Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное образовательное
учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Факультет инфокоммуникационных технологий

ЛАБОРАТОРНАЯ РАБОТА №2 «Запросы на выборку и модификацию данных, представления и индексы в PostgreSQL»

Выполнила:
студент : Аль-Мошки Исмаил Абдулвахаб
группа: K32401

Проверили:
Говорова Марина Михайловна

Санкт-Петербург
2023

Цель работы: овладеть практическими навыками создания представлений и запросов на выборку данных к базе данных PostgreSQL, использования подзапросов при модификации данных и индексов.

Оборудование: компьютерный класс.

Программное обеспечение: СУБД PostgreSQL, pgadmin 4.

Практическое задание:

1. Создать запросы и представления на выборку данных к базе данных PostgreSQL (согласно индивидуальному заданию, часть 2 и 3).

Вариант 12. БД «Прокат автомобилей»

Описание предметной области: Компания предоставляет прокат автомобилей. В пункт проката обращаются клиенты, данные о которых регистрируют в базе. Цена проката зависит от марки автомобиля, технических характеристик и года выпуска.

Для проката авто с клиентом заключается договор, в котором фиксируется период проката, вид страховки, стоимость страховки, залоговая стоимость. Залоговая стоимость возвращается полностью или частично клиенту, в зависимости от страховки, аварий и штрафов. Если залоговая стоимость уже возвращена клиенту, но на авто в компанию пришел штраф, то он оплачивается компанией, а не клиентом. При передаче авто клиенту составляется акт о передаче автомобиля клиенту. При возвращении автомобиля также составляется акт о передаче авто компании.

Если клиент не вернул автомобиль в срок и не оформил продление, ему назначается штраф за каждый час просрочки.

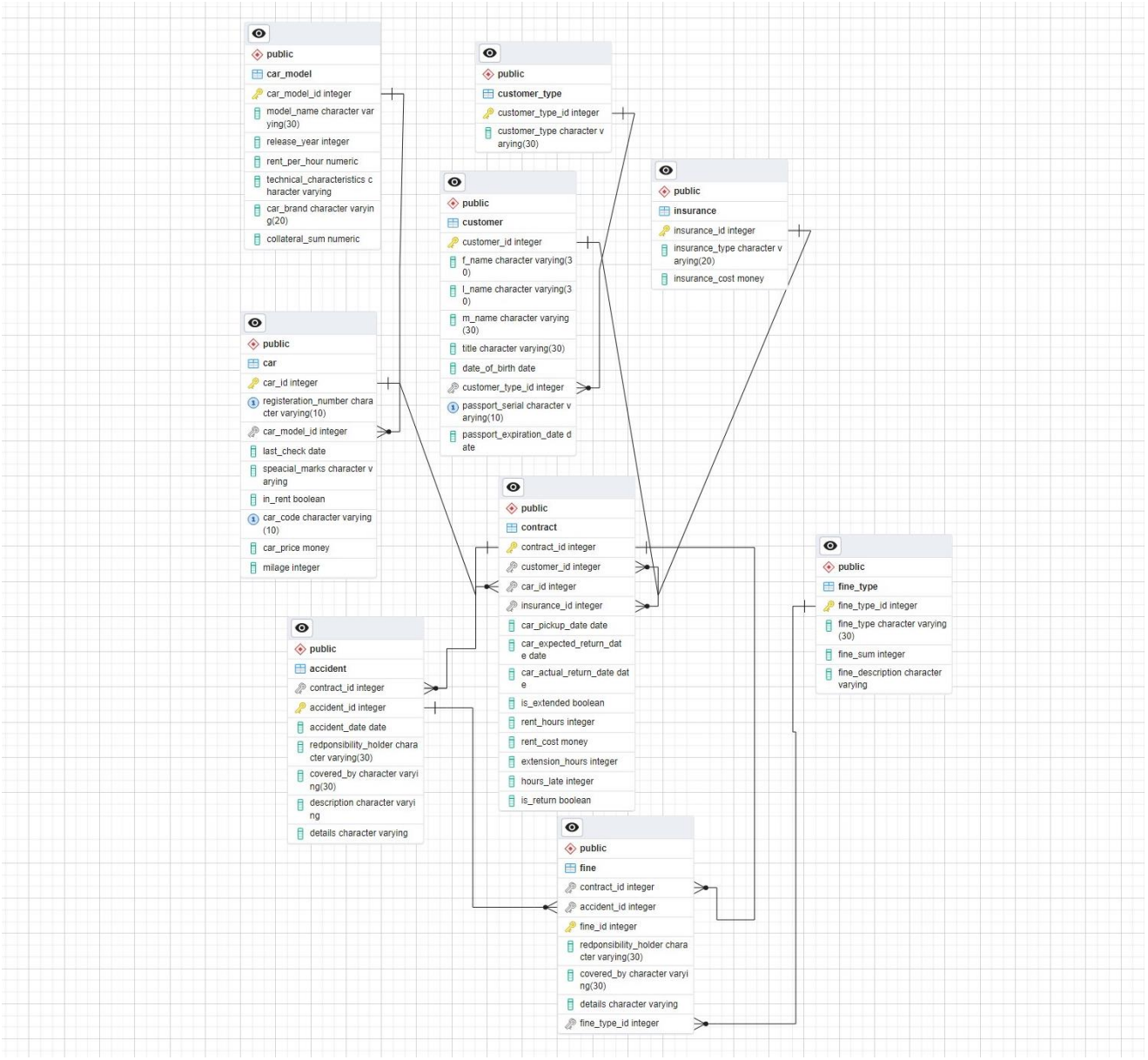
Постоянным клиентам предоставляются скидки.

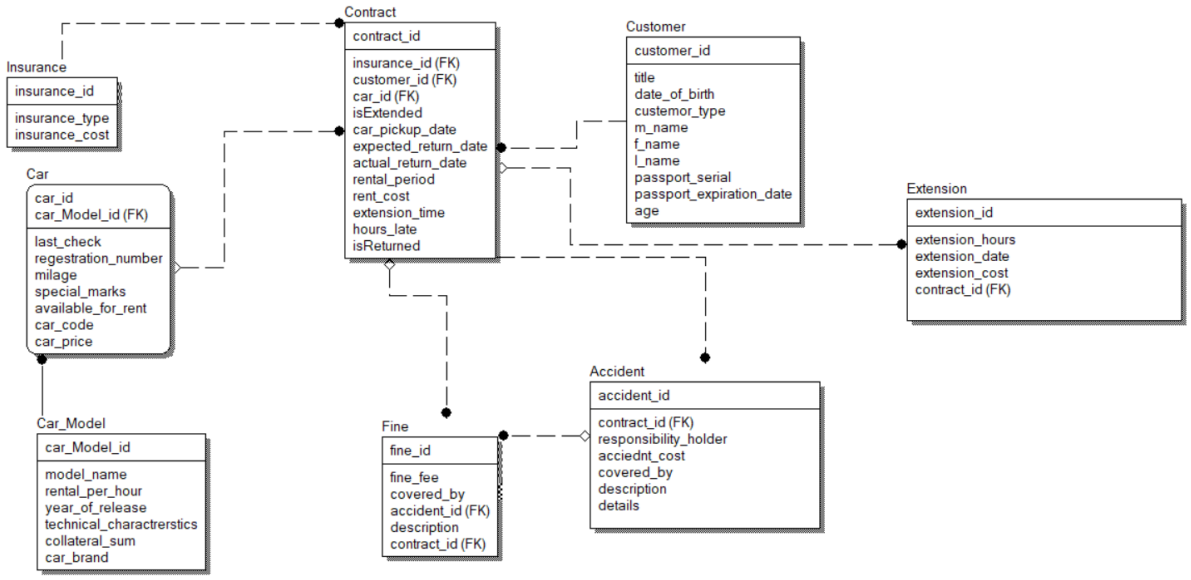
В системе необходимо хранить историю штрафов и аварий автомобилей.

Цены на прокат автомобилей могут меняться.

