661311 Logiciels Statistiques: ETL processing and Visualisation in R

Mohamed OUHOURANE
Roland H. G. DOSSA

Sunday, 04th March, 2018

Introduction

En 1970 E.F. Codd publie, un article sur *Un modèle relationnel de données pour les grandes banques de données partagées*. C'est un article qui présentait un modèle de stockage et de manipulation des données à l'aide de tableaux. Quelques temps après la publication de cet article de Codd, l'IBM (**I**nternational **B**usiness **M**achines) s'est mis à travailler sur la création d'une base de données relationnelle. Entre 1979/1982, l'Oracle, Relational Technology et l'IBM ont tous mis en place des bases de données relationnelles à caractère commercial. C'est à partir de 1986 que ces derniers ont commencé à utiliser le SQL (Structured Query Language) comme langage de requêtes. Le langage SQL fut normalisé par l'American National Standards Institute (ANSI) en 1986, d'où parfois l'appelation du SQL standard comme étant "**SQL ANSI**". Cette norme a fait l'objet d'une mise à jour entre 1989 et 2008. La première mise à jour a été celle de 1989, en suite en 1992 (appelée SQL2), en 1999 (appelée SQL3), en 2003 (appelée SQL 2003), en 2006 (appelée SQL 2006) enfin en 2008 (appelée SQL 2008). Toutes les bases de données relationnelles majeures supportent cette norme mais chacune a ses propres extensions propriétaires.

À travers ce travail, il sera question de manipuler des données de format SQL en format R et ce via un document RMarkdown.

Ce travail s'organisera principalement autour de trois grands chapitres. Le chapitre 1 met en exergue la base de données qui sera utilisé. En effet, ce chapitre nous permettra de faire ressortir les différentes composantes que contient la base de données **Northwind** (qui fera objet de transformation dans notre étude) afin d'avoir une bonne compréhension de celle-ci. Le chapitre 2 mettra en exergue le processus **ETL**. À travers ce chapitre, nous procéderons aux différentes transformations telles que l'Extraction, la Transformation et le Loading. Ici, nous montrerons les différentes techniques qui nous permettront de passer du **système transactionnel OLTP** (Online transaction processing) vers le **systèmes OLAP** (OnLine Analytical Processing). En d'autres terme, nous montrerons les différents procédés qui nous permettront de passer de la base de données originale à une la base de données DataWareHouse sur laquelle nous ferions des visualisations. Le troisième et dernier chapitre nous permettra de mettre en relation quelques éléments obtenus après le processus ETL. Il sera question de procéder à des graphiques, série chronologique etc.

Description de la base de données

Dans le cadre de ce projet nous exploiterons la base de donnée **Northwind**. En effet, la base de données Northwind est un exemple de base de données fournie avec l'application Microsoft Access. Fondamentalement, la base de données concerne une société nommée "Northwind Traders". Elle prend en compte toutes les transactions de vente qui se produisent entre ladite société, c'est-à-dire les commerçants Northwind et ses clients, ainsi que les transactions d'achat entre cette société et ses fournisseurs. La **figure 1** montre le cheminement des différentes transactions.

Source.

La base de données contient les détails suivants:

- Offreurs/Vendeurs de Northwind: ils offrent les produits à la compagnie.
- Clients de Northwind: Ils achètent les produits au près des vendeurs de Northwind.
- Employés de la société de Northwind: Ils travaillent pour le compte de la société Northwind.
- Information sur les produits: ce sont les produits commercialisés sur lesquels la société de Northwind fait les inventaires détaillés.
- Livreurs: les informations sur les différents livreurs des produits des vendeurs aux clients.
- Transactions de commandes d'achat: il s'agit des transactions des commandes entre les vendeurs et la société.
- Transactions de commandes de vente: Il s'agit des informations détaillées sur les transactions effectuées entre les clients et la société.
- Inventaires des transactions: il s'agit de toutes les informations sur les transactions se déroulant dans l'inventaire des factures et sur celles des factures portées contre la commande. La **figure 2** montre la nomenclature de la base de donnée:

Notons que cette nommenclature est le plus souvent désignée par l'appelation **système transactionnel OLTP** qui signifie en anglais **Online transaction processing**. Ce derdernier est défini comme étant un traitement transactionnel en ligne qui sert à effectuer des modifications d'informations en temps réel. On le retrouve essentiellement dans des opérations commerciales comme les opérations bancaires, ou l'achat de bien divers (dans notre cas). L'objectif de l'utilisation d'un tel système est de pouvoir insérer, et interpréter pour des besoins divers, les données de la base de données, en toute sécurité.

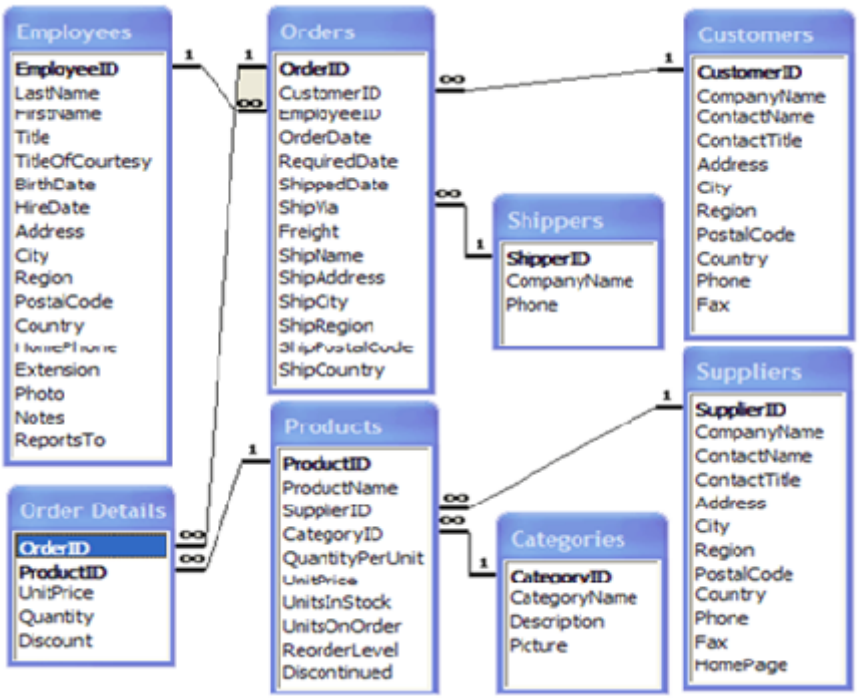


Figure 1: Structures de la base de données

Source.

Modélisation dimensionnelle

À l'aide des systèmes OLTP qui servent, en général, de source de données pour les **systèmes OLAP** (OnLine Analytical Processing) qui sont quant à eux, source d'analyse des données qui vont permettre d'aboutir à la décision. Ainsi, il va donc falloir extraire des systèmes OLTP, les éléments qui feront l'objet de notre analyse dans le cas cette étude, on parle de la **modélisation dimensionnelle OLAP**.

