DATA STORAGE AND MANAGEMENT STRATEGY

Project 4

SEPT 28TH, 2024
ANKIT BHARTI
(CS 651 03) Cloud Computing & Big Data Analytic
Steven Thomas

Introduction

As I continued with the process of this task, I realized that an excellent and easily manageable storage of data is very essential and is in fact a necessity when it comes to the data types I work with, and specifically the activities at this firm – popularly known as the Samsung consumer electronics. However, for the way this component of the project was framed, I propose to harness NoSQL as my software of choice in resource provisioning and job execution. This is because, such unstructured data as customer reviews, product logs as well as transaction data, tends to have huge volumes of data and a lot of it is not chronologically structured data

Database Technology Selection

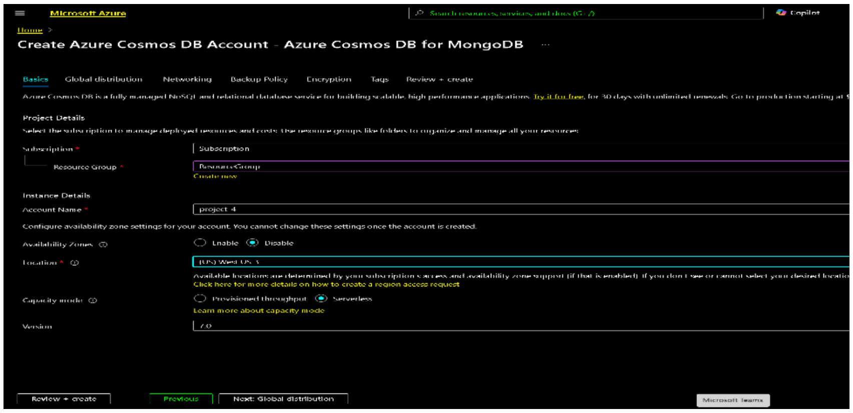
I evaluated various parameters such as the data type, system access frequency, and potential expansion of usage. The incoming data essentially comprises two sources; that is, online reviews left by our customers, logs generated upon product usage or service, as well as actual orders from customers, which can be characterized as semi-structured data. Such kind of data does not fit well in the scope of the traditional relational databases like SQLite and the relevance of NoSQL databases come to play as it can accommodate more variations of data structures due to lack of schemas.

In this case, nothing but Azure Cosmos DB, a NoSQL database as a service (DBaaS) of Microsoft, was my ultimate choice. Cosmos DB provides several unique qualifications:

- It can accommodate multiple database models like document-oriented, key-value pair, and even graph databases.
- It provides prompt data access and availability across the globe, that is very important here at Samsung, which has quite a few installations across the world.
- Azure Cosmos DB can work with other services in the Azure platform which I strongly depend on in this study and this assist in the flow of data and analytics.

Implementation in Cloud Environment

Setting Up Azure Cosmos DB: I set up Azure Cosmos DB to provide the solution. Setting up a Cosmos DB account was the first task. I referred to MongoDB API since it is good for non-structured data like product logs and reviews.



Create Azure Cosmos DB Account - Azure Cosmos DB for MongoDB

Azure Cosmos DB is a fully managed NoSQL and relational database service for building scalable, high performance applications. Try it for free, for 30 days with unlimited renewals. Go to production starting at $

Project Details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * Subscription

Resource Group * ResourceGroup

Create new

Instance Details

Account Name * project 4

Configure availability zone settings for your account. You cannot change these settings once the account is created.

Availability Zones ○ Enable ● Disable

Location * (US) West US 3

Available locations are determined by your subscription's access and availability zone support (if that is enabled). If you don't see or cannot select your desired locatio

