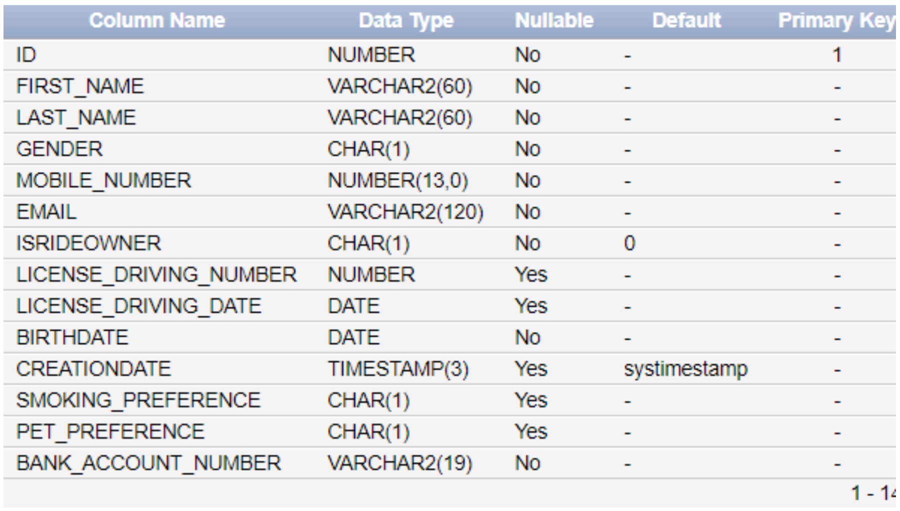
Here are the different relation schemas of our database:

Table MEMBER



Column Name	Data Type	Nullable	Default	Primary Key
ID	NUMBER	No	-	1
FIRST_NAME	VARCHAR2(60)	No	-	-
LAST_NAME	VARCHAR2(60)	No	-	-
GENDER	CHAR(1)	No	-	-
MOBILE_NUMBER	NUMBER(13,0)	No	-	-
EMAIL	VARCHAR2(120)	No	-	-
ISRIDEOWNER	CHAR(1)	No	0	-
LICENSE_DRIVING_NUMBER	NUMBER	Yes	-	-
LICENSE_DRIVING_DATE	DATE	Yes	-	-
BIRTHDATE	DATE	No	-	-
CREATIONDATE	TIMESTAMP(3)	Yes	systimestamp	-
SMOKING_PREFERENCE	CHAR(1)	Yes	-	-
PET_PREFERENCE	CHAR(1)	Yes	-	-
BANK_ACCOUNT_NUMBER	VARCHAR2(19)	No	-	-
				1 - 1

This table involves all the information that the members can fill during their subscription. Some are mandatory such as the personal information (*First name, Last name, Gender*, …) and *bank account number* and some others, concerning their preferences about the co-travellers, are not.

Regarding the preferences, the member can fill them and there are 2 of them. The first one is about *Smoking* in the car. He will need to answer "Y" for "Yes" if he doesn't mind. Otherwise, he will answer "N" for "No". This is the same principle for the preference about having a *pet* in the vehicle.

In this table, the attribute ISRIDEOWNER can be filled by "N" (=No) or "Y" (=Yes). If the member answers ISRIDEOWNER by "Y", the 2 following attributes license_driving_number and license_driving_date (the date the driver obtained his driving license) will become mandatory. In SQL, it means that they become "not null".

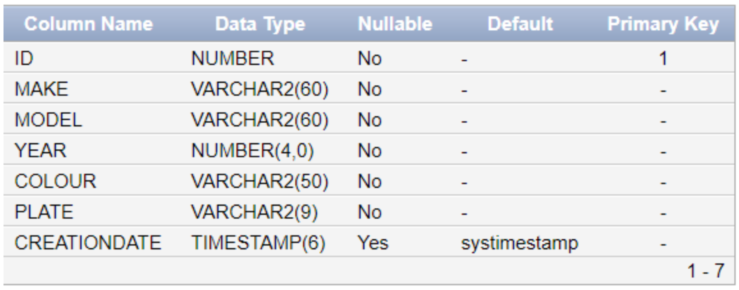
We implemented this constraint as followed:

```
CONSTRAINT "CHK_LICENSEFORDRIVER" CHECK ("ISRIDEOWNER"='Y' AND
"LICENSE_DRIVING_DATE" IS NOT NULL AND "LICENSE_DRIVING_NUMBER" IS
NOT NULL OR "ISRIDEOWNER"='N') ENABLE)
```

This table is one of the main tables of our database and gathers all the users of the platform, either they are passengers or drivers.

