



source : istockphoto

PROJET DE BASE DE DONNÉES

Agence de location de voitures

DEVISME Claire
GARCIA Claire

Mme Kessaci

Table des matières

I – Présentation du projet	4
Contexte	4
Analyse.....	4
Données à calculer : calculs de coût.....	5
Difficultés du projet.....	6
II – Le modèle conceptuel	7
III - Le modèle relationnel	9
Voiture	9
Révision.....	9
Nettoyage.....	9
Location.....	10
Réservation.....	10
Situation	10
IV - Les requêtes de créations de table BaseVoit.sql	11
CATEGORIE	11
EMPLACEMENT.....	11
MARQUE	12
VOITURE	13
SITUATION.....	13
NETTOYAGE	14
REVISION.....	14
VILLE	14
CONDUCTEUR.....	15
LOCATION	16
RESERVATION	16
V – Organisation des pages web	18
VI – Guide d'utilisation	19
Accueil page.html	19
Réserver pour la 1ère fois client.php Reserver.php.....	20
Réserver en étant déjà client DejaClient.php ReserveDC.php	22
Retourner un véhicule Retour.php Validation.php Facture.php	23
Ajouter un véhicule AjoutVoit.php AjoutVoit2.php.....	24
Supprimer un véhicule SupVoit.php	25

Entretien d'une voiture Entretien.html	26
VI – Explication des requêtes de la page AjoutVoit2.php.....	27
VII – Bilan personnel	28
Claire DEVISME	28
Claire GARCIA	28

I – Présentation du projet

Contexte

Dans le but d'améliorer sa gestion de location de voitures, l'entreprise X nous a demandé de créer une application permettant de gérer plus facilement ses locations. Celle-ci permettra de gérer à la fois les réservations, les locations, mais aussi l'entretien du véhicule, c'est-à-dire le nettoyage ainsi que la révision.

Nous avons donc réalisé le cahier des charges suivant, qui décrit notre démarche.

Analyse

Chaque voiture sera caractérisée par son numéro d'immatriculation, son nombre de kilomètres au compteur, son nombre de places – entre 2 et 9 places maximum –, sa marque, sa catégorie – citadine, berline, monospace/SUV, minibus cités par ordre croissant d'importance –, son emplacement dans le parc, son nettoyage et sa révision.

On recueille les dates de révision et de nettoyage de chaque voiture pour gérer l'entretien des voitures du parc. On pourra notamment éditer une liste mensuelle des véhicules dont on prévoit la révision. Une voiture sera envoyée à la révision chaque fois qu'elle effectue au moins 20 000 nouveaux kilomètres par rapport au kilomètres au compteur. La voiture est de ce fait bloquée au début du mois qui suit et ceci pour le mois entier. La réservation sera impossible sur une période située entre 2 mois dans ce cas (par exemple du 29 au 2). Une voiture est de même placée au nettoyage à chaque 5 locations mais n'implique aucun blocage pour la journée, le nettoyage sera réalisé dès le lendemain du retour, avant l'ouverture de l'agence. En outre, un entretien, tout aussi bien physique que technique, peut être nécessaire en fonction de son état lors du retour de la location.

Une voiture pourra être garée à un nouvel emplacement à chaque retour de location. Mais en emplacement peut être vide si aucune voiture n'y est garée. La révision aussi bien que le nettoyage, ne nécessite pas de modifier l'emplacement où le véhicule est garé, soit celui-ci est fait sur place, soit l'emplacement est conservé en cas d'éventuel retour d'entretien.

De l'autre côté se situe le client et conducteur, il est défini par un numéro de permis, un nom, un prénom, une date de naissance, une adresse, une commune liée à son code postal, un mail, un numéro de téléphone portable et de fixe ainsi qu'un numéro d'identité pouvant correspondre soit au numéro de carte d'identité, soit au numéro de passeport.

Pour effectuer une réservation, le client remplit un formulaire de recherche d'après les critères qu'il souhaite : il choisit une catégorie de voiture – parmi citadine, berline, monospace/SUV, minibus –, un nombre de place souhaité au minimum – entre 2 et 9 –, la date de départ de sa location et son retour – au plus tard la réservation peut

être faite le jour même et/ou sur place –, ainsi que le nombre de kilomètres qu'il prévoit de faire durant la période de location sélectionnée. Suite à cette requête, le client obtient une liste de véhicules potentiellement louables (ou pas) – si aucun véhicule n'est disponible en référence à ces critères, on affichera l'impossibilité de réservation –. Par contre, s'il a réservé un véhicule après un autre client qui a dépassé les délais, il ne pourra pas louer le véhicule effectivement prévu pour lui. L'agence devra alors lui trouver un autre véhicule correspondant « au moins » à ses critères. En effet, il sera surclassé, et ceci sans surcoût, en cas de non disponibilité du véhicule recherché. De plus, aucune réservation ne sera disponible si entre les 2 dates prévues, le nombre de véhicules loués est supérieur à 70% du nombre de véhicules total composant le parc.

Une fiche de location pourra être établie le jour de la récupération du véhicule, précisant le nom du client, la catégorie et la marque de la voiture louée, son kilométrage actuel, la durée prévue de la location en jours, son prix à la journée – qui dépend du nombre de kilomètres prévus par le client et de la catégorie du véhicule loué, sauf cas de surclassement, dans ce cas on prendra le prix de la catégorie recherchée par le client –, le coût prévisionnel total suivant le nombre de kilomètres prévus sur la durée prévue de location, l'emplacement auquel le véhicule est rangé permettant au client de le retirer (par exemple à la place A18) ainsi que le montant de l'acompte versé – correspondant au montant d'une journée de location, qui dépend de la catégorie de véhicule –.

Au retour de la location, une facture est établie précisant en plus de la fiche de location, le nombre de kilomètres effectivement parcourus et le prix à payer – correspondant au prix à la journée pour la catégorie de véhicule loué, ou recherché si application de la règle de surclassement il y a, multiplié par le nombre de jours prévus –. Un surcoût sera appliqué pour un client ayant loué un véhicule pour une période supérieure à celle prévue – on appliquera un surcoût de 20% par rapport au prix à la journée du véhicule, sur les journées additionnelles –.