Click here for more details on how to create a region access request

Capacity mode ○ Provisioned throughput ● Serverless

Learn more about capacity mode

Version 7.0

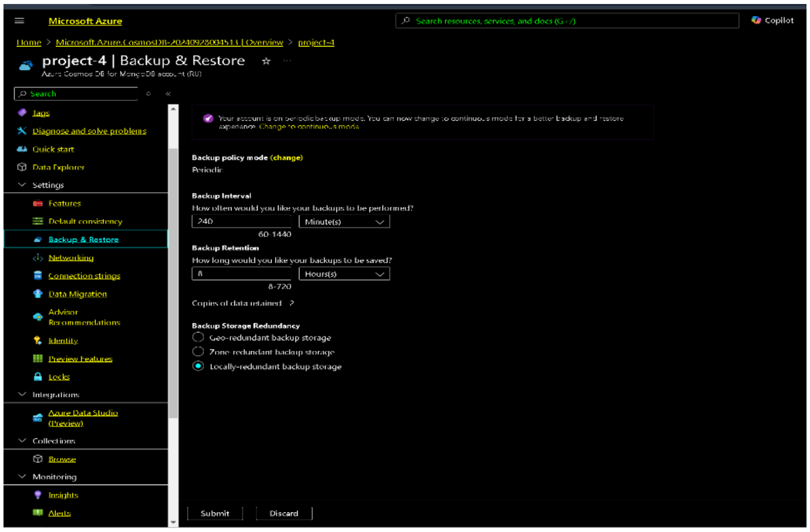
After achieving the previous step, I generated the folders where the data is going to be stored. I organized data in folders such as User-reviews, E-commerce transaction logs and User-activities. The main aim of the folders is for ease of referencing when searching for data and the ability to efficiently query the data later. I went ahead and set up automatic scaling of such that the system can easily accommodate heavier loads of data as it grows.

Efficient Data Storage and Retrieval: The speed at which such data is delivered is critical because queries can be automatically translated and returned to nature using state of Cosmos DB and its indexing strategies. Among these strategies, the automatic indexing has been of much help in handling complicated queries without the need for manual handling. Any form of index is very important but, in a case, where more data is used then more types of indexes will be needed. This is where sharding, which is the partitioning of data across multiple servers, comes in, to allow large datasets to be handled without or with higher performance during both read and write operations.

Moreover, I ensured that appropriate consistency levels were implemented for the given data. Most of the time, I settled for Session Consistency because it tends to be good enough without sacrificing performance or poles integrity compromising the data.

Database Management and Optimization

Backup and Recovery Strategy: A robust backup and data recovery strategy was put into place given the importance of the data's integrity and the potential risks involved in losing any of it. This involved leveraging, among other tools, Azure's in-built ones. The native tool provided by Microsoft offers a simple way to automate backing up of the data which is present in the form of Cosmo DB collections instituted within a schedule. The service also provides a pain-free recovery advantage with the ability to restore the database to a certain point without having to go undergo complications following data loss or data corruption. More so, with Cosmos Db and its geo-replication feature, the usual practice of disaster recovery is being advanced using another level of data replication across different regions.



project-4 | Backup & Restore

Azure Cosmos DB for MongoDB account (RU)

Your account is on periodic backup mode. You can now change to continuous mode for a better backup and restore experience. Change to continuous mode

Backup policy mode (change)

Periodic

Backup Interval

How often would you like your backups to be performed?

240 | Minute(s)

60-1440

Backup Retention

How long would you like your backups to be saved?

8 | Hours(s)

8-720

Copies of data retained 2

Backup Storage Redundancy

○ Geo-redundant backup storage

○ Zone-redundant backup storage

● Locally-redundant backup storage

Monitoring and Maintenance: For proper system care practices, I activated Azure Monitor and started to keep an eye on certain aspects, for instance, handling performance improvement on the DB. Among the key factors Azure Monitor helps me to track are the system's utilization of the CPU unit, query speed, and additional capabilities that do not necessarily relate to storage capacity. This can be seen in the fact that in case the time taken to execute a query is more than a given threshold, one can decide whether to restructure and optimize the queries or rather reconfigure the partitioning approach.

Conclusion

It is, therefore, reasonable to assert that the employment of Azure Cosmos DB for data storage in this report is both relevant and effective. The solution's ability to grow without constraints and integrate seamlessly with the Microsoft Azure ecosystem makes it the best choice for managing the large and highly varied datasets assigned to Samsung Corporation. Through common practices, such as scaling out, creating indexes, distributing data over different servers, and taking backup, I managed to setup an effective database system that offers efficient data storage and retrieval, and takes care of problems associated to data inconsistency and lack of access to data. This information storage and management system, by virtue of being based on NoSQL, is especially good for customer feedback, device logs and sales transaction details which involve ever changing record sets enhancing prospects for further implementation of the project.

Citations

1. Leavitt, N. (2010). Will NoSQL databases live up to their promise? *Computer*, 43(2), 12-14. https://doi.org/10.1109/MC.2010.58

2. Microsoft Azure. (n.d.). Azure Cosmos DB: A globally distributed, multi-model database service. Retrieved from https://azure.microsoft.com/en-us/services/cosmos-db/