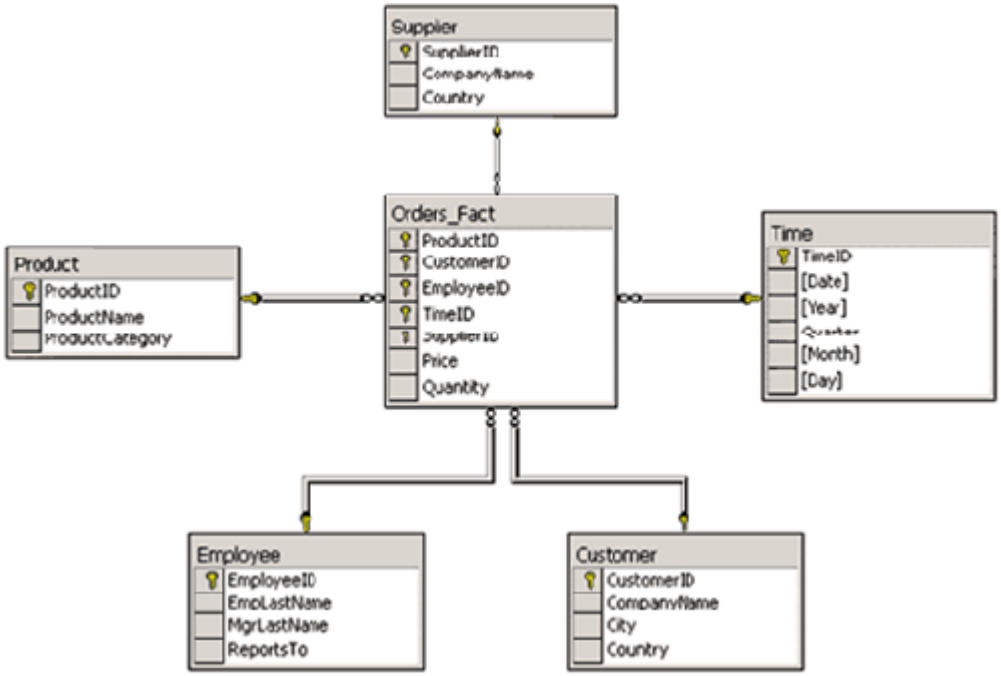
En effet, la modélisation dimensionnelle structure les données d'une façon très différente de la structure fréquemment utilisée par les modélisateurs. Elle produit ce qu'on appelle le modèle dimensionnel ou communément le *schéma en étoile*. Elle est la structure de données la plus utilisée et la plus appropriée aux requêtes et analyses d'entrepôts de données. Non seulement simple à créer, mais elle est également stable et intuitivement compréhensible par les utilisateurs finaux. Nous rencontrons le plus souvent deux types de base de modèle dimensionnel:

1- Le schéma en flocons de neige

2- Le schéma en étoile

Dans le cadre de notre travail, nous nous attellerons sur le dernier type.

En effet, le schéma en étoile consiste en une grande table de **faits** et un cercle d'autres tables qui contient les éléments descriptifs du fait, appelés **dimensions**. Lorsqu'il s'agit de l'illustrer, ce modèle ressemble à une étoile, c'est d'ailleurs l'origine du terme en *étoile*. La **figure 3** suivante montre le schéma en étoile pour le compte de notre travail :



Processus ETL (Extracting Transformation Loading)

Avant toute chose, il est important de procéder à l'installation du **RODBC**. Ce dernier implémente la connectivité de base de données ODBC avec des bases de données conformes où les pilotes existent sur le système hôte.

```
#install.packages("RODBC")
library(RODBC)
```

Les données **Northwind** étant des données enregistrées dans l'aplication Microsoft Access, il est donc important de procéder à la connection de cette base avant toute utilisation. De même, il a été créé une connection pour la nouvelle base DataWareHouse (*NorthwindDataWarehouse2000*). La commande ci-dessous est celle utilisée pour l'accomplissement de cette tâche.

```
setwd("C:/Users/usager/Dropbox/Partage_Roland/ProjetHEC/Redaction")

########################################
## connection opérationnelle "ConnOper" ##
########################################
path1 <-"C:/Users/usager/Dropbox/Partage_Roland/ProjetHEC/Redaction/Northwind.MDB"
ConnOper<-odbcConnectAccess(path1)

#####################################
## connection DataWareHouse "ConnDW" ##
#####################################
path2 <- "C:/Users/usager/Dropbox/Partage_Roland/ProjetHEC/Redaction/NorthwindDataWarehouse2000.mdb"
ConnDW<-odbcConnectAccess(path2)
```

Aussi, notons-nous que pour des raison de bon fonctionnement lors de l'execution des nos codes dans le ce fichier *Rmd* utilisé dans le cadre notre étude, de même que dans le but de l'utilisation de la base DataWareHouse, il est nécessaire de procéder à la suppression de tous les champs contenus dans ladite base. Ce qui fait appel aux comandes ci-dessous:

```
sqlDrop(ConnDW,"Suppliers",errors=FALSE)
sqlDrop(ConnDW,"Customers",errors=FALSE)
sqlDrop(ConnDW,"Employees",errors=FALSE)
sqlDrop(ConnDW,"Products",errors=FALSE)
sqlDrop(ConnDW,"DimDates",errors=FALSE)
sqlDrop(ConnDW,"OrdersFact",errors=FALSE)
```

Comme nous pouvons le constater, la base DataWareHouse, est belle et bien vide. Nous pourrons donc procéder au transfert des données dans celle-ci.

```
sqlTables(ConnDW)$TABLE_NAME[which(sqlTables(ConnDW)$TABLE_TYPE=="TABLE")]
```

```
# character(0)
```

En général, une procédure d'extraction nécessite l'utilisation des commandes *sqlQuery* et *sqlSave*. La première nécessite deux paramètres très important à savoir la *connection* vers la base de données opérationnelle et la *requête SQL* qui traduit les champs à extraire. La deuxième quant à elle permet de sauvegarder les champs extraits dans la nouvelle base de données DataWareHouse.

Les éléments ou encore *dimension* (sous de la forme de tableau) à extraire sont ceux inscrits sur notre schéma en étoile (voir *figure 3*). Il s'agit entre autre de :

Dimension Suppliers

```
Suppliers <- data.frame(sqlQuery(ConnOper, "select SupplierID, CompanyName,
                                 Country from Suppliers"))
head(Suppliers)
```

```
#   SupplierID                       CompanyName Country
# 1          1                     Exotic Liquids      UK
# 2          2         New Orleans Cajun Delights     USA
# 3          3          Grandma Kelly's Homestead     USA
# 4          4                      Tokyo Traders   Japan
# 5          5 Cooperativa de Quesos 'Las Cabras'   Spain
# 6          6                         Mayumi's   Japan
```

```
sqlSave(ConnDW, Suppliers, addPK=TRUE,rownames = FALSE)
```

Dimension Customers

```
Customers <- data.frame(sqlQuery(ConnOper, "select CustomerID, CompanyName,
                                           City, Country from Customers"))
head(Customers)
```

```
#   CustomerID                       CompanyName        City Country
# 1      ALFKI                Alfreds Futterkiste      Berlin Germany
# 2      ANATR Ana Trujillo Emparedados y helados México D.F.  Mexico
# 3      ANTON            Antonio Moreno Taquería México D.F.  Mexico
# 4      AROUT                   Around the Horn      London      UK
# 5      BERGS                Berglunds snabbköp       Luleå  Sweden
# 6      BLAUS            Blauer See Delikatessen    Mannheim Germany
```

```
sqlSave(ConnDW, Customers, addPK=TRUE,rownames = FALSE)
```

Dimension Employees

```
Employees <- data.frame(sqlQuery(ConnOper, "select EmployeeID, Title,
                                           Country from Employees"))
head(Employees)
```

```
#   EmployeeID                Title Country
# 1          1  Sales Representative     USA
# 2          2 Vice President, Sales     USA
# 3          3  Sales Representative     USA
# 4          4  Sales Representative     USA
# 5          5         Sales Manager      UK
# 6          6  Sales Representative      UK
```

```
sqlSave(ConnDW, Employees, addPK=TRUE,rownames = FALSE)
```

Notons que les dimensions ci-haut ont été extraites de la base opérationnelle vers la base DataWareHouse en utilisant une requête simple sur une table qui ne nécessite aucune aucune jointure avec d'autres tables. Par contre les autres mentionnées ci-desous nécessite une attention particulière. En effet, leur extraction est réalisée par l'entremise d'une jointure de plusiers tables. Il s'agit de:

