A Project Report

Used Car Database Management System

Group 2

Ajinkya Sanjay Kobal

Deekshitha Merlyn Paul

Prasanth Dadala

Sahitya Batchu

Sangram Dedge

BAN5501-02-S24_Data Management & SQL for Analytics

Prof. Theyab Alhwiti

Date: - 3rd May 2024

ABSTRACT

The automobile sector has been greatly impacted by the rapidly expanding e-commerce scene, as evidenced by the rise in online used car purchases. This trend makes navigating the intricacies of the used automobile market imperative for a user-centric and efficient database system. This report explores the conception and execution of a painstakingly created relational database system meant especially for this kind of platform. The project carefully follows accepted guidelines for database design, giving data integrity, normalization, and scalability first priority. This guarantees data correctness, reduces duplication, and permits smooth market expansion in the future. Key features include user management, maintaining complete vehicle specifications, processing user bids on listed autos, and enabling the creation of elaborate car adverts.

The project recognizes the potential for future improvements even if the primary system concentrates on current features. These include adding features like user reviews and ratings to improve the user experience, automating post-sale procedures like title transfers and car history reports, and integrating a secure payment processing system. This project prepares the way for an online used automobile marketplace platform that is user-friendly by building the groundwork for a reliable and scalable database system. This results in improved customer experience, more efficient transactions, and eventually a flourishing online automobile market.

INTRODUCTION

Consumer behavior has changed dramatically in the digital age, with a movement toward online shopping occurring in many different industries. This trend has become well-established in the automobile industry, where there is an exponential increase in demand for old cars that are bought online. An efficient database system is essential for handling the intricate web of user interactions, transactions, and vehicle ads in a dynamic used automobile marketplace to adapt to this changing terrain.

But it's no longer sufficient to just build a database system that works. The underlying database must be able to support this increase without sacrificing user experience or speed as the online used car market expands and draws in more users. Our research is focused on this crucial area, which is optimizing a relational database system for performance and scalability in the used automobile market.

This project attempts to fulfill this exact demand and is being carried out in combination with the Data Management & SQL for Analysis Course. We'll develop a dependable relational database system especially for a used automobile marketplace by utilizing modern database design concepts and thoroughly researching scalability and performance optimization strategies. In addition to addressing the difficulties in handling the intricate relationships that arise between buyers, sellers, and automobile listings, this system will make sure that it can successfully adjust and flourish even in the face of the market's rapid expansion.

We shall base our investigation on the following research question:

What are the best ways to achieve optimal scalability and performance in a used vehicle marketplace with growing numbers of car listings, users, and transaction volume through relational database schema and system architecture?

Through the investigation of this research question, the project will produce a database system that is future-proof and supports a vibrant used automobile industry. We will explore different database technologies, design patterns, and performance optimization strategies to make sure the system can handle an increasing number of users with ease, keep query processing times low, and ultimately promote an easy-to-use online vehicle marketplace.

[Dataset: - LINK]

PROJECT OBJECTIVES

This project aims to design and implement a robust relational database system specifically tailored for a used car marketplace. However, to ensure its success in a dynamic online environment, we will prioritize scalability and performance alongside core functionalities. Here's a breakdown of the detailed project objectives:

1. Simulating a Scalable Relational Database System:

Design tables and relationships to effectively handle user data, such as registration details, user kinds (sellers/buyers), and possibly location data. This is known as data modeling for user management.

2. Enabling User Flexibility:

Allow users to create and maintain a single account that makes it easier for them to buy and sell items on the marketplace: unified user accounts.

3. Facilitating Car Advertisement Creation:

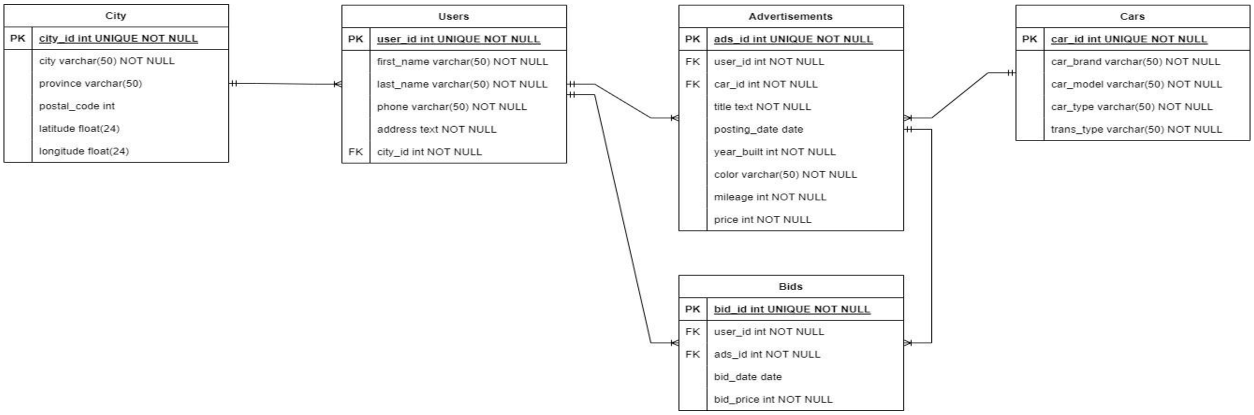
Enable sellers to generate comprehensive automobile adverts by providing them with a system that captures vital details including the car's make, model, year, mileage, and condition. Allowing sellers to set a desired selling price for each automobile they offer is known as selling price management.

4. Additional Considerations for Scalability and Performance:

Future-Proof Design: Provide flexibility in the database schema to allow for future improvements, like the addition of new features or car attributes (like color and features) or the inclusion of user reviews and ratings.

Data Normalization: Use normalization strategies to reduce redundant data and enhance data quality. As the amount of data increases, this will improve the effectiveness of data manipulation and querying.

Figure 1,
ER Diagram,


City
PK city_id int UNIQUE NOT NULL
city varchar(50) NOT NULL
province varchar(50)
postal_code int
latitude float(24)
longitude float(24)
Users
PK user_id int UNIQUE NOT NULL
first_name varchar(50) NOT NULL
last_name varchar(50) NOT NULL
phone varchar(50) NOT NULL
address text NOT NULL
FK city_id int NOT NULL
Advertisements
PK ads_id int UNIQUE NOT NULL
FK user_id int NOT NULL
FK car_id int NOT NULL
title text NOT NULL
posting_date date
year_built int NOT NULL
color varchar(50) NOT NULL
mileage int NOT NULL
price int NOT NULL
Cars
PK car_id int UNIQUE NOT NULL
car_brand varchar(50) NOT NULL
car_model varchar(50) NOT NULL
car_type varchar(50) NOT NULL
trans_type varchar(50) NOT NULL
Bids
PK bid_id int UNIQUE NOT NULL
FK user_id int NOT NULL
FK ads_id int NOT NULL
bid_date date
bid_price int NOT NULL

(Fig 1. ER Diagram - Used Cars)

Figure 2,

Data Dictionary,

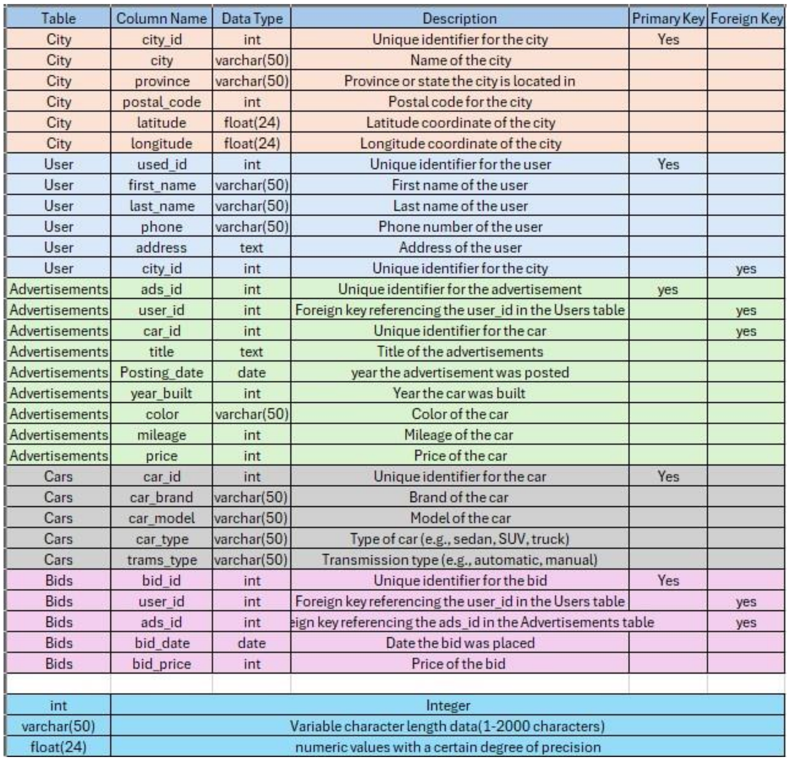


Table	Column Name	Data Type	Description	Primary Key	Foreign Key
City	city_id	int	Unique identifier for the city	Yes	
City	city	varchar(50)	Name of the city		
City	province	varchar(50)	Province or state the city is located in		
City	postal_code	int	Postal code for the city		
City	latitude	float(24)	Latitude coordinate of the city		
City	longitude	float(24)	Longitude coordinate of the city		
User	used_id	int	Unique identifier for the user	Yes	
User	first_name	varchar(50)	First name of the user		
User	last_name	varchar(50)	Last name of the user		
User	phone	varchar(50)	Phone number of the user		
User	address	text	Address of the user		
User	city_id	int	Unique identifier for the city		yes
Advertisements	ads_id	int	Unique identifier for the advertisement	yes	
Advertisements	user_id	int	Foreign key referencing the user_id in the Users table		yes
Advertisements	car_id	int	Unique identifier for the car		yes
Advertisements	title	text	Title of the advertisements		
Advertisements	Posting_date	date	year the advertisement was posted		
Advertisements	year_built	int	Year the car was built		
Advertisements	color	varchar(50)	Color of the car		
Advertisements	mileage	int	Mileage of the car		
Advertisements	price	int	Price of the car		
Cars	car_id	int	Unique identifier for the car	Yes	
Cars	car_brand	varchar(50)	Brand of the car		
Cars	car_model	varchar(50)	Model of the car		
Cars	car_type	varchar(50)	Type of car (e.g., sedan, SUV, truck)		
Cars	trams_type	varchar(50)	Transmission type (e.g., automatic, manual)		
Bids	bid_id	int	Unique identifier for the bid	Yes	
Bids	user_id	int	Foreign key referencing the user_id in the Users table		yes
Bids	ads_id	int	eign key referencing the ads_id in the Advertisements table		yes
Bids	bid_date	date	Date the bid was placed		
Bids	bid_price	int	Price of the bid		
int	Integer				
varchar(50)	Variable character length data(1-2000 characters)				
float(24)	numeric values with a certain degree of precision				

