

1. Вывести список всех клиентов (таблица customer).

```
SELECT * FROM customer;
```

customer 1 X Execution plan - 1

	customer_id	store_id	first_name	last_name	email	address_id	activebool	cr
1	524	1	Jared	Ely	jared.elly@sakilacustomer.org	530	[v]	
2	1	1	Mary	Smith	mary.smith@sakilacustomer.org	5	[v]	
3	2	1	Patricia	Johnson	patricia.johnson@sakilacustomer.org	6	[v]	
4	3	1	Linda	Williams	linda.williams@sakilacustomer.org	7	[v]	
5	4	2	Barbara	Jones	barbara.jones@sakilacustomer.org	8	[v]	
6	5	1	Elizabeth	Brown	elizabeth.brown@sakilacustomer.org	9	[v]	
7	6	2	Jennifer	Davis	jennifer.davis@sakilacustomer.org	10	[v]	
8	7	1	Maria	Miller	maria.miller@sakilacustomer.org	11	[v]	
9	8	2	Susan	Wilson	susan.wilson@sakilacustomer.org	12	[v]	
10	9	2	Margaret	Moore	margaret.moore@sakilacustomer.org	13	[v]	
11	10	1	Dorothy	Taylor	dorothy.taylor@sakilacustomer.org	14	[v]	
12	11	2	Lisa	Anderson	lisa.anderson@sakilacustomer.org	15	[v]	
13	12	1	Nancy	Thomas	nancy.thomas@sakilacustomer.org	16	[v]	
14	13	2	Karen	Jackson	karen.jackson@sakilacustomer.org	17	[v]	
15	14	2	Betty	White	betty.white@sakilacustomer.org	18	[v]	
16	15	1	Helen	Harris	helen.harris@sakilacustomer.org	19	[v]	
17	16	2	Sandra	Martin	sandra.martin@sakilacustomer.org	20	[v]	
18	17	1	Donna	Thompson	donna.thompson@sakilacustomer.org	21	[v]	
19	18	2	Carol	Garcia	carol.garcia@sakilacustomer.org	22	[v]	
20	19	1	Ruth	Martinez	ruth.martinez@sakilacustomer.org	23	[v]	
21	20	2	Sharon	Robinson	sharon.robinson@sakilacustomer.org	24	[v]	
22	21	1	Michelle	Clark	michelle.clark@sakilacustomer.org	25	[v]	

2. Вывести имена и фамилии клиентов с именем Carolyn.

```
• SELECT first_name, last_name  
  FROM customer  
 WHERE first_name = 'Carolyn';
```

customer 1 X

Execution plan - 1

SELECT first_name, last_name FROM customer WHERE Enter a SQL expression to filter results (use Ctrl+Space)

Grid AZ first_name AZ last_name

1 Carolyn Perez

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

```
SELECT CONCAT(first_name, ' ', last_name) AS full_name
FROM customer
WHERE first_name LIKE '%ary%'
    OR last_name LIKE '%ary%';
```

Results 1 × Execution plan - 1

Select * from customer where first_name like '%ary%' or last_name like '%ary%' Enter a SQL expression to filter results (use Ctrl+Space)

	AZ full_name
1	Mary Smith
2	Rosemary Schmidt
3	Richard Mcrary
4	Gary Coy
5	Tim Cary
6	Zachary Hite
7	Ruben Geary
8	Adrian Clary
9	Daryl Larue

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

```
SELECT *  
FROM payment  
ORDER BY amount DESC  
LIMIT 20;
```

