UNIVERSITE GALATASARAY
FACULTÉ D'INGÉNIERIE ET DE TECHNOLOGIE

CONSTRUCTION D'UN PIPELINE DE TRAITEMENT DE DONNÉES POUR LES DONNÉES DE DIFFÉRENTS FORMATS

(FARKLI FORMATTAKİ VERİLER İÇİN BİR VERİ İŞLEME BORU HATTI OLUŞTURMA)

PROJET DE FIN D'ETUDES

Doğa Yağmur YILMAZ

Département : GENIE INFORMATIQUE
Directrice du projet de fin d'études : DR. Sultan Nezihe TURHAN

JUIN 2023

PRÉFACE

Je tiens à remercier ma chère professeure Sultan N. Turhan qui a contribué à la formation de mon projet de fin d’études, et Kariyer.net qui m'a donné l'opportunité de préparer un projet de TUBITAK en utilisant la technique du pipeline créée dans ce projet.

Doğa Yağmur Yılmaz

Juin, 2023

TABLE DES METIÈRES

LISTE DES NOTATIONS ET DES TERMES

ETL: Extract Transform Load

ELT: Extract Load Transform

Data Lake : Le Lac de Donnée

Data Warehouse : L’Entrepôt de Données

Business Intelligence : L’Intelligence d'Affaires

Cloud Database : Base de Données en Nuage

Batch Processing : Traitement par lots

KVKK : Loi sur la protection des données personnelles

Yök : le Conseil de l'enseignement supérieur de la Turquie

ORC (Optimized Row Columnar) : Lignes optimisées Colonnes

RESUME

L'objectif du projet est de réaliser l'intégration des données en créant un pipeline de traitement des données pour des données de différents formats qui proviennent de différentes sources.

Dans le cadre du projet, Il est créé une base de données orientée colonnes d'académiciens, sur laquelle des requêtes peuvent être exécutées, en extrayant et en intégrant les données des académiciens à partir de deux sources différentes.

Les données ont été collectées à partir de deux sources différentes. La première source est "Yök Académique", qui contient les informations académiques et de contact des académiciens. Les données de Yök Académique ont été extraites à l'aide d'un code de scrape écrit manuellement en utilisant la librairie nodejs pupeteer. La raison de l'utilisation de Javascript dans le code de scrape est que la page Yök Académique est rendue de manière dynamique, c'est-à-dire que les nouvelles informations s'ouvrent en cliquant quelque part. Afin d'imiter cette structure et d'interagir avec l'interface, l'utilisation de Javascript est obligatoire. En même temps, la page de Yök académique a un bouton pour télécharger les données listées sur la page en tant qu'un tableau html. Dans le même code de scrape, les données sauvegardées ont été converties au format csv/excel afin de rendre les données extraites utilisables. Comme deuxième source, Google scholar a été utilisé pour récupérer des articles et des domaines d'expertise spécialisés d'académiciens. Les données de Google scholar ont été extraites à l'aide d'un code de scrape écrit en Python en utilisant les librairies Scholarly et Scrapy. La librairie Scholarly est une librairie de Python qui permet d'extraire des données de Google scholar. C'est pourquoi le langage Python a été préféré. Les titres des articles et les domaines de spécialisation de tous les académiciens ont été extraits et sauvegardés sous forme de fichiers json.

Le fait que les données soient collectées dans différents formats s'explique par le fait qu'elles sont téléchargées et sauvegardées dans le format html de type Excel fourni par la plateforme Yök Académique à l'aide d'un bouton. Google Scholar ne dispose pas d'une solution propre pour télécharger les données. De plus, l'adresse IP qui demande à plusieurs reprises l'accès à la page est bloquée. Pour cette raison, les données accessibles au public sur Google scholar ont été obtenues par scraping avec plusieurs machines différentes à des intervalles de temps et sauvegardées au format json.

Les données dans différents formats ont été téléchargées vers des systèmes de base de données en nuage "Azure Blob Storage, CosmosDB et MongoDB" afin d'être stockées dans une zone unique en tenant compte des facteurs de coût, d'accès et d'évolutivité.

Le "Blob Storage", qui est une architecture de "lac de données" acceptant des données de différents formats sous leur forme brute (raw), a été choisi comme unité de stockage en nuage. En même temps, grâce à son intégration avec Data Factory, on a pensé qu'un pipeline pourrait être créé au sein des systèmes Azure pour la transformation et l'analyse des données.

En installant Data Factory sur le stockage Blob, deux fichiers de formats différents ont été collectés dans un seul fichier json à l'aide d'un code de python et les données ont été converties au format de données "parquet" et "ORC" à l'aide de Data Factory afin d'exécuter des requêtes sur les données de manière efficace. Les données converties à l'aide de Data Factory ont été réinscrites dans le stockage Blob.

Dans l'étape suivante, bien qu'il ait été souhaité d'utiliser Spark pour analyser les données, cette méthode n'a pas été préférée en raison des contraintes de coût et de compte étudiant créées sur Azure.

Si l'on s'en tient à l'idée d'analyser les données à l'aide de Spark, d'autres moyens d'accéder à Spark par le cloud ont été étudiés. Il a été constaté que Google Cloud peut extraire des données de Blob Storage et fournir un accès Spark via dataproc à l'aide d'une api. En outre, il a été constaté que Spark dispose d'une

connexion au service BigQuery, où les données volumineuses peuvent être interrogées dans un flux. À ce stade, étant donné qu'il serait plus efficace en termes de performances d'exécuter des requêtes via BigQuery en éliminant Spark, c'est cette voie qui a été suivie.

Comme il a été observé que les données écrites au format parquet sur Blob Storage étaient corrompues, les données au format ORC ont été extraites de Blob Storage vers Google BigQuery. La requête "select * from table" a été exécutée sur les données extraites et il a été observé que la sortie tabulaire donnait des résultats corrects. Une autre fonctionnalité offerte par BigQuery dans la section Query Results est l'affichage des résultats au format json. À ce stade, le fichier au format Json a été téléchargé et comparé au fichier json intégré avant la conversion et il a été observé que les données n'étaient pas corrompues.

Par conséquent, le rapport présente un pipeline créé sur des systèmes en nuage.

ÖZET

Proje kapsamında bir akademisyen veri tabanı oluşturmak amaçlanmaktadır. Bu amaç doğrultusunda iki farklı kaynak üzerinden akademisyenlere ait veriler çekilip birleştirilerek üzerinde sorgu çalıştırılabilir sütun odaklı bir veri tabanı oluşturulacaktır.

İki farklı kaynaktan veriler toplanmıştır. İlk kaynak olarak akademisyenlerin akademik ve iletişim bilgilerinin yer aldığı "Yök Akademik" kullanılmıştır. Yök akademik'ten veriler nodejs puppeteer kütüphanesi kullanılarak manual olarak yazılmış bir scrape kodu yardımıyla çekilmiştir. Scrape kodunda Javascript kullanılmasının nedeni Yök Akademik sayfasının dinamik olarak render'lanması yani bir yerlere tıklanarak yeni bilgilerin açılmasıdır. Bu yapıyı taklit edebilmek ve arayüzle etkileşebilmek için javascript kullanımı zorunludur. Aynı zamanda yök akademik sayfa sayfa listelenen verileri html table olarak indirmek için bir butona sahiptir ve indirilen veriler excel görünümlü html table formatındadır. Aynı scrape kodu içerisinde çekilen verinin kullanılabilir hale gelmesi için kaydedilen verinin csv/excel formatına dönüşümü sağlanmıştır. İkinci kaynak olarak akademisyenlere ait makaleleri ve özelleşmiş uzmanlık alanlarını çekmek için Google scholar kullanılmıştır. Google scholar üzerinden veriler Scholarly ve Scrapy kütüphanesi kullanılarak Python ile yazılan bir scrape koduyla çekilmiştir. Scholarly kütüphanesi Google scholar'dan scrape yapmaya yarayan bir Python kütüphanesidir. Bu nedenle dil olarak Python kullanımı tercih edilmiştir. Tüm akademisyenlerin makale başlıkları ve uzmanlık alanları çekilerek json dosyası halinde kaydedilmiştir.

Verilerin farklı formatta toplanmasının sebebi yök akademik platformunun buton yardımıyla kendi sağladığı excel görünümlü html table formatında indirilip kaydedilmesidir. Google scholar'ın veriyi indirmek için kendisinin sağladığı bir çözüm bulunmamaktadır. Üstelik art arda istek atan ip adresinin sayfaya erişimi engellenmektedir. Bu nedenle Google scholar üzerindeki public erişimli veri birkaç farklı makine ile zaman aralıklı olarak scrape yoluyla elde edilmiş ve json formatında kaydedilmiştir.

Elde edilen farklı formattaki veriler maliyet, erişim ve ölçeklenebilirlik etkenleri göz önünde bulundurularak tek bir alanda depolanmak için bulut veri tabanı sistemlerine "Azure Blob Storage, CosmosDB ve MongoDB" yüklenmiştir.

Veri akıtılan bulut sistemleri içerisinden farklı formattaki datayı ham haliyle kabul eden bir "veri gölü" mimarisi olan "Blob Storage" bulut depolama birimi olarak seçilmiştir. Aynı zamanda Data Factory ile entegrasyonu sayesinde veri dönüşümü ve analiz için azure sistemleri içerisinde bir pipeline oluşturulabileceği düşünülmüştür.

Blob storage üzerine Data Factory kurularak farklı formattaki iki dosyanın bir Python kodu yardımıyla tek json dosyasında toplanması ve veri üzerinde verimli bir şekilde sorgu çalıştırılabilmesi için verinin "parquet" ve "ORC" veri formatına dönüşümü sağlanmıştır. Data Factory yardımıyla dönüştürülen veri tekrar Blob Storage üzerine yazılmıştır.

Sonraki aşamada veri üzerinde analiz yapabilmek için spark'tan faydalanmak istense de Azure üzerinde yaratılan maliyet ve öğrenci hesabı kısıtları sebebiyle bu yöntem tercih edilememiştir.

Spark üzerinden analiz fikrine sadık kalınarak başka yollarla Cloud üzerinden Spark erişimi araştırılmıştır. Google Cloud'un Blob Storage'dan veri çekebildiği ve api yardımıyla dataproc üzerinden Spark erşimi sağladığı görülmüştür. Buna ek olarak büyük verinin akış halinde sorgulanabildiği BigQuery hizmetiyle de Spark'ın bağlantısı olduğu bilgisine erişilmiştir. Bu noktada aradan Spark'ı çıkartarak BigQuery üzerinden sorgu çalıştırmak performans açısından verim sağlayacağı için bu yol izlenmiştir.

