HW3

Подготовка

1) Установка DBeaver
2) Запуск контейнера postgres:16 через `docker compose up --build -d` с параметрами:

```
- ports:
   - "5432:5432"
-environment:
   POSTGRES_USER: postgres
   POSTGRES_PASSWORD: postgres123
   POSTGRES_DB: dvd_rental
```

3) Восстановление базы из бэкапа `./backups/dvd-rental.backup`
   - `docker exec -i HW3_postgress pg_restore -U postgres -d dvd_rental --clean --if-exists /backups/dvd-rental.backup`
4) Подключение DBeaver по указанным параметрам в п.2 по `localhost`
5) ER-диаграмма базы данных dvd_rental:

SQL и получение данных

1) Вывести список всех клиентов (таблица customer).

```
SELECT * FROM public.customer;
```

2) Вывести имена и фамилии клиентов с именем Carolyn.

```
SELECT first_name, last_name  FROM public.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 *
FROM (
    SELECT first_name || ' ' || last_name AS full_name
    FROM public.customer
) t
WHERE full_name LIKE '%ary%';
```

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

```
SELECT *
FROM public.payment
ORDER BY amount desc
LIMIT 20;
```

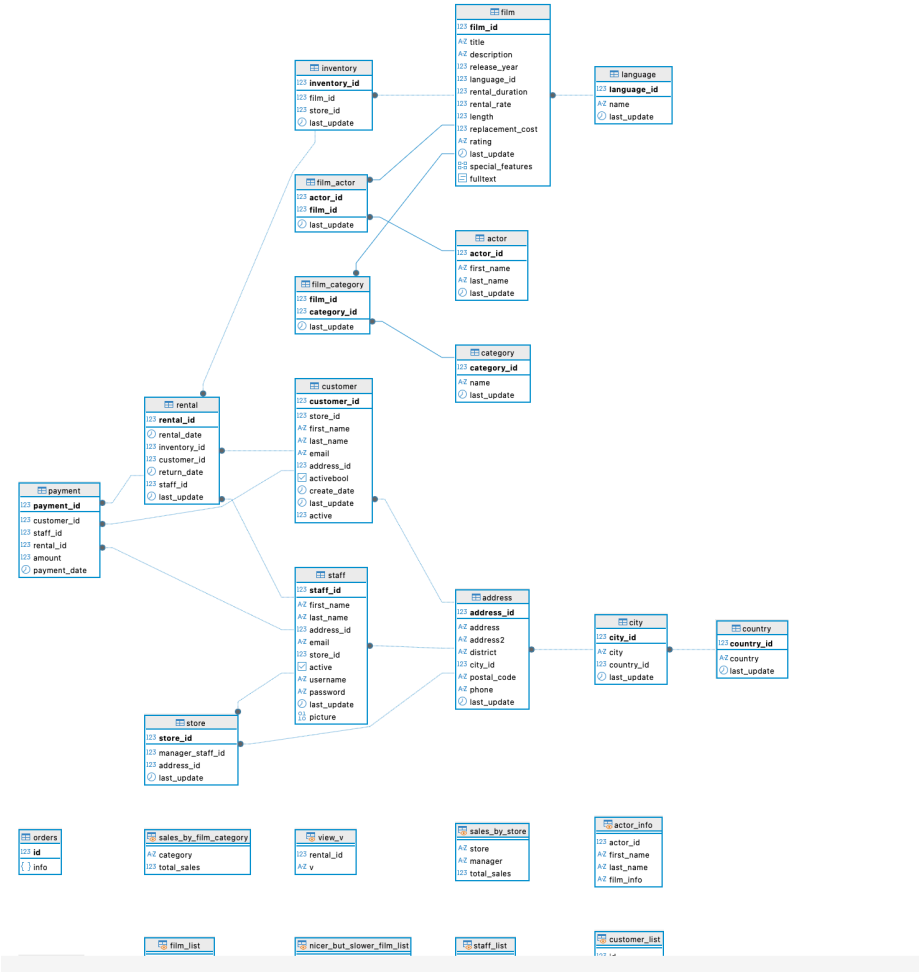

Figure 1: ER diagram

	customer_id	store_id	first_name	last_name	email	address_id
2	1	1	Mary	Smith	mary.smith@sakilacustomer.org	5
3	2	1	Patricia	Johnson	patricia.johnson@sakilacustomer.org	6
4	3	1	Linda	Williams	linda.williams@sakilacustomer.org	7
5	4	2	Barbara	Jones	barbara.jones@sakilacustomer.org	8
6	5	1	Elizabeth	Brown	elizabeth.brown@sakilacustomer.org	9
7	6	2	Jennifer	Davis	jennifer.davis@sakilacustomer.org	10
8	7	1	Maria	Miller	maria.miller@sakilacustomer.org	11
