Подзапросы

- Легко
- Нормально
- Сложно

№		Задание	Решение
1.		Напишите запрос, который возвращает адреса и почтовые индексы всех адресов, расположенных в London. Не используйте JOIN для этой задачи.	Select address, postal_code from address where city_id in (Select city_id from city where city like 'London');
2.		Найдите клиентов ни разу не бравших в аренду фильмы с участием EMILY DEE. В качестве результата выведите таблицу с колонками last_name, first_name - фамилия и имя клиента. Отсортируйте список по фамилии клиента.	Select distinct customer.last_name, customer.first_name from customer where not exists (select null from rental inner join inventory on rental.inventory_id=inventory.inventory_id inner join film_actor on inventory.film_id=film_actor.film_id inner join actor on film_actor.actor_id=actor.actor_id where (actor.first_name='EMILY') and (actor.last_name='DEE') and (customer.customer_id=rental.customer_id)) order by customer.last_name asc;
3.		В случае утери, кражи, порчи или невозврата арендованного диска с клиента взимается стоимость замены (replacement_cost). Найдите в базе данных фильмы с самой высокой стоимостью замены используя условие с **под-запросом**. Напишите запрос, который возвращает поля film_id, title и replacement_cost в возрастающем порядке поля film_id.	select film_id, title, replacement_cost from film where replacement_cost=(select max(replacement_cost) from film) group by film_id order by film_id asc;
4.		Напишите SQL запрос, чтобы найти фильмы, стоимость проката которых выше средней стоимости всех фильмов. Используйте подзапрос для расчета среднего рейтинга. Полученная таблица должна включать следующие столбцы: film_id - идентификатор фильма, title - название фильма и rental_rate - стоимость проката фильма. Отсортируйте результат по убыванию арендной ставки.	select film_id, title, rental_rate from film where rental_rate>(select avg(rental_rate) from film) order by rental_rate desc;
5.		Создайте запрос SQL, чтобы найти клиентов, которые взяли напрокат больше фильмов, чем среднее количество прокатов среди всех клиентов. Используйте подзапрос для расчета среднего количества аренд.	select distinct rental.customer_id, customer.first_name, customer.last_name, count(rental.rental_id) as rental_count from customer inner join rental on customer.customer_id=rental.customer_id group by rental.customer_id

		Результирующая таблица должна содержать следующие столбцы: customer_id – уникальный идентификатор клиента, first_name – имя клиента, last_name — фамилия клиента. rental_count — количество взятых напрокат фильмов	having rental_count>(select count(distinct rental_id)/count(distinct customer_id) from rental);
6.		В этом задании вы нашли среднее время проката фильма (в днях). Напишите запрос, чтобы получить список фильмов, у которых время проката ниже среднего. Отобразите результат в таблице со столбцами film_id, title и average_rental_time. Отсортируйте таблицу по столбцу film_id.	select film.film_id, film.title, ROUND (AVG(DATEDIFF(rental.return_date,rental.rental_date))) as average_rental_time from film inner join inventory on film.film_id=inventory.film_id inner join rental on inventory.inventory_id=rental.inventory_id group by film.film_id having ROUND (AVG(DATEDIFF(rental.return_date,rental.rental_date)))<(Select ROUND (AVG(DATEDIFF(return_date,rental_date))) from rental) order by film.film_id asc;
7.		Найдите фильмы из базы данных Sakila, в которых нет записей об актерах. Решите задачу без использования JOIN-ов (используя условие NOT EXISTS). Выведите результат с полями title, release_year, отсортированными по названию фильма.	select distinct title, release_year from film where not EXISTS (select null from film_actor where film.film_id=film_actor.film_id) order by title asc;
8.		Рейтинг **NC-17** — это рейтинг фильмов, классифицированных как подходящие только для взрослых. Напишите запрос для поиска всех актеров, которые никогда не снимались в фильмах с этим рейтингом, используя условие NOT IN. Выведите результирующую таблицу из двух столбцов first_name и last_name, отсортированных по фамилии в алфавитном порядке.	select distinct actor.first_name, actor.last_name from actor where actor.actor_id not in (select distinct actor.actor_id from actor inner join film_actor on actor.actor_id=film_actor.actor_id inner join film on film_actor.film_id=film.film_id where film.rating='NC-17') order by actor.last_name asc;