

Projet sur les bases de données du Big Data :

Etudes des bases de Données NoSql à partir de critères

GROUPE 5 - Azure Cosmos DB

Préambule

Le travail ici demandé fait suite au cours « Les Bases de Données du Big Data ». Ces bases de données sont aussi souvent appelées Bases de Données NoSql. L'objectif est, à partir de critères et d'un schéma de données prédéfinis, d'évaluer une base de données Nosql par **groupe de cinq étudiants**. Ce TP permet, en plus de l'examen, d'évaluer le cours « Les Bases de Données du Big Data » (50% de la note).

Table des matières

Identité du moteur NoSql	6
Editeur	6
Version initiale (Nr. De version et date)	6
Version actuelle (Nr. De version et date)	6
Modèles de données supportés(Clé/Valeur, Orienté document, Orienté Colonnes, Orienté Graphe)	6
Gestion du schéma (sans schéma, dynamique, statique, mixte)	6
Support de SQL (DDL, DML)	6
Support des indexes secondaires	6
Langage de développement du sgbd nosql	7
Support / Pérennité (communauté , etc....)	7
API Supportés	7
Théorème CAP (CP, AP, AC)	7
Méthode de partitionnement	7
Méthode de réplication	8
Concept de consistance	8
Concept de durabilité	8
Clés étrangères	8
Support de références (REF)	8
Licences et prix	9
Différents types de versions (communautaire, entreprise, ...)	9
Audience dans le marché	9
Tables et tables filles	10
Clé avec major et minor key	10
Gestion des utilisateurs	11
Gestion des droits	11
Gestion des namespaces ou databases	11
Systèmes d'exploitations supportés	11

Disponible en mode DBaaS	12
Support du Map/Reduce	12
Lien vers la documentation technique y compris les API	12
Typage (none, static, dynamique)	12
Applications communautaires l'utilisant	12
Domaines d'applications	12
Architecture du moteur NoSql	13
Montée en charge	13
Gestion de la disponibilité	13
Procédure d'installation	13
Modélisation et chargement des données	14
Mises à jours des données (insert, update, delete)	14
Maj de données partie Key/value	14
Maj de données partie Document	14
Maj de données partie Orientée colonnes	14
Maj de données partie Relationnelles	14
Maj de données partie Graphes	15
Interrogation des données	15
Les résultats de l'étude	16
Répartition du travail	16
Où s'inscrire pour constituer son groupe de 5 et choisir son SGBD NoSql	17

Description du projet

Afin de mener à bien le projet de l'UE Bases de Données pour le Big Data, nous avons formé un groupe de 5 étudiants. Après concertation, nous avons choisis de travailler sur la base de données Azure Cosmos DB.

Après analyse, nous avons constaté que Azure Cosmos DB peut stocker des données selon les 5 modèles suivants :

- Relational
- Key-Value
- Document
- Graph
- Column

Dans un but d'apprentissage, nous avons cherché à exploiter toutes les capacités de Cosmos DB. Nous sommes donc partis sur une infrastructure cloud multimodel avec une architecture logicielle (Java) adéquate. Cette architecture nous a permis de pouvoir exploiter le potentiel de l'infrastructure et de gérer ces 5 types de données différents au sein de Azure Cosmos DB.

L'exploitation de ces différents types de données nous a amenés à utiliser 5 APIs différentes:

- API SQL pour les données de type Relational
- Table API pour les données de type Key-Value
- API for MongoDB pour les données de type Document
- Gremlin API pour les données de types Graph
- Cassandra API pour les données de type Column

Par conséquent, l'architecture de notre projet Java a été réalisée en se basant sur ces 5 APIs. De ce fait, nous avons dédié la majeure partie de ce projet à mettre en place les DAOs des multiples APIs de manière générique et réutilisable. Nous n'avons donc pas pu terminé l'intégralité de la partie 5.

Lien du github :

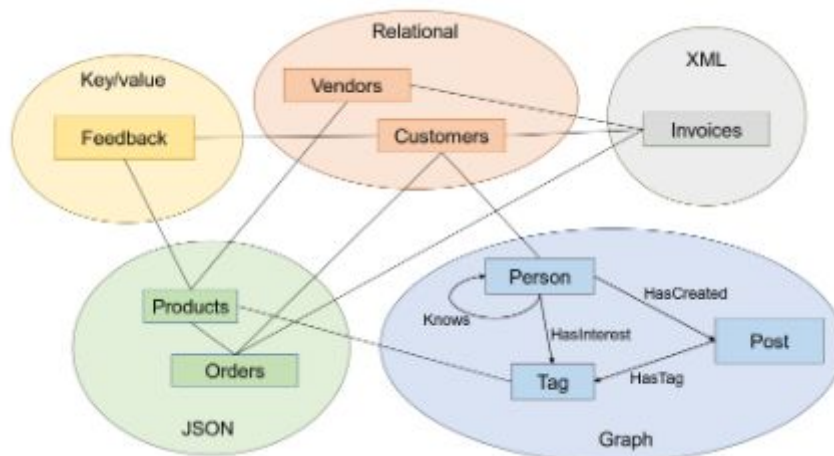
https://github.com/AdrienAudouard/TP-Transversal-M1MIAGE_-NOSQL_BD-2018_2019