payment 1 × Execution plan - 1

	123 payment_id	123 customer_id	123 staff_id	123 rental_id	123 amount	payment_date
1	22,650	204	2	15,415	11.99	2007-03-22 22:17:22.996
2	29,136	13	2	8,831	11.99	2007-04-29 21:06:07.996
3	28,814	592	1	3,973	11.99	2007-04-06 21:26:57.996
4	24,553	195	2	16,040	11.99	2007-03-23 20:47:59.996
5	20,403	362	1	14,759	11.99	2007-03-21 21:57:24.996
6	24,866	237	2	11,479	11.99	2007-03-02 20:46:39.996
7	23,757	116	2	14,763	11.99	2007-03-21 22:02:26.996
8	28,799	591	2	4,383	11.99	2007-04-07 19:14:17.996
9	20,152	336	1	15,073	10.99	2007-03-22 09:29:41.996
10	18,272	544	2	1,434	10.99	2007-02-15 16:59:12.996
11	19,764	292	1	12,739	10.99	2007-03-18 20:43:44.996
12	18,290	550	1	3,272	10.99	2007-02-21 03:46:53.996
13	19,815	297	1	12,472	10.99	2007-03-18 10:27:14.996
14	19,856	301	1	15,201	10.99	2007-03-22 14:53:08.996
15	20,244	345	1	14,702	10.99	2007-03-21 19:28:29.996
16	18,153	511	2	2,966	10.99	2007-02-20 06:07:59.996
17	18,175	516	1	1,718	10.99	2007-02-16 13:20:28.996
18	19,336	221	1	2,660	10.99	2007-02-19 09:18:28.996
19	19,481	260	1	2,091	10.99	2007-02-17 16:37:30.996
20	18,367	572	2	1,889	10.99	2007-02-17 02:33:38.996

```
SELECT *
FROM address
WHERE address_id IN (
    SELECT address_id
    FROM store
);
```

address 1 Execution plan - 1

	address_id	address	address2	district	city_id	postal_code	phone	last_update
1	1	47 MySakila Drive	[NULL]	Alberta	300			2006-02-15 09:45:30.000
2	2	28 MySQL Boulevard	[NULL]	QLD	576			2006-02-15 09:45:30.000

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

```
SELECT
    EXTRACT(DAY FROM payment_date)::INTEGER AS day,
    EXTRACT(MONTH FROM payment_date)::INTEGER AS month,
    EXTRACT(ISODOW FROM payment_date)::INTEGER AS weekday
FROM payment;
```

Results 1 × Execution plan - 1

	day	month	weekday
1	15	2	4
2	16	2	5
3	16	2	5
4	19	2	1
5	20	2	2
6	21	2	3
7	17	2	6
8	20	2	2
9	20	2	2
10	16	2	5
11	16	2	5
12	17	2	6
13	17	2	6
14	18	2	7
15	20	2	2
16	21	2	3
17	15	2	4
18	15	2	4
19	16	2	5

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
    customer_id,
    rental_date::DATE AS rental_date, -- Влияет только на вывод, а не на фильтрацию
    staff_id
FROM rental
WHERE rental_date >= '2005-06-01' -- Здесь нет приведений к типу, поэтому этот запрос может использовать индекс.
    AND rental_date < '2005-07-01';
```

rental 1 × Execution plan - 1

Enter a SQL expression to filter results (use Ctrl+Space)

Grid

Text

Record

	customer_id	rental_date	staff_id
1	416	2005-06-14	2
2	516	2005-06-14	1
3	239	2005-06-14	2
4	285	2005-06-14	1
5	310	2005-06-14	1
6	592	2005-06-14	1
7	49	2005-06-14	1
8	264	2005-06-14	2
9	46	2005-06-14	1
10	323	2005-06-14	2
11	481	2005-06-14	1
12	139	2005-06-14	2
13	595	2005-06-14	2
14	284	2005-06-14	2
15	306	2005-06-14	1
16	191	2005-06-14	2
17	95	2005-06-15	2
18	197	2005-06-15	2
19	512	2005-06-15	1

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

