Projet 8 : Déployez un Modèle dans le Cloud

Mentor : Amine HADJ-YOUCEF

Etudiant : Marin DUCHEMIN

OPENCLASSROOMS

Plan de la Présentation

I ~ Problématique et Jeu de Données

II ~ Architecture Big Data dans le Cloud

III ~ Chaîne de Traitement des Images

Conclusions et Perspectives

I ~ Problématique

Start-up de l’AgriTech: “Fruits!”

Besoin :

- Moteur de classification de photos de fruits

Fruits!

Objectifs :

- Prétraitement des images

- Réduction de dimensions

I ~ Problématique

Contraintes : Mise à l'Echelle

- Architecture "Big Data"
- Utilisation du Cloud

Solutions Possibles :

- Amazon Web Service (AWS)
- Microsoft Azure
- Google Cloud
- IBM Cloud
- etc

I ~ Jeu de Données

Jeu de données :

- Image de 100 x 100 pixels
- Plus de 67 000 images de fruit
- 131 types de fruit









I ~ Problématique

Obstacles du Projet :

- Grand nombre d’images → espace informatique important
- Solutions gratuites du cloud insuffisantes

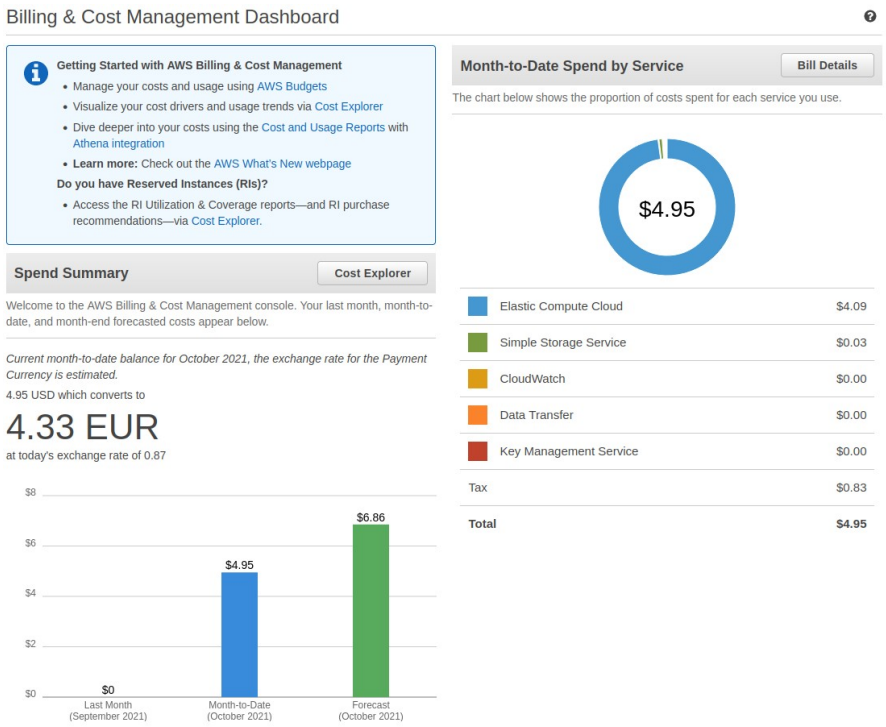
Solutions potentielles:

- Utilisation de techniques “Big Data” (calcul distribué)
- Developpement et tests en local avec lancement sur le cloud