БД должна содержать следующий минимальный набор сведений: ФИО. Паспортные данные. Код должности. Наименование должности. Оклад. Обязанности. Код марки. Наименование. Технические характеристики. Описание. Код автомобиля. Регистрационный номер. Номер кузова. Номер двигателя. Год выпуска. Пробег. Цена автомобиля. Цена проката. Дата последнего ТО. Специальные отметки. Отметка о возврате. Код клиента. ФИО. Адрес. Телефон. Паспортные данные. Дата и время выдачи автомобиля. На сколько часов. Дата и время возврата автомобиля. Данные о нарушениях. Данные об авариях. Дата продления. Часов продления.

Схема базы данных:



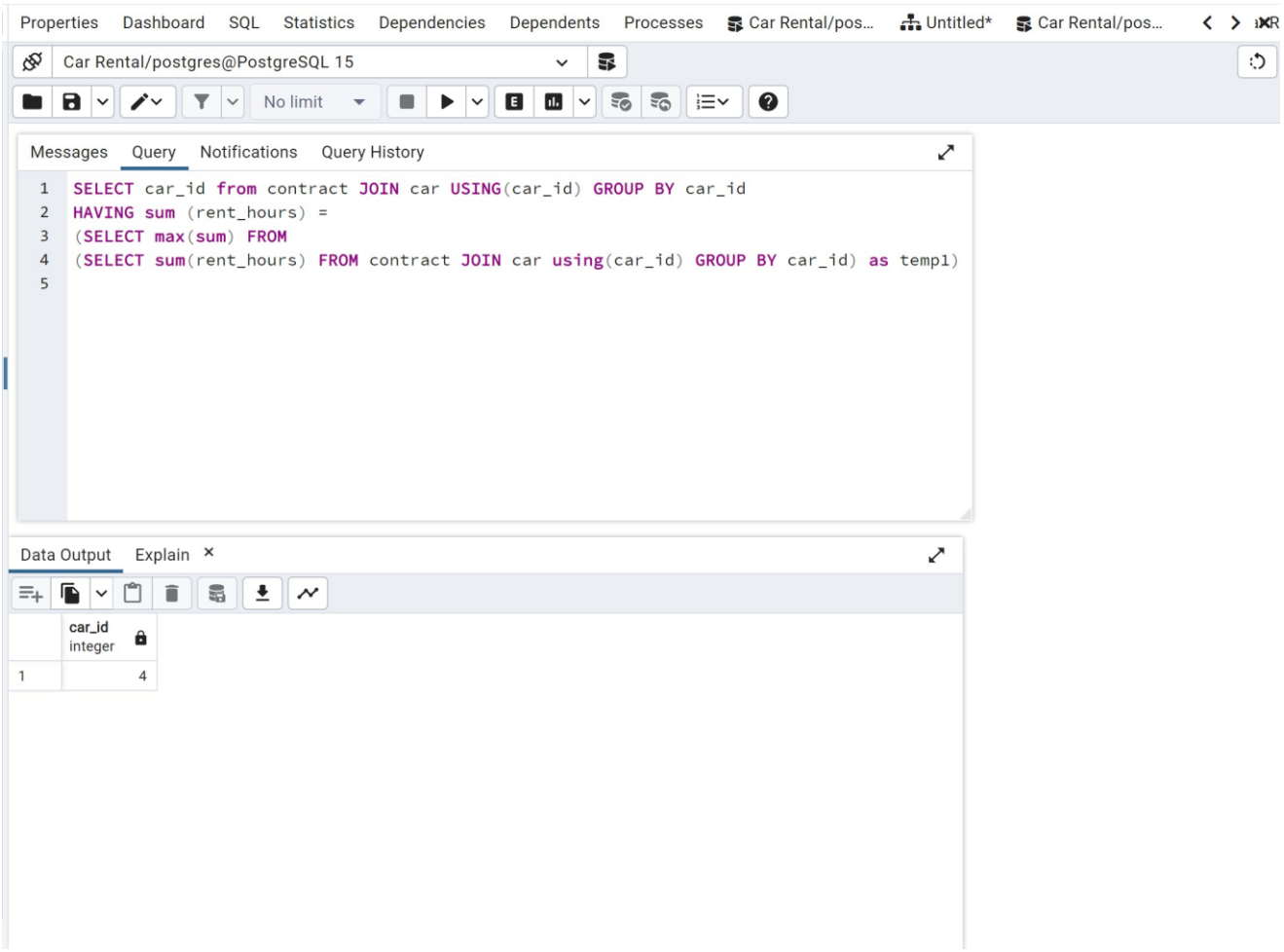


Задание 2. Создать запросы:

- Какой автомобиль находился в прокате максимальное количество часов?

SELECT car_id from contract JOIN car USING(car_id) GROUP BY car_id HAVING sum (rent_hours) =

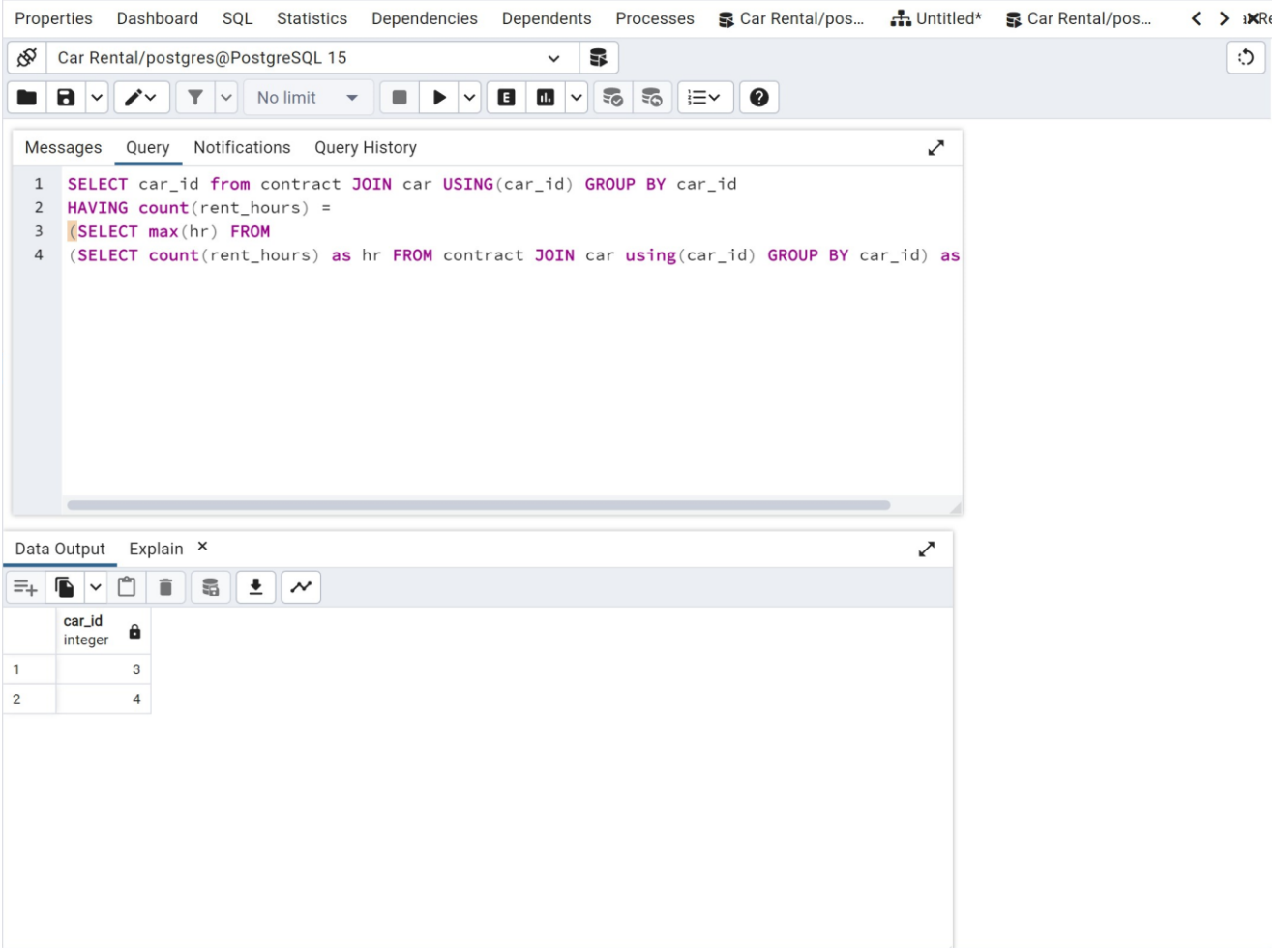
(SELECT max(sum) FROM
(SELECT sum(rent_hours) FROM contract JOIN car using(car_id) GROUP BY car_id) as temp1)



- Автомобили какой марки чаще всего брались в прокат?