```
SELECT title, description, length
FROM film
WHERE length BETWEEN 60 AND 120
    AND release_year > 2000
ORDER BY length DESC
LIMIT 20
```

film 1 X Execution plan - 1

SELECT title, description, length FROM film WHERE Enter a SQL expression to filter results (use Ctrl+Space)

	AZ title	AZ description	123 length
1	Dolls Rage	A Thrilling Display of a Pioneer And a Frisbee who must Escape a Teacher in The Outback	120
2	Lock Rear	A Thoughtful Character Study of a Squirrel And a Technical Writer who must Outtrace a Student in Ancient Japan	120
3	Calendar Gunfight	A Thrilling Drama of a Frisbee And a Lumberjack who must Sink a Man in Nigeria	120
4	Dazed Punk	A Action-Packed Story of a Pioneer And a Technical Writer who must Discover a Forensic Psychologist in An Abandoned Amusement Park	120
5	Order Betrayed	A Amazing Saga of a Dog And a Shark who must Challenge a Cat in The Sahara Desert	120
6	Karate Moon	A Astounding Yarn of a Womanizer And a Dog who must Reach a Waitress in A MySQL Convention	120
7	Untouchables Sunrise	A Amazing Documentary of a Woman And a Astronaut who must Outtrace a Teacher in An Abandoned Fun House	120
8	Rage Games	A Fast-Paced Saga of a Astronaut And a Secret Agent who must Escape a Hunter in An Abandoned Amusement Park	120
9	Command Darling	A Awe-Inspiring Tale of a Forensic Psychologist And a Woman who must Challenge a Database Administrator in Ancient Japan	120
10	Identity Lover	A Boring Tale of a Composer And a Mad Cow who must Defeat a Car in The Outback	119
11	Apocalypse Flamingos	A Astounding Story of a Dog And a Squirrel who must Defeat a Woman in An Abandoned Amusement Park	119
12	Dumbo Lust	A Touching Display of a Feminist And a Dentist who must Conquer a Husband in The Gulf of Mexico	119
13	Games Bowfinger	A Astounding Documentary of a Butler And a Explorer who must Challenge a Butler in A Monastery	119
14	Strangers Graffiti	A Brilliant Character Study of a Secret Agent And a Man who must Find a Cat in The Gulf of Mexico	119
15	Bugsy Song	A Awe-Inspiring Character Study of a Secret Agent And a Boat who must Find a Squirrel in The First Manned Space Station	119
16	Fidelity Devil	A Awe-Inspiring Drama of a Technical Writer And a Composer who must Reach a Pastry Chef in A U-Boat	118
17	Backlash Undefeated	A Stunning Character Study of a Mad Scientist And a Mad Cow who must Kill a Car in A Monastery	118
18	Paths Control	A Astounding Documentary of a Butler And a Cat who must Find a Frisbee in Ancient China	118
19	Jawbreaker Brooklyn	A Stunning Reflection of a Boat And a Pastry Chef who must Succumb a A Shark in A Jet Boat	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 AS payment_date,  
amount  
FROM payment  
WHERE payment_date >= '2007-04-01'  
    AND payment_date < '2007-05-01'  
    AND amount <= 4  
ORDER BY  
    amount DESC,  
    payment_date ASC;
```

film 1 X Execution plan - 1

	A-Z title	A-Z description	▼	123 length	▼	
3	Calendar Gunfight	A Thrilling Drama of a Frisbee And a Lumberjack who must Sink a Man in Nigeria		120		
4	Dazed Punk	A Action-Packed Story of a Pioneer And a Technical Writer who must Discover a Forensic Psychologist in An Abandoned Amusement Park		120		
5	Order Betrayed	A Amazing Saga of a Dog And a A Shark who must Challenge a Cat in The Sahara Desert		120		
6	Karate Moon	A Astounding Yarn of a Womanizer And a Dog who must Reach a Waitress in A MySQL Convention		120		
7	Untouchables Sunrise	A Amazing Documentary of a Woman And a Astronaut who must Outrage a Teacher in An Abandoned Fun House		120		
8	Rage Games	A Fast-Paced Saga of a Astronaut And a Secret Agent who must Escape a Hunter in An Abandoned Amusement Park		120		
9	Command Darling	A Awe-Inspiring Tale of a Forensic Psychologist And a Woman who must Challenge a Database Administrator in Ancient Japan		120		
10	Identity Lover	A Boring Tale of a Composer And a Mad Cow who must Defeat a Car in The Outback		119		
11	Apocalypse Flamingos	A Astounding Story of a Dog And a Squirrel who must Defeat a Woman in An Abandoned Amusement Park		119		
12	Dumbo Lust	A Touching Display of a Feminist And a Dentist who must Conquer a Husband in The Gulf of Mexico		119		
13	Games Bowfinger	A Astounding Documentary of a Butler And a Explorer who must Challenge a Butler in A Monastery		119		
14	Strangers Graffiti	A Brilliant Character Study of a Secret Agent And a Man who must Find a Cat in The Gulf of Mexico		119		
15	Bugsy Song	A Awe-Inspiring Character Study of a Secret Agent And a Boat who must Find a Squirrel in The First Manned Space Station		119		
16	Fidelity Devil	A Awe-Inspiring Drama of a Technical Writer And a Composer who must Reach a Pastry Chef in A U-Boat		118		
17	Backlash Undefeated	A Stunning Character Study of a Mad Scientist And a Mad Cow who must Kill a Car in A Monastery		118		

