L. Bougueroua

TP BDD
M2 - UPMC

Documentations en lignes

DB Designer 4 : http://www.fabforce.net/dbdesigner4/docs.php
PostgresSQL : http://www.postgresql.org/docs/8.3/interactive/index.html
http://postgresql.developpez.com/faq/

Mise en œuvre des outils

1) Mise en œuvre des outils pour l'utilisation d'un SGBD PostgreSQL

Connexion au SGBD PostgreSQL via ODBC

1) Lancer le Panneau de configuration → Outils d'administration → Sources de données (ODBC)
2) Accéder à l'onglet « Sources de données système » (figure 1)
3) Sélectionner PostgresSQL30W et lancer la commande « Configurer »

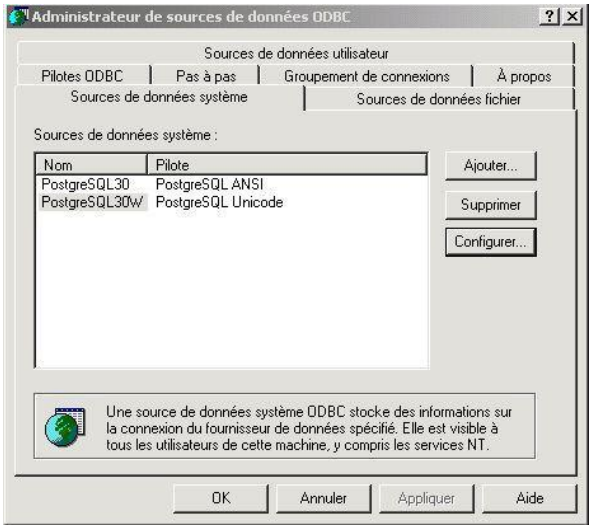

Figure 1 : Administrateur de sources de données ODBC

4) Configuration du pilote ODBC (figure 2) :

- Le nom de la base : CAVE
- Le serveur : 10.2.254.1
- Le port : 5432
- user-name : (votre compte de binôme pour le TP)
- password : (le mot de passe)

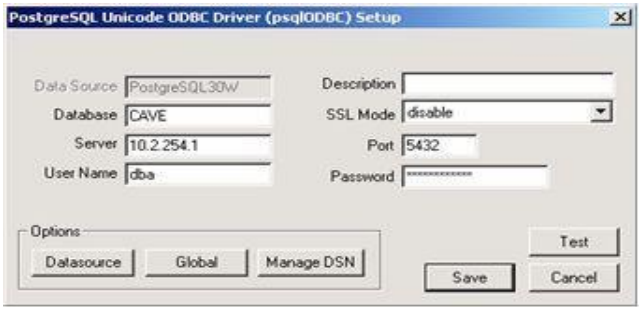

Figure 2 : configuration du pilote ODBC pour PostgreSQL

Mise en place pour DB Designer

1) Lancer DB Designer (*la documentation pour la modélisation se trouve à l'adresse url suivante : http://www.fabforce.net/dbdesigner4/doc/index.html*)
2) Lancer à partir du menu: *database → connect to database (figure 3)*

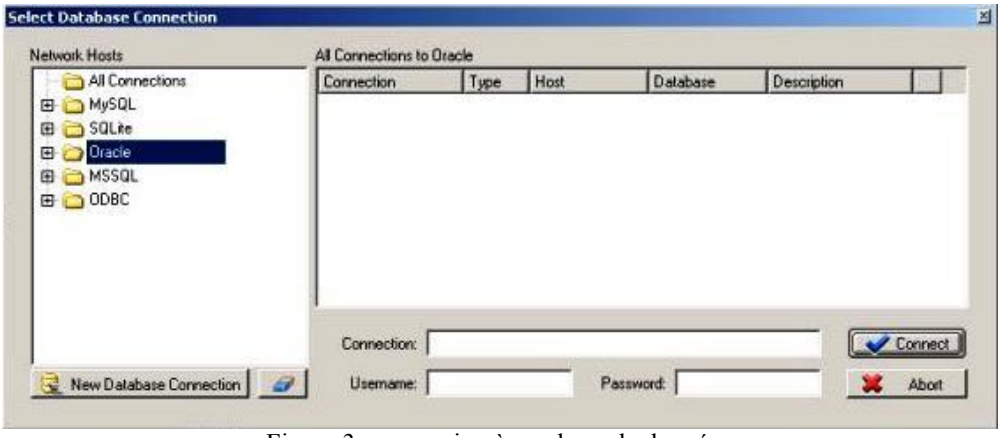

Figure 3 : connexion à une base de données

3) Lancer la commande : « *New Database Connection* »
4) Configuration du pilote ODBC (figure 4) :

