

Royaume du Maroc  
Haut-commissariat au plan  
Ecole des Sciences de l'Information  
Année Universitaire : 2023-2024  
Module : Data Modeling



## *Rapport de projet Data Modeling :*

Analyse et Prédiction de désabonnement des  
clients à travers Denodo

3<sup>ème</sup> ACI - ICSD

*Réalisé par :*  
*ELGHAZI Soufiane*  
*MAASRI Amine*  
*OUAHIB Yassine*  
*DAHMOU Youssef*

**Année Universitaire : 2023/2024**

# Table des matières

Liste des figures.....	3
Liste des Tableaux : .....	4
Introduction .....	5
I. Chapitre I : La virtualisation des données : Théorique .....	6
1. Contexte : .....	6
2. Data Virtualisation : Définition et avantages : .....	6
3. Benchmark des outils de virtualisation : Tableau comparatif.....	8
4. Denodo : Architecture et Installation .....	8
5. Conclusion : .....	10
II. Chapitre 2 : La Virtualisation des données : Pratique.....	11
1. Introduction : .....	11
2. Sources des données : .....	11
• Architecture : .....	11
• Bases de données : .....	12
• Création des Connections et des Vues virtuelles : .....	15
3. Conclusion : .....	21
III. Machine Learning et Dashbording : .....	22
1. Introduction : .....	22
2. ML avec Spark .....	22
3. Dashboard avec Power BI : .....	26
4. Conclusion : .....	28
Conclusion : .....	29
Bibliographie : .....	30

# Liste des figures

Figure 1 : Définition de Data virtualisation.....	6
Figure 2 : Avantages de data virtualisation .....	7
Figure 3 : Denodo Architecture .....	9
Figure 4 : Création de compte sous Denodo site .....	10
Figure 5 : Télécharger la version Community .....	10
Figure 6: Architecture des sources de données .....	11
Figure 7 : Création de la table Oracle.....	12
Figure 8 : Remplir la table Oracle avec Talend .....	12
Figure 9 : Chargement de Table MySQL.....	12
Figure 10 : Vérification de remplissage des données vers MySQL.....	12
Figure 11 : Création de la table Cassandra.....	13
Figure 12 : Remplir la table de Cassandra.....	13
Figure 13 : Table Cassandra Rempli .....	13
Figure 14 : Fichier Excel .....	13
Figure 15 : Table Sous Snowflake .....	14
Figure 16 : Table avec Hadoop HDFS.....	14
Figure 17 : Connexion réussie avec Oracle .....	15
Figure 18 : Requête sur La vue d'Oracle.....	15
Figure 19 : JDBC Driver pour MySQL .....	16
Figure 20 : Connexion réussie avec MySQL.....	16
Figure 21 : Requête sur La vue de MySQL.....	16
Figure 22 : Connexion réussie avec Cassandra .....	17
Figure 23 : Requête sur La vue de Cassandra .....	17
Figure 24 : Connexion réussie avec Excel.....	18
Figure 25 : Requête sur La vue de Excel .....	18
Figure 26 : Connexion réussie avec Snowflake .....	19
Figure 27 : Requête sur La vue de Snowflake .....	19
Figure 28 : Connexion réussie avec HDFS .....	20
Figure 29 : Requête sur La vue de HDFS.....	20
Figure 30 : Jointure des Vues.....	21
Figure 31 : Création de Spark session .....	22
Figure 32 : Connexion ODBC avec Python.....	22
Figure 33 : Requête SQL et création de curseur .....	23
Figure 34 : Convertir pandas DataFrame en Spark Dataframe .....	23
Figure 35 : Analyse et visualisation de churn en fonction d'autre features .....	23
Figure 36 : Clustering et Elbow Method.....	24
Figure 37 : Algorithmes de classification et Evaluations .....	25
Figure 38 : Random Forest Après SMOTE .....	25
Figure 39 : Spark Pipeline de Modèle Random Forest .....	26
Figure 40 : Power BI Odbc Installation.....	26
Figure 41 : Power BI connection1.....	27
Figure 42 : Power BI connection2.....	27
Figure 43 : Power BI Dashboard .....	28

## Liste des Tableaux :

<b>Tableau 1 :</b> Data Virtualisation VS Data Warehousing .....	7
<b>Tableau 2 :</b> Tableau Comparatif des outils de virtualisation .....	8

## Liste des Abréviations :

Abréviation	Acronyme
HDFS	Hadoop Distributed File System
JDBC	Java Database Connectivity
ODBC	Open Database Connectivity
BI	Business Intelligence
CSV	Comma-Separated Values
ML	Machine Learning
SMOTE	Synthetic Minority Over-sampling Technique
ANN	Réseau de Neurones Artificiels
XGBOOST	eXtreme Gradient Boosting
ROC	Receiver Operating Characteristic
AUC	Area Under the Curve

# Introduction

Le projet de data virtualisation que nous avons entrepris repose sur la mise en œuvre de Denodo en tant qu'outil central pour créer une couche abstraite permettant l'intégration transparente de données provenant de diverses sources. Dans le paysage complexe de la gestion des données, notre objectif est d'explorer les capacités de virtualisation offertes par Denodo pour faciliter l'accès, la gestion et l'analyse de données hétérogènes. Les différentes sources de données, allant des bases de données relationnelles classiques aux systèmes distribués Big Data, offrent un défi significatif que nous chercherons à résoudre en utilisant Denodo comme pierre angulaire de notre architecture.

Nous allons travailler avec des jeux de données pertinents sur le Churn ou désabonnement des clients, provenant de diverses sources telles que Snowflake, Oracle, Cassandra, Excel, MySQL et Hadoop (HDFS). Ces données variées reflètent la diversité des défis auxquels sont confrontées les entreprises modernes dans la gestion de leurs informations.

À travers ce rapport, nous détaillerons les étapes que nous avons suivies, de la sélection de Denodo comme outil de virtualisation à la création de vues virtuelles, l'utilisation de Spark (Py Spark) pour des tâches de machine Learning visant à prédire le désabonnement des clients, et enfin la visualisation des résultats via Power BI dans un objectif de reporting.

# I. Chapitre I : La virtualisation des données : Théorique

## 1. Contexte :

L'analyse du churn, ou désabonnement des clients, est devenue un impératif stratégique pour les entreprises évoluant dans des environnements concurrentiels. La capacité à anticiper les tendances de désabonnement offre un avantage significatif, justifiant ainsi notre démarche de mise en œuvre d'un projet de data virtualisation. Ce chapitre plonge dans l'écosystème complexe de la gestion des données, en mettant en lumière les défis inhérents et la nécessité d'une approche telle que la virtualisation pour surmonter ces obstacles.

## 2. Data Virtualisation : Définition et avantages :

La data virtualisation est une approche technologique qui permet d'accéder et d'intégrer des données provenant de sources hétérogènes, distribuées et disparates, sans nécessiter de mouvement physique des données. Elle crée une couche abstraite virtuelle au-dessus des différentes sources de données, fournissant ainsi une vue unifiée et intégrée, indépendamment de la localisation ou du format original des données. L'objectif principal de la data virtualisation est de simplifier et d'accélérer l'accès aux données pour les utilisateurs, tout en optimisant la gestion et en minimisant les coûts liés à la duplication des données. Cette approche favorise l'agilité dans la prise de décision en offrant une vision unifiée et en temps réel des informations disponibles au sein d'une organisation.

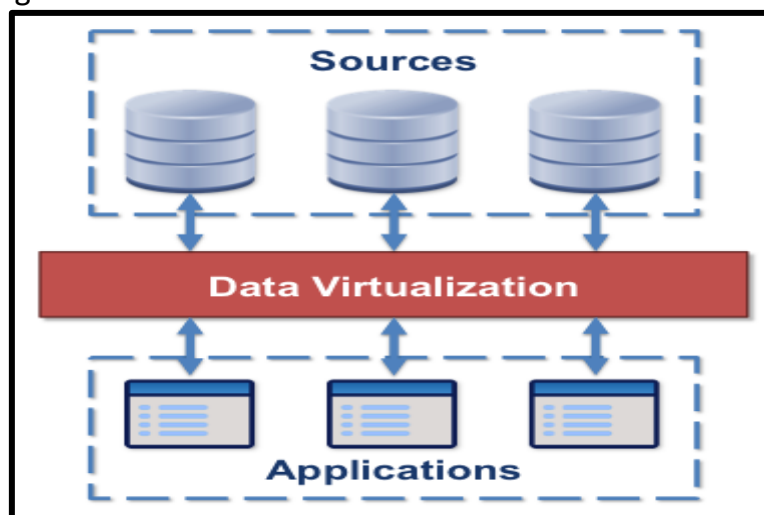


Figure 1 : Définition de Data virtualisation

- **Pourquoi la virtualisation des données ?**

La virtualisation des données est adoptée pour plusieurs raisons, reflétant les défis complexes auxquels sont confrontées les organisations modernes dans la gestion de leurs données. Voici quelques motivations clés pour la virtualisation des données :

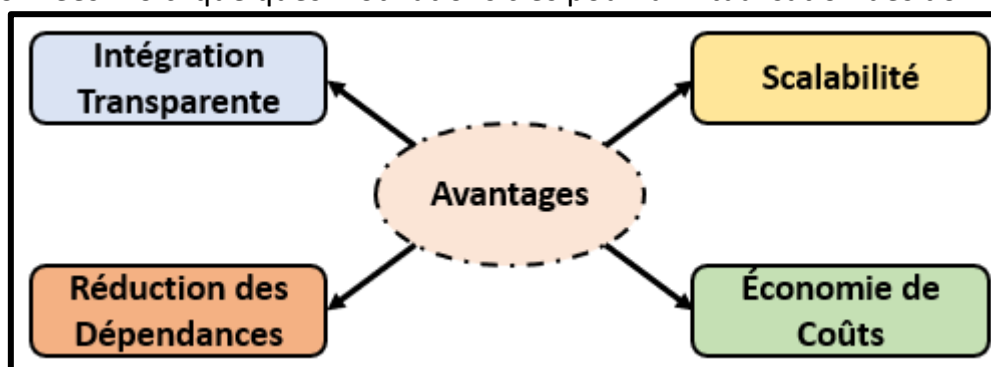


Figure 2 : Avantages de data virtualisation

- **Data virtualisation VS Data Warehousing :**

Tableau 1 : Data Virtualisation VS Data Warehousing

Data Virtualisation	Data Warehousing
La virtualisation des données peut traiter facilement les données présentes dans les entrepôts de données, de manière plus rapide que ces derniers. Cependant, le seul inconvénient de la virtualisation des données est qu'elle n'est pas conçue pour traiter des quantités massives de données. Elle a été créée pour produire des insights rapides et opportuns, soulageant les utilisateurs des méthodes conventionnelles d'accès aux données telles que l'ETL et le stockage des données.	Le terme "Data Warehousing" lui-même donne une idée vague de ce processus. Le data warehousing est un processus visant à collecter, gérer et traiter de grandes quantités de données afin de dévoiler des informations significatives. Le Data Warehouse est au cœur de tout système d'informatique décisionnelle (BI) car la BI est créée à des fins d'analyse des données. Sans les entrepôts de données, nous nous retrouverions avec une masse de données sans aucune vision en temps réel.