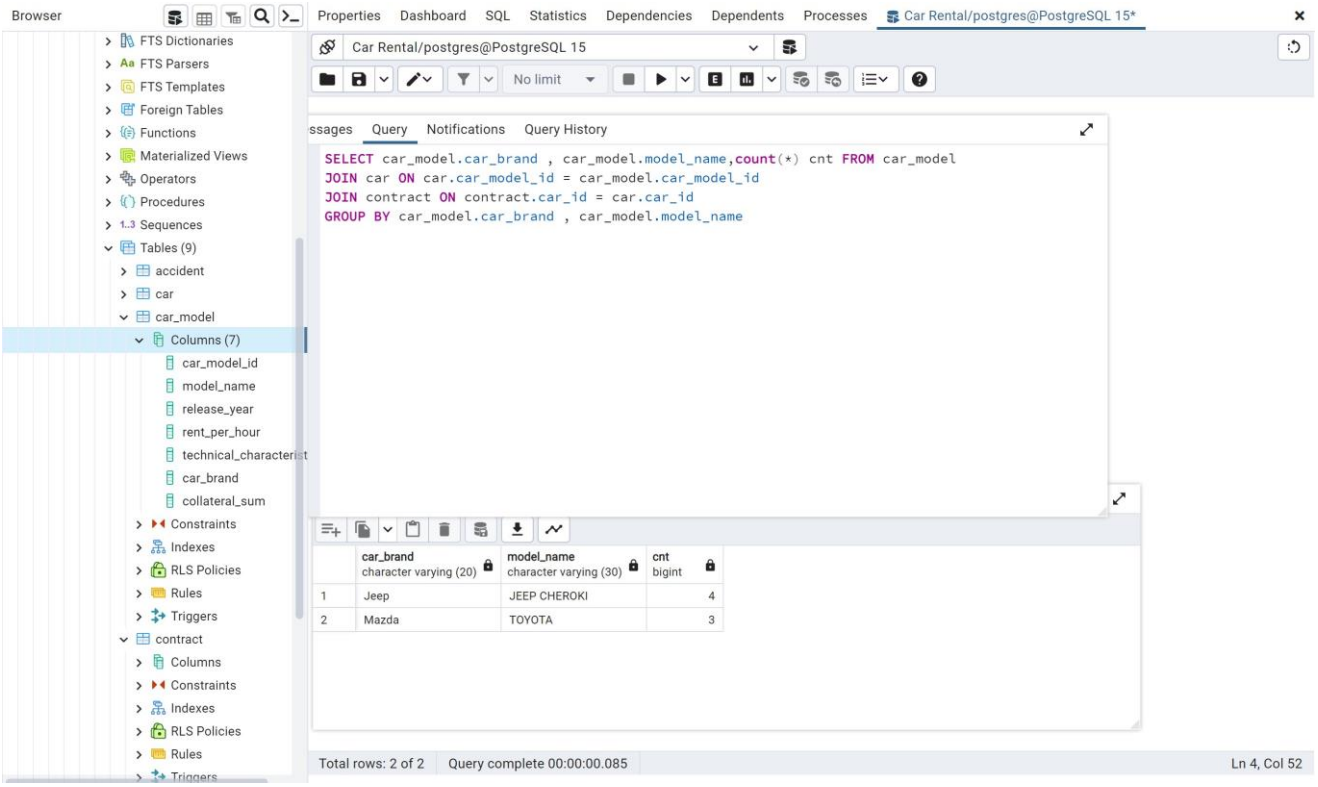
SELECT car_id from contract JOIN car USING(car_id) GROUP BY car_id
HAVING count(rent_hours) =

(SELECT max(count) FROM (SELECT sum(rent_hours) FROM contract JOIN car using(car_id) GROUP BY car_id) as temp1)



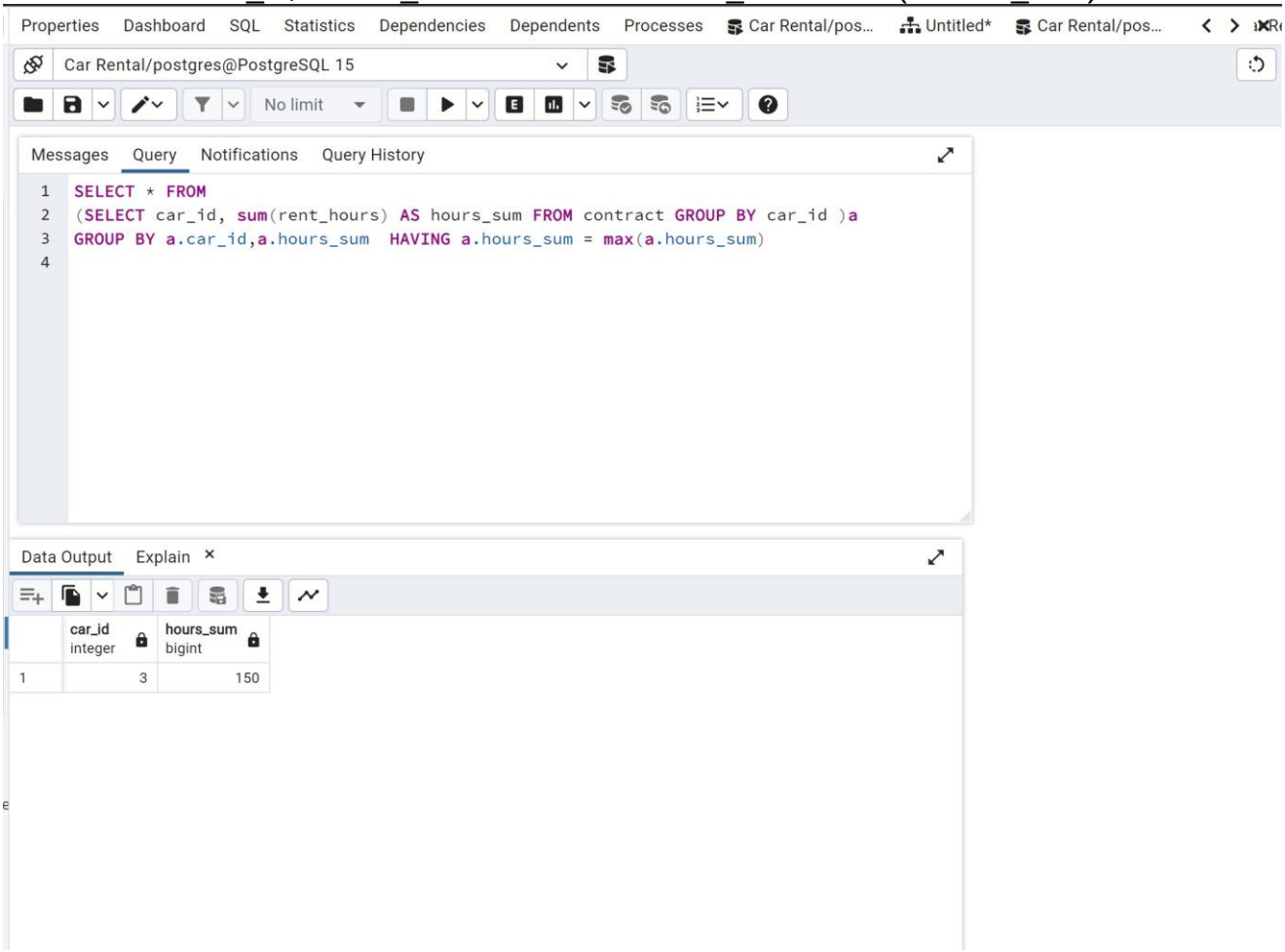
- Определить убытки от простоя автомобилей за вчерашний день.
- Вывести данные автомобиля, имеющего максимальный пробег.

