

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN



SEMINAR CHƯƠNG 5

Môn học : Dữ liệu lớn

Học kỳ II (2021-2022)

Chủ đề:

APACHE PINOT

Sinh viên thực hiện:

- | | |
|---------------------|----------------|
| 1. Trần Nhật Tân | MSSV: 19522177 |
| 2. Bùi Ngọc Thành | MSSV: 19522220 |
| 3. Huỳnh Quốc Khánh | MSSV: 19521677 |
| 4. Lê Thế Tiệm | MSSV: 19522330 |

GVHD: TS. Nguyễn Hồ Duy Tri

Lớp: IS405.N11.HTCL

Thành phố Hồ Chí Minh, tháng 11 năm 2022

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN



SEMINAR CHƯƠNG 5

Môn học : Dữ liệu lớn

Học kỳ II (2021-2022)

Chủ đề:

APACHE PINOT

Sinh viên thực hiện:

- | | |
|---------------------|----------------|
| 1. Trần Nhật Tân | MSSV: 19522177 |
| 2. Bùi Ngọc Thành | MSSV: 19522220 |
| 3. Huỳnh Quốc Khánh | MSSV: 19521677 |
| 4. Lê Thế Tiệm | MSSV: 19522330 |

GVHD: TS. Nguyễn Hồ Duy Tri

Lớp: IS405.N11.HTCL

Thành phố Hồ Chí Minh, tháng 11 năm 2022

MỤC LỤC

LỜI CẢM ƠN	4
NHẬN XÉT CỦA GIẢNG VIÊN	5
CHƯƠNG I: APACHE PINOT	1
1.1 TỔNG QUAN ĐỀ TÀI.....	1
1.2 KHÁI NIỆM.....	2
1.2.1 Lịch sử ra đời.....	2
1.3 KIẾN TRÚC PINOT	3
1.3.1 Mô hình lưu trữ.....	3
1.3.2 Các thành phần của Pinot.....	4
1.4 Luồng hoạt động của Pinot	5
1.4.1 Batch Data Flow	6
1.4.2 Real-time Data Flow	7
1.4.3 Truy vấn trong Pinot.....	8
1.5 Tính năng của Apache Pinot	8
1.5.1 Ưu-Nhược điểm của Pinot	9
1.5.2 Tại sao nên sử dụng Apache Pinot?	9
CHƯƠNG 2: APACHE SPARK	10
2.1 Khái niệm.....	10
2.2 CÁC THÀNH PHẦN CỦA SPARK	12
2.3 KIẾN TRÚC CỦA SPARK.....	12
2.4 ƯU ĐIỂM – NHƯỢC ĐIỂM.....	14
2.5 SO SÁNH APACHE PINOT VÀ APACHE SPARK	15
CHƯƠNG 3: CÀI ĐẶT	18

3.1 Cách thiết lập cụm Pinot.....	18
3.2 điều kiện tiên quyết.....	18
3.3 Khởi động Zookeeper	18
3.4 Setup Zooinspector	19
3.5 Khởi động Controller.....	20
3.6 Khởi động Broker.....	22
3.7 Khởi động Server.....	23
CHƯƠNG 4: MINH HỌA VẬN HÀNH XỬ LÝ DỮ LIỆU LỚN.....	26
THAM KHẢO	32
Hết.	33

LỜI CẢM ƠN

Trên thực tế không có sự thành công nào mà không gắn liền với những sự hỗ trợ, giúp đỡ dù ít hay nhiều, dù trực tiếp hay gián tiếp của người khác. Với lòng biết ơn sâu sắc nhất, đầu tiên nhóm chúng em xin gửi lời cảm ơn chân thành đến tập thể quý Thầy Cô Trường Đại học Công nghệ thông tin – Đại học Quốc gia TP.HCM và quý Thầy Cô khoa Hệ thống thông tin đã giúp cho nhóm có những kiến thức cơ bản làm nền tảng để thực hiện đề tài này.

Đặc biệt nhóm chúng em xin gửi lời cảm ơn chân thành tới thầy Nguyễn Hồ Duy Tri – giảng viên lý thuyết và thực hành môn Dữ liệu lớn đã tận tình giúp đỡ, trực tiếp chỉ bảo, hướng dẫn nhóm trong suốt quá trình làm đề tài. Nhờ đó, chúng em đã tiếp thu được nhiều kiến thức bổ ích trong việc vận dụng cũng như kỹ năng làm đề tài. Nếu không có những lời hướng dẫn, dạy bảo của Thầy thì nhóm chúng em nghĩ đề tài này của nhóm rất khó có thể hoàn thiện được. Một lần nữa, em xin chân thành cảm ơn Thầy. Ngoài ra, để đề tài được hoàn thành thì không thể nào cảm ơn những người đã làm ra đó, cảm ơn các bạn các thành viên trong nhóm đã chăm chỉ và chịu khó hoàn thành nhiệm vụ đúng tiến độ.

Cuối cùng, cảm ơn tất cả các thành viên trong nhóm đã làm việc hết công suất để hoàn thành tốt đề tài của mình. Xin chân thành cảm ơn!

[illegible]

CHƯƠNG I: APACHE PINOT

1.1 TỔNG QUAN ĐỀ TÀI

Hiện nay, Big data đã trở nên quá phổ biến, nó cũng là những vấn đề mà từ những công ty lớn cho đến nhiều start-up sẽ phải đối mặt. Từ đó cũng đã kéo theo sự xuất hiện, phát triển của nhiều công cụ và framework để giúp xử lý và lưu trữ dữ liệu. Có một vài framework phổ biến như Hadoop, Spark, Hive, Pig, Storm và Zookeeper. Trong bài này chúng ta sẽ thảo luận về Apache Pinot.

Có 2 loại xử lý Big Data:

- Batch Processing
- Real-time Processing

Xử lý trên data được thu thập theo thời gian được gọi là Batch Processing. Ví dụ, manager của ngân hàng muốn xử lý data trong một tháng vừa qua để biết số lượng tấm séc (chi phiếu) bị hủy trong tháng vừa qua.

Xử lý dựa trên dữ liệu tức thời trả về kết quả tức thời được gọi là Real-time Processing. Ví dụ, quản lý ngân hàng nhận được cảnh báo gian lận ngay sau khi giao dịch gian lận xảy ra.

1.2 KHÁI NIỆM



Apache Pinot là kho dữ liệu phân tán, mã nguồn mở được viết bằng ngôn ngữ Java. Pinot được xây dựng để phục vụ phân tích dữ liệu Real-time với độ trễ thấp.

Cái tên Pinot được lấy ý tưởng từ những cây nho Pinot được ép thành chất lỏng để sản xuất nhiều loại rượu vang khác nhau. Những nhà sáng lập cơ sở dữ liệu đã chọn cái tên này như một phép ẩn dụ để phân tích một lượng lớn dữ liệu từ nhiều định dạng tệp hoặc từ các nguồn dữ liệu trực tuyến khác nhau.

1.2.1 Lịch sử ra đời

Ban đầu, dự án Pinot được bắt đầu xây dựng tại LinkedIn vào năm 2013 nhằm cung cấp nhiều loại sản phẩm hướng tới người dùng và doanh nghiệp. Sản phẩm phân tích đầu tiên ở LinkedIn đã sử dụng Pinot là bản thiết kế lại tính năng của trang mạng xã hội, cho phép các thành viên kiểm tra ai đã xem hồ sơ của họ trong thời gian thực.

Dự án Pinot được cấp giấy phép bởi Apache 2.0 nên đã trở thành mã nguồn mở và được LinkedIn tài trợ cho quỹ phần mềm Apache vào tháng 6 năm 2019.

Được phát triển bởi các kỹ sư tại LinkedIn. Nó được tạo ra để mở rộng hiệu suất truy vấn dựa trên số lượng Node nhất định trong một Cluster. Có nghĩa là khi thêm nhiều Node hơn, hiệu suất truy vấn sẽ luôn được cải thiện. Pinot được sử dụng bởi các công ty lớn như Uber, Microsoft, Factual.

1.3 KIẾN TRÚC PINOT

1.3.1 Mô hình lưu trữ

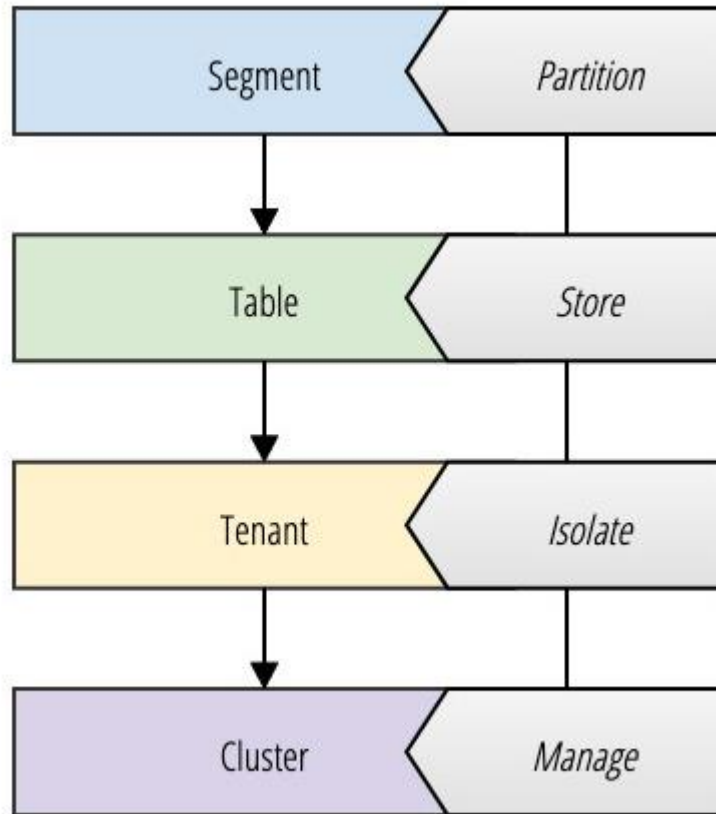


Table: đóng vai trò như là một cơ sở dữ liệu.