### 3. Benchmark des outils de virtualisation : Tableau comparatif

Tableau 2 : Tableau Comparatif des outils de virtualisation

CRITÈRES	DENODO	CDATA	IBM CLOUD PAK FOR DATA	APACHE DRILL
Flexibilité des Connexions	Étendue, prise en charge de diverses sources (relationnelles, fichiers, etc.)	Ciblée, spécialisation dans des connecteurs spécifiques pour différentes sources	Variée, connecteurs pour diverses sources (bases de données, services cloud, fichiers)	Large, support pour diverses sources de données NoSQL (Hadoop )
Performance	Élevée, optimisation des requêtes et cache intelligent	Variable, dépendant du connecteur spécifique utilisé	Dépend de la configuration et de l'infrastructure sous-jacente	Acceptable, peut varier en fonction de la complexité des requêtes
Gestion de la Sécurité	Robuste, gestion centralisée des autorisations et des politiques de sécurité	Variable, dépend du connecteur spécifique et de la source de données	Fonctionnalités de sécurité avancées, gestion des accès et confidentialité	Dépend de la configuration de la sécurité et des politiques spécifiques
Scalabilité	Évolutive, capable de gérer des volumes importants de données et de connexions	Variable, dépend du connecteur utilisé et de la configuration	Hautement scalable, adapté à des charges de travail variées	Évolutif, en fonction de l'architecture et de la configuration
Interface Utilisateur	Intuitive, conviviale et riche en fonctionnalités	Variable, dépend du connecteur utilisé, mais généralement conviviale	Conviviale, offrant des fonctionnalités avancées pour la préparation des données	Interface utilisateur généralement conviviale
Coût	Coût initial et coût total de possession compétitifs	Variable, dépend du nombre et du type de connecteurs nécessaires	Dépend du modèle de tarification et des fonctionnalités requises	Souvent open source, mais dépend des besoins de l'entreprise
Écosystème et Intégration	Intégration aisée avec divers outils et technologies	Intégration variable en fonction du connecteur spécifique	Intégration avec des solutions IBM et d'autres outils d'analyse	Intégration avec diverses sources de données et outils d'analyse

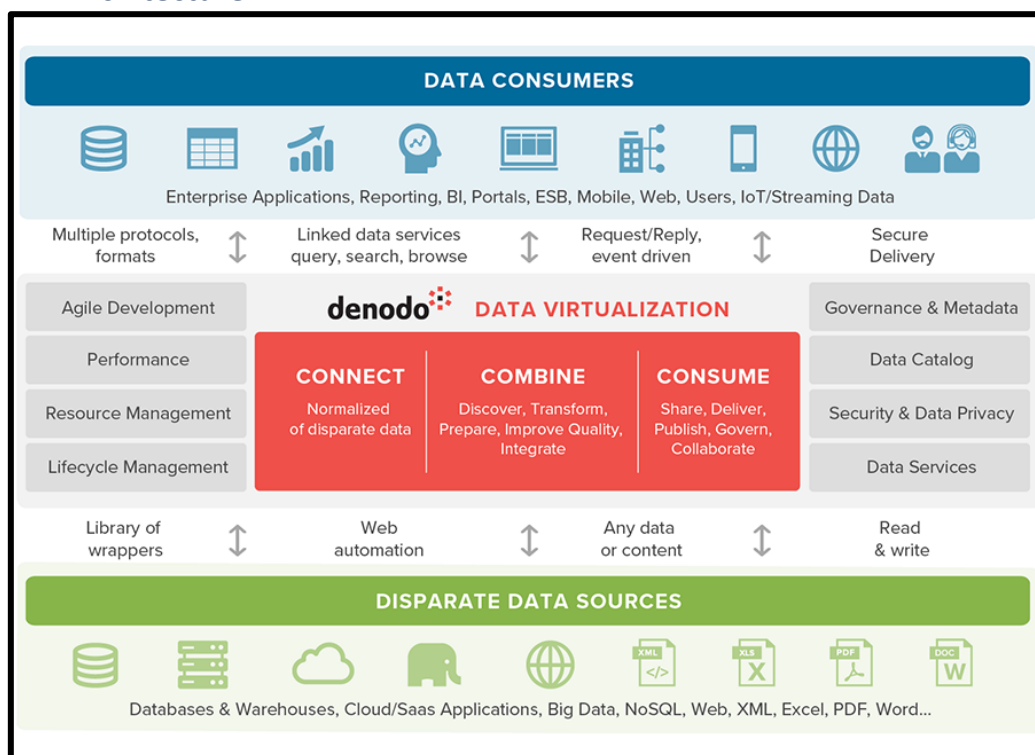
### 4. Denodo : Architecture et Installation

- Pourquoi Denodo :

Denodo a été rigoureusement sélectionné comme la solution convenante pour notre projet, principalement en raison de sa polyvalence et de sa capacité à relever les défis complexes inhérents à la gestion de données diverses. Évitant les complications liées aux plugins de connexions avec différentes bases de données rencontrées dans des solutions open source telles qu'Apache Drill, ainsi que les obstacles associés à IBM et CData, où l'utilisation de JDBC payant a entravé notre progression, Denodo s'est imposé comme le choix évident. Son aptitude à fournir une flexibilité dans la prise en charge de diverses sources de données, associée à une performance élevée et une intégration aisée avec d'autres outils, renforce sa réputation. De plus, la possibilité de travailler avec Denodo communauté sans payer des coûts prohibitifs a été un critère décisif, justifiant ainsi sa renommée dans le domaine de la virtualisation des données pour ce projet spécifique.



- **Architecture :**

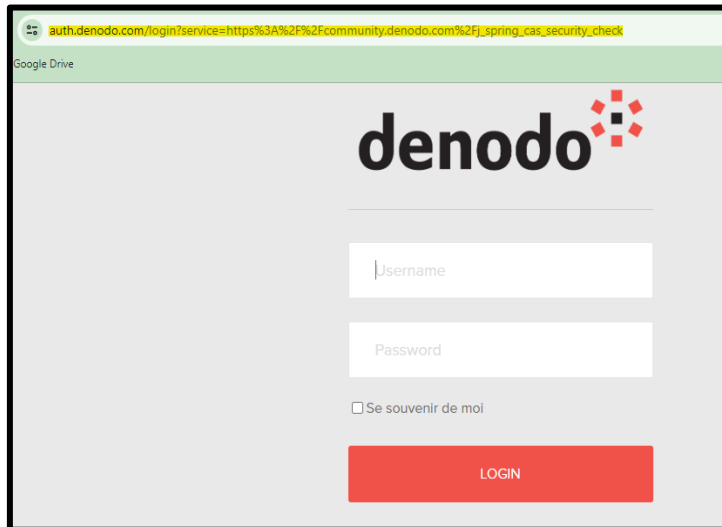


**Figure 3 : Denodo Architecture**

Au cœur de son fonctionnement se trouve le serveur Denodo, qui agit comme une couche d'abstraction entre les sources de données hétérogènes et les utilisateurs finaux. Denodo utilise un modèle de métadonnées pour créer des vues virtuelles, éliminant ainsi la nécessité de déplacer physiquement les données. Les composants clés comprennent le moteur d'exécution des requêtes, qui optimise la récupération des données, et le générateur de vues, qui facilite la création de connexions et de vues virtuelles. Les utilisateurs accèdent aux données à travers l'interface conviviale de Denodo, bénéficiant ainsi d'une vision unifiée et intégrée, sans se soucier de la complexité sous-jacente des sources de données. Cette architecture permet à Denodo de fournir une virtualisation des données efficace, flexible et performante.

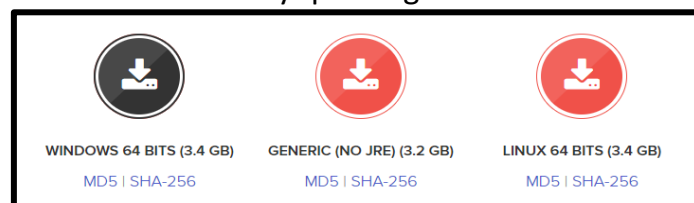
- **Installation :**

Pour l'installation, veuillez aller au site officiel de Denodo, créer un compte



**Figure 4 : Création de compte sous Denodo site**

Ensuite installer la version Community qui est gratuite.



**Figure 5 : Télécharger la version Community**

## **5. Conclusion :**

Ce chapitre a jeté les bases de notre exploration, établissant l'importance du churn dans le contexte des affaires modernes et justifiant l'adoption de la data virtualisation. À travers le tableau comparatif des outils, nous avons identifié Denodo comme l'outil central, dont l'architecture et l'installation feront l'objet de la prochaine étape. Cette section prépare le terrain pour une compréhension approfondie de la manière dont Denodo fonctionnera au sein de notre projet global de virtualisation des données.

## II. Chapitre 2 : La Virtualisation des données : Pratique

### 1. Introduction :

Dans ce deuxième chapitre, notre attention se tourne vers la concrétisation de la virtualisation des données. Après avoir établi la fondation théorique dans le chapitre précédent, nous aborderons désormais la mise en œuvre pratique de la virtualisation. Ce chapitre s'ouvre sur une exploration des sources de données variées, allant des bases de données relationnelles aux systèmes Big Data distribués. Nous détaillerons également l'architecture mise en place, la création de connexions, et la manière dont les vues virtuelles prennent forme pour constituer une couche abstraite intégrative.

### 2. Sources des données :

- **Architecture :**

L'architecture de nos connections vers Denodo se schématisent comme cela :

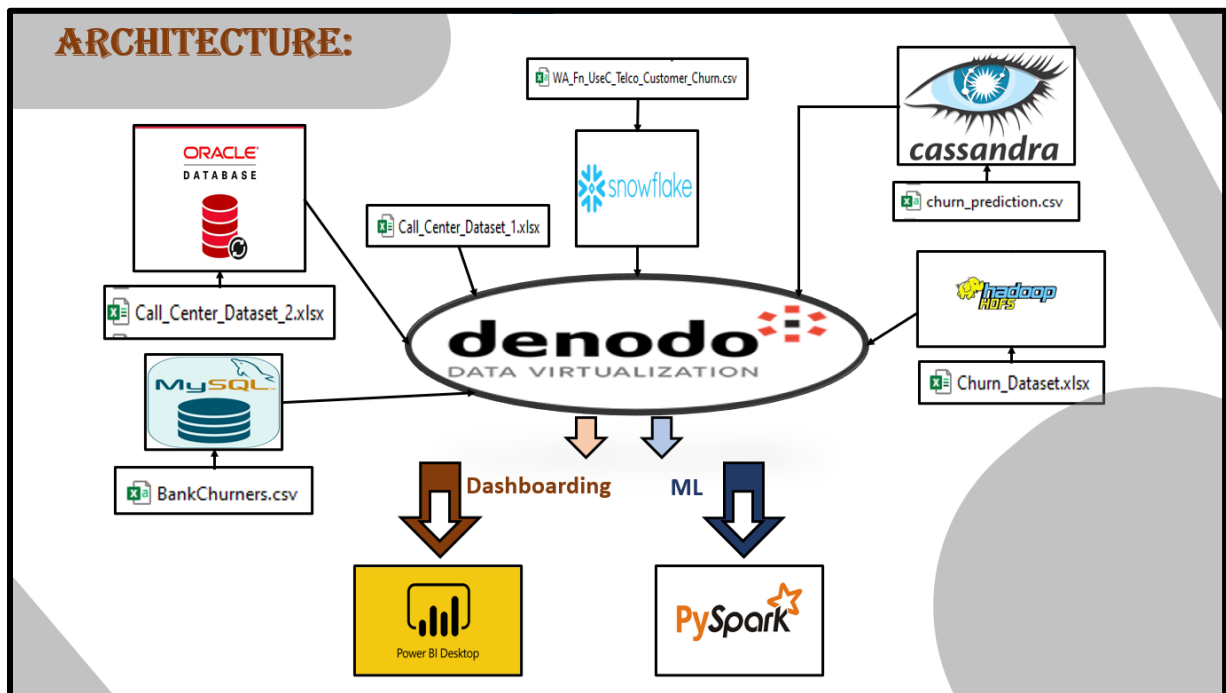


Figure 6: Architecture des sources de données

- **Bases de données :**

On a essayé d'explorer divers types de bases de données qui sont :

