République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Benyoucef Benkhedda – Alger 1

Faculté des Sciences

Département de Mathématiques et Informatique

Master 1 Ingénierie des Systèmes Informatiques Intelligents (ISII)

Entrepôt de Données

Projet TP

Élaboré par :

M^lle Hanafi Zohra Groupe 02

M^lle Tesbia Lylia Groupe 02

Année universitaire
2019 – 2020

Table des matières

- # Introduction :

L'entreposage de données est une phase du processus décisionnel qui supporte efficacement le processus OLAP, il est né dans les entreprises pour l'aide à la prise de décision. Ainsi, les principaux utilisateurs de ces technologies font partie intégrante de l'entreprise, Nous pouvons citer les secteurs qui peuvent tirer profit des outils d'aide à la décision comme les banques et les assurances, ainsi que ceux de l'automobile et des institutions médicales…etc.

- # Présentation de la base de données :

« Finantial » est une base de données dédiée aux opérations bancaires. Elle contient 606 prêts bancaires réussis et 76 non réussis ainsi que leurs informations et transactions. Elle est composée de 55 colonnes et de plus d'un million de lignes réparties sur 8 tables liées selon le schéma ci-dessous.

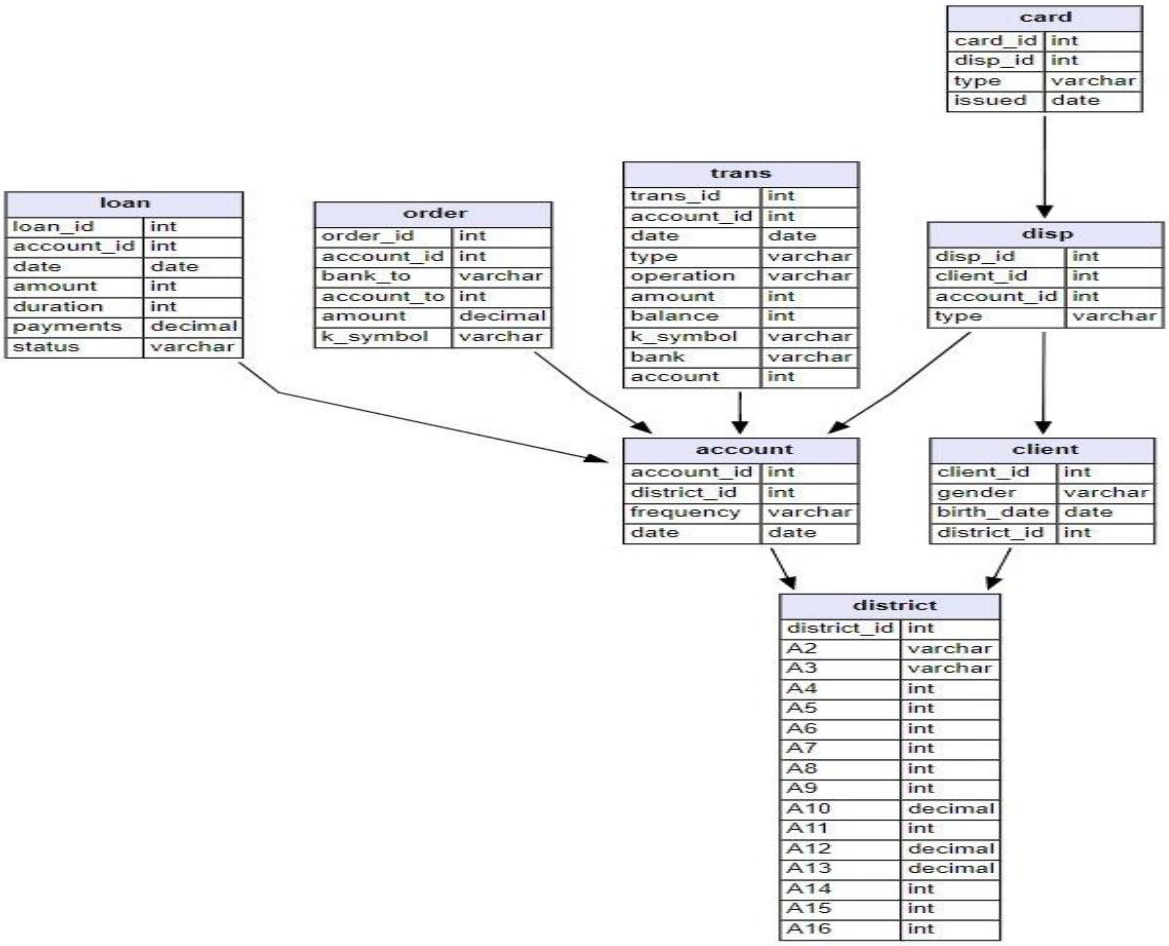

Figure 1 Schema de la base de données opérationnelles

Le tableau ci-dessous détaille les caractéristiques du jeu de données :

Relation	Description
Account	Elle contient 4500 enregistrements, chaque, enregistrement décrit les caractéristiques d'un compte