- Le nom de la connexion :
- Le pilote (driver) : ODBC
- Le nom de l'ODBC DNS : PostgresSQL30W
- user-name : (votre compte de binôme pour le TP)
- password : (le mot de passe)

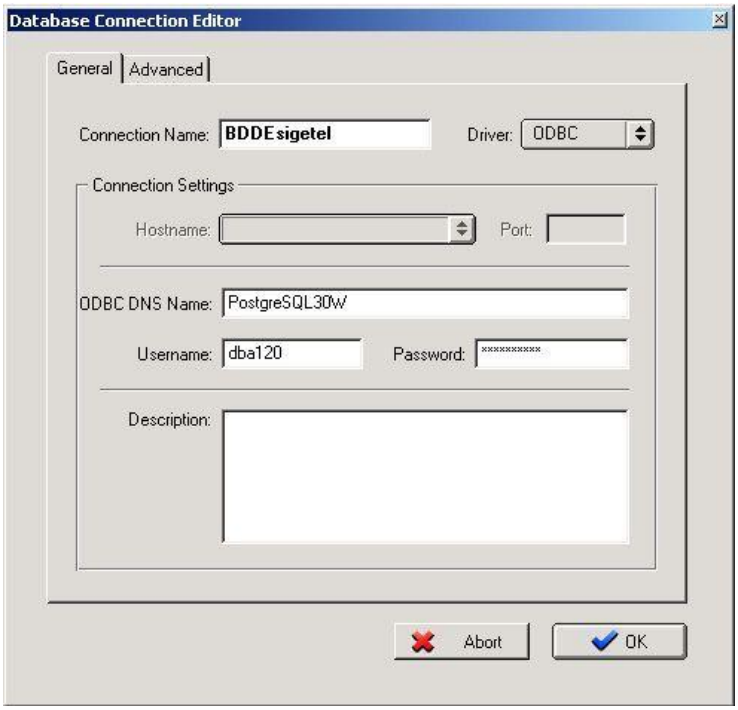

Figure 4 : éditeur de connexion à une base de données

Maintenant vous pouvez par exemple récupérer l'ensemble des objets qui se trouve dans la base de données : à partir du menu: *database → reverse engineering*

5) L'éditeur de DB Designer est présenté dans la figure 5 (*la documentation pour la modélisation se trouve à l'adresse url suivante : http://www.fabforce.net/dbdesigner4/doc/index.html*)

