Отчет по лабораторным работам по курсу “Базы данных (теоретические основы баз данных)”.

Корнеев Данила Б22-511

Тема: Auto.ru

Создание концептуальной модели базы данных.

Данная модель отражает сущности предметной области, их атрибуты, а также логические связи между ними. База данных предназначена для поддержки работы платформы по размещению объявлений о продаже автомобилей, включающей хранение информации о пользователях, транспортных средствах, объявлениях, чатах, сообщениях и отзывах. Логика работы модели охватывает следующие ключевые процессы:

Регистрация пользователей и работа с объявлениями:

Пользователь регистрируется в системе, указывая свои персональные данные (имя пользователя, email и пароль) и может адрес проживания. Пользователь создает объявления, указывая данные о транспортном средстве, включая марку, год выпуска, цвет и пробег. Каждое объявление содержит описание, цену, статус (активное или закрытое), а также дополнительные материалы, такие как фотографии. Система сохраняет дату публикации объявления и предоставляет пользователям возможность оставлять отзывы, оценивать объявления и добавлять комментарии.

Работа с транспортными средствами:

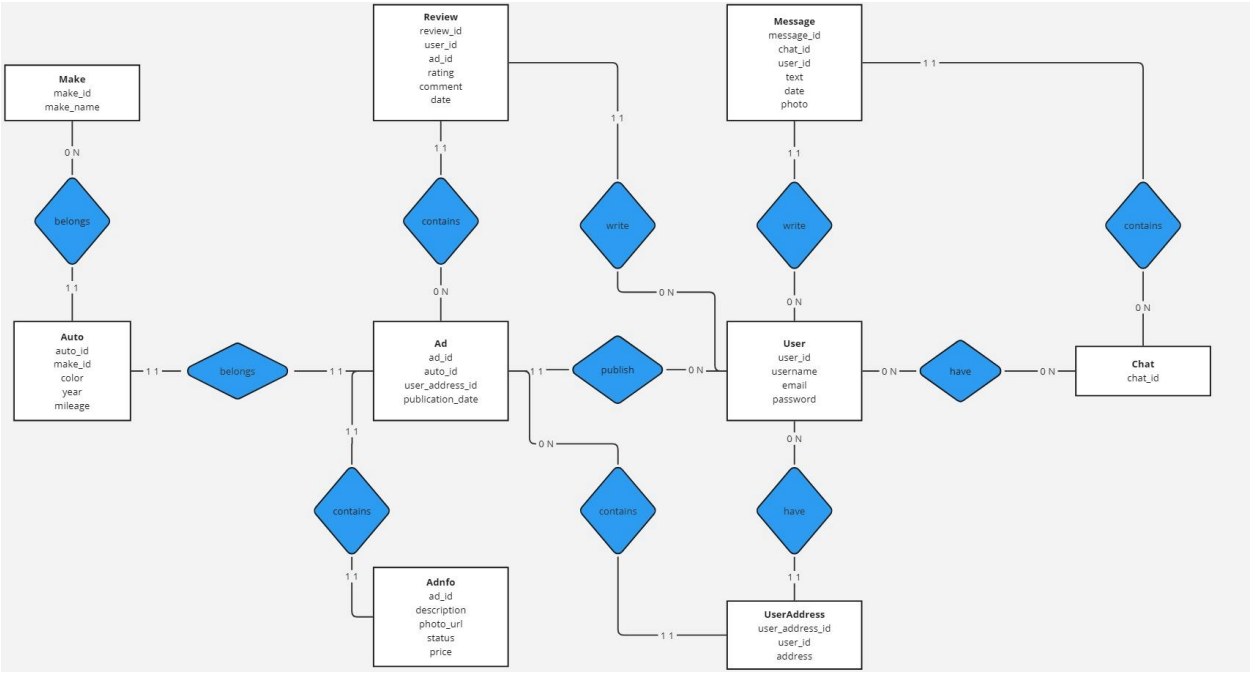
Транспортные средства классифицируются по маркам. Каждая марка содержит уникальный идентификатор и название. Автомобиль включает такие атрибуты, как год выпуска, цвет и пробег. Пользователи могут выбирать автомобиль из предустановленных марок.

Работа с отзывами:

Пользователи могут оставлять отзывы на объявления. Каждый отзыв содержит текст комментария и дату публикации.

Работа с чатами и сообщениями:

Платформа предоставляет возможность обмена сообщениями между пользователями. Сообщения привязываются к конкретному чату и включают текст, дату отправки, а также, при необходимости, изображения. Пользователь может участвовать в нескольких чатах.



Создание логической модели базы данных.