1. Schéma et les données de l'étude

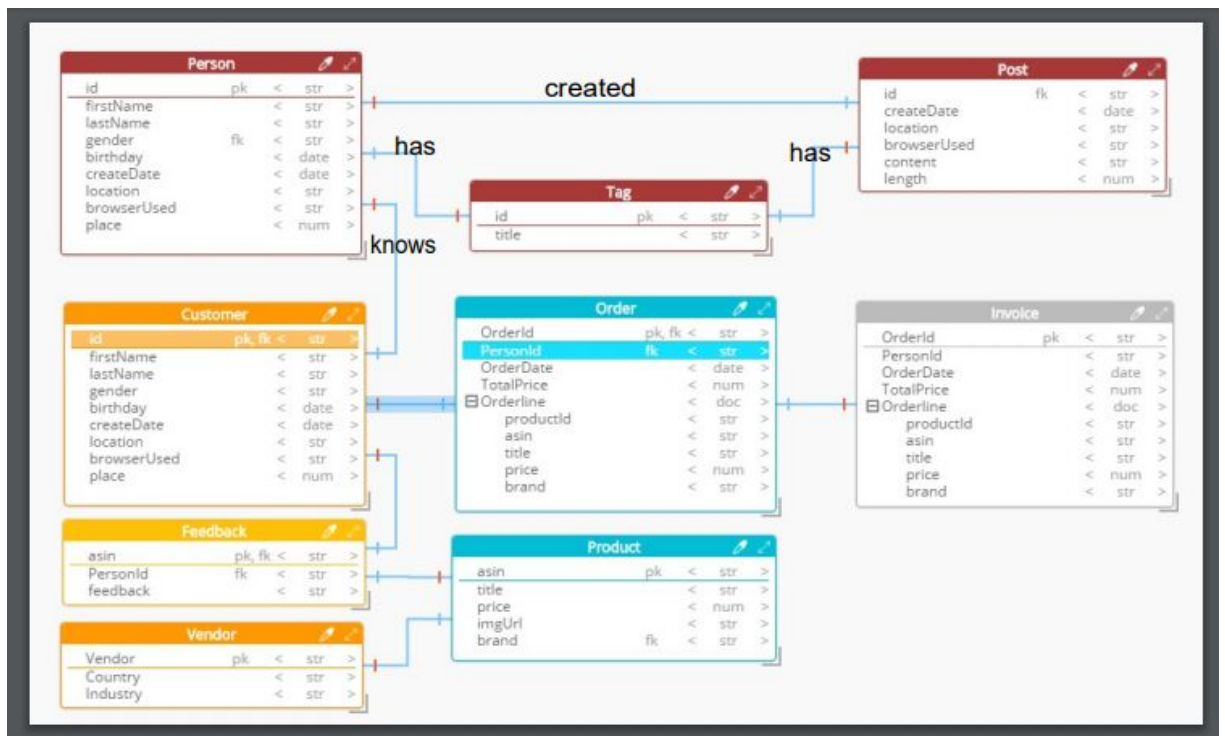
1.1 Schéma de données

Le schéma de données ci-dessous de vente de produits est un schéma multi-modèle. Il y a là les modèles suivants :

- Le modèle clé-Valeur pour gérer les FeedBack,
- Le modèle relationnel pour gérer Vendors et Customers
- Le modèle XML pour gérer Invoices (ce sera pour vous un modèle orienté colonnes)
- Le modèle JSON (documents) pour gérer Products et Orders
- Le modèle Graph pour gérer Person, Tag et Post



Le but initial de cette application était de gérer chaque modèle avec le moteur BD qui va. Le but qui vous est assigné ici est de gérer chaque modèle avec un seul moteur NoSQL que vous aurez choisi. Si le SGBD NoSQL supporte tous ces modèles, alors utilisez les. Si un modèle n'est pas possible avec avec votre SGBD NoSQL alors convertissez en ce qui est possible avec votre SGBD.



Le schéma ci-dessus représente le détail des structures par modèle. Faites attention, les cardinalités n'apparaissent pas. Vous devez en tenir compte.

L'ensemble du projet est disponible sur jalon :

Nom du fichier à télécharger : ?

1.2 Les données de l'étude

Les données pour chaque modèle sont disponibles dans le dossier NoSQL\data du zip à télécharger sur jalon.

