

Analyse de la Clientèle d'un Concessionnaire Automobile pour la Recommandation de Modèle

Rendu intermédiaire

Groupe 5



Quan ZHANG
Yajuan LUO
Yue ZHAO
Chaymae FAZAZI-IDRISSI

Table des matières

Contents

1	Description	3
2	Préparation	4
2.1	Nettoyage de la base de données	4
2.1.1	Nettoyage dans fichiers .csv	4
2.1.2	Nettoyage dans Oracle sous SQL	4
3	Analyse	8
3.1	Charger les données dans R	8
3.2	Fusion des fichiers	8
3.2.1	Fusion entre Catalogue.csv et Immatriculations.csv	9
3.2.2	Fusion entre Clients.csv et la fusion comme 3.2.1	10
3.3	Visuellement	11
3.4	Tree de	11
4	Conclusion	12

1 Description

Pour ce projet on doit concevoir un outil qui va permettre à un concessionnaire automobile de cibler les véhicules qui peuvent intéresser ses clients plus précisément :

- un outil rapide qui peut évaluer le type de véhicule qui peut intéresser ses clients en se basant sur les caractéristiques de chaque client et les différents besoins
- envoyer une documentation précise sur le véhicule le plus adéquat pour des clients sélectionnés par son service marketing

Pour cela on doit faire une méthode de gestion de projet et un plan de mise en œuvre, pour l'analyse aussi il va nous falloir les techniques de data mining, machine learning et deep learning qu'on va utiliser dans notre projet pour répondre à cette problématique en se basant sur des fichiers de données qu'on a à disposition (Catalogue, Immatriculations, Marketing) et les données des clients.

2 Préparation

2.1 Nettoyage de la base de données

2.1.1 Nettoyage dans fichiers .csv

Nous allons trouver et corriger les erreurs de syntaxe:

- Nous remplaçons (è, é) par (e, e) dans tous les fichiers .csv;
- Dans fichier clients.csv: Nous remplaçons (Masculin, Homme, Féminin) par (M, M, F, F);

Pour l'instant, nous ne traitons pas ces valeurs vides et inconnus (grâce à `filter()` dans `library(dplyr)` sous R ou bien nettoyage sous SQL, nous allons sélectionner les données correctes), par exemple:

535	48 F	159	En Couple		FALSE	9652 KH 90
536	36 M	1270	celibataire	0	FALSE	6249 OO 45
537	52 M	594	celibataire	0	FALSE	7653 EU 72
538	41 F	1385	En Couple ?		FALSE	5239 SI 80

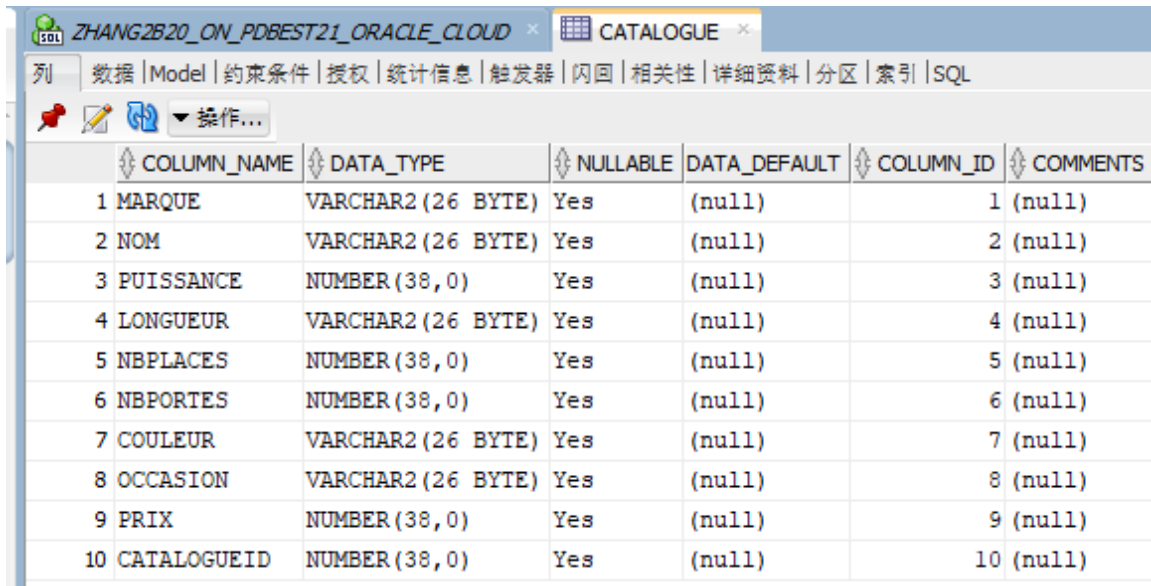
Figure 1: exemple des valeurs vides et inconnus