- a. **Relationnelle : Oracle.**

Tous d'abord on crée la table oracle :

```
SQL> select table_name from user_tables;
TABLE_NAME
-----
CHURN

SQL> select * from churn;
no rows selected

SQL> describe churn;
Name Null? Type
-----
CALL_ID VARCHAR2(255)
ANSWERED VARCHAR2(255)
RESOLVED VARCHAR2(255)
SPEED_ANSWER NUMBER
AVGTALKDURATION TIMESTAMP(6)
SATISFACTION_RATING NUMBER

SQL> |
```

Figure 7 : Création de la table Oracle

Ensuite on l'a rempli en utilisant **Talend** :

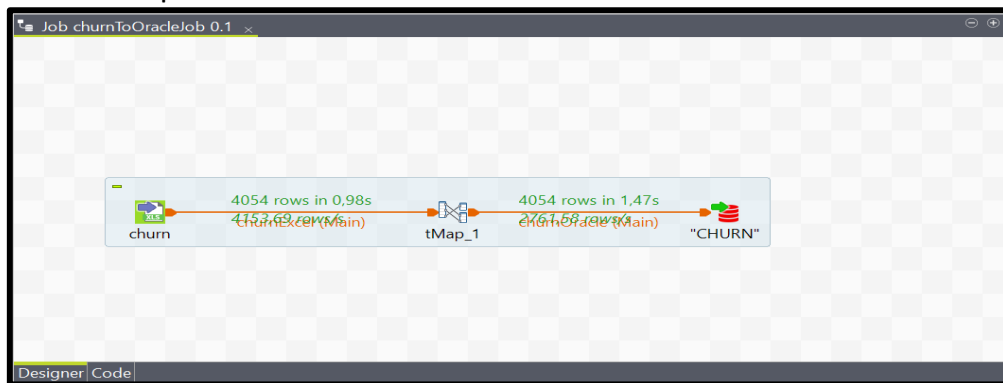


Figure 8 : Remplir la table Oracle avec Talend

- b. **Relationnelle : MySQL**

On charge les données d'après le csv vers MySQL

```
1 LOAD DATA INFILE "D:/Downloads/Data/Clean_Data/BankChurnersCopie.csv"
2 INTO TABLE churn
3 FIELDS TERMINATED BY ','
4 ENCLOSED BY '"'
5 LINES TERMINATED BY '\n'
6 IGNORE 1 ROWS;
```

Figure 9 : Chargement de Table MySQL

Ensuite en jet un coup d'œil sur les données

CLIENTNUM	Attrition_Flag	Customer_Age	Gender	Dependent_count	Education_Level	Marital_Status	Income_Category	Card_Category	Months_on_book	Total_Relationship_Count
708 805 383	Existing Customer	45	M	3	High School	Married	\$60K - \$80K	Blue		39
818 770 008	Existing Customer	49	F	5	Graduate	Single	Less than \$40K	Blue		44
713 982 108	Existing Customer	51	M	3	Graduate	Married	\$80K - \$120K	Blue		36
769 911 858	Existing Customer	40	F	4	High School	Unknown	Less than \$40K	Blue		34
709 106 358	Existing Customer	40	M	3	Uneducated	Married	\$60K - \$80K	Blue		21
713 061 558	Existing Customer	44	M	2	Graduate	Married	\$40K - \$60K	Blue		36
810 347 208	Existing Customer	51	M	4	Unknown	Married	\$120K +	Gold		46
818 906 208	Existing Customer	32	M	0	High School	Unknown	\$60K - \$80K	Silver		27
710 930 508	Existing Customer	37	M	3	Uneducated	Single	\$60K - \$80K	Blue		36
719 661 558	Existing Customer	48	M	2	Graduate	Single	\$80K - \$120K	Blue		36
708 790 833	Existing Customer	42	M	5	Uneducated	Unknown	\$120K +	Blue		31
710 821 833	Existing Customer	65	M	1	Unknown	Married	\$40K - \$60K	Blue		54
710 599 683	Existing Customer	56	M	1	College	Single	\$80K - \$120K	Blue		36
816 082 233	Existing Customer	35	M	3	Graduate	Unknown	\$60K - \$80K	Blue		30
712 396 908	Existing Customer	57	F	2	Graduate	Married	Less than \$40K	Blue		48
714 885 258	Existing Customer	44	M	4	Unknown	Unknown	\$80K - \$120K	Blue		37
709 967 358	Existing Customer	48	M	4	Post-Graduate	Single	\$80K - \$120K	Blue		36
753 327 333	Existing Customer	41	M	3	Unknown	Married	\$80K - \$120K	Blue		34
806 160 108	Existing Customer	61	M	1	High School	Married	\$40K - \$60K	Blue		56
709 327 383	Existing Customer	45	F	2	Graduate	Married	Unknown	Blue		37
806 165 208	Existing Customer	47	M	1	Doctorate	Divorced	\$60K - \$80K	Blue		42
708 508 758	Attrited Customer	62	F	0	Graduate	Married	Less than \$40K	Blue		49

Figure 10 : Vérification de remplissage des données vers MySQL

### c. No SQL : Cassandra

On crée le Keyspace et la table churn dans cassandra

```
CREATE TABLE mykeyspace.churn (
  customer_id int PRIMARY KEY,
  age int,
  average_monthly_balance_prevq float,
  average_monthly_balance_prevq2 float,
  branch_code int,
  churn int,
  city float,
  current_balance float,
  current_month_balance float,
  current_month_credit float,
  current_month_debit float,
  customer_nw_category int,
  dependents float,
  gender text,
  last_transaction_date date,
  occupation text,
  previous_month_balance float,
  previous_month_credit float,
  previous_month_debit float,
  previous_month_end_balance float,
  vintage int
```

Figure 11 : Création de la table Cassandra

Ensuite Copy les données pour remplir la table.

```
cqlsh:mykeyspace> COPY churn (customer_id, age, average_monthly_balance_prevq, average_monthly_balance_prevq2, branch_code, churn, city, current_balance, current_month_balance, current_month_credit, current_month_debit, customer_nw_category, dependents, gender, last_transaction_date, occupation, previous_month_balance, previous_month_credit, previous_month_debit, previous_month_end_balance, vintage) FROM '/home/amine/churn1.csv' WITH HEADER = true;
Using 7 child processes
```

Figure 12 : Remplir la table de Cassandra

Enfin en affiche les données pour vérifier leurs existence

```
customer_id | age | average_monthly_balance_prevq | average_monthly_balance_prevq2 | branch_code | churn | city | current_balance | current_month_balance | current_month_credit | current_month_debit | customer_nw_category | dependents | gender | last_transaction_date | occupation | previous_month_balance | previous_month_credit | previous_month_debit | previous_month_end_balance | vintage
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----
501 | 23 | 45 | 2.18 | 5003.79004 | 1223.59998 | 3588.76001 | 1020 | 2 | 0 | Male | 2019-12-08 | self_employed | 198.34998 | 1548 | 79.79 | 744.72998 | 6859.18018 | 70.73 | 1545.73999 | 2652.54004 | 9214.61035 | 2019-12-31 | self_employed | 1850.66003 | 5000.33008 | 0.33 | 5124.0498 | 2378
```

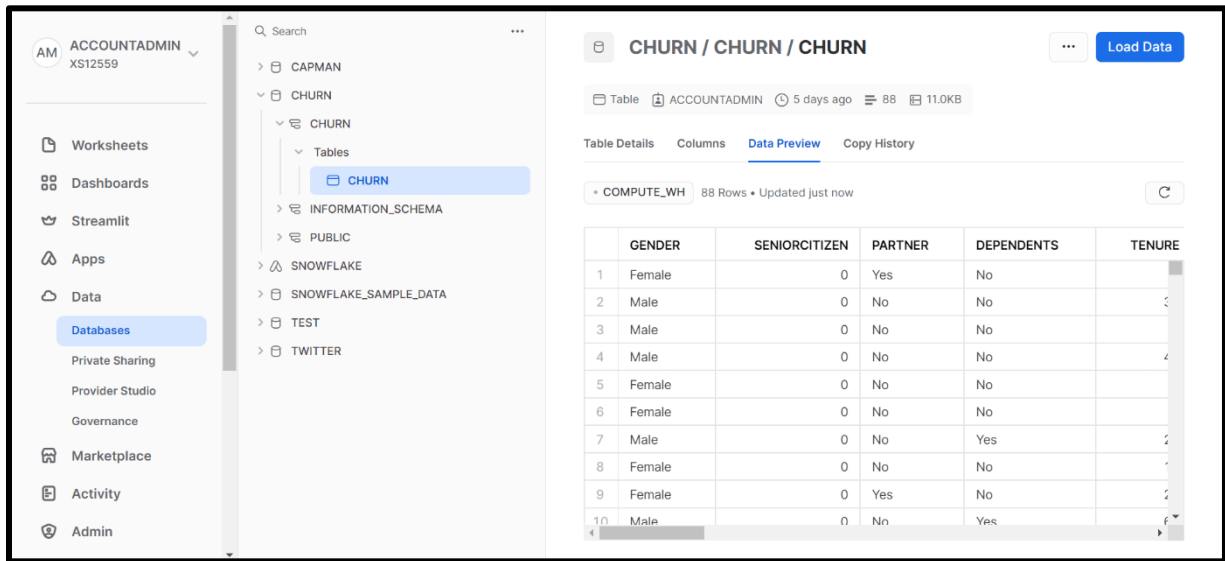
Figure 13 : Table Cassandra Rempli

### d. Fichiers : Excel

	A	B	C	D
1	Call_Id	Date	Agent	Department
2	ID0001	2015-01-01 09:12:58	Diane	Washing Machine
3	ID0002	2015-01-01 09:12:58	Becky	Air Conditioner
4	ID0003	2015-01-01 09:47:31	Stewart	Washing Machine
5	ID0004	2015-01-01 09:47:31	Greg	Washing Machine

Figure 14 : Fichier Excel

## e. Cloud Data Warehouse : Snowflake



	GENDER	SENIORCITIZEN	PARTNER	DEPENDENTS	TENURE
1	Female	0	Yes	No	
2	Male	0	No	No	
3	Male	0	No	No	
4	Male	0	No	No	
5	Female	0	No	No	
6	Female	0	No	No	
7	Male	0	No	Yes	
8	Female	0	No	No	
9	Female	0	Yes	No	
10	Male	0	No	Yes	

Figure 15 : Table Sous Snowflake

Ensuite en affiche les données de Snowflake.

## f. Système Distribué Big Data : Hadoop Hdfs

```
amine@DELL-VOSTRO3420:~$ hadoop fs -copyFromLocal /home/amine/Churn_Dataset.xlsx /amine
amine@DELL-VOSTRO3420:~$ hadoop fs -ls /amine
Found 1 items
-rw-r--r--  1 amine supergroup        652190 2024-01-02 00:38 /amine/Churn_Dataset.xlsx
```

Figure 16 : Table avec Hadoop HDFS

Pour HDFS on remonte les données d'après notre machine vers HDFS.