1.3 Liste des moteurs NoSql à choisir

Moteurs Clé-Valeur : REDIS, Amazon DynamoDB, Microsoft Azur CosmosDB, RIAK

Moteurs orientés colonnes : Cassandra, Hbase, Big Table de Google, Microsoft Azur Table store

Moteurs orientés documents : CouchBase, CouchDB, OrientDB, MarkLogic

Moteurs orientés graphes : Neo4J, ArangoDB, Virtuoso, Giraph

2. Identité du moteur NoSql

Vous devez présenter sur quelques lignes chaque item ci-dessous du moteur nosql que vous aurez à évaluer.

2.2 Editeur

Azure Cosmos DB est un moteur NoSql développé et distribué par Microsoft. Il a été lancé en mai 2017, en étant intégré directement aux solutions Azure afin de venir étoffer les PaaS existantes.

2.3 Version initiale (Nr. De version et date)

Cosmos DB est sorti en version 1.0 en mai 2017.

2.4 Version actuelle (Nr. De version et date)

Microsoft ne met pas à disposition de numéro de version actuel de CosmosDB.

2.5 Modèles de données supportés(Clé/Valeur, Orienté document, Orienté Colonnes, Orienté Graphe)

Azure Cosmos DB est multi-modèle, lors de la création d'un projet on est libre de choisir le modèle de données que l'on veut. Il propose plusieurs API en fonction du modèle que l'on veut utiliser.

Un modèle est associé à une base de données lors de sa création (donc pas de multi-modèle dynamique).

2.6 Gestion du schéma (sans schéma, dynamique, statique, mixte)

Cosmos DB fonctionne sans schéma.

2.7 Support de SQL (DDL, DML)

Tous les modèles supportent les opérations DDL et DML.

2.8 Support des indexes secondaires

Cosmos DB indexe automatiquement, il n'a pas besoin d'index secondaire.

2.9 Langage de développement du sgbd nosql

Le langage dans lequel a été développé Cosmos DB n'est pas connu.

2.10 Support / Pérennité (communauté , etc....)

D'après le site db-engines.com, Azure Cosmos DB est 26^{ème} au classement des SGBD et 4^{ème} au classement des SGBD orientés document et clé / valeur. C'est donc un SGBD très populaire avec une grosse communauté.

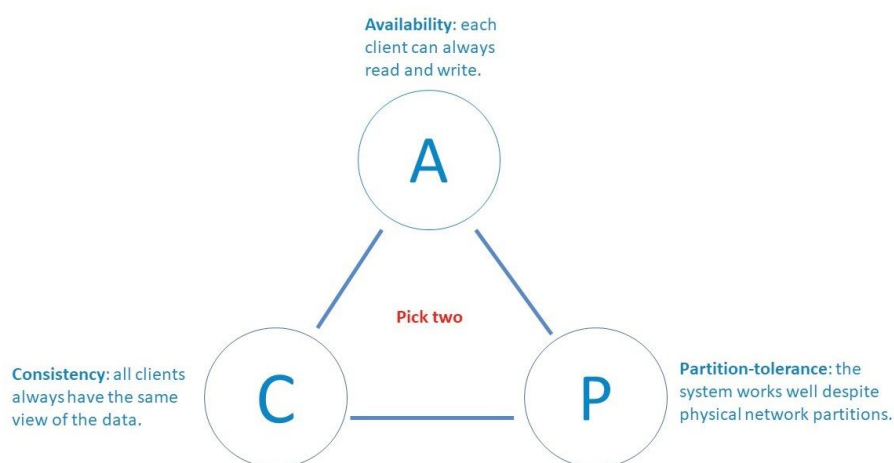
De plus, Microsoft fournit une documentation très complète en français et en anglais.

2.11 API Supportés

Cosmos DB supporte 5 apis:

- API SQL : pour utiliser SQL;
- API Gremlin : pour créer une base de données orientées graphe;
- API Mongo DB : utiliser les modèle document avec Cosmos DB ;
- API Table : pour la migration de données vers la version premium de Azure Cosmos DB;
- API Cassandra : migrer des données de Cassandra vers Azure Comos DB.

2.12 Théorème CAP (CP, AP, AC)



Cosmos DB permet grâce à des options de choisir en fonction de nos besoins CP, AP ou AC. Pour une base de données en Key-value, la configuration par défaut est CP (consistency & partition tolerance)

2.13 Méthode de partitionnement

Azure Cosmos DB utilise le partitionnement à l'échelle avec des conteneurs individuels. Ils sont gérés en fonction des besoins de performances.

Un conteneur contient des éléments qui sont divisés en sous ensemble appelés « partitions logiques ». Les éléments sont réparties dans les partitions logiques en fonction d'une clé de partition associé à chaque élément.

