имена клиентов (имя + фамилия в одной колонке), у которых имя или фамилия содержат подстроку ary (например: Mary, Geary).

```
select
        concat(first_name, ' ', last_name) Клиенты
from
        customer
where
        (last_name like '%ary%')
        or
        (first_name like '%ary%');
```



4. Вывести 20 самых крупных транзакций (таблица payment).

```
select
        *
from
        payment
order by
        amount desc
limit 20;
```

payment 1

select * from payment order b | Enter a SQL expression to filter results (use Ctrl+Space)

Grid | Text | Record

	payment_id	customer_id	staff_id	rental_id	amount	payme
1	22,650	204	2	15,415	11.99	2007-03
2	29,136	13	2	8,831	11.99	2007-04
3	28,814	592	1	3,973	11.99	2007-04
4	24,553	195	2	16,040	11.99	2007-03-
5	20,403	362	1	14,759	11.99	2007-03
6	24,866	237	2	11,479	11.99	2007-03-
7	23,757	116	2	14,763	11.99	2007-03-
8	28,799	591	2	4,383	11.99	2007-04
9	20,152	336	1	15,073	10.99	2007-03-
10	18,272	544	2	1,434	10.99	2007-02
11	19,764	292	1	12,739	10.99	2007-03-
12	18,290	550	1	3,272	10.99	2007-02-
13	19,815	297	1	12,472	10.99	2007-03
14	19,856	301	1	15,201	10.99	2007-03-
15	20,244	345	1	14,702	10.99	2007-03
16	18,153	511	2	2,966	10.99	2007-02-
17	18,175	516	1	1,718	10.99	2007-02

5. Вывести адреса всех магазинов, используя подзапрос.

```
select
        address
from
        address
where
        address_id in (
        select
                address_id
        from
                store
        );
```

address 1

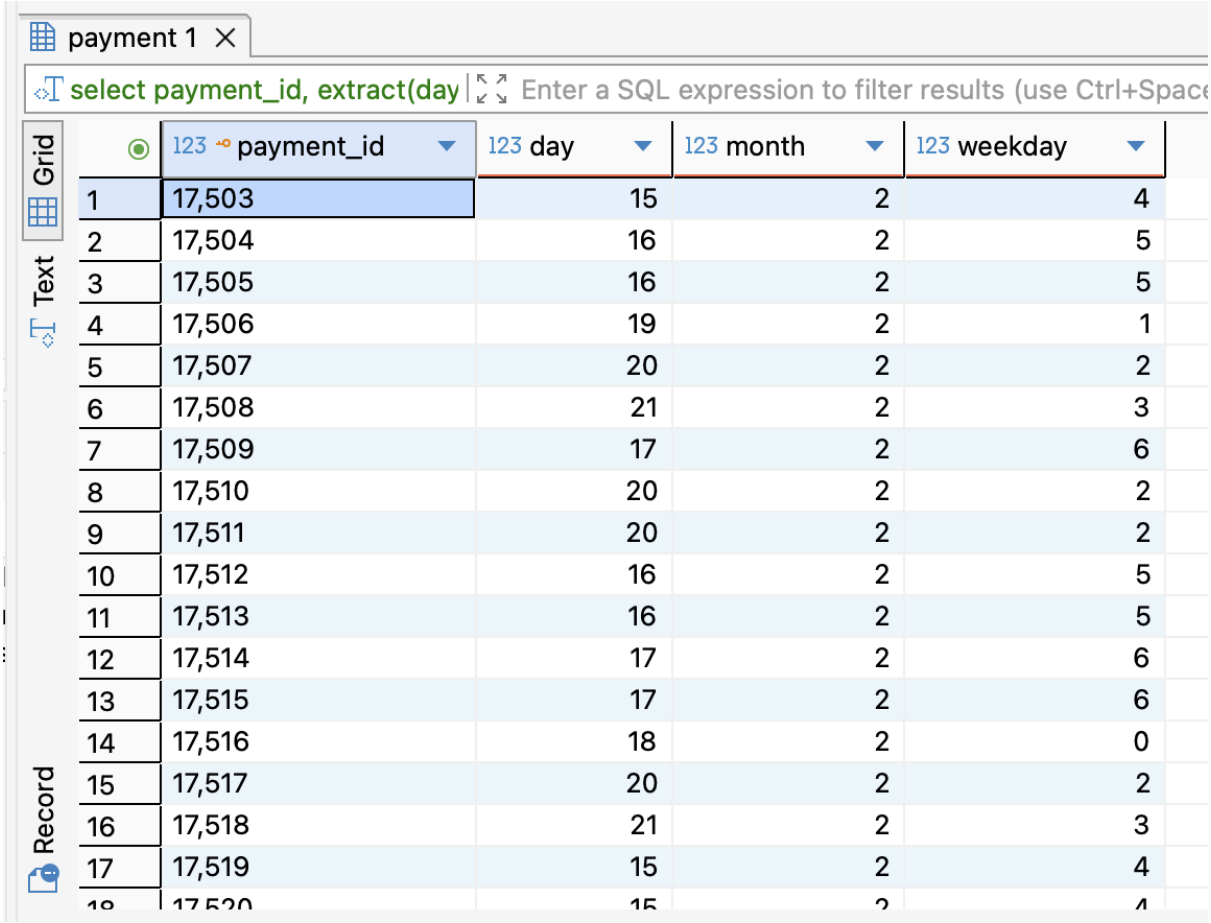
select address from address w | Enter a SQL expression to

Grid | Text

	address
1	47 MySakila Drive
2	28 MySQL Boulevard

6. Для каждой оплаты вывести число, месяц и день недели в числовом формате (Понедельник – 1, Вторник – 2 и т.д.).

```
select
        payment_id,
        extract(day from payment_date) as day,
        extract(month from payment_date) as month,
        extract(dow from payment_date) as weekday
from
        payment;
```



payment 1

select payment_id, extract(day | Enter a SQL expression to filter results (use Ctrl+Space

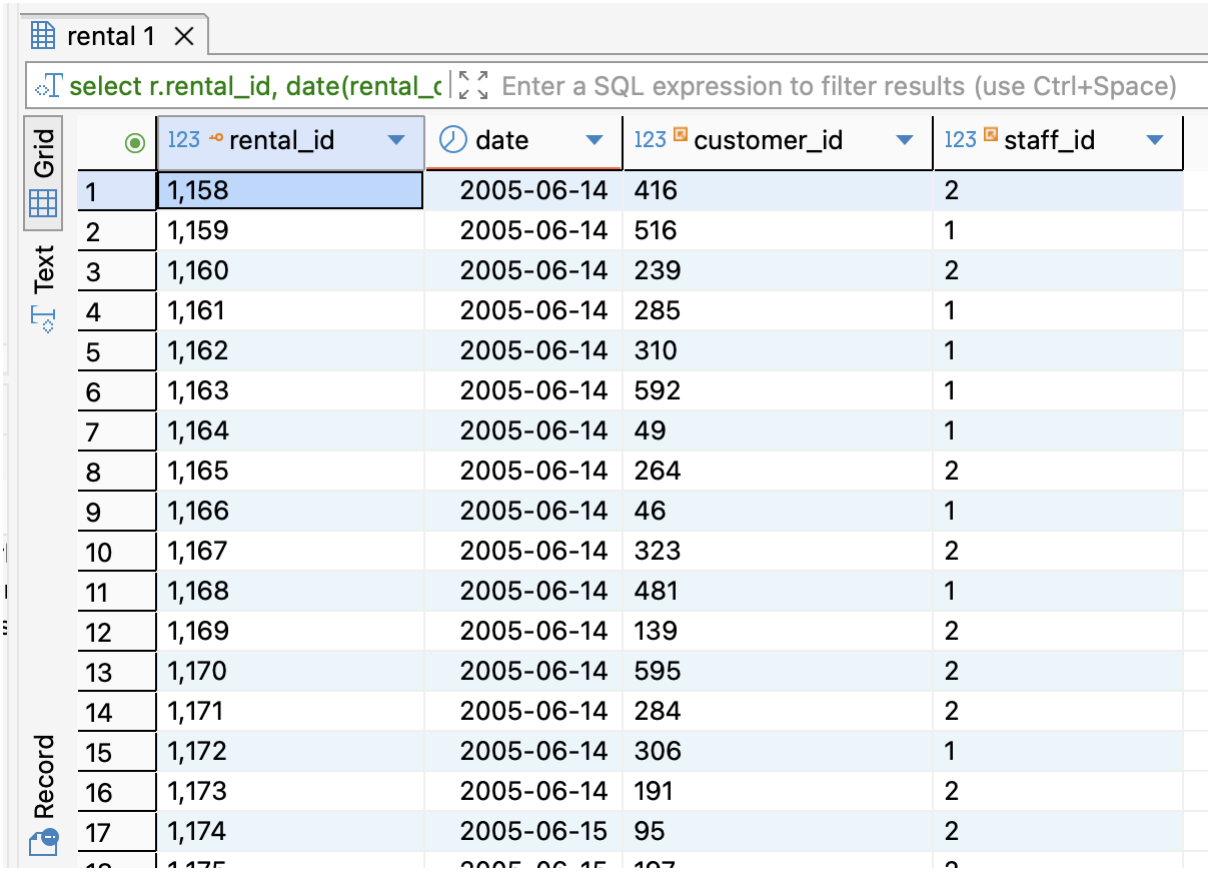
Grid | Text | Record

	payment_id	day	month	weekday
1	17,503	15	2	4
2	17,504	16	2	5
3	17,505	16	2	5
4	17,506	19	2	1
5	17,507	20	2	2
6	17,508	21	2	3
7	17,509	17	2	6
8	17,510	20	2	2
9	17,511	20	2	2
10	17,512	16	2	5
11	17,513	16	2	5
12	17,514	17	2	6
13	17,515	17	2	6
14	17,516	18	2	0
15	17,517	20	2	2
16	17,518	21	2	3
17	17,519	15	2	4

7. Вывести, кто (customer_id), когда (rental_date, приведенная к типу date) и у кого (staff_id) брал диски в аренду в июне 2005 года.

```
select
        r.rental_id,
        date(rental_date),
        customer_id,
        staff_id
from
        rental r
```

```
where
        date_part('year', r.rental_date) = 2005
        and
        date_part('month', r.rental_date) = 6;
```



rental 1

select r.rental_id, date(rental_c | Enter a SQL expression to filter results (use Ctrl+Space)

