|  |  |  |  |
| --- | --- | --- | --- |
|  | |  | | --- | | **Université Moulay Ismail**  **Faculté Des Sciences et Techniques Errachidia** | |  |

Une image contenant texte

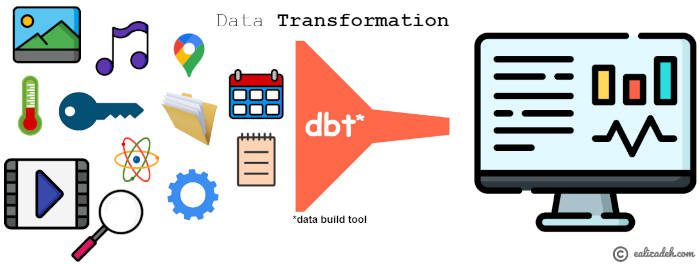
Description générée automatiquement

***Master : Système d’information décisionnel et imagerie***

***Module : Big Data***

**Rapport de Mini projet intitulé :**

***DATA TRANSFORMATION DBT***



|  |  |
| --- | --- |
| **Réalisé par :**   * **SABIRI MOHAMED** | **Encadré par :**   * **Prof. AMINE BENHAMZA** |

**Année Universitaire 2021/2022**

Table des matières

[I. Introduction 3](#_Toc95147064)

[II. Définitions 3](#_Toc95147065)

[III. Informations générales 3](#_Toc95147066)

[IV. Fonctionnalités Principales 4](#_Toc95147067)

[V. Connectivité et integration 5](#_Toc95147068)

[VI. Déploiement 5](#_Toc95147069)

[VII. Les Pratiques 7](#_Toc95147070)

[VIII. Feuille Route 7](#_Toc95147071)

[IX. Références 7](#_Toc95147072)

## **Introduction**

De nos jour la recherche et l’exploitation des outils d’analyse des base données NoSQL ou Big Data constitue un point très importante pour les analystes dans le processus ETL ou ELT (Extraction, Transformation, Chargement de données), alors que dans chaque un de ces opération nous somme dans l’obligation d’intégration des outils pour garder la fiabilité des résultats ainsi que la rapidité de traitement de données. Dans ce sujet je vais aborder l’outils BDT (Data Build Tool) outils utilisé dans l’étape de transformation des données dans les entrepôts de données.

