

Université des Sciences et Technologies Houari Boumediene  
Faculté d'Informatique

Master I  
Ingénierie de Logiciels  
Bases de Données Avancées

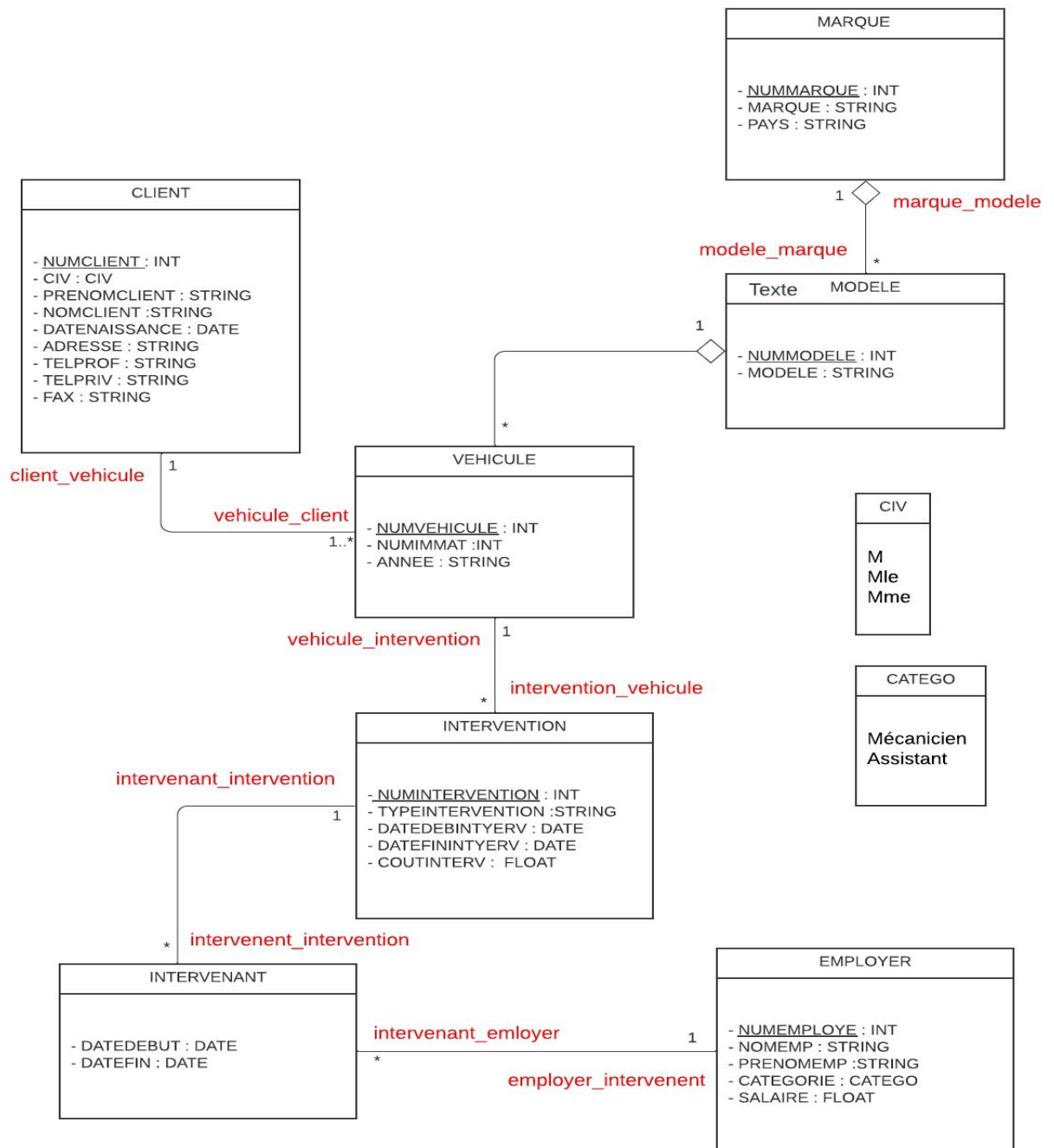
Projet  
SQL3-Oracle et NoSQL (MongoDB)

BOUMAZA MUSTAPHA MOUNIR  
AMELLAL YACINE

# Partie I : Relationnel-Objet

## A- Modélisation orientée objet :

### *1-Transformation de schéma relationnel en un schéma Objet (diagramme de classes)*



## B- Création des TableSpaces et utilisateur :

## 2- Creation des tables spaces :

```
SQL> CREATE TABLESPACE SQL3_TBS
 2 DATAFILE 'c:\mytbs.dat'
 3 SIZE 100M
 4 AUTOEXTEND ON
 5 NEXT 10M
 6 MAXSIZE UNLIMITED;

Tablespace crÚÚ.

SQL>
SQL> CREATE TEMPORARY TABLESPACE SQL3_TempTBS
 2 TEMPFILE 'c:\mytemptbs.dat'
 3 SIZE 50M
 4 AUTOEXTEND ON
 5 NEXT 5M
 6 MAXSIZE UNLIMITED;

Tablespace crÚÚ.
```

## 3-Creation d'utilisateur :

```
SQL> CREATE USER SQL3 IDENTIFIED BY psw
 2 DEFAULT TABLESPACE SQL3_TBS
 3 TEMPORARY TABLESPACE SQL3_TempTBS
 4 QUOTA UNLIMITED ON SQL3_TBS;

Utilisateur crÚÚ.
```

## 4-Donner tous les privilèges à cet utilisateur :

```
SQL> GRANT ALL PRIVILEGES TO SQL3;

Autorisation de privilèges (GRANT) acceptÚe.
```

## C- Langage de définition de données :

*5-Creation des types nécessaires et toutes les associations qui existent :*

```
SQL> create type tclient;
      2 /

Type créé.

SQL>
SQL> create type tvehicule;
      2 /

Type créé.

SQL>
SQL> create type tmodele;
      2 /

Type créé.

SQL>
SQL> create type tmarque;
      2 /

Type créé.

SQL>
SQL> create type tintervention;
      2 /

Type créé.

SQL> create type tintervenant;
      2 /

Type créé.

SQL>
SQL> create type temployeur;
      2 /

Type créé.
```

```

SQL>
SQL> -- cette instruction permet de définir une table imbriquée de référence des objets de type tvehicule
SQL> create type t_set_ref_vehicule as table of ref tvehicule;
    2 /

Type créé.

SQL>
SQL> -- cette instruction permet de définir une table imbriquée de référence des objets de type tmodele
SQL> create type t_set_ref_modele as table of ref tmodele;
    2 /

Type créé.

SQL>
SQL> -- cette instruction permet de définir une table imbriquée de référence des objets de type tintervention
SQL> create type t_set_ref_intervention as table of ref tintervention;
    2 /

Type créé.

SQL>
SQL> -- cette instruction permet de définir une table imbriquée de référence des objets de type tintervenant
SQL> create type t_set_ref_intervenant as table of ref tintervenant;
    2 /

Type créé.