- **Création des Connexions et des Vues virtuelles :**

- a. **Oracle :**

On commence par la connexion avec la table d'Oracle

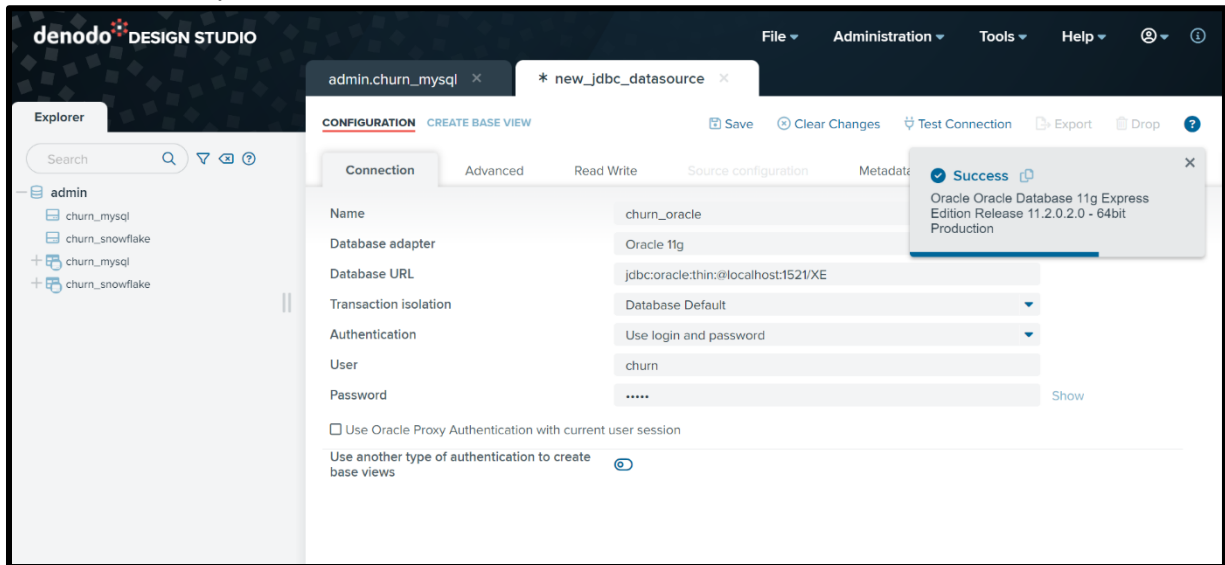


Figure 17 : Connexion réussie avec Oracle

Ensuite on crée la vue Virtuelle et taper la requête d'affichage des données

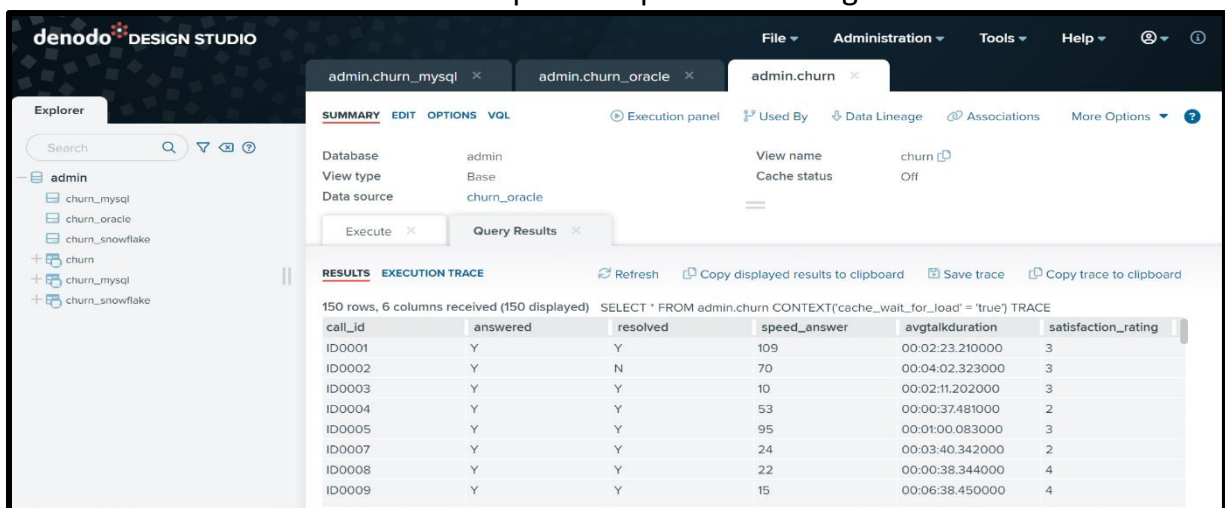


Figure 18 : Requête sur La vue d'Oracle

## b. MySQL :

Pour le cas de MySQL on a besoin d'un JDBC Driver :

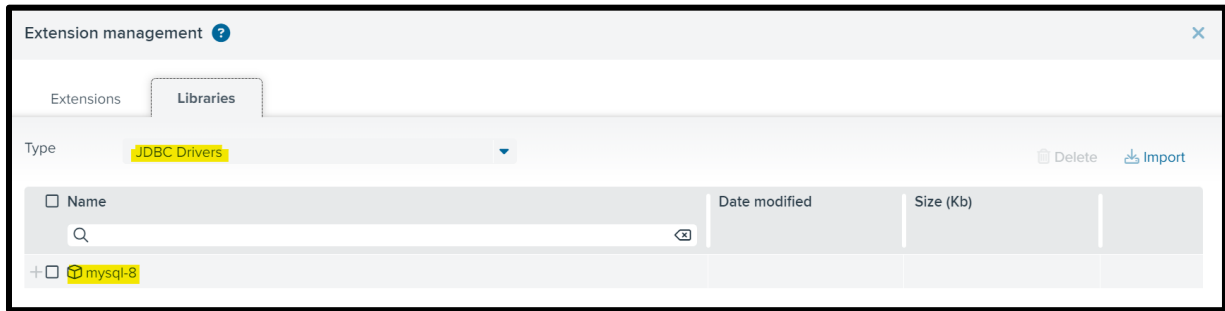


Figure 19 : JDBC Driver pour MySQL

Ensuite on test la connexion :

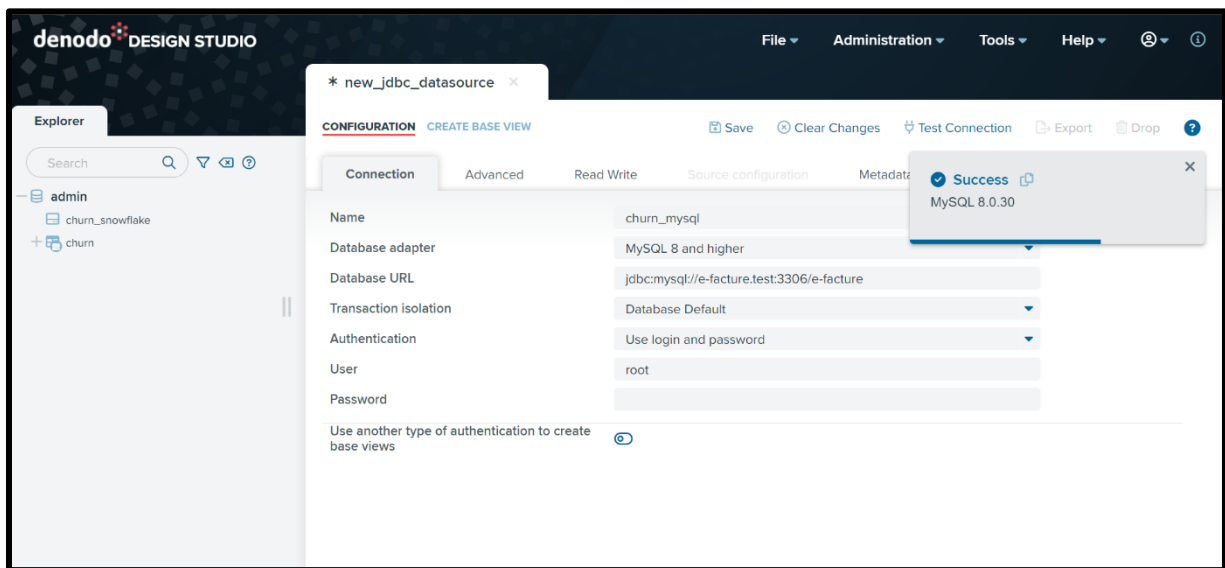


Figure 20 : Connexion réussie avec MySQL

Enfin on crée la vue Virtuelle et taper la requête d'affichage des données

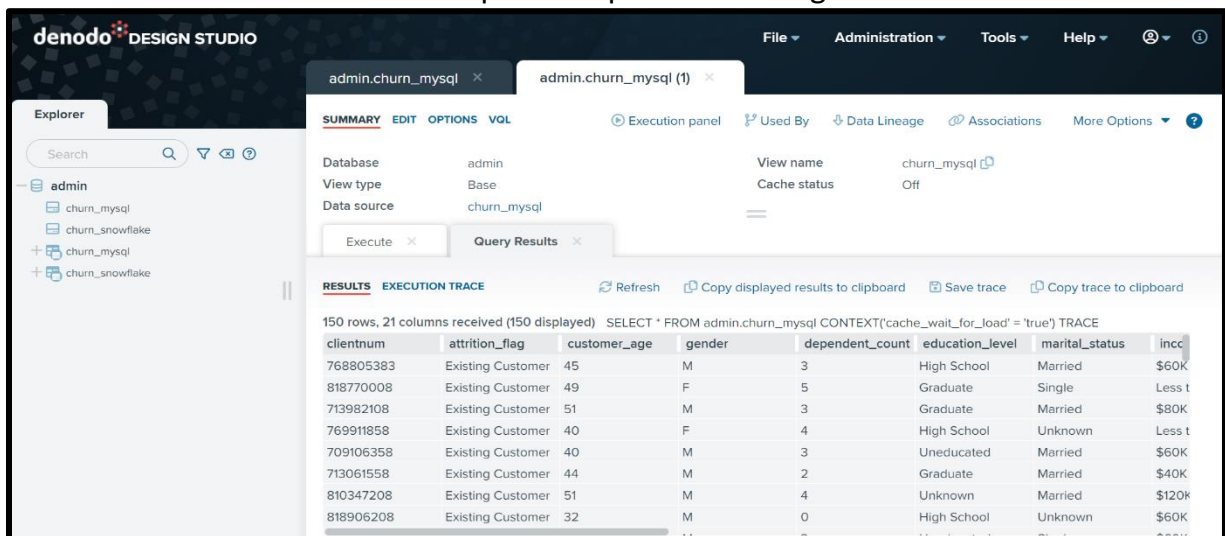


Figure 21 : Requête sur La vue de MySQL



### c. Cassandra :

De la même façon on commence par le test de connexion :