Client	Elle contient 5369 enregistrements, chaque enregistrement décrit les caractéristiques d'un client.
Disposition	Elle contient 5369 enregistrements, chaque enregistrement relie un client à un compte, c'est-à-dire que cette relation décrit les droits des clients à gérer des comptes.
Order	Elle contient 6471 enregistrements, chaque enregistrement décrit les caractéristiques d'un ordre de paiement.
Transaction	Elle contient 1056320 enregistrements, chaque enregistrement décrit une transaction faite par un compte.
Loan	Elle contient 682 enregistrements, chaque enregistrement décrit un prêt accordé pour un compte donné.
District	Elle contient 77 enregistrements, chaque enregistrement donne des informations sur une région donnée.
Card	Elle contient 892 enregistrements, chaque enregistrement détaille les informations d'une carte de crédit.

Notre but est de proposer une conception et une implémentation des différentes phases du processus qui permet la création d'un entrepôt de données à partir d'une base de données opérationnelle.

Pour cela on a commencé par analyser la base de données opérationnelle afin d'établir un suivi pour cette banque.

Après avoir analysé les données, on est passé à l'étape de conception et construction de l'entrepôt de données qui traitera les points suivants :

1. La conception et la modélisation de l'entrepôt de données.
2. La conception de la zone d'ETL.

• Modélisation de l'entrepôt de données

Nous allons d'abord décrire le processus d'affaires traité et définir son grain, ses dimensions, ainsi que la table des faits et ses mesures, ensuite on passera à la modélisation de ce processus à travers un schéma en étoile.

➢ Conception du processus de suivi des « Prêt » :

Ce processus vise à analyser l'état du prêt, s'il est réussi ou non ainsi que le reste à payer, le montant payé ainsi que le nombre de mois payé selon plusieurs axes d'analyse (**client**, **région**, **date)**.

➢ Grain de l'activité

Chaque ligne dans la table de fait (PrêtFact) représente l'état d'un prêt, son montant versé, le reste à payer ainsi que le nombre de mois payé par un certain **client**, sur **une région** établie à **une date** donnée.

➢ Présentation des dimensions

Ce processus est analysé selon les dimensions suivantes :

1. Dimension client,
2. Dimension région,
3. Dimension date avec une granularité de un jour (24h).

➢ Les mesures

Etat_paid : l'état d'un prêt si c'est un prêt réussit ou non (1 pour réussit 0 sinon).

Reste : le reste à payer.

Payed : le montant versé.

Paid_monthe : le nombre de mois payé.

➢ Schéma en étoile du processus suivi des prêts

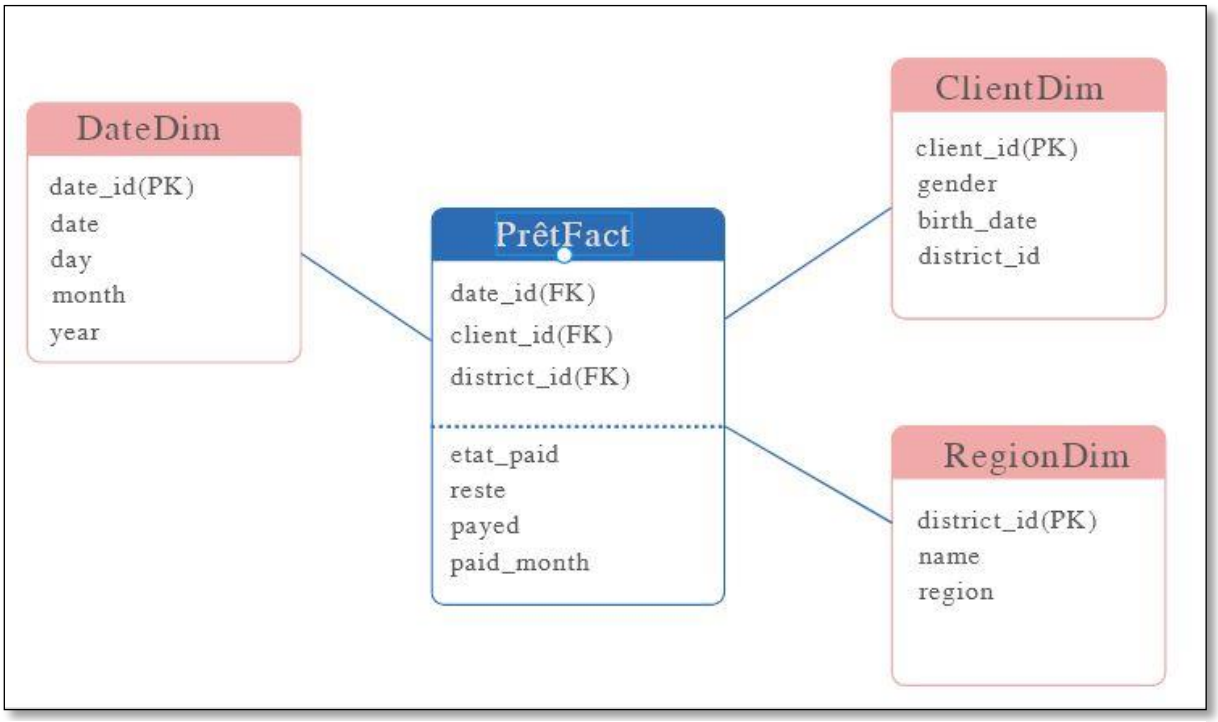

Figure 2 Modélisation du schéma en étoile

• Conception de la zone d'alimentation (ETL)

