Database Management System

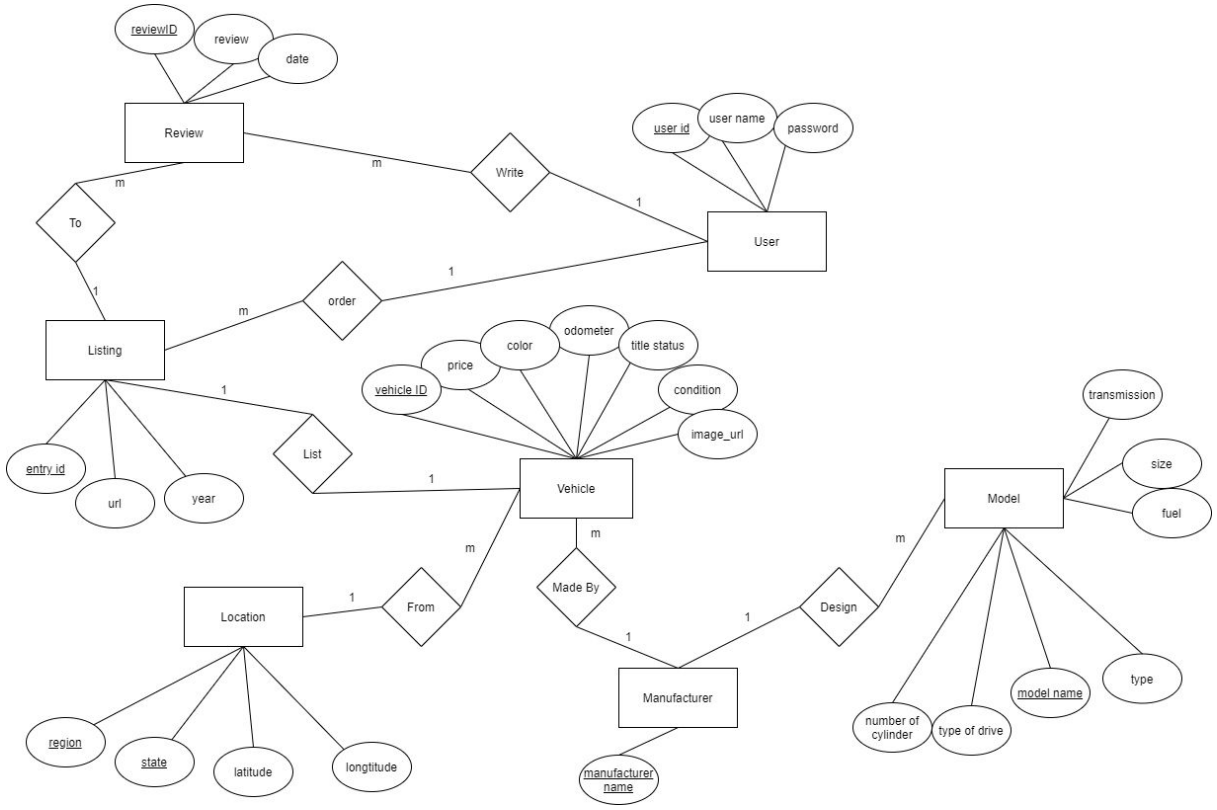
Project Name:
Used Cars Management

System Phase III: Database Schema Construction

Team member:Jiaxu Zhu, Haocheng Song, Jiajing Liao, Yufan Chen

Table of Content

Entity Relationship Diagram Design



In this section, we designed the ER schema diagram for our Used Car Management System to describe related data as entities, relationship and attributes. Here is the detailed description and explanation:

1) We create table User to store the information of our system users, they may have interest in our listed vehicles, so they need their username and password to login our system, and each user has an unique user ID for identification.
2) Users browse data from the system by looking at Listings can they can also make orders. The table Listing contains the url of each vehicle, the year each entry is posted and each entry in the list has an unique ID. A user can make as many orders as he wants, so the Cardinality is 1 : m in this relationship.
3) Users can also write reviews about a specific listing, each review is identified by its own ID in the management system, with its review content and posted date. A user can write several reviews, and a listing can be reviewed by several users too.
4) An entry in table Listing lists information about a specific vehicle, so the Cardinality is 1 : 1, and of course there should be an entity called Vehicle, which contains an ID, with some basic properties about a car like color, odometer, price, title, condition and some images. A vehicle is made by a manufacturer, which must have a name, and a car is from some location, which combines state and region.
5) For a manufacturer, it designs some car models, and a model also has some information, and uses model name as its key.