Blob Storage üzerine parquet formatıyla yazılan verinin bozulduğu gözlemlendiğinden ORC formatındaki veri Blob Storage'tan Google BigQuery'e çekilmiştir. Çekilen veri üzerinde "select * from table" sorgusu çalıştırılarak tablo şeklindeki çıktının düzgün sonuçlar verdiği görülmüştür. BigQuery'nin Query Rresults

bölümünde sunduğu bir başka özellik çıktıların json formatında gösterilmesidir. Bu noktada Json formatındaki dosya indirilerek dönüşümden önceki birleştirilmiş json dosyası ile karşılaştırılmış ve verinin bozulmadığı gözlemlenmiştir.

Sonuç olarak raporda bulut sistemleri üzerinden oluşturulmuş bir pipeline sunulmaktadır.

EXPLICATION DE LA SUJET ET DE L'EFFET DU TRAVAIL

L'objectif du projet est de réaliser l'intégration des données en créant un pipeline de traitement des données pour des données de différents formats qui proviennent de différentes sources. Les données sous différents formats collectées à partir de différentes sources seront stockées sous une forme semi-structurée en coulant dans une zone unique au cours de la première étape. Elles seront ensuite stockées dans une source de données centralisée en deux étapes. En premier étape, les données seront stockées dans un "Lac " et en deuxième étape, un "Entrepôt de Données" sera modélisé et data seraient collectés à cet endroit. De cette manière, des données propres et organisées, qui constituent la partie la plus importante de chaque projet, seront fournies et prêtes à être utilisées dans les processus d'analyse et d'Intelligence d'Affaires.

Dans le cadre du projet, diverses sources seront utilisées pour collecter des informations sur les académiciens en effectuant le traitement par lots, afin de constituer une base de données des académiciens en Turquie. Tout d'abord, les informations "nom, prénom, ORCID, ID de chercheur, titre, université, position de cadre, domaine principal, domaine scientifique, spécialités et e-mail" des académiciens en Turquie seront collectées à partir de la plateforme "Yök Académique". Ensuite, les articles des académiciens seront extraits de "Google Scholar" pour réduire leurs domaines d'expertise à des titres plus spécifiques. Les domaines de travail seront déterminés plus précisément dans les titres d'articles extraits, ce qui permettra d'accéder à des informations sur des domaines d'expertise exacts. Les correspondances entre les académiciens sur "Yök Académique" et "Google Scholar" seront réalisées sans confusion grâce aux identifiants fournis par "Google Scholar" pour chaque faculté et département de chaque université.

Le but donc est de faciliter l'accès à des données précises et fiables sur les académiciens en Turquie, ce qui pourrait être utile pour les recherches et les analyses. Le traitement par lots permettra également de traiter un grand nombre de données rapidement et efficacement. Ce pipeline facilitera donc la collecte, le stockage et la gestion de données académiques en Turquie.

MÉTHODES ET TECHNOLOGIES À UTILISER

BATCH PROCESSING

Le batch processing est un processus de traitement en lot dans lequel une grande quantité de données est traitée en une seule fois. Dans le batch processing, les données sont traitées en blocs et les résultats sont stockés pour être examinés ultérieurement. Il s'agit d'un flux de travail composé d'une série de tâches interconnectées.

LAC DE DONNÉE

Il s'agit d'une zone de stockage où des données de différents formats provenant de différentes sources sont stockées ensemble dans leur forme originale. Le stockage des données s'effectue par le processus ELT. Lorsque les données sont prêtes à être utilisées, le processus de transformation est exécuté. L'état des données avant qu'elles ne soient structurées est accessible via le lac de données.

ENTREPÔT DE DONNÉE

Les données sous différents formats provenant de différentes sources sont transformées et stockées dans une source de données centrale contenant des données qualitatives et quantitatives. L'entrepôt de données offre une grande facilité d'utilisation et d'analyse des données volumineuses (big data). Il fournit des informations structurées et les organise dans des schémas définis pour les besoins de l'entrepôt de données. Les données sont stockées en exécutant le processus ETL.

GOOGLE BIG QUERY

BigQuery est le puissant service de base de données analytique en nuage de Google, conçu pour les plus grands ensembles de données. Il permet aux utilisateurs d'exécuter en quelques secondes des requêtes rapides, de type SQL. BigQuery est un entrepôt de données d'entreprise entièrement géré qui aide à gérer et à analyser des données grâce à des fonctionnalités intégrées. L'architecture sans serveur de BigQuery permet d'utiliser des requêtes SQL pour répondre aux plus grandes questions de votre entreprise sans aucune gestion de l'infrastructure. L'une des principales caractéristiques de l'architecture de BigQuery est la séparation du stockage et du calcul. Cela permet à BigQuery de faire évoluer le stockage et le calcul de manière indépendante, en fonction de la demande. Les requêtes fédérées permettent de lire des données provenant de sources externes, tandis que le streaming prend en charge les mises à jour continues des données.

CLOUD DATABASE SYSTÈMES

Une base de données en nuage est une base de données qui est hébergée dans un environnement de cloud computing. Cela signifie que la base de données est stockée et gérée à distance à partir d'un centre de données distant, plutôt que d'être hébergée localement sur le matériel informatique de l'entreprise. Les bases de données dans le cloud peuvent être configurées pour fonctionner dans des environnements de cloud public ou hybride, et sont souvent proposées sous forme de services gérés ou d'instances de machine virtuelle cloud.

MONGO DB

MongoDB est une application de base de données NoSQL qui utilise le système de base de données relationnel, basée sur des documents et open source. MongoDB stocke les données dans des documents basés sur BSON, qui est un format de données de type JSON, ses champs sémantiques varient d'un document à l'autre et la structure des données peut changer au fil du temps.

AZURE BLOB STORAGE

Azure Blob Storage est une solution de stockage de données volumineuses en différents formats proposée sur la plateforme de cloud computing Microsoft Azure. Azure Blob Storage est intégré à la services d'Azure tels que Data Factory, Spark et Databricks.

AZURE COSMOSDB

Azure Cosmos DB est un service de base de données cloud de Microsoft qui prend en charge plusieurs modèles de données et offre une évolutivité horizontale élastique, une distribution mondiale, une faible latence et des garanties de haute disponibilité. Azure Cosmos DB prend en charge plusieurs modèles de données, dont le modèle de données de documents, de graphiques, de clés-valeurs et de colonnes. Le modèle de données de documents est le plus couramment utilisé et stocke les données sous forme de documents JSON, offrant une grande flexibilité dans la définition de la structure des données. En outre, Azure Cosmos DB prend également en charge plusieurs API -notamment SQL, MongoDB, Cassandra- pour offrir une compatibilité avec les applications existantes et les compétences en matière de développement de base de données. CosmosDB est intégré à d'autres services Azure tels que Azure Data Factory, Azure Databricks et Azure Synapse Analytics, permettant ainsi une intégration facile avec l'ensemble de la plateforme Azure pour des analyses plus avancées.

RECHERCHE DE NORMES

KVKK

En Turquie, la loi KVKK est la principale norme à respecter lors de l'extraction de données, y compris les données publiques. Selon la loi KVKK, les données personnelles ne peuvent être collectées qu'avec le

consentement explicite de la personne concernée. Les organisations doivent également informer les personnes concernées de la finalité du traitement des données et des tiers auxquels les données peuvent être transférées. Les organisations doivent également prendre des mesures pour protéger les données personnelles contre toute perte, vol, accès non autorisé, divulgation ou destruction.

Étant donné que nous ne collectons aucune donnée qui n'a pas été rendue publique dans le cadre du projet, nous agissons conformément à cette loi.

CALENDRIER DE TÂCHES À EFFECTUER

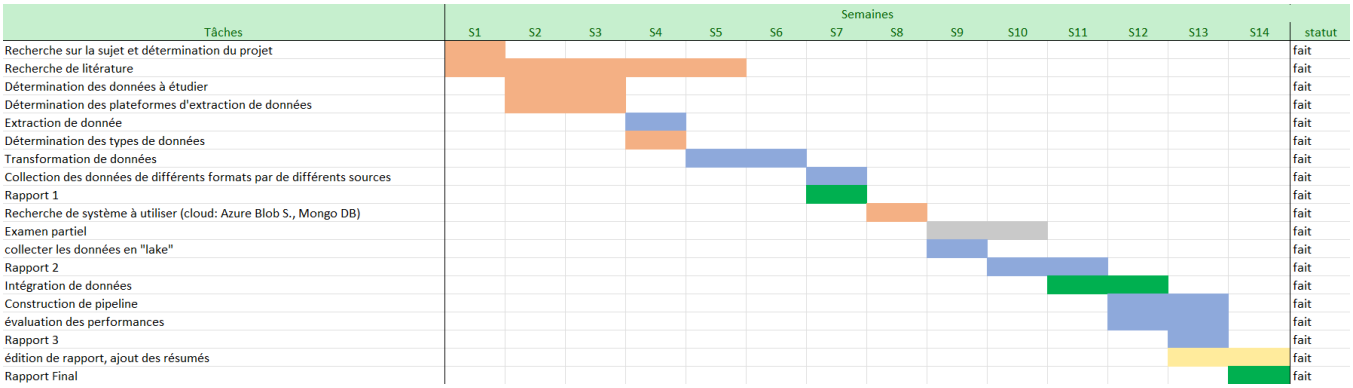


Image 1. Calendrier de tâches

REVUE DE LITTÉRATURE

1. An Overview of Current Trends in Data Ingestion and Integration

La collecte et l'intégration de données constituent un facteur important dans l'analyse de données, notamment avec l'émergence de données volumineuses et à haute vélocité. Les approches existantes, axées principalement sur « le traitement par lots », ont dû être adaptées pour répondre aux nouvelles tendances en matière de traitement et de stockage de données. En même temps, les outils d'ingestion de données ont gagné en popularité car ils ont soutenu de nouvelles tendances de développement et d'analyse. De plus, les principaux fournisseurs ont déplacé leur attention vers les solutions « cloud », ce qui a affecté à la fois les outils existants et émergents. Dans cet article, les outils d'ingestion et d'intégration de données de pointe sont examinés de manière systématique et individuelle, avec identification de leurs caractéristiques clés. En outre, les catégories d'outils actuelles sont observées et comparées, ainsi que la liste des outils qu'ils englobent, en fonction de leurs caractéristiques. [1]