2.1.2 Nettoyage dans Oracle sous SQL

Nous changeons "2eme voiture" par "deuxiemeVoiture". Quand nous chargeons les données, nous allons cocher pour que les données puissent nullable(default null).

Ensuite, nous allons nettoyer chaque table:

CATALOGUE



	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	MARQUE	VARCHAR2 (26 BYTE)	Yes	(null)	1 (null)	
2	NOM	VARCHAR2 (26 BYTE)	Yes	(null)	2 (null)	
3	PUISSANCE	NUMBER (38, 0)	Yes	(null)	3 (null)	
4	LONGUEUR	VARCHAR2 (26 BYTE)	Yes	(null)	4 (null)	
5	NBPLACES	NUMBER (38, 0)	Yes	(null)	5 (null)	
6	NBPORTES	NUMBER (38, 0)	Yes	(null)	6 (null)	
7	COULEUR	VARCHAR2 (26 BYTE)	Yes	(null)	7 (null)	
8	OCCASION	VARCHAR2 (26 BYTE)	Yes	(null)	8 (null)	
9	PRIX	NUMBER (38, 0)	Yes	(null)	9 (null)	
10	CATALOGUEID	NUMBER (38, 0)	Yes	(null)	10 (null)	

Figure 2: Table: CATALOGUE

Il s'agit d'un tableau très basique, nous pouvons donc constater qu'il n'y a pas d'erreurs dans ce tableau. Donc nous changeons rien dedans.

CLIENTS

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	AGE	NUMBER(38,0)	Yes	(null)	1 (null)	
2	SEXE	VARCHAR2(26 BYTE)	Yes	(null)	2 (null)	
3	TAUX	VARCHAR2(26 BYTE)	Yes	(null)	3 (null)	
4	SITUATIONFAMILIALE	VARCHAR2(26 BYTE)	Yes	(null)	4 (null)	
5	NBENFANTSACHARGE	NUMBER(38,0)	Yes	(null)	5 (null)	
6	DEUXIEMEVOITURE	VARCHAR2(26 BYTE)	Yes	(null)	6 (null)	
7	IMMATRICULATION	VARCHAR2(26 BYTE)	Yes	(null)	7 (null)	