(Fig 2. Data Dictionary - Used Cars)

We can build a database system with a solid basis for user interaction, essential marketplace features, and the capacity to accommodate future expansion by concentrating on these specific goals. This will create the foundation for a successful internet marketplace for secondhand cars.

PROJECT SCOPE

This project focuses on designing and implementing a core relational database system specifically designed for a used car marketplace. The primary objective is to establish a foundation for managing essential data elements that facilitate user interaction, car listings, and the bidding process. Here's a breakdown of the project scope:

User Management System:

- Create and implement a user table to hold the data associated with user registration, such as email addresses, passwords (securely hashed), usernames, and possibly optional information like phone numbers.
- Create a field to indicate if the user is a vendor or a buyer in order to activate role-based features in the marketplace.
- For possible logistics management (outside the purview of this project), take into consideration adding a location table that is related to the user table and enables users to indicate where they are.

Car Advertisement Management:

- Create a specific table to record information from auto advertisements, such as the vehicle's make, model, year, mileage, and condition.
- Provide a mechanism that allows sellers to list a price for each vehicle they promote.

- To connect ads to sellers, create links between the automobile advertisement table and other pertinent tables (such as the user table).

Bidding System:

- Create a table to record bid details, such as the date of the bidding and the offered price.
- To track user bids on automobile listings, create links between the bid table and the user table and the car advertisement table.

Database Design Principles:

- Follow accepted data integrity guidelines by using foreign keys to connect similar data across tables and primary keys to identify records uniquely.
- Utilize data normalization strategies to reduce duplication and boost data management effectiveness.
- By including a flexible schema design that can handle future expansion in the number of users, automobile listings, and transaction volume, you can take future scalability into consideration**.**

Transaction Processing: Managing financial transactions after a bid is accepted falls outside the scope of this project. Integration with a secure payment processing system would be a future enhancement.

Post-Sale Activities: Management of activities following a sale, such as title transfer or vehicle history reports, is not included in this project's scope.

Advanced Functionalities: Features like user reviews and ratings, while valuable for a comprehensive marketplace, are beyond the scope of this project but can be considered for future development.

The project will deliver a fundamental database system that drives essential marketplace capabilities by concentrating on certain in-scope activities. Future scalability to support expansion and possible improvements for a successful online platform for used car marketplaces will be considered in the design.

PROJECT DESCRIPTION

This project delves into the design and implementation of a relational database system tailored specifically for a used car marketplace. The system aims to efficiently manage the complexities of this online platform, catering to both buyers and sellers.

User Flexibility and Interaction:

- Users will have the ability to sign up for the marketplace as consumers or sellers, giving them access to a variety of activities.
- Advertisements for the cars they wish to sell can be made by sellers. These commercials will include pertinent vehicle information, enabling prospective customers to make well-informed decisions.
- Customers have the option to peruse current automobile ads and place bids on vehicles that catch their eye.

Data Integrity and Scalability:

- Data integrity will be given top priority in the database schema by following accepted guidelines. This entails using foreign keys to create associations between pertinent tables (users, ads, automobiles, and bids) and primary keys to identify records uniquely.

- As the user base and automobile listings expand, data normalization techniques will be used to reduce data redundancy and improve data management effectiveness.
- Future scalability will be considered in the design, along with flexibility to allow for possible future improvements and a rise in the volume of data in the marketplace.

Optimizing Performance and User Experience:

- To optimize query performance, great care will be used when creating the database schema and queries. This means that consumers looking through vehicle listings or managing their accounts will get speedier search results and more effective data retrieval.
- It is essential to establish robust associations between different elements in the database schema. This will enable smooth communication inside the marketplace platform between users, ads, vehicle information, and the bidding procedure.

The overall goal of this project is to provide a solid database foundation that supports an easy-to-use and effective online used automobile marketplace for both buyers and sellers. The system is intended to manage the intricacies of the commercial landscape while guaranteeing scalability to support further expansion.

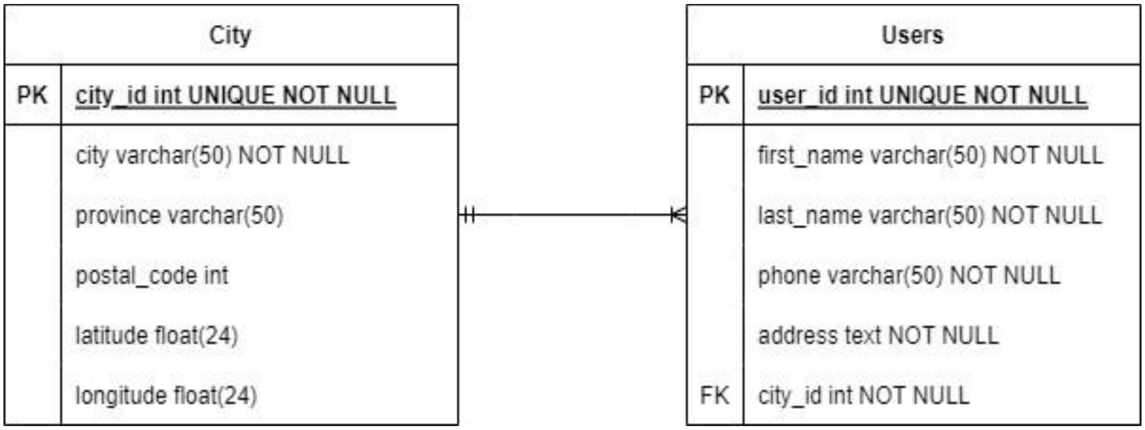
PROJECT COMPONENTS

➢ Designing Database

Every table needs to have a distinct **Primary Key (PK)** for every entry to guarantee data integrity. A table's individual records can be recognized and distinguished using the primary key. Furthermore, **Foreign Keys (FK)**, which create relationships between tables, can be used to combine the values from fields in other tables. Relationships between related data in different database tables can be linked thanks to foreign keys. The **NOT NULL** constraint will be used to indicate essential fields.

➢ Users & Location

User information is essential to guaranteeing precise identification of buyers and sellers during transactions. Furthermore, the users' locations are crucial in figuring out the logistics of car shipments. Taking these things into account, we suggest the following design:



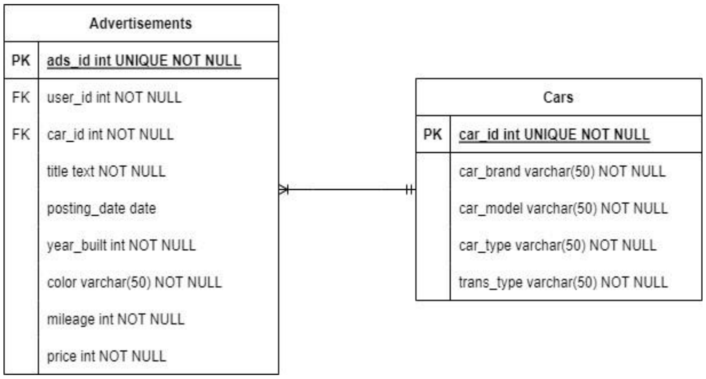
	City
PK	city_id int UNIQUE NOT NULL
	city varchar(50) NOT NULL
	province varchar(50)
	postal_code int
	latitude float(24)
	longitude float(24)

	Users
PK	user_id int UNIQUE NOT NULL
	first_name varchar(50) NOT NULL
	last_name varchar(50) NOT NULL
	phone varchar(50) NOT NULL
	address text NOT NULL
FK	city_id int NOT NULL

A distinct City Table will be made to guarantee data integrity and get rid of redundancies. Additionally, this design will support the potential for many users to live in the same city, creating a **one-to-many relationship.**

➢ Advertisements & Cars

Important details about the vehicles up for grabs should be entered into the Advertisements Table using the following required fields:



	Advertisements
PK	ads_id int UNIQUE NOT NULL
FK	user_id int NOT NULL
FK	car_id int NOT NULL
	title text NOT NULL
	posting_date date
	year_built int NOT NULL
	color varchar(50) NOT NULL
	mileage int NOT NULL
	price int NOT NULL

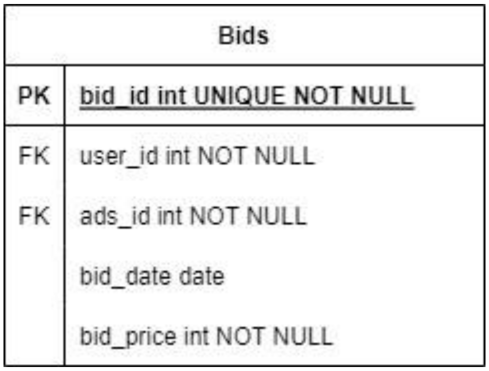
	Cars
PK	car_id int UNIQUE NOT NULL
	car_brand varchar(50) NOT NULL
	car_model varchar(50) NOT NULL
	car_type varchar(50) NOT NULL
	trans_type varchar(50) NOT NULL

The Cars Table is created as a separate entity in order to prevent redundancy. A one-to-many relationship is created with the Advertisements Table since each advertisement should specifically offer information about a single car type, even though a single car type may appear in multiple advertisements.