```

```

SQL> -- création de type tclient
SQL> create or replace type tclient as object (NUMCLIENT INTEGER,
    2 CIV varchar2(5),
    3 PRENOMCLIENT varchar2(50),
    4 NOMCLIENT varchar2(50),
    5 DATENAISSANCE DATE,
    6 ADRESSE varchar2(100),
    7 TELPROF varchar2(20),
    8 TELPRIV varchar2(20),
    9 FAX varchar2(20),
    10 client_vehicule t_set_ref_vehicule);
    11 /

```

Type créé.

```

SQL> -- création de type tmarque
SQL> create or replace type tmarque as object (NUMMARQUE INTEGER,
    2 MARQUE varchar2(30),
    3 PAYS varchar2(30),
    4 MARQUE_MODELE t_set_ref_modele);
    5 /

```

Type créé.

```

SQL>
SQL> -- création de type tmodele
SQL> create or replace type tmodele as object (NUMMODELE INTEGER,
    2 NUMMARQUE ref tmarque,
    3 MODELE varchar2(30),
    4 modele_vehicule t_set_ref_vehicule);
    5 /

```

Type créé.

Type crÚÚ.

Type crÚÚ.

Type crÚÚ.

Type crÚÚ.



## 6-Définition Des méthodes :

### La 1er methode :

```
SQL>
SQL> -- 1er methode : Calculer pour chaque employé, le nombre des interventions effectuées.
SQL>
SQL> -- signateur
SQL>
SQL> alter type temployer add member function nb_interventions return integer cascade;

Type modifié.

SQL>
SQL> --body
SQL>
SQL> -- Définition du corps de la méthode pour calculer le nombre d'interventions effectuées par un employé
SQL>
SQL> CREATE OR REPLACE TYPE BODY temployer AS
  2  MEMBER FUNCTION nb_interventions RETURN INTEGER IS
  3    nombre_interventions INTEGER := 0;
  4  BEGIN
  5    nombre_interventions := self.employer_intervenant.COUNT;
  6    RETURN nombre_interventions;
  7  END;
  8  END;
  9  /

Corps de type créé.
```

### La 2em methode :

```
SQL>
SQL> -- signateur
SQL>
SQL> alter type tmodele add member function count_vehicules return integer cascade;

Type modifié.

SQL>
SQL> --body
SQL>
SQL> -- Définition du corps de la méthode pour calculer le nombre de véhicules pour un modèle
SQL> CREATE OR REPLACE TYPE BODY tmodele AS
  2    -- Méthode pour calculer le nombre de véhicules pour le modèle
  3    MEMBER FUNCTION count_vehicules RETURN INTEGER IS
  4      total_vehicules INTEGER := 0;
  5    BEGIN
  6      -- Parcours de la collection des références de véhicules
  7      FOR i IN 1..self.MODELE_VEHICULE.COUNT LOOP
  8        -- Incrémentation du compteur pour chaque référence de véhicule
  9        total_vehicules := total_vehicules + 1;
 10      END LOOP;
 11
 12      -- Retour du nombre total de véhicules pour le modèle
 13      RETURN total_vehicules;
 14    END count_vehicules;
 15  END;
 16  /

Corps de type créé.
```

### La 3em methode :

```
SQL>
SQL> -- 3em methode : Calculer pour chaque modèle, le nombre de véhicules.
SQL>
SQL> -- signateur
SQL>
SQL> alter type tmodele add member function calcul_vehicules return integer cascade;

Type modifié.

SQL>
SQL> --body
SQL>
SQL> -- Définition du corps de la méthode pour calculer le nombre de véhicules pour un modèle
SQL>
SQL> CREATE OR REPLACE TYPE BODY tmodele AS
  2 MEMBER FUNCTION calcul_vehicules RETURN INTEGER IS
  3   nombre_vehicules INTEGER := 0;
  4 BEGIN
  5   nombre_vehicules := self.modele_vehicule.COUNT;
  6   RETURN nombre_vehicules;
  7 END;
  8 END;
  9 /

Corps de type créé.
```

### La 4em methode :

```
SQL> -- signateur
SQL>
SQL> alter type tclient add member function lister_vehicules RETURN t_set_ref_vehicule cascade;

Type modifié.

SQL>
SQL> --body
SQL>
SQL> -- Définition du corps de la méthode pour lister les véhicules de chaque client
SQL>
SQL> CREATE OR REPLACE TYPE BODY tclient AS
  2 MEMBER FUNCTION lister_vehicules RETURN t_set_ref_vehicule IS
  3 BEGIN
  4 RETURN self.client_vehicule;
  5 END;
  6 END;
  7 /

Corps de type créé.
```



### La 5em methode :

```
SQL> -- signateur
SQL>
SQL> alter type tmarque add member function chiffre_affaire return NUMBER cascade;

Type modifi  .

SQL>
SQL> --body
SQL>
SQL> -- D  finition du corps de la m  thode pour calculer le chiffre d'affaires pour chaque marque
SQL>
SQL> CREATE OR REPLACE TYPE BODY tmarque AS
  2   MEMBER FUNCTION chiffre_affaire RETURN NUMBER
  3   IS
  4       l_chiffre_affaires NUMBER := 0;
  5   BEGIN
  6       -- Parcourir tous les mod  les de la marque
  7       FOR m IN (SELECT VALUE(mod) FROM TABLE(self.MARQUE_MODELE) mod)
  8       LOOP
  9           -- Parcourir tous les v  hicules de chaque mod  le
 10          FOR v IN (SELECT VALUE(veh) FROM TABLE(m.modele_vehicule) veh)
 11          LOOP
 12              -- Parcourir toutes les interventions de chaque v  hicule
 13              FOR i IN (SELECT VALUE(inter) FROM TABLE(v.vehicule_intervention) inter)
 14              LOOP
 15                  -- Ajouter le co  t de l'intervention au chiffre d'affaires
 16                  l_chiffre_affaires := l_chiffre_affaires + i.COUTINTERV;
 17              END LOOP;
 18          END LOOP;
 19      END LOOP;
 20      RETURN l_chiffre_affaires;
 21  END chiffre_affaire;
 22  END;
 23  /
```

Avertissement : Corps de type cr     avec erreurs de compilation.