10. Показать имена, фамилии и идентификаторы всех клиентов с именами Jack, Bob или Sara, чья фамилия содержит букву «р». 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') -- если нужно регистронезависимо, то можно добавить LOWER(), но это не будет использовать индекс. Хорошо использовать индекс!
      AND last_name ILIKE '%p%'
ORDER BY customer_id ASC;
```

	AZ Имя	AZ Фамилия	123 ➔ Идентификатор
1	Sara	Perry	84
2	Bob	Pfeiffer	564

```
▶ △ ⊜ --DROP TABLE IF EXISTS students;
▶ + △ CREATE TABLE students (
    id SERIAL PRIMARY KEY,
    first_name TEXT NOT NULL,
    last_name TEXT NOT NULL,
    age INTEGER NOT NULL,
    birth_date DATE NOT NULL,
    address TEXT NOT NULL
);
⊜ INSERT
    INTO students (id, first_name, last_name, age, birth_date, address)
VALUES (1, 'Александр', 'Иванов', 24, '2005-08-24', 'г. Москва, ул. Ленина, д. 82'); -- Автоинкремент не обновится, если его не обновить вручную
SELECT setval('students_id_seq', (SELECT MAX(id) FROM students));
SELECT * FROM students;
⊜ INSERT INTO students (first_name, last_name, age, birth_date, address)
VALUES
    ('Мария', 'Никонова', 19, '1994-07-22', 'г. Севастополь, Нахимовский пр., д. 5'),
    ('Станислав', 'Сидоров', 21, '2004-11-30', 'г. Новосибирск, ул. Гагарина, д. 66'),
    ('Екатерина', 'Владимирова', 43, '1972-01-10', 'г. Екатеринбург, ул. Мира, д. 22');
SELECT * FROM students;
⊜ DELETE FROM students
    WHERE id = 2;
SELECT * FROM students;
DROP TABLE students;
SELECT * FROM students;|
```

Statistics 1 × Execution plan - 1



SQL Error [42P01]: ERROR: relation "students" does not exist
Position: 17

```
CREATE TABLE students (
    id SERIAL PRIMARY KEY,
    first_name TEXT NOT NULL,
    last_name TEXT NOT NULL,
    age INTEGER NOT NULL,
    birth_date DATE NOT NULL,
    address TEXT NOT NULL
```

12. Вывести количество уникальных имен клиентов.

```
• SELECT COUNT(DISTINCT first_name) AS unique_names_number
      FROM customer;
```

Results 1 × Execution plan - 1

Enter a SQL expression to filter results (use Ctrl+Space)

Grid	123 unique_names_number
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.

```
SELECT
    amount,
    ARRAY_AGG(payment_date::DATE) AS payment_dates,
    COUNT(*) AS payment_count,
    SUM(amount) AS total_amount
FROM payment
GROUP BY amount
ORDER BY payment_count DESC, amount DESC
LIMIT 5;
```

payment 1 × Execution plan - 1

SELECT amount, ARRAY_AGG(payment_date::DATE), COUNT(*), SUM(amount) FROM payment GROUP BY amount ORDER BY COUNT(*) DESC, amount DESC LIMIT 5;

Enter a SQL expression to filter results (use Ctrl+Space)

Grid

	amount	payment_dates	payment_count	total_amount
1	4.99	2007-02-16 [+3423]	3,424	17,085.76
2	2.99	2007-02-19 [+3232]	3,233	9,666.67
3	0.99	2007-02-17 [+2719]	2,720	2,692.8
4	5.99	2007-02-21 [+1187]	1,188	7,116.12
5	6.99	2007-02-16 [+1016]	1,017	7,108.83

Text

Value

4.99

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

```
• SELECT
    store_id,
    COUNT(*) AS items_count
FROM inventory
GROUP BY store_id
```

inventory 1 × Execution plan - 1

SELECT store_id, COUNT(*) AS items_count FROM Enter a SQL expression to filter results (use Ctrl+Space)

	store_id	items_count
1	1	2,270
2	2	2,311

Grid Text

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

• **SELECT**

