

Université Sorbonne Paris Nord (USPN)	Année Universitaire
Institut Galilée (IG)	2020 / 2021
Laboratoire d'Informatique de Paris Nord (LIPN)	La Maison des Sciences Numérique (La MSN)
La Data e(s)t le monde de demain	□□□ M. F. Boufarès
faouzi.boufares@sorbonne-paris-nord.fr	boufares@lipn.univ-paris13.fr
La Data/La Donnée	Bases de Données Avancées - Entrepôts de Données



**Projet Annuel : DES BASES AUX ENTREPÔTS DE DONNÉES,
Données structurées ou NON structurées (Oracle, MySQL ou MongoDB ?)
From PDF Files to SQL or NoSQL Data Bases**

Exploration des Curriculum-Vitae (CV) des étudiant.e.s
qui candidatent pour le Maser 2 Informatique Exploration Informatique des Données et Décisionnel.

Réalisé le 17 décembre 2020 (Partie 1) par l'équipe :

<i>Groupe</i>	<i>Binôme</i>	<i>NOM</i>	<i>Prénom</i>
G7	B09	LE	Minh Hao
		OULD HADDA	Tiziri
	B20	CHAOUCHE	Yacine
		TERAA	Youcef Oussama
	B12	ADEL	Salah-Eddine
		BOUHOUT	Adil
<i>Encadrant (Enseignant-Chercheur) : M. F. BOUFARES</i>			

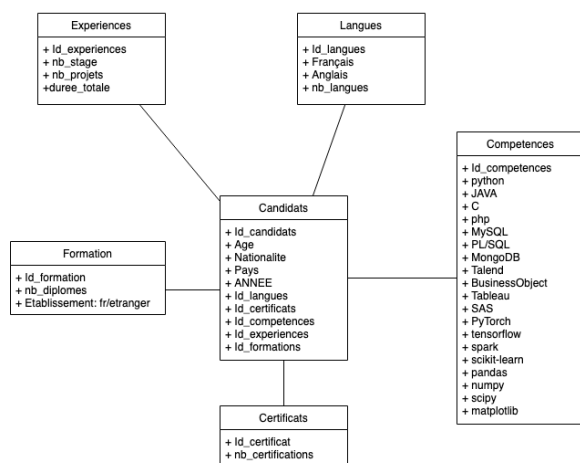
ORACLE
DATABASE

MySQL

mongoDB

Table des matières

1. Introduction	3
2. Schémas de Base de Données : Structures de données & Méta-Données	4
2.1. Le schéma conceptuel de données (EER)	5
2.2. Le schéma conceptuel de données (UML)	6
2.3. Le schéma logique de données (Schéma relationnel)	7
2.4. Le dictionnaire de données & les métadonnées	9
2.5. Expression des contraintes sur les données	12
3. Exemple de contenu de la base de données (instance de BD)	13
3.1. Exemple d'instances de la BD des CV	14
3.2. Création des structures de données (SQL-ORACLE)	15
3.3. Insertion des données dans une table et qualité des données (SQL/PLSQL-ORACLE)	19
3.4. Interrogations & Manipulations (SQL/PLSQL-ORACLE)	20
4. Sujet du Projet Annuel	21
4.1. : Conception & Implantation de Bases de Données	21
4.2. : Conception & Implantation d'Entrepôts de Données	21
Schéma de la Data Warehouse :	25



L'intégration de données :	26
Insertion des données MySQL vers le schéma de la base de données Data Warehouse :	26
Table de fait candidats :	33
Job d'insertion dans la table dimension compétences :	34
Job d'insertion dans la table dimension langues :	35
Sous Jobs d'insertion dans les tables dimensions formation, certifications et expériences :	37
5. Dataviz avec Tableau Software	38
5. Conclusion	42

1. Introduction

Le système d'information de l'Université Sorbonne Paris Nord (USPN) est articulé autour de la base de données de nom BD-USPN. Elle contient les données relatives aux : (i) Ressources Humaines (Tels que les Enseignant.e.s, les Chercheur.se.s , les Ingénieur.e.s, les Administratifs) (ii) aux Etablissements (Tels que les Universités, les Ecoles, les Instituts, Hôpitaux, Entreprises, Organismes, Editeurs) partenaires, (iii) Eudiant.e.s (Tels que les anciens, les actuels, les candidats nationaux et internationaux retenus et non retenus). Une partie de cette base de données fera l'objet de notre étude.

En effet, dans le cadre de notre projet annuel intitulé « **DES BASES AUX ENTREPÔTS DE DONNÉES, données structurées ou NON structurées (Oracle, MySQL ou MongoDB ?) ; From PDF Files to SQL or NoSQL Data Bases** ».

Il s'agit de la construction de données structurées à partir de données NON structurées (à partir de documents papiers ou numériques qui correspondent aux Curriculum-Vitae (CV) des étudiant.e.s qui candidatent pour le Maser 2 Informatique Exploration Informatique des Données et Décisionnel. Cette BD devra contenir les candidatures sur plusieurs années afin d'étudier le profil des candidat.e.s en entrée et évidemment ceux en sortie c'est-à-dire les diplômé.e.s.

Elle est décrite par le schéma conceptuel ainsi que le schéma relationnel (les tables) ci-après.

Ce document se présente comme suit. Le premier paragraphe.... Dans la deuxième section.... En conclusion.

2. Schémas de Base de Données : Structures de données & Méta-Données

Le schéma conceptuel de données (SCD) ou le modèle conceptuel de données (MCD) peut être décrit selon plusieurs formalismes.

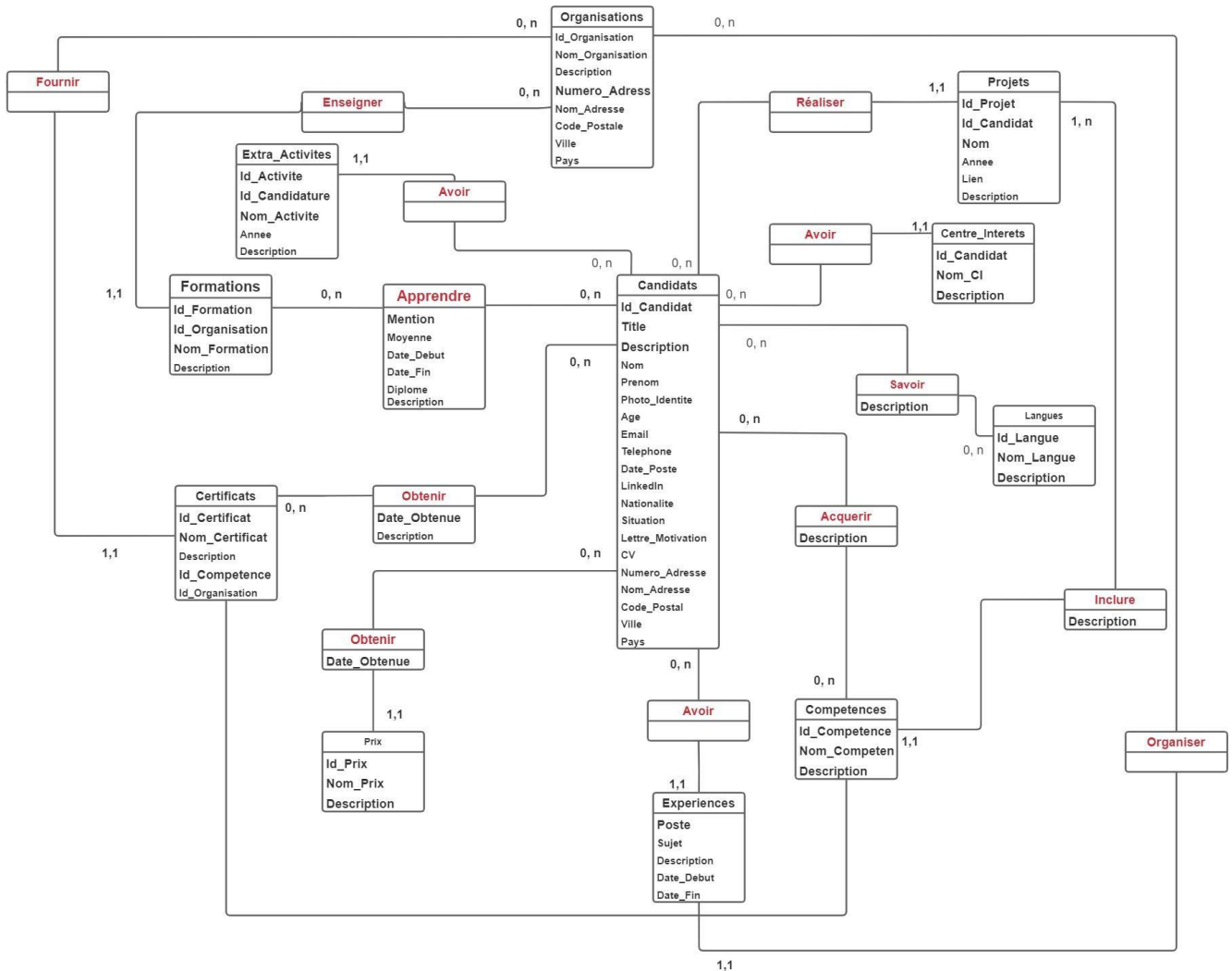
Dans le formalisme « Entity-Relationship (ER) ou Extended Entity-Relationship (EER), appelé aussi EA », la description des données comporte plusieurs entités (ou type d'entité = les rectangles) reliées entre elle par des associations (ou type d'association = les ovales), les arcs sont étiquetés par des contraintes de cardinalité (Exemple 1,1 ; 1,n et 0,n). La figure 1 représente le schéma conceptuel de la Base de Données, selon le formalisme Entité/Association (EA).

Dans le formalisme « Unified Modeling Language : UML », La description des données comporte plusieurs classes reliées entre elles par des arcs étiquetés par des contraintes de multiplicité (Exemple 1 ; 1,* et *) :

La structure (le schéma) de la BD est donnée ci-dessous.

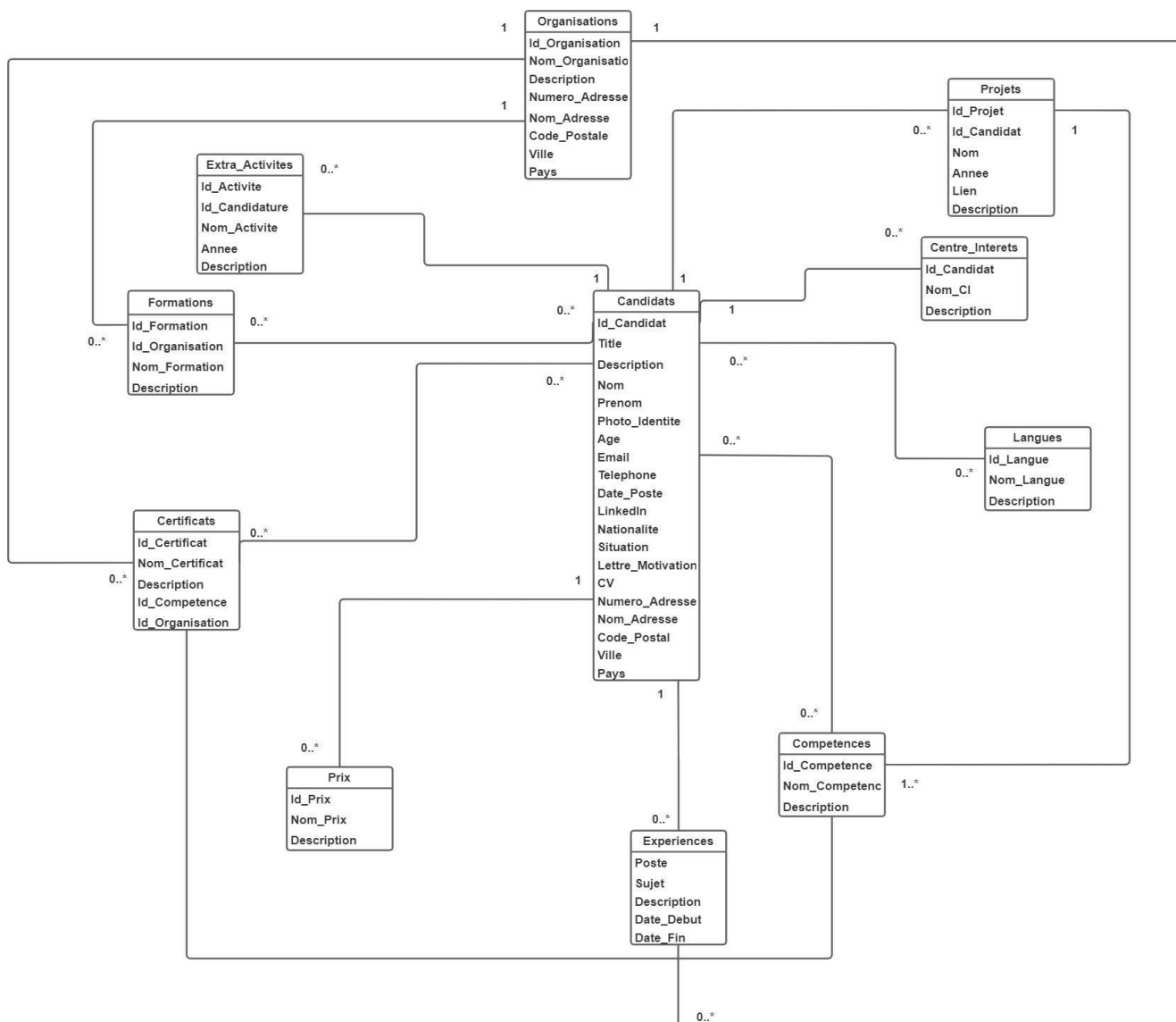
2.1. Le schéma conceptuel de données (EER)

Le schéma conceptuel de données (SCD) ou le modèle conceptuel de données (MCD) est décrit ci-dessous selon formalisme « Entity-Relationship (ER, EA) ou Extended Entity-Relationship (EER) ».



2.2. Le schéma conceptuel de données (UML)

Le **schéma conceptuel de données** (SCD) ou le **modèle conceptuel de données** (MCD) est décrit ci-dessous selon formalisme « Unified Modeling Language : UML ».