SELECT * FROM car GROUP BY car_id HAVING milage = (SELECT max(milage) FROM car)



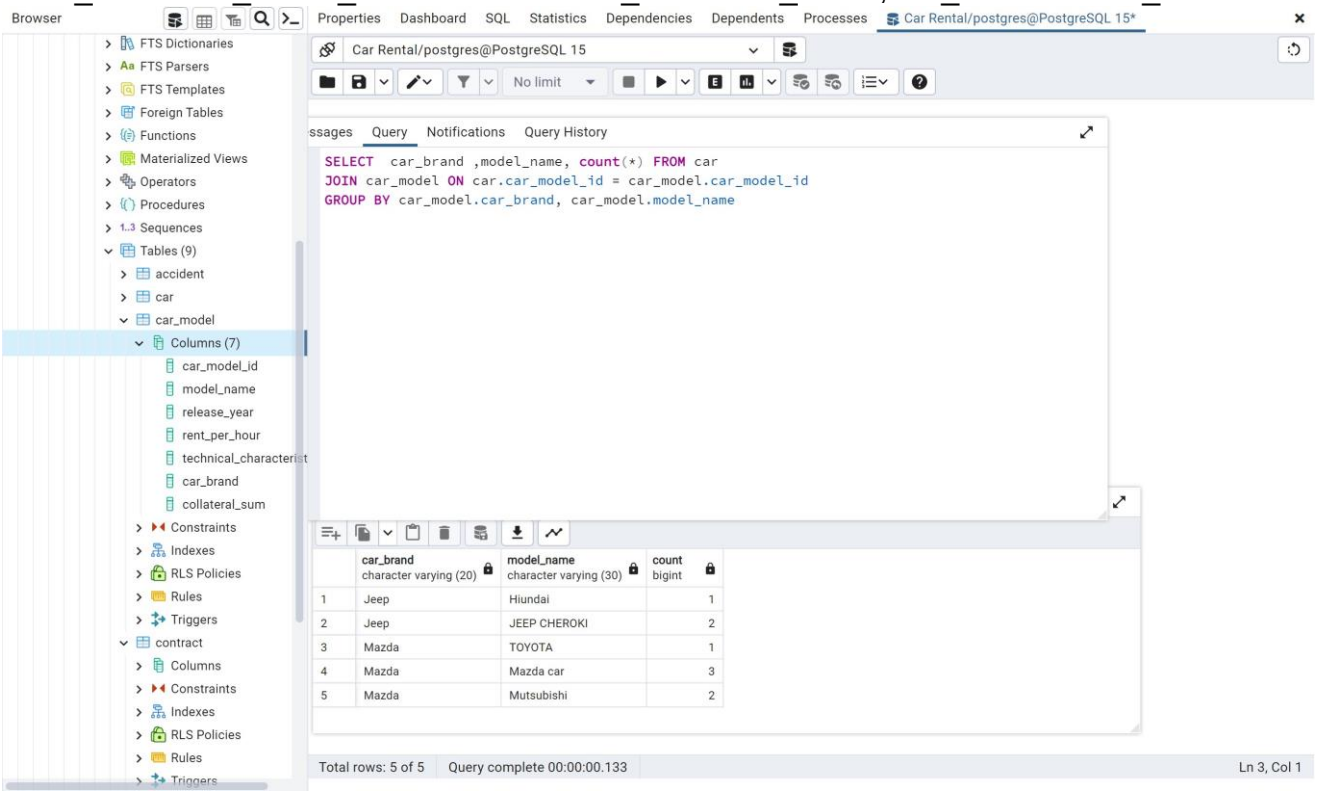
- Какой автомобиль суммарно находился в прокате дольше всех.

```
SELECT * FROM
(SELECT car_id, sum(rent_hours) AS hours_sum FROM contract GROUP BY car_id )a
GROUP BY a.car_id,a.hours_sum  HAVING a.hours_sum = max(a.hours_sum)
```



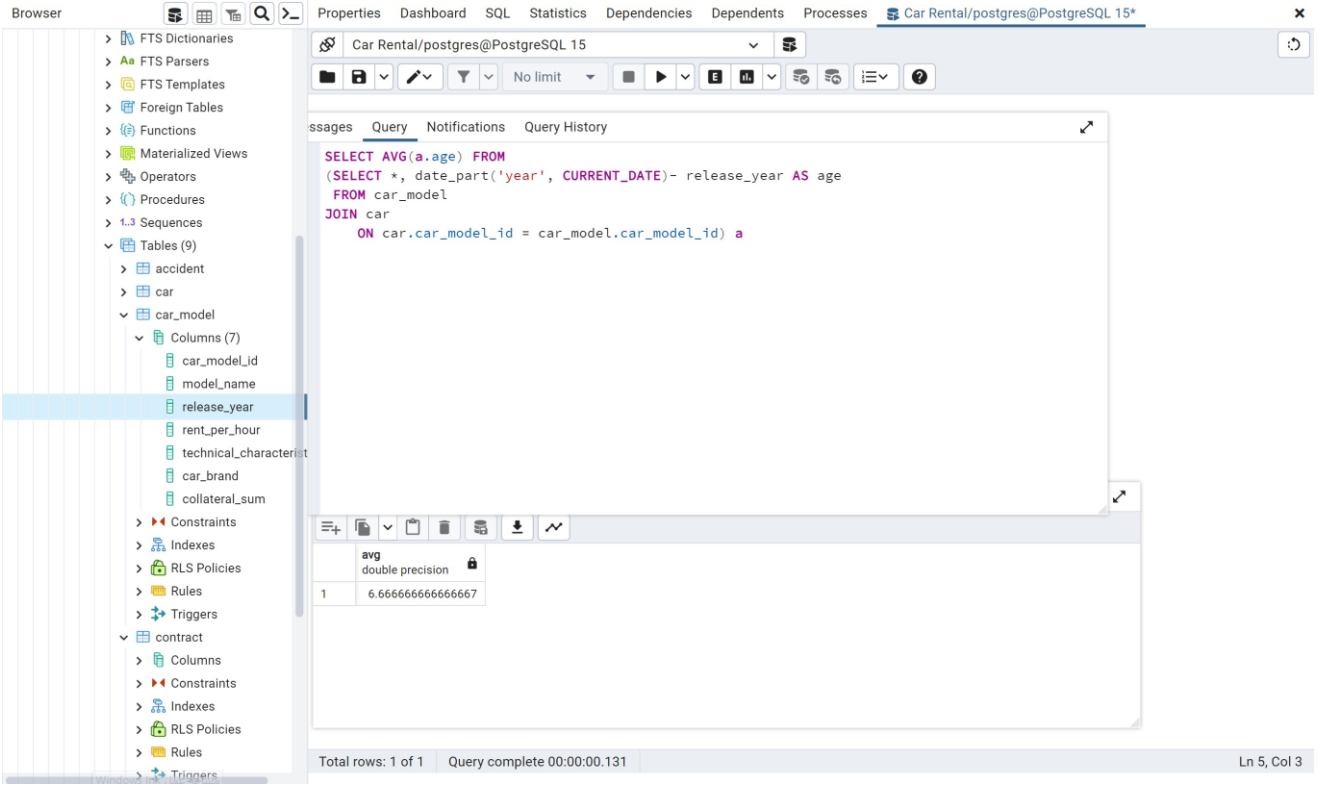
- Определить, каким количеством автомобилей каждой марки и модели владеет компания.

SELECT car_brand ,model_name, count(*) FROM car JOIN car_model ON car.car_model_id = car_model.car_model_id GROUP BY car_model.car_brand, car_model.model_name



- Определить средний “возраст” автомобилей компании.

