Efficient Relational Database Design for Used Car Marketplace Platform

For Database Administrators

Artikel ini bertujuan untuk membahas langkah-langkah dalam merancang sebuah sistem basis data yang memadai untuk mendukung operasional platform jual beli mobil bekas secara online. Artikel ini akan menjelaskan secara terperinci tentang desain struktur tabel, hubungan antar tabel, serta kunci primer dan kunci asing yang diperlukan untuk memastikan integrasi data yang efisien dan efektif. Dengan demikian, artikel ini akan memberikan panduan komprehensif bagi pembaca dalam merancang dan mengimplementasikan sistem basis data untuk aplikasi jual beli mobil bekas online.

Project's objectives

Membangun sebuah relational database untuk sebuah website penjualan mobil bekas. Setiap pengguna (seller) dapat menawarkan mobil bekasnya melalui iklan yang akan ditampilkan di situs web. Sebelum dapat menjual mobil, pengguna harus melengkapi profil mereka dengan informasi seperti nama, kontak, dan lokasi. Setiap iklan akan berisi informasi detail tentang mobil yang ditawarkan, termasuk merek, model, jenis body, tipe transmisi, tahun pembuatan, dan deskripsi tambahan seperti warna dan jarak tempuh. Pengguna (buyer) dapat mencari mobil berdasarkan lokasi penjual, merek, dan jenis body mobil. Fitur penawaran harga tersedia hanya jika penjual mengaktifkan fitur tawar, tetapi transaksi pembelian dilakukan di luar aplikasi dan tidak termasuk dalam lingkup proyek ini.Berikut sejumlah tujuan spesifik yang harus dicapai dalam rancangan sistem:

1. **Mengelola Profil Pengguna:** Sistem akan memungkinkan pengguna (*seller*) untuk membuat dan mengelola profil mereka dengan informasi seperti nama, kontak, dan lokasi. Ini diperlukan sebelum mereka dapat menawarkan mobil bekas mereka melalui iklan.
2. **Pengelolaan Iklan Mobil:** Pengguna (*seller*) dapat membuat iklan untuk mobil bekas mereka, yang akan ditampilkan di situs web. Setiap iklan akan berisi informasi detail tentang mobil yang ditawarkan, seperti merek, model, jenis body, tipe transmisi, tahun pembuatan, dan deskripsi tambahan seperti warna dan jarak tempuh.
3. **Pencarian Mobil Berdasarkan Kriteria:** Pengguna (*buyer*) dapat mencari mobil bekas berdasarkan lokasi penjual, merek, dan jenis body mobil. Ini memungkinkan pengguna untuk menemukan mobil yang sesuai dengan preferensi dan lokasi mereka.
4. **Fitur Tawar Harga (Opsional):** Fitur penawaran harga tersedia jika penjual mengaktifkan fitur tawar. Ini memungkinkan pembeli untuk menawar harga produk. Namun, transaksi pembelian dilakukan di luar aplikasi dan tidak termasuk dalam lingkup proyek.

Designing The Database

Dalam perancangan database untuk sistem penjualan mobil bekas yang efektif, langkah-langkah yang diperlukan dimulai dengan pemahaman yang mendalam tentang tujuan akhir dari aplikasi tersebut. Ini melibatkan identifikasi objek-objek kunci yang diperlukan untuk database, termasuk informasi tentang pengguna, iklan mobil, dan detail penawaran. Setelah tabel-tabel tersebut diidentifikasi, langkah selanjutnya adalah menentukan hubungan antara tabel-tabel tersebut dengan menentukan jenis hubungan antar tabel. Penerapan aturan bisnis juga merupakan tahap penting dalam perancangan database. Ini mencakup penerapan *constraint* untuk memastikan integritas data, seperti memastikan setiap iklan mobil memiliki merek dan model, serta memastikan bahwa kuantitas mobil tidak boleh bernilai negatif. Hasil dari perancangan ini adalah Diagram Hubungan Entitas (ERD), yang merupakan representasi visual dari struktur database dan hubungan antar tabel.

Pernyataan Misi

Dalam tahap awal proyek pembuatan database relasional untuk aplikasi penjualan mobil bekas, langkah pertama adalah merumuskan pernyataan misi yang akan menjadi panduan dalam menentukan objek-objek yang diperlukan untuk database. Pernyataan misi ini penting untuk

mengidentifikasi tujuan inti dari proyek ini yaitu terkait proses perancangan struktur database yang efektif dan efisien. Dengan demikian, perumusan misi menjadi landasan yang kuat untuk memastikan bahwa database yang dibangun akan sesuai dengan kebutuhan dan tujuan akhir dari aplikasi penjualan mobil bekas.

1. Memastikan setiap pengguna aplikasi melengkapi data diri mereka sebelum dapat menawarkan mobil bekas, termasuk nama, kontak, lokasi domisili, dan informasi revelan lainnya.
2. Mengembangkan fitur iklan yang memungkinkan pengguna untuk menampilkan informasi detail produk, termasuk merek, model, jenis body mobil, tipe, tahun pembuatan, dan deskripsi lainnya.
3. Menyediakan fitur pencarian yang memungkinkan pengguna mencari mobil berdasarkan lokasi penjual, merek, dan jenis body mobil.
4. Menyediakan fitur penawaran harga (bid) jika penjual mengizinkannya, sehingga memungkinkan calon pembeli untuk melakukan negosiasi harga.

Membuat Struktur Tabel

Langkah berikutnya adalah mengidentifikasi objek-objek yang esensial untuk mencapai tujuan yang telah ditetapkan pada bagian pernyataan misi project. Objek-objek ini kemudian akan diterjemahkan menjadi tabel-tabel dalam database yang sedang dibangun. Selanjutnya, akan ditetapkan atribut-atribut yang relevan untuk setiap tabel, serta kunci-kunci yang diperlukan untuk memastikan integritas dan keterhubungan data secara efisien.

Pernyataan Misi	Nama Tabel
Memastikan setiap pengguna aplikasi melengkapi data diri mereka sebelum dapat menawarkan mobil bekas, termasuk nama, kontak, lokasi domisili, dan informasi revelan lainnya.	`User_Data; City`
Mengembangkan fitur iklan yang memungkinkan pengguna untuk menampilkan informasi detail produk, termasuk merek, model, jenis body mobil, tipe, tahun pembuatan, dan deskripsi lainnya.	`Advertisement`

Menyediakan fitur pencarian yang memungkinkan pengguna mencari mobil berdasarkan lokasi penjual, merek, dan jenis body mobil.	Car
Menyediakan fitur penawaran harga (bid) jika penjual mengizinkannya, sehingga memungkinkan calon pembeli untuk melakukan negosiasi harga.	Bids

Langkah selanjutnya adalah mengidentifikasi setiap tabel, memberikan deskripsi yang jelas, dan menetapkan atribut-atribut yang relevan untuk setiap tabel tersebut. Langkah ini krusial dalam merancang struktur database agar sesuai dengan kebutuhan untuk menyimpan informasi dari aplikasi penjualan mobil bekas. Dengan merinci setiap tabel dan atributnya dengan cermat, akan memungkinkan pembangunan database yang efisien dan dapat mendukung semua fungsi yang diperlukan dalam operasional aplikasi.

Nama Tabel	Deskripsi Tabel
User_Data	Tabel ini digunakan untuk menyimpan informasi tentang pengguna (user) aplikasi penjualan mobil bekas. Setiap baris dalam tabel ini mewakili satu pengguna dengan informasi lengkap terkait identitas dan kontaknya.
City	Tabel ini digunakan untuk menyimpan informasi tentang kota-kota di mana pengguna aplikasi penjualan mobil bekas berada. Setiap baris dalam tabel ini merepresentasikan satu kota beserta detailnya.
Advertisement	Tabel ini digunakan untuk menyimpan informasi tentang iklan yang diposting oleh pengguna aplikasi penjualan mobil bekas. Setiap baris dalam tabel ini merepresentasikan satu iklan yang berisi detail tentang mobil bekas yang ditawarkan.
Car	Tabel ini digunakan untuk menyimpan informasi tentang mobil bekas yang ditawarkan dalam aplikasi penjualan mobil bekas. Setiap baris dalam tabel ini merepresentasikan satu mobil bekas yang dapat diiklankan oleh pengguna.
Bids	Tabel ini digunakan untuk menyimpan informasi tentang tawaran harga yang diajukan oleh calon pembeli untuk sebuah iklan mobil bekas dalam aplikasi penjualan mobil bekas. Setiap baris dalam tabel ini merepresentasikan satu tawaran harga yang diajukan.

Tabel `City`

Atribut	Deskripsi Atribut	Kunci	
`city_id`	ID unik yang digunakan untuk mengidentifikasi setiap kota dalam database.	CK	**PK**
`city_name`	Nama dari setiap kota	CK	AK
`latitude`	Koordinat lintang geografis dari lokasi kota		
`longitude`	Koordinat bujur geografis dari lokasi kota		

Tabel `city` memuat informasi tentang kota – kota dalam database. Berikut adalah penjelasan singkat untuk setiap atribut beserta karakteristik dan limitasi dari setiap atribut:

A. `city_id`:

- **Deskripsi Atribut**: Id unik yang digunakan untuk mengidentifikasi setiap kota dalam database.
- **Karakteristik**:
  a. Merupakan *Primary Key* yang memastikan keunikan setiap baris dalam tabel.
  b. Nilai dari `city_id` tidak boleh sama dengan yang lain.
  c. Digunakan untuk mengidentifikasi setiap kota secara unik dalam aplikasi.

B. `city_name`:

- **Deskripsi Atribut**: Nama dari setiap kota.
- **Karakteristik**:
  a. Menyimpan nama dari masing-masing kota.
  b. Nama kota ini digunakan untuk tujuan deskriptif dan identifikasi dalam aplikasi.
  c. Meskipun tidak secara eksplisit nama kota harus unik, dalam prakteknya sering kali nama kota digunakan sebagai identifikasi unik.

C. `latitude`:

- **Deskripsi Atribut**: Koordinat lintang geografis dari lokasi kota.
- **Karakteristik**:
  a. Menyimpan nilai koordinat lintang dari lokasi kota.
  b. Digunakan untuk penentuan lokasi geografis dalam aplikasi.
  c. Nilai harus dalam rentang tertentu yang mencakup lintang geografis valid.

D. `longitude`:

- **Deskripsi Atribut:** Koordinat bujur geografis dari lokasi kota.
- **Karakteristik:**
  a. Menyimpan nilai koordinat bujur dari lokasi kota.
  b. Digunakan untuk penentuan lokasi geografis dalam aplikasi, bersama dengan atribut latitude.
  c. Nilai harus dalam rentang tertentu yang mencakup bujur geografis valid.