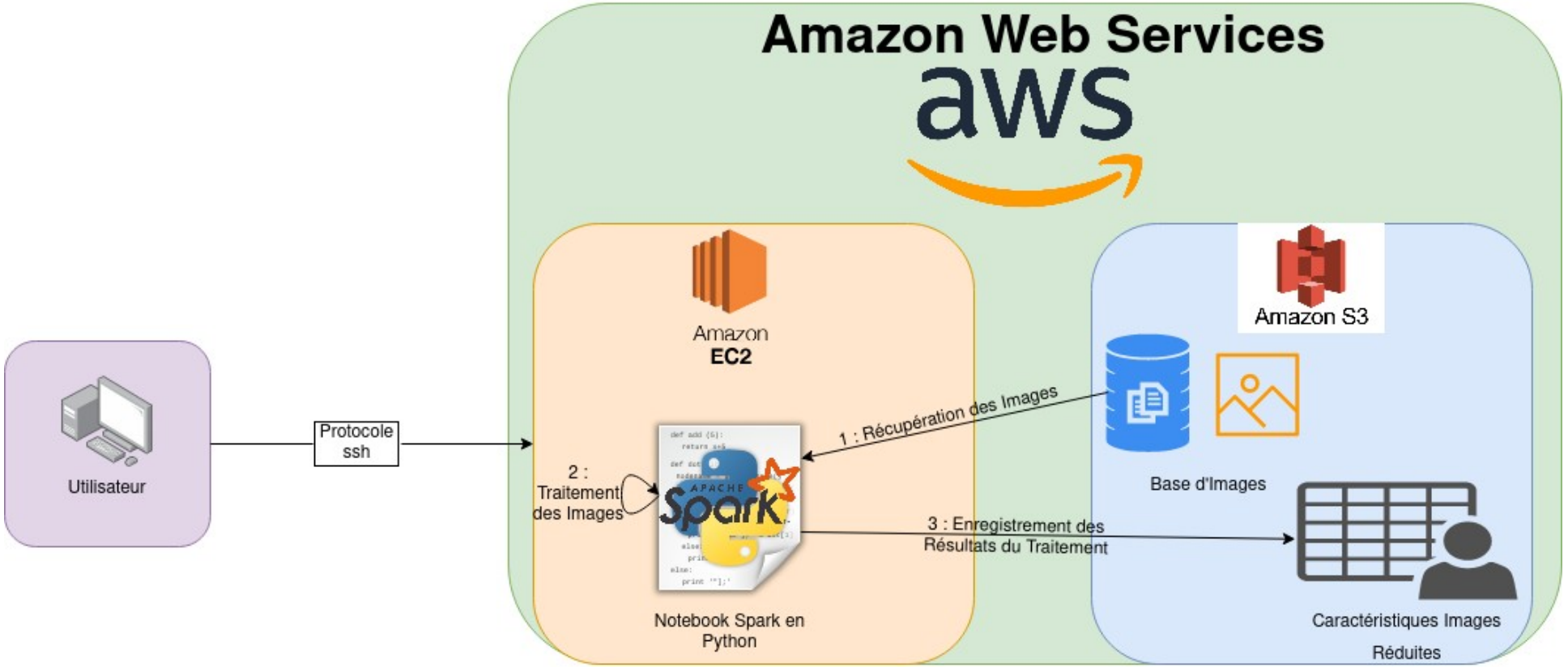
I ~ Limitations personnelles

Réduction du coût d’utilisation :

- Tests en local des nouveaux outils
  - PySpark
  - Boto3
- Nombre limité d’images en ligne
- Temps d’instance minimum