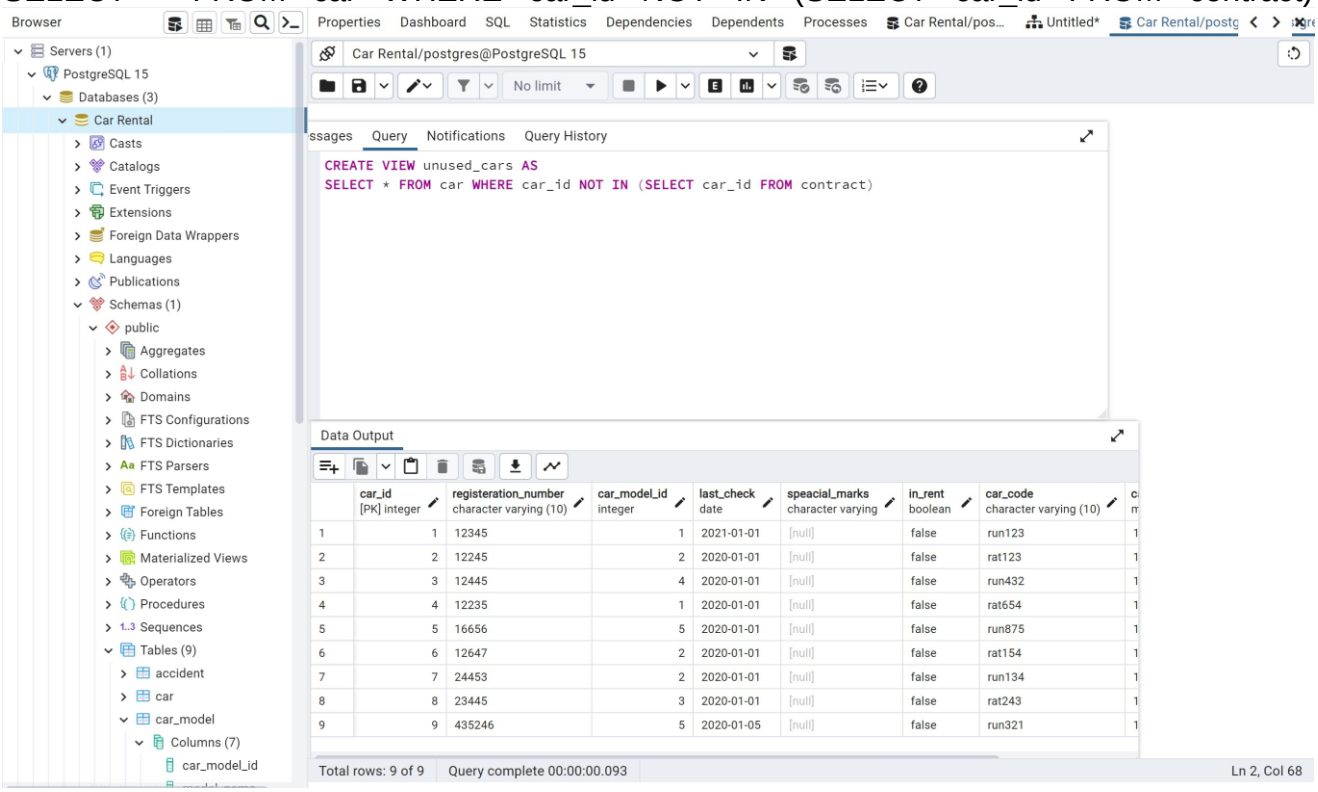
SELECT AVG(a.age) FROM (SELECT *, date_part('year', CURRENT_DATE)- release_year AS age FROM car_model
JOIN car ON car.car_model_id = car_model.car_model_id) a



Задание 3. Создать представление:

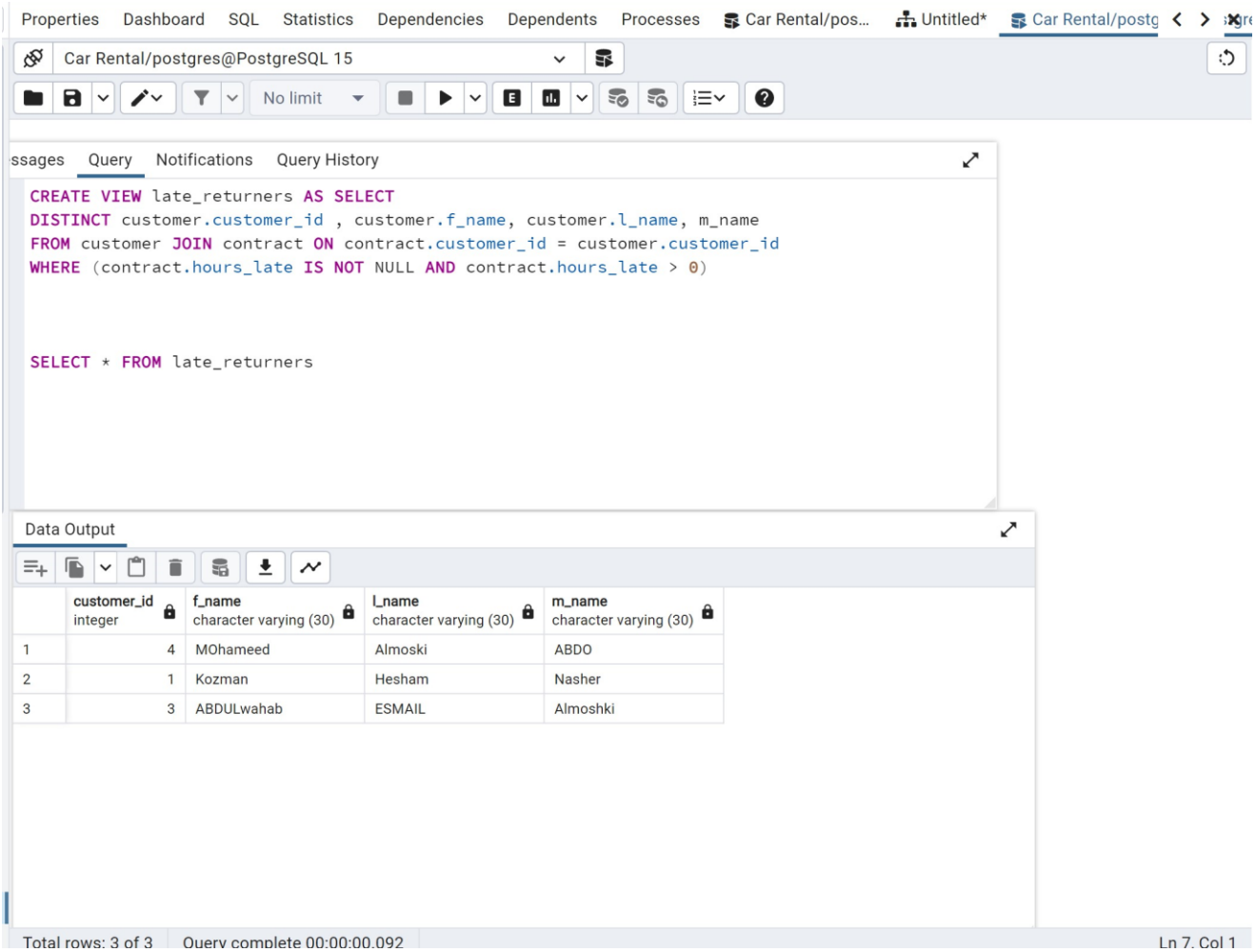
- Какой автомобиль ни разу не был в прокате?

CREATE VIEW unused_cars AS
SELECT * FROM car WHERE car_id NOT IN (SELECT car_id FROM contract)



- Вывести данные клиентов, не вернувших автомобиль вовремя.

CREATE VIEW late_returners AS SELECT
DISTINCT customer.customer_id , customer.f_name, customer.l_name, m_name
FROM customer JOIN contract ON contract.customer_id = customer.customer_id
WHERE (contract.hours_late IS NOT NULL AND contract.hours_late > 0)



2.Составить 3 запроса на модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов.

1-INSERT

Update car SET milage = ((SELECT max(milage) FROM car) +1000) WHERE car_id = 1

Properties Dashboard SQL Statistics Dependencies Dependents Processes Car Rental/pos... Untitled* Car Rental/pos...

