BigQuery Columnar Storage

강사 : 고병화

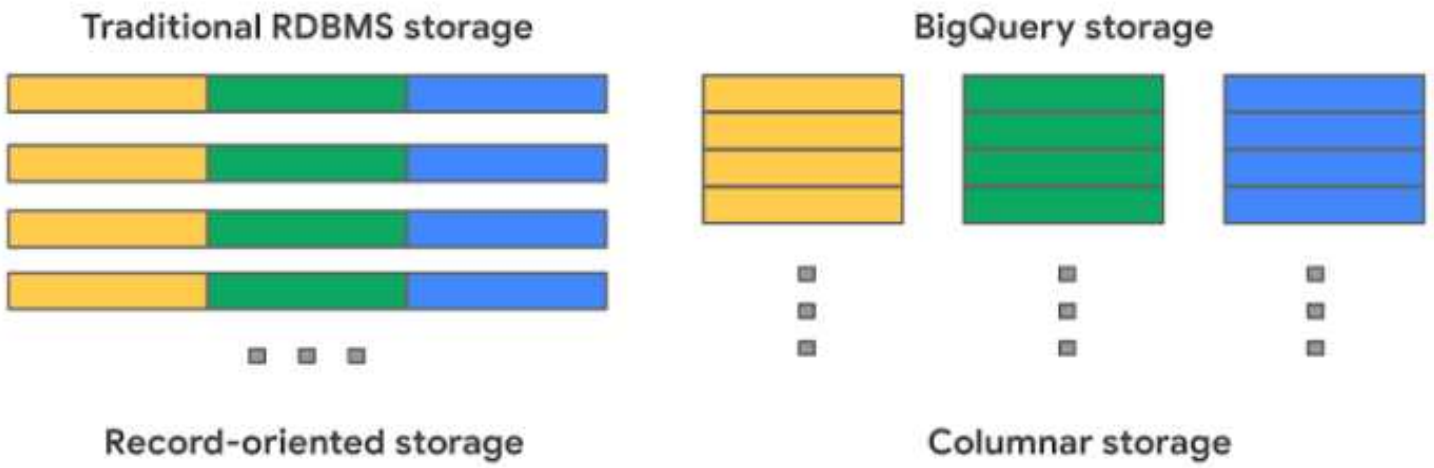
Record Oriented Storage vs Columnar Storage

전통적인 관계형 데이터베이스에서는 데이터를 한 행씩 레코드지향 스토리지에 저장한다. 이로 인해 트랜잭션 업데이트 등의 OLTP 유스케이스에서 훌륭하게 동작한다. 발생하는 트랜잭션에 대해서 테이블 업데이트시 하나의 레코드 단위로 읽거나 쓰거나 하기 때문이다.
반대로 어떤 컬럼내의 값들의 총합을 계산하는 것과 같은 집계 수행시에는 전체 테이블의 내용을 메모리에 올려야 하는 부담이 생긴다.

빅쿼리는 컬럼지향 스토리지를 사용하는데 여기서는 각각의 컬럼이 분리된 파일 블럭에 저장된다. 이로 인해 빅쿼리는 OLAP 유스케이스에 이상적으로 작동한다.
집계 연산이 필요한 경우 전체 테이블을 메모리에 올리는 대신 집계에 필요한 컬럼이 저장된 파일 블럭들 만 스토리지에서 읽어 내면 된다.

https://velog.io/@jaytiger/BigQuery-Architecture-Storage-Internals

BigQuery Columnar Storage 포맷



https://cloud.google.com/blog/topics/developers-practitioners/bigquery-admin-reference-guide-storage

BigQuery Columnar Storage 포맷의 특징

record를 column으로 분리하여 각각 다른 storage volume에 저장하여 매우 빠른 속도로 읽기가 가능하다
반대로 업데이트 속도는 느려진다. BigQuery는 record를 업데이트 하지 않음
BigQuery는 OLAP(Online Analytic Processing)에 적합하다