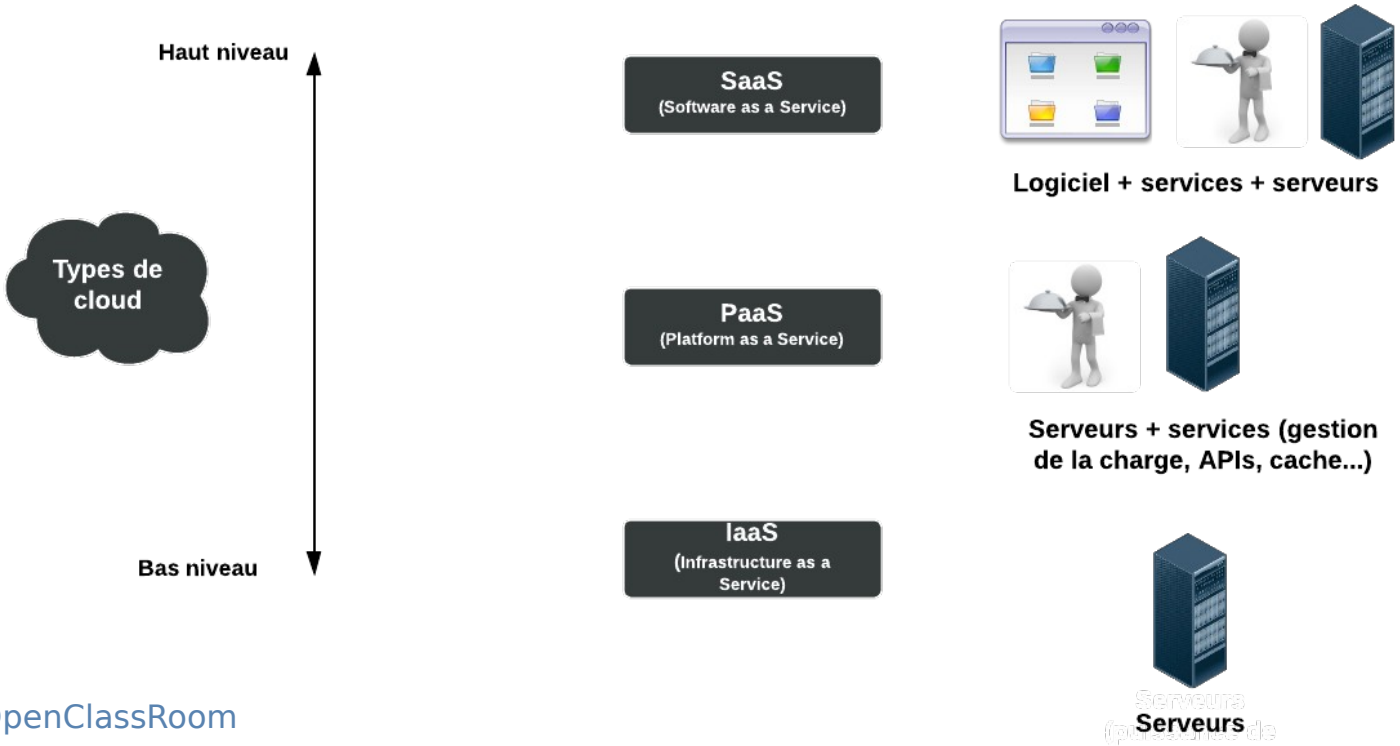


I ~ Solution proposée



II ~ Présentation du Cloud

Définition du Cloud :



Crédit Image : OpenClassRoom

II ~ Serveur de Traitement

Serveur EC2 (Elastic Cloud Computing) de AWS:

- Premier service (historiquement)

- Premier service (utilisation)
  - Configurable (généraliste)
  - Mise en place rapide
  - Ajustable aux besoins (élastique)

Type d'instance	vCPU	Mémoire (Gio)	Stockage (Go)	Performances de mise en réseau	Processeur physique	Fréquence d'horloge (GHz)
t2.nano	1	0,5	EBS uniquement	Faibles	Série Intel Xeon	Jusqu'à 3,3
t2.micro	1	1	EBS uniquement	Faibles à modérées	Série Intel Xeon	Jusqu'à 3.3
t2.small	1	2	EBS uniquement	Faibles à modérées	Série Intel Xeon	Jusqu'à 3.3
t2.medium	2	4	EBS uniquement	Faibles à modérées	Série Intel Xeon	Jusqu'à 3.3
t2.large	2	8	EBS uniquement	Faibles à modérées	Série Intel Xeon	Jusqu'à 3.0
t2.xlarge	4	16	EBS uniquement	Modérées	Série Intel Xeon	Jusqu'à 3.0

II ~ Serveur de Traitement

Lancement d’un serveur EC2 sur AWS :

- Choix d’une zone géographique
  - Latence
  - Client
  - Coût
- Choix d’une instance
  - t2.micro
  - t2.medium
  - etc
- Choix d’une Amazon Machine Image (AMI)
  - Installation OS

II ~ Serveur de Traitement

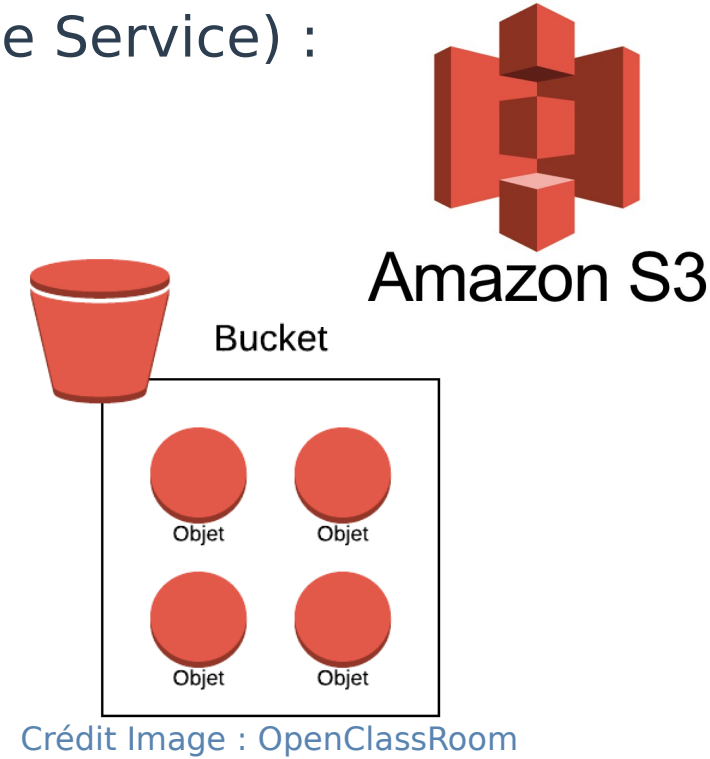
Accès à distance du serveur EC2 :