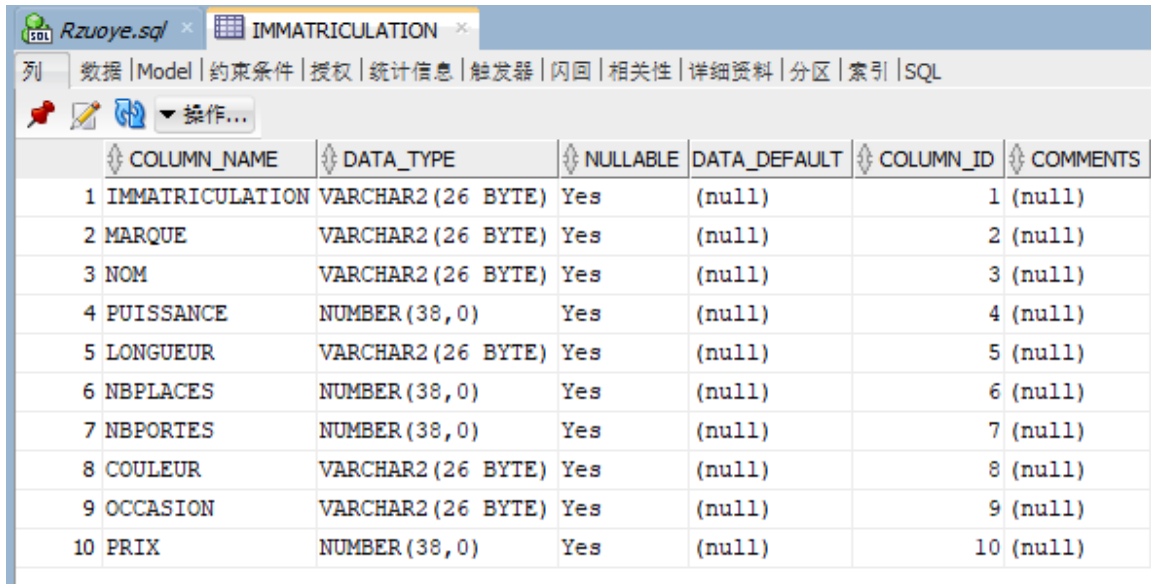
Figure 3: Table: CLIENTS

Les opérations dans table clients sous SQL:

```

1  --Pourque Age soit dans [18, 84]
2  select * from clients where age < 18 or age > 84;
3  DELETE from clients where age < 18 or age > 84;
4
5  --Domain de valeurs 'SEXE': 'F','M'
6  select * from clients where SEXE != 'M' and SEXE != 'F';
7  delete from clients where SEXE != 'M' and SEXE != 'F';
8
9  --Domain de valeurs 'taux': [544, 74185]
10 select * from clients where taux = ' ';
11 delete from clients where taux = ' ';
12 select * from clients where taux = '?';
13 delete from clients where taux = '?';
14 select * from clients where taux < 544 or taux > 74185;
15 delete from clients where TO_NUMBER(taux) < 544 or TO_NUMBER(taux) > 74185;
16
17 --Pour SITUATIONFAMILIALE, on fait group by pour voir les erreurs possibles
18 select sum(age),SITUATIONFAMILIALE from clients group by SITUATIONFAMILIALE;
19 delete from clients where SITUATIONFAMILIALE = '?';
20 delete from clients where SITUATIONFAMILIALE = ' ';
21 delete from clients where SITUATIONFAMILIALE = 'N/D';
22
23 --Domain de valeurs NBENFANTSACHARGE: [0, 4]
24 select * from clients where NBENFANTSACHARGE < 0 or NBENFANTSACHARGE > 4;
25 delete from clients where NBENFANTSACHARGE < 0 or NBENFANTSACHARGE > 4;
26
27 --DEUXIEMEVOITURE: TRUE or FALSE
28 select * from clients where DEUXIEMEVOITURE != 'TRUE' and DEUXIEMEVOITURE != 'FALSE';
29 delete from clients where DEUXIEMEVOITURE != 'TRUE' and DEUXIEMEVOITURE != 'FALSE';
30
31 -- on touche rien pour les valeurs IMMATRICULATION car il y a pas d'errers comme ' ', '?', 'N/D'