2. TOSCAdata: Modeling data pipeline applications in TOSCA

Les outils modernes de gestion de données cloud ne sont pas suffisamment matures pour intégrer les applications cloud génériques avec la plateforme sans serveur en raison du manque de normes matures et stables. Pour résoudre ce problème, les auteurs proposent une extension de la norme TOSCA (Topology and Orchestration Specification for Cloud Applications), appelée TOSCAdata, qui se concentre sur la modélisation des applications cloud basées sur les pipelines de données. TOSCAdata fournit un ensemble de modèles TOSCA indépendamment déployables, planifiables, évolutifs et réutilisables qui gèrent efficacement le flux et la transformation des données de manière linéaire. Les auteurs démontrent l'applicabilité des modèles TOSCAdata proposés en prenant une application cloud basée sur le Web dans le contexte de la promotion du tourisme comme cas d'utilisation. TOSCAdata fournit une solution unifiée à la communauté de développement de logiciels qui fonctionne avec une solution de gestion de données open-source (Apache NiFi) et une solution commerciale (AWS Datapipeline), qui peut être étendue pour prendre en charge un plus

grand nombre de plates-formes de gestion de données commerciales. Les auteurs prévoient également d'améliorer la scalabilité et le niveau de sécurité des données, ainsi que la performance des applications cloud basées sur les pipelines de données TOSCA.[2]

3. Implementing Big Data Lake for Heterogeneous Data Sources

Les villes modernes connectées utilisent de plus en plus les avancées des TIC pour améliorer leurs services et la qualité de vie de leurs habitants. Cependant, la collecte, l'intégration et l'analyse de toutes les sources de données hétérogènes disponibles dans les villes sont un défi. Cet article suggère une approche de « data lake » construite sur des technologies de Big Data pour rassembler toutes les données en vue d'une analyse ultérieure. La plateforme décrite ici permet la collecte, le stockage, l'intégration et l'analyse ultérieure des résultats. Cette solution est la première tentative d'intégration d'un ensemble diversifié de sources de données provenant de quatre villes pilotes dans le cadre du projet CUTLER (développement urbain côtier à travers les prismes de la résilience). Le travail présente la première solution mise en œuvre pour CUTLER, visant à collecter, stocker et traiter des données hétérogènes provenant de quatre villes pilotes. L'article décrit le processus de collecte de données auprès de divers intervenants, la conception de la plateforme, appuyée sur les exigences des données diverses, la mise en œuvre de la plateforme, les défis et les limitations rencontrés, ainsi que des solutions proposées pour l'avenir. Jusqu'à présent, dans le cadre de CUTLER, des données provenant de 57 sources ont été collectées, traitées et stockées dans la plateforme, répondant aux besoins des pilotes.[3]

4. Data pre-processing pipeline generation for AutoETL

Le prétraitement des données joue un rôle clé dans le processus d'analyse de données, allant de la correction des erreurs à la sélection des caractéristiques les plus pertinentes pour la phase d'analyse. Les scientifiques des données ne peuvent pas facilement prévoir l'impact des prototypes de prétraitement et ont besoin d'une méthode pour discriminer entre eux et trouver les plus pertinents pour leur étude en cours. Dans ce travail, une méthode générique est développée pour construire des pipelines de prétraitement automatiques (AutoETL). Les résultats ont montré que l'optimisation des prototypes de pipeline efficaces fournit 90% de la précision prédictive de l'analyse en comparaison avec une recherche exhaustive, mais avec un temps de traitement 24 fois plus petit. La méthode proposée permet de guider l'instantiation des transformations pour faciliter la recherche de meilleures instantiations. [4]

5. Modelling Data Pipelines

Il existe plusieurs autres défis liés au transport des données de leur source à leur destination. Les pipelines de données sont mis en place pour automatiser le flux de données et réduire l'intervention humaine. Les pipelines ETL/ELT sont des représentations abstraites des pipelines de bout en bout. Pour exploiter pleinement le potentiel du pipeline de données, les auteurs essaient de comprendre les activités qui y sont liées et comment elles sont connectées dans un pipeline de données de bout en bout. Cette étude donne un aperçu de la conception d'un modèle conceptuel de pipeline de données qui peut être utilisé comme langage de communication entre différentes équipes de données. Le modèle de pipeline de données présenté dans cet article est un modèle conceptuel validé par une étude de cas exploratoire. Les sources de données peuvent être de différents types. Le modèle conceptuel de pipeline de données proposé dans cet article a des nœuds et des connecteurs qui effectuent les activités dans le flux de données. Selon les besoins, il peut être utilisé par n'importe quelle organisation pour n'importe quelle application de données en créant des instances.[5]

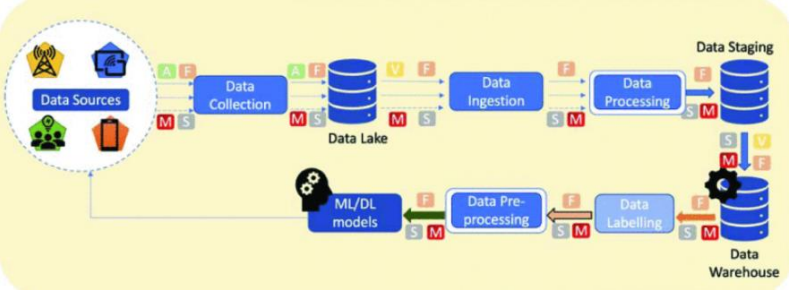


Image.2 (Fig. 6. Conceptual model of data pipeline)

6. Big Data Pipeline with ML-Based and Crowd Sourced Dynamically Created and Maintained Columnar Data Warehouse for Structured and Unstructured Big Data

Le système décrit dans l'article crée automatiquement et maintient dynamiquement son entrepôt de données en tant que partie de son pipeline Big Data, et ce pour les données structurées, semi-structurées et non structurées. Il utilise l'apprentissage automatique pour identifier et créer des dimensions, établir des relations entre des données de différentes sources et créer les dimensions correspondantes. Le système optimise dynamiquement les dimensions en fonction des données crowdsourcées fournies par les utilisateurs finaux ainsi que des analyses de requêtes. Ce système résout le défi de création et de maintenance d'un entrepôt de données en automatisant les tâches correspondantes. Cependant, l'analyse NLP doit être améliorée pour mieux gérer les données non structurées et il reste du travail à faire pour améliorer l'identification d'entités utiles en tant que dimensions d'intérêt. [6]

MODÈLE DE HAUT NIVEAU

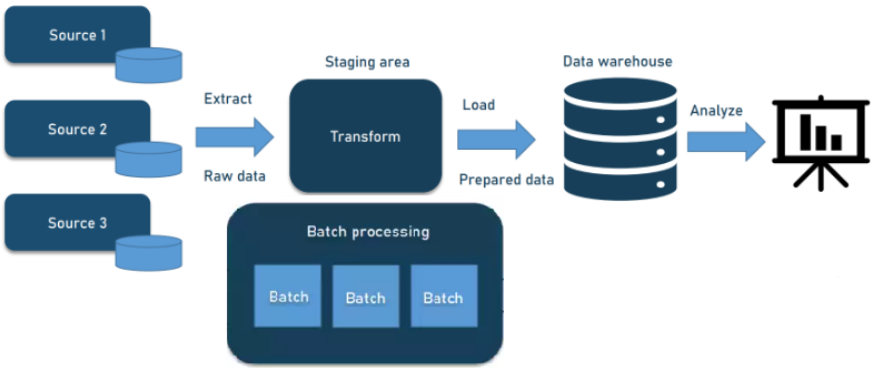


Image 3. ETL Pipeline Architecture, The logic behind batch data processing, Altexsoft

COMPOSANTS DU PROJET

YOK ACADÉMİQUE

"Yök Académique" est une plateforme en ligne créée par le Conseil de l'enseignement supérieur de Turquie (Yök) pour gérer les données académiques des universités turques. Elle contient des informations sur les académiciens turcs, notamment leur nom, prénom, titre, université, domaine principal, domaine scientifique, spécialités, adresse e-mail et ID de chercheur. La plateforme permet également de gérer les processus liés à l'enseignement supérieur en Turquie, tels que la sélection et la nomination des académiciens, l'évaluation de

la qualité de l'enseignement et de la recherche, ainsi que la gestion des bourses d'études et des programmes d'échanges académiciens.

Composants de Yök Académique Dataset

Université : université où travaille académicien

ORCID (Open Researcher and Contributor ID) : un identifiant numérique unique et persistant attribué à chaque chercheur. ORCID fournit un moyen de centraliser et de connecter les informations de recherche et de carrière d'un chercheur.

Chercheur ID : un identifiant numérique unique donné par système de Yök

Titre : une référence à la position hiérarchique au sein de l'institution académicien

Nom-Prénom

Position de cadre : une référence à une personne qui enseigne et mène des activités de recherche dans une université

Domaine principal : une référence à la discipline ou au domaine d'expertise principal dans lequel il mène des activités de recherche et d'enseignement

Domaine scientifique : une référence à la branche spécifique de la science dans laquelle il travaille et mène des recherches

Mot-clé : est un terme ou une phrase qui décrit le sujet ou le domaine d'intérêt spécifique de ses recherches

E-Mail : contact universitaire

URL : Le lien de la page Yök académique contenant les informations personnelles de l'académicien

GOOGLE SCHOLAR

Google Scholar est un moteur de recherche académique gratuit proposé par Google. Il permet aux utilisateurs de trouver des articles de recherche, des thèses, des livres, des résumés et des citations provenant de diverses sources, y compris des universités, des éditeurs scientifiques et des sites web professionnels. Les résultats de recherche de Google Scholar sont souvent considérés comme plus fiables et plus précis que les résultats de recherche généraux, car ils sont basés sur des sources académiciens et scientifiques de qualité. Google Scholar est souvent utilisé par les académiciens, les étudiants et les chercheurs pour trouver des articles pertinents sur un sujet de recherche particulier, ainsi que pour évaluer l'impact de leurs propres recherches en suivant les citations de leurs articles publiés.