Données à calculer : calculs de coût

➤ On établira le prix d'une location de la manière suivante :

- $\text{SURCOUT} = \text{prixKm} \times 0.2 \times (\text{nbKmEff} - \text{nbKmPrev})$
- $\text{COUT}_{\text{PREV}} = \text{prixKm} \times \text{nbKmPrev}$
- $\text{ACOMPTE} = \text{prixKm} \times (\text{nbKmPrev} / (\text{dateRetourPrev} - \text{dateDebut}))$

$\text{PRIX}_{\text{total}} = \text{COUT}_{\text{PREV}} - \text{ACOMPTE} (+ \text{SURCOUT})$
--

Le prix total sera calculé à partir du coût prévisionnel, duquel on déduira l'acompte et auquel on pourra ajouter l'éventuel surcoût.

- **On a pour cela établi le montant au kilomètre pour chaque catégorie de véhicule :**

	Catégorie de Véhicule	Prix du kilomètre	Prix du kilomètre supplémentaire (+20%)
1	Citadine	0.58 €	+ 0.70 €
2	Berline	0.76 €	+ 0.91 €
3	Monospace / SUV	0.87 €	+ 1.04 €
4	Minibus	1.29 €	+ 1.55 €

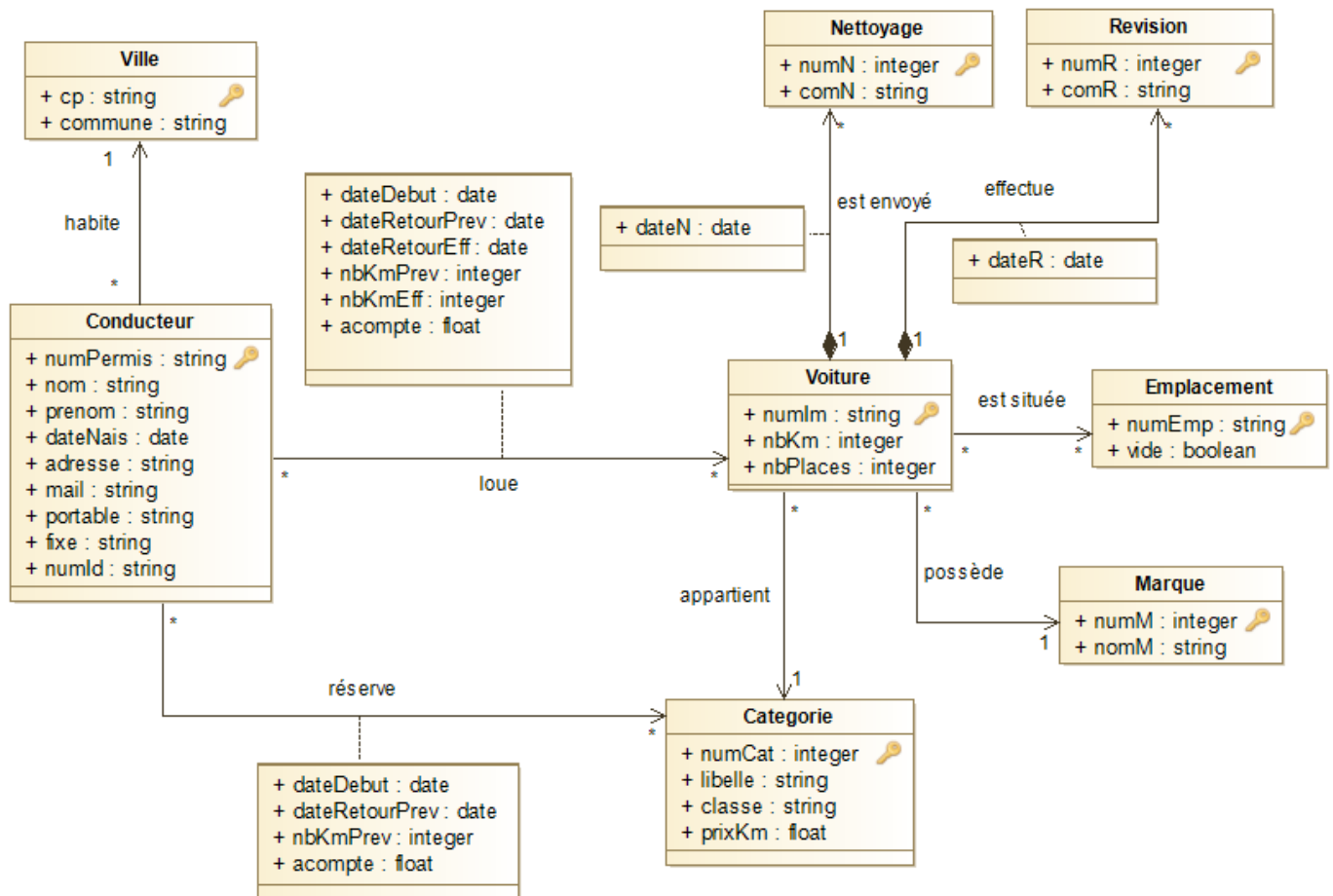
Tableau décrivant les prix kilométriques en fonction de la catégorie de véhicule

Difficultés du projet

Il va nous falloir constituer de zéro une application, à partir du peu d'information donné mais aussi recueilli et réaliser notre base de données ainsi que notre site web.

II – Le modèle conceptuel

A l'aide de l'analyse effectuée sur le fonctionnement de l'entreprise dans la gestion de ses différents véhicules et des informations recueillies auprès de ses membres, nous avons pu établir le schéma conceptuel ci-dessous.



Une **Voiture** possède comme identifiant un numéro d'immatriculation unique (string). On lui attribue un nombre de places (entier) et un kilométrage (entier).

