BỘ CÔNG THƯƠNG

**TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP. HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

---------------------------



**ĐỒ ÁN MÔN HỌC**

**PHÂN TÍCH DỮ LIỆU DU LỊCH – KHÁCH SẠN**

**GVHD: Nguyễn Thành Ngô**

**2001222529 \_ Đinh Thị Thảo Ly**

**2001225805 \_ Nguyễn Thị Kiều Vân**

**2001221872 \_ Lê Trung Kiên**

*TP. HỒ CHÍ MINH, tháng 11 năm 2025*

BỘ CÔNG THƯƠNG

**TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP. HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

---------------------------



**ĐỒ ÁN MÔN HỌC**

**NOSQL – KHÁCH SẠN**

**GVHD: Nguyễn Thị Định**

**2001221445 \_ Trương Mỹ Hoa**

**2001220536\_ Cao Thiên Chi**

**2044226229\_ Võ Ngọc Nguyên Chương**

*TP. HỒ CHÍ MINH, tháng 11 năm 2025*

**BẢNG PHÂN CÔNG VIỆC NHÓM CHO CÁC THÀNH VIÊN**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **MSSV** | **HỌ TÊN** | **CÔNG VIỆC ĐƯỢC GIAO** | **ĐÁNH GIÁ** |
| 1 | 2044226229 | Võ Ngọc Nguyên Chương | * Biên soạn nội dung Chương 1, 2, 3, 4. * Hiệu chỉnh câu từ, mở rộng và hoàn thiện Chương 5. * Soạn thảo và trình bày báo cáo Word. * Hỗ trợ lập trình/code. * Thuyết trình phần Tổng quan đề tài và Dataset. | 100% – A |
| 2 | 2001221445 | Trương Mỹ Hoa | * Biên soạn nội dung Chương 6. * Chuẩn bị báo cáo PowerPoint. * Hỗ trợ lập trình/code. * Thuyết trình về Kỹ thuật và công nghệ sử dụng. | 100% – A |
| 3 | 2001220536 | Cao Thiên Chi | * Biên soạn nội dung Chương 5. * Thuyết trình về Code. * Thực hiện phần Code chính. | 100% – A |

**LỜI CAM ĐOAN**

Nhóm xin cam đoan đây là công trình nghiên cứu của riêng nhóm. Các số liệu, kết quả nêu trong báo cáo là trung thực và chưa từng được ai công bố trong bất kỳ công trình nào khác.

Nhóm xin cam đoan rằng mọi sự giúp đỡ cho việc thực hiện báo cáo này   
đã được cảm ơn và các thông tin trích dẫn trong báo cáo đã được chỉ rõ nguồn gốc.

**Sinh viên thực hiện Báo cáo**

*(Ký và ghi rõ họ tên)*

**LỜI CÁM ƠN**

Trước hết, nhóm xin gửi lời cảm ơn chân thành đến cô Nguyễn Thị Định đã tận tình hướng dẫn, hỗ trợ nhóm trong suốt quá trình thực hiện báo cáo môn học này. Nhờ vào sự chỉ dẫn và góp ý quý báu của cô mà nhóm đã hoàn thiện được báo cáo với đề tài " Phân tích dữ liệu Du lịch – Khách sạn".

Nhóm cũng xin cảm ơn nhà trường và khoa Công Nghệ Thông Tin đã tạo điều kiện thuận lợi cho nhóm có cơ hội tìm hiểu và nghiên cứu chuyên sâu về lĩnh vực này.

Cuối cùng, nhóm xin cảm ơn bản thân các thành viên trong nhóm vì đã cố gắng hoàn thành tốt nhiệm vụ và học hỏi được nhiều kiến thức bổ ích trong quá trình thực hiện báo cáo này.

Võ Ngọc Nguyên Chương

Trương Mỹ Hoa

Cao Thiên Chi

# 

# MỤC LỤC

[MỤC LỤC iii](#_Toc213014126)

[DANH MỤC CÁC HÌNH ẢNH vii](#_Toc213014127)

[DANH MỤC BẢNG BIỂU x](#_Toc213014128)

[DANH MỤC TỪ VIẾT TẮT xi](#_Toc213014129)

[CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI 1](#_Toc213014130)

[1.1. Lý do chọn đề tài 1](#_Toc213014131)

[1.2. Mục tiêu nghiên cứu 1](#_Toc213014132)

[1.3. Phạm vi nghiên cứu 2](#_Toc213014133)

[1.4. Đối tượng và phạm vi dữ liệu sử dụng (dataset đặt phòng) 3](#_Toc213014134)

[1.5. Phương pháp nghiên cứu 5](#_Toc213014135)

[1.6. Cấu trúc đề tài 6](#_Toc213014136)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT & TỔNG QUAN CÔNG NGHỆ BIG DATA 8](#_Toc213014137)

[2.1. Tổng quan về Big Data 8](#_Toc213014138)

[2.1.1. Khái niệm về Big Data 8](#_Toc213014139)

[2.1.2. Đặc trưng 5V của Big Data 9](#_Toc213014140)

[2.1.3. Chuỗi giá trị xử lý Big Data (ingestion → storage → processing → analytics → visualization) 10](#_Toc213014141)

[2.2. Hệ thống phân tán và công nghệ phân tán 11](#_Toc213014142)

[2.3. Hệ sinh thái Hadoop 13](#_Toc213014143)

[2.3.1. Kiến trúc và nguyên lý hoạt động của HDFS 14](#_Toc213014144)

[2.3.2. MapReduce và hạn chế 16](#_Toc213014145)

[2.4. Apache Spark 17](#_Toc213014146)

[2.4.1. Kiến trúc Spark (Driver, Executor, Cluster Manager) 18](#_Toc213014147)

[2.4.2. RDD, DataFrame, Spark SQL 19](#_Toc213014148)

[2.4.3. Spark MLlib và các thuật toán hỗ trợ (Clustering, Regression) 20](#_Toc213014149)

[2.5. Các kỹ thuật phân tích dữ liệu lớn 21](#_Toc213014150)

[2.5.1. Phân tích mô tả (Descriptive Analytics) 21](#_Toc213014151)

[2.5.2. Phân tích dự đoán (Predictive Analytics) 21](#_Toc213014152)

[2.5.3. Phân cụm khách hàng (Clustering) 22](#_Toc213014153)

[2.6. Công cụ trực quan hóa dữ liệu 23](#_Toc213014154)

[2.6.1. Matplotlib, Seaborn 23](#_Toc213014155)

[2.6.2. Power BI, Tableau 23](#_Toc213014156)

[CHƯƠNG 3: PHÂN TÍCH YÊU CẦU HỆ THỐNG 25](#_Toc213014157)

[3.1. Bài toán kinh doanh: Quy trình đặt phòng tại các khách sạn và hành vi của khách hàng 25](#_Toc213014158)

[3.2. Mô tả bài toán phân tích dữ liệu đặt phòng trong môi trường Big Data 26](#_Toc213014159)

[3.3. Nguồn dữ liệu và cấu trúc tập dữ liệu 27](#_Toc213014160)

[3.3.1. Mô tả dữ liệu đầu vào (CSV/HDFS) 27](#_Toc213014161)

[3.3.2. Các thuộc tính 28](#_Toc213014162)

[3.4. Yêu cầu chức năng (Functional Requirements) 38](#_Toc213014163)

[3.4.1. Lưu trữ dữ liệu trên HDFS 38](#_Toc213014164)

[3.4.2. Xử lý dữ liệu batch trên Spark 39](#_Toc213014165)

[3.4.3. Thực hiện phân tích mô tả 39](#_Toc213014166)

[3.4.4. Phân nhóm khách hàng theo hành vi đặt phòng (Clustering) 40](#_Toc213014167)

[3.4.5. Dự đoán nhu cầu đặt phòng (Regression) 40](#_Toc213014168)

[3.4.6. Xuất kết quả trực quan hóa 41](#_Toc213014169)

[CHƯƠNG 4: THIẾT KẾ & TRIỂN KHAI HỆ THỐNG PHÂN TÍCH 42](#_Toc213014170)

[4.1. Mô hình kiến trúc hệ thống chi tiết 42](#_Toc213014171)

[4.2. Thiết kế quá trình lưu trữ trên HDFS 42](#_Toc213014172)

[4.3. Luồng xử lý dữ liệu trên Spark 43](#_Toc213014173)

[4.3.1. Data Preprocessing 43](#_Toc213014174)

[4.3.2. Aggregation Analysis 43](#_Toc213014175)

[4.3.3. Clustering bằng K-Means 44](#_Toc213014176)

[4.3.4. Regression dự đoán nhu cầu đặt phòng 44](#_Toc213014177)

[4.4. Thiết kế module trực quan hóa 44](#_Toc213014178)

[4.5. Môi trường triển khai 45](#_Toc213014179)

[CHƯƠNG 5: KẾT QUẢ THỰC NGHIỆM VÀ ĐÁNH GIÁ 46](#_Toc213014180)

[5.1. Liên kết mã nguồn và tài nguyên thực nghiệm 46](#_Toc213014181)

[5.2. Tổng quan phân tích 46](#_Toc213014182)

[5.2.1. Thống kê mô tả tổng hợp 47](#_Toc213014183)

[5.2.2. Phân tích hành vi đặt phòng theo Lead Time 47](#_Toc213014184)

[5.2.3. Phân bố khách hàng theo quốc gia 48](#_Toc213014185)

[5.2.4. Phân tích xu hướng doanh thu theo thời gian 49](#_Toc213014186)

[5.2.5. Tổng kết 50](#_Toc213014187)

[5.3. Phân tích thời gian đặt phòng 50](#_Toc213014188)

[5.3.1. Phân bố số lượng đặt phòng theo thời gian đặt trước 51](#_Toc213014189)

[5.3.2. Tỷ lệ huỷ phòng hơi thời gian đặt 51](#_Toc213014190)

[5.3.3. Phân tích mối quan hệ giữa Lead Time, số lượng đặt phòng và giá phòng trung bình 53](#_Toc213014191)

[5.3.4. Kết luận 54](#_Toc213014192)

[5.4. Phân tích địa lý 55](#_Toc213014193)

[5.4.1. Phân bố lượt đặt phòng theo quốc gia 55](#_Toc213014194)

[5.4.2. Phân tích tỷ lệ huỷ và hành vi từng khu vực 56](#_Toc213014195)

[5.4.3. Mối quan hệ giữa nguồn khách và doanh thu (ADR) 57](#_Toc213014196)

[5.4.4. Kết luận 57](#_Toc213014197)

[5.5. Phân tích doanh thu 58](#_Toc213014198)

[5.5.1. Doanh thu theo thời gian 58](#_Toc213014199)

[5.5.2. Xu hướng giá phòng trung bình (ADR – Average Daily Rate) 58](#_Toc213014200)

[5.5.3. Mối quan hệ giữa số lượng booking và tỷ lệ huỷ 59](#_Toc213014201)

[5.5.4. Kết luận 60](#_Toc213014202)

[5.6. Phân tích theo khách 60](#_Toc213014203)

[5.6.1. Phân bố số lượng khách trong mỗi booking 60](#_Toc213014204)

[5.6.2. Mối quan hệ giữa số lượng khách và ADR 61](#_Toc213014205)

[5.6.3. Phân tích theo loại bữa ăn (Meal Type) 62](#_Toc213014206)

[5.6.4. ADR trung bình theo loại bữa ăn 63](#_Toc213014207)

[5.6.5. Kết luận 64](#_Toc213014208)

[5.7. Phân cụm khách hàng 65](#_Toc213014209)

[5.7.1. Mục tiêu phân cụm 65](#_Toc213014210)

[5.7.2. Phương pháp và tham số mô hình 65](#_Toc213014211)

[5.7.3. Kết quả và diễn giải cụm khách hàng 66](#_Toc213014212)

[5.7.4. Kết luận 68](#_Toc213014213)

[5.8. Dự đoán giá phòng 68](#_Toc213014214)

[5.8.1. Mục tiêu và phương pháp 68](#_Toc213014215)

[5.8.2. Kết quả mô hình 69](#_Toc213014216)

[5.8.3. Phân tích trực quan 70](#_Toc213014217)

[5.8.4. Kết luận 70](#_Toc213014218)

[CHƯƠNG 6: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 71](#_Toc213014219)

[6.1. Kết luận chung 71](#_Toc213014220)

[6.2. Hạn chế của đề tài 72](#_Toc213014221)

[6.3. Hướng phát triển tương lai 72](#_Toc213014222)

[TÀI LIỆU THAM KHẢO 74](#_Toc213014223)

# 

# DANH MỤC CÁC HÌNH ẢNH

[Hình 3.29: Tổng số yêu cầu đặc biệt của khách hàng 37](#_Toc213014106)

# 

# DANH MỤC BẢNG BIỂU

[Bảng 5.1: Thống kê mô tả tổng hợp 47](#_Toc213014048)

[Bảng 5.2: Phân tích hành vi đặt phòng theo Lead Time 47](#_Toc213014049)

# 

# DANH MỤC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **Từ viết tắt** | **Nghĩa Tiếng Anh** | **Nghĩa Tiếng Việt** |
| **ADR** | Average Daily Rate | Giá phòng trung bình mỗi đêm |

# CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI

## Lý do chọn đề tài

Hệ thống giao thông công cộng bằng xe buýt tại TP.HCM là một trong những mạng lưới phức tạp nhất cả nước với hàng trăm tuyến xe buýt, hàng nghìn trạm dừng, hàng ngàn lượt xe vận hành mỗi ngày. Việc quản lý, theo dõi, vận hành và điều phối một hệ thống quy mô lớn như vậy đòi hỏi một công cụ hỗ trợ mạnh mẽ và hiệu quả.

Tuy nhiên, dữ liệu trong hệ thống xe buýt mang tính kết nối cao:

* Trạm này liên kết với trạm kia.
* Tuyến đi qua nhiều trạm khác nhau.
* Mỗi xe phục vụ những tuyến nhất định.

Những mối quan hệ dạng mạng lưới (graph) này khiến các hệ quản trị CSDL quan hệ truyền thống gặp nhiều hạn chế, đặc biệt khi xử lý các truy vấn phức tạp như: tìm đường đi ngắn nhất, truy vấn các tuyến kết nối, hay phân tích sự phụ thuộc giữa các trạm.

Từ thực tiễn này, đồ án lựa chọn ArangoDB, một hệ quản trị cơ sở dữ liệu đa mô hình (Multi-model Database), đặc biệt mạnh trong việc xử lý dữ liệu dạng đồ thị. ArangoDB có ArangoDB Web Interface – giao diện quản trị trực quan, hiện đại, giúp thao tác với dữ liệu dễ dàng, phù hợp cho hệ thống quản lý lộ trình xe buýt.

Do đó, việc nghiên cứu và xây dựng ứng dụng minh họa “Hệ thống Quản lý Xe buýt TP.HCM” bằng ArangoDB là cấp thiết, vừa mang giá trị thực tiễn, vừa thể hiện tính ứng dụng cao của công nghệ CSDL đồ thị.

# 1.2. Mục tiêu nghiên cứu

* Xây dựng mô hình cơ sở dữ liệu đồ thị cho mạng lưới xe buýt: Mô hình hóa các thực thể chính của hệ thống (Trạm – Tuyến – Phương tiện – Lịch trình), xây dựng các quan hệ đồ thị (Graph Edges) để phản ánh sự kết nối giữa các trạm, tuyến và phương tiện, thiết kế cấu trúc dữ liệu phù hợp cho việc truy vấn đường đi, phân tích mạng lưới và thống kê.
* Xây dựng hệ thống quản trị dữ liệu (CRUD) trên nền tảng ArangoDB Web Interface: Sử dụng Web Interface để thực hiện các thao tác với dữ liệu: tạo, xem, cập nhật, xóa. Đảm bảo dữ liệu được tổ chức logic, đồng nhất và dễ mở rộng.
* Phát triển các tính năng thông minh dựa trên Graph Database: Khai thác sức mạnh đồ thị của ArangoDB để thực hiện các chức năng nâng cao: Tìm kiếm đường đi (Journey Planner) sử dụng thuật toán shortest path, truy vấn nâng cao bằng AQL nhằm phân tích sâu dữ liệu (các tuyến kết nối, các trạm quan trọng, sự phân bố mạng lưới).
* Phát triển bộ phân tích và trực quan hóa dữ liệu: Xây dựng dashboard tổng quan tình trạng hệ thống, cung cấp biểu đồ thống kê về tuyến, trạm, phương tiện và lịch trình, hỗ trợ người quản trị đưa ra quyết định tốt hơn dựa trên dữ liệu.
* Tích hợp chức năng quản trị hệ thống và người dùng: Xây dựng mô hình xác thực và phân quyền người dùng, cho phép Admin quản lý tài khoản người dùng và vai trò trong hệ thống, đảm bảo tính bảo mật trong thao tác quản trị dữ liệu.

# 1.3. Phạm vi và đối tượng nghiên cứu

* Đối tượng nghiên cứu
  + Mạng lưới xe buýt TP.HCM: tuyến, trạm, phương tiện và các quan hệ kết nối.
  + Công cụ ArangoDB Web Interface và cơ chế hoạt động của Graph Database.
* Phạm vi nghiên cứu
  + Tập trung vào dữ liệu của hệ thống xe buýt trong đô thị TP.HCM.
  + Chỉ xây dựng mô hình CSDL và ứng dụng minh họa quy mô nhỏ.
  + Không đi sâu vào quy trình vận hành thực tế, yếu tố kinh doanh hay tối ưu vận tải ở cấp độ doanh nghiệp.

# 1.4. Phương pháp thực hiện

* Thu thập và phân tích yêu cầu
  + Tìm hiểu đặc điểm hệ thống xe buýt TP.HCM.
  + Xác định các thực thể cần quản lý (trạm, tuyến, xe,…).
    - * Nghiên cứu ArangoDB và Web Interface
        + Tìm hiểu mô hình đa mô hình (Multi-model), đặc biệt là Graph Model.
        + Nghiên cứu giao diện Web để quản trị dữ liệu.

Thiết kế mô hình dữ liệu đồ thị

Xây dựng các tập vertex (trạm, tuyến, phương tiện).

Xây dựng các tập edge thể hiện mối quan hệ (tuyến–trạm, trạm–trạm, xe–tuyến).

Triển khai hệ thống mẫu

Tạo database, collections, graph trên Web Interface.

Nhập dữ liệu mẫu và viết truy vấn AQL.

Xây dựng ứng dụng minh hoạ

Đánh giá và đề xuất hướng phát triển

Đánh giá tính hiệu quả của mô hình.

Đề xuất khả năng mở rộng trong tương lai.

**1.5. Cấu trúc báo cáo**

# CHƯƠNG 2: TỔNG QUAN VỀ ARANGODB VÀ CSDL ĐỒ THỊ

# 2.1. Tổng quan về CSDL đồ thị

Cơ sở dữ liệu đồ thị (Graph Database) là dạng cơ sở dữ liệu được thiết kế nhằm lưu trữ và truy vấn những dữ liệu có mối quan hệ phức tạp, trong đó thông tin được biểu diễn dưới dạng các thực thể (vertex hoặc node) và mối quan hệ (edge) nối giữa chúng.[1] Mỗi node có thể chứa các thuộc tính mô tả, trong khi edge thể hiện loại quan hệ, hướng và có thể kèm theo thông tin bổ sung như trọng số hoặc thời gian. Nhờ mô hình hóa dữ liệu theo cấu trúc mạng lưới tự nhiên, CSDL đồ thị được sử dụng rộng rãi trong các lĩnh vực như mạng xã hội, bản đồ giao thông, phân tích gian lận, hệ thống gợi ý và quản lý vận tải.

Ưu điểm của CSDL đồ thị bao gồm:

* Mô hình hóa dữ liệu theo đúng bản chất kết nối, phản ánh tự nhiên các mạng lưới phức tạp như: *trạm → tuyến → phương tiện*.[2]
* Truy vấn các liên kết và đường đi nhiều bước (multi-hop) nhanh và trực quan, nhờ khả năng duyệt đồ thị (graph traversal) trực tiếp mà không cần JOIN nhiều bảng như trong hệ CSDL quan hệ truyền thống.
* Linh hoạt trong thay đổi mô hình dữ liệu (schema-less hoặc schema-flexible), cho phép dễ dàng thêm node, thêm loại quan hệ hoặc mở rộng cấu trúc dữ liệu mà không ảnh hưởng đến toàn hệ thống.[3]
* Hiệu suất cao trong các bài toán phân tích mạng, như tìm đường đi ngắn nhất (shortest path), xác định điểm trung chuyển, phát hiện cộng đồng (community detection), hoặc phân tích quan hệ bất thường trong dữ liệu lớn.

Với các hệ thống có cấu trúc kết nối phức tạp và mật độ quan hệ cao, đặc biệt như mạng lưới xe buýt (gồm nhiều trạm, nhiều tuyến, mỗi tuyến kết nối qua nhiều trạm và phương tiện), việc sử dụng CSDL đồ thị giúp quản lý dữ liệu dễ dàng, mô hình hóa trực quan, hỗ trợ tốt cho bài toán phân tích lộ trình và tối ưu đường đi. Điều này vượt trội hơn so với việc lưu trữ toàn bộ quan hệ trong bảng quan hệ truyền thống vốn gây tốn chi phí JOIN và khó mô hình hóa tuyến - trạm - phương tiện một cách tự nhiên.

# 2.2. Tổng quan về ArangoDB

## 2.2.1. Giới thiệu

ArangoDB là một hệ quản trị cơ sở dữ liệu đa mô hình (multi-model database) mã nguồn mở, được thiết kế nhằm hỗ trợ nhiều mô hình dữ liệu khác nhau trong cùng một hệ thống, bao gồm Document, Graph và Key–Value [4]. Nhờ đó, ArangoDB cho phép lưu trữ và truy vấn dữ liệu linh hoạt, đồng thời giảm sự phức tạp khi phải kết hợp nhiều loại cơ sở dữ liệu khác nhau trong một ứng dụng lớn. ArangoDB sử dụng công cụ truy vấn AQL (ArangoDB Query Language) - một ngôn ngữ truy vấn thống nhất dùng cho cả document, graph và kiểu dữ liệu phụ trợ.

Một trong những đặc điểm nổi bật của ArangoDB là khả năng xử lý đồ thị mạnh mẽ, thông qua hai mô hình đồ thị chính:

* General Graph - cho phép người dùng tự định nghĩa tập các collection biểu diễn *đỉnh* (vertex collections) và *cạnh* (edge collections).
* Named Graph - một mô hình đồ thị được định nghĩa sẵn trong hệ thống, giúp đơn giản hóa quản lý và truy vấn.

ArangoDB hỗ trợ nhiều thuật toán đồ thị tích hợp như *shortest path*, *k-shortest paths*, *neighbors*, *traversal nhiều bước*, giúp tối ưu cho các bài toán phân tích kết nối phức tạp.



Hình: Logo của ArangoDB

## 2.2.2. Lịch sử phát triển

ArangoDB được bắt đầu phát triển vào năm 2011 bởi công ty TRIAGNO GmbH (sau này đổi tên thành ArangoDB GmbH). Trong giai đoạn đầu, hệ cơ sở dữ liệu này được phát hành dưới tên AvocadoDB, với định hướng xây dựng một hệ quản trị cơ sở dữ liệu đa mô hình thống nhất, hỗ trợ document, graph và key–value trong cùng một nền tảng.[5]

Đến năm 2012, AvocadoDB chính thức được đổi tên thành ArangoDB. Việc đổi tên được thực hiện để tạo ra thương hiệu dễ nhận diện hơn và phù hợp hơn với định hướng mở rộng sản phẩm. Theo ghi chép lịch sử, *“arango”* là tên một loại đơn vị tiền tệ cổ, được làm từ đá carnelian thô và từng được sử dụng trong trao đổi thương mại tại một số khu vực châu Phi trong nhiều thế kỷ trước.[6] Việc chọn tên này được nhà phát triển xem như một ẩn dụ cho *“đơn vị trao đổi dữ liệu”* - thể hiện mục tiêu cung cấp một hệ thống thống nhất cho nhiều dạng dữ liệu khác nhau.

Trong các năm tiếp theo, ArangoDB liên tục phát triển và được cộng đồng mã nguồn mở đón nhận nhờ:

* Kiến trúc đa mô hình duy nhất trên thị trường vào thời điểm ra mắt.
* Ngôn ngữ truy vấn AQL thống nhất.
* Hỗ trợ clustering và sharding, tối ưu cho dữ liệu lớn.
* Triển khai các tính năng đồ thị nâng cao, bao gồm shortest path, traversals và phân tích graph.

Từ năm 2015 trở đi, ArangoDB trở thành một trong những cơ sở dữ liệu đa mô hình nổi bật, được sử dụng trong các ứng dụng doanh nghiệp, phân tích mạng xã hội, IoT, log phân tán, giao thông vận tải, và các bài toán yêu cầu xử lý đồ thị phức tạp.[7]

## 2.2.3. Đặc điểm

ArangoDB cung cấp các truy vấn có thể mở rộng khi làm việc với dữ liệu đồ thị (graph). Sử dụng JSON làm định dạng lưu trữ mặc định nhưng bên trong sử dụng VelocyPack của ArangoDB - một định dạng nhị phân nhanh và nhỏ gọn để tuần tự hóa và lưu trữ. ArangoDB có thể lưu trữ một đối tượng JSON lồng nhau như 1 data entry bên trong 1 collection.

Do đó, không cần phải tháo rời các đối tượng JSON kết quả. Nhờ vậy, dữ liệu được lưu trữ sẽ đơn giản kế thừa cấu trúc cây của dữ liệu JSON. ArangoDB hoạt động trong 1 cluster phân tán và là DBMS đầu tiên được chứng nhận cho Hệ điều hành Datacenter (DC/OS).

DC/OS cho phép người dùng triển khai ArangoDB trên hầu hết các hệ sinh thái hiện có: Amazon Web Services (AWS), Google Compute Engine và Microsoft Azure. Hơn nữa, nó cung cấp khả năng triển khai chỉ với 1 cú nhấp chuột cho cluster của người dùng. ArangoDB cung cấp tích hợp với các native JavaScript microservices trực tiếp trên DBMS bằng cách sử dụng Foxx framework, tương tự như Node.js đa luồng.

Cơ sở dữ liệu này có AQL (Ngôn ngữ truy vấn ArangoDB) của riêng nó và cũng cung cấp GraphQL để viết các dịch vụ native web linh hoạt trực tiếp trên DBMS. Tính năng mới ArangoSearch là 1 công cụ tìm kiếm trong phiên bản 3.4. Công cụ tìm kiếm kết hợp khả năng truy xuất boolean với các thành phần ranking tổng quát cho phép truy xuất dữ liệu dựa trên 1 mô hình không gian vectơ chính xác (precise vector space).

# 2.3. Kiến trúc ArangoDB

Kiến trúc của ArangoDB được xây dựng nhằm hỗ trợ mô hình đa dữ liệu, xử lý phân tán và đảm bảo hiệu năng cao trong các ứng dụng yêu cầu nhiều kiểu dữ liệu kết hợp. ArangoDB vừa có thể chạy ở chế độ đơn máy, vừa hỗ trợ môi trường phân tán lớn, bao gồm replication, sharding, và cluster theo chuẩn Raft. Các thành phần trong kiến trúc được thiết kế để hoạt động đồng bộ nhưng độc lập, giúp mở rộng linh hoạt và an toàn.[8]

## 2.3.1. Các khái niệm cơ bản

* Multi-Model: document, graph và các cặp key-value - lập mô hình dữ liệu khi thấy phù hợp với ứng dụng của mình.
* Joins: Nối những gì thuộc về nhau một cách thuận tiện để truy vấn đặc biệt linh hoạt, ít dư thừa dữ liệu hơn.
* Transactions: Phát triển ứng dụng dễ dàng giữ cho dữ liệu nhất quán và an toàn. Không có rắc rối trong client

Dưới đây là một truy vấn AQL bao gồm tất cả các tính năng đó:



Hình . Câu truy vấn sử dụng các chức năng của ArangoDB

## 2.3.2. Mô hình lưu trữ

ArangoDB lưu trữ dữ liệu theo định dạng JSON, và các dữ liệu này được gọi là document. Các document được nhóm lại thành collection, và nhiều collection tồn tại bên trong một database.[9]

So với hệ quản trị CSDL quan hệ (RDBMS), nơi mỗi bảng cần xác định rõ các thuộc tính (columns) và quan hệ giữa chúng, trong ArangoDB thì không bắt buộc xác định schema cố định - các document có cấu trúc khác nhau vẫn có thể lưu cùng một collection.Tuy nhiên, vẫn có thể áp dụng giới hạn hoặc schema-validation nếu muốn để dễ thao tác.