Par exemple, chaque élément à une valeur unique pour la propriété « UserID ». Si « UserID » est utilisé en tant que clé de partitionnement alors une partition logique sera créée pour chaque utilisateur.

2.14 Méthode de réplication

Azure Cosmos DB déplace automatiquement les partitions logiques d'un conteneur à l'autre et d'un serveur à l'autre pour répartir la charge sur le plus de serveurs possible.

Il y a des contraintes à prendre en compte lors du choix d'une clé de partition :

- une partition logique ne doit pas dépasser 10Go ;
- un conteneur a un débit maximum de 400 requêtes par seconde qu'il doit répartir sur ces différentes partitions logiques.

2.15 Concept de consistance

Les BD Azure Cosmos DB ont 2 types de consistance. la consistance forte et la consistance éventuelle

- la consistance forte permet une programmabilité des données élevée. Elle a cependant pour conséquence d'augmenter la latence dans un état stable mais aussi de diminuer la disponibilité des données en cas de panne
- la consistance éventuelle rend plus difficile la programmation avec les données. cependant elle réduit la latence et rend les données disponibles à tout moment.

2.16 Concept de durabilité

Le concept de durabilité est pris en charge par Azure Cosmos DB

Azure Cosmos DB assure la durabilité de ses instances, et l'améliore en fonction de la réplication des données par région, avec une durabilité plus importante et un temps de redémarrage de l'application plus rapide pour un nombre d'instances/régions plus importants : <https://docs.microsoft.com/en-us/azure/cosmos-db/consistency-levels-tradeoffs#rto>.

2.17 Clés étrangères

Puisque Cosmos DB supporte le SQL, il supporte également les clés étrangères.

2.18 Support de références (REF)

Les bases de données Azure Cosmos DB peuvent être configurés de telle sorte qu'elle support les requêtes SQL. Le langage SQL supporte la notion de Référence. Par conséquent, les bases de données Azur Cosmos DB supporte également les références.

2.19 Licences et prix

Azure Cosmos DB facture à la consommation, les tarifs sont de :

- 0,008\$/heure les 100 requêtes par seconde ;
- 0,250\$ le Go de stockage.

A titre d'exemple, si on souhaite avoir un débit de 400 requêtes par seconde pendant un mois (730h pour un mois de 30 jours) et 10 Go on devrait payer 25,86\$:

Débit des requêtes : $4(* 100 \text{ requêtes/seconde}) * 730 \text{ (heures)} * 0,008(\text{taux horaire}) = 23,36\$$

Stockage : $10(\text{Go}) * 0,250(\text{prix du Go/Mois}) = 2,50\$$

Total = $23,36 + 2,50 = 25,86\$$

2.20 Différents types de versions (communautaire, entreprise, ...)

Deux versions sont disponibles. Les tarifs varient en fonction du débit approvisionné (exprimé en unité de requête par seconde, RU/s) et du stockage SSD local consommé (facturé en Go).

Région : Devise: Afficher la tarification par :

Capacité de réserve pour le débit approvisionné

Économisez jusqu'à 65 % sur les coûts et bénéficiez de contrats de niveau de service garantissant une disponibilité améliorée, tout en allégeant la charge de planification de la capacité avec la tarification de la capacité de réserve d'Azure Cosmos DB. Réservez un débit approvisionné pendant un ou trois ans avec un paiement unique, et partagez-le entre toutes les régions, API, comptes et abonnements liés à une inscription donnée. Pour en savoir plus, voir la page de [documentation](#).