- Communication via protocole ssh
  - openssh (Linux, Mac)
  - Putty (Windows)
- Installation de l'environnement
  - Python via requirements.txt
- Transfert des fichiers via FileZilla
- Exécution du Notebook via la console terminale

II ~ Serveur de Stockage

Serveur de Stockage S3 (Simple Storage Service) :

- Service de stockage en ligne
- Pseudo gratuit
- Configuration simple des droits d’accès
- Versionning et chiffrage possible



Crédit Image : OpenClassRoom

II ~ Serveur de Stockage

Communication avec S3: Boto3

- Software Development Kit (SDK) de AWS
- En Python
- API orientée-objet

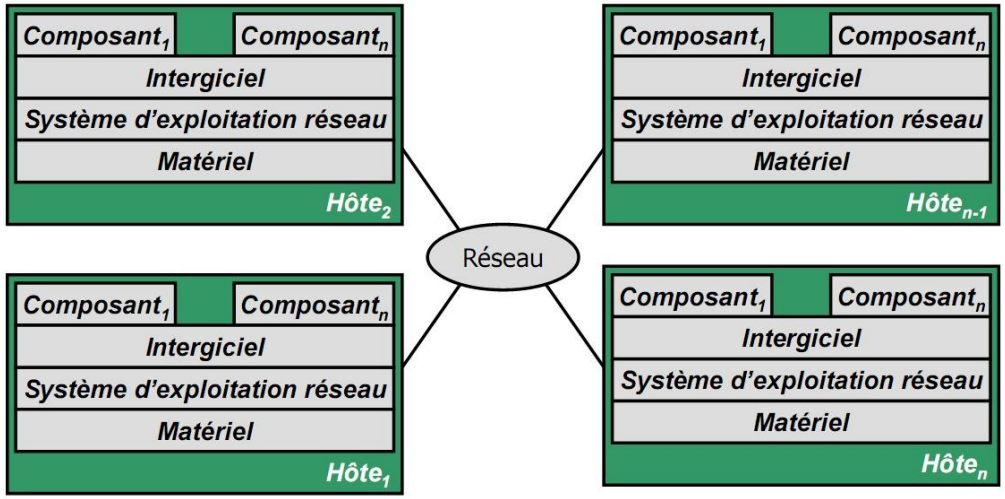
```python
# S3 Variables
access_key = '[redacted]'
secret_key = '[redacted]'
REGION = 'eu-west-1'
bucket_name = 'md-ocr-p8-bucket'

client = boto3.client('s3',
                      aws_access_key_id=access_key,
                      aws_secret_access_key=secret_key
                     )
s3 = boto3.resource(
    's3',
    aws_access_key_id=access_key,
    aws_secret_access_key=secret_key
    )
bucket = s3.Bucket(bucket_name)
```