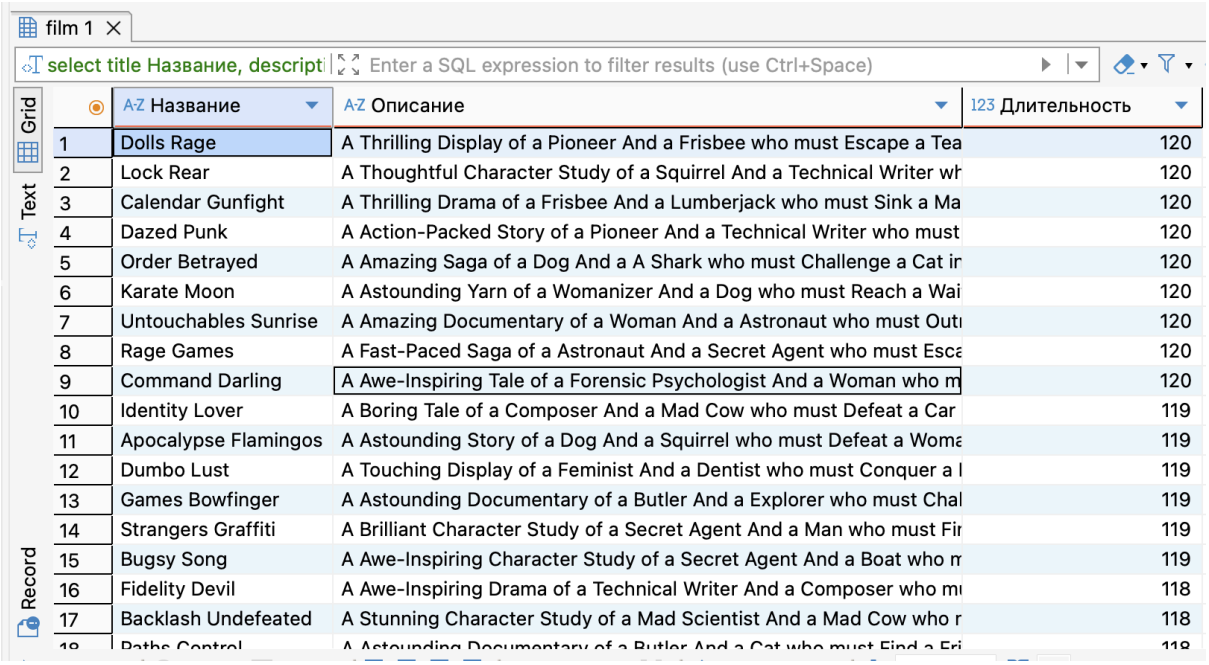
Grid | Text | Record

	rental_id	date	customer_id	staff_id
1	1,158	2005-06-14	416	2
2	1,159	2005-06-14	516	1
3	1,160	2005-06-14	239	2
4	1,161	2005-06-14	285	1
5	1,162	2005-06-14	310	1
6	1,163	2005-06-14	592	1
7	1,164	2005-06-14	49	1
8	1,165	2005-06-14	264	2
9	1,166	2005-06-14	46	1
10	1,167	2005-06-14	323	2
11	1,168	2005-06-14	481	1
12	1,169	2005-06-14	139	2
13	1,170	2005-06-14	595	2
14	1,171	2005-06-14	284	2
15	1,172	2005-06-14	306	1
16	1,173	2005-06-14	191	2
17	1,174	2005-06-15	95	2

8. Вывести название, описание и длительность фильмов (таблица film), выпущенных после 2000 года, с длительностью от 60 до 120 минут включительно. Показать первые 20 фильмов с наибольшей длительностью.

```
select
        title Название,
        description Описание,
        length Длительность
from
        film
where
        release_year >= 2000
        and
        length between 60 and 120
order by
        length desc
```

```
limit 20;
```



film 1

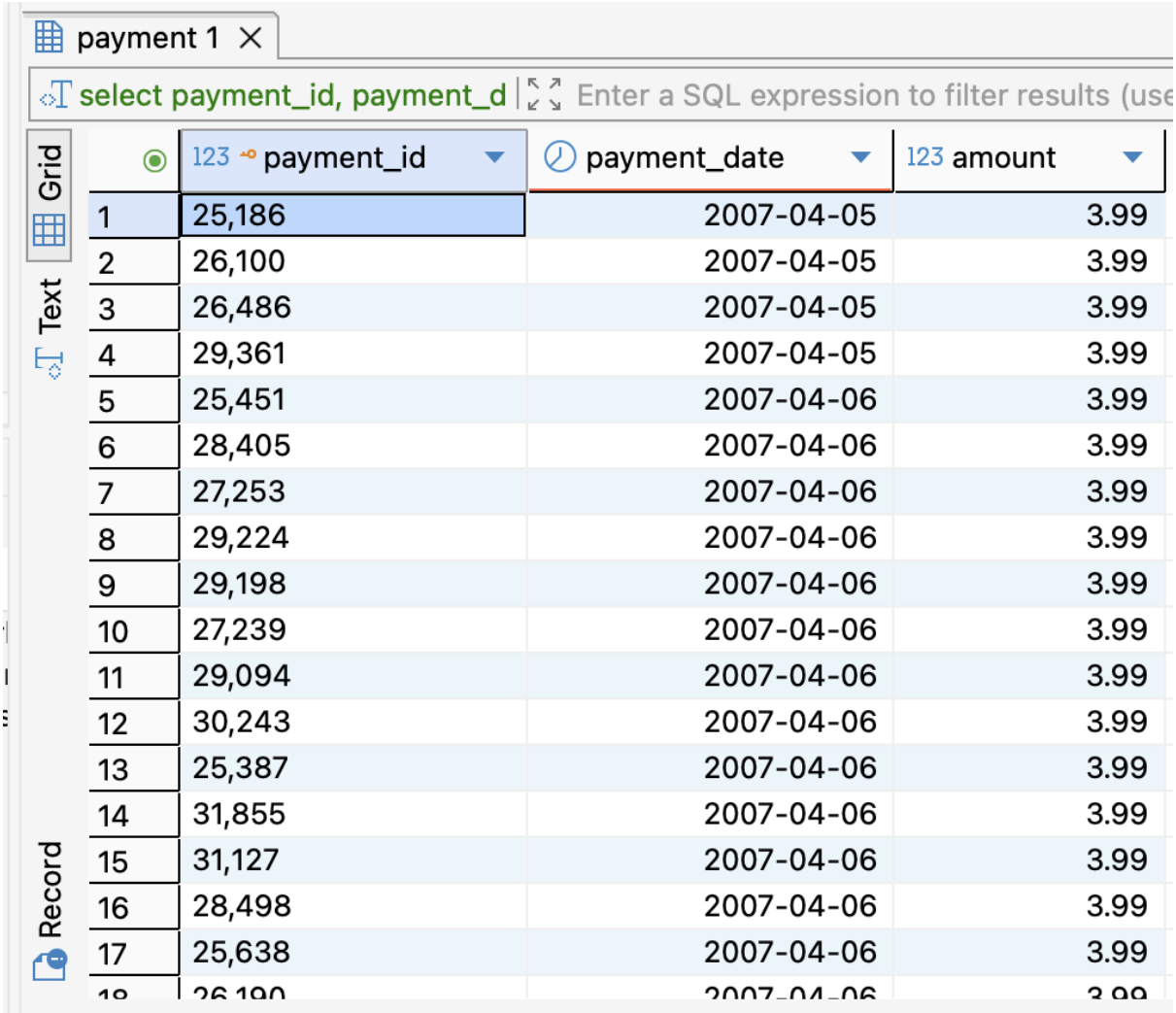
select title Название, descript | Enter a SQL expression to filter results (use Ctrl+Space)

Grid | Text | Record

	Название	Описание	Длительность
1	Dolls Rage	A Thrilling Display of a Pioneer And a Frisbee who must Escape a Tea	120
2	Lock Rear	A Thoughtful Character Study of a Squirrel And a Technical Writer wh	120
3	Calendar Gunfight	A Thrilling Drama of a Frisbee And a Lumberjack who must Sink a Ma	120
4	Dazed Punk	A Action-Packed Story of a Pioneer And a Technical Writer who must	120
5	Order Betrayed	A Amazing Saga of a Dog And a A Shark who must Challenge a Cat in	120
6	Karate Moon	A Astounding Yarn of a Womanizer And a Dog who must Reach a Wai	120
7	Untouchables Sunrise	A Amazing Documentary of a Woman And a Astronaut who must Outı	120
8	Rage Games	A Fast-Paced Saga of a Astronaut And a Secret Agent who must Esca	120
9	Command Darling	A Awe-Inspiring Tale of a Forensic Psychologist And a Woman who m	120
10	Identity Lover	A Boring Tale of a Composer And a Mad Cow who must Defeat a Car	119
11	Apocalypse Flamingos	A Astounding Story of a Dog And a Squirrel who must Defeat a Woma	119
12	Dumbo Lust	A Touching Display of a Feminist And a Dentist who must Conquer a I	119
13	Games Bowfinger	A Astounding Documentary of a Butler And a Explorer who must Chal	119
14	Strangers Graffiti	A Brilliant Character Study of a Secret Agent And a Man who must Fir	119
15	Bugsy Song	A Awe-Inspiring Character Study of a Secret Agent And a Boat who m	119
16	Fidelity Devil	A Awe-Inspiring Drama of a Technical Writer And a Composer who mu	118
17	Backlash Undefeated	A Stunning Character Study of a Mad Scientist And a Mad Cow who r	118
18	Paths Control	A Astounding Documentary of a Butler And a Cat who must Find a Fri	118