L'alimentation de l'entrepôt de donnée se déroule par le processus d'alimentation de données (ETL). Ce processus est utilisé pour alimenter l'entrepôt donné à partir des bases de données opérationnelles. Ainsi, il permet l'extraction, le nettoyage et l'importation des données des différentes sources et il les charge dans un entrepôt de données (data Warehouse) en temps réel.

❖ Extraction

Cette étape consiste à extraire les données de leurs sources, fichiers plats auxquelles nous avons un accès direct via notre poste de travail, et à les charger dans la zone de préparation de données.

Dans notre cas on va utiliser **l'extraction complète** qui consiste à remplacer tout le contenu de la table plutôt que chercher à identifier les changements survenus. Ce type est utilisé pour le chargement initial de l'entrepôt de données.

❖ Transformation

Nous procédant à la transformation et nettoyage des données c'est la 2eme étape du processus ETL, après avoir analysé les données et vu les anomalies tirées des systèmes sources, certaines transformations étaient nécessaires :

- ✓ **La suppression des tables inutiles** :

On a supprimé les deux tables « Order, Card » car on n'en aura pas besoin pour extraire des données ou effectuer des calculs.

- ✓ **Suppression des champs inutiles** :

La table « district » qui contient a la base 16 champs qui ne sont pas tous utile donc on a réduit le nombre de ses attributs a 3 attributs qui sont tous jugé nécessaire dans notre travail qui sont « district_id, A2,A3 » , la table « loan » on a supprimé le champ « status», la table « account » on a supprimé le champ « frequency», la table « transaction » on a enlevé les champs « operation, bank,account».

Ce traitement a été effectué sur la zone de préparation de données dans laquelle on a effectué une extraction complète. C'est-à-dire on a copié notre base de données opérationnelle complète dans une zone intermédiaire.

```
String modifier_account = "ALTER TABLE account DROP COLUMN frequency";
String modifier_tans = "ALTER TABLE transaction DROP COLUMN operation, DROP COLUMN bank,"
        + "DROP COLUMN account";
String modifier_loan = "ALTER TABLE loan DROP COLUMN status";
String modifier_district = "ALTER TABLE district DROP COLUMN A4 ,"
        + "DROP COLUMN A5 , DROP COLUMN A6 , DROP COLUMN A7 , DROP COLUMN A8 ,"
        + "DROP COLUMN A9 , DROP COLUMN A10 , DROP COLUMN A11 , DROP COLUMN A12 ,"
        + "DROP COLUMN A13 , DROP COLUMN A14 , DROP COLUMN A15 , DROP COLUMN A16";



Connection connection = tab.connexion_db(db_name,user,mdp);
Statement statement1= connection.createStatement();
Statement statement2= connection.createStatement();
Statement statement3= connection.createStatement();
Statement statement4= connection.createStatement();
statement1.executeUpdate(modifier_account);
statement2.executeUpdate(modifier_tans);
statement3.executeUpdate(modifier_loan);
statement4.executeUpdate(modifier_district);
```

Figure 3 Suppression des champs inutiles

- ✓ **Formatage de la date :**

On a tous d'abords créer une table 'date' qui représente la date d'un prêt, après on a fait une extraction par jour, mois et année.

```
//fonction pour extract data from Date:
public void ExtractDataDate(String bdd_name, String user,String mdp) throws SQLException  {
Connection connection = DriverManager.getConnection("jdbc:postgresql://localhost:5432/"+bdd_name, user, mdp);
Statement statement = connection.createStatement();


//creation de la table date son identifiant est auto increment (serial):
statement.executeUpdate("CREATE TABLE IF NOT EXISTS  Date ( date_id SERIAL PRIMARY KEY, date date, day int, month int, year int );");

//la requette qui nous permet d'extraire le jour le mois et l'annee d'une date donnee
ResultSet result = statement.executeQuery(" WITH ItemDate as (select date as d from loan) select d, extract (day from d) as jour_d,"
        + "extract (month from d) as month_d,extract (year from d) as year_d from ItemDate ;");
//on insere ensuite dans la table date en utilisant prepareStatement ():
String query ="insert into Date ( date_id , date , day , month , year  )"
                        + " values (DEFAULT, ?, ?, ?, ?)";

while (result.next()) {
        PreparedStatement preparedStmt = connection.prepareStatement(query);

        preparedStmt.setDate(1,result.getDate("d"));
        preparedStmt.setInt (2, result.getInt("jour_d"));
        preparedStmt.setInt(3, result.getInt("month_d"));
        preparedStmt.setInt(4, result.getInt("year_d"));

        preparedStmt.execute();
                }
result.close();
```