Une voiture appartient à une et une seule catégorie. Une catégorie peut en revanche correspondre à plusieurs voitures. D'où la cardinalité (1, *) entre Catégorie et Voiture. Nous avons donc créé la classe **Catégorie** avec comme identifiant NumCat (string) et comme attribut, son libellé (caractère), sa classe (entier), son prix au kilomètre (réel).

Une voiture possède de plus une marque mais une marque peut disposer de différents modèles de voitures donc aussi de différentes voitures. Une **Marque** sera caractérisée par un numéro (string) et un nom (string).

On crée aussi la classe **Emplacement** pour la situer sur le parking. Celle-ci permettra de savoir si l'emplacement est vide ou non. Elle contiendra les informations suivantes : le numéro d'emplacement (string) comme clé primaire et sa disponibilité (booléen).

Une voiture peut être envoyée à la révision ou au nettoyage mais à une seule révision ou nettoyage correspond une unique voiture. D'où la cardinalité (1, *) entre Voiture et Révision ou entre Voiture et Nettoyage. De plus, on ne conserve les révisions et nettoyages que si la voiture existe encore au sein du garage. D'où la composition entre Voiture et chaque entretien respectivement.

Pour la **Révision**, on inscrira un numéro d'identification (string), on précisera sa date (date) et éventuellement un commentaire décrivant les réparations à effectuer. La classe ainsi créée permettra à l'entreprise d'éditer une liste mensuelle des véhicules qui nécessitent une révision, qui se fera tous les 20 000 km. Les véhicules suivants seront donc bloqués dès lors et ceci jusqu'à la fin du mois suivant.

De même, pour le **Nettoyage**, on conservera un numéro permettant l'identification (string), on recueillera la date (date) et potentiellement un commentaire précisant les directives principales.

On doit aussi recueillir les informations du **Conducteur** pour que l'entreprise ai accès au fichier clients. On utilisera comme clé son numéro de permis (string) qui permet d'avoir un identifiant unique pour chaque utilisateur et de même format quel que soit la nationalité du potentiel client (contrairement au numéro d'identité). On recueille aussi son nom (string), son prénom (string), sa date de naissance (date), son adresse, son adresse mail (string), son portable ainsi que son numéro de fixe (string pour permettre la prise en compte du premier caractère étant principalement '0', par exemple pour 0362545895). Un numéro d'identité sera de même donné (numéro de carte d'identité ou passeport).

On crée de plus la classe **Ville** pour éviter les doublons de villes pour les clients provenant d'un même endroit (on aura notamment une majorité de personnes aux alentours de l'agence qui feront partie du fichier client). On identifiera celle-ci tout naturellement par son code postal (string) et on prendra aussi son nom de commune (string).

Un conducteur peut louer plusieurs voitures, de même une voiture peut être associée à plusieurs conducteurs pour une date donnée. D'où les cardinalités respectives (*, *) entre le Conducteur et la Voiture. On conservera à chaque location du client, les dates de début et de fin prévue et effective (date), le nombre de kilomètres à la fois prévu et effectif (entier), le montant de l'acompte (réel). On pourra éditer une fiche de location correspondante.

Un conducteur peut réserver plusieurs catégories et à une catégorie est associé plusieurs conducteurs. On recueillera les dates de début et de retour prévisionnel (date), le nombre de kilomètres prévus (entier) ainsi que l'acompte (réel).

Une location engendre dans tous les cas une facture qu'il faudra éditer.

III - Le modèle relationnel

Grâce au modèle conceptuel précédent, on a établi le modèle relationnel qui suit :

Categorie (numcat, libelle, classe, prixKm)

Emplacement (numEmp, vide)

Marque (numM, nomM)

Voiture (numlm, nbKm, nbPlaces, numCat#, numM#)

Situation (numlm#, numEmp#)

Nettoyage (numN, numlm#, comN, dateN)

Revision (numR, numlm#, comR, dateR)

Ville (cp, commune)

Conducteur (numPermis, nom, prenom, dateNais, adresse, mail, portable, fixe, numld, cp#)

Location (numPermis#, numlm#, dateDebut, dateRetourPrev, dateRetourEff, nbKmPrev, nbKmEff, acompte)

Reservation (numPermis#, numCat#, dateDebut, dateRetourPrev, nbKmPrev, acompte)

Voiture

La classe **Voiture** possède deux clés étrangères, numCat et numEmp, car elle est reliée à la classe **Catégorie** avec une cardinalité (1, *) et reliée à la table emplacement avec une cardinalité (1, 0..1).

Révision

La classe **Révision** possède une clé étrangère en référence à la clé primaire numlm de Voiture, étant liée à la classe Voiture. Elle forme avec numR la clé primaire de la Révision car les 2 classes sont liées par leur cycle de vie, il s'agit d'une relation de composition, on a la cardinalité (1, *).

Nettoyage

La classe **Nettoyage** possède de même une clé étrangère en référence à la clé primaire numlm de Voiture, étant liée à la classe Voiture. Elle aussi forme avec numR la clé primaire de la Révision car les 2 classes sont liées par leur cycle de vie, il s'agit d'une relation de composition, on a la cardinalité (1, *).

Location

De plus, on crée la relation **Location** qui possède deux clés étrangères, numPermis, numIm car on a une cardinalité (*, *) entre la classe Voiture (d'où la clé primaire associée numPermis) et la classe Conducteur (d'où la clé primaire associée numIm). Ces deux clés constituent la clé primaire de la relation Location ainsi créée.

Réservation

Enfin on crée la relation **Réservation** qui possède deux clés étrangères, numPermis, numIm, elle aussi, car on a une cardinalité (*, *) entre la classe Voiture (d'où la clé primaire associée numPermis) et la classe Conducteur (d'où la clé primaire associée numIm).

Ces deux clés constituent la clé primaire de la relation Réservation ainsi créée.

