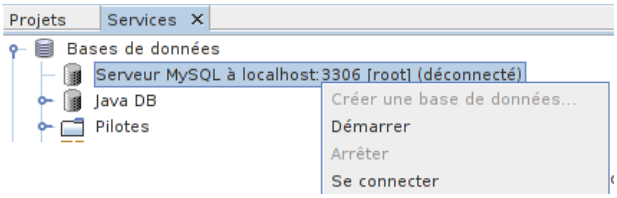
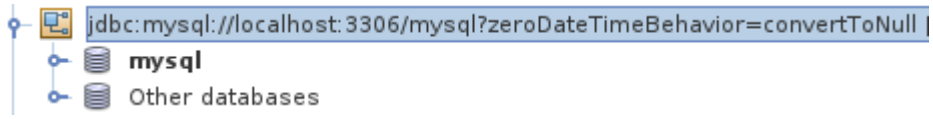
Gérer les bases de données depuis NetBeans

I- Exemple 1 : MySql

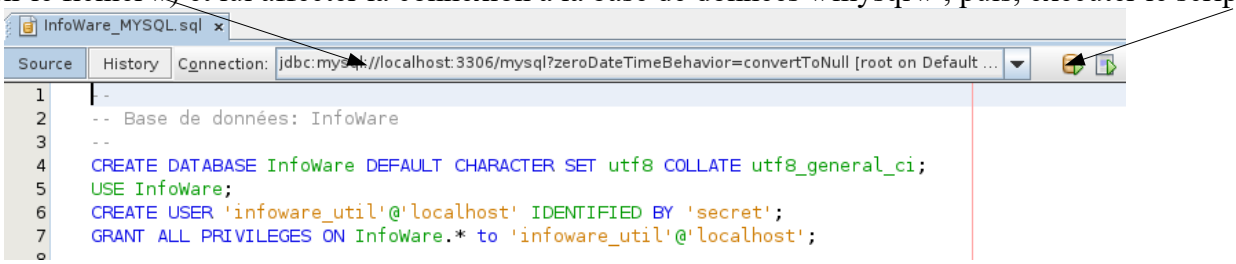
1. Démarrer le service MariaDB
2. Sélectionner l'onglet « Services » - « Bases de données »



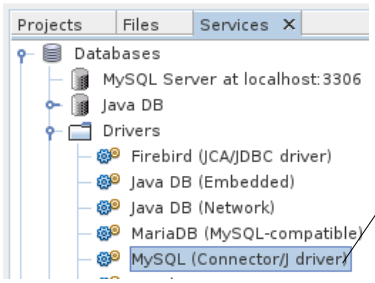
3. Ouvrir la connexion par défaut à la BD « mysql »



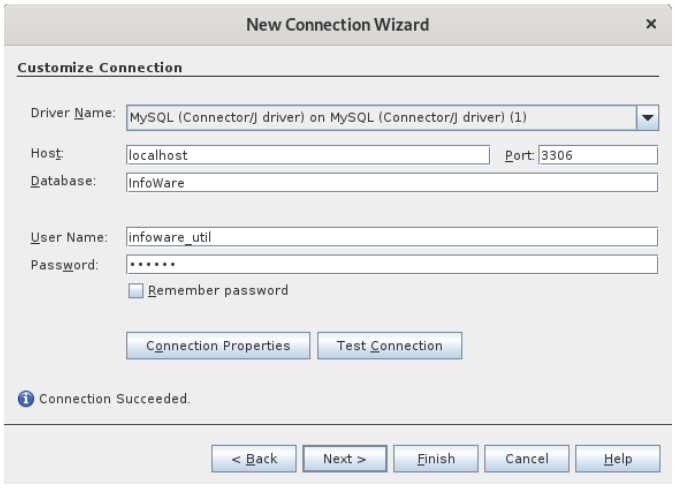
4. Ouvrir le fichier .sql de création de la base de données dans un onglet (menu « Fichier » - commande « Ouvrir le fichier ») et lui affecter la connexion à la base de données « mysql » ; puis, exécuter le script .