Selanjutnya adalah memvisualisasikan nilai dari tabel `city` untuk setiap atribut dengan maksud agar pemeriksaan dapat dilakukan secara lebih cermat terkait hubungan antar atribut dalam tabel untuk mengidentifikasi kemungkinan dependensi parsial dan transitif. Pendekatan ini memungkinkan untuk menentukan apakah ada atribut yang bergantung hanya pada sebagian atribut lainnya (dependensi parsial), atau apakah ada atribut yang bergantung pada atribut non-kunci melalui atribut lain dalam tabel (dependensi transitif). Ini memungkinkan untuk memastikan bahwa struktur data yang dirancang memenuhi kebutuhan aplikasi dengan baik, menghindari masalah seperti redundansi data dan anomali pengelolaan data.

city_id	city_name	latitude	longitude
1	Jakarta	-62.088	1.068.456
2	Bandung	-69.175	1.076.191
3	Surabaya	-72.575	1.127.521
4	Semarang	-69.932	1.104.203
5	Yogyakarta	-77.956	1.103.695
6	Medan	35.952	986.722
7	Makassar	-51.477	1.194.327
8	Palembang	-29.761	1.047.751

Dependensi parsial terjadi ketika sebuah atribut non-kunci bergantung pada sebagian dari kunci utama, bukan keseluruhan kunci utama. Melalui visualisasi tabel `city` diatas, sepertinya tidak ada dependensi parsial yang terlihat. Setiap atribut non-kunci (`city_name`, `latitude`, `longitude`) sepenuhnya tergantung pada seluruh kunci utama (`city_id`). Tidak ada atribut non-kunci yang hanya bergantung pada sebagian dari kunci utama. Selain itu juga tidak terjadi dependensi transitif, dimana dependensi transitif terjadi ketika sebuah atribut non-kunci bergantung pada atribut non-kunci lain melalui kunci utama. Dalam tabel `city` yang diberikan, tidak terdapat dependensi transitif yang terlihat. Setiap atribut non-kunci (`city_name`, `latitude`, `longitude`) bergantung langsung pada kunci utama (`city_id`) dan tidak bergantung pada atribut non-kunci lainnya.

Tabel `User_Data`

Atribut	Deskripsi Atribut	Kunci	
`user_id`	Nomor identifikasi unik untuk setiap pengguna dalam database	CK	**PK**
`first_name`	Nama depan pengguna		
`last_name`	Nama akhir pengguna		
`birth_date`	Tanggal lahir pengguna		
`contact_num`	Nomor kontak pengguna	CK	AK
`email`	Alamat email pengguna	CK	AK
`address`	Alamat lengkap tempat tinggal pengguna		
`city_id`	Id kota tempat tinggal pengguna		FK

Tabel `User_Data` menggambarkan informasi pengguna dalam aplikasi penjualan mobil bekas. Berikut adalah penjelasan singkat untuk setiap atribut beserta karakteristik dan limitasi dari setiap atribut:

A. `user_id`:

- **Deskripsi Atribut**: Nomor identifikasi unik untuk setiap pengguna dalam database.
- **Kunci**: PK (Primary Key)
- **Karakteristik**:
  a. Menyediakan identifikasi unik untuk setiap pengguna.
  b. Nilainya harus unik dan tidak boleh sama dengan user_id lain.
  c. Digunakan sebagai kunci utama untuk mengidentifikasi setiap entitas pengguna.

B. `first_name:`

- **Deskripsi Atribut**: Nama depan pengguna.
- **Karakteristik**:
    a. Menyimpan nama depan dari setiap pengguna.
    b. Tidak harus unik, karena beberapa pengguna dapat memiliki nama depan yang sama

C. `last_name`:

- **Deskripsi Atribut**: Nama akhir pengguna.
- **Karakteristik**:
    a. Menyimpan nama belakang dari setiap pengguna.
    b. Tidak harus unik, karena beberapa pengguna dapat memiliki nama belakang yang sama,

D. `birth_date`:

- **Deskripsi Atribut**: Tanggal lahir pengguna.
- Karakteristik:
    a. Menyimpan tanggal lahir dari setiap pengguna.
    b. Digunakan untuk keperluan verifikasi usia atau statistik demografis untuk pengguna.
    c. Tidak ada limitasi spesifik, tetapi biasanya dalam format tanggal.

E. `contact_num`:

- **Deskripsi Atribut**: Nomor kontak pengguna.
- **Karakteristik**:
    a. Menyimpan nomor kontak atau telepon dari setiap pengguna.
    b. Harus unik, setiap pengguna tidak dapat memiliki nomor kontak yang sama.

F. `email`:

- **Deskripsi Atribut**: Alamat email pengguna.
- **Karakteristik**:
  a. Menyimpan alamat email unik dari setiap pengguna.

G. `address`:

- **Deskripsi Atribut**: Alamat lengkap tempat tinggal pengguna.
- **Karakteristik**:
  a. Menyimpan alamat tempat tinggal lengkap dari setiap pengguna, termasuk detail seperti nama jalan, nomor rumah, dan informasi tambahan lain.

H. `city_id`:

- **Deskripsi Atribut**: ID kota tempat tinggal pengguna.
- **Karakteristik**:
  a. Menyimpan ID kota tempat tinggal pengguna.
  b. Digunakan untuk menghubungkan pengguna dengan data geografis seperti nama kota, negara bagian, atau negara.
  c. Merujuk ke kunci utama atau kunci unik pada tabel kota (`city`).

Dengan memvisualisasikan nilai dari tabel `User_Data` untuk setiap atribut, analisis yang cermat terhadap hubungan di antara mereka memungkinkan identifikasi terhadap kemungkinan adanya gejala dependensi parsial dan transitif. Pendekatan ini memungkinkan pengamatan terhadap atribut yang mungkin bergantung hanya pada sebagian atribut lainnya (dependensi parsial), serta pengamatan terhadap kemungkinan adanya atribut yang bergantung pada atribut non-kunci melalui atribut lain dalam tabel (dependensi transitif). Dengan demikian, peninjauan menyeluruh melalui tabel visualisasi terhadap hubungan antar atribut dalam tabel User_Data menjadi penting untuk memastikan desain database yang optimal dan memenuhi kebutuhan fungsional aplikasi. Ini membantu dalam mengidentifikasi dan mengatasi potensi redundansi data, anomali pengelolaan data sehingga integritas data yang dikelola dapat dipastikan efisien dan efektif.

user_id	first_name	last_name	birth_date	contact_num	email	address	city_id
1	Adi	Wibowo	01/01/1990	8123456789	adi.wibowo@example.com	15 Jl. Gatot Subroto	6
2	Budi	Susilo	15/03/1992	8234567890	budi.susilo@example.com	89 Jl. Gatot Subroto	5
3	Citra	Kusuma	20/05/1988	8345678901	citra.kusuma@example.com	55 Jl. Sudirman	7
4	Dewi	Wijaya	10/07/1995	8456789012	dewi.wijaya@example.com	19 Jl. Diponegoro	4
5	Eka	Santoso	05/09/1985	8567890123	eka.santoso@example.com	53 Jl. Diponegoro	2
6	Fajar	Purnomo	12/11/1998	8678901234	fajar.purnomo@example.com	85 Jl. Gatot Subroto	5
7	Gita	Pratama	25/02/1993	8789012345	gita.pratama@example.com	5 Jl. Gatot Subroto	8
8	Hadi	Putra	30/04/1987	8890123456	hadi.putra@example.com	15 Jl. Sudirman	2
9	Indra	Mulyono	18/08/1991	8901234567	indra.mulyono@example.com	67 Jl. Sudirman	2
10	Joko	Wijaya	03/12/1989	9012345678	joko.wijaya@example.com	85 Jl. Gatot Subroto	6

Dari tabel visualisasi tersebut, terbukti tabel `User_Data` tidak menunjukkan adanya gejala dependensi parsial atau transitif. Setiap atribut, seperti `first_name`, `last_name`, `birth_date`, `contact_num`, `email`, `address`, dan `city_id` berdiri sendiri dan tidak bergantung pada bagian-bagian lain dalam tabel. Misalnya, atribut `first_name` tidak bergantung pada atribut lainnya seperti `last_name` atau `birth_date`. Begitu juga, tidak ada atribut yang bergantung pada atribut non-kunci melalui atribut lainnya. Sebagai contoh, atribut `address` tidak bergantung pada atribut `city_id` atau atribut lainnya. Dengan kata lain, semua informasi dalam tabel tersebut tersusun secara independen dan tidak saling tergantung.

Tabel `Car`

Atribut	Deskripsi Atribut	Kunci	
`car_id`	Nomor identifikasi unik untuk setiap mobil dalam database	CK	**PK**
`car_brand`	Merek atau produsen mobil		
`car_model`	Model spesifik dari mobil		
`car_type`	Jenis bodi atau kategori mobil		
`car_transmission`	Jenis transmisi mobil		
`year`	Tahun pembuatan atau perakitan mobil		

Tabel `Car` menggambarkan informasi tentang mobil. Berikut adalah penjelasan singkat untuk setiap atribut:

A. `car_id`:

- **Deskripsi atribut**: Nomor identifikasi unik untuk setiap mobil dalam database
- **Karakteristik:**
    a. Merupakan kunci utama (*Primary Key*) yang memastikan keunikan setiap baris dalam tabel.
    b. Digunakan untuk mengidentifikasi setiap mobil secara unik.

B. `car_brand`:

- **Deskripsi atribut**: Merek atau produsen mobil.
- **Karakteristik:**
    a. Menyimpan informasi tentang merek mobil, contoh nilai: Toyota, Honda, Nissan.

C. `car_model`:

- **Deskripsi atribut**: Model spesifik dari mobil.
- **Karakteristik:**
    a. Menyimpan model tertentu dari merek mobil, contoh misalnya Toyota **Camry**, Honda **Civic**, atau Nissan **X-Trail**.

D. `car_type`:

- **Deskripsi atribut**: Jenis bodi atau kategori mobil.
- **Karakteristik**:
    a. Menyimpan informasi tentang jenis bodi mobil, contoh nilai: Sedan, SUV, Hatchback.

E. `car_transmission`:

- **Deskripsi atribut**: Jenis transmisi mobil.
- Karakteristik:
    a. Menyimpan informasi tentang jenis transmisi mobil, apakah manual atau otomatis.