Figure 4 création de la table date

Puis, après le formatage de la date on supprime les dates dupliquées.

```
PreparedStatement preparedStmt = connection.prepareStatement("DELETE FROM date " +
                        "  WHERE date IN (SELECT date " +
                        "                   FROM (SELECT row_number() OVER (PARTITION BY date), date " +
                        "                           FROM date) x " +
                        "                  WHERE x.row_number > 1);");

preparedStmt.executeUpdate();
connection.close();
}
```

Figure 5 Formatage de la table date

- ✓ **Calculs des mesures :**

Ou on a calculé l'ensemble des mesures dont a besoin :

- **paid_month** : qui est le nombre de mois que le client a payé, il représente le nombre de transactions où k-symbole est égale à 'UVER'.
- **etat_paid :** après le calcul du nombre des mois payants, on le compare avec la duration de prêt (dans la table loan) s'ils sont égaux, c'est que c'un prêt réussit s'ils sont différents, c'est que c'est un prêt non réussit.
- **payed** : c'est le nombre de mois payants multiplié par le payement nécessaire par mois (l'attribut payments dans la table loan).

❖ **reste :** c'est la soustraction le montant total du prêt (l'attribut amount dans loan) et le payement effectuer 'payed'.

```
//creation de la table transation_loan :
statement.executeUpdate("CREATE TABLE IF NOT EXISTS  Tra_load ( id_tra_load SERIAL PRIMARY KEY, client_id int, paid_month int, duration int,payed float, "
                    + "reste float, amount float,gender varchar,date date,etat_paid int, id_district int);");
ResultSet result = statement.executeQuery("SELECT c.client_id, count (k_symbol) as paid_month , l.duration, (count (k_symbol)*l.payments)as payed,  "
                                          + "l.amount-(count (k_symbol)*l.payments)as reste,l.amount,c.gender,l.date," +
                                          "(CASE WHEN count (k_symbol)=l.duration  THEN 1" +
                                                  " ELSE 0" +
                                          " END" +
                                          " ) as etat,c.district_id" +
                                        "from transaction t, loan l,disp di, client c" +
                                        "where l.account_id = t.account_id and t.k_symbol ='UVER' and l.account_id=di.account_id and "
                                        + "c.client_id= di.client_id and di.type='OWNER'" +
                                        "group by c.client_id, l.duration,l.payments,l.amount,c.gender,l.date,c.district_id" +
                                        "order by l.date");
    //ensuite on insere:
String query ="insert into Tra_load (id_tra_load,client_id,paid_month,duration,payed,reste,amount,gender,date,etat_paid,id_district)"
                        + " values (DEFAULT, ?, ?, ?, ?,?,?,?,?,?,?)";

while (result.next()) {
    PreparedStatement preparedStmt = connection.prepareStatement(query);

        preparedStmt.setInt(1,result.getInt(1));
        preparedStmt.setInt (2, result.getInt(2));
        preparedStmt.setInt(3, result.getInt(3));
        preparedStmt.setFloat(4, result.getFloat(4));
        preparedStmt.setFloat(5,result.getFloat(5));
        preparedStmt.setFloat(6,result.getFloat(6));
        preparedStmt.setString(7, result.getString(7));
        preparedStmt.setDate(8, result.getDate(8));
```

Figure 6 Calculs des mesures

❖ Chargement

C'est la dernière étape du processus ETL, les données sont chargées dans un entrepôt central. Nous pouvons maintenant combinées, agrégés et chargés dans des cubes ou Datamart si cela est jugé nécessaire, dans cette phase nous effectuons un :

- Chargement de la dimension temps.
- Chargement des dimensions.
- Chargement des tables de faits.

✓ Chargement de la dimension temps

La dimension de date est une dimension clé dans un entrepôt de données, Elle est principalement une dimension statique qui ne nécessite pas de mise à jour quotidienne cette dimension est chargée qu'une seule fois durant le cycle de vie de l'entrepôt de données. Nous avons donc construit cette dimension comme étant un calendrier constitué de dates qui correspondent à notre fait 'Prêt'.

```
public void LoadDateDim(String bdd_name_ource,String bdd_name_target, String user,String mdp) throws SQLException {
    Connection connection = DriverManager.getConnection("jdbc:postgresql://localhost:5432/"+bdd_name_ource, user, mdp);
    Connection connection2 = DriverManager.getConnection("jdbc:postgresql://localhost:5432/"+bdd_name_target, user, mdp);
    Statement statement = connection.createStatement();
    ResultSet result = statement.executeQuery("select date_id, date, day, month, year from date");

    String query ="insert into datedim (date_id, date, day, month, year) VALUES (?, ?, ?, ?, ?)";

    while (result.next()) {
            PreparedStatement preparedStmt = connection2.prepareStatement(query);

            preparedStmt.setInt(1, result.getInt("date_id"));
            preparedStmt.setDate(2,result.getDate("date"));
            preparedStmt.setInt (3, result.getInt("day"));
            preparedStmt.setInt(4, result.getInt("month"));
            preparedStmt.setInt(5, result.getInt("year"));

            preparedStmt.execute();

            }
```

Figure 7 Chargement de la dimension temps

✓ Chargement des autres dimensions

On suppose que les tables sont inchangées, d'où elles nécessitent un seul chargement.

