AWS 기반 데이터 웨어하우징

2016년 3월

고지 사항

이 문서는 정보 제공 목적으로만 제공됩니다. 본 문서의 발행일 당시 AWS의 현재 제품 및 실행방법을 설명하며, 예고 없이 변경될 수 있습니다. 고객은 본 문서에 포함된 정보나 AWS 제품 또는 서비스의 사용을 독립적으로 평가할 책임이 있으며, 각 정보 및 제품은 명시적이든 묵시적이든 어떠한 종류의 보증 없이 "있는 그대로" 제공됩니다. 본 문서는 AWS, 그 계열사, 공급업체 또는 라이선스 제공자로부터 어떠한 보증, 표현, 계약 약속, 조건 또는 보증을 구성하지 않습니다. 고객에 대한 AWS의 책임 및 의무는 AWS 계약에 준거합니다. 본 문서는 AWS와 고객 간의 어떠한 계약도 구성하지 않으며 이를 변경하지도 않습니다.

목차

요약

전 세계 기업의 데이터 엔지니어, 데이터 분석가 및 개발자들은 데이터 웨어하우징을 클라우드로 마이그레이션하여 성능을 개선하고 비용을 낮추기 위한 방안을 모색하고 있습니다. 이 백서에서는 분석 및 데이터 웨어하우징 아키텍처에 대한 최신 전략을 설명하고, 이 아키텍처를 구축하기 위해 AWS(Amazon Web Services)에서 사용할 수 있는 서비스를 소개하며, 이러한 서비스를 사용하여 데이터 웨어하우징 솔루션을 구축하기 위한 일반적인 설계 패턴을 제시합니다.

서론

현대 사회에서 데이터와 분석은 비즈니스에 있어 없어서는 안 될 요소입니다. 거의 모든 대기업에서는 자체 트랜잭션 처리 시스템 및 기타 데이터베이스를 비롯한 다양한 소스의 데이터를 통해 보고 및 분석 용도로 데이터 웨어하우스를 구축하고 있습니다.

하지만 하나 또는 여러 데이터 소스로부터의 정보를 받아 보관하는 중앙 저장소인 데이터 웨어하우스를 구축하고 실행하는 것은 복잡하고 비용이 많이 드는 일입니다. 대부분의 데이터 웨어하우징 시스템은 구축 작업이 복잡하고, 갖춰야 하는 소프트웨어와 하드웨어에 수십억 원의 비용이 들며, 계획, 조달, 구축 및 배포 프로세스에 수개월이 걸릴 수 있습니다. 초기 투자를 통해 데이터 웨어하우스를 구축한 후에는 쿼리를 빠르게 실행하고 데이터 손실을 방지하기 위해 데이터베이스 관리자 팀을 고용해야 합니다.

또한 기존의 데이터 웨어하우스는 확장이 어렵습니다. 데이터 볼륨이 증가하거나 더 많은 사용자에게 분석 및 보고를 제공하기 위해서는, 쿼리 성능의 저하를 감수하거나 비용이 많이 드는 업그레이드 프로세스에 시간과 노력을 투자해야 합니다. 실제로 일부 IT 팀에서는 기존의 서비스 수준 계약(SLA)을 유지하기 위해 데이터 증가나 쿼리 추가를 자제하고 있습니다. 많은 기업에서는 기존의 데이터베이스 공급업체와 원만한 관계를 유지하는 문제로 고심하고 있습니다. 관리형 시스템을 위해 하드웨어를 업그레이드하거나, 만료된 계약 라이선스에 대해 장기전의 협상에 들어갈 수밖에 없습니다. 한 데이터 웨어하우스 엔진에서 확장 한도에 도달하면 동일한 공급업체의 또 다른 엔진으로 마이그레이션해야 하며 이러한 엔진은 서로 다른 SQL 시맨틱을 사용합니다.

Amazon Redshift는 기능과 성능을 희생하지 않으면서 데이터 웨어하우스 시스템 구축에 드는 비용과 노력을 상당히 절감함으로써 기업의 데이터 웨어하우징 개념을 바꿔 놓았습니다. Amazon Redshift는 신속하고 완벽하게 관리되는 페타바이트 규모의 데이터 웨어하우징 솔루션으로서, 기존 비즈니스 인텔리전스(BI) 도구를 사용하여 대량의 데이터를 합리적인 비용으로 간편하게 분석할 수 있게 해 줍니다. Amazon Redshift를 사용하면 1/10의 비용으로 대량 병렬 처리(MPP)를 수행하는 컬럼 방식 데이터 웨어하우징 엔진의 성능을 얻을 수 있습니다. 약정이 없는 시간당 $0.25의 소규모로 시작하여, 1년에 테라바이트당 $1,000의 규모로 확장할 수 있습니다.

2013년 2월에 출시된 Amazon Redshift는 현재 가장 빠른 성장을 보이는 AWS 서비스 중 하나가 되었으며, 업종과 규모에 상관없이 수많은 고객이 사용하고 있습니다. Amazon Redshift로 마이그레이션한 기업에는 NTT DOCOMO, FINRA, Johnson & Johnson, Hearst, Amgen, NASDAQ 등이 있습니다. 이렇게 빠른 성장을 보인 Amazon Redshift는 *Forrester Wave: 2015년 4분기 엔터프라이즈 데이터 웨어하우스* 보고서에서 선도적인 솔루션으로 인정받았습니다.[1]

이 백서에서는 온프레미스의 데이터 웨어하우징 영역을 클라우드로 전환함에 따라 발생하는 전략적 이점을 활용하는 데 필요한 정보를 제공합니다.

- 최신 분석 아키텍처
- 해당 아키텍처에서 사용할 수 선택할 수 있는 데이터 웨어하우징 기술
- Amazon Redshift와 그 차별화된 기능에 대한 심층 분석
- Amazon Redshift 및 기타 서비스를 사용하여 AWS 기반의 완전한 데이터 웨어하우징 시스템을 구축하기 위한 청사진
- 다른 데이터 웨어하우징 솔루션에서의 마이그레이션 및 AWS 파트너 에코시스템 활용을 위한 팁

최신 분석 및 데이터 웨어하우징 아키텍처

*데이터 웨어하우스*는 하나 또는 여러 데이터 소스로부터의 정보를 보관하는 중앙 저장소입니다. 일반적으로 데이터는 트랜잭션 시스템 및 기타 관계형 데이터베이스에서 데이터 웨어하우스로 유입되며, 이러한 데이터는 정형, 반정형, 비정형의 구조를 가집니다. 이 데이터는 일반 케이던스에서 처리 및 변형되고 수집됩니다. 데이터 과학자, 비즈니스 분석가 및 의사결정자 등과 같은 사용자들은 BI 도구, SQL 클라이언트 및 스프레드시트를 통해 데이터에 액세스합니다.

데이터 웨어하우스를 구축하는 이유는 무엇일까요? 트랜잭션이 기록되는 온라인 트랜잭션 처리(OLTP) 데이터베이스에서 분석 쿼리를 직접 실행하지 않는 이유는 무엇일까요? 이에 대한 답을 찾기 위해 데이터 웨어하우스와 OLTP 데이터베이스 간의 차이를 알아보겠습니다. 데이터 웨어하우스는 일괄 쓰기 작업과 대량의 데이터 읽기에 최적화되어 있