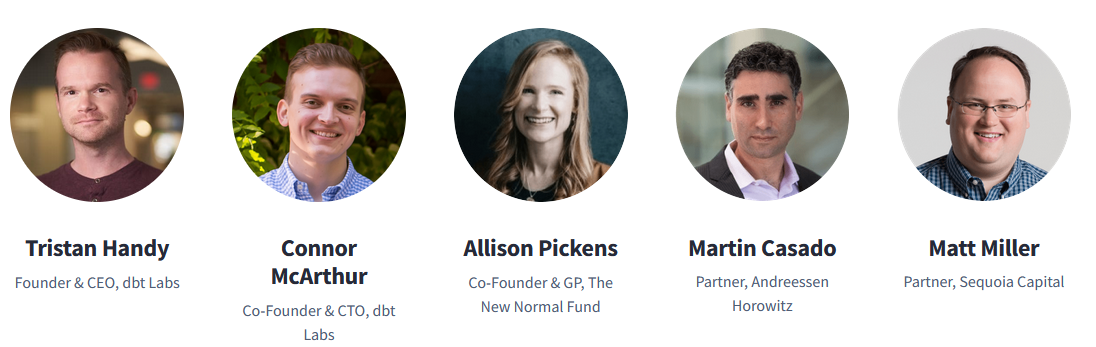
## **Définitions**

* Data Build Tool

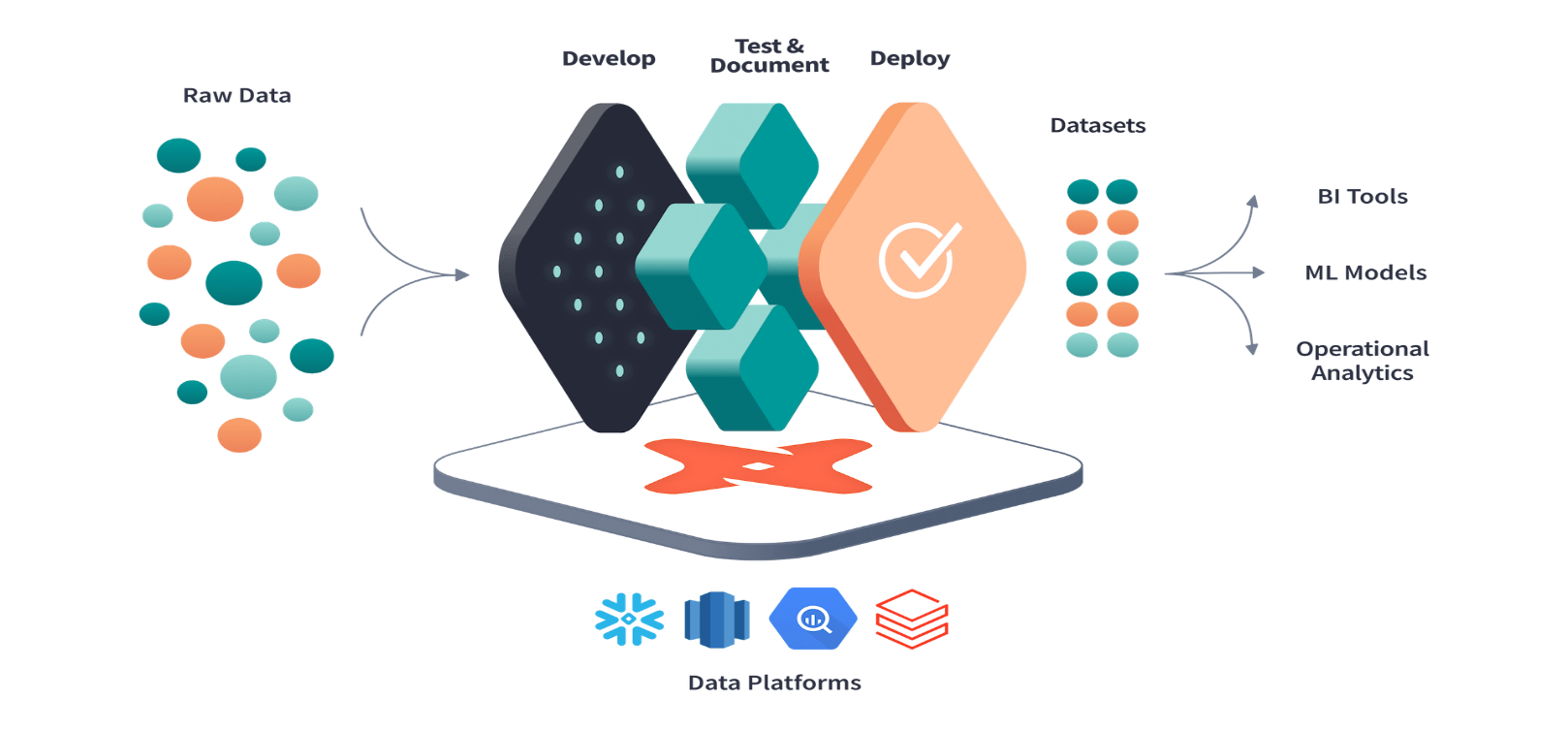
DBT outils permettant les Analytics et les ingénieurs d'analyse de faire la transformation les données dans leurs entrepôts en écrivant simplement des instructions de sélection. il gère la transformation de ces instructions select en tables et vues, aussi effectue les processus T dans ELT(Extract, Load, Transform) par contre il n'extrait ni ne charge de données, mais il est extrêmement efficace pour transformer des données déjà chargées dans les entrepôt.

## **Informations générales**

* **Historique**
* **2016 : commencé chez** [**RJMetrics**](https://en.wikipedia.org/wiki/RJMetrics)
* **2018 : acquis par Talend, avec lancement du 1ere produit commercial**
* **2020 : En avril 2020, dbt Labs a annoncé sa série A dirigée par Andreessen Horowitz, En novembre, dbt Labs a annoncé sa série B dirigée par Andreessen Horowitz et Sequoia.**
* **2021 : dbt Labs a relevé sa série C dirigée par Altimeter, Sequoia et Andreessen Horowitz.**
* **Site Internet**
* **https://docs.getdbt.com/**
* **Dépôt GitHub**
* **https://github.com/dbt-labs/**
* **Taille de la communauté**
* **Octobre 2018 : mise à jour du nombre d'utilisateurs dbt au début du post de 100 à 280**
* **Septembre 2019 : mise à jour du nombre d'utilisateurs dbt au début de la publication de 280 à 850**
* **Mai 2021 : nombre d'utilisateurs dbt mis à jour au début de la publication à 5000**
* **Fondateurs**



## **Fonctionnalités Principales**



Dans le processus ETL le DBT est utilisé pour objectif de faire la transformation de données, et comme illustrer dans le workflow, on distingue les fonctionnalité suivantes qui sont déduite de trois couches constituant le model DBT :

* **Develop : développer**

Écrivez des modèles SQL modulaires avec des instructions SELECT et la fonction ref() - dbt gère la corvée de gestion des dépendances.