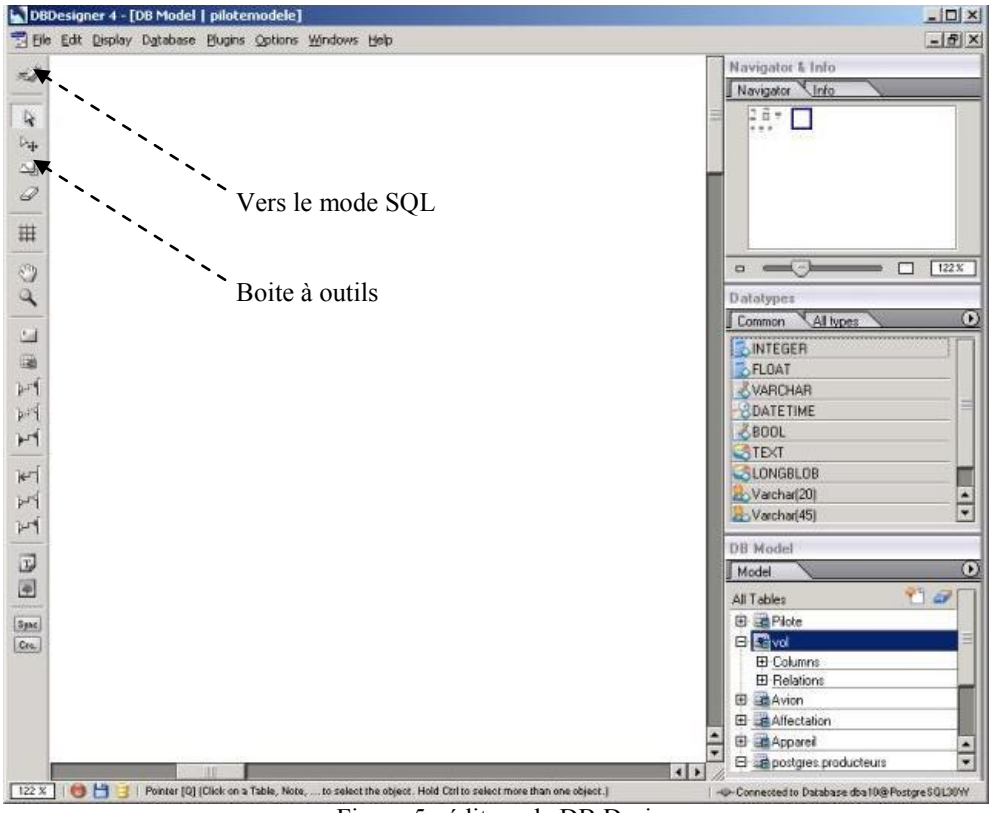

Figure 5 : éditeur de DB Designer

La figure suivante (figure 6), montre la possibilité de basculer entre le mode schéma conceptuel et l'éditeur de requêtes SQL.

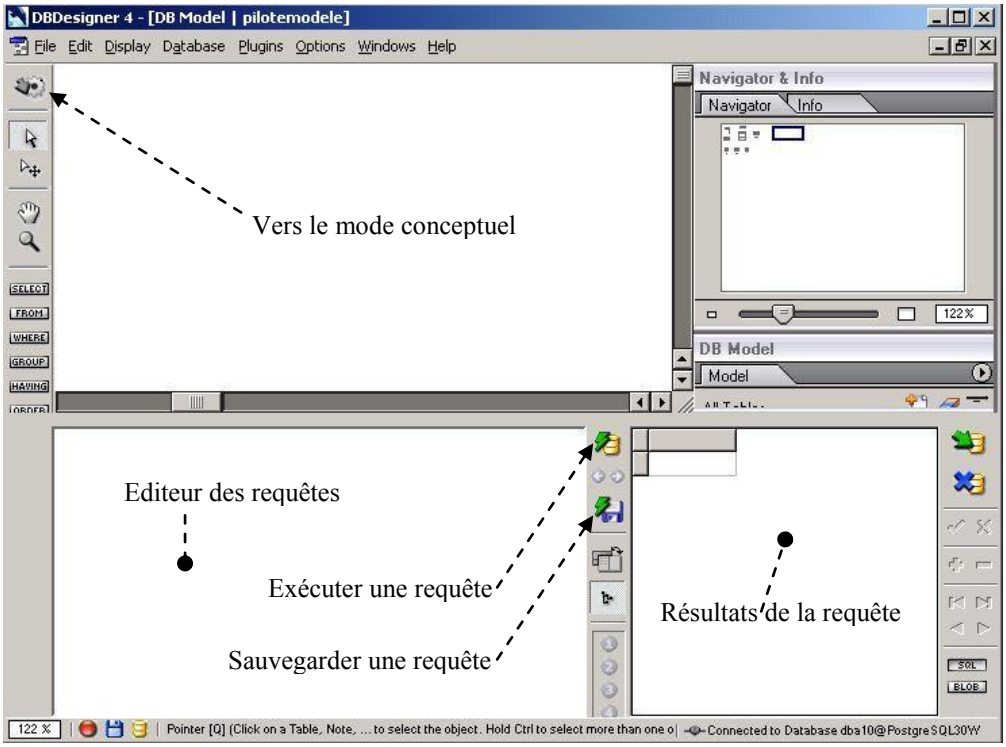

Figure 6 : éditeur de modèles

TP N°1 : Création d'une BDD

Une compagnie d'aviation gère ses vols, ses pilotes, ses appareils au moyen d'une base de données relationnelle. Le schéma de la base est défini ci-dessous :

VOL *(novol, vildep, vilar, dep_h, dep_mm, ar_h, ar_mn)*

PILOTE *(nopilot, nom, adresse, salaire, comm, embauche)*

APPAREIL *(codetype, nbplace, design)*

AVION *(nuavion, type, anneeserv, nom, nbhvol)*

AFFECTATION *(vol, date_vol, pilote, nbpass, avion)*

Partie 1 :