F. `year`:

- **Deskripsi atribut**: Tahun pembuatan atau perakitan mobil.
- **Karakteristik**:
    a. Menyimpan tahun pembuatan mobil

Dengan memvisualisasikan nilai dari tabel `Car` untuk setiap atribut, dapat dilakukan pemeriksaan yang mendalam terhadap hubungan di antara mereka untuk mengidentifikasi kemungkinan dependensi parsial dan transitif. Pendekatan ini memungkinkan untuk menentukan apakah terdapat atribut yang bergantung hanya pada sebagian atribut lainnya (dependensi parsial), atau apakah ada atribut yang bergantung pada atribut non-kunci melalui atribut lain dalam tabel (dependensi transitif).

car_id	car_brand	car_model	car_type	car_transmission	year
1	Toyota	Camry	Sedan	Automatic	2015
2	Honda	Civic	Sedan	Manual	2018
3	Nissan	X-Trail	SUV	Automatic	2017
4	Ford	Focus	Hatchback	Manual	2019
5	Chevrolet	Cruze	Sedan	Automatic	2016
6	BMW	3 Series	Sedan	Automatic	2020
7	Mercedes-Benz	C-Class	SUV	Automatic	2018
8	Audi	A4	Sedan	Automatic	2017
9	Hyundai	Accent	Hatchback	Manual	2016
10	Kia	Rio	Hatchback	Automatic	2019

Dari tabel visualisasi di atas tidak menunjukkan adanya dependensi parsial. Setiap atribut seperti `car_brand`, `car_model`, `car_type`, `car_transmission`, dan `year` tidak bergantung pada subset atribut lainnya. Misalnya, `car_brand` tidak bergantung pada atribut lain seperti `car_model` atau `car_type`. Semua atribut bersifat mandiri dan tidak tergantung pada atribut lainnya. Selain itu juga tidak menunjukkan adanya dependensi transitif. Dependensi transitif terjadi ketika atribut non-kunci bergantung pada atribut non-kunci melalui atribut lain dalam tabel. Dalam tabel visualisasi tersebut, tidak ada atribut non-kunci yang bergantung pada atribut non-kunci melalui atribut lainnya. Sebagai contoh, atribut `car_model` tidak bergantung pada atribut non-kunci seperti `car_type` atau atribut lainnya untuk nilainya. Oleh karena itu, tidak ada dependensi transitif yang teridentifikasi.

Tabel `Advertisement`

Atribut	Deskripsi Atribut	Kunci	
`ads_id`	Nomor identifikasi unik untuk setiap iklan dalam database.	CK	**PK**
`title`	Menyimpan judul atau deskripsi singkat dari iklan yang diposting		
`date_posted`	Periode tanggal dan waktu iklan diposting		
`odometer`	Kilometer yang telah ditempuh oleh mobil yang diiklankan		
`color`	Warna mobil yang diiklankan		
`price`	Harga mobil yang diiklankan		
`user_id`	Nomor identifikasi pengguna yang memposting iklan		FK
`car_id`	Nomor identifikasi mobil yang diiklankan		FK

Tabel `Advertisement`:

A. `ads_id`:

- **Deskripsi Atribut**: Nomor identifikasi unik untuk setiap iklan dalam database.
- **Karakteristik**:
  a. Bertindak sebagai kunci utama (*Primary Key*) untuk memastikan keunikan setiap baris dalam tabel.

B. `title`:

- **Deskripsi Atribut**: Menyimpan judul atau deskripsi singkat dari iklan yang diposting.
- **Karakteristik**:
  a. Memberikan deskripsi singkat tentang judul atau konten iklan.

C. `date_posted`:

- **Deskripsi Atribut**: Periode tanggal dan waktu iklan diposting.
- **Karakteristik**:
  a. Menyimpan informasi tentang kapan iklan dipublikasikan, membantu dalam pelacakan riwayat iklan.

D. `odometer`:

- **Deskripsi Atribut**: Kilometer yang telah ditempuh oleh mobil yang diiklankan.
- **Karakteristik**:
  a. Memberikan informasi tentang seberapa jauh mobil tersebut telah digunakan, penting bagi pembeli yang mempertimbangkan kondisi mobil.

E. `color`:

- **Deskripsi Atribut**: Warna mobil yang diiklankan.
- **Karakteristik**:
  a. Menyediakan deskripsi visual tentang mobil yang diiklankan, membantu pembeli dalam pemilihan mobil.

F. `price`:

- **Deskripsi Atribut**: Harga mobil yang diiklankan.
- **Karakteristik**:
  a. Menyimpan informasi tentang harga jual mobil, memungkinkan pembeli untuk menilai mobil berdasarkan harga yang diiklankan.

G. `user_id`:

- **Deskripsi Atribut**: Nomor identifikasi pengguna yang memposting iklan.
- **Karakteristik**:
  a. Mengidentifikasi pengguna yang melakukan aktifias pengiklanan, memungkinkan pelacakan aktivitas pengguna.

H. `car_id`:

- **Deskripsi Atribut**: Nomor identifikasi mobil yang diiklankan.
- **Karakteristik**:
  a. Menghubungkan iklan dengan informasi spesifik tentang mobil, seperti merek, model, dan tahun, memfasilitasi pencarian dan pemilihan mobil oleh pembeli.

Selanjutnya untuk menganalisis dependensi parsial dan transitif pada databse, perlu dilakukan pemeriksaan hubungan antara atribut – atribut dalam tabel `Advertisement` dan memvisualisasikan nilai untuk setiap atribut, sebagai berikut :

ads_id	title	date_posted	odometer	color	price	user_id	car_id
1	Excellent Condition - Low Mileage	2022-05-01 08:30:00	50000	Red	200000000	1	1
2	Great Deal - Perfect for City Driving	2022-05-02 10:15:00	40000	Blue	180000000	2	2
3	Sporty Look - Luxury Experience	2022-05-03 13:45:00	30000	Black	250000000	3	3
4	Elegant Style - Performance Redefined	2022-05-04 09:20:00	20000	White	220000000	4	5
5	Budget Friendly - Compact and Efficient	2022-05-05 11:00:00	10000	Gray	150000000	5	6
6	Excellent Condition - Low Mileage	2022-05-06 14:30:00	45000	Red	190000000	6	1
7	Great Deal - Perfect for City Driving	2022-05-07 08:00:00	35000	Blue	170000000	7	2
8	Sporty Look - Luxury Experience	2022-05-08 10:45:00	25000	Black	240000000	8	4
9	Elegant Style - Performance Redefined	2022-05-09 13:15:00	15000	White	210000000	9	4
10	Budget Friendly - Compact and Efficient	2022-05-10 09:30:00	5000	Gray	160000000	10	1

Tabel visualisasi untuk tabel `Advertisement` di atas tidak menunjukkan adanya dependensi parsial. Setiap atribut, seperti `title`, `date_posted`, `odometer`, `color`, `price`, `user_id`, dan `car_id`, tidak bergantung pada subset atribut lainnya. Misalnya, atribut `title` tidak bergantung pada atribut lain seperti `odometer` atau `color`. Semua atribut bersifat mandiri dan tidak tergantung pada atribut lainnya. Tabel tersebut juga tidak menunjukkan adanya dependensi transitif. Dependensi transitif terjadi ketika atribut non-kunci bergantung pada atribut non-kunci melalui atribut lain dalam tabel. Dalam tabel `Advertisement`, tidak ada atribut non-kunci yang bergantung pada atribut non-kunci melalui atribut lainnya. Sebagai contoh, atribut `price` tidak bergantung pada atribut `odometer` atau atribut lain. Oleh karena itu, tidak ada dependensi transitif yang teridentifikasi sehingga tidak perlu dilakukan normalisasi tabel.

Tabel `Bids`

Atribut	Deskripsi Atribut	Kunci	
`bid_id`	Nomor identifikasi unik untuk setiap penawaran dalam database	CK	**PK**
`bid_price`	Harga yang ditawarkan dalam suatu penawaran		
`bid_posted`	Periode tanggal dan waktu penawaran diposting		
`ads_id`	Nomor identifikasi unik untuk setiap iklan yang diberi penawaran		FK
`bid_buyer_id`	Nomor identifikasi pembeli yang mengajukan penawaran		FK

Tabel `Bids`:

A. `bid_id`:

- **Deskripsi Atribut**: Nomor identifikasi unik untuk setiap penawaran dalam database.
- **Karakteristik**:
  a. Berperan sebagai kunci utama (*Primary Key*) untuk memastikan keunikan setiap baris dalam tabel.

B. `bid_price`:

- **Deskripsi Atribut**: Harga yang ditawarkan dalam suatu penawaran.
- **Karakteristik**:
  a. Menyimpan nilai harga yang diajukan oleh pengguna untuk suatu iklan.

C. `bid_posted`:

- **Deskripsi Atribut**: Periode tanggal dan waktu penawaran diposting.
- **Karakteristik**:
  a. Menyimpan informasi tentang kapan penawaran diajukan, membantu dalam pelacakan riwayat penawaran.

D. `ads_id`:

- **Deskripsi Atribut**: Nomor identifikasi unik untuk setiap iklan yang diberi penawaran.
- **Karakteristik**:
  a. Menghubungkan penawaran dengan iklan tertentu, memungkinkan pelacakan penawaran yang terkait dengan iklan.
  b. Pembeli dapat melakukan penawaran (*bid*) harga mobil tertentu apabila penjual mengizinkan fitur tawar.

E. `bid_buyer_id`:

- **Deskripsi Atribut**: Nomor identifikasi pengguna (*buyer*) yang mengajukan penawaran.
- **Karakteristik**:
  a. Mengidentifikasi pengguna (*buyer*) yang melakukan penawaran, memfasilitasi pelacakan aktivitas penawaran oleh pengguna tertentu.