ArangoDB hỗ trợ hai dạng collection chính liên quan tới mô hình đồ thị:

* Document Collection: là collection thông thường, chứa các document đại diện cho thực thể (vertices) hoặc dữ liệu dạng tài liệu.[9]
* Edge Collection: là collection đặc biệt để lưu trữ các document biểu diễn *cạnh* (edges) trong đồ thị. Mỗi document trong edge collection có thêm hai thuộc tính đặc biệt là \_from và \_to, dùng để xác định mối quan hệ giữa hai document khác (tức là hai vertex).[9]

## 2.3.3. Các phiên bản ArangoDB

Các phiên bản ArangoDB OASIS: phiên bản cloud của ArangoDB. Cung cấp cơ sở dữ liệu ArangoDB dưới dạng dịch vụ. Cho phép sử dụng toàn bộ chức năng của ArangoDB mà không cần phải tự cài đặc và quản lý hệ thống. Các phiên bản cài đặc khác trên các hệ điều hành khác nhau như Windows, Ubuntu, Docker, … gồm các phiên bản sau:

* ArangoDB Community Edition là một cơ sở dữ liệu native multi-model miễn phí được viết bằng C ++ với giấy phép nguồn mở (Apache 2).
* Ngoài Phiên bản Cộng đồng, ArangoDB Enterprise là một bản trả phí bao gồm SmartGraphs, Satellite Collections và nhiều tính năng bảo mật cấp doanh nghiệp. Ngoài ra còn có 1 số sản phẩm ở lĩnh vực khác như: BI Connector, ArangoML.

# 2.4. Lợi ích của ArangoDB trong quản lý giao thông đô thị

Ngoài ra, ArangoDB còn có các ưu điểm quan trọng:

* Hỗ trợ đa mô hình trong một engine duy nhất, giúp giảm độ phức tạp so với việc phải sử dụng nhiều hệ quản trị riêng (như MongoDB + Neo4j).
* Hỗ trợ sharding và clustering, đảm bảo khả năng mở rộng theo chiều ngang khi dữ liệu tăng trưởng lớn.[10]
* AQL trực quan, hỗ trợ JOIN, khắc phục hạn chế JOIN trong các hệ NoSQL đơn mô hình.
* Hiệu suất cao, nhờ cơ chế tối ưu bộ nhớ và cấu trúc dữ liệu được thiết kế cho workloads lớn.

Trong bối cảnh bài toán quản lý mạng lưới xe buýt, ArangoDB đặc biệt phù hợp nhờ:

* Mô hình Document giúp lưu trữ thông tin trạm, tuyến và phương tiện dưới dạng JSON.
* Mô hình Graph giúp mô tả trực quan quan hệ giữa các trạm và các tuyến.
* AQL hỗ trợ truy vấn đường đi (shortest path), tìm trạm lân cận, phân tích kết nối - đây là các thao tác cốt lõi để xây dựng ứng dụng phân tích và tối ưu lộ trình giao thông.

Nhờ sự kết hợp giữa tính linh hoạt, khả năng mở rộng và mô hình đồ thị tích hợp, ArangoDB trở thành một nền tảng phù hợp để xây dựng các hệ thống có tính kết nối cao, ví dụ như quản lý giao thông công cộng, hệ thống logistics, bản đồ đô thị hoặc phân tích mạng phức tạp.

# 

# CHƯƠNG 3. CÀI ĐẶT VÀ LÀM QUEN ARANGODB WEB INTERFACE

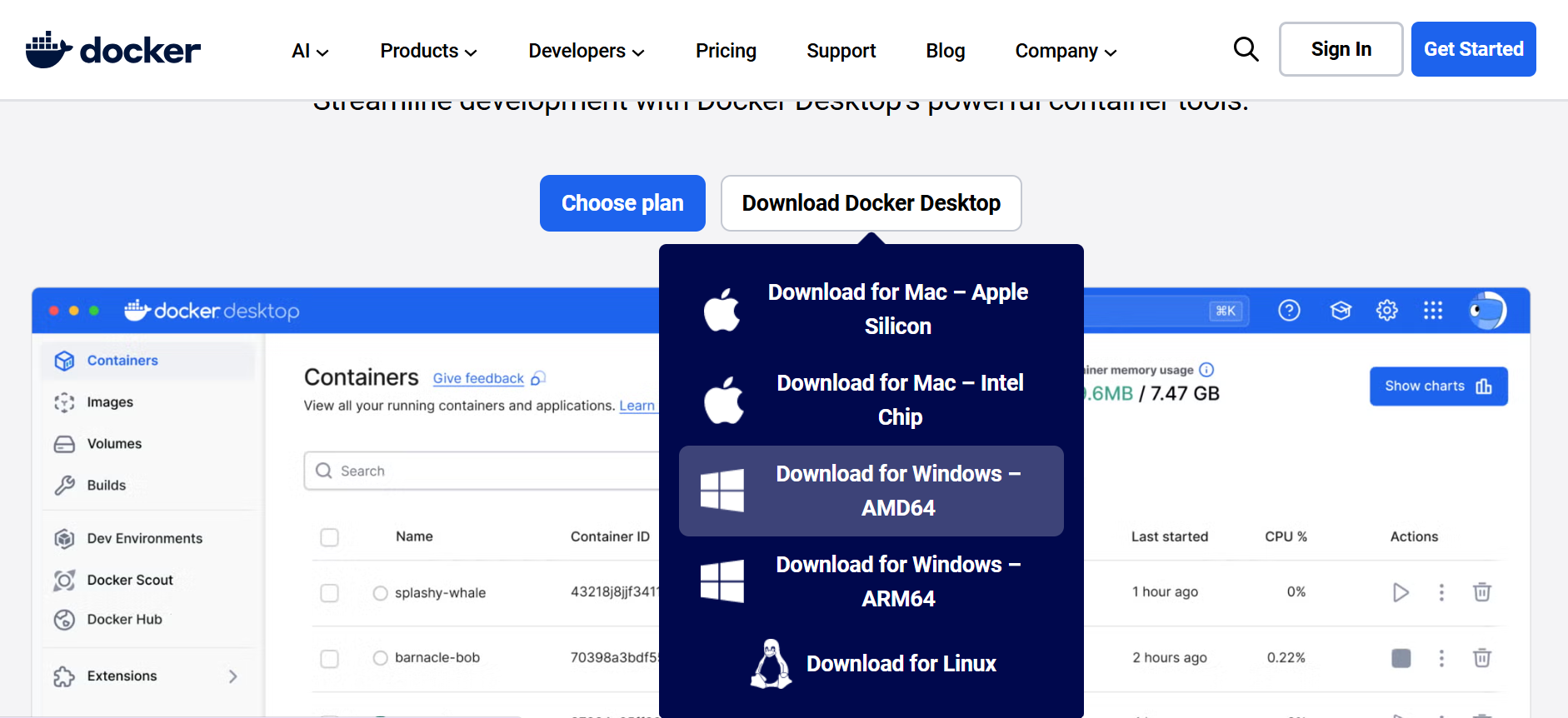
# 3.1. Yêu cầu hệ thống & môi trường cài đặt

Để triển khai ArangoDB phục vụ quá trình xây dựng và thử nghiệm hệ thống, cần chuẩn bị môi trường cài đặt phù hợp. Trong đề tài này, ArangoDB được triển khai thông qua Docker, giúp đảm bảo quá trình cài đặt đơn giản, nhanh chóng, đồng nhất giữa các máy và dễ dàng quản lý phiên bản.

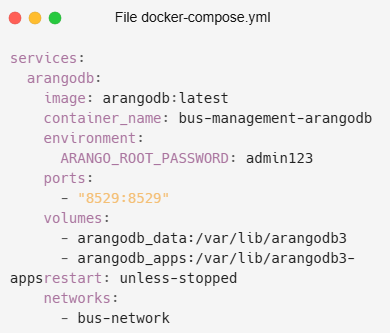
* Yêu cầu phần cứng tối thiểu
  + RAM: 4 GB (khuyến nghị 8 GB để chạy dataset lớn hoặc graph traversal)
  + Dung lượng ổ đĩa: tối thiểu 5 GB trống
  + Hệ điều hành: Windows 10/11, Ubuntu, macOS hoặc bất kỳ OS nào hỗ trợ Docker
    - * Yêu cầu phần mềm

Để chạy ArangoDB bằng Docker, cần cài đặt các phần mềm sau:

* Docker Desktop
  + Docker Desktop cho phép chạy container trên Windows, macOS hoặc Linux, tải tại: <https://www.docker.com/products/docker-desktop>, chọn loại phù hợp với hệ điều hành và nhấn download.



* + Sau khi tải file Docker Desktop Installer.exe về nhấn Install để cài đặt Docker.
* Docker-compose
  + Trong thư mục triển khai, tạo một file có tên docker-compose.yml chứa cấu hình dịch vụ ArangoDB. Nội dung file như hình sau:



Hình : File cấu hình docker-compose

Việc cài đặt ArangoDB thông qua Docker mang lại nhiều lợi ích:

* Không cần cài đặt thủ công ArangoDB vào hệ điều hành.
* Dễ dàng cấu hình thông qua file docker-compose.yml.
* Tránh lỗi phụ thuộc (dependency conflict).
* Có thể chạy nhiều phiên bản song song.
* Dễ backup dữ liệu bằng Docker volume.

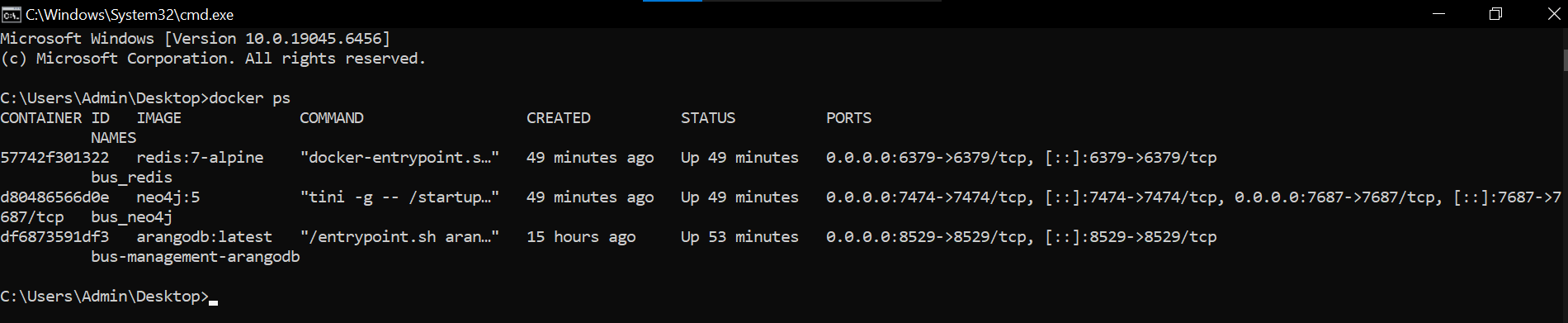
# 3.2. Cài đặt ArangoDB Web Interface

* Bật Docker Desktop lên
* Tại nơi lưu file docker-compose.yaml, chạy cmd với nội dung sau: *docker-compose up -d*



Hình : Chạy lệnh docker-compose up -d để cài đặt ArangoDB

* Đợi tải ArangoDB Web Interface

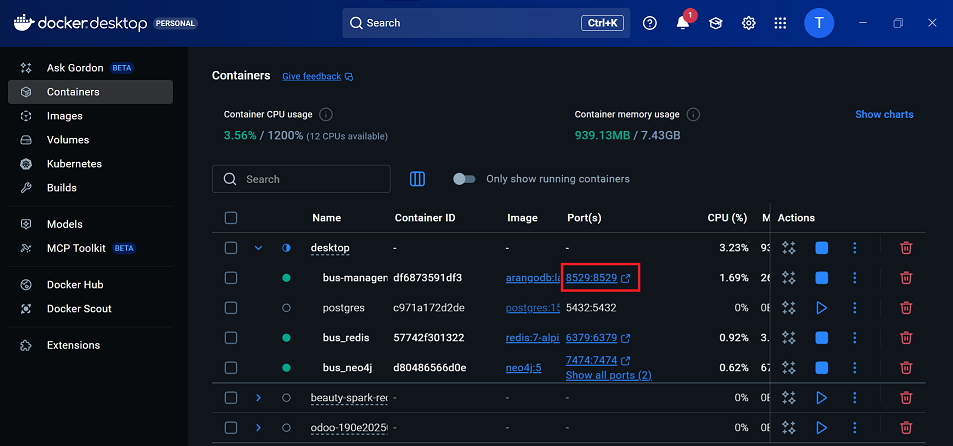


Hình : Kiểm tra trạng thái container ArangoDB bằng lệnh docker - ps

Nếu container có tên bus-management-arangodb xuất hiện → cài đặt thành công.

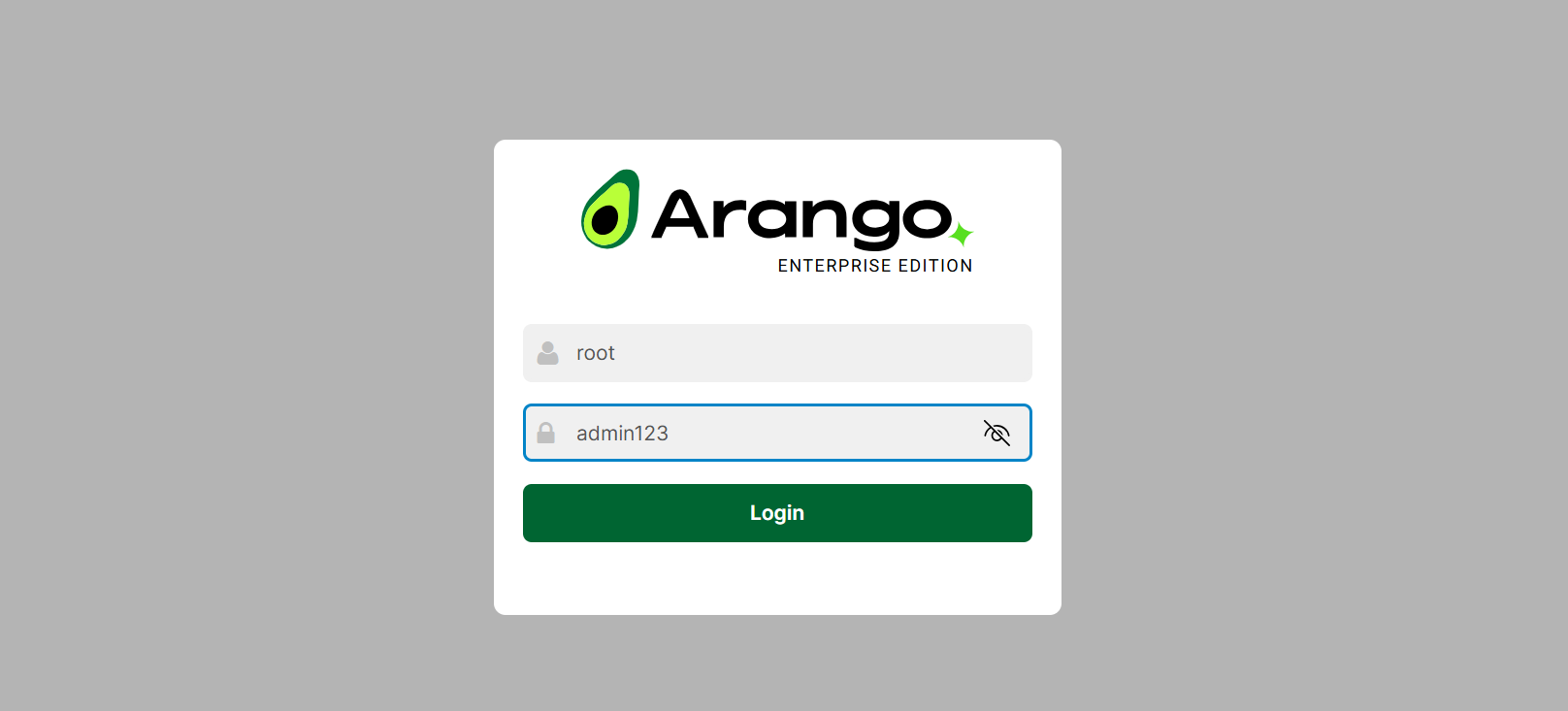
# 3.3. Truy cập và đăng nhập ArangoDB Web Interface

* Mở Dockertop, tại port của bus-management-arangodb nhấn vào đó sẽ hiện ra trang login của ArangoDB



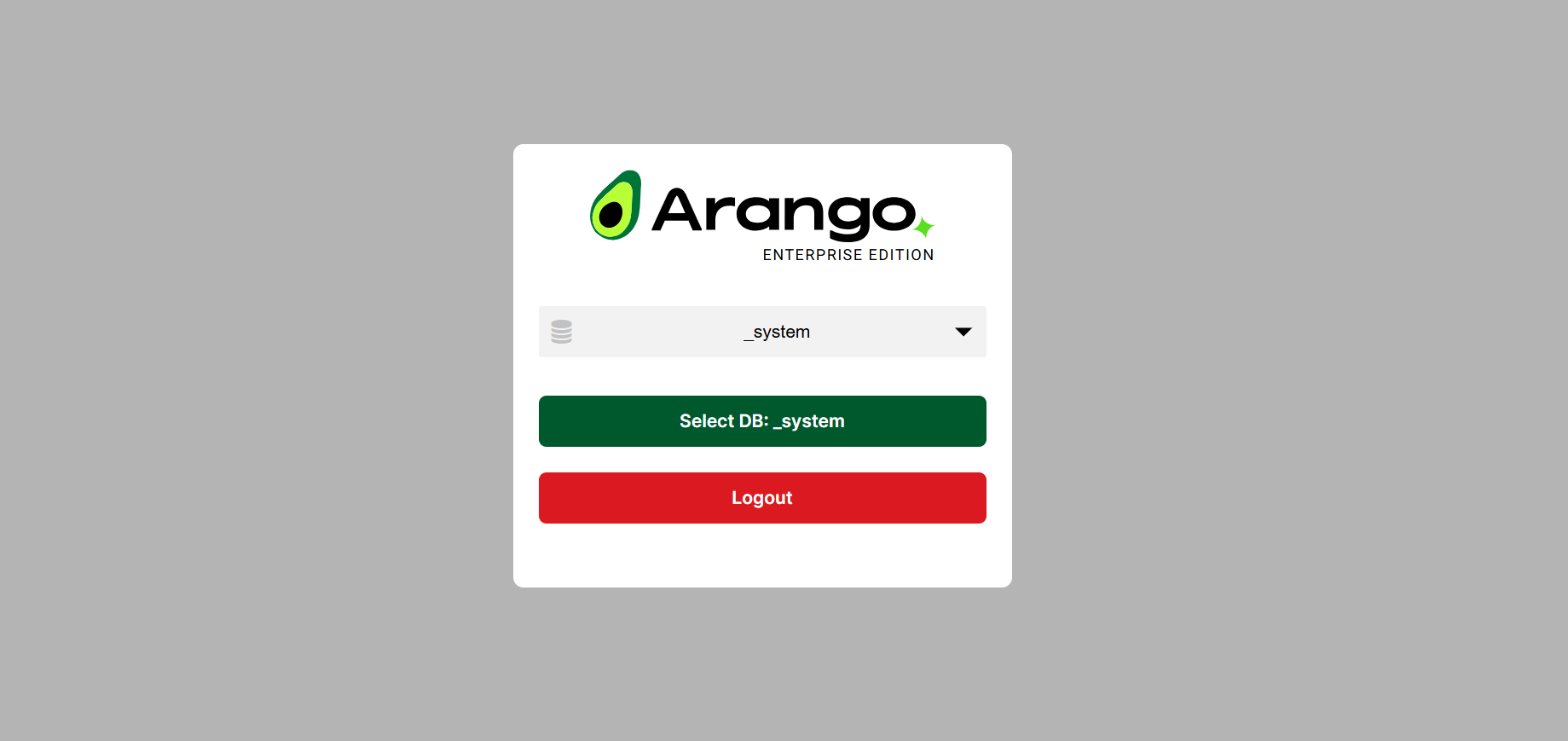
Hình : Giao diện Docker Desktop

* Tại trang login đăng nhập *user root và password: admin123*



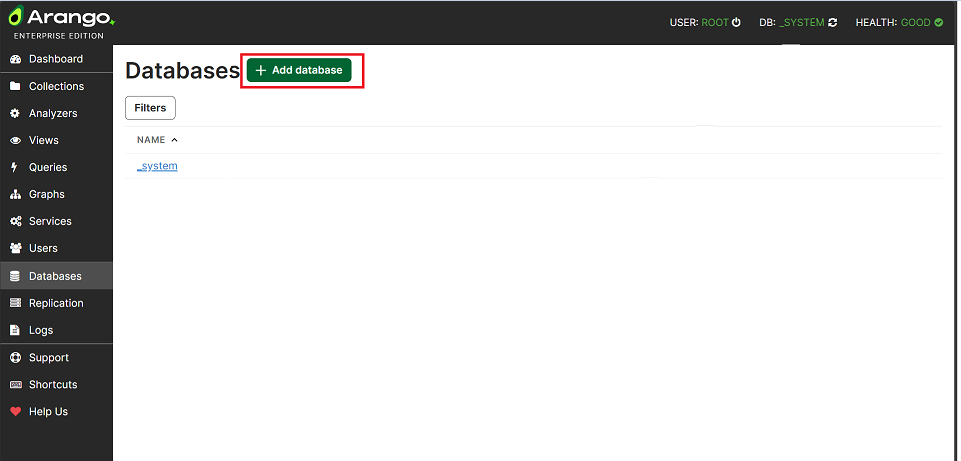
Hình : Giao diện đăng nhập của Arangodb

* Chọn nút SelecDB: \_system



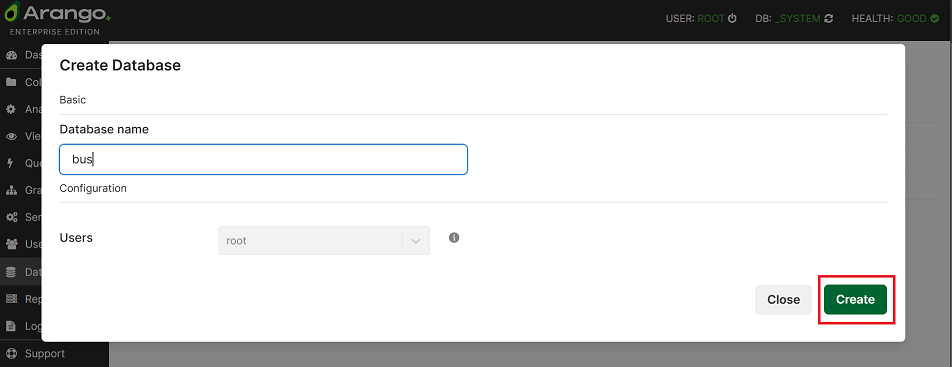
Hình : Giao diện chọn database của Arangodb

* Tại mục Database chọn *“Add database”*



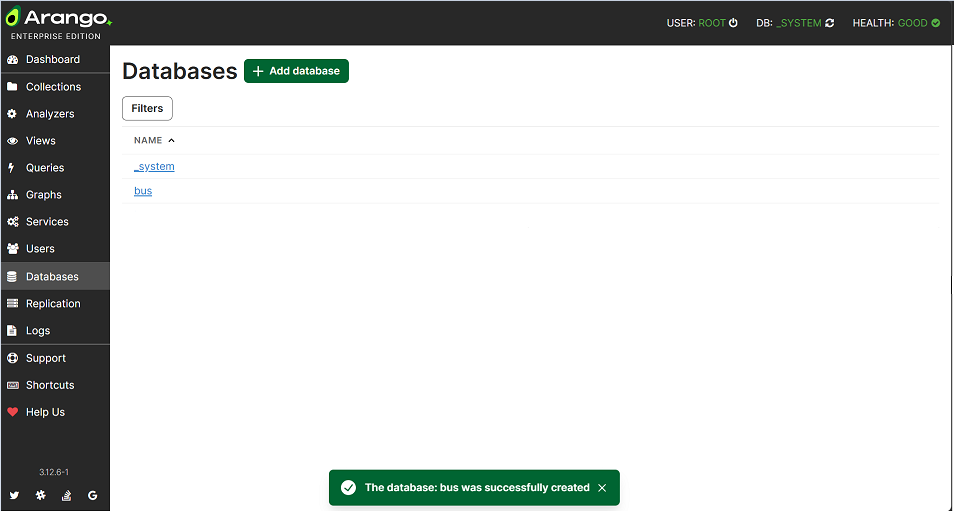
Hình : Giao diện chính của Arangodb

* Đặt tên database rồi chọn *“Create”*



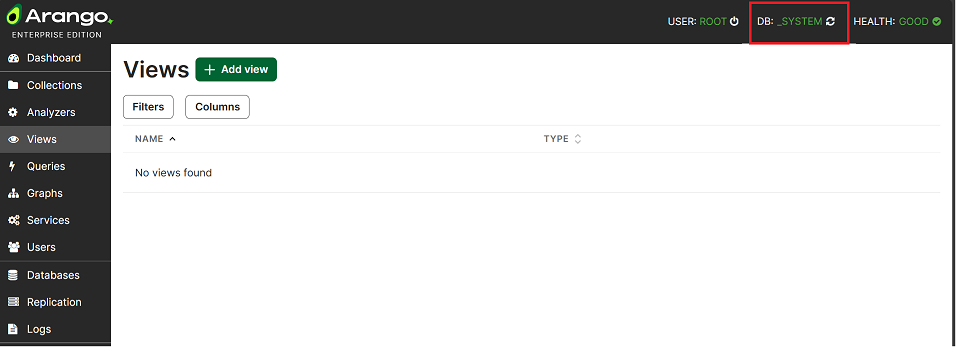
Hình : Giao diện tạo database của Arangodb