Situation

On crée en plus la relation **Situation** pour créer un lien entre Voiture et emplacement. Celle-ci possèdera donc les clés primaires de chacune des classes auxquelles elle est reliée, elles constitueront la clé primaire de la nouvelle classe Situation ainsi créée.

IV - Les requêtes de créations de table | BaseVoit.sql

Lors de la première version, nous avons omis les contraintes de domaine (CHECK), nous avons seulement précisés les NOT NULL. Il fut de même pour les contraintes d'intégrité (ON DELETE CASCADE), que nous avons par la suite rajouté.

```
DROP TABLE if exists REVISION cascade;
DROP TABLE if exists NETTOYAGE cascade;
DROP TABLE if exists SITUATION cascade;
DROP TABLE if exists VOITURE cascade;
DROP TABLE if exists EMPLACEMENT cascade;
DROP TABLE if exists CATEGORIE;
DROP TABLE if exists CONDUCTEUR;
DROP TABLE if exists LOCATION;
DROP TABLE if exists VILLE;
DROP TABLE if exists MARQUE;
DROP TABLE if exists RESERVATION;
```

CATEGORIE

```
--
-- Structure de la table CATEGORIE
--
CREATE TABLE CATEGORIE (
  numCat VARCHAR(5) NOT NULL,
  libelle VARCHAR(20) NOT NULL,
  classe VARCHAR(30) NOT NULL,
  prixKm FLOAT NOT NULL,
  CHECK (prixKm > 0),
  PRIMARY KEY (numCat)
);
--
-- Contenu de la table CATEGORIE
--
insert into CATEGORIE values ('CAT1', 'Minibus', 'Classe1', 1.29);
insert into CATEGORIE values ('CAT2', 'Monospace/SUV', 'Classe2', 0.87);
insert into CATEGORIE values ('CAT3', 'Berline', 'Classe3', 0.76);
insert into CATEGORIE values ('CAT4', 'Citadine', 'Classe4', 0.58);
```

EMPLACEMENT

```
--
-- Structure de la table EEMPLACEMENT
```

```
--
CREATE TABLE EMPLACEMENT (
numEmp VARCHAR (5) NOT NULL,
vide BOOLEAN NOT NULL,
PRIMARY KEY (numEmp)
);
--
-- Contenu de la table EMPLACEMENT
--
insert into EMPLACEMENT values ('E01', '0');
insert into EMPLACEMENT values ('E02', '1');
insert into EMPLACEMENT values ('E03', '0');
insert into EMPLACEMENT values ('E04', '0');
insert into EMPLACEMENT values ('E05', '0');
insert into EMPLACEMENT values ('E06', '1');
insert into EMPLACEMENT values ('E07', '0');
insert into EMPLACEMENT values ('E08', '0');
insert into EMPLACEMENT values ('E09', '0');
insert into EMPLACEMENT values ('E10', '0');
insert into EMPLACEMENT values ('E11', '1');
insert into EMPLACEMENT values ('E12', '1');
insert into EMPLACEMENT values ('E13', '1');
insert into EMPLACEMENT values ('E14', '1');
```

MARQUE

```
--
-- Structure de la table MARQUE
--
CREATE TABLE MARQUE (
numM VARCHAR NOT NULL,
nomM VARCHAR (30) NOT NULL,
PRIMARY KEY (numM)
);
--
-- Contenu de la table MARQUE
--
insert into MARQUE values ('M1', 'Peugeot');
insert into MARQUE values ('M2', 'Renault');
insert into MARQUE values ('M3', 'Citroën');
insert into MARQUE values ('M4', 'Fiat');
insert into MARQUE values ('M5', 'BMW');
```

VOITURE

```
--  
-- Structure de la table VOITURE  
--  
CREATE TABLE VOITURE (  
  numIm VARCHAR (10) NOT NULL,  
  nbKm INTEGER NOT NULL,  
  nbPlaces INTEGER NOT NULL,  
  numCat VARCHAR constraint numCat references CATEGORIE(numCat) on delete cascade,  
  numM VARCHAR constraint numM references MARQUE(numM) on delete cascade,  
  PRIMARY KEY (numIm)  
);  
--  
-- Contenu de la table VOITURE  
--  
insert into VOITURE values ('AA-316-BN', 14600,4,'CAT1','M1');  
insert into VOITURE values ('BV-528-XR', 11000,5,'CAT2','M2');  
insert into VOITURE values ('PR-622-VC', 15000,5,'CAT1','M3');  
insert into VOITURE values ('OL-824-PC', 3300,7,'CAT3','M5');  
insert into VOITURE values ('MK-928-TA', 9800,7,'CAT2','M3');  
insert into VOITURE values ('SX-026-PK', 10000,9,'CAT4','M4');  
insert into VOITURE values ('UY-864-PL', 15000,9,'CAT3','M1');  
insert into VOITURE values ('HV-432-RX', 6400,9,'CAT4','M4');
```

SITUATION

```
--  
-- Structure de la table SITUATION  
--  
CREATE TABLE SITUATION (  
  numIm VARCHAR constraint numIm references VOITURE(numIm) on delete cascade,  
  numEmp VARCHAR constraint numEmp references EMPLACEMENT(numEmp) on delete cascade,  
  PRIMARY KEY (numIm,numEmp) );  
--  
-- Contenu de la table SITUATION  
--  
insert into SITUATION values ('AA-316-BN','E01');  
insert into SITUATION values ('BV-528-XR','E03');  
insert into SITUATION values ('PR-622-VC','E04');  
insert into SITUATION values ('OL-824-PC','E05');  
insert into SITUATION values ('MK-928-TA','E07');  
insert into SITUATION values ('SX-026-PK','E09');  
insert into SITUATION values ('UY-864-PL','E08');  
insert into SITUATION values ('HV-432-RX','E10');
```