Composants de Google Scholar Dataset

scholar_id : un identifiant numérique unique donné par google scholar

source : la première page source où l'on obtient les informations principales

name: nom de l'académicien

url_picture : URL de source

affiliation: information de l'université, du département et du titre

organization: identifiant unique donné pour l'université + département

interests : Les sujets et les domaines que l'académicien travaille

email_domain: nom de domaine e-mail de l'université

citedby: nombre total de citations que l'auteur a reçues

publications : publications de l'académicien

{

container_type : type de publication ex. : articles de recherche, des thèses, des livres, des résumés

source: la deuxième page source où l'on obtient les informations bib {

title: le titre de l'article

pub_year: année de publication de l'article

citation: titre de l'article que l'auteur fait la référence

}

num_citations : Le nombre de citations que l'article a reçues

}


```
    "publications"
],
"scholar_id": "3YA6oMkAAAAJ",
"source": "AUTHOR_PROFILE_PAGE",
"name": "E. Ertugrul Karsak",
"affiliation": "Professor of Industrial Engineering, Galatasaray University",
"organization": 811678055528458522,
"interests": [
    "Decision Analysis",
    "Multi-Criteria Decision Making",
    "Data Envelopment Analysis",
    "Product Development",
    "Supplier Selection"
```


LES ETAPES

Extraction de Données (Extract)

1- Un code de scrape a été écrit pour extraire automatiquement toutes les informations des académiciens via la plateforme "Yök Académique". Il existe 306 enregistrements d'académiciens.

2- Un code de scrape a été écrit pour extraire automatiquement les informations des articles des académiciens ainsi que leurs domaines de spécialisation actuels spécifiés via Google Scholar. Ils ont été enregistrés dans des JSON fichiers organisés par université. Il existe 137 enregistrements d'académiciens et 3625 enregistrements d'articles.

Transformation et Nettoyage de Données (Transform)

Les données extraites de Yök académique peuvent être récupérées sous forme de table HTML. Des scripts ont été écrits pour convertir ces données en format CSV/Excel et JSON.

Les codes de conversion ont été écrits en JavaScript pour que le processus d'extraction (scraping) de données et de conversion soit terminé en une seule script.

Chargement de Données (Load)

Les données extraites seront stockées dans un système de base de données tel qu'un data lake. Ensuite, un modèle sera utilisé pour les exploiter. On a besoin d'un système de base de données pour stocker les deux ensembles de données en ensemble. Compte tenu de l'évolutivité, du coût et de l'accessibilité, l'architecture en nuage a été préférée.

1- Azure Blob Storage

En raison de la limite de crédit, toutes les données ne peuvent pas être traitées. Les données seront chargées en filtrant les académiciens de l'Université Galatasaray en utilisant un compte étudiant.

Blob Storage se présente comme une structure de type lac de donnée avec une interface utilisateur pratique. On peut importer directement les données sous forme de fichiers sans aucune opération de transformation. Il est créé un compte de stockage, un conteneur et deux blobs pour stocker les données dans le même emplacement. Il est chargé les données provenant de Yök Académique en format « .xlsx » et les données provenant de Google Scholar en format « .json » .

On a, donc, créé un data lake où on stocke les données brutes(raw) de différents formats.

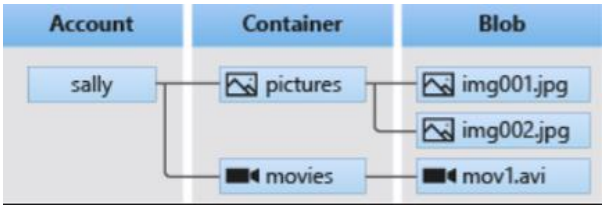


Image 4. Blob Storage Architecture, Microsoft Azure

À ce stade, on utilise 2 méthodes pour montrer les options d'accès aux données :

1- Blob à accès privé

2- Blob à accès public : toute personne disposant de l'adresse du fichier correspondant peut visualiser le fichier brut via un navigateur web (en lecture seule, sans possibilité de modification) ou le télécharger.

Les deux ensembles de données sont importés en environ 20 à 30 secondes.

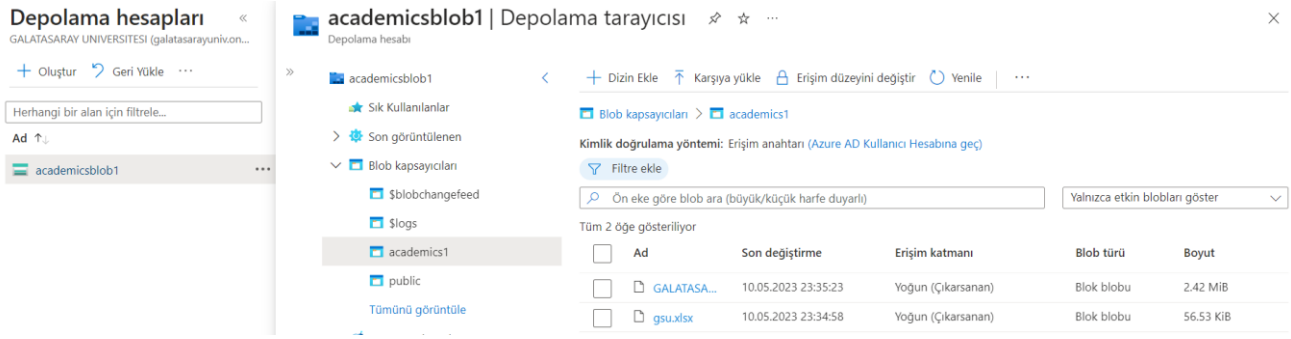


Image 5. Compte de Stockage, Conteneur, blobs (fichiers), Azure Blob Storage

Compte de Stockage, Conteneur, les blobs (fichiers) montrés dans l'Image 5 correspondent respectivement à Academicsblob1, Academics1, Gsu.xlsx et Galatasaray Üniversitesi.json.

2-Azure CosmosDB

Azure CosmosDB, une base de données NoSQL et relationnelle, ne demande pas en réalité une donnée structurée mais utilise un type de base de données relationnelle. Il est donc nécessaire de soumettre les données à une opération de transformation.

Il est écrit un script en javascript qui transforme la table HTML récupérée lors de l'extraction de données en JSON, puis il est chargé les données transformées en format JSON.

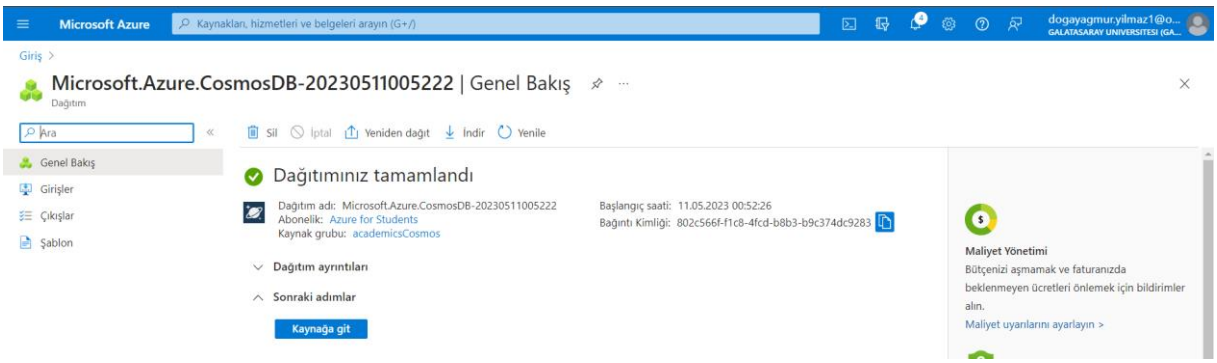


Image 6. Création de Compte de Stockage, Azure CosmosDB

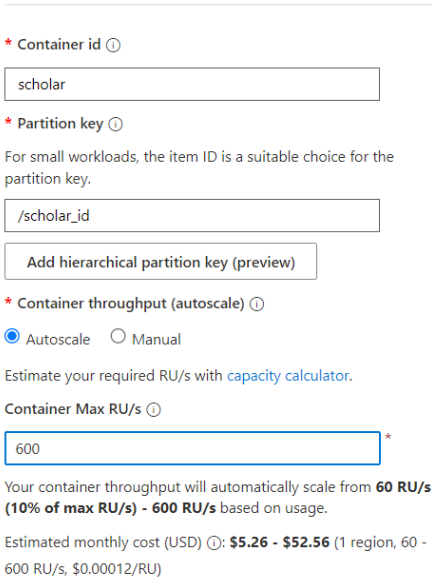


Image 7. Création de Conteneur "scholar", Azure CosmosDB

Il dure 3 minutes pour charger les données qu'on extraites de Google Scholar. Les données extraites de Yök Académique ont été chargées en 30 secondes.

3-MongoDB

MongoDB est un système de base de données orienté documents. Il ne prend pas en charge les fichiers « .xlsx », il accepte le format « csv ». Cependant, les enregistrements des données contient des caractères turcs et presque toutes les colonnes sont de type objet et les caractères turcs sont encodés en csv, ce qui entraîne une apparence erronée des données. Toutefois, si les données qui semblent corrompues en format csv sont converties en « .xlsx » à l'aide d'un convertisseur ou « d'Excel », elles fournissent une sortie correcte.

Il est utilisé mongosh pour établir une connexion avec MongoDB, ainsi que pour créer une base de données et une collection. Il est ensuite utilisé mongoimport pour importer les fichiers ".csv" et ".json" dans la base de données.

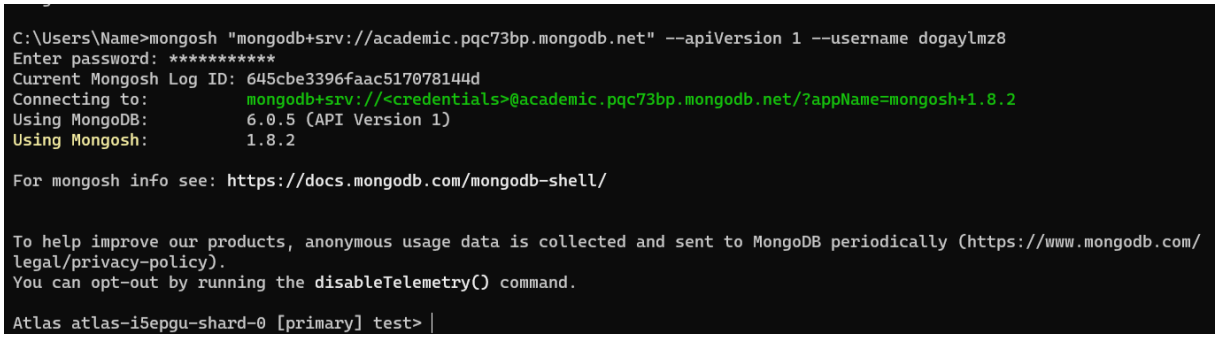


Image 8. Connection à MongoDB avec Mongosh

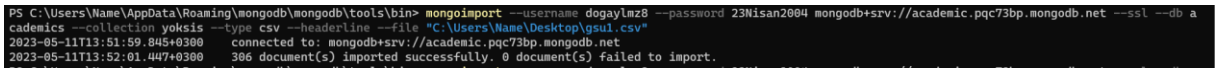


Image 9. Mongoimport fichier csv

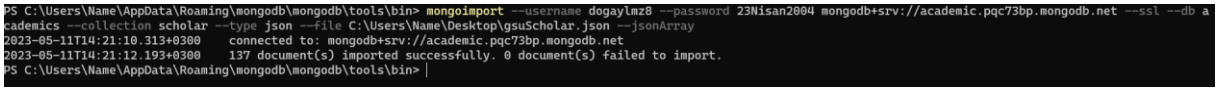


Image 10. Mongoimport fichier JSON

EVALUATION GENERALE D'ETUDE

Les données des académiciens ont été recueillies lors de la phase préliminaire de l'étude à partir de deux sources présentées en deux manières distinctes. Tout d'abord, les données ont été obtenues sous forme de fichier csv à partir du système " Yök Académique", choisi comme source de données principale. Les articles des académiciens dont les données ont été acquises à partir du système précédent ont été récupérés sous format json à partir de "Google Scholar", qui a été utilisé comme deuxième source de données.