Voici un exemple du chargement de l'une des tables dimension :

```
public void LoadClientDim(String bdd_name_ource,String bdd_name_target, String user,String mdp) throws SQLException {
    Connection connection = DriverManager.getConnection("jdbc:postgresql://localhost:5432/"+bdd_name_ource, user, mdp);
    Connection connection2 = DriverManager.getConnection("jdbc:postgresql://localhost:5432/"+bdd_name_target, user, mdp);
    Statement statement = connection.createStatement();
    ResultSet result = statement.executeQuery("select client_id, gender, birth_date, district_id from client");

    String query ="insert into clientdim(client_id, gender, birth_date, district_id)VALUES ( ?, ?, ?, ?)";

    while (result.next()) {
        PreparedStatement preparedStmt = connection2.prepareStatement(query);

        preparedStmt.setInt(1, result.getInt("client_id"));
        preparedStmt.setString(2,result.getString("gender"));
        preparedStmt.setDate (3, result.getDate("birth_date"));
        preparedStmt.setInt(4, result.getInt("district_id"));

        preparedStmt.execute();

        }
result.close();
connection.close();
connection2.close();
    }
```

Figure 8 Chargement des autres dimensions

✓ Chargement de la table de faits

Après avoir chargé toutes les dimensions nous passons aux tables de faits. La source est la même que pour les dimensions, c'est-à-dire la zone de préparation de données. Pour chaque enregistrement de table des faits dans l'entrepôt de données où la clé primaire de notre table de fait est l'ensemble des clés primaire des tables de dimensions.

```
public void LoadFactTable(String bdd_name_ource,String bdd_name_target, String user,String mdp) throws SQLException {
    Connection connection = DriverManager.getConnection("jdbc:postgresql://localhost:5432/"+bdd_name_ource, user, mdp);
    Connection connection2 = DriverManager.getConnection("jdbc:postgresql://localhost:5432/"+bdd_name_target, user, mdp);
    Statement statement = connection.createStatement();
    ResultSet result = statement.executeQuery("select d.date_id, tl.client_id, tl.id_district, tl.etat_paid, tl.reste,tl.payed, tl.paid_month " +
                       "from date d, tra_load tl " +
                       " where d.date= tl.date ");
    String query ="insert into presfact(date_id, client_id, district_id, etat_paid, reste, payed, paid_month)VALUES (?, ?, ?, ?, ?, ?, ?);";

    while (result.next()) {
        PreparedStatement preparedStmt = connection2.prepareStatement(query);
        preparedStmt.setInt(1, result.getInt(1));
        preparedStmt.setInt(2,result.getInt(2));
        preparedStmt.setInt(3,result.getInt(3));
        preparedStmt.setInt (4, result.getInt(4));
        preparedStmt.setFloat (5, result.getFloat(5));
        preparedStmt.setFloat (6, result.getFloat(6));
        preparedStmt.setInt (7, result.getInt(7));
        preparedStmt.execute();
            }
```

Figure 9 Chargement de la table de faits

• Interface graphique

Dont le but de faciliter l'interaction avec l'utilisateur, on a créé une application Desktop permettant à l'utilisateur d'effectuer les opérations suivant :

➢ L’authentification :

L’utilisateur doit saisir le nom d’utilisateur et le mot de passe du serveur pour s’authentifier.

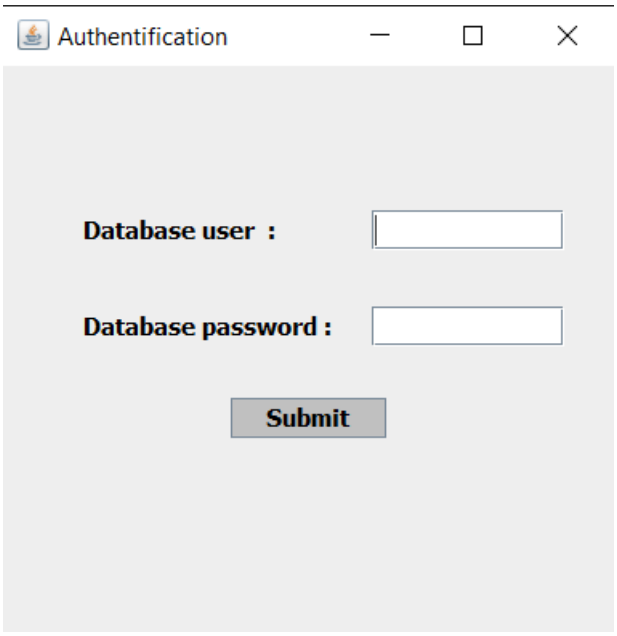

Figure 10 Authentification à l'interface graphique

➢ Les opérations de l’application :