- **Bids**

The offering price and the bidding date are the two most significant fields in this table.

The remaining data can be retrieved from other tables using foreign keys.



	Bids
PK	bid_id int UNIQUE NOT NULL
FK	user_id int NOT NULL
FK	ads_id int NOT NULL
	bid_date date
	bid_price int NOT NULL

It displays a many-to-one relationship with both the Advertisements Table and the Users Table, indicating that a single advertisement may receive more than one bid, and that a user may submit more than one bid on the same or separate advertisements.

DATABASE ENVIRONMENT

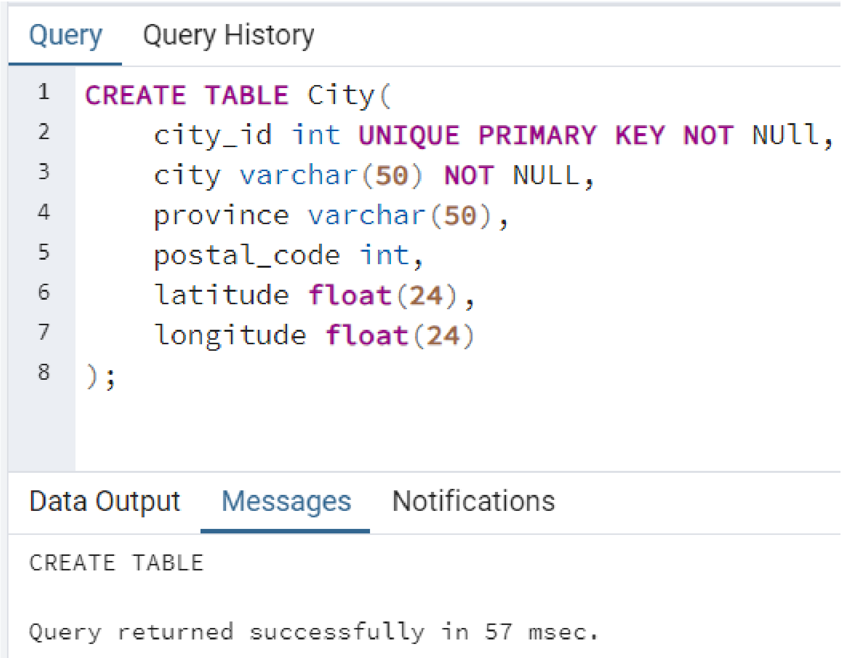
Client Profile: This project designs a database for a **used car marketplace**. It manages user data, car listings (make, model, year, etc.), bids, and facilitates searching & bidding. This helps with:

- User-friendly buying & selling
- Efficient car listing management
- Transparent bidding system
- Data-driven insights for sellers

User Profile: This project builds upon the core functionalities of a used car marketplace database, prioritizing scalability, and performance for a dynamic online environment. Here's a breakdown of the key objectives:

- Scalable Data Model
- Unified User Accounts
- Comprehensive Car Listings
- Future-Proof Design
- Performance Optimization

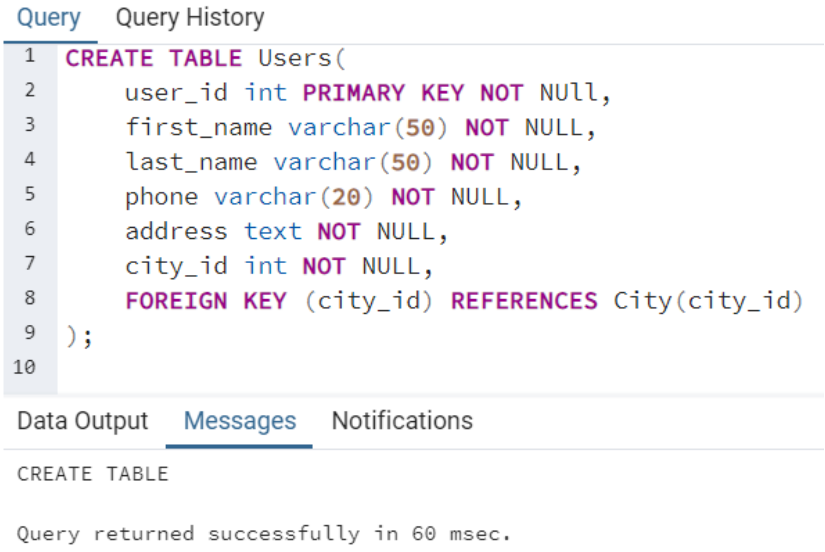
Interface: The interface is easy to use, and the screenshots are attached below.



```
CREATE TABLE City(
    city_id int UNIQUE PRIMARY KEY NOT NUll,
    city varchar(50) NOT NULL,
    province varchar(50),
    postal_code int,
    latitude float(24),
    longitude float(24)
);
```

```
CREATE TABLE

Query returned successfully in 57 msec.
```

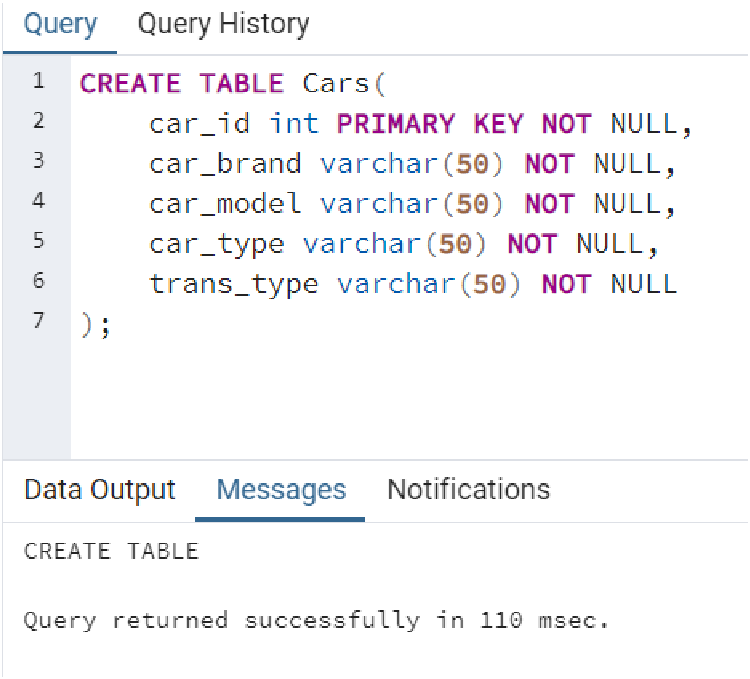
The query is about creating a table named City with various attributes including city_id, city, province, postal code, latitude, and longitude.



```
CREATE TABLE Users(
    user_id int PRIMARY KEY NOT NUll,
    first_name varchar(50) NOT NULL,
    last_name varchar(50) NOT NULL,
    phone varchar(20) NOT NULL,
    address text NOT NULL,
    city_id int NOT NULL,
    FOREIGN KEY (city_id) REFERENCES City(city_id)
);

```

```
CREATE TABLE

Query returned successfully in 60 msec.
```

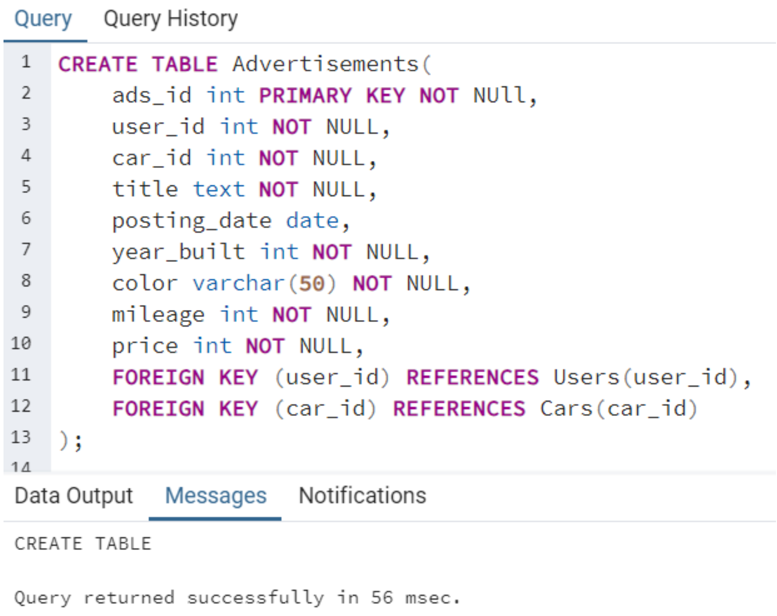
The above query successfully created a table named Users with columns for user ID, username, password, email, phone number, and user type.



```sql
CREATE TABLE Cars(
    car_id int PRIMARY KEY NOT NULL,
    car_brand varchar(50) NOT NULL,
    car_model varchar(50) NOT NULL,
    car_type varchar(50) NOT NULL,
    trans_type varchar(50) NOT NULL
);
```

```
CREATE TABLE

Query returned successfully in 110 msec.
```