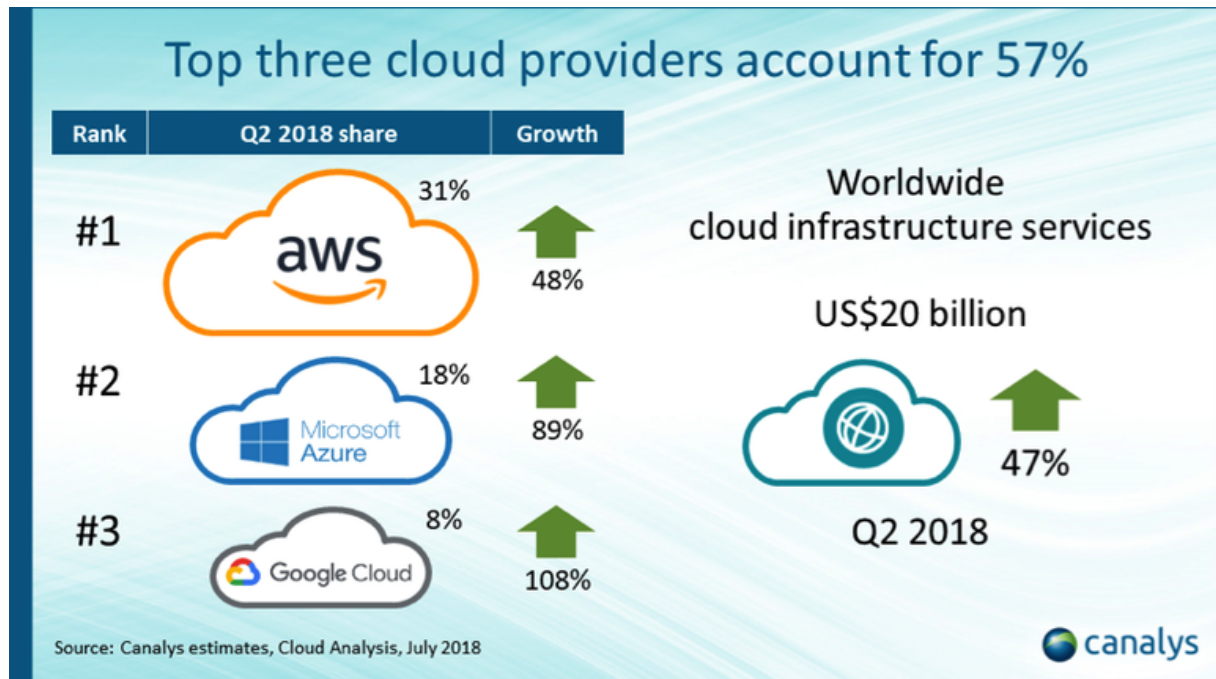
DÉBIT	RÉSERVATION D'1 AN		RÉSERVATION DE 3 ANS	
	ÉCRITURE DANS UNE SEULE RÉGION	ÉCRITURE DANS PLUSIEURS RÉGIONS	ÉCRITURE DANS UNE SEULE RÉGION	ÉCRITURE DANS PLUSIEURS RÉGIONS
TARIF/ÉCONOMIES	TARIF POUR 100 RU/S (ÉCONOMIES SUR LE PAIEMENT À L'UTILISATION)	TARIF POUR 100 RU/S (ÉCONOMIES SUR LE PAIEMENT À L'UTILISATION)	TARIF POUR 100 RU/S (ÉCONOMIES SUR LE PAIEMENT À L'UTILISATION)	TARIF POUR 100 RU/S (ÉCONOMIES SUR LE PAIEMENT À L'UTILISATION)
50K premières RU/s	0,0058 € (~48%)	0,0108 € (~51%)	0,0051 € (~54%)	0,0095 € (~57%)
450 000 unités de requête/s suivantes	0,0051 € (~54%)	0,0095 € (~57%)	0,0044 € (~60%)	0,0081 € (~63%)
2 500 000 RU/s par la suite	0,0048 € (~57%)	0,0088 € (~60%)	0,0038 € (~66%)	0,0068 € (~69%)
Plus de 3 000 000 de RU/s	0,0038 € (~66%)	0,0068 € (~69%)	0,0027 € (~75%)	0,0048 € (~78%)

Tout débit que vous fournissez en excédent de votre capacité de réserve est facturé selon le tarif de débit standard approvisionné.

2.21 Audience dans le marché

Microsoft est positionné comme 2ème fournisseur de PaaS, faisant de ses solutions Azure des outils utilisés par un grand nombre de clients à travers le monde.

CosmosDB en elle-même est 26ème sur le [classement de popularité de db-engines](#), avec un bien meilleur classement en termes de modèles NoSQL (voir ci-dessous).



Source : <https://www.lebigdata.fr/microsoft-azure-parts-marche-cloud>

Score 27.59

Rank #26 Overall

#4 Document stores

#2 Graph DBMS

#4 Key-value stores

#3 Wide column stores

Source : <https://db-engines.com/en/ranking>

2.22 Tables et tables filles

L'héritage entre les tables n'est pas assuré avec Azures Cosmos DB.

2.23 Clé avec major et minor key

Azure Cosmos DB supporte le principe de clé avec major et minor key. Dans la base de données Cosmos DB Key-Value (Azure Table API), nous disposons de “partition keys” et “row keys”, qui agissent ici respectivement comme des “major keys” et “minor keys”.