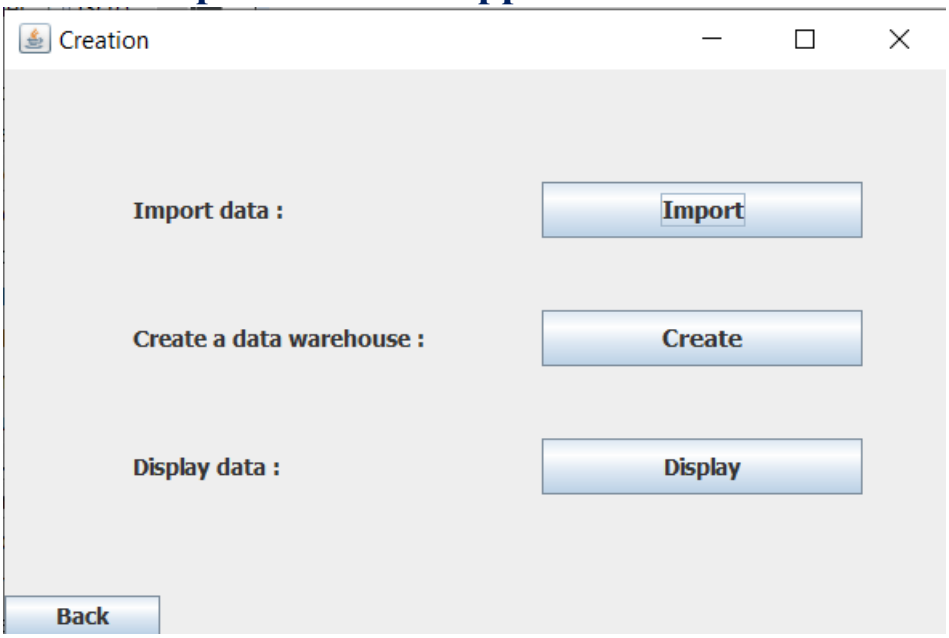

Figure 11 Opérations possibles de l'interface graphiques

1. Importation d’une base de données :

L’utilisateur doit d’abord introduire le nom d’utilisateur et le mot de passe du serveur ainsi que le nom de la base de données qui sera créée.

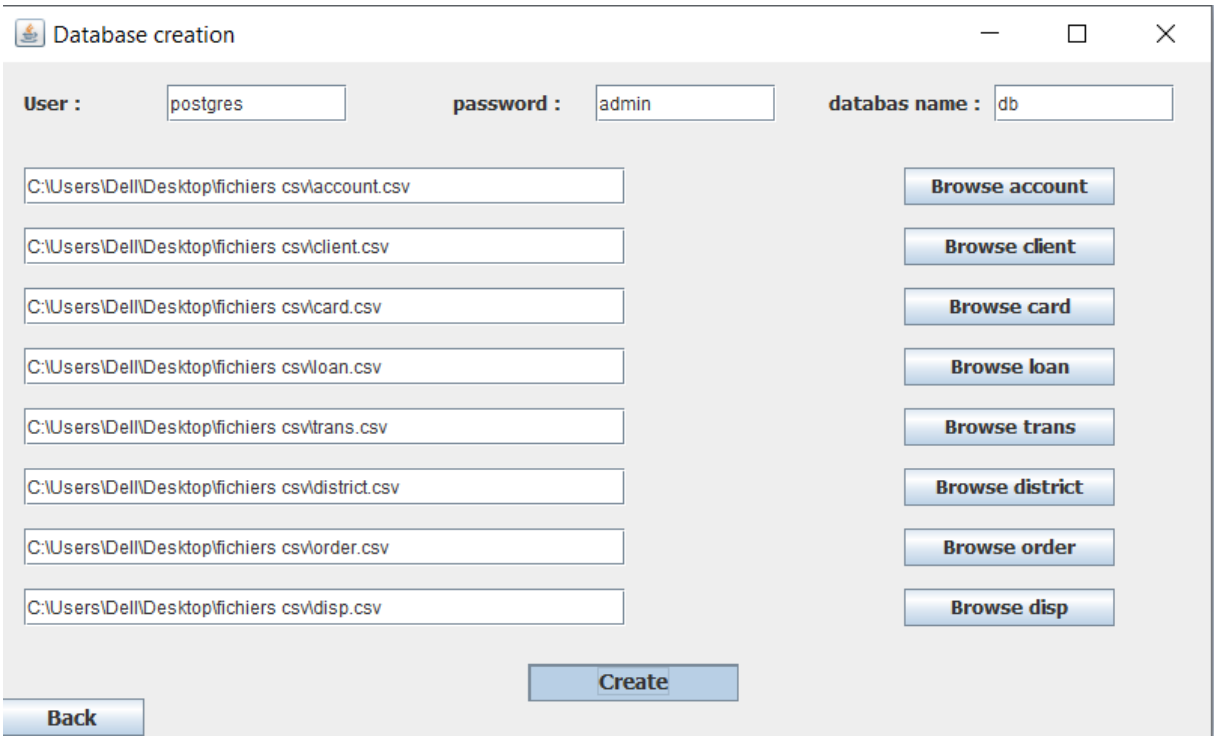

Figure 12 Création d'une base de données

Ensuite, il devra sélectionner le chemin dans son ordinateur de chaque table qui sera sous forme de fichier csv.

