



---

## Annexe

---

---

### 3.1] La Base de données

#### A) Présentation de l'existant

#### B) Test à effectuer

---

---

### 3.2] Besoin de la MS2R

- La Base de données

#### A) La Base De Données

La Base de données est comme on l'a précisé dans la partie précédente « incomplète », on ne peut toujours pas faire le lien entre les postes de la MS2R ainsi que les logiciels présents sur ceux-ci, et il n'y avait par ailleurs aucun moyen de différencier chaque poste en fonction de son service. Il a fallu donc créer de nouvelles tables après avoir ajouté quelques nouveaux postes et nouveaux logiciels en relation avec la mission 1 et 2.

Ses nouvelles tables contiendront toutes les données concernant ces informations manquantes et aussi de faire les liens entre les logiciels et les postes tout en faisant en sorte qu'un poste puisse avoir plusieurs logiciels mais aussi de créer une table qui contiendra les informations de chaque service, voici donc comment on a procédé pour la création de ses nouvelles tables

#### B) Les Tables

##### La Table SERVICE

- Elle doit avoir une clé primaire qui correspondra à un numéro permettant de faire le lien entre les postes et cette table
- Elle doit contenir les informations concernant le bâtiment de ce service (sa location) ;
- Elle doit aussi contenir le nom du service auquel elle est associée ;

On créera donc la Table SERVICE qui accueillera toutes ses informations

---

```
CREATE TABLE IF NOT EXISTS `service` (  
  `NumService` INT(1) NOT NULL AUTO_INCREMENT,  
  `NomService` VARCHAR(15) CHARACTER SET cpl251 NOT NULL,  
  `Batiment` VARCHAR(20) COLLATE utf8mb4_unicode_ci NOT NULL,  
  PRIMARY KEY (`NumService`),  
  KEY `NumService` (`NumService`)  
) ENGINE=INNODB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci AUTO_INCREMENT=4 ;
```

Ensuite on lui intègre les informations des postes :

```
INSERT INTO `service` (`NumService`, `NomService`, `Batiment`) VALUES  
(1, 'Administration', 'A 1er étage'),  
(2, 'Logistique', 'A 2ème étage'),  
(3, 'Projet', 'B 1er étage');
```

Enfin on peut effectuer le lien direct avec la table Poste en lui ajoutant un nouveau champ qui contiendra l'information du service pour chaque poste tout en spécifiant que le champ Numservice ne devra pas être contraire à la clé primaire de la table POSTE :

```
ALTER TABLE poste  
ADD `NumService` INT(1) NOT NULL  
KEY `NumService` (`NumService`);  
  
ALTER TABLE `poste`  
ADD CONSTRAINT `poste_ibfk_1`  
FOREIGN KEY (`NumService`)  
REFERENCES `service` (`NumService`);
```

Après avoir effectué ses modifications pour l'insertion de la table service il ne restait plus qu'à assimiler chaque poste à un service dans cette Base de données (grâce à la requête SQL : UPDATE).

## La Table INSTALLER

- Elle permettra de faire le lien entre la table POSTE et LOGICIEL
- Elle doit permettre d'insérer plusieurs logiciels différents pour un poste

Cette table INSTALLER a été créée dans le but d'assimiler plusieurs logiciels différents à un seul poste. En effet si l'on aurait simplement créé un champ dans la table POSTE en clé étrangère celui-ci n'aurait pu accueillir qu'un seul information (donc un seul logiciel), il a donc fallu créer une table concaténant permettant de faire une base de données plus « logique ».

Pour commencer on a créé la table INSTALLER ayant pour clé primaire IdLogiciel et IdPoste afin de subvenir aux besoins comme on l'a expliqué au-dessus :

```
CREATE TABLE IF NOT EXISTS `installer` (  
  `IdPoste` INT(5) NOT NULL,  
  `IdLogiciel` INT(5) NOT NULL,  
  PRIMARY KEY (`IdPoste`, `IdLogiciel`),  
  KEY `IdLogiciel` (`IdLogiciel`)  
) ENGINE=INNODB DEFAULT CHARSET=latin1;
```

Ensuite après la création de la table INSTALLER, on effectue les contraintes d'intégrité référentielle entre les deux tables POSTE et LOGICIEL :