Dengan memeriksa nilai dari setiap atribut dalam tabel `Bids`, dapat dilakukan analisa terkait hubungan di antara mereka untuk menemukan kemungkinan dependensi parsial dan transitif. Pendekatan ini memungkinkan untuk menentukan apakah ada atribut yang bergantung hanya pada sebagian atribut lainnya (dependensi parsial), atau apakah ada atribut yang bergantung pada atribut non-kunci melalui atribut lain dalam tabel (dependensi transitif).

bid_id	bid_price	bid_posted	ads_id	bid_buyer_id
1	150000000	2022-05-01 08:30:00	1	11
2	160000000	2022-05-02 10:15:00	2	12
3	170000000	2022-05-03 13:45:00	3	14
4	100000000	2022-05-04 09:20:00	5	14
5	100000000	2022-05-05 11:00:00	6	15
6	180000000	2022-05-06 14:30:00	1	15
7	90000000	2022-05-07 08:00:00	2	16
8	215000000	2022-05-08 10:45:00	4	18
9	190000000	2022-05-09 13:15:00	4	29
10	140000000	2022-05-10 09:30:00	1	20

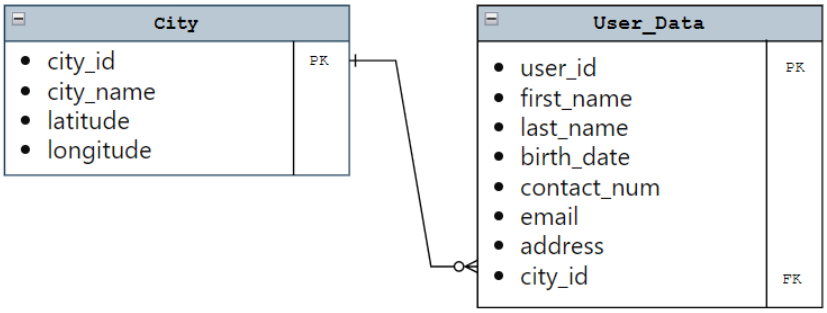
Tidak terdapat dependensi parsial yang terlihat dalam visualisasi tabel `Bids` di atas. Setiap atribut seperti `bid_id`, `bid_price`, `bid_posted`, `ads_id`, dan `bid_buyer_id` tidak bergantung pada subset atribut lainnya. Semua atribut independen dan tidak tergantung pada atribut lain. selain itu tabel `Bids` juga tidak menunjukkan adanya dependensi transitif. Dependensi transitif terjadi ketika atribut non-kunci bergantung pada atribut non-kunci melalui atribut lain dalam tabel. Namun, dalam tabel ini, tidak ada atribut non-kunci yang bergantung pada atribut non-kunci melalui atribut lainnya.

Menentukan Hubungan Antar Tabel

Langkah selanjutnya setelah penentuan tabel dan atribut yaitu akan ditentukan hubungan antara tabel-tabel yang telah dibuat. Ini akan dijelaskan dengan tipe hubungan (*one-to-one*, *one-to-many*, atau *many-to-many*) dan kunci yang sesuai.

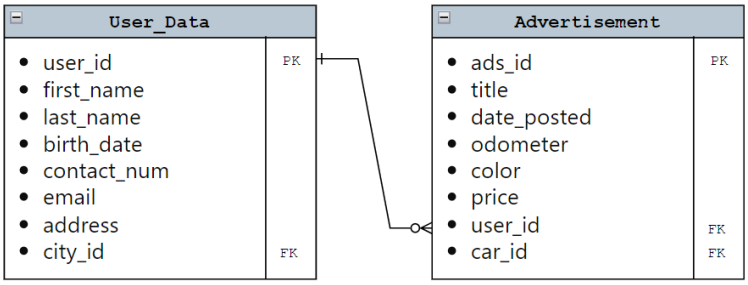
	City	User_Data	Car	Advertisement	Bids
City		*1 to many*			
User_Data	*many to 1*			*1 to many*	*1 to many*
Car				*1 to many*	
Advertisement		*many to 1*	*many to 1*		*1 to many*
Bids		*many to 1*		*many to 1*	

A. Tabel : `City - User_Data`



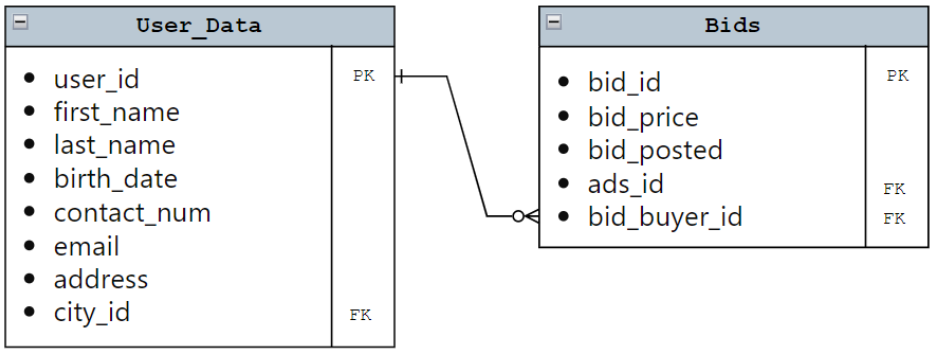
Setiap kota (`City`) dapat memiliki banyak pengguna (`User_Data`), tetapi setiap pengguna hanya terkait dengan satu kota tertentu saja.

B. Tabel : `User_Data - Advertisement`



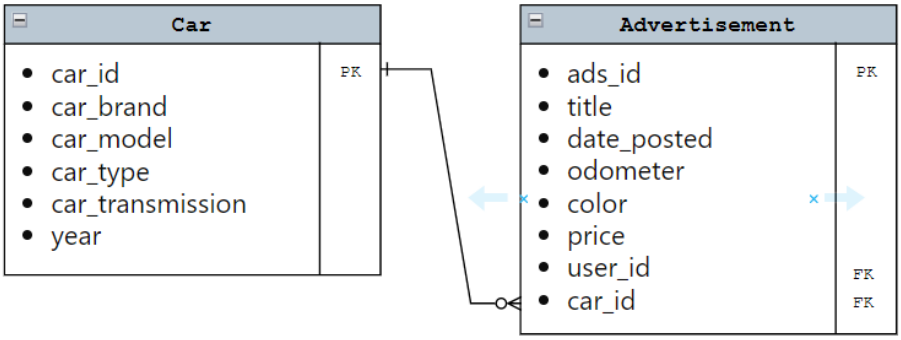
Setiap pengguna (`User_Data`) dapat memposting banyak iklan (`Advertisement`), tetapi setiap iklan diposting hanya terkait dengan satu pengguna tertentu.

C. **Tabel :** `User_Data - Bids`



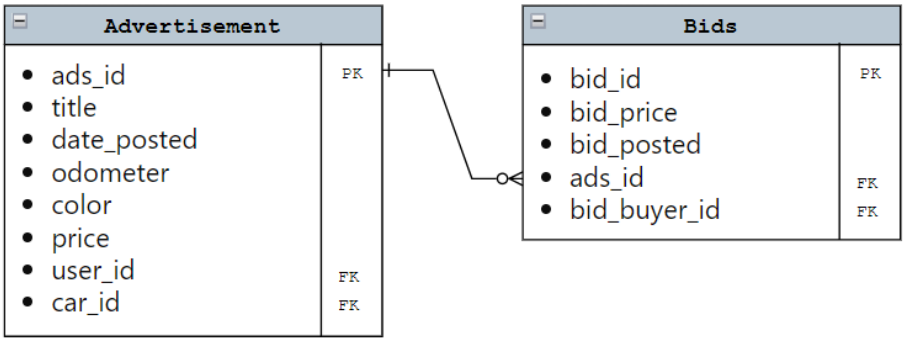
Setiap pembeli (`User_Data`) dapat membuat banyak penawaran harga (`Bids`), tetapi setiap penawaran harga hanya terkait dengan satu pembeli tertentu.

D. **Tabel :** `Car - Advertisement`



Setiap mobil (`Car`) dapat diiklankan dalam banyak iklan (`Advertisement`), tetapi setiap iklan hanya mempromosikan satu mobil.

E. **Tabel :** `Advertisement - Bids`



Setiap iklan (`Advertisement`) dapat menerima banyak penawaran harga dari pembeli (`Bids`), tetapi setiap penawaran harga oleh pembeli hanya diajukan untuk satu iklan.

Aturan Bisnis

Aturan bisnis (*Business Rules*) adalah pilar fundamental dalam desain database yang memastikan konsistensi, keakuratan, dan kepatuhan data terhadap kebijakan bisnis yang telah ditetapkan. Dengan memiliki aturan bisnis yang terdefinisi dengan jelas, pengelola basis data dapat menetapkan batasan dan pedoman yang mengatur penyusunan, penyimpanan, dan pengolahan data. Hal ini menjadi penting karena dapat mencegah terjadinya kesalahan atau inkonsistensi data yang berpotensi memengaruhi pengambilan keputusan bisnis serta kinerja operasional secara keseluruhan..

Tabel `City`

Business Rule	Constraint
1. **Ketentuan Identifikasi Unik: Setiap kota harus memiliki nomor identifikasi unik yang disimpan dalam atribut `city_id`. Nomor ini harus unik dan tidak boleh sama dengan `city_id` lain.** 2. **Ketentuan Nama Kota: Setiap kota harus memiliki nama yang dinyatakan dalam atribut `city_name`. Nama kota harus unik.** 3. **Informasi Geografis: Setiap kota dapat memiliki informasi geografis berupa koordinat lintang dan bujur yang disimpan dalam atribut `latitude` dan `longitude`** 4. **Hubungan dengan Tabel `User_Data`:** Atribut `city_id` di tabel `User_Data` merupakan *foreign key* yang menghubungkan pengguna dengan kota tempat tinggal mereka. Setiap nilai `city_id` harus merujuk ke nilai yang valid pada atribut `city_id` di tabel `City`. 5. **Tipe Partisipasi Opsional : setiap entitas dalam tabel `City` tidak diwajibkan untuk memiliki setidaknya satu entri yang terkait dalam tabel `User_Data`. Dalam konteks ini, partisipasi opsional akan mengizinkan beberapa kota untuk tidak memiliki pengguna yang tinggal di dalamnya** 6. **Parameter Tindakan:** Ketika ada upaya penghapusan atau pembaruan pada entri kota dalam tabel `City`, parameter **RESTRICT** memastikan bahwa tindakan tersebut dibatasi atau dicegah jika ada entri terkait dalam tabel `User_Data`	**NOT NULL** untuk field : city_id city_name latitude longitude Parameter relasi untuk tabel `City` – `User_Date` **ON DELETE RESTRICT** **ON UPDATE RESTRICT;**