9	8	2	Susan	Wilson	susan.wilson@sakilacustomer.org	12
10	9	2	Margaret	Moore	margaret.moore@sakilacustomer.org	13
11	10	1	Dorothy	Taylor	dorothy.taylor@sakilacustomer.org	14
12	11	2	Lisa	Anderson	lisa.anderson@sakilacustomer.org	15
13	12	1	Nancy	Thomas	nancy.thomas@sakilacustomer.org	16
14	13	2	Karen	Jackson	karen.jackson@sakilacustomer.org	17
15	14	2	Betty	White	betty.white@sakilacustomer.org	18

Figure 2: ER diagram

	A-Z first_name	A-Z last_name
	Carolyn	Perez

Figure 3: ER diagram

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

Figure 4: ER diagram

	payment_id	customer_id	staff_id	rental_id	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

Figure 5: ER diagram

5. Вывести адреса всех магазинов, используя подзапрос.

```
SELECT address
FROM public.address
WHERE address_id in (
    SELECT address_id
    FROM store
)
```

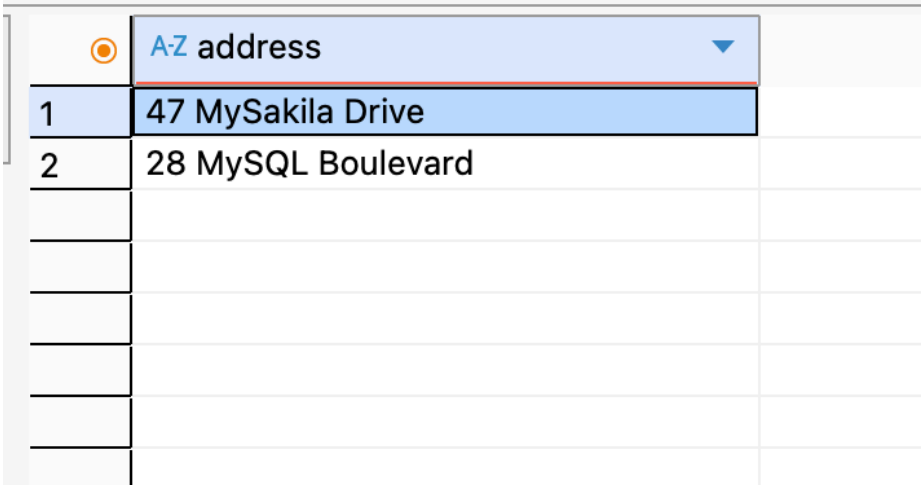

Figure 6: ER diagram

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(ISODOW FROM payment_date)    AS weekday,
    TO_CHAR(payment_date, 'Day') AS weekday_text
FROM payment;
```

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,
    CAST(rental_date AS date),
    staff_id
FROM rental
WHERE rental_date >= '2005-06-01' AND rental_date < '2005-07-01';
```

	payment_id	day	month	weekday	weekday_text
1	17,503	15	2	4	Thursday
2	17,504	16	2	5	Friday
3	17,505	16	2	5	Friday
4	17,506	19	2	1	Monday
5	17,507	20	2	2	Tuesday
6	17,508	21	2	3	Wednesday
7	17,509	17	2	6	Saturday
8	17,510	20	2	2	Tuesday
9	17,511	20	2	2	Tuesday
10	17,512	16	2	5	Friday
11	17,513	16	2	5	Friday
12	17,514	17	2	6	Saturday
13	17,515	17	2	6	Saturday
14	17,516	18	2	7	Sunday