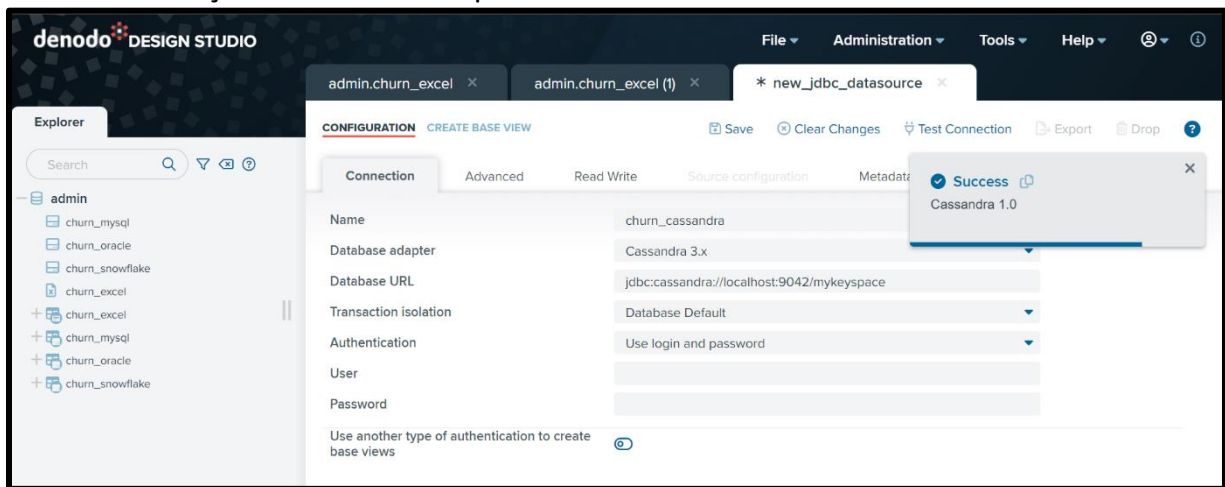


Figure 22 : Connexion réussie avec Cassandra

Ensuite on crée la vue Virtuelle et taper la requête d'affichage des données

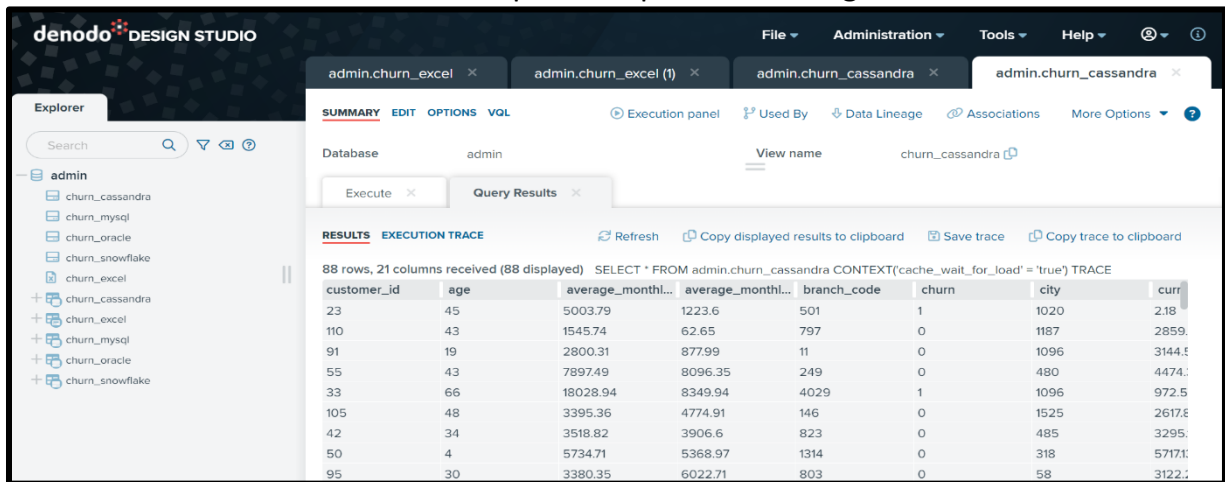


Figure 23 : Requête sur La vue de Cassandra

#### d. Excel :

Test connexion :

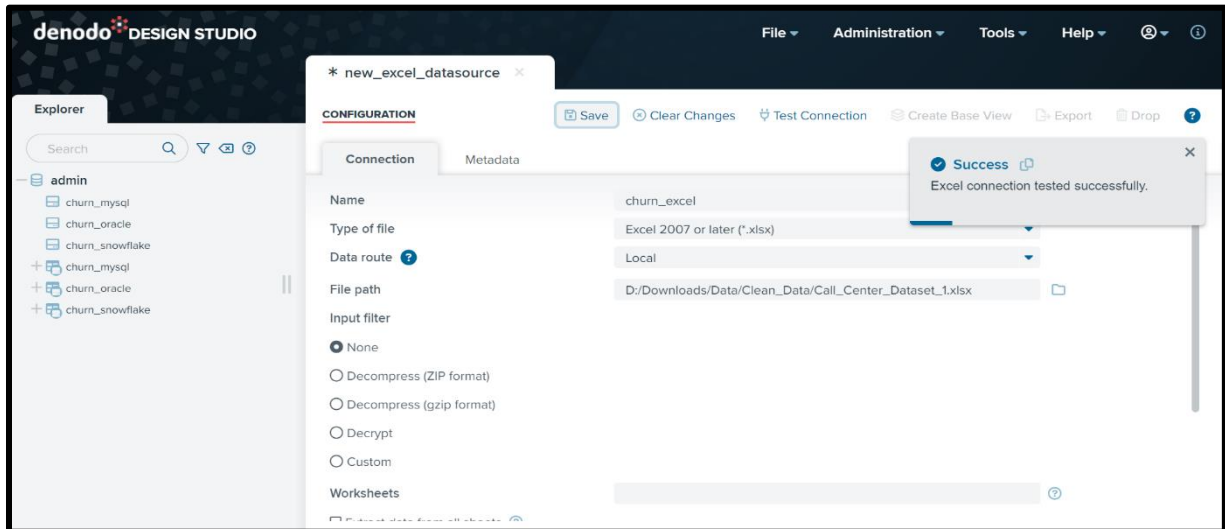


Figure 24 : Connexion réussie avec Excel

Création de la vue Virtuelle et taper la requête d'affichage des données

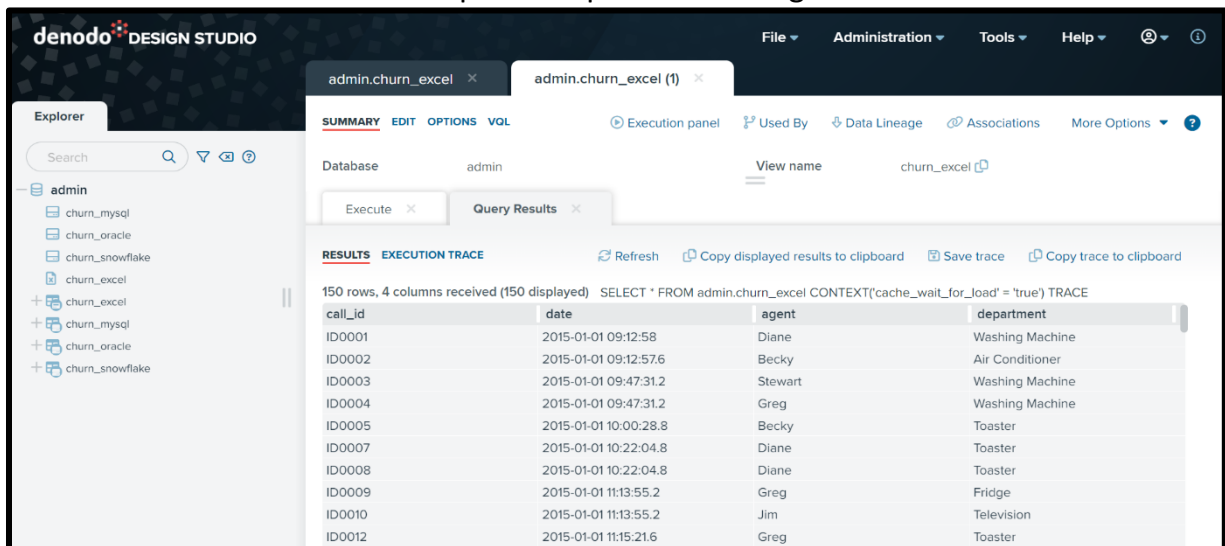


Figure 25 : Requête sur La vue de Excel

### e. Snowflake :

Test de la connexion :