9. Найти все платежи (таблица payment), совершенные в апреле 2007 года, стоимость которых не превышает 4 долларов. Вывести идентификатор платежа, дату (без времени) и сумму платежа. Отсортировать платежи по убыванию суммы, а при равной сумме — по более ранней дате.

```
select
        payment_id,
        payment_date::date,
        amount
from
        payment
where
        (payment_date between '2007-04-01'
                and '2007-04-30')
        and
        amount <= 4
order by
        amount desc,
        payment_date asc;
```


payment 1

select payment_id, payment_d | Enter a SQL expression to filter results (use

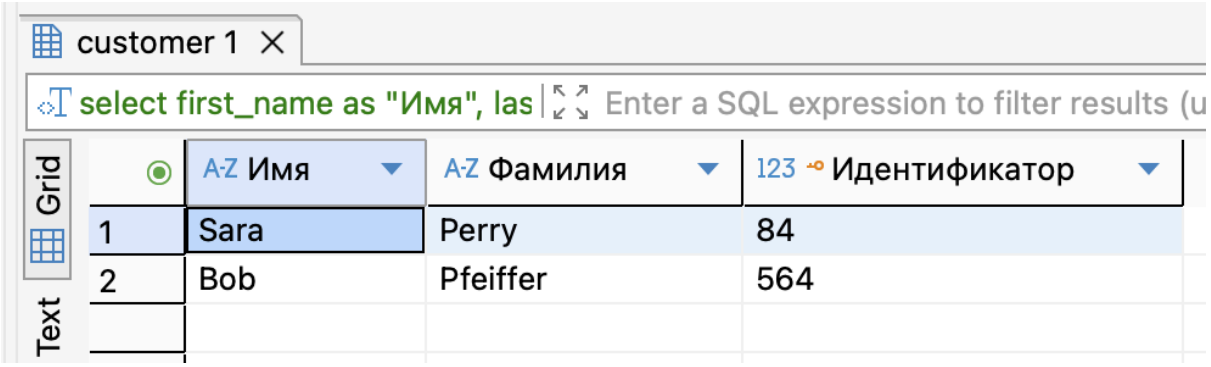
Grid | Text | Record

	payment_id	payment_date	amount
1	25,186	2007-04-05	3.99
2	26,100	2007-04-05	3.99
3	26,486	2007-04-05	3.99
4	29,361	2007-04-05	3.99
5	25,451	2007-04-06	3.99
6	28,405	2007-04-06	3.99
7	27,253	2007-04-06	3.99
8	29,224	2007-04-06	3.99
9	29,198	2007-04-06	3.99
10	27,239	2007-04-06	3.99
11	29,094	2007-04-06	3.99
12	30,243	2007-04-06	3.99
13	25,387	2007-04-06	3.99
14	31,855	2007-04-06	3.99
15	31,127	2007-04-06	3.99
16	28,498	2007-04-06	3.99
17	25,638	2007-04-06	3.99
18	26,190	2007-04-06	3.99

10. Показать имена, фамилии и идентификаторы всех клиентов с именами Jack, Bob или Sara, чья фамилия содержит букву «p». Переименовать колонки: с именем — в «Имя», с идентификатором — в «Идентификатор», с фамилией — в «Фамилия». Отсортировать клиентов по возрастанию идентификатора.

```
select
        first_name as "Имя",
        last_name as "Фамилия",
        customer_id as "Идентификатор"
from
        customer
where
        first_name in ('Jack', 'Bob', 'Sara')
        and
        (last_name like '%p%'
                or last_name like '%P%')
order by
```

```
customer_id asc;
```



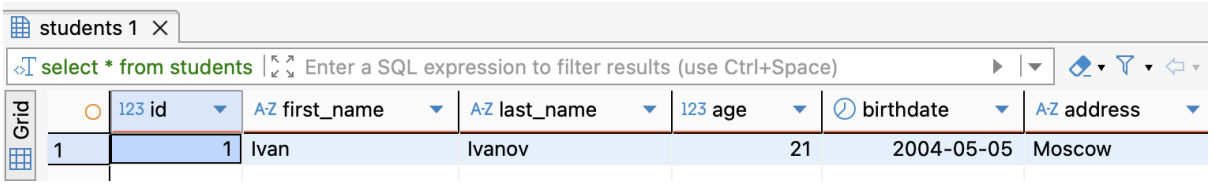
11. Работа с собственной таблицей студентов
    - Создать таблицу студентов с полями: имя, фамилия, возраст, дата рождения и адрес. Все поля должны запрещать внесение пустых значений (NOT NULL).

```
create table students(
id serial primary key not null,
first_name varchar not null,
last_name varchar not null,
age int not null,
birthdate date not null,
address text not null);
```

- Внести в таблицу одного студента с id > 50.

```
insert
	into
	students(first_name, last_name, age, birthdate, address)
values ('Ivan',
'Ivanov',
21,
'2004-05-05',
'Moscow');
select
	*
from
	students;
```

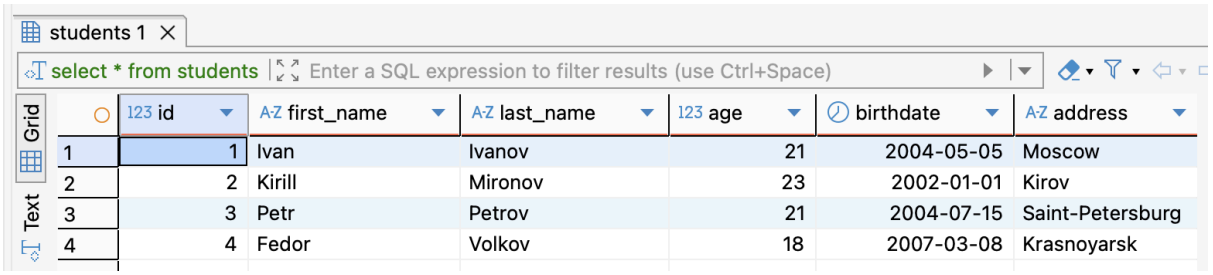
- Просмотреть текущие записи таблицы.



- Внести несколько записей одним запросом, используя автоинкремент id.

```
insert
	into
	students(first_name, last_name, age, birthdate, address)
values
('Kirill',
'Mironov',
23,
'2002-01-01',
'Kirov'),
('Petr',
'Petrov',
21,
'2004-07-15',
'Saint-Petersburg'),
('Fedor',
'Volkov',
18,
'2007-03-08',
'Krasnoyarsk')
;
```

- Снова просмотреть текущие записи таблицы.

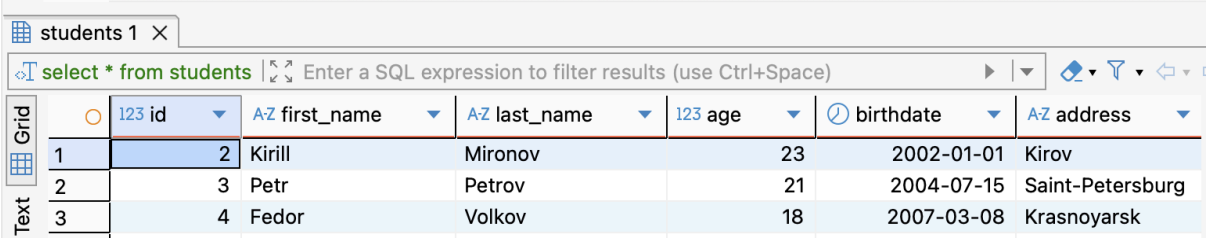


- Удалить одного выбранного студента.

```
delete
from
	students
where
```

```
id = 1;
```

- Вывести полный список студентов.

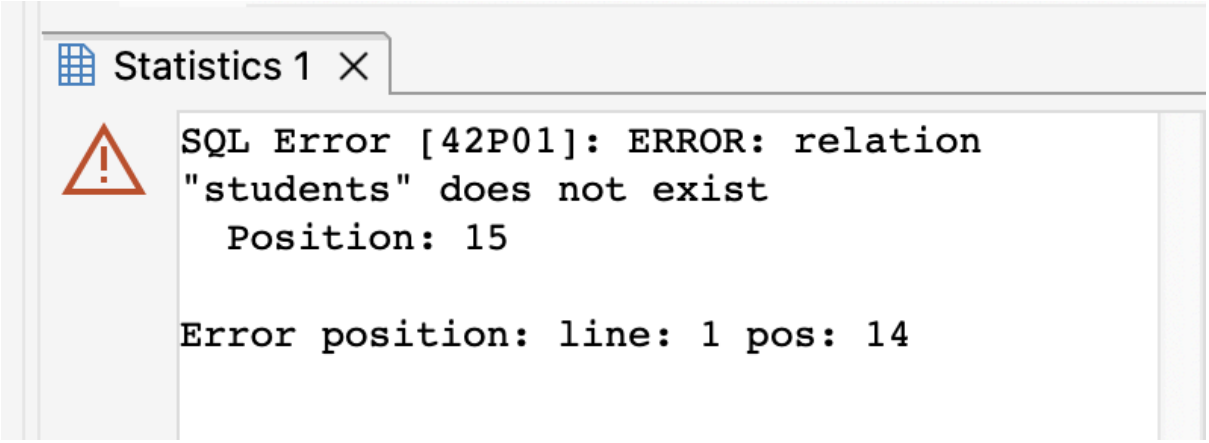


	123 id	A-Z first_name	A-Z last_name	123 age	birthdate	A-Z address
1	2	Kirill	Mironov	23	2002-01-01	Kirov
2	3	Petr	Petrov	21	2004-07-15	Saint-Petersburg
3	4	Fedor	Volkov	18	2007-03-08	Krasnoyarsk

- Удалить таблицу студентов.

```
drop table students;
```

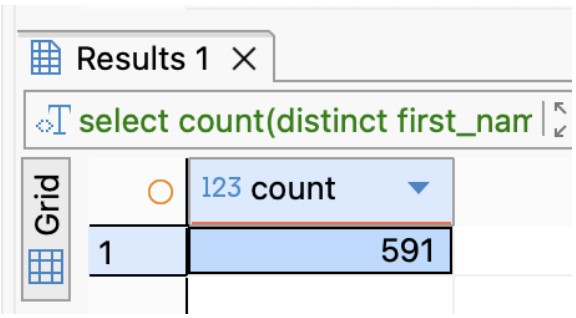
- Выполнить запрос на выборку из таблицы студентов и вывести его результат (показать, что таблица удалена).