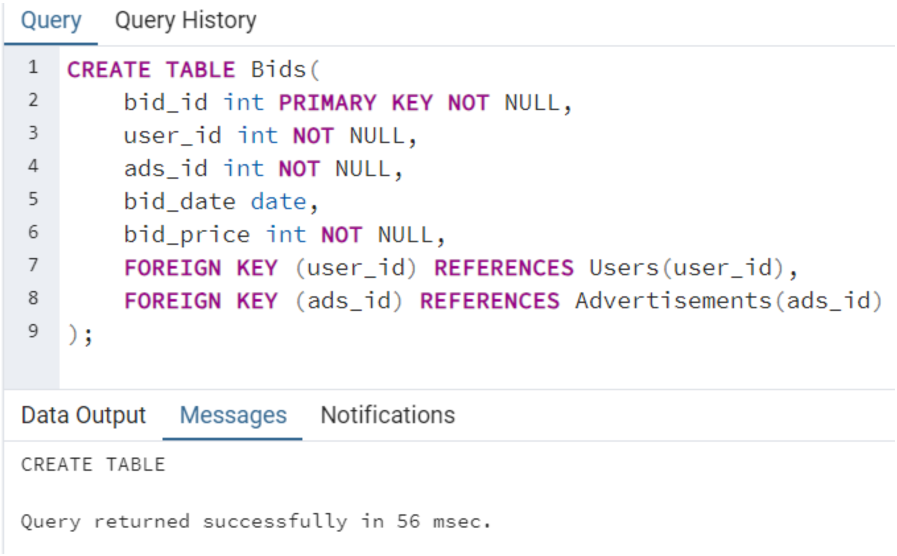
The query successfully created a table named Cars. The Cars table stores details like car ID, make, model, year, and car type.



```sql
CREATE TABLE Advertisements(
    ads_id int PRIMARY KEY NOT NUll,
    user_id int NOT NULL,
    car_id int NOT NULL,
    title text NOT NULL,
    posting_date date,
    year_built int NOT NULL,
    color varchar(50) NOT NULL,
    mileage int NOT NULL,
    price int NOT NULL,
    FOREIGN KEY (user_id) REFERENCES Users(user_id),
    FOREIGN KEY (car_id) REFERENCES Cars(car_id)
);
```

```
CREATE TABLE

Query returned successfully in 56 msec.
```

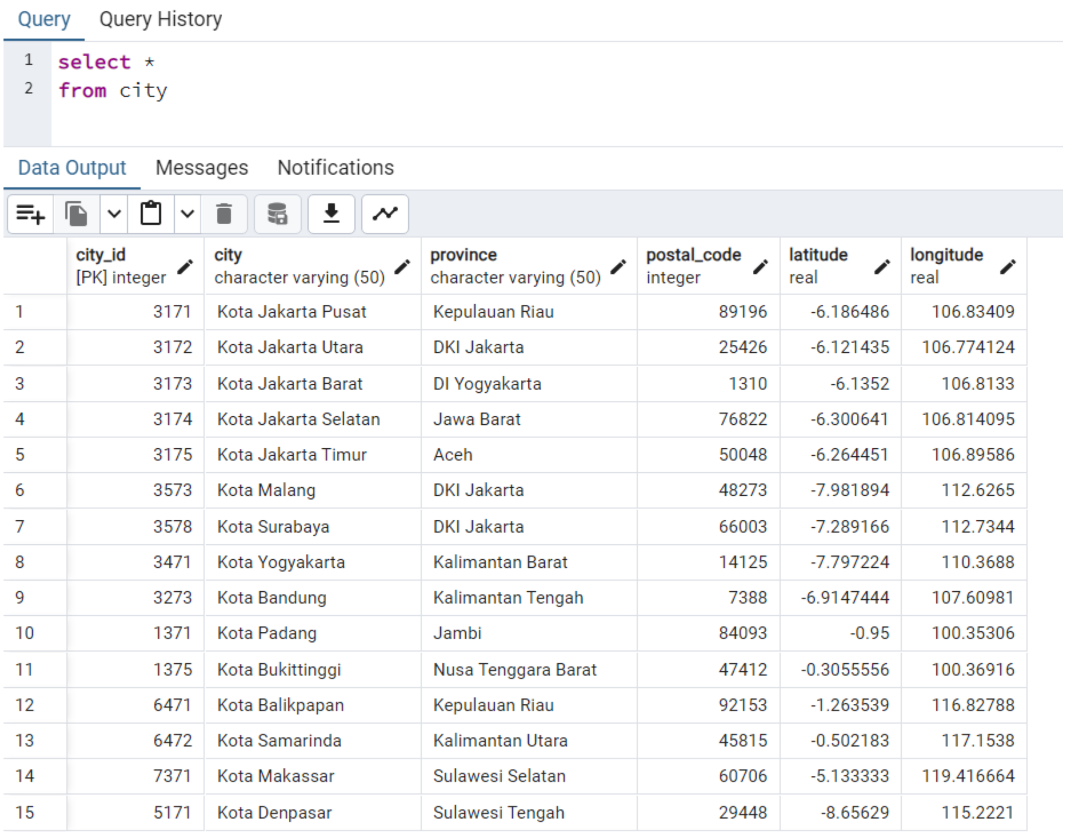
The above relational database schema is for a used car marketplace. It includes tables for users, cars, advertisements, bids, and potential locations. Foreign keys link these tables to establish relationships between data points.



```
CREATE TABLE Bids(
    bid_id int PRIMARY KEY NOT NULL,
    user_id int NOT NULL,
    ads_id int NOT NULL,
    bid_date date,
    bid_price int NOT NULL,
    FOREIGN KEY (user_id) REFERENCES Users(user_id),
    FOREIGN KEY (ads_id) REFERENCES Advertisements(ads_id)
);
```

```
CREATE TABLE

Query returned successfully in 56 msec.
```

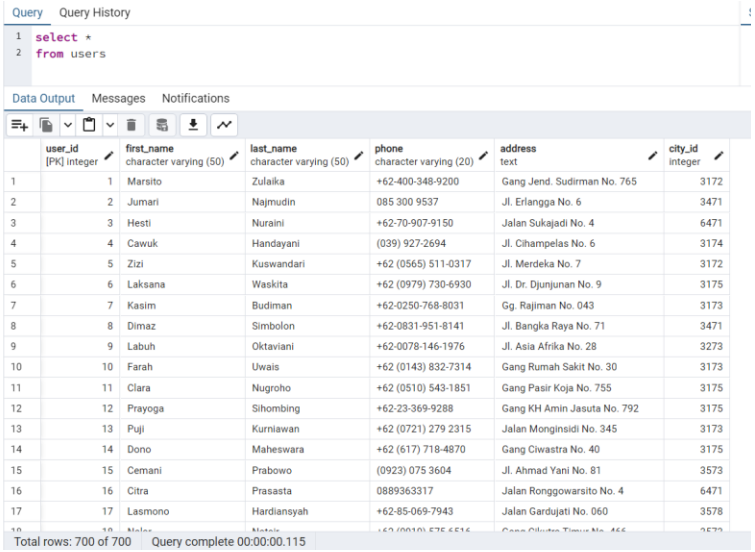
This SQL query updates an existing table named Bids in a relational database. It likely modifies a specific field, such as the bid amount, based on a certain condition.



```
select *
from city
```

	city_id [PK] integer	city character varying (50)	province character varying (50)	postal_code integer	latitude real	longitude real
1	3171	Kota Jakarta Pusat	Kepulauan Riau	89196	-6.186486	106.83409
2	3172	Kota Jakarta Utara	DKI Jakarta	25426	-6.121435	106.774124
3	3173	Kota Jakarta Barat	DI Yogyakarta	1310	-6.1352	106.8133
4	3174	Kota Jakarta Selatan	Jawa Barat	76822	-6.300641	106.814095
5	3175	Kota Jakarta Timur	Aceh	50048	-6.264451	106.89586
6	3573	Kota Malang	DKI Jakarta	48273	-7.981894	112.6265
7	3578	Kota Surabaya	DKI Jakarta	66003	-7.289166	112.7344
8	3471	Kota Yogyakarta	Kalimantan Barat	14125	-7.797224	110.3688
9	3273	Kota Bandung	Kalimantan Tengah	7388	-6.9147444	107.60981
10	1371	Kota Padang	Jambi	84093	-0.95	100.35306
11	1375	Kota Bukittinggi	Nusa Tenggara Barat	47412	-0.3055556	100.36916
12	6471	Kota Balikpapan	Kepulauan Riau	92153	-1.263539	116.82788
13	6472	Kota Samarinda	Kalimantan Utara	45815	-0.502183	117.1538
14	7371	Kota Makassar	Sulawesi Selatan	60706	-5.133333	119.416664
15	5171	Kota Denpasar	Sulawesi Tengah	29448	-8.65629	115.2221

The image shows a screenshot of a relational database schema for a used car marketplace. It includes tables for users, cars, advertisements, and bids, likely with relationships established through foreign keys.