Dimension Products

En effet, étant connecté à un autre tableau, le processus d'extraction de la dimension "Products", nécessite une jointure de deux tables "Products" et "Categories" étant donné que nous avons besoin ici du nom de la Categories pour un bon affichage lors de la visualisation . Ainsi, l'on procède à la jointures à travers la clé (*ProductID*) qui les relie. D'où la commande ci-dessous:

```
Products <-sqlQuery(ConnOper, "
                        SELECT
                                              A.ProductID,
                                              A.ProductName,
                                              B.CategoryName
                        FROM
                                              Products A,
                                              Categories B
                        WHERE
                                              A.CategoryID = B.CategoryID
                       ")
head(Products)
```

```
#   ProductID        ProductName CategoryName
# 1         1               Chai    Beverages
# 2         2              Chang    Beverages
# 3        24 Guaraná Fantástica    Beverages
# 4        34      Sasquatch Ale    Beverages
# 5        35     Steeleye Stout    Beverages
# 6        38      Côte de Blaye    Beverages
```

```
sqlSave(ConnDW, Products, addPK=TRUE,rownames = FALSE)
```

La génération de la dimension Date nécessite l'installation du package "**lubridate**" d'où la commande ci-dssous.

```
#install.packages("lubridate")
library(lubridate)
```

Dimension Dates

La dimension Date est une table qui a un enregistrement par jour. Selon la période utilisée dans l'entreprise, l'on pourrait définir le début et la fin de la dimension Date. Par exemple, la dimension Date peut commencer du 1er janvier 1980 au 31 décembre 2030. Dans notre cas elle commence du 31 Décembre 1993 au 25 Juin 1999. À partir de cette dimension Date, nous générons les Semaines-Mois-Trimestres-Années. Notons que la dimension Date fait intervenir une clé étrangère. Dans notre cas, la clé étrangère est l'élément *DimDates*. Cette clé est celle qui permettra de relier la dimension date au tableau des effects (*Orderfacts*).

```
n=2000 # Séquence de départ
SeqDate <- as.Date(1:n, origin = "1993-12-31")
DimDates<-data.frame(cbind(DateID=1:n, date=SeqDate,Mois=month(SeqDate,abbr=FALSE,
        label = TRUE),Quarter=quarter(SeqDate,with_year = TRUE),Annee=year(SeqDate)))
DimDates$DateID <- as.integer(DimDates$DateID)
head(DimDates)
```

```
#   DateID date Mois Quarter Annee
# 1      1 8766    1  1994.1  1994
# 2      2 8767    1  1994.1  1994
# 3      3 8768    1  1994.1  1994
```

```
# 4      4 8769    1  1994.1  1994
# 5      5 8770    1  1994.1  1994
# 6      6 8771    1  1994.1  1994
```

```
sqlSave(ConnDW, DimDates, addPK=TRUE,rownames = FALSE)
```

Table des faits: OrdersFact

L'OrderFact est une table de faits ou une entité de faits définie dans un schéma en étoile ou en flocon qui, stocke des mesures d'évaluation de l'activité comme par exemple la *Quantité* et le *Prix de Vente*. Les tables ou les entités de faits contiennent également des clés externes qui les relient aux tables de dimensions. Les clés externes lient chaque ligne de données de la table de faits aux dimensions et aux niveaux correspondants. Les entités de faits sont celles qui réunissent toutes les autres tables de dimensions, leur extraction vient en dernière position. La création l'OrderFact se fera en suivant quelques étapes qu'il urge d'expliquer.

- Par une requête de jointure entre les tables Orders details et Order, l'on extrait les différentes colonnes de chacune des tables. De même, nous créyons également une colonne *Date.*

```
OrdersFac <-sqlQuery(ConnOper, "
                        SELECT
                     A.ProductID,
                     B.CustomerID,
                     B.EmployeeID,
                     B.OrderDate,
                     A.UnitPrice,
                     A.Quantity,
                     0 as DateID
                     FROM
                     [Order Details] A,
                     Orders B
                     WHERE
                     A.OrderID = B.OrderID
                     ")
```

- Dans les données de la requête précédente, on a un champ OrderDate qui est une date. Nous transformons cette date en une clé (DateID). Ainsi donc pour chaque Date l'on cherche l'ID numérique correspondant en prenant comme référence le début de la dimension date (31 Décembre 1993).

```
######################
## Ajouter date dim ##
######################
Dates2=as.Date(OrdersFac$OrderDate)
for (i in 1:length(Dates2)){
  OrdersFac$DateID[i] <- which(as.Date(1:n, origin = "1994-01-01")==Dates2[i])
}
```

- On sélectionne ensuite les champs qui nous intéresse.

```
OrdersFact<-data.frame(OrdersFac[,c(1,2,3,7,5,6)])
head(OrdersFact)
```

```
#   ProductID CustomerID EmployeeID DateID UnitPrice Quantity
# 1        11      VINET          5    215      14.0       12
# 2        42      VINET          5    215       9.8       10
# 3        72      VINET          5    215      34.8        5
# 4        14      TOMSP          6    216      18.6        9
```

```
# 5         51       TOMSP           6    216       42.4        40
# 6         41       HANAR           4    219        7.7        10
```

- Pour ajouter le Supplier ID dans la table des faits, nous exécutons une requête par jointure des tables "Products" et "Suppliers" à travers la clé *SupplierID*

```
#########################
## Ajouter suppliers ID ##
#########################
SupplProdID  <-sqlQuery(ConnOper, "
                      SELECT
                    A.ProductID,
                    B.SupplierID
                    FROM
                    Products A,
                    Suppliers B
                    WHERE
                    A.SupplierID = B.SupplierID
                    ")
```

- En fin, l'on procède à une juxtaposition (par la commande *merge*) du résultat des deux requêtes précédentes selon le champ *ProductsID*

```
#####################################
## merge SupplProdID et OrdersFact ##
#####################################

OrdersFact <- merge(OrdersFact, SupplProdID, by="ProductID")

#############################################################
## Sauvegarde de OrdersFact dans la base DataWareHouse ##
#############################################################

sqlSave(ConnDW, OrdersFact, addPK=TRUE,rownames = FALSE)
```

Comme nous pouvons le constater, la commande ci-dessous nous permet de voir que les tables sont bien créées dans la base DataWareHouse. En effet, la commande *sqlTables* permet de voir les différents tableaux presents dans la base DataWareHouse.

```
sqlTables(ConnDW)$TABLE_NAME[which(sqlTables(ConnDW)$TABLE_TYPE=="TABLE")]
```

```
# [1] "Customers"  "DimDates"   "Employees"  "OrdersFact" "Products"
# [6] "Suppliers"
```

En général, dans un système commercial (Microsoft SQL Server) l'on crée un *Cub OLAP* qui contiendrait toutes les informations qui serviront à générer des rapports pré-synthétisés. Dans notre étude, nous lançons directement nos requêtes à partir de la base DataWareHouse.

Une fois la procédure ETL terminée, nous pouvons maintenant procéder à la visualisation des données.

Visualisation

À ce niveau, comme mentionné en introduction et comme son nom l'indique, il sera question de procéder à une visualisation des différents éléménts obtenus dans la base DataWareHouse après extraction et transformation. En effet, nous aurons à produire des graphiques dans le but de faire ressortir les analyses pertinentes issues des différentes transactions enregistrées. Pour la production de bons graphiques, l'outil priorisé serait la librairie **ggplot2**. Aussi, aurions-nous à recourir à d'autres outils. La commande ci-dessous donne un aperçu de ces derniers.

```
library(ggplot2)
library(lubridate)
library(forecast)
library(grid)
library(gridExtra)
library(RGraphics)
library(plyr)
```