NETTOYAGE

```
--  
-- Structure de la table NETTOYAGE  
--  
CREATE TABLE NETTOYAGE (  
  numN VARCHAR NOT NULL,  
  dateN DATE,  
  comN VARCHAR (300),  
  numIm VARCHAR constraint numIm references VOITURE(numIm) on delete cascade,  
  PRIMARY KEY (numN,numIm)  
);  
--  
-- Contenu de la table NETTOYAGE  
--  
insert into NETTOYAGE values ('N001',TO_DATE('02/02/2018','DD/MM/YYYY'),NULL,'HV-432-RX');  
insert into NETTOYAGE values ('N002',TO_DATE('21/04/2018','DD/MM/YYYY'),NULL,'MK-928-TA');  
insert into NETTOYAGE values ('N003',TO_DATE('06/04/2018','DD/MM/YYYY'),NULL,'PR-622-VC');  
insert into NETTOYAGE values ('N004',TO_DATE('13/09/2018','DD/MM/YYYY'),NULL,'BV-528-XR');  
insert into NETTOYAGE values ('N005',TO_DATE('15/06/2018','DD/MM/YYYY'),NULL,'UY-864-PL');
```

REVISION

```
--  
-- Structure de la table REVISION  
--  
CREATE TABLE REVISION (  
  numR VARCHAR NOT NULL,  
  dateR DATE,  
  comR VARCHAR (300),  
  numIm VARCHAR constraint numIm references VOITURE(numIm) on delete cascade,  
  PRIMARY KEY (numR,numIm)  
);  
--  
-- Contenu de la table REVISION  
--  
insert into REVISION values ('R1',TO_DATE('28/08/2018','DD/MM/YYYY'),NULL,'AA-316-BN');  
insert into REVISION values ('R2',TO_DATE('15/06/2018','DD/MM/YYYY'),NULL,'SX-026-PK');  
insert into REVISION values ('R3',TO_DATE('03/04/2018','DD/MM/YYYY'),NULL,'UY-864-PL');  
insert into REVISION values ('R4',TO_DATE('05/02/2018','DD/MM/YYYY'),NULL,'SX-026-PK');  
insert into REVISION values ('R5',TO_DATE('13/09/2018','DD/MM/YYYY'),NULL,'OL-824-PC');
```

VILLE

```
--
```

```

-- Structure de la table VILLE
--
CREATE TABLE VILLE (
cp VARCHAR (10) NOT NULL,
commune VARCHAR (50) NOT NULL,
PRIMARY KEY (cp)
);
--
-- Contenu de la table VILLE
--
insert into VILLE values ('33000','Bordeaux');
insert into VILLE values ('31000','Toulouse');
insert into VILLE values ('59000','Lille');
insert into VILLE values ('69000','Lyon');
insert into VILLE values ('13000','Marseille');
insert into VILLE values ('64000','Pau');
insert into VILLE values ('35000','Rennes');

```

CONDUCTEUR

```

--
-- Structure de la table CONDUCTEUR
--
CREATE TABLE CONDUCTEUR (
numPermis VARCHAR (20) NOT NULL,
nom VARCHAR (50) NOT NULL,
prenom VARCHAR (30) NOT NULL,
dateNais DATE NOT NULL,
adresse VARCHAR (50) NOT NULL,
mail VARCHAR (30) NOT NULL,
portable VARCHAR (12) NOT NULL,
fixe VARCHAR (12),
numId VARCHAR (30) NOT NULL,
cp VARCHAR constraint cp references VILLE(cp) on delete cascade,
PRIMARY KEY (numPermis)
);
--
-- Contenu de la table CONDUCTEUR
--
insert into CONDUCTEUR values
('C1','Robert','Albert',TO_DATE('06/05/1968','DD/MM/YYYY'),'11 rue
Fouchet','robert.albert@gmail.com','0612345678',NULL,'01','33000');
insert into CONDUCTEUR values
('C2','Henri','Berry',TO_DATE('13/09/1988','DD/MM/YYYY'),'46 rue
Mout','henry.berry@gmail.com','0612345699',NULL,'02','59000');

```

insert into CONDUCTEUR values

('C3','Carla','Amstrong',TO_DATE('24/12/1978','DD/MM/YYYY'),'37 rue André','carla.amstrong@gmail.com','0600345699',NULL,'03','69000');

LOCATION

--

-- Structure de la table LOCATION

--

CREATE TABLE LOCATION (

dateDebut **DATE NOT NULL,**

dateretourPrev **DATE NOT NULL,**

dateretourEff **DATE NOT NULL,**

nbKmPrev **INTEGER NOT NULL,**

nbKmEff **INTEGER NOT NULL,**

acompte **FLOAT,**

numPermis **VARCHAR constraint** numPermis **references** CONDUCTEUR(numPermis) **on delete cascade,**

numIm **VARCHAR constraint** numIm **references** VOITURE(numIm) **on delete cascade,**

PRIMARY KEY (numPermis,numIm)

);

--

-- Contenu de la table LOCATION

--

insert into LOCATION values

(TO_DATE('12/05/2018','DD/MM/YYYY'),TO_DATE('15/09/2018','DD/MM/YYYY'),TO_DATE('15/09/2018','DD/MM/YYYY'),**2000,2150,NULL,'C2','AA-316-BN');**

insert into LOCATION values

(TO_DATE('12/02/2018','DD/MM/YYYY'),TO_DATE('15/03/2018','DD/MM/YYYY'),TO_DATE('15/04/2018','DD/MM/YYYY'),**200,200,NULL,'C3','OL-824-PC');**

insert into LOCATION values

(TO_DATE('01/01/2018','DD/MM/YYYY'),TO_DATE('01/12/2018','DD/MM/YYYY'),TO_DATE('11/12/2018','DD/MM/YYYY'),**100000,100300,NULL,'C1','PR-622-VC');**

