In this reading, you'll explore the concept of keys in more depth, so you have a better understanding of how to choose a primary key from a list of "candidate keys", and how to connect tables together with foreign keys in SQL.

A relational database is a collection of data that is managed and maintained in a database management system such as Oracle or MySQL. A relational database enables you to retrieve every single piece of stored data. This can be done by specifying:

- the name of the target table (or tables),
- the name of the required column (or columns),
- and the primary key of the table.

In a relational database, the primary key can be selected from any candidate key attribute that contains a unique instance value in each row of the table.

Review the following vehicle table. Can you identify the candidate key attributes?

Vehicle table

Vehicle ID	Owner ID	Car plate number	Owner phone number
D01	Ow01	PL02NY	0738297294
D02	Ow02	SN02L2	0725021582
D03	Ow03	PK02L2	0765021583

This table includes three candidate keys:

- the vehicle ID,
- the car plate number,
- and the owner's phone numbers.

Each of these three keys holds unique values in the related column in all rows. Therefore, each of those attributes can potentially act as a primary key column for the vehicle table and could be used to uniquely identify each record in the table. You might notice that there are unique values in all rows that belong to the owner ID. So, is the owner ID a fourth candidate key?

Well, it is correct that the owner ID column has unique values in all rows of the table in the given example. However, it's possible that the same owner may buy another car and will then have multiple cars. So, it's not a good idea to choose this column, because if someone owns multiple cars then the owner ID will no longer be unique. For example, if an owner buys three cars, then their owner ID repeats itself three times in the table. This means that it's no longer unique and thus cannot be chosen as a primary key.

But which of the three attributes would be the best candidate to act as a primary key? The vehicle plate number could be problematic. The owner might choose to change the vehicle plate number by purchasing a private plate number. In this instance, the vehicle plate number associated with the owner would change. This would require updating the plate number wherever it exists in the database. However, in case none of the values are updated, or if they're updated with a wrong number, this will cause confusion and can sometimes even cause serious errors and problems.

Similarly, the owner phone number can't be considered as a primary key because this candidate key value may change if the owner changes their phone number. This will then lead to the same type of problem that you could encounter with the number plate candidate key.

When you design a table in your database, you must make sure to choose a candidate key with a value that cannot change. The vehicle ID candidate key is unique and not expected to change so it should be your primary key. The entity relationship diagram below shows the primary key of the table in bold and underlined.

Vehicle	
PK	Vehicle ID VARCHAR(10)
	OwnerID VARCHAR(10)
	PlateNumber VARCHAR(10)
	PhoneNumber INT

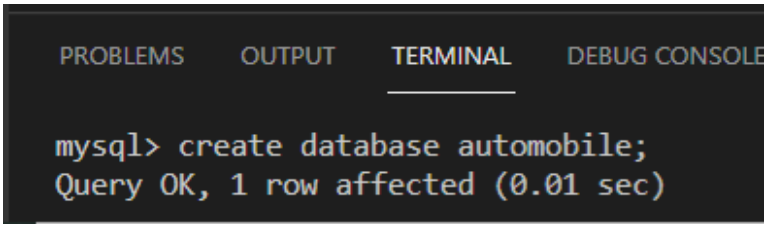
Do you know how to create this table with SQL inside a database called automobile? If yes, please create it now, and remember to define the vehicle ID as a primary key.
Otherwise, complete the following step-by-step instructions inside MySQL in your machine on the Coursera platform.

Create the database

Inside the terminal, write the "CREATE DATABASE" command followed by the name of your new database. In this case, the database name is named "automobile". Finally, add a semi-colon at the end of the statement and click enter on the keyboard to execute the query.

CREATE DATABASE automobile;

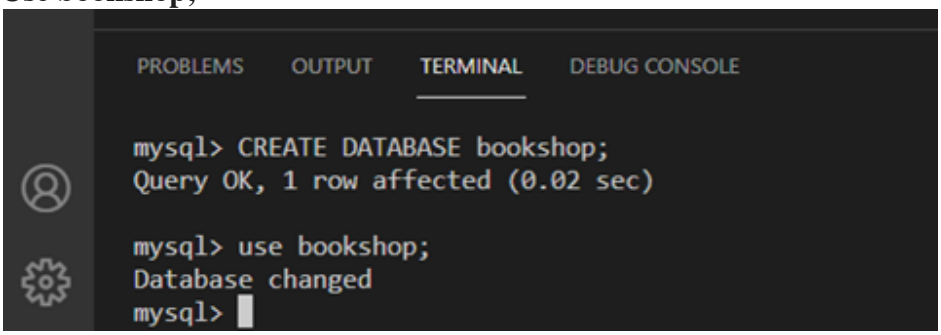
Below is a screenshot of the create database statement inside the terminal.