Car Rental/postgres@PostgreSQL 15

No limit

Messages Query Notifications Query History

```
1  SELECT * FROM car
2
3  Update car SET milage = (SELECT max(milage) FROM car ) +1000 WHERE car_model = 3
4
```

Data Output Explain

	car_model_id integer	last_check date	speacial_marks character varying	in_rent boolean	car_code character varying (10)	car_price money	milage integer
1	1	2021-01-01	[null]	false	run123	12,000.00 ر.س.	6000
2	2	2020-01-01	[null]	false	rat123	11,000.00 ر.س.	6000
3	4	2020-01-01	[null]	false	run432	12,000.00 ر.س.	6000
4	1	2020-01-01	[null]	false	rat654	11,500.00 ر.س.	6000
5	5	2020-01-01	[null]	false	run875	13,000.00 ر.س.	6000
6	2	2020-01-01	[null]	false	rat154	12,000.00 ر.س.	6000
7	2	2020-01-01	[null]	false	run134	11,000.00 ر.س.	6000
8	3	2020-01-01	[null]	false	rat243	16,000.00 ر.س.	6000

Properties Dashboard SQL Statistics Dependencies Dependents Processes Car Rental/pos... Untitled* Car Rental/pos...

Car Rental/postgres@PostgreSQL 15

No limit

Messages Query Notifications Query History

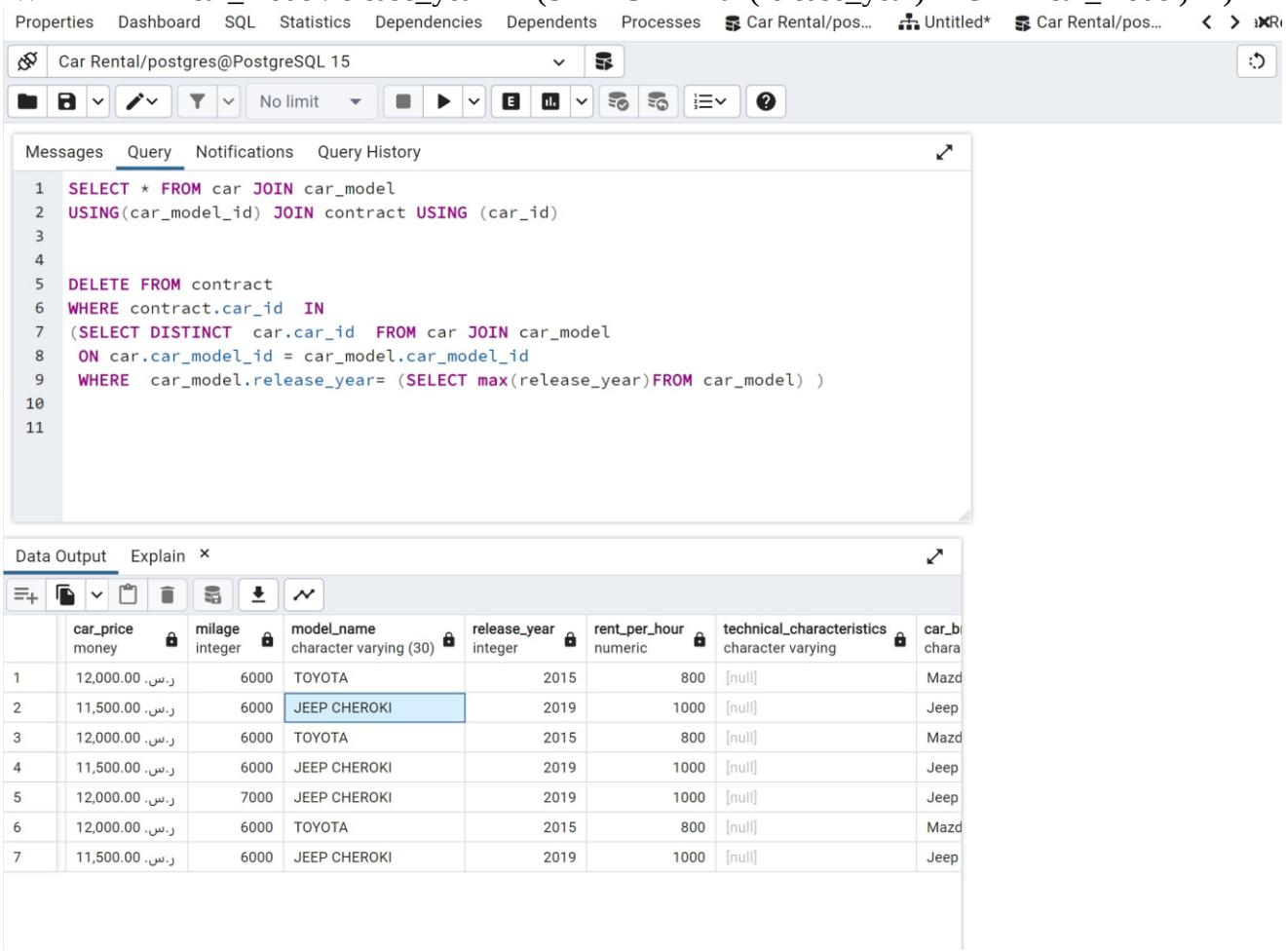
```
1  Update car SET milage = ((SELECT max(milage) FROM car ) +1000) WHERE car_id = 1
2
3  SELECT * FROM car
4
```