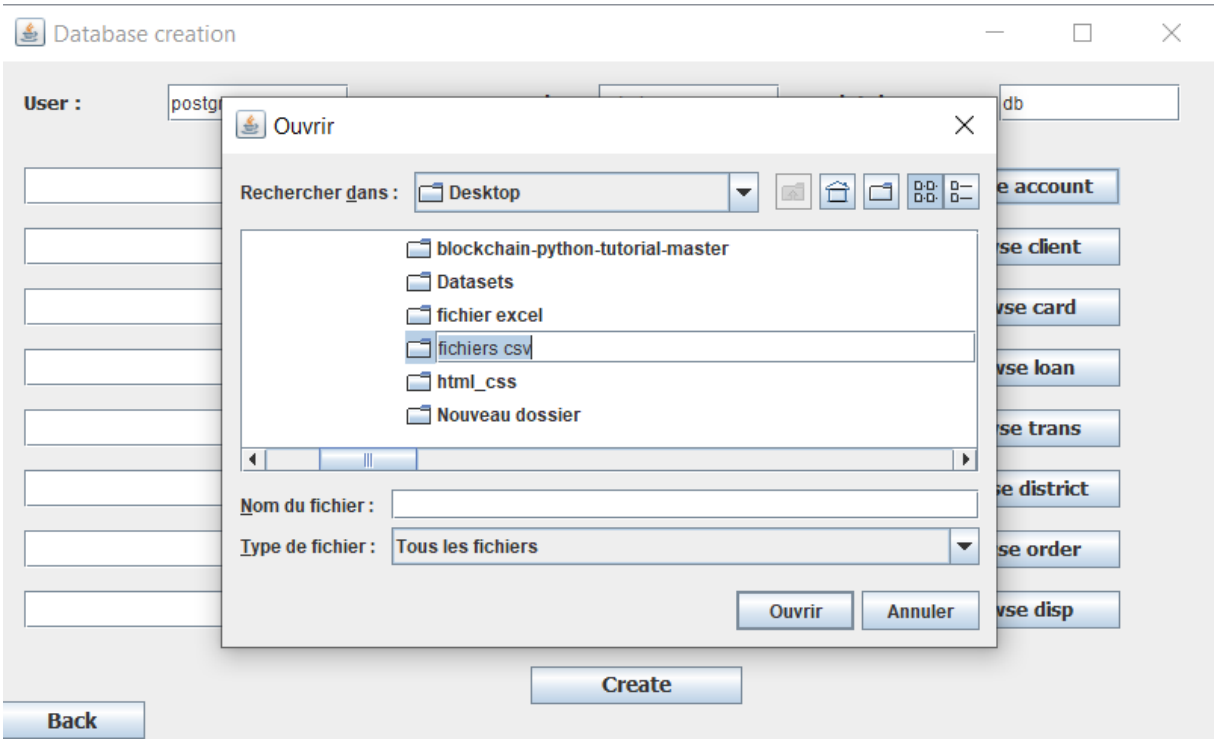

Figure 13 Explication du choix d'un fichier plat

2. Création d'un entrepôt de données :

L'utilisateur peut aussi créer un entrepôt de données en saisissant le nom d'utilisateur et le mot de passe du serveur, le nom de l'entrepôt de données à créer, le nom de la base de données depuis laquelle il va créer cet entrepôt, et le nom de la base de données intermédiaire.

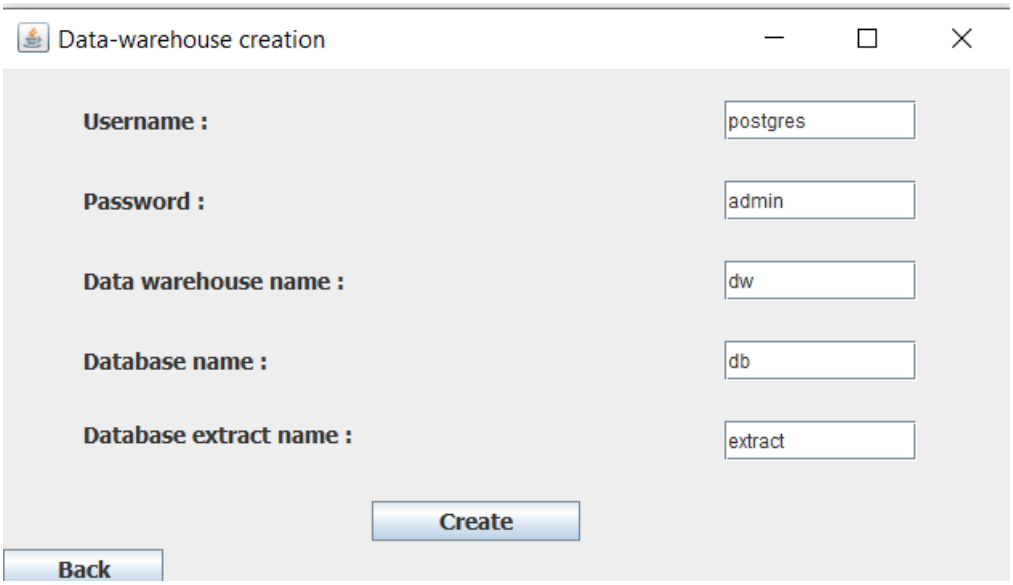

Figure 14 Création d'un entrepôt de données

3. Visualiser les différentes tables de l'entrepôt de données

L'utilisateur doit saisir le nom d'utilisateur et le mot de passe du serveur ainsi que le nom de l'entrepôt de données pour visualiser les différentes tables et leurs données de cet entrepôt.