Well done, you have now created the automobile database.

Create a table
Use the database to create the vehicles table inside the automobile database. You need to choose the automobile database first by typing the following SQL syntax:

Use bookshop;



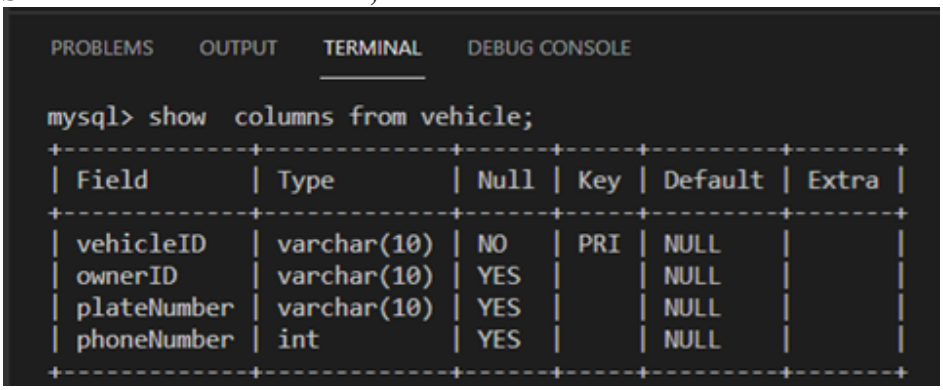
Write the create table statement
To create the vehicle table, write an SQL statement that contains the CREATE TABLE command followed by the name of the table (vehicle in this case) and open parenthesis to define the table's columns including the vehicle ID, owner ID, plate number and phone

number. Of course, each column should be assigned a suitable datatype, as shown in the previous ER diagram.
Once all required columns have been defined, you must add a closing parenthesis and a semi-colon at the end of the SQL statement as follows:
CREATE TABLE vehicle(vehicleID varchar(10), ownerID varchar(10), plateNumber varchar(10), phoneNumber INT);
Click enter to execute the SQL statement. The output result is displayed below.

To show the table you have already created, you can type the following syntax:
Show tables;

The vehicle table is displayed inside the automobile database.

To show the structure of the vehicle table, you can type:
Show columns from vehicle;



Now any candidate key that has not been chosen to act as the primary key of the vehicle table is called an alternate key. That's the plate number and phone number.
Also, it's important to remember that a primary key can be composed of one single simple attribute or multiple attributes (a composite key), which is composed of two or more attributes that form a unique value in each row of the table. This usually happens when a single attribute cannot be found to act as a primary key. If you have any doubt about what a composite key is and how to create it, please review the primary key video where the concept was discussed in detail.
Now let's build another table to maintain data about the vehicles' owners. This table includes information about the owner's name and address as illustrated below.

Owner table

Owner ID	Owner name	Owner address
Ow01	Amjad Omer	110, Elephant Way
Ow02	Hans Henderson	120, Dragon Way
Ow03	Paulo Galdames	130, Giraffe Avenue

The owner ID is the primary key in this table, as it will always be unique. Whereas the owner's name could be the same for different owners, and the owner address could be the same for different owners living at the same address. This would violate the uniqueness requirement for the primary key.

Now create the Owner table in the automobile database. Remember to declare the ownerID as the primary key of the table.

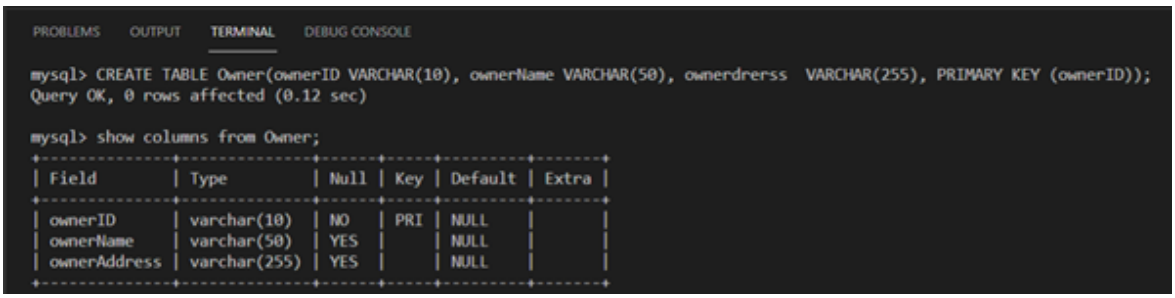
In case you have any difficulty with this task, write the following SQL statement in the terminal section and click enter button to execute the query.