RESERVATION

--

-- Structure de la table RESERVATION

--

CREATE TABLE RESERVATION (

dateDebut **DATE NOT NULL,**

dateretourPrev **DATE NOT NULL,**

nbKmPrev **INTEGER NOT NULL,**

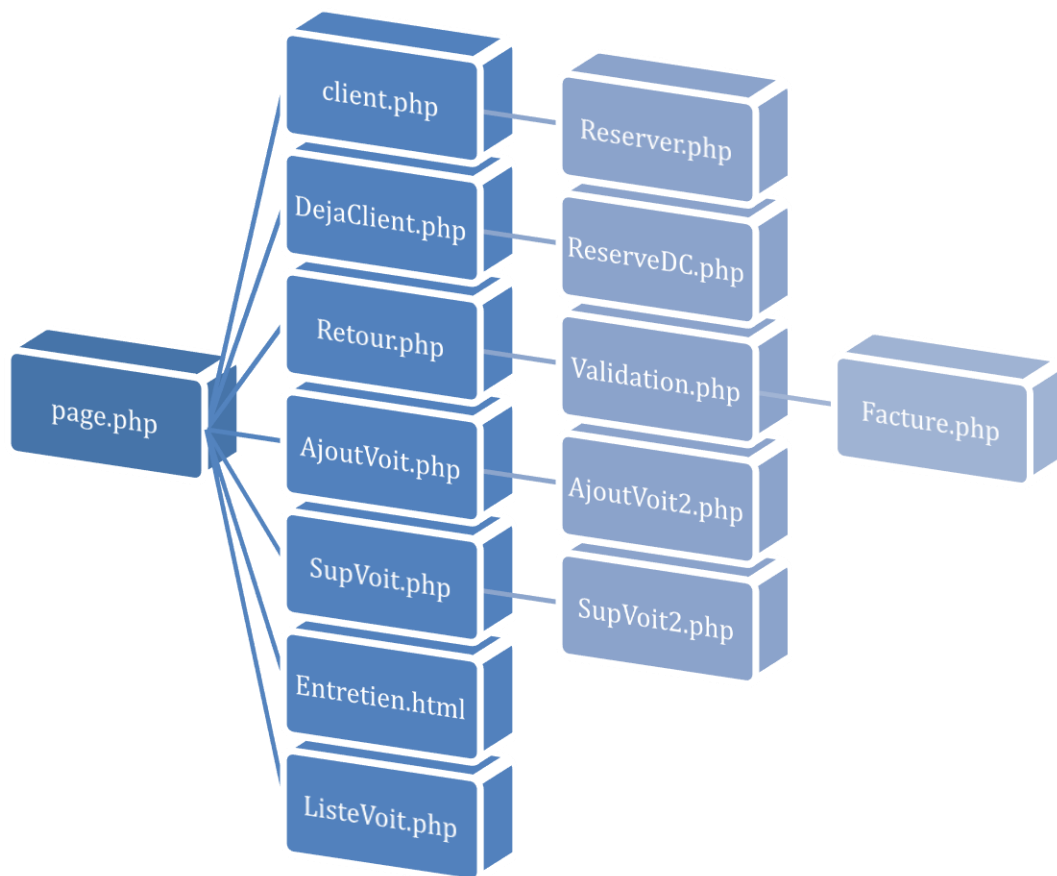
acompte **FLOAT,**


```

numPermis VARCHAR constraint numPermis references CONDUCTEUR(numPermis) on
delete cascade,
numCat VARCHAR constraint numCat references CATEGORIE(numCat) on delete cascade,
CHECK (nbKmPrev>0),
PRIMARY KEY (numPermis,numCat)
);
--
-- Contenu de la table RESERVATION
--
insert into RESERVATION values
(TO_DATE('12/05/2019','DD/MM/YYYY'),TO_DATE('15/09/2019','DD/MM/YYYY'),2000,NULL,
'C2','CAT1');
insert into RESERVATION values
(TO_DATE('12/05/2020','DD/MM/YYYY'),TO_DATE('15/09/2020','DD/MM/YYYY'),2000,NULL,
'C1','CAT1');

```

V – Organisation des pages web



VI – Guide d'utilisation

Accueil | page.html



← → ↻ ⓘ houplin.studserv.deule.net/~cgarcia/projet/page.html

Bonjour!

Bienvenue dans notre agence de location

Réserver pour la première fois Je suis déjà client

Retour d'une voiture

Ajouter une voiture Supprimer une voiture Entretien

Mes voitures

Notre page d'accueil permet d'établir le menu des différentes actions possibles :

- Réserver un véhicule en tant que nouveau client
- Réserver un véhicule avec mon numéro client
- Retourner un véhicule
- Ajouter un véhicule
- Supprimer un véhicule
- Inventorier la liste des entretiens d'un véhicule

Réserver pour la 1ère fois | [client.php](#) [Reserver.php](#)

← → ↻ houplin.studserv.deule.net/~cgarcia/projet/client.php

Bonjour client!

Enregistrement client

Nom Prénom

Date de Naissance

Número carte d'identité ou de passeport

Número de permis

Adresse

N° et libellé de voie

Code postal Ville

Coordonnées

Mail

Número de portable

Número de fixe

Critères réservation

Nombre de places souhaites minimum : ▼

Categorie : ☒ Citadine ☐ Berline ☐ Monospace/Suv ☐ Minibus

Date depart : Date retour :

Nombre de kilometres prévus : km

Lors de sa première réservation, le client doit saisir les informations demandées dans le cadre “Enregistrement client” dans le formulaire ci-dessus. Il s’agit de ses données personnelles, qui seront ensuite intégrées à la base de données de l’entreprise. Le client n’aura donc plus besoin de les resaisir. Seul son numéro de permis lui sera demandé pour l’identifier.

Une fois ses informations saisies, le client choisit ses critères de réservation dans le cadre correspondant.

Prenons l'exemple ci-dessous, Madame Fabienne EME a bien rempli les informations la concernant. Elle aura comme identifiant par la suite 'C55'.

← → ↻ ⓘ Non sécurisé | houplin.studserv.deule.net/~cgarcia/projet/client.php

Bonjour client!