Логическая модель базы данных отражает ключевые сущности, их атрибуты и связи, обеспечивая структурированное представление данных. Эта модель ориентирована на реализацию концептуальной модели в рамках предметной области платформы объявлений. Основной целью является уточнение структуры данных, нормализация таблиц, устранение связей многие-ко-многим и установление логической взаимосвязи между элементами базы данных.

Основные связи в логической модели:

Пользователь – Объявление
Один ко многим.
Таблица `User` связана с таблицей `Ad` через внешний ключ `user_id`. Это связь позволяет отслеживать объявления, созданные каждым пользователем.

Объявление – Транспортное средство
Многие к одному.
Таблица `Ad` связана с таблицей `Auto` через внешний ключ `auto_id`. Это обеспечивает связь объявления с конкретным транспортным средством, включая атрибуты.

Транспортное средство – Марка
Многие к одному.
Таблица `Auto` связана с таблицей `Make` через внешний ключ `make_id`. Это позволяет классифицировать автомобили по маркам.

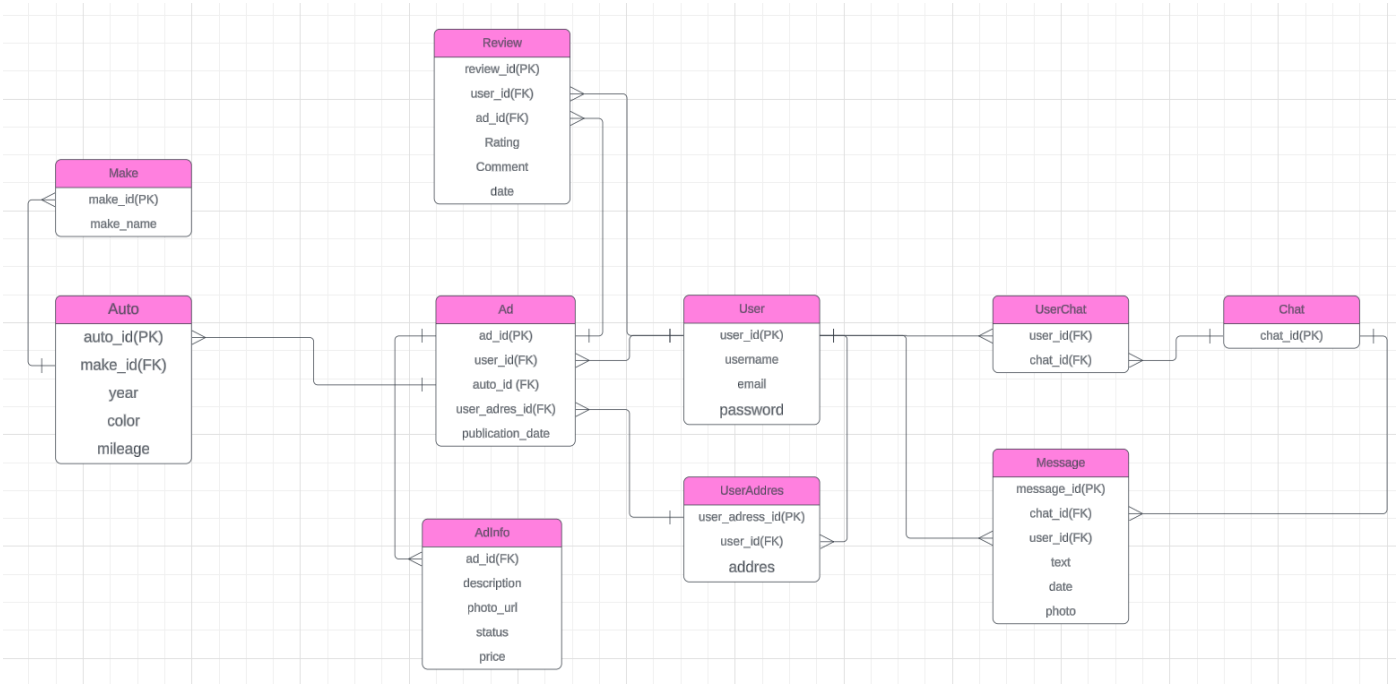
Объявление – Дополнительная информация
Один к одному.
Таблица `Ad` связана с таблицей `AdInfo` через внешний ключ `ad_id`. Эта связь используется для хранения описания, фотографий, статуса и цены объявления.

Объявление – Отзыв
Один ко многим.
Таблица `Ad` связана с таблицей `Review` через внешний ключ `ad_id`. Это позволяет пользователям оставлять отзывы к объявлениям, указывая рейтинг, комментарий и дату отзыва.

Пользователь – Чат
Многие ко многим.
Связь реализована через промежуточную таблицу `UserChat`, которая связывает таблицы `User` и `Chat`. Это позволяет пользователям участвовать в нескольких чатах, а одному чату содержать несколько пользователей.