2.3. Le schéma logique de données (Schéma relationnel)

Le Schéma relationnel de la Base de Données (Modèle Logique de Données MLD) est composé de plusieurs tables. Elles sont présentées ci-dessous :

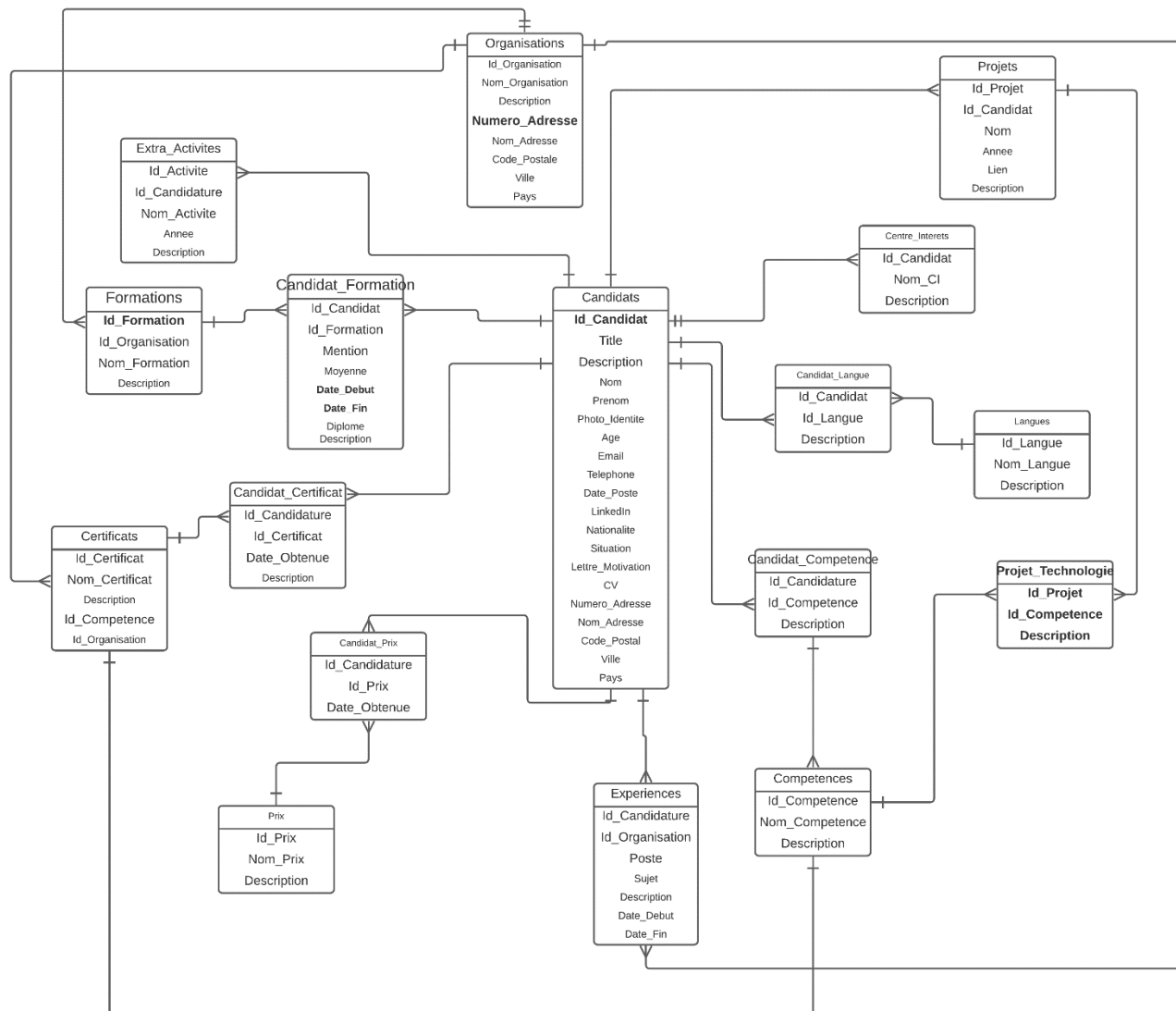


Table des candidats :

CANDIDATS(Ń ID_CANDIDAT, TITLE, DESCRIPTION, NOM, PRENOM, PHOTO_IDENTITE, AGE, EMAIL, TELEPHONE, DATE_POSTE, LINKEDIN, NATIONALITE, SITUATION, LETTRE_MOTIVATION, CV)

Table des formations :

FORMATIONS(Ń ID_ORGANISATION, ID_CANDIDAT, ORGANISATION, DATE_DEBUT, DATE_FIN, NOM_FORMATION, MENTION, MOYENNE, DIPLOME, PAYS, DESCRIPTION)

Table des expériences :

EXPÉRIENCES(Ń ID_EXPERIENCE, ID_CANDIDAT, POSTE, SUJET, DESCRIPTION, LOCATION, ORGANISATION, DATE_DÉBUT, DATE_FIN)

Table des détails des projets :

PROJETS(Ń ID_PROJET, ID_CANDIDAT, NOM, ANNÉE, LIEN, COMPÉTENCES, OUTILS, DESCRIPTION)

Table des détails des compétences :

COMPÉTENCES(\tilde{N} ID_COMPETENCE, ID_CANDIDAT , NOM_COMPETENCE, DESCRIPTION)

Table des détails des certificats :

CERTIFICATS(\tilde{N} ID_CERTIFICAT, ID_CANDIDAT , NOM_CERTIFICAT, ANNÉE, ORGANISATION, DESCRIPTION)

Table des détails des prix :

PRIX(\tilde{N} ID_PRIX , NOM_PRIX, ANNÉE, ORGANISATION, DESCRIPTION)

Table des détails des langues :

LANGUES(ID_CANDIDAT , \tilde{N} NOM_LANGUE, NIVEAU, DESCRIPTION)

Table des détails des centre_interets :

CENTRE_INTERETS(\tilde{N} NOM_CI, ID_CANDIDAT , DESCRIPTION)

Table des détails des extra_activités:

EXTRA_ACTIVITÉS(\tilde{N} ID_ACTIVITE, ID_CANDIDAT , NOM_ACTIVITÉS, ANNÉE, DESCRIPTION)

Table des détails des adresses :

ADRESSES(\tilde{N} ID_ADRESSE, ID_CANDIDAT , NUMERO_ADRESSE, NOM_ADRESSE, CP_ADRESSE, VILLE_ADRESSE, PAYS_ADRESSE)

Un dictionnaire de données contient les détails de toutes les descriptions des différentes colonnes de toutes les tables de la BD.

Ces descriptifs constituent une partie des métadonnées de la BD.

Le dictionnaire de données de la BD est détaillé ci-dessous :

2.4. Le dictionnaire de données & les métadonnées

Le **dictionnaire de données** de la BD est le suivant (ces descriptifs constituent un sous ensemble des **métadonnées** de la BD) :

Table	Attribut - Colonne	Signification - Description	Type de données	Contrainte
CANDIDATS	ID CANDIDAT	Numéro Identité du Candidat	Caractères/Texte, 20	Unique, Non nul
CANDIDATS	TITLE	Titre de CV du candidat	Caractères/Texte, 300	
CANDIDATS	DESCRIPTION	Le message du candidat pour les recruteurs.	Caractères/Texte, 300	
CANDIDATS	NOM	Nom du candidat	Caractères/Texte, 20	Non nul, en MAJUSCULE
CANDIDATS	PRENOM	Prénom du candidat	Caractères/Texte, 100	Non nul, la première lettre en Majuscule
CANDIDATS	PHOTO_IDENTITE	Chemin de la photo dans le système	Caractères/Texte, 250	en format /documents/photos/PH_yyyy_NOM_Prenom.jpg
CANDIDATS	AGE	Date de naissance du candidat	Entier	
CANDIDATS	EMAIL	Email du candidat	Caractères/Texte, 100	vérifie une expression régulière
CANDIDATS	TELEPHONE	Numéro de téléphone du candidat	Caractères/Texte, 20	vérifie une expression régulière
CANDIDATS	DATE_POSTE	La date où le candidat a posté le CV	Date	en format dd/mm/yyyy
CANDIDATS	LINKEDIN	Lien LinkedIn du candidat	Caractères/Texte, 100	
CANDIDATS	NATIONALITE	Nationalité du candidat	Caractères/Texte, 50	en MAJUSCULE
CANDIDATS	SITUATION	État actuel du candidat sur promotion	Caractères/Texte, 20	dans la liste ('admis(e)', 'refusé(e)', 'en attente')
CANDIDATS	LETTRÉ_MOTIVATION	Chemin de la Lettre de motivation dans le system	Caractères/Texte, 250	en format /documents/lms/LM_yyyy_NOM_Prenom.pdf
CANDIDATS	CV	Chemin du CV dans le system	Caractères/Texte, 250	Non nul, en format /documents/cvs/CV_yyyy_NOM_Prenom.pdf

Table	Attribut - Colonne	Signification - Description	Domaine - Type de données	Contrainte
FORMATIONS	ID_ORGANISATION	Identifiant de l'organisation scolaire dans système	Caractères/Texte, 20	Unique, Non nul
FORMATIONS	ORGANISATION	Nom de l'organisation scolaire	Caractères/Texte, 100	Non nul
FORMATIONS	DATE_DEBUT	Date où le candidat a commencé à étudier la formation	Date	en format mm/yyyy ou yyyy
FORMATIONS	DATE_FIN	Date où le candidat a fini la formation	Date	en format mm/yyyy ou yyyy
FORMATIONS	NOM_FORMATION	Nom de formation que le candidat a étudié	Caractères/Texte, 100	
FORMATIONS	MENTION	Mention du candidat à la formation	Caractères/Texte, 20	
FORMATIONS	MOYENNE	Moyenne du candidat à la formation	Réel, 4	dans l'intervalle [0,20]
FORMATIONS	DIPLOME	Niveau obtenu après la formation	Caractères/Texte, 10	
FORMATIONS	PAYS	Pays de l'organisation	Caractères/Texte, 30	en MAJUSCULE
FORMATIONS	DESCRIPTION	Description de l'organisation et de la formation	Caractères/Texte, 300	

Table	Attribut - Colonne	Signification - Description	Domaine - Type de données	Contrainte
-------	--------------------	-----------------------------	---------------------------	------------

EXPERIENCES	ID_EXPERIENCE	Identifiant de l'expérience du candidat	Caractères/Texte, 20	Unique, Non nul
EXPERIENCES	POSTE	Poste du candidat au niveau de l'expérience	Caractères/Texte, 50	
EXPERIENCES	SUJET	Sujet de l'expérience	Caractères/Texte, 300	
EXPERIENCES	DESCRIPTION	Description de l'expérience	Caractères/Texte, 1000	
EXPERIENCES	LOCATION	Location où le candidat a fait l'expérience	Caractères/Texte, 300	
EXPERIENCES	ORGANISATION	Nom d'organisation au niveau de l'expérience	Caractères/Texte, 100	Non nul
EXPERIENCES	DATE_DEBUT	Date où candidat a commencé	Date	en format mm/yyyy
EXPERIENCES	DATE_FIN	Date où candidat a terminé	Date	en format mm/yyyy

Table	Attribut - Colonne	Signification - Description	Domaine - Type de données	Contrainte
PROJETS	ID_PROJET	Identifiant de l'expérience du candidat	Caractères/Texte, 20	Unique, Non nul
PROJETS	NOM	Nom de projet du candidat	Caractères/Texte, 250	Non nul
PROJETS	ANNEE	Année ou projet était réalisé	Caractères/Texte, 4	yyyy
PROJETS	LIEN	Lien de projet	Caractères/Texte, 250	
PROJETS	COMPÉTENCES	Compétence obtenue dans projet, séparées par des virgules	Caractères/Texte, 300	
PROJETS	OUTILS	Outils utilisés dans projet, séparés par des virgules	Caractères/Texte, 300	
PROJETS	DESCRIPTION	Description de projet	Caractères/Texte, 1000	

Table	Attribut - Colonne	Signification - Description	Domaine - Type de données	Contrainte
COMPETENCES	ID_COMPETENCES	Identifiant de la compétence du candidat	Caractères/Texte, 20	Unique, Non nul
COMPETENCES	NOM_COMPETENCE	Nom de compétence du candidat dans le CV	Caractères/Texte, 100	Non nul
COMPETENCES	DESCRIPTION	Description de compétence au niveau candidat	Caractères/Texte, 300	

Table	Attribut - Colonne	Signification - Description	Domaine - Type de données	Contrainte
LANGUES	NOM_LANGUE	Langue du candidat dans le CV	Caractères/Texte, 20	Non nul
LANGUES	NIVEAU	Niveau de langue du candidat	Caractères/Texte, 50	
LANGUES	DESCRIPTION	Description de niveau de langue	Caractères/Texte, 100	

Table	Attribut - Colonne	Signification - Description	Domaine - Type de données	Contrainte
CENTRE_INTERETS	NOM_CI	Nom de centre d'intérêt du candidat	Caractères/Texte, 50	Non nul
CENTRE_INTERETS	DESCRIPTION	Description de centre d'intérêt du candidat	Caractères/Texte, 300	

Table	Attribut - Colonne	Signification - Description	Domaine - Type de données	Contrainte
CERTIFICATS	ID_CERTIFICAT	Identifiant de certificat du candidat	Caractères/Texte, 20	Unique
CERTIFICATS	NOM	Nom de certificat du candidat dans le CV	Caractères/Texte, 100	Non nul
CERTIFICATS	ANNEE	Année où le candidat a participé l'activité	Caractères/Texte, 4	en format yyyy
CERTIFICATS	ORGANISATION	Nom de l'organisation qui a fourni le certificat	Caractères/Texte, 100	
CERTIFICATS	DESCRIPTION	Description de certificat du candidat	Caractères/Texte, 300	

Table	Attribut - Colonne	Signification - Description	Domaine - Type de données	Contrainte
PRIX	ID_PRIX	Identifiant de prix du candidat	Caractères/Texte, 20	Unique
PRIX	NOM	Nom de prix du candidat dans le CV	Caractères/Texte, 100	Non nul
PRIX	ANNÉE	Année de prix du candidat dans le CV	Caractères/Texte, 4	en format yyyy
PRIX	ORGANISATION	Organisation fournit le prix	Caractères/Texte, 100	
PRIX	DESCRIPTION	Description de prix du candidat dans le CV	Caractères/Texte, 300	

Table	Attribut - Colonne	Signification - Description	Domaine - Type de données	Contrainte
EXTRA_ACTIVITES	ID_ACTIVITE	Identifiant de l'activité du candidat	Caractères/Texte, 20	
EXTRA_ACTIVITES	NOM	Nom de l'activité	Caractères/Texte, 100	Non nul
EXTRA_ACTIVITES	ANNEE	Année où le candidat a participé l'activité	Caractères/Texte, 4	en format yyyy
EXTRA_ACTIVITES	DESCRIPTION	Description de l'activité	Caractères/Texte, 300	

Table	Attribut - Colonne	Signification - Description	Domaine - Type de données	Contrainte
ADRESSES	ID_ADRESSE	Identifiant de l'adresse du candidat	Caractères/Texte, 20	Unique, Non nul
ADRESSES	NUMERO_ADRESSE	Numéro de l'adresse	Caractères/Texte, 10	
ADRESSES	NOM_ADRESSE	Nom de l'adresse	Caractères/Texte, 30	Non nul
ADRESSES	CP_ADRESSE	Code postal	Caractères/Texte, 20	
ADRESSES	VILLE_ADRESSE	Ville de candidat	Caractères/Texte, 20	Non nul
ADRESSES	PAYS_ADRESSE	Pays de candidat	Caractères/Texte, 20	Non nul

2.5. Expression des contraintes sur les données

L'expression des contraintes sur les données est une étape primordiale pour valider les données. La liste des contraintes à valider est donnée ci-dessous.

- Le nom du candidat doit être en **majuscule**, les valeurs sont du type **alphabétique**, il peut contenir un tiret (-) s'il est composé.
 - Le première lettre du prénom du candidat doit être en **majuscule**, les valeurs sont du type, il peut contenir un tiret (-) s'il est composé.
 - l'âge du candidat doit être supérieur à **20**.
 - La nationalité du candidat doit être en **majuscule**.
 - Des **expressions régulières** pour définir le mail, le téléphone du candidat.
 - La situation du candidat prend une valeur dans la liste suivante {**admis(e), refusé(e), en attente**}.
 - Le fichier de la photo d'identité doit être nommé de cette façon (PH_YYYY_NOM_PRENOM).
 - Le fichier de la lettre de motivation doit être nommé de cette façon (LM_YYYY_NOM_PRENOM).
 - Le fichier de CV doit être nommé de cette façon (CV_YYYY_NOM_PRENOM).
 - Date début de la formation doit être de la forme **mm/yyyy(mois/année)**.
 - Date fin de la formation doit être de la forme **mm/yyyy(mois/année)**.
 - La mention prend des valeurs du type **alphabétique**.
 - La moyenne prend des valeurs **numériques** comprises en 0 et 20.
 - Le pays de la formation doit être en **majuscule**.
 - La location de l'expérience doit être en **majuscule**.
 - Date début de l'expérience doit être de la forme **mm/yyyy(mois/année)**.
 - Date fin de l'expérience doit être de la forme **mm/yyyy(mois/année)**.
 - L'année du projet doit être de la forme **yyyy(année)**.
 - L'année du certificat doit être de la forme **yyyy(année)**.
 - L'année du prix doit être de la forme **yyyy(année)**.
 - L'année de l'extra activité doit être de la forme **yyyy(année)**.
 - Le numéro dans la voie de l'adresse prend des valeurs **numériques** plus une valeur dans la liste suivante (**BIS, TER, QUATER**) ou des valeurs **numériques** seulement.
 - Le voie de l'adresse doit commencer par valeur dans la liste suivante {RUE, BOULEVARD, AVENUE, QUAI, IMPASSE, PONT, PLACE, SQUARE, ALLÉE, VOIE, MONTÉE, ESPLANADE, ROUTE, VOIRIE, CITÉ, CHEMIN, PARVIS}.
- *** Un candidat doit suivre au moins une formation.
- *** Un candidat doit réaliser au moins un projet.