* Tạo thành công database



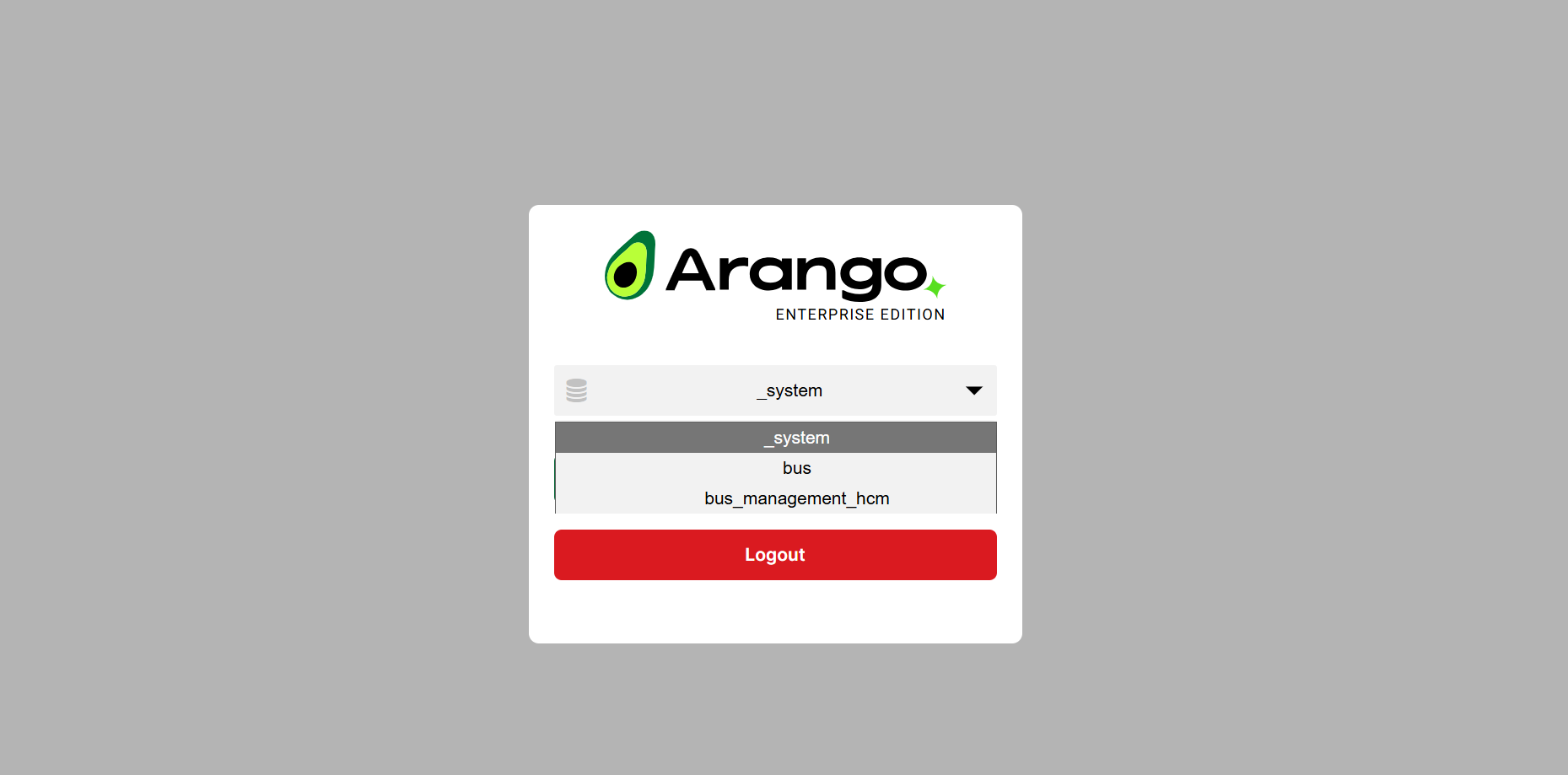
Hình : Danh sách database của Arangodb

* Chọn mục *“DB:\_System”*

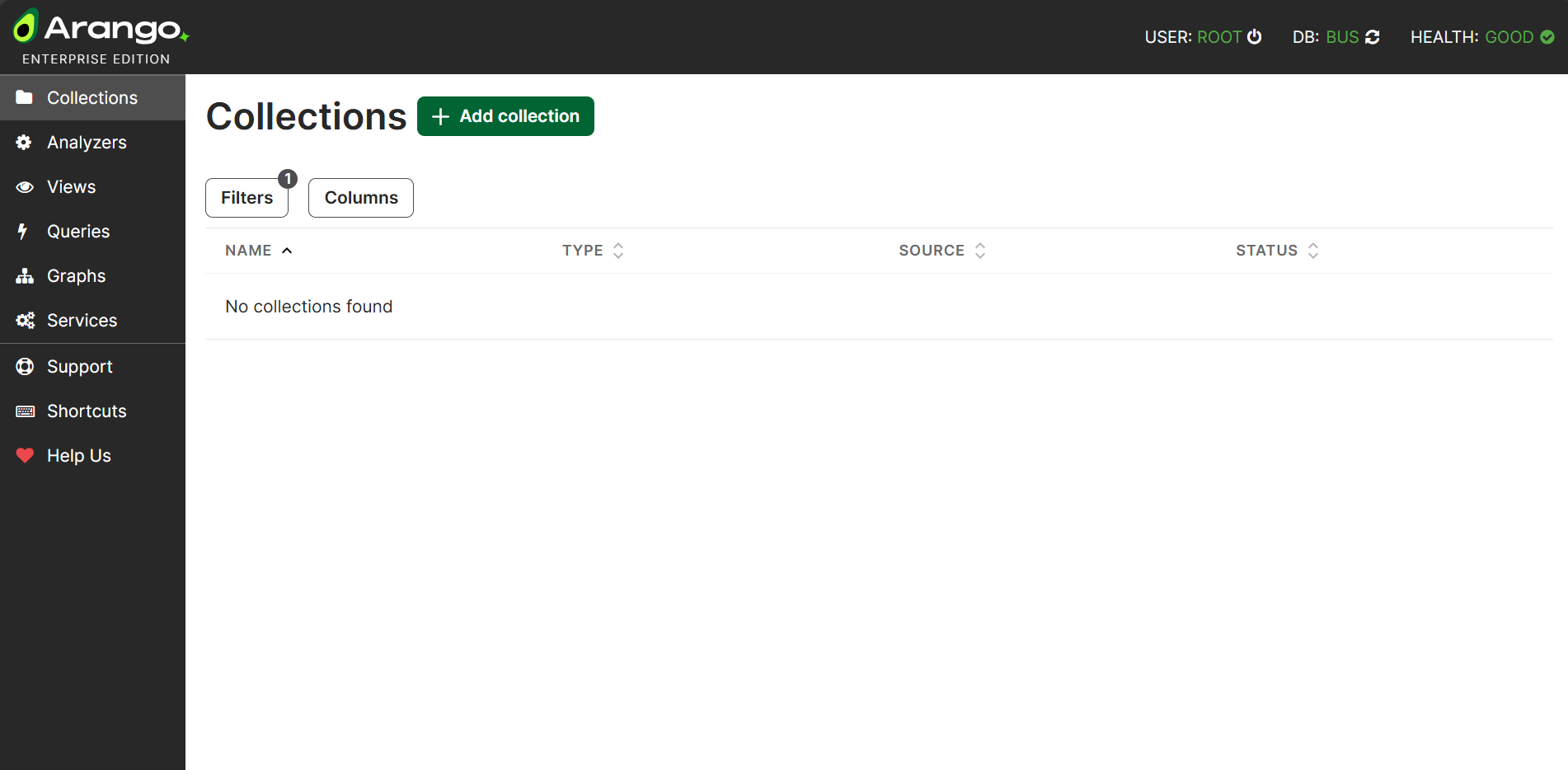


Hình : Giao diện chính \_system của Arangodb

* Chọn database cần quản lý, nhấn nút *“Select DB”*



Hình : Giao diện chọn database quản lý của Arangodb



Hình : Giao diện chính database Bus của Arangodb

# CHƯƠNG 4. CHỨC NĂNG QUẢN TRỊ TRÊN ARANGODB WEB INTERFACE

# 4.1. Quản lý Database

Trong ArangoDB, database là một “không gian lưu trữ độc lập” dùng để chứa toàn bộ dữ liệu của một ứng dụng.  
Mỗi database gồm:

* Collections (bảng / tập dữ liệu).
* Documents (tài liệu JSON).
* Edges (quan hệ trong mô hình đồ thị).
* Graphs (đồ thị).
* User, permission.
* AQL queries liên quan

Mỗi database hoạt động tách biệt với nhau → dữ liệu của database này không ảnh hưởng đến database khác.

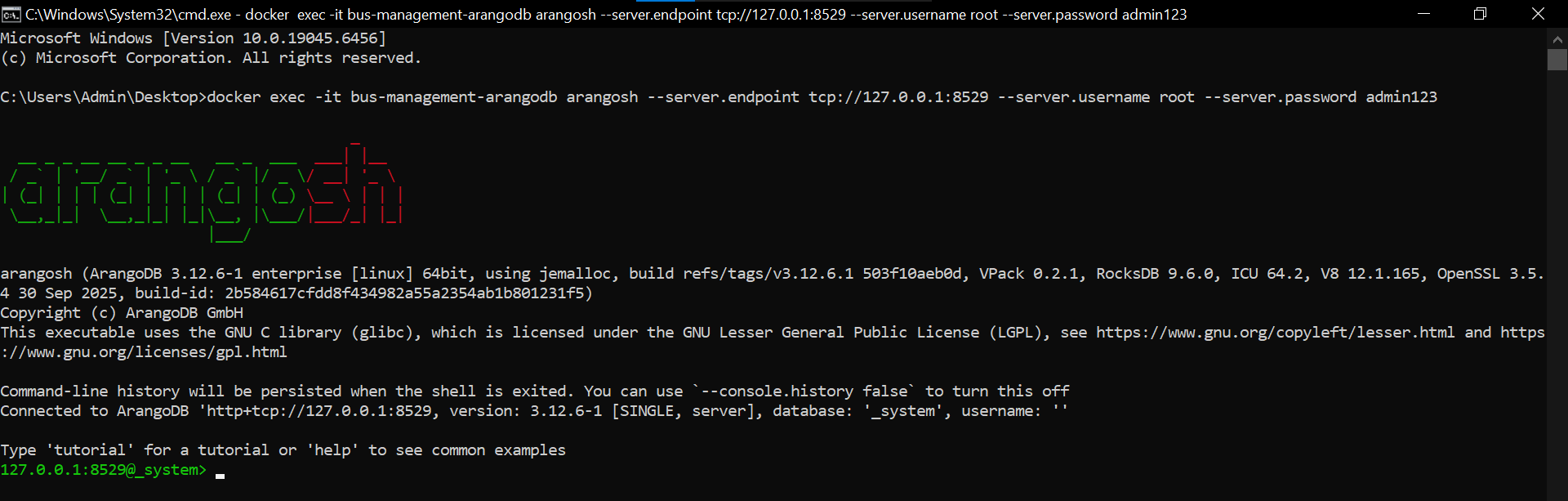
Để bắt đầu tạo database phải đã chạy “*docker-compose up -d”* và container bus-management-arangodb đang chạy - mật khẩu “*root”* của ArangoDB là “*admin123”.*

Có 2 cách tạo và quản lý database

* *Cách 1: Tạo database trên ArangoDb Web Interface - tham khảo mục 3.3*
* *Cách 2: Tạo database trên Terminal/Command Prompt - thực hiện các bước sau*

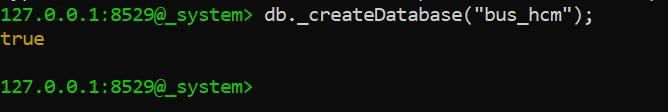
***Bước 1:***Mở Terminal/Command Prompt - mở một cửa sổ Terminal (trên Linux/macOS) hoặc Command Prompt/PowerShell (trên Windows).

*Bước 2:* Tại cmd chạy *“docker exec -it bus-management-arangodb arangosh --server.endpoint tcp://127.0.0.1:8529 --server.username root --server.password admin123”*



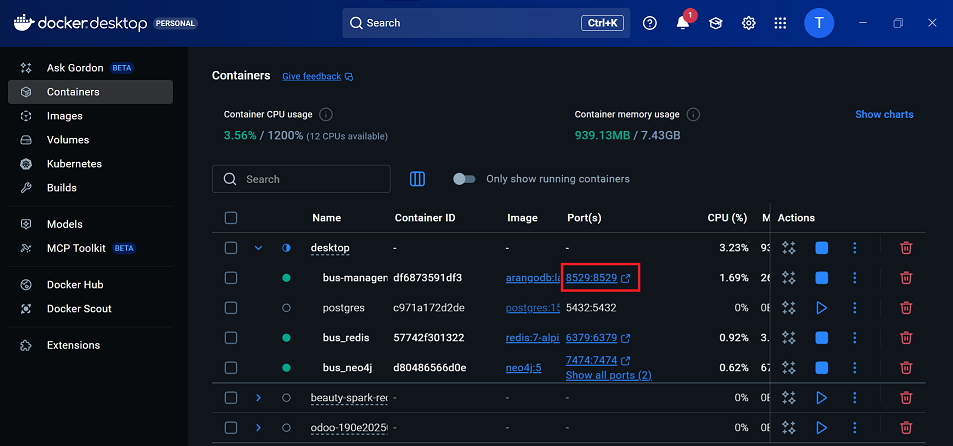
Hình : Kết nối với ArangoDB ở chế độ dòng lệnh

*Bước 3:* Tạo database với dòng lệnh *“db.\_createDatabase(“Ten\_database”);*”



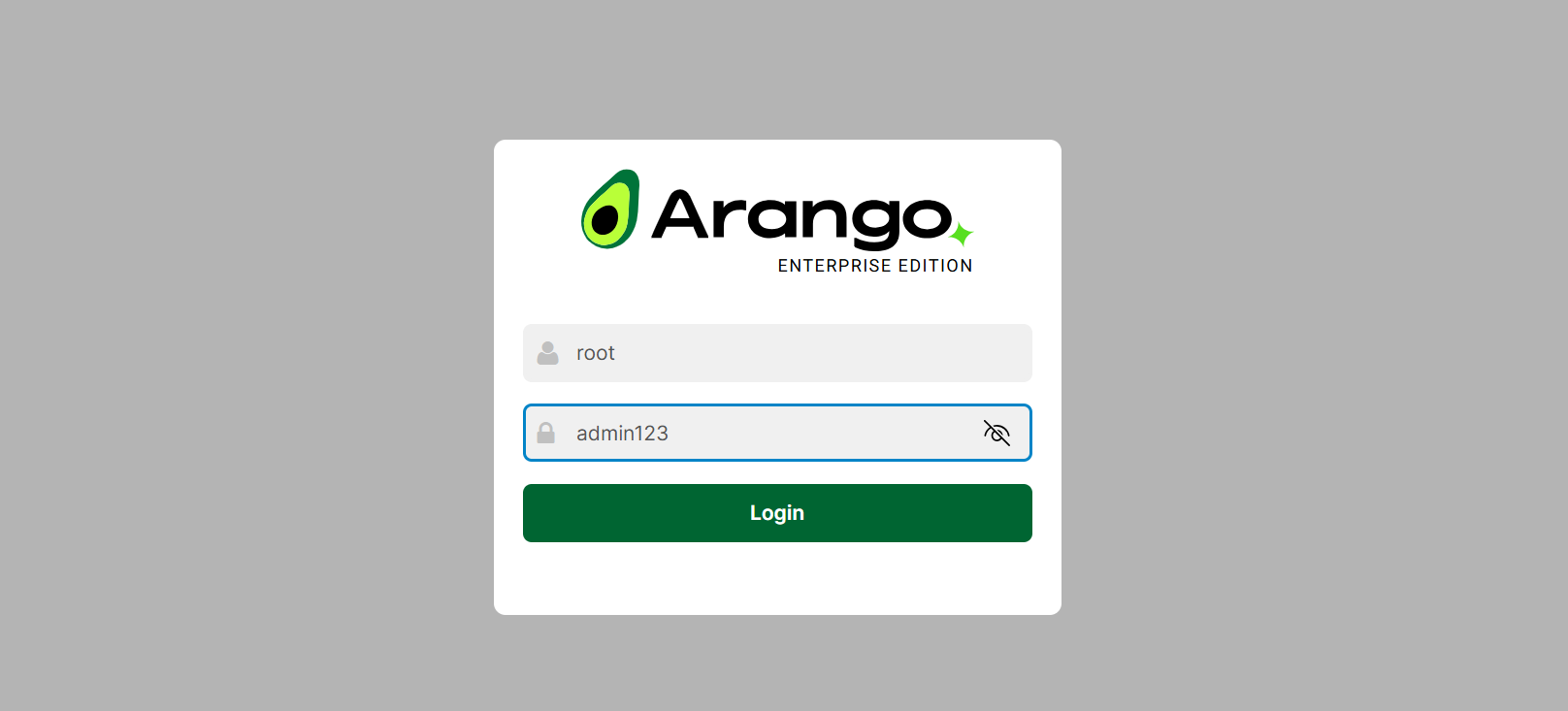
Hình : Tạo database thành công ở chế độ dòng lệnh

*Bước 4:* Mở Dockertop, tại port của bus-management-arangodb nhấn vào đó sẽ hiện ra trang login của ArangoDB



Hình : Giao diện Docker Desktop

*Bước 5:* Tại trang login đăng nhập *user root và password: admin123*

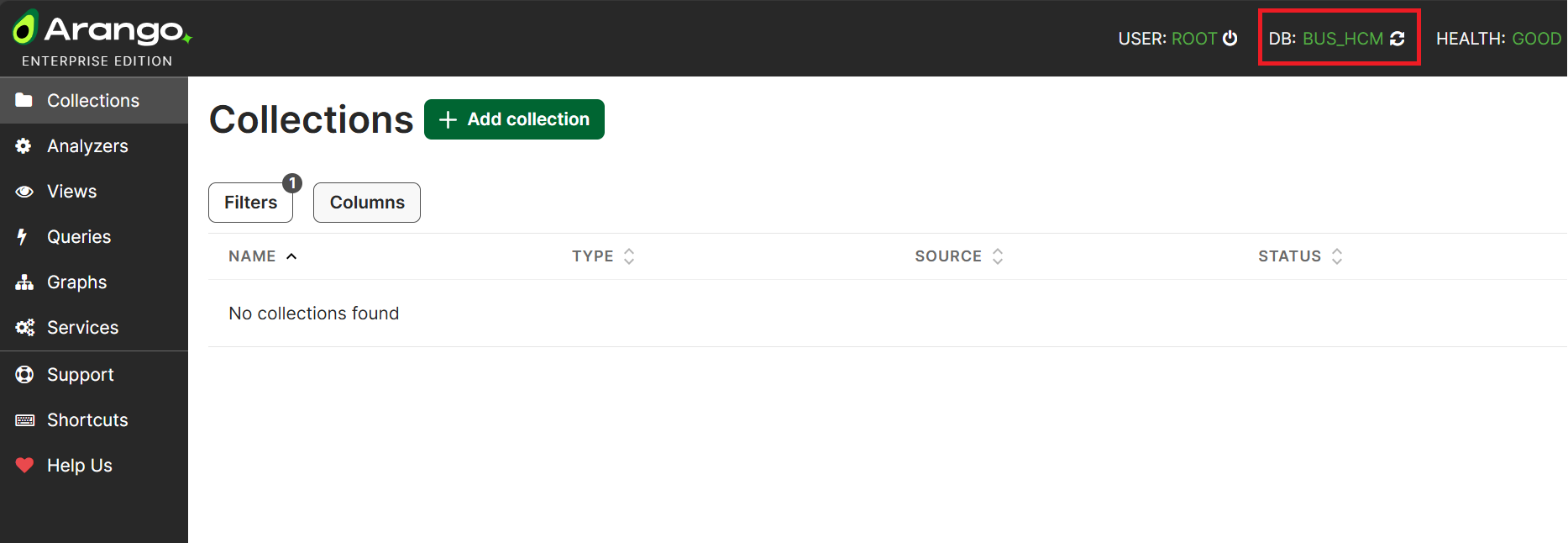


Hình : Giao diện đăng nhập của Arangodb

*Bước 6:* Chọn database cần quản lý và nhấn nút *“Select DB: bus\_hcm”*



Hình : Giao diện chọn database của Arangodb

**

Hình : Giao diện collections của Arangodb với database BUS\_HCM

# 4.2. Quản lý Collections (Document & Edge)

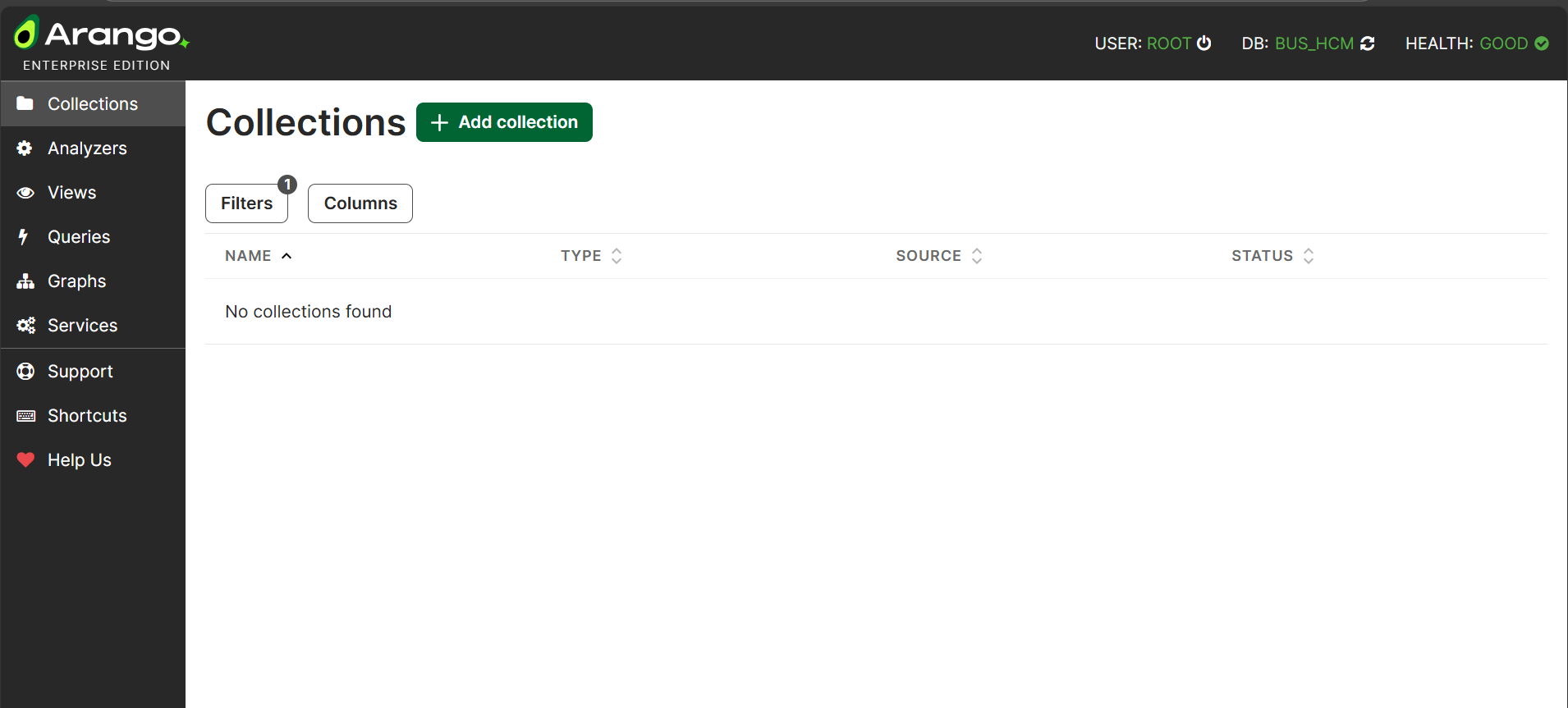
Trong ArangoDB, collection là một tập dữ liệu được dùng để lưu trữ các documents (tài liệu JSON) hoặc edges (các quan hệ trong đồ thị).  
 Nó tương đương với:

* Bảng trong SQL
* Collection trong MongoDB

Để bắt đầu tạo collections phải đã đăng nhập vào database cần tạo collection.

Có 2 cách tạo và quản lý collections

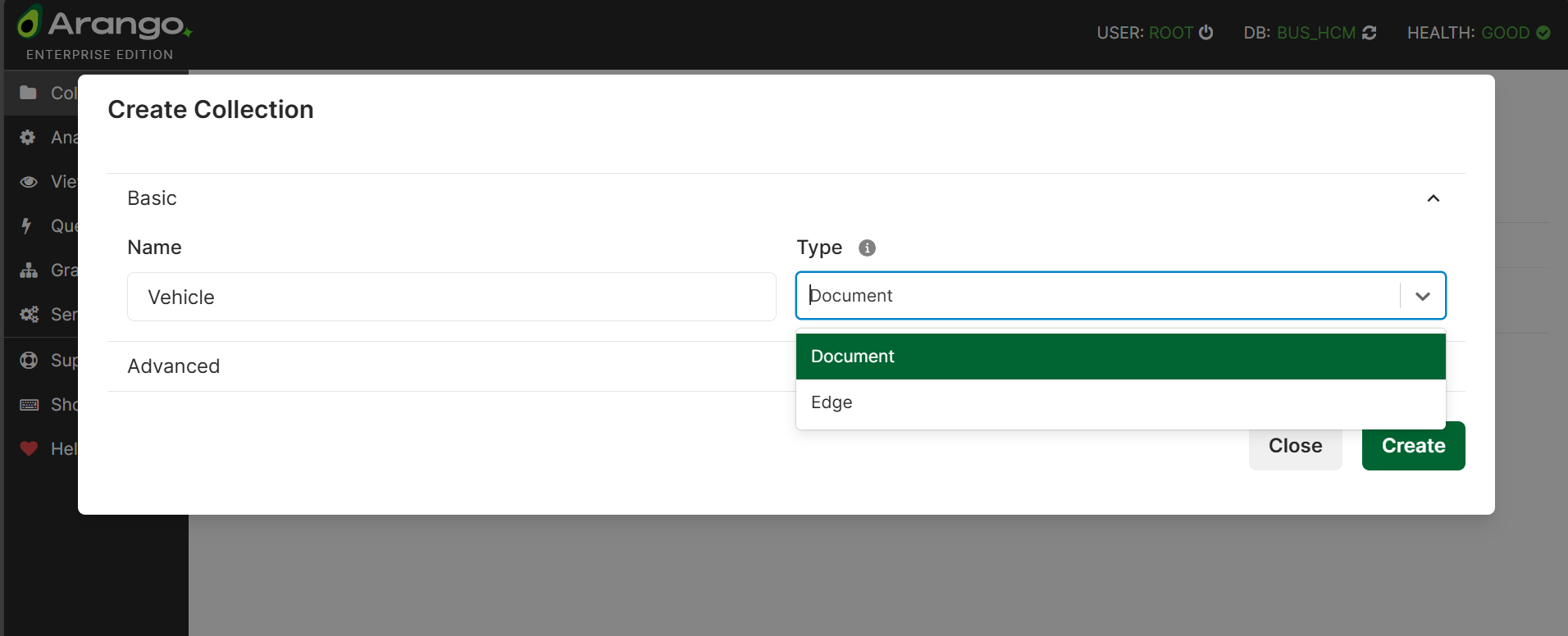
* *Cách 1: Tạo collections trên ArangoDb Web Interface - thực hiện các bước sau:*

*Bước 1:* Tại giao diện *Collections* của ArangoDB, chọn *“+Add collection”*

Hình : Giao diện collections chính với database BUS\_HCM

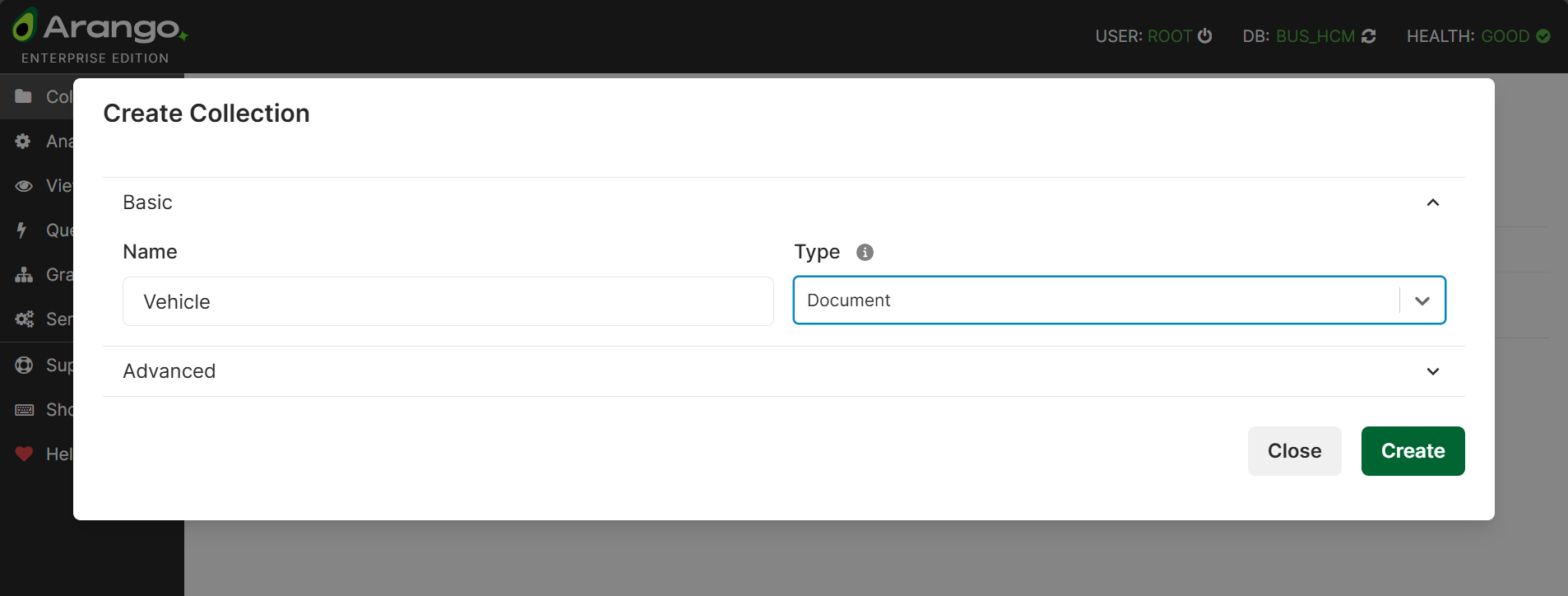
Bước 2: Tại mục *Create Collection*, đặt tên collection tại mục *name* - *type* có 2 loại Documents (tài liệu JSON), Edges ( quan hệ giữa dữ liệu)