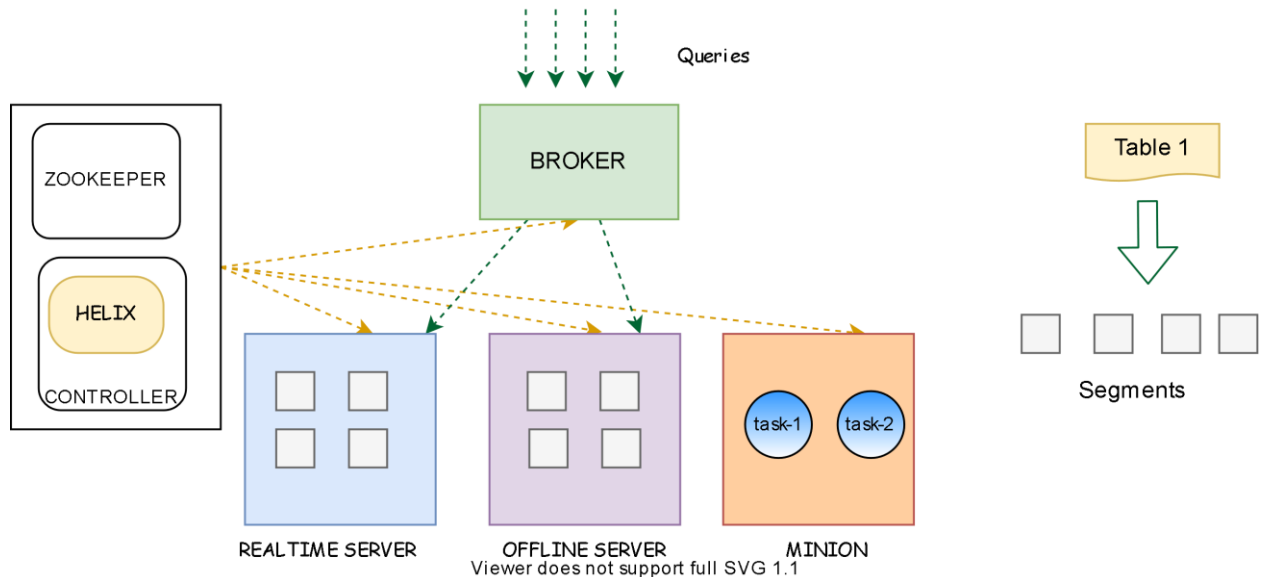
Segment: vì Pinot có kiến trúc hệ thống phân tán theo chiều ngang, nên các kỹ sư đã hi vọng rằng kích thước của bảng sẽ tăng dần theo thời gian. Để đạt được điều này, tất cả dữ liệu cần được phân phối trên nhiều Node. Và Pinot đã chia thành công dữ liệu thành các phần nhỏ hơn gọi là Segment

Tenant: một bảng sẽ được liên kết với một Tenant. Điều này cho phép tất cả các bảng thuộc không gian logic được nhóm lại dưới tên của Tenant và sẽ tách biệt với các Tenant khác.

Cluster: là một nhóm Tenant hay hay còn là một nhóm tập hợp các nút. Thông thường thì chỉ có một cụm cho mỗi môi trường hoặc trung tâm dữ liệu. Tại LinkedIn, một cụm lớn nhất bao gồm hơn 1000 nút được phân bổ trên trung tâm dữ

liệu. Có thể thêm nút vào cụm bằng cách tăng hiệu suất và tính khả dụng của truy vấn.

1.3.2 Các thành phần của Pinot



Các thành phần chính của Pinot gồm: Controller, Broker, Server, Minion. Pinot sử dụng Apache Helix trong kiến trúc của nó như một tác nhân bên trong các thành phần khác nhau và sử dụng Zookeeper để duy trì và điều phối trạng thái của các cụm.

- **Helix và Zookeeper:** tất cả các Server và Broker được điều khiển bởi Helix. Helix là một framework quản lý các phân vùng và bản sao trong hệ thống phân tán. Truy vấn sẽ được tối ưu hóa khi Helix cập nhật cấu hình định tuyến giữa các Node dựa trên nơi lưu trữ dữ liệu. Helix sử dụng Zookeeper để duy trì trạng thái của cụm, mỗi thành phần của cụm Pinot lấy địa chỉ Zookeeper để làm thao số khởi động.
- **Controller:** là thành phần cốt lõi thúc đẩy tính nhất quán và định tuyến trong một cụm Pinot, có khả năng hiển thị trạng thái của các thành phần khác trong cụm. Controller thường là thành phần đầu tiên được khởi động sau Zookeeper vì vai trò của Controller như là sự tham gia của Helix để điều khiển trạng thái

của các thành phần khác. Để khởi động Controller, cần có địa chỉ Zookeeper và tên Cluster. Khi đó, Controller sẽ tự động tạo ra một Cluster thông qua Helix nếu như nó chưa tồn tại.

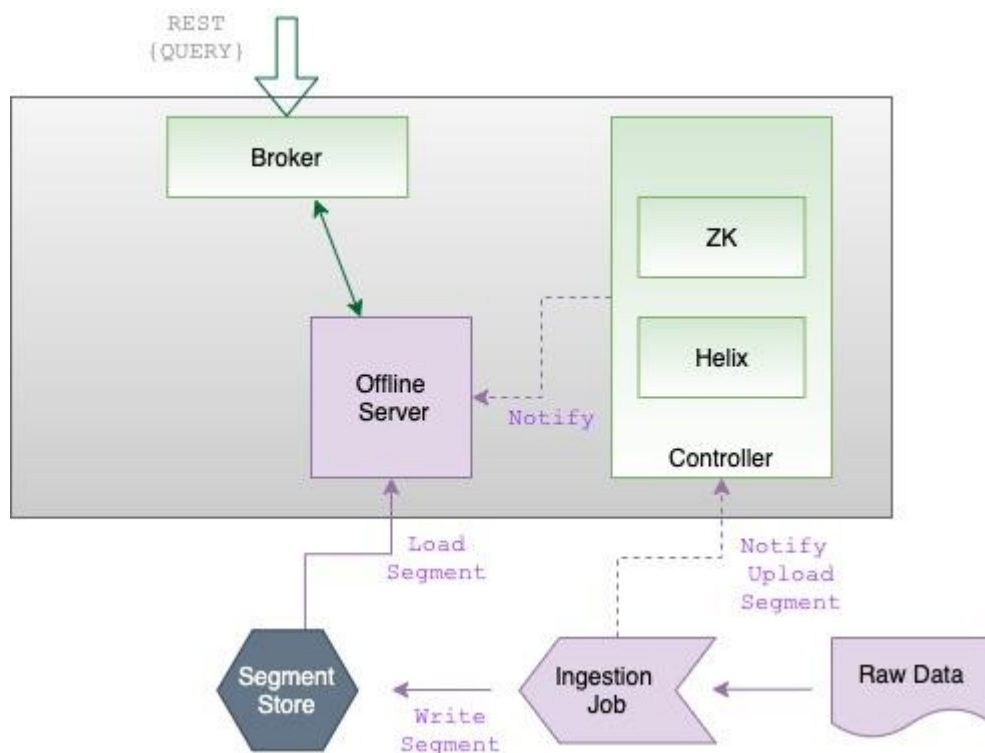
- **Broker:** trách nhiệm của Broker là định tuyến truy vấn nhất định đến một phiên bản máy chủ thích hợp. Broker sẽ thu nhập và kết hợp các phản hồi từ máy chủ và đưa ra kết quả cuối cùng, sau đó gửi về cho khách hàng tùy thuộc vào yêu cầu khách hàng. Để Broker hoạt động, cần ba thứ chính: tên Cluster, địa chỉ Zookeeper, tên Broker.
- **Servers:** lưu trữ các phân đoạn và thực hiện hầu hết các công việc nặng nhọc trong quá trình xử lý truy vấn. Mặc dù kiến trúc cho thấy có 2 loại máy chủ là Offline Server và Real-time server, nhưng chính máy chủ đó sẽ không biết nó sẽ là Offline hay Real-time.
- **Minion:** là thành phần tùy chọn và không bắt buộc cần phải có để bắt đầu Pinot. Minion được sử dụng để xóa dữ liệu từ Cluster Pinot.

1.4 Luồng hoạt động của Pinot

Logical Table trong Apache Pinot được hình thành dưới dạng Offline hoặc Real-time. Lý do có hai loại Table bởi vì mỗi loại có mô hình trạng thái khác nhau. Trong khi Real-time Tables sử dụng các dữ liệu luồng như Apache Kafka, Amazon Kinesis để lưu trữ trong khoảng thời gian ngắn và thể hiện tỉ lệ truy vấn dựa theo tốc độ nhập, thì Offline Tables sử dụng các máy ảo có dung lượng lưu trữ lớn như Amazon S3 và HDFS để lưu trữ dữ liệu lâu hơn.