*** Un candidat doit avoir au moins compétence.

*** Un candidat doit avoir au moins une adresse.

3. Exemple de contenu de la base de données (instance de BD)

Une instance de la base de données des CV est donnée ci-dessous. Les scripts SQL correspondants sont donnés dans les fichiers :

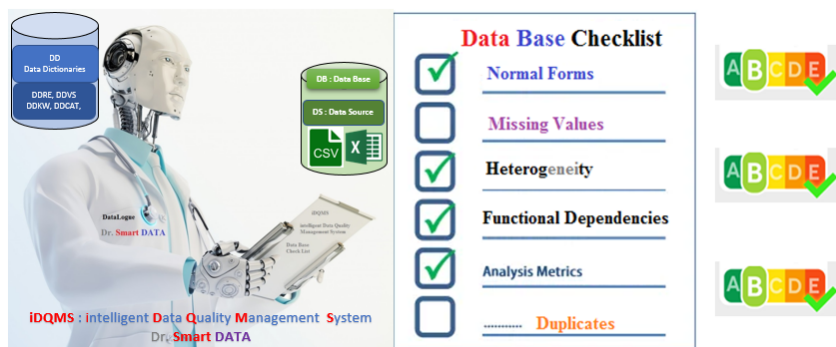
- de création des structures des tables de nom : **G7_CreatDon_CV.sql**
- de création des données de nom : **G7_InsertDon_CV.sql**
- de manipulation des données de nom : **G7_ManipDon_CV.sql**

Un jeu de test est primordial pour valider notre travail. Nous donnons, dans ce qui suit, les détails de ce jeu de données.

Il va falloir définir très précisément des « règles » (des contraintes très précises) sur les données (Construire/Définir les métadonnées) afin d'assurer/améliorer la qualité des données

Exemple : Le nom d'un(e) client(e) doit être :

1. NON vide, alphabétique,
2. en MAJUSCULE,
3. sans espace superflu (devant et entre les mots),
4. les seuls caractères spéciaux autorisés sont – et les lettres françaises avec accent...



3.1. Exemple d'instances de la BD des CV

Une instance inspirée des CV de la promotion M2EID2 de l'année 2020-2021 est donnée ci-dessous.

Table des candidats :

ID_CANDIDAT	TITLE	DESCRIPTION	NOM	PRENOM	PHOTO_IDENTITE
1 16	REC HERCHE UN...	D A T A E N G I ...	CHAMBRIER	Julian	/documents/photos/PH_2020_CHAMBRIER_Julian.jpg
2 17	Candidature M...	(null)	CHAOUCHE	Yacine	/documents/photos/PH_2020_CHAOUCHE_Yacine.jpg
3 18	Data Scientist	(null)	CHATTAT	Naima	/documents/photos/PH_2020_CHATTAT_Naima.jpg
4 19	(null)	Actuellement\mét...	CHFADI	Sara	/documents/photos/PH_2020_CHFADI_Sara.jpg
5 20	CV Stage	(null)	DJENADI	Sabrina	/documents/photos/PH_2020_DJENADI_Sabrina.jpg
6 11	CV_BERHILI_Faiza	Recherche de sta...	BERHILI	Faiza	/documents/photos/PH_2020_BERHILI_Faiza.jpg
7 12	CV_BIAN_Yiping	(null)	BIAN	Yiping	(null)
8 1	INGÉNIEURE DA...	À la recherche d...	SABOUNI	Soukayna	/documents/photos/PH_2020_SABOUNI_Soukayna.jpg

Table des formations de candidats :

ID_ORGANISATION	ID_CANDIDAT	ORGANISATION
1 13	11	Lycée Maurice Utrillo Stains
2 14	11	Institut Galilée - Université Sorbonne Paris Nord
3 15	11	Institut Galilée - Université Sorbonne Paris Nord
4 16	11	Institut Galilée - Université Sorbonne Paris Nord
5 17	12	Université BEIHANG
6 18	12	Université Paris 13
7 19	12	Université Paris 13
8 21	12	Université Paris 13
9 22	12	Université Paris 13
10 24	1	École Nationale des Sciences Appliquées

Table des expériences de candidats :

ID_EXPERIENCE	ID_CANDIDAT	POSTE	SUJET	DESCRIPTION
1 18	1	Stage de Fin d'Études Business Analyst	(null)	Analyse et conception de la plateforme e-commerce\ Ondatherm Prime SAP
2 19	1	Stage Développeur Full Stack	(null)	Conception et la Réalisation d'une application web pour le\ compte de D
3 20	1	Stage Analyst	(null)	Benchmark sur l'étude de l'opportunité ICT, cas de la\ cybersécurité .

3.2. Création des structures de données (SQL-ORACLE)

La création des tables, en SQL2, est détaillée ci-dessous.

```
CREATE TABLE CANDIDATS (
  ID_CANDIDAT          int primary key auto_increment,
  TITLE                nvarchar(300),
  DESCRIPTION           nvarchar(300),
  NOM                  nvarchar(20),
  PRENOM               nvarchar(100),
  PHOTO_IDENTITE       nvarchar(250),
  AGE                  int,
  EMAIL                 nvarchar(100),
  TELEPHONE            nvarchar(20),
  DATE_POSTE           Date,
  LINKEDIN              nvarchar(100),
  NATIONALITE           nvarchar(50),
  SITUATION             nvarchar(20),
  LETTRE_MOTIVATION    nvarchar(250),
  CV                   nvarchar(250),
  NUMERO_ADRESSE        nvarchar(10),
  NOM_ADRESSE           nvarchar(30),
  Code_Postal          nvarchar(10),
  VILLE                 nvarchar(50),
  PAYS                  nvarchar(100)
);
```

```
create table extra_activite (
  id_activite int primary key auto_increment,
  id_candidat int,
  nom_activite nvarchar(100),
  annee nvarchar(4),
  description nvarchar(300),
  constraint foreign key(id_candidat) references candidats(id_candidat)
);
```

```
CREATE TABLE CENTRE_INTERETS (
  ID_CANDIDAT int,
  NOM_CI      nvarchar(50),
  DESCRIPTION nvarchar(300),
  CONSTRAINT FK_CENINT_Candidats FOREIGN KEY(ID_CANDIDAT) REFERENCES
  CANDIDATS (ID_CANDIDAT));
```

```
create table Organisations (
  Id_Organisation int primary key auto_increment,
  Nom_Organisation nvarchar(100),
  Description nvarchar(300),
  Numero_Adresse nvarchar(10),
  Nom_Adresse nvarchar(30),
```

```

Code_Postale nvarchar(10),
Ville nvarchar(50),
Pays nvarchar(100)
);

CREATE TABLE FORMATIONS(
    Id_Formation int primary key auto_increment,
    ID_ORGANISATION int,
    Nom_Formation nvarchar(100),
    Description nvarchar(300),
    CONSTRAINT FOREIGN KEY(ID_ORGANISATION) REFERENCES ORGANISATIONS
(ID_ORGANISATION)
);

create table candidat_formation (
    Id_Candidat int,
    Id_Formation int,
    Mention nvarchar(20),
    Moyenne float(4,2),
    Diplome nvarchar(100),
    Date_Debut nvarchar(10), -- *** UPDATE
    Date_Fin nvarchar(10), -- *** UPDATE
    Description nvarchar(300),
    CONSTRAINT FOREIGN KEY(Id_Candidat) REFERENCES Candidats (Id_Candidat),
    CONSTRAINT FOREIGN KEY(Id_Formation) REFERENCES Formations (Id_Formation)
);

create table competences (
    Id_Competence int primary key auto_increment,
    Nom_Competence nvarchar(100),
    Description nvarchar(300)
);

create table certificats (
    Id_Certificat int primary key auto_increment,
    Nom_Certificat nvarchar(200),
    Id_Competence int,
    Id_Organisation int,
    description nvarchar(300),
    constraint foreign key(Id_Competence) references Competences(Id_Competence),
    constraint foreign key(Id_Organisation) references organisations(Id_Organisation)
);

create table candidat_certificats (
    Id_Candidat int,
    Id_Certificat int,
    Date_Obtenue nvarchar(10),
    Description nvarchar(300),
    constraint foreign key(Id_Candidat) references candidats(Id_Candidat),
    constraint foreign key(Id_Certificat) references certificats(Id_Certificat)
);

```


);

```
create table prix (  
  id_prix int primary key auto_increment,  
  nom_prix nvarchar(200),  
  description nvarchar(300)  
);
```

```
create table candidat_prix (  
  id_candidat int,  
  id_prix int,  
  date_obtenu nvarchar(10),  
  description nvarchar(300),  
  constraint foreign key(Id_Candidat) references candidats(Id_Candidat),  
  constraint foreign key(id_prix) references prix(id_prix)  
);
```

```
create table langues (  
  id_langue int primary key auto_increment,  
  nom_langue nvarchar(100),  
  description nvarchar(300)  
);
```

```
create table candidat_langue (  
  id_candidat int,  
  id_langue int,  
  niveau nvarchar(20),  
  description nvarchar(300),  
  constraint foreign key(Id_Candidat) references candidats(Id_Candidat),  
  constraint foreign key(id_langue) references langues(id_langue)  
);
```

```
create table candidat_competence (  
  id_candidat int,  
  id_competence int,  
  description nvarchar(300),  
  constraint foreign key(Id_Candidat) references candidats(Id_Candidat),  
  constraint foreign key(id_competence) references competences(id_competence)  
);
```

```
create table projets (  
  id_projet int primary key auto_increment,  
  id_candidat int,  
  nom nvarchar(100),  
  annee nvarchar(4),  
  lien nvarchar(200),  
  description nvarchar(300),  
  constraint foreign key(Id_Candidat) references candidats(Id_Candidat)  
);
```

```
create table projet_technologie (  
  id_projet int,  
  id_competence int,  
  description nvarchar(300),  
  constraint foreign key(id_competence) references competences(id_competence),  
  constraint foreign key(id_projet) references projets(id_projet)  
);
```

```
create table experiences (  
  id_candidat int,  
  id_organisation int,  
  poste nvarchar(100),  
  sujet nvarchar(200),  
  date_debut nvarchar(10),  
  date_fin nvarchar(10),  
  description nvarchar(300),  
  constraint foreign key(id_candidat) references candidats(id_candidat),  
  constraint foreign key(id_organisation) references organisations(id_organisation)  
);
```

3.3. Insertion des données dans une table et qualité des données (SQL/PLSQL-ORACLE)

L'insertion des données dans une table devra se faire de manière à respecter toutes les contraintes définies précédemment.

La « **bonne et complète** » insertion des données dans une table devrait se faire avec une **procédure PL/SQL** qui permet de **contrôler** le contenu de chaque colonne et non seulement avec la commande SQL « **INSERT INTO NomTable (...) VAUES (...);** » :

```
CREATE OR REPLACE PROCEDURE InsertDon_NomTable (Val1, Val2, ...) IS
...
BEGIN
  Anomalie:=FALSE;
  IF ... THEN ...
  IF Anomalie = FALSE THEN -- Si TOUT va Bien, aucune anomalie selon les contraintes !
    INSERT INTO NomTable (Col1, Col2, ...) VAUES (Val1, Val2, ...);
    COMMIT ;
  END IF ;
END;
```

3.4. Interrogations & Manipulations (SQL/PLSQL-ORACLE)

Nous avons défini des requêtes pour interroger la base de données en basant les besoins de l'évaluation d'admission :

- Les candidats qui maîtrisent plus qu'une langue
- Sélectionner les candidats ayant des certificats
- Afficher les candidats et leurs formations
- Afficher les candidats, leur nombre de prix et leurs prix
- Les candidats en liste d'attente
- Les candidats qui ont déjà réalisé des projets avec python

Les requêtes et les résultats sont présentés dans le fichier de manipulation **G7_ManipDon_CV.sql**

4. Sujet du Projet Annuel

Le sujet de notre projet annuel est « DES BASES AUX ENTREPÔTS DE DONNÉES, données structurées ou NON structurées (Oracle, MySQL ou MongoDB ?) ». Il s'agit d'un travail à réaliser par groupe d'étudiant.e.s.

Le travail à faire consiste à :

- Concevoir & développer une interface...
- Présentation orale à faire la semaine qui précède l'examen du mois de mars (fin des cours).
- Rapport (Word/PDF) + Diapositives (PPT/PDF)
- Codes (Oracle, MySQL et MongoDB)

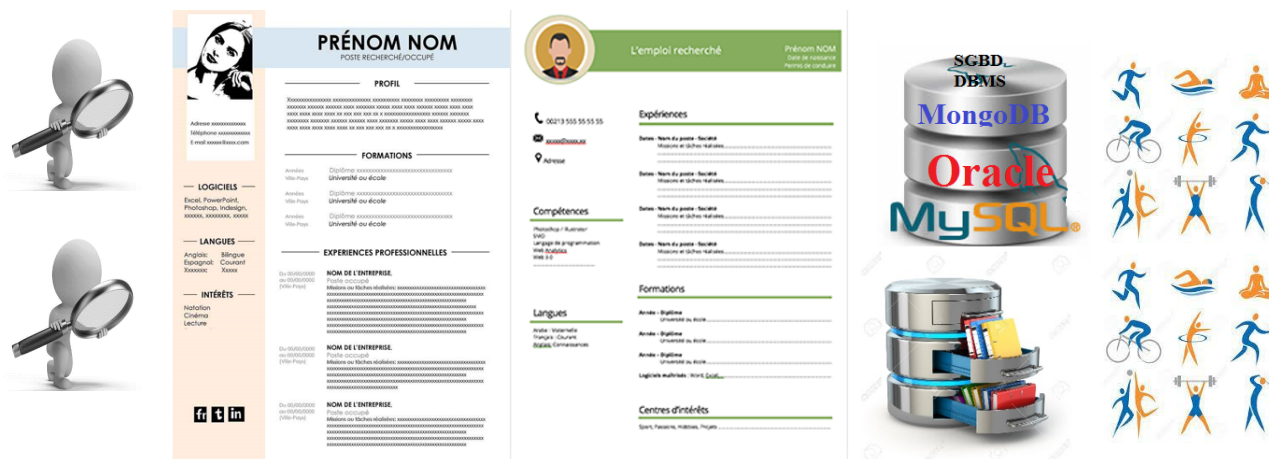
4.1. : Conception & Implantation de Bases de Données

Construction de **données structurées** à partir de **données NON structurées** (à partir de documents papiers ou numériques qui correspondent à des CVs ; **From PDF Files to SQL or NoSQL Data Bases**).

8.1.1. □ Donnez le schéma conceptuel (en utilisant l'un des deux formalismes UML ou E/A), le **schéma logique relationnel structuré SQL** et le **schéma logique Non-relationnel Non-structuré NoSQL** de la BD qui permet de stocker les données pertinentes que l'on peut trouver dans les CVs des personnes qui candidatent pour une inscription en Master 2 Informatique « Exploration Informatique et Décisionnel » (M2EID2).