At first, we had created another table DRIVER filled with the members who wanted to offer a ride and thus had to enter their Driving license number. But then, a specialist in database models told us that it would not be optimal to do so because it would implicate another table with a lot of data (and the computation time for some Queries would be higher). Consequently, for the sake of efficiency, we decided to create only one table storing all the members and then creating a constraint (described in the part "Table CAR") that allows only the members who have filled the information about their driving license number and date to add a car.

Table CAR



Column Name	Data Type	Nullable	Default	Primary Key
ID	NUMBER	No	-	1
MAKE	VARCHAR2(60)	No	-	-
MODEL	VARCHAR2(60)	No	-	-
YEAR	NUMBER(4,0)	No	-	-
COLOUR	VARCHAR2(50)	No	-	-
PLATE	VARCHAR2(9)	No	-	-
CREATIONDATE	TIMESTAMP(6)	Yes	systimestamp	-

1 - 7

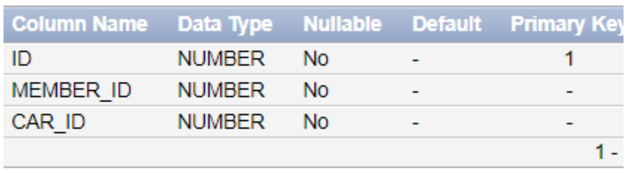
The driver also has to fill some details about the car(s) he will use for the rides he offers. These details involve the *Make* of the car, the *Model*, the *Year*, the *Colour* and the *Plate* of the vehicle. This information is mandatory since they will help the co-travellers to recognize the car when the driver comes to pick them up. The type of the attribute *Plate* is varchar2(9) since the platform can be used in all Europe and none European car plate can excess 9 characters.

An important part that we wanted to implement, was that we wanted to make sure that a member who isn't a driver can't create a car. Therefore, we created a trigger that returns an error message "Not possible to link a car to a member who is not a driver" when a non-driver member registers a car:

```
CREATE OR REPLACE TRIGGER  "car_member_isrideowner_TRG"

BEFORE

insert or update on member_car

for each row

declare

lc_isrideowner member.isrideowner%type;

begin

select mbr.isrideowner

into lc_isrideowner

from member mbr

where id = :new.member_id;
```

```
IF lc_isrideowner='N' then raise_application_error (-20999,'Pas
possible de lier un véhicule à un membre qui n''est pas
conducteur!'); end if;

end;
```

Table MEMBER_CAR

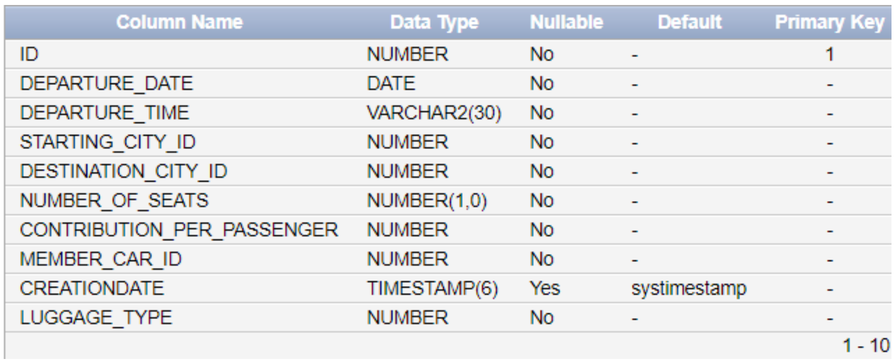


Column Name	Data Type	Nullable	Default	Primary Key
ID	NUMBER	No	-	1
MEMBER_ID	NUMBER	No	-	-
CAR_ID	NUMBER	No	-	-

1 -

This table is only used to create a Many-to-Many relationship between the tables DRIVER and CAR. Indeed, this allows a member to register multiple cars and that a car can belong to multiple members. This will be explained more in details in the next section of this report concerning the relationships.