Figure 7: ER diagram

	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

Figure 8: ER diagram

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,
    description,
    length
FROM film
WHERE (length BETWEEN 60 AND 120)  AND release_year >= 2000
ORDER BY length DESC
LIMIT 20;
```

	title	description	length
1	Rage Games	A Fast-Paced Saga of a Astronaut And a Secret Agent who must Escape a Hunter in An Abandoned Amus	120
2	Untouchables Sunrise	A Amazing Documentary of a Woman And a Astronaut who must Outrace a Teacher in An Abandoned Fun	120
3	Dolls Rage	A Thrilling Display of a Pioneer And a Frisbee who must Escape a Teacher in The Outback	120
4	Order Betrayed	A Amazing Saga of a Dog And a A Shark who must Challenge a Cat in The Sahara Desert	120
5	Dazed Punk	A Action-Packed Story of a Pioneer And a Technical Writer who must Discover a Forensic Psychologist in	120
6	Command Darling	A Awe-Inspiring Tale of a Forensic Psychologist And a Woman who must Challenge a Database Administra	120
7	Calendar Gunfight	A Thrilling Drama of a Frisbee And a Lumberjack who must Sink a Man in Nigeria	120
8	Karate Moon	A Astounding Yarn of a Womanizer And a Dog who must Reach a Waitress in A MySQL Convention	120
9	Lock Rear	A Thoughtful Character Study of a Squirrel And a Technical Writer who must Outrace a Student in Ancient	120
10	Identity Lover	A Boring Tale of a Composer And a Mad Cow who must Defeat a Car in The Outback	119
11	Games Bowfinger	A Astounding Documentary of a Butler And a Explorer who must Challenge a Butler in A Monastery	119

Figure 9: ER diagram

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,
    CAST(payment_date AS date) AS payment_date_only,
    amount
FROM payment
WHERE (payment_date >= DATE '2007-04-01' AND payment_date < DATE '2007-05-01')
    AND amount <= 4
ORDER BY amount DESC, payment_date
```

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 "  ",
    last_name AS "    ",
    customer_id AS "       "
FROM customer
WHERE first_name IN ('Jack', 'Bob', 'Sara') AND last_name ILIKE '%p%'
```

	payment_id	payment_date_only	amount
1	26,486	2007-04-05	3.99
2	26,100	2007-04-05	3.99
3	25,186	2007-04-05	3.99
4	29,361	2007-04-05	3.99
5	30,243	2007-04-06	3.99
6	27,743	2007-04-06	3.99
7	27,239	2007-04-06	3.99
8	26,692	2007-04-06	3.99
9	27,253	2007-04-06	3.99
10	25,360	2007-04-06	3.99
11	31,855	2007-04-06	3.99
12	26,261	2007-04-06	3.99
13	29,990	2007-04-06	3.99
14	29,224	2007-04-06	3.99
15	26,704	2007-04-06	3.99

Figure 10: ER diagram

```
ORDER BY customer_id
```

	Имя	Фамилия	Идентификатор
1	Sara	Perry	84
2	Bob	Pfeiffer	564

Figure 11: ER diagram

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).
- Внести в таблицу одного студента с id > 50.
- Просмотреть текущие записи таблицы.
- Внести несколько записей одним запросом, используя автоинкремент id.
- Снова просмотреть текущие записи таблицы.
- Удалить одного выбранного студента.
- Вывести полный список студентов.
- Удалить таблицу студентов.
- Выполнить запрос на выборку из таблицы студентов и вывести его результат (показать, что таблица удалена).