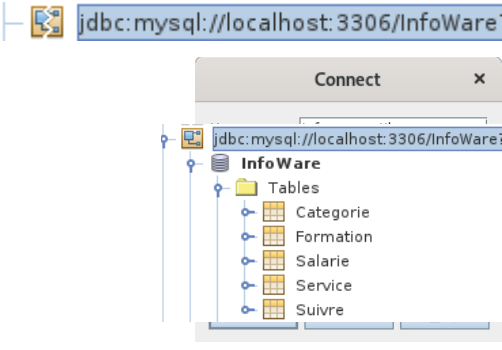
5. Créer une nouvelle connexion nouvelle « connexion » NetBeans sur la base de données « InfoWare » :



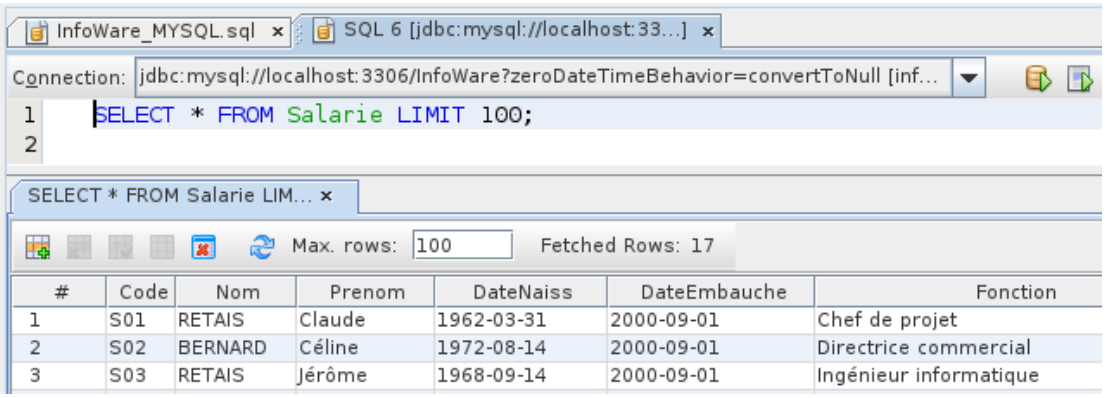
Clic droit puis « Connect using » ;
1- choisir le pilote MySql
2- le serveur (ici localhost)
3- le nom de la base de données (InfoWare)
4- le nom d'utilisateur et le mot de passe (Cf. script SQL)
5- cliquer sur « Test Connection »
Si le test est concluant, terminer la paramétrage pour enregistrer la connexion



6. Une nouvelle connexion est apparue ; ouvrir cette connexion (clic droit - Connecter) :

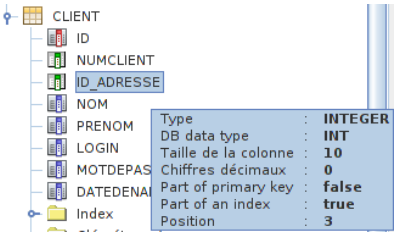


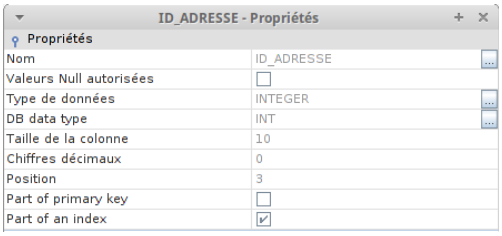
7. Pour afficher le contenu d'une table :
   clic droit sur la table
   puis « View data »



A partir de cette vue, on peut ajouter, supprimer, modifier des enregistrements de la table.

8. Pour exécuter toute autre requête :
   clic droit sur la table puis « Execute command »
9. Pour consulter la structure d'une table, développer le niveau correspondant puis afficher les propriétés.



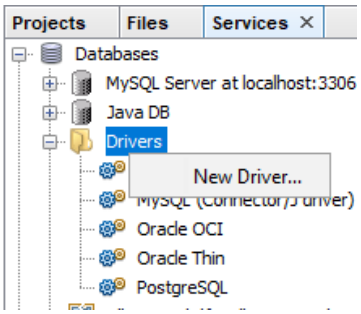


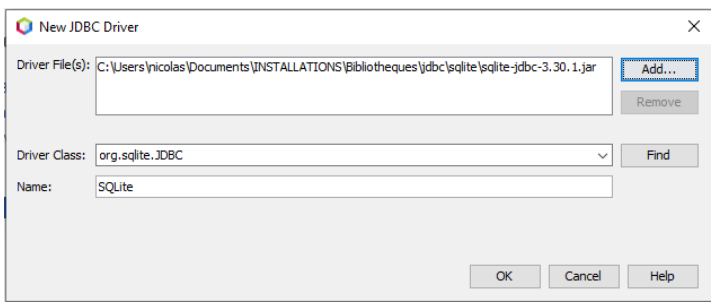
II-Exemple 2 : SQLite

Les procédures sont identiques, excepté pour la connexion. Les bases de données sont des fichiers locaux dont on précise le chemin d'accès dans la chaîne de connexion.
Il faut :