En utilisant DB Designer, tracer le schéma de la base de données en intégrant:

- L'ensemble des tables
- Les relations entre tables
- Les domaines des différents types
- Les contraintes possibles (définition des clés primaires et étrangère, gestion des valeurs null, surveillance des domaines de définition des attributs)

Remarques importantes (valable pour l'utilisation de DB Designer avec le SGBD PostgreSQL):

- Il faut utiliser que des relations binaires
- Dans chaque table :
  - éviter l'incrémentation automatique d'un attribut (AI)
  - éviter l'utilisation des index

Partie 2 :

Basculer en mode requêtes (*Query mode*) :

0. Changer vos mots de passe : ALTER USER login WITH PASSWORD 'password';
1. Créer un nouveau schéma dans la BDD dont le nom est votre login
2. Créer les tables en utilisant le script automatiquement (fichiers→Exporter → script de création SQL)
3. Modifier votre script en incluant les instructions de suppressions des tables juste avant la création des tables
4. Rédiger un nouveau script pour l'insertion des données ("TPinitialisation.sql" par exemple) à l'aide des données fournis dans la page suivante.
5. Rédiger une vue qui donne la date du dernier vol réalisé pour chaque avion. Afficher le numéro de l'avion, son label, la date et le vol.
6. Exprimer à partir de cette vue l'action suivante: consultation de la vue pour l'avion 8567
7. Rédiger une vue qui donne pour chaque pilote son nombre d'affectations. Cette vue comportera le numéro du pilote, son nom et le nombre d'affectations de ce pilote.
8. Exprimer à partir de cette vue l'action suivante : ajout de 3 à chaque nombre d'affectation de pilote

9. Donner pour chaque pilote qui est passé par PARIS, le numéro de vol et la date correspondante. Afficher le numéro de pilote, le nom de pilote, le numéro de vol et la date du vol.
10. Donner pour chaque pilote, la liste des villes à partir desquelles il n'a jamais décollé. Afficher le numéro de pilote, le nom de pilote et la vile.

Informations sur la Base

La table **VOL**

NOVOL	VILDEP	VILAR	DEP_H	DEP_MM	ARR_H	ARR_MM
AF8810	Paris	Djerba	9	0	11	45
AF8809	Djerba	Paris	12	45	15	40
IW201	Lyon	Fort de France	9	45	15	25
IW655	La Havane	Paris	19	55	12	35
IW433	Paris	St-Martin	17	0	8	20
IW924	Sidney	Colombo	17	25	22	30
IT319	Bordeaux	Nice	10	35	11	45
AF3218	Marseille	Francfort	16	45	19	10
AF3530	Lyon	Londres	8	0	8	40
AF3538	Lyon	Londres	18	35	19	15
AF3570	Marseille	Londres	9	35	10	20

La table **PILOTE**

NOPILOTE	NOM	ADRESSE	SAL	COMM	EMBAUCHE
1333	Fois	Nantes	99000	0	01-MAR-92
6589	Duval	Paris	98600	5580	12-MAR-92
7100	Martin	Paris	85600	16000	01-APR-93
6548	Barre	Lyon	82680		01-DEC-92
1243	Collet	Paris	89000	0	01-FEB-90
5543	Delorme	Paris	91850	9850	01-FEB-92
3452	André	Nice	92570	2000	12-DEC-92
3421	Berger	Reims	83700		28-DEC-92
6723	Martin	Orsay	93150		15-MAY-92
3843	Gaucher	Cachan	87600		20-NOV-92
3465	Pic	Tours	88650		15-JUL-93

La table **APPAREIL**

CODE	NB PLACE	DESIGN
74E	150	BOEING 747-400 COMBI
AB3	130	AIRBUS A300
741	100	BOEING 747-100
SEC	80	CONCORDE
734	450	BOEING 737-400

La table AVION

NUAVION	TYPE	ANSERV	LABEL	NBHEUREVOL
8832	734	1998	Ville de Paris	16000
8567	734	1998	Ville de Reims	8000
8467	734	2005	Le Sud	600
7693	741	1998	Pacifique	34000
8556	AB3	2004		12000
8432	AB3	2001	Malte	10600
8118	74E	2002		11800