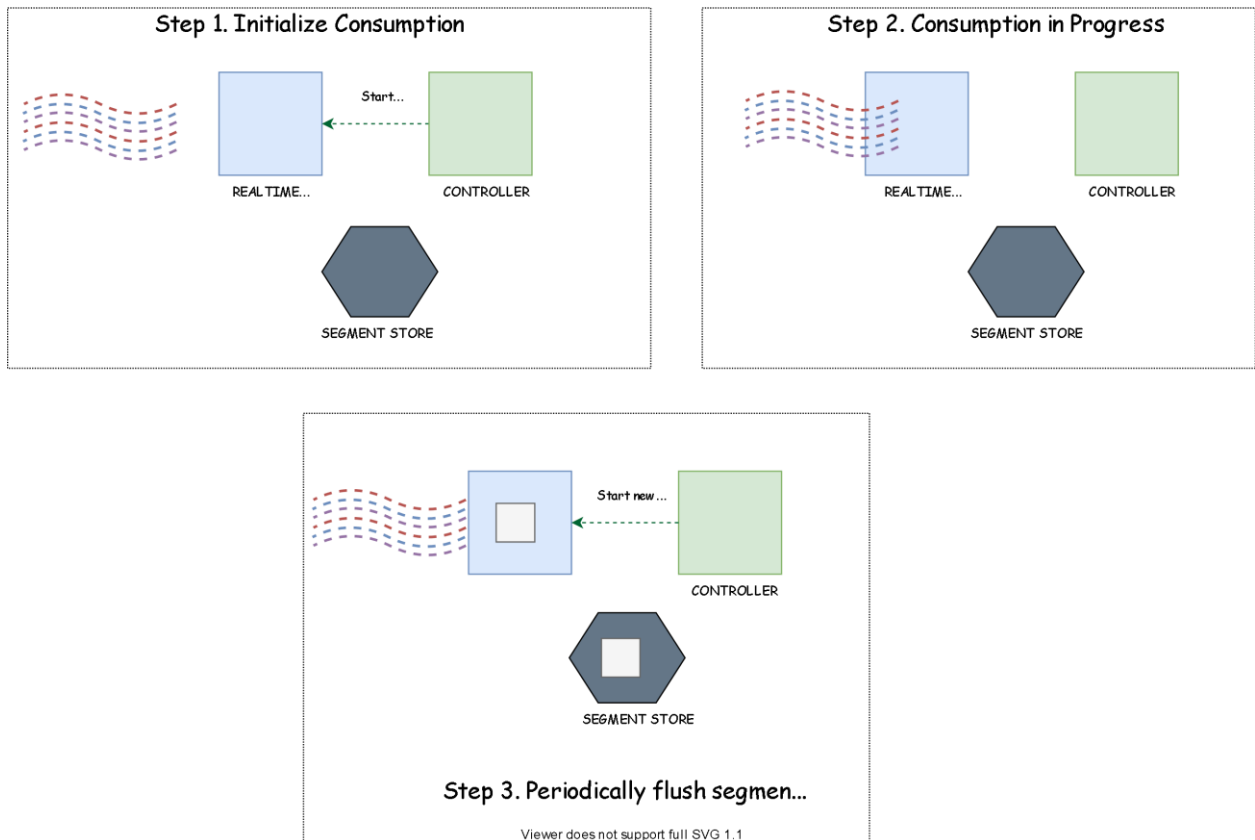
1.4.1 Batch Data Flow

Trong chế độ Batch, Pinot nhập dữ liệu thông qua việc chuyển dữ liệu thô thành các Segments (đại diện cho một phần dữ liệu của Table). Sau đó, nó sẽ lưu trữ các Segments vào trong kho (Cluster's segment store) và thông báo cho Controller. Sau khi xử lý thông tin, Helix sẽ cập nhật trạng thái của cấu hình trong Zookeeper. Sau đó, Offline Server sẽ được thông báo về các Segments mới và các Segments này sẽ được tải xuống từ kho, cuối cùng được thêm vào danh sách truy vấn các Segments.



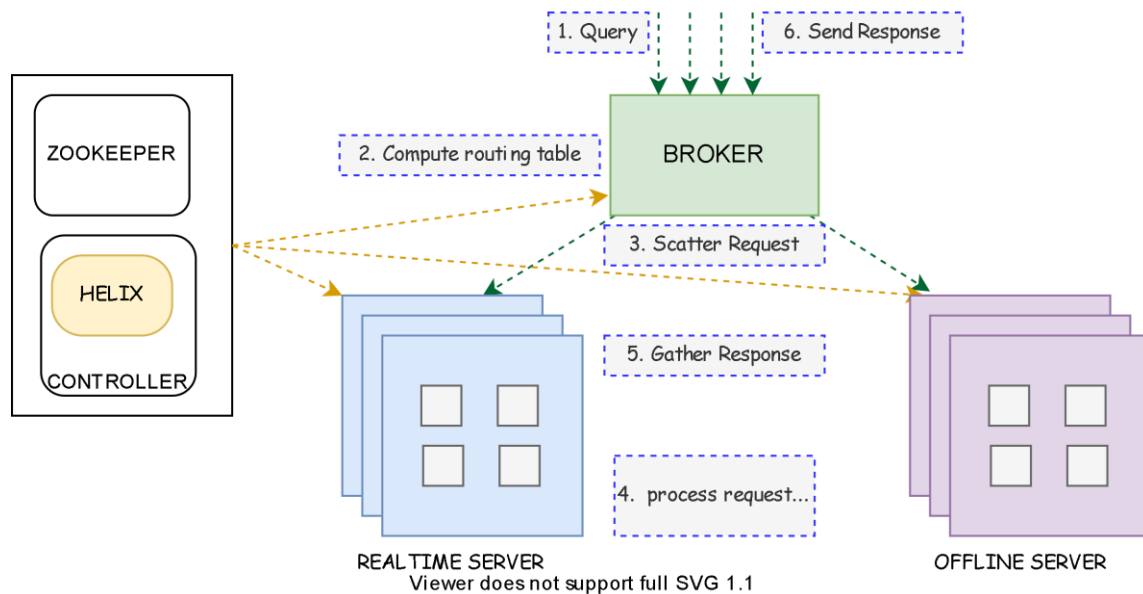
1.4.2 Real-time Data Flow

Đối với luồng Real-time, Controller sẽ tạo ra chỗ trống mới trong Zookeeper dùng để tiêu thụ các Segment. Sau đó, Helix sẽ thông báo cho máy chủ về các Segment mới, đổi lại sẽ phải bắt đầu nhập dữ liệu từ các nguồn trực tuyến. Broker sẽ xử lý các thay đổi, phát hiện các Segment mới và thêm vào danh sách các truy



vấn Segment.

1.4.3 Truy vấn trong Pinot



Broker sẽ nhận các truy vấn và kiểm tra các yêu cầu đối với Segment tới bảng định tuyến máy chủ (Server routing table) trong khi phân chia các yêu cầu giữa Offline và Real-time Servers. Các yêu cầu sau đó sẽ được lọc và tổng hợp bởi hai Tables và trả về kết quả cho Broker. Cuối cùng Broker sẽ trả lại thông tin thu thập được cho khách hàng.

1.5 Tính năng của Apache Pinot

Có thể nói được các chỉ mục: Apache Pinot hỗ trợ các công cụ hỗ trợ nói các chỉ mục như Forward Index, Star-free Index, Inverted Index,...

Tối ưu hóa truy vấn tùy thuộc Segment và siêu dữ liệu truy vấn.

Lưu trữ một lượng lớn bảng dữ liệu theo cột thay vì theo hàng và cung cấp lược đồ nén ví dụ như Fixed Bit Length, Run Length,...

Cho phép người dùng truy vấn các trường dưới dạng giá trị được phân tách bằng dấu phẩy vì nó hỗ trợ việc có thể có nhiều giá trị trong một trường dữ liệu.

Ngôn ngữ truy vấn của Pinot cho phép người dùng tổng hợp, lựa chọn, sắp xếp, lọc, nhóm các truy vấn trên dữ liệu.

Có thể mở rộng và có khả năng chịu lỗi.

1.5.1 Ưu-Nhược điểm của Pinot

Ưu điểm:

- Tính khả dụng cao: Pinot được xây dựng để phục vụ các truy vấn phân tích có độ trễ thấp. Hệ thống sẽ tiếp tục làm việc cho dù có xảy ra bất kỳ lỗi nào.
- Khả năng mở rộng theo chiều ngang: bằng cách thêm các nút mới mỗi khi khối lượng công việc thay đổi.
- Độ trễ và Lưu trữ: cung cấp độ trễ thấp cho dù đang ở thông lượng cao. Các tính năng như Star-free indexing đã được phát triển để đạt được điều này.
- Dữ liệu bất biến: tất cả dữ liệu được lưu trữ là bất biến.
- Thay đổi cấu hình động: các thao tác như thêm bảng mới, mở rộng cụm, nhập dữ liệu, sửa đổi cấu hình Index phải được thực hiện mà không ảnh hưởng đến tính khả dụng hoặc hiệu suất của truy vấn.

Nhược điểm:

- Chỉ hỗ trợ phương thức JDBC.
- Chức năng còn hạn chế và trong giai đoạn phát triển.

1.5.2 Tại sao nên sử dụng Apache Pinot?

Sử dụng cơ sở dữ liệu cột – Columnar Database: về cốt lõi, Apache pinot là một cơ sở dữ liệu hướng cột có thể được truy vấn bằng cách sử dụng tập hợp con của SQL.

- Tính năng lập chỉ mục cho phép phát triển nhanh chóng các loại chỉ mục mới, liên tục đẩy nhanh hiệu suất.
- Trong khi OLAP thực thi truy vấn chậm và lưu trữ không hiệu quả, Pinot đã thực hiện hầu hết các truy vấn trong vòng chưa đầy 1 giây,

đồng thời cung cấp yêu cầu về lưu trữ đã được cải thiện so với các cơ sở dữ liệu khác.

Nhập dữ liệu trong thời gian thực

- Điều này đã cho phép người dùng nhập hàng triệu dữ liệu mỗi giây đồng thời thực hiện chúng ngay lập tức.
- Sự kết hợp giữa nhập và tính khả dụng của truy vấn đã mở ra trải nghiệm mới cho người dùng mà trước đây chưa thể đạt được trên quy mô lớn mà không có bộ nhớ đệm

Khả năng chịu lỗi và khả năng mở rộng: bởi vì là một hệ thống phân tán, nên Pitno có thể mở rộng ra hàng ngàn nút. Các nút này hoạt động như một hệ thống duy nhất, phản hồi các truy vấn đồng thời, và dữ liệu sẽ được sao chép giữa các node để chịu lỗi.

Đáng tin cậy: được sử dụng phổ biến bởi các ứng dụng, thương hiệu nổi tiếng như Weibo, Microsoft Team, 7 eleven, Uber...