```
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
);
```

Updated Rows	0
Execute time	0.011s
Start time	Sun Dec 14 17:44:13 MSK 2025
Finish time	Sun Dec 14 17:44:13 MSK 2025
Query	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)

Figure 12: ER diagram

```
INSERT INTO students (id, first_name, last_name, age, birth_date, address)
VALUES (53, 'Ivan', 'Ivanov', 25, '2000-04-13', 'Moscow');
```

Updated Rows	1
Execute time	0.005s
Start time	Sun Dec 14 17:44:43 MSK 2025
Finish time	Sun Dec 14 17:44:43 MSK 2025
Query	INSERT INTO students (id, first_name, last_name, age, birth_date, address)
	VALUES (53, 'Ivan', 'Ivanov', 25, '2000-04-13', 'Moscow')

Figure 13: ER diagram

```
SELECT * FROM students;
```

```
INSERT INTO students (first_name, last_name, age, birth_date, address)
VALUES
```

	123 id	A-Z first_name	A-Z last_name	123 age	birth_date	A-Z address
1	53	Ivan	Ivanov	25	2000-04-13	Moscow

Figure 14: ER diagram

```
('Andrey', 'Andreev', 24, '2001-04-13', 'Saint Petersburg'),
('Maksim', 'Maksimov', 23, '2002-04-13', 'Saratov'),
('Roman', 'Romanov', 22, '2003-04-13', 'Ufa');
```

Name	Value
Updated Rows	3
Execute time	0.002s
Start time	Sun Dec 14 17:47:47 MSK 2025
Finish time	Sun Dec 14 17:47:47 MSK 2025
Query	INSERT INTO students (first_name, last_name, age, birth_date, address)
	VALUES
	('Andrey', 'Andreev', 24, '2001-04-13', 'Saint Petersburg'),
	('Maksim', 'Maksimov', 23, '2002-04-13', 'Saratov'),
	('Roman', 'Romanov', 22, '2003-04-13', 'Ufa')

Figure 15: ER diagram

```
SELECT * FROM students;
```

123 id	A-Z first_name	A-Z last_name	123 age	birth_date	A-Z address
53	Ivan	Ivanov	25	2000-04-13	Moscow
1	Andrey	Andreev	24	2001-04-13	Saint Petersburg
2	Maksim	Maksimov	23	2002-04-13	Saratov
3	Roman	Romanov	22	2003-04-13	Ufa

Figure 16: ER diagram

```
DELETE FROM students
WHERE id = 2;

SELECT * FROM students;

DROP TABLE students;

SELECT * FROM students;
```

id	first_name	last_name	age	birth_date	address
53	Ivan	Ivanov	25	2000-04-13	Moscow
1	Andrey	Andreev	24	2001-04-13	Saint Petersburg
3	Roman	Romanov	22	2003-04-13	Ufa

Figure 17: ER diagram

Updated Rows	0
Execute time	0.007s
Start time	Sun Dec 14 17:53:24 MSK 2025
Finish time	Sun Dec 14 17:53:24 MSK 2025
Query	DROP TABLE students

Figure 18: ER diagram

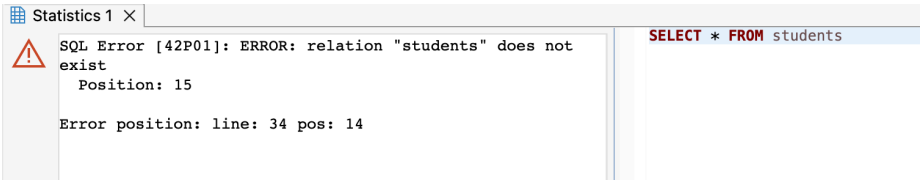

Figure 19: ER diagram

JOIN и агрегатные функции

12. Вывести количество уникальных имен клиентов.