Tabel User_Data

Business Rule	Constraint
1. **Ketentuan Identifikasi Unik: Setiap pengguna harus memiliki nomor identifikasi unik yang disimpan dalam atribut `user_id`** 2. **Ketentuan Nama Pengguna: Setiap pengguna harus memiliki nama depan dan nama belakang yang disimpan dalam atribut `first_name` dan `last_name`. Nama depan dan nama belakang tidak harus unik, karena beberapa pengguna dapat memiliki nama yang sama.** 3. **Informasi Tanggal Lahir: Setiap pengguna harus memiliki tanggal lahir yang disimpan dalam atribut `birth_date`. Format tanggal lahir *Date*** 4. **Nomor Kontak Unik: Setiap pengguna harus memiliki nomor kontak atau telepon unik yang disimpan dalam atribut `contact_num`.** 5. **Alamat Email Unik: Setiap pengguna harus memiliki alamat email yang unik, disimpan dalam atribut `email`.** 6. **Alamat Lengkap Tempat Tinggal: Setiap pengguna harus memiliki alamat lengkap tempat tinggal yang disimpan dalam atribut `address`. Alamat ini harus mencakup detail seperti nama jalan, nomor rumah, dan informasi tambahan lainnya.** 7. **ID Kota Tempat Tinggal Pengguna: Setiap pengguna harus terkait dengan kota tempat tinggal mereka melalui atribut `city_id` (FK) yang merujuk ke ID kota pada tabel `City`** 8. **Tipe Partisipasi Mandatory: Setiap pengguna dapat memposting banyak iklan (`Advertisement`), tetapi setiap iklan diposting hanya terkait dengan satu pengguna tertentu.** 9. **Tipe Partisipasi Mandatory: Setiap pembeli dapat membuat banyak penawaran harga (`Bids`), tetapi setiap penawaran harga hanya terkait dengan satu pembeli tertentu.** 10. **Parameter Tindakan: RESTRICT:** Ketika ada upaya penghapusan atau pembaruan pada entri pengguna dalam tabel `User_Data`, parameter **RESTRICT** memastikan bahwa tindakan tersebut dibatasi atau dicegah jika ada entri terkait dalam tabel `Advertisement` atau `Bids`. Hal ini membantu menjaga integritas referensial antara tabel-tabel yang terkait.	**NOT NULL** untuk field : user_id first_name last_name birth_date contact_num email address Parameter relasi untuk tabel `User_Data – City – Advertisement – Bids` **ON DELETE RESTRICT** **ON UPDATE RESTRICT;**

Tabel `Car`

Business Rule	Constraint
1. **Ketentuan Identifikasi Unik: Setiap mobil harus memiliki nomor identifikasi unik yang disimpan dalam atribut `car_id`.** 2. **Merek atau Produsen Mobil: Setiap mobil harus memiliki informasi tentang merek atau produsen yang disimpan dalam atribut `car_brand`.** 3. **Model Spesifik Mobil: Setiap mobil harus memiliki model spesifik yang disimpan dalam atribut `car_model`.** 4. **Jenis Bodi atau Kategori Mobil: Setiap mobil harus memiliki informasi tentang jenis bodi atau kategori yang disimpan dalam atribut `car_type`.** 5. **Jenis Transmisi Mobil: Setiap mobil harus memiliki informasi tentang jenis transmisi yang disimpan dalam atribut `car_transmission`.** 6. **Tahun Pembuatan atau Perakitan Mobil: Setiap mobil harus memiliki informasi tentang tahun pembuatan atau perakitan yang disimpan dalam `atribut year`.** 7. **Hubungan dengan Tabel Advertisement: Atribut `car_id` di tabel `Car` menghubungkan mobil dengan iklan yang terkait dalam tabel `Advertisement`.** 8. **Parameter Tindakan:** Ketika ada upaya penghapusan atau pembaruan pada entri mobil dalam tabel `Car`, parameter **RESTRICT** memastikan bahwa tindakan tersebut dibatasi atau dicegah jika ada entri terkait dalam tabel `Advertisement`.	**NOT NULL** untuk field : car_id car_brand car_model car_type car_transmission year Parameter relasi untuk tabel `Car - Advertisement` **ON DELETE RESTRICT** **ON UPDATE RESTRICT;**

Tabel `Advertisement`

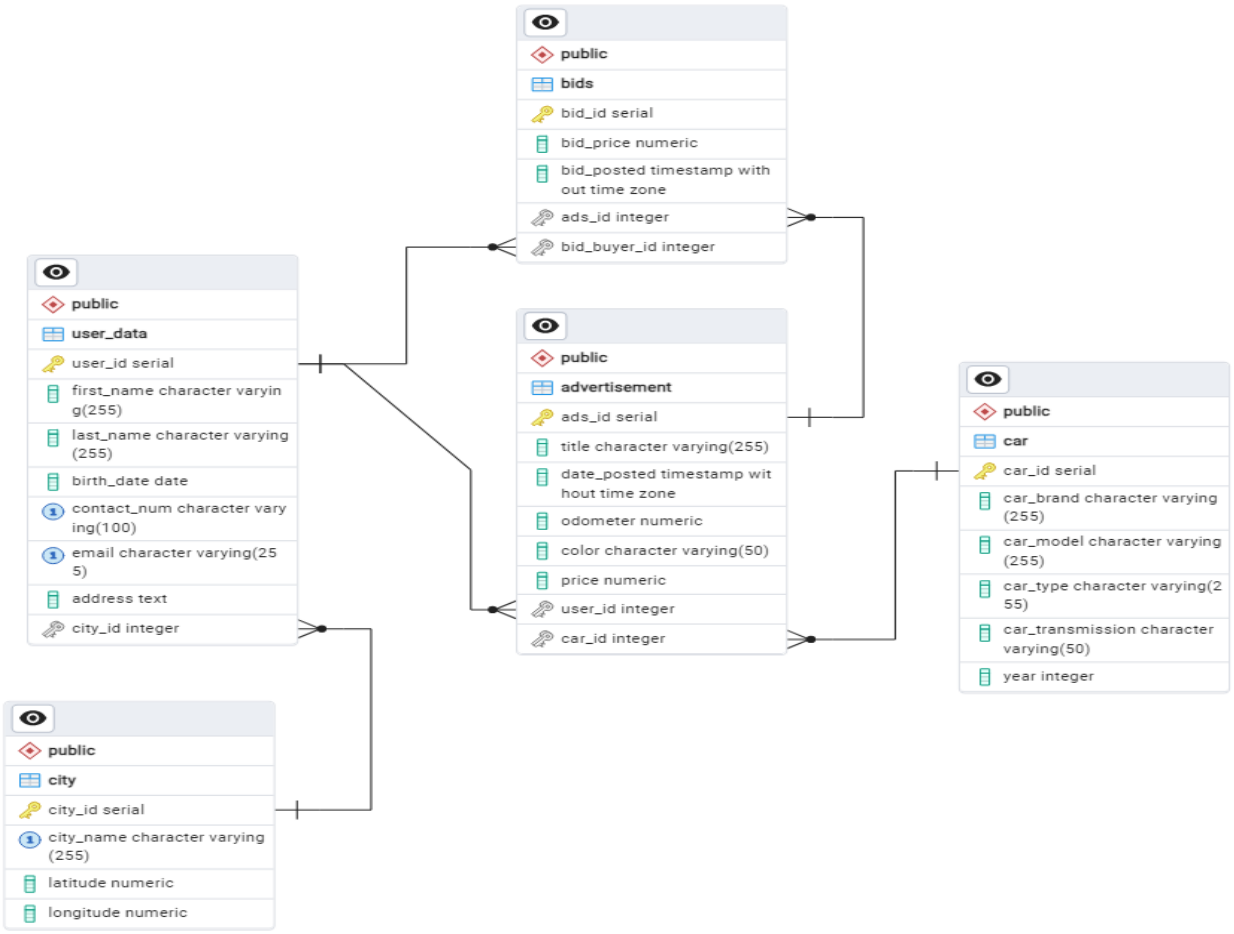
Business Rule	Constraint
1. **Ketentuan Identifikasi Unik: Setiap iklan harus memiliki nomor identifikasi unik yang disimpan dalam atribut `ads_id`.**	**NOT NULL** untuk field :
2. **Judul atau Deskripsi Singkat Iklan: Setiap iklan harus memiliki judul atau deskripsi singkat yang menjelaskan konten iklan yang disimpan dalam atribut `title`.**	ads_id
3. **Periode Tanggal dan Waktu Iklan Diposting: Setiap iklan harus memiliki informasi tentang tanggal dan waktu iklan diposting yang disimpan dalam atribut `date_posted`.**	title
4. **Kilometer yang Telah Ditempuh oleh Mobil: Setiap iklan harus mencakup informasi tentang jumlah kilometer yang telah ditempuh oleh mobil yang diiklankan yang disimpan dalam atribut `odometer`.**	date_posted
5. **Warna Mobil yang Diiklankan: Setiap iklan mencakup informasi tentang warna mobil yang disimpan dalam atribut `color`.**	odometer
6. **Harga Mobil yang Diiklankan: Setiap iklan harus mencakup informasi tentang harga mobil yang diiklankan yang disimpan dalam `price`.**	color
	price
	Parameter relasi untuk tabel `Advertisement – User_Data – Car – Bids` adalah :
	ON DELETE RESTRICT
	ON UPDATE RESTRICT;

Tabel `Bids`

Business Rule	Constraint
1. **Ketentuan Identifikasi Unik: Setiap penawaran harga oleh pembeli harus memiliki nomor identifikasi unik yang disimpan dalam `bid_id`.**	**NOT NULL** untuk field :
2. **Keterkaitan dengan Tabel Advertisement: Setiap penawaran harga yang dilakukan pembeli harus terkait dengan satu iklan tertentu.**	bid_id
3. **Keterkaitan dengan Tabel User_Data (Pembeli): Setiap penawaran harga oleh pembeli harus terkait dengan satu pembeli tertentu.**	bid_price
4. **Ketentuan Harga Penawaran: Atribut `bid_price` harus menyimpan nilai harga yang ditawarkan dalam penawaran dengan limitasi nilai > 0**	bid_posted
5. Periode Penawaran Diposting: Atribut `bid_posted` harus menyimpan informasi tentang periode tanggal dan waktu penawaran diposting.	Parameter relasi untuk tabel `Bids – User_Data – Advertisement` adalah
	ON DELETE RESTRICT
	ON UPDATE RESTRICT;

Implementasi Design

Hasil dari desain database ini akan berupa Diagram Hubungan Entitas (ERD). Setelah ERD dibuat, langkah selanjutnya adalah menerapkan hasil ERD ke dalam database menggunakan PostgreSQL dan Bahasa Definisi Data (DDL). Dengan demikian, proses merancang database telah memenuhi kebutuhan aplikasi *e-library* dan menyediakan solusi yang efektif untuk manajemen buku, penahanan, dan peminjaman dalam ekosistem *e-library*.