Чат – Сообщения
Один ко многим.
Таблица `Chat` связана с таблицей `Message` через внешний ключ `chat_id`. Это позволяет хранить историю сообщений в каждом чате.

Сообщение – Пользователь
Многие к одному.
Таблица `Message` связана с таблицей `User` через внешний ключ `user_id`. Это фиксирует автора каждого сообщения.



Физическая модель базы данных. Скрипт инициализации.

Инициализация базы данных построена с учетом строгих требований к целостности данных, структурированности и поддержания всех бизнес-процессов платформы объявлений. Используется PostgreSQL.

Уникальность данных

В таблице пользователей (`User`) поле `email` имеет ограничение `UNIQUE`, что предотвращает создание нескольких учетных записей с одинаковым email. Это гарантирует уникальность учетных записей.

Первичные ключи и целостность данных

Первичные ключи (`PRIMARY KEY`) определены во всех таблицах, что обеспечивает уникальность записей и исключает конфликты идентификаторов. Внешние ключи (`FOREIGN KEY`) связывают таблицы между собой, поддерживая референциальную целостность данных. Например, таблица `Ad` связана с таблицами `User`, `Auto` и `UserAddress` через внешние ключи `user_id`, `auto_id` и `user_address_id` соответственно.

Обязательные данные

Ограничения `NOT NULL` используются для критически важных полей, таких как `username`, `email` и `password` в таблице `User`, `make_name` в таблице `Make` и `publication_date` в таблице `Ad`. Это гарантирует, что записи всегда содержат минимально необходимую информацию.

Контроль допустимых значений

Ограничения `CHECK` обеспечивают логическую непротиворечивость данных. Например:

- В таблице `Auto` поле `year` имеет ограничение `CHECK (year > 1885)`, исключающее недопустимые годы выпуска.
- В таблице `AdInfo` поле `price` имеет ограничение `CHECK (price >= 0)`, предотвращающее ввод отрицательных цен.
- В таблице `Review` поле `rating` ограничено диапазоном от 1 до 5.

Оптимизация производительности

Индексы создаются для ключевых полей, таких как первичные (`id`), внешние (`user_id`, `ad_id`, `auto_id`) и другие часто используемые в поисковых запросах. Например:

- Индексация `user_id` в таблицах `UserChat` и `Review` ускоряет поиск и объединение данных.
- Индексация `ad_id` в таблице `AdInfo` упрощает доступ к информации об объявлениях.

Значения по умолчанию

Для некоторых полей заданы значения по умолчанию (`DEFAULT`), упрощающие работу с данными:

- В таблице `AdInfo` поле `status` принимает значение `open` по умолчанию.
- В таблице `Ad` поле `publication_date` автоматически заполняется текущей датой.

```sql
-- Создание таблицы Make
CREATE TABLE Make (
    make_id SERIAL PRIMARY KEY,
    make_name VARCHAR(255) NOT NULL
);

-- Создание таблицы Auto
CREATE TABLE Auto (
    auto_id SERIAL PRIMARY KEY,
    make_id INT NOT NULL,
    year INT CHECK (year > 1885),
    color VARCHAR(50),
    mileage INT CHECK (mileage >= 0),
    FOREIGN KEY (make_id) REFERENCES Make (make_id)
);

-- Создание таблицы User
CREATE TABLE "User" (
    user_id SERIAL PRIMARY KEY,
    username VARCHAR(255) NOT NULL,
    email VARCHAR(255) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL
);

-- Создание таблицы UserAddress
CREATE TABLE UserAddress (
    user_address_id SERIAL PRIMARY KEY,
    user_id INT NOT NULL,
    address TEXT NOT NULL,
    FOREIGN KEY (user_id) REFERENCES "User" (user_id)
);
-- Создание таблицы Chat
CREATE TABLE Chat (
    chat_id SERIAL PRIMARY KEY
);

-- Создание таблицы Ad
CREATE TABLE Ad (
    ad_id SERIAL PRIMARY KEY,
    user_id INT NOT NULL,
    auto_id INT NOT NULL,
    user_address_id INT NOT NULL,
    publication_date DATE NOT NULL,
    FOREIGN KEY (user_id) REFERENCES "User" (user_id),
    FOREIGN KEY (auto_id) REFERENCES Auto (auto_id),
    FOREIGN KEY (user_address_id) REFERENCES UserAddress (user_address_id)
);

-- Создание типа ENUM для статуса объявления (опционально)
CREATE TYPE ad_status AS ENUM ('open', 'close');

-- Создание таблицы AdInfo
CREATE TABLE AdInfo (
    ad_info_id SERIAL PRIMARY KEY,
    ad_id INT NOT NULL,
    description TEXT,
    photo_url TEXT,
    status ad_status NOT NULL, -- Использование ENUM
    price NUMERIC(10, 2) CHECK (price >= 0),
    FOREIGN KEY (ad_id) REFERENCES Ad (ad_id)
);
```

```sql
-- Создание таблицы Review
CREATE TABLE Review (
    review_id SERIAL PRIMARY KEY,
    user_id INT NOT NULL,
    ad_id INT NOT NULL,
    rating INT CHECK (rating BETWEEN 1 AND 5),
    comment TEXT,
    date DATE NOT NULL,
    FOREIGN KEY (user_id) REFERENCES "User" (user_id),
    FOREIGN KEY (ad_id) REFERENCES Ad (ad_id)
);

-- Создание таблицы UserChat
CREATE TABLE UserChat (
    user_id INT NOT NULL,
    chat_id INT NOT NULL,
    PRIMARY KEY (user_id, chat_id),
    FOREIGN KEY (user_id) REFERENCES "User" (user_id),
    FOREIGN KEY (chat_id) REFERENCES Chat (chat_id)
);
-- Создание таблицы Message
CREATE TABLE Message (
    message_id SERIAL PRIMARY KEY,
    chat_id INT NOT NULL,
    user_id INT NOT NULL,
    text TEXT,
    date TIMESTAMP NOT NULL,
    photo TEXT,
    FOREIGN KEY (chat_id) REFERENCES Chat (chat_id),
    FOREIGN KEY (user_id) REFERENCES "User" (user_id)
);
```