```
SELECT count(distinct(first_name))
FROM customer;
```

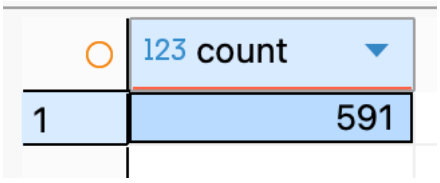

Figure 20: ER diagram

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(CAST(payment_date AS date)),
    count(amount) AS payment_count,
    sum(amount) AS payment_summ
FROM payment
GROUP BY amount
ORDER BY payment_count DESC
LIMIT 5
```

	123 amount	array_agg	123 payment_count	123 payment_summ
1	4.99	2007-02-16 [+34	3,424	17,085.76
2	2.99	2007-02-19 [+32	3,233	9,666.67
3	0.99	2007-02-17 [+27	2,720	2,692.8
4	5.99	2007-02-21 [+118	1,188	7,116.12
5	6.99	2007-02-16 [+10	1,017	7,108.83

Figure 21: ER diagram

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

```
SELECT
    store_id,
    count(inventory_id)
FROM inventory
GROUP BY inventory.store_id
```

	123 store_id	123 count
1	1	2,270
2	2	2,311

Figure 22: ER diagram

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

```
SELECT
    s.store_id ,
    a.address ,
    c.city,
    c2.country
FROM store s
JOIN address a ON a.address_id = s.address_id
JOIN city c ON c.city_id = a.city_id
JOIN country c2 ON c2.country_id  = c.country_id
```

	123 store_id	A-Z address	A-Z city	A-Z country
1	1	47 MySakila Drive	Lethbridge	Canada
2	2	28 MySQL Boulevard	Woodridge	Australia

Figure 23: ER diagram

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

```
SELECT first_name || ' ' || last_name AS full_name
FROM customer c

UNION ALL

SELECT first_name || ' ' || last_name  AS full_name
FROM staff s;
```

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

```
SELECT DISTINCT first_name
FROM customer
EXCEPT
```

	A-Z full_name
1	Jared Ely
2	Mary Smith
3	Patricia Johnson
4	Linda Williams
5	Barbara Jones
6	Elizabeth Brown
7	Jennifer Davis
8	Maria Miller
9	Susan Wilson
10	Margaret Moore
11	Dorothy Taylor
12	Lisa Anderson
13	Nancy Thomas

Figure 24: ER diagram

```
SELECT DISTINCT first_name
FROM staff;
```

	A-Z first_name
1	Danny
2	Amber
3	Johnnie
4	Edward
5	Cindy
6	Amy
7	Earl
8	Nancy
9	Carolyn
10	Geraldine
11	Rene
12	Kenneth
13	Ray
14	Adrian

Figure 25: ER diagram

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,
    CAST (rental_date AS date) AS rental_date,
    staff_id
FROM rental
WHERE rental_date >= DATE '2005-06-01' AND rental_date < DATE '2005-07-01'
```

	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

Figure 26: ER diagram

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 p.customer_id,
    ROUND(AVG(p.amount),2) AS mean_payments
FROM payment AS p
GROUP BY p.customer_id
HAVING COUNT(p.amount) >= 40;
```

	customer_id	mean_payments
1	144	4.74
2	526	4.97
3	148	4.7

Figure 27: ER diagram

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
```

```
    fa.actor_id,
    a.first_name || ' ' || a.last_name AS actor_name,
    count(*) AS film_count
FROM film_actor AS fa
JOIN actor AS a ON a.actor_id = fa.actor_id
GROUP BY fa.actor_id , actor_name
ORDER BY film_count DESC
LIMIT 1
```

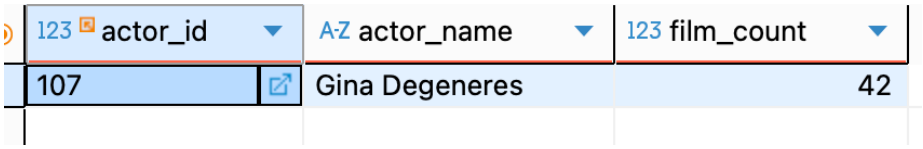

Figure 28: ER diagram

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', (SELECT MIN(rental_date) FROM rental)),
            date_trunc('month', (SELECT MAX(rental_date) FROM rental)),
            interval '1 month'
        ) AS month
)
SELECT
    TO_CHAR(m.month, 'YYYY-MM') AS month,
    ROUND(COALESCE(SUM(p.amount), 0), 1) AS revenue
FROM months m
LEFT JOIN rental AS r ON date_trunc('month', r.rental_date) = m.month
LEFT JOIN payment AS p ON p.rental_id = r.rental_id
GROUP BY m.month
ORDER BY m.month;
```