JOIN и агрегатные функции

12. Вывести количество уникальных имен клиентов.

```
select
	count(distinct first_name)
from
	customer;
```



	123 count
1	591

13. Вывести 5 самых часто встречающихся сумм оплаты: саму сумму, даты таких оплат, количество платежей с этой суммой и общую сумму этих платежей.

```
with cte as
(
select
	payment.amount as amount,
		count(payment.amount) as most_freq_amount,
		sum(payment.amount) as sum_amount
from
	payment
group by
	payment.amount
)
select
	amount,
	most_freq_amount,
	sum_amount
from
	cte
order by
	most_freq_amount desc
limit 5;
```

payment 1

with cte as (select payment.a | Enter a SQL expression to filter results (use Ctrl+

Grid | Text

	123 amount	123 most_freq_amount	123 sum_amount
1	4.99	3,424	17,085.76
2	2.99	3,233	9,666.67
3	0.99	2,720	2,692.8
4	5.99	1,188	7,116.12
5	6.99	1,017	7,108.83

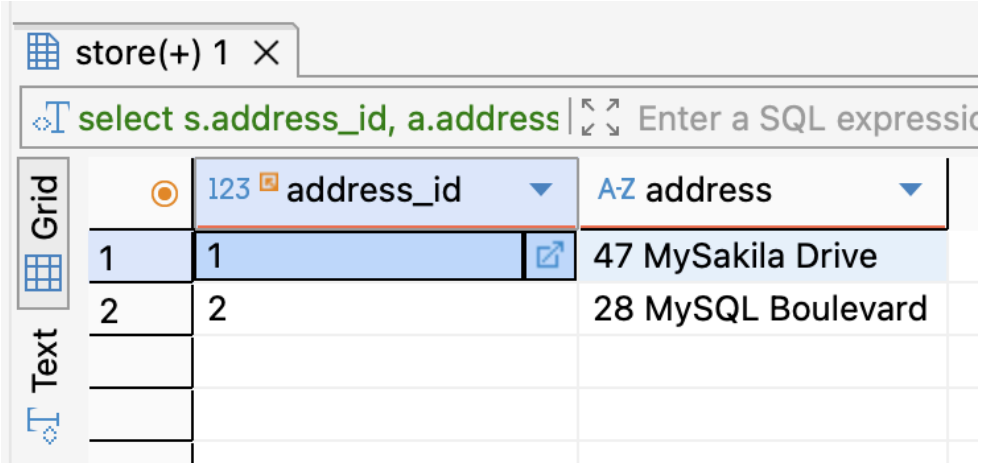
14. Вывести количество ячеек (записей) в инвентаре для каждого магазина.

```
select
	store_id,
	count(inventory_id) as amount
```

```
from
        inventory
group by
        store_id;
```

15. Вывести адреса всех магазинов, используя соединение таблиц (JOIN).

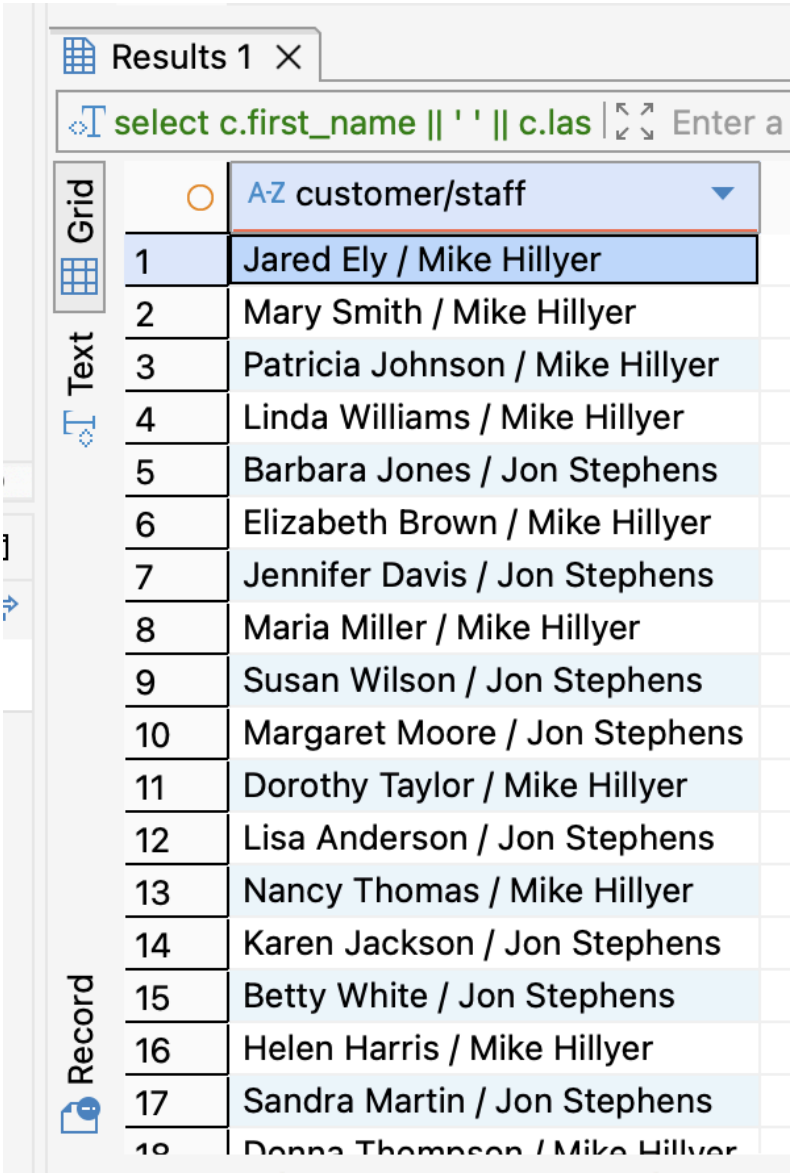
```
select
        s.address_id,
        a.address
from
        address a
inner join store s
  on
        s.address_id = a.address_id;
```



16. Вывести полные имена всех клиентов и всех сотрудников в одну колонку (объединенный список).

```
select
```

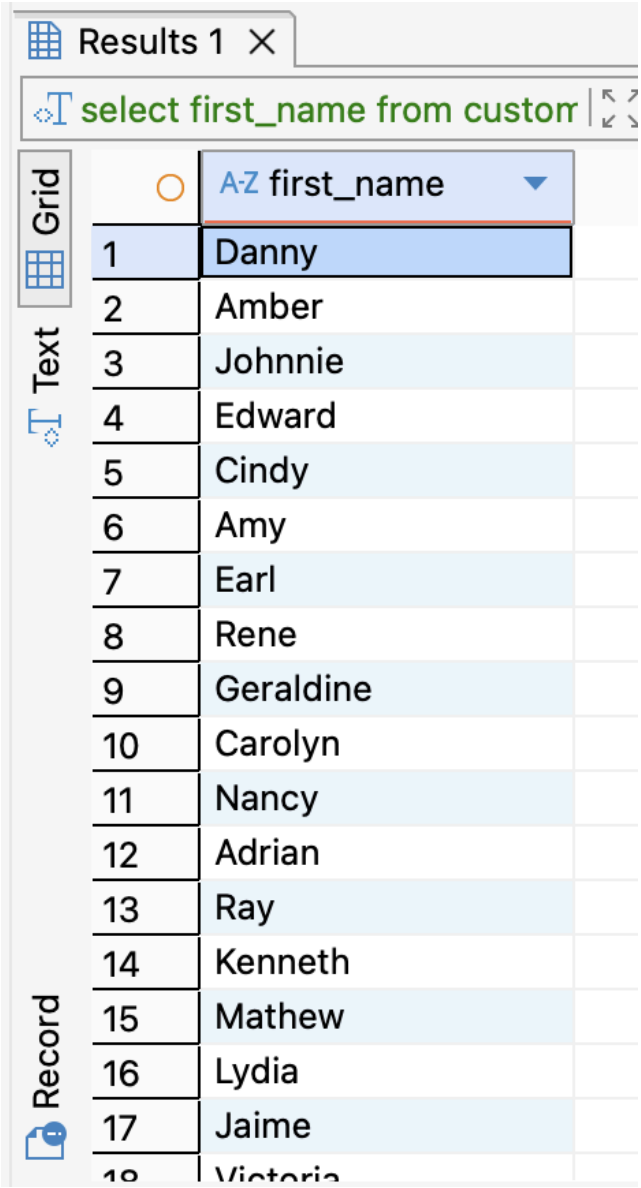
```
        c.first_name || ' ' || c.last_name || ' / ' ||
  s.first_name || ' ' || s.last_name as "customer/staff"
from
        customer c
inner join staff s
  on
        s.store_id = c.store_id;
```



17. Вывести имена клиентов, которые не совпадают ни с одним именем сотрудников (операция EXCEPT или аналог).

```
select
        first_name
from
        customer
except
```

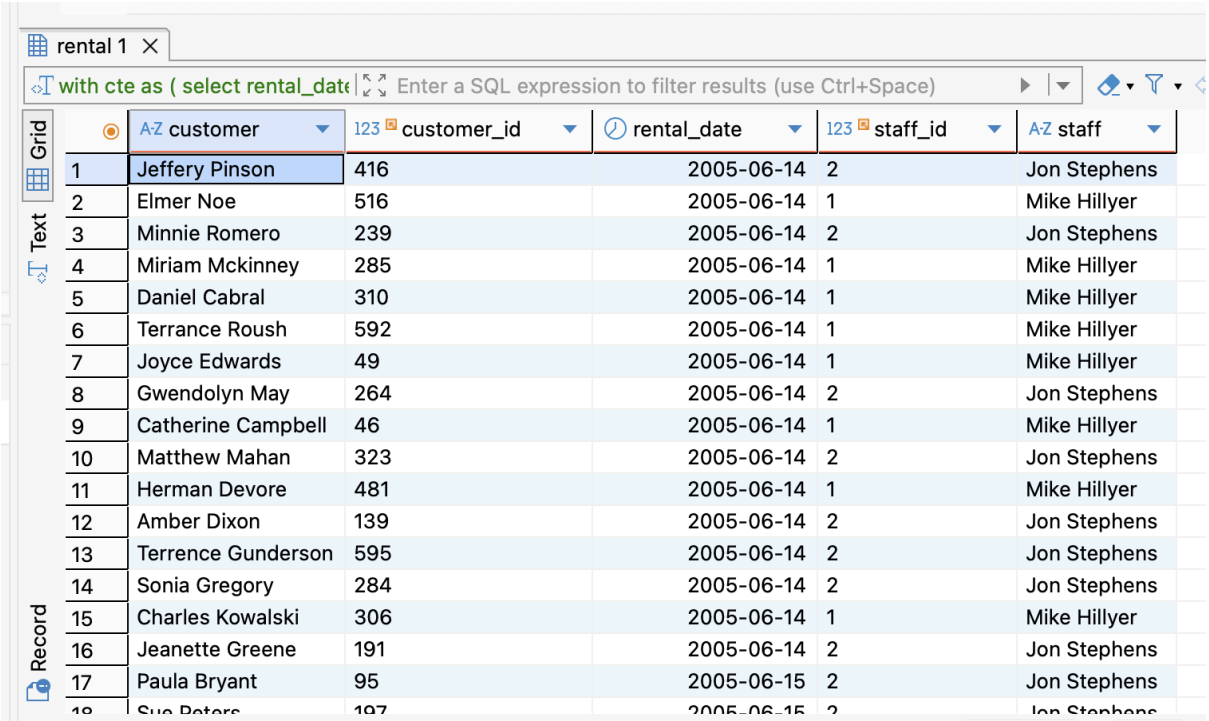
```
   (
select
                first_name
from
        staff);
```