## 7-Creation Des Tables :

```
SQL>
SQL> create table client OF tclient (
  2     NUMCLIENT PRIMARY KEY,
  3     CONSTRAINT civ_check CHECK (CIV IN ('M', 'Mle', 'Mme'))
  4 )
  5 nested table client_vehicule store as table_client_vehicule;

Table cr   e.

SQL>
SQL>
SQL> create table marque of tmarque (PRIMARY KEY(NUMMARQUE))
  2 nested table MARQUE_MODELE store as table_MARQUE_MODELE;

Table cr   e.

SQL>
SQL>
SQL> create table modele of tmodele (
  2     NUMMODELE PRIMARY KEY,
  3     CONSTRAINT fk_marque FOREIGN KEY (NUMMARQUE) REFERENCES marque
  4 )
  5 nested table modele_vehicule store as table_modele_vehicule;

Table cr   e.

SQL>
SQL>
SQL> create table vehicule of tvehicule (
  2     NUMVEHICULE PRIMARY KEY,
  3     CONSTRAINT fk_client FOREIGN KEY (NUMCLIENT) REFERENCES client,
  4     CONSTRAINT fk_modele FOREIGN KEY (NUMMODELE) REFERENCES modele
  5 )
  6 nested table vehicule_intervention store as table_vehicule_intervention;

Table cr   e.
```

```
SQL> create table intervention of tintervention (
  2     NUMINTERVENTION PRIMARY KEY,
  3     CONSTRAINT fk_vehicule FOREIGN KEY (NUMVEHICULE) REFERENCES vehicule
  4 )
  5 nested table intervention_intervenant store as table_intervention_intervenant;

Table cr   e.

SQL>
SQL>
SQL> create table employer OF temployer (
  2     NUMEMPLOYER PRIMARY KEY,
  3     CONSTRAINT categorie_check CHECK (CATEGORIE in ('M  canicien','Assistant'))
  4 )
  5 nested table employer_intervenant store as table_employer_intervenant;

Table cr   e.

SQL>
SQL>
SQL> create table intervenant of tintervenant (
  2     CONSTRAINT fk_intervention FOREIGN KEY (NUMINTERVENTION) REFERENCES intervention,
  3     CONSTRAINT fk_employer FOREIGN KEY (NUMEMPLOY  ) REFERENCES employer
  4 );

Table cr   e.
```

## D- Langage de manipulation de données :

*Capture de quelques tables apres le remplissage de toutes les tables .*

### Table CLIENT:

```
SQL> select *from client;
```

NUMCLIENT	CIV	PRENOMCLIE	NOMCLIENT	DATENAIS	ADRESSE	TELP	PROF	TELPRIV	FAX	CLIENT_VEHICULE
1	Mme	Cherifa	MAHBOUBA	08/08/57	CITE 1013 LOGTS BT 6 1 Alger	0561381813		0562458714		T_SET_REF_VEHICULE(0 0002202085E00BA7B7BB 3457E8C9213B1F2D868D BD2F9588315A54F6588E 3D8DC9C0E3D5C)
2	Mme	Lamia	TAHMI	31/12/55	CITE BACHEDJARA BAT IMENT 38 -Bach Djerr ah-Alger	0562467849		0561392487		T_SET_REF_VEHICULE(0 000220208EB67A38F85D 34A398D2FB93BFD943CF 5D2F9588315A54F6588E 3D8DC9C0E3D5C)
NUMCLIENT	CIV	PRENOMCLIE	NOMCLIENT	DATENAIS	ADRESSE	TELP	PROF	TELPRIV	FAX	CLIENT_VEHICULE
3	Mle	Ghania	DIAF	31/12/55	43, RUE ABDERRAHMANE SBAA BELLE VUE-EL H ARRACH-ALGER	0523894562		0619430945	0562784254	T_SET_REF_VEHICULE(0 00022020859B44C76646 145EF937C9792E827915 9D2F9588315A54F6588E 3D8DC9C0E3D5C, 00002 20208BBD8AB4567F1403 490FB4C4D23791DA4D2F 9588315A54F6588E3D8D C9C0E3D5C)
NUMCLIENT	CIV	PRENOMCLIE	NOMCLIENT	DATENAIS	ADRESSE	TELP	PROF	TELPRIV	FAX	CLIENT_VEHICULE
4	Mle	Chahinaz	MELEK	27/06/55	HLM AISSAT IDIR CAGE 9 3 EME ETAGE-EL HA RRACH ALGER	0634613493		0562529463		T_SET_REF_VEHICULE(0 0002202088C7D78098DB B4689B0CFC18E43182AA 4D2F9588315A54F6588E 3D8DC9C0E3D5C)
5	Mme	Noura	TECHTACHE	22/03/49	16, ROUTE EL DJAMILA -AIN BENIAN-ALGER	0562757834		0562757843	0562757843	T_SET_REF_VEHICULE()

20	M	Younes	CHALAH		CITE DES 60 LOGTS BT D N 48-NACIRIA-BOUM ERDES			0561358279		T_SET_REF_VEHICULE(0 0002202083C09A36EB5F 340CC993BCD6E6A9184 3D2F9588315A54F6588E 3D8DC9C0E3D5C, 00002 202082542884F98ED47C F818C4F76D34C87D4D2F 9588315A54F6588E3D8D
NUMCLIENT	CIV	PRENOMCLIE	NOMCLIENT	DATENAIS	ADRESSE	TELP	PROF	TELPRIV	FAX	CLIENT_VEHICULE
21	M	Boubeker	BARKAT	08/11/35	CITE MENTOURI N 71 B T AB SMK Constantine	0561824538		0561326179		T_SET_REF_VEHICULE()
22	M	Seddik	HMIA		25 RUE BEN YAHYIA-JI JEL	0562379513			0562493627	T_SET_REF_VEHICULE(0 000220208A6633D57427 54CF38B2FAE8E27B6548 FD2F9588315A54F6588E 3D8DC9C0E3D5C, 00002 2020876C7F792C82148B
NUMCLIENT	CIV	PRENOMCLIE	NOMCLIENT	DATENAIS	ADRESSE	TELP	PROF	TELPRIV	FAX	CLIENT_VEHICULE
23	M	Lamine	MERABAT	13/09/65	CITE JEANNE D ARC EC RAN B2-GAMBETTA - OR AN	0561724538		0561724538		T_SET_REF_VEHICULE()

23 ligne(s) s lection  e(s).