Table RIDE



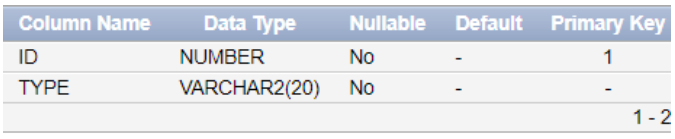
Column Name	Data Type	Nullable	Default	Primary Key
ID	NUMBER	No	-	1
DEPARTURE_DATE	DATE	No	-	-
DEPARTURE_TIME	VARCHAR2(30)	No	-	-
STARTING_CITY_ID	NUMBER	No	-	-
DESTINATION_CITY_ID	NUMBER	No	-	-
NUMBER_OF_SEATS	NUMBER(1,0)	No	-	-
CONTRIBUTION_PER_PASSENGER	NUMBER	No	-	-
MEMBER_CAR_ID	NUMBER	No	-	-
CREATIONDATE	TIMESTAMP(6)	Yes	systimestamp	-
LUGGAGE_TYPE	NUMBER	No	-	-

1 - 10

When a driver wants to offer a ride, he has to create it by filling some mandatory information. He needs to indicate the Departure date and time, the city from which he will start his trip and the destination city, the number of seats in his car, the contribution asked per passenger and the kind of luggage allowed. The data type of the attribute Luggage_Type is "Number" because it references the Primary Key of the table LUGGAGE_TYPE which will be explained in the next sub-section. In parallel, the data type of the attributes Starting-City_ID and Destination_City_ID is also "Number" because they also refer to the primary of the table CITY which will be explained below.

A ride is linked to a unique combination Member/Car thanks to the table MEMBER_CAR that links both tables.

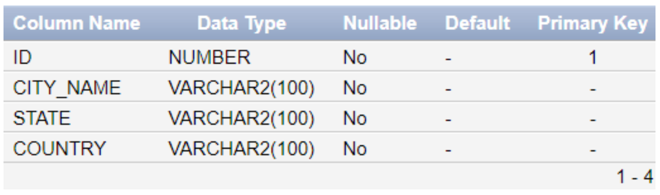
Table LUGGAGE_TYPE



Column Name	Data Type	Nullable	Default	Primary Key
ID	NUMBER	No	-	1
TYPE	VARCHAR2(20)	No	-	-
				1 - 2

As previously explained above, by creating a ride, the driver has to indicate the type of luggage he will allow. The table LUGGAGE_TYPE stores three categories of luggage: Small (20x20) (Id = 1), Medium (30x50) (Id=2) and Large (45x80) (Id=3). For a specific ride, the driver will have to choose from this list what will be allowed in his car.

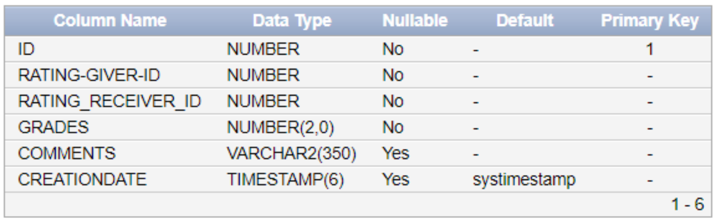
Table CITY



Column Name	Data Type	Nullable	Default	Primary Key
ID	NUMBER	No	-	1
CITY_NAME	VARCHAR2(100)	No	-	-
STATE	VARCHAR2(100)	No	-	-
COUNTRY	VARCHAR2(100)	No	-	-
				1 - 4

The table CITY stores a list of all the cities served by the platform. It should include the relevant state and country information for each city. When a driver creates a ride, he has to pick one of these cities as the departure city and another as the destination city.