Les données extraites sont stockées dans les bases de données des systèmes cloud tels qu'Azure Blob Storage, Azure CosmosDB et MongoDB. Parmi les systèmes où les données sont chargées, le stockage en blob dans l'architecture "Data Lake", où les données brutes peuvent être directement chargées, a été choisi comme zone de stockage.

Les données provenant de Yök Académique, qui ont été converties en « csv » pour être stockées dans MongoDB, ont également été chargées dans le stockage blob (lac de données) pour des raisons de commodité dans les processus de conversion.

ORC (Optimized Row Columnar) et Parquet sont deux formats columnar de fichiers big data. Parquet est généralement plus adapté aux analyses en écriture unique et en lecture multiple, tandis qu'ORC convient mieux aux opérations de lecture intensive. ORC est optimisé pour les données Hive, qui sont adaptées à la structure de l'entrepôt de données sur laquelle les requêtes SQL peuvent être exécutées.

Puisque des requêtes seront exécutées dans cette étude, il serait approprié de convertir les données au format parquet avant de les donner à Spark. Étant donné que Data Factory ne parvient pas à convertir le fichier de données mappées reçu de Blob au format parquet, il convient d'utiliser le format « orc », qui est utilisé pour stocker efficacement les données Hive. Ce format a été conçu pour surmonter les limites des autres formats de fichier et convient pour les requêtes.

Grâce à un script Python, deux fichiers distincts ont été réunis en un seul fichier « json » et convertis au format «ORC» à l'aide d'azure data factory installé sur le stockage blob.

Étant donné que CosmosDB demande des données au format json, les données extraites de Yök Académique doivent être converties manuellement, à l'aide de code, avant d'être placées dans la première unité de stockage.

Pour créer le pipeline, il est préférable d'utiliser une structure dans laquelle les données peuvent être stockées dans la structure du lac de données et les données peuvent être converties au format souhaité en installant une data factory. Afin d'analyser les données avec la connexion Spark, l'option Blob Storage a été utilisée. Spark est un outil de traitement de données rapide et en mémoire qui prend en charge les analyses de big data et inclut un support SQL et Python comme SparkSql et PySpark.

Azure propose une version d'essai gratuite en deux manières. Le compte étudiant et le compte free trial permettent d'utiliser des services spécifiques avec certaines restrictions. Malheureusement, l'utilisation de Spark avec le compte étudiant n'est pas ouverte, car après la création réussie du compte Spark, le noyau nécessaire pour créer un cluster n'est pas défini dans le compte donc un cluster ne peut pas être créé et utilisé sur Spark ou les Databricks basées sur Spark avec le compte étudiant. Étant donné que le compte d'essai gratuit a également une limite de 4 noyaux et qu'au moins 8 noyaux sont nécessaires pour créer un cluster sur Spark, il n'est pas approprié d'utiliser Spark sur Azure pour des raisons de coût.

Une autre option est le nuage de Google. En fournissant un accès à Azure Blob Storage, il est possible de transférer les données combinées et transformées par Azure Data Factory vers Google Cloud. Il est également possible d'effectuer un processus d'analyse sans serveur en faisant progresser les données reçues sur Google Cloud vers le traitement par lots Apache Spark à l'aide de dataproc (via une connexion api). En outre, Google fournit une connexion BigQuery pour les requêtes SQL sur Spark.

Dans le projet, qui progresse comme traitement par lots à la première étape, il est en fait nécessaire de développer un pipeline en tenant compte du modèle de streaming afin de maintenir les informations des académiciens à jour et d'inclure de nouveaux articles dans la base de données à créer.

À ce stade, l'utilisation de la fonction de streaming de BigQuery sera une approche logique pour le projet.

Image 11. Modèle en Google Cloud, Google BigQuery

La base de données source présentée dans l'image 11 est une représentation du stockage Blob. Elle fait référence aux données de « data lake » qui sont fusionnées et réécrites dans le blob au format « Orc » à l'aide de « data factory ». Dans la phase de Datastream (flux de données), les données du Blob sont transférées vers Google Cloud pour réaliser Bigquery. Dans Google Cloud, les unités de stockage et de calcul sont séparées. Par conséquent, les données extraites du Blob le sont à des fins de traitement et non de stockage.

PROBLEMES RENCONTRES ET LES SOLUTIONS

1- Lors de la Scrape, Google Scholar ne dispose pas d'une solution propre pour télécharger les données. De plus, l'adresse IP qui demande à plusieurs reprises l'accès à la page est bloquée. Pour cette raison, les données accessibles au public sur google scholar ont été retirées par scraping avec plusieurs machines différentes à des intervalles de temps et sauvegardées au format json.

2- Étant donné que le projet a été mené sur des comptes d'étudiants ou des essais gratuits de comptes en nuage, toutes les données retirées n'ont pas pu être exploitées. Les données des universitaires de l'université Galatasaray ont donc été conservées séparément et l'étude a été réalisée sur ces données.

3- Comme la connexion Spark ne pouvait pas être realisée par Azure, d'autres solutions cloud capables d'extraire des données en format ORC d'un stockage blob et de fournir une connexion Spark ont été recherchées. Dans cette direction, la solution cloud de google a été mise en place pour travailler avec le modèle de flux.

4- Comme la connexion Spark ne pouvait pas être fournie, la mise en correspondance des données a été assurée par un script de python et la conversion de format des données a été assurée à l'aide d'Azure data factory dans un seul fichier.

5- Étant donné que les données au format ORC écrites dans le blob peuvent être extraites directement vers le nuage Google et utilisées dans les opérations BigQuery, il n'est pas nécessaire de disposer d'une connexion spark en cas d'utilisation de BigQuery.

6- La corruption et la perte de données s'étant produites lors de la conversion du format de données en parquet via data factory, ORC, qui est un autre format en colonnes sur lequel des requêtes peuvent être exécutées, a été préféré.

PARAMETRES A UTILISER DANS L'ANALYSE DES PERFORMANCES

1- Accessibilité
2- Perte de Données
3- Stockage de Données Brutes (Raw)
4- La durée de Conversion
5- La durée des Résultats de l'Analyse/ Requête
6- Le Coût
7- Reconnaissance de Format de Données Correspondant

VERIFICATION ET PREMIERS RESULTATS DU SYSTEME

Conversion de Format de Données

Le processus de pipeline qui prend le fichier « gsu-all.json » blob qui se trouve dans le container « academics2 », le convertit en « all.orc »blob et l'écrit en container « orcall » a été réalisé avec succès en Azure data factory.