8.1.2. □ Donnez une instance (quelques lignes, au moins 50, ou objets ou enregistrements réalistes) de votre BD !

8.1.3. □ Donnez TROIS implémentations sur Oracle, MySQL et MongoDB



4.2. : Conception & Implantation d'Entrepôts de Données

En disposant de données (stockées dans la BD des CV) sur une période de 10 années,

4.2.1. □ Concevez un entrepôt de données qui permet d'analyser profondément le profil des candidats en M2EID2 ! (Vous êtes libres de prendre les hypothèses que vous voulez !).

4.2.2. □ Donnez une instance (quelques lignes, au moins 50, ou objets ou enregistrements réalistes) de votre ED !

4.2.3. □ Donnez TROIS implémentations sur Oracle, MySQL et MongoDB



{ name: mongo, type: DB }

3.3. MongoDB:

MongoDB est un système de base de donnée orienté objet, dynamique, stable, scalable et sans SQL. Les objets sont stockés sous format JSON dans des documents séparés. Au lieu de stocker les données sous format de tables avec des valeurs, on utilise une hiérarchie et un système d'objet JSON pour créer un système plus adaptatif.

L'intérêt de MongoDB, c'est d'implémenter un système à hautes performances avec une scalabilité parfaite. Le système est extrêmement simple à installer, il fonctionne sous tout les système d'exploitation et il existe des librairies pour le manipuler dans tout les langages.

Notre implémentation : Afin de s'entraîner pour le stage, j'ai chois de travailler sur deux procédures parmi les plusieurs possibilité qu'on pouvait faire :

- Insertion des données avec un fichier d'extension Json dans une la base de données, afin d'avoir un peu plus de travaille a faire sur le pre processing des données avec tableau, dans notre BDD MongoDB les données ne sont pas structuré et on a la possibilité d'insérer toutes types d'information pour chaque attribut.

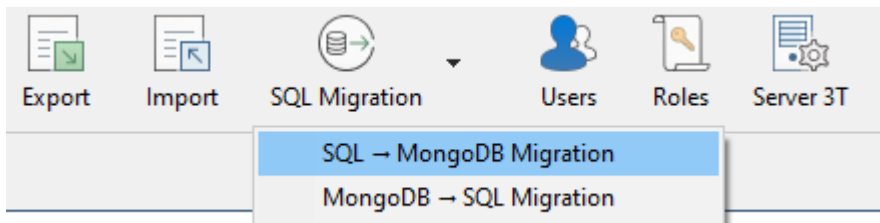
- Migration des données à partir de la base de données SQL grace à l'outil **Studio 3T**

Studio 3T comme IDE MongoDB : Studio 3T fait exactement cela en fournissant une interface utilisateur graphique dotée d'éditeurs avec saisie semi-automatique et coloration syntaxique, validation JSON intégrée, génération automatique de code de requête en sept langues et de nombreuses autres fonctionnalités qui vous aident à travailler plus rapidement et à gagner du temps.

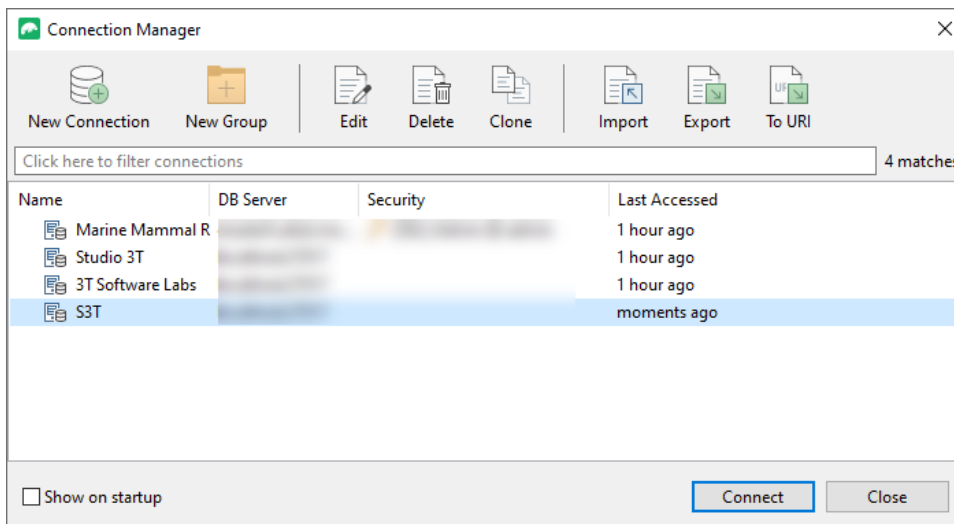
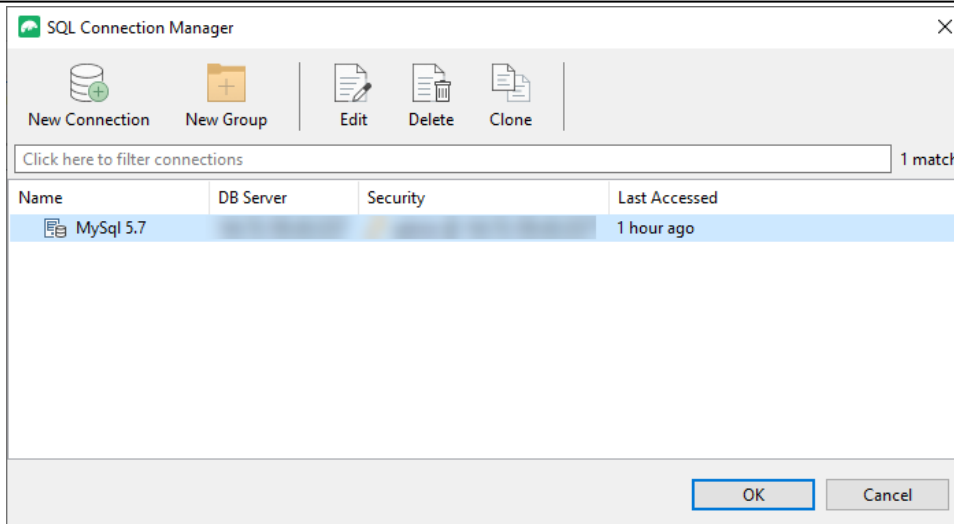
Source : <https://studio3t.com/>

Connectez-vous à un serveur SQL : Juste après avoir ouvert l'onglet Migration SQL vers MongoDB, la première chose à faire est de vous connecter à un serveur de base de données relationnelle. Les serveurs actuellement en charge sont MySQL, PostgreSQL, Oracle Database, Microsoft SQL Server et Sybase.

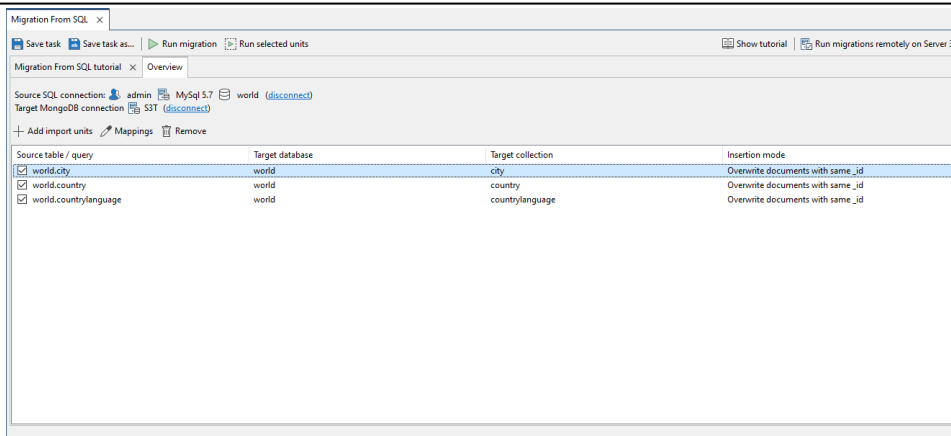
- Cliquez sur le bouton Définir la connexion SQL.
- Sélectionnez / configurez une connexion SQL via le Gestionnaire de connexion SQL.



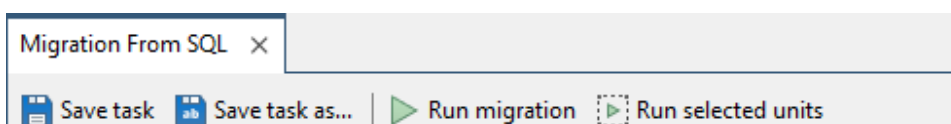
The image shows a 'New SQL Connection' dialog box. It has a title bar with a close button. The 'Name' field is set to 'new-sql-connection'. The 'Group' dropdown is set to '<root level>'. There are three tabs: 'Server' (selected), 'SSH Tunnel', and 'Custom Options'. Under the 'Server' tab, the 'SQL Server Type' dropdown is set to 'MySQL Server'. The 'Host' field is '127.0.0.1', the 'Port' field is '3306', the 'Username' field is 'root', the 'Password' field is masked with dots, and the 'Database' field is empty. At the bottom, there are buttons for 'Test Connection', 'Test on Server 3T', 'Save' (highlighted), and 'Cancel'.



Une fois connecté à un serveur, cliquez sur le bouton Ajouter des unités d'importation. Choisissez Avec la table source.



Une boîte de dialogue apparaîtra dans laquelle vous pourrez explorer les tables de votre serveur SQL et en sélectionner autant que vous le souhaitez.



Une fois que vous avez confirmé, une nouvelle unité d'importation sera créée pour chaque table.

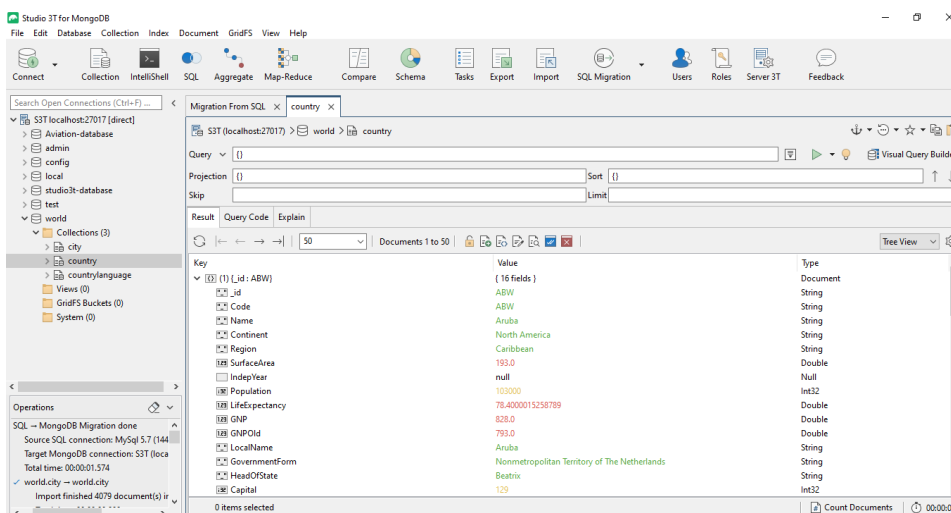
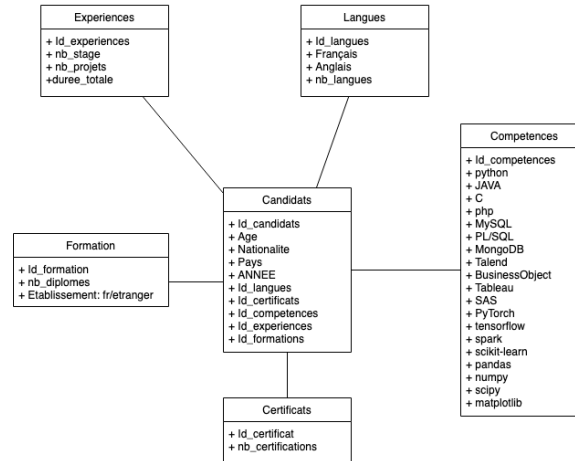


Schéma de la Data Warehouse :

Data Warehouse ou Entrepôt de données est un système transversal, qui complète les systèmes opérationnels, il contient de grandes quantités de données provenant de diverses sources, sauvees sous un schéma de données unique, et résidant à un endroit unique.

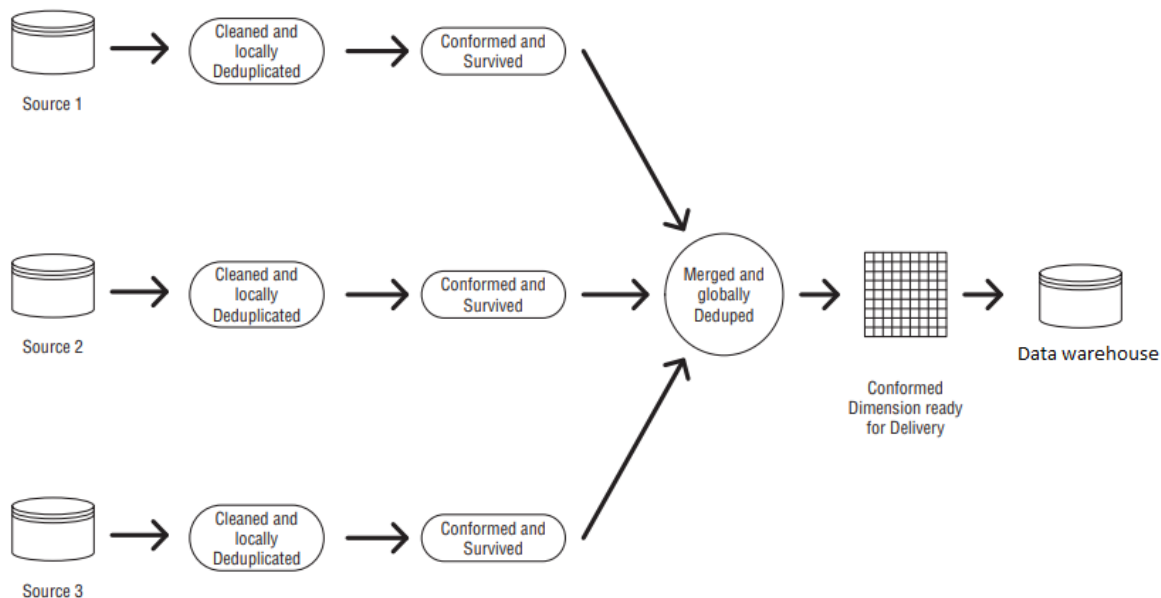
Le schéma de data warehouse est réalisé par un schéma en étoile, ou la table de fait étant le candidat (ou candidature) le reliant aux différentes tables des dimensions par une relation 1-1 représentant chacune les caractéristiques de chaque candidature.



Ce schéma sera utilisé dans l'analyse afin de répondre aux questions sur les caractéristiques communes de toutes les candidatures en faisant une comparaison sur les différentes compétences, nombre d'expériences (Experiences.duree_totale), les langues que maîtrise le candidat ou en cours les certifications obtenues.

L'intégration de données :

En gros, on fait l'intégration de données avec trois parties : Nettoyer les données locales, Intégrer et Dédupliquer les données depuis plusieurs sources, et Capturer les données incrémentales.