CHƯƠNG 2: APACHE SPARK

2.1 Khái niệm



Là một framework mã nguồn mở tính toán cụm, được phát triển sơ khởi vào năm 2009 bởi AMPLab, sau đó được trao cho Apache Software Foundation vào năm 2013 và được phát triển cho đến nay.

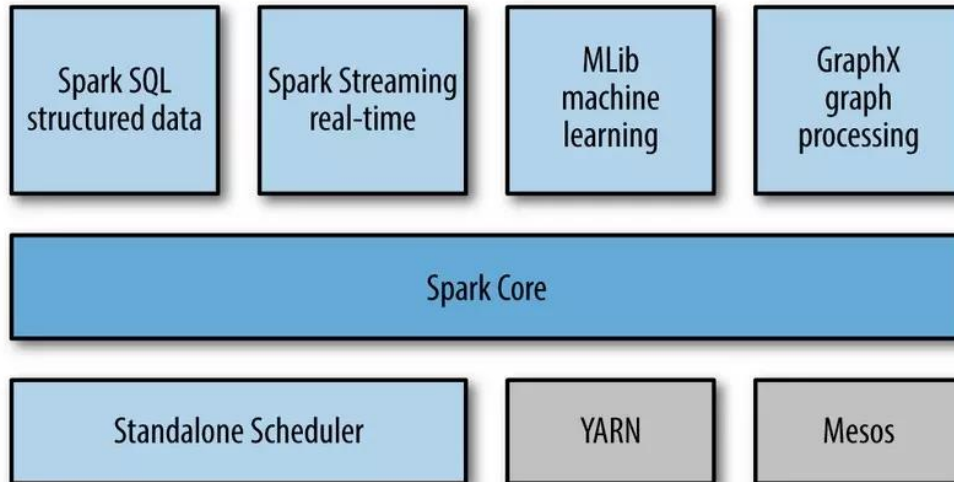
Spark cho phép người dùng xây dựng các mô hình dự đoán nhanh chóng với việc tính toán được thực hiện trên một nhóm máy tính. Nó có thể tính toán cùng lúc trên toàn bộ tập dữ liệu mà không cần phải trích xuất mẫu thử nghiệm.

Tốc độ xử lý của Spark có được do việc tính toán được thực hiện cùng lúc trên nhiều máy khác nhau. Đồng thời việc tính toán được thực hiện ở bộ nhớ trong (in-memories) hay thực hiện hoàn toàn trên RAM.

Spark không có hệ thống File của riêng mình nên nó sử dụng hệ thống file khác như HDFS, Cassandra, S3,... Bên cạnh đó Spark còn hỗ trợ nhiều kiểu định dạng file khác nhau như file text, csv,json,... và đồng thời không phụ thuộc vào bất kì hệ thống file nào.

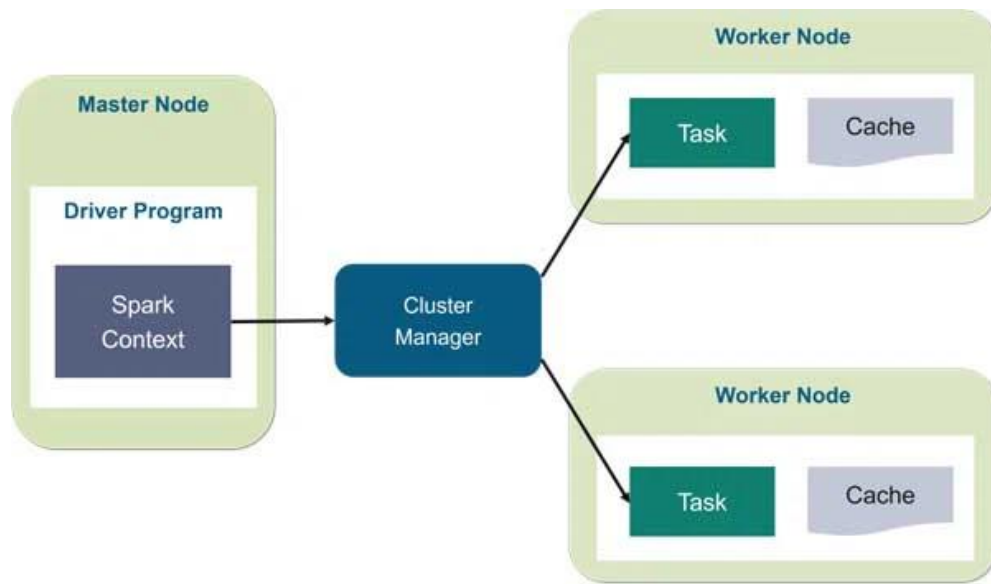
2.2 CÁC THÀNH PHẦN CỦA SPARK

Gồm 5 thành phần chính: Spark Core, Spark Streaming, Spark SQL, Mlib và GraphX.



- Spark Core: cung cấp những chức năng cơ bản nhất của Spark như lập lịch cho các tác vụ, quản lý bộ nhớ, tương tác với hệ thống lưu trữ,... Đặc biệt, Spark Core còn cung cấp API để định nghĩa Resilient Distributed Dataset, là tập hợp các item được phân tán trên các Node của cụm và có thể xử lý song song.
- Spark SQL: cho phép truy vấn dữ liệu cấu trúc qua các câu lệnh SQL. Spark SQL có thể thao tác với nhiều nguồn dữ liệu như Hive tables, Parquet, và JSON.
- Spark Streaming: cung cấp API để dễ dàng xử lý dữ liệu Stream.
- Mlib(Machine Learning Library): là nền tảng học máy phân tán trên Spark. Cung cấp nhiều thuật toán học máy như Classification, Regression, Clustering,...
- GraphX: thư viện để xử lý đồ thị.

2.3 KIẾN TRÚC CỦA SPARK



Về cơ bản, Apache Spark bao gồm hai thành phần chính: trình điều khiển (driver) và trình thực thi (executors). Trình điều khiển dùng để chuyển đổi mã của người dùng thành nhiều tác vụ (tasks) có thể được phân phối trên các nút xử lý (worker nodes).

Trình thực thi chạy trên các nút xử lý và thực hiện các nhiệm vụ được giao cho chúng. Spark cũng có thể chạy ở chế độ cụm độc lập chỉ yêu cầu khung Apache Spark và JVM trên mỗi máy trong cụm. Tuy nhiên, sử dụng các công cụ quản lý cụm như trung gian giữa hai thành phần giúp tận dụng tài nguyên tốt hơn và cho phép phân bổ theo yêu cầu. Trong doanh nghiệp, Apache Spark có thể chạy trên Apache Mesos, Kubernetes và Docker Swarm.

2.4 ƯU ĐIỂM – NHƯỢC ĐIỂM

Ưu điểm:

- **Đơn giản và dễ sử dụng:** Apache Spark được phát triển để giúp người dùng tiếp cận dễ dàng hơn với công nghệ tính toán song song. Người dùng chỉ cần trang bị những kiến thức về database, lập trình Python hoặc Scala là đã có thể sử dụng được.
- **Khả năng, tốc độ phân tích thời gian thực ấn tượng:** Spark có thể xử lý hàng loạt dữ liệu thời gian thực. Tốc độ xử lý có thể lên đến hàng triệu sự kiện mỗi giây. Việc nhận dữ liệu từ nguồn và xử lý dữ liệu diễn ra gần như đồng thời. Bên cạnh đó, Spark còn hữu ích cho việc phát hiện gian lận khi thực hiện các giao dịch ngân hàng.
- **Hỗ trợ bởi những thư viện cấp cao:** Spark nhận được sự hỗ trợ của các thư viện cấp cao như truyền dữ liệu trực tuyến, truy vấn SQL, học máy và xử lý đồ thị. Không chỉ giúp tăng hiệu suất cho nhà phát triển, những thư viện tiêu chuẩn này còn đảm bảo sự kết nối liền mạch cho các quy trình làm việc phức tạp.
- **Khả năng tương thích cao và hỗ trợ nhiều ngôn ngữ lập trình:** Apache Spark có thể tương thích với tất cả các định dạng tệp và nguồn dữ liệu được hỗ trợ bởi cụm Hadoop. Ngôn ngữ lập trình sử dụng được là Scala, Java, Python và R.

Nhược điểm:

- Spark không có hệ thống File riêng, do đó nó phải phụ thuộc vào một số nền tảng khác như Hadoop hoặc một nền tảng đám mây như S3, Google Cloud Storage.
- Đòi hỏi rất nhiều ram để chạy bộ nhớ, do đó tốn rất nhiều chi phí.

2.5 SO SÁNH APACHE PINOT VÀ APACHE SPARK

Cả hai đều là các công cụ mã nguồn mở được phát triển trong khuôn khổ tổ chức của Apache Foundation. Mỗi giải pháp có thể được sử dụng như một giải pháp độc lập, nhưng chúng thường được tích hợp vào môi trường dữ liệu lớn, ví dụ như Hadoop (YARN, HDFS và thường là Apache Kafka). Cả Spark và Flink đều cung cấp một loạt các tính năng và API tương tự, ví dụ hỗ trợ cho các truy vấn SQL, cũng như xử lý hàng loạt và luồng.