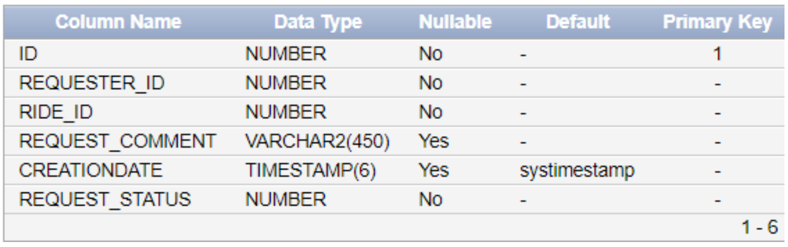
Table RATING



Column Name	Data Type	Nullable	Default	Primary Key
ID	NUMBER	No	-	1
RATING-GIVER-ID	NUMBER	No	-	-
RATING_RECEIVER_ID	NUMBER	No	-	-
GRADES	NUMBER(2,0)	No	-	-
COMMENTS	VARCHAR2(350)	Yes	-	-
CREATIONDATE	TIMESTAMP(6)	Yes	systimestamp	-
				1 - 6

We have created this table in order to implement a rating system. At the end of the trip, the rider can evaluate and add comments about the driver and vice versa. Our grading system is on a scale from 1 to 10 and comments are optional. The goal of this rating is to provide useful information for the rider to choose a driver but also for a driver to accept a rider. But, it also enables the members to notice what should be improved for the next rides.

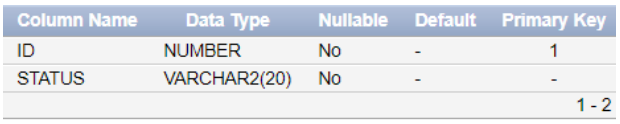
Table REQUEST



Column Name	Data Type	Nullable	Default	Primary Key
ID	NUMBER	No	-	1
REQUESTER_ID	NUMBER	No	-	-
RIDE_ID	NUMBER	No	-	-
REQUEST_COMMENT	VARCHAR2(450)	Yes	-	-
CREATIONDATE	TIMESTAMP(6)	Yes	systimestamp	-
REQUEST_STATUS	NUMBER	No	-	-
				1 - 6

This table stores the information related to the passengers' requests. The members can look at the list of the rides offered by the drivers and put in a request for a specific trip. He can also add a comment if he wants to, but this is not mandatory. The data type of the attribute Request_Status is "Number" because it references the Primary key of the table REQUEST_STATUS explained in the next sub-section.

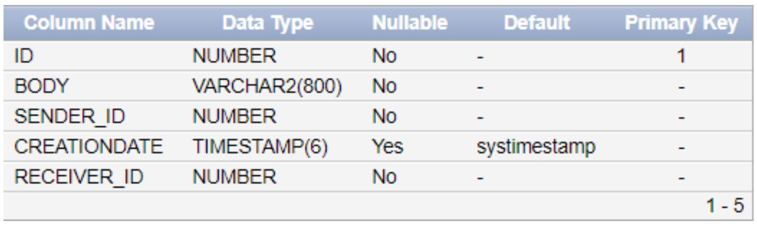
Table REQUEST_STATUS



Column Name	Data Type	Nullable	Default	Primary Key
ID	NUMBER	No	-	1
STATUS	VARCHAR2(20)	No	-	-
				1 - 2

This table stores the three possible value of the Request' status: Pending (Id=1), Approved(Id=2), Rejected(Id=3). The table REQUEST will be automatically updated depending if the driver approves or rejects the request.

Table MESSAGE



Column Name	Data Type	Nullable	Default	Primary Key
ID	NUMBER	No	-	1
BODY	VARCHAR2(800)	No	-	-
SENDER_ID	NUMBER	No	-	-
CREATIONDATE	TIMESTAMP(6)	Yes	systimestamp	-
RECEIVER_ID	NUMBER	No	-	-
				1 - 5

This table is used to enable the members to communicate with each other. They may, for example, ask some details about the ride, about the other passengers, ask some information about the destination city, where exactly he can be picked up and dropped off, …

With this table, a user can send a message to another. The attribute Body contains the message content.