18. Вывести, кто (customer_id), когда (rental_date, приведенная к типу date) и у кого (staff_id) брал диски в аренду в июне 2005 года.

```
with cte as (
select
                rental_date::date,
                customer_id,
                staff_id
from
```

```
        rental
where
        extract(year from rental_date) = 2005
        and extract(month from rental_date) = 06)
select
        c.first_name || ' ' || c.last_name as customer,
        cte.customer_id,
        cte.rental_date,
        cte.staff_id,
        s.first_name || ' ' || s.last_name as staff
from
        cte
inner join customer c
  on
        cte.customer_id = c.customer_id
inner join staff s
  on
        s.staff_id = cte.staff_id;
```

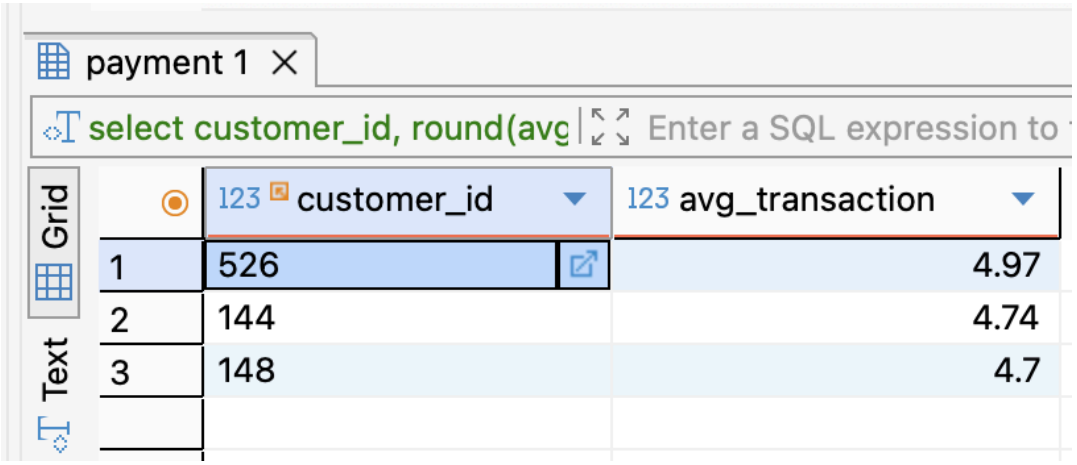


	customer	customer_id	rental_date	staff_id	staff
1	Jeffery Pinson	416	2005-06-14	2	Jon Stephens
2	Elmer Noe	516	2005-06-14	1	Mike Hillyer
3	Minnie Romero	239	2005-06-14	2	Jon Stephens
4	Miriam Mckinney	285	2005-06-14	1	Mike Hillyer
5	Daniel Cabral	310	2005-06-14	1	Mike Hillyer
6	Terrance Roush	592	2005-06-14	1	Mike Hillyer
7	Joyce Edwards	49	2005-06-14	1	Mike Hillyer
8	Gwendolyn May	264	2005-06-14	2	Jon Stephens
9	Catherine Campbell	46	2005-06-14	1	Mike Hillyer
10	Matthew Mahan	323	2005-06-14	2	Jon Stephens
11	Herman Devore	481	2005-06-14	1	Mike Hillyer
12	Amber Dixon	139	2005-06-14	2	Jon Stephens
13	Terrence Gunderson	595	2005-06-14	2	Jon Stephens
14	Sonia Gregory	284	2005-06-14	2	Jon Stephens
15	Charles Kowalski	306	2005-06-14	1	Mike Hillyer
16	Jeanette Greene	191	2005-06-14	2	Jon Stephens
17	Paula Bryant	95	2005-06-15	2	Jon Stephens

19. Вывести идентификаторы всех клиентов, у которых 40 и более оплат. Для каждого такого клиента посчитать средний размер транзакции, округлить его до двух знаков после запятой и вывести в отдельном столбце.

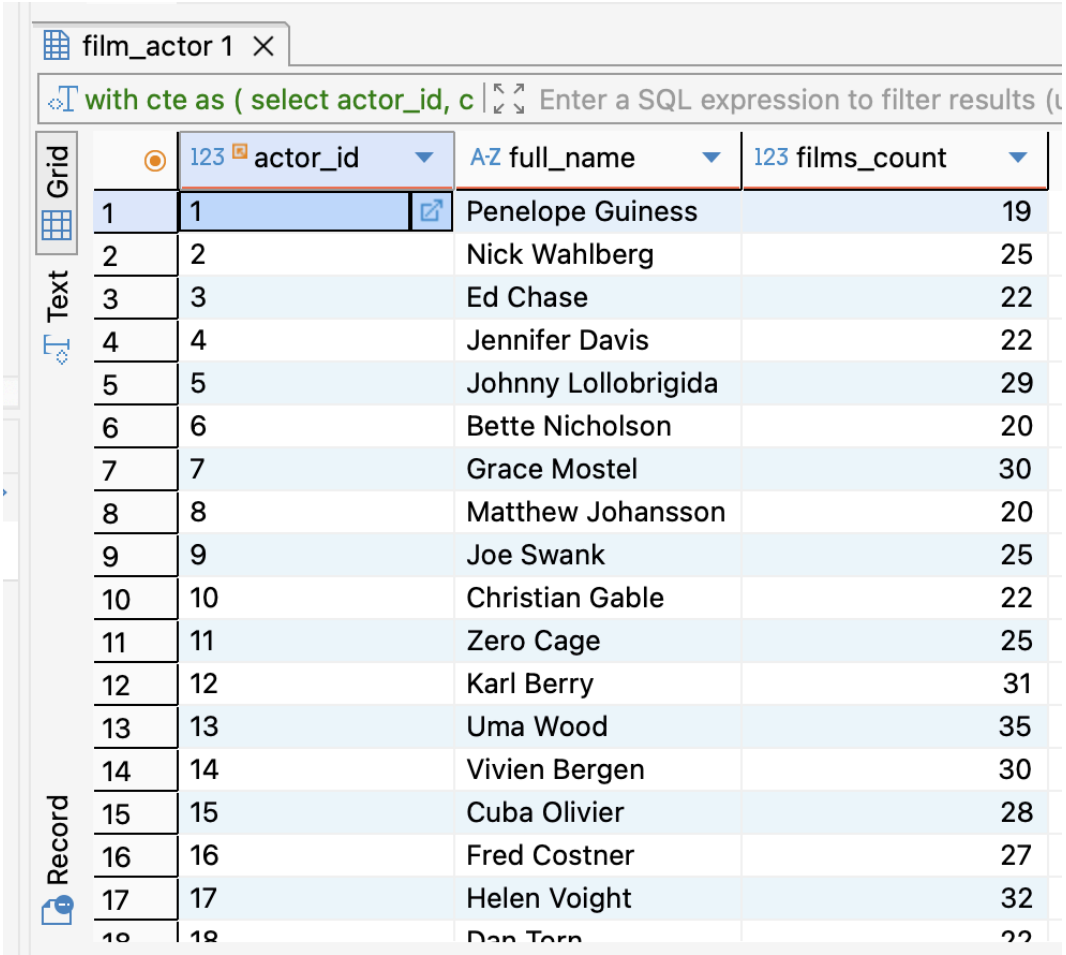
```
select
        customer_id,
```

```
        round(avg(amount), 2) avg_transaction
from
        payment
group by
        customer_id
having
        (count(*) >= 40)
order by
        avg_transaction desc;
```



20. Вывести идентификатор актера, его полное имя и количество фильмов, в которых он снялся.

```
with cte as
  (
select
        actor_id,
                count(*) films_count
from
        film_actor
group by
                actor_id)
select
        cte.actor_id,
        a.first_name || ' ' || a.last_name full_name,
        cte.films_count
from
        cte
inner join actor a
  on
        a.actor_id = cte.actor_id;
```

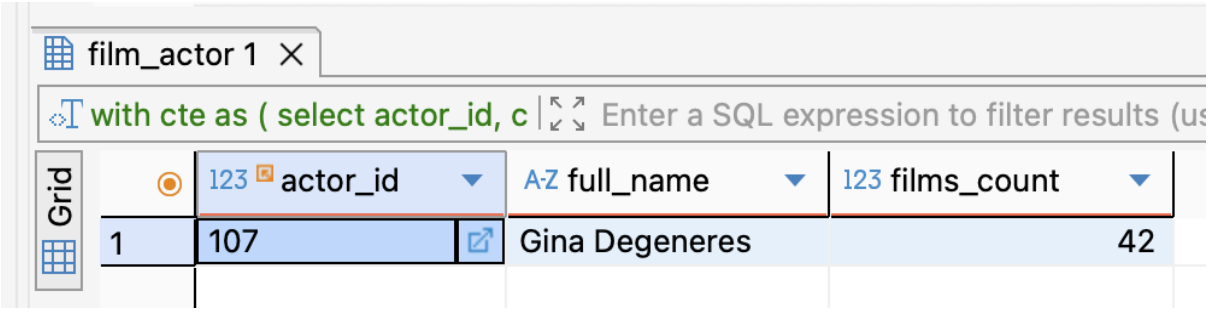


	actor_id	full_name	films_count
1	1	Penelope Guiness	19
2	2	Nick Wahlberg	25
3	3	Ed Chase	22
4	4	Jennifer Davis	22
5	5	Johnny Lollobrigida	29
6	6	Bette Nicholson	20
7	7	Grace Mostel	30
8	8	Matthew Johansson	20
9	9	Joe Swank	25
10	10	Christian Gable	22
11	11	Zero Cage	25
12	12	Karl Berry	31
13	13	Uma Wood	35
14	14	Vivien Bergen	30
15	15	Cuba Olivier	28
16	16	Fred Costner	27
17	17	Helen Voight	32

Определить актера, снявшегося в наибольшем количестве фильмов (группировать по id актера).