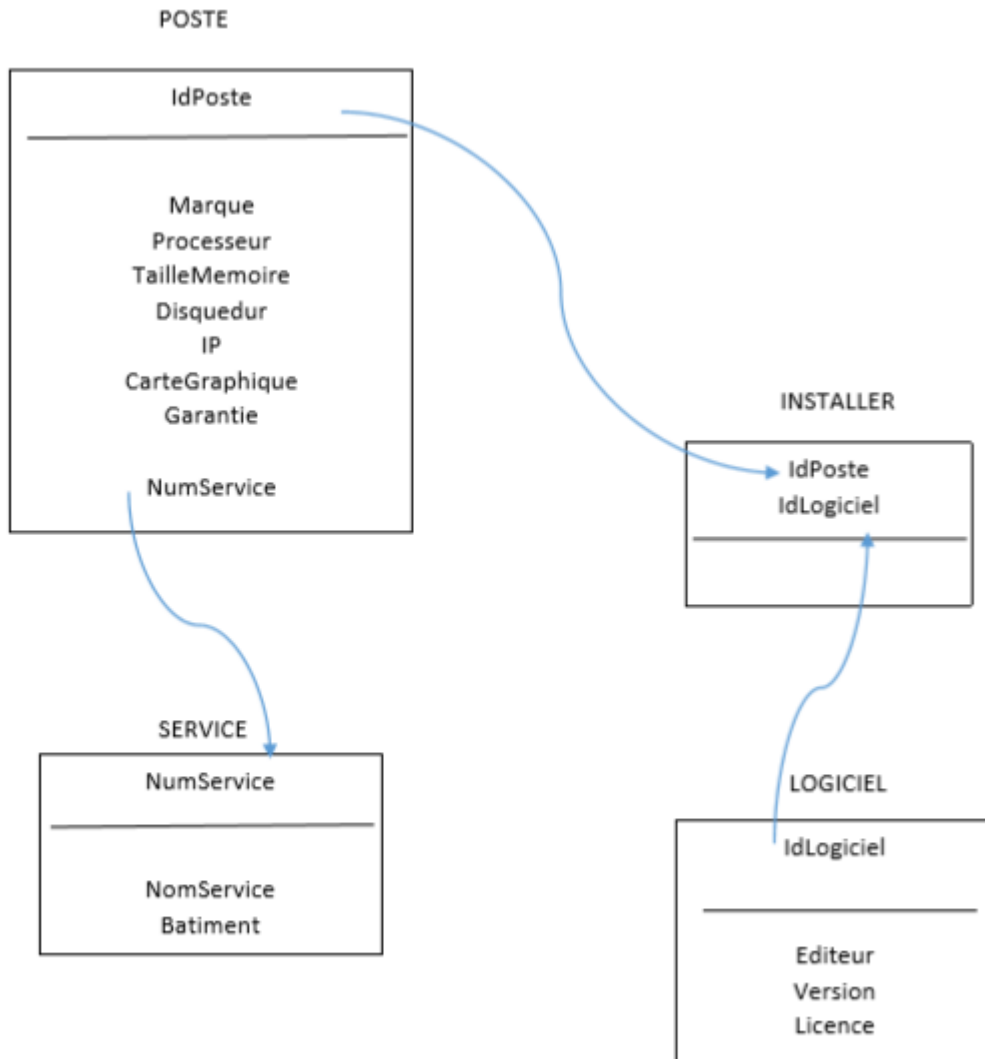
```
ALTER TABLE `installer`  
  ADD CONSTRAINT `installer_ibfk_2`  
  FOREIGN KEY (`IdLogiciel`)  
  REFERENCES `logiciel` (`IdLogiciel`)  
  ON DELETE CASCADE ON UPDATE CASCADE,  
  
  ADD CONSTRAINT `installer_ibfk_3` FOREIGN KEY  
  (`IdPoste`) REFERENCES `poste` (`IdPoste`)  
  ON DELETE CASCADE ON UPDATE CASCADE;
```

Et enfin on lui insère des données en faisant bien attention que ses données correspondent à ID de poste ou de logiciel existant :

```
INSERT INTO `installer` (`IdPoste`, `IdLogiciel`) VALUES  
(1, 10),  
(2, 10),  
(3, 10),  
(16, 10),  
(17, 10),  
(18, 10),  
(26, 10),  
(4, 20),
```

La base de données est donc après ses requêtes SQL prêtes à accueillir les informations des postes de la MS2R et totalement prête pour la création de l'application dans la Mission 4.