Data Definition Language (DDL)

Setelah ERD untuk *Car Marketplace Platform* dibuat, langkah selanjutnya adalah menerapkan hasil ERD tersebut ke dalam database menggunakan *PostgreSQL* dan Bahasa Definisi Data (DDL). Dengan demikian, proses perancangan database telah memenuhi kebutuhan aplikasi platform pasar mobil dan menyediakan solusi yang efektif untuk manajemen mobil, iklan, penawaran, dan interaksi antara pengguna dalam ekosistem platform tersebut.

```
-- Membuat tabel City
CREATE TABLE City (
    city_id SERIAL PRIMARY KEY,
    city_name VARCHAR(255) UNIQUE NOT NULL,
    latitude DECIMAL,
    longitude DECIMAL
);

-- Membuat tabel User_Data
CREATE TABLE User_Data (
    user_id SERIAL PRIMARY KEY,
    first_name VARCHAR(255),
    last_name VARCHAR(255),
    birth_date DATE,
    contact_num VARCHAR(100) UNIQUE NOT NULL,
    email VARCHAR(255) UNIQUE NOT NULL,
    address TEXT NOT NULL,
    city_id INTEGER NOT NULL,
    FOREIGN KEY (city_id) REFERENCES City(city_id)
);

-- Membuat tabel Car
CREATE TABLE Car (
    car_id SERIAL PRIMARY KEY,
    car_brand VARCHAR(255),
    car_model VARCHAR(255),
    car_type VARCHAR(255),
    car_transmission VARCHAR(50),
    year INTEGER
);
```

```
-- Membuat tabel Advertisement
CREATE TABLE Advertisement (
    ads_id SERIAL PRIMARY KEY,
    title VARCHAR(255),
    date_posted TIMESTAMP,
    odometer DECIMAL,
    color VARCHAR(50),
    price DECIMAL,
    user_id INTEGER NOT NULL,
    car_id INTEGER NOT NULL,
    FOREIGN KEY (user_id) REFERENCES User_Data(user_id),
    FOREIGN KEY (car_id) REFERENCES Car(car_id)
);

-- Membuat tabel Bids
CREATE TABLE Bids (
    bid_id SERIAL PRIMARY KEY,
    bid_price DECIMAL,
    bid_posted TIMESTAMP,
    ads_id INTEGER NOT NULL,
    bid_buyer_id INTEGER NOT NULL,
    FOREIGN KEY (ads_id) REFERENCES Advertisement(ads_id,
    FOREIGN KEY (bid_buyer_id) REFERENCES User_Data(user_id)
);
```

Populating Dummy Dataset

Setelah tahapan merancang desain basis data selesai dilakukan, langkah selanjutnya adalah menciptakan sebuah *dummy dataset* untuk masing-masing tabel yang telah direncanakan. Pada tahap ini, keseluruhan *dataset dummy* harus dipastikan sesuai dengan aturan dan logika relasional dari basis data yang telah dirancang sebelumnya. Hal ini penting agar dataset tersebut tidak hanya mengikuti struktur tabel yang telah dibuat, tetapi juga mencerminkan hubungan antartabel yang telah ditetapkan. Dengan demikian, *dataset dummy* tersebut menjadi representasi yang akurat dari cara data sebenarnya akan disimpan dan dihubungkan di dalam basis data yang telah direncanakan. Proses pembuatan dataset dummy ini memegang peranan penting dalam pengembangan basis data, karena akan membantu dalam menguji fungsionalitas basis data sebelum implementasi sebenarnya dilakukan. Sehingga, dengan melakukan tahapan ini secara cermat dan teliti, akan meminimalisir kemungkinan terjadinya kesalahan atau ketidakcocokan dalam struktur dan logika relasional basis data yang telah dirancang.

Create Dummy Dataset

Hasil dari *dummy* dataset ini akan disimpan dalam format CSV agar dapat dengan mudah diakses dan dimanipulasi. Setiap atribut dalam dataset ini akan sesuai dengan karakteristik yang telah dijelaskan sebelumnya. Proses pembuatan *dummy* data ini akan menggunakan bahasa pemrograman *Python*. Di awal, akan terlebih dahulu dilakukan definisi terhadap tipe data untuk setiap atribut sesuai dengan kebutuhan. Kemudian, langkah berikutnya adalah menghasilkan *dummy* data dengan memperhatikan aturan yang telah ditetapkan dalam desain database. Setelah *dummy* dataset dibuat, langkah selanjutnya adalah menyimpannya dalam format CSV. Ini memungkinkan dataset untuk dapat dengan mudah diimpor dan digunakan oleh sistem lainnya. Dengan demikian, *dummy* dataset ini akan menjadi sebuah representasi yang sesuai dengan kebutuhan dan dapat digunakan untuk pengembangan serta pengujian sistem lebih lanjut.

Tabel `City`

Atribut	Deskripsi Atribut	Kunci	
city_id	ID unik yang digunakan untuk mengidentifikasi setiap kota dalam database.	CK	**PK**
city_name	Nama dari setiap kota	CK	AK
latitude	Koordinat lintang geografis dari lokasi kota		
longitude	Koordinat bujur geografis dari lokasi kota		

```
import pandas as pd
from faker import Faker

# Inisialisasi Faker
fake = Faker('id_ID')

# Generate dummy data
num_cities = 30
cities_data = []

# Set untuk memeriksa keunikan nama kota
unique_city_names = set()

while len(cities_data) < num_cities:
    city_name = fake.city()
    if city_name not in unique_city_names:
        latitude = fake.latitude()
        longitude = fake.longitude()
        cities_data.append({
            'city_name': city_name,
            'latitude': latitude,
            'longitude': longitude
        })
        unique_city_names.add(city_name)

# Buat DataFrame dari data dummy
cities_df = pd.DataFrame(cities_data)
cities_df['city_id'] = range(1, num_cities + 1)

# Tambahkan city_id sebagai Primary Key
cities_df.set_index('city_id', inplace=True)

# Tampilkan DataFrame
cities_df
```

Tabel `User_Data`

Atribut	Deskripsi Atribut	Kunci	
user_id	Nomor identifikasi unik untuk setiap pengguna dalam database	CK	**PK**
first_name	Nama depan pengguna		
last_name	Nama akhir pengguna		
birth_date	Tanggal lahir pengguna		
contact_num	Nomor kontak pengguna	CK	AK
email	Alamat email pengguna	CK	AK
address	Alamat lengkap tempat tinggal pengguna		
city_id	Id kota tempat tinggal pengguna		FK

```
import pandas as pd
from faker import Faker
import random

# Inisialisasi Faker
fake = Faker('id_ID')

# Fungsi untuk generate email
def generate_email(first_name, last_name):
    return first_name.lower() + '.' + last_name.lower() + '@example.com'

# Generate dummy data
num_users = 200
users_data = []

# Ambil unique city_id dari cities_df
unique_city_ids = cities_df.index.tolist()

for i in range(num_users):
    # Generate unique first name and last name combination
    first_name = fake.first_name()
    last_name = fake.last_name()
    full_name = first_name + ' ' + last_name

    # Generate unique email
    email = generate_email(first_name, last_name)

    # Generate random city_id
    city_id = random.choice(unique_city_ids)

    # Generate random birth date
    birth_date = fake.date_of_birth(minimum_age=18, maximum_age=90)

    # Generate unique contact number
```

```
    contact_num = fake.phone_number()

    # Generate random address
    address = fake.street_address()

    users_data.append({
        'user_id': i + 1,
        'first_name': first_name,
        'last_name': last_name,
        'birth_date': birth_date,
        'contact_num': contact_num,
        'email': email,
        'address': address,
        'city_id': city_id
    })

# Buat DataFrame dari data dummy
users_df = pd.DataFrame(users_data)

# Tampilkan DataFrame
users_df
```

Tabel `Car`

Atribut	Deskripsi Atribut	Kunci	
car_id	Nomor identifikasi unik untuk setiap mobil dalam database	CK	**PK**
car_brand	Merek atau produsen mobil		
car_model	Model spesifik dari mobil		
car_type	Jenis bodi atau kategori mobil		
car_transmission	Jenis transmisi mobil		
year	Tahun pembuatan atau perakitan mobil		

```
import pandas as pd
import random

# Generate unique car_id
car_ids = [i + 1 for i in range(20)]

# Generate 20 different car brands
car_brands = ['Toyota', 'Honda', 'Nissan', 'Mitsubishi', 'Suzuki',
              'Ford', 'Chevrolet', 'Hyundai', 'Kia', 'BMW',
              'Mercedes-Benz', 'Audi', 'Lexus', 'Volvo', 'Volkswagen',
              'Ferrari', 'Lamborghini', 'Porsche', 'Tesla', 'Jaguar']
```

```
# Generate 20 different car models
car_models = ['Corolla', 'Civic', 'Altima', 'Outlander', 'Swift',
              'Fiesta', 'Cruze', 'Elantra', 'Optima', '3 Series',
              'E-Class', 'A4', 'RX', 'S60', 'Golf', '488 GTB',
              'Aventador', '911', 'Model S', 'F-Type']

# Generate 20 different car types
car_types = ['Sedan', 'Hatchback', 'SUV', 'MPV', 'Coupe',
             'Convertible', 'Pickup', 'Van', 'Wagon', 'Crossover',
             'Luxury', 'Sport', 'Compact', 'Electric', 'Hybrid',
             'Minivan', 'Truck', 'Roadster', 'Supercar', 'Limousine']

# Generate random car transmission types (True or False)
car_transmissions = [random.choice([True, False]) for _ in range(20)]

# Generate random car years from 1980 to 2021
car_years = [random.randint(1980, 2021) for _ in range(20)]

# Create dictionary for dummy data
cars_data = {
    'car_id': car_ids,
    'car_brand': car_brands,
    'car_model': car_models,
    'car_type': car_types,
    'car_transmission': car_transmissions,
    'year': car_years
}

# Create DataFrame from dummy data
cars_df = pd.DataFrame(cars_data)

# Tampilkan DataFrame
cars_df
```