II ~ Architecture Big Data

Obstacles liés au Big Data

- Volume
- Vélocité
- Variété



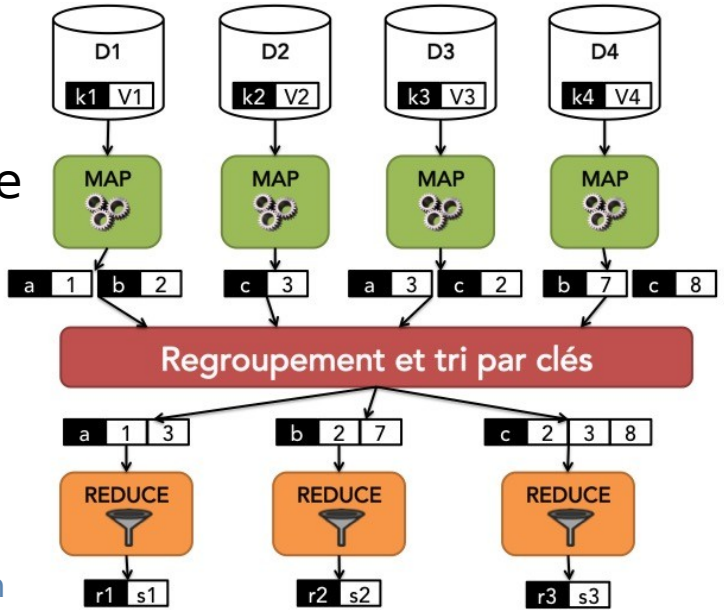
Solution : Calcul Distribué

- Noeuds de calcul autonomes
- Communication inter-noeuds via un cluster
- Passage à l'échelle horizontale plutôt que verticale

II ~ Architecture Big Data

Hadoop :

- Infrastructure logicielle pour le calcul distribué
- Basé sur les opérations Map et Reduce



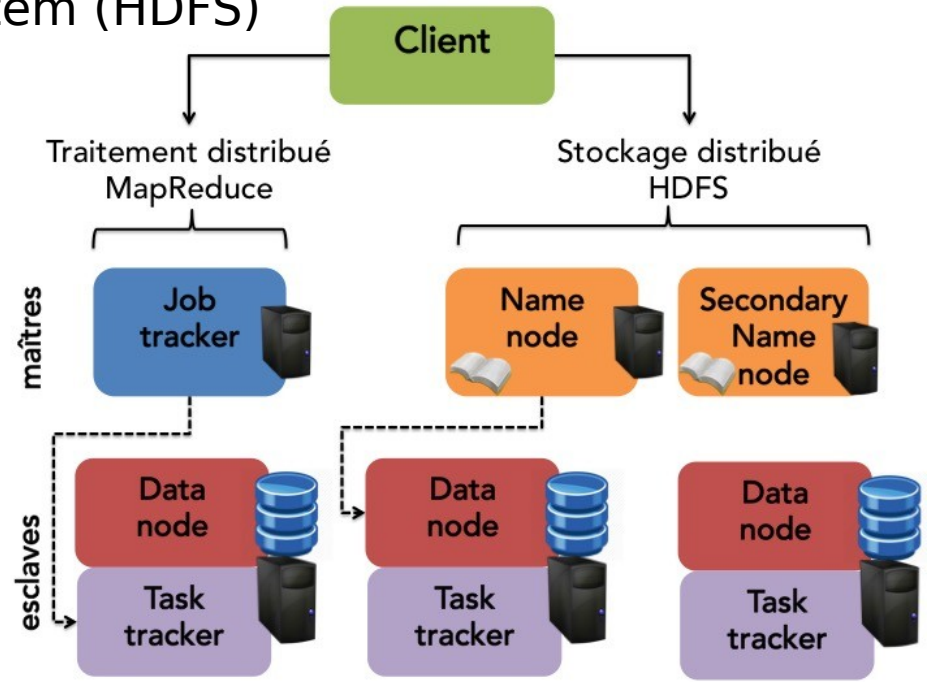
Crédit Image :
OpenClassRoom

II ~ Architecture Big Data

Hadoop :