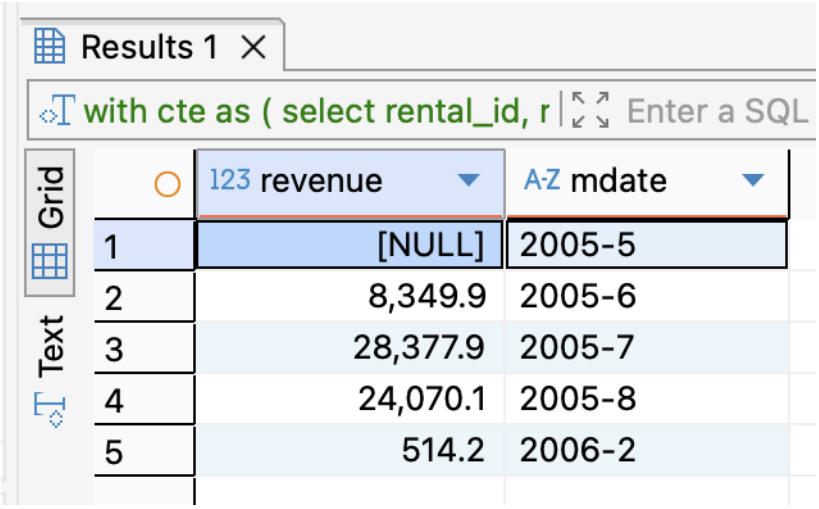
```
with cte as
  (
select
        actor_id,
                count(*) films_count
from
        film_actor
group by
        actor_id
order by
        count(*)
      desc
limit 1
  )
select
        cte.actor_id,
        a.first_name || ' ' || a.last_name full_name,
        cte.films_count
```

```
from
        cte
inner join actor a on
        a.actor_id = cte.actor_id
order by
        films_count desc ;
```



21. Посчитать выручку по каждому месяцу работы проката. Месяц должен определяться по дате аренды (rental_date), а не по дате оплаты (payment_date). Округлить выручку до одного знака после запятой. Отсортировать строки в хронологическом порядке. В отчете должен присутствовать месяц, в который не было выручки (нет данных о платежах).

```
with cte as
  (
select
                rental_id,
                rental_date::date
from
        rental
  )
select
        round(sum(p.amount), 1) revenue,
        extract(year from rental_date) || '-' || extract(month from rental_date) mdate
from
        cte
left join payment p
  on
        p.rental_id = cte.rental_id
group by
        mdate
order by
        mdate;
```

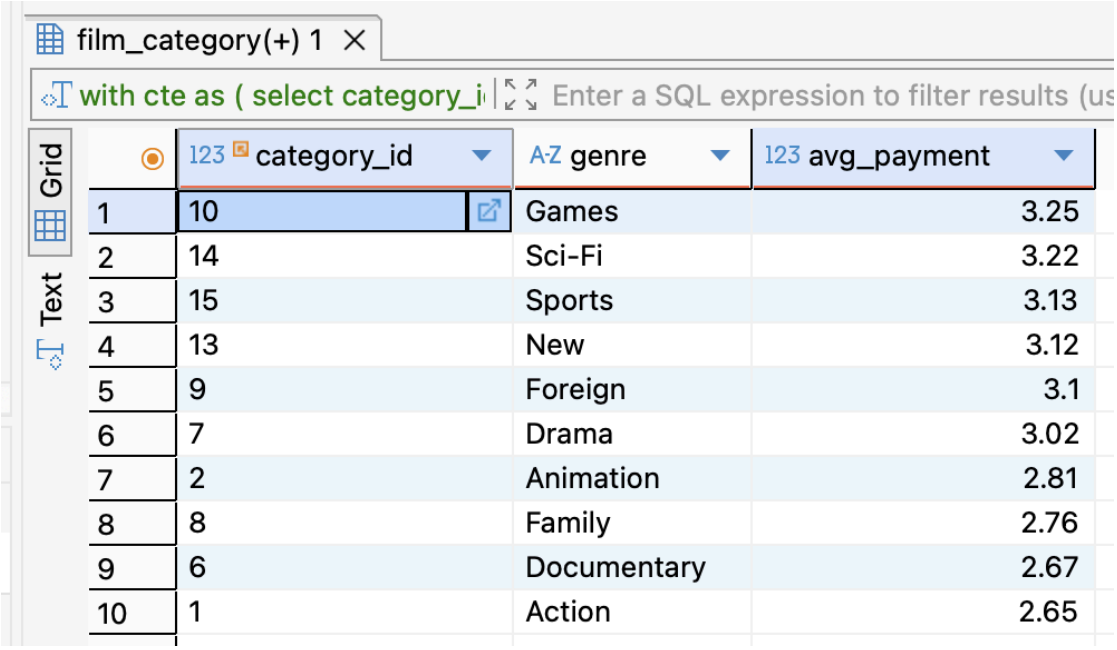


	123 revenue	A-Z mdate
1	[NULL]	2005-5
2	8,349.9	2005-6
3	28,377.9	2005-7
4	24,070.1	2005-8
5	514.2	2006-2

22. Найти средний платеж по каждому жанру фильма. Отобразить только те жанры, к которым относится более 60 различных фильмов. Округлить средний платеж до двух знаков после запятой и дать понятные названия столбцам. Отсортировать жанры по убыванию среднего платежа.

```
with cte as (
select
	category_id
from
	film_category
group by
	category_id
having
	count(*) > 60
),
cat as (
select
	f.film_id,
	f.category_id,
	c.name as genre
from
	film_category f
inner join category c on
	c.category_id = f.category_id
where
	f.category_id in (
	select
		category_id
	from
		cte)
)
select
```

```
        cat.category_id,
        cat.genre,
        ROUND(AVG(f.rental_rate), 2) as avg_payment
from
        cat
inner join film f on
        f.film_id = cat.film_id
group by
        cat.category_id,
        cat.genre
order by
        avg_payment desc;
```

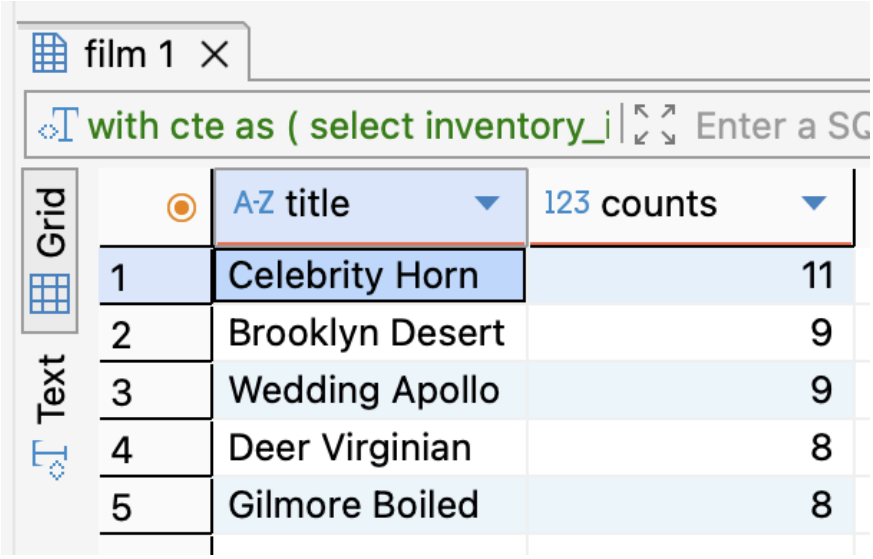


	category_id	genre	avg_payment
1	10	Games	3.25
2	14	Sci-Fi	3.22
3	15	Sports	3.13
4	13	New	3.12
5	9	Foreign	3.1
6	7	Drama	3.02
7	2	Animation	2.81
8	8	Family	2.76
9	6	Documentary	2.67
10	1	Action	2.65

23. Определить, какие фильмы чаще всего берут напрокат по субботам. Вывести названия первых 5 самых популярных фильмов. При одинаковой популярности отдать предпочтение фильму, который идет раньше по алфавиту.

```
with cte as (
select
        inventory_id,
        rental_date
from
        rental
where
        extract(DOW from rental_date) = 6
)
select
        f.title,
```

```
        COUNT(*) as counts
from
        cte
inner join inventory i on
        i.inventory_id = cte.inventory_id
inner join film f on
        i.film_id = f.film_id
group by
        f.title
order by
        counts desc,
        f.title asc
limit 5;
```