Table MODELE:

```
SQL> select *from modele;
```

NUMMODELE	NUMMARQUE	MODELE	MODELE_VEHICULE
2	000022020887DBA3DCD0 EF42C3953F8DC8D06103 467F3EC7594CC547948A F58298B5A0CB69	Diablo	T_SET_REF_VEHICULE(0 0002202085E00BA7B7BB 3457E8C9213B1F2DB68D BD2F9588315A54F6588E 3D8DC9C0E3D5C, 00002 2020876C7F792C82148B 0AEF45714324C05ABD2F 9588315A54F6588E3D8D C9C0E3D5C)
3	0000220208BF7398C741	Série 5	T_SET_REF_VEHICULE(0
NUMMODELE	NUMMARQUE	MODELE	MODELE_VEHICULE
	E448B381468E13280837 FD7F3EC7594CC547948A F58298B5A0CB69		000220208FDB1EBEAF4 D4CC38448C8A20DACECB 8D2F9588315A54F6588E 3D8DC9C0E3D5C)
4	0000220208B848E562C9 064D8CB12EEA278771A1 547F3EC7594CC547948A F58298B5A0CB69	NSX	T_SET_REF_VEHICULE()
5	0000220208FE1E2C8A4F	Classe C	T_SET_REF_VEHICULE(0
NUMMODELE	NUMMARQUE	MODELE	MODELE_VEHICULE
	1C4A048296293B21CA14 F17F3EC7594CC547948A F58298B5A0CB69		0002202080D22A4090E5 64F699C248ABD495E068 8D2F9588315A54F6588E 3D8DC9C0E3D5C)

25	0000220208C80805B382 E1433E962AA64327ED18 217F3EC7594CC547948A	Séville	T_SET_REF_VEHICULE(0 0002202084B87E3929F0 8452BACF4ECAC6304148
NUMMODELE	NUMMARQUE	MODELE	MODELE_VEHICULE
	F58298B5A0CB69		3D2F9588315A54F6588E 3D8DC9C0E3D5C)
26	000022020837954E218F CE45E482F167ED3DB3BF E37F3EC7594CC547948A F58298B5A0CB69	95 Cabriolet	T_SET_REF_VEHICULE()
27	0000220208BF7398C741 E448B381468E13280837 FD7F3EC7594CC547948A	TT Coupè	T_SET_REF_VEHICULE()
NUMMODELE	NUMMARQUE	MODELE	MODELE_VEHICULE
	F58298B5A0CB69		
28	000022020866454F0795 17434CA0B5BFB2F372E3 747F3EC7594CC547948A F58298B5A0CB69	F 355	T_SET_REF_VEHICULE()
29	00002202087488008C40 7447738C81489A8B2AA6 837F3EC7594CC547948A F58298B5A0CB69	POLO	T_SET_REF_VEHICULE()
NUMMODELE	NUMMARQUE	MODELE	MODELE_VEHICULE

28 ligne(s) sélectionné(s).

## E- Langage d'interrogation de données :

9-

La requête :

```
SQL> SELECT Deref(m.NUMMARQUE).MARQUE AS MARQUE,  
2         m.MODELE AS MODELE  
3 FROM modele m;
```

Resultat :

MARQUE	MODELE
LAMBORGHINI	Diablo
AUDI	Série 5
ALFA-ROMEO	NSX
MERCEDES	Classe C
RENAULT	Safrane
VENTURI	400 GT
LOTUS	Esprit
PEUGEOT	605
TOYOTA	Prévia
FERRARI	550 Maranello
ROLLS-ROYCE	Bentley-Continental

MARQUE	MODELE
ALFA-ROMEO	Spider
MASERATI	Evoluzione
PORSCHE	Carrera
PORSCHE	Boxter
VOLVO	S 80
CHRYSLER	300 M
BMW	M 3
JAGUAR	XJ 8
PEUGEOT	406 Coupé
VENTURI	300 Atlantic
MERCEDES	Classe E

MARQUE	MODELE
LEXUS	GS 300
CADILLAC	Séville
SAAB	95 Cabriolet

25 ligne(s) sélectionné(s).

10-

La requête :

```
SQL> SELECT v.NUMVEHICULE
2  FROM vehicule v
3  WHERE EXISTS (
4      SELECT 1
5      FROM intervention i
6      WHERE i.NUMVEHICULE = REF(v)
7  );
```

Resultat :

```
NUMVEHICULE
-----
1
2
3
6
8
10
14
17
20
21
22

NUMVEHICULE
-----
25
28

13 ligne(s) sélectionné(s).
```



11-

La requête et le resultat :

```
SQL> SELECT AVG(DATEFININTRV - DATEDEBINTRV) AS duree_moyenne_intervention
  2  FROM intervention;

DUREE_MOYENNE_INTERVENTION
-----
                1,890625
```

*AVG est une fonction en SQL qui calcule la moyenne d'une colonne numérique dans un ensemble de résultats. Elle prend en entrée une colonne numérique et retourne la moyenne de toutes les valeurs de cette colonne dans les lignes sélectionnées par la requête.*

*Dans le contexte de la requête précédente, AVG est utilisée pour calculer la moyenne des durées des interventions. Elle prend en compte toutes les durées calculées pour chaque intervention (c'est-à-dire la différence entre la date de fin et la date de début de chaque intervention) et en retourne la moyenne.*

12-

La requête et le resultat :

```
SQL>
SQL> SELECT SUM(COUTINTERV) AS Montant_Global
  2  FROM intervention
  3  WHERE COUTINTERV > 30000;

MONTANT_GLOBAL
-----
        202000
```

12-

La requête et le resultat :

```
SQL> SELECT e.NUMEMPLOYER, e.NOMEMP, e.PRENOMEMP, e.CATEGORIE, COUNT(DISTINCT Deref(it.NUMINTERVENTION).NUMINTERVENTION) AS NB_INTERVENTIONS
  2 FROM employer e ,intervenent it ,intervention i
  3 WHERE e.NUMEMPLOYER = it.NUMEMPLOYE.NUMEMPLOYER
  4 AND Deref(it.NUMINTERVENTION).NUMVEHICULE = i.NUMVEHICULE
  5 GROUP BY e.NUMEMPLOYER, e.NOMEMP, e.PRENOMEMP, e.CATEGORIE
  6 ORDER BY NB_INTERVENTIONS DESC;
```