	Apache Pinot	Apache Spark
Ngôn ngữ triển khai	Java	Scala
Hệ điều hành	Tất cả hệ điều hành với Java 11 hoặc hơn	Linux, OS X, Window
Xử lý luồng	Streaming, Batch	Spark Streaming
Thông lượng	Cao	Cao
API và các phương thức truy cập khác	JDBC	JDBC, ODBC
Hỗ trợ ngôn ngữ lập trình	Go, Java, Python	Java, Python, Scala, R
Truy vấn SQL	PQL tương tự như SQL	Spark SQL
Kết nối	Kafka, Kubernetes, GCP, Azure, AWS3, ...	Kafka, Amazon S3, ElasticSearch, Twitter, ...
Khả năng chịu lỗi	Nhiều controller và một trong số chúng sẽ hoạt động như một	Đảm bảo chính xác một lần

	đơn vị chỉ huy. Nếu đơn vị chỉ huy gặp sự cố hoặc chết, một đơn vị chỉ huy khác sẽ tự động được chọn	
Độ trễ	Độ trễ thấp trong khoảng mili giây	Độ trễ thấp trong khoảng mili giây

So sánh Apache Pinot và Spark SQL

Tên	Apache Pinot	Spark SQL
Mô tả	Kho dữ liệu OLAP phân tán theo thời gian thực, được thiết kế để trả lời các truy vấn OLAP với độ trễ thấp	Spark SQL là một thành phần trên đầu trang của 'Spark Core' để xử lý dữ liệu có cấu trúc
Mô hình cơ sở dữ liệu chính	DBMS quan hệ	DBMS quan hệ
Phát triển	Apache Software Foundation và những người đóng góp	Quỹ phần mềm Apache
Giấy phép	Mã nguồn mở	Mã nguồn mở
Ngôn ngữ triển khai	Java	Scala
Hệ điều hành máy chủ	Tất cả hệ điều hành có Java JDK11 trở lên	Hệ điều hành Linux X Windows

Lược đồ dữ liệu	Có	Có
.SQL	Ngôn ngữ truy vấn giống SQL	Các câu lệnh DML và DDL giống SQL
API và các phương pháp truy cập khác	JDBC .	JDBC ODBC .
Các ngôn ngữ lập trình được hỗ trợ	Go Java Python	Java Python R Scala .
Phương pháp phân vùng	Phân vùng ngang	Sử dụng Spark Core
Khái niệm người dùng	Kh	Không

CHƯƠNG 3: CÀI ĐẶT

3.1 Cách thiết lập cụm Pinot

Trong hướng dẫn, chúng em sẽ thiết lập cụm Pinot với các thành phần sau:

- 1 zookeeper
- 2 controllers
- 2 brokers
- 2 servers .Sau khi cụm được thiết lập và chạy, chúng ta sẽ xem cách tải dữ liệu vào Pinot và truy vấn nó.

3.2 điều kiện tiên quyết

Trước khi bắt đầu, hãy đảm bảo đầy đủ các điều kiện tiên quyết này:

- Tải xuống dữ liệu mẫu và cấu hình (<https://github.com/npawar/pinot-tutorial.git>)
- Tải xuống bản phát hành Pinot mới nhất(<https://pinot.apache.org>)
- Cài đặt Java 9 trở lên(<https://openjdk.java.net>)
- Cài đặt Apache Maven 3.5.0 trở lên (<https://maven.apache.org>)
- Cài đặt Zooinspector(<https://github.com/zzhang5/zooinspector>)
- Tải xuống Apache Kafka binary 2.4 hoặc cao hơn (<https://kafka.apache.org>)

3.3 Khởi động Zookeeper

Sử dụng tập lệnh khởi chạy trong thư mục từ bản phát hành Pinot.

bin/pinot-admin.sh StartZookeeper -zkPort 2181



```
~/org/apache/maven/maven-monitor/2.2.1/maven-monitor-2.2.1.jar
tavis@tavis-virtual-machine: ~/apache-pinot-0.11.0-bin
tavis@tavis-virtual-machine: ~/zooinspector$ cd
tavis@tavis-virtual-machine: $ cd apache-pinot-0.11.0-bin
tavis@tavis-virtual-machine: ~/apache-pinot-0.11.0-bin$ bin/pinot-admin.sh StartZ
ookeeper -zkPort 2181
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/tavis/apache-pinot-0.11.0-bin/lib/pinot-
all-0.11.0-jar-with-dependencies.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/tavis/apache-pinot-0.11.0-bin/plugins/pl
not-input-format/pinot-parquet/pinot-parquet-0.11.0-shaded.jar!/org/slf4j/impl/S
taticLoggerBinder.class]
```


3.4 Setup Zooinspector

Build

- `$git clone https://github.com/zzhang5/zooinspector.git`
- `$cd zooinspector/`
- `$mvn clean package`

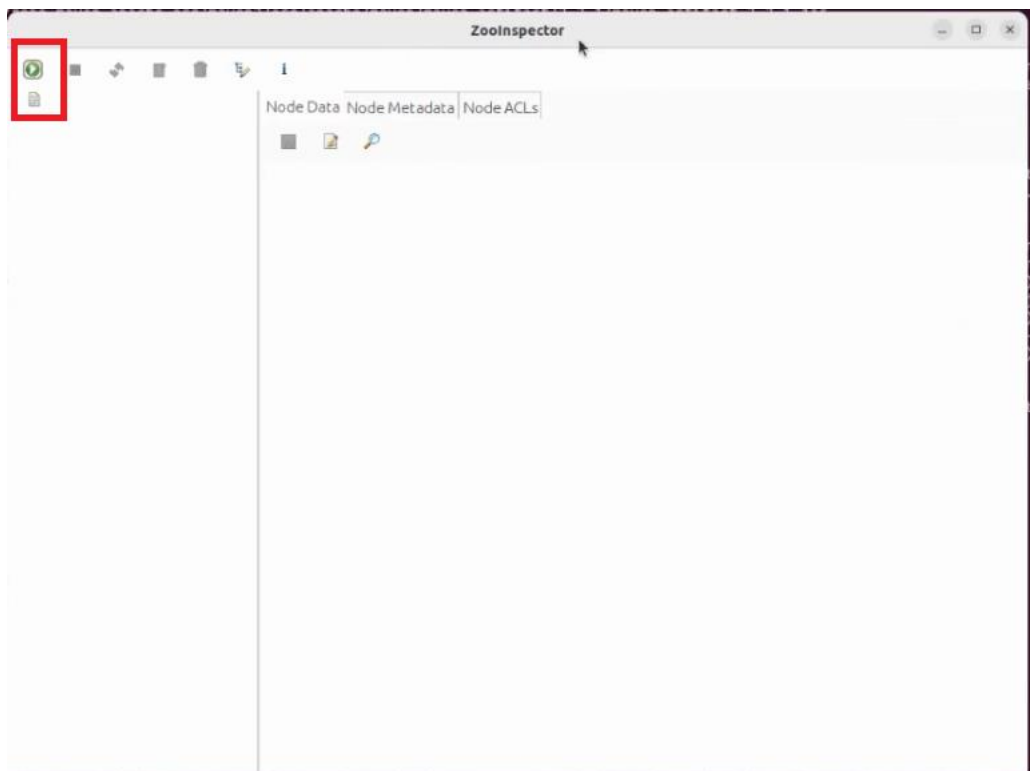
Run

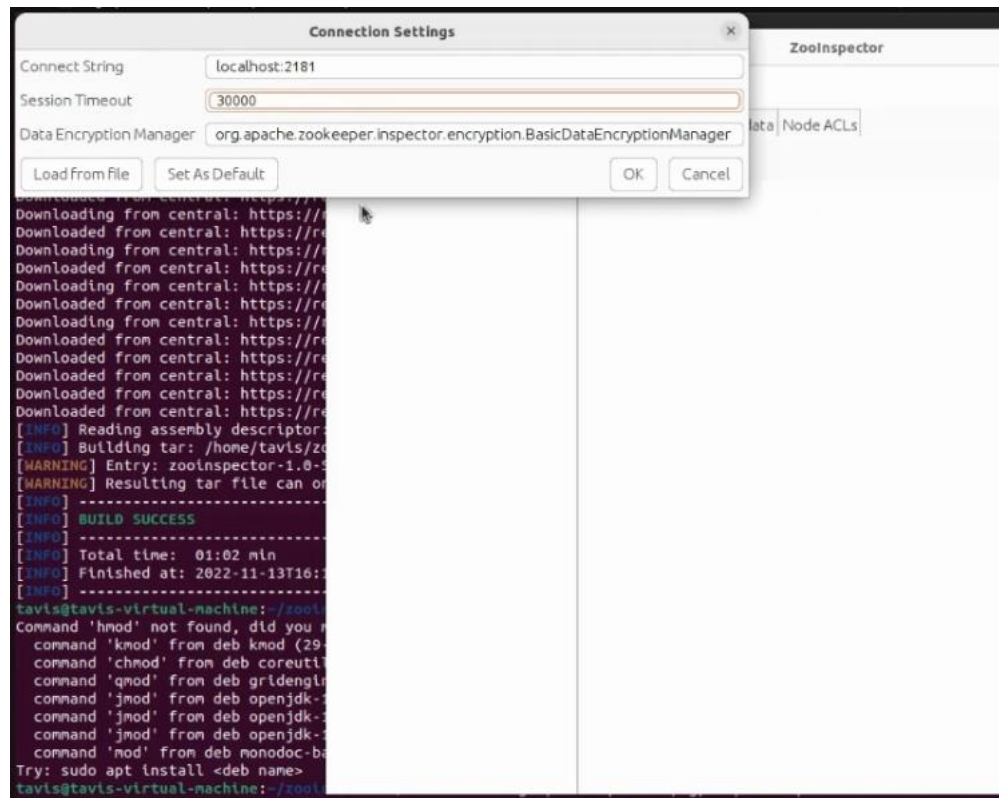
- `$chmod +x target/zooinspector-pkg/bin/zooinspector.sh`
- `$target/zooinspector-pkg/bin/zooinspector.sh`

```
try: sudo apt install <deb name>
tavis@tavis-virtual-machine:~/zooinspector$ chmod +x target/zooinspector-pkg/bin/zooinspector.sh
tavis@tavis-virtual-machine:~/zooinspector$ target/zooinspector-pkg/bin/zooinspector.sh
defaultHostsList: [localhost:2181]
init jtree: javax.swing.JTree[0,0,0x0,invalid,alignmentX=0.0,alignmentY=0.0,border=javax.swing.plaf.synth.SynthBe
okesStopCellEditing=false,largeModel=false,rootVisible=true,rowHeight=-1,scrollsOnExpand=false,showsRootHandles=tr
```