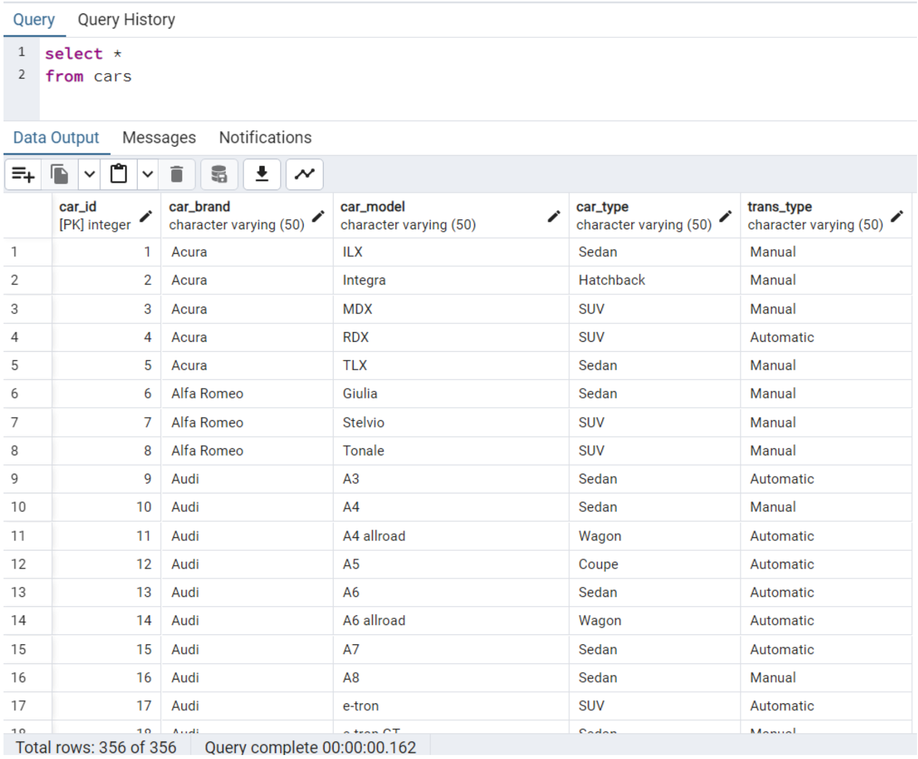


```
select *
from users
```

	user_id [PK] integer	first_name character varying (50)	last_name character varying (50)	phone character varying (20)	address text	city_id integer
1	1	Marsito	Zulaika	+62-400-348-9200	Gang Jend. Sudirman No. 765	3172
2	2	Jumari	Najmudin	085 300 9537	Jl. Erlangga No. 6	3471
3	3	Hesti	Nuraini	+62-70-907-9150	Jalan Sukajadi No. 4	6471
4	4	Cawuk	Handayani	(039) 927-2694	Jl. Cihampelas No. 6	3174
5	5	Zizi	Kuswandari	+62 (0565) 511-0317	Jl. Merdeka No. 7	3172
6	6	Laksana	Waskita	+62 (0979) 730-6930	Jl. Dr. Djunjunan No. 9	3175
7	7	Kasim	Budiman	+62-0250-768-8031	Gg. Rajiman No. 043	3173
8	8	Dimaz	Simbolon	+62-0831-951-8141	Jl. Bangka Raya No. 71	3471
9	9	Labuh	Oktaviani	+62-0078-146-1976	Jl. Asia Afrika No. 28	3273
10	10	Farah	Uwais	+62 (0143) 832-7314	Gang Rumah Sakit No. 30	3173
11	11	Clara	Nugroho	+62 (0510) 543-1851	Gang Pasir Koja No. 755	3175
12	12	Prayoga	Sihombing	+62-23-369-9288	Gang KH Amin Jasuta No. 792	3175
13	13	Puji	Kurniawan	+62 (0721) 279 2315	Jalan Monginsidi No. 345	3173
14	14	Dono	Maheswara	+62 (617) 718-4870	Gang Ciwastra No. 40	3175
15	15	Cemani	Prabowo	(0923) 075 3604	Jl. Ahmad Yani No. 81	3573
16	16	Citra	Prasasta	0889363317	Jalan Ronggowarsito No. 4	6471
17	17	Lasmono	Hardiansyah	+62-85-069-7943	Jalan Gardujati No. 060	3578

Total rows: 700 of 700 Query complete 00:00:00.115

The above query retrieves data from a table named users.

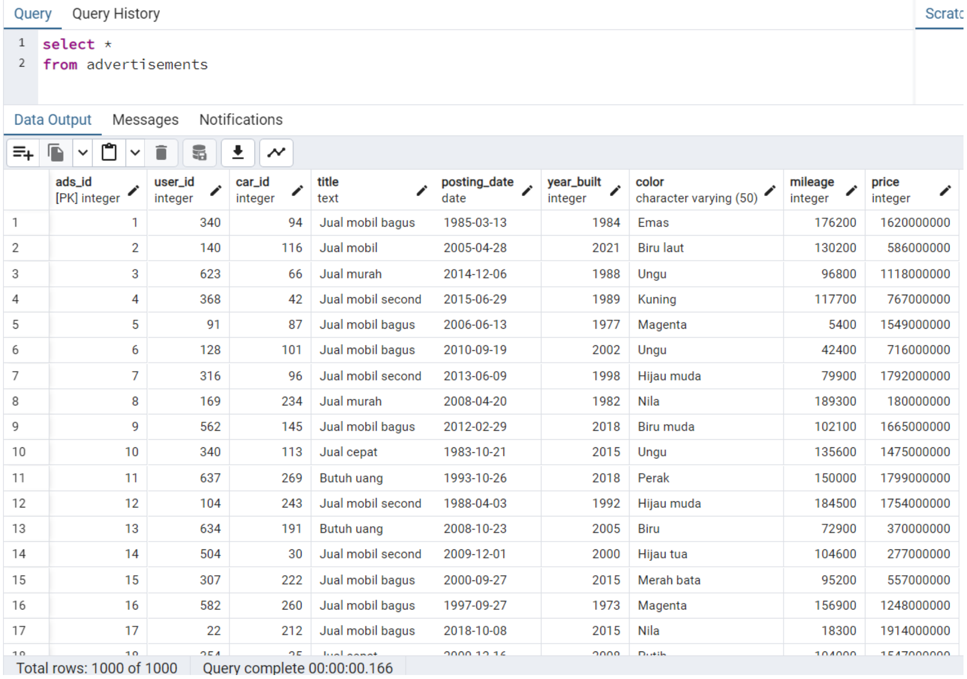


```
select *
from cars
```

	car_id [PK] integer	car_brand character varying (50)	car_model character varying (50)	car_type character varying (50)	trans_type character varying (50)
1	1	Acura	ILX	Sedan	Manual
2	2	Acura	Integra	Hatchback	Manual
3	3	Acura	MDX	SUV	Manual
4	4	Acura	RDX	SUV	Automatic
5	5	Acura	TLX	Sedan	Manual
6	6	Alfa Romeo	Giulia	Sedan	Manual
7	7	Alfa Romeo	Stelvio	SUV	Manual
8	8	Alfa Romeo	Tonale	SUV	Manual
9	9	Audi	A3	Sedan	Automatic
10	10	Audi	A4	Sedan	Manual
11	11	Audi	A4 allroad	Wagon	Automatic
12	12	Audi	A5	Coupe	Automatic
13	13	Audi	A6	Sedan	Automatic
14	14	Audi	A6 allroad	Wagon	Automatic
15	15	Audi	A7	Sedan	Automatic
16	16	Audi	A8	Sedan	Manual
17	17	Audi	e-tron	SUV	Automatic

Total rows: 356 of 356 Query complete 00:00:00.162

The above query retrieves data from a table named Cars.

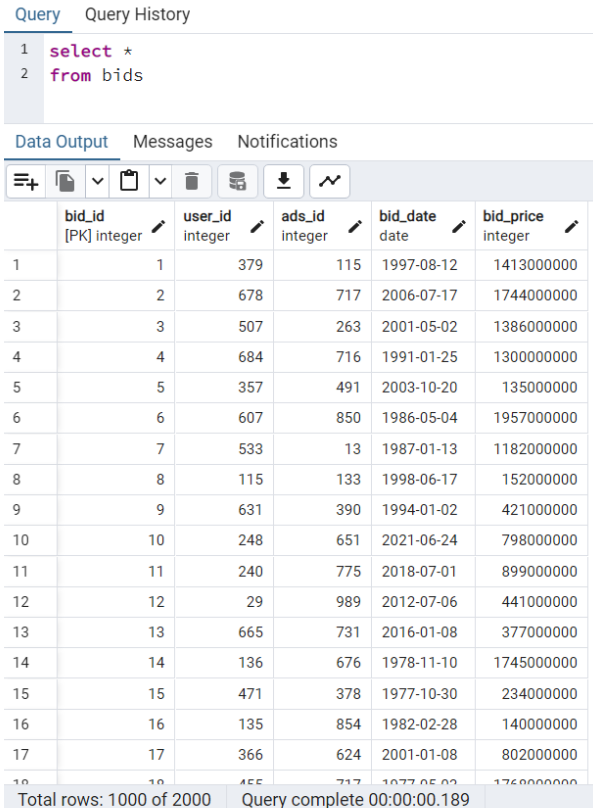


```sql
select *
from advertisements
```

	ads_id [PK] integer	user_id integer	car_id integer	title text	posting_date date	year_built integer	color character varying (50)	mileage integer	price integer
1	1	340	94	Jual mobil bagus	1985-03-13	1984	Emas	176200	1620000000
2	2	140	116	Jual mobil	2005-04-28	2021	Biru laut	130200	586000000
3	3	623	66	Jual murah	2014-12-06	1988	Ungu	96800	1118000000
4	4	368	42	Jual mobil second	2015-06-29	1989	Kuning	117700	767000000
5	5	91	87	Jual mobil bagus	2006-06-13	1977	Magenta	5400	1549000000
6	6	128	101	Jual mobil bagus	2010-09-19	2002	Ungu	42400	716000000
7	7	316	96	Jual mobil second	2013-06-09	1998	Hijau muda	79900	1792000000
8	8	169	234	Jual murah	2008-04-20	1982	Nila	189300	180000000
9	9	562	145	Jual mobil bagus	2012-02-29	2018	Biru muda	102100	1665000000
10	10	340	113	Jual cepat	1983-10-21	2015	Ungu	135600	1475000000
11	11	637	269	Butuh uang	1993-10-26	2018	Perak	150000	1799000000
12	12	104	243	Jual mobil second	1988-04-03	1992	Hijau muda	184500	1754000000
13	13	634	191	Butuh uang	2008-10-23	2005	Biru	72900	370000000
14	14	504	30	Jual mobil second	2009-12-01	2000	Hijau tua	104600	277000000
15	15	307	222	Jual mobil bagus	2000-09-27	2015	Merah bata	95200	557000000
16	16	582	260	Jual mobil bagus	1997-09-27	1973	Magenta	156900	1248000000
17	17	22	212	Jual mobil bagus	2018-10-08	2015	Nila	18300	1914000000
18	18	354	35	Jual cepat	2000-12-16	2008	Putih	104000	1547000000

Total rows: 1000 of 1000 Query complete 00:00:00.166

The above query retrieves data from a table named Advertisements.