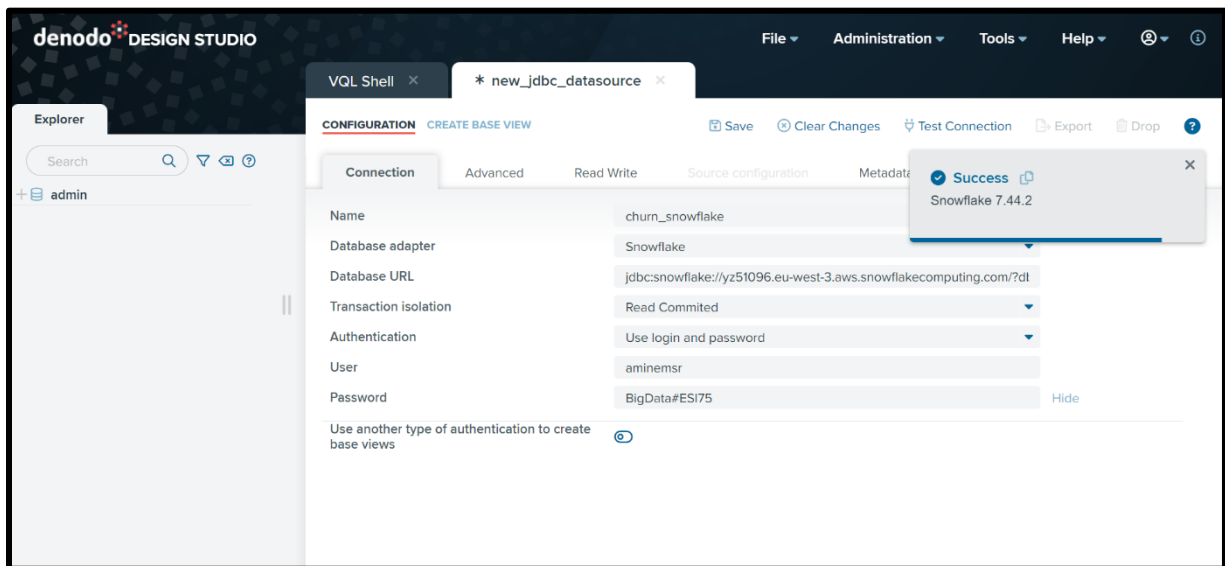


Figure 26 : Connexion réussie avec Snowflake

Ensuite on crée la vue Virtuelle et taper la requête d’affichage des données

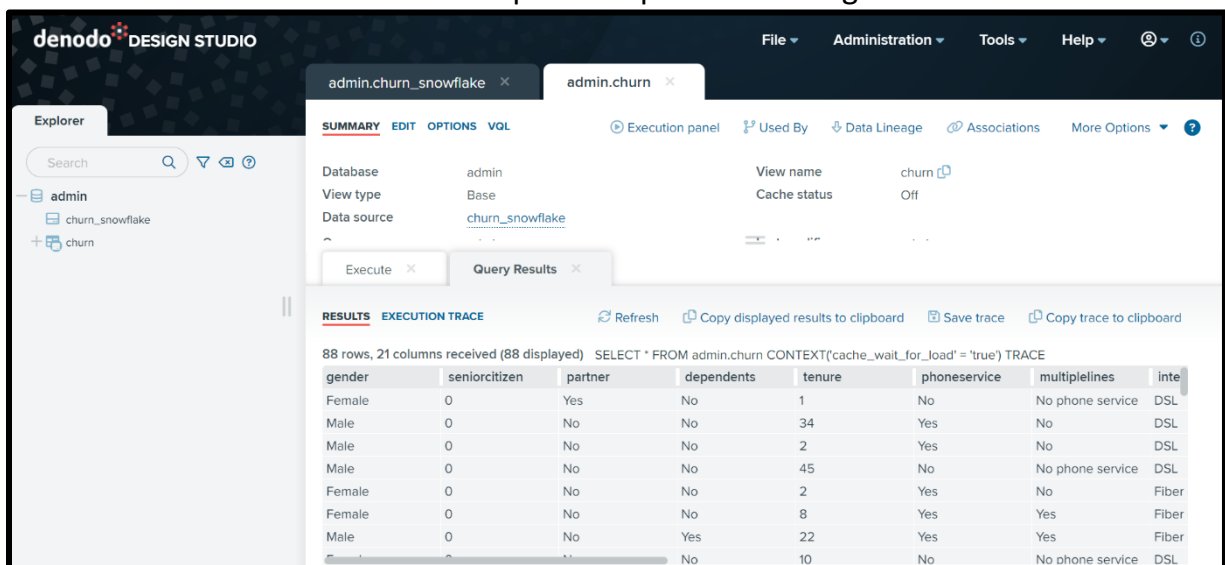


Figure 27 : Requête sur La vue de Snowflake

## f. Hadoop Hdfs :

Tester la connexion avec Hdfs

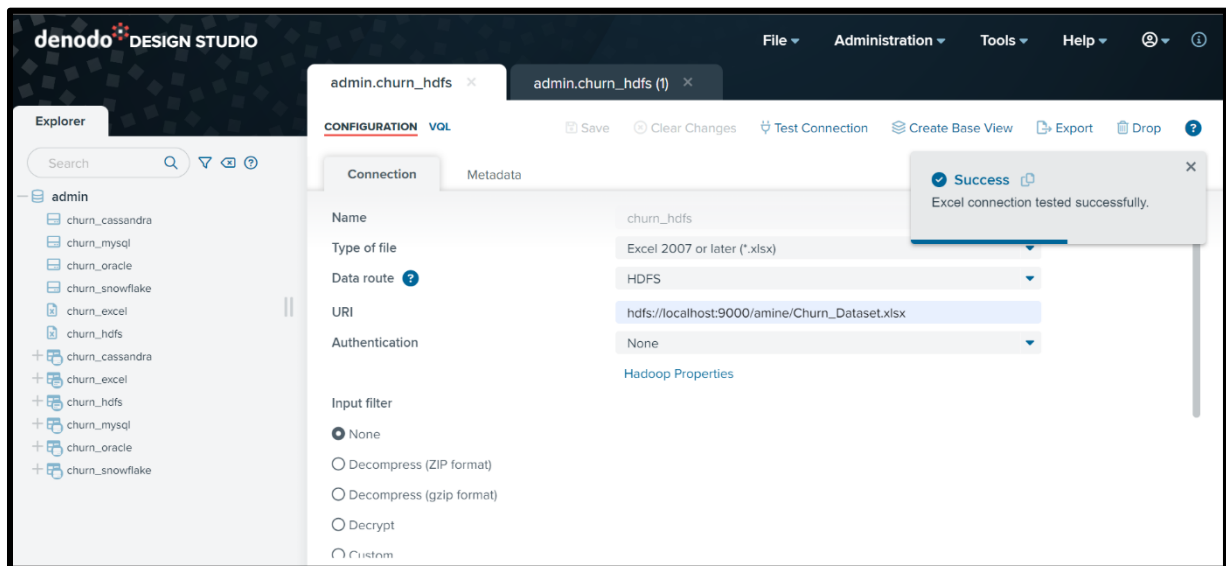


Figure 28 : Connexion réussie avec HDFS

Et créer la vue Virtuelle et taper la requête d'affichage des données

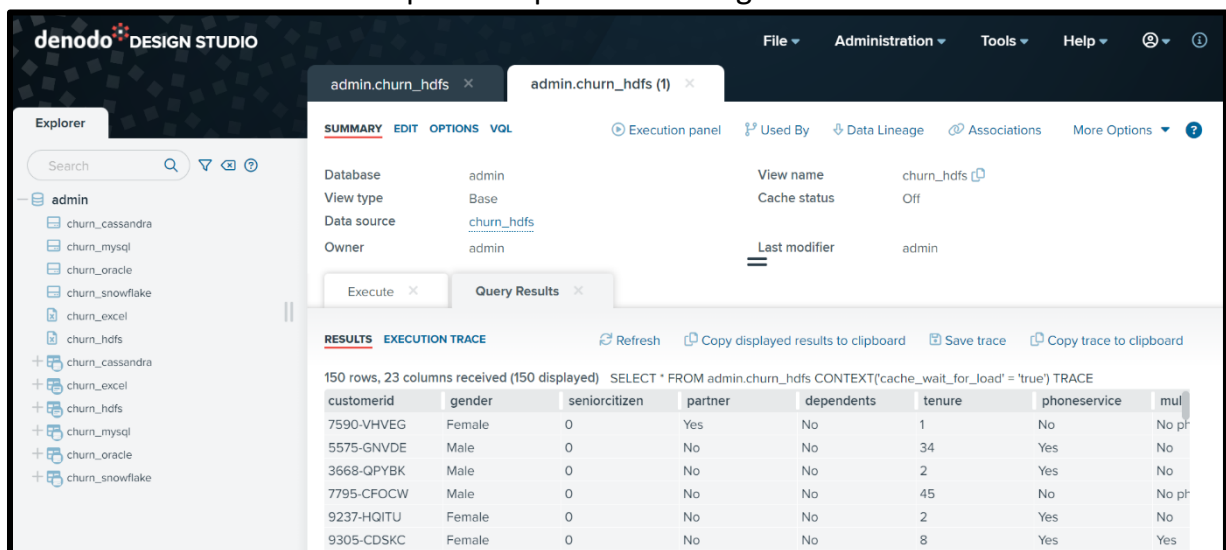


Figure 29 : Requête sur La vue de HDFS

Enfin en crée une vue avec la jointure d'ensemble de vues :

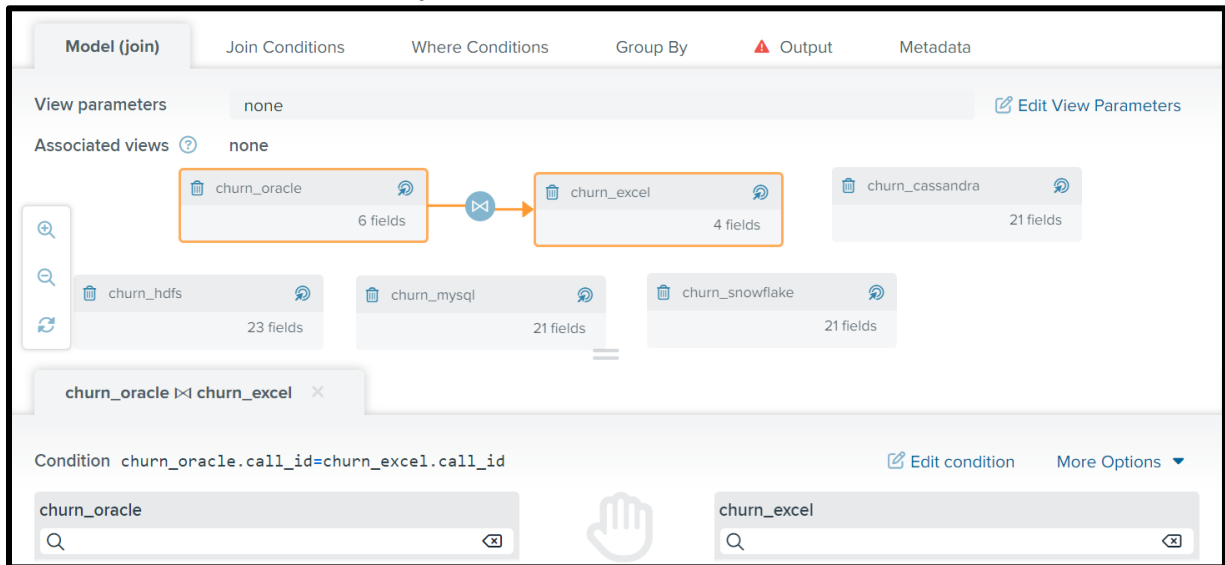


Figure 30 : Jointure des Vues

### 3. Conclusion :

Ce chapitre a concrétisé la virtualisation des données en explorant les sources variées et en détaillant la mise en place d'une couche abstraite avec Denodo. De la gestion des bases de données à la création de vues virtuelles, cette étape a établi une base solide pour la phase pratique du projet. La transition vers le prochain chapitre promet une exploration approfondie des applications du machine Learning et de la visualisation à partir de cette couche virtuelle. Ce passage pratique renforce notre compréhension opérationnelle et prépare le terrain pour des insights avancés.

### III. Machine Learning et Dashbording :

#### 1. Introduction :

Ce Chapitre III marque une évolution significative de notre projet, plongeant dans le monde du machine Learning (ML) et de la création de tableaux de bord. Après avoir établi une solide couche abstraite dans le Chapitre II, nous explorons maintenant la valeur ajoutée de la prédiction du désabonnement des clients via Spark (Py Spark) et la visualisation des résultats à travers Power BI. Ce chapitre présente ainsi une perspective pratique, où les données virtualisées prennent vie à des fins de prise de décision et de communication.

#### 2. ML avec Spark

Dans cette section on va essayer de faire des analyses de churn

- **Connexion avec Python :**

On commence par la création de Spark Session :

```
import findspark
findspark.init()

from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("ChurnAnalysis").getOrCreate()
```

Figure 31 : Création de Spark session

Ensuite on crée la connexion avec Denodo :

```
import pyodbc as dbdriver
from socket import gethostname

denodoserver_name = "localhost"
odbcdriver = "DenodoODBC Unicode(x64)"
denodoserver_odbc_port = "9996"
denodoserver_database = "admin"
denodoserver_uid = "admin"
denodoserver_pwd = "admin"
client_hostname = gethostname()
useragent = "%s-%s" % (dbdriver.__name__, client_hostname)

cnxn = dbdriver.connect(
    driver = odbcdriver,
    server = denodoserver_name,
    port = denodoserver_odbc_port,
    database = denodoserver_database,
    uid = denodoserver_uid,
    pwd = denodoserver_pwd,
    useragent = useragent,
)
```

Figure 32 : Connexion ODBC avec Python

Ensuite on charge le jeu de donnée avec la requête selecte vers une liste

```
## Query to be sent to the Denodo VDP Server
query = "select * from churn_snowflake"

## Define a cursor and execute the results
cur = cnxn.cursor()
cur.execute(query)

results = cur.fetchall()
results

[('Female', Decimal('0'), 'Yes', 'No', Decimal('1'), 'No', 'No phone service', 'DSL', 'No', 'Yes', 'No', 'No', 'No', 'Month-to-month', 'Yes', 'Electronic check', 29.85, 29.85, 'No\r\n', 1.0),
 ('Male', Decimal('0'), 'No', 'No', Decimal('34'), 'Yes', 'No', 'DSL', 'Yes', 'No', 'Yes', 'No', 'No', 'One year', 'No', 'Mailed check', 56.95, 1889.5, 'No\r\n', 0.0),
 ('Male', Decimal('0'), 'No', 'No', Decimal('2'), 'Yes', 'No', 'DSL', 'Yes', 'Yes', 'No', 'No', 'No', 'Month-to-month', 'Yes', 'Mailed check', 53.85, 108.15, 'Yes\r\n', 1.0),
 ('Male', Decimal('0'), 'No', 'No', Decimal('45'), 'No', 'No phone service', 'DSL', 'Yes', 'No', 'Yes', 'Yes', 'No', 'No',
```

Figure 33 : Requête SQL et création de curseur

Enfin en converti la liste vers pandas data frame puis vers Spark data frame :

**Après en convertit la liste vers pandas dataframe ensuite de pandas vers Spark Dataframe**

```
columns = ['Gender', 'SeniorCitizen', 'Partner', 'Dependents', 'Tenure', 'PhoneService', 'MultipleLines', 'InternetService', 'OnMnlyService', 'EmailService', 'TechSupport', 'StreamingService', 'TVService', 'InternetService', 'OnMnlyService', 'EmailService', 'TechSupport', 'StreamingService', 'TVService']
df = pd.DataFrame([tuple(row) for row in results], columns=columns)

df = spark.createDataFrame(df)
```