BigQuery Columnar Storage 포맷 - Capacitor

내부적으로 빅쿼리는 데이터를 Capacitor라는 컬럼지향 형식으로 저장한다.
각각의 필드 또는 컬럼의 값들이 분리되어 저장되면 파일을 읽어들일 때의 오버헤드는 실제로 읽고 있는 필드의 갯수에 비례하게 된다.
각 컬럼들이 각자 하나씩의 파일에 저장된다는 의미는 아니다.
각 컬럼이 최적의 성능을 내도록 독립적으로 압축된 후 하나의 파일 블럭에 저장된다
Capacitor가 근사(approximation)모델을 구축해서 데이터 타입(아주 긴 문자열 vs. 정수형)이나 사용패턴(예를 들어, 어떤 컬럼은 WHERE 조건절에서 필터의 용도로 자주 사용된 다거나)과 같은 연관된 인자들을 고려하여 행(row)을 다시 섞거나(reshuffle) 열(column)을 인코딩(encode)한다
모든 컬럼이 인코딩 되는 동안에 빅쿼리는 데이터에 대한 다양한 통계 데이터를 수집한다.통계 데이터는 따로 보관되었다가 향후 쿼리가 실행되는 동안 사용되어진다.

https://velog.io/@jaytiger/BigQuery-Architecture-Storage-Internals

BigQuery Columnar Storage 포맷 - Capacitor

UID	First Name	Last Name	Age	City	Purchases

349872349873	Ashuk	Patel	34	Chicago	[
					{
					"pur_id": 3459872980,
					"UID": 349872349873,
					"Product": "Google Home",
					"Price": 100.01
					},
					{
					"pur_id": 38479782,
					"UID": 349872349873,
					"Product": "Pixel 4",
					"Price": 550.5
					},
					{
					"pur_id": 8937492,
					"UID": 349872349873,
					"Product": "Pixel 4 case",
					"Price": 20.23
					}
					]

42398714298	Jessie	Walters	20	New York	[
					{
					""pur_id"": 3459872980,
					""UID"": 349872349873,
					""Product"": ""Google Home"",
					""Price"": 100.01
					}
					]

3498734871	Lisa	LaBlenc	54	Austin	[{...}]
34598792358	Greg	Smith	28	Boston	{...}]

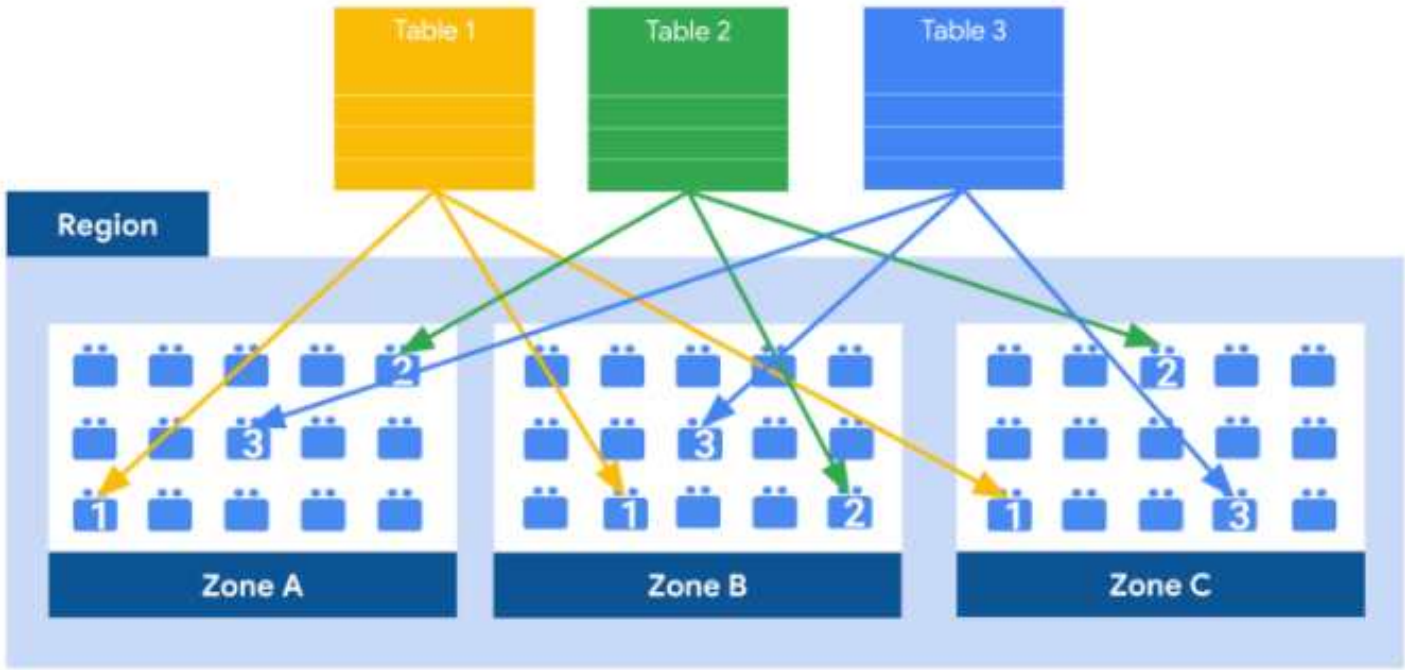
https://velog.io/@jaytiger/BigQuery-Architecture-Storage-Internals