```

IMMATRICULATION


	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	IMMATRICULATION	VARCHAR2(26 BYTE)	Yes	(null)	1	(null)
2	MARQUE	VARCHAR2(26 BYTE)	Yes	(null)	2	(null)
3	NOM	VARCHAR2(26 BYTE)	Yes	(null)	3	(null)
4	PUISSANCE	NUMBER(38,0)	Yes	(null)	4	(null)
5	LONGUEUR	VARCHAR2(26 BYTE)	Yes	(null)	5	(null)
6	NBPLACES	NUMBER(38,0)	Yes	(null)	6	(null)
7	NBPORTES	NUMBER(38,0)	Yes	(null)	7	(null)
8	COULEUR	VARCHAR2(26 BYTE)	Yes	(null)	8	(null)
9	OCCASION	VARCHAR2(26 BYTE)	Yes	(null)	9	(null)
10	PRIX	NUMBER(38,0)	Yes	(null)	10	(null)

Figure 4: Table: IMMATRICULATION

Les opérations dans table IMMATRICULATION sous SQL:

```

1  --Afin d'utiliser Group by, on doit nettoyer au moins un column, par exemple 'PRIX'
2  --Domain de valeurs PRIX: [7500, 101300]
3  select * from IMMATRICULATION where prix < 7500 or prix > 101300;
4  --on touche pas si'l y a pas d'erreurs
5
6  --Pour les valeurs marque:
7  select sum(prix), MARQUE from IMMATRICULATION group by MARQUE;
8  --on touche pas si'l y a pas d'erreurs
9
10 --Pour les valeurs nom:
11 select sum(prix), nom from IMMATRICULATION group by nom;
12 --on touche pas si'l y a pas d'erreurs
13
14 --Domain de valeurs PUISSANCE: [55, 507]
15 select * from IMMATRICULATION where PUISSANCE < 55 or PUISSANCE > 507;
16 --on touche pas si'l y a pas d'erreurs
17
18 --Pour les valeurs longueur:
19 select sum(prix), longueur from IMMATRICULATION group by longueur;
20 --on touche pas si'l y a pas d'erreurs
21
22 --Domain de valeurs NBPLACES: [5, 7]
23 select * from IMMATRICULATION where NBPLACES < 5 or NBPLACES > 7;
24 --on touche pas si'l y a pas d'erreurs
25
26 --Domain de valeurs NBPORTES: [3, 5]
27 select * from IMMATRICULATION where NBPORTES < 3 or NBPORTES > 5;
28 --on touche pas si'l y a pas d'erreurs
29

```

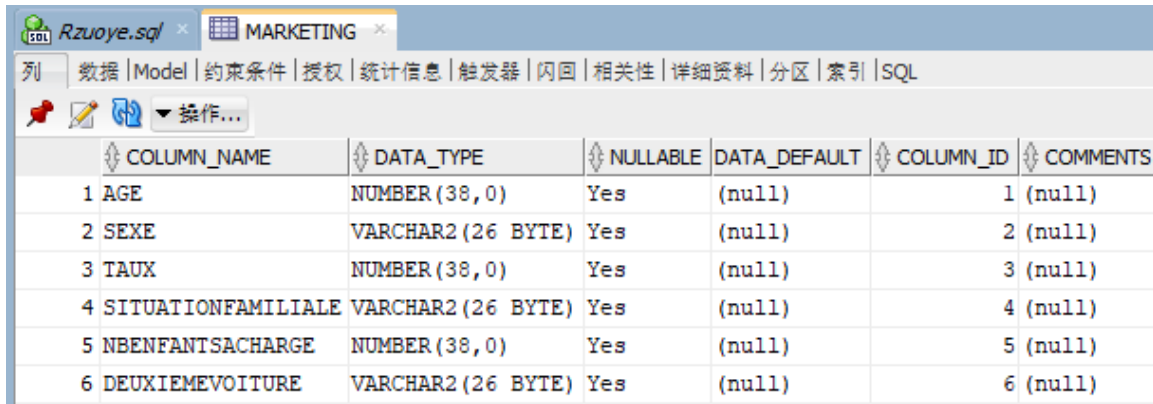
```

30 --Pour les valeurs COULEUR:
31 select sum(prix), COULEUR from IMMATRICULATION group by COULEUR;
32 --on touche pas si'l y a pas d'erreurs
33
34 --Pour les valeurs OCCASION:
35 select sum(prix), OCCASION from IMMATRICULATION group by OCCASION;
36 --on touche pas si'l y a pas d'erreurs