Serie chronologique

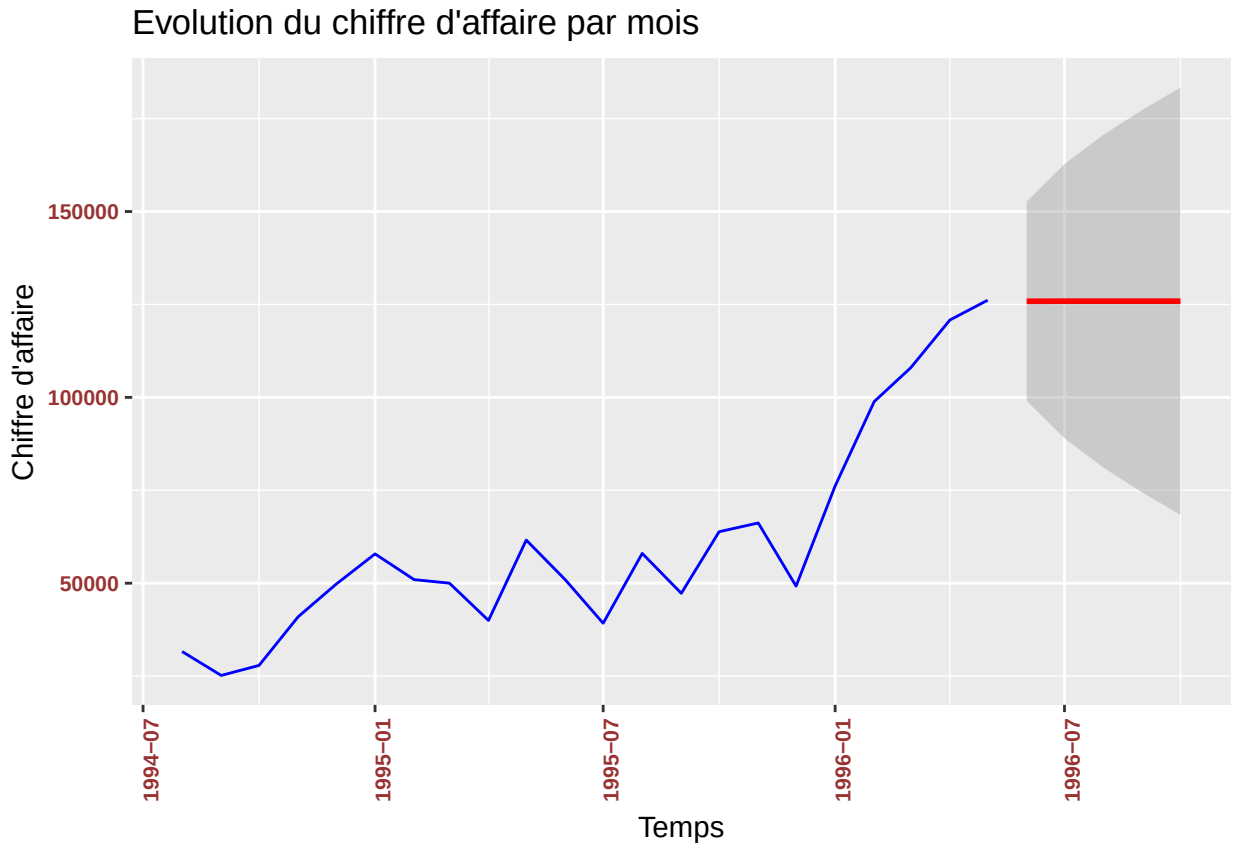
Comme l'indique son nom, les séries chronologiques sont mises en exergue dans le but d'analyser l'évolution des transaction au cours du temps. Ici, nous aurions à visualiser à chaque mois les évolutions du chiffre d'affaire et des commandes au cours du temps lors des transactions.

L'observation du chiffre d'affaire et des commandes à travers le temps, nécessite une travail au préalable. En effet, pour obtenir le chiffre d'affaire par mois, nous exécutons une requête de jointure entre les tables "DimDates" et "OrdersFacts" avec comme clé de liaison la *DateID*. La colonne chiffre d'affaire est ainsi obtenue par le produit des colonnes "Prix" et "Qauntity". L'on procède à l'aggrégation des du résultat des requêtes précédentes par année et par mois simultanément. En suite nous appliquons le modèle **ARIMA**(voir procédure ARIMA sous R) sur le chiffre d'affaire obtenu. Pour le second graphique, l'on a conduit la même démarche sauf qu'à ce niveau nous comptons juste le nombre de commandes effectuées par mois et par année à travers la fonction *COUNT*.

```
  #########
## Requête ##
  ########

ChAffr <-sqlQuery(ConnDW,"
                 SELECT
                  A.UnitPrice,
                  A.Quantity,
                  B.Annee,
                  B.Quarter,
                  B.Mois,
                  B.DateID,
                  A.Quantity*A.UnitPrice as Total
                  FROM
                  OrdersFact A,
                  DimDates B
                  WHERE
                  A.DateID = B.DateID" )


#####################################################################
## Calcul de l'évolution du chiffre d'affaire au cours du temps##
```

```
#####################################################################

### Aggrégation des données par années et par mois

h=5 ##Horizon de prédiction sur cinq(05) mois
ChAffAgrMois=aggregate(ChAffr$Total ~ ChAffr$Annee + ChAffr$Mois, ChAffr, sum)
names(ChAffAgrMois) = c("Annee","Mois","Total")
ChAffAgrMois <- ChAffAgrMois[order(ChAffAgrMois$Annee, ChAffAgrMois$Mois),]
ChAffAgrMois <- ChAffAgrMois[-dim(ChAffAgrMois)[1],]
objetTS = ts(ChAffAgrMois$Total, frequency = 12,
             start = c(ChAffAgrMois$Annee[1],ChAffAgrMois$Mois[1]))

### Application du modèle ARIMA

fit <- arima(objetTS, order=c(1,1,1))
pred <- predict(fit, n.ahead=h)
sample<-ChAffAgrMois$Total
forecast<-pred$pred
upper<-forecast+2*pred$se
lower<-forecast-2*pred$se
debut=paste0(ChAffAgrMois$Annee[1],"-",ChAffAgrMois$Mois[1],"-01")
seqDate=seq(as.POSIXct(debut), by = "month", length.out = length(ChAffAgrMois$Annee)+h)

### Plot
df1 = data.frame(time = seqDate[1:length(ChAffAgrMois$Annee)],
                 M = sample, isin = "observations")
df2 = data.frame(time = seqDate[(length(ChAffAgrMois$Annee)+1):
      (length(ChAffAgrMois$Annee)+h)], M = forecast , isin    = "my_forecast")
df3 = data.frame(time = seqDate[(length(ChAffAgrMois$Annee)+1):
      (length(ChAffAgrMois$Annee)+h)], M = upper ,isin = "upper_bound")
df4 = data.frame(time = seqDate[(length(ChAffAgrMois$Annee)+1):
      (length(ChAffAgrMois$Annee)+h)], M = lower, isin = "lower_bound")

PLOT11= ggplot(df1, aes(x = time, y = M)) + geom_line(colour='blue') + geom_smooth(aes(x=time, y=M, yma
      colour='red', data=df2, stat='identity') + theme(axis.text.x = element_text(face="bold", color="#
                          axis.text.y = element_text(face="bold", color="#993333",size=8, angle=0)) + la
```