CREATE TABLE Owner(ownerID VARCHAR(10), ownerName VARCHAR(50), ownerdrerss VARCHAR(255), PRIMARY KEY (ownerID));

Let's now examine the structure of the owner table by executing the following SQL query:

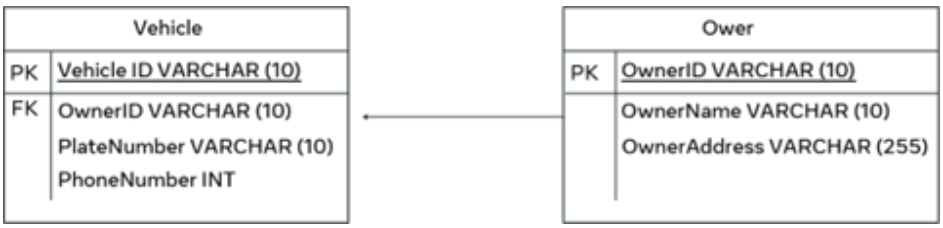
Show columns from vehicle;

The output of the query in the image below depicts the owner table structure, including the columns or fields, data types, key and default values.



By now you have two tables created in the automobile database. The vehicle table contains information about the vehicle ID, owner ID, plate number and phone number. Whereas the owner table contains information about the owner ID, owner name and address.

You may have noticed that the owner ID is a common attribute, as it exists in both tables. However, the key difference is that it is a primary key in the owner table. This means it must be unique in each row of the table. Yet it might be duplicated in the vehicle table, because the same owner might have multiple vehicles. This means that the owner ID in the vehicle table is a suitable choice for a foreign key that connects the vehicle table with the owner table as shown in the following ER diagram.



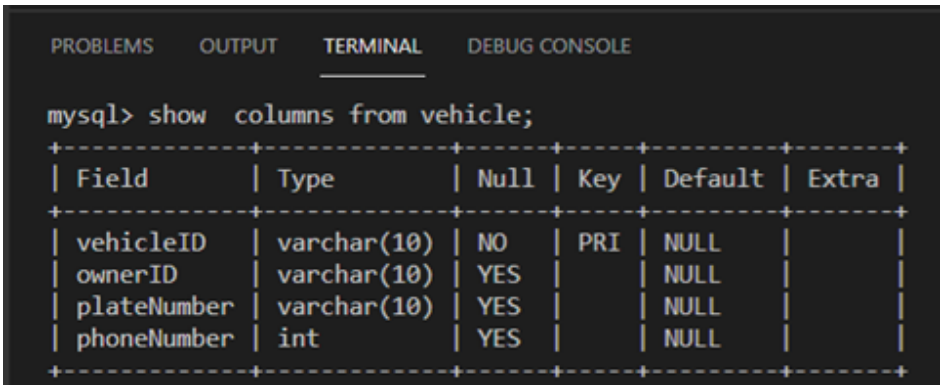
This diagram shows the cross-reference between the two tables. The owner ID column represents the foreign key in the vehicle table that refers to the external primary key column in the owner table. This ensures that each vehicle is associated with the right owner. Also, according to the diagram, the relationship is one-to-many, where each owner may own many cars.

To create this relationship in the actual database you need to modify the vehicle table structure to make the owner ID a foreign key. This can be done by using the “ALTER” command to change the structure of the vehicle table. You can also use the "ADD” command to define the owner ID as the foreign key. Finally, you can use a "REFERENCES” keyword to reference it with the primary key in the owner table.

ALTER TABLE vehicle ADD FOREIGN KEY (ownerID) REFERENCES owner (ownerID);

Once the above alter statement is executed, type the following SQL statement to show the new structure of the vehicle table:

Show columns from vehicle;



You will notice that the owner ID key has been changed into a MUL key, which is one of the three possible values for the "key" attribute in MySQL.

- PRI comes from primary; this means it’s a primary key.
- UNI comes from unique; this means it’s a unique key.
- MUL comes from multiple. If the key is MUL, it means that the related column is permitted to contain the same value in multiple cells of that column.

In this reading, you learned how to choose the primary key from a set of candidate keys. In addition, you learned how to alter the structure of the table to associate two tables by using the foreign key.