```

Après vérification, il n'y a aucune erreur dans ce tableau

MARKETING



	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	AGE	NUMBER(38,0)	Yes	(null)	1	(null)
2	SEXE	VARCHAR2(26 BYTE)	Yes	(null)	2	(null)
3	TAUX	NUMBER(38,0)	Yes	(null)	3	(null)
4	SITUATIONFAMILIALE	VARCHAR2(26 BYTE)	Yes	(null)	4	(null)
5	NBNFANTSACHARGE	NUMBER(38,0)	Yes	(null)	5	(null)
6	DEUXIEMEVOITURE	VARCHAR2(26 BYTE)	Yes	(null)	6	(null)

Figure 5: Table: MARKETING

Les opérations dans table MARKETING sous SQL:

```

1 --Domain de valeurs 'taux': [544, 74185]
2 select * from MARKETING where taux < 544 or taux > 74185;
3 delete from MARKETING where taux < 544 or taux > 74185;

```

IL FAUT PAS OUBLIER:

```

1 COMMIT;

```

3 Analyse

3.1 Charger les données dans R

```

1 install.packages("RJDBC")
2 library(RJDBC)
3
4 drv <- RJDBC::JDBC(driverClass = "oracle.jdbc.OracleDriver", classPath = Sys.glob("C:/Users/12506/
   OneDrive/Desktop/ESTIA_3A/R/Oracle/drivers/*"))
5
6 ##classPath : add path to drivers jdbc
7
8 #Connexion OK
9 conn <- dbConnect(drv, "jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=144.21.67.201)(
   PORT=1521))(CONNECT_DATA=(SERVICE_NAME=pdbest21.631174089.oraclecloud.internal)))", "ZHANG2B20",
   "ZHANG2B2001")
10
11 allTables <- dbGetQuery(conn, "SELECT owner, table_name FROM all_tables where owner = 'BABEAU2B20'")
12
13 tableCatalogue <- dbGetQuery(conn, "select * from Catalogue")
14 tableClients <- dbGetQuery(conn, "select * from Clients")
15 tableIm <- dbGetQuery(conn, "select * from IMMATRICULATION")
16 tableMar <- dbGetQuery(conn, "select * from MARKETING")
17 View(tableCatalogue)
18 View(tableClients)
19 View(tableIm)
20 View(tableMar)

```








Data		
allTables	5 obs. of 2 variables	
conn	Formal class JDBCConnection	
drv	Formal class JDBCdriver	
tableCatalogue	270 obs. of 10 variables	
tableClients	11065 obs. of 7 variables	
tableIm	1048575 obs. of 10 variables	
tableMar	9 obs. of 6 variables	

Figure 6: Les tables apres les nettoyages

3.2 Fusion des fichiers

- Le fichier Clients.csv contient les informations sur les clients ayant les véhicules vendus cette année.
- Le fichier Immatriculations.csv contient les informations sur les véhicules vendus cette année.
- le fichier Catalogue.csv identifier des catégories de véhicules.

Afin d'éviter une prédiction introuvable (Différents types de voitures ont des paramètres différents) et donner la recommandation aux clients, nous prédisons un paramètre chaque fois (marque, nom, puissance....).

Pour la recommandation de modèle, au début de ce projet, nous avons ajouté un column "catalogueId" dans fichier Catalogue.csv. Grâce à ce column, la fusion entre Catalogue.csv et Immatriculations.csv nous permet d'obtenir une relation entre **IMMATRICULATION** et **CATALOGUEID** et la fusion entre Immatriculations.csv et Clients.csv nous permet d'obtenir une relation entre **les paramètres de clients** et **CATALOGUEID**.