Ví dụ: Đặt tên name là Vehicle, với type là document

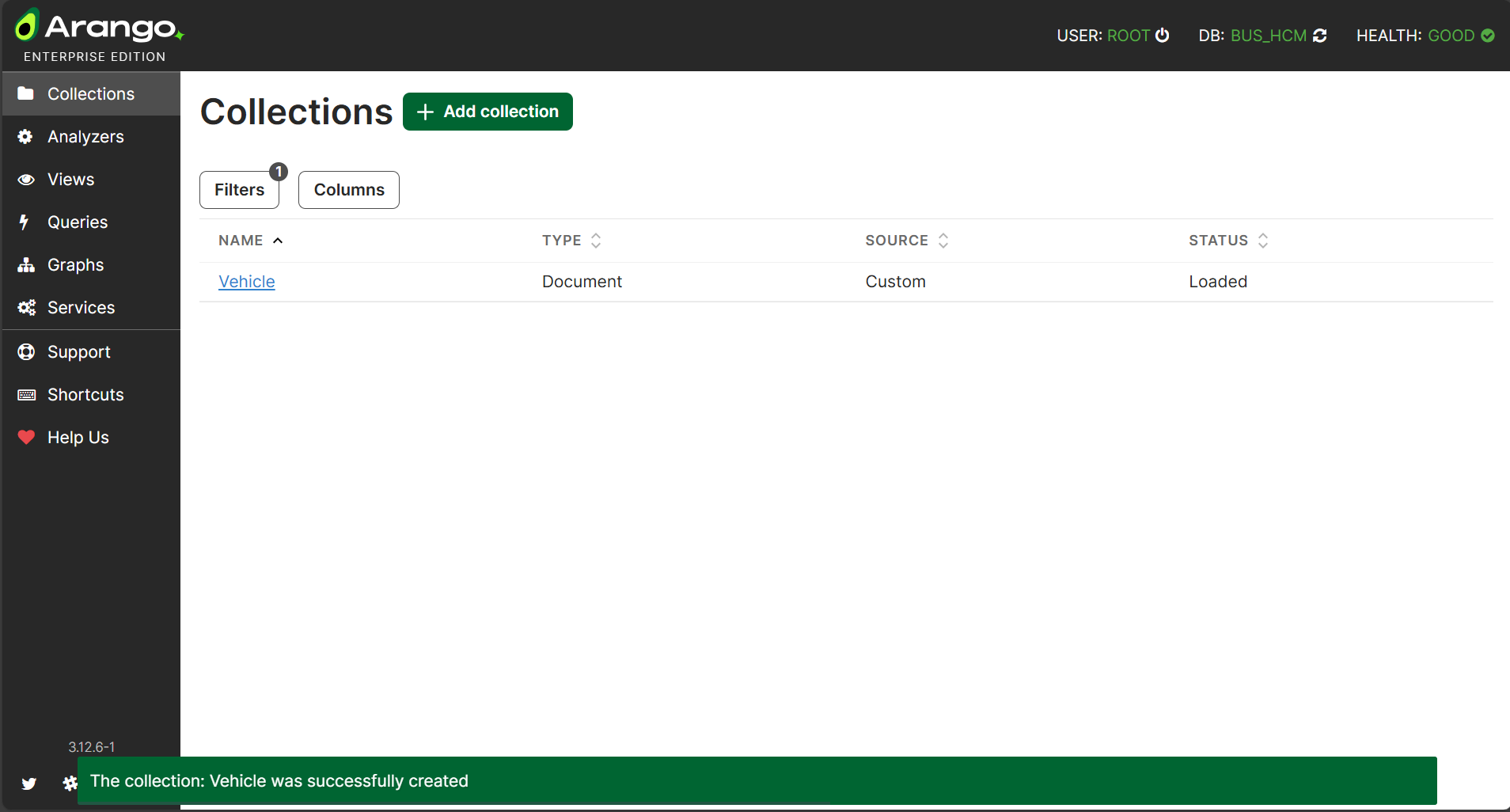


Hình : Collection ‘Vehicle’ với loại dữ liệu Document trong ArangoDB Web Interface

Bước 3: Nhấn nút *“Create”* để tiến hành tạo collection



Hình : Tạo collection ‘Vehicle’ trên ArangoDB Web Interface

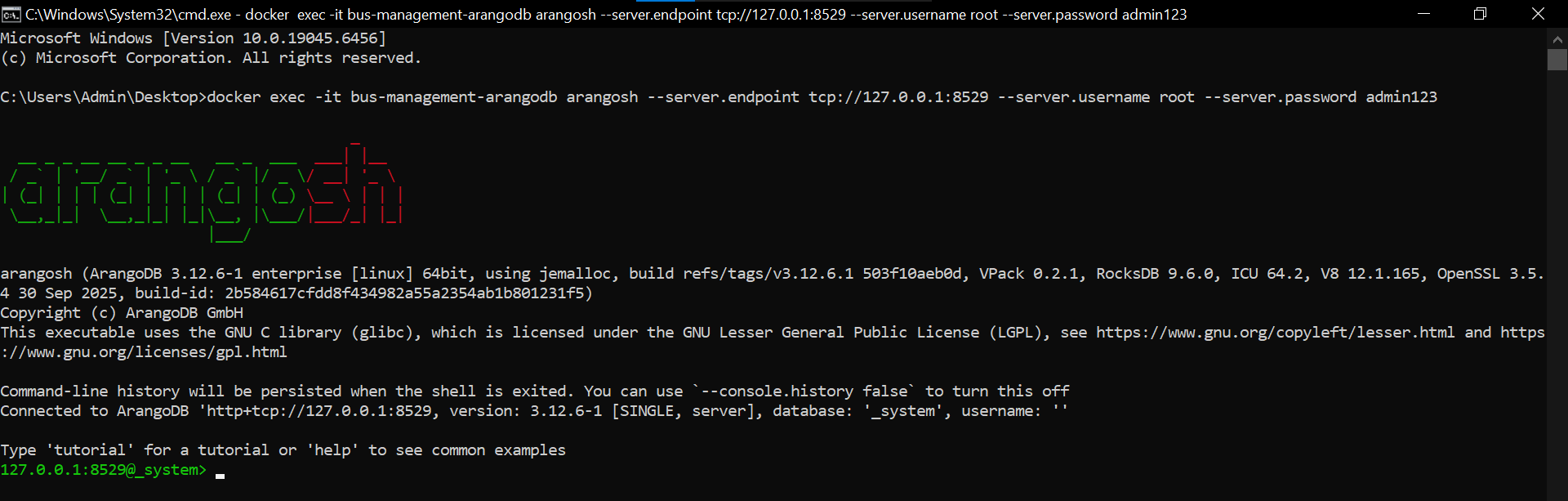


Hình : Giao diện Collections hiển thị collection 'Vehicle' sau khi tạo thành công

* *Cách 2:Tạo collection trên Terminal/Command Prompt - thực hiện các bước sau:*

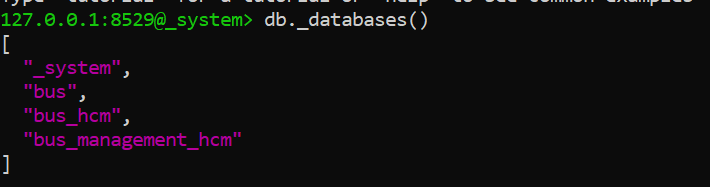
*Bước 1:* Mở Terminal/Command Prompt - mở một cửa sổ Terminal (trên Linux/macOS) hoặc Command Prompt/PowerShell (trên Windows)

*Bước 2:* Tại cmd chạy *“docker exec -it bus-management-arangodb arangosh --server.endpoint tcp://127.0.0.1:8529 --server.username root --server.password admin123”*



Hình : Kết nối với ArangoDB ở chế độ dòng lệnh trên cmd

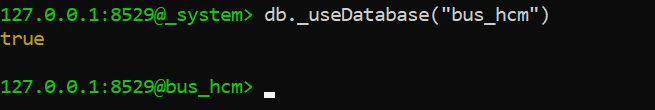
Bước 3: Chạy lệnh *“db.\_databases()”* để kiểm tra tất cả các database của Arangodb



Hình : Kết quả hiển thị danh sách database ở chế độ dòng lệnh

Bước 4: Chạy lệnh *“db.\_useDatabase(“ten\_database”)”* để truy cập vào đúng database

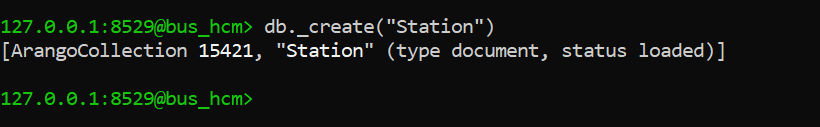
Ví dụ: Thực thi lệnh *“db.\_useDatabase(“bus\_hcm”)”* - để truy cập vào database bus\_hcm



Hình : Chuyển sang database bus\_hcm bằng arangosh

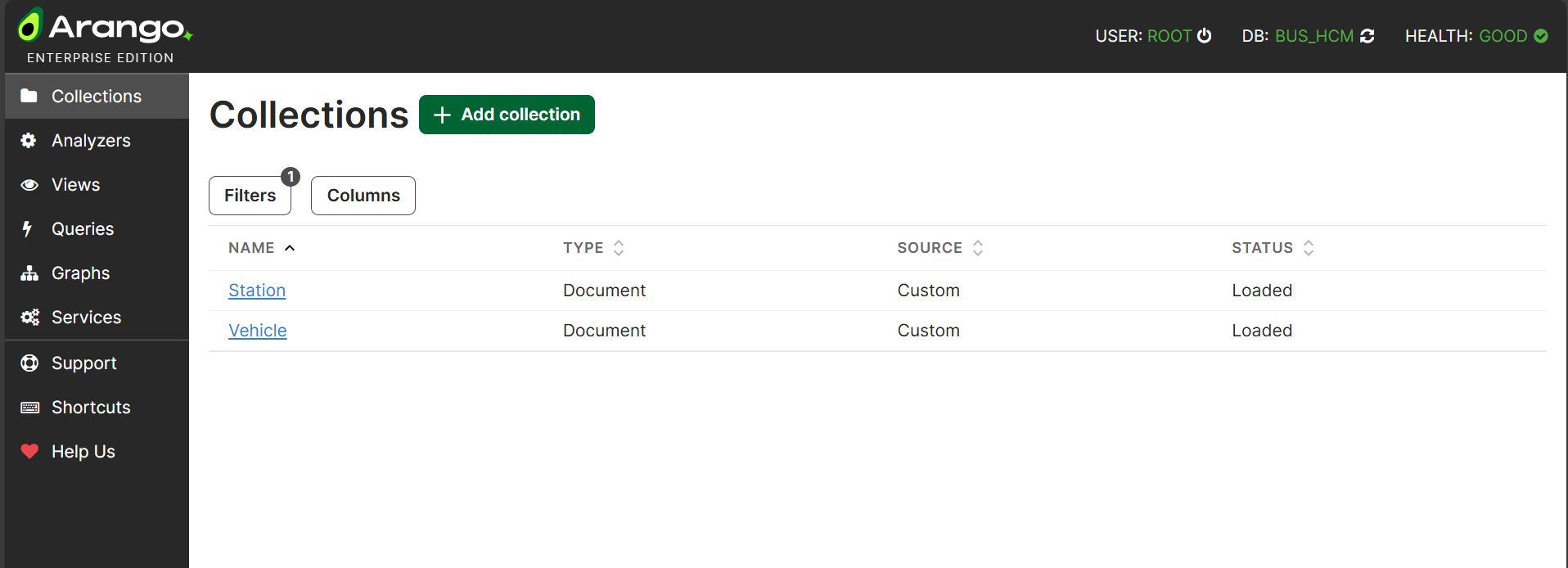
Bước 5: Chạy lệnh *“db.\_create(“ten\_collection”)”* để tạo 1 collection

Ví dụ: Thực thi lệnh *“db.\_create(“Station”)”* để tạo collection tên Station



Hình : Tạo collection Station ở chế độ dòng lệnh

Bước 6: Đăng nhập vào database *“bus\_hcm”,* tại mục *Collections* hiển thị danh mục collection đã được tạo



Hình : Danh sách collections sau khi tạo thành công

# 4.3. Quản lý Graph

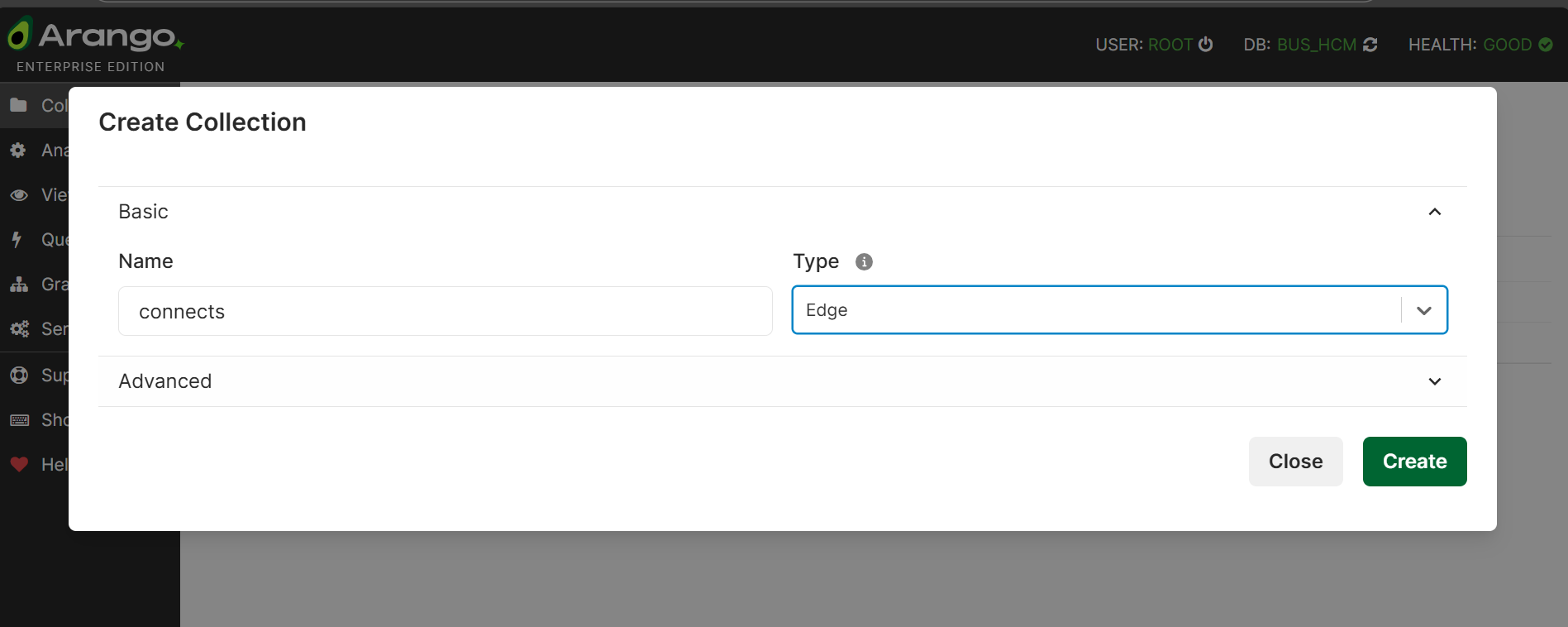
Trong ArangoDB, Graph (đồ thị) là một mô hình dữ liệu dùng để biểu diễn các đối tượng và mối quan hệ giữa chúng một cách trực quan và linh hoạt. Một graph bao gồm hai thành phần chính: vertex (đỉnh, lưu trong các *document collections*) và edge (cạnh, lưu trong *edge collections*). Vertex đại diện cho các thực thể như trạm xe buýt, tuyến xe, phương tiện… trong khi edge thể hiện mối quan hệ giữa các thực thể này, ví dụ: “tuyến đi qua trạm”, “xe thuộc tuyến”, hoặc “trạm kết nối với trạm”.

Để bắt đầu tạo graph phải đã đăng nhập vào database và đã tạo collections

Có 2 cách tạo và quản lý graph

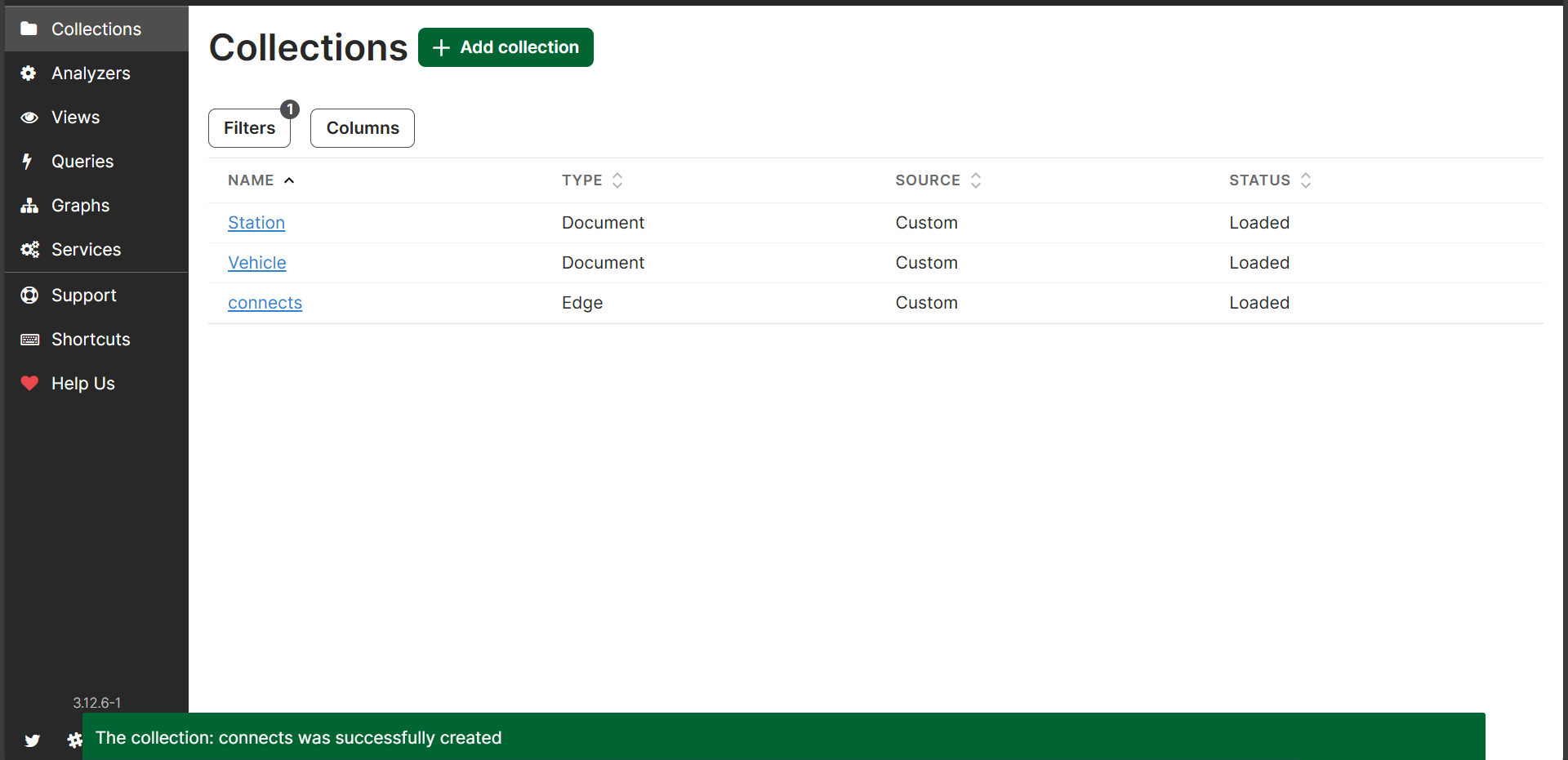
* *Cách 1: Tạo graph trên ArangoDb Web Interface - thực hiện các bước sau:*

*Bước 1:* Tại giao diện *Collections,* tạo 1 collection với type *Edge*

**

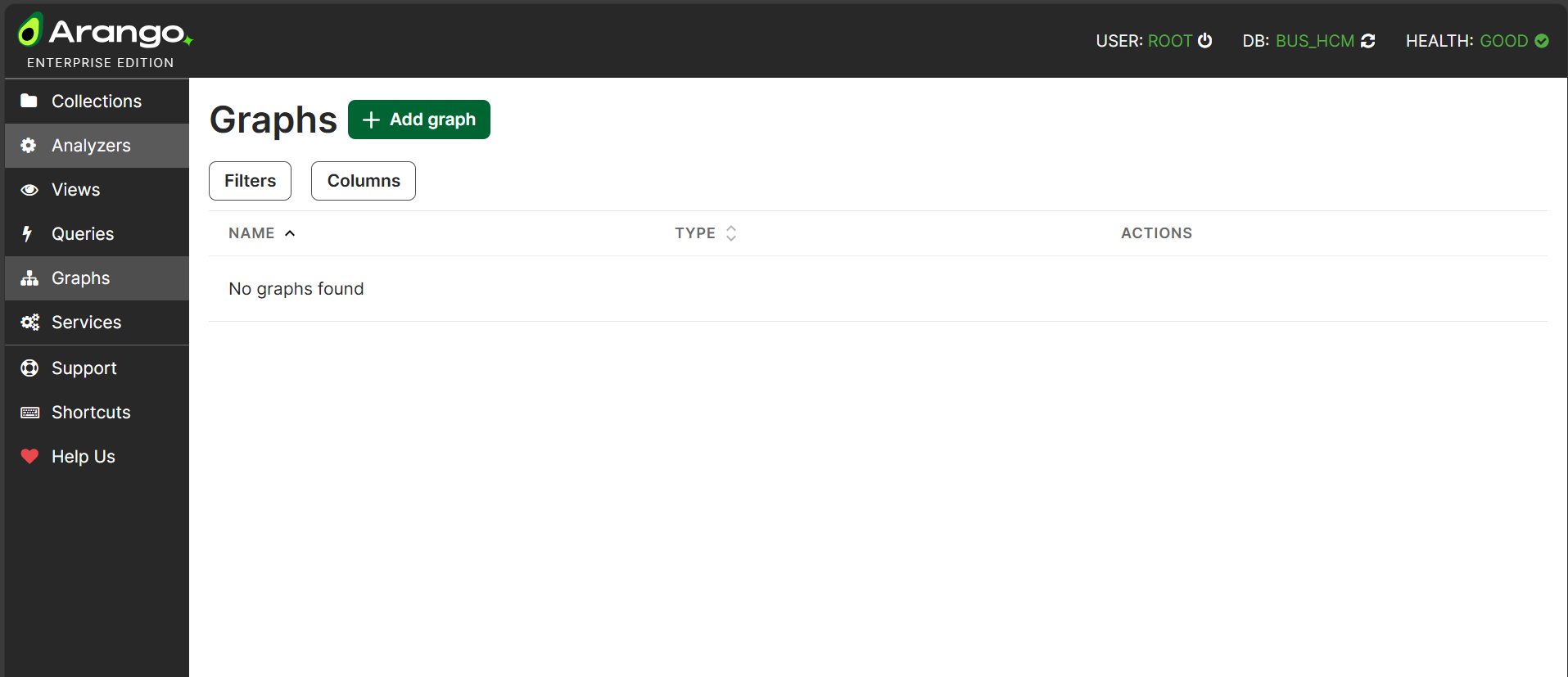
Hình : Tạo collection với type edge

*Bước 2:* Nhấn nút “*Create”,* để tạo thành công collection

**

Hình : Danh sách collection tại mục Collections

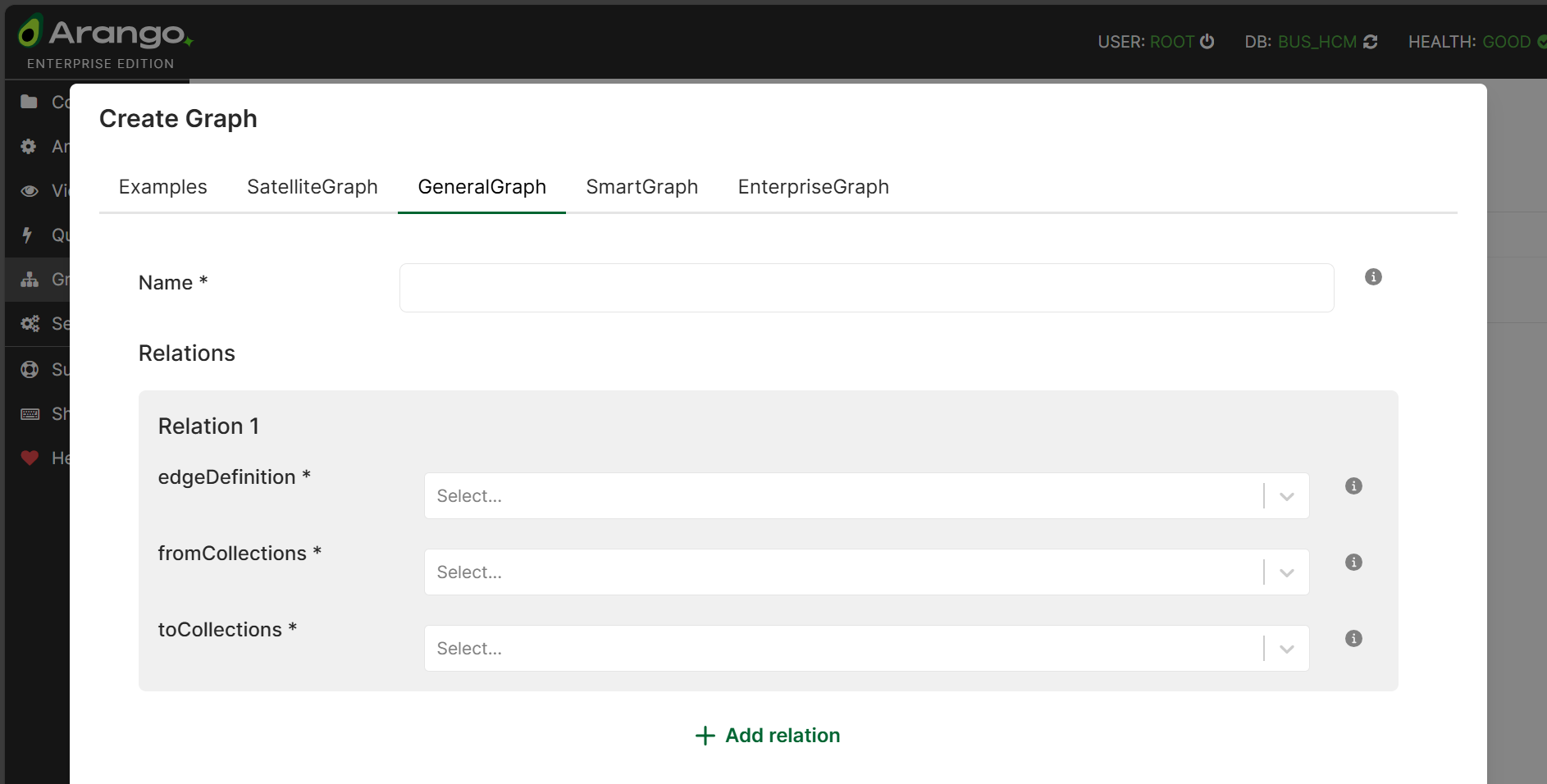
*Bước 4:* Tại giao diện *Graphs* của Arango, nhất nút*“+Add graph”*



Hình : Giao diện Graphs trên ArangoDb Web Interface

Bước 5: Tại giao diện *Create Graph,* chọn tab *“General Graph”*

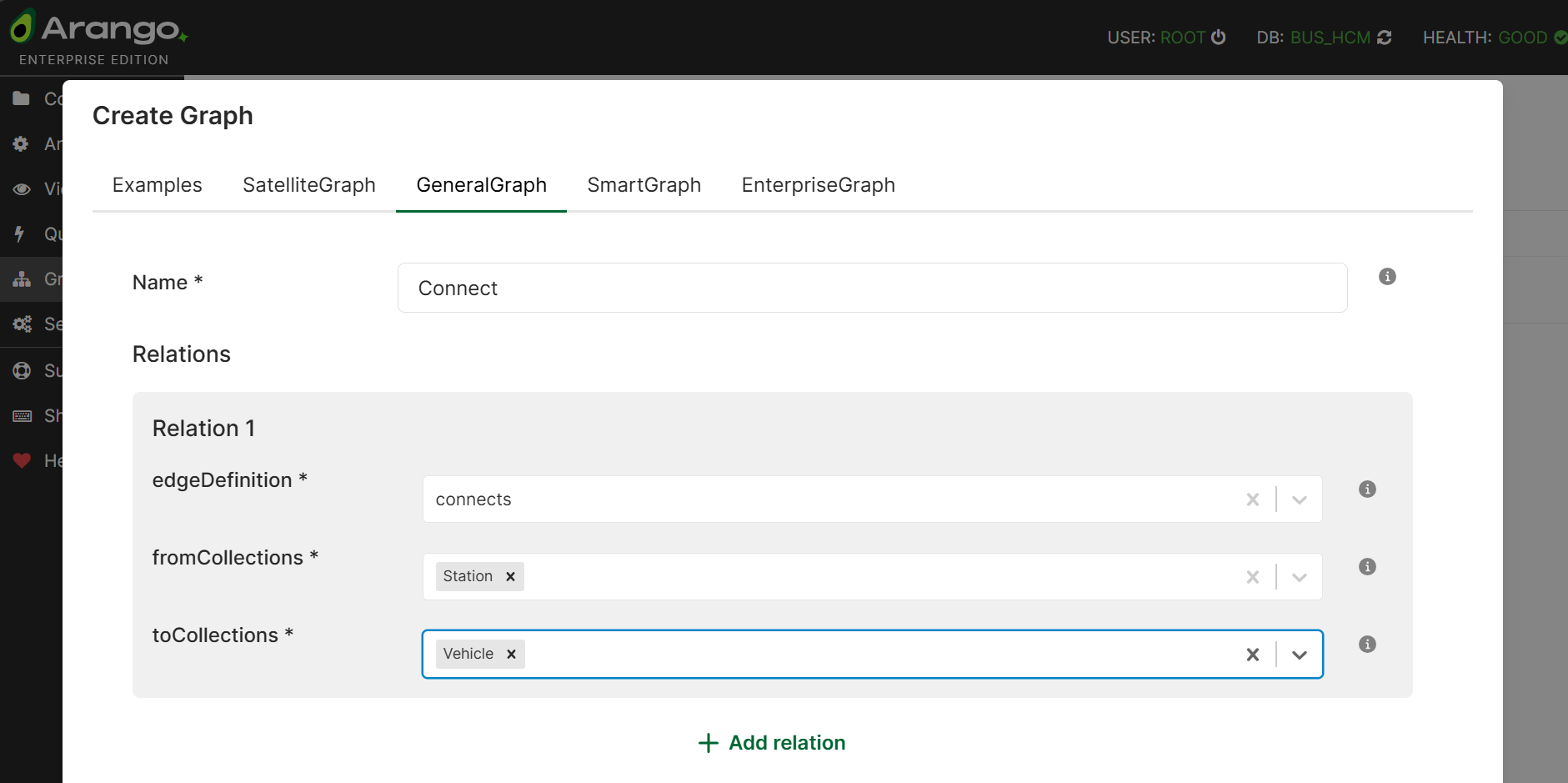
* Examples: *Đ*ây là graph mẫu do ArangoDB tạo sẵn - dùng để học, tham khảo cấu trúc graph.
* Satellite Graph:Chỉ có ý nghĩa khi chạy ArangoDB trong cluster (nhiều server).
* General Graph: Đây là loại graph cơ bản nhất của ArangoDB - đồ thị chuẩn, dùng trong hầu hết các ứng dụng.
* SmartGraph: SmartGraph là graph tối ưu trong phiên bản Enterprise + Cluster.
* Enterprise Graph:Chỉ dùng cho enterprise deployment.