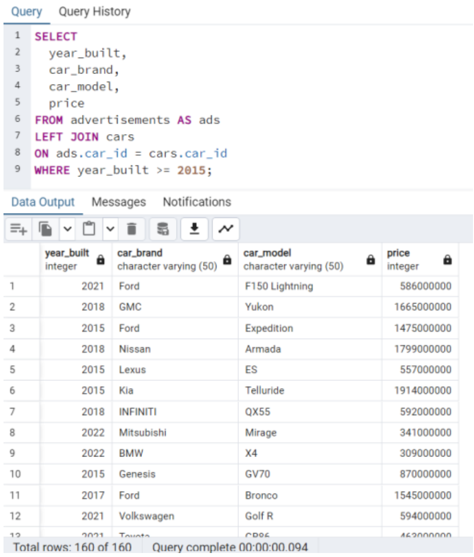


```sql
select *
from bids
```

	bid_id [PK] integer	user_id integer	ads_id integer	bid_date date	bid_price integer
1	1	379	115	1997-08-12	1413000000
2	2	678	717	2006-07-17	1744000000
3	3	507	263	2001-05-02	1386000000
4	4	684	716	1991-01-25	1300000000
5	5	357	491	2003-10-20	135000000
6	6	607	850	1986-05-04	1957000000
7	7	533	13	1987-01-13	1182000000
8	8	115	133	1998-06-17	152000000
9	9	631	390	1994-01-02	421000000
10	10	248	651	2021-06-24	798000000
11	11	240	775	2018-07-01	899000000
12	12	29	989	2012-07-06	441000000
13	13	665	731	2016-01-08	377000000
14	14	136	676	1978-11-10	1745000000
15	15	471	378	1977-10-30	234000000
16	16	135	854	1982-02-28	140000000
17	17	366	624	2001-01-08	802000000
18	18	455	717	1977-05-02	1768000000

Total rows: 1000 of 2000 Query complete 00:00:00.189

The above query retrieves data from a table named bids.

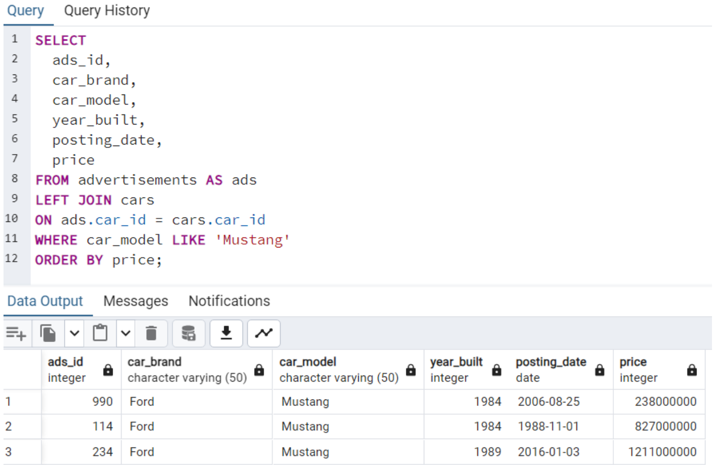


```sql
SELECT
  year_built,
  car_brand,
  car_model,
  price
FROM advertisements AS ads
LEFT JOIN cars
ON ads.car_id = cars.car_id
WHERE year_built >= 2015;
```

	year_built integer	car_brand character varying (50)	car_model character varying (50)	price integer
1	2021	Ford	F150 Lightning	586000000
2	2018	GMC	Yukon	1665000000
3	2015	Ford	Expedition	1475000000
4	2018	Nissan	Armada	1799000000
5	2015	Lexus	ES	557000000
6	2015	Kia	Telluride	1914000000
7	2018	INFINITI	QX55	592000000
8	2022	Mitsubishi	Mirage	341000000
9	2022	BMW	X4	309000000
10	2015	Genesis	GV70	870000000
11	2017	Ford	Bronco	1545000000
12	2021	Volkswagen	Golf R	594000000
13	2021	Toyota	GR86	463000000

Total rows: 160 of 160 Query complete 00:00:00.094

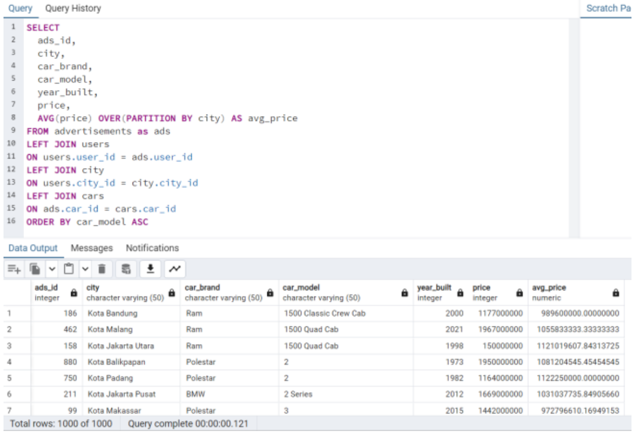
The above query retrieves data from two tables. It joins the advertisements and cars tables, potentially to find all car advertisements based on specific criteria.



```sql
SELECT
  ads_id,
  car_brand,
  car_model,
  year_built,
  posting_date,
  price
FROM advertisements AS ads
LEFT JOIN cars
ON ads.car_id = cars.car_id
WHERE car_model LIKE 'Mustang'
ORDER BY price;
```

	ads_id integer	car_brand character varying (50)	car_model character varying (50)	year_built integer	posting_date date	price integer
1	990	Ford	Mustang	1984	2006-08-25	238000000
2	114	Ford	Mustang	1984	1988-11-01	827000000
3	234	Ford	Mustang	1989	2016-01-03	1211000000

The image depicts the query joining two tables in a relational database. It combines data from the “cars” and “advertisements” tables, likely to retrieve car listings with specific details like car model or price.

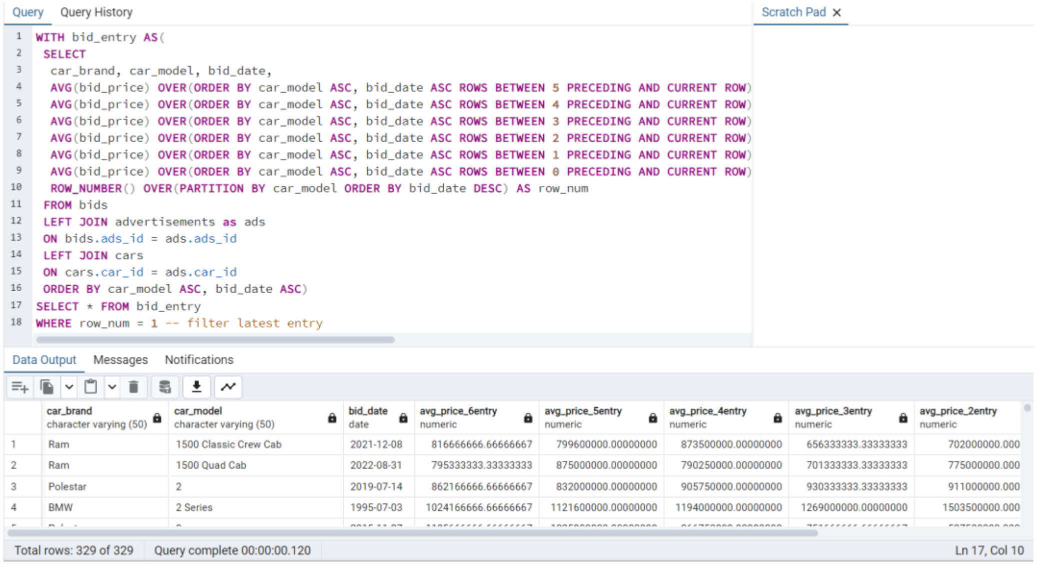


```sql
SELECT
  ads_id,
  city,
  car_brand,
  car_model,
  year_built,
  price,
  AVG(price) OVER(PARTITION BY city) AS avg_price
FROM advertisements as ads
LEFT JOIN users
ON users.user_id = ads.user_id
LEFT JOIN city
ON users.city_id = city.city_id
LEFT JOIN cars
ON ads.car_id = cars.car_id
ORDER BY car_model ASC
```

	ads_id integer	city character varying (50)	car_brand character varying (50)	car_model character varying (50)	year_built integer	price integer	avg_price numeric
1	186	Kota Bandung	Ram	1500 Classic Crew Cab	2000	1177000000	989600000.00000000
2	462	Kota Malang	Ram	1500 Quad Cab	2021	1967000000	1055833333.33333333
3	158	Kota Jakarta Utara	Ram	1500 Quad Cab	1998	150000000	1121019607.84313725
4	880	Kota Balikpapan	Polestar	2	1973	1950000000	1081204545.45454545
5	750	Kota Padang	Polestar	2	1982	1164000000	1122250000.00000000
6	211	Kota Jakarta Pusat	BMW	2 Series	2012	1669000000	1031037735.84905660
7	99	Kota Makassar	Polestar	3	2015	1442000000	972796610.16949153

Total rows: 1000 of 1000 Query complete 00:00:00.121

The query updates the price of a car advertisement in a database for a used car marketplace.