2.24 Gestion des utilisateurs

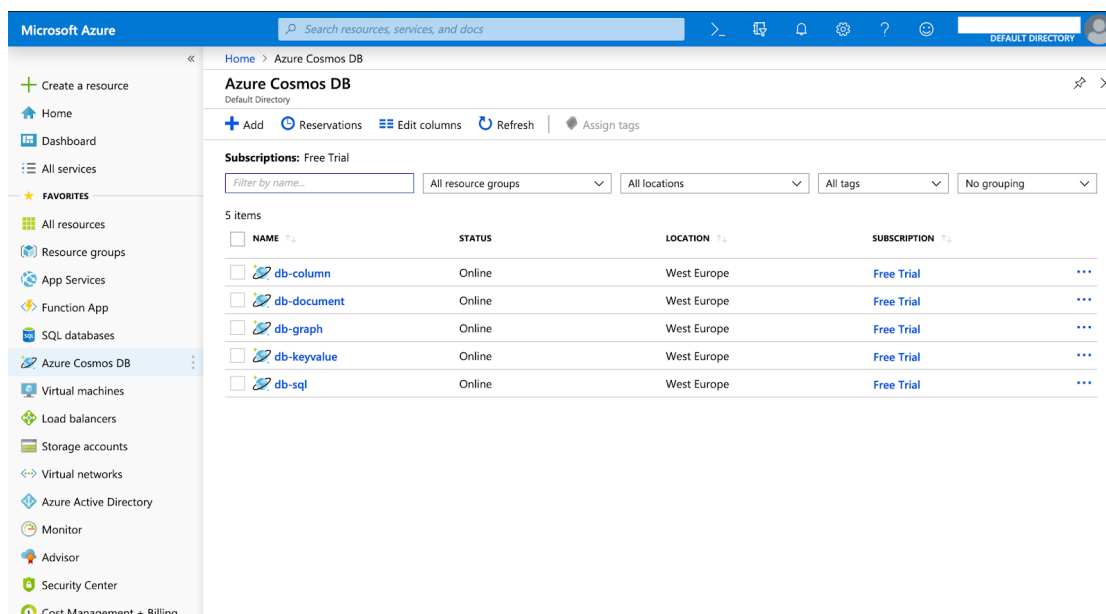
Il est possible de partager l'accès à chaque base de données à des utilisateurs d'un Azure Active Directory. Chaque utilisateur peut avoir des droits différents, et il existe des rôles prédéfinis permettant de donner plus ou moins de liberté aux utilisateurs.

2.25 Gestion des droits

Il est possible de définir pour tous les utilisateurs les droits qu'ils ont pour chaque ressource. Elles peuvent être établies par une de nombreuses règles prédéfinies dans Azure Cosmos DB. En somme, il suffit d'ajouter un utilisateur à la ressource Cosmos DB en lui assignant un rôle sur celle-ci, après avoir ajouté l'utilisateur à l'Active Directory correspondant.

2.26 Gestion des namespaces ou databases

Sur Cosmos DB, la gestion des databases se fait par le biais d'une interface web depuis portal.azure.com



The screenshot shows the Azure portal interface for managing Azure Cosmos DB resources. The left sidebar contains a navigation menu with options like 'Create a resource', 'Home', 'Dashboard', 'All services', and 'FAVORITES'. The 'FAVORITES' section lists various services, with 'Azure Cosmos DB' currently selected. The main content area displays the 'Azure Cosmos DB' resource page, which includes a table of 5 items. The table columns are 'NAME', 'STATUS', 'LOCATION', and 'SUBSCRIPTION'. The items listed are 'db-column', 'db-document', 'db-graph', 'db-keyvalue', and 'db-sql', all with a status of 'Online' and located in 'West Europe'. Each item has a 'Free Trial' subscription. The top bar of the portal shows the 'Microsoft Azure' logo, a search bar, and a user profile icon.

NAME	STATUS	LOCATION	SUBSCRIPTION
db-column	Online	West Europe	Free Trial
db-document	Online	West Europe	Free Trial
db-graph	Online	West Europe	Free Trial
db-keyvalue	Online	West Europe	Free Trial
db-sql	Online	West Europe	Free Trial

2.27 Systèmes d'exploitations supportés

Accessible en web, Azure Cosmos DB est hébergé dans les serveurs de Microsoft, il n'est pas possible de l'héberger sur ses serveurs ou en local.

2.28 Disponible en mode DBaaS

Azure Cosmos DB n'est utilisable qu'en version cloud, donc en PaaS/DBaaS (Platform/DataBase as a Service).

2.29 Support du Map/Reduce

La fonctionnalité Map/Reduce est prise en charge par le moteur NoSQL Azure Cosmos DB. Elle assure notamment en partie le bon fonctionnement du scaling PaaS/Cloud de Cosmos DB, puisqu'elle permet de distribuer la charge de données entre les instances parallèles qui peuvent se créer au besoin lorsque le nombre de requête ou la masse de données à traiter sont importants.

2.30 Lien vers la documentation technique y compris les API

<https://docs.microsoft.com/en-us/azure/cosmos-db/>

2.31 Typage (none, static, dynamique)

Azure Cosmos DB supporte le typage des données. Le typage est statique.