NUMEMPLOYER	NOMEMP	PRENOMEMP	CATEGORIE	NB_INTERVENTIONS
59	BELHAMIDI	Mourad	Mécanicien	4
60	IGOUDJIL	Redouane	Assistant	3
55	HADJ	Zouhir	Assistant	2
56	OUSSEDIK	Hakim	Mécanicien	2
57	ABAD	Abdelhamid	Assistant	2
53	LACHEMI	Bouزيد	Mécanicien	2
62	RAHALI	Ahcene	Mécanicien	2
65	MOHAMMEDI	Mustapha	Mécanicien	2
64	BADI	Hatem	Assistant	2
54	BOUCHEMLA	Elias	Assistant	2
63	CHAQUI	Ismail	Assistant	1

NUMEMPLOYER	NOMEMP	PRENOMEMP	CATEGORIE	NB_INTERVENTIONS
66	FEKAR	Abdelaziz	Assistant	1
67	SAIDOUNI	Wahid	Mécanicien	1

13 ligne(s) sélectionné(s).

## Partie II : NoSQL – Modèle orienté « documents »

## A- Modélisation orientée document :

On suppose que la plupart des requêtes sur la base vont porter sur les véhicules, leur marque et leurs interventions par les employés (voir exemples de requêtes plus bas).

Proposition d'une modélisation orientée document de la base de données décrite dans la partie I :

```

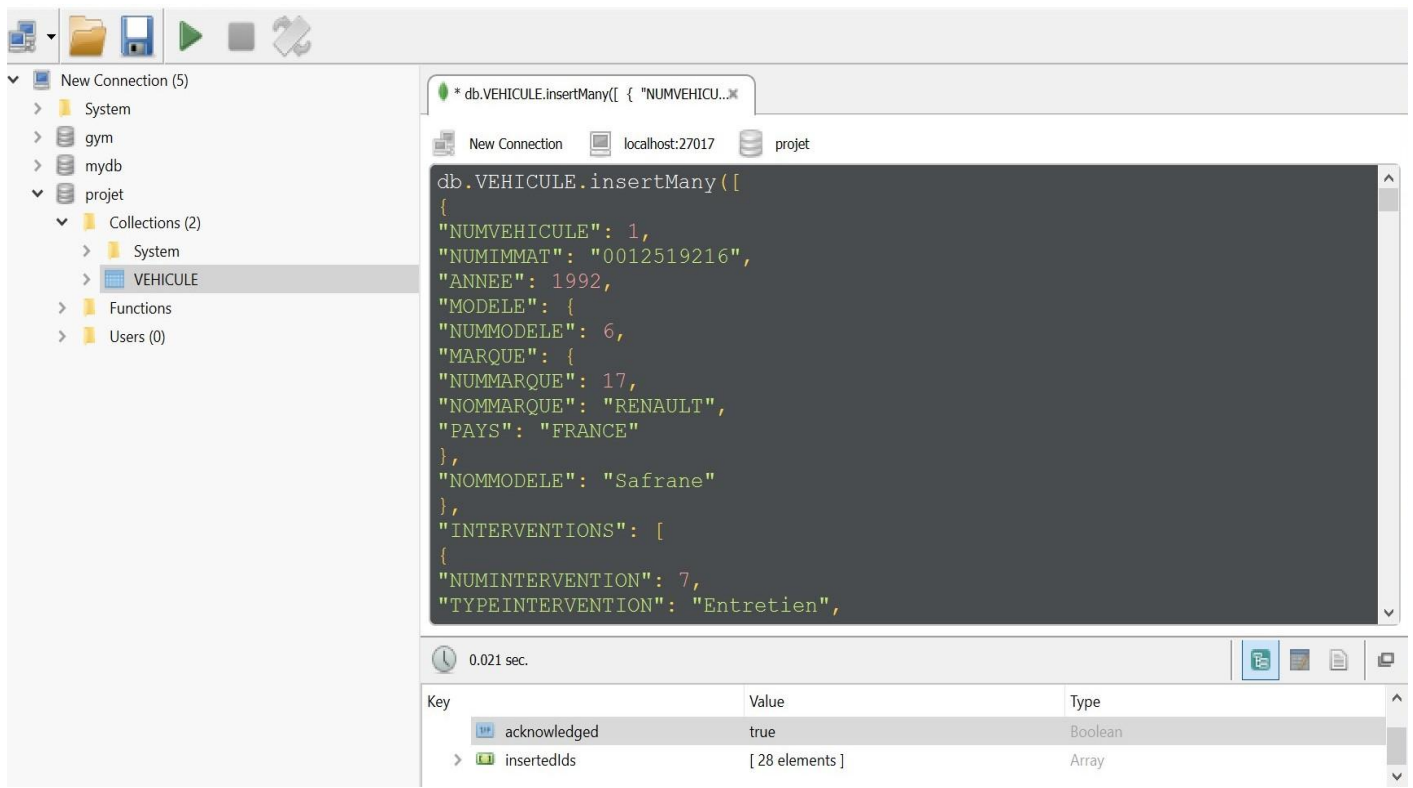
-----collection vehicule-----{
"NUMVEHICULE": ,
"NUMIMMAT": "",
"ANNEE": ,
"MODELE": {
    "NUMMODELE": ,
    "MARQUE": {
        "NUMMARQUE": ,
        "NOMMARQUE": "",
        "PAYS": ""
    },
    "NOMMODELE": ""
},
},
"INTERVENTIONS": [
    {
        "NUMINTERVENTION": ,
        "TYPEINTERVENTION": "",
        "DATEDEBINTERV": ,
        "DATEFININTERV": ,
        "COUTINTERV": ,
        "EMPLOYE": [
            {
                "NUMEMPLOYE": ,
                "NOMEMP": "",
                "PRENOMEMP": "",
                "CATEGORIE": "",
                "SALAIRE": ,
                "DATEDEBUT": ,
                "DATEFIN":
            },
            ]
        },
    ]
}

```

### Justification de notre choix de conception :

Cette modélisation est centrée sur les véhicules et regroupe toutes les informations liées à un véhicule dans un seul document, y compris les INTERVENTIONS et les employés impliqués. Cela facilitera les requêtes courantes sur les véhicules, leurs marques et leurs INTERVENTIONS par les employés, comme indiqué dans l'énoncé. L'imbrication des données liées aux clients, modèles, marques et intervenants permet d'éviter les jointures coûteuses entre les collections et d'avoir toutes les informations nécessaires dans un seul document.