Nous allons analyser les relations entre les informations sur les clients et CATALOGUEID et predire pour la Recommandation de Modèle pour les clients sélectionnés par le service marketing dans Marketing.csv.

3.2.1 Fusion entre Catalogue.csv et Immatriculations.csv

```
1 ##charger tous les libraries possible utilises
2 library(rvest)
3 library(ggplot2)
4 library(dplyr)
5 library(scales)
6 library(maps)
7 library(mapproj)
8 library(plotly)
9 library(rpart)
10 library(rpart.plot)
11 library(C50)
12 library(tree)
13 library(ROCR)
14 library(randomForest)
15 library(e1071)
16 library(naivebayes)
17 library(nnet)
18 library(kknn)
19
20 tableIm.f <- merge(tableIm, tableCatalogue, by = c("MARQUE", "NOM", "PUISSANCE", "LONGUEUR", "NBPORTES", "
    COULEUR", "OCCASION", "PRIX"))
21
22 tableIm.f <- tableIm.f[c(9, 12)]
23 View(tableIm.f)
```

	IMMATRICULATION	CATALOGUEID
654094	7820 YI 94	117
654095	1543 JP 74	117
654096	6692 ZE 81	117
654097	8969 DO 95	117
654098	4260 SV 64	117
654099	7166 PE 13	117
654100	9870 OA 14	117
654101	895 AW 29	117
654102	7120 LQ 69	117
654103	249 AO 23	117
654104	8500 NA 93	117
654105	6372 OG 71	117
654106	4467 FX 42	117
654107	7107 QQ 84	117
654108	895 KO 72	117
654109	7008 CI 08	117

Figure 7: Fusion entre Catalogue.csv et Immatriculations.csv

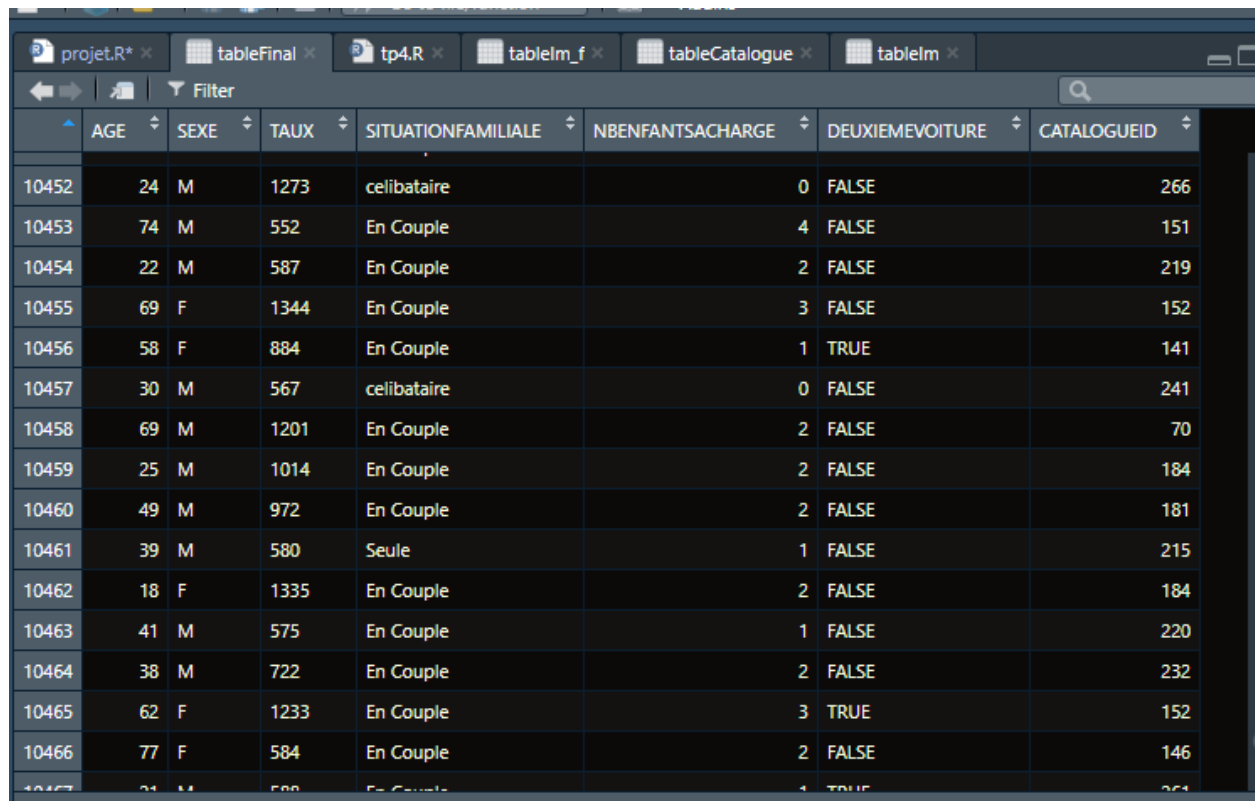
3.2.2 Fusion entre Clients.csv et la fusion comme 3.2.1

```

1 tableFinal <- merge(tableClients, tableIm_f, by = "IMMATRICULATION")
2 View(tableFinal)
3
4 ## supprimer column "IMMATRICULATION"
5 tableFinal <- subset(tableFinal, select=-IMMATRICULATION)
6 View(tableFinal)