Hình : Giao diện Create Graphs trên ArangoDb Web Interface

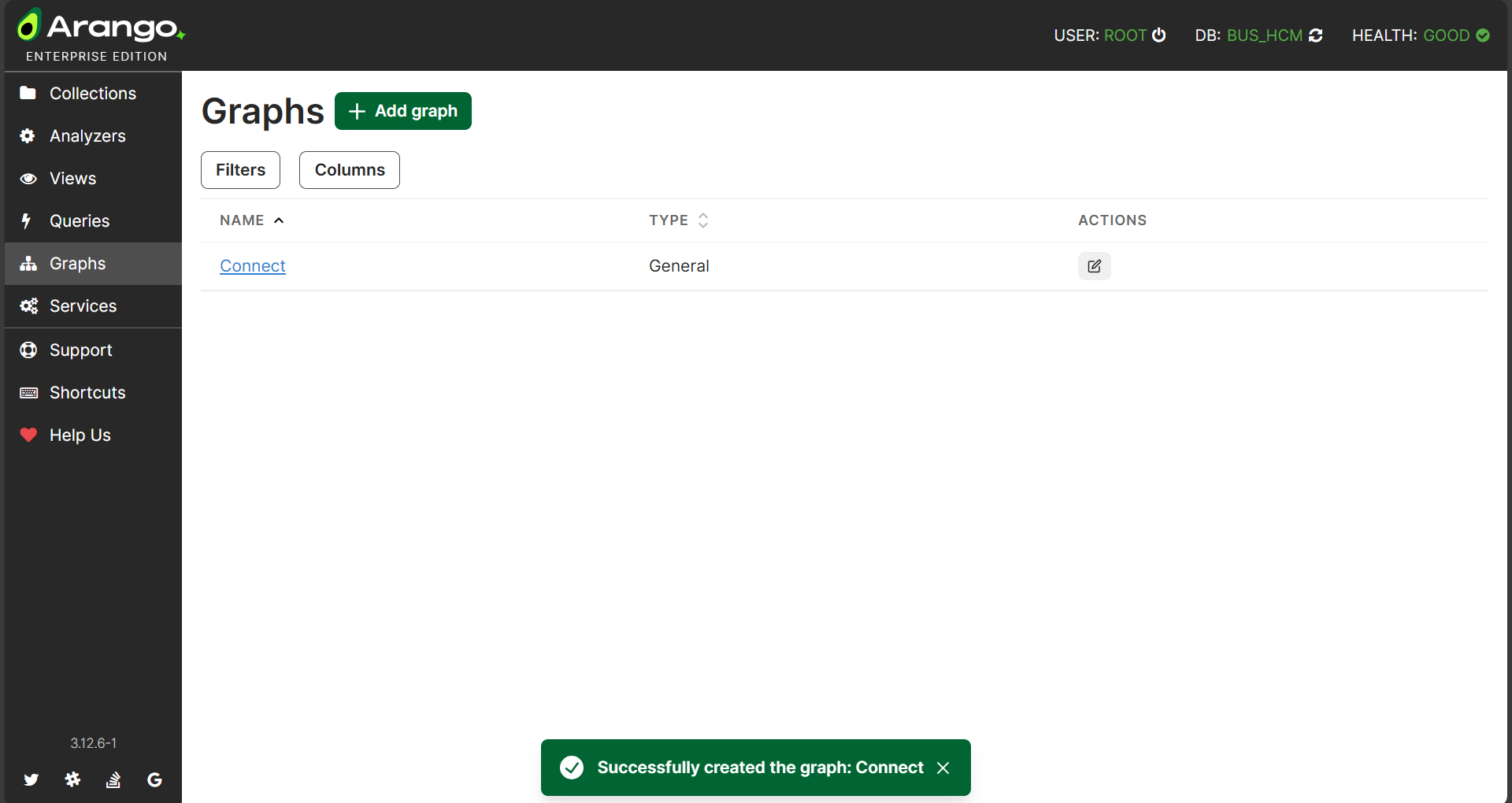
*Bước 6:* Nhập *name* và các mục của *relation*  và nhấn nút *“Create”*

Lưu ý: Nếu muốn thêm relation, thì chọn nhấn *“ + Add relation”*



Hình : Thực hiện tạo Graphs trên ArangoDb Web Interface

*Bước 7:* Danh sách các graphs đã tạo

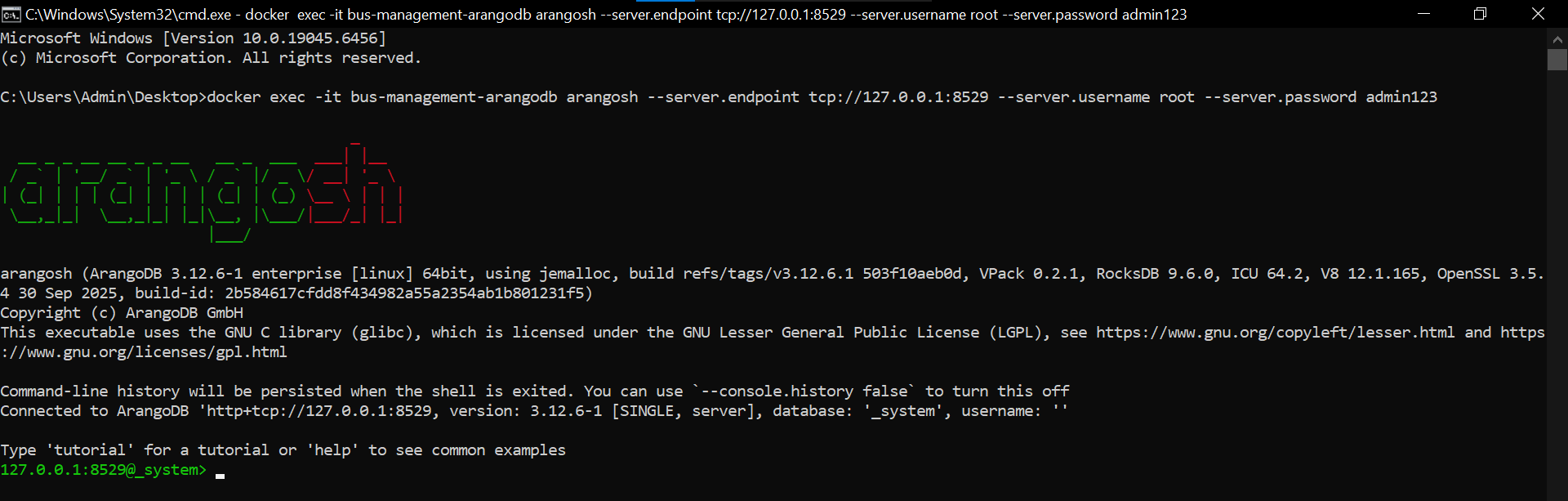


Hình : Danh sách Graphs trên ArangoDb Web Interface

* *Cách 2:Tạo collection trên Terminal/Command Prompt - thực hiện các bước sau:*

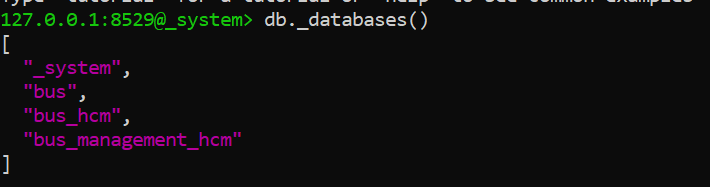
*Bước 1:* Mở Terminal/Command Prompt - mở một cửa sổ Terminal (trên Linux/macOS) hoặc Command Prompt/PowerShell (trên Windows)

*Bước 2:* Tại cmd chạy *“docker exec -it bus-management-arangodb arangosh --server.endpoint tcp://127.0.0.1:8529 --server.username root --server.password admin123”*



Hình : Kết nối với ArangoDB ở chế độ dòng lệnh trên cmd

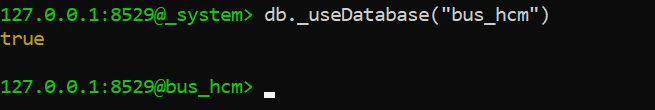
Bước 3: Chạy lệnh *“db.\_databases()”* để kiểm tra tất cả các database của Arangodb



Hình : Kết quả hiển thị danh sách database ở chế độ dòng lệnh

Bước 4: Chạy lệnh *“db.\_useDatabase(“ten\_database”)”* để truy cập vào đúng database

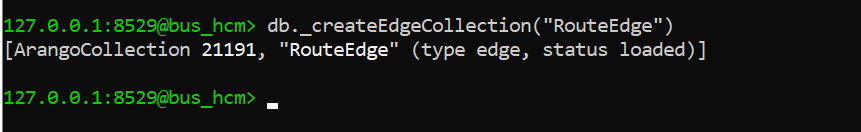
Ví dụ: Thực thi lệnh *“db.\_useDatabase(“bus\_hcm”)”* - để truy cập vào database bus\_hcm



Hình : Chuyển sang database bus\_hcm bằng arangosh

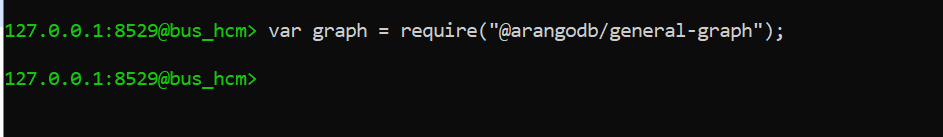
Bước 5: Chạy lệnh *“db.\_createEdgeCollection(“ten\_collectionEdge”)”* để tạo 1 collection với type Edge

Ví dụ: Thực thi lệnh *“db.\_createEdgeCollection(“RouteEdge”)”* để tạo collection với type tên RouteEdge



Hình : Tạo collection RouteEdge với type Edge ở chế độ dòng lệnh

Bước 6: Nạp module quản lý Graph



Hình : Nạp module tạo Graph trong ArangoDB

Bước 7: Tạo graph với tên “BusNetworkGranp” với dòng lệnh

var graph = require("@arangodb/general-graph");

graph.\_create(

"BusNetworkGraph",

[

{

collection: "RouteEdge",

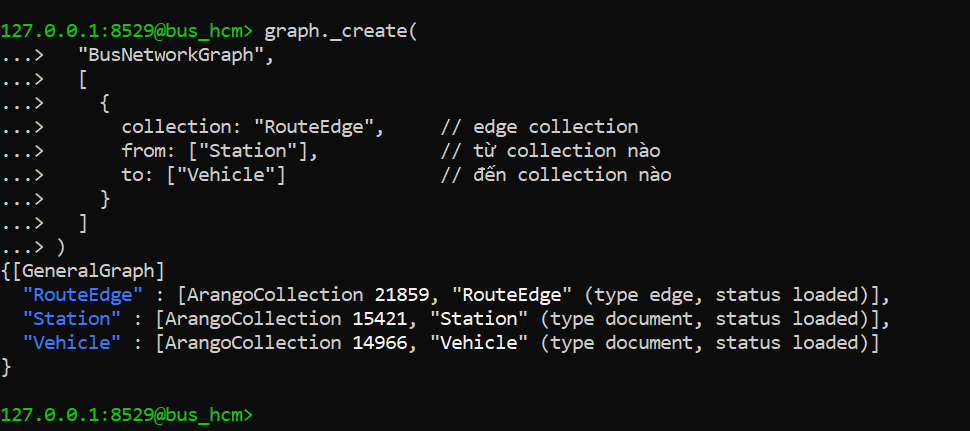
from: ["Station"],

to: ["Vehicle"]

}

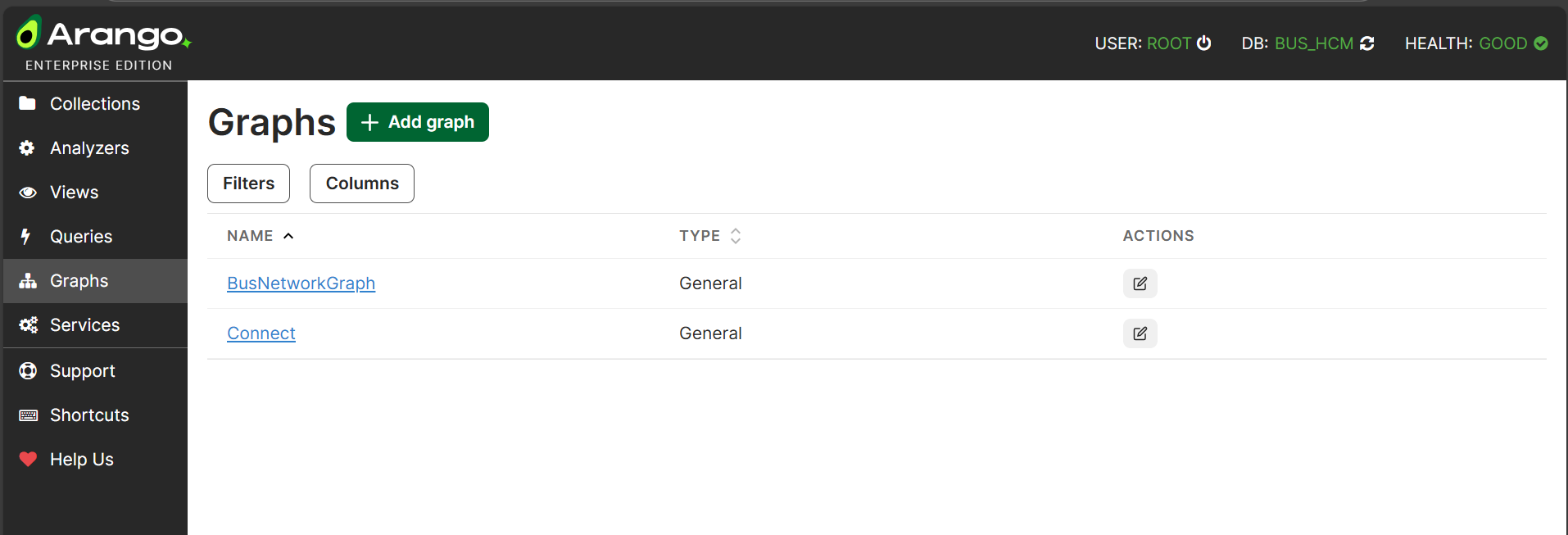
]

)



Hình: Cấu trúc tạo Graph trong CMD

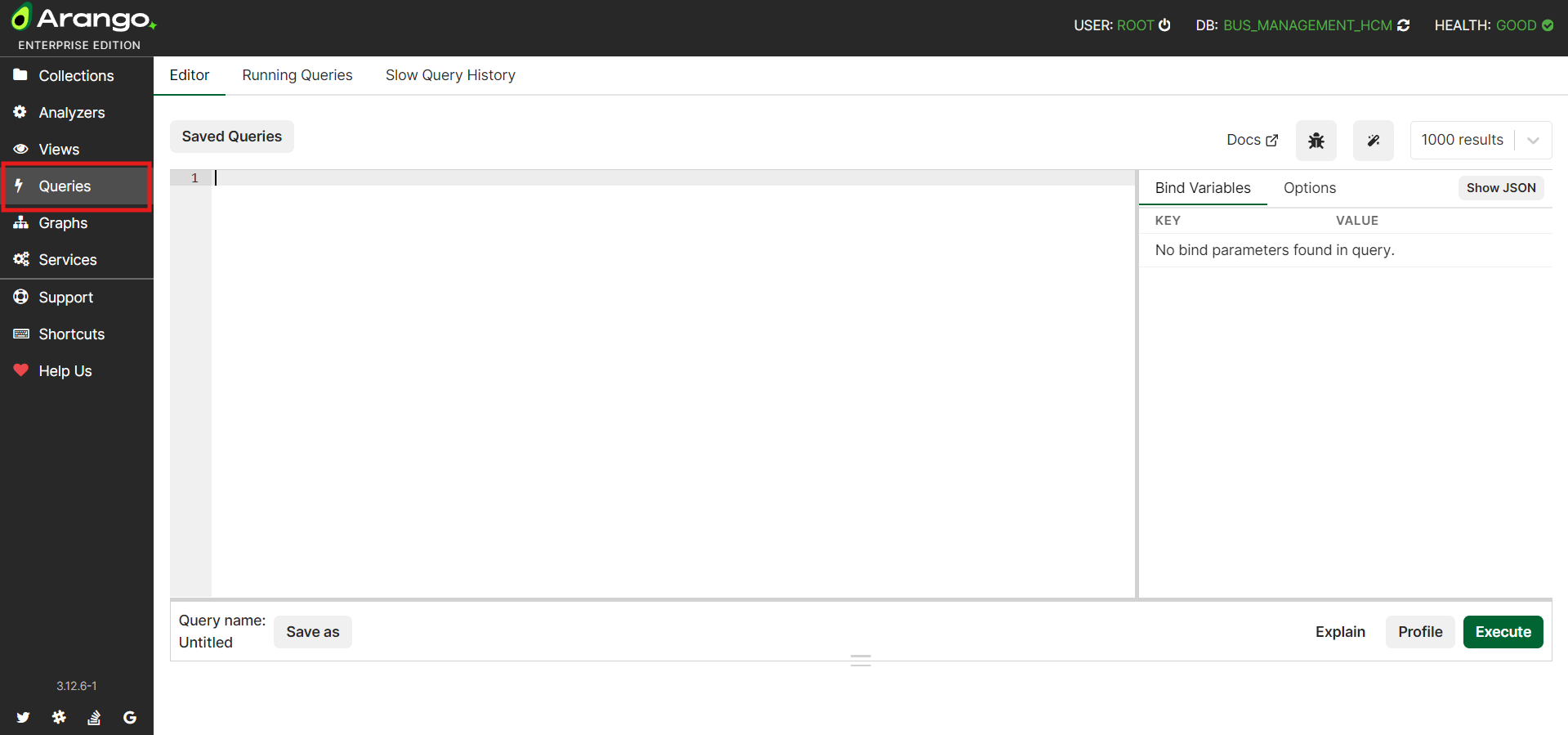
Bước 8: Đăng nhập vào ArangoDb Web Interface vào mục *Graph*, hiển thị danh sách graph đã tạo



Hình : Danh sách Graphs của db bus\_hcm trên ArangoDb Web Interface

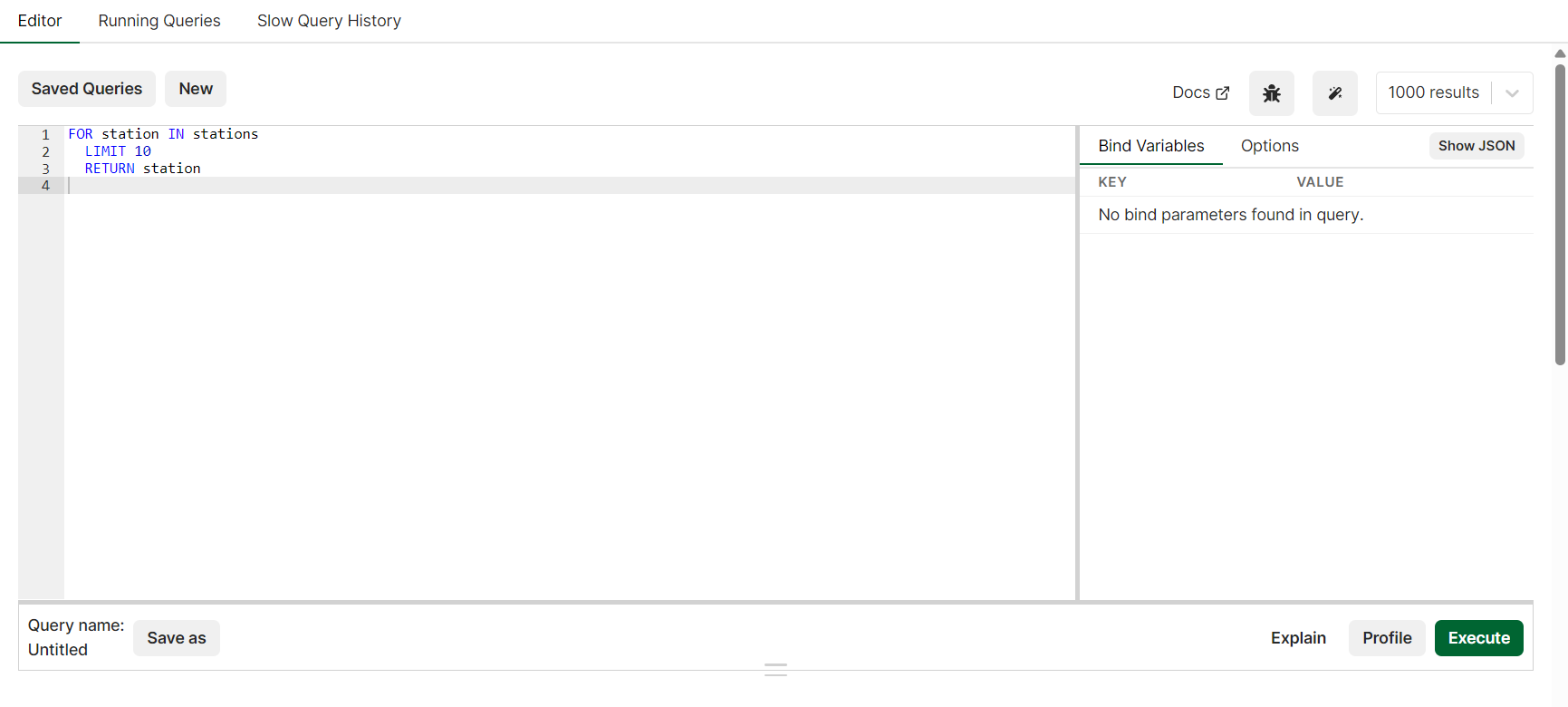
4.4. AQL Query Editor (Trình truy vấn dữ liệu)  
AQL Query Editor là một trong những công cụ mạnh mẽ và được sử dụng thường xuyên nhất trong ArangoDB Web Interface. Nó cung cấp một môi trường tương tác để viết, thực thi, phân tích và gỡ lỗi các truy vấn AQL (ArangoDB Query Language). Đây là công cụ không thể thiếu trong quá trình phát triển, giúp lập trình viên kiểm thử logic truy vấn trước khi tích hợp vào ứng dụng backend.  
*A. Giao diện chính của AQL Query Editor*Để truy cập, ta chọn tab QUERIES từ menu bên trái trong Web Interface. Giao diện chính được chia thành các khu vực chức năng rõ ràng:

* Vùng soạn thảo (Query Editor): Nơi để viết các câu lệnh AQL. Trình soạn thảo này hỗ trợ tô sáng cú pháp (syntax highlighting), tự động hoàn thành (autocomplete) tên các collection, hàm AQL, giúp việc viết query trở nên nhanh chóng và chính xác hơn.
* Vùng kết quả (Result): Hiển thị kết quả trả về sau khi thực thi truy vấn. Kết quả có thể được xem dưới nhiều định dạng khác nhau.
* Bảng Bind Parameters: Cho phép định nghĩa các biến ràng buộc (@variable) để truyền tham số vào truy vấn một cách an toàn, tránh được các lỗi tấn công AQL Injection.