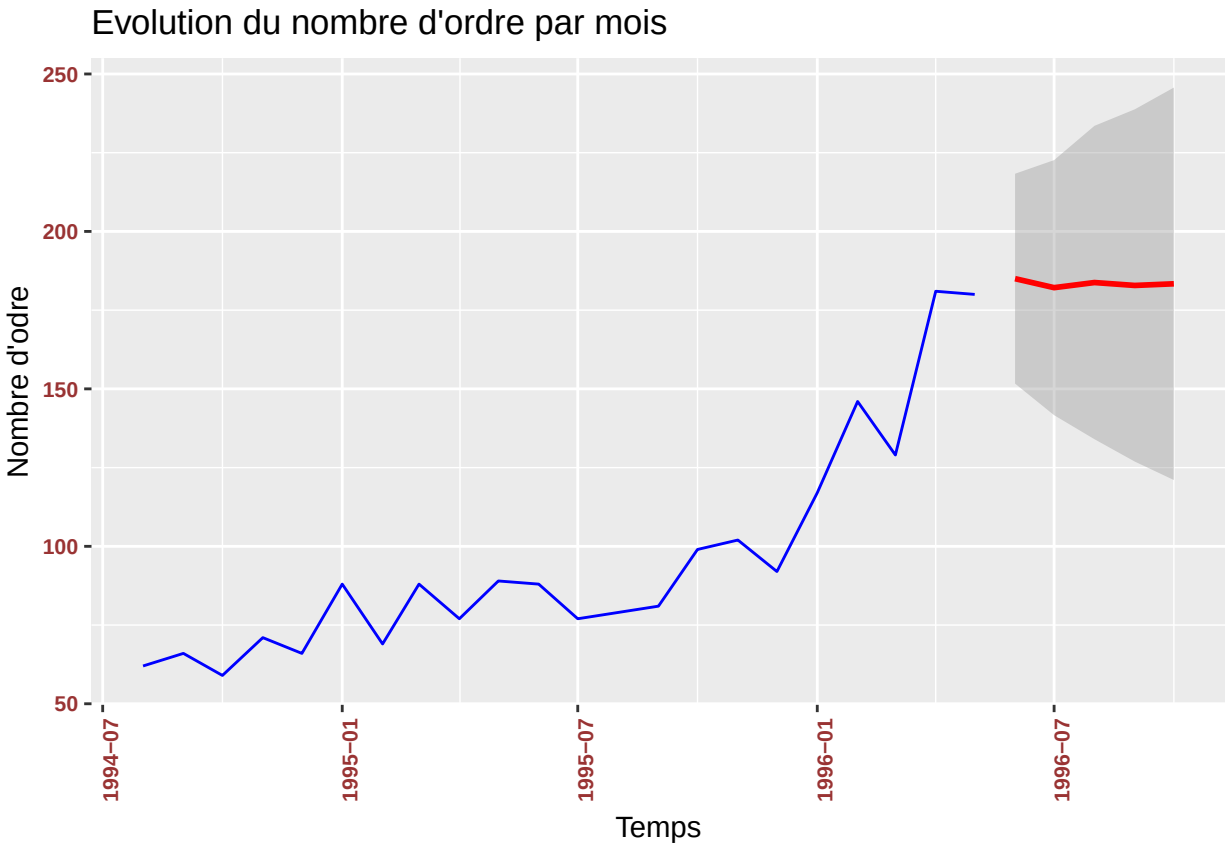
Evolution du chiffre d'affaire mensuel au cours du temps



En se basant sur le graphique ci-dessus, nous constatons à première vue que la tendance du chiffre d'affaire se trouve être également à la hausse. Ce qui se confirme par la hausse de la demande des produits. Cette hausse du chiffre d'affaire est restée constante sur les cinq (05) prochains mois avec un intervalle de confiance de 95%. Ceci fait suite aux prédictions faites dans le future. Cette tendance prévisionnelle a été mis en place par un modèle de prédiction à travers la fonction *FORECAST*.

```
#####################################################################
## Calcul de Evolution du nombre d'order (commandes) par mois ##
#####################################################################

### Aggrégation des données par années et par mois

h=5 ##Horizon de prédiction sur cinq(05) mois
NombreOdreMois <- count(ChAffr$DateID, c('ChAffr$Annee','ChAffr$Mois'))
names(NombreOdreMois) = c("Annee","Mois","Effectif")
NombreOdreMois <- NombreOdreMois[order(NombreOdreMois$Annee, NombreOdreMois$Mois),]
NombreOdreMois <- NombreOdreMois[-dim(NombreOdreMois)[1],]
objetTS = ts(NombreOdreMois$Effectif, frequency = 12,
           start = c(NombreOdreMois$Annee[1],NombreOdreMois$Mois[1]))

### Application du modèle ARIMA

fit <- arima(objetTS, order=c(1,1,1))
pred <- predict(fit, n.ahead=h)
sample<-NombreOdreMois$Effectif
forecast<-pred$pred
upper<-forecast+2*pred$se
lower<-forecast-2*pred$se
debut=paste0(NombreOdreMois$Annee[1],"-",NombreOdreMois$Mois[1],"-01")
seqDate=seq(as.POSIXct(debut), by = "month", length.out = length(NombreOdreMois$Annee)+h)

### Plot

df11 = data.frame(time = seqDate[1:length(NombreOdreMois$Annee)],
      M = sample, isin = "observations")
df22 = data.frame(time = seqDate[(length(NombreOdreMois$Annee)+1):
      (length(NombreOdreMois$Annee)+h)], M = forecast , isin    = "my_forecast")
df33 = data.frame(time = seqDate[(length(NombreOdreMois$Annee)+1):
      (length(NombreOdreMois$Annee)+h)], M = upper ,isin = "upper_bound")
df44 = data.frame(time = seqDate[(length(NombreOdreMois$Annee)+1):
      (length(NombreOdreMois$Annee)+h)], M = lower, isin = "lower_bound")
PLOT2= ggplot(df11, aes(x = time, y = M)) + geom_line(colour='blue') + geom_smooth(aes(x=time, y=M, yma
     colour='red', data=df22, stat='identity') + theme(axis.text.x = element_text(face="bold", color="#
     axis.text.y = element_text(face="bold", color="#993333",size=8, angle=0)) + labs(x = "Temps",y="Nor
```

Evolution du nombre d'ordre mensuelle au cours du temps



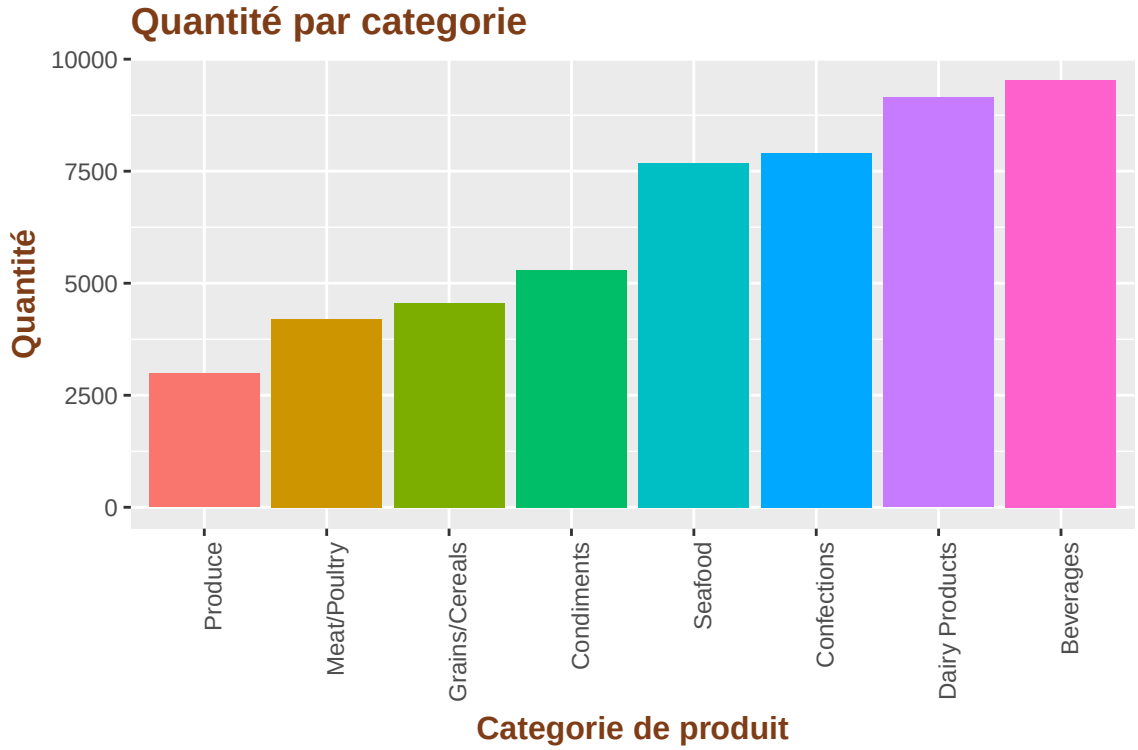
De la figure ci-dessus, il en résulte une tendance à la hausse des commandes des produits. Cette hausse des commandes des produits est restée constante sur les cinq (05) prochains mois avec un intervalle de confiance de 95%. Ceci fait suite aux prédictions faites dans le future.