Enregistrement client

Nom Prénom

Date de Naissance

Numéro carte d'identité ou de passeport

Numéro de permis

Adresse

N° et libellé de voie

Code postal Ville

Coordonnées

Mail

Numéro de portable

Numéro de fixe

Critères réservation

Nombre de places souhaites minimum :

Catégorie : ☐ Citadine ☐ Berline ☐ Monospace/Suv ☒ Minibus

Date depart Date retour

Nombre de kilometres prévus : km

Une fois le formulaire rempli, le client clique sur le bouton 'Reserver'. Il est ensuite redirigé vers la page [Reserver.php](#) où les informations saisies seront prises en compte et ajoutées à notre base de données. De plus, une réservation sera créée en fonction de la catégorie de véhicules choisie. Ajouté à cela, un prix approximatif de cette location est calculé à partir de la catégorie du véhicule souhaité et du nombre de kilomètre parcouru.

Sur notre exemple, Madame Fabienne EME a choisi un minibus pour réaliser environ 300 km. Le prix de sa location est donc de 900 euros.

← → ↻ ⓘ houplin.studserv.deule.net/~cgarcia/projet/Reserver.php

Client ajoute

Reservation ajoute

Le prix total est de '900' euros

Réserver en étant déjà client | DejaClient.php ReserveDC.php

← → ↻ Non sécurisé | houplin.studserv.deule.net/~cgarcia/projet/DejaClient.php

Bonjour client!

Enregistrement client

Nom

Prénom

Numéro de permis

Critères réservation

Nombre de places souhaites minimum :

Categorie : ☐ Citadine ☐ Berline ☐ Monospace/Suv ☐ Minibus

Date depart : Date retour :

Nombre de kilometres prevus : km

Reserver

Si un client a déjà fait une réservation de véhicule dans notre entreprise, son nom et ses informations personnelles sont déjà présentes dans la base. Il lui suffit de rentrer son nom, son prénom ainsi que son numéro de permis pour pouvoir retrouver les informations le concernant.

Puis pour réaliser sa réservation, il renseigne ses critères. Il clique ensuite sur le bouton 'Reserver' situé en bas de la page pour valider.

Par exemple, Madame Fabienne EME, pour sa deuxième réservation, saisit seulement son nom, son prénom et son numéro de permis. Puis elle sélectionne ses critères de réservation.

← → ↻ houplin.studserv.deule.net/~cgarcia/projet/DejaClient.php

Bonjour client!

Enregistrement client

Nom

Prénom

Numéro de permis

Critères réservation

Nombre de places souhaites minimum :

Categorie : ☒ Citadine ☐ Berline ☐ Monospace/Suv ☐ Minibus

Date depart : Date retour :

Nombre de kilometres prevus : km

Reserver

Le client est suite à cela redirigé sur la page [ReserveDC.php](#) où la réservation est prise en compte grâce aux critères choisis. De plus comme précédemment, le calcul du prix est estimé.

La deuxième réservation de Madame Fabienne EME va coûter environ 180 euros.

Retourner un véhicule | [Retour.php](#) [Validation.php](#) [Facture.php](#)

← → ↻ ⓘ houplin.studserv.deule.net/~cgarcia/projet/Retour.php

Merci d avoir utilisé notre service

Enregistrement retour

Numéro de permis

Date depart : Date retour prevus :

Nombre de kilometres prevus : km

Date retour effective :

Nombre de kilometres réellement parcourus

Quel emplacement

Besoin nettoyage : ☐

Besoin révision : ☐

Au retour de la location, il est nécessaire d'enregistrer le véhicule et les informations concernant la location qui a eu lieu. On proposera aussi un éventuel nettoyage ou une éventuelle révision en cas de besoin occasionnel. On validera ensuite le retour.

On est ensuite emmené sur le page 'Validation.php' qui nous confirme notre retour et qui attribue les informations de la location dans la table location dans la base de données. De plus un bouton 'Facture' nous permet d'obtenir une facture à partir du code de la page 'Facture.php'.

Le code de ces pages proposant encore des d'erreurs, nous ne pouvons pas vous proposer d'exemple.

Ajouter un véhicule | AjoutVoit.php AjoutVoit2.php

← → ↻ ⓘ houplin.studserv.deule.net/~cgarcia/projet/AjoutVoit.php

Bienvenue

Ajouter une voiture

Numéro d'immatriculation

Nombre kilomètre

Nombre places

Marque

Categorie : ☐ Citadine ☐ Berline ☐ Monospace/Suv ☐ Minibus

Emplacement libre

Ajouter

Lors de l'achat de nouveaux véhicules, l'entreprise a besoin d'ajouter des nouvelles voitures à sa base de données. On a pour cela créé l'onglet 'Ajouter une voiture'. On entre donc les caractéristiques de la voiture à ajouter dans le formulaire. Puis on choisit un emplacement pour la nouvelle voiture parmi les emplacements libres. En effet, une requête sélectionne les emplacements vides du parc de l'entreprise.

Par exemple l'entreprise souhaite ajouter une voiture immatriculée 'GH-456-YU'. Elle remplit les informations sur le véhicule puis choisit de le placer sur l'emplacement E11.

← → ↻ ⓘ Non sécurisé | houplin.studserv.deule.net/~cgarcia/projet/AjoutVoit.php

Bienvenue

Ajouter une voiture

Numéro d'immatriculation

Nombre kilomètre

Nombre places

Marque

Categorie : ☐ Citadine ☒ Berline ☐ Monospace/Suv ☐ Minibus

Emplacement libre

Ajouter

En cliquant sur 'Ajouter', nous sommes redirigés sur '**AjoutVoit2.php**'. Les informations précédemment saisies sont ajoutées à la base de l'entreprise dans la classe voiture. Un premier lien est créé entre marque et voiture et un deuxième est créé entre voiture et emplacement dans l'association situation. C'est ce qui va

permettre de changer la valeur du booléen de l'attribut 'vide' dans la classe emplacement. En effet, l'emplacement choisi précédemment va devenir occupé.