```

Finalement, nous avons obtenu une table qui contient tous les champs nous interessent:



	AGE	SEXE	TAUX	SITUATIONFAMILIALE	NBENFANTSACHARGE	DEUXIEMEVOITURE	CATALOGUEID
10452	24	M	1273	celibataire	0	FALSE	266
10453	74	M	552	En Couple	4	FALSE	151
10454	22	M	587	En Couple	2	FALSE	219
10455	69	F	1344	En Couple	3	FALSE	152
10456	58	F	884	En Couple	1	TRUE	141
10457	30	M	567	celibataire	0	FALSE	241
10458	69	M	1201	En Couple	2	FALSE	70
10459	25	M	1014	En Couple	2	FALSE	184
10460	49	M	972	En Couple	2	FALSE	181
10461	39	M	580	Seule	1	FALSE	215
10462	18	F	1335	En Couple	2	FALSE	184
10463	41	M	575	En Couple	1	FALSE	220
10464	38	M	722	En Couple	2	FALSE	232
10465	62	F	1233	En Couple	3	TRUE	152
10466	77	F	584	En Couple	2	FALSE	146
10467	31	M	588	En Couple	1	TRUE	264

Figure 8: La table finale









Ensuite, nous allons analyser les relations parmi les informations de clients et modèle de voiture.

3.3 Visuellement

3.4 Tree de

4 Conclusion

Après la dernière fusion, on obtient `tableFinal` qui contient plus de lignes que `tableClients`. C-à-d, il y a encore des erreurs à nettoyer. Mais nous allons ignorer ces erreurs (0.1% n'influence pas)

Data		
▶ <code>allTables</code>	5 obs. of 2 variables	
▶ <code>conn</code>	Formal class <code>JDBCConnection</code>	
▶ <code>drv</code>	Formal class <code>JDBCdriver</code>	
▶ <code>tableCatalogue</code>	270 obs. of 10 variables	
▶ <code>tableClients</code>	11065 obs. of 7 variables	
▶ <code>tableFinal</code>	11082 obs. of 7 variables	
▶ <code>tableIm</code>	1048575 obs. of 10 variables	
▶ <code>tableIm_f</code>	1048575 obs. of 2 variables	
▶ <code>tableMar</code>	9 obs. of 6 variables	