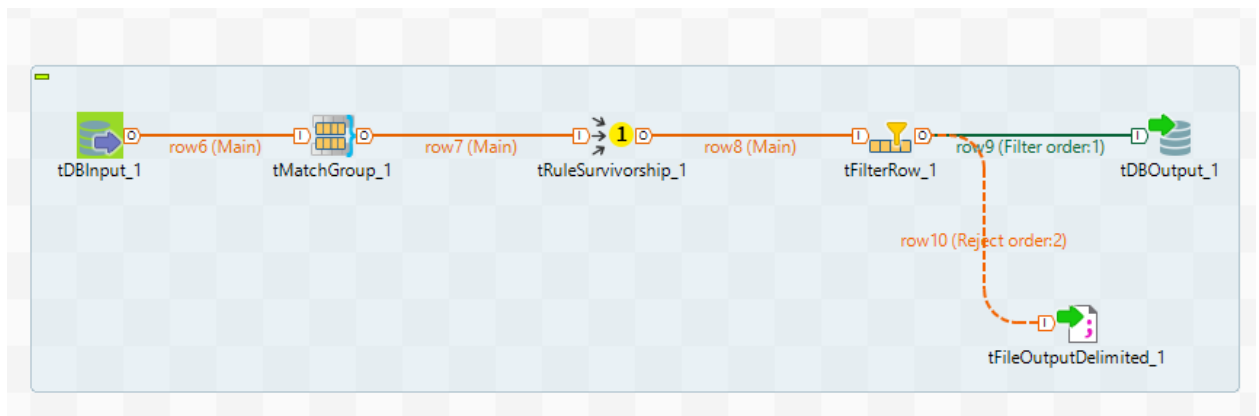


1. Nettoyer les données locales

À côté des triggers, on utilise encore Talend pour contrôler la qualité de données et les doublons.

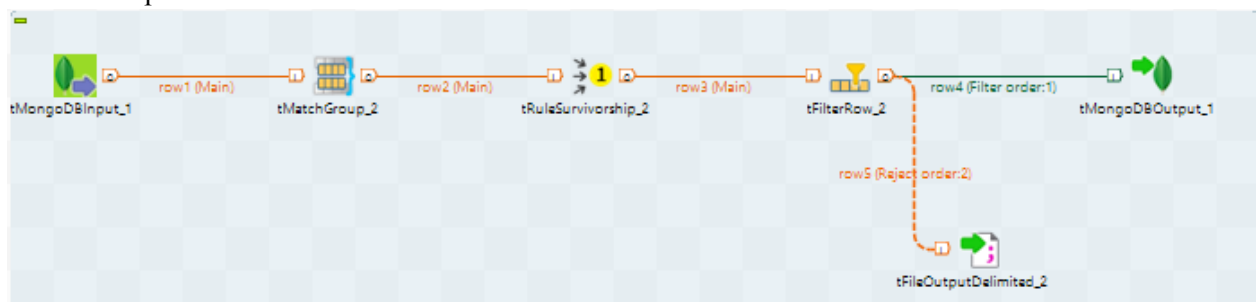
Sur Talend, le processus de dédupliquer à plusieurs étapes suivantes :

- On fait la réunion de groupe des doublons par **tMatchGroup**. On configure les calculs, les algorithmes pour déterminer les doublons. Par exemple, avec la table **CANDIDATS**, on utilise les colonnes et les algorithmes suivantes : Nom (Soundex FR), Prenom (Soundex FR), Téléphone (Hamming algorithm), Email (Hamming) et Nationalité (Exact – ignore case) afin connaître les doublons.
- Ensuite, on choisit les valeurs à garder dans le record master par **tRuleSurvivorship**. Par exemple, on garde une bonne valeur **Email** (approprié pour une regex) dans plusieurs valeurs Email et une valeur **Prenom** plus longue.
- Après, on utilise **tFilterRow** pour filtrer les records master (master = true) et les records jetés aux 2 flux.
- Enfin, on modifie le record master, supprime les records jetés et mets les records doublons dans un fichier csv pour revoir après.



Il y a des étapes pareilles pour les tables que l'on veut dédupliquer avant l'intégration de données comme la table Formations, table Experiences, table Langue, ...

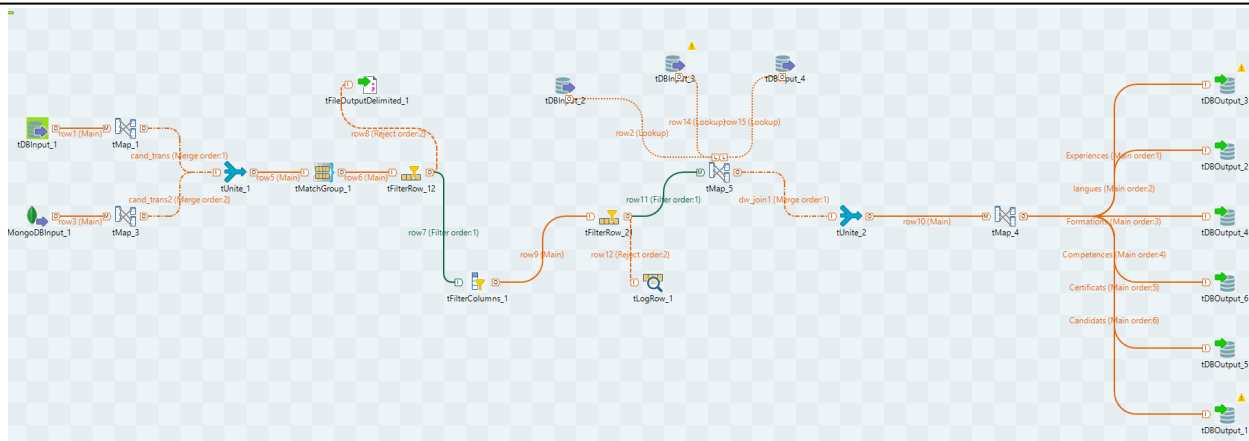
Pour la base de données comme MongoDB, il est juste différent sur la requête de collection de première étape. Les autres étapes sont maintenues.



2. Intégrer et Dédupliquer les données depuis plusieurs sources

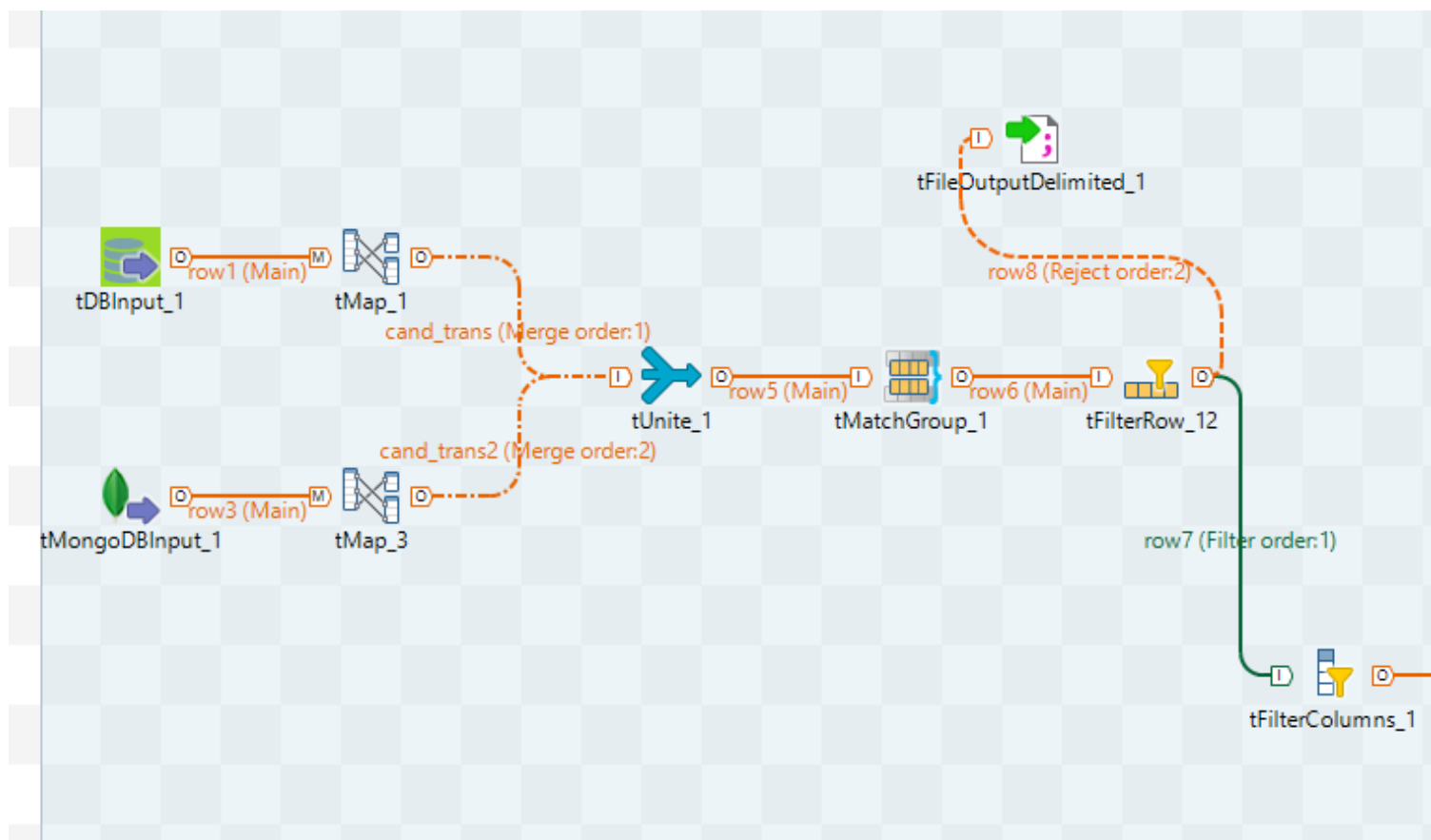
Après première partie, on intégrera les données venant de plusieurs sources comme MySQL et MongoDB. L'intégration de données est un travail compliqué et il faut utiliser plusieurs sous-systèmes pour gérer les données dans l'entrepôt de données et dans plusieurs sources. Par exemple, le système Master Data Management, le système Reference Data Management, le système Change Data Capture,...

Dans ce projet, nous avons fait un ETL pour intégrer les données depuis deux sources MySQL et MongoDB comme l'image suivant :



Il est possible d'avoir les doublons (1-1) entre deux sources. Par conséquent, on doit supprimer des doublons avant d'insérer à l'entrepôt de données. Cette première portion est de dédupliquer des doublons en utilisant **tMap**, **tUnite**, **tMatchGroup**, **tFilterRow** et **tFilterColumn**.

tMap est de transformer les valeurs aux formes appropriées avec l'entrepôt de données. Après, **tUnite** est d'intégrer les données venant de plusieurs sources. Ensuite, on fait une déduplication avec **tMatchGroup** et **tFilterRow** pour filtrer les records masters qui pourront être traités encore.

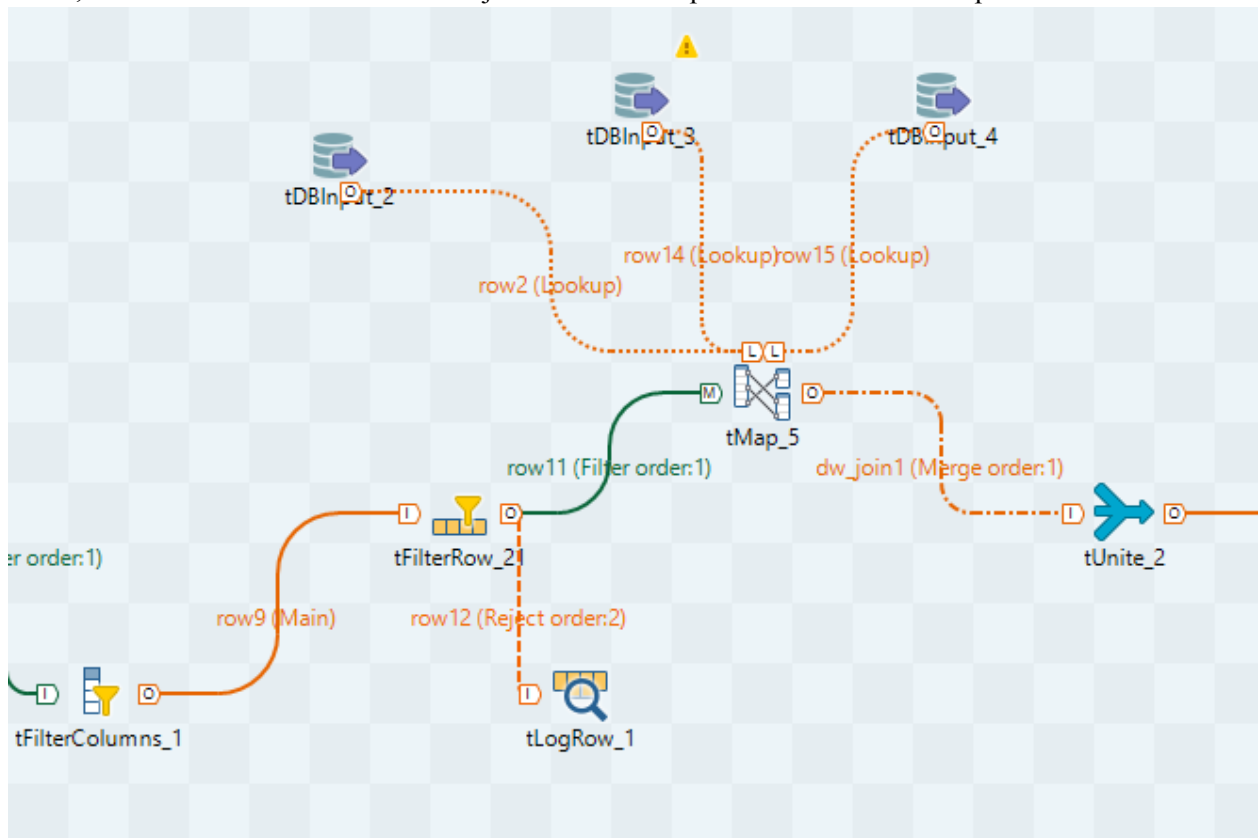


Après la déduplication, on va collecter les données nécessaires pour l'entrepôt de données. On utilise **tFilterRow** afin de séparer les records correspondants avec les sources. Par exemple, tFilterRow a deux sorties, une pour MySQL et une pour MongoDb.