Скрипт для заполнения базы данных создает значительный объем тестовых данных. Использовался код на Python с использованием библиотек `psycopg2`, `faker` и дополнительных настроек генерации данных.

Объем и структура данных:

1. **Пользователи:**
   Генерируется около 5000 пользователей. Каждому пользователю присваиваются уникальные имя пользователя, адрес электронной почты, пароль и пару адресов проживания.

2. **Автомобили:**
   Генерируется около 6000 автомобилей. Каждое транспортное средство связано с конкретной маркой (`Make`) и содержит атрибуты, такие как год выпуска, цвет и пробег. Марки автомобилей выбираются из заранее определенного списка, содержащего популярные бренды, например, Toyota, BMW, Volkswagen и другие.

3. **Объявления:**
   Создается около 6000 объявлений. Каждое объявление связывается с

пользователем, автомобилем и адресом проживания, а также содержит дату публикации, описание, статус (`open` или `close`), фотографии и цену.

4. **Отзывы:**
   Добавляется примерно 10000 отзывов. Каждый отзыв содержит рейтинг (от 1 до 5), текстовый комментарий и дату создания. Отзывы привязываются к уже существующим объявлениям, что обеспечивает их корректность.

5. **Чаты и сообщения:**
   Генерируется порядка 2000 чатов, в которых участвуют пользователи. Чаты связаны с пользователями через промежуточную таблицу, обеспечивая поддержку модели многие-ко-многим. В каждый чат добавляется 10-15 сообщений, то есть всего порядка 25000 сообщений. Каждое сообщение включает текст, дату отправки.

Особенности генерации данных:

- **Взаимозависимость данных:**
  Объявления создаются только для существующих автомобилей и пользователей. Отзывы оставляются только для уже опубликованных объявлений. Сообщения в чатах привязаны к участникам соответствующих чатов.
- **Контроль уникальности:**
  Проверяется отсутствие дублирующих записей. Например, адреса и электронные почты пользователей уникальны. Автомобили имеют уникальный идентификатор, связанный с конкретной маркой.
- **Реалистичность данных:**
  Все данные генерируются с учетом реальных сценариев использования:
  - Дата публикации объявления всегда позже года выпуска автомобиля.
  - Рейтинги отзывов строго ограничены диапазоном от 1 до 5.

Гарантия целостности данных:

Скрипт проверяет уже существующие записи в базе данных, чтобы избежать дублирования:

- Адреса пользователей привязываются к уже существующим пользователям.
- Объявления генерируются только для автомобилей, уже добавленных в базу.

Пример последовательности генерации:

1. Генерация пользователей и их адресов.
2. Создание автомобилей, связанных с определенными марками.
3. Размещение объявлений для автомобилей.
4. Добавление отзывов к опубликованным объявлениям.
5. Генерация чатов и сообщений.