## **B- Remplir la collection :**

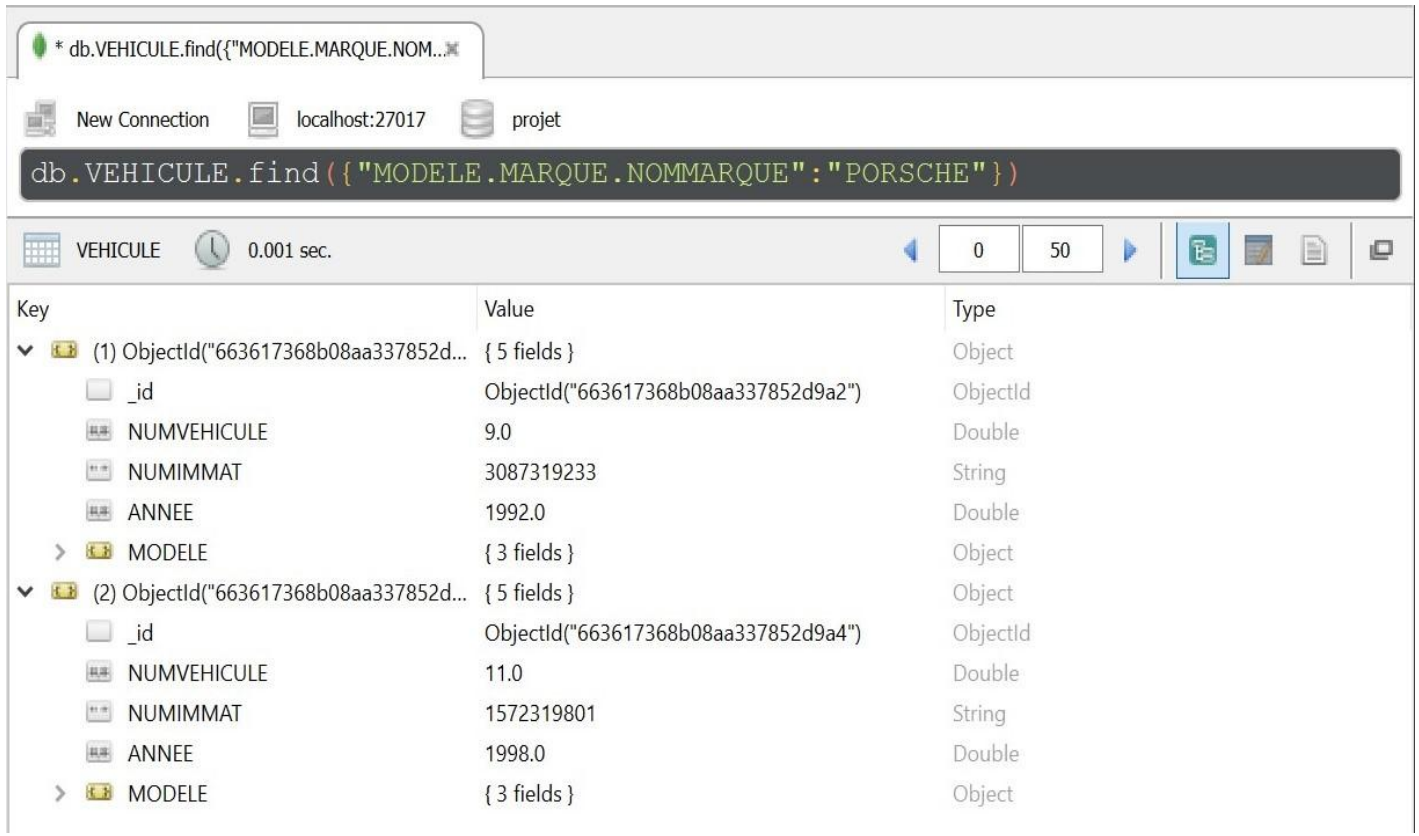


The screenshot displays a MongoDB management interface. On the left, a sidebar shows the database structure: 'New Connection (5)' with sub-items 'System', 'gym', 'mydb', and 'projet'. Under 'projet', there are 'Collections (2)' including 'System' and 'VEHICULE', along with 'Functions' and 'Users (0)'. The 'VEHICULE' collection is selected. The main editor shows a JavaScript command: `* db.VEHICULE.insertMany([ { "NUMVEHICU...`. Below the command, a status bar indicates '0.021 sec.'. At the bottom, a table summarizes the operation results:

Key	Value	Type
acknowledged	true	Boolean
insertedIds	[ 28 elements ]	Array

## C- Répondre aux requêtes :

### 1- Afficher tous les véhicules de la marque «PORSCHE»:



The screenshot shows the MongoDB Compass interface. The query bar contains the following query:

```
db.VEHICULE.find({"MODELE.MARQUE.NOMMARQUE": "PORSCHE"})
```

The results are displayed in a table with columns: Key, Value, and Type. The table shows two documents.

Key	Value	Type
▼ (1) ObjectId("663617368b08aa337852d...") { 5 fields }	{ 5 fields }	Object
_id	ObjectId("663617368b08aa337852d9a2")	ObjectId
NUMVEHICULE	9.0	Double
NUMIMMAT	3087319233	String
ANNEE	1992.0	Double
> MODELE	{ 3 fields }	Object
▼ (2) ObjectId("663617368b08aa337852d...") { 5 fields }	{ 5 fields }	Object
_id	ObjectId("663617368b08aa337852d9a4")	ObjectId
NUMVEHICULE	11.0	Double
NUMIMMAT	1572319801	String
ANNEE	1998.0	Double
> MODELE	{ 3 fields }	Object

```
> use projet
switched to db projet
> db.createCollection("VEHICULE")
{ "ok" : 1 }
> db.VEHICULE.find({"MODELE.MARQUE.NOMMARQUE": "PORSCHE"}).count()
2
>
```

2- Récupérer dans une nouvelle collection Véhicules Interventions, les matricules des véhicules et le nombre total de s interventions par véhicule ; la collection devra être ordonnée par ordre décroissant du nombre des interventions :

```
db.VEHICULE.aggregate([
{
  $match: { // from
    INTERVENTIONS: { $exists: true } //pour récupérer les véhicules qui ont des interventions
  }
},
{
  $project: {
    NUMIMMAT: 1, // pour récupérer le matricule des vehicules
    NBINTERVENTIONS: { $size: "$INTERVENTIONS" } // pour récupérer le nombre total des interventions
  }
},
{
  $sort: { NBINTERVENTIONS: -1 } //sort=utiliser pour ordonner une collection dans ce cas ordonner de maniere décroissante le nombre des interventions avec -1
},
{
  $out: "vehicules_interventions" //nom de la nouvelle collection qui contient les immatricules des vehicules et le nombre d intervention de cette vehicule
},
]);
```