- se procurer le connecteur JDBC pour SQLite ;
- déclarer un nouveau connecteur pour SQLite dans l'onglet « Services » de NetBeans ;
- à partir de ce connecteur, créer une nouvelle connexion vers la base de données ; vérifier que le fichier correspondant a été créé ;
- exécuter le script de création des tables et de leur contenu à partir de cette connexion.

1. **Copier le connecteur JDBC pour SQLite**
   Le connecteur JDBC pour SQLite est fourni (*sqlite-jdbc-3.36.0.3.jar*) ; le copier dans le répertoire où vous stockez habituellement tous vos .jar ;
2. **Créer un nouveau connecteur pour SQLite**
   - Onglet « Services » - Databases – Drivers : clic droit puis « New Driver »
   - Parcourir les répertoires pour trouver le connecteur SQLite et valider

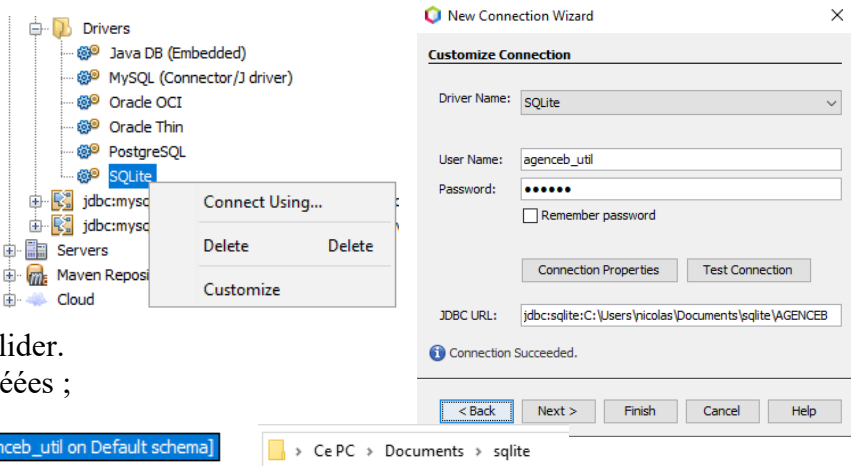




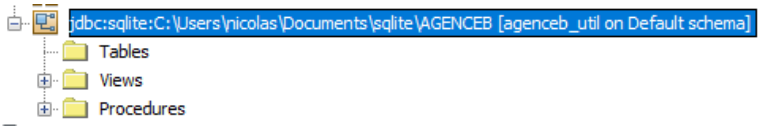
3. Créer une nouvelle connexion vers la base de données

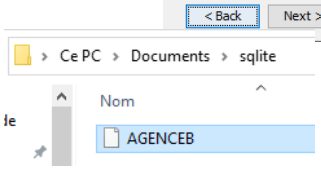
(la base de données sera créée par la même occasion

- Dans le répertoire « Documents », créer un sous répertoire « sqlite » où seront stockées les BDD embarquées SQLite ;
- Onglet « Services » - Databases – Drivers - SQLite puis « Connect Using » ;
- URL de la BDD :
  - Linux : `jdbc:sqlite:/home/<login>/Documents/sqlite/AGENCEB`
  - Windows : `jdbc:sqlite:C:\\Users\\<nom utilisateur>\\Documents\\sqlite\\AGENCEB`
- UserName, Password : agenceb_util/secret ;

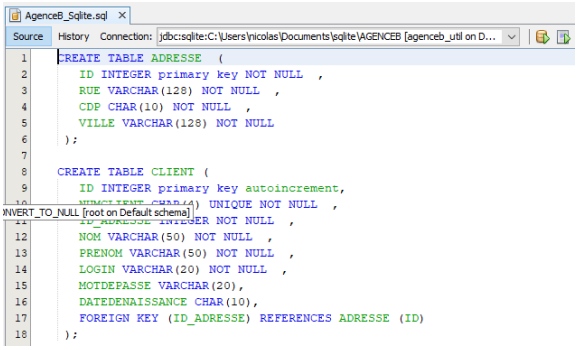


- Tester la connexion avant de valider.
- La BDD et la connexion sont créées ;