In our ER diagram, for each entity set, it must have a primary key. And for each relationship, the Cardinalities are also marked.

Relational Schema

1) **User**(**user_id: int**, username: varchar, password: varchar)
   User table has three attributes.
   user_id is primary key and it is increased automatically;
   username is user created that must be unique;
   password is also user created for authorization and login.
   Each user has to register an account before doing any other operations.

2) **Review**(**review_id:** int, review: varchar, date: DATETIME)
   Review table has three attributes.
   review_id is primary key and automatically increased;
   review means the actual content of this review, each review is given by an user;
   date is the time when this review was posted, it should be recorded like 'YYYY-MM-DD 00:00:00';
   Reviews are given by users, they show users' attitude towards lists.

3) **Listing**(**entry_id: int,** url: varchar, year: int)
   Listing table has three attributes.
   entry_id is primary key and automatically increased;
   url records the link of the related information about that vehicle. In table Listing, the detailed information is not contained;
   year is the year that this entry was posted on our management system.

4) **Vehicle**(**vehicle_ID: int**, price: int, color: varchar, odometer: varchar, title_status: varchar, condition: varchar, image_url: varchar)
   Vehicle table has seven attributes.
   vehicle_ID is primary key and automatically increased;
   price means the price of that vehicle;
   color is the color of the vehicle;
   odometer means the current milage of the car;
   title_status means the maintenance of each car (i.e. clean title, rebuilt);
   condition defines the current condition of the car (i.e. excellent);
   image_url: For each car, there are some pictures of if that can be browsed by users to get a deep understanding of the car.

5) **Location**(**region: varchar**, **state: varchar**, latitude: varchar, longitude: varchar)
   Location table has four attributes.
   The combination of ***region*** and ***state*** is primary key, a regino cannot uniquely identifies the location because in different states, there can be two different cities that have the same name;
   For each region, the ***latitude*** and ***longitude*** are also recorded to give a more detailed description of location.

6) **Manufacutrer**(**name**)
   Manufacturer table has one attribute.
   name is just the name of a specific manufacturer (i.e. Ford, BMW)

Each car is made by a manufacturer, and a manufacturer may design several models.

7) **Model(model_name: varchar**, number_of_cylinder: int, type_of_drive: varchar, type: varchar, transmission: varchar, size: int, fuel: varchar)
Mode table has seven attributes.
model_name is primary key, each model must has a unique model name in the real world;
number_of_cylinder is one of the properties for a specific model. Normally, the number of cylinders in a car can be 4~8;
type_of_drive: It defines whether a car in this model is four-wheel drive, front-drive or rear-drive;
type: It defines the body design of a model (i.e. pickup, SUV, sedan);
transmission defines whether a car in this model is manual or automatic;
size means the size of the cars in this model and internal space;
fuel: It illustrates what kind of fuel is required for this specific model.

SQL Tables

Below are the screenshots for initializing empty SQL tables corresponding to our relational schema.

```
create table "USER"(
        user_id integer,
        username varchar(50) UNIQUE,
        passaword varchar(50) NOT NULL,
        primary key (user_id)
);
```

	COLUMN_NAME	DATA_TYPE
1	USER_ID	NUMBER(38,0)
2	USERNAME	VARCHAR2(50 BYTE)
3	PASSAWORD	VARCHAR2(50 BYTE)

Figure 1. Create User Table with three attributes USER_ID,USERNAME and PASSWORD

```
create table LISTING(
        entry_id integer,
        "URL" varchar(200),
        "YEAR" integer,
        ordered_by integer,
        primary key (entry_id),
        foreign key (ordered_by) REFERENCES "USER"(user_id)
);
```

	COLUMN_NAME	DATA_TYPE
1	ENTRY_ID	NUMBER(38,0)
2	URL	VARCHAR2(200 BYTE)
3	YEAR	NUMBER(38,0)
4	ORDERED_BY	NUMBER(38,0)

Figure 2. Create Listing Table with four attributes ENTRY_ID, URL, YEAR and ORDERED_BY

```
create table REVIEW(
        review_id integer,
        review varchar(2000),
        "DATE" TIMESTAMP,
        written_by integer,
        to_listing integer,
        primary key (review_id),
        foreign key (written_by) REFERENCES "USER"(user_id),
        foreign key (to_listing) REFERENCES "LISTING"(entry_id)
);
```