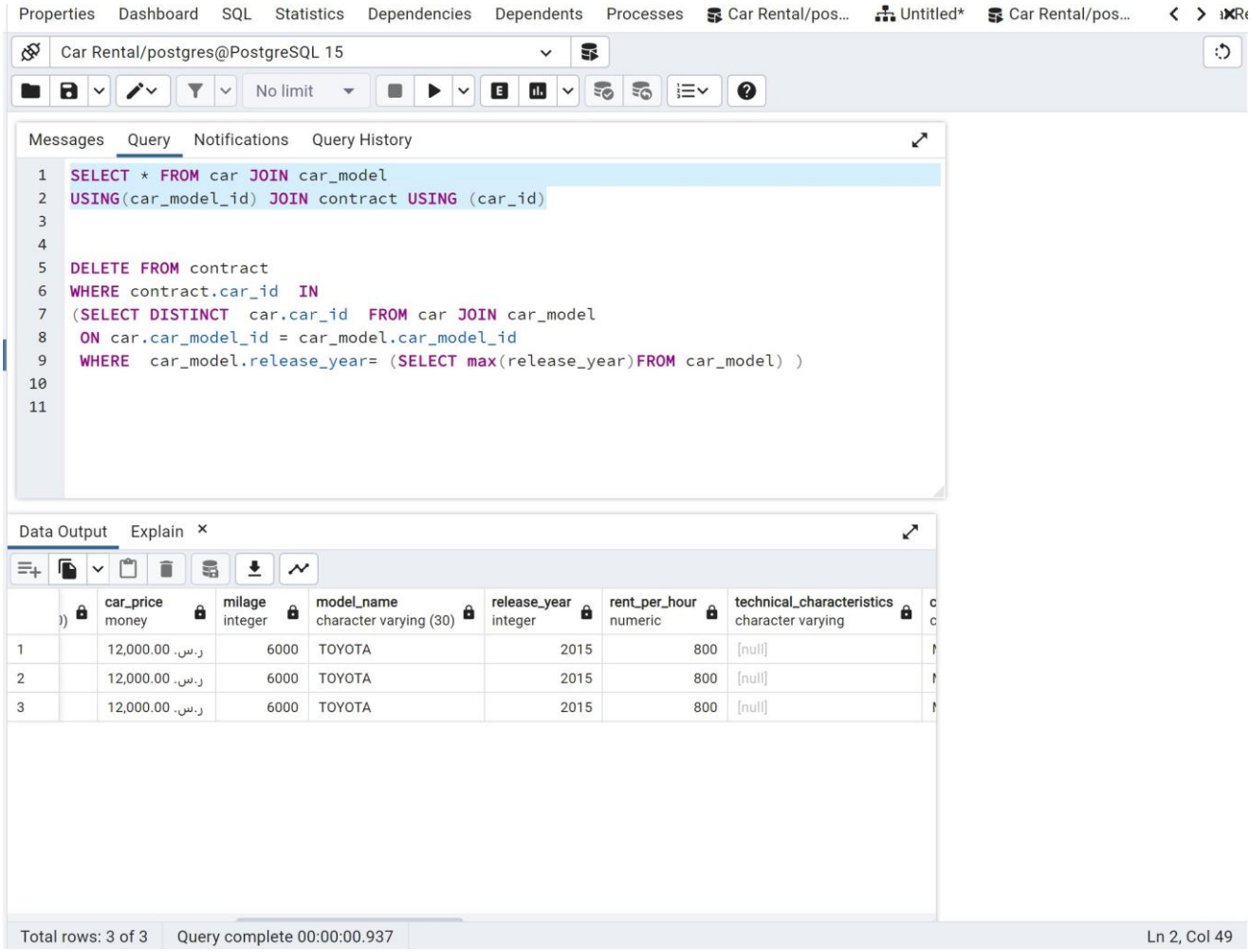
Data Output Explain

	car_model_id integer	last_check date	speacial_marks character varying	in_rent boolean	car_code character varying (10)	car_price money	milage integer
1	2	2020-01-01	[null]	false	rat123	11,000.00 ر.س.	6000
2	4	2020-01-01	[null]	false	run432	12,000.00 ر.س.	6000
3	1	2020-01-01	[null]	false	rat654	11,500.00 ر.س.	6000
4	5	2020-01-01	[null]	false	run875	13,000.00 ر.س.	6000
5	2	2020-01-01	[null]	false	rat154	12,000.00 ر.س.	6000
6	2	2020-01-01	[null]	false	run134	11,000.00 ر.س.	6000
7	3	2020-01-01	[null]	false	rat243	16,000.00 ر.س.	6000
8	1	2021-01-01	[null]	false	run123	12,000.00 ر.س.	7000

2-DELETE

```
DELETE FROM contract
WHERE contract.car_id  IN
(SELECT DISTINCT  car.car_id  FROM car JOIN car_model
 ON car.car_model_id = car_model.car_model_id
 WHERE  car_model.release_year=  (SELECT  max(release_year)FROM  car_model)  )
```





3- UPDATE

UPDATE contract SET car_actual_return_date = CURRENT_DATE WHERE contract_id = 5

Properties Dashboard SQL Statistics Dependencies Dependents Processes Car Rental/pos... Untitled* Car Rental/pos...

Car Rental/postgres@PostgreSQL 15

No limit

Messages Query Notifications Query History

```
1  ALTER TABLE contract ALTER COLUMN rent_cost TYPE INTEGER USING rent_cost::integer
2  UPDATE contract  SET  rent_cost = CAST((CAST(rent_cost AS decimal)-100) AS MONEY)
3  WHERE car_id IN
4  (SELECT contract.car_id  FROM insurance JOIN contract USING (insurance_id ) WHERE insura
5
6
7  SELECT * FROM contract
```

Data Output Explain

	car_actual_return_date date	is_extended boolean	rent_hours integer	rent_cost money	extension_hours integer	hours_late integer	is_return boolean
1	[null]	false	70	3,900.00 ر.س.	[null]	[null]	false
2	[null]	false	40	4,900.00 ر.س.	[null]	[null]	false
3	[null]	false	40	4,900.00 ر.س.	[null]	30	false

Properties Dashboard SQL Statistics Dependencies Dependents Processes Car Rental/pos... Untitled* Car Rental/pos...

Car Rental/postgres@PostgreSQL 15

No limit

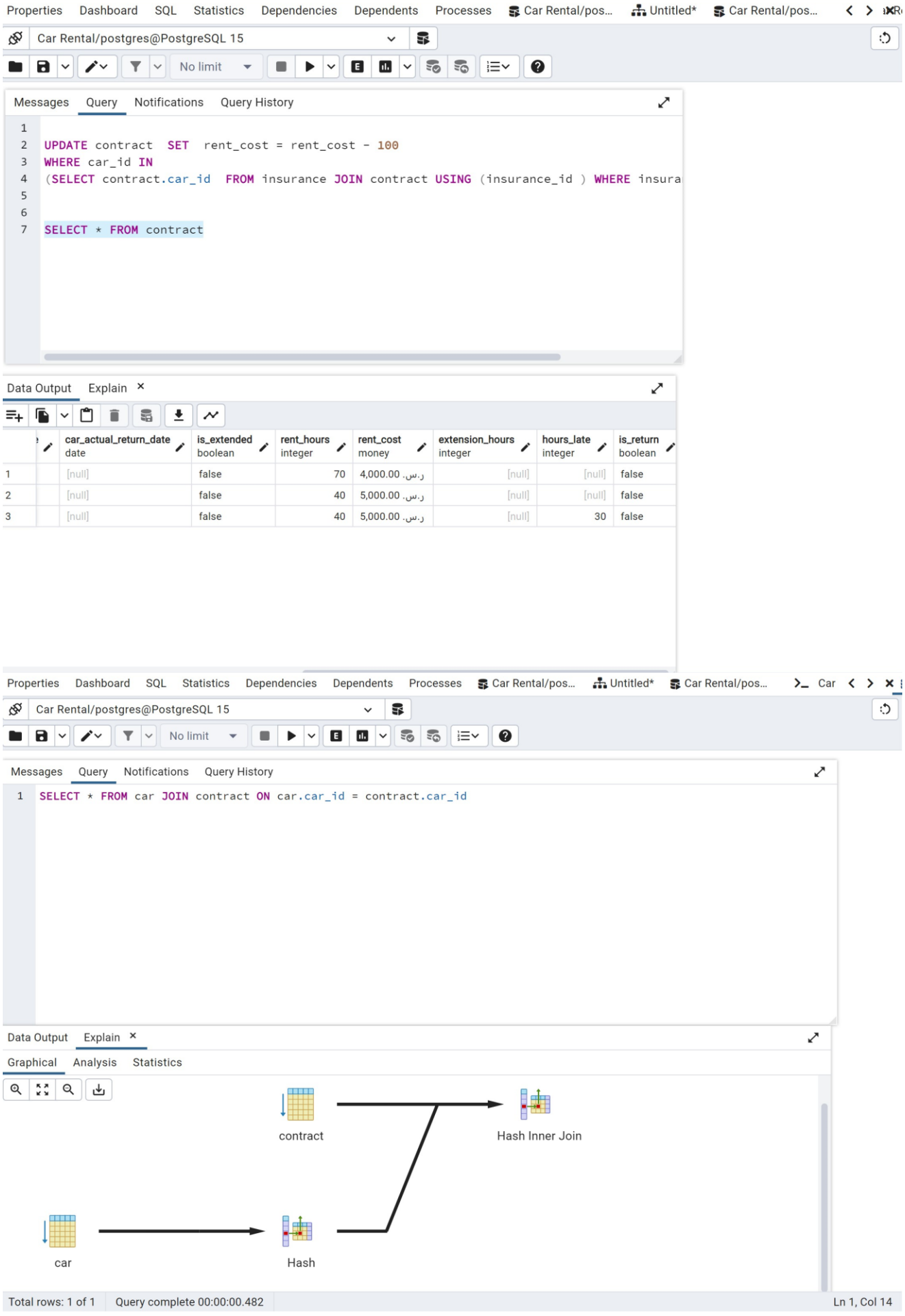
Messages Query Notifications Query History

```
1   rent_cost = CAST((CAST(rent_cost AS decimal)-100) AS MONEY)
2
3  .id  FROM insurance JOIN contract USING (insurance_id ) WHERE insurance_type = 'Casual');
4
5  :t
```