La table AFFECTATION

VOL	DATE VOL	PILOTE	AVION	NBPASSAGERS
IW201	01-MAR-05	6723	8567	310
IW201	02-MAR-05	6723	8832	26
AF3218!	12 -JUN-05	6723	7693	83
AF3530	12-NOV-05	6723	8432	178
AF3530	13-NOV-05	6723	8432	155
AF3533	21-DEC-05	6723	8118	110
IW201	03-MAR-05	1333	8567	356
IW201	12-MAR-05	6589	8467	211
AF8810	02-MAR-05	7100	8556	160
IT319	02-MAR-05	3452	8432	105
IW433	22-MAR-05	3421	8556	178
IW655	23-MAR-05	6548	8118	118
1W655	20-DEC-05	1243	8467	402
IW655	18-JAN-05	5643	8467	398
IW924	30-APR-05	8843	8832	412
IW201	01-MAY-05	6548	8432	156
AF8810	02-MAY-05	6589	7693	88
AF3218	01-SEP-05	8843	7693	98
AF3570	12-SEP-05	1243	7693	56
AF3570	12-SEP-05	1299	8118	180

TP N°2 : requêtes SQL

Nous souhaitons utiliser le modèle relationnel pour gérer une cave à vin. La base de données représentant cette cave possède le schéma relationnel suivant:

VINS *(num, cru, annee, degre)*

PRODUCTEURS *(num, nom, prenom, region)*

RECOLTES *(nprod, nvin, quantite)*

Un vin est caractérisé par un ***numéro***, un ***cru***, une ***année*** de production et un ***degré***. L'ensemble des vins est représenté par la relation VINS. La clé de la relation VINS est l'attribut ***num***.

Un producteur est caractérisé par un ***numéro***, un ***nom***, un ***prénom*** et une ***région***. L'ensemble des producteurs est représenté par la relation PRODUCTEURS. La clé de la relation PRODUCTEURS est l'attribut ***num***. Un producteur produit un ou plusieurs vins. Réciproquement, un vin est produit par un ou plusieurs producteurs (éventuellement par aucun !).

L'ensemble des productions est représenté par la relation RECOLTES. Un tuple de la relation RECOLTES représente une production particulière d'un vin de numéro ***nvin*** par un producteur de numéro ***nprod*** en une certaine quantité. La clé de la relation RECOLTES est le groupe d'attributs (***nvin, nprod***).

Cette base est déjà construite, elle appartient au schéma *postgres* : pour l'utilisée il faut passer le nom du schéma + le nom de la table (exemple : ***postgres.vins***).

L'objectif du TP est d'analyser cette base de données en répondant à la liste des questions suivante:

1. Afficher l'ensemble des tables du schéma postgres (voir la documentation)
2. Afficher les informations contenues dans la relation VINS.
3. Donner la liste des producteurs qui n'ont pas de prénom.
4. Donner la liste des régions de production des vins.
5. Donner la liste des vins de 1980 ordonnée par degré.
6. Donner la liste par ordre alphabétique des noms et des prénoms des producteurs de vins n'appartenant pas aux régions suivantes : Corse, Beaujolais, Bourgogne et Rhône.
7. Quelle est la liste des crus récoltés en 1979 ordonnée par numéro de producteur? Afficher le cru, le numéro du producteur et la quantité produite.
8. Quels sont les noms des producteurs du cru Etoile, leurs régions et la quantité de vins récoltés?
9. Quel est le nombre de récoltes?
10. Combien y-a-t-il de producteurs de vin dans la région Savoie et Jura?

11. Combien y-a-t-il de producteurs de vin ayant récoltés au moins un vin dans la région Savoie et Jura?
12. Quelles sont les quantités de vin produites par région. Donner la liste ordonnée par quantité décroissante.
13. Quelle est la quantité de vin produite de degré >12?
14. Quel est le cru ou les crus au plus fort degré?
15. Donner la liste ordonnée des crus.
16. Donner la liste ordonnée des crus récoltés.
17. Quel est donc le cru non récolté?
18. Donner la liste ordonnée des crus et la quantité par cru?
19. Quel est le degré moyen des crus?
20. Quel sont les crus (ordonnés par degré et année) de degré supérieur au degré moyen des crus?
21. Donner la liste des noms et prénoms des producteurs produisant au moins trois crus.
22. Retrouver tous les couples de producteurs habitant la même région. Les Tuples du résultat seront de la forme num1, nom1, num2, nom2, région. La présence d'un Tuple avec num1 et num2 interdit la présence d'un Tuple avec num2, num1.
23. Pour un producteur, un cru est significatif s'il a été produit en quantité supérieure à 200 litres. Calculer pour chaque producteur le nombre de crus significatifs récoltés.
24. Quelles sont les régions dont la quantité de vin produite est supérieure à 5000 litres ? Afficher la région et la quantité.
25. Donner les numéros et les noms des producteurs qui produisent au moins tous les vins produits par le producteur 35. Vous utilisez l'instruction EXISTS.
26. Retrouver les numéros et les noms des producteurs qui produisent au moins tous les vins produits par le producteur 35. Vous utilisez l'instruction COUNT.