### C) Schéma de la BDD



#### D) Requête SQL

### Bilan de la Mission 3

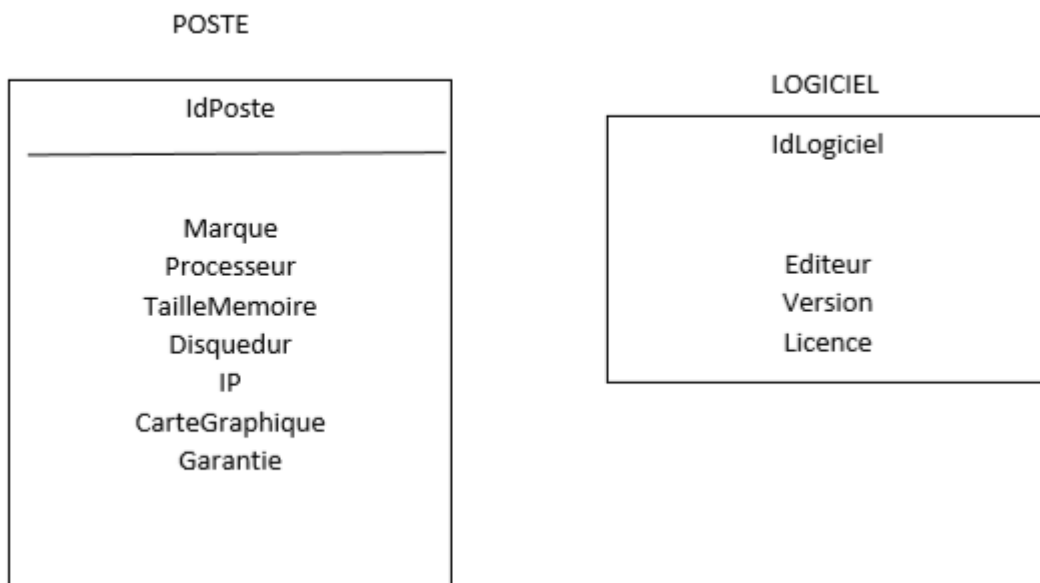
### 3.1] La Base de données

#### C) Présentation de l'existant

La base de données nous a été donnée avec deux tables concernant les postes ainsi que les logiciels présents sur ceux-ci.

Cette Base de données devait donc être revue car il n'y avait aucun lien entre les tables, ce qui est contraire au principe d'une Base de données.

Cette Base de données se présente ainsi :



Comme on peut le constater il n'y a aucun lien entre les logiciels et les postes, on ne peut donc pas savoir quels logiciels ont été installés sur quelle poste par exemple.

## D) Test à effectuer

Malgré le fait que la BDD n'était pas complète, il a fallu faire quelques requêtes afin de voir si celle-ci était opérationnel. Voilà donc les trois requêtes à effectuer ainsi que leurs résultats sous SQL Yog qui été demandé dans le cahier des charges :

- La première requête devait retourner la liste détaillée des postes qui ne sont pas pourvus de processeur de type Intel, ce qui nous amener as effectuer cette requête :

```
SELECT *  
FROM poste  
WHERE Processeur NOT LIKE "%Intel%";
```

Après avoir effectué cette requête sous SQLYog on obtenait ce résultat qui nous donnais tous les postes n'ayant pas un processeur Intel ainsi que toutes les informations des postes :

IdPoste	Marque	Processeur	TailleMemoire	DisqueDur	IP	CarteGraphique
8	Apple	A6X	6 Go	64 Go	172.16.0.8	Puce graphique
9	Compaq	Athlon II P360	4 Go	320 Go	172.16.0.9	ATI Mobility Rad
10	Compaq	Athlon II P360	8 Go	600 Go	172.16.15.254	ATI Mobility Rad

- La deuxième requête devait quant à elle donner la liste détaillée des postes qui ont au plus 4Go de RAM ou 500 Go de HD, on effectuer donc cette requête :

```
SELECT *  
FROM poste  
WHERE TailleMemoire<=4 OR DisqueDur <=500  
GROUP BY Marque;
```