- Hadoop Distributed File System (HDFS)
  - Distribué
  - Répliqué
  - Optimisé

- Limites :
  - Que deux opérations
  - Temps de traitement

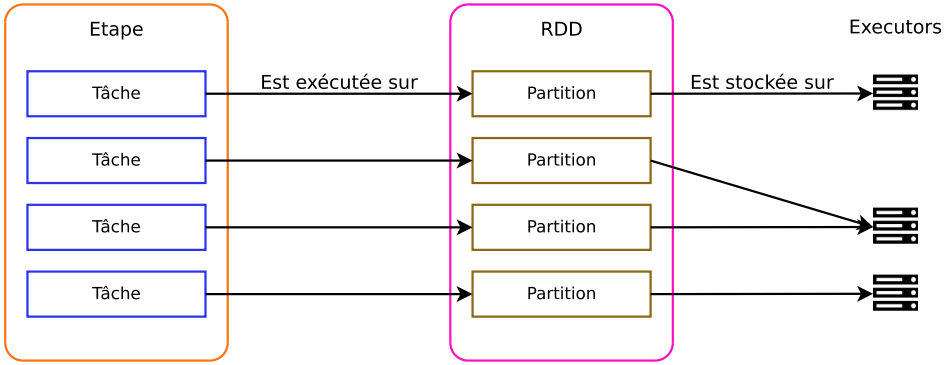


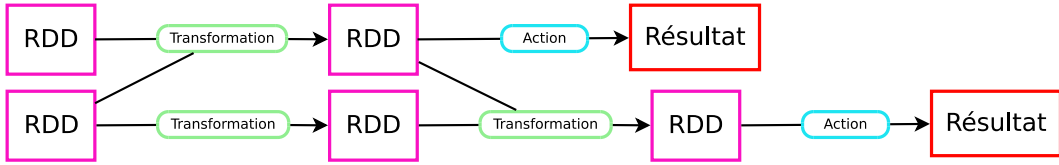
Crédit Image :
OpenClassRoom

II ~ Architecture Big Data

Apache Spark : Hadoop Plus Rapide

- Vitesse augmentée
  - RAM plutôt que DD (Go/s x ~100)
- Resilient Distributed Datasets (RDD)
  - Fonctions ‘lazy’