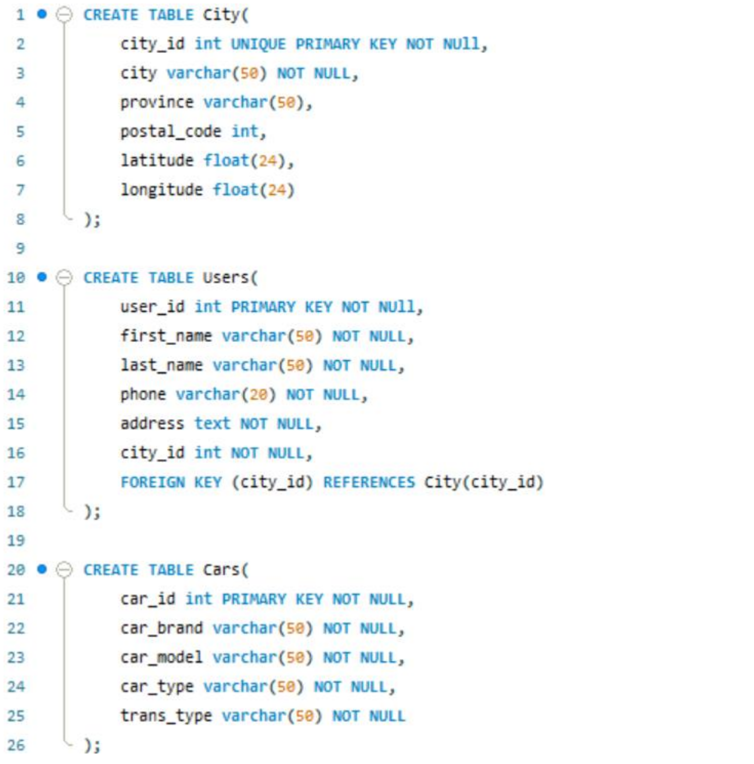
```sql
WITH bid_entry AS(
 SELECT
  car_brand, car_model, bid_date,
  AVG(bid_price) OVER(ORDER BY car_model ASC, bid_date ASC ROWS BETWEEN 5 PRECEDING AND CURRENT ROW)
  AVG(bid_price) OVER(ORDER BY car_model ASC, bid_date ASC ROWS BETWEEN 4 PRECEDING AND CURRENT ROW)
  AVG(bid_price) OVER(ORDER BY car_model ASC, bid_date ASC ROWS BETWEEN 3 PRECEDING AND CURRENT ROW)
  AVG(bid_price) OVER(ORDER BY car_model ASC, bid_date ASC ROWS BETWEEN 2 PRECEDING AND CURRENT ROW)
  AVG(bid_price) OVER(ORDER BY car_model ASC, bid_date ASC ROWS BETWEEN 1 PRECEDING AND CURRENT ROW)
  AVG(bid_price) OVER(ORDER BY car_model ASC, bid_date ASC ROWS BETWEEN 0 PRECEDING AND CURRENT ROW)
  ROW_NUMBER() OVER(PARTITION BY car_model ORDER BY bid_date DESC) AS row_num
 FROM bids
 LEFT JOIN advertisements as ads
 ON bids.ads_id = ads.ads_id
 LEFT JOIN cars
 ON cars.car_id = ads.car_id
 ORDER BY car_model ASC, bid_date ASC)
SELECT * FROM bid_entry
WHERE row_num = 1 -- filter latest entry
```

	car_brand character varying (50)	car_model character varying (50)	bid_date date	avg_price_6entry numeric	avg_price_5entry numeric	avg_price_4entry numeric	avg_price_3entry numeric	avg_price_2entry numeric
1	Ram	1500 Classic Crew Cab	2021-12-08	816666666.66666667	799600000.00000000	873500000.00000000	656333333.33333333	702000000.000
2	Ram	1500 Quad Cab	2022-08-31	795333333.33333333	875000000.00000000	790250000.00000000	701333333.33333333	775000000.000
3	Polestar	2	2019-07-14	862166666.66666667	832000000.00000000	905750000.00000000	930333333.33333333	911000000.000
4	BMW	2 Series	1995-07-03	1024166666.66666667	1121600000.00000000	1194000000.00000000	1269000000.00000000	1503500000.000

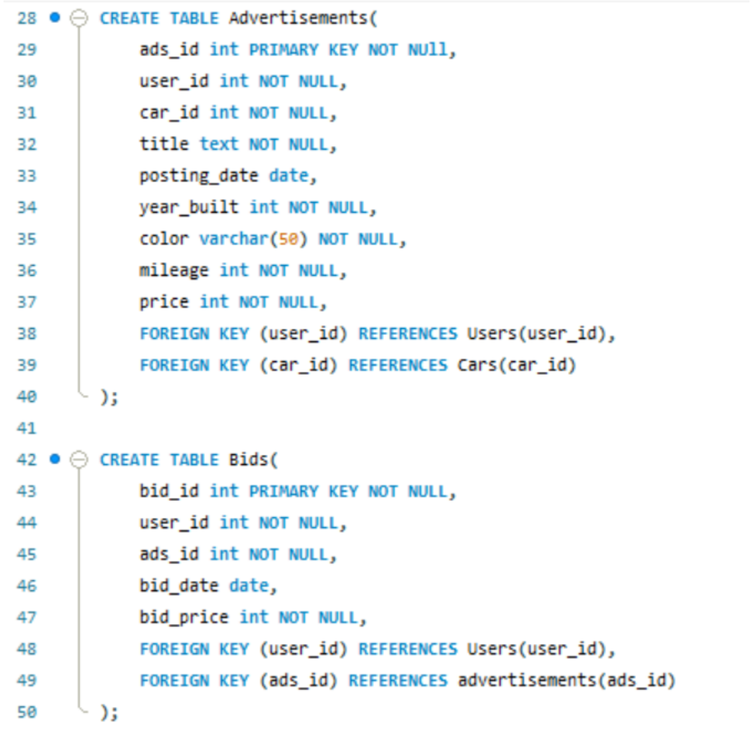
Total rows: 329 of 329 Query complete 00:00:00.120 Ln 17, Col 10

This query is updating the avg_price field (average price) in a table named bid_entry. It calculates the average price for each car model, ordered by ascending bid date, within a window of 5 preceding rows.

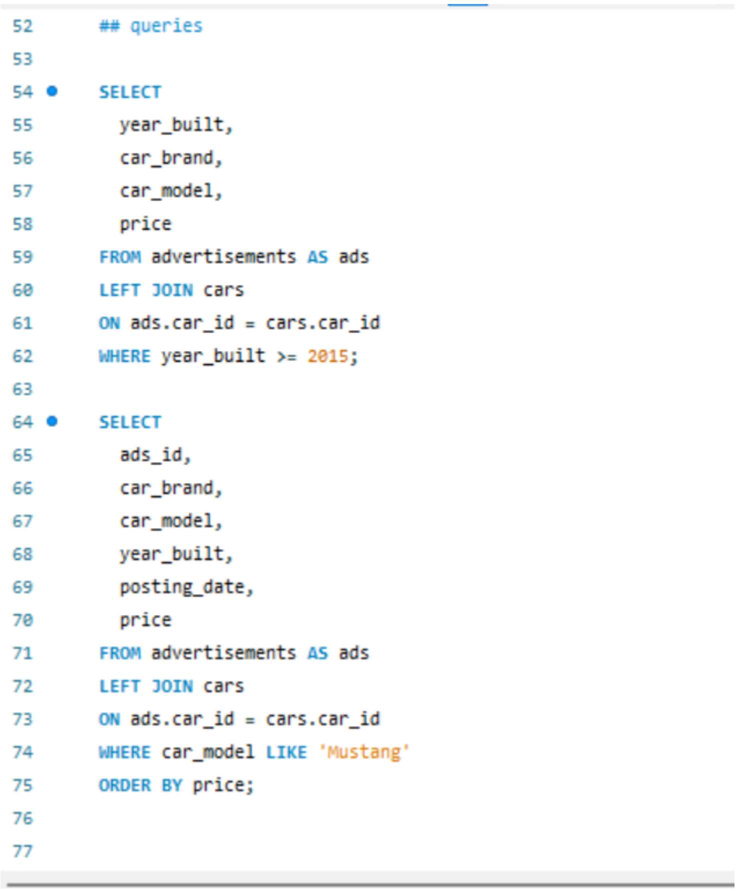
DDL STATEMENTS:



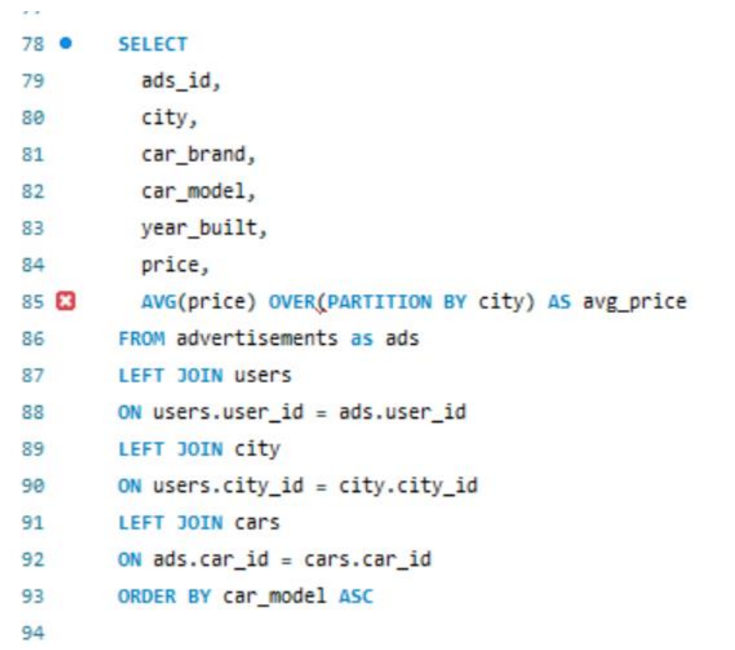
```
CREATE TABLE City(
    city_id int UNIQUE PRIMARY KEY NOT NUll,
    city varchar(50) NOT NULL,
    province varchar(50),
    postal_code int,
    latitude float(24),
    longitude float(24)
);

CREATE TABLE Users(
    user_id int PRIMARY KEY NOT NUll,
    first_name varchar(50) NOT NULL,
    last_name varchar(50) NOT NULL,
    phone varchar(20) NOT NULL,
    address text NOT NULL,
    city_id int NOT NULL,
    FOREIGN KEY (city_id) REFERENCES City(city_id)
);

CREATE TABLE Cars(
    car_id int PRIMARY KEY NOT NULL,
    car_brand varchar(50) NOT NULL,
    car_model varchar(50) NOT NULL,
    car_type varchar(50) NOT NULL,
    trans_type varchar(50) NOT NULL
);
```



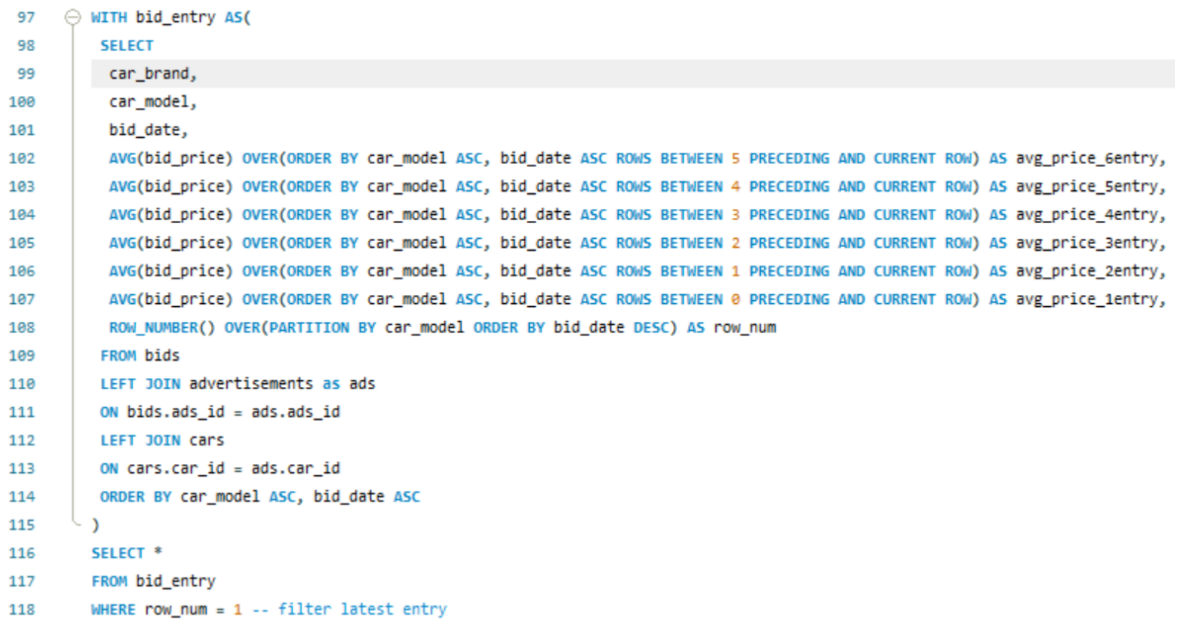
```
CREATE TABLE Advertisements(
    ads_id int PRIMARY KEY NOT NUll,
    user_id int NOT NULL,
    car_id int NOT NULL,
    title text NOT NULL,
    posting_date date,
    year_built int NOT NULL,
    color varchar(50) NOT NULL,
    mileage int NOT NULL,
    price int NOT NULL,
    FOREIGN KEY (user_id) REFERENCES Users(user_id),
    FOREIGN KEY (car_id) REFERENCES Cars(car_id)
);

CREATE TABLE Bids(
    bid_id int PRIMARY KEY NOT NULL,
    user_id int NOT NULL,
    ads_id int NOT NULL,
    bid_date date,
    bid_price int NOT NULL,
    FOREIGN KEY (user_id) REFERENCES Users(user_id),
    FOREIGN KEY (ads_id) REFERENCES advertisements(ads_id)
);
```



```
## queries

SELECT
  year_built,
  car_brand,
  car_model,
  price
FROM advertisements AS ads
LEFT JOIN cars
ON ads.car_id = cars.car_id
WHERE year_built >= 2015;

SELECT
  ads_id,
  car_brand,
  car_model,
  year_built,
  posting_date,
  price
FROM advertisements AS ads
LEFT JOIN cars
ON ads.car_id = cars.car_id
WHERE car_model LIKE 'Mustang'
ORDER BY price;
```



```
SELECT
  ads_id,
  city,
  car_brand,
  car_model,
  year_built,
  price,
  AVG(price) OVER(PARTITION BY city) AS avg_price
FROM advertisements as ads
LEFT JOIN users
ON users.user_id = ads.user_id
LEFT JOIN city
ON users.city_id = city.city_id
LEFT JOIN cars
ON ads.car_id = cars.car_id
ORDER BY car_model ASC
```



```
WITH bid_entry AS(
 SELECT
  car_brand,
  car_model,
  bid_date,
  AVG(bid_price) OVER(ORDER BY car_model ASC, bid_date ASC ROWS BETWEEN 5 PRECEDING AND CURRENT ROW) AS avg_price_6entry,
  AVG(bid_price) OVER(ORDER BY car_model ASC, bid_date ASC ROWS BETWEEN 4 PRECEDING AND CURRENT ROW) AS avg_price_5entry,
  AVG(bid_price) OVER(ORDER BY car_model ASC, bid_date ASC ROWS BETWEEN 3 PRECEDING AND CURRENT ROW) AS avg_price_4entry,
  AVG(bid_price) OVER(ORDER BY car_model ASC, bid_date ASC ROWS BETWEEN 2 PRECEDING AND CURRENT ROW) AS avg_price_3entry,
  AVG(bid_price) OVER(ORDER BY car_model ASC, bid_date ASC ROWS BETWEEN 1 PRECEDING AND CURRENT ROW) AS avg_price_2entry,
  AVG(bid_price) OVER(ORDER BY car_model ASC, bid_date ASC ROWS BETWEEN 0 PRECEDING AND CURRENT ROW) AS avg_price_1entry,
  ROW_NUMBER() OVER(PARTITION BY car_model ORDER BY bid_date DESC) AS row_num
 FROM bids
 LEFT JOIN advertisements as ads
 ON bids.ads_id = ads.ads_id
 LEFT JOIN cars
 ON cars.car_id = ads.car_id
 ORDER BY car_model ASC, bid_date ASC
)
SELECT *
FROM bid_entry
WHERE row_num = 1 -- filter latest entry
```

CONCLUSION

A great opportunity exists in the dynamic world of online marketplaces for the construction of a relational database system tailored for the used automobile industry. Delivering a dependable and scalable solution that meets the changing needs of consumers and sellers in the automobile industry has been the goal of this project. Utilizing cutting-edge technologies and following accepted best practices in database architecture, we have laid the groundwork for an intuitive and effective online platform. With an eye to the future, this well-thought-out database system provides a solid basis for expansion. While user experience emphasizes a seamless and effective exchange for both buyers and sellers, scalability assures that the system can support substantial expansion. This creates the foundation for a vibrant ecology of online used car marketplaces, which in turn streamlines the entire process of purchasing and selling cars.

REFERENCES

- Used Cars Data Analysis and Visualization (EDA). (n.d.). Kaggle.com. Retrieved March 14, 2024, from https://www.kaggle.com/code/ismailsefa/used-cars-data-analysis-and-visualization-eda
- Why Dealer Management Software? (n.d.). Pagoda Dealer Management Software. Retrieved March 14, 2024, from https://pagodadms.com/blog/why-dealer-management-software/
- Wikipedia Contributors. (2019, March 1). Customer-relationship management. Wikipedia; Wikimedia Foundation. https://en.wikipedia.org/wiki/Customer_relationship_management
- Case Studies. (n.d.). Resources.automotivemastermind.com. Retrieved March 14, 2024, from https://resources.automotivemastermind.com/car-dealer-case-studies
- Ahuja's, J. (n.d.). Best Used Luxury Car Dealers India, Bigboytoyz. Www.bigboytoyz.com; Ahuja's. https://www.bigboytoyz.com/
- Yalcin, S. (n.d.). SellAnyCar.com | Sell Any Car to us! Uae.sellanycar.com. https://uae.sellanycar.com/
- "A Scalable Database Architecture for E-commerce Applications" by J. Yang, J. Wang, and H. Zhao (2019).
- "Designing an Online Auction System for E-commerce" by H. Lin, W. Yu, and Z. Zheng (2018).
- MySQL Documentation: https://dev.mysql.com/doc/
- "Database Systems: Concepts and Design" by Ramesh Agarwal and Vipin Ramakrishnan (2020).
- "Data Management for E-commerce: Concepts and Techniques" by P.K. Sinha (2017).