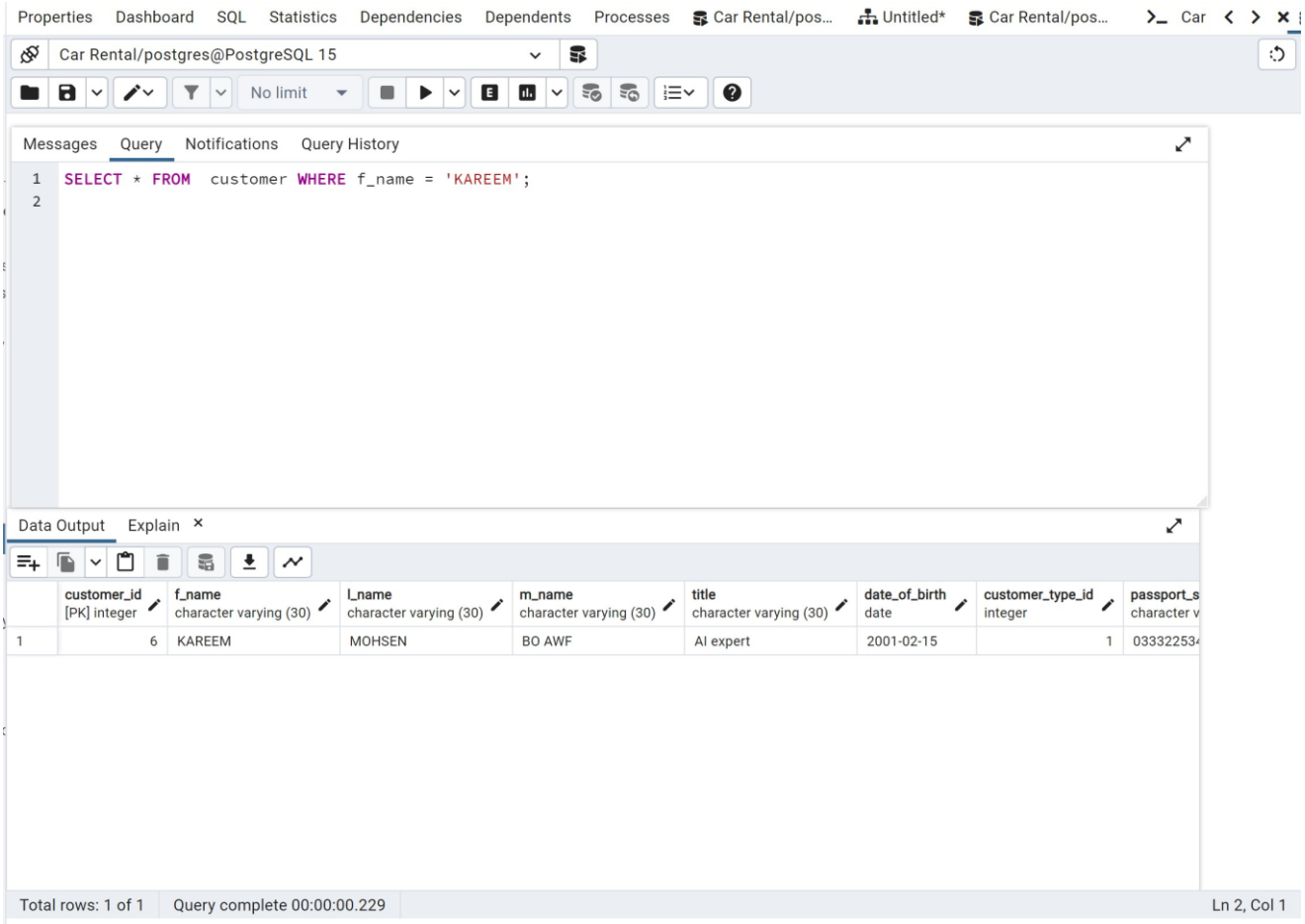
Data Output Explain

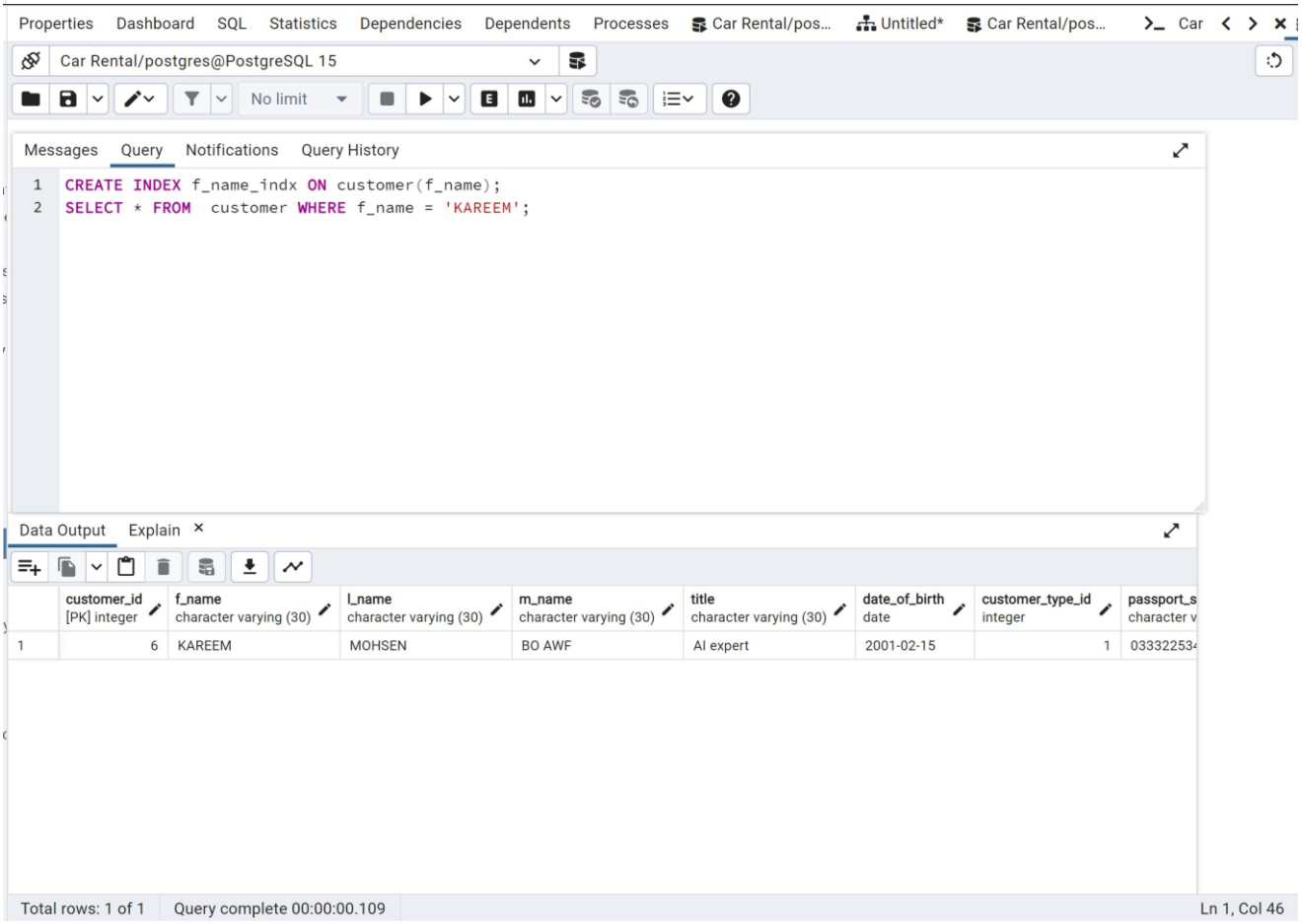
	car_actual_return_date date	is_extended boolean	rent_hours integer	rent_cost money	extension_hours integer	hours_late integer	is_return boolean
1	[null]	false	70	3,800.00 ر.س.	[null]	[null]	false
2	[null]	false	40	4,800.00 ر.س.	[null]	[null]	false
3	[null]	false	40	4,800.00 ر.س.	[null]	30	false

3. Изучить графическое представление запросов и просмотреть историю запросов.



4. Создать простой и составной индексы для двух произвольных запросов и сравнить время выполнения запросов без индексов и с индексами. Для получения плана запроса использовать команду EXPLAIN.





Индексирование сводит к минимуму время поиска, но оно может отнимать много времени в таблицах, в которые поступает много вставок.

Вывод:

В этой лабораторной работе я освоил команды DML SQL в postgresql и научился создавать сложные запросы и подзапросы.