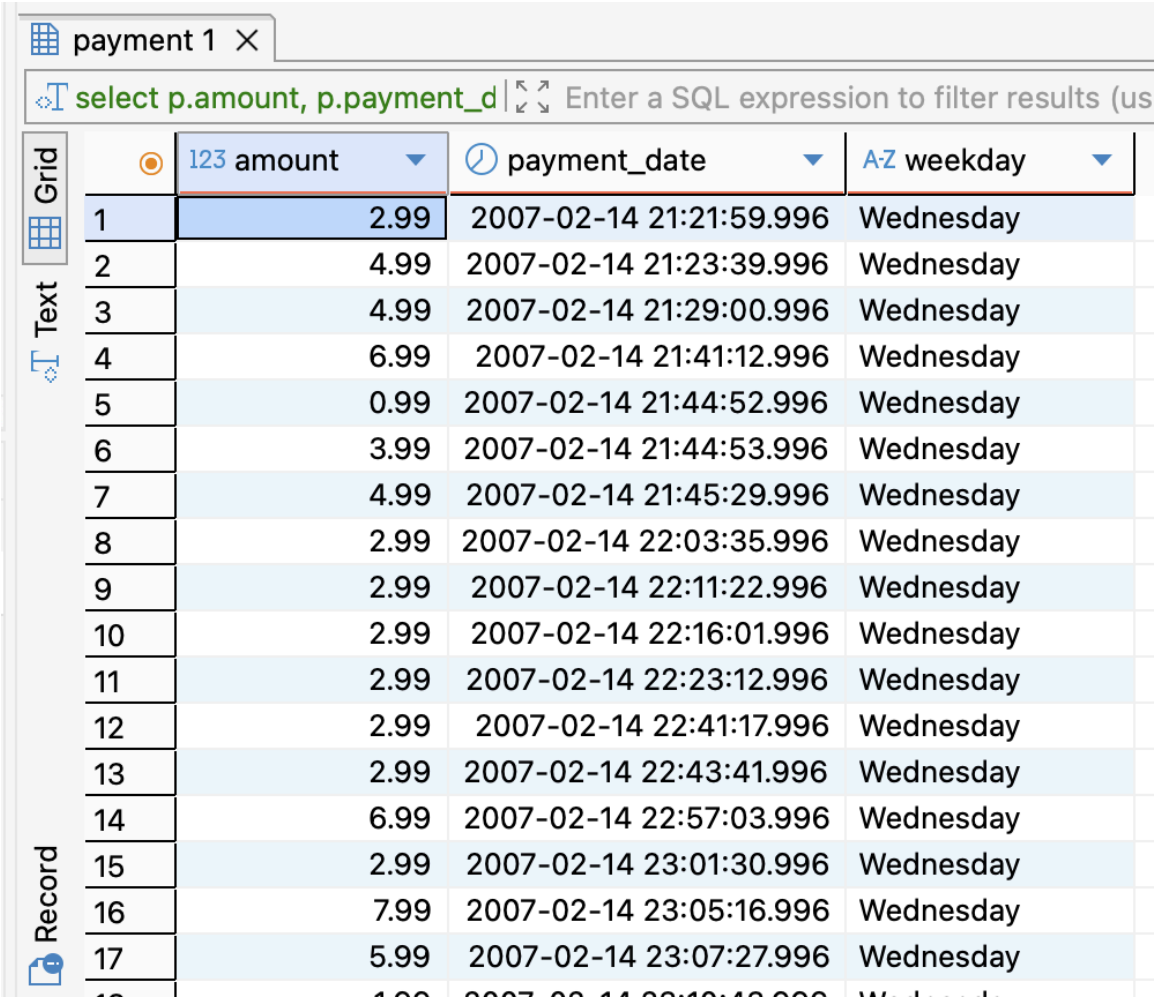


	A-Z title	123 counts
1	Celebrity Horn	11
2	Brooklyn Desert	9
3	Wedding Apollo	9
4	Deer Virginian	8
5	Gilmore Boiled	8

Оконные функции и простые запросы

24. Для каждой оплаты вывести сумму, дату и день недели (название дня недели текстом).

```
select
        p.amount,
        p.payment_date,
        to_char(p.payment_date, 'Day') as weekday
from
        payment p
order by
        p.payment_date
```



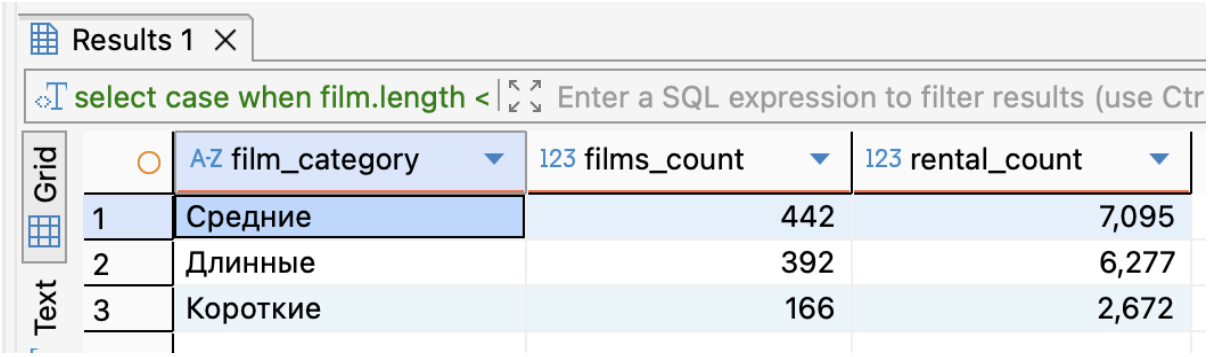
	123 amount	payment_date	A-Z weekday
1	2.99	2007-02-14 21:21:59.996	Wednesday
2	4.99	2007-02-14 21:23:39.996	Wednesday
3	4.99	2007-02-14 21:29:00.996	Wednesday
4	6.99	2007-02-14 21:41:12.996	Wednesday
5	0.99	2007-02-14 21:44:52.996	Wednesday
6	3.99	2007-02-14 21:44:53.996	Wednesday
7	4.99	2007-02-14 21:45:29.996	Wednesday
8	2.99	2007-02-14 22:03:35.996	Wednesday
9	2.99	2007-02-14 22:11:22.996	Wednesday
10	2.99	2007-02-14 22:16:01.996	Wednesday
11	2.99	2007-02-14 22:23:12.996	Wednesday
12	2.99	2007-02-14 22:41:17.996	Wednesday
13	2.99	2007-02-14 22:43:41.996	Wednesday
14	6.99	2007-02-14 22:57:03.996	Wednesday
15	2.99	2007-02-14 23:01:30.996	Wednesday
16	7.99	2007-02-14 23:05:16.996	Wednesday
17	5.99	2007-02-14 23:07:27.996	Wednesday

25.

- Распределить фильмы по трем категориям в зависимости от длительности:
  - «Короткие» — менее 70 минут;
  - «Средние» — от 70 минут (включительно) до 130 минут (не включая 130);
  - «Длинные» — от 130 минут и более.
- Для каждой категории необходимо:
  - посчитать количество прокатов (то есть сколько раз фильмы этой категории брались в аренду);
  - посчитать количество фильмов, которые относятся к этой категории и хотя бы один раз сдавались в прокат.
- Фильмы, у которых не было ни одного проката, не должны учитываться в подсчете количества фильмов в категории. Продумать, какой тип соединения таблиц нужно использовать, чтобы этого добиться.

select

```
        case
                when film.length < 70 then 'Короткие'
                when film.length >= 70
                and film.length < 130 then 'Средние'
                else 'Длинные'
        end as film_category,
        count(distinct film.film_id) as films_count,
        count(r.rental_id) as rental_count
from
        film
left join inventory i on
        film.film_id = i.film_id
left join rental r on
        i.inventory_id = r.inventory_id
group by
        film_category
order by
        rental_count desc
```

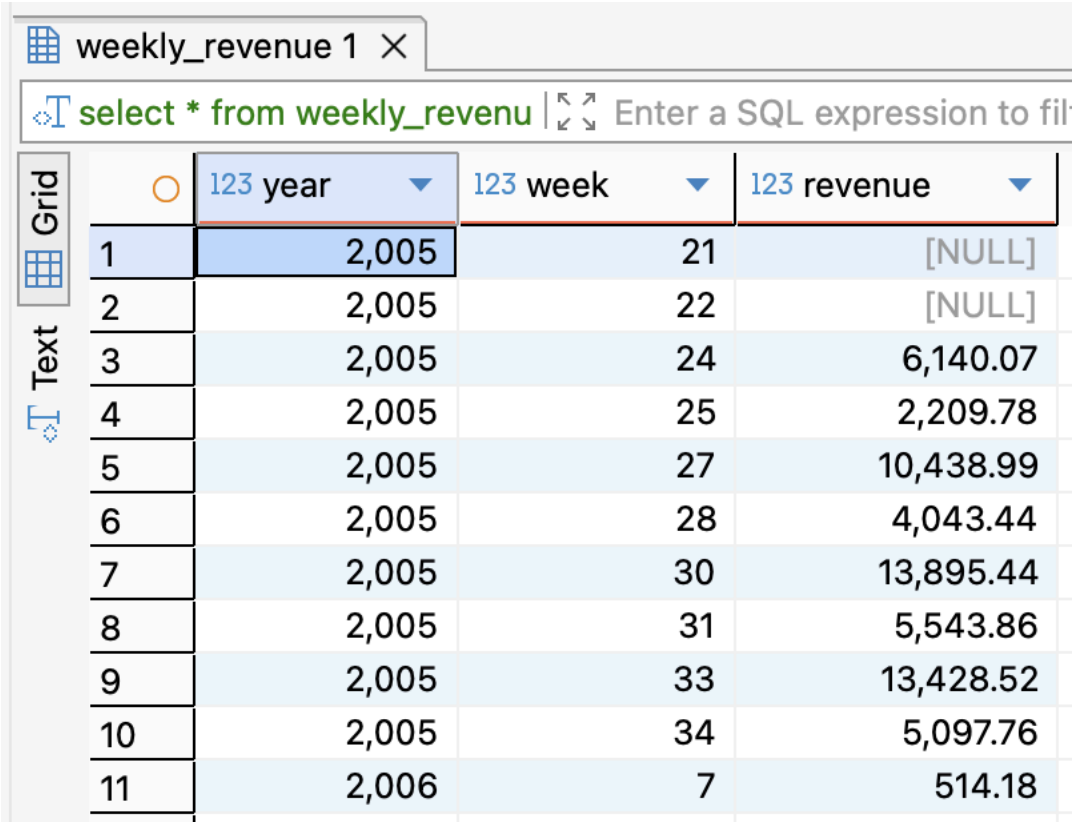


	film_category	films_count	rental_count
1	Средние	442	7,095
2	Длинные	392	6,277
3	Короткие	166	2,672