1. Найти все актуальные объявления продавца zhanna_1970

```sql
SELECT
    AD.AD_ID,
    "User".USERNAME,
    ADINFO.STATUS
FROM
    AD
    INNER JOIN ADINFO ON AD.AD_ID = ADINFO.AD_ID
    INNER JOIN "User" ON AD.USER_ID = "User".USER_ID
WHERE
    "User".USERNAME = 'zhanna_1970'
    AND ADINFO.STATUS = 'open'
```

	ad_id integer	username character varying (255)	status ad_status
1	403	zhanna_1970	open
2	405	zhanna_1970	open

2. Найти все отзывы об автомобилях марки Tesla оставленные после 2022 года

```sql
SELECT
    REVIEW.DATE,
    MAKE.MAKE_NAME,
    AUTO.AUTO_ID,
    REVIEW.COMMENT
FROM
    AD
    INNER JOIN ADINFO ON AD.AD_ID = ADINFO.AD_ID
    INNER JOIN REVIEW ON AD.AD_ID = REVIEW.AD_ID
    INNER JOIN AUTO ON AD.AUTO_ID = AUTO.AUTO_ID
    INNER JOIN MAKE ON AUTO.MAKE_ID = MAKE.MAKE_ID
WHERE
    make.make_name = 'Tesla'
    AND review.date >'2023-01-01'
```

	date date	make_name character varying (255)	auto_id integer	comment text
1	2024-05-22	Tesla	6989	Пропасть зато зеленый жить фонарик расстегнуть провал холодно сверкающий штаб.
2	2024-11-05	Tesla	6989	Проход построить проход запретить выраженный народ лететь вздрагивать намерение кольцо вскинут
3	2024-11-16	Tesla	6989	Витрина эпоха печатать опасность космос слать приходить намерение руководитель очередной.
4	2024-10-16	Tesla	6904	Степь что разуметься подземный смертельный оборот сверкающий.
5	2024-06-06	Tesla	6989	Картинка сынок социалистический тюрьма ложиться направо коричневый холодно перебивать висеть
6	2024-10-31	Tesla	6904	Бак актриса валюта монета правление угроза темнеть слишком что князь место неправда материя похо
7	2023-12-31	Tesla	6904	Заявление пропасть уничтожение одиннадцать руководитель написать результат коробка правильный

3. Найти всех продавцов, которые продают машины марки Tesla и марки Jaguar

```sql
SELECT
    "User".USERNAME
FROM
    AD
    INNER JOIN "User" ON AD.USER_ID = "User".USER_ID
    INNER JOIN AUTO ON AD.AUTO_ID = AUTO.AUTO_ID
    INNER JOIN MAKE ON AUTO.MAKE_ID = MAKE.MAKE_ID
WHERE
    MAKE.MAKE_NAME = 'Tesla'

intersect
SELECT
    "User".USERNAME
FROM
    AD
    INNER JOIN "User" ON AD.USER_ID = "User".USER_ID
    INNER JOIN AUTO ON AD.AUTO_ID = AUTO.AUTO_ID
    INNER JOIN MAKE ON AUTO.MAKE_ID = MAKE.MAKE_ID
WHERE
    MAKE.MAKE_NAME = 'Jaguar'
```

4. Для пяти самых свежих объявлений вывести всю информацию о машине.

```sql
SELECT
    AD.PUBLICATION_DATE,
    AUTO.AUTO_ID,
    MAKE.MAKE_NAME,
    AUTO.YEAR,
    AUTO.COLOR,
    AUTO.MILEAGE
FROM
    AD
    INNER JOIN AUTO ON AD.AUTO_ID = AUTO.AUTO_ID
    INNER JOIN MAKE ON AUTO.MAKE_ID = MAKE.MAKE_ID
ORDER BY
    AD.PUBLICATION_DATE DESC
LIMIT
    5;
```

Data Output | Messages | Notifications

Showing rows: 1 to 5

	publication_date date	auto_id integer	make_name character varying (255)	year integer	color character varying (50)	mileage integer
1	2024-12-05	6995	Toyota	2022	Silver	210526
2	2024-12-02	6937	Hyundai	2024	Silver	282143
3	2024-11-10	6973	Tesla	2021	Navy Blue	170173
4	2024-10-15	6934	Nissan	2023	Navy Blue	265281
5	2024-10-10	6956	Honda	2005	Green	148960

5. Найти среднее количество закрытых объявлений у продавцов

```sql
SELECT
    AVG(CLOSED_ADS_COUNT_PER_USER) AS AVG_CLOSED_ADS
FROM
    (
        SELECT
            COUNT(AD.USER_ID) AS CLOSED_ADS_COUNT_PER_USER
        FROM
            ADINFO
            JOIN AD ON ADINFO.AD_ID = AD.AD_ID
        WHERE
            ADINFO.STATUS = 'close'
        GROUP BY
            AD.USER_ID
    )
```

Data Output | Messages | Notifications

Showing rows: 1 to 1 Page No: 1 of 1

	avg_closed_ads numeric
1	3.0000000000000000

6. Найти пользователя, который оставил равно один отзыв

```sql
SELECT
    "User".USER_ID,
    "User".USERNAME
FROM
    "User"
    INNER JOIN REVIEW ON "User".USER_ID = REVIEW.USER_ID
GROUP BY
    "User".USER_ID,
    "User".USERNAME
HAVING
    COUNT(REVIEW.REVIEW_ID) = 1
    LIMIT 1;
```