Notons que nous pouvons également visualiser l'évolution du chiffre d'affaire et des commandes des produits par **Jour**, par **Trimestre** et par **Année**.

Nombre d'articles commandé par categorie de produits

```
ChAffCateg <- sqlQuery(ConnDW, "
                        SELECT
                                A.UnitPrice,
                                A.Quantity,
                                B.CategoryName,
                                A.Quantity*A.UnitPrice as Total
                                FROM
                                OrdersFact A,
                                Products B
                                WHERE
                                A.ProductID = B.ProductID")




################################################################
## Calcul du nombre de Quantité par categorie de produits ##
################################################################

ChAffCategAgQ=aggregate(Quantity ~ CategoryName, ChAffCateg, sum)

ChAffCategAgQ <-data.frame(PRODUITS=ChAffCategAgQ$CategoryName,
                           QUANTITE=ChAffCategAgQ$Quantity)

ChAffCategAgQ$PRODUITS<-factor(ChAffCategAgQ$PRODUITS,
                levels = ChAffCategAgQ$PRODUITS[order(ChAffCategAgQ$QUANTITE)])
```

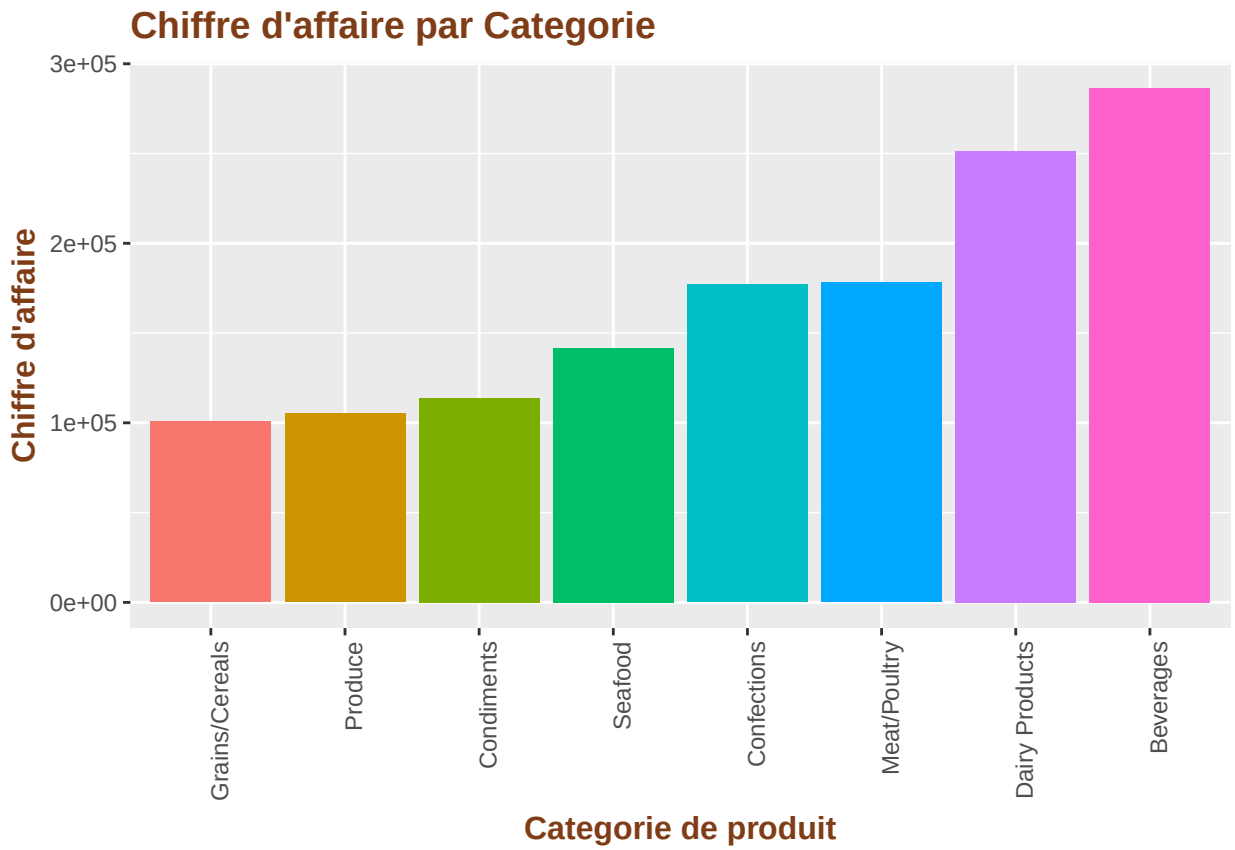
Du graphique ci_dessous, il en ressort que parmi les huit (08) différentes catégories de produits achetées, la catégories **Beverages** est celle la plus demandée (avec une quantité de plus 8500 tonnes) suivies respcetivement de **Dairy Products**, **Confections**, **Seafood**, **Condiments**, **Grains/Cereals**, **Meat/Poultry** et **Produce**.



Chiffre d'affaire par categorie de produits

```
#############################################################
## Calcul du chiffre d'affaire par categorie de produits ##
#############################################################

ChAffCategAgV=aggregate(Total ~ CategoryName, ChAffCateg, sum)

ChAffCategAgV <-data.frame(PRODUITS=ChAffCategAgV$CategoryName,
                                  ChiffreAffaire=ChAffCategAgV$Total)

ChAffCategAgV$PRODUITS<-factor(ChAffCategAgV$PRODUITS,
                    levels = ChAffCategAgV$PRODUITS[order(ChAffCategAgV$ChiffreAffaire)])
```

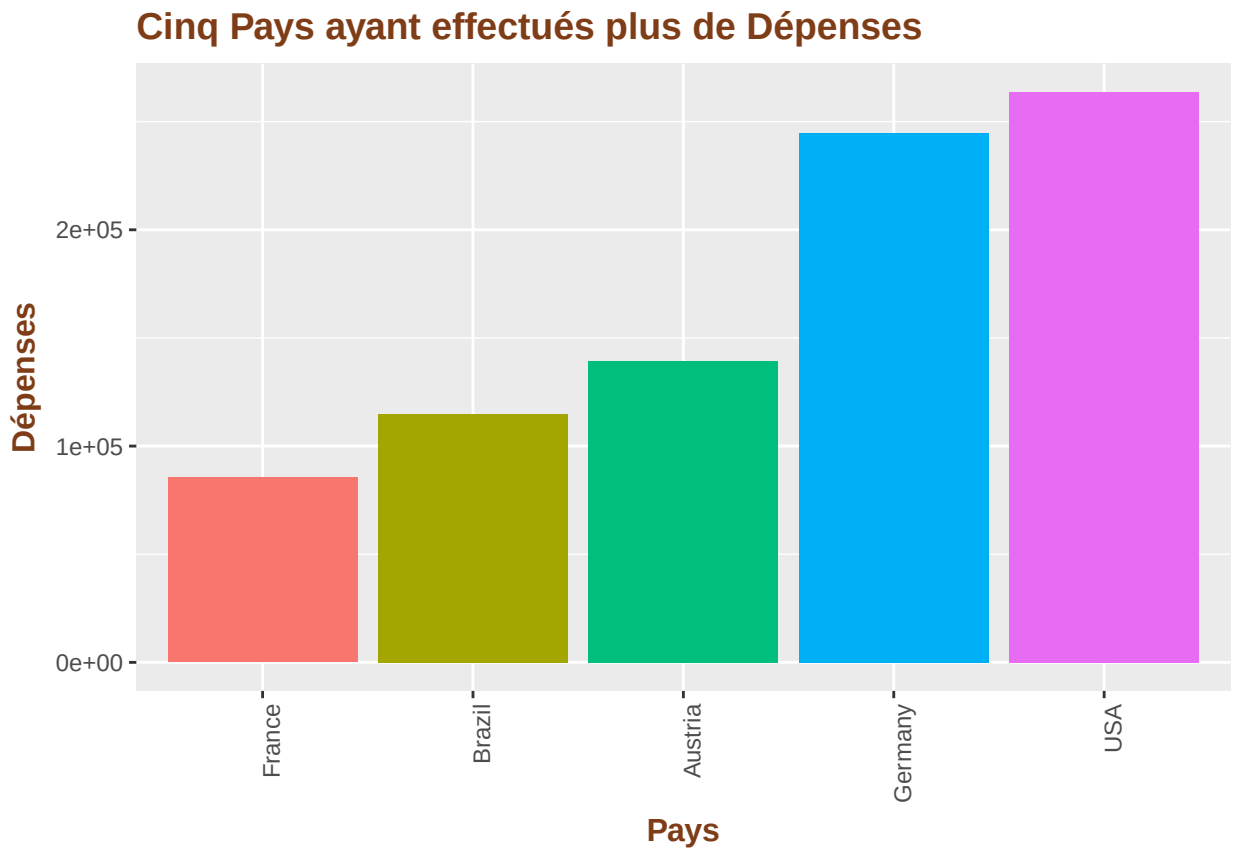
Comme le montre la figure ci-dessous pour ce qui concerne le chiffe d'affaire des différentes catégories, on constate que la catégorie **Beverages** réunie à elle seule près de $300,000$\$ suivies respcetivement de celles des catégories **Dairy Products**, **Confections**, **Seafood**, **Condiments**, **Grains/Cereals**, **Meat/Poultry** et **Produce**. En effet, celà semble normal dans la mesure où il s'agit du même ordre en ce qui concerne les quantité. La hiérarchie n'a été que respectée.