**2-Nettoyage et traitement**

**Ensuite on explore le jeu de donnée:**

```
df.printSchema()
df.show(10, truncate=False)
```

```
root
 |-- customerID: string (nullable = true)
 |-- gender: string (nullable = true)
```

Figure 34 : Convertir pandas DataFrame en Spark Dataframe

### • Analyse avec Py Spark :

Parmi les analyses qu'on a effectué sur le jeu de données ; on a visualisé la tenure et monthly charges avec la variable cible qui est churn :

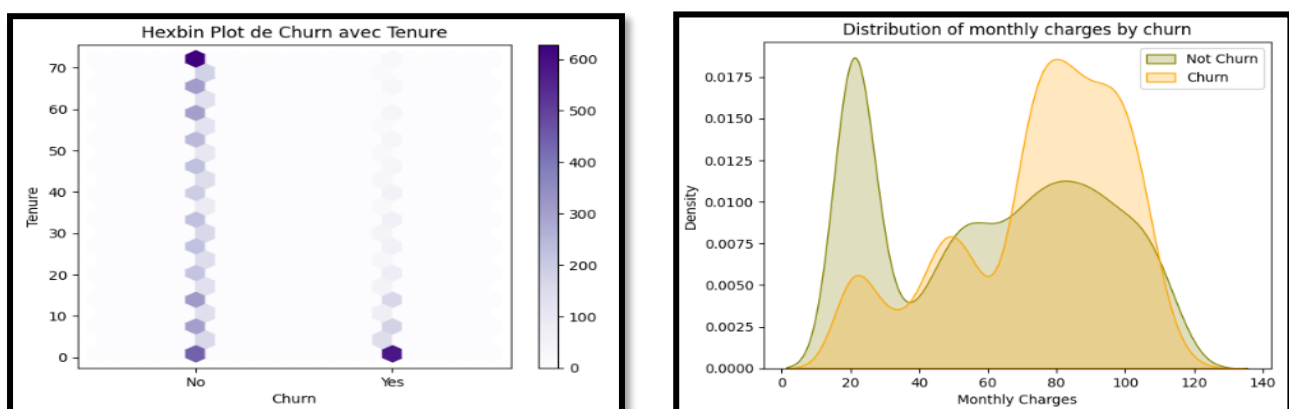


Figure 35 : Analyse et visualisation de churn en fonction d'autres features

On remarque que La densité de pointe pour les clients ayant résilié leur abonnement est légèrement plus basse que pour les clients n'ayant pas résilié leur abonnement, à environ 0-1 ans d'abonnement. Cela suggère que les clients ayant résilié leur abonnement sont plus susceptibles d'avoir une courte durée d'abonnement. Ainsi que Pour la distribution des charges mensuelles, nous observons que les valeurs entre 60 et 120 présentent une densité élevée de désabonnement. D'autre part, les personnes chargées moins ont tendance à rester.

- **Segmentation / Clustering :**

En utilisant K-Means, nous avons réussi à extraire 4 clusters, déduits du diagramme de la méthode ELBOW.

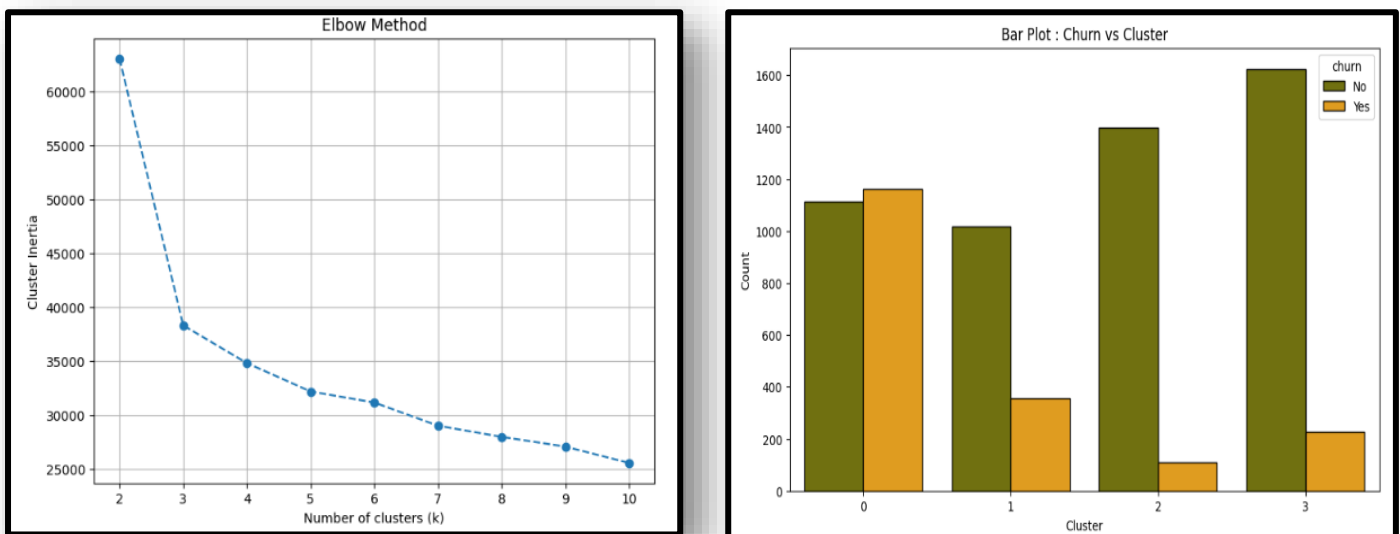


Figure 36 : Clustering et Elbow Method

- **Cluster 2 :** Clients potentiels
- **Cluster 1 :** Clients mécontents
- **Cluster 3 :** Clients fidèles
- **Cluster 0 :** Clients insatisfaits



- **Classification :**

Pour la tache de classification on essayer plusieurs algorithmes de classification qui sont :

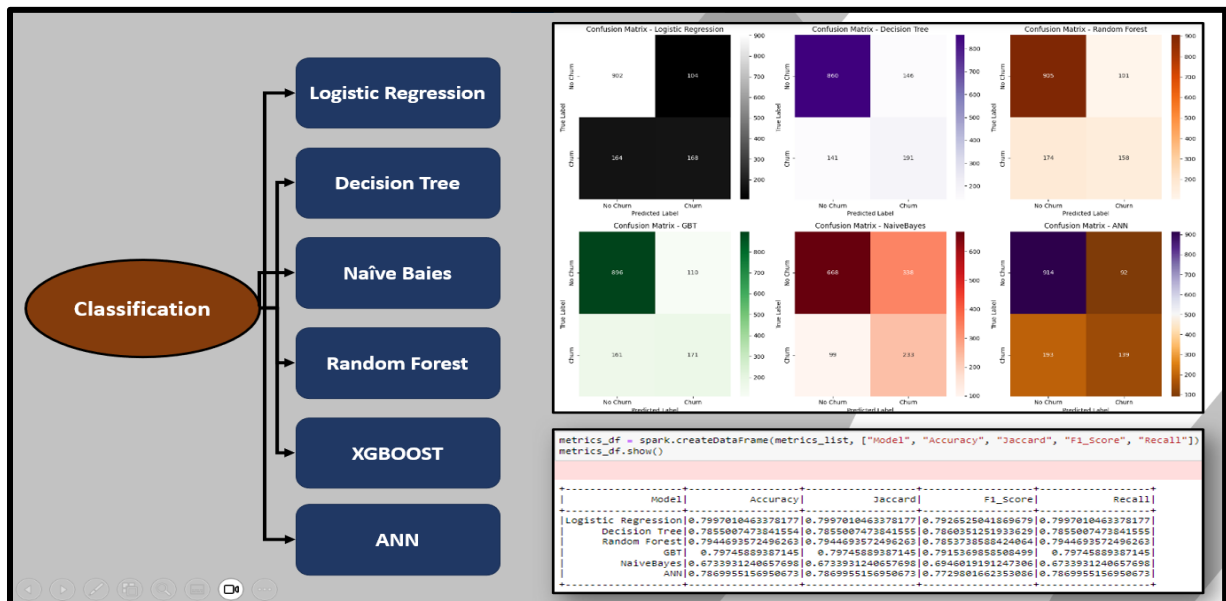


Figure 37 : Algorithmes de classification et Evaluations

Mais le problème c'est que à cause de problème de déséquilibre des données, les modèles sont biaisés vers la classe majoritaire qui est dans ce cas la classe Non Churn. Pour cela on a adopté la méthode SMOTE :

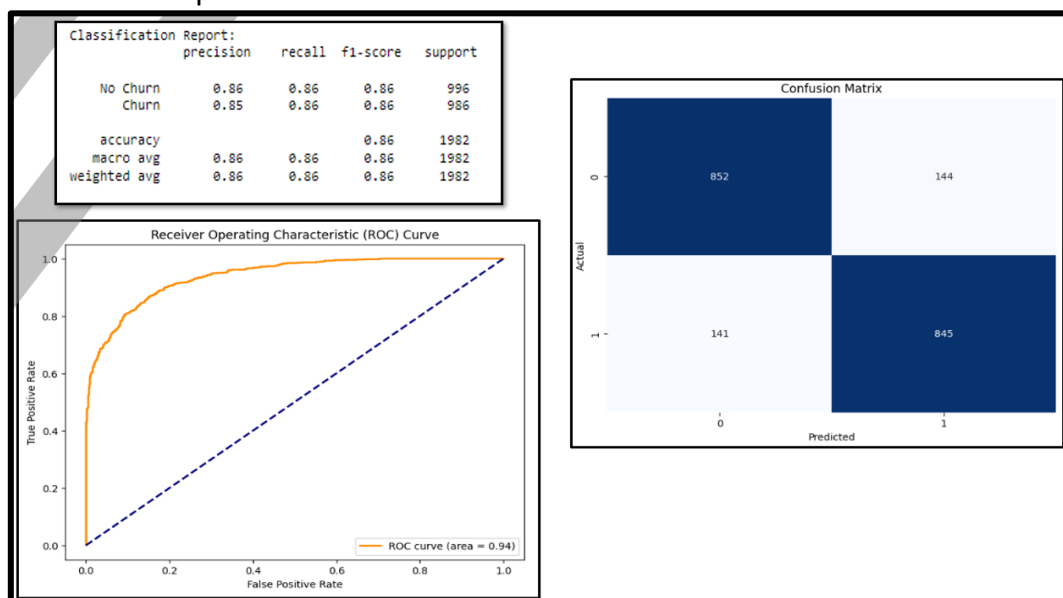


Figure 38 : Random Forest Après SMOTE

On remarque que vraiment l'oversampling avec la méthode SMOTE a donné des résultats assez améliorés pour le modèle Random Forest.