2.32 Applications communautaires l'utilisant

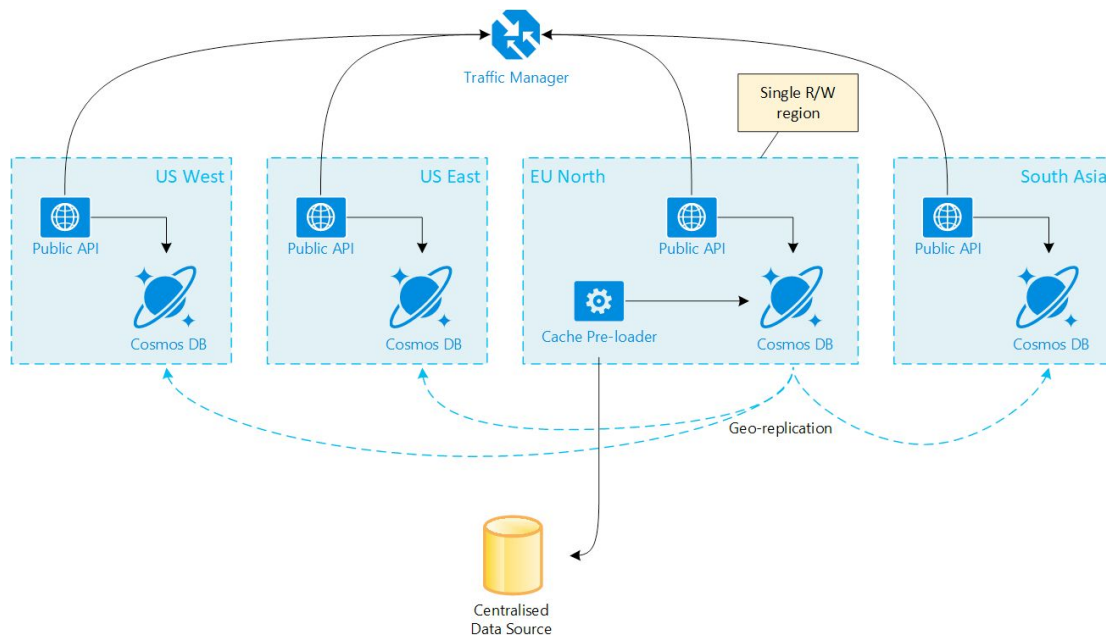
Plusieurs applications avec de nombreux utilisateurs utilisent Azure Cosmos DB:

- Asos.com: site de vente en ligne de vêtements avec plus de 15 millions d'utilisateurs à travers le monde
- Jet.com: Site de e-commerce appartenant au groupe Walmart
- American Cancer Society

2.33 Domaines d'applications

Azure Cosmos DB est fait pour être utilisé dans des applications web, mobile, gaming et IOT qui ont besoins d'être scalable automatiquement, d'avoir des performances stables, de temps de réponse de l'ordre des millisecondes et d'avoir des données sans schéma.

2.34 Architecture du moteur NoSql



2.35 Montée en charge

Un des avantages de Azure Cosmos DB est la possibilité de pouvoir monter en charge aisément. Du fait, ce système de répartition qui est liée au stockage et au trafic utilisé, l'utilisateur peut faire évoluer comme il le souhaite le débit ou le stockage dans un ou plusieurs régions qu'il possède.

2.36 Gestion de la disponibilité

Dans un BD Azure Cosmos DB, les données sont disponibles à 99.99%. Cette haute disponibilité est due à toutes les répliques que les BD Azure Cosmos DB effectuent dans ces différentes régions. De ce fait, si un serveur tombe en panne, on peut récupérer les données sur un autre serveur sur lequel les données ont été répliquées.

2.37 Procédure d'installation

L'utilisation de Azure Cosmos DB ne nécessite pas d'installation particulière. En effet, une plateforme en ligne est disponible afin d'accéder à la BD. Pour une utilisation en programmation, l'installation de l'API correspondante est nécessaire.

Cosmos DB propose également des projets d'exemples dans plusieurs langages en intégrant les endpoints/connection strings nécessaires, qui permettent aux utilisateurs de se lancer facilement.

3. Modélisation et chargement des données

L'ensemble du chargement des données a été effectué au sein de notre programme Java, afin de gérer de manière flexible les particularités des formats .csv, .json et .xml.

La classe "LoadDataFileMain" s'occupe d'appeler les diverses méthodes d'import des données, en reliant nos utilitaires de lecture de fichiers à nos DAOs pour la création d'objets.

4. Mises à jours des données (insert, update, delete)

Nous nous sommes orientés vers une gestion DAO (Data Access Object) afin de permettre une manipulation précise et générique des objets de nos bases de données NoSQL.

Chaque type de donnée dispose d'un modèle et d'un controller associé, afin de respecter les bonnes pratiques de développement objet.

4.2 Maj de données partie Key/value

La manipulation de données Key-Value (Feedback) peut être testée dans la classe "KeyValueMain" qui utilise l'API Azure Table.

4.3 Maj de données partie Document

La manipulation de données Document (Product, Order) peut être testée dans la classe "DocumentMain" qui utilise l'API MongoDB.