TP N°3 : BDD et JDBC

A partir du code de l'exemple (voir annexe A), rédiger un programme java contenant :

1. Une méthode ***connexion()*** regroupant l'ensemble des instructions nécessaires pour la connexion à la base de données.
2. Une méthode ***connexion_erreur()*** pour la gestion des problèmes de connexion.
3. Une méthode ***deconnexion()*** pour la déconnexion.
4. Une méthode ***requete_simple()*** permettant l'exécution d'une requête simple dont le résultat se limite à un unique Tuple. Application avec la requête: "quelles sont les données du cru dont le numéro est égal à 30?"
5. Une méthode ***requete_table()*** permettant l'exécution et l'affichage d'une table: Application avec la requête : "Liste des producteurs (nom, prénom) de la région Corse".
6. Une méthode ***requete_parametree (int annee)*** comportant un paramètre (année) permettant d'effectuer une restriction. Application avec la requête :"Quels sont les producteurs des crus (nom, prenom, cru) de l'année (**annee**)"
7. Une méthode ***requete_region_web()*** permettant de générer un fichier HTML (voir annexe B). Application avec la requête: "Donner la liste ordonnée des régions".
8. Ajouter l'ouverture automatique du fichier html (voir annexe C)
9. Généraliser l'affichage en format html des requêtes.

ANNEXE A – Exemple:

```
import java.sql.*;
public class MaClasseJDBC{

        public static void main(String args[]){
                String url = "jdbc:postgresql://10.2.254.1:5432/CAVE";
                Connection con=null;
                Statement stmt;
                String query = "SELECT NUM, NOM FROM postgres.PRODUCTEURS;";

                try {
                        Class.forName("org.postgresql.Driver");
                        con = DriverManager.getConnection(url,"login", "password");/
                        stmt = con.createStatement();
                        ResultSet rs = stmt.executeQuery(query);
                        System.out.println("Liste des producteurs:");
                        System.out.println("Num          nom");
                        while (rs.next()){
                                int i = rs.getInt(1); // num
                                String n = rs.getString(2); // nom
                                System.out.println(i + " "+n );
                                }
```

```
            }catch(SQLException ee){
                        ee.printStackTrace();
            }catch (Exception e){
                        e.printStackTrace();
            }
            if (con!=null) try {con.close();} catch(Exception e){}
      }
}
```

ANNEXE B – HTML:

SOURCE DU PROGRAMME: HTML à générer

```
<HTML>
 <HEAD> <TITLE>Liste des regions de production</TITLE> </HEAD>
 <BODY>
<CENTER>
 <H2>Liste des régions de production</H2>
 <TABLE BORDER=1>
 <TR BGCOLOR=RED ALIGN=CENTER>
   <TH>NUMERO</TH>
   <TH>Label</TH>
 </TR>
 <TR>
   <TD BGCOLOR=WHITE>1</TD>
   <TD BGCOLOR=POWERBLUE>Alsace  </TD>
</TR>
<TR>
  <TD BGCOLOR=WHITE>2</TD>
  <TD BGCOLOR=POWERBLUE>Beaujolais </TD>
</TR>
.....
</TABLE>
</CENTER>
</BODY>
</HTML>
```

```
//Sauvegarde dans un fichier
DataOutputStream ecrire = new DataOutputStream(new FileOutputStream("res.html"));
```

ANNEXE C :

```
String[] callAndArgs = { "E:\\Apps\\Mozilla\\FireFox\\firefox.exe","res.html"};
Process process =Runtime.getRuntime().exec(callAndArgs);
```

Projet BDD

La compagnie de transport TT3 (Tout Transport, Tout Type, Tout Temps) a choisi de se diversifier. Son domaine d'activité principal est lié aux taxis et aux transports de groupes. Cette diversification récente a entraîné des problèmes de gestion et la direction a compris que les difficultés étaient en partie incriminables à l'obsolescence de son système d'information. La décision d'informatiser a été prise. Le PDG de la compagnie, sur les conseils d'un ami, a décidé de procéder à une modélisation du champ d'application avant d'acheter les ordinateurs personnels que lui réclament les gestionnaires de la compagnie (proverbe du routier: "Ne pas mettre la charrue avant les bœufs").