Et voici la Pipeline Spark de modèle Random Forest qui a donné les meilleures performances :

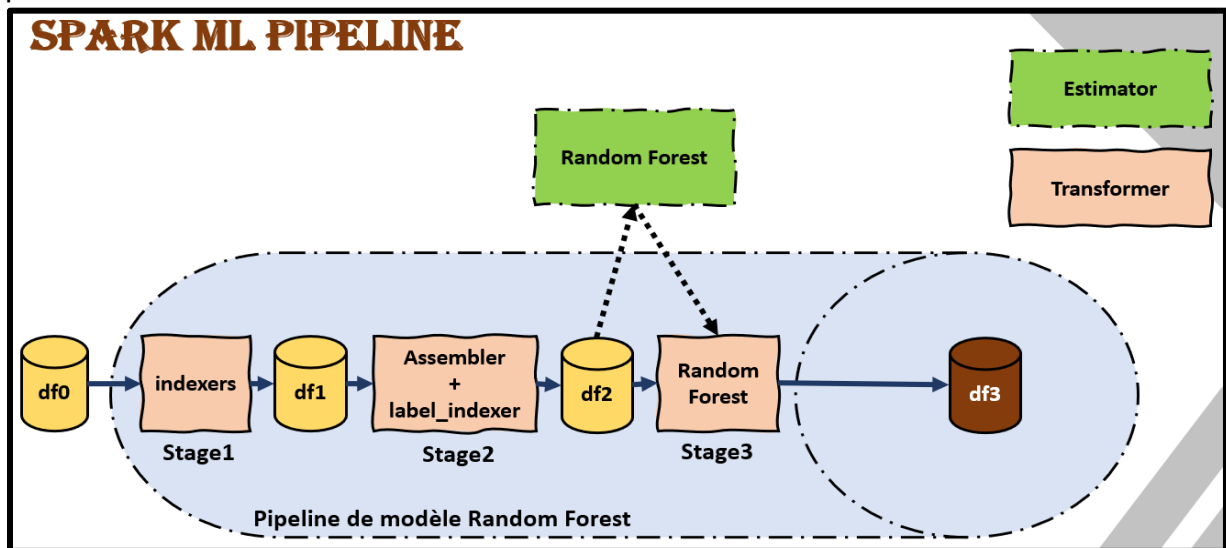


Figure 39 : Spark Pipeline de Modèle Random Forest

### 3. Dashboard avec Power BI :

- **Odbc Installation :**

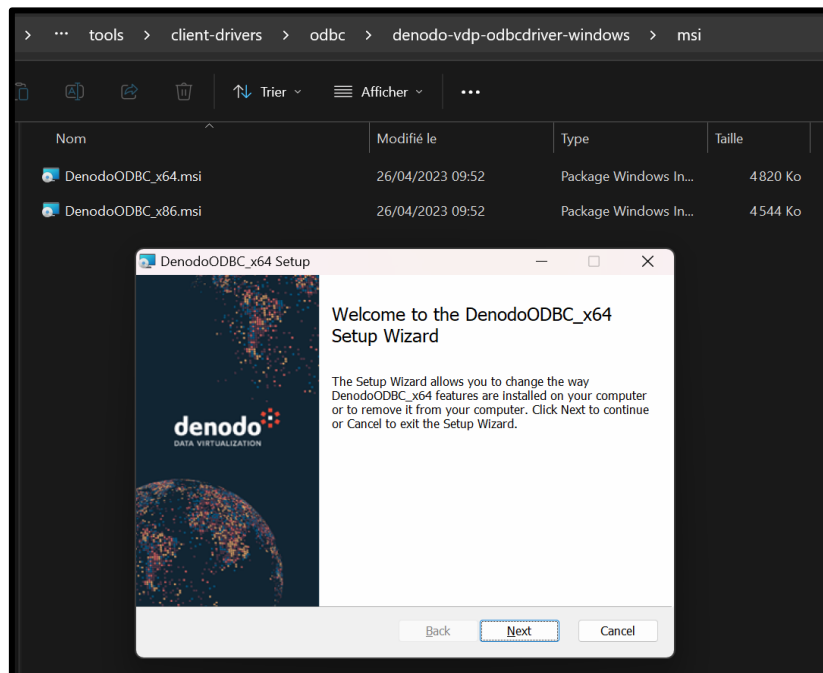
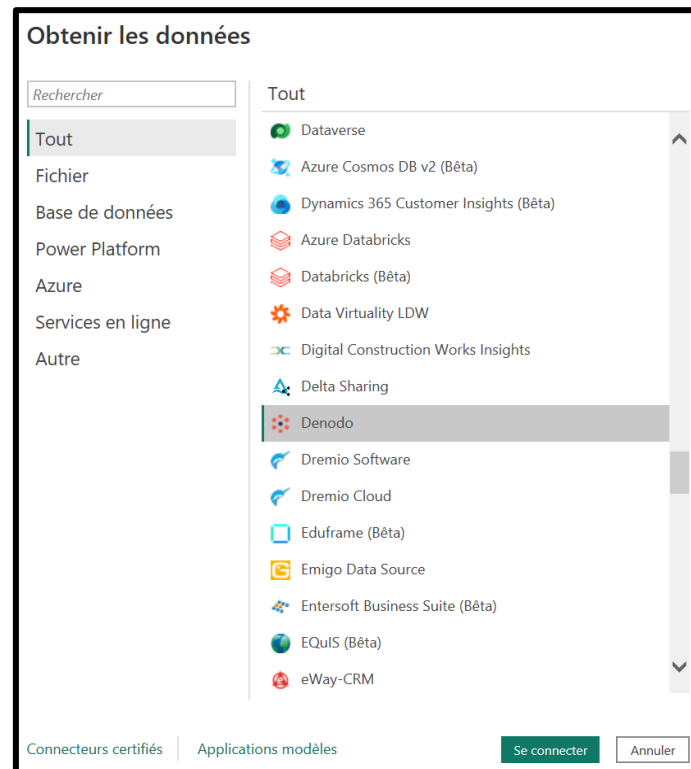


Figure 40 : Power BI Odbc Installation

- **Connection avec Denodo :**



**Figure 41 : Power BI connection1**

The screenshot shows the 'Denodo Connector' configuration window. It includes the following fields and options:

- DSN or Connection String** (with an information icon): A text box containing the string `SERVER=localhost;DATABASE=admin;PORT=9996`.
- Enable debug mode (default: false) (facultatif)** (with an information icon): A dropdown menu currently showing 'false'.
- Mode de connectivité des données** (with an information icon): Two radio button options:
  - ☐ Importer
  - ☒ DirectQuery

**Figure 42 : Power BI connection2**

- **Dashboard :**

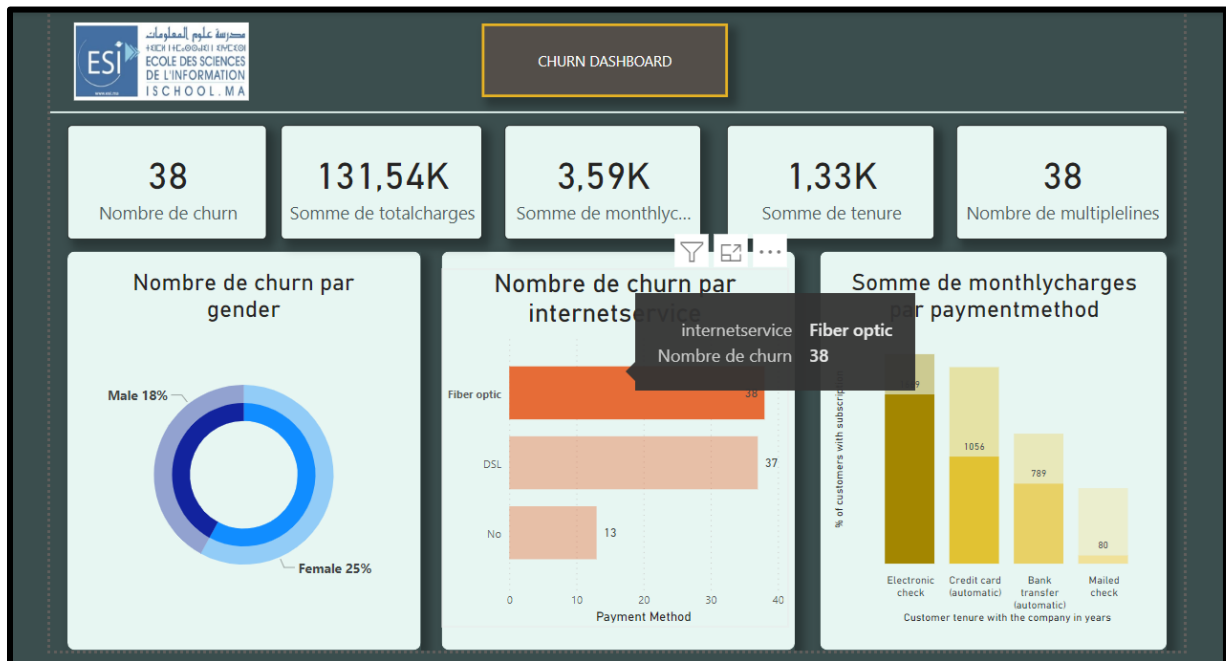


Figure 43 : Power BI Dashboard

#### 4. Conclusion :

Ce chapitre a ouvert la porte à des applications avancées de nos données virtualisées. L'utilisation de Spark pour la machine Learning offre des perspectives de prédiction du désabonnement des clients, tandis que Power BI donne vie à ces insights par le biais de tableaux de bord interactifs. Cette convergence entre le monde du ML et de la visualisation démontre la puissance de la data virtualisation dans des contextes pratiques. La transition vers la conclusion générale du rapport nous permettra de réfléchir à l'ensemble du projet, en mettant en avant les accomplissements et les enseignements tirés de cette exploration multidimensionnelle des données.

## Conclusion :

Ce projet de data virtualisation a été une exploration approfondie des capacités de Denodo en tant qu'outil central dans la gestion et l'analyse de données provenant de sources disparates. En utilisant des données de churn provenant de différentes technologies et environnements, nous avons pu démontrer la flexibilité de Denodo dans la création de vues virtuelles sur une couche abstraite.

La diversité des sources, allant des bases de données relationnelles aux systèmes distribués en passant par les fichiers et les données en streaming, a mis à l'épreuve notre approche. Cependant, l'intégration réussie de ces sources diverses nous a permis d'extraire des insights pertinents sur le désabonnement des clients.

L'utilisation de Spark (Py Spark) pour des tâches de machine Learning a ajouté une dimension prédictive à notre projet, en mettant en évidence la valeur ajoutée de la virtualisation des données pour des applications plus avancées. La visualisation des résultats à l'aide de Power BI a facilité la communication des insights générés, renforçant ainsi l'aspect stratégique de notre démarche.

En conclusion, ce projet a démontré que la data virtualisation, en particulier avec Denodo, offre une solution robuste pour intégrer et exploiter des données variées, offrant ainsi des avantages tangibles dans la prise de décision et la prévision des tendances futures. Les enseignements tirés de cette expérience peuvent être appliqués avec succès dans des contextes similaires, ouvrant ainsi la voie à une gestion plus efficace et stratégique des données au sein des organisations.

# Bibliographie :

[1] Data Virtualisation « Définition » <https://datavalue-consulting.com/data-virtualization-definition/>

(Consulté le 28/12/2023).

[2] Churn Analysis : « 3 Steps to Understanding Why Customers Leave » <https://www.chargebee.com/blog/churn-analysis/>

(Consulté le 28/12/2023).

[3] Denodo « Denodo Download » [https://www.denodo.com/en/denodo-platform/denodo-express?utm\\_source=Denodo-web&utm\\_medium=Try-Denodo](https://www.denodo.com/en/denodo-platform/denodo-express?utm_source=Denodo-web&utm_medium=Try-Denodo)

(consulté le 29/12/2023).

[4] Cassandra « Cassandra Documentation » <https://cassandra.apache.org/doc/latest/>

(Consulté le 29/12/2023).

[5] Snowflake « Snowflake Documentations » <https://www.snowflake.com/en/>

(Consulté le 01/01/2024).

[6] Hadoop « Denodo & Hadoop » <https://community.denodo.com/kb/en/view/document/Denodo%20and%20Hadoop>

(Consulté le 01/01/2024).

[7] Power BI « Denodo & Power BI » <https://community.denodo.com/kb/en/view/document/Denodo%20and%20Hadoop>

(Consulté le 01/01/2024).

[8] Python « Denodo & Python » <https://community.denodo.com/kb/en/view/document/How%20to%20connect%20to%20Denodo%20from%20Python%20-%20a%20starter%20for%20Data%20Scientists>

(Consulté le 01/01/2024).