- Для дальнейших заданий считать, что создана таблица weekly_revenue, в которой для каждой недели и года хранится суммарная выручка компании за эту неделю (на основании данных о прокатах и платежах).

```
create table weekly_revenue as
select
        extract(year from rental_date) as year,
        extract(week from rental_date) as week,
        sum(amount) as revenue
from
        rental r
left join payment p on
        p.rental_id = r.rental_id
group by
        1,
```

```
        2
order by
        1,
        2
```



weekly_revenue 1

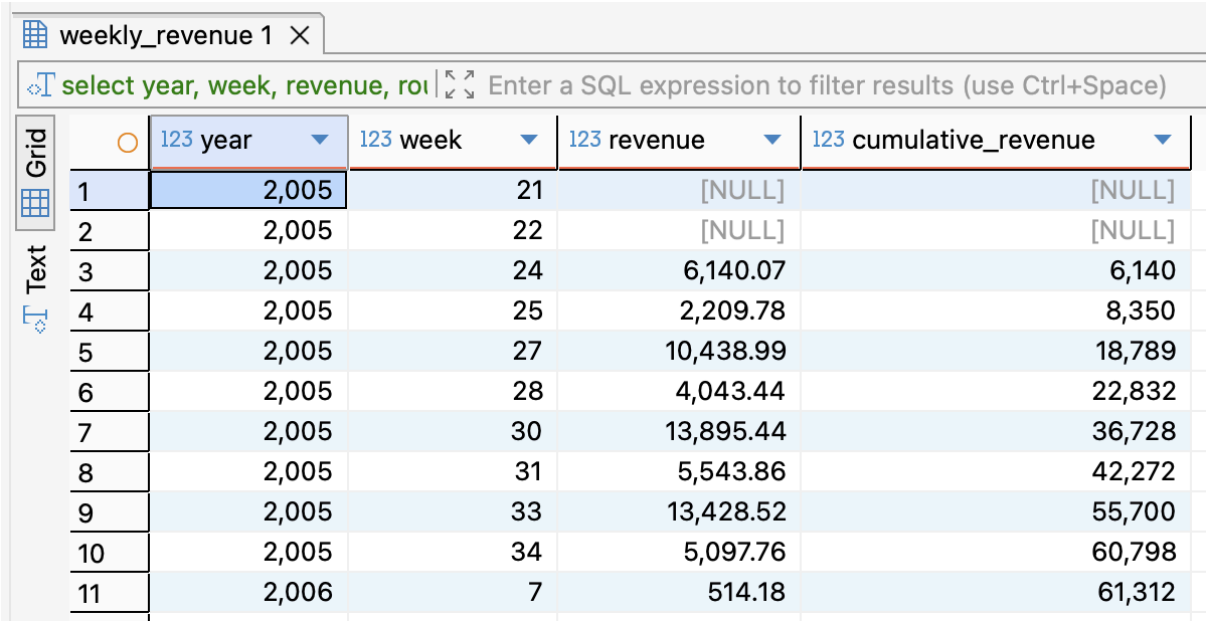
select * from weekly_revenu | Enter a SQL expression to fil

Grid | Text

	123 year	123 week	123 revenue
1	2,005	21	[NULL]
2	2,005	22	[NULL]
3	2,005	24	6,140.07
4	2,005	25	2,209.78
5	2,005	27	10,438.99
6	2,005	28	4,043.44
7	2,005	30	13,895.44
8	2,005	31	5,543.86
9	2,005	33	13,428.52
10	2,005	34	5,097.76
11	2,006	7	514.18

26. На основе таблицы weekly_revenue рассчитать накопленную (кумулятивную) сумму недельной выручки бизнеса. Вывести все столбцы таблицы weekly_revenue и добавить к ним столбец с накопленной выручкой. Накопленную выручку округлить до целого числа.

```
select
        year,
        week,
        revenue,
        round(sum(revenue) over (order by year, week), 0) as cumulative_revenue
from
        weekly_revenue
```



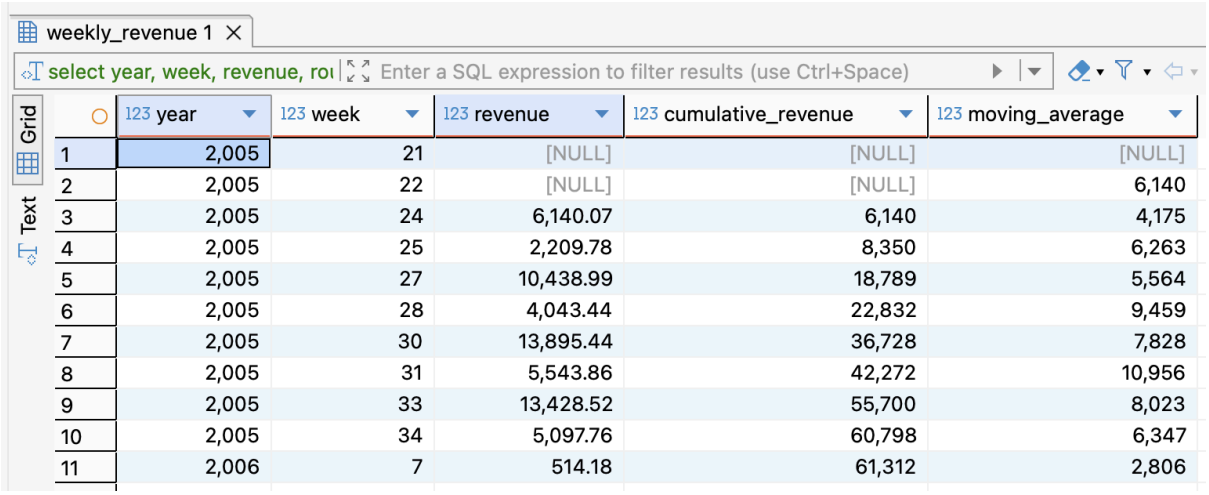
	123 year	123 week	123 revenue	123 cumulative_revenue
1	2,005	21	[NULL]	[NULL]
2	2,005	22	[NULL]	[NULL]
3	2,005	24	6,140.07	6,140
4	2,005	25	2,209.78	8,350
5	2,005	27	10,438.99	18,789
6	2,005	28	4,043.44	22,832
7	2,005	30	13,895.44	36,728
8	2,005	31	5,543.86	42,272
9	2,005	33	13,428.52	55,700
10	2,005	34	5,097.76	60,798
11	2,006	7	514.18	61,312

27. На основе таблицы weekly_revenue рассчитать скользящую среднюю недельной выручки, используя для расчета три недели: предыдущую, текущую и следующую. Вывести всю таблицу weekly_revenue и добавить:

- столбец с накопленной суммой выручки;
- столбец со скользящей средней недельной выручки.

Скользящую среднюю округлить до целого числа.

```
select
        year,
        week,
        revenue,
        round(sum(revenue) over (order by year, week), 0) as cumulative_revenue,
        round(avg(revenue) over (order by year, week rows between 1 preceding
and 1 following), 0) as moving_average
from
        weekly_revenue
```



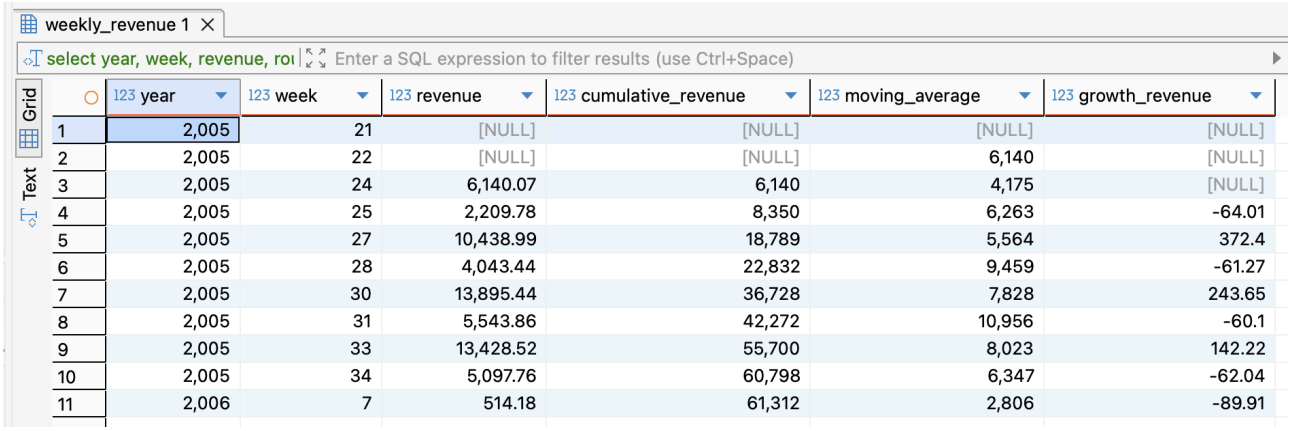
	123 year	123 week	123 revenue	123 cumulative_revenue	123 moving_average
1	2,005	21	[NULL]	[NULL]	[NULL]
2	2,005	22	[NULL]	[NULL]	6,140
3	2,005	24	6,140.07	6,140	4,175
4	2,005	25	2,209.78	8,350	6,263
5	2,005	27	10,438.99	18,789	5,564
6	2,005	28	4,043.44	22,832	9,459
7	2,005	30	13,895.44	36,728	7,828
8	2,005	31	5,543.86	42,272	10,956
9	2,005	33	13,428.52	55,700	8,023
10	2,005	34	5,097.76	60,798	6,347
11	2,006	7	514.18	61,312	2,806

28. Рассчитать прирост недельной выручки бизнеса в процентах по сравнению с предыдущей неделей.
Прирост в процентах определяется как:
(текущая недельная выручка – выручка предыдущей недели) / выручка предыдущей недели × 100%.
Вывести всю таблицу weekly_revenue и добавить:

- столбец с накопленной суммой выручки;
- столбец со скользящей средней;
- столбец с приростом недельной выручки в процентах.

Значение прироста в процентах округлить до двух знаков после запятой.

```
select
        year,
        week,
        revenue,
        round(sum(revenue) over (order by year, week), 0) as cumulative_revenue,
        round(avg(revenue) over (order by year, week rows between 1 preceding
and 1 following), 0) as moving_average,
        round(
    case when lag(revenue, 1) over (order by year, week) = 0 then 0
        else (revenue - lag(revenue, 1) over (order by year, week)) / lag(revenue,
1) over (order by year, week) * 100
    end, 2
    ) as growth_revenue
from
        weekly_revenue
```


weekly_revenue 1

select year, week, revenue, ro | Enter a SQL expression to filter results (use Ctrl+Space)

Grid | Text

	year	week	revenue	cumulative_revenue	moving_average	growth_revenue
1	2,005	21	[NULL]	[NULL]	[NULL]	[NULL]
2	2,005	22	[NULL]	[NULL]	6,140	[NULL]
3	2,005	24	6,140.07	6,140	4,175	[NULL]
4	2,005	25	2,209.78	8,350	6,263	-64.01
5	2,005	27	10,438.99	18,789	5,564	372.4
6	2,005	28	4,043.44	22,832	9,459	-61.27
7	2,005	30	13,895.44	36,728	7,828	243.65
8	2,005	31	5,543.86	42,272	10,956	-60.1
9	2,005	33	13,428.52	55,700	8,023	142.22
10	2,005	34	5,097.76	60,798	6,347	-62.04
11	2,006	7	514.18	61,312	2,806	-89.91