```
a.address,  
a.address2,  
a.address_id AS address_table_address_id, -- для примера  
s.address_id AS store_table_address_id -- для примера  
FROM store s JOIN address a  
ON s.address_id = a.address_id;
```

address(+) 1 X Execution plan - 1

SELECT a.address, a.address2, a.address_id AS ad | Enter a SQL expression to filter results (use Ctrl+Space)

Grid	A-Z address	A-Z address2	123 ↗ address_table_address_id	123 ↗ store_table_address_id
1	47 MySakila Drive	[NULL]	1	1
2	28 MySQL Boulevard	[NULL]	2	2

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

```
AI ◉ SELECT first_name || ' ' || last_name AS full_name
      FROM staff
UNION
SELECT first_name || ' ' || last_name AS full_name -- Проигнорируется, но не ошибка
      FROM customer
```

Results 1 ×

Execution plan - 1

	A-Z full_name
1	Austin Cintron
2	Hugh Waldrop
3	Steven Curley
4	Derek Blakely
5	Michele Grant
6	Lee Hawks
7	Dwight Lombardi
8	Leo Ebert
9	Carmen Owens
10	Pedro Chestnut
11	Pauline Henry
12	Edgar Rhoads
13	Jay Robb
14	Peggy Myers
15	Mark Rinehart
16	Ronald Weiner
17	Teresa Rogers
18	Tina Simmons

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

```
• SELECT DISTINCT first_name  
    FROM customer  
EXCEPT  
SELECT DISTINCT first_name  
    FROM staff  
ORDER BY first_name;
```

Results 1 × Execution plan - 1

SELECT DISTINCT first_name FROM customer EXT Enter a SQL expression to filter results (use Ctrl+Space)

	AZ first_name	Value
1	Aaron	
2	Adam	
3	Adrian	
4	Agnes	
5	Alan	
6	Albert	
7	Alberto	
8	Alex	
9	Alexander	
10	Alfred	
11	Alfredo	
12	Alice	
13	Alicia	
14	Allan	
15	Allen	
16	Allison	
17	Alma	
18	Alvin	

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.

```
• SELECT  
    customer_id,  
    rental_date::DATE,  
    staff_id  
FROM rental  
WHERE rental_date >= '2005-06-01' AND rental_date < '2005-07-01'; -- Одинарные кавычки для строк, дат, времени, строк и INSERT. Двойные кавы
```

rental 1 × Statistics 1 Execution plan - 1

Grid Text Record

customer_id rental_date staff_id

	customer_id	rental_date	staff_id
1	416	2005-06-14	2
2	516	2005-06-14	1
3	239	2005-06-14	2
4	285	2005-06-14	1
5	310	2005-06-14	1
6	592	2005-06-14	1
7	49	2005-06-14	1
8	264	2005-06-14	2
9	46	2005-06-14	1
10	323	2005-06-14	2
11	481	2005-06-14	1
12	139	2005-06-14	2
13	595	2005-06-14	2
14	284	2005-06-14	2
15	306	2005-06-14	1
16	191	2005-06-14	2
17	95	2005-06-15	2
18	197	2005-06-15	2

- 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) AS average_payment,
    COUNT(*) AS payments_count -- Для самопроверки
  FROM payment
 GROUP BY customer_id
--WHERE payments_count > 40 -- Порядок выполнения: 1. FROM → 2. WHERE → 3. GROUP BY → 4. HAVING → 5. SELECT → 6. ORDER BY
--HAVING payments_count > 40 -- SELECT тоже ещё не выполнился!
 HAVING COUNT(*) > 40
 ORDER BY payments_count ASC
```

payment 1 X

SELECT customer_id,ROUND(AVG(amount), 2) AS Enter a SQL expression to filter results (use Ctrl+Space)

	customer_id	average_payment	payments_count
1	526	4.97	42
2	148	4.7	45

20. Вывести идентификатор актера, его полное имя и количество фильмов, в которых он снялся. Определить актера, снявшегося в наибольшем количестве фильмов (группировать по id актера).

```
SELECT
    a.actor_id,
    a.first_name || ' ' || a.last_name AS full_name,
    COUNT(*) AS films_count
FROM actor a
JOIN film_actor fa
ON a.actor_id = fa.actor_id
GROUP BY a.actor_id, first_name, a.last_name
ORDER BY films_count DESC, full_name ASC
LIMIT 1;
```

actor 1

Enter a SQL expression to filter results (use Ctrl+Space)