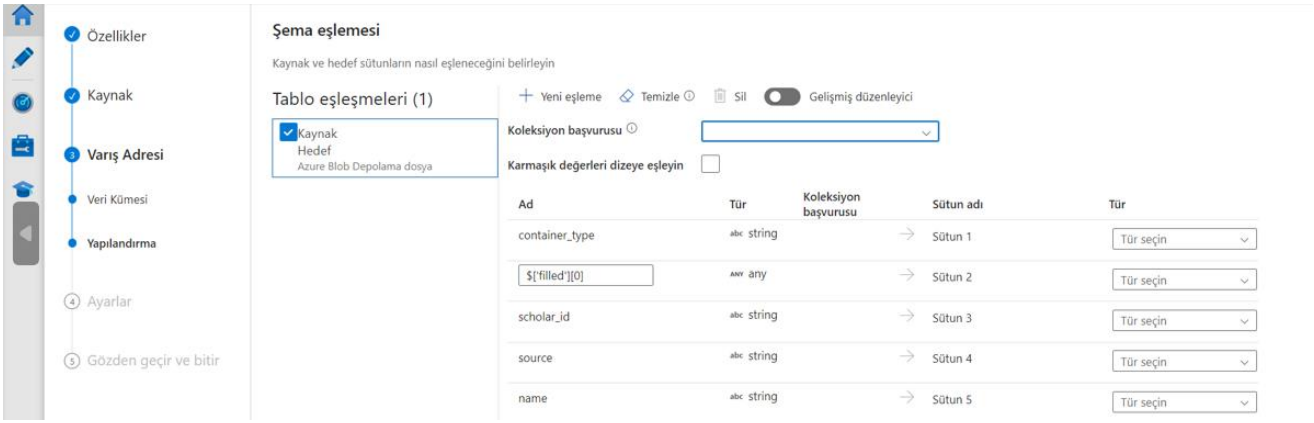


Image 12. Création du pipelineORC Pipeline, Azure Data Factory

Image 13. Prévisualisation des données d'entrée, Azure Data Factory

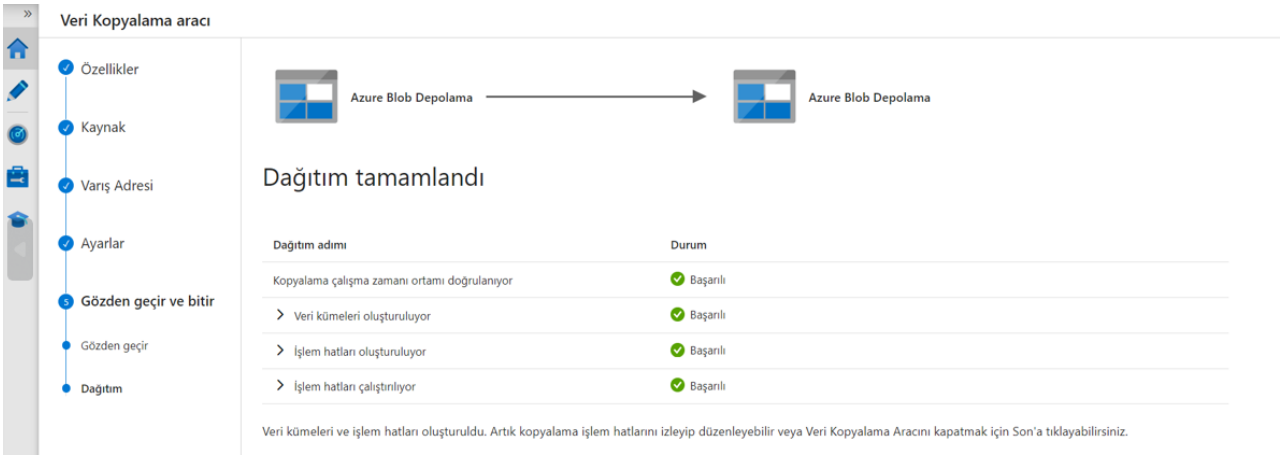


Image 14. Réalisation de la Conversion,pipelineORC, Azure Data Factory

Datastream en Google Cloud

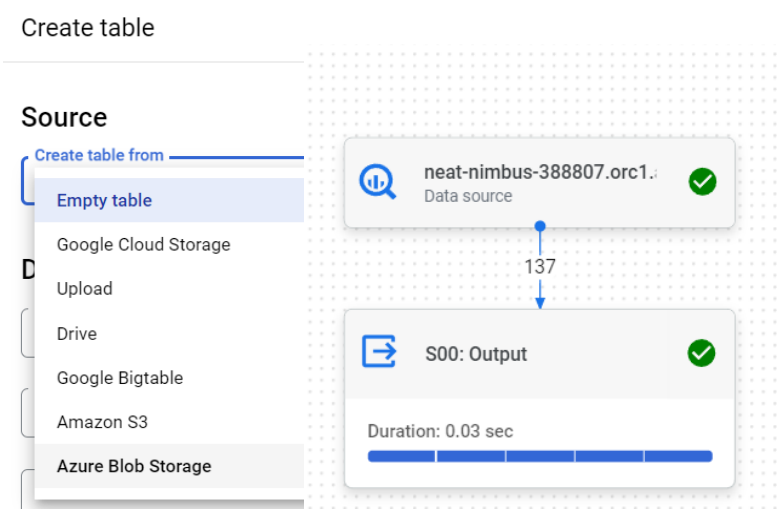
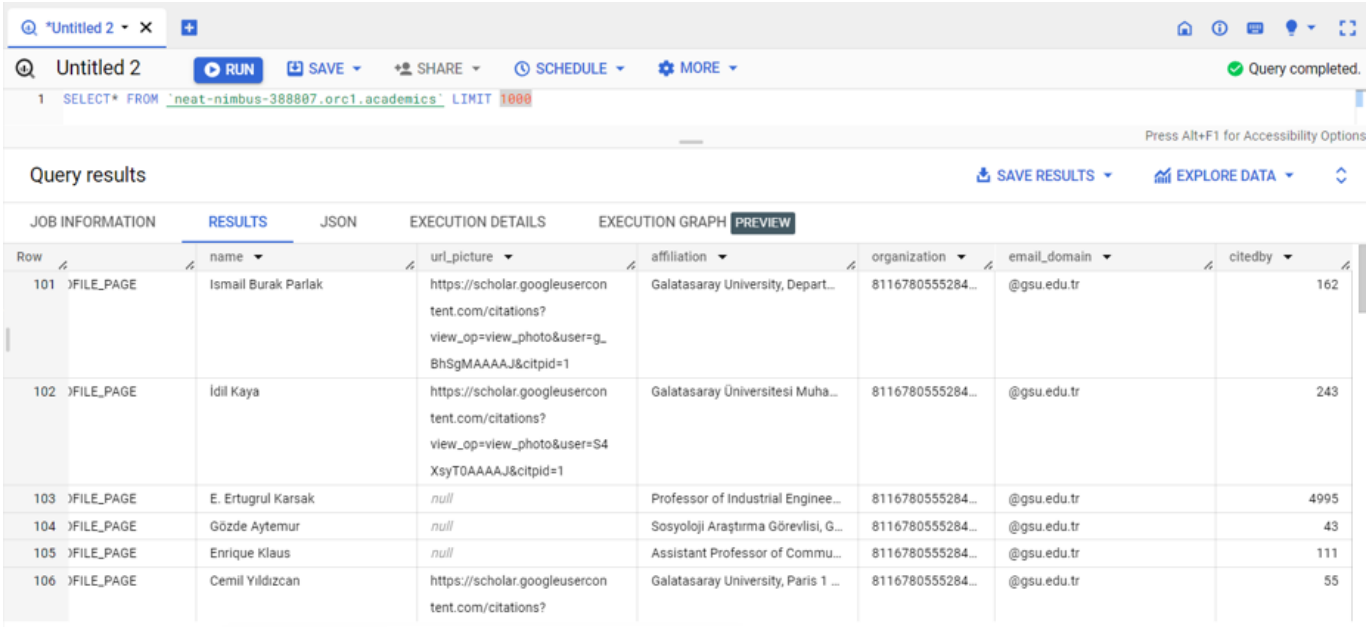


Image 15. Extraction de Données d'Azure Blob Storage en format ORC

Google BigQuery / Requêtes sur Data

Une fois les données transférées au bigquery, elles ont été exécutées à l'aide d'une requête SQL par défaut. L'image de n'importe quelle partie des données défilant est présentée ci-dessous.



Row		name	url_picture	affiliation	organization	email_domain	citedby
101	)FILE_PAGE	Ismail Burak Parlak	https://scholar.googleusercontent.com/citations?view_op=view_photo&user=g_BhSgMAAAAJ&citpid=1	Galatasaray University, Depart...	8116780555284...	@gsu.edu.tr	162
102	)FILE_PAGE	İdil Kaya	https://scholar.googleusercontent.com/citations?view_op=view_photo&user=S4XsyT0AAAAJ&citpid=1	Galatasaray Üniversitesi Muha...	8116780555284...	@gsu.edu.tr	243
103	)FILE_PAGE	E. Ertugrul Karsak	null	Professor of Industrial Enginee...	8116780555284...	@gsu.edu.tr	4995
104	)FILE_PAGE	Gözde Aytemur	null	Sosyoloji Araştırma Görevlisi, G...	8116780555284...	@gsu.edu.tr	43
105	)FILE_PAGE	Enrique Klaus	null	Assistant Professor of Commu...	8116780555284...	@gsu.edu.tr	111
106	)FILE_PAGE	Cemil Yıldızcan	https://scholar.googleusercontent.com/citations?	Galatasaray University, Paris 1 ...	8116780555284...	@gsu.edu.tr	55

Image 16. Résultat de Requête sur Data en Format ORC

L'affichage au format json de la requête exécutée sur data en format ORC et ses résultats à l'écran est le suivant. Comme tous les résultats ne peuvent pas être affichés, le fichier au format json est téléchargé à locale.

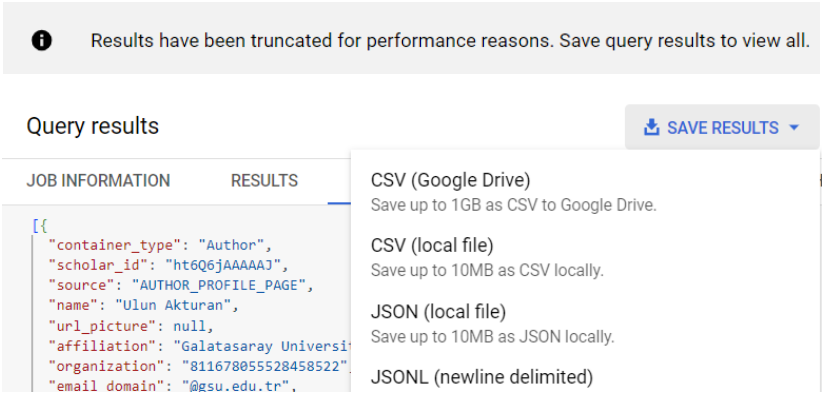


Image 17.1. Téléchargement des Résultats sous Forme de Fichier json

La requête est exécutée sur le data provenant de Blob Storage et l'information sur la 101ème ligne est affiché dans le cadre de données aléatoirement sélectionné.


```
{
    "container_type": "Author",
    "filled": [
        "basics",
        "publications"
    ],
    "scholar_id": "g_BhSgMAAAAJ",
    "source": "AUTHOR_PROFILE_PAGE",
    "name": "Ismail Burak Parlak",
    "url_picture": "https://scholar.googleusercontent.com/citations?view_op=view_photo&user=g_BhSgMAAAAJ&citpid=1",
    "affiliation": "Galatasaray University, Department of Computer Engineering",
    "organization": 81167805552845852 2,
    "interests": [
        "Natural Language Processing",
        "Data Science",
        "Image & Video Processing",
        "Deep Learning",
        "Medical Informatics"
    ],
    "email_domain": "@gsu.edu.tr",
    "homepage": "https://avesis.gsu.edu.tr/bparlak",
    "citedby": 162,
    "publications": [
        {
            "container_type": "Publication",
            "source": "AUTHOR_PUBLICATION_ENTRY",
            "bib": {
                "title": "Finite-interval-valued Type-2 Gaussian fuzzy numbers applied to fuzzy TODIM in a healthcare problem",
                "pub_year": "2020",
                "citation": "Engineering Applications of Artificial Intelligence 87, 103352, 2020"
            },
            "filled": false,
            "author_pub_id": "g_BhSgMAAAAJ:_FxGoFyzp5QC",
            "num_citations": 87,
            "citedby_url": "https://scholar.google.com/scholar?oi=bibs&hl=en&cites=17284985530621533961",
            "cites_id": [
```


...


```
    ]
],
"Üniversite": "GALATASARAY ÜNİVERSİTESİ",
"ORCID": "0000-0002-0887-4226",
"Araştırmacı ID": "132155",
"Unvan": "DOKTOR ÖĞRETİM ÜYESİ",
"Ad Soyad": "İSMAİL BURAK PARLAK",
"Kadro Yeri": "GALATASARAY ÜNİVERSİTESİ/MÜHENDİSLİK VE TEKNOLOJİ FAKÜLTESİ/BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ/BİLGİSAYAR BİLİMLERİ ANABİLİM DALI/",
"Temel Alan": "Mühendislik Temel Alanı",
"Bilim Alan": "Bilgisayar Bilimleri ve Mühendisliği",
"Anahtar Kelime": "Biyoenformatik ; Yapay Zeka ; Yapay Öğrenme",
"E-Posta": "bparlak[at]gsu.edu.tr",
"Araştırmacı GUID ID": "01955822F961DFA5",
"URL": "https://akademik.yok.gov.tr/AkademikArama/AkademisyenGorevOgrenimBilgileri?sira=nT3KGP3694fUamKe2z0D-w&authorId=01955822F961DFA5"
}
```


Image 17.2 Visualisation de Fichier ORC en JSON qui a été Requête

En réunissant les données provenant de Yök Académique et de Google Scholar, les informations ont été écrites dans un seul fichier sous la forme de données de « google scholar + données de yök académique ».