	COLUMN_NAME	DATA_TYPE
1	REVIEW_ID	NUMBER(38,0)
2	REVIEW	VARCHAR2(2000 BYTE)
3	DATE	TIMESTAMP(6)
4	WRITTEN_BY	NUMBER(38,0)
5	TO_LISTING	NUMBER(38,0)

Figure 3. Create REVIEW Table with five attributes REVIEW_ID, REVIEW, DATE, WRITTEN_BY and TO_LISTING

```
create table "LOCATION"(
        region varchar(50),
        "STATE" varchar(50),
        latitude varchar(50),
        longitude varchar(50),
        primary key (region, "STATE")
);
```

	COLUMN_NAME	DATA_TYPE
1	REGION	VARCHAR2(50 BYTE)
2	STATE	VARCHAR2(50 BYTE)
3	LATITUDE	VARCHAR2(50 BYTE)
4	LONGITUDE	VARCHAR2(50 BYTE)

Figure 4. Create LOCATION Table with four attributes REGION, STATE, LATITUDE and LONGITUDE

```
create table MANUFACUTRER(
        manufacutrer_name varchar(50),
        primary key (manufacutrer_name)
);
```

COLUMN_NAME	DATA_TYPE
MANUFACUTRER_NAME	VARCHAR2(50 BYTE)

Figure 5. Create MANUFACTURER Table with single attribute MANUFACTURER_NAME

```
create table "MODEL"(
        model_name varchar(50),
        number_of_cylinder integer,
        type_of_drive varchar(50),
        "TYPE" varchar(50),
        transmission varchar(50),
        "SIZE" integer,
        fuel varchar(50),
        designed_by varchar(50),
        primary key (model_name),
        foreign key (designed_by) REFERENCES MANUFACUTRER(manufacutrer_name)
);
```

	COLUMN_NAME	DATA_TYPE
1	MODEL_NAME	VARCHAR2(50 BYTE)
2	NUMBER_OF_CYLINDER	NUMBER(38,0)
3	TYPE_OF_DRIVE	VARCHAR2(50 BYTE)
4	TYPE	VARCHAR2(50 BYTE)
5	TRANSMISSION	VARCHAR2(50 BYTE)
6	SIZE	NUMBER(38,0)
7	FUEL	VARCHAR2(50 BYTE)
8	DESIGNED_BY	VARCHAR2(50 BYTE)

Figure 6. Create MODEL Table with eight attributes MODEL_NAME, NUMBER_OF_CYLINDER, TYPE_OF_DRIVE, TYPE, TRANSMISSION, SIZE, FUEL and DESIGNE_BY

```
create table VEHICLE(
        vehicle_id integer,
        price integer,
        color varchar(50),
        odometer integer,
        title_status varchar(50),
        condition varchar(50),
        image_url varchar(200),
        listed_in integer,
        region varchar(50),
        "STATE" varchar(50),
        made_by varchar(50),

        primary key (vehicle_id),
        foreign key (listed_in) REFERENCES LISTING(entry_id),
        foreign key (region, "STATE") REFERENCES "LOCATION"(region, "STATE"),
        foreign key (made_by) REFERENCES MANUFACUTRER(manufacutrer_name)
);
```

	COLUMN_NAME	DATA_TYPE
1	VEHICLE_ID	NUMBER(38,0)
2	PRICE	NUMBER(38,0)
3	COLOR	VARCHAR2(50 BYTE)
4	ODOMETER	NUMBER(38,0)
5	TITLE_STATUS	VARCHAR2(50 BYTE)
6	CONDITION	VARCHAR2(50 BYTE)
7	IMAGE_URL	VARCHAR2(200 BYTE)
8	LISTED_IN	NUMBER(38,0)
9	REGION	VARCHAR2(50 BYTE)
10	STATE	VARCHAR2(50 BYTE)
11	MADE_BY	VARCHAR2(50 BYTE)

Figure 7. Create VEHICLE Table with eleven attributes VEHICLE_ID, PRICE, COLOR, ODOMETER, TITLE_STATUS, CONDITION, IMAGE_URL, LISTED_IN, REGION, STATE and MADE_BY