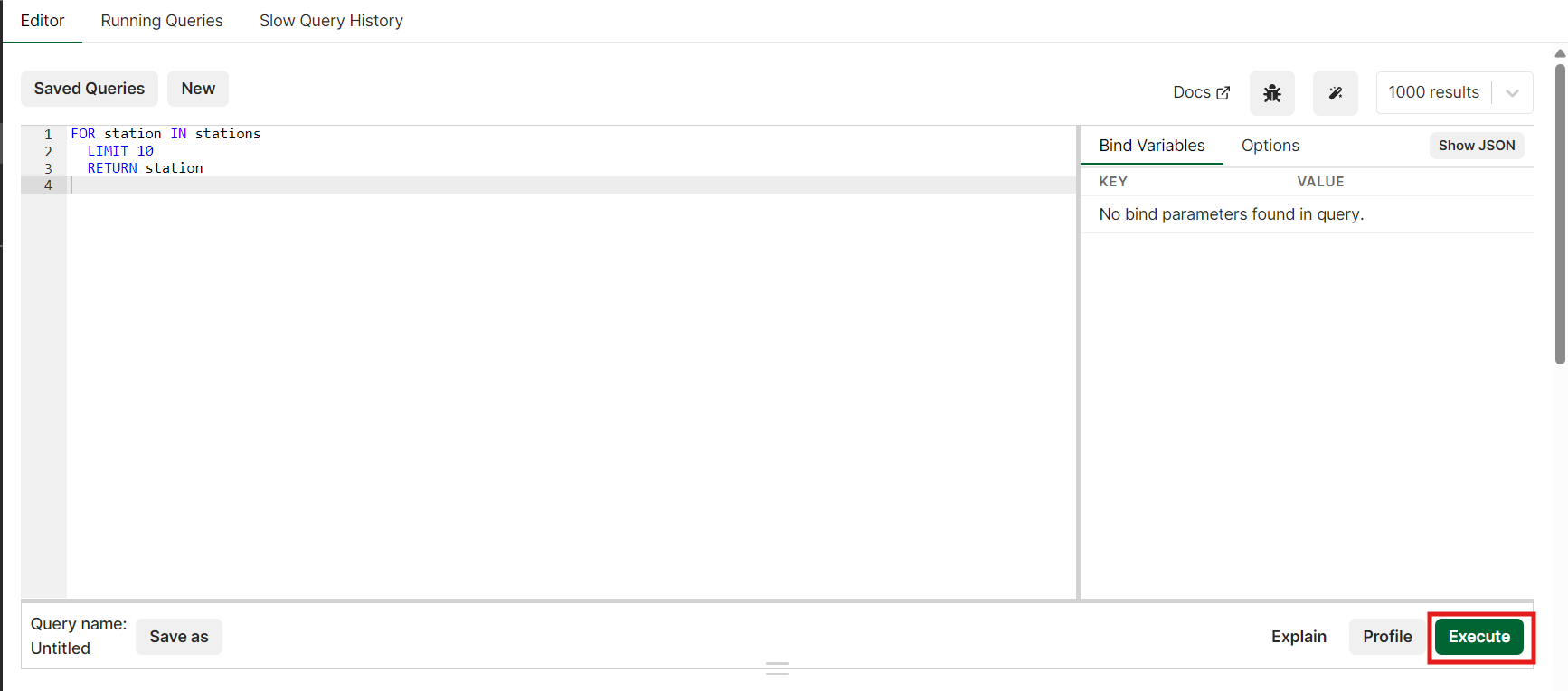
  
Các nút chức năng:

* Execute: Chạy truy vấn hiện tại.
* Explain: Phân tích và hiển thị kế hoạch thực thi (execution plan) của truy vấn mà không thực sự chạy nó.
* Save: Lưu truy vấn lại để tái sử dụng sau này.
* Query History: Xem lại lịch sử các truy vấn đã thực thi.

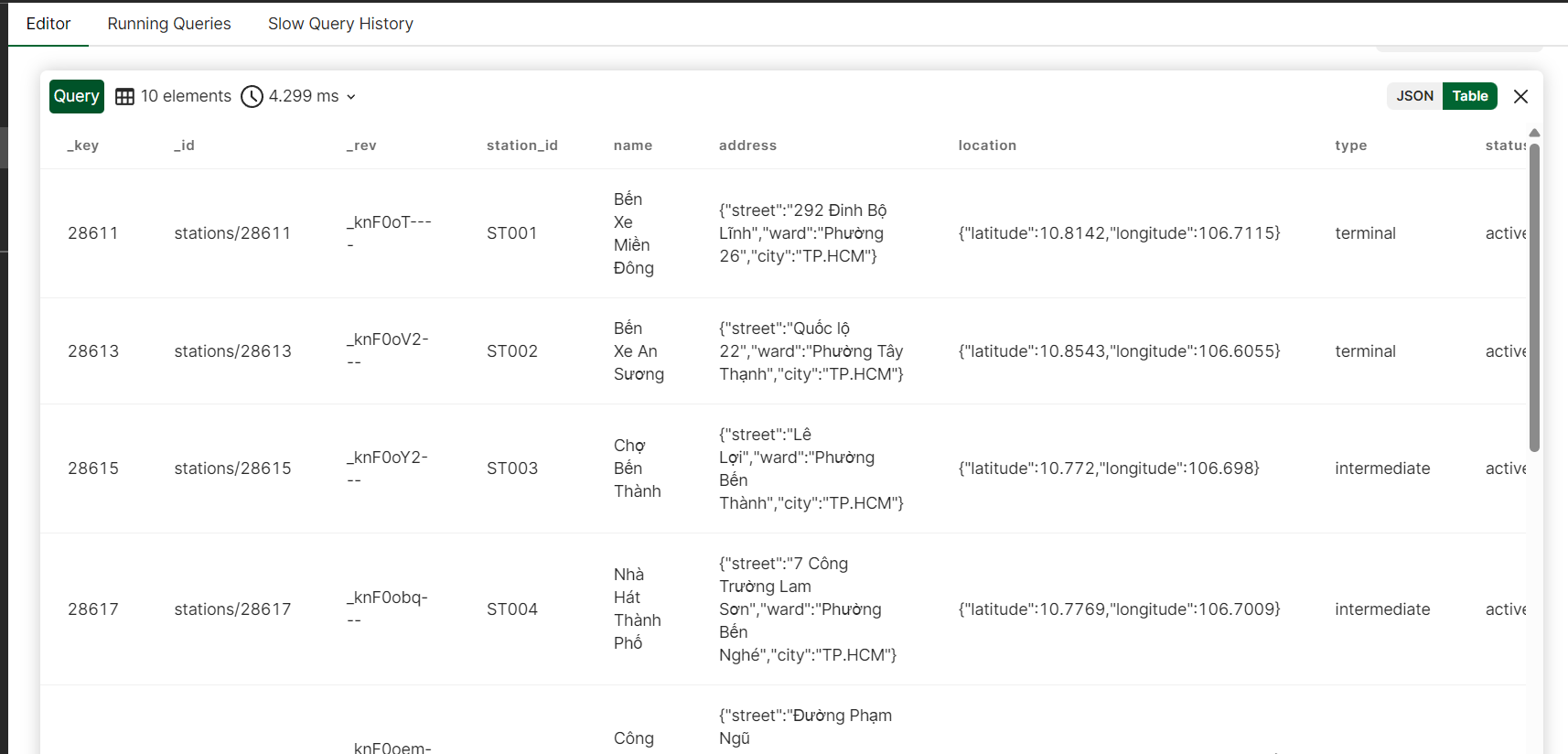
*B. Thực thi một truy vấn cơ bản*Bước 1: Viết truy vấn  
Trong vùng soạn thảo, nhập một câu lệnh AQL đơn giản. Ví dụ, để lấy 10 trạm xe buýt đầu tiên:  
*FOR station IN stations  
LIMIT 10  
RETURN station*

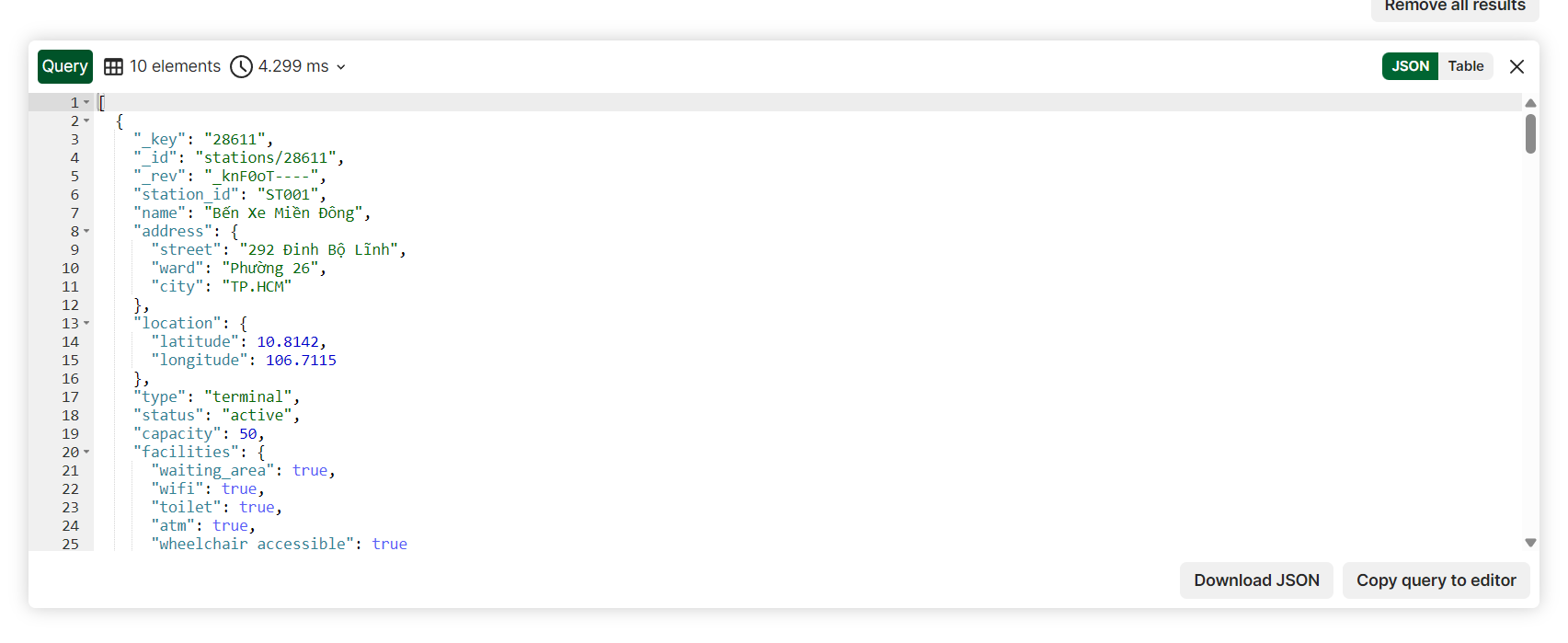
**

Bước 2: Thực thi  
Nhấn nút Execute (hoặc dùng phím tắt Ctrl + Enter).

Bước 3: Xem kết quả  
Kết quả sẽ được hiển thị ngay lập tức trong vùng bên dưới. Bạn có thể chuyển đổi giữa các chế độ xem:

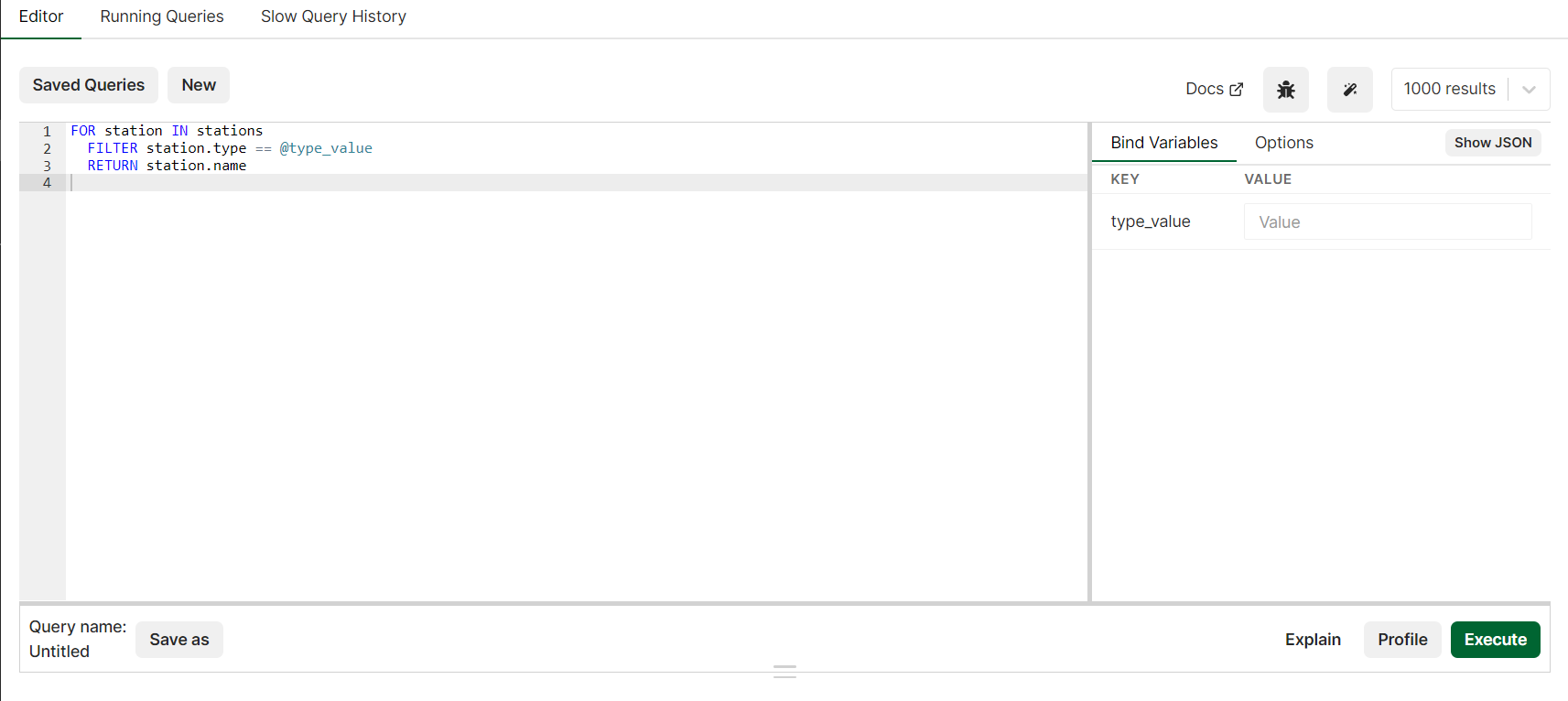
* JSON: Hiển thị kết quả dưới dạng JSON, dễ đọc và copy.
* Table: Hiển thị kết quả dưới dạng bảng, phù hợp cho dữ liệu có cấu trúc phẳng.
* Graph: Nếu kết quả trả về là một đồ thị (các đỉnh và cạnh), chế độ này sẽ trực quan hóa đồ thị đó. Đây là tính năng cực kỳ hữu ích để gỡ lỗi các truy vấn traversal.



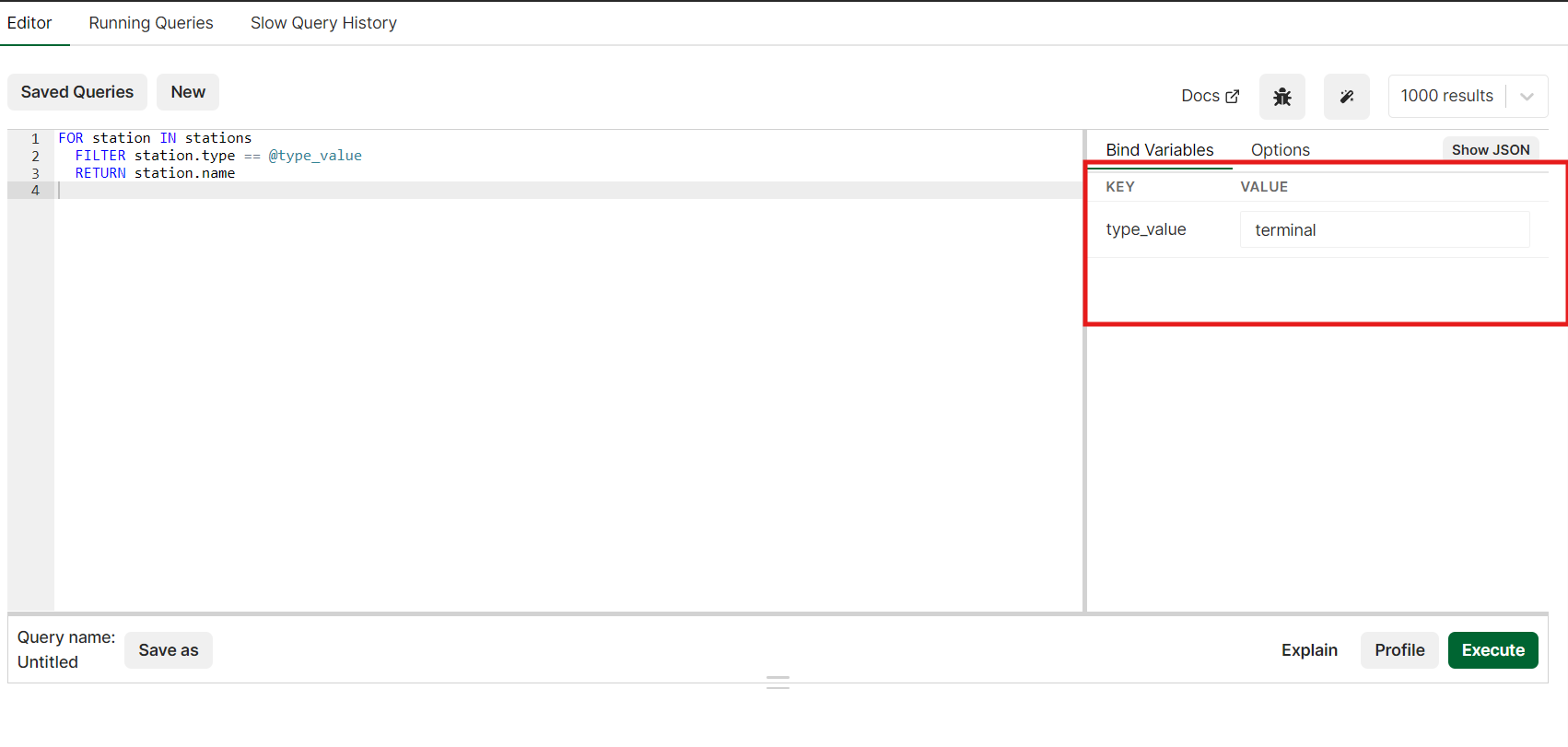


*C. Sử dụng Bind Parameters để tăng tính bảo mật và tái sử dụng*Việc nối chuỗi để tạo câu truy vấn là một hành vi không an toàn. Bind Parameters cho phép truyền giá trị vào truy vấn một cách an toàn.  
Bước 1: Viết truy vấn với biến ràng buộc

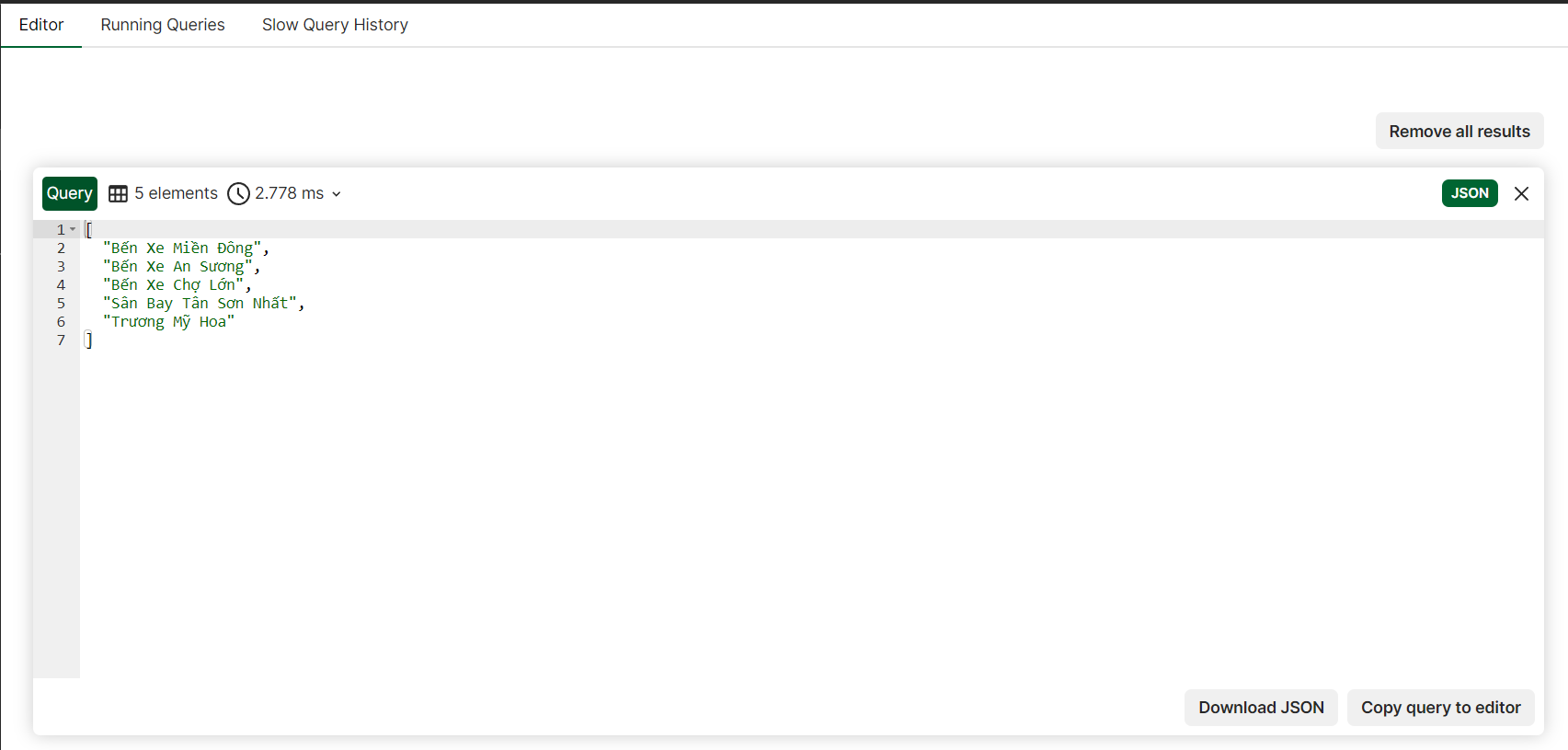
Viết một truy vấn tìm kiếm trạm xe buýt theo loại, sử dụng biến @type\_value.  
*FOR station IN stations  
FILTER station.type == @type\_value  
RETURN station.name*



Bước 2: Định nghĩa giá trị cho biến  
Trong bảng Bind Parameters, nhập type\_value vào cột Key và terminal vào cột Value.



Bước 3: Thực thi  
Nhấn Execute. ArangoDB sẽ thay thế @type\_value bằng giá trị terminal một cách an toàn và thực thi truy vấn.

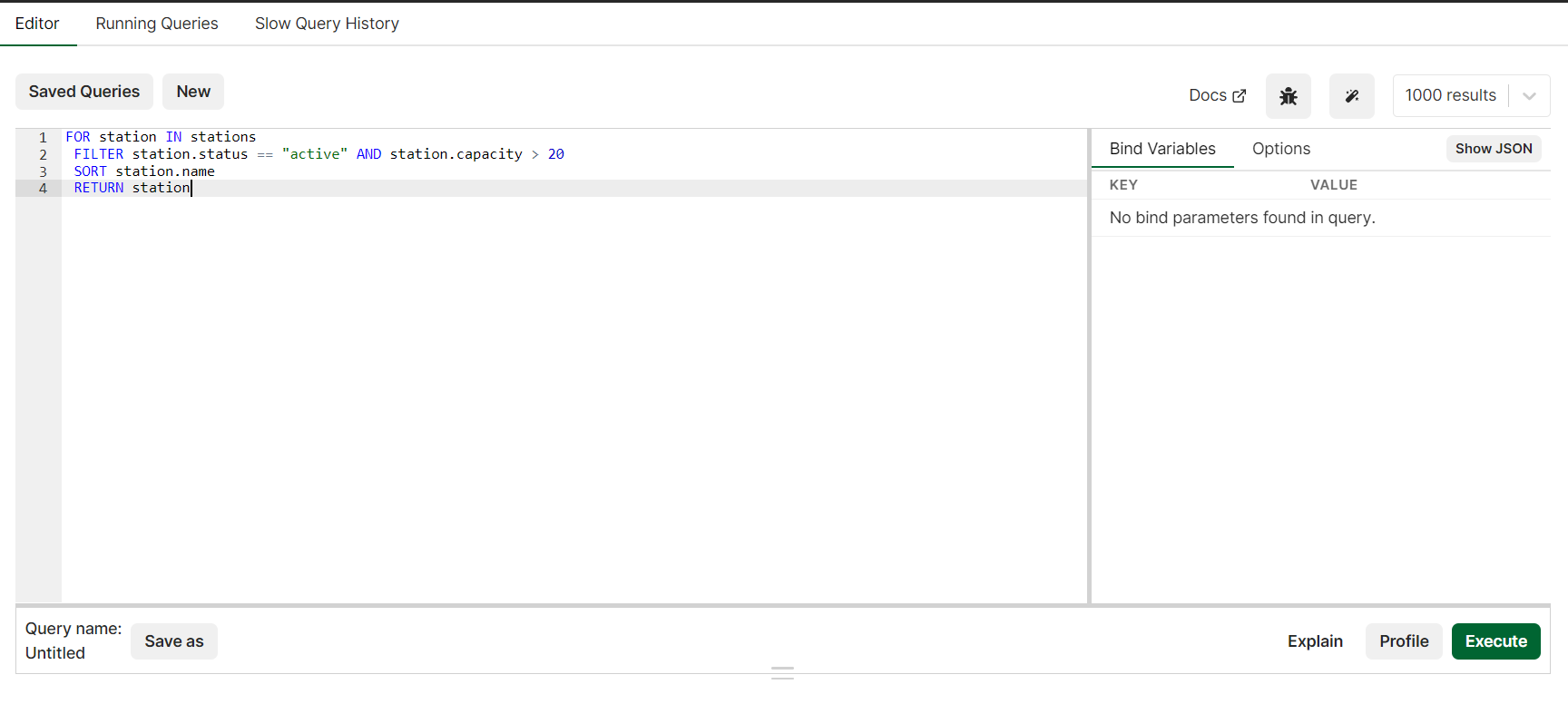


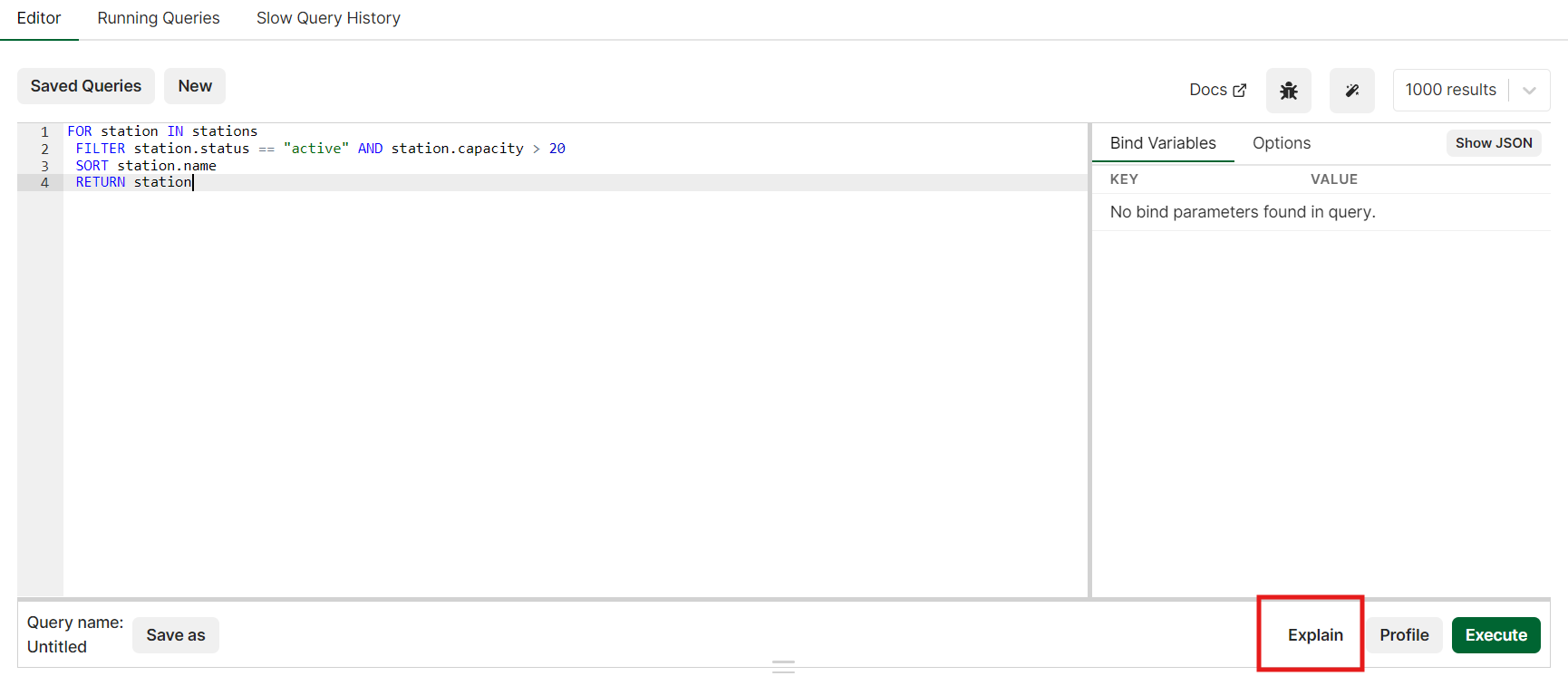
*D. Phân tích và Tối ưu hóa truy vấn với*Tính năng Explain là công cụ mạnh mẽ nhất để hiểu cách ArangoDB thực thi một câu truy vấn và tìm ra điểm nghẽn về hiệu năng.  
**Bước 1**: Nhập truy vấn cần phân tích  
Ví dụ, một truy vấn lọc phức tạp:  
*FOR station IN stations*