Kết nối Zookeeper với Zooinspector

- Nhấn vào biểu tượng để kết nối





- Sau khi kết nối sẽ xuất hiện thư mục Zookeeper



3.5 Khởi động Controller

Sử dụng tập lệnh khởi chạy trong thư mục từ bản phát hành Pinot

- Controller 1
`bin/pinot-admin.sh StartController \`
`-zkAddress localhost:2181 \`
`-clusterName PinotCluster \`
`-controllerPort 9001`

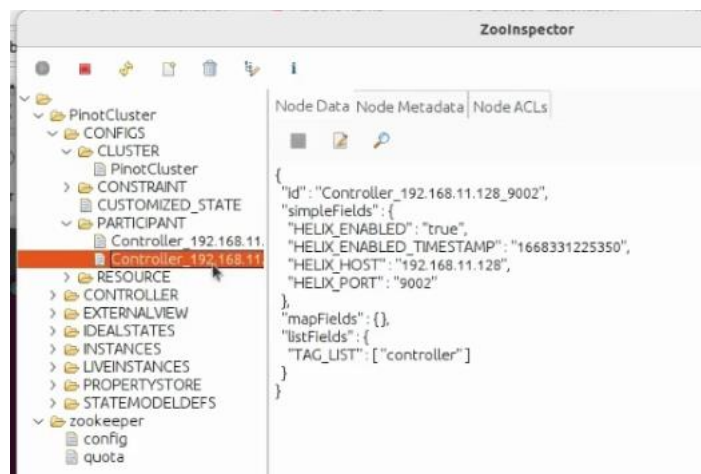
```
tavis@tavis-virtual-machine: ~/apache-pinot-0.11.0-bin
tavis@tavis-virtual-machine:~/apache-pinot-0.11.0-bin$ cd
tavis@tavis-virtual-machine: $ cd apache-pinot-0.11.0-bin
tavis@tavis-virtual-machine:~/apache-pinot-0.11.0-bin$ bin/pinot-admin.sh StartC
ontroller \
  -zkAddress localhost:2181 \
  -clusterName PinotCluster \
  -controllerPort 9801
```

- Controller 2

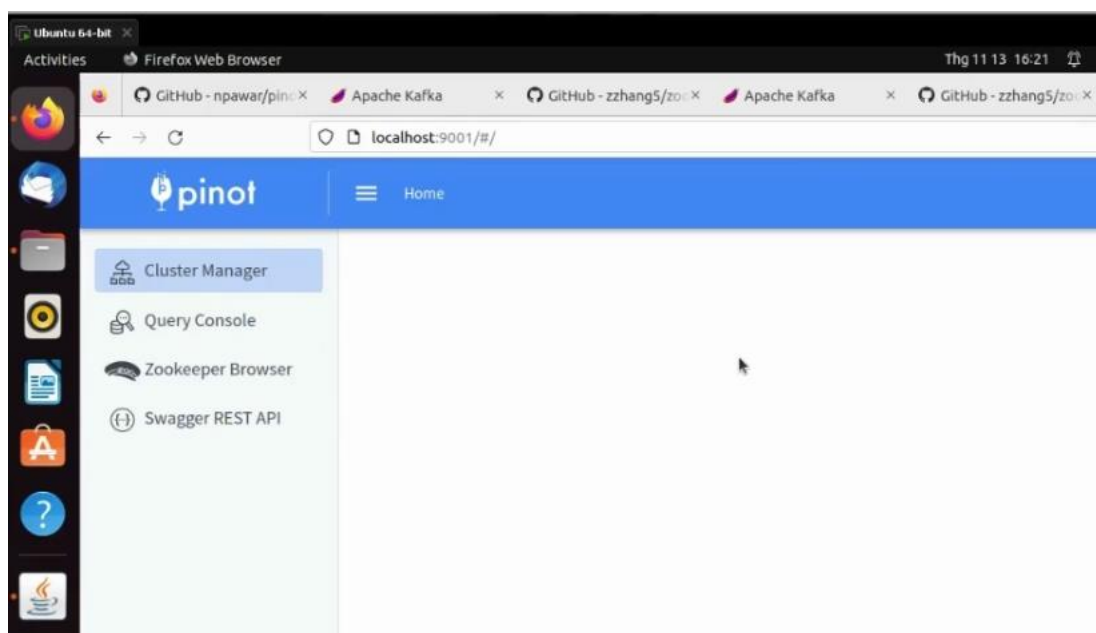
```
bin/pinot-admin.sh StartController \
  -zkAddress localhost:2181 \
  -clusterName PinotCluster \
  -controllerPort 9002
```

```
tavis@tavis-virtual-machine: ~/apache-pinot-0.11.0-bin
tavis@tavis-virtual-machine:~/apache-pinot-0.11.0-bin$ bin/pinot-admin.sh StartC
ontroller \
  -zkAddress localhost:2181 \
  -clusterName PinotCluster \
  -controllerPort 9002
```

Xem lại thay đổi trong Zookeeper



Chuyển đến cửa sổ trình duyệt và nhập localhost:9001



Có 4 lựa chọn ở đây:

- Cluster Manager
- Query Console
- Zookeeper Browser
- Rest API

3.6 Khởi động Broker

Sử dụng tập lệnh khởi chạy trong thư mục từ bản phát hành Pinot

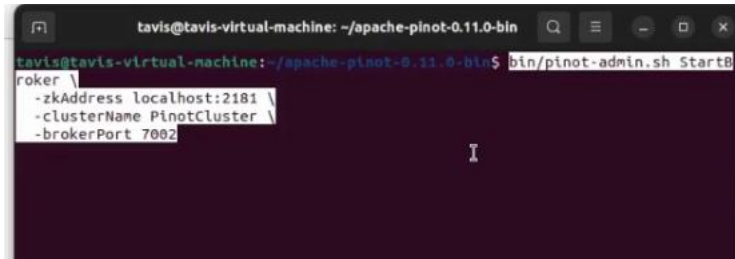
- Broker 1

```
bin/pinot-admin.sh StartBroker \  
-zkAddress localhost:2181 \  
-clusterName PinotCluster \  
-brokerPort 7001
```

```
tavis@tavis-virtual-machine: ~/apache-pinot-0.11.0-bin  
tavis@tavis-virtual-machine:~/apache-pinot-0.11.0-bin$ bin/pinot-admin.sh StartBroker \  
-zkAddress localhost:2181 \  
-clusterName PinotCluster \  
-brokerPort 7001
```

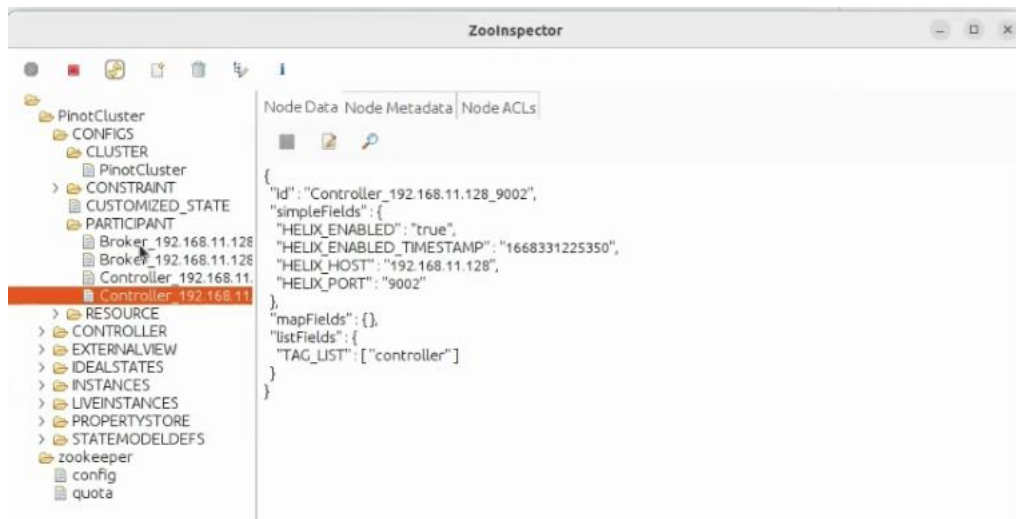
- Broker 2

```
bin/pinot-admin.sh StartBroker \
  -zkAddress localhost:2181 \
  -clusterName PinotCluster \
  -brokerPort 7002
```



```
tavis@tavis-virtual-machine: ~/apache-pinot-0.11.0-bin
tavis@tavis-virtual-machine:~/apache-pinot-0.11.0-bin$ bin/pinot-admin.sh StartBroker \
  -zkAddress localhost:2181 \
  -clusterName PinotCluster \
  -brokerPort 7002
```

Xem lại thay đổi trong Zookeeper



3.7 Khởi động Server

Sử dụng tập lệnh khởi chạy trong thư mục từ bản phát hành Pinot

- Server 1

```
bin/pinot-admin.sh StartServer \
  -zkAddress localhost:2181 \
  -clusterName PinotCluster \
  -serverPort 8001 -serverAdminPort 8011
```

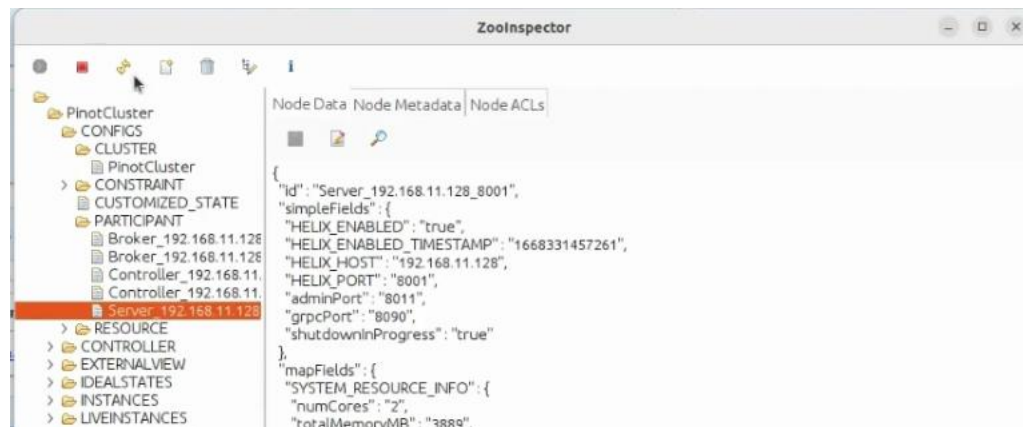
```
tavis@tavis-virtual-machine: ~/apache-pinot-0.11.0-bin
tavis@tavis-virtual-machine:~/apache-pinot-0.11.0-bin$ bin/pinot-admin.sh StartServer \
-zkAddress localhost:2181 \
-clusterName PinotCluster \
-serverPort 8001 -serverAdminPort 8011
```