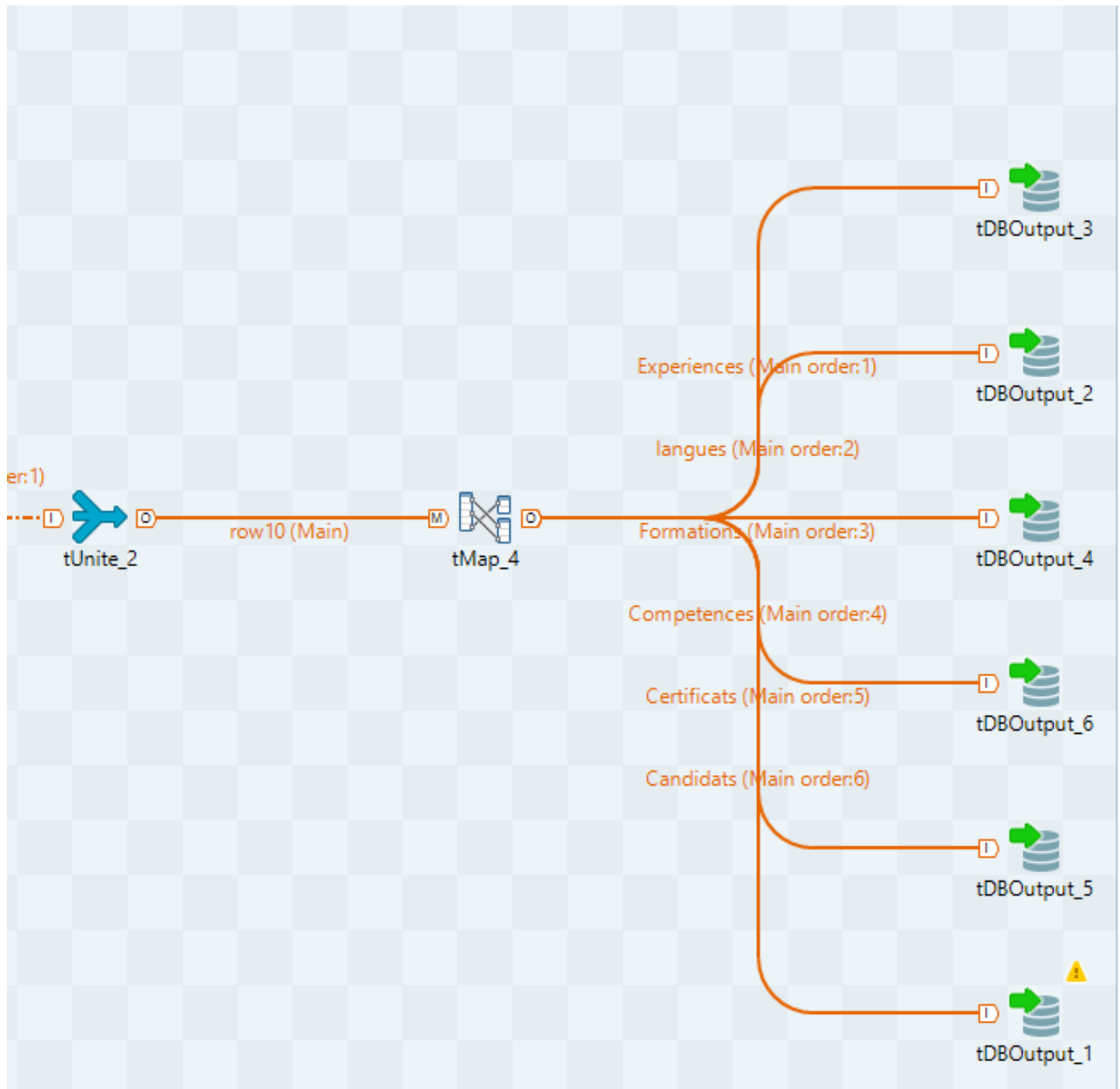
tMap est de joindre avec des autres tables (formations, experiences, langues) pour collecter les autres données comme nombre de formations d'un candidat, nombre de langues d'un candidat ou ses compétences, ...

En fait, il y aura **tMap** et **tMongodbInput** au lieu de **tLogRow** pour collecter des données dans MongoDB. Mais on a une difficulté avec les requêtes de Mongo et on essaie de résoudre cette difficulté avant la présentation.

Enfin, les données de deux sources sont jointes au **tUnite** pour les insérer à l'entrepôt de données.



Enfin, il faut ajouter les données à l'entrepôt de données en partant des tables dimensions d'abord : **EXPERIENCES**, **LANGUES**, **FORMATIONS**, **COMPETENCES**, **CERTIFICATS**. Après les tables dimensions, on insère des données à la table fait **FAIT_CANDIDATS** en gardant les clés étrangères des tables dimensions. On a réussi à intégrer les données depuis plusieurs sources au l'entrepôt de données avec le schéma correspondant de l'entrepôt de données.

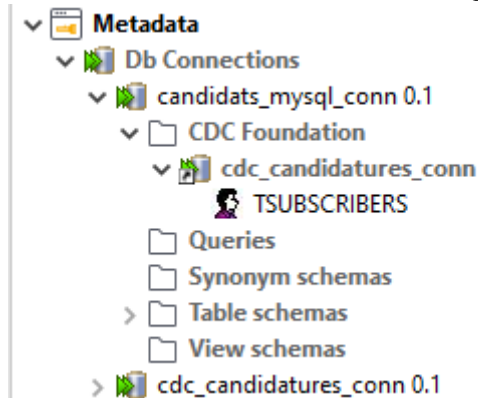


3. Capturer de données ajoutées incrémentales

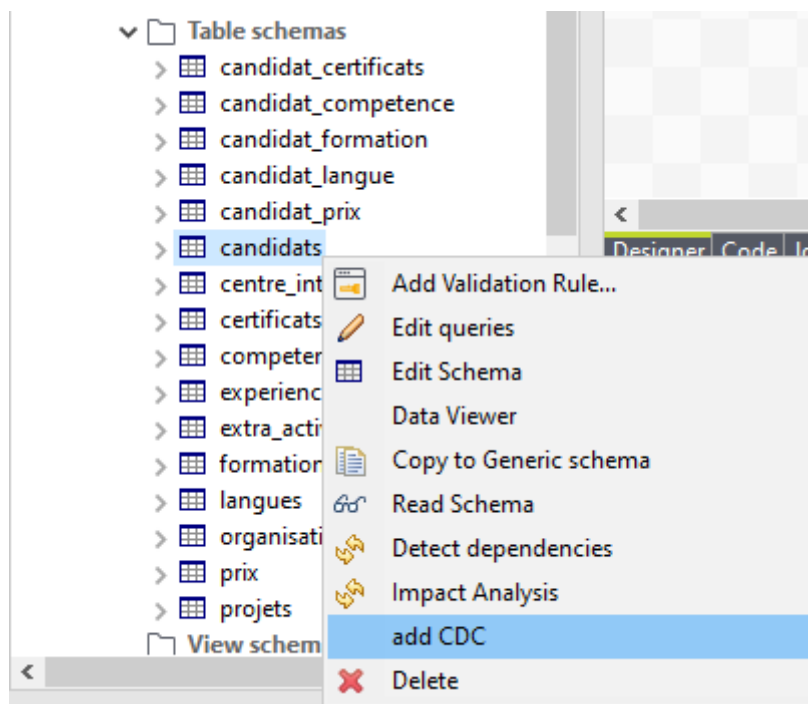
Pendant le chargement historique initial de l'entrepôt de données, la capture des modifications du contenu des données n'est pas importante car on charge toutes les données à partir d'un point dans le temps.

Cependant, de nombreuses tables d'entrepôt de données sont si grandes qu'elles ne peuvent pas être rafraîchies pendant chaque cycle ETL. Vous devez avoir une capacité de transférer seulement les changements pertinents aux données source depuis la dernière mise à jour.


Sur Talend, on peut utiliser la fonctionnalité Change Data Capture pour tracer des modifications. On doit créer une base de données qui suit et stocke les modifications des données de la base de données principale.



Après, on peut ajouter une capture de modification à la table qu'on veut suivre les modifications.



Talend va créer les vues et les triggers pour voir et suivre des modifications sur les données. On peut ajouter des CDC aux tables importantes comme Candidats, Formations, Langues, etc...


 Create Subscriber and Execute SQL Script

— □ ×

```

alter table `cdc_candidatures`.`TSUBSCRIBERS` modify `TALEND_CDC_TABLE_TO_WATCH` VARCHAR(50)
INSERT INTO `cdc_candidatures`.`TSUBSCRIBERS`
(
  `TALEND_CDC_TABLE_TO_WATCH`,
  `TALEND_CDC_SUBSCRIBER_NAME`,
  `TALEND_CDC_CREATION_DATE`
)
values ('candidatures. formations', 'candidats_sub', sysdate());

CREATE TABLE `cdc_candidatures`.`TCDC_ formations`
(
  `TALEND_CDC_SUBSCRIBERS_NAME` VARCHAR(50) NOT NULL,
  `TALEND_CDC_STATE` VARCHAR(1),
  `TALEND_CDC_TYPE` VARCHAR(1),
  `TALEND_CDC_CREATION_DATE` DATETIME,
  `Id_ Formation` INT(10) NOT NULL
);

```

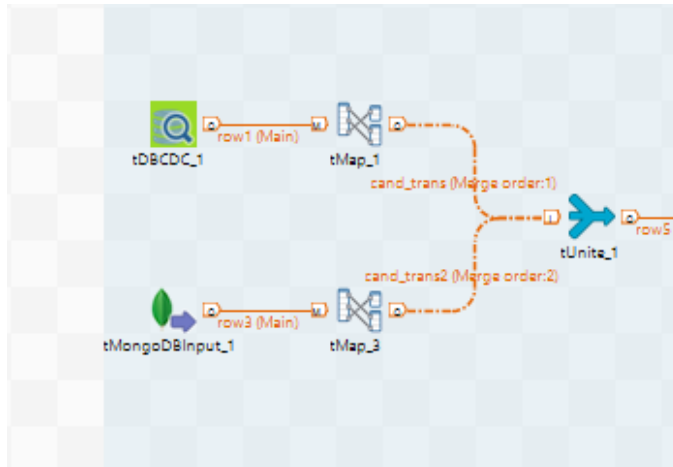
Events to catch: ☒ Insert ☒ Update ☒ Delete

Subscriber Name:

Execute

Close

Il y a seulement une différence qu'on va utiliser **tDBCDC** au lieu de **tDBInput** pour les cycles ETLs prochaines. Maintenant, Talend ne supporte pas CDC pour MongoDB, on pourra utiliser une méthode alternative avec les colonnes **LAST_MODIFIED_TIME** pour tracer des modifications.

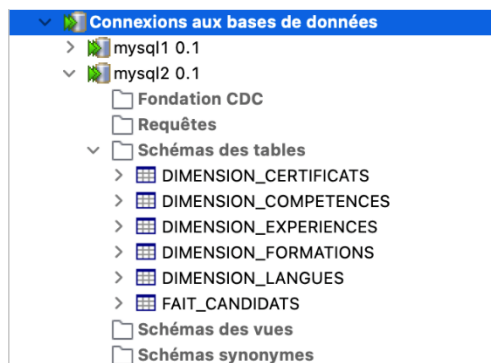


Insertion des données MySQL vers le schéma de la base de données Data Warehouse :

Après l'étape de data-cleaning et de déduplication les données subissent un ensemble de transformation avec le logiciel Talend afin de passer d'un schéma initial utilisé dans la base de données relationnel à un schéma de datawarehouse qui stockera les tables dimensions ainsi que la table de faits.

Après création des connexion MySQL dans Talend et la récupération des schémas utilisés pour les tables sources et destinations (voir figure ci-dessous), on pourra ensuite facilement instancier en tant qu'objet du type :

- 'tDBInput' : pour les connexions MySQL qui serviront en lecture (select) dans une table de la base de données d'entrée (relationnel).
- 'tDBOutput' : pour les connexions MySQL qui serviront en écriture (insertion/update) dans une table de la base de données de sortie (DW).



Sur cette figure on notera :

- mysql1 : base de données relationnel,
- mysql2 : data warehouse

Important : Afin de permettre une mise à jour sans erreurs dans nos tables, on a choisi dans le champ « Action sur les données », l'option : « Insert or update on duplicate key or unique index ».

"FAIT_CANDIDATS" (tDBOutput_1) (MySQL)

Paramètres simples

Database: MySQL Apply

Type de propriété: Référentiel Bases de données (MySQL):mysql2

Version de la base de données: Mysql 8

☐ Utiliser une connexion existante

Hôte: "localhost" Port: "3306"

Base de données: "datawarehouse"

Utilisateur: "root" Mot de passe: "*****"

Table: "FAIT_CANDIDATS"

Action sur la table: Défaut Action sur les données: Insert or update on duplicate key or unique

Schéma: Référentiel Bases de données (MySQL):mysql2 - FAIT_C... Modifier le schéma Sync colonnes

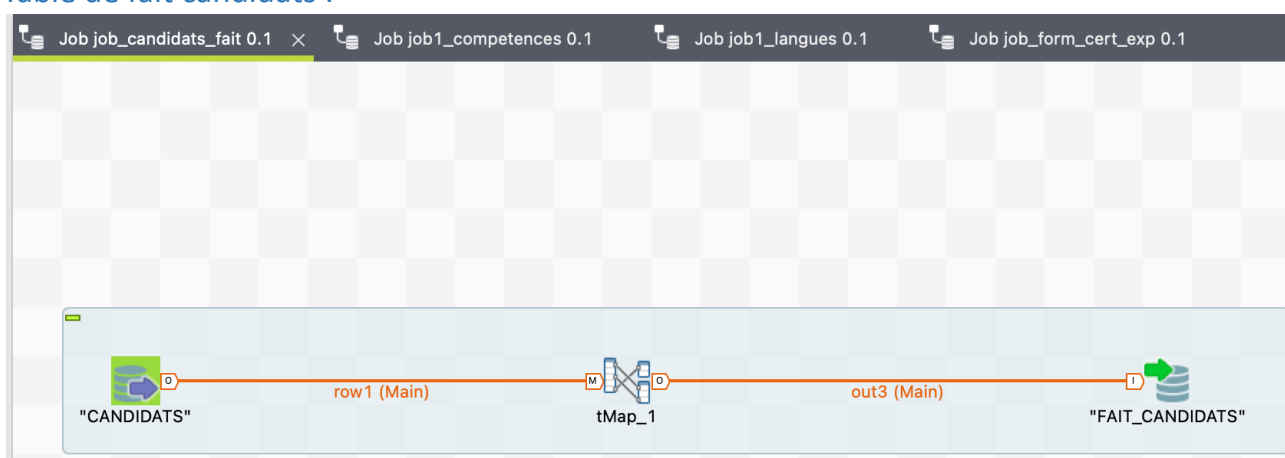
Source de données

This option only applies when deploying and running in the Talend Runtime

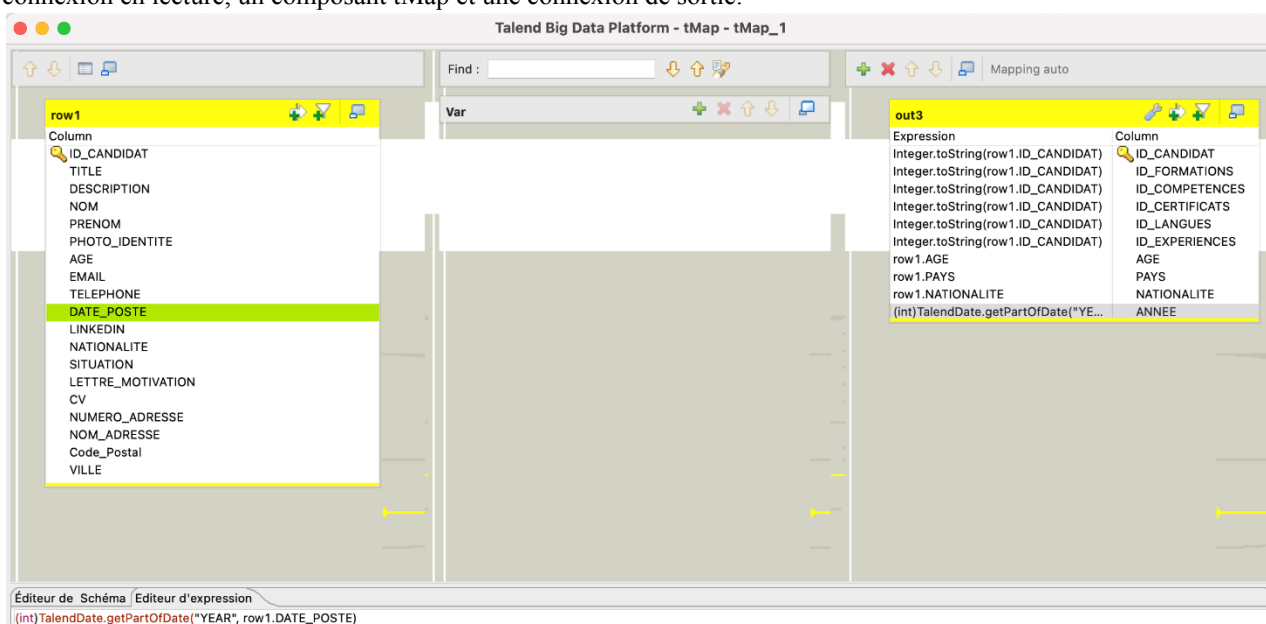
☐ Spécifier l'alias de la source de données

☐ Arrêter en cas d'erreur

Table de fait candidats :



Le schéma ci-dessus illustre l'utilisation des différents composants dans ce cas (simple) en utilisant une instance de connexion en lecture, un composant tMap et une connexion de sortie.