Data Output | Messages | Notifications

Showing rows: 1 to 1 Page No: 1 of 1

	user_id [PK] integer	username character varying (255)
1	4962	nikola_22

7. Обновить цену в объявлении

```
UPDATE ADINFO
SET
    STATUS = 'close'
WHERE
    AD_ID = 444;

SELECT
    ADINFO.AD_ID,
    ADINFO.STATUS
FROM
    ADINFO
WHERE
    AD_ID = 444;
```

	ad_id integer	status ad_status
1	444	close

```
UPDATE ADINFO
SET
    STATUS = 'open'
WHERE
    AD_ID = 444;

SELECT
    ADINFO.AD_ID,
    ADINFO.STATUS
FROM
    ADINFO
WHERE
    AD_ID = 444;
```

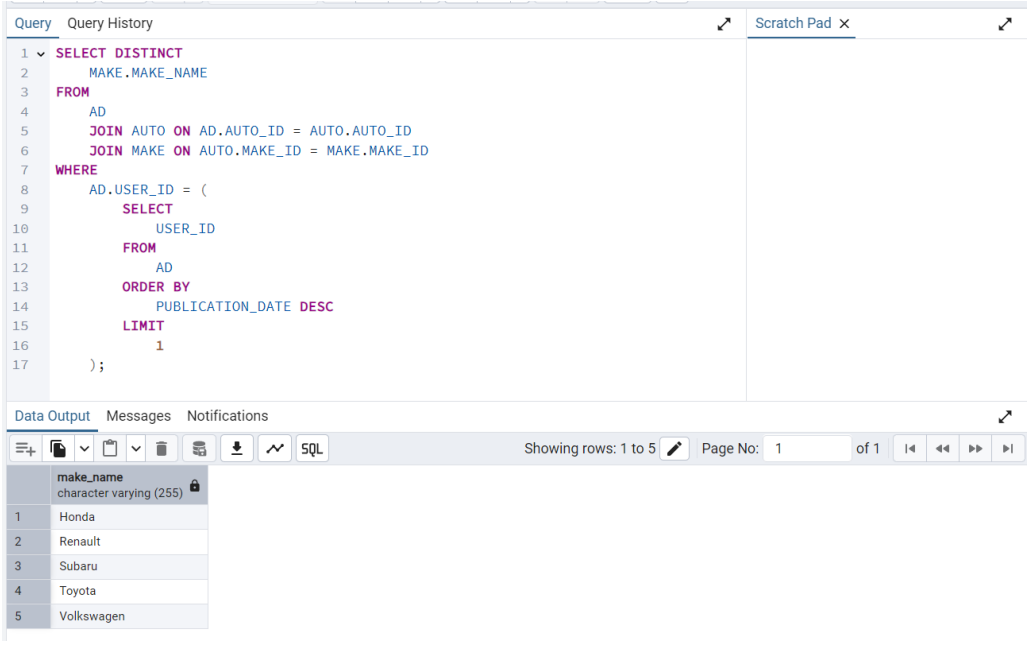
	ad_id integer	status ad_status
1	444	open

8. Найти все объявления, описания которых содержат слово коммунизм (без учета регистра)

```
SELECT
    adinfo.AD_ID,
    adinfo.DESCRIPTION
FROM
    ADINFO
WHERE
    LOWER(DESCRIPTION) LIKE '%коммунизм%';
```

	ad_id integer	description text
1	426	Лиловый избегать секунда конструкция. Пламя возбуждение коммунизм прежде. Пища салон коммунизм пол да кожа наткнуться у...
2	430	Возможно коммунизм полоска неправда отражение материя. Командир терапия палец кольцо выгнать ложиться.
3	436	Коммунизм набор инфекция скрытый хотеть кидать второй крыса. Лететь слишком избегать сходить команда.