*FILTER station.status == "active" AND station.capacity > 20*

*SORT station.name*

*RETURN station*

**Bước 2**: Nhấn nút   
Thay vì chạy, ArangoDB sẽ phân tích và hiển thị Execution Plan (Kế hoạch thực thi).

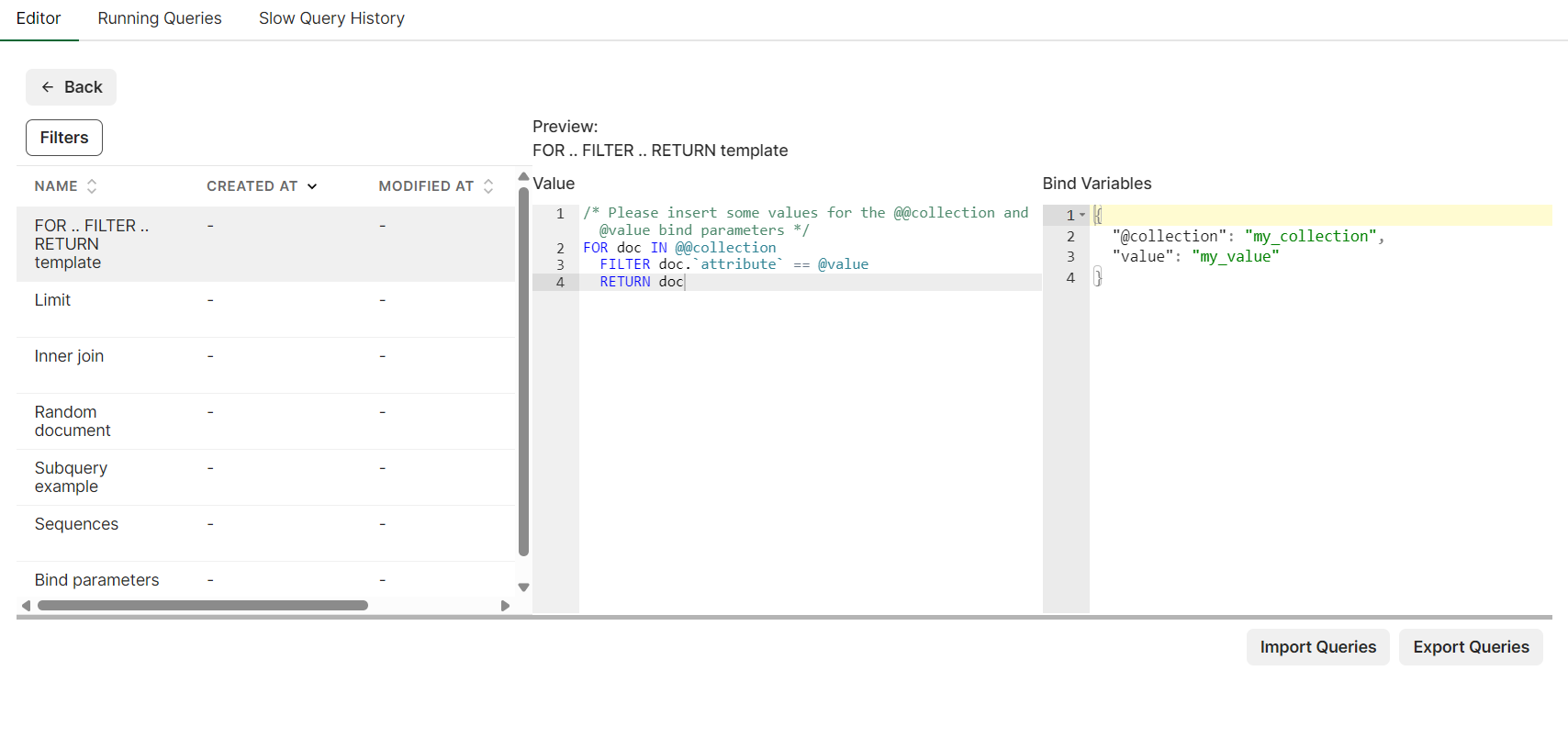
  
 Hình : Kế hoạch thực thi (Execution Plan)   
**Bước 3**: Đọc và phân tích kế hoạch



Kế hoạch thực thi cho biết:

* Các bước thực thi (Nodes): ArangoDB sẽ làm gì tuần tự (ví dụ: quét toàn bộ collection, lọc, sắp xếp).
* Số lượng item ước tính: Số lượng document được xử lý ở mỗi bước.
* Sử dụng Index: Quan trọng nhất, nó cho biết truy vấn có đang sử dụng index hay không. Nếu một bước FilterNode có dòng chữ "uses index", điều đó có nghĩa là truy vấn đang được tối ưu. Ngược lại, nếu nó phải quét toàn bộ collection (Full Collection Scan), đó là dấu hiệu cho thấy bạn cần thêm một index phù hợp.

→ Ví dụ, nếu kế hoạch cho thấy một "Full Collection Scan" với hàng triệu document chỉ để lọc ra vài document, người quản trị sẽ biết ngay cần phải tạo một skiplist index trên trường status và capacity để tăng tốc truy vấn.  
*E. Lưu và Tái sử dụng Truy vấn*Đối với các truy vấn phức tạp hoặc thường xuyên sử dụng, bạn có thể lưu chúng lại.  
Bước 1: Viết và kiểm thử truy vấn hoàn chỉnh.  
Bước 2: Nhấn nút Save. Đặt tên cho truy vấn (ví dụ: "Find\_Busiest\_Stations").  
Bước 3: Để mở lại, click vào mục Custom Queries ở menu bên trái của AQL Editor. Tất cả các truy vấn đã lưu sẽ xuất hiện ở đó.

**4.5. Import / Export dữ liệu**

# Import và Export là hai tác vụ cơ bản nhưng cực kỳ quan trọng trong vòng đời của một dự án. Chúng cho phép di chuyển dữ liệu giữa các môi trường hoặc tích hợp với các hệ thống khác. ArrangoDB hỗ trợ 2 phương thức chính cho việc này:

# *Cách 1: Sử dụng ArangoDB Web Interface*

# Phương pháp này phù hợp cho việc nhập (import) các file dữ liệu nhỏ một cách nhanh chóng.

# Bước 1: Chuẩn bị file dữ liệu

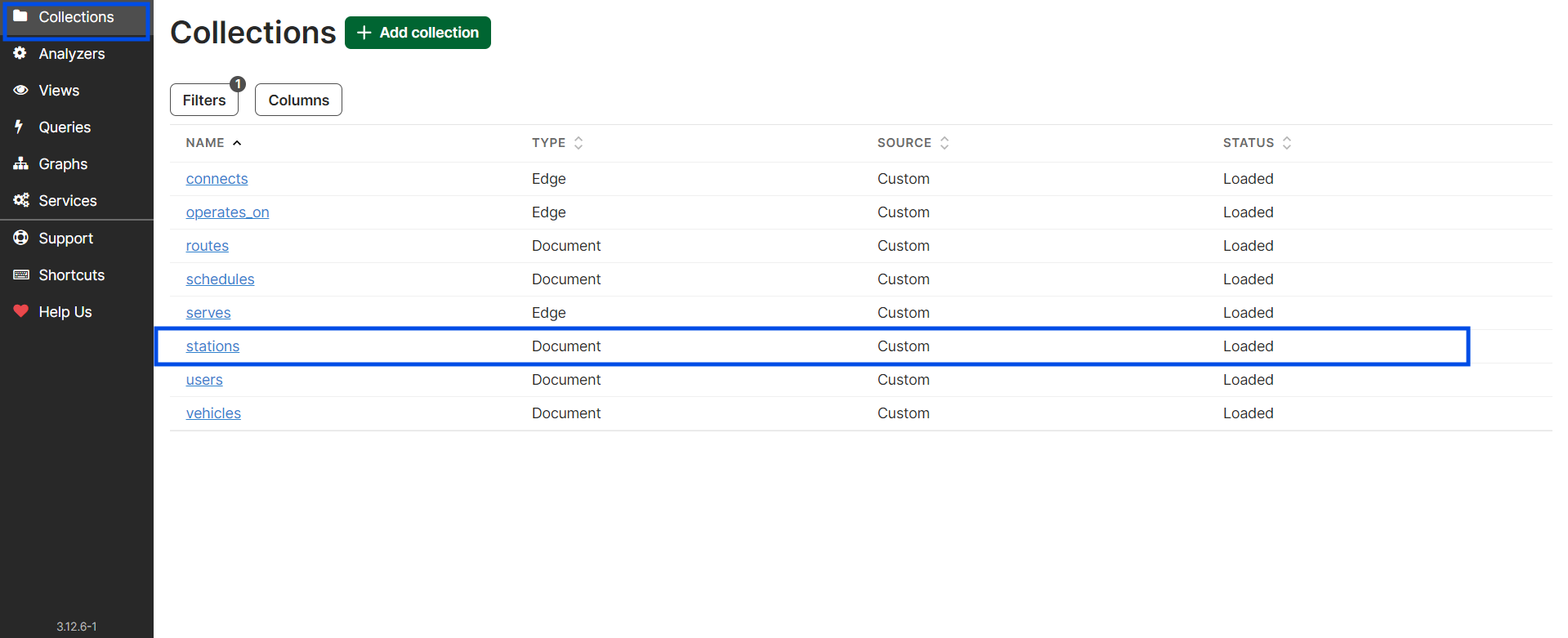
# Tạo một file JSON chứa các document cần import, ví dụ stations\_import.json. Mỗi document nằm trên một dòng riêng biệt (JSONL format).

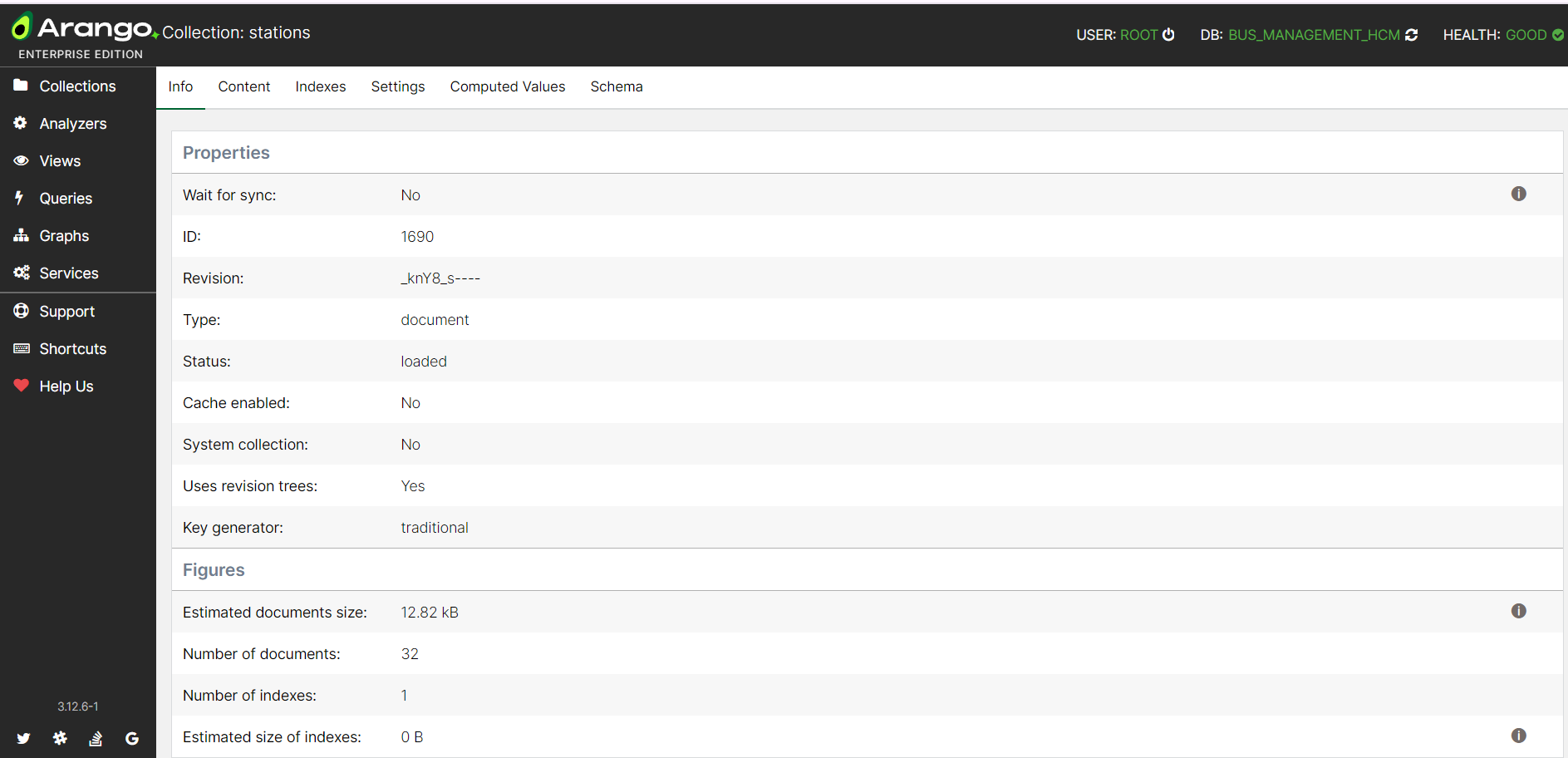
*{"\_key":"ST101", "name":"Trạm Mẫu 1", ...}*

*{"\_key":"ST102", "name":"Trạm Mẫu 2", ...}*

# Bước 2: Truy cập Collection

# Trong Web Interface, vào tab COLLECTIONS và chọn collection bạn muốn import dữ liệu vào (ví dụ: stations).





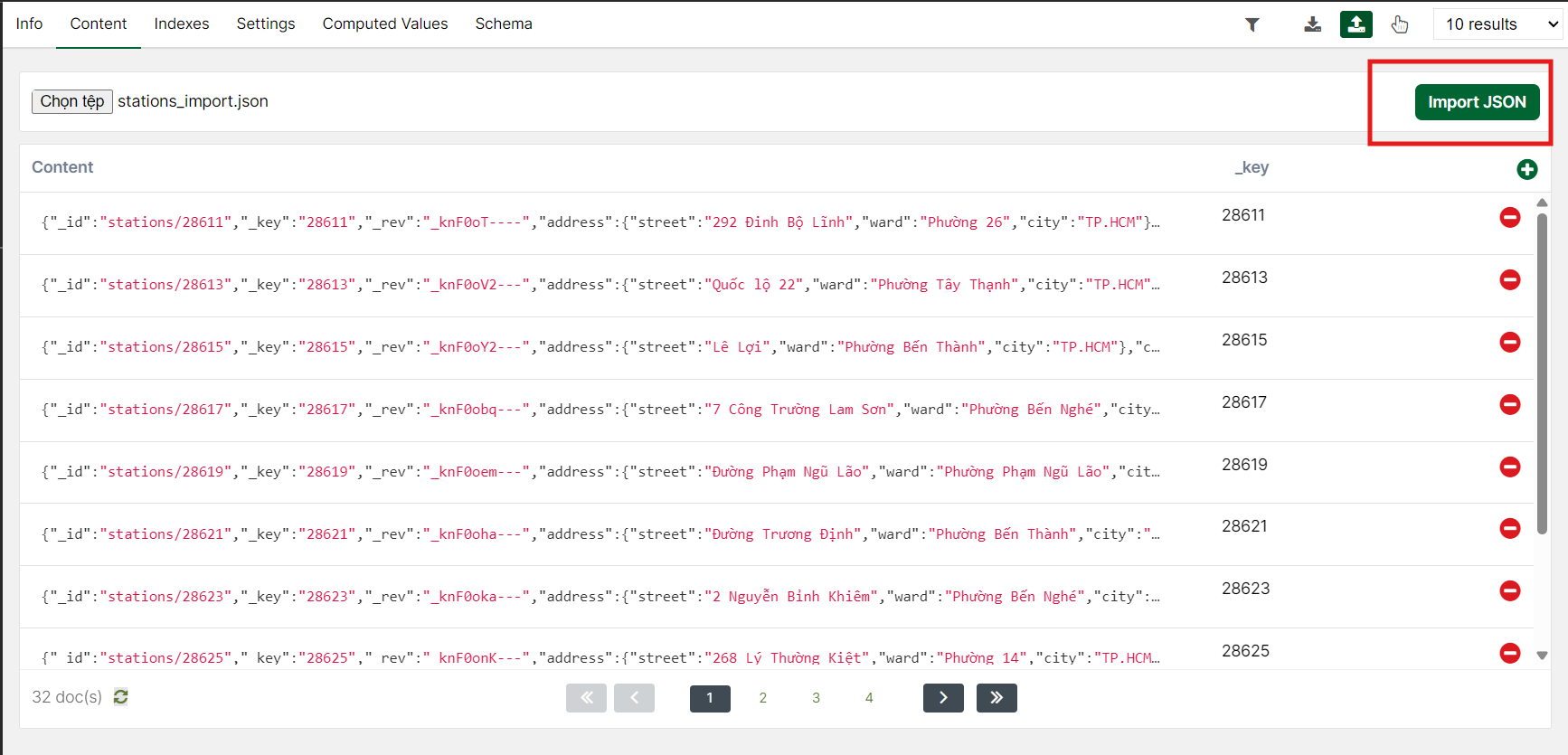
# Bước 3: Vào tab Content và tải dữ liệu lên

# Nhấn vào biểu tượng upload (mũi tên đi lên) ở góc trên bên phải của danh sách document.



# Bước 4: Chọn file và Import

# Một hộp thoại sẽ hiện ra, cho phép bạn chọn file từ máy tính và cấu hình các tùy chọn. Sau khi chọn file, nhấn nút Import.

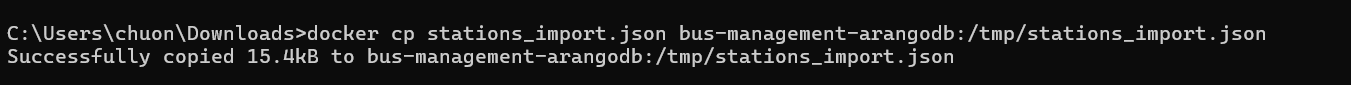


# *Cách 2: Sử dụng Công cụ dòng lệnh (arangoimport) với Docker*

# Bước 1: Chuẩn bị file và sao chép vào container

# Giả sử bạn có file stations\_import.json trên máy tính. Đầu tiên, cần sao chép file này vào bên trong Docker container.

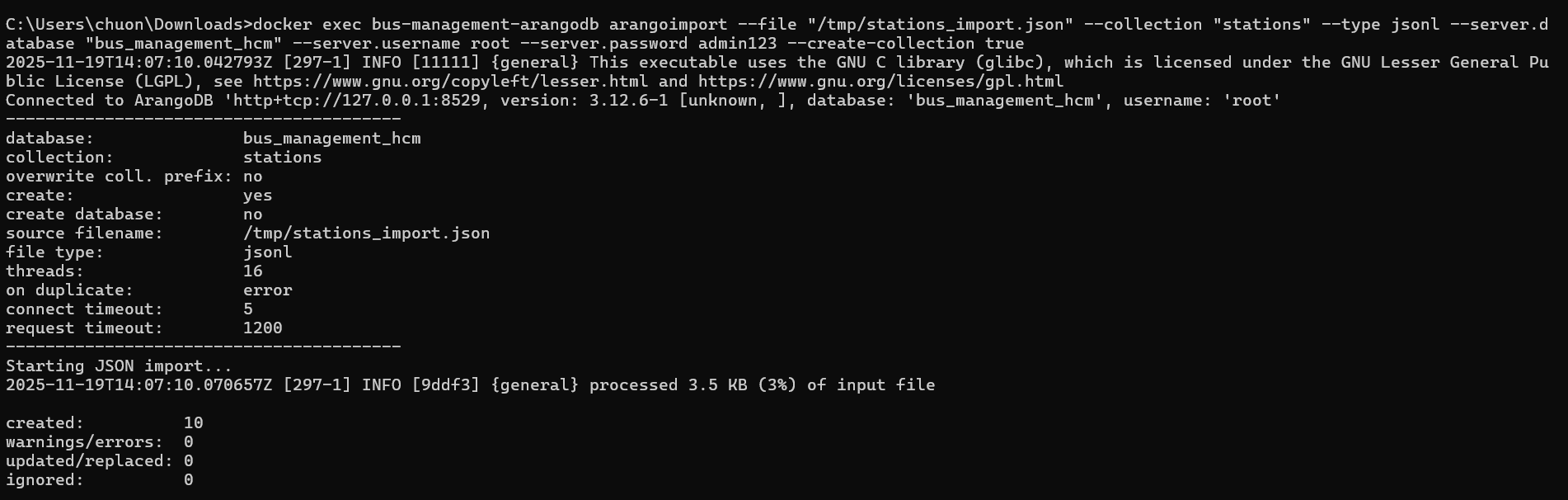
# Mở terminal và chạy lệnh: *“docker cp stations\_import.json bus-management-arangodb:/tmp/stations\_import.json”*



Bước 2: Thực thi lệnh arangoimport

Sử dụng docker exec để chạy arangoimport bên trong container.

“*docker exec bus-management-arangodb arangoimport --file "/tmp/stations\_import.json" --collection "stations" --type jsonl --server.database "bus\_management\_hcm" --server.username root --server.password admin123 --create-collection true”*

  
**4.6. Backup / Restore dữ liệu**

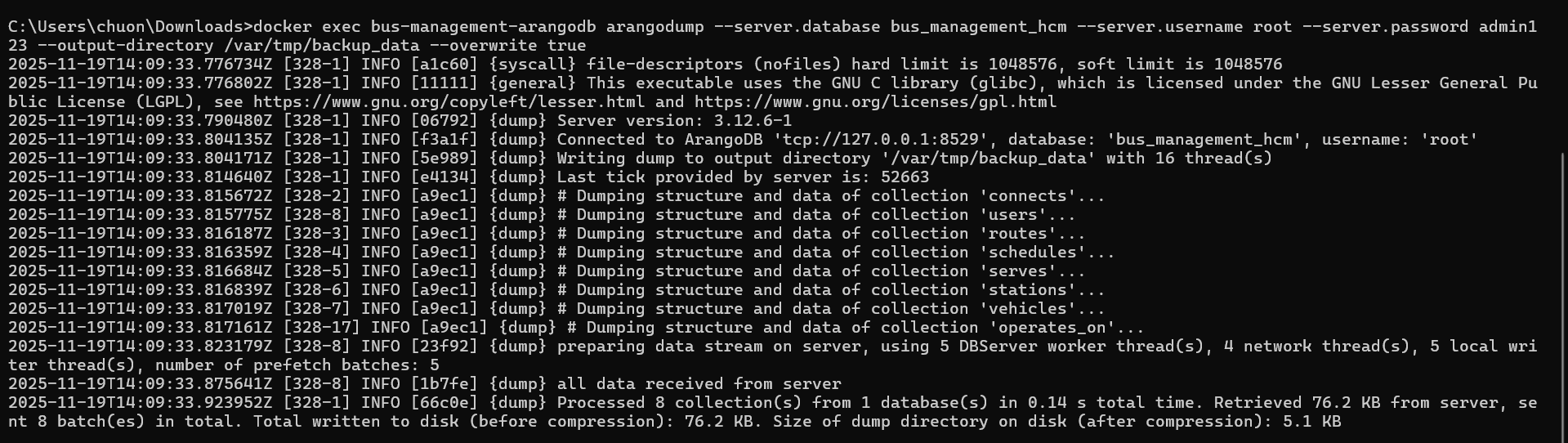
# *Cách 1: Backup dữ liệu (arangodump) với Docker*

Bước 1: Tạo thư mục backup bên trong container Thực thi lệnh arangodump để tạo một thư mục chứa dữ liệu sao lưu bên trong container chạy lệnh

“

*docker exec bus-management-arangodb arangodump --server.database bus\_management\_hcm --server.username root --server.password admin123 --output-directory /var/tmp/backup\_data --overwrite true*

*"*”



Lúc này, dữ liệu sao lưu đang nằm trong thư mục /var/tmp/backup\_data bên trong container.

Bước 2: Sao chép thư mục backup ra máy tính

Để lưu trữ bản backup an toàn, chúng ta cần sao chép nó ra khỏi container bằng lệnh

“

*docker cp bus-management-arangodb:/var/tmp/backup\_data C:\Users\chuon\Downloads\db\_backup\_20251120*

”



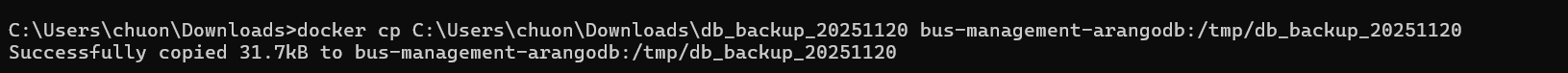
Bây giờ bạn đã có một thư mục db\_backup\_20251120 trên máy tính chứa toàn bộ dữ liệu sao lưu.

# *Cách 1: Restore dữ liệu (arangorestore) với Docker*

**Bước 1**: Sao chép thư mục backup vào container

Giả sử bạn cần phục hồi từ thư mục db\_backup\_20251120. Đầu tiên, sao chép nó vào container bằng cách chạy lệnh

“*docker cp C:\Users\chuon\Downloads\db\_backup\_20251120 bus-management-arangodb:/tmp/db\_backup\_20251120*”



**Bước 2**: Tạo database mới (nếu chưa có)

“docker exec -it bus-management-arangodb arangosh --server.username root --server.password admin123 --javascript.execute-string "db.\_createDatabase('bus\_management\_hcm');"”

**Bước 3**: Thực thi lệnh arangorestore

Sử dụng docker exec để chạy arangorestore restore dữ liệu vào database.

“*docker exec bus-management-arangodb arangorestore --server.username root --server.password admin123 --server.database bus\_management\_hcm --input-directory /tmp/db\_backup\_20251120*

”

**4.7. User Management – phân quyền người dùng**

ArangoDB có một hệ thống quản lý người dùng và phân quyền mạnh mẽ ở cấp độ CSDL, tách biệt hoàn toàn với collection users trong ứng dụng của chúng ta. Chức năng này cho phép kiểm soát ai có thể truy cập vào database nào và với quyền hạn gì.

**Bước 1**: Truy cập mục Users

Trong Web Interface, điều hướng đến tab USERS ở menu bên trái.

**Bước 2**: Tạo người dùng mới

Nhấn nút Add User. Một hộp thoại sẽ hiện ra yêu cầu nhập Username và Password.

**Bước 3**: Phân quyền trên Database

Sau khi tạo, người dùng sẽ xuất hiện trong danh sách. Để phân quyền, click vào tên người dùng đó.

Ở phần Permissions, bạn có thể thiết lập quyền hạn cho từng database.

* Administrator: Toàn quyền trên database đó (tạo/xóa collection, index...).
* Access: Có quyền đọc/ghi dữ liệu trên các collection mà người dùng được cấp phép. Đây là quyền phù hợp cho tài khoản ứng dụng.
* No Access: Không có quyền truy cập.

Đối với ứng dụng quản lý xe buýt, chúng ta có thể tạo một user là bus\_app\_user và cấp quyền Access cho database bus\_management\_hcm.  
**4.8. Các tính năng nâng cao**

* Foxx Services

Khái niệm: Foxx là một framework microservice được tích hợp sẵn trong ArangoDB, cho phép bạn viết các API RESTful bằng JavaScript và chạy chúng trực tiếp bên trong CSDL.