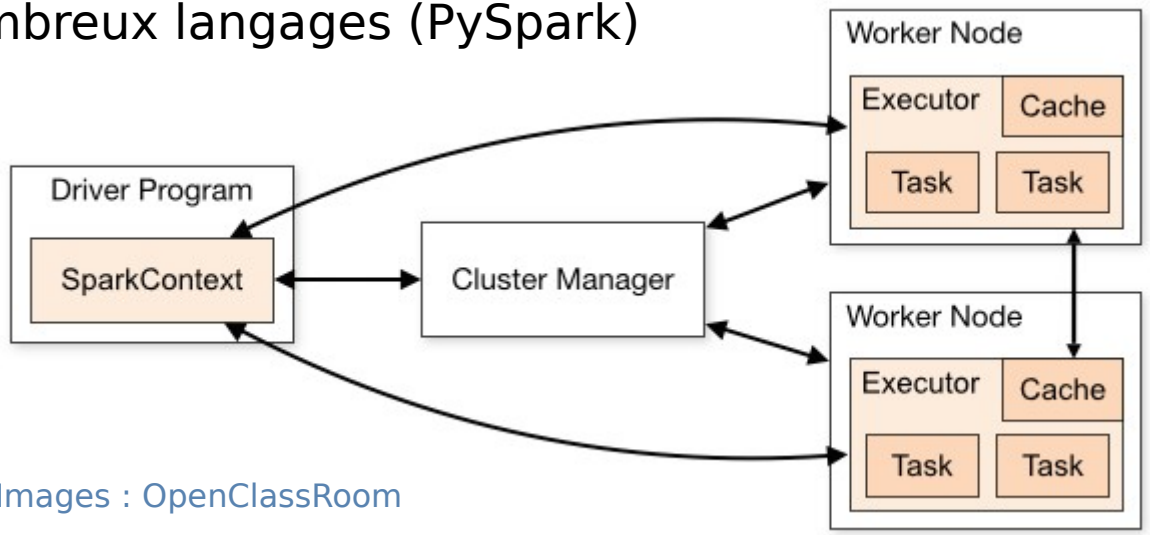
Crédit Images : OpenClassRoom

II ~ Architecture Big Data

Apache Spark : Hadoop Généralisé

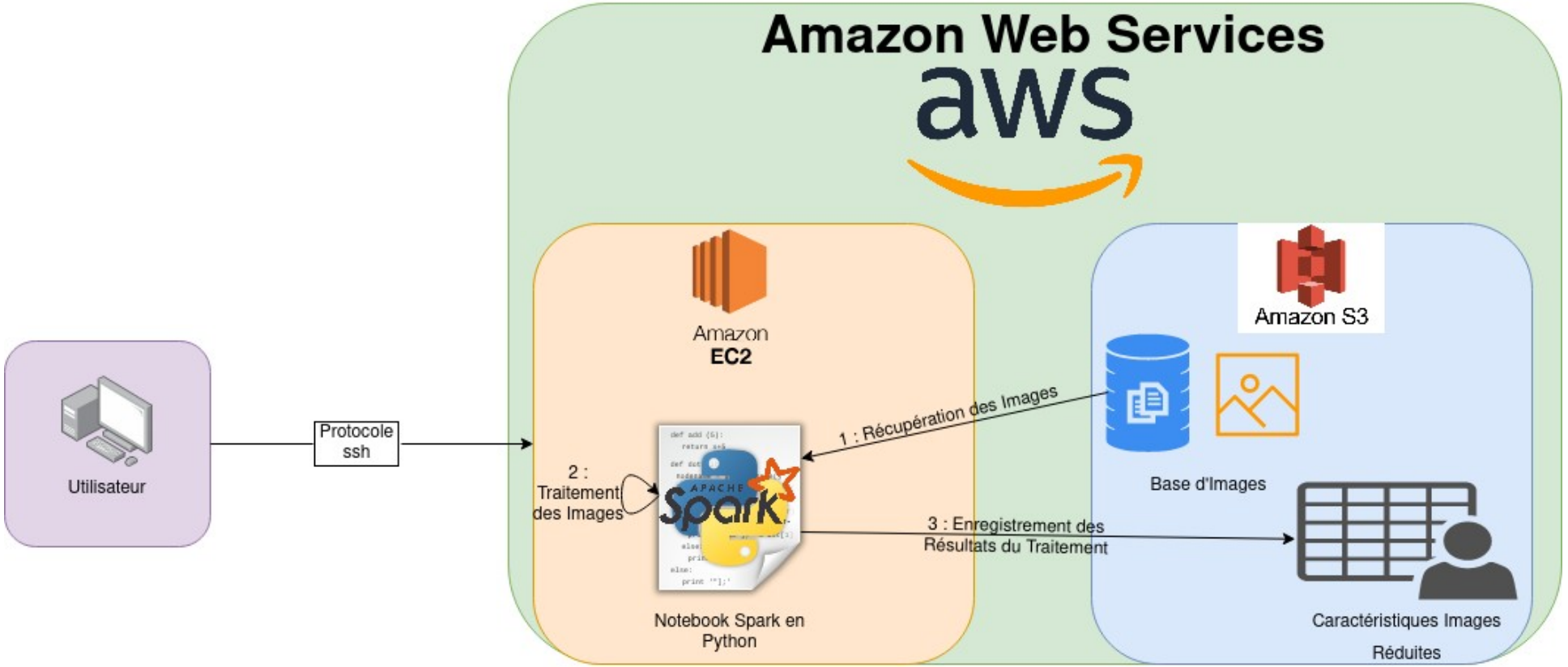
- API adaptés : SQL, ML, Streaming, GraphX, Core

- Nombreux langages (PySpark)



Crédit Images : OpenClassRoom

III ~ Solution proposée



III ~ Chaîne de Traitement des Images

Etape 1 : Récupérer les Images du S3

- Utilisation de boto3

- Difficulté de mise en place via PySpark
  - Incompatibilité boto3 / PySpark
  - Passage par une liste de couple (lien, image)

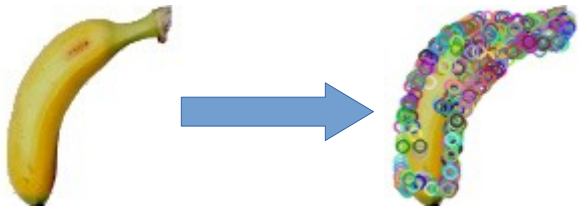
```
item_list = []
for s3_file in bucket.objects.filter(Prefix=image_path):
    item_list.append((s3_file.key, bucket.Object(s3_file.key).get().get('Body').read()))

item_list_rdd = sc.parallelize(item_list)
```

III ~ Chaîne de Traitement des Images

Etape 2 : Extraction des Descripteurs

- Prétraitement de l’image
  - Niveau de gris
  - Exposition
  - Contraste
  - Bruit



- Creation Descripteurs
  - Méthode ORB (openCV)