On obtenait par la suite ce résultat qui nous donnais donc les postes qui ont 500Go de disque dur ou plus de 4Go de RAM ainsi que les informations e chaque poste correspondant à cette recherche :

IdPoste	Marque	Processeur	TailleMemoire	DisqueDur	IP	CarteGraphique
1	Acer	Intel i5	4 Go	500 Go	172.16.0.1	NVIDIA Quadro 1000M
8	Apple	A6X	6 Go	64 Go	172.16.0.8	Puce graphique
4	Asus	Intel i5	6 Go	500 Go	172.16.0.4	Mobile Intel HM77
9	Compaq	Athlon II P360	4 Go	320 Go	172.16.0.9	ATI Mobility Radeon

- La troisième requête devait elle nous donner le nombre de logiciels recensés par éditeur, ce qui donner cette requête :

```
SELECT Editeur, COUNT(idlogiciel)
FROM logiciel
GROUP BY Editeur;
```

On obtenait par la suite ce résultat qui nous donné le nombre de logiciels présent dans cette BDD pour chaque éditeur, on avait par exemple pour le système d'exploitation Debian 2 logiciels au total pour cette édition :

Editeur	COUNT(idlogiciel)
Apple	1
Debian	2
Kaspersky	1
Microsoft	4
Opera	1
pfSense	1
Ubuntu	2
(NULL)	(NULL)

Après avoir effectué ces requêtes SQL on pouvait donc distinguer l'incohérence de la Base de données et le travail à effectuer pour que celle-ci soit utilisable pour la réalisation de l'application de gestion des postes (Mission 4)

## 3.2] Besoin de la MS2R

### E) La Base De Données

La Base de données est comme on l'a précisé dans la partie précédente « incomplète », on ne peut toujours pas faire le lien entre les postes de la MS2R ainsi que les logiciels présents sur ceux-ci, et il n'y avait par ailleurs aucun moyen de différencier chaque poste en fonction de son service.

Il a fallu donc créer de nouvelles tables après avoir ajouté quelques nouveaux postes et nouveaux logiciels en relation avec la mission 1 et 2.

Ses nouvelles tables contiendront toute les données concernant ces informations manquantes et aussi de faire les liens entre les logiciels et les postes tout en faisant en sorte qu'un poste puisse avoir plusieurs logiciels mais aussi de créer une table qui contiendrai les informations de chaque service, voici donc comment on a procédé pour la création de ses nouvelles tables



## F) Les Tables

### La Table SERVICE

- Elle doit avoir une clé primaire qui correspondra à un numéro permettant de faire le lien entre les postes et cette table
- Elle doit contenir les informations concernant le bâtiment de ce service (sa location) ;
- Elle doit aussi contenir le nom du service auquel elle est associée ;

On créera donc la Table SERVICE qui accueillera toute ses informations

```
CREATE TABLE IF NOT EXISTS `service` (  
  `NumService` INT(1) NOT NULL AUTO_INCREMENT,  
  `NomService` VARCHAR(15) CHARACTER SET cpl251 NOT NULL,  
  `Batiment` VARCHAR(20) COLLATE utf8mb4_unicode_ci NOT NULL,  
  PRIMARY KEY (`NumService`),  
  KEY `NumService` (`NumService`)  
) ENGINE=INNODB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci AUTO_INCREMENT=4 ;
```

Ensuite on lui intègre les informations des postes :

```
INSERT INTO `service` (`NumService`, `NomService`, `Batiment`) VALUES  
(1, 'Administration', 'A 1er étage'),  
(2, 'Logistique', 'A 2ème étage'),  
(3, 'Projet', 'B 1er étage');
```

Enfin on peut effectuer le lien direct avec la table Poste en lui ajoutant un nouveau champ qui contiendra l'information du service pour chaque poste tout en spécifiant que le champ Numservice ne devra pas être contraire à la clé primaire de la table POSTE :

```
ALTER TABLE poste  
ADD `NumService` INT(1) NOT NULL  
KEY `NumService` (`NumService`);  
  
ALTER TABLE `poste`  
ADD CONSTRAINT `poste_ibfk_1`  
FOREIGN KEY (`NumService`)  
REFERENCES `service` (`NumService`);
```

Après avoir effectué ses modifications pour l'insertion de la table service il ne restait plus qu'à assimiler chaque poste à un service dans cette Base de données (grâce à la requête SQL : UPDATE).