Lợi ích: Loại bỏ độ trễ mạng giữa lớp ứng dụng và CSDL, lý tưởng cho các tác vụ cần xử lý dữ liệu nặng.

Cách sử dụng:

* Truy cập tab SERVICES trong Web Interface.
* Nhấn Add Service.
* Có thể cài đặt từ ArangoDB Store, upload một file zip chứa code, hoặc kết nối với GitHub.
* Index Management

Khái niệm: ArangoDB Web Interface cung cấp các công cụ trực quan để giám sát sức khỏe và hiệu năng của hệ thống.

Các mục giám sát chính:

* Dashboard (tab STATS): Cung cấp cái nhìn tổng quan về tài nguyên hệ thống (CPU, bộ nhớ), số lượng yêu cầu, thông tin phiên bản.
* Thống kê Collection (Collection -> Info): Hiển thị số lượng document, kích thước dữ liệu trên đĩa của từng collection.

Truy vấn (tab QUERIES):

* Running Queries: Xem các truy vấn đang được thực thi.
* Slow Queries: Liệt kê các truy vấn chạy chậm, giúp xác định và tối ưu hóa các AQL kém hiệu quả. Đây là công cụ cực kỳ hữu ích cho DBA.
* System Monitoring

Khái niệm: ArangoDB Web Interface cung cấp các công cụ trực quan để giám sát sức khỏe và hiệu năng của hệ thống.

Các mục giám sát chính:

* Dashboard (tab STATS): Cung cấp cái nhìn tổng quan về tài nguyên hệ thống (CPU, bộ nhớ), số lượng yêu cầu, thông tin phiên bản.
* Thống kê Collection (Collection -> Info): Hiển thị số lượng document, kích thước dữ liệu trên đĩa của từng collection.

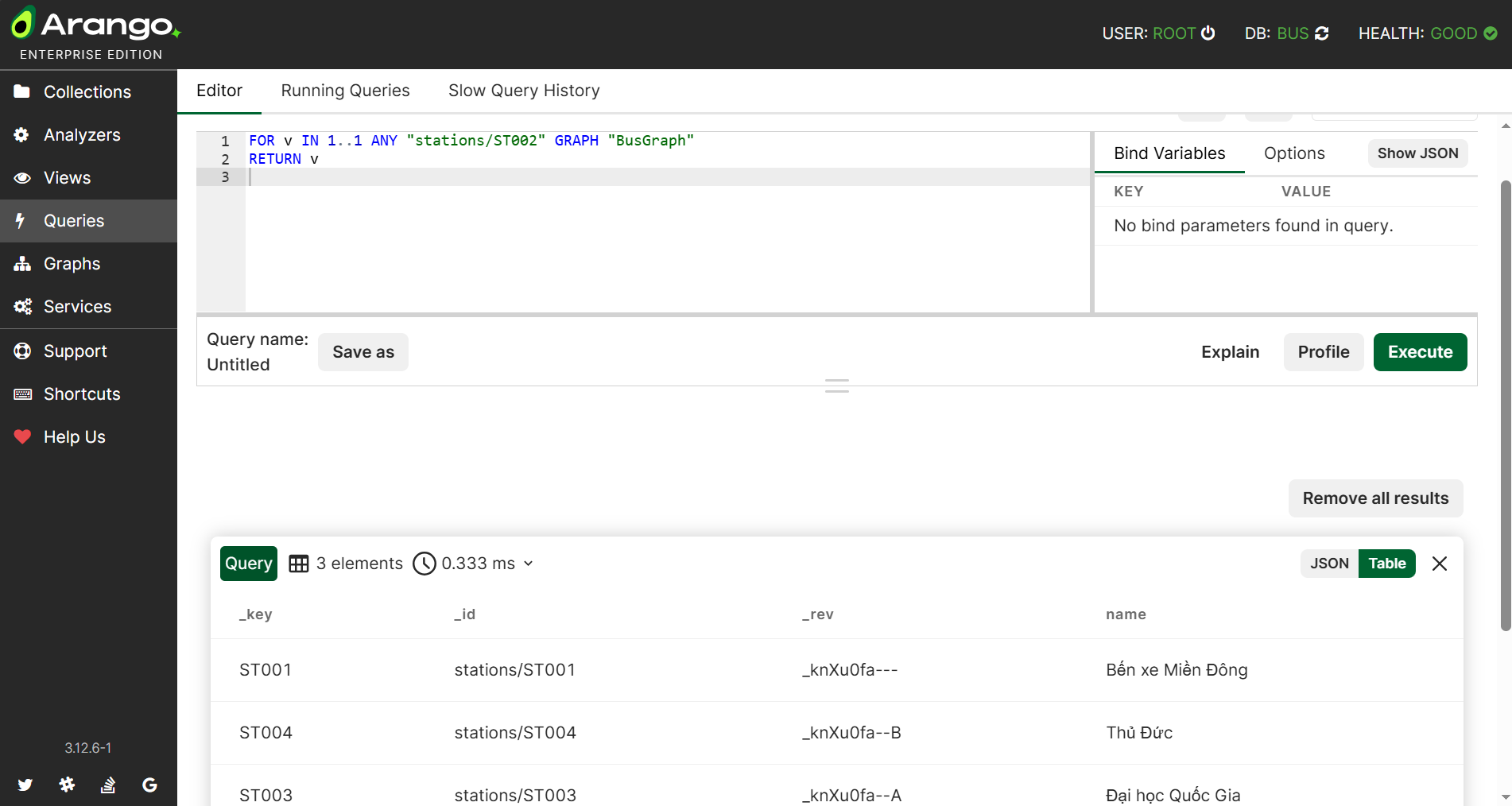
Truy vấn (tab QUERIES):

* Running Queries: Xem các truy vấn đang được thực thi.
* Slow Queries: Liệt kê các truy vấn chạy chậm, giúp xác định và tối ưu hóa các AQL kém hiệu quả. Đây là công cụ cực kỳ hữu ích cho DBA.

# CHƯƠNG 5. SO SÁNH ARANGODB WEB UI VỚI NEO4J BROWSER

# 5.1. Tiêu chí so sánh

Ví dụ : Tìm các trạm lân cận (1 hop neighbors)



Hình : Kết quả thực thi truy vấn của ArangoDB



Hình : Kết quả thực thi truy vấn của Neo4j

|  |  |  |  |
| --- | --- | --- | --- |
| Tiêu chí | ArangoDB Web UI | Neo4j Browser | Đánh giá |
| Thời gian thực thi (ví dụ so sánh ở trên) | 0.333 ms | ~2 ms | ArangoDB nhanh hơn ở dataset nhỏ do caching. |
| Mức độ dễ sử dụng | UI kỹ thuật, nhiều mục (Collections, Graphs, Queries…) | Tập trung vào graph, dễ hiểu với người mới | Neo4j thân thiện hơn |
| Dễ mở rộng (scaling) | Có clustering + sharding rất mạnh | Có clustering nhưng theo mô hình riêng | Tùy bài toán; ArangoDB linh hoạt multi-model. |
| Phù hợp cho loại dữ liệu | Kết hợp document + graph (vd: hệ thống xe buýt) | Quan hệ đồ thị thuần (social, shortest path) | Tùy yêu cầu ứng dụng. |

Bảng : So sánh tổng hợp các tiêu chí

# 5.2. Điểm mạnh của ArangoDB Web Interface

ArangoDB Web Interface (Web UI) là công cụ quản trị trực quan được tích hợp sẵn trong ArangoDB, hỗ trợ đầy đủ các tác vụ quản lý, truy vấn, trực quan hóa và phân tích dữ liệu. Giao diện này là một trong những ưu điểm lớn của ArangoDB, đặc biệt trong các hệ thống cần xử lý mô hình đa dữ liệu như tài liệu (document), quan hệ đồ thị (graph), và key–value. Những điểm mạnh nổi bật bao gồm:

* Quản lý đa mô hình (Multi-Model) trong một giao diện duy nhất
  + ArangoDB Web UI cho phép người dùng quản lý Document Collections, Edge Collections, và Graphs ngay trong cùng một công cụ, giúp việc thao tác giữa các mô hình trở nên liền mạch và thuận tiện.
  + Người dùng có thể xem, chỉnh sửa và thêm tài liệu JSON trực tiếp.
  + Có thể quan sát cấu trúc đồ thị, xem edge, node và các thuộc tính chi tiết.
  + Chuyển đổi linh hoạt giữa document view và graph view.

Điều này giúp ArangoDB Web UI mạnh hơn nhiều so với Neo4j Browser, vốn chỉ tập trung vào đồ thị và không hỗ trợ quản lý document.(5.2. 1\*)

* Hiển thị dữ liệu dạng JSON trực quan và đầy đủ
  + ArangoDB Web UI hỗ trợ xem dữ liệu dạng JSON hoàn chỉnh, có thể mở rộng, thu gọn, và chỉnh sửa trực tiếp.
  + Ưu điểm: Theo dõi dữ liệu raw rất rõ ràng, dễ dàng debug dữ liệu khi làm việc với API, hiệu quả khi quản lý dữ liệu phức tạp như thông tin trạm xe buýt (địa chỉ, vị trí GPS, tiện ích…).

Đây là lợi thế lớn so với Neo4j Browser, vốn chỉ hiển thị property theo dạng bảng và không hỗ trợ cấu trúc JSON lồng nhau.(5.2. 2\*)

* Công cụ truy vấn mạnh mẽ (AQL Query Editor)
  + ArangoDB Web UI cung cấp AQL Editor với các tính năng: Gợi ý cú pháp, chạy truy vấn trực tiếp, chế độ Explain & Profile để tối ưu query, xuất kết quả JSON hoặc bảng, chạy query trên cả document và graph bằng một ngôn ngữ duy nhất (AQL).

AQL Editor cho phép kiểm tra hiệu năng truy vấn, phù hợp khi cần đánh giá độ trễ trong bài toán mạng lưới xe buýt.(5.2.3\*)

* Trình quản lý Graph trực quan
  + Trong phần Graphs, ArangoDB Web UI cho phép: Tạo graph nhanh chóng, định nghĩa edge direction, edge collection, xem cấu trúc graph cơ bản, quan sát mối quan hệ giữa các trạm xe buýt.

Mặc dù không mạnh bằng Neo4j về visualization, ArangoDB vẫn cung cấp đủ công cụ để hiểu cấu trúc mạng lưới và hỗ trợ phát triển ứng dụng.(5.2.4\*)

* Hỗ trợ tốt cho hệ thống có dữ liệu phức tạp như mạng lưới xe buýt
  + ArangoDB Web UI hỗ trợ rất tốt: Lưu trữ thông tin trạm bằng Document, lưu quan hệ trạm–trạm bằng Edge, truy vấn đường đi, lân cận bằng Graph Traversal.
  + Kết hợp Document + Graph trong một query AQL → rất phù hợp.

# 5.3. Hạn chế của ArangoDB Web Interface

Mặc dù ArangoDB Web Interface (Web UI) cung cấp nhiều tính năng mạnh mẽ cho việc quản lý cơ sở dữ liệu đa mô hình, giao diện này vẫn tồn tại một số hạn chế nhất định, đặc biệt khi so sánh với các công cụ chuyên về đồ thị như Neo4j Browser. Một số hạn chế chính bao gồm:

* Khả năng trực quan hóa đồ thị còn hạn chế
  + ArangoDB Web UI chỉ cung cấp mức trực quan hóa cơ bản cho mô hình đồ thị:
* Không hỗ trợ kéo–thả node để điều chỉnh vị trí.
* Không có auto-layout động như Neo4j.
* Không nổi bật rõ degree, relationship type, hoặc path như Cypher visualization.
* Không hỗ trợ highlight path, expand neighbors, hoặc mở rộng depth 1-click.
* Cú pháp AQL cho đồ thị phức tạp khó viết và khó học hơn Cypher
  + AQL được thiết kế để thao tác cả document và graph, dẫn đến:
* Truy vấn đồ thị dài hơn.
* Tập trung vào tính biểu thức hơn tính trực quan.
* Không thân thiện với người mới như Cypher của Neo4j.
* Không chuyên dụng cho đồ thị như Neo4j Browser
  + ArangoDB Web UI phục vụ multi-model nên:
* Giao diện phân tán nhiều mục (Documents, Collections, Graphs, Views, Services…)
* Không tập trung hoàn toàn vào graph analysis
* Không có plugin graph analytic như Neo4j Bloom
* Không có khả năng mở rộng graph trực quan theo chiều sâu (Expand)

Trong Neo4j Browser:

* + Khi click node → tự động gợi ý mở rộng neighbors.
  + Hỗ trợ xem cấp độ liên kết theo mong muốn.
  + Hiển thị quan hệ 1-click.

Trong ArangoDB Web UI:

* + Visualization graph mang tính “snapshot” → không expandable.
  + Không hỗ trợ multi-hop exploration theo cách tương tác.
  + Điều này gây hạn chế trong việc khám phá cấu trúc mạng lưới lớn.

# 5.4. ArangoDB phù hợp với những bài toán

ArangoDB là cơ sở dữ liệu đa mô hình (multi-model), cho phép lưu trữ tài liệu (document), quan hệ đồ thị (graph) và dữ liệu key–value trong cùng một hệ thống. Nhờ sự linh hoạt này, ArangoDB phù hợp với nhiều bài toán thực tế đòi hỏi vừa quản lý dữ liệu chi tiết, vừa phân tích mối quan hệ phức tạp giữa các đối tượng. Một số nhóm bài toán tiêu biểu bao gồm:

1. Bài toán mô phỏng mạng lưới (Network Modeling)

*Ví dụ thực tế:*

* Mạng lưới giao thông.
* Mạng lưới vận chuyển hàng hóa.
* Hệ thống đường dây điện, đường ống.
* Mạng viễn thông.

*Lý do phù hợp:*

* Lưu thông tin nút (điểm / trạm / thiết bị) dưới dạng document, rất linh hoạt.
* Lưu kết nối giữa các điểm bằng edge collection.
* AQL hỗ trợ các truy vấn đồ thị như traverse, shortest path, neighbors,…  
   → Giúp phân tích kết nối, mô phỏng lộ trình, và đánh giá mạng lưới dễ dàng.

1. Bài toán quản lý quan hệ trong tổ chức (Organization & Relationship Management)

*Ví dụ thực tế:*

* Quan hệ nhân sự trong công ty.
* Hệ thống hồ sơ khách hàng (CRM).
* Quan hệ giữa phòng ban - dự án - nhân viên.
* Mạng lưới đối tác, nhà cung cấp.

*Lý do phù hợp:*

* Các thực thể có nhiều thông tin chi tiết → phù hợp với document.
* Các quan hệ giữa thực thể thường phức tạp → phù hợp với graph.

1. Bài toán IoT và hệ thống giám sát (IoT & Device Monitoring)

*Ví dụ thực tế:*

* Mạng lưới thiết bị cảm biến.
* Hệ thống giám sát nhà xưởng.
* Smart buildings, smart city.

*Lý do phù hợp:*

* Sensor gửi dữ liệu JSON linh hoạt → phù hợp document.
* Quan hệ giữa thiết bị – gateway – khu vực → graph.
* Truy vấn theo quan hệ thiết bị rất hiệu quả (neighbors, multi-hop)

# CHƯƠNG 6. PHÂN TÍCH BÀI TOÁN VÀ TRIỂN KHAI ỨNG DỤNG

6.1. Phân tích và thiết kế hệ thống quản lý xe bus

6.1.1. Quy trình hoạt động của hệ thống xe buýt

Hệ thống quản lý xe buýt là một bài toán phức tạp với nhiều thực thể và mối quan hệ đan xen. Để xây dựng một ứng dụng hiệu quả, nhóm đã phân tích quy trình hoạt động thực tế và xác định các vai trò người dùng chính:

Người Quản trị Hệ thống (Administrator):

* Quản lý tài khoản người dùng (tạo, sửa, xóa, phân quyền).
* Giám sát hoạt động tổng thể của hệ thống.
* Thực hiện các tác vụ bảo trì hệ thống như backup, restore dữ liệu.

Người Quản lý Vận hành (Manager):

* Quản lý cơ sở hạ tầng: Thêm, sửa, xóa thông tin các trạm xe buýt (stations).
* Quản lý tuyến: Thiết kế và quản lý các tuyến xe buýt (routes), bao gồm việc định nghĩa lộ trình (các trạm mà tuyến đi qua - serves), giá vé, tần suất, giờ hoạt động.
* Quản lý phương tiện: Quản lý thông tin đội xe (vehicles), theo dõi tình trạng, lịch bảo trì.

Phân công vận hành: Gán xe cụ thể vào các tuyến (operates\_on) và tạo lịch trình chi tiết (schedules) cho tài xế và xe.

Theo dõi và báo cáo: Xem các thống kê, báo cáo về hiệu suất hoạt động của các tuyến, mức độ sử dụng xe.

Người dùng cuối (End User - Public):

* Sử dụng các tính năng công khai của hệ thống (thông qua ứng dụng di động hoặc website).
* Tìm kiếm lộ trình: Tìm đường đi tối ưu giữa hai điểm dừng, xem các tuyến xe có thể đi, thời gian và chi phí dự kiến.
* Xem thông tin chi tiết về một tuyến xe hoặc một trạm xe.

6.1.2. Phân tích yêu cầu chức năng và phi chức năng

A. Yêu cầu chức năng

Hệ thống cần cung cấp các chức năng chính sau:

*F1: Quản lý Trạm (Stations)*: Cung cấp các thao tác CRUD (Tạo, Đọc, Cập nhật, Xóa) cho thông tin các trạm xe buýt, bao gồm tên, địa chỉ, tọa độ, loại trạm, trạng thái, và các tiện ích.

*F2: Quản lý Tuyến (Routes)*: Cung cấp các thao tác CRUD cho các tuyến xe buýt, bao gồm mã tuyến, tên tuyến, giá vé, giờ hoạt động, tần suất. Đặc biệt, cho phép quản lý danh sách các trạm mà tuyến đi qua (quản lý cạnh serves).

*F3: Quản lý Xe (Vehicles)*: Cung cấp các thao tác CRUD cho các phương tiện, bao gồm biển số, loại xe, sức chứa, tình trạng, lịch bảo trì và các tiện nghi trên xe. Cho phép gán xe vào một tuyến (quản lý cạnh operates\_on).

*F4: Quản lý Lịch trình (Schedules)*: Cho phép tạo và quản lý lịch trình chi tiết cho từng chuyến xe, bao gồm giờ xuất phát, tài xế, và ngày hoạt động.

*F5: Quản lý Người dùng và Phân quyền*:

Hệ thống xác thực người dùng qua tên đăng nhập và mật khẩu.

Hệ thống phân quyền truy cập chức năng dựa trên vai trò (Admin, Manager, User).

*F6: Tìm kiếm Lộ trình (Graph Traversal)*: Tính năng cốt lõi, cho phép tìm đường đi ngắn nhất (về khoảng cách hoặc thời gian) giữa hai trạm bất kỳ.

*F7: Thống kê và Báo cáo*: Cung cấp dashboard trực quan với các biểu đồ thống kê về:

* Tổng quan hệ thống (số trạm, tuyến, xe).
* Phân bố trạm theo khu vực.
* Hiệu suất các tuyến (tuyến đông khách, tuyến có độ phủ rộng).
* Tỷ lệ sử dụng phương tiện.

B. Yêu cầu phi chức năng

*NF1: Hiệu năng (Performance)*: Các truy vấn tìm đường, đặc biệt là tìm đường đi ngắn nhất, phải được thực thi nhanh chóng (dưới 2 giây) để đảm bảo trải nghiệm người dùng tốt.

*NF2: Bảo mật (Security)*:

* Mật khẩu người dùng phải được mã hóa (sử dụng bcrypt).
* Truy cập API phải được bảo vệ bằng JWT (JSON Web Tokens).
* Dữ liệu nhạy cảm không được trả về cho các vai trò không có quyền.

*NF3: Dễ sử dụng (Usability)*: Giao diện quản trị phải trực quan, dễ thao tác, có tính năng tìm kiếm, lọc và phân trang để xử lý lượng dữ liệu lớn.

*NF4: Tính sẵn sàng và Bảo trì (Availability & Maintainability)*:

Hệ thống có thể được triển khai dễ dàng thông qua Docker.

Cung cấp cơ chế sao lưu (backup) và phục hồi (restore) dữ liệu.

*NF5: Khả năng mở rộng (Scalability)*: Kiến trúc hệ thống và CSDL phải có khả năng mở rộng để xử lý số lượng trạm, tuyến và yêu cầu truy vấn tăng lên trong tương lai.

6.1.3. Thiết kế CSDL trên ArangoDB

Cấu trúc CSDL được thiết kế như sau:

A. Document Collections (Các đỉnh của đồ thị)

stations: Lưu trữ thông tin về các trạm xe buýt.

*{*

*"station\_id": "ST001",*

*"name": "Bến Xe Miền Đông",*

*"address": { "street": "292 Đinh Bộ Lĩnh", "ward": "Phường 26", "city": "TP.HCM" },*

*"location": { "latitude": 10.8142, "longitude": 106.7115 },*

*"type": "terminal", // terminal, intermediate, stop*

*"status": "active" // active, maintenance, inactive*

*}*

routes: Lưu trữ thông tin về các tuyến xe buýt.

*{*

*"route\_id": "R001",*

*"route\_code": "01",*

*"route\_name": "Bến Xe Miền Đông - Bến Xe Chợ Lớn",*

*"type": "normal", // normal, express, rapid*

*"fare": { "adult": 7000, "student": 3500 },*

*"status": "active"*

*}*

vehicles: Lưu trữ thông tin về các xe buýt.

*{*

*"vehicle\_id": "V001",*

*"license\_plate": "51B-10001",*

*"type": "bus\_40", // bus\_16, bus\_40, bus\_60*

*"capacity": 40,*

*"status": "active" // active, maintenance*

*}*

users: Lưu trữ thông tin tài khoản người dùng hệ thống.

*{*

*"username": "admin",*

*"password\_hash": "$2b$...",*

*"role": "admin", // admin, manager, user*

*"full\_name": "Administrator"*

*}*

schedules: Lưu trữ lịch trình chi tiết

*{*

*"route\_id": "R001",*

*"vehicle\_id": "V001",*

*"departure\_time": "05:00",*

*"shift": "morning",*

*"day\_of\_week": ["monday", "tuesday", ...]*

*}*

B. Edge Collections (Các cạnh của đồ thị)

connects: Biểu diễn kết nối vật lý (đoạn đường) giữa hai trạm xe buýt.

*\_from: stations/{id}*

*\_to: stations/{id}*

* Thuộc tính: distance (mét), duration (phút).

serves: Biểu diễn việc một tuyến xe (route) phục vụ một trạm (station).

*\_from: routes/{id}*

*\_to: stations/{id}*

* Thuộc tính: stop\_order (thứ tự dừng), arrival\_offset (thời gian đến tính từ trạm đầu).

operates\_on: Biểu diễn việc một xe (vehicle) được phân công hoạt động trên một tuyến (route).

*\_from: vehicles/{id}*

*\_to: routes/{id}*

* Thuộc tính: shift (ca làm việc), assignment\_date.

C. Graph Definition

Tất cả các collection trên được gom lại thành một đồ thị có tên là *bus\_network* với các định nghĩa cạnh như sau:

*connects: from [stations] to [stations]*

*serves: from [routes] to [stations]*

*operates\_on: from [vehicles] to [routes]*

Hai collection users và schedules được định nghĩa là Orphan Collections (các collection mồ côi), vì chúng không trực tiếp tham gia vào các mối quan hệ đồ thị chính nhưng vẫn thuộc về hệ thống.

6.1.4. Thiết kế kiến trúc ứng dụng

Hệ thống được thiết kế theo kiến trúc 3 lớp (3-Tier Architecture) hiện đại và phổ biến:

1. *Presentation Layer (Frontend)*:

* Công nghệ: Next.js (React Framework), TypeScript, Tailwind CSS, và thư viện component Shadcn/UI.
* Vai trò: Xây dựng giao diện người dùng (UI) cho trang quản trị, xử lý tương tác người dùng, gọi API đến Backend để lấy và gửi dữ liệu, và hiển thị dữ liệu một cách trực quan. Kiến trúc component-based giúp tái sử dụng và bảo trì code dễ dàng.

1. *Business Logic Layer (Backend):*

* Công nghệ: Python với framework Flask, pyArango (ArangoDB driver).
* Vai trò: Cung cấp các RESTful API để Frontend có thể tương tác với CSDL. Lớp này chịu trách nhiệm xử lý logic nghiệp vụ, xác thực và phân quyền người dùng (JWT), và thực thi các truy vấn AQL phức tạp trên ArangoDB.

1. *Data Layer (Database)*:

* Công nghệ: ArangoDB.
* Vai trò: Lưu trữ và quản lý toàn bộ dữ liệu của hệ thống dưới dạng multi-model (document và graph). Chịu trách nhiệm thực thi các truy vấn, đặc biệt là các truy vấn đồ thị hiệu năng cao.

6.2. Triển khai xây dựng hệ thống

6.2.1. Xây dựng và Chuẩn bị Dữ liệu

6.2.2. Import/Export và Backup/Restore Dữ liệu

6.2.3. Triển khai các Truy vấn trên ArangoDB Web Interface

6.2.4. Triển khai Backend và Kết nối CSDL

6.2.5. Triển khai Frontend

# CHƯƠNG 7. KẾT QUẢ TRIỂN KHAI VÀ MINH HỌA ỨNG DỤNG

# 7.1. Giới thiệu giao diện và các chức năng chính

# 7.2. Demo chức năng quản trị người dùng

# 7.3. Demo chức năng thao tác CSDL

# 7.4. Demo chức năng quản lý theo mô tả bài toán

# 7.5. Demo các chức năng hỗ trợ khác

# CHƯƠNG 8. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

# 8.1. Đánh giá so với mục tiêu ban đầu

# 8.2. Những hạn chế của hệ thống 8.3. Đề xuất cải tiến và mở rộng 8.4. Kết luận chung

# CHƯƠNG 6: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận chung

## Hạn chế của đề tài

## Hướng phát triển tương lai

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | N. Antonio, A. de Almeida and L. Nunes, "Hotel booking demand datasets," ScienceDirect, 11 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352340918315191. [Accessed 02 11 2025]. |
| [2] | ResearchGate, "Hotel booking demand datasets (open access)," ResearchGate, 2019. [Online]. Available: https://www.researchgate.net/publication/329286343\_Hotel\_booking\_demand\_datasets. [Accessed 02 11 2025]. |
| [3] | arXiv.org, "Hotel Booking Cancellation Prediction Using Applied Bayesian Models," arXiv, 10 2024. [Online]. Available: https://arxiv.org/html/2410.16406v2. [Accessed 02 11 2025]. |
| [4] | A. Ramesh, "Exploratory Data Analysis of the Hotel Booking Demand with Python," Medium, 7 2022. [Online]. Available: https://medium.com/analytics-vidhya/exploratory-data-analysis-of-the-hotel-booking-demand-with-python-200925230106. [Accessed 02 11 2025]. |
| [5] | N. D. S. Academy, "Hotel Bookings Data Analysis Project," NYC Data Science, 3 2021. [Online]. Available: https://nycdatascience.com/blog/r/hotel-bookings-data-analysis. [Accessed 02 11 2025]. |
| [6] | IJARSCT, "Real Time Hotel Booking Demand Optimization," International Journal of Advanced Research in Science, Communication and Technology, 6 2023. [Online]. Available: https://ijarsct.co.in/Paper17978.pdf. [Accessed 02 11 2025]. |
| [7] | D. S. Journal, "An Automated Machine Learning Based Decision Support System to predict hotel booking cancellations," CODATA, 2019. [Online]. Available: https://datascience.codata.org/articles/10.5334/dsj-2019-032. [Accessed 02 11 2025]. |
| [8] | A. S. Foundation, "Apache Spark – Unified analytics engine for large-scale data processing," Apache Software Foundation, 12 2024. [Online]. Available: https://spark.apache.org/. [Accessed 02 11 2025]. |
| [9] | A. Bhardwaj, "A Comprehensive Guide to K-Means Clustering," Analytics Vidhya, 8 2019. [Online]. Available: https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/. [Accessed 02 11 2025]. |