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
```

	A-Z month	123 revenue
1	2005-05	0
2	2005-06	8,349.9
3	2005-07	28,377.9
4	2005-08	24,070.1
5	2005-09	0
6	2005-10	0
7	2005-11	0
8	2005-12	0
9	2006-01	0
10	2006-02	514.2

Figure 29: ER diagram

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

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,
    count(f.title)
FROM rental AS r
JOIN inventory AS i ON i.inventory_id = r.inventory_id
```

	A-Z category_name	123 category_count	123 avg_payment
	Sports	73	4.53
	Drama	61	4.32
	Foreign	67	4.13
	Documentary	63	4
	Animation	64	3.99
	Action	61	3.9
	Family	67	3.88

Figure 30: ER diagram

```
JOIN film AS f ON f.film_id = i.film_id
WHERE EXTRACT(ISODOW FROM r.rental_date) = 6
GROUP BY f.title
ORDER BY count(f.title) DESC, f.title
LIMIT 5
```

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

Figure 31: ER diagram

Оконные функции и простые запросы

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

```
SELECT
    amount,
    CAST(payment_date AS date) AS payment_date,
    TO_CHAR(payment_date, 'Day') AS weekday_text
FROM payment
```

	123 amount	payment_date	A-Z weekday_text
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

Figure 32: ER diagram

25. Для каждой оплаты вывести:

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

```
WITH base AS (
    SELECT
        p.amount,
        p.payment_date::date AS payment_date,
        TO_CHAR(p.payment_date, 'FMDay') AS weekday_text,
        f.film_id,
        CASE
            WHEN f.length < 70 THEN '    '
            WHEN f.length >= 70 AND f.length < 130 THEN '    '
```

```
            ELSE '    '
        END AS len_category
    FROM payment p
    JOIN rental r ON r.rental_id = p.rental_id
    JOIN inventory i ON i.inventory_id = r.inventory_id
    JOIN film f ON f.film_id = i.film_id
),
films_per_category AS (
    SELECT
        len_category,
        COUNT(DISTINCT film_id) AS films_count
    FROM base
    GROUP BY len_category
)
SELECT
    b.amount,
    b.payment_date,
    b.weekday_text,
    b.len_category,
    COUNT(*) OVER (PARTITION BY b.len_category) AS rentals_count,
    fpc.films_count
FROM base b
JOIN films_per_category fpc ON fpc.len_category = b.len_category
ORDER BY b.payment_date;
```

amount	payment_date	weekday_text	len_category	rentals_count	films_count
4.99	2007-02-14	Wednesday	Длинные	5,707	376
5.99	2007-02-14	Wednesday	Средние	6,459	424
3.99	2007-02-14	Wednesday	Средние	6,459	424
7.99	2007-02-14	Wednesday	Длинные	5,707	376
5.99	2007-02-14	Wednesday	Средние	6,459	424
4.99	2007-02-14	Wednesday	Длинные	5,707	376
4.99	2007-02-14	Wednesday	Длинные	5,707	376
2.99	2007-02-14	Wednesday	Длинные	5,707	376

Figure 33: ER diagram

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 (
    year INTEGER NOT NULL,
    week INTEGER NOT NULL,
    revenue NUMERIC(10, 2) NOT NULL,
    PRIMARY KEY (year, week)
);
```

```
INSERT INTO weekly_revenue (year, week, revenue)
SELECT
    EXTRACT(YEAR FROM p.payment_date)::int AS year,
    EXTRACT(WEEK FROM p.payment_date)::int AS week,
    SUM(p.amount) AS revenue
FROM payment p
GROUP BY
    EXTRACT(YEAR FROM p.payment_date),
    EXTRACT(WEEK FROM p.payment_date)
ORDER BY year, week;