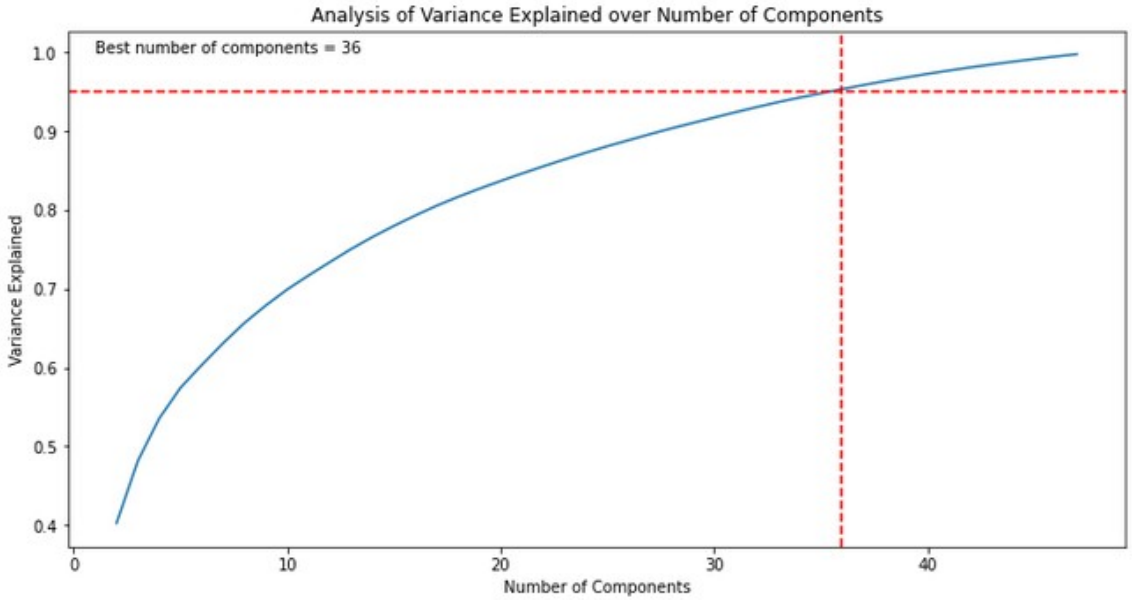
- Sauvegarde dans un dictionnaire
  - Clé : Identifiant de l’image
  - Valeur : Matrice 2D des descripteurs

III ~ Chaîne de Traitement des Images

Etape 3 : Réduction de Dimensions

- Extraction des Features des images
  - Empilement vertical des descripteurs
  - Création de “Bags of Visual Words”
  - Extraction et mise à l’échelle des descripteurs pertinents

- Utilisation d’une ACP
  - Nombre de composantes minimum pour une variance expliquée de 95%
  - Passage de 48 à 36 composantes



III ~ Chaîne de Traitement des Images

Etape 4 : Sauvegarde des Résultats sur le S3

- Difficulté de mise en place via PySpark
  - Incompatibilité boto3 / PySpark
- Enregistrement en local
  - Fichier temporaire
- Sauvegarde permanente du fichier temporaire
  - Via boto3
  - Sur le S3

```
ID_list = des_df.select("ID").rdd.flatMap(lambda x: x).collect()

for i in range(len(ID_list)):
    ID = ID_list[i]
    split_ID = re.compile("_")
    fruit_type = split_ID.split(ID)[-1]
    split_fruit = re.compile(f"_{fruit_type}")
    fruit = split_fruit.split(ID)[0]
    features = X_PCA[i].tolist()
    np.savetxt("temp_reduced_features.csv", features, delimiter=',')
    s3_path = f"Ressources/Reduced Features/{fruit_type}/{fruit}_reduced_features.csv"
    client.upload_file("temp_reduced_features.csv", bucket_name, s3_path)
```

Conclusion et Perspectives

Conclusion : Chaine de Traitement Big Data Réussie

- Découverte du calcul distribué (PySpark)
- Déploiement sur AWS
  - EC2
  - S3
- Connexion à distance (ssh)
- Limites :
  - Temps de traitement relativement long
    - 365s pour 2316 images réparties dans 5 types de fruits
  - Surtout dû à la communication avec le S3
    - 120s pour récupérer les données
    - 180s pour sauvegarder les résultats
    - 65s de traitement

Conclusion et Perspectives

Axes de Poursuite :

- Utiliser des serveurs plus puissants
- Utiliser des réseaux de neurones pour extraire les “features”
- Trouver un système de sauvegarde différent
  - Un seul fichier
  - Logiciel parquet
  - etc
- Utiliser Databricks directement plutôt que AWS
- Utiliser le serveur pour faire un cluster Spark
  - Utiliser un scrypt plutôt qu’un notebook
  - Interroger à distance avec des requêtes

Merci de votre attention

Réserve

Protocole ssh :

- Connexion au serveur

```
ssh -i "/home/muninn/.ssh/OCR_P8.pem" ec2-user@ec2-54-228-168-107.eu-west-1.compute.amazonaws.com
```

- Connexion au Jupyter Lab

```
ssh -i "/home/muninn/.ssh/OCR_P8.pem" -CNL localhost:2412:localhost:2412 ec2-user@54.228.168.107
```

Réserve

Méthode ORB :

- Oriented FAST and rotated BRIEF