Grid	123 actor_id	AZ full_name	123 films_count
1	107	Gina Degeneres	42

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 months AS (
    SELECT GENERATE_SERIES(
        DATE_TRUNC('month', MIN(rental_date)),
        DATE_TRUNC('month', MAX(rental_date)),
        '1 month'
    )::DATE AS month_start
    FROM rental
)
SELECT
    m.month_start AS month,
    COALESCE(ROUND(SUM(p.amount), 1), 0.0) AS revenue
FROM months m
LEFT JOIN rental r ON DATE_TRUNC('month', r.rental_date)::DATE = m.month_start
LEFT JOIN payment p ON r.rental_id = p.rental_id
GROUP BY m.month_start
ORDER BY m.month_start;
```

Results 1

	month	revenue
1	2005-05-01	0
2	2005-06-01	8,349.9
3	2005-07-01	28,377.9
4	2005-08-01	24,070.1
5	2005-09-01	0
6	2005-10-01	0
7	2005-11-01	0
8	2005-12-01	0
9	2006-01-01	0
10	2006-02-01	514.2

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.

SELECT

```
c.name AS genre,
ROUND(AVG(p.amount), 2) AS avg_payment
FROM category c
JOIN film_category fc ON c.category_id = fc.category_id
JOIN film f ON fc.film_id = f.film_id
JOIN inventory i ON f.film_id = i.film_id
JOIN rental r ON i.inventory_id = r.inventory_id
JOIN payment p ON r.rental_id = p.rental_id
GROUP BY c.category_id, c.name
HAVING COUNT(DISTINCT f.film_id) > 60
ORDER BY avg_payment DESC;
```

category 1 X

SQL SELECT c.name AS genre, ROUND(AVG(p.amount), 2) AS avg_payment FROM category c JOIN film_category fc ON c.category_id = fc.category_id JOIN film f ON fc.film_id = f.film_id JOIN inventory i ON f.film_id = i.film_id JOIN rental r ON i.inventory_id = r.inventory_id JOIN payment p ON r.rental_id = p.rental_id GROUP BY c.category_id, c.name HAVING COUNT(DISTINCT f.film_id) > 60 ORDER BY avg_payment DESC; Enter a SQL expression to filter results (use Ctrl+Space)

	AZ genre	123 avg_payment
1	Sports	4.53
2	Drama	4.32
3	Foreign	4.13
4	Documentary	4
5	Animation	3.99
6	Action	3.9
7	Family	3.88

- 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.

● SELECT

```
f.title  
FROM film f  
JOIN inventory i ON f.film_id = i.film_id  
JOIN rental r ON i.inventory_id = r.inventory_id  
WHERE EXTRACT(ISODOW FROM r.rental_date) = 6  
GROUP BY f.film_id, f.title  
ORDER BY COUNT(*) DESC, f.title  
LIMIT 5;
```

film 1 ×

SELECT title FROM film JOIN inventory JOIN rental | Enter a SQL expression to filter results (use Ctrl+Space)

Grid

Text

	AZ title
1	Celebrity Horn
2	Brooklyn Desert
3	Wedding Apollo
4	Deer Virginian
5	Gilmore Boiled

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

```
• SELECT
    amount,
    payment_date::DATE AS payment_date,
    TO_CHAR(payment_date, 'Day') AS weekday
  FROM payment;
```

payment 1 ×

	amount	payment_date	weekday
1	7.99	2007-02-15	Thursday
2	1.99	2007-02-16	Friday
3	7.99	2007-02-16	Friday
4	2.99	2007-02-19	Monday
5	7.99	2007-02-20	Tuesday
6	5.99	2007-02-21	Wednesday
7	5.99	2007-02-17	Saturday
8	5.99	2007-02-20	Tuesday
9	2.99	2007-02-20	Tuesday
10	4.99	2007-02-16	Friday
11	6.99	2007-02-16	Friday
12	0.99	2007-02-17	Saturday
13	0.99	2007-02-17	Saturday

25. (1)

Для каждой оплаты вывести:
сумму платежа;
дату платежа;
день недели, соответствующий дате платежа, в текстовом виде (например: «понедельник», «вторник» и т.п.).

SELECT