Le champ d'application couvert par les impératifs de la gestion de cette compagnie de transport concerne le parc de véhicules, son entretien, l'administration de chauffeurs et de leur emploi du temps, la gestion des appels des clients à la centrale téléphonique.

Le texte qui suit est une description du champ d'application tel qu'il apparaît à la suite d'une réunion avec les différents cadres de la direction (les mots en style gras sont les constituants qui sont retenus dans la modélisation).

Compte rendu de la réunion:

Pour le chef mécanicien, un véhicule est identifié par un numéro de châssis. Chaque véhicule possède un numéro de plaque ainsi qu'une date de mise en service. Le parc de véhicules est divisé en plusieurs types. Un type est connu par le modèle, par exemple: Mercedes 300. Un véhicule ne peut bien entendu appartenir qu'à un seul modèle. La description d'un modèle permet de connaître le nombre de personnes pouvant prendre place dans les véhicules de ce modèle; le type de carburant consommé; la catégorie de permis que doit posséder le chauffeur; le type de boîte à vitesses et le poids du modèle.
Un des objectifs de ce système d'information est de surveiller la consommation journalière en carburant des véhicules. Ainsi une augmentation de cette consommation sera le signe que le moteur nécessite un réglage. A chaque fois que le chauffeur fait le plein, il remplit une fiche indiquant le numéro de plaque, la date, le nombre de kilomètres roulés depuis le dernier plein, la quantité et le type de carburant mis dans le réservoir.

L'équipe des mécaniciens s'occupe de l'entretien. Ce qui est mémorisé sur l'entretien des véhicules est donné par une description associée à une date et un numéro de châssis.

Le responsable de la planification a besoin des informations suivantes pour établir l'emploi du temps de ces chauffeurs. Il dispose jusqu'à maintenant de fiches sur les chauffeurs où l'on trouve le numéro du chauffeur, son nom, son prénom, son adresse. La fiche contient aussi un emplacement où sont notées les catégories de permis que possèdent les chauffeurs.

L'emploi du temps des chauffeurs et des véhicules est défini sur un grand tableau qui occupe une paroi entière de son bureau. Les numéros des châssis des véhicules se trouvent sur les entêtes de ligne. Les entêtes de colonne sont des dates subdivisées en trois tranches horaires. Un seul numéro de chauffeur est inscrit dans une case du tableau.

Le responsable de planification doit faire attention à ce que les modèles conduits par les chauffeurs soient compatibles avec les permis qu'ils possèdent.

Le central téléphonique pour gérer les appels téléphoniques est équipé d'un système qui permet aux chauffeurs de donner leur position en indiquant la zone dans laquelle ils sont inoccupés avec un véhicule. Lorsqu'un client demande un taxi, il suffit de lui assigner un véhicule dans sa zone Je prise en charge. Si aucun taxi ne se trouve dans la zone, il faut trouver le plus proche véhicule. Afin d'effectuer cette recherche d'une manière optimale, il existe des *tabelles*, qui, pour chaque heure de la journée indique le temps de parcours d'une zone à une autre. On encode ainsi la variation de fluidité du trafic dans la ville au cours de la journée.

La compagnie TT3 est répartie dans la ville en stations où sont garés les véhicules. Une station possède un numéro de station. La station se trouve dans une zone. Un véhicule est associé à une seule station. Un chauffeur est aussi assigné à une et une seule station. C'est à cette station qu'il vient prendre et rendre son véhicule.

Ce TP est doit être traité comme un microprojet comprenant :

1. En utilisant DB Designer, tracer le schéma de la base de données.
2. Créer la base de données sous PostgreSQL
3. Ecrire des requêtes SQL caractéristiques permettant de vérifier le contenu des diverses tables ainsi que la cohérence du schéma relationnel.
4. Ecrire une requête SQL permettant de connaître le ou les véhicules les plus proches lorsqu'un client appelle d'une zone donnée.
5. Rédiger un programme java permettant de dialoguer avec la BDD