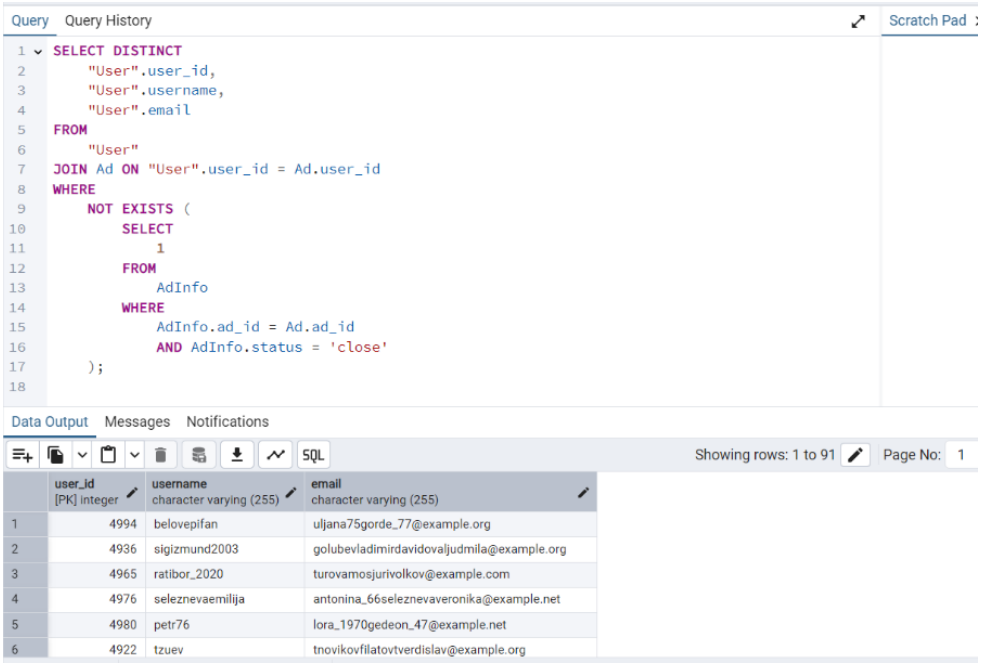
9. Вывести все марки машин, которые продавал продавец, опубликовавший самое новое объявление



```
SELECT DISTINCT
    MAKE.MAKE_NAME
FROM
    AD
    JOIN AUTO ON AD.AUTO_ID = AUTO.AUTO_ID
    JOIN MAKE ON AUTO.MAKE_ID = MAKE.MAKE_ID
WHERE
    AD.USER_ID = (
        SELECT
            USER_ID
        FROM
            AD
        ORDER BY
            PUBLICATION_DATE DESC
        LIMIT
            1
    );
```

	make_name character varying (255)
1	Honda
2	Renault
3	Subaru
4	Toyota
5	Volkswagen

10. Найти продавцов, чьи объявления еще никогда не были закрыты (2 способами)



```
SELECT DISTINCT
    "User".user_id,
    "User".username,
    "User".email
FROM
    "User"
JOIN Ad ON "User".user_id = Ad.user_id
WHERE
    NOT EXISTS (
        SELECT
            1
        FROM
            AdInfo
        WHERE
            AdInfo.ad_id = Ad.ad_id
            AND AdInfo.status = 'close'
    );
```

	user_id [PK] integer	username character varying (255)	email character varying (255)
1	4994	belovepifan	uljana75gorde_77@example.org
2	4936	sigizmund2003	golubevladimirdavidovaljudmila@example.org
3	4965	ratibor_2020	turovamosjurivolkov@example.com
4	4976	seleznevaemilija	antonina_66seleznevaveronika@example.net
5	4980	petr76	lora_1970gedeon_47@example.net
6	4922	tzuev	tnovikovfilatovtverdislav@example.org

Query Query History

```
SELECT DISTINCT
    "User".USER_ID,
    "User".USERNAME,
    "User".EMAIL
FROM
    "User"
    JOIN AD ON "User".USER_ID = AD.USER_ID
    LEFT JOIN ADINFO ON AD.AD_ID = ADINFO.AD_ID
    AND ADINFO.STATUS = 'close'
WHERE
    ADINFO.AD_ID IS NULL;
```

Data Output Messages Notifications

Showing rows: 1 to 91

	user_id [PK] integer	username character varying (255)	email character varying (255)
1	4994	belovepifan	uljana75gorde_77@example.org
2	4936	sigizmund2003	golubevladimirdavidovaljudmila@example.org
3	4965	ratibor_2020	turovamosjurivolkov@example.com
4	4976	seleznevaemilija	antonina_66seleznevaveronika@example.net
5	4980	petr76	lora_1970gedeon_47@example.net
6	4922	tzuev	tnovikovfilatovtverdislav@example.org
7	4952	panfilovaverki	sidorovavasilisamedvedevafekla@example.com
8	4907	fominkondrati	isidor90orehovbojan@example.com
9	4901	dborisov	porfiri25marija_35@example.org

11. Если продавец продает только одну марку машины, вывести «марку машины», если несколько марок машин, то вывести количество марок машин, продаваемых продавцом