* **Test & Document : Tester et documenter**

Testez chaque modèle avant la production et partagez la documentation générée dynamiquement avec toutes les parties prenantes des données.

* **Deploye : Contrôle de version et CI/CD**

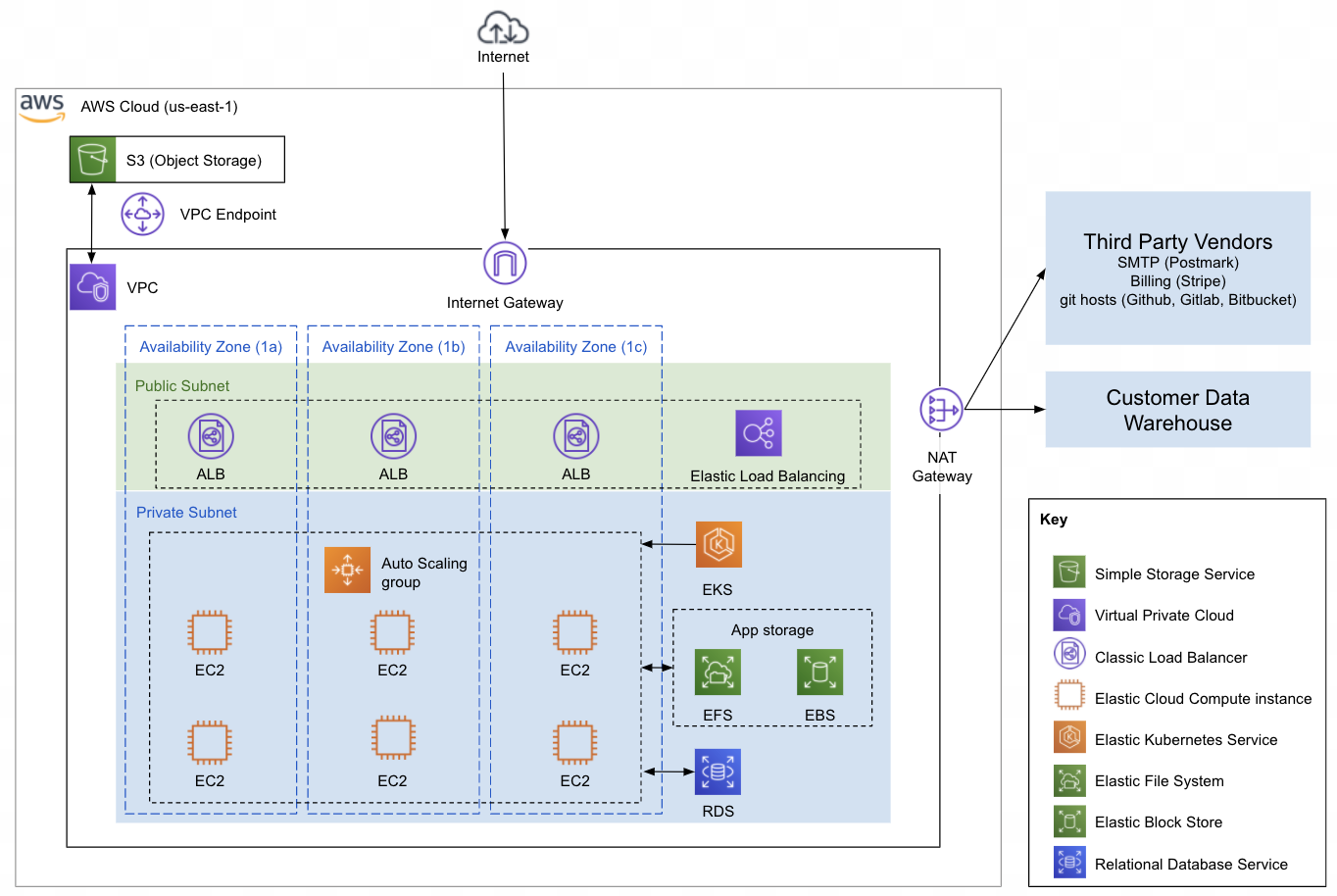
Déployez en toute sécurité à l'aide d'environnements de développement. Le contrôle de version activé par Git permet la collaboration et un retour aux états précédents.

## **Connectivité et integration**

Comme objectif de BDT est de faire la transformation des données dans les bases de données, Entrepôt de données ou dans les plate-forme et clous, son fonctionnement dépond de la possibilité à intégrer avec le type de SGSB des données à transformer, alors que son utilisation fonctionne par utilisation des adaptateur de fonctionnement.

## **Déploiement**

Architecture de déploiement réseau de DBT.