Tabel **Advertisement**

Atribut	Deskripsi Atribut	Kunci	
ads_id	Nomor identifikasi unik untuk setiap iklan dalam database.	CK	**PK**
title	Menyimpan judul atau deskripsi singkat dari iklan yang diposting		
date_posted	Periode tanggal dan waktu iklan diposting		
odometer	Kilometer yang telah ditempuh oleh mobil yang diiklankan		
color	Warna mobil yang diiklankan		
price	Harga mobil yang diiklankan		
user_id	Nomor identifikasi pengguna yang memposting iklan		FK
car_id	Nomor identifikasi mobil yang diiklankan		FK

```
import pandas as pd
import random
from faker import Faker
import datetime

# Generate unique ads_id
ads_ids = [i + 1 for i in range(126)]

# Generate short descriptions for ads
fake = Faker('id_ID')
titles = [fake.sentence(nb_words=6) for _ in range(126)]

# Generate random date_posted
start_date = datetime.datetime(2023, 1, 1)
end_date = datetime.datetime.now()
date_posted = [fake.date_time_between(start_date=start_date,
end_date=end_date) for _ in range(126)]

# Generate random odometer values
odometer = [random.randint(100, 100000) for _ in range(126)]

# Generate random colors for ads
colors = ['Black', 'White', 'Silver', 'Red', 'Blue', 'Gray', 'Green',
'Yellow', 'Orange',
          'Purple', 'Brown', 'Gold', 'Pink', 'Beige', 'Turquoise', 'Maroon',
'Navy']
color = [random.choice(colors) for _ in range(126)]

# Generate random prices for ads
price = [random.randint(100000000, 150000000) for _ in range(126)]

# Generate random user_ids from users_df
```

```
user_ids = random.choices(users_df['user_id'], k=126)

# Generate random car_ids from cars_df
car_ids = random.choices(cars_df['car_id'], k=126)

# Create dictionary for dummy data
ads_data = {
    'ads_id': ads_ids,
    'title': titles,
    'date_posted': date_posted,
    'odometer': odometer,
    'color': color,
    'price': price,
    'user_id': user_ids,
    'car_id': car_ids
}

# Create DataFrame from dummy data
ads_df = pd.DataFrame(ads_data)

# Tampilkan DataFrame
ads_df
```

Tabel **Bids**

Atribut	Deskripsi Atribut	Kunci	
bid_id	Nomor identifikasi unik untuk setiap penawaran dalam database	CK	**PK**
bid_price	Harga yang ditawarkan dalam suatu penawaran		
bid_posted	Periode tanggal dan waktu penawaran diposting		
ads_id	Nomor identifikasi unik untuk setiap iklan yang diberi penawaran		FK
bid_buyer_id	Nomor identifikasi pembeli yang mengajukan penawaran		FK

```
import pandas as pd
import random
from faker import Faker
import datetime

# Generate unique bid_id
bid_ids = [i + 1 for i in range(80)]

# Generate random bid prices
```

```
bid_prices = [random.randint(80000000, 99000000) for _ in range(80)]

# Generate random bid_posted dates
start_date = datetime.datetime(2023, 1, 1)
end_date = datetime.datetime.now()
bid_posted = [fake.date_time_between(start_date=start_date,
end_date=end_date) for _ in range(80)]

# Generate random ads_id from ads_df
ads_id = random.choices(ads_df['ads_id'], k=80)

# Generate random bid_buyer_id from users_df
bid_buyer_id = random.choices(users_df['user_id'], k=80)

# Create dictionary for dummy data
bids_data = {
    'bid_id': bid_ids,
    'bid_price': bid_prices,
    'bid_posted': bid_posted,
    'ads_id': ads_id,
    'bid_buyer_id': bid_buyer_id
}

# Create DataFrame from dummy data
bids_df = pd.DataFrame(bids_data)

# Tampilkan DataFrame
bids_df
```

Input Dummy Dataset Into The Database

Setelah dataset CSV dibuat, langkah selanjutnya adalah memuatnya ke dalam basis data. Untuk melakukan hal ini, perlu menggunakan perintah SQL yang sesuai dengan basis data yang digunakan.

```
-- import dummy data city
COPY
      city
FROM 'D:\FINAL PROJECT\cities_df.csv'
DELIMITER ','
CSV
HEADER

-- import dummy data User_Data
COPY
      user_data
FROM 'D:\FINAL PROJECT\users_df.csv'
DELIMITER ','
CSV
HEADER

-- import dummy data Car
COPY
      car
FROM 'D:\FINAL PROJECT\cars_df.csv'
DELIMITER ','
CSV
HEADER

-- import dummy data Advertisement
COPY
      advertisement
FROM 'D:\FINAL PROJECT\ads_df.csv'
DELIMITER ','
CSV
HEADER

-- import dummy data bids
```

```
COPY
      bids
FROM 'D:\FINAL PROJECT\bids_df.csv'
DELIMITER ','
CSV
HEADER
```

Creating Transactional Query

Setelah database yang dibuat memiliki record data, akan dilakukan query berdasarkan 5 perintah *transactional query*, sebagai berikut :

1. **Mencari mobil keluaran 2015 ke atas**

Query :

```
SELECT *
FROM Car
WHERE year >= 2015;
```

Output :

	car_id [PK] integer	car_brand character varying (255)	car_model character varying (255)	car_type character varying (255)	car_transmission character varying (50)	year integer
1	7	Chevrolet	Cruze	Pickup	False	2017
2	8	Hyundai	Elantra	Van	False	2015
3	13	Lexus	RX	Compact	False	2018
4	14	Volvo	S60	Electric	False	2020

2. **Menambahkan satu data bid produk baru**

Query :

```
INSERT INTO Bids (bid_id,bid_price, bid_posted, ads_id,
bid_buyer_id)
VALUES (81, 10000000, '2024-05-05 12:00:00', 1, 123);
```

Output :

75	75	82195774	2023-11-28 15:09:34.573587	77	101
76	76	93111664	2023-06-30 12:44:25.747501	18	13
77	77	88918233	2023-02-08 11:04:09.351123	122	194
78	78	95185934	2023-11-28 11:04:36.714353	87	13
79	79	91144836	2023-02-24 11:45:02.545939	85	177
80	80	80312603	2024-01-07 17:52:16.043091	1	20
81	81	10000000	2024-05-05 12:00:00	1	123

3. Melihat semua mobil yg dijual 1 akun dari yg paling baru

Query :

```
SELECT    a.ads_id,    a.title,    a.date_posted,    c.car_brand,
c.car_model, c.year
FROM Advertisement a
JOIN Car c ON a.car_id = c.car_id
WHERE a.user_id = 179
ORDER BY a.date_posted DESC;
```

Output :

	ads_id integer	title character varying (255)	date_posted timestamp without time zone	car_brand character varying (255)	car_model character varying (255)	year integer
1	81	Nobis mollitia ea similique corrupti.	2024-03-16 21:40:59.44981	Kia	Optima	2010
2	101	Modi repudiandae a dignissimos amet.	2023-12-27 07:32:23.163639	Honda	Civic	2008
3	90	Error quaerat velit eum aperiam amet ducimus.	2023-11-07 01:47:02.447685	Ford	Fiesta	1994
4	84	Expedita cupiditate porro illo iusto aspernatur aspernat...	2023-08-13 08:20:05.794013	Porsche	911	1981

4. Mencari mobil bekas yang termurah berdasarkan keyword

Query :

```
SELECT
    Car.car_id,
    Car.car_brand,
    Car.car_model,
    Advertisement.price
FROM
    Car
JOIN
    Advertisement ON Car.car_id = Advertisement.car_id
WHERE
    Car.car_brand = 'Honda'
ORDER BY
    Advertisement.price ASC
```

Output :

	car_id integer	car_brand character varying (255)	car_model character varying (255)	price numeric
1	2	Honda	Civic	105357185
2	2	Honda	Civic	107245144
3	2	Honda	Civic	108933364
4	2	Honda	Civic	146816622

5. Mencari mobil bekas yang terdekat berdasarkan sebuah id kota, jarak terdekat dihitung berdasarkan latitude longitude. Perhitungan jarak dapat dihitung menggunakan rumus jarak euclidean berdasarkan latitude dan longitude. City_id = 2

Query :

```
SELECT
    Advertisement.ads_id,
    Advertisement.title,
    Advertisement.price,
    City.city_id,
    City.city_name,
    SQRT(POW((City.latitude - City_Target.latitude), 2) +
POW((City.longitude - City_Target.longitude), 2)) AS distance
FROM
    Advertisement
JOIN
    User_Data ON Advertisement.user_id = User_Data.user_id
JOIN
    City ON User_Data.city_id = City.city_id
CROSS JOIN
    (SELECT latitude, longitude FROM City WHERE city_id = 3173)
AS City_Target
ORDER BY
    distance ASC;
```

Output :

	ads_id integer	price numeric	city_id integer	city_name character varying (255)	distance numeric
1	124	104206436	1	Tasikmalaya	0.0000000000000000
2	22	138324860	1	Tasikmalaya	0.0000000000000000
3	55	140018722	1	Tasikmalaya	0.0000000000000000
4	52	119670579	20	Madiun	33.731725530117967
5	102	106865656	20	Madiun	33.731725530117967
6	62	120647872	9	Tangerang	42.716967746088837
7	126	106874157	9	Tangerang	42.716967746088837

Creating Analytical Query

1. **Ranking popularitas model mobil berdasarkan jumlah bid**

Query :

```
  SELECT
    Car.car_model,
    COUNT(Bids.bid_id) AS total_bids
FROM
    Car
JOIN
    Advertisement ON Car.car_id = Advertisement.car_id
JOIN
    Bids ON Advertisement.ads_id = Bids.ads_id
GROUP BY
    Car.car_model
ORDER BY
    total_bids DESC;
```

Output :

	car_model character varying (255)	total_bids bigint
1	Corolla	12
2	Swift	11
3	Cruze	7
4	488 GTB	5
5	Optima	5
6	911	5
7	3 Series	4
8	S60	4
9	Fiesta	4
10	Golf	4
11	Elantra	4
12	Outlander	4
13	E-Class	3
14	Aventador	2

2. **Membandingkan harga mobil berdasarkan harga rata-rata per kota**

Query :

```
SELECT
    c.city_name,
    cr.car_brand,
    cr.car_model,
    cr.year,
    a.price AS price,
    AVG(a.price) OVER (PARTITION BY c.city_id) AS avg_car_city
FROM
    Advertisement a
JOIN
    User_Data u ON a.user_id = u.user_id
JOIN
```

```
    City c ON u.city_id = c.city_id
JOIN
    Car cr ON a.car_id = cr.car_id
ORDER BY
    c.city_name, cr.car_brand, cr.car_model, cr.year;
```

Output :

	city_name character varying (255)	car_brand character varying (255)	car_model character varying (255)	year integer	price numeric	avg_car_city numeric
1	Balikpapan	Ford	Fiesta	1994	141246596	132865116.00000000
2	Balikpapan	Honda	Civic	2008	146816622	132865116.00000000
3	Balikpapan	Honda	Civic	2008	105357185	132865116.00000000
4	Balikpapan	Kia	Optima	2010	122027168	132865116.00000000
5	Balikpapan	Lexus	RX	2018	115681827	132865116.00000000
6	Balikpapan	Porsche	911	1981	149531434	132865116.00000000
7	Balikpapan	Porsche	911	1981	149394980	132865116.00000000
8	Banjarmasin	Lamborghini	Aventador	1998	121839284	120341402.250000000000
9	Banjarmasin	Suzuki	Swift	1980	132437793	120341402.250000000000
10	Banjarmasin	Tesla	Model S	2003	111551317	120341402.250000000000
11	Banjarmasin	Toyota	Corolla	1993	115537215	120341402.250000000000
12	Batam	Kia	Optima	2010	126935270	130377319.40000000