- Server 2

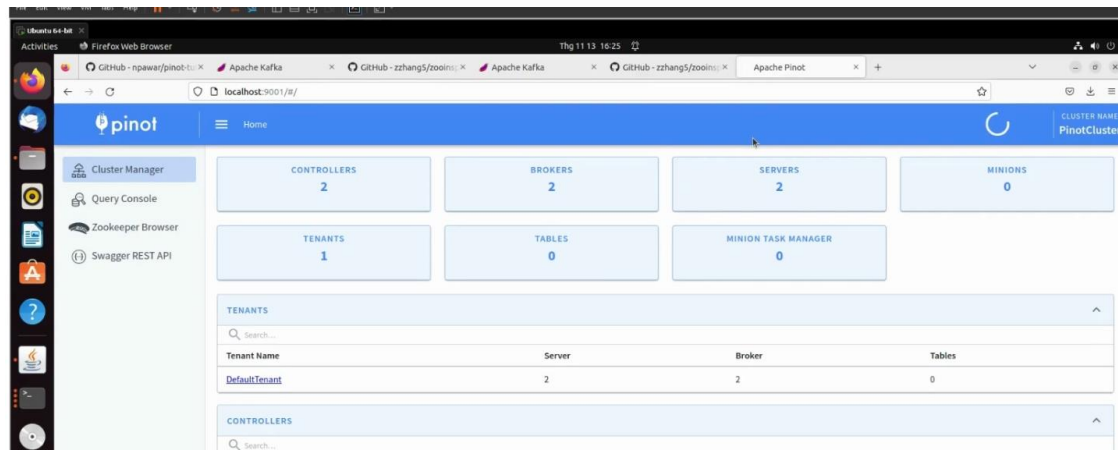
```
bin/pinot-admin.sh StartServer \
-zkAddress localhost:2181 \
-clusterName PinotCluster \
-serverPort 8002 -serverAdminPort 8012
```

```
tavis@tavis-virtual-machine: ~/apache-pinot-0.11.0-bin
tavis@tavis-virtual-machine:~/apache-pinot-0.11.0-bin$ bin/pinot-admin.sh StartServer \
-zkAddress localhost:2181 \
-clusterName PinotCluster \
-serverPort 8002 -serverAdminPort 8012
```

Xem lại thay đổi trong Zookeeper



Xem trên cửa sổ trình duyệt



Cụm được thiết lập! Khám phá cụm bằng Zooinspector hoặc <http://localhost:9001>.

Về cơ bản, Apache Spark bao gồm hai thành phần chính: trình điều khiển (driver) và trình thực thi (executors). Trình điều khiển dùng để chuyển đổi mã của người dùng thành nhiều tác vụ (tasks) có thể được phân phối trên các nút xử lý (worker nodes).

Trình thực thi chạy trên các nút xử lý và thực hiện các nhiệm vụ được giao cho chúng. Spark cũng có thể chạy ở chế độ cụm độc lập chỉ yêu cầu khung Apache Spark và JVM trên mỗi máy trong cụm. Tuy nhiên, sử dụng các công cụ quản lý cụm như trung gian giữa hai thành phần giúp tận dụng tài nguyên tốt hơn và cho phép phân bổ theo yêu cầu. Trong doanh nghiệp, Apache Spark có thể chạy trên Apache Mesos, Kubernetes và Docker Swarm.

CHƯƠNG 4: MINH HỌA VẬN HÀNH XỬ LÝ DỮ LIỆU LỚN

TẠO CẤU HÌNH LƯỢC ĐỒ

```
{
  "schemaName": "transcript",
  "dimensionFieldSpecs": [
    {
      "name": "studentID",
      "dataType": "INT"
    },
    {
      "name": "firstName",
      "dataType": "STRING"
    },
    {
      "name": "lastName",
      "dataType": "STRING"
    },
    {
      "name": "gender",
      "dataType": "STRING"
    },
    {
      "name": "subject",
      "dataType": "STRING"
    }
  ],
}
```



```

"metricFieldSpecs": [
  {
    "name": "score",
    "dataType": "FLOAT"
  }
],
"dateTimeFieldSpecs": [{
  "name": "timestampInEpoch",
  "dataType": "LONG",
  "format" : "1:MILLISECONDS:EPOCH",
  "granularity": "1:MILLISECONDS"
}]
}

```

TẠO CẤU HÌNH BẢNG

```

{
  "tableName": "transcript",
  "tableType": "OFFLINE",
  "segmentsConfig": {
    "replication": 1,
    "timeColumnName": "timestampInEpoch",
    "timeType": "MILLISECONDS",
    "retentionTimeUnit": "DAYS",
    "retentionTimeValue": 365
  },
  "tenants": {
    "broker": "DefaultTenant",
    "server": "DefaultTenant"
  },
}

```

```

"tableIndexConfig": {
  "loadMode": "MMAP"
},
"ingestionConfig": {
  "batchIngestionConfig": {
    "segmentIngestionType": "APPEND",
    "segmentIngestionFrequency": "DAILY"
  }
},
"metadata": {}
}

```

TẢI LÊN LƯỢC ĐỒ VÀ BẢNG

```

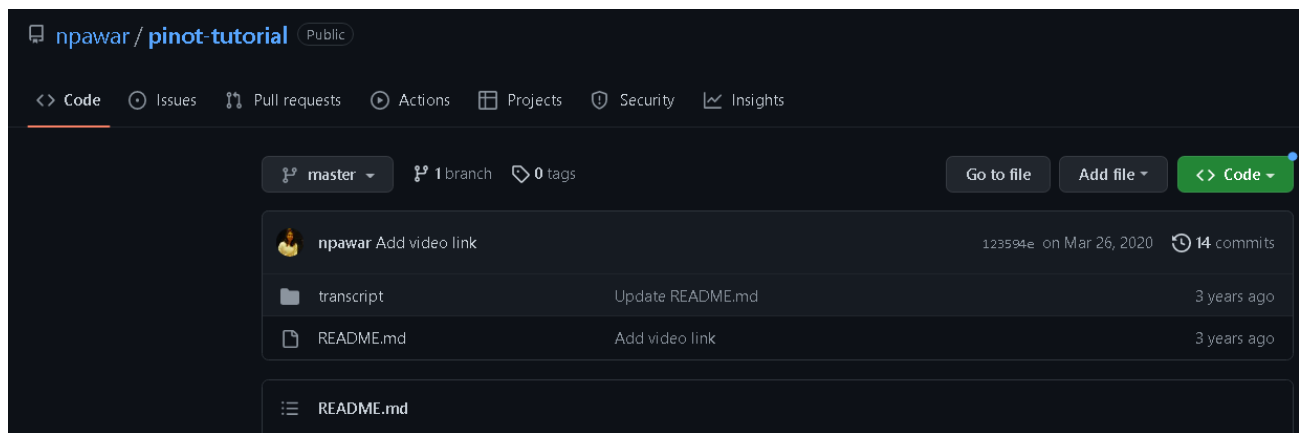
bin/pinot-admin.sh AddTable \
  -tableConfigFile /path/to/table-config.json \
  -schemaFile /path/to/table-schema.json -exec

```

NHẬP DỮ LIỆU TỪ MỘT DATASET BAO GỒM CÁC BƯỚC SAU:

1. Xác định lược đồ
2. Xác định cấu hình bảng
3. Tải lên cấu hình Lược đồ và Bảng
4. Tải dữ liệu lên

Tải data mẫu về máy theo link sau: [npawar/pinot-tutorial: Sample files for Pinot tutorial \(github.com\)](https://npawar.github.io/pinot-tutorial/sample-files-for-pinot-tutorial/)