The screenshot shows the MongoDB Compass interface. On the left, the 'Collections' list includes 'System', 'VEHICULE', and 'vehicules\_interventions'. The 'vehicules\_interventions' collection is selected. The main panel displays the aggregation pipeline: `db.VEHICULE.aggregate([ { $match: { INTERVENTIONS: { $exists: true } } }, { $project: { NUMIMMAT: 1, NBINTERVENTIONS: { $size: '$INTERVENTIONS' } } }, { $sort: { NBINTERVENTIONS: -1 } }, { $out: 'vehicules_interventions' } ])`. Below the pipeline, the results are shown in a table with columns 'Key', 'Value', and 'Type'. The table contains 13 rows, each representing a vehicle with its ID and a set of 3 fields.

Key	Value	Type
> (1) ObjectId("663617368b08aa337852d99a")	{ 3 fields }	Object
> (2) ObjectId("663617368b08aa337852d9ad")	{ 3 fields }	Object
> (3) ObjectId("663617368b08aa337852d99b")	{ 3 fields }	Object
> (4) ObjectId("663617368b08aa337852d99c")	{ 3 fields }	Object
> (5) ObjectId("663617368b08aa337852d99f")	{ 3 fields }	Object
> (6) ObjectId("663617368b08aa337852d9a1")	{ 3 fields }	Object
> (7) ObjectId("663617368b08aa337852d9a3")	{ 3 fields }	Object
> (8) ObjectId("663617368b08aa337852d9a7")	{ 3 fields }	Object
> (9) ObjectId("663617368b08aa337852d9aa")	{ 3 fields }	Object
> (10) ObjectId("663617368b08aa337852d9ae")	{ 3 fields }	Object
> (11) ObjectId("663617368b08aa337852d9af")	{ 3 fields }	Object
> (12) ObjectId("663617368b08aa337852d9b2")	{ 3 fields }	Object
> (13) ObjectId("663617368b08aa337852d9b5")	{ 3 fields }	Object



3-Dans une collection véhicule bcp\_pannes, récupérer les véhicules dont le nombre des interventions dépasse 6 pannes : ( il y'a pas qui dépasse 2pannes)

```
db.VEHICULE.aggregate([
  {
    $match: {
      "INTERVENTIONS": { $exists: true },
    }
  },
  {
    $project: {
      "NUMVEHICULE": 1,
      "NUMIMMAT": 1,
      "NBINTERVENTION": { $size: "$INTERVENTIONS" }
    }
  },
  {
    $match: {
      "NBINTERVENTION": { $gte: 2 }
    }
  },
  {
    $out: "vehicule_bcp_pannes" // Écrire les résultats dans la collection "vehicule_bcp_pannes"
  }
]);
```

The screenshot shows the MongoDB Compass interface. On the left, the 'Collections' list includes 'vehicule\_bcp\_pannes'. The main panel displays the results of the aggregation query. The query bar shows `db.getCollection('vehicule_bcp_pannes').find({})`. The results table shows two documents:

Key	Value	Type
(1) ObjectId("663617368b08aa337852d99a")	{ 4 fields }	Object
_id	ObjectId("663617368b08aa337852d99a")	ObjectId
NUMVEHICULE	1.0	Double
NUMIMMAT	0012519216	String
NBINTERVENTION	2	Int32
(2) ObjectId("663617368b08aa337852d9ad")	{ 4 fields }	Object
_id	ObjectId("663617368b08aa337852d9ad")	ObjectId
NUMVEHICULE	20.0	Double
NUMIMMAT	1234319707	String
NBINTERVENTION	2	Int32

#### 4- Récupérer dans une collection employe-interv, toutes les interventions d'un employé :

```
db.VEHICULE.aggregate([
{
  $unwind: "$INTERVENTIONS" // Décompose le tableau "interventions" pour avoir un document par intervention
},
{
  $unwind: "$INTERVENTIONS.EMPLOYE" // Décompose le tableau "employes" à l'intérieur de chaque intervention pour avoir un document par employé
},
{
  $group: { // Regroupe les documents par numéro d employé
    _id: "$INTERVENTIONS.EMPLOYE.NUMEMPLOYE", // Clé de regroupement basée sur le numéro d employé
    NBINTERVENTIONS: { $sum: 1 }, // Compte le nombre d interventions pour chaque employé
    INTERVENTIONS: { $push: "$INTERVENTIONS" }, // Ajoute chaque intervention à un tableau pour chaque employé
  }
},
{
  $project: { // Restructure les documents de sortie
    _id: 0, // Supprime le champ "_id" par défaut
    NUMEMPLOYER: "$_id", // Crée un champ "NUMEMPLOYER" à partir de la clé de regroupement "_id"
    NBINTERVENTIONS: 1, // Conserve le champ "NBINTERVENTIONS"
    INTERVENTIONS: 1, // Conserve le champ "INTERVENTIONS"
  }
},
{
  $sort: { NUMEMPLOYER: 1 } //ordonner les numéro d employé par ordre croissant
},
{
  $out: "employeinterv" // écrit les résultats dans la collection "employeinterv"
}
]);
```

The screenshot shows the MongoDB Compass interface. On the left, the 'Collections' list includes 'VEHICULE' and 'employeinterv'. The 'employeinterv' collection is selected, showing 14 documents. The top toolbar displays the aggregation pipeline: `* db.vehicules.aggregate([ { $unwind: "$INTERVENTIONS" } ])`. The main panel shows the results of the aggregation, with columns for Key, Value, and Type. The results are 14 documents, each containing an ObjectId, a numeric value (1-14), and an array of 4 fields.