```
amount AS "Сумма платежа",
payment_date::DATE AS "Дата платежа",
CASE EXTRACT(ISODOW FROM payment_date)
    WHEN 1 THEN 'понедельник'
    WHEN 2 THEN 'вторник'
    WHEN 3 THEN 'среда'
    WHEN 4 THEN 'четверг'
    WHEN 5 THEN 'пятница'
    WHEN 6 THEN 'суббота'
    WHEN 7 THEN 'воскресенье'
END AS "День недели"
FROM payment;
```

payment 1 X

T SELECT amount AS "Сумма платежа", payment_date::DATE AS "Дата платежа", CASE EXTRACT(ISODOW FROM payment_date) WHEN 1 THEN 'понедельник' WHEN 2 THEN 'вторник' WHEN 3 THEN 'среда' WHEN 4 THEN 'четверг' WHEN 5 THEN 'пятница' WHEN 6 THEN 'суббота' WHEN 7 THEN 'воскресенье' END AS "День недели" FROM payment;			
	123 Сумма платежа	Дата платежа	AZ День недели
1	7.99	2007-02-15	четверг
2	1.99	2007-02-16	пятница
3	7.99	2007-02-16	пятница
4	2.99	2007-02-19	понедельник
5	7.99	2007-02-20	вторник
6	5.99	2007-02-21	среда
7	5.99	2007-02-17	суббота
8	5.99	2007-02-20	вторник
9	2.99	2007-02-20	вторник
10	4.99	2007-02-16	пятница
11	6.99	2007-02-16	пятница
12	0.99	2007-02-17	суббота

25. (2)

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.

SELECT

```
film_id,  
title,  
length,  
CASE  
    WHEN length < 70 THEN 'Короткие'  
    WHEN length >= 70 AND length < 130 THEN 'Средние'  
    WHEN length >= 130 THEN 'Длинные'  
END AS category  
FROM film  
ORDER BY length;
```

film 1 X

SELECT film_id, title, length, CASE WHEN length Enter a SQL expression to filter results (use Ctrl+Space)

Grid	123 ⚠ film_id	A-Z title	123 length	A-Z category	
Text	1	730	Ridgemont Submarine	46	Короткие
Record	2	469	Iron Moon	46	Короткие
Text	3	15	Alien Center	46	Короткие
Text	4	504	Kwai Homeward	46	Короткие
Text	5	505	Labyrinth League	46	Короткие
Text	6	237	Divorce Shining	47	Короткие
Text	7	398	Hanover Galaxy	47	Короткие
Text	8	869	Suspects Quills	47	Короткие
Text	9	393	Halloween Nuts	47	Короткие
Text	10	407	Hawk Chill	47	Короткие
Text	11	247	Downhill Enough	47	Короткие
Text	12	784	Shanghai Tycoon	47	Короткие

25. (3)

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

```
category,
COUNT(*) AS rentals_count,
COUNT(DISTINCT film_id) AS films_with_rentals_count
FROM (
    SELECT
        f.film_id,
        f.length,
        CASE
            WHEN f.length < 70 THEN 'Короткие'
            WHEN f.length >= 70 AND f.length < 130 THEN 'Средние'
            ELSE 'Длинные'
        END AS category
    FROM film f
    INNER JOIN inventory i ON f.film_id = i.film_id
    INNER JOIN rental r ON i.inventory_id = r.inventory_id
) AS rented_films
GROUP BY category
ORDER BY
CASE category
    WHEN 'Короткие' THEN 1
    WHEN 'Средние' THEN 2
    WHEN 'Длинные' THEN 3
END;
```

Results 1 X

SELECT category, COUNT(*) AS rentals_count, COUNT(DISTINCT film_id) AS films_with_rentals_count

Enter a SQL expression to filter results (use Ctrl+Space)

Grid

Text

	category	rentals_count	films_with_rentals_count
1	Короткие	2,672	158
2	Средние	7,095	424
3	Длинные	6,277	376

• 25. (4)

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 r.rental_date) AS year,
    EXTRACT(WEEK FROM r.rental_date) AS week,
    SUM(p.amount) AS total_revenue
FROM rental r
JOIN payment p ON r.rental_id = p.rental_id
GROUP BY
    EXTRACT(YEAR FROM r.rental_date),
    EXTRACT(WEEK FROM r.rental_date)
ORDER BY year, week;
```

