Giới thiệu về Azure Cosmos DB

Azure Cosmos DB là hệ quản trị cơ sở dữ liệu NoSQL được cung cấp bởi Microsoft. Nó có khả năng tương thích với một số hệ thống NoSQL phổ biến hiện nay như:

MongoDB: Hệ thống NoSQL được sử dụng rộng rãi cho các ứng dụng web và di động.

Cassandra: Hệ thống NoSQL được sử dụng cho các ứng dụng phân tán lớn.

Dưới đây là một số điểm nổi bật của Azure Cosmos DB:

Khả năng mở rộng: Cosmos DB có khả năng mở rộng cao, có thể đáp ứng nhu cầu của các ứng dụng có lượng dữ liệu lớn và truy cập cao.

Hiệu suất cao: Cosmos DB cung cấp hiệu suất truy vấn nhanh chóng và ổn định.

Tính sẵn sàng cao: Cosmos DB đảm bảo tính sẵn sàng cao cho dữ liệu của bạn với khả năng chịu lỗi và tự phục hồi.

Bảo mật: Cosmos DB cung cấp các tính năng bảo mật mạnh mẽ để bảo vệ dữ liệu của bạn.

Dễ sử dụng: Cosmos DB cung cấp nhiều giao diện lập trình ứng dụng (API) dễ sử dụng, giúp bạn dễ dàng truy cập và quản lý dữ liệu.

Ngoài ra, Cosmos DB còn có một số tính năng khác như:

Hỗ trợ nhiều mô hình dữ liệu: Cosmos DB hỗ trợ nhiều mô hình dữ liệu khác nhau như tài liệu, bảng, key-value và đồ thị.

Tích hợp với các dịch vụ Azure khác: Cosmos DB có thể tích hợp dễ dàng với các dịch vụ Azure khác như Azure Data Factory, Azure Machine Learning và Azure Stream Analytics.

Cosmos DB lưu trữ dữ liệu dưới dạng các tài liệu (document) được biểu diễn bằng JSON (JavaScript Object Notation). Các tài liệu này được tổ chức thành các nhóm gọi là container.

Dưới đây là một số điểm chính về cách thức lưu trữ dữ liệu trong Cosmos DB:

Tài liệu: Một tài liệu là một tập hợp các trường được xác định bởi một khóa. Các trường trong mỗi tài liệu có thể khác nhau và một trường có thể chứa các tài liệu con.

JSON: JSON là một định dạng dữ liệu nhẹ và dễ đọc được sử dụng để biểu diễn các tài liệu trong Cosmos DB.

Container: Container là một nhóm các tài liệu được tổ chức cùng nhau. Container có thể được chia thành các phân vùng để cải thiện hiệu suất truy vấn.

Phân vùng: Phân vùng là một cách chia dữ liệu trong container thành các nhóm nhỏ hơn. Phân vùng được thực hiện dựa trên khóa phân vùng, là một trường trong tài liệu.

Bài viết này sẽ giới thiệu một số công cụ phổ biến để tải dữ liệu vào Cosmos DB, bao gồm:

Công cụ Data Migration:

Công dụng: Dành cho việc di chuyển dữ liệu từ các hệ thống khác sang Cosmos DB.

Ví dụ: Di chuyển dữ liệu từ MongoDB sang Cosmos DB.

Ưu điểm: Dễ sử dụng, hỗ trợ nhiều nguồn dữ liệu.

Nhược điểm: Ít tùy chỉnh hơn các công cụ khác.

Azure Data Factory:

Công dụng: Dành cho việc xây dựng các quy trình ETL (Extract, Transform, Load) phức tạp.

Ví dụ: Tải dữ liệu từ nhiều nguồn khác nhau vào Cosmos DB, sau đó thực hiện các thao tác chuyển đổi dữ liệu trước khi lưu trữ.

Ưu điểm: Mạnh mẽ, linh hoạt, có thể kết hợp với nhiều dịch vụ Azure khác.

Nhược điểm: Cấu hình phức tạp hơn các công cụ khác.

Thư viện BulkExecutor:

Công dụng: Dành cho việc tải dữ liệu ồ ạt vào Cosmos DB bằng cách viết mã ứng dụng.

Ví dụ: Viết ứng dụng C# để tải dữ liệu từ file CSV vào Cosmos DB.

Ưu điểm: Hiệu suất cao, linh hoạt.

Nhược điểm: Yêu cầu kiến thức lập trình.

Data Explorer:

Công dụng: Dành cho việc tải dữ liệu thủ công vào Cosmos DB và thực hiện các truy vấn thử nghiệm.

Ví dụ: Tải một vài tài liệu JSON vào Cosmos DB để kiểm tra cấu hình.

Ưu điểm: Dễ sử dụng, trực quan.

Nhược điểm: Không phù hợp cho việc tải dữ liệu ồ ạt.

Lựa chọn công cụ nào phù hợp với bạn phụ thuộc vào một số yếu tố:

Lượng dữ liệu bạn cần tải.

Mức độ phức tạp của quy trình tải dữ liệu.

Kỹ năng và kinh nghiệm của bạn với các công cụ khác nhau.