Dans notre exemple notre voiture immatriculée 'GH-456-YU' est ajoutée à notre base ainsi que ses caractéristiques. De plus, l'emplacement E11 est dorénavant occupé par cette nouvelle voiture.



← → ↻ ⓘ houplin.studserv.deule.net/~cgarcia/projet/AjoutVoit2.php

Voiture ajoutée

Supprimer un véhicule | SupVoit.php



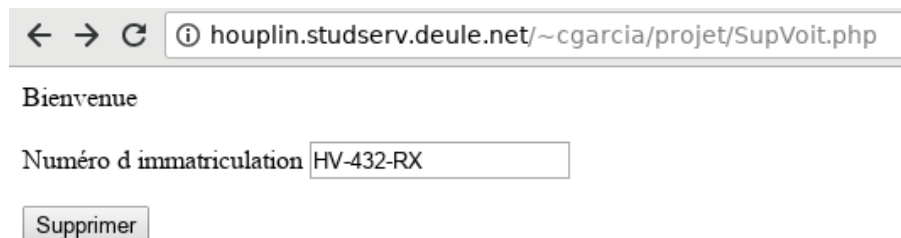
← → ↻ ⓘ houplin.studserv.deule.net/~cgarcia/projet/SupVoit.php

Bienvenue

Numéro d immatriculation

L'entreprise peut avoir également besoin de supprimer un de ses véhicules. Pour cela elle a besoin du numéro d'immatriculation de la voiture qu'elle souhaite supprimer.

Par exemple, si elle souhaite supprimer la voiture immatriculée 'HV-432-RX'.



← → ↻ ⓘ houplin.studserv.deule.net/~cgarcia/projet/SupVoit.php

Bienvenue

Numéro d immatriculation

En cliquant sur 'supprimer', la page '**SupVoit2.php**' va s'exécuter et supprimer de la base, la voiture correspondant au numéro d'immatriculation saisi ainsi que toutes les relations existantes avec celle-ci. De plus, l'emplacement sur lequel était situé le véhicule deviendra disponible.

Dans notre exemple, le véhicule était situé sur l'emplacement 'E14'. Cet emplacement est à présent disponible.



← → ↻ ⓘ houplin.studserv.deule.net/~cgarcia/projet/SupVoit2.php

Voiture Supprimée

L'emplacement 'E14' est disponible

Liste des véhicules | ListeVoit.php

← → ↻ ⓘ houplin.studserv.deule.net/~cgarcia/projet/ListeVoit.php

La liste des voiture

Numero immatriculation	Nb kilomètre	nb Place	Categorie
OL-824-PC	3300	7	Berline
HV-432-RX	3000	4	Citadine
AA-316-BN	14600	4	Minibus
PR-622-VC	15000	5	Minibus
BV-528-XR	11000	5	Monospace/SUV
MK-928-TA	9800	7	Monospace/SUV

L'entreprise peut également afficher tous les véhicules qu'elle possède. Elles sont classées par catégorie.

Entretien d'une voiture | Entretien.html

← → ↻ ⓘ houplin.studserv.deule.net/~cgarcia/projet/Entretien.html

Bienvenue

Numéro d'immatriculation

Initialement, l'entreprise devait pouvoir gérer les entretiens de ses véhicules. Par manque de temps, les codes liés à cette requête n'ont pas été réalisés.

VI – Explication des requêtes de la page AjoutVoit2.php

Expliquons plus en détails la page SupVoit2.php :

A partir du numéro d'immatriculation d'une voiture nous voulons supprimer celle-ci ainsi que toutes les associations liées à cette voiture. De plus, nous devons rendre disponible l'emplacement sur lequel était le véhicule.

Nous commençons donc par rechercher sur le numéro d'emplacement de la voiture :

```
"Select numEmp from SITUATION where situation.numIm='".$numIm.'";"
```

Nous rangeons ensuite cette valeur dans la variable \$numeroemp.

```
$nEmp=pg_query($base, $Emp);  
$numemp = pg_fetch_array($nEmp);  
$numeroemp=$numemp['numemp'];
```

Une fois cette requête exécutée, il faut changer la valeur de l'attribut 'vide' qui correspond au booléen de la disponibilité de l'emplacement (f=false, l'emplacement est disponible, t=true, l'emplacement est vide).

```
$ChgtEmp="update EMPLACEMENT set vide='t' where numemp='$numeroemp';";
```

Enfin, nous pouvons supprimer le véhicule souhaité avec la requête :

```
" Delete from VOITURE where numIm='".$numIm.'";"
```

Tous les liens avec la voiture supprimée vont être supprimés grâce au 'ON DELETE CASCADE' dans la base de données.

VII – Bilan personnel

Claire DEVISME

Pour ma part, les cours de Base de Données et ce projet m'ont fait découvrir de nouvelles choses étant donné que je n'avais aucune connaissance dans ce domaine. J'ai beaucoup aimé la matière de par sa logique mais je rencontre encore pas mal de difficultés, c'est pourquoi le projet fût bénéfique. Le travail en binôme nous a permis de réaliser un projet certes non abouti, mais en partie fonctionnel.

De plus, le projet permet de se rendre compte de l'utilisation complète des notions vues ce semestre : conception et analyse avec création du modèle conceptuel puis traduction en modèle relationnel, et ensuite la réalisation avec la création de la base de données puis la rédaction des scripts php en intégrant les requêtes sql.

Claire GARCIA

Nous avons passé beaucoup de temps à l'analyse des besoins au détriment de la conception. C'est pour cela que nous présentons aujourd'hui un projet en cours et non terminé. J'en ai donc tiré la conclusion que la gestion du temps est très importante lors d'un travail en groupe.

Le travail en groupe permet d'élargir nos connaissances et nos horizons. En effet, je trouve intéressant de pouvoir croiser plusieurs réflexions et d'aller plus loin dans l'analyse.

Ce projet m'a également apporté les compétences de base du php qui me seront utiles dans mon futur professionnel.