Le composant tMap est très utilisé dans Talend car il permet de diriger le flux de données en entrée vers celui de la sortie élément par élément en effectuant diverses opérations intermédiaires. Ces opérations sont décrites en insérant un code Java dans le champ expression de chaque sortie.

Dans notre schéma, l'ID de chaque dimension (clés étrangères) sera égale à celui de l'ID du candidat cela nous permettra de faciliter la recherche des différents éléments.

Depuis la table source « CANDIDATS » on récupère d'autres éléments intéressants à analyser dans la partie visualisation de données :

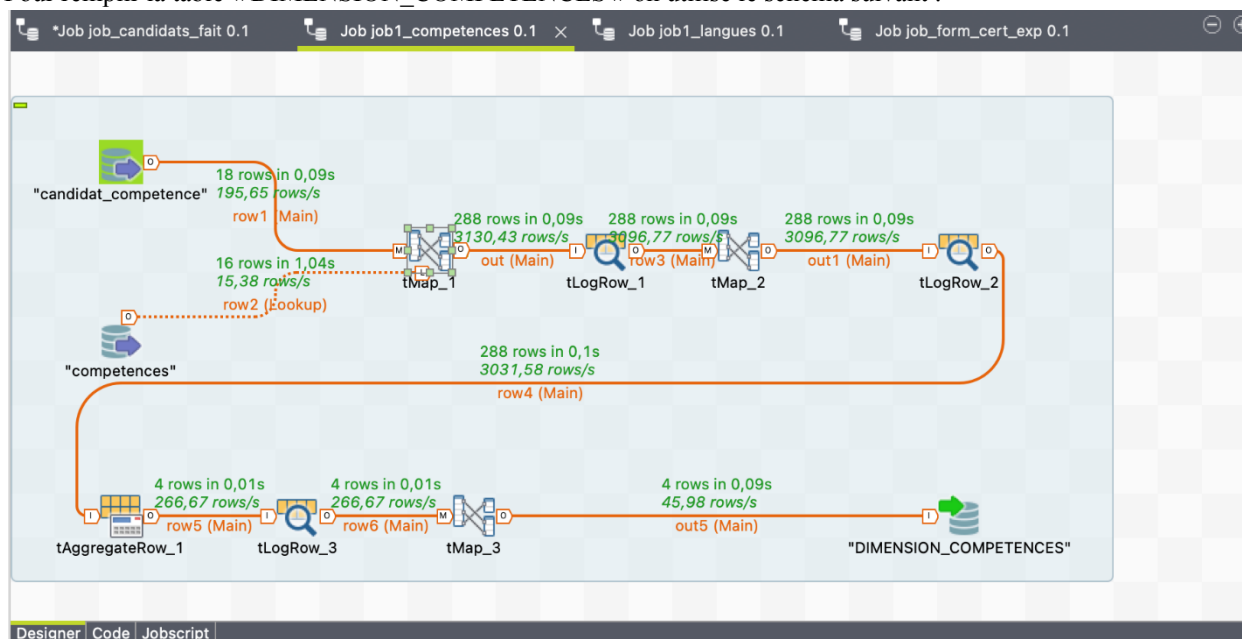
- AGE
- PAYS (de résidence)
- NATIONALITE

Vu que dans le projet, on s'intéresse aussi aux CV des dix dernières années, on utilisera le champ « DATE_POSTE » pour récupérer l'année de candidature (ANNEE).

Dans le champ expression de cette sortie, on utilisera la classe Java « TalendDate » fournie par Talend et qui permet de faire de nombreuses opérations sur les objets Java de type Date. Dans ce cas, nous avons utilisé la méthode getPartDate pour extraire l'année de la date du poste du CV.

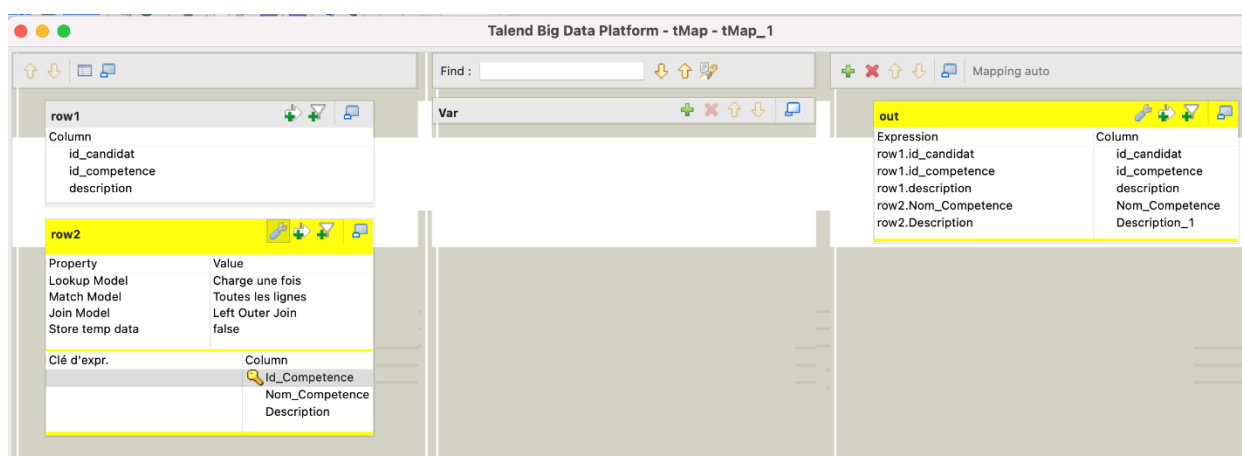
Job d'insertion dans la table dimension compétences :

Pour remplir la table « DIMENSION_COMPETENCES » on utilise le schéma suivant :



Les opérations effectuées sont les suivantes :

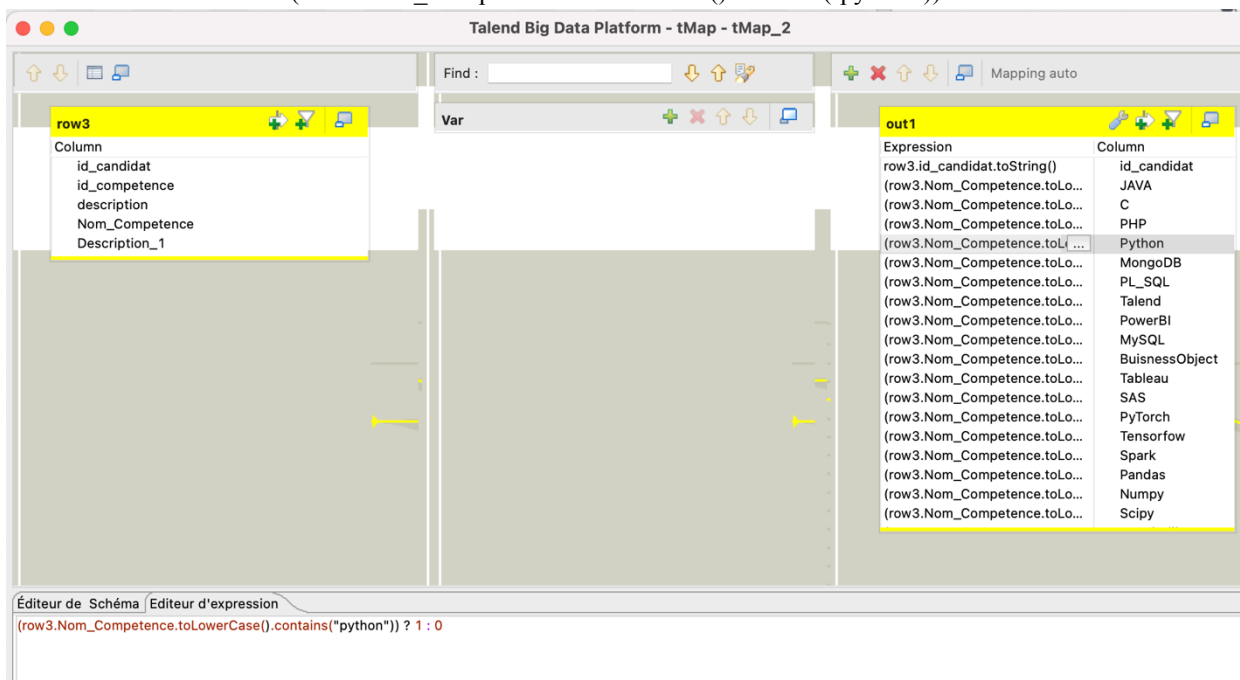
1. On fait tout d'abord une jointure sur les tables sources « CANDIDAT_COMPETENCES » et « COMPETENCES » en utilisant la clé primaire de la compétence « ID_COMPETENCE ». Cela nous permettra d'avoir dans une et même table l'ID du candidat et les informations liées aux compétences de ces candidats.



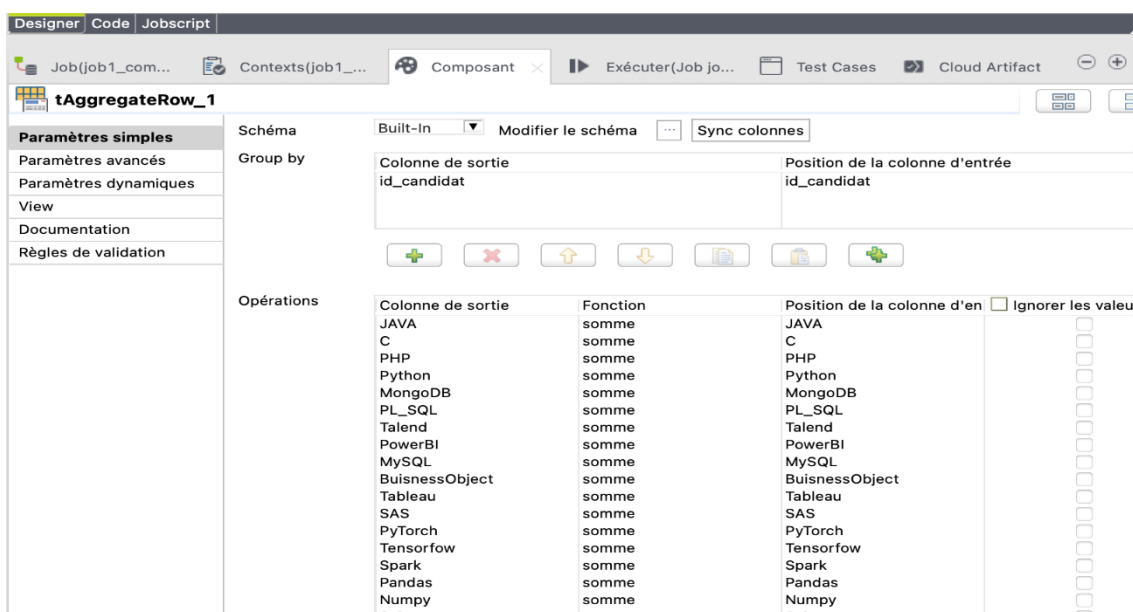
2. Grace à un autre tMap, on crée par la suite plusieurs nouvelles variables (de type entier), une par type compétence, qu'on initialisera 1 lorsque le nom de la compétence contiendra la compétence associée (sinon 0).
 - o Exemple pour le cas du langage Python (voir figure ci-dessous) :

- la sortie out1.Python sera généré avec le code suivant :

```
(row3.Nom_Competence.toLowerCase().contains('python')) ? 1 : 0
```

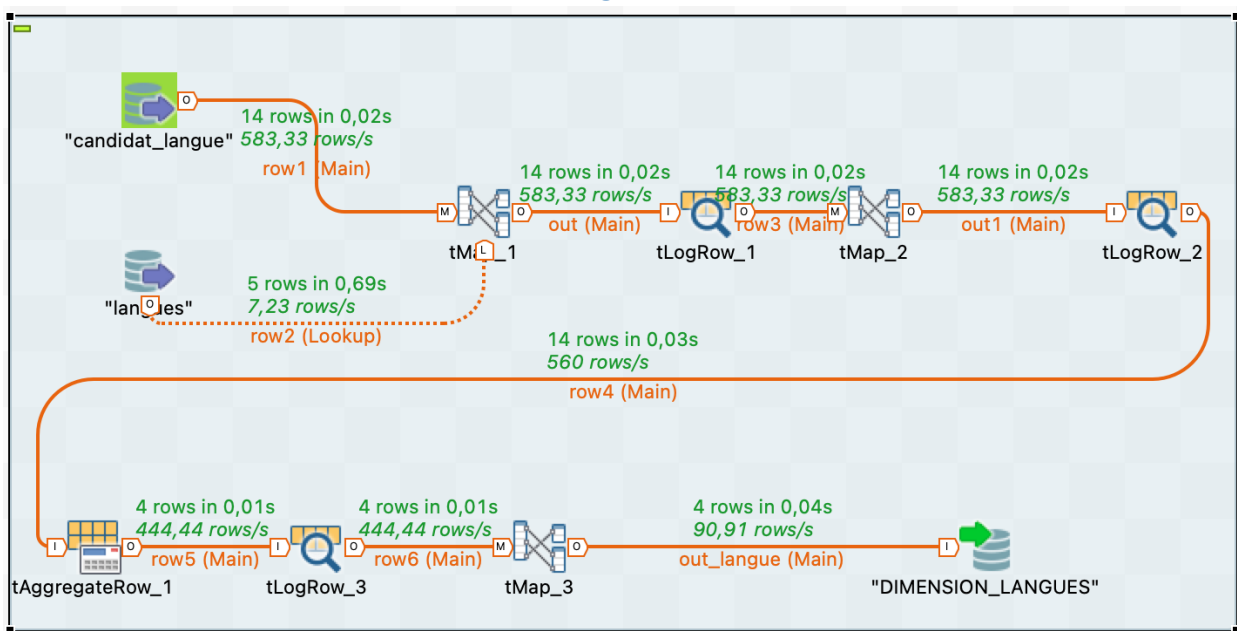


- En utilise ensuite un tAggregateRow qui nous permettra de faire un regroupement des entrées de la table résultante avec par « id candidat » en utilisant la fonction d'agrégation **SOMME** sur les variables nouvellement créés. Lorsque cette somme n'est pas nulle cela veut dire que le candidat a cette compétence.



- Enfin la dernière étape consiste à respecter le format de sortie, en transformant avec un autre tMap ces nouvelles variables en boolean (avec un simple test if >0).