A titre d'exemple, on examine les résultats des requêtes des lignes 54 et 55 et la compatibilité des données avec le fichier au format json dans lequel on a réuni les deux fichiers avant la conversion des données.

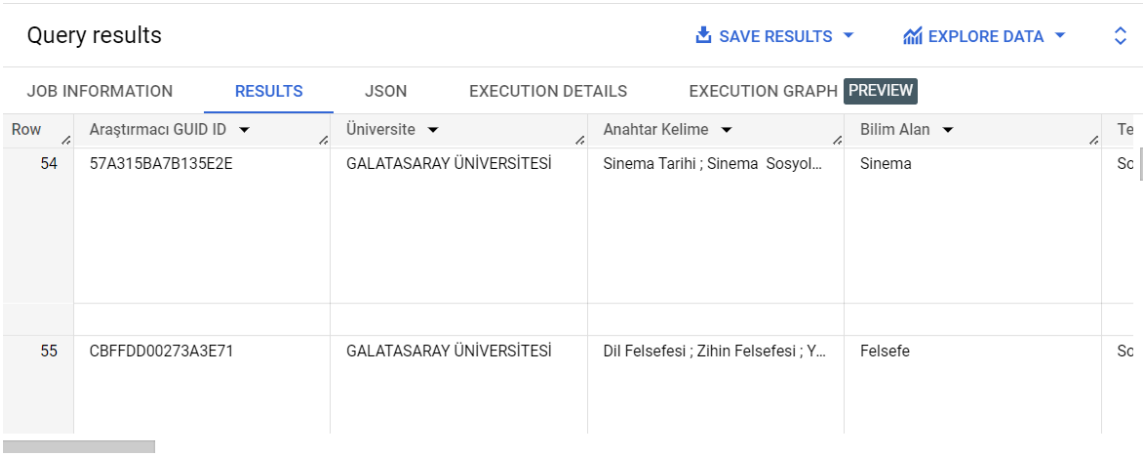


Image 18.0

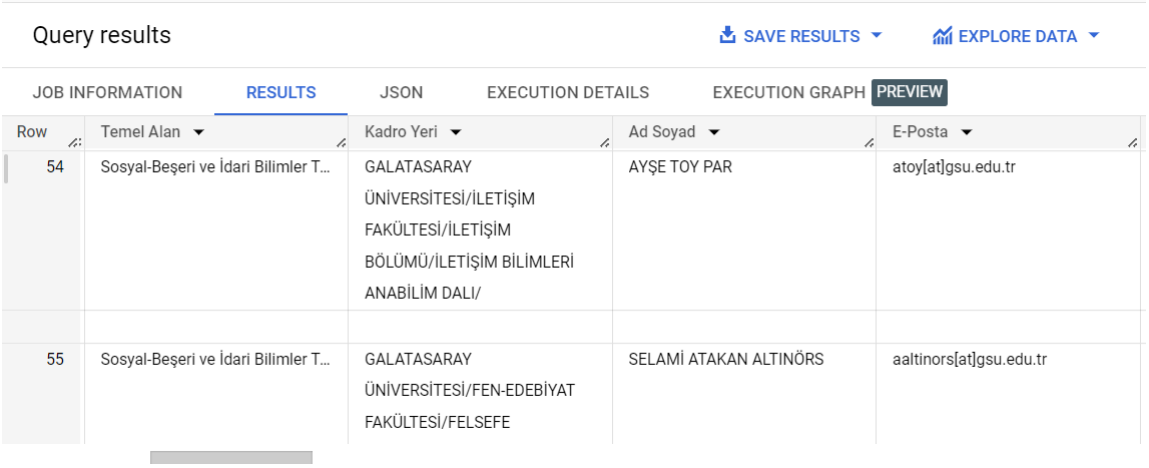


Image 18.1

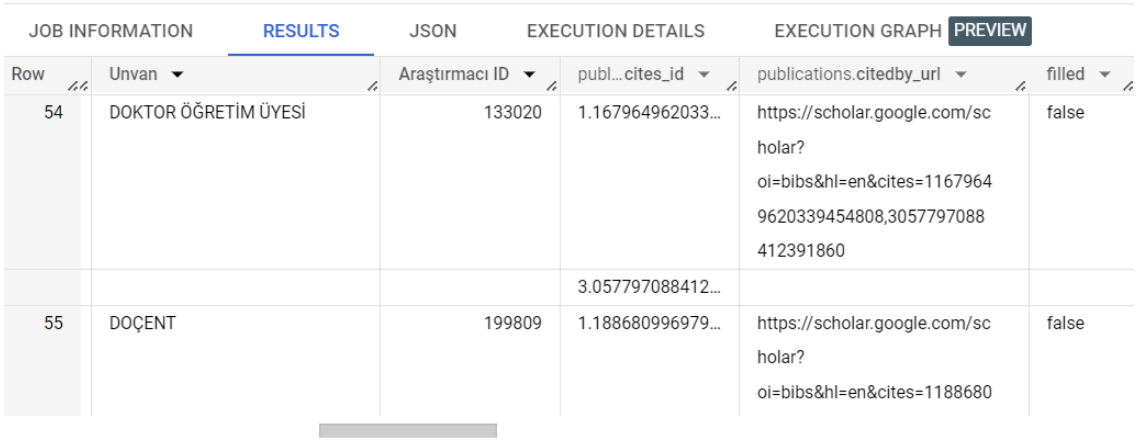


Image 18.2

Row	num_citations	publications.bib.citation	pu... pub_year	publications.bib.title	publications.source
54	9		2009	El kapılarında yeşilçam: 1970-1990 arası Türkiye'de dış göç-sinema ilişkisi	AUTHOR_PUBLICATION_ENTRY
55	38	Erciyes Üniversitesi Sosyal Bilimler Enstitüsü Dergisi 1 (28), 389-401, 2010	2010	Düşünce ile dil arasındaki ilişki...	AUTHOR_PUBLICATION_ENTRY

Image 18.3

Row	publications.container_type	email_domain	URL	source
54	Publication	@gsu.edu.tr	https://akademik.yok.gov.tr/AkademikArama/AkademisyenGorevOgrenimBilgileri?sira=_D2bDdARDMCOgjHVGzFVZw&authorId=57A315	AUTHOR_PROFILE_PAGE
55	Publication	@gsu.edu.tr	https://akademik.yok.gov.tr/AkademikArama/AkademisyenGorevOgrenimBilgileri?	AUTHOR_PROFILE_PAGE

Image 18.4

Row	source	ORCID	url_picture	scholar_id
54	AUTHOR_PROFILE_PAGE	0000-0001-7638-2373	https://scholar.googleusercontent.com/citations?view_op=view_photo&user=iKQuRIoAAAAJ&citpid=1	iKQuRIoAAAAJ
55	AUTHOR_PROFILE_PAGE	0000-0001-6072-288X	https://scholar.googleusercontent.com/citations?view_op=view_photo&user=cl	clTUAfwAAAAJ

Image 18.5

Row	affiliation	citedby	organization	name	filled
54	İletişim Fakültesi, Galatasaray ...	9	8116780555284...	Ayse Toy Par	basics
					publications
55	Galatasaray Üniversitesi	91	8116780555284...	S.Atakan ALTINÖRS	basics

Image 18.6

Row	filled	homepage	interests	container_type
54	basics	*null*	Sinema	Author
	publications		Türk Sinema Tarihi	
55	basics	*null*	Dil felsefesi	Author

Image 18.7


```
"container_type": "Author",
"filled": [
    "basics",
    "publications"
],
"scholar_id": "iKQuRloAAAAJ",
"source": "AUTHOR_PROFILE_PAGE",
"name": "Ayse Toy Par",
"url_picture": "https://scholar.googleusercontent.com/citations?view_op=view_photo&user=iKQuRloAAAAJ&citpid=1",
"affiliation": "İletişim Fakültesi, Galatasaray Üniversitesi",
"organization": 811678055528458522,
"interests": [
    "Sinema",
    "Türk Sinema Tarihi"
],
"email_domain": "@gsu.edu.tr",
"citedby": 9,
"publications": [
    {
        "container_type": "Publication",
        "source": "AUTHOR_PUBLICATION_ENTRY",
        "bib": {
            "title": "El kapılarında yeşilçam: 1970-1990 arası Türkiye'de dış göç-sinema ilişkisi",
            "pub_year": "2009",
            "citation": ""
        },
        "filled": false,
        "author_pub_id": "iKQuRloAAAAJ:u5HHmVD_uO8C",
        "num_citations": 9,
        "citedby_url": "https://scholar.google.com/scholar?oi=bibs&hl=en&cites=11679649620339454808,3057797088412391860",
        "cites_id": [
            "11679649620339454808",
            "3057797088412391860"
        ]
    }

],
"Üniversite": "GALATASARAY ÜNİVERSİTESİ",
"ORCID": "0000-0001-7638-2373",
"Araştırmacı ID": "133020",
"Unvan": "DOKTOR ÖĞRETİM ÜYESİ",
"Ad Soyad": "AYŞE TOY PAR",
"Kadro Yeri": "GALATASARAY ÜNİVERSİTESİ/İLETİŞİM FAKÜLTESİ/İLETİŞİM BÖLÜMÜ/İLETİŞİM BİLİMLERİ ANABİLİM DALI/",
"Temel Alan": "Sosyal-Beşeri ve İdari Bilimler Temel Alanı",
"Bilim Alan": "Sinema",
"Anahtar Kelime": "Sinema Tarihi ; Sinema  Sosyolojisi",
"E-Posta": "atoy[at]gsu.edu.tr",
"Araştırmacı GUID ID": "57A315BA7B135E2E",
"URL": "https://akademik.yok.gov.tr/AkademikArama/AkademisyenGorevOgrenimBilgileri?sira=_D2bDdARDMCOgjHVGzFVZw&authorId=57A315BA7B135E2E"
```