Tendance des cinq pays ayant effectué plus de dépenses

Du graphique ci-dessous, il en résulte que des cinq(05) ayant effectué plus de dépenses, les **USA** sont en pôle position suivis de la Germany, l'Australi, du Brazil et de la France respectivement en deuxième, troisième, quatrième et cinquième positio derrière les USA. En effet, les USA sont réputés être le pays qui dépense le plus quand il s'agit non seulement de l'armement mais également dans les produits consommables. Aussi, faudrait-t-il remarqué que les demandes de ces pays sont dûes au nombre d'invididus vivant dans ces pays.

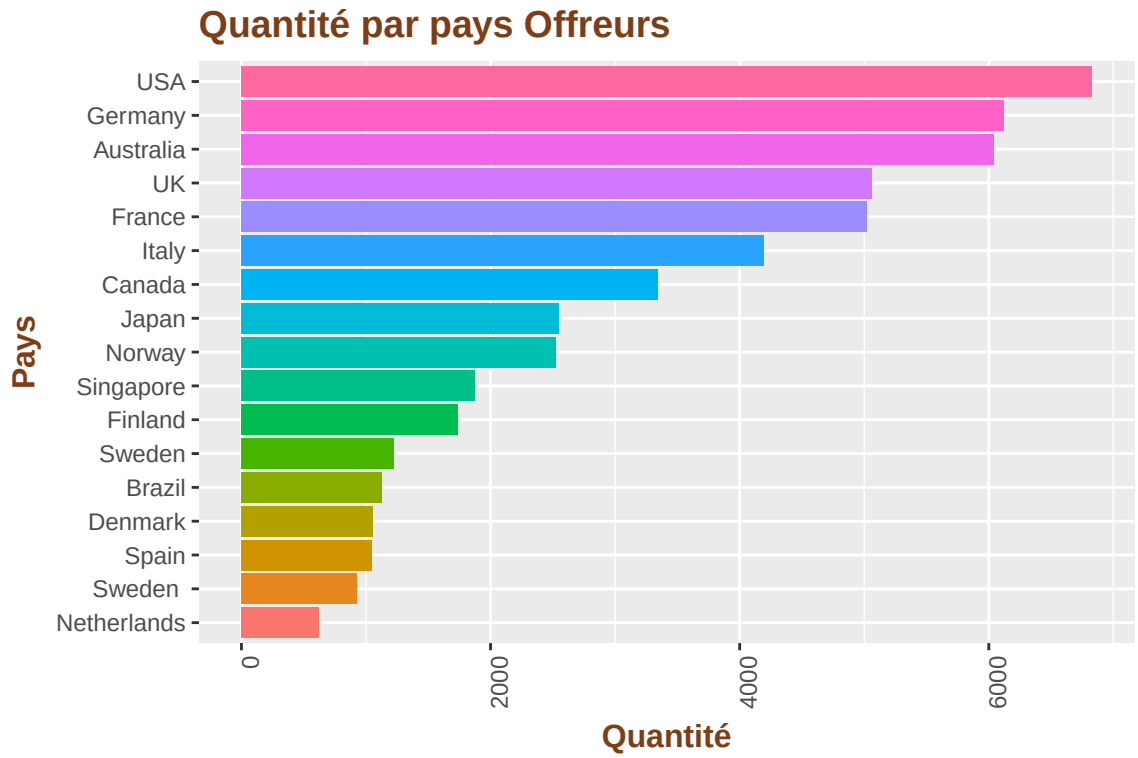
```
ChiffreAffairePaye <- sqlQuery(ConnDW, "
                        SELECT
                               A.UnitPrice,
                               A.Quantity,
                               B.Country,
                               A.Quantity*A.UnitPrice as Total
                         FROM
                               OrdersFact A,
                               Customers B
                         WHERE
                               A.CustomerID = B.CustomerID")

ChiffreAffairePayeAgr=aggregate(Total~Country, ChiffreAffairePaye, sum)
ChiffreAffairePayeAgr =(ChiffreAffairePayeAgr[order(ChiffreAffairePayeAgr$Total,
                                                    decreasing = TRUE),])[1:5,]


ChiffreAffairePayeAgr <-data.frame(Pays=ChiffreAffairePayeAgr$Country,
                                   Depenses=ChiffreAffairePayeAgr$Total)
ChiffreAffairePayeAgr$Pays<-factor(ChiffreAffairePayeAgr$Pays,
          levels = ChiffreAffairePayeAgr$Pays[order(ChiffreAffairePayeAgr$Depenses)])

ggplot(data = ChiffreAffairePayeAgr,aes(Pays,
               Depenses,fill=Pays))+guides(fill=FALSE)+geom_bar(stat = "identity")+
  labs(title = "Cinq Pays ayant effectués plus de Dépenses", y = "Dépenses", x = "Pays" )+
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        title  = element_text(colour = "#7F3D17",size=12, face='bold'))
```


Cinq Pays ayant effectués plus de Dépenses
Dépenses
2e+05
1e+05
0e+00
France
Brazil
Austria
Germany
USA
Pays