### La Table INSTALLER

- Elle permettra de faire le lien entre la table POSTE et LOGICIEL
- Elle doit permettre d'insérer plusieurs logiciels différents pour un poste

Cette table INSTALLER a été créée dans le but d'assimiler plusieurs logiciels différents à un seul poste. En effet si l'on aurait simplement créé un champ dans la table POSTE en clé étrangère celui-ci n'aurait pu accueillir qu'une seule information (donc un seul logiciel), il a donc fallu créer une table concaténant permettant de faire une base de données plus « logique ».

Pour commencer on a créé la table INSTALLER ayant pour clé primaire IdLogiciel et IdPoste afin de subvenir aux besoins comme on l'a expliqué au-dessus :

```
CREATE TABLE IF NOT EXISTS `installer` (  
  `IdPoste` INT(5) NOT NULL,  
  `IdLogiciel` INT(5) NOT NULL,  
  PRIMARY KEY (`IdPoste`,`IdLogiciel`),  
  KEY `IdLogiciel` (`IdLogiciel`)  
) ENGINE=INNODB DEFAULT CHARSET=latin1;
```

Ensuite après la création de la table INSTALLER, on effectue les contraintes d'intégrité référentielle entre les deux tables POSTE et LOGICIEL :

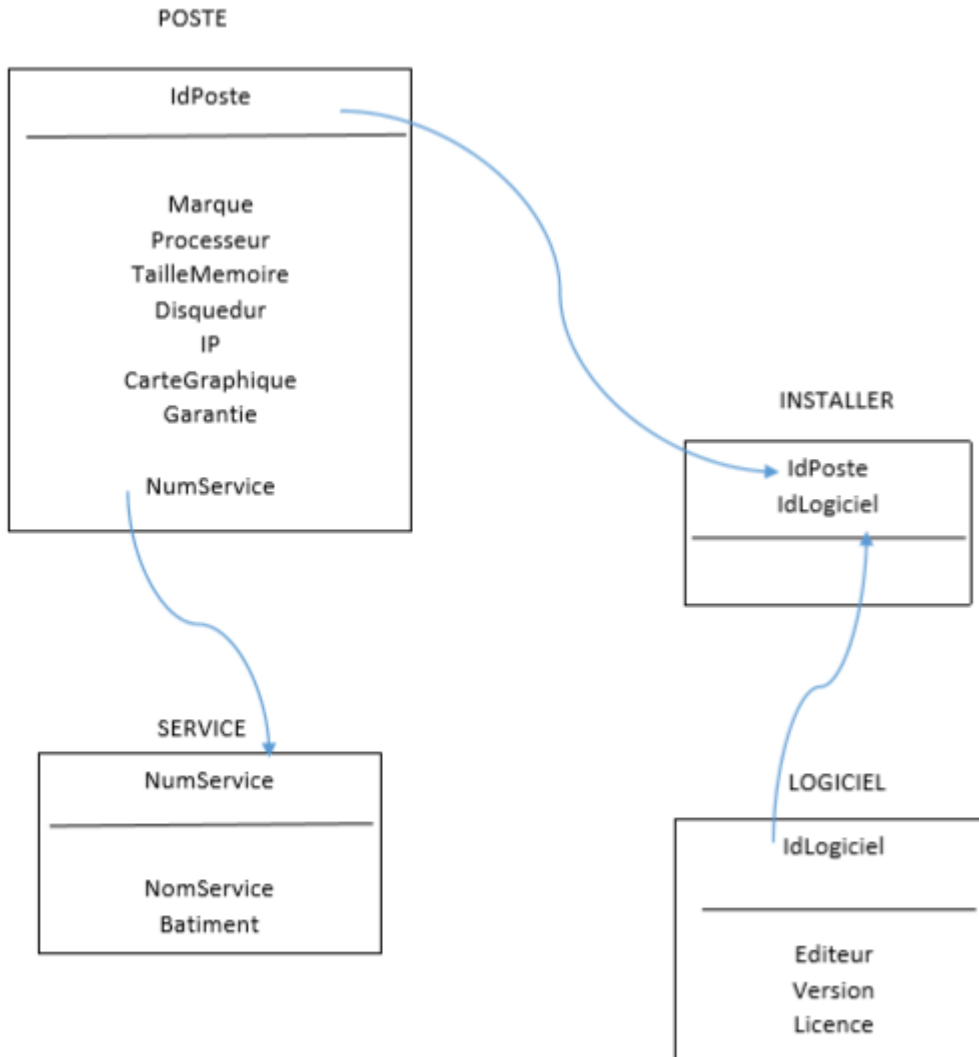
```
ALTER TABLE `installer`  
  ADD CONSTRAINT `installer_ibfk_2`  
  FOREIGN KEY (`IdLogiciel`)  
  REFERENCES `logiciel` (`IdLogiciel`)  
  ON DELETE CASCADE ON UPDATE CASCADE,  
  
  ADD CONSTRAINT `installer_ibfk_3` FOREIGN KEY  
  (`IdPoste`) REFERENCES `poste` (`IdPoste`)  
  ON DELETE CASCADE ON UPDATE CASCADE;
```