18.8

MODELE DU PIPELINE

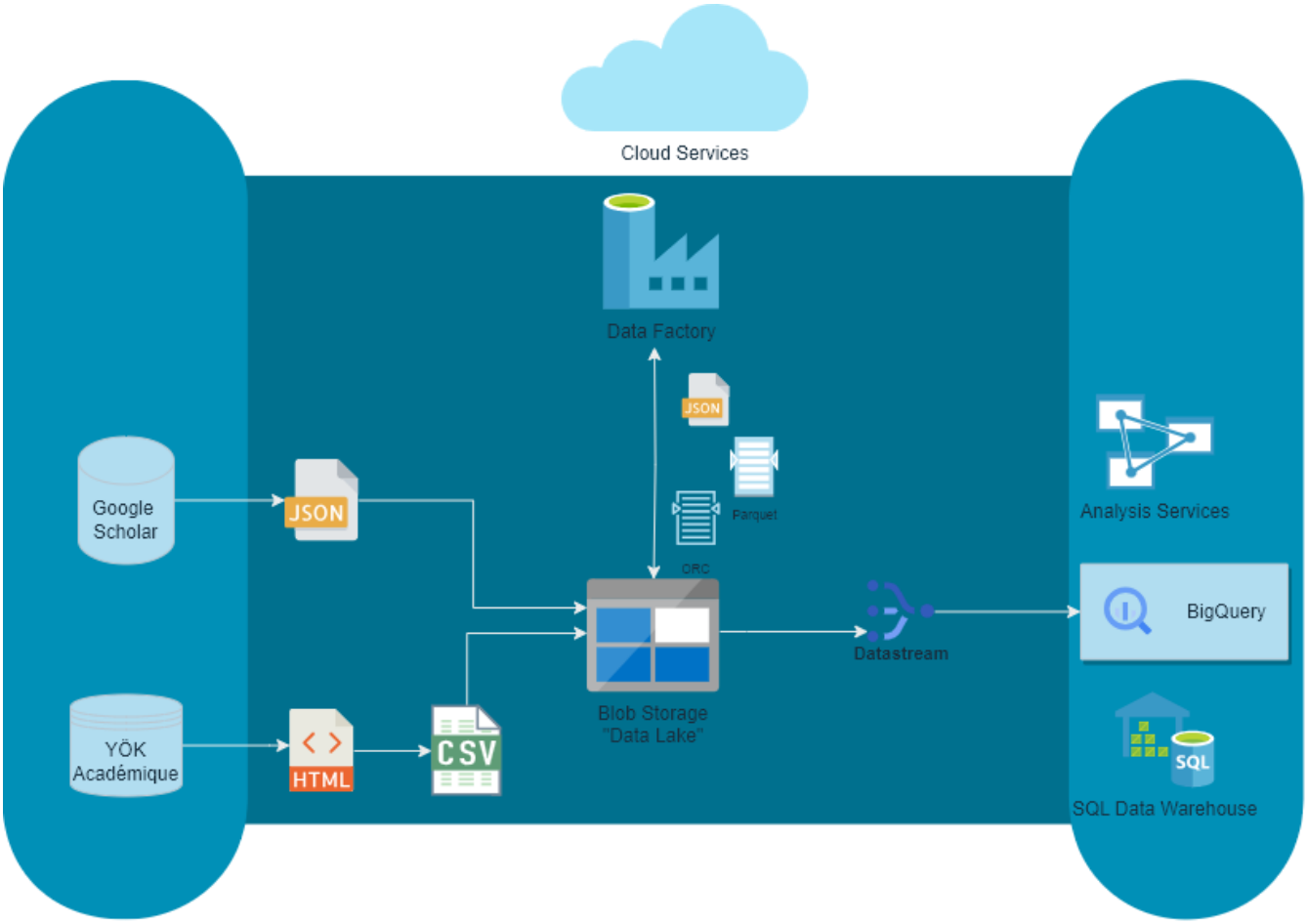


Image 19. Modèle de Pipeline de Traitement des Données Finales

ANALYSE DES PERFORMANCES

Accessibilité

Lorsque notre objectif était d'analyser les données à l'aide de spark, on a rencontré des problèmes d'accès, alors qu'il n'existe aucun problème d'accès avec Google Bigquery.

Perte de Données

Afin d'exécuter une requête SQL sur ces données, on a d'abord converti les données au format parquet, mais une corruption des données s'est produite. Lorsque on a utilisé le format ORC, les données ont été préservées et il n'y a pas eu de problème pour sql. On a vérifié cette situation et ajouté sur le document dans la section « Google BigQuery / Requêtes sur Data » comme la série des Images 18.x.

Stockage de Données Brutes-Raw (non traitées)

Alors que mongodb et cosmosdb ne nous permettent pas de déposer directement les données raw, le stockage blob nous a permis de créer une structure de lac de données en acceptant directement les données raw.

La Durée de Conversion

En raison de contraintes du compte d'étudiant, aucune des conversions de format n'a pris beaucoup de temps car on a travaillé avec une quantité limitée de données.

La durée des Résultats de l'Analyse/ Requête

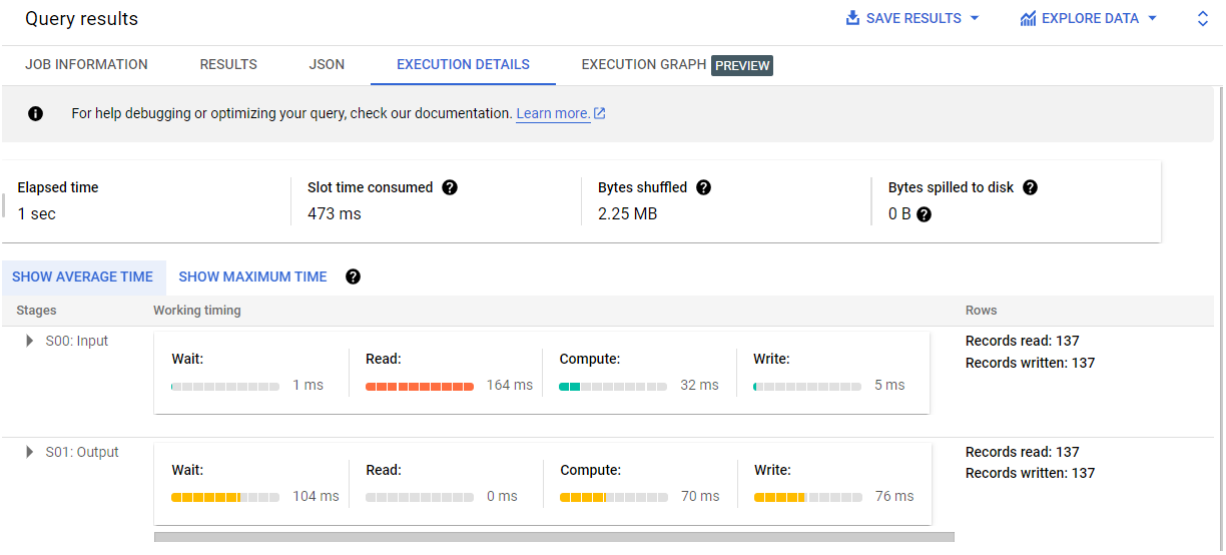


Image 19. Exécution Details, Google BigQuery

Le coût

Ce paramètre a causé de nombreux problèmes qu'ils sont expliqués dans les sections précédentes.

Reconnaissance de format de données correspondant

Google Bigquery n'a pas pu reconnaître le schéma des données lorsqu'on a envoyé des données au format json et qu'on a essayé de le requêter. Cette situation montre une fois de plus que la conversion des données en colonnes est nécessaire pour la requête.

CONCLUSION

Deux ensembles de données sous différents formats provenant de différentes sources sont stockés dans le stockage Blob qui est une architecture de type "lac de données". Puis elles ont été intégrées par des infos «université et prénom-nom» à l'aide d'un code de scrape et converties au format ORC (colonné) via Data Factory avec succès.

Les données ont été requêtées en les coulant au Google BigQuery. La requête a permis d'obtenir des informations sur les académiciens et leurs articles sous forme de tableaux.

En raison de contraintes liées au coût des systèmes en nuage, seule la partie de l'université Galatasaray des données extraites a été utilisée.

A la fin d'étude, le base de données contient 137 académiciens et leurs articles.

BIBLIOGRAPHIE

[1] T. Hlupić and J. Puniš, "An Overview of Current Trends in Data Ingestion and Integration," *2021 44th International Convention on Information, Communication and Electronic Technology (MIPRO)*, Opatija, Croatia, 2021, pp. 1265-1270, doi: 10.23919/MIPRO52101.2021.9597149.

[2] Chinmaya Kumar Dehury, Pelle Jakovits, Satish Narayana Srirama, Giorgos Giotis, Gaurav Garg, TOSCAdata: Modeling data pipeline applications in TOSCA, Journal of Systems and Software, Volume 186, 2022, 111164, ISSN 0164-1212, doi: 10.1016/j.jss.2021.111164.

[3] G. Pal, K. Atkinson and G. Li, "Managing Heterogeneous Data on a Big Data Platform: A Multi-criteria Decision Making Model for Data-Intensive Science," 2020 IEEE International Conference on Big Data and Smart Computing (BigComp), Busan, Korea (South), 2020, pp. 229-239, doi: 0.1109/BigComp48618.2020.00-69.

[4] Joseph Giovanelli, Besim Bilalli, Alberto Abelló, Data pre-processing pipeline generation for AutoETL, Information Systems, Volume 108, 2022, 101957, ISSN 0306-4379, doi: 10.1016/j.is.2021.101957.

[5] A. Raj, J. Bosch, H. H. Olsson and T. J. Wang, "Modelling Data Pipelines," *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, Portoroz, Slovenia, 2020, pp. 13-20, doi: 10.1109/SEAA51224.2020.00014.

[6] K. Ghane, "Big Data Pipeline with ML-Based and Crowd Sourced Dynamically Created and Maintained Columnar Data Warehouse for Structured and Unstructured Big Data," 2020 3rd International Conference on Information and Computer Technologies (ICICT), San Jose, CA, USA, 2020, pp. 60-67, doi: 10.1109/ICICT50521.2020.00018.