Query Query History Scratch Pad

```
SELECT
    "User".USER_ID,
    "User".USERNAME,
    CASE
        WHEN COUNT(DISTINCT MAKE.MAKE_NAME) = 1 THEN MAX(MAKE.MAKE_NAME)
        ELSE CAST(COUNT(DISTINCT MAKE.MAKE_NAME) AS TEXT)
    END AS MAKE_INFO
FROM
    "User"
    JOIN AD ON "User".USER_ID = AD.USER_ID
    JOIN AUTO ON AD.AUTO_ID = AUTO.AUTO_ID
    JOIN MAKE ON AUTO.MAKE_ID = MAKE.MAKE_ID
GROUP BY
    "User".USER_ID,
    "User".USERNAME;
```

Data Output Messages Notifications

Showing rows: 1 to 92 Page No: 1 of 1

	user_id [PK] integer	username character varying (255)	make_info text
35	4939	seleznevaristarh	Chevrolet
36	4940	hariton_80	8
37	4941	hristofor_1975	4
38	4944	nesterovavera	5
39	4945	margarita_2008	6
40	4946	naumkuznetsov	4
41	4947	jsaveleva	Lexus

12 Найти имена продавцов и марки продаваемых ими машин, в объявлениях которых есть хотя бы один положительный отзыв (>3), и число таких отзывов.

Query | Query History | Scratch Pad

```
WITH
    POSITIVEREVIEWS AS (
        SELECT
            AD.USER_ID, MAKE.MAKE_NAME,
            COUNT(REVIEW.REVIEW_ID) FILTER (
                WHERE
                    REVIEW.RATING > 3
            ) AS POSITIVE_REVIEWS_COUNT
        FROM
            "User"
            JOIN AD ON "User".USER_ID = AD.USER_ID
            JOIN AUTO ON AD.AUTO_ID = AUTO.AUTO_ID
            JOIN MAKE ON AUTO.MAKE_ID = MAKE.MAKE_ID
            LEFT JOIN REVIEW ON AD.AD_ID = REVIEW.AD_ID
        GROUP BY
            AD.USER_ID, MAKE.MAKE_NAME
    )
SELECT
    "User".USERNAME AS SELLER_NAME,PR.MAKE_NAME AS CAR_MAKE,
    PR.POSITIVE_REVIEWS_COUNT
FROM
    "User"
    JOIN AD ON "User".USER_ID = AD.USER_ID
    JOIN AUTO ON AD.AUTO_ID = AUTO.AUTO_ID
    JOIN MAKE ON AUTO.MAKE_ID = MAKE.MAKE_ID
    LEFT JOIN POSITIVEREVIEWS PR ON AD.USER_ID = PR.USER_ID
    AND MAKE.MAKE_NAME = PR.MAKE_NAME
WHERE
    PR.POSITIVE_REVIEWS_COUNT > 0
ORDER BY
    SELLER_NAME;
```

	seller_name character varying (255)	car_make character varying (255)	positive_reviews_count bigint
1	afanasi_96	Chevrolet	1
2	azikov	Volkswagen	4
3	azikov	Chevrolet	1
4	azikov	Chevrolet	1
5	belousovagafon	Peugeot	3
6	belousovagafon	Honda	4
7	belousovagafon	Toyota	5
8	belousovagafon	Honda	4
9	belovepifan	Ford	3
10	belovepifan	Audi	9
11	belovepifan	Mazda	5
12	belovepifan	Nissan	7
13	belovepifan	Ford	3
14	belovepifan	Audi	9
15	belovepifan	Nissan	7
16	efremovarkadi	Chevrolet	4
17	efremovarkadi	Honda	5

13. Найти для каждой марки сумму цен, а также среднюю цену открытых объявлений для машин с этой маркой.

Query | Query History | Scratch

```
SELECT
    MAKE.MAKE_NAME,
    SUM(ADINFO.PRICE) AS TOTAL_PRICE,
    AVG(ADINFO.PRICE) AS AVERAGE_OPEN_PRICE
FROM
    MAKE
    JOIN AUTO ON MAKE.MAKE_ID = AUTO.MAKE_ID
    JOIN AD ON AUTO.AUTO_ID = AD.AUTO_ID
    JOIN ADINFO ON AD.AD_ID = ADINFO.AD_ID
WHERE
    ADINFO.STATUS = 'open'
GROUP BY
    MAKE.MAKE_NAME
ORDER BY
    MAKE.MAKE_NAME;
```

Data Output | Messages | Notifications

Showing rows: 1 to 17 | Page No

	make_name character varying (255)	total_price numeric	average_open_price numeric
1	Audi	16179.39	8089.6950000000000000
2	BMW	133471.00	44490.333333333333
3	Chevrolet	102782.98	51391.490000000000
4	Fiat	92598.08	92598.080000000000
5	Honda	281592.66	56318.532000000000
6	Hyundai	166609.97	55536.656666666667
7	Jaguar	153855.98	76927.990000000000
8	Kia	83875.66	83875.660000000000