• `SELECT *`

```
FROM weekly_revenue;
```

weekly_revenue 1 ×

Enter a SQL expression to filter results (use Ctrl+Space)

Grid	year	week	total_revenue
1	2,005	24	6,140.07
2	2,005	25	2,209.78
3	2,005	27	10,438.99
4	2,005	28	4,043.44
5	2,005	30	13,895.44
6	2,005	31	5,543.86
7	2,005	33	13,428.52
8	2,005	34	5,097.76
9	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,  
total_revenue,  
ROUND(SUM(total_revenue) OVER (ORDER BY year, week))::INTEGER AS cumulative_revenue  
FROM weekly_revenue  
ORDER BY year, week;
```

weekly_revenue 1 ×

SELECT year, week, total_revenue, ROUND(SUM(

Enter a SQL expression to filter results (use Ctrl+Space)

Grid	123 year	123 week	123 total_revenue	123 cumulative_revenue
1	2,005	24	6,140.07	6,140
2	2,005	25	2,209.78	8,350
3	2,005	27	10,438.99	18,789
4	2,005	28	4,043.44	22,832
5	2,005	30	13,895.44	36,728
6	2,005	31	5,543.86	42,272
7	2,005	33	13,428.52	55,700
8	2,005	34	5,097.76	60,798
9	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,  
total_revenue,  
ROUND(SUM(total_revenue) OVER (ORDER BY year, week))::INTEGER AS cumulative_revenue,  
ROUND(AVG(total_revenue) OVER (  
    ORDER BY year, week  
    ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING  
)::INTEGER AS moving_3_weeks_avg  
FROM weekly_revenue  
ORDER BY year, week;
```

weekly_revenue 1 ×

SELECT year, week, total_revenue, ROUND(SUM(total_revenue) OVER (ORDER BY year, week))::INTEGER AS cumulative_revenue, ROUND(AVG(total_revenue) OVER (ORDER BY year, week ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING)::INTEGER AS moving_3_weeks_avg FROM weekly_revenue ORDER BY year, week;					
Grid	123 year	123 week	123 total_revenue	123 cumulative_revenue	123 moving_3_weeks_avg
1	2,005	24	6,140.07	6,140	4,175
2	2,005	25	2,209.78	8,350	6,263
3	2,005	27	10,438.99	18,789	5,564
4	2,005	28	4,043.44	22,832	9,459
5	2,005	30	13,895.44	36,728	7,828
6	2,005	31	5,543.86	42,272	10,956
7	2,005	33	13,428.52	55,700	8,023
8	2,005	34	5,097.76	60,798	6,347
9	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:
$$(\text{текущая недельная выручка} - \text{выручка предыдущей недели}) / \text{выручка предыдущей недели} \times 100\%.$$

Вывести всю таблицу `weekly_revenue` и добавить:
столбец с накопленной суммой выручки;
столбец со скользящей средней;
столбец с приростом недельной выручки в процентах.

```
SELECT
    year,
    week,
    total_revenue,
    ROUND(SUM(total_revenue) OVER (ORDER BY year, week))::INTEGER AS cumulative_revenue,
    ROUND(AVG(total_revenue) OVER (
        ORDER BY year, week
        ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING
    ))::INTEGER AS moving_3_weeks_avg,
    ROUND(
        (total_revenue - LAG(total_revenue, 1) OVER (ORDER BY year, week))
        / NULLIF(LAG(total_revenue, 1) OVER (ORDER BY year, week), 0)
        * 100,
        2
    ) AS revenue_growth_pct
FROM weekly_revenue
ORDER BY year, week;
```

grid weekly_revenue 1

	year	week	total_revenue	cumulative_revenue	moving_3_weeks_avg	revenue_growth_pct
1	2,005	24	6,140.07	6,140	4,175	[NULL]
2	2,005	25	2,209.78	8,350	6,263	-64.01
3	2,005	27	10,438.99	18,789	5,564	372.4
4	2,005	28	4,043.44	22,832	9,459	-61.27
5	2,005	30	13,895.44	36,728	7,828	243.65
6	2,005	31	5,543.86	42,272	10,956	-60.1
7	2,005	33	13,428.52	55,700	8,023	142.22
8	2,005	34	5,097.76	60,798	6,347	-62.04
9	2,006	7	514.18	61,312	2,806	-89.91