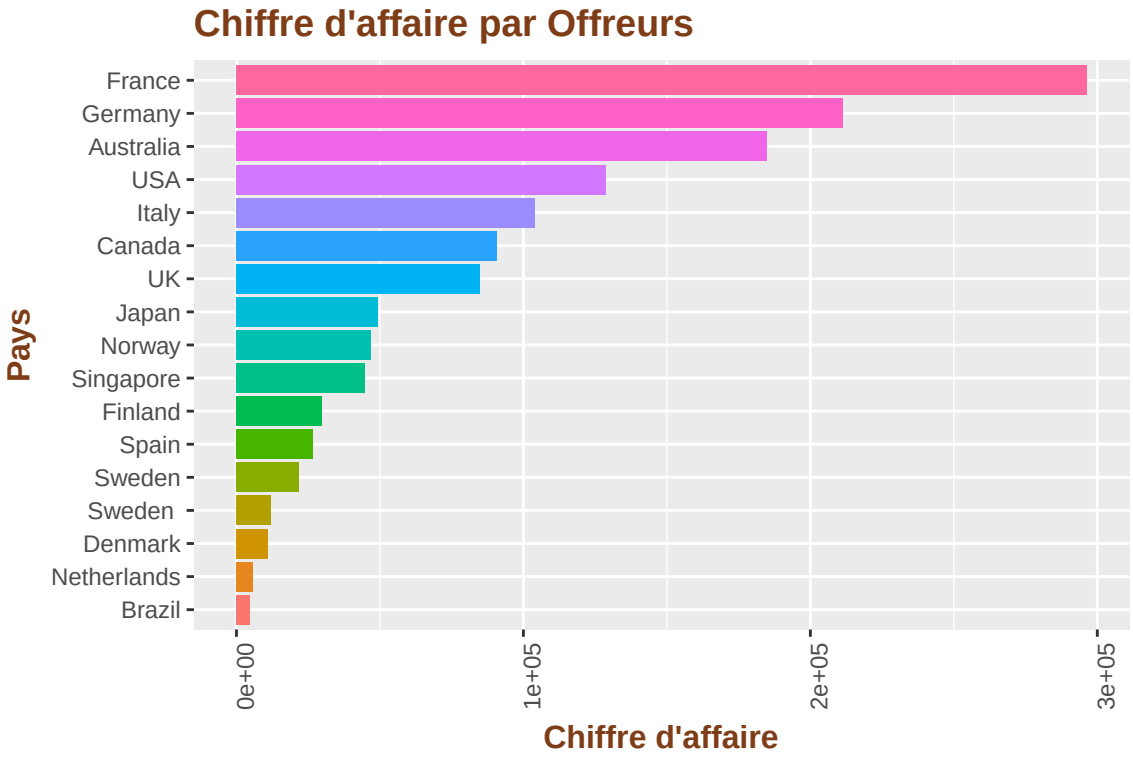
Quantités vendues et chiffre d'affaire des pays offreurs (Vendeurs)

```
######### nombre d'article vendu par offreurs et leur chiffre d'affaire
ChAffSupl <- sqlQuery(ConnDW, "
                      SELECT
                      A.UnitPrice,
                      A.Quantity,
                      B.Country,
                      A.Quantity*A.UnitPrice as Total
                      FROM
                      OrdersFact A,
                      Suppliers B
                      WHERE
                      A.SupplierID = B.SupplierID")



#####################################################################
## Calcul du nombre de quanttités offertes par les Vendeurs ##
#####################################################################

ChAffSuplAgQ=aggregate(Quantity ~ Country, ChAffSupl, sum)

ChAffSuplAgQ <-data.frame(Pays=ChAffSuplAgQ$Country,
                          QUANTITE=ChAffSuplAgQ$Quantity)
ChAffSuplAgQ$Pays<-factor(ChAffSuplAgQ$Pays,
                  levels = ChAffSuplAgQ$Pays[order(ChAffSuplAgQ$QUANTITE)])

################################################
## Calcul du Chiffres d'affaire des Vendeurs ##
################################################

ChAffSuplAgV=aggregate(Total ~ Country, ChAffSupl, sum)

ChAffSuplAgV <-data.frame(Pays=ChAffSuplAgV$Country,
                          ChiffreAffaire=ChAffSuplAgV$Total)
ChAffSuplAgV$Pays<-factor(ChAffSuplAgV$Pays,
                   levels = ChAffSuplAgV$Pays[order(ChAffSuplAgV$ChiffreAffaire)])
```

De la figure ci-dessous, il revient de dire qu'au nombre des 17 pays ayant offert leurs produits, les USA sont ceux ayant vendus plus leurs produits avec une quantité de plus de 6000 articles (toutes catégories confondues). Les Pays-Bas quant à eux sont ceux ayant moins vendu leurs produits et ce avec une quantité de moins de 1000 articles (toutes catégories confondues) durant la période considérée.



Contrairement à ce que l'on pouvait s'attendre, il a été constaté que le pays ayant effectué le chiffre d'affaire le plus élevé est la France alors que celui-ci était le cinquième pays ayant vendus plus de produits après le Royaume Unis, l'Australie l'Allemagne et les USA. De même celui ayant effectué le chiffre d'affaire le plus bas s'avère être le Brasil. En effet, ce constat pouvait s'expliquer par la différence des prix au niveau de chaque produit. Les produits les plus vendus parpourraient s'avérer être ceux ayant un prix bas.



Le tableau ci-dessous nous donne une idée sur les quantités vendus par les pays impliqués dans les transactions enregistrées au niveau de la base de données opérationnelle et de leurs chiffres d'affaires.

Pays	QUANTITE	ChiffreAffaire
USA	6828	128844.15
UK	5064	84710.60
Sweden	928	12072.60
Sweden	1223	21789.80
Spain	1050	26768.80
Singapore	1878	44935.80
Norway	2526	46897.20
Netherlands	623	5901.35
Japan	2551	49211.50
Italy	4197	104011.50
Germany	6120	211540.09
France	5023	296381.75
Finland	1736	29804.00
Denmark	1056	10884.50
Canada	3344	90899.70
Brazil	1125	4782.60
Australia	6045	185022.65

Croissement de deux variables qualitatives: Trimestre et Categorie (Tableau de Contingence)

Pour le croisement de ces deux variables, nous mergeons deux requêtes selon le "ProductID". La première est similaire à celle de la série chronologique faite ci-haut. La deuxième consiste à chercher le nom de la catégorie correspondant au "ProductID". En suite, nous utilisons la fonction *XTABS* pour obtenir un tableau croisé dynamique. Ainsi, chaque cellule obtenue représente le nombre d'articles vendus par Catégorie et par Trimestre.

```
ChAffQuarter <-sqlQuery(ConnDW, "
                                SELECT
                                A.UnitPrice,
                                A.Quantity,
                                A.ProductID,
                                B.Quarter,
                                A.Quantity*A.UnitPrice as Total
                                FROM
                                OrdersFact A,
                                DimDates B
                                WHERE
                                (A.DateID = B.DateID)")

ProdIDCateg <-sqlQuery(ConnDW, "
                                SELECT
                                      A.ProductID,
                                      A.CategoryName
                                FROM
                                      Products A")

MergeData=merge(ProdIDCateg, ChAffQuarter, by = "ProductID")
TbCroise=xtabs(MergeData$Quantity ~ MergeData$CategoryName+paste("Trimestre",
                    substr(MergeData$Quarter, 6, 6)), MergeData)
tbl2 <-  tableGrob(round(TbCroise, 0))
grid.arrange(tbl2, ncol=1)
```

	Trimestre 1	Trimestre 2	Trimestre 3	Trimestre 4
Beverages	2839	3239	1593	1861
Condiments	1561	1592	894	1251
Confections	2670	2399	1067	1770
Dairy Products	2858	2622	1589	2080
Grains/Cereals	1533	1185	833	1011
Meat/Poultry	1602	956	601	1040
Produce	832	1142	501	515
Seafood	2183	2213	1256	2029

Pour tester l'indépendance entre ces deux variables qualitatives, nous utilisons ici le test de *Khi-deux*. Il en découle le résultat ci-dessou.

```
test <- chisq.test(TbCroise)
if (test$p.value<0.05)
  print(" Il ya une dependance entre les variables: trimestre et Catogorie de produit")
```

```
# [1] " Il ya une dependance entre les variables: trimestre et Catogorie de produit"
```

La répartition du chiffre d'affaire par title d'employés

Ici, nous exécutons une requête de jointure des tables "OrdersFact" et "Employees" pour calculer le montant des ventes pour chaque ordre détail par employé. Puis nous procédons à l'aggrégation de cette requête par "titre d'employé".

```
ChAffEmpTil <- sqlQuery(ConnDW, "
                        SELECT
                              A.UnitPrice,
                              A.Quantity,
                              B.Title,
                              A.Quantity*A.UnitPrice as Total
                        FROM
                              OrdersFact A,
                              Employees B
                        WHERE
                              A.employeeID = B.employeeID")
ChAffEmpTilAg=aggregate(Total~Title, ChAffEmpTil, sum)

library(plotrix)
pie3D(ChAffEmpTilAg$Total, labels = ChAffEmpTilAg$Title,
      main = "Chiffre d'affaire par titre d'empoyés",
      explode=0.1, radius=.9, labelcex = 0.5,  start=1)
```