**Selon le schéma précèdent on distingue les sections suivantes :**

* **VPC** : Dans les deux déploiements hébergés, l'infrastructure d'application dbt Cloud réside dans un VPC. AWS géré par dbt Labs. L'une des principales différences entre le déploiement de production et le déploiement à locataire unique est que le déploiement à locataire unique fournit un VPC dédié pour un seul client.
* **EKS**: **Elastic Kubernetes Service** permet de gérer les ressources d'application dbt Cloud. **EKS** offre un degré élevé de fiabilité et d'évolutivité pour l'application dbt Cloud.
* **CLB** : un ou plusieurs équilibreurs de charge classiques AWS vivant dans un sous-réseau public sont exploités dans les environnements de déploiement hébergés pour distribuer le trafic entrant sur plusieurs instances EC2 dans le cluster EKS.
* **EC2** : les déploiements dbt Cloud hébergés exploitent un cluster de nœuds de travail AWS EC2 pour exécuter l'application dbt Cloud.
* **EBS** : Afin de stocker les données d'application, dbt Cloud exploite AWS Elastic Block Store monté sur les instances EC2 décrites ci-dessus.
* **EFS** : Un système de fichiers élastique AWS est provisionné pour les déploiements hébergés afin de stocker et de gérer les fichiers locaux à partir de dbt Cloud IDE.
* **S3** : AWS Simple Storage Service (S3) est utilisé pour stocker les journaux d'application et les artefacts dbt Cloud (tels que ceux générés à partir des exécutions de tâches dbt).
* **RDS** : L'application hébergée dbt Cloud exploite AWS Postgres RDS pour stocker les informations d'application telles que les comptes, les utilisateurs, les environnements, etc. Notez que, comme expliqué dans la section Interaction de l'entrepôt de données ci-dessus, aucune donnée d'un entrepôt associé n'est jamais stockée dans cette base de données.

## **Les Pratiques**

**Les recommandations des bonnes pratiques DBT :**

1. **Utiliser la fonction de références.**
2. **Limiter les références aux données brutes**
3. **Renommer et reformuler les champs une fois**
4. **Décomposez des modèles complexes en plus petits morceaux**
5. **Regroupez vos modèles dans des répertoires**
6. **Ajoutez des tests à vos modèles**
7. **Considérez l'architecture de l'information de votre entrepôt de données**
8. **Utiliser la syntaxe de sélection de modèle lors d'une exécution locale**
9. **Conseils professionnels pour les flux de travail**
10. **N'exécutez que des modèles modifiés pour tester les modifications (« CI mince »)**
11. **Limiter les données traitées en développement**
12. **Utiliser des crochets pour gérer les privilèges sur les objets créés par dbt**
13. **Séparez les transformations centrées sur la source et centrées sur l'entreprise**
14. **Gestion des espaces blancs générés par Jinja**

**Les principaux commande de DBT**

* [**build**](https://docs.getdbt.com/reference/commands/build)**: build and test all selected resources (models, seeds, snapshots, tests)**
* [**clean**](https://docs.getdbt.com/reference/commands/clean) **(CLI only): deletes artifacts present in the dbt project**
* [**compile**](https://docs.getdbt.com/reference/commands/compile)**: compiles (but does not run) the models in a project**
* [**debug**](https://docs.getdbt.com/reference/commands/debug) **(CLI only): debugs dbt connections and projects**
* [**deps**](https://docs.getdbt.com/reference/commands/deps)**: downloads dependencies for a project**
* [**docs**](https://docs.getdbt.com/reference/commands/cmd-docs) **: generates documentation for a project**
* [**init**](https://docs.getdbt.com/reference/commands/init) **(CLI only): initializes a new dbt project**
* [**list**](https://docs.getdbt.com/reference/commands/list) **(CLI only): lists resources defined in a dbt project**
* [**parse**](https://docs.getdbt.com/reference/commands/parse) **(CLI only): parses a project and writes detailed timing info**
* [**run**](https://docs.getdbt.com/reference/commands/run)**: runs the models in a project**
* [**seed**](https://docs.getdbt.com/reference/commands/seed)**: loads CSV files into the database**
* [**snapshot**](https://docs.getdbt.com/reference/commands/snapshot)**: executes "snapshot" jobs defined in a project**
* [**source**](https://docs.getdbt.com/reference/commands/source)**: provides tools for working with source data (including validating that sources are "fresh")**
* [**test**](https://docs.getdbt.com/reference/commands/test)**: executes tests defined in a project**
* [**rpc**](https://docs.getdbt.com/reference/commands/rpc) **(CLI only): runs an RPC server that clients can submit queries to**
* [**run-operation**](https://docs.getdbt.com/reference/commands/run-operation)**: runs arbitrary maintenance SQL against the database**

## **Feuille Route**

## **Références**

**https://docs.getdbt.com/docs/introduction**