3. **Dari penawaran suatu model mobil, cari perbandingan tanggal user melakukan bid dengan bid selanjutnya beserta harga tawar yang diberikan**

Query :

```
WITH FirstBids AS (
    SELECT
        a.ads_id,
        a.user_id,
        b.bid_id AS first_bid_id,
        b.bid_price AS first_bid_price,
        b.bid_posted AS first_bid_date,
        ROW_NUMBER() OVER (PARTITION BY a.ads_id ORDER BY
b.bid_posted) AS bid_order
    FROM
        Advertisement a
    JOIN
        Bids b ON a.ads_id = b.ads_id
),
NextBids AS (
    SELECT
        a.ads_id,
        a.user_id,
        b.bid_id AS next_bid_id,
        b.bid_price AS next_bid_price,
        b.bid_posted AS next_bid_date,
        ROW_NUMBER() OVER (PARTITION BY a.ads_id ORDER BY
b.bid_posted) AS bid_order
```

```
    FROM
        Advertisement a
    JOIN
        Bids b ON a.ads_id = b.ads_id
)
SELECT
    cr.car_brand,
    cr.car_model,
    fb.user_id,
    fb.first_bid_date,
    nb.next_bid_date,
    fb.first_bid_price,
    nb.next_bid_price
FROM
    FirstBids fb
JOIN
    NextBids nb ON fb.ads_id = nb.ads_id AND fb.user_id =
nb.user_id AND fb.bid_order + 1 = nb.bid_order
JOIN
    Car cr ON cr.car_id = (SELECT car_id FROM Advertisement WHERE
ads_id = fb.ads_id)
ORDER BY
    cr.car_brand, cr.car_model, fb.user_id;
```

Output :

	car_brand character varying (255)	car_model character varying (255)	user_id integer	first_bid_date timestamp without time zone	next_bid_date timestamp without time zone	first_bid_price numeric	next_bid_price numeric
1	BMW	3 Series	51	2023-03-04 05:42:41.530314	2023-04-13 22:53:38.213546	84941912	83287512
2	BMW	3 Series	51	2023-04-13 22:53:38.213546	2023-07-05 11:52:15.076738	83287512	90552660
3	Chevrolet	Cruze	153	2023-07-02 00:52:17.374861	2024-02-20 16:34:39.77762	80631417	97785205
4	Chevrolet	Cruze	166	2023-01-29 05:16:27.615873	2023-03-18 15:32:55.597496	97588864	82784557
5	Chevrolet	Cruze	166	2023-03-18 15:32:55.597496	2023-03-19 18:47:16.030454	82784557	86269057
6	Ferrari	488 GTB	185	2023-02-24 11:45:02.545939	2023-08-22 17:25:53.029037	91144836	87682011
7	Ferrari	488 GTB	197	2023-03-09 22:24:46.710202	2023-12-11 01:16:05.132126	82499503	91153812
8	Ford	Fiesta	168	2024-01-07 17:52:16.043091	2024-05-05 12:00:00	80312603	10000000
9	Kia	Optima	21	2023-11-01 03:14:57.330885	2024-05-06 02:14:42.31701	81344215	88484295
10	Kia	Optima	155	2023-01-31 02:19:01.641664	2023-03-27 09:31:58.613809	95450285	81967721
11	Kia	Optima	155	2023-03-27 09:31:58.613809	2023-08-17 13:40:56.578306	81967721	97447827

4. **Membandingkan persentase perbedaan rata-rata harga mobil berdasarkan modelnya dan rata-rata harga bid yang ditawarkan oleh customer pada 6 bulan terakhir (4% bobot)**
   a. Difference adalah selisih antara rata-rata harga model mobil(avg_price) dengan rata-rata harga bid yang ditawarkan terhadap model tersebut(avg_bid_6month)
   b. Difference dapat bernilai negatif atau positif
   c. Difference_percent adalah persentase dari selisih yang telah dihitung, yaitu dengan cara difference dibagi rata-rata harga model mobil(avg_price) dikali 100%
   d. Difference_percent dapat bernilai negatif atau positif

Query :

```
WITH CarAvgPrice AS (
    SELECT
        car_brand,
        car_model,
        AVG(price) AS avg_price
    FROM
        Car
    JOIN
        Advertisement ON Car.car_id = Advertisement.car_id
    GROUP BY
        car_brand, car_model
),
BidAvgPrice AS (
    SELECT
        car_brand,
        car_model,
        AVG(bid_price) AS avg_bid_6month
    FROM
        Car
    JOIN
        Advertisement ON Car.car_id = Advertisement.car_id
    JOIN
        Bids ON Advertisement.ads_id = Bids.ads_id
    WHERE
        Bids.bid_posted >= NOW() - INTERVAL '6 month'
    GROUP BY
        car_brand, car_model
)
SELECT
    c.car_brand,
    c.car_model,
    ROUND(cap.avg_price) AS avg_price,
    ROUND(bap.avg_bid_6month) AS avg_bid_6month,
    ROUND(cap.avg_price - bap.avg_bid_6month) AS difference,
    ROUND(((cap.avg_price - bap.avg_bid_6month) / cap.avg_price)
* 100, 4) AS difference_percent
FROM
```

```
    CarAvgPrice cap
JOIN
    BidAvgPrice bap ON cap.car_brand = bap.car_brand AND
cap.car_model = bap.car_model
JOIN
    Car c ON cap.car_brand = c.car_brand AND cap.car_model =
c.car_model;
```

Output :

	car_brand character varying (255)	car_model character varying (255)	user_id integer	first_bid_date timestamp without time zone	next_bid_date timestamp without time zone	first_bid_price numeric	next_bid_price numeric
1	BMW	3 Series	51	2023-03-04 05:42:41.530314	2023-04-13 22:53:38.213546	84941912	83287512
2	BMW	3 Series	51	2023-04-13 22:53:38.213546	2023-07-05 11:52:15.076738	83287512	90552660
3	Chevrolet	Cruze	153	2023-07-02 00:52:17.374861	2024-02-20 16:34:39.77762	80631417	97785205
4	Chevrolet	Cruze	166	2023-01-29 05:16:27.615873	2023-03-18 15:32:55.597496	97588864	82784557
5	Chevrolet	Cruze	166	2023-03-18 15:32:55.597496	2023-03-19 18:47:16.030454	82784557	86269057
6	Ferrari	488 GTB	185	2023-02-24 11:45:02.545939	2023-08-22 17:25:53.029037	91144836	87682011
7	Ferrari	488 GTB	197	2023-03-09 22:24:46.710202	2023-12-11 01:16:05.132126	82499503	91153812
8	Ford	Fiesta	168	2024-01-07 17:52:16.043091	2024-05-05 12:00:00	80312603	10000000
9	Kia	Optima	21	2023-11-01 03:14:57.330885	2024-05-06 02:14:42.31701	81344215	88484295
10	Kia	Optima	155	2023-01-31 02:19:01.641664	2023-03-27 09:31:58.613809	95450285	81967721
11	Kia	Optima	155	2023-03-27 09:31:58.613809	2023-08-17 13:40:56.578306	81967721	97447827

5. **Dari penawaran suatu model mobil, cari perbandingan tanggal user melakukan bid dengan bid selanjutnya beserta harga tawar yang diberikan**

Query :

```
WITH RankedBids AS (
    SELECT
        car_brand,
        car_model,
        bid_price,
        bid_posted,
        ROW_NUMBER() OVER (PARTITION BY car_brand, car_model
ORDER BY bid_posted DESC) AS rn
    FROM
        Advertisement
    JOIN
        Car ON Advertisement.car_id = Car.car_id
    JOIN
        Bids ON Advertisement.ads_id = Bids.ads_id
    WHERE
        bid_posted >= NOW() - INTERVAL '6 month'
)
SELECT
    car_brand,
    car_model,
    MAX(CASE WHEN rn = 6 THEN bid_price END) AS m_min_6,
    MAX(CASE WHEN rn = 5 THEN bid_price END) AS m_min_5,
```

```
    MAX(CASE WHEN rn = 4 THEN bid_price END) AS m_min_4,
    MAX(CASE WHEN rn = 3 THEN bid_price END) AS m_min_3,
    MAX(CASE WHEN rn = 2 THEN bid_price END) AS m_min_2,
    MAX(CASE WHEN rn = 1 THEN bid_price END) AS m_min_1
FROM
    RankedBids
GROUP BY
    car_brand, car_model;
```

Output :

	car_brand character varying (255)	car_model character varying (255)	m_min_6 numeric	m_min_5 numeric	m_min_4 numeric	m_min_3 numeric	m_min_2 numeric
1	Chevrolet	Cruze	[null]	[null]	[null]	[null]	84318652
2	Ferrari	488 GTB	[null]	[null]	[null]	[null]	[null]
3	Ford	Fiesta	[null]	[null]	80312603	80675230	86279121
4	Hyundai	Elantra	[null]	[null]	[null]	[null]	[null]
5	Kia	Optima	[null]	[null]	[null]	[null]	[null]
6	Lamborghini	Aventador	[null]	[null]	[null]	[null]	[null]
7	Lexus	RX	[null]	[null]	[null]	[null]	[null]
8	Mercedes-Benz	E-Class	[null]	[null]	[null]	[null]	[null]
9	Mitsubishi	Outlander	[null]	[null]	[null]	[null]	93847233
10	Porsche	911	[null]	[null]	[null]	97855787	87980421
11	Suzuki	Swift	[null]	[null]	[null]	95185934	94336404
12	Tesla	Model S	[null]	[null]	[null]	[null]	[null]
13	Toyota	Corolla	[null]	[null]	[null]	97655279	86169642
14	Volkswagen	Golf	[null]	[null]	[null]	81473274	87857572
15	Volvo	S60	[null]	[null]	[null]	[null]	[null]

Berikut adalah tautan ke presentasi proyek basis data relasional yang mencakup deskripsi tabel, atribut, serta hubungan antar tabel yang digunakan dalam aplikasi penjualan mobil bekas: Link Presentasi Proyek Basis Data Relasional. Anda juga dapat mengakses repositori GitHub proyek ini untuk melihat kode dan dokumentasi lengkap: Link GitHub Proyek Basis Data Relasional.