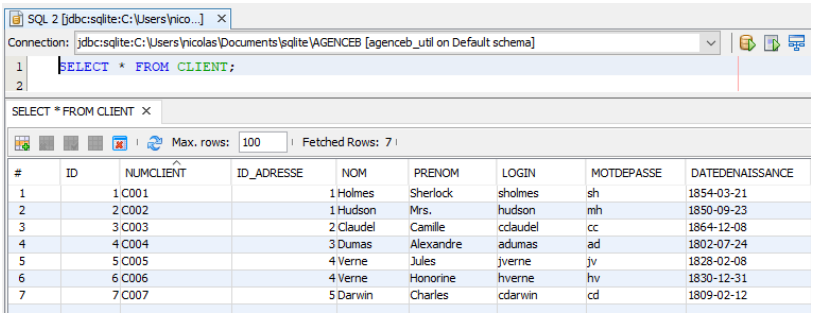


4. Exécuter le script de création des tables et de leur contenu à partir de cette connexion



```
CREATE TABLE ADRESSE  (
    ID INTEGER primary key NOT NULL  ,
    RUE VARCHAR(128) NOT NULL  ,
    CDP CHAR(10) NOT NULL  ,
    VILLE VARCHAR(128) NOT NULL
  );

CREATE TABLE CLIENT (
    ID INTEGER primary key autoincrement,
    NUMCLIENT CHAR(4) UNIQUE NOT NULL  ,
    ID_ADRESSE INTEGER NOT NULL  ,
    NOM VARCHAR(50) NOT NULL  ,
    PRENOM VARCHAR(50) NOT NULL  ,
    LOGIN VARCHAR(20) NOT NULL  ,
    MOTDEPASSE VARCHAR(20),
    DATEDENAISSANCE CHAR(10),
    FOREIGN KEY (ID_ADRESSE) REFERENCES ADRESSE (ID)
  );
```



```
SELECT * FROM CLIENT;
```

#	ID	NUMCLIENT	ID_ADRESSE	NOM	PRENOM	LOGIN	MOTDEPASSE	DATEDENAISSANCE
1	1	C001	1	Holmes	Sherlock	sholmes	sh	1854-03-21
2	2	C002	1	Hudson	Mrs.	hudson	mh	1850-09-23
3	3	C003	2	Claudel	Camille	cclaudel	cc	1864-12-08
4	4	C004	3	Dumas	Alexandre	adumas	ad	1802-07-24
5	5	C005	4	Verne	Jules	jverne	jv	1828-02-08
6	6	C006	4	Verne	Honorine	hverne	hv	1830-12-31
7	7	C007	5	Darwin	Charles	cdarwin	cd	1809-02-12

II.1 Conversion de dates en Java

1. Le type java.time.LocalDate

Depuis la version 8 de Java, le type conseillé pour enregistrer les dates est LocalDate du paquetage java.time.

1.1. Conversion LocalDate vers String :

utiliser un objet de type DateTimeFormatter :

```
LocalDate uneDate = LocalDate.of(2018,4,17);  //jeu d'essai
DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy-MM-dd");
String maDateStr = dtf.format(uneDate);
```

1.2. Conversion String vers LocalDate :

utiliser un objet de type DateTimeFormatter :

```
String uneDateStr = "17/04/2018"; // jeu d'essai
DateTimeFormatter dtf = DateTimeFormatter.ofPattern("dd/MM/yyyy");
LocalDate d = LocalDate.parse(uneDateStr, dtf);
```

2. Le type java.sql.Date

Lorsqu'il lit une donnée issue d'un champ du type SQl DATE, JDBC instancie un objet de type java.sql.Date (sous-type de java.util.Date) : `java.sql.Date ddn  = rs.getDate("DATEDENAISSANCE") ;`

2.1. Convertir une date de type *« java.sql.Date »* en « *LocalDate* »

```
LocalDate uneDate = rs.getDate("DATEDENAISSANCE").toLocalDate() ;
```

2.2. Convertir une date de type *« LocalDate »* en « *java.sql.Date* »

```
LocalDate uneDate = LocalDate.of(1999,1,15) ;
java.sql.Date ddn  = java.sql.Date.valueOf(uneDate) ;
```