Job d'insertion dans la table dimension langues :



Comme dans la table dimension compétences, on utilise le même principe pour créer les nouvelles variables suivantes (même opérations utilisées) :

- Français : si le candidat parle français
- Anglais : si le candidat parle l'anglais
- Nb_langue : nombre de langue parlé par le candidat

La figure ci-dessous illustre l'équivalent de l'étape 3 cité dans la description du Job de la dimension compétence.

tAggregateRow_1

Paramètres simples

Paramètres avancés

Paramètres dynamiques

View

Documentation

Règles de validation

Schéma

Built-In

Modifier le schéma

Sync colonnes

Group by

Colonne de sortie

id_candidat

Position de la colonne d'entrée

id_candidat

Opérations

Colonne de sortie	Fonction	Position de la colonne d'ent	Ignorer les valeurs null
francais	somme	francais	<input type="checkbox"/>
anglais	somme	anglais	<input type="checkbox"/>
nb_langue	count	id_candidat	<input type="checkbox"/>

Sous Jobs d'insertion dans les tables dimensions formation, certifications et expériences :

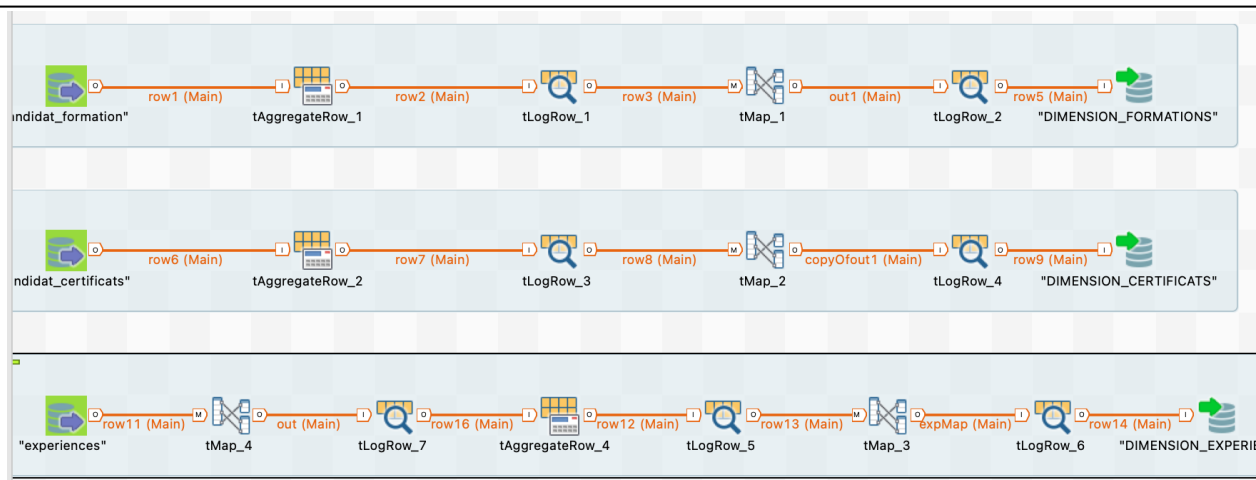
Les jobs suivants permettent de calculer les différents compteurs utilisés dans les tables dimensions associés tel que décrit dans leurs schémas en utilisant les composants de base de Talend.

Dans le dernier sous Job, on a rajouté le calcul de la durée total des expériences pour quantifier en nombre de mois l'expérience du candidat.

Dans un premier temps la durée de chaque expérience passe par un composant tMap qui calcul la différence en nombre de mois de chaque expérience :

```
o (int)TalendDate.diffDate(row11.date_fin, row11.date_debut,"MM",true)
```

Puis avec une fonction d'agrégation on somme l'ensemble des durées obtenu précédemment (SOMME). Le résultat en sortie est envoyé en tant que type entier.



Remarque importante :

Dans l'exécution des Job il est important de lancer d'abord les jobs des tables dimension avant de remplir la table candidat. Cela permet d'éviter des erreurs d'insertion car les clés sont déclarées en tant que clés étrangères avec une références au tables dimension.

5. Dataviz avec Tableau Software



Qu'est-ce que la data visualisation, ou dataviz ?

"Une image vaut mille mots."

Confucius n'avait pas encore d'ordinateur lorsqu'il a prononcé cette phrase. Elle est aujourd'hui plus vraie que jamais face aux montagnes de données collectées en entreprise. Elles restent complexes à analyser, expliquer et partager sous leur forme brute.

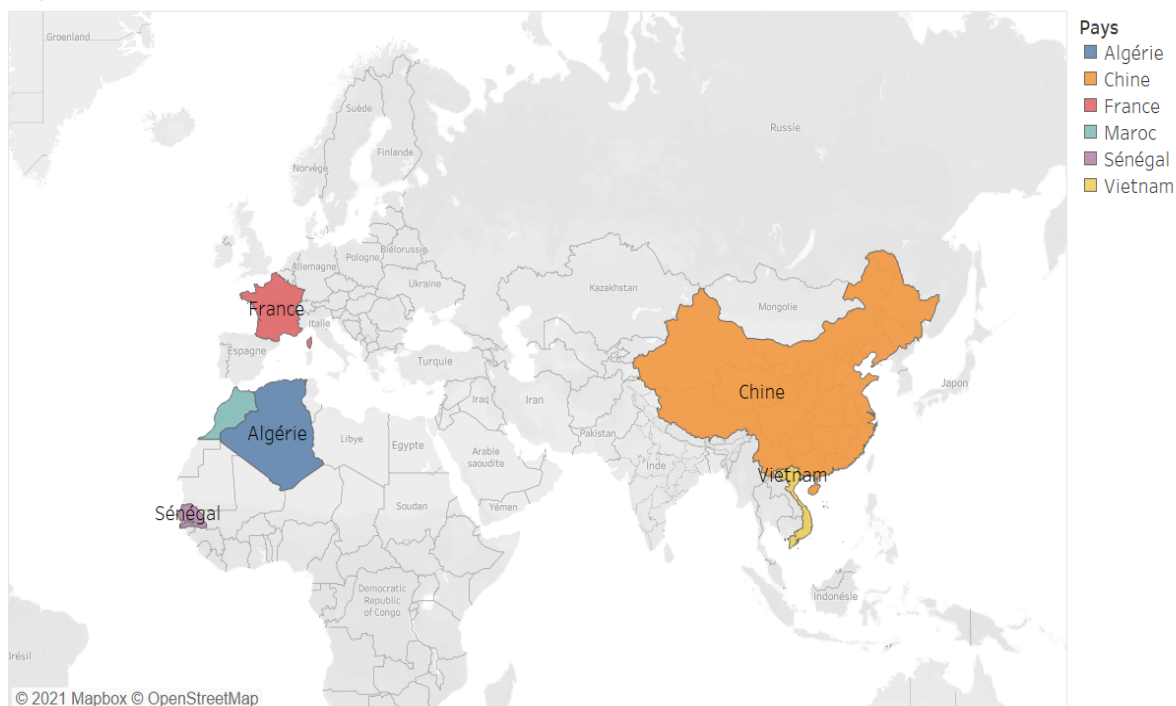
Nous avons déjà tous passé des heures à analyser des données à partir de tableaux Excel interminables et complexes.

La « dataviz » vient mettre un terme à cette tâche fastidieuse. Elle consiste à communiquer des chiffres ou des informations brutes en les transformant en objets visuels : points, barres, courbes, cartographies...

En alliant fonctionnalités simples et esthétisme, elle offre un gain de temps conséquent dans la recherche et l'analyse des données. C'est aussi un outil de communication puissant.

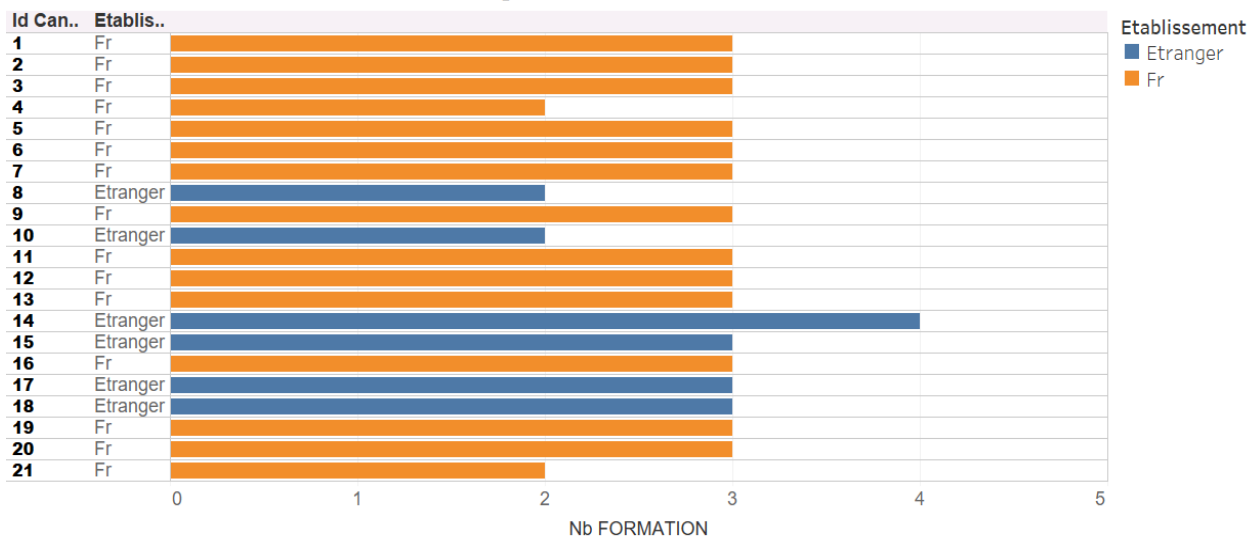
Pour visualiser les données extraites à partir des Cv on a travaillé sur « **Tableau Software** » un logiciel en ligne considérée comme un des **leaders du marché de la dataviz** aux côtés de **Microsoft** et **Qlik**. Cette solution permet d'analyser les données avec une approche tableau de bord et data visualisation totalement intuitive.

Pays des candidats



On peut voir sur cette carte avec couleurs les pays d'où viennent les candidats : France, Algérie, Maroc, Sénégal, Chine, Vietnam.

Etablissements et nombre de formation de chaque candidat



Ce diagramme à barres nous montre l'établissement de chaque candidat (établissement français ou étranger) ainsi que le nombre de formations suivies dans ce dernier.

Age et nationalité de chaque candidat

Id Cand..	Nationalite	AGE
1	Française	22
2	Vietnamien	22
3	Algérienne	22
4	Française	24
5	Française	26
6	Chinois	30
7	Française	37
8	Marocaine	25
9	Française	22
10	Marocaine	25
11	Vietnamienne	Null
12	Algérienne	Null
13	Sénégalaise	Null
14	Marocaine	Null
15	Marocaine	Null
16	Française	21
17	Sénégalaise	Null
18	Marocaine	Null
19	Française	23
20	Française	24
21	Algérienne	25

On peut voir ce premier tableau la nationalité et l'âge de chaque candidat.

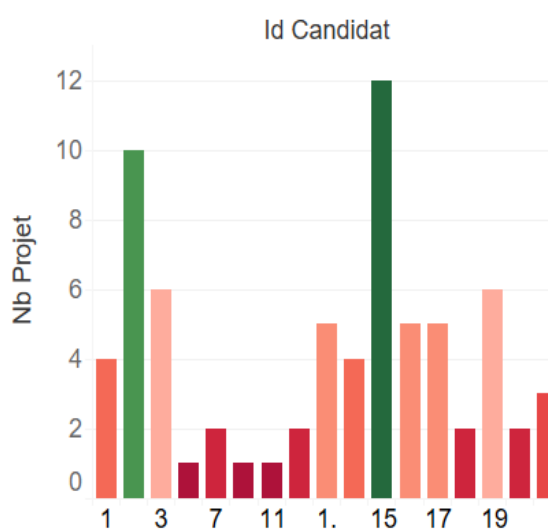
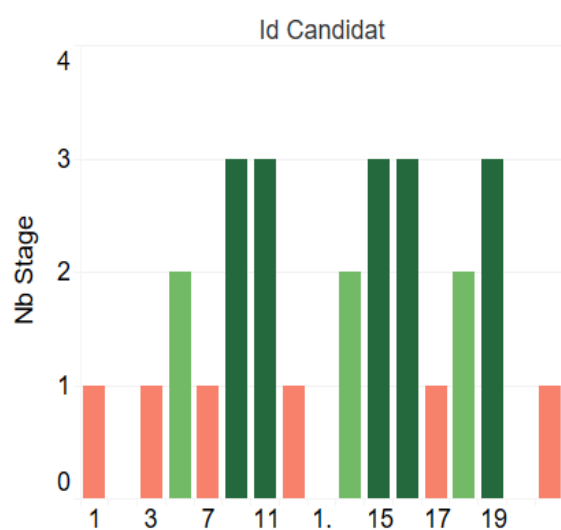
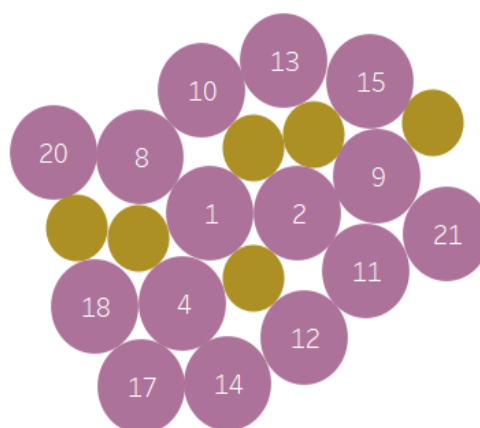
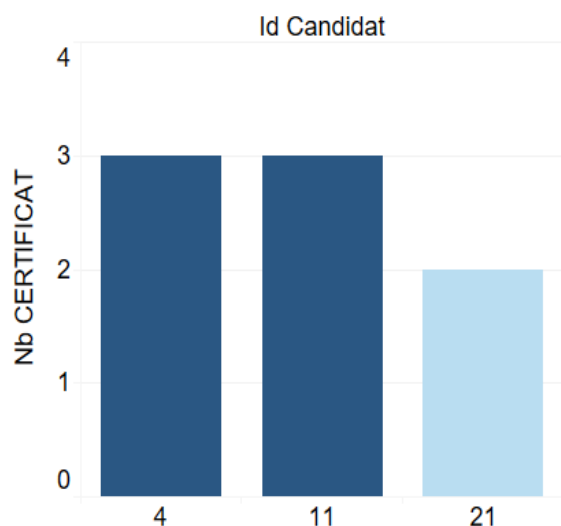
Les compétences de chaque candidat

Id Cand..	Busines..	C	Java	Mongo ..	My SQL	Power BI	PI Sql	Sas	Python	Tableau	Spark	PHP	Talend
1	1	1	1	1	1	0	1	1	1	1	0	0	1
2	1	1	1	1	1	1	1	1	1	1	0	0	1
3	1	1	1	1	1	1	1	1	1	1	0	1	1
4	0	0	1	1	1	0	1	0	1	0	0	0	1
5	0	1	1	1	1	0	1	0	1	0	0	0	1
6	0	1	1	0	1	0	1	0	0	0	0	0	0
7	1	1	1	0	1	0	1	1	1	1	0	1	0
9	1	1	1	1	1	1	1	1	1	1	0	0	1
11	0	0	1	1	1	0	1	0	1	0	1	0	1
12	0	1	1	1	1	0	1	1	1	0	0	0	0
13	0	1	1	0	1	0	0	1	1	0	0	0	0
15	1	1	1	0	1	1	1	1	1	0	0	0	1
16	0	1	1	0	0	0	0	0	1	0	0	1	0
17	1	0	1	1	1	1	1	0	1	1	0	0	1
18	0	1	1	1	1	1	1	1	1	0	0	0	1
19	0	1	1	0	1	0	1	0	1	0	0	0	0
20	0	1	1	1	1	0	1	1	1	0	0	1	0
21	0	1	1	1	1	0	1	1	1	1	1	0	0

On peut voir sur ce deuxième tableau les compétences de chaque candidat (1 si c'est mentionné sur le CV Sinon 0).

Expériences - projets - certificats - langues des candidats

Nb Langues
1 2



Enfin ce Dashboard nous montre avec le diagramme à bulles les langues mentionnées sur les CV (dans notre cas 2 langues : français et anglais ou une seule) ainsi que le nombre de projets, stages réalisés par chaque candidat et le nombre de certificats obtenus avec les trois diagrammes à barres.

5. Conclusion

Ce projet a été une belle expérience pour le groupe, ça nous a permis de renforcer notre capacité de travaillé en groupe et d'aborder un sujet très intéressant, avec plusieurs problématique (structuration des tables, insertions des données, automatisation des scripts ...etc.) en plus en un temps réduit.