Lỗi HTTP 429 xảy ra khi bạn ghi dữ liệu vào container Cosmos DB với tốc độ vượt quá dung lượng thông lượng (throughput) được cấu hình cho container đó. Lỗi này có thể xảy ra khi bạn:

Tải dữ liệu ồ ạt vào Cosmos DB.

Thực hiện nhiều thao tác ghi dữ liệu đồng thời.

Để khắc phục lỗi HTTP 429, bạn cần thực hiện các bước sau:

Xác định container đang gặp lỗi.

Kiểm tra dung lượng thông lượng hiện tại của container.

Tăng dung lượng thông lượng cho container.

Giả sử bạn có một container Cosmos DB lưu trữ thông tin về khách hàng. Bạn có thể phân vùng dữ liệu dựa trên quốc gia của khách hàng.

Khóa phân vùng trong trường hợp này sẽ là trường "country".

Lợi ích của việc phân vùng:

Cải thiện hiệu suất truy vấn: Khi dữ liệu được phân vùng, Cosmos DB có thể chỉ truy cập các phân vùng có chứa dữ liệu liên quan đến truy vấn của bạn. Điều này giúp cải thiện hiệu suất truy vấn.

Tăng khả năng mở rộng: Phân vùng cho phép bạn chia dữ liệu thành các nhóm nhỏ hơn, giúp Cosmos DB có thể mở rộng để đáp ứng nhu cầu lưu trữ dữ liệu ngày càng tăng.

Cải thiện tính sẵn sàng: Khi dữ liệu được phân vùng, việc mất một phân vùng sẽ không ảnh hưởng đến các phân vùng khác. Điều này giúp cải thiện tính sẵn sàng của dữ liệu.

Lưu ý:

Việc lựa chọn khóa phân vùng là rất quan trọng. Khóa phân vùng nên được chọn dựa trên cách thức truy vấn dữ liệu của bạn.

Bạn có thể thay đổi khóa phân vùng của container sau khi tạo.

Nội dung chính:

Mục đích chính của các API Table, MongoDB và Cassandra là hỗ trợ các ứng dụng hiện có được viết bằng chính những hệ quản trị cơ sở dữ liệu NoSQL đó.

Nếu bạn đang xây dựng một ứng dụng và cơ sở dữ liệu mới, nên sử dụng SQL API hoặc Gremlin API.

Cosmos DB cung cấp nhiều lựa chọn để tải dữ liệu lên và truy vấn dữ liệu trong cơ sở dữ liệu Cosmos DB.

Giải thích chi tiết:

Cosmos DB hỗ trợ nhiều API khác nhau để truy vấn và thao tác với dữ liệu.

Ba API Table, MongoDB và Cassandra được thiết kế để tương thích với cú pháp truy vấn của các hệ quản trị cơ sở dữ liệu NoSQL phổ biến cùng tên. Điều này giúp việc di chuyển các ứng dụng hiện có sang Cosmos DB dễ dàng hơn, vì bạn không cần phải thay đổi nhiều về code truy vấn dữ liệu.

Tuy nhiên, nếu bạn đang xây dựng một ứng dụng mới, Microsoft khuyên bạn nên sử dụng SQL API (cho phép truy vấn theo cú pháp SQL quen thuộc) hoặc Gremlin API (dành cho dữ liệu dạng đồ thị) vì chúng được thiết kế riêng cho Cosmos DB và có thể tận dụng tối đa các tính năng của dịch vụ này.

Ngoài ra, đoạn văn còn cho biết Cosmos DB cung cấp nhiều công cụ để tải dữ liệu lên (như Data Migration, Azure Data Factory) và truy vấn dữ liệu (như SQL API, MongoDB API, ...).

Data Migration là một công cụ giúp bạn nhập dữ liệu từ nhiều nguồn khác nhau vào Cosmos DB.

Các nguồn dữ liệu hỗ trợ bao gồm: file JSON, MongoDB, SQL Server, file CSV, Azure Table Storage, Amazon DynamoDB, HBase và các container Cosmos khác.

Bạn có thể tải công cụ Data Migration từ GitHub.

Công cụ sẽ hướng dẫn bạn từng bước trong quá trình di chuyển dữ liệu vào cơ sở dữ liệu Cosmos DB.

Bạn cần cung cấp nguồn dữ liệu (chọn từ danh sách được liệt kê) và đích đến (là cơ sở dữ liệu và container Cosmos DB).

Data Migration có thể thêm dữ liệu vào container hiện có hoặc tạo một container mới nếu container bạn chọn chưa tồn tại.

Ngoài nhập dữ liệu, Data Migration cũng cho phép bạn xuất dữ liệu từ container Cosmos DB sang file JSON, lưu trữ cục bộ hoặc trên Azure Blob storage.

Đối với lượng dữ liệu lớn, Data Migration có thể sử dụng nhiều luồng đồng thời để phân đoạn dữ liệu thành các khối nhỏ và tải các khối này song song. Mỗi luồng hoạt động như một kết nối riêng biệt với cơ sở dữ liệu.

Tóm tắt: Data Migration là một công cụ hữu ích để di chuyển dữ liệu từ các nguồn khác nhau vào Cosmos DB, hỗ trợ cả nhập và xuất dữ liệu với khả năng xử lý khối lượng dữ liệu lớn.