SELECT *
FROM weekly_revenue;
```

	123 year	123 week	123 revenue
1	2,007	7	4,923.98
2	2,007	8	3,427.86
3	2,007	9	5,358.29
4	2,007	11	5,443.2
5	2,007	12	13,085.07
6	2,007	14	6,562.62
7	2,007	15	7,911.84
8	2,007	17	8,361.11
9	2,007	18	5,723.89
10	2,007	20	514.18

Figure 34: ER diagram

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)) AS cumulative_revenue
FROM weekly_revenue
```

	123 year	123 week	123 revenue	123 cumulative_revenue
1	2,007	7	4,923.98	4,924
2	2,007	8	3,427.86	8,352
3	2,007	9	5,358.29	13,710
4	2,007	11	5,443.2	19,153
5	2,007	12	13,085.07	32,238
6	2,007	14	6,562.62	38,801
7	2,007	15	7,911.84	46,713
8	2,007	17	8,361.11	55,074
9	2,007	18	5,723.89	60,798
10	2,007	20	514.18	61,312

Figure 35: ER diagram

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)) AS cumulative_revenue,
    ROUND(
        (
            COALESCE(LAG(revenue) OVER (ORDER BY year, week), 0)
          + revenue
          + COALESCE(LEAD(revenue) OVER (ORDER BY year, week), 0)
        ) /
        (
            (CASE WHEN LAG(revenue)  OVER (ORDER BY year, week) IS NULL THEN 0 ELSE 1 END)
          + 1
          + (CASE WHEN LEAD(revenue) OVER (ORDER BY year, week) IS NULL THEN 0 ELSE 1 END)
        )
    ) AS moving_3_weeks_avg_revenue
FROM weekly_revenue;
```

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

	year	week	revenue	cumulative_revenue	moving_3_weeks_avg_revenue
1	2,007	7	4,923.98	4,924	4,176
2	2,007	8	3,427.86	8,352	4,570
3	2,007	9	5,358.29	13,710	4,743
4	2,007	11	5,443.2	19,153	7,962
5	2,007	12	13,085.07	32,238	8,364
6	2,007	14	6,562.62	38,801	9,187
7	2,007	15	7,911.84	46,713	7,612
8	2,007	17	8,361.11	55,074	7,332
9	2,007	18	5,723.89	60,798	4,866
10	2,007	20	514.18	61,312	3,119

Figure 36: ER diagram

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)) AS cumulative_revenue,
    ROUND(
        (
            COALESCE(LAG(revenue) OVER (ORDER BY year, week), 0)
          + revenue
          + COALESCE(LEAD(revenue) OVER (ORDER BY year, week), 0)
        ) /
        (
            (CASE WHEN LAG(revenue)  OVER (ORDER BY year, week) IS NULL THEN 0 ELSE 1 END)
          + 1
          + (CASE WHEN LEAD(revenue) OVER (ORDER BY year, week) IS NULL THEN 0 ELSE 1 END)
        )
    ) AS moving_3_weeks_avg_revenue,
    ROUND(
        (revenue - COALESCE(LAG(revenue) OVER (ORDER BY year, week), revenue)) / COALESCE(LA
    ,2) AS percentage_increase
FROM weekly_revenue;
```

		week	revenue	cumulative_revenue	moving_3_weeks_avg_revenue	percentage_increase
1	,007	7	4,923.98	4,924	4,176	0
2	,007	8	3,427.86	8,352	4,570	-30.38
3	,007	9	5,358.29	13,710	4,743	56.32
4	,007	11	5,443.2	19,153	7,962	1.58
5	,007	12	13,085.07	32,238	8,364	140.39
6	,007	14	6,562.62	38,801	9,187	-49.85
7	,007	15	7,911.84	46,713	7,612	20.56
8	,007	17	8,361.11	55,074	7,332	5.68
9	,007	18	5,723.89	60,798	4,866	-31.54
10	,007	20	514.18	61,312	3,119	-91.02

Figure 37: ER diagram