Et enfin on lui insère des données en faisant bien attention que ses données correspondent à ID de poste ou de logiciel existant :

```
INSERT INTO `installer` (`IdPoste`, `IdLogiciel`) VALUES  
(1, 10),  
(2, 10),  
(3, 10),  
(16, 10),  
(17, 10),  
(18, 10),  
(26, 10),  
(4, 20),
```

La base de données est donc après ses requêtes SQL prête à accueillir les informations des postes de la MS2R et totalement prête pour la création de l'application dans la Mission 4.

## G) Schéma de la BDD



#### H) Requête SQL

Maintenant que la Base de données est prête on peut effectuer les requêtes demandés dans le cahier des charges, voici donc les deux requêtes permettant de vérifier que la base de données est bel et bien terminée et fonctionnel :

- La première requête demande était de donner l'éditeur et la version des logiciels installés sur le poste préparé à la mission 1, ce poste étant le poste N°999 :

```
SELECT Version, Editeur  
FROM poste INNER JOIN installer
```

```
ON poste.IdPoste=installer.IdPoste
INNER JOIN logiciel
ON installer.IdLogiciel=logiciel.IdLogiciel
WHERE Poste.IdPoste=999 ;
```

Ce qui nous donné donc ce résultat ou l'on avait tous les logiciels présents sur la machine virtuelle préparer à la mission 1 :

Version	Editeur
Windows 10 Professional	Microsoft
PDF Creator 3.3.0.15261	pdfforge GmbH
Firefox 18.5.0.0	Mozilla
OpenOffice	Microsoft
Libre Office 6.0	Libre Document Fonda
Thunderbird 18.5.0.0	Mozilla
(NULL)	(NULL)

- La deuxième requête était de donné le nombre de logiciels installés par poste :










```
SELECT poste.IdPoste, COUNT(logiciel.IdLogiciel) AS "Nombre de logiciels par poste"
FROM poste INNER JOIN installer
ON poste.IdPoste=installer.IdPoste
INNER JOIN logiciel
ON installer.IdLogiciel=logiciel.IdLogiciel
GROUP BY poste.IdPoste
```

On avait par la suite ce résultat nous donnant donc pour chaque poste son nombre de logiciels y étant installé :

IdPoste	Nombre de logiciels par poste
11	5
12	5
13	5
14	5
15	5
16	5
17	5
18	5
19	1
20	1
21	5

## Bilan de la Mission 3

Le travail sur la Base de données as été terminé en fonction du cahier des charges, en voici un rappel du cahier des charges avec les tâches qui ont été réalisé lors du travail sur cette mission :

<i>Tâches dans le cahier des charges</i>	<i>Tâches réalisé</i>
1 Requête (liste détaillée des postes qui ne sont pas pourvus de processeur de type Intel)	
2 requête (liste détaillée des postes qui ont au plus 4Go de RAM ou 500 Go de HD)	
3 requête (Le nombre de logiciels recensés par éditeur)	
Connaitre l'ensemble des logiciels installés sur chaque poste	
Enregistrement de la machine virtuel de la Mission 1	
4 requête (L'éditeur et la version des logiciels installés sur le poste préparé à la mission 1)	
5 requête (Le nombre de logiciels installés par poste)	
Insertion des Services	
Schéma relationnel	

La mission 3 concernant la Base de données as donc été réalisé dans son ensemble, toutes les attentes et les requêtes ont été effectuer dans sa totalité en fonction du cahier des charges données.