암호화와 내구성 관리(Encryption and managed durability)

빅쿼리의 영속 계층 (Persistence Layer)은 구글의 분산 파일 시스템인 Colossus가 제공하고 있다. 이 곳에서 데이터는 자동으로 압축되고 암호화되고 복제되고 분산된다.
Colossus는 삭제 인코딩(erasure encoding)이라는 것을 통해 내구성을 보장하고 있다. 삭제 인코딩은 데이터를 쪼개서 서로 다른 디스크들에 골고루 여분의 (데이터)조각들이 저장될 수 있도록 한다.
또한 데이터의 내구성과 가용성을 보장하기 위해서 이 데이터들은 데이터셋을 생성한 권역(Region)의 서로 다른 가용존(AZ)에 복제된다.
이것은 데이터가 서로 다른 전력 체계와 네트워크를 가진 서로 다른 건물에 보관된다는 의미이다.
좀 더 안전하게 멀티리전을 사용하는 경우 빅쿼리는 복사본 하나를 더 권역 외 복제본(off-region replica)으로 만든다.
이런 방식으로 데이터는 주요 재해 발생시에도 복구가 가능하게 된다.

https://velog.io/@jaytiger/BigQuery-Architecture-Storage-Internals

암호화와 내구성 관리(Encryption and managed durability)



https://velog.io/@jaytiger/BigQuery-Architecture-Storage-Internals

BigQuery 구조

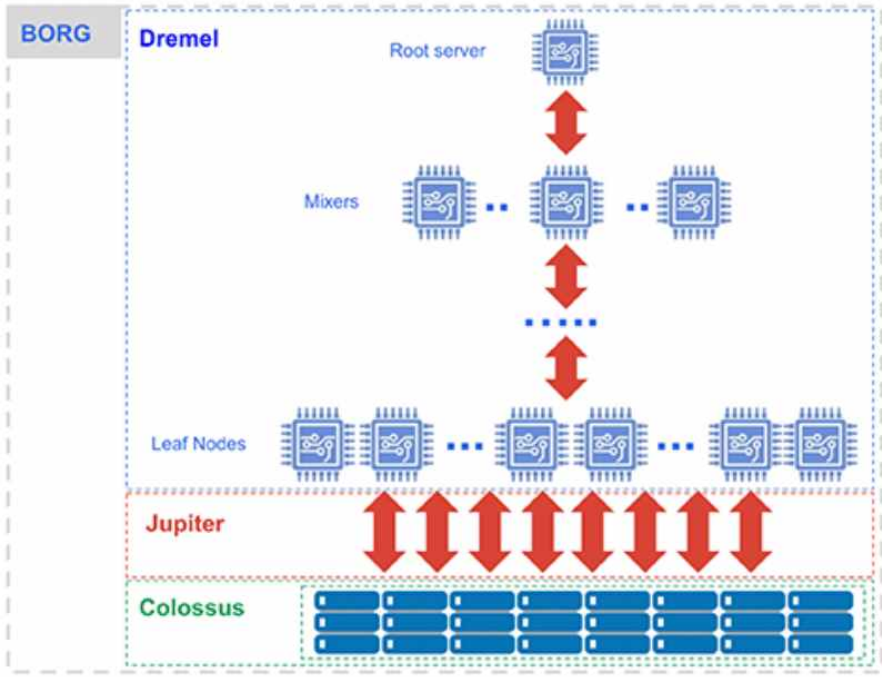
빅 쿼리는 크게 네 가지 구성 요소로 이루어져 있다

Dremel(Compute): 방대한 분산 노드들에서 SQL 쿼리를 실행, Execution Engine

Colossus(Storage): 데이터를 저장하고 실시간 처리를 할 수 있는 구글의 분산 파일 시스템, Distributed Storage

Jupiter(Network): Compute와 Storage 사이의 통신 담당

Borg(Orchestration): 이 모든 분산 노드들을 조율 및 운영, 쿠버네티스의 전신



https://yeomko.tistory.com/29