Chỉnh sửa cấu hình file transcript

```
tavis@tavis-virtual-machine: ~/pinot-tutorial/transcript

executionFrameworkSpec:
  name: 'standalone'
  segmentGenerationJobRunnerClassName: 'org.apache.pinot.plugin.ingestion.batch.standalone.SegmentGenerationJobRunner'
  segmentTarPushJobRunnerClassName: 'org.apache.pinot.plugin.ingestion.batch.standalone.SegmentTarPushJobRunner'
  segmentUriPushJobRunnerClassName: 'org.apache.pinot.plugin.ingestion.batch.standalone.SegmentUriPushJobRunner'
jobType: SegmentCreationAndTarPush
inputDirURI: '$BASE_DIR/rawdata/'
includeFileNamePattern: 'glob:**/*.*csv'
outputDirURI: '$BASE_DIR/segments/'
overwriteOutput: true
pinotFSSpecs:
  - scheme: file
    className: org.apache.pinot.spi.filesystem.LocalPinotFS
recordReaderSpec:
  dataFormat: 'csv'
  className: 'org.apache.pinot.plugin.inputformat.csv.CSVRecordReader'
  configClassName: 'org.apache.pinot.plugin.inputformat.csv.CSVRecordReaderConfig'
tableSpec:
  tableName: 'transcript'
pinotClusterSpecs:
  - controllerURI: 'http://localhost:9001'
~
~
```

```
tavis@tavis-virtual-machine: ~/pinot-tutorial/transcript

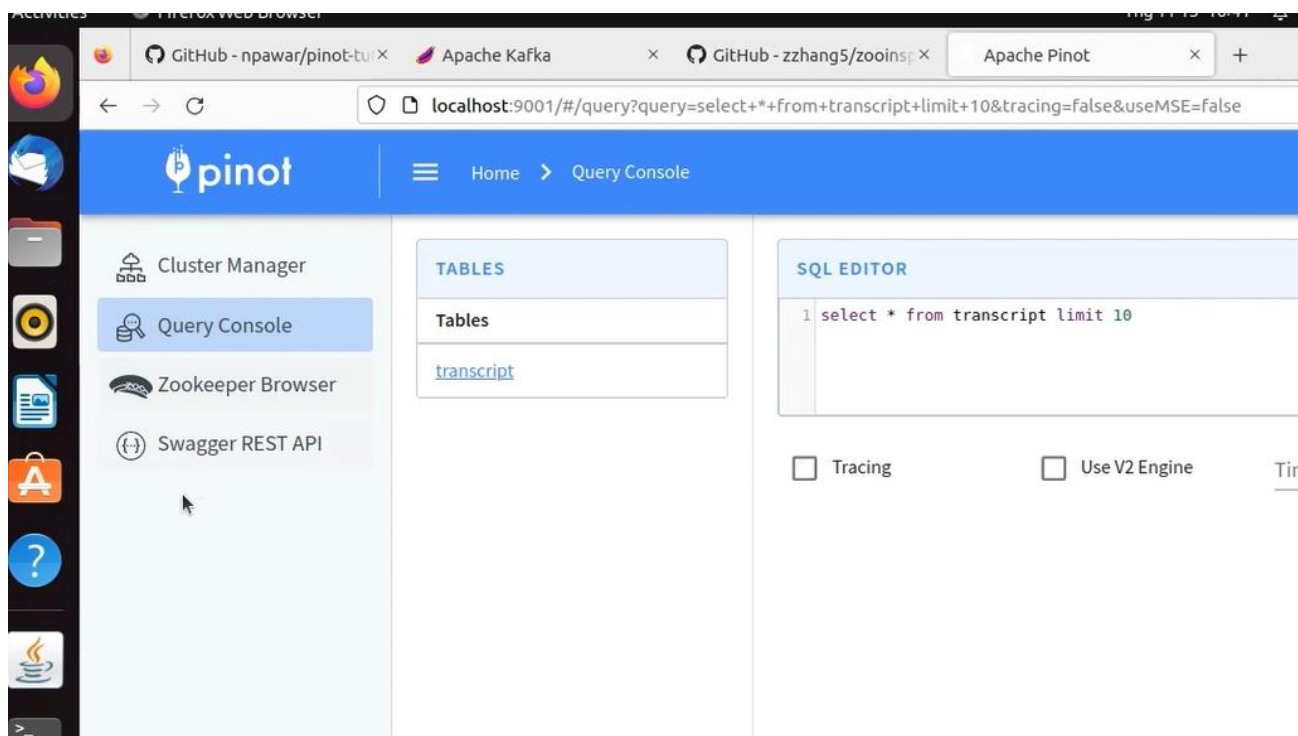
executionFrameworkSpec:
  name: 'standalone'
  segmentGenerationJobRunnerClassName: 'org.apache.pinot.plugin.ingestion.batch.standalone.SegmentGenerationJobRunner'
  segmentTarPushJobRunnerClassName: 'org.apache.pinot.plugin.ingestion.batch.standalone.SegmentTarPushJobRunner'
  segmentUriPushJobRunnerClassName: 'org.apache.pinot.plugin.ingestion.batch.standalone.SegmentUriPushJobRunner'
jobType: SegmentCreationAndTarPush
inputDirURI: /rawdata/
includeFileNamePattern: 'glob:**/*.*csv'
outputDirURI: '$BASE_DIR/segments/'
overwriteOutput: true
pinotFSSpecs:
  - scheme: file
    className: org.apache.pinot.spi.filesystem.LocalPinotFS
recordReaderSpec:
  dataFormat: 'csv'
  className: 'org.apache.pinot.plugin.inputformat.csv.CSVRecordReader'
  configClassName: 'org.apache.pinot.plugin.inputformat.csv.CSVRecordReaderConfig'
tableSpec:
  tableName: 'transcript'
pinotClusterSpecs:
  - controllerURI: 'http://localhost:9001'
~
~
```

```
es Terminal Thg 11 13 16:45
tavis@tavis-virtual-machine: ~/pinot-tutorial/transcript

executionFrameworkSpec:
  name: 'standalone'
  segmentGenerationJobRunnerClassName: 'org.apache.pinot.plugin.ingestion.batch.standalone.SegmentGenerationJobRunner'
  segmentTarPushJobRunnerClassName: 'org.apache.pinot.plugin.ingestion.batch.standalone.SegmentTarPushJobRunner'
  segmentUriPushJobRunnerClassName: 'org.apache.pinot.plugin.ingestion.batch.standalone.SegmentUriPushJobRunner'
jobType: SegmentCreationAndTarPush
inputDirURI: '/home/tavis/pinot-tutorial/transcript/rawdata/'
includeFileNamePattern: 'glob:**/*.*csv'
outputDirURI: '/home/tavis/pinot-tutorial/transcript/segments/'
overwriteOutput: true
pinotFSSpecs:
  - scheme: file
    className: org.apache.pinot.spl.filesystem.LocalPinotFS
recordReaderSpec:
  dataFormat: 'csv'
  className: 'org.apache.pinot.plugin.inputformat.csv.CSVRecordReader'
  configClassName: 'org.apache.pinot.plugin.inputformat.csv.CSVRecordReaderConfig'
tableSpec:
  tableName: 'transcript'
pinotClustersSpecs:
  - controllerURI: 'http://localhost:9001'
  ~
  ~
```

Add file vào Pinot bằng câu lệnh:

```
1552ea1a936ad145056f7033]
tavis@tavis-virtual-machine: ~/apache-pinot-0.11.0-bin$ bin/pinot-admin.sh AddTable -tableConfigFile $BASE_DIR/transcript-table-offline.json -schemaFile $BASE_DIR/transcript-schema.json -controllerPort 9001 -exec
```



Thêm data vào table đã tạo:

```
WARNING: Please consider reporting this to the maintainers of org.codehaus.groovy.reflection.CachedClass
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
Executing command: AddTable -tableConfigFile /home/tavis/pinot-tutorial/transcript-table-offline.json -schemaFile /home/tavis/pinot-tutorial/http -controllerHost 192.168.11.128 -controllerPort 9001 -user null -password [hidden] -exec
{"unrecognizedProperties": {}, "status": "TableConfigs transcript successfully added"}
tavis@tavis-virtual-machine: ~/apache-pinot-0.11.0-bin$ bin/pinot-admin.sh LaunchDataIngestionJob -jobSpecFile $BASE_DIR/batch-job-spec.yml
```

Home > Query Console

TABLES

Tables

[transcript](#)

TRANSCRIPT SCHEMA

Column	Type
studentID	INT
firstName	STRING
lastName	STRING
gender	STRING
subject	STRING
score	FLOAT

☐ Tracing ☐ Use V2 Engine Timeout (in Milliseconds) RUN QUERY

QUERY RESPONSE STATS

timeUsedMs	numDocsScanned	totalDocs	numServersQueried	numServersResponded	numSegmentsQueried	numSegmentsProcessed	numSegmentsMatched	numC
63	4	4	1	1	1	1	1	0

EXCEL CSV COPY Show JSON fr

QUERY RESULT

firstName	gender	lastName	score	studentID	subject	timestamp
Lucy	Female	Smith	3.8	200	Maths	1570863600000
Lucy	Female	Smith	3.5	200	English	1571036400000
Bob	Male	King	3.2	201	Maths	1571900400000
Nick	Male	Young	3.6	202	Physics	1572418800000

Thực hiện các câu lệnh truy vấn:

TABLES

Tables

[transcript](#)

TRANSCRIPT SCHEMA

Column	Type
studentID	INT
firstName	STRING
lastName	STRING
gender	STRING
subject	STRING
score	FLOAT

☐ Tracing ☐ Use V2 Engine Timeout (in Milliseconds)

SQL EDITOR

```
1 select count(*) from transcript limit 10
```

QUERY RESPONSE STATS

timeUsedMs	numDocsScanned	totalDocs	numServersQueried	numServersResponded
431	4	4	1	1

EXCEL CSV COPY

QUERY RESULT

count(*)
4

SQL EDITOR

1 select sum(score) from transcript group by subject

☐ Tracing
☐ Use V2 Engine
Timeout (in Milliseconds)

QUERY RESPONSE STATS

Search...

timeUsedMs	numDocsScanned	totalDocs	numServersQueried	numServersResponded	numSeg
356	4	4	1	1	1

EXCEL
CSV
COPY

QUERY RESULT

Search...

sum(score)
7
3.5

THAM KHẢO

1. [Apache Pinot vs. Apache Spark Comparison \(sourceforge.net\)](https://sourceforge.net)
2. [GitHub - npawar/pinot-tutorial: Các tệp mẫu cho hướng dẫn Pinot](https://github.com/npawar/pinot-tutorial)
3. Apache Spark là gì? Tổng quan thông tin cần biết về Apache Spark(link: <https://bizflycloud.vn/tin-tuc/apache-spark-20210726115949898.htm>)
4. Pinot introduction(link: <https://docs.pinot.apache.org/>)
5. Tìm hiểu về Apache Spark(link: <https://viblo.asia/p/tim-hieu-ve-apache-spark-ByEZkQQW5Q0>)
6. Tổng quan về Apache Spark cho hệ thống Big Data(link: <https://viblo.asia/p/tong-quan-ve-apache-spark-cho-he-thong-big-data-RQqKLxR6K7z>)

7. Apache Pinot architecture(link: <https://www.decipherzone.com/blog-detail/apache-pinot-architecture>)
8. Compare Apache Pinot vs Apache Spark(link: <https://db-engines.com/en/system/Apache+Pinot%3BSpark+SQL%3BdBASE>)
9. Official website [Apache Pinot™: Realtime distributed OLAP datastore | Apache Pinot™](#)
- 10.Official document [Introduction - Apache Pinot Docs](#).

Hết.