4.4 Maj de données partie Orientée colonnes

La manipulation de données Column (Invoice) peut être testée dans la classe "ColumnMain" qui utilise l'API Cassandra.

4.5 Maj de données partie Relationnelles

La manipulation de données Relationnelles peut être testée dans la classe "RelationalMain" qui utilise l'API SQL d'Azure.

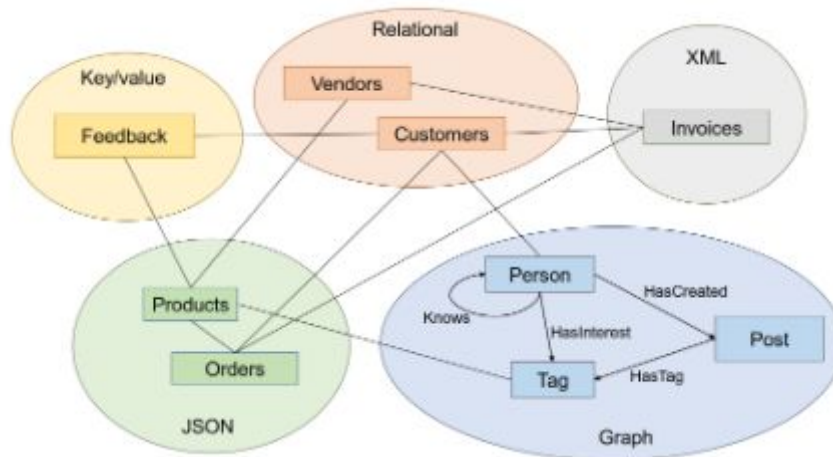
4.6 Maj de données partie Graphes

La manipulation de données Graph peut être testée dans la classe “GraphMain” qui utilise l’API Gremlin.

5. Interrogation des données

Ecrire les programmes java qui répondent aux questions ci-dessous.

Rappel de l’infrastructure de données :



- Query 1. For a given customer, find his/her all related data including profile, orders, invoices, feedback, comments, and posts in the last month, return the category in which he/she has bought the largest number of products, and return the tag which he/she has engaged the greatest times in the posts.
- Query 2. For a given product during a given period, find the people who commented or posted on it, and had bought it. (=4 tables)
- Query 3. For a given product during a given period, find people who have undertaken activities related to it, e.g., posts, comments, and review, and return sentences from these texts that contain negative sentiments.
- Query 4. Find the top-2 persons who spend the highest amount of money in orders. Then for each person, traverse her knows-graph with 3-hop to find the friends, and finally return the common friends of these two persons.
- Query 5. Given a start customer and a product category, find persons who are this customer's friends within 3-hop friendships in Knows graph, besides, they have bought

products in the given category. Finally, return feedback with the 5-rating review of those bought products.

- Query 6. Given customer 1 and customer 2, find persons in the shortest path between them in the subgraph, and return the TOP 3 best sellers from all these persons' purchases.
- Query 7. For the products of a given vendor with declining sales compare to the former quarter, analyze the reviews for these items to see if there are any negative sentiments.
- Query 8. For all the products of a given category during a given year, compute its total sales amount, and measure its popularity in the social media.
- Query 9. Find top-3 companies who have the largest amount of sales at one country, for each company, compare the number of the male and female customers, and return the most recent posts of them.
- Query 10. Find the top-10 most active persons by aggregating the posts during the last year, then calculate their RFM (Recency, Frequency, Monetary) value in the same period, and return their recent reviews and tags of interest.

6. Les résultats de l'étude

- Rapport d'identification de votre moteur Nosql (remplir pour le cela le chapitre 2 de ce document)
- Les programmes et scripts de chargement de données (remplir chapitre 3). Joindre les scripts et/ou les programmes dans des fichiers textes
- Les programmes et scripts de mise à jour de données (remplir le chapitre 4). Joindre les programmes et les scripts dans des fichiers textes
- Les programmes de consultation des données (remplir le chapitre 5), joindre les programmes dans des fichiers textes

7. Répartition du travail

- 5 Membres par groupe (activités par membre): chap. 2 (7 propriétés), chap. 3 (chargement des données d'1 modèle), chap. 4 (mise à jour des données d'1 modèle), chap. 5 (2 requêtes).
- Vous devez lors de la restitution identifier ce que chaque membre a fait

8. Où s'inscrire pour constituer son groupe de 5 et choisir son SGBD NoSql

- Voici le lien pour s'inscrire et choisir un moteur nosql :
https://docs.google.com/spreadsheets/d/1DKc54OqkBICPKrG39drzGs_W-eVTa-U-Ye-I5rWIYG8/edit?usp=sharing
- Un moteur nosql ne peut pas être choisi par deux groupes.