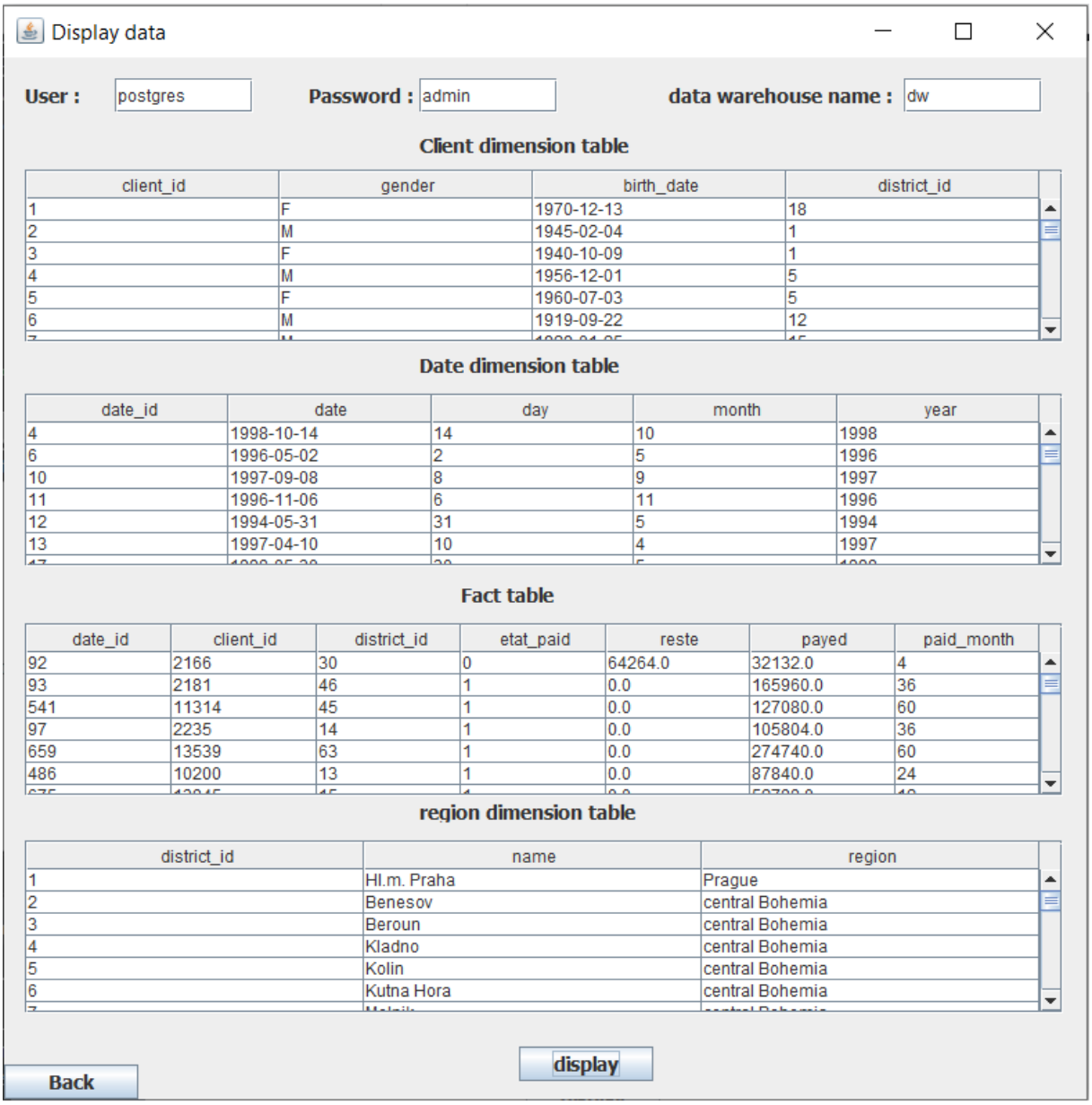

Figure 15 Visualiser les tables de l'entrepôt de données

Conclusion

Dans ce projet nous avons implémenté les différentes phases du processus qui permet la création d'un entrepôt de données à partir d'une base de données opérationnelle, ou on a commencé par présenter une modélisation de la solution proposée ce qui a permis de produire le schéma en étoile partant par le schéma relationnel source obtenu à partir des fichiers plats CSV.