Key	Value	Type
> (1) ObjectId("66362465acb800c3878c32c4")	{ 4 fields }	Object
> (2) ObjectId("66362465acb800c3878c32c5")	{ 4 fields }	Object
> (3) ObjectId("66362465acb800c3878c32c6")	{ 4 fields }	Object
> (4) ObjectId("66362465acb800c3878c32c7")	{ 4 fields }	Object
> (5) ObjectId("66362465acb800c3878c32c8")	{ 4 fields }	Object
> (6) ObjectId("66362465acb800c3878c32c9")	{ 4 fields }	Object
> (7) ObjectId("66362465acb800c3878c32ca")	{ 4 fields }	Object
> (8) ObjectId("66362465acb800c3878c32cb")	{ 4 fields }	Object
> (9) ObjectId("66362465acb800c3878c32cc")	{ 4 fields }	Object
> (10) ObjectId("66362465acb800c3878c32cd")	{ 4 fields }	Object
> (11) ObjectId("66362465acb800c3878c32ce")	{ 4 fields }	Object
> (12) ObjectId("66362465acb800c3878c32cf")	{ 4 fields }	Object
> (13) ObjectId("66362465acb800c3878c32d0")	{ 4 fields }	Object
> (14) ObjectId("66362465acb800c3878c32d1")	{ 4 fields }	Object

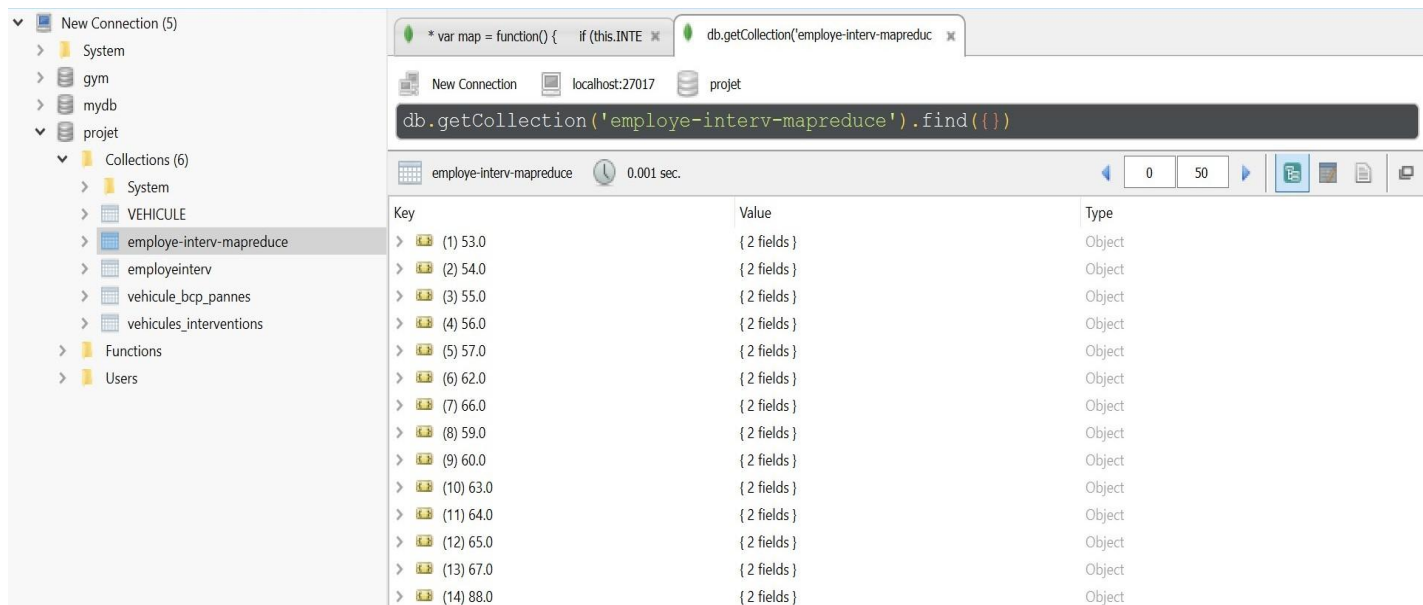
5- Augmenter de 8000DA, le salaire des employés de catégorie « Mécanicien» :

```
db.VEHICULE.updateMany(  
  { "INTERVENTIONS.EMPLOYE.CATEGORIE": "Mécanicien" },  
  { $inc: { "interventions.$[].EMPLOYE.$[emp].salaire": 8000 } },  
  { arrayFilters: [{ "emp.CATEGORIE": "Mécanicien" }] }  
);
```

## 6- Reprendre la 4ième requête à l'aide du paradigme Map-Reduce:

```
var map = function() {
  if (this.INTERVENTIONS) {
    this.INTERVENTIONS.forEach(function(INTERVENTIONS) {
      INTERVENTIONS.EMPLOYE.forEach(function(EMPLOYE) {
        emit(EMPLOYE.NUMEMPLOYE, { interventions: [INTERVENTIONS] });
      });
    });
  }
};

var reduce = function(NUMEMPLOYE, values) {
  var result = { interventions: [] };
  values.forEach(function(value) {
    result.interventions = result.interventions.concat(value.interventions);
  });
  return result;
};
// la collection dans laquelle le résultat sera placé
db.VEHICULE.mapReduce(
  map,
  reduce,
  {
    out: "employe-interv-mapreduce"
  }
);
```



Key	Value	Type
> (1) 53.0	{ 2 fields }	Object
> (2) 54.0	{ 2 fields }	Object
> (3) 55.0	{ 2 fields }	Object
> (4) 56.0	{ 2 fields }	Object
> (5) 57.0	{ 2 fields }	Object
> (6) 62.0	{ 2 fields }	Object
> (7) 66.0	{ 2 fields }	Object
> (8) 59.0	{ 2 fields }	Object
> (9) 60.0	{ 2 fields }	Object
> (10) 63.0	{ 2 fields }	Object
> (11) 64.0	{ 2 fields }	Object
> (12) 65.0	{ 2 fields }	Object
> (13) 67.0	{ 2 fields }	Object
> (14) 88.0	{ 2 fields }	Object

## **D. Analyse par rapport à ces requêtes**

Ces requêtes permettent de récupérer et manipuler les données de la collection "véhicule" de différentes manières :

1. Filtrer les véhicules par marque.
2. Créer une nouvelle collection avec les matricules et le nombre d'interventions par véhicule, triée par ordre décroissant du nombre d'interventions.
3. Créer une nouvelle collection avec les véhicules ayant plus de 6 interventions.
4. Créer une nouvelle collection regroupant les interventions par employé.
5. Mettre à jour les salaires des employés de catégorie "Mécanicien".
6. Utiliser le paradigme Map-Reduce pour regrouper les interventions par employé.

Ces requêtes font appel à différentes opérations MongoDB comme `find`, `aggregate`, `project`, `unwind`, `group out`, `updateMany` et `mapReduce`. Elles permettent de filtrer, projeter, regrouper et transformer les données de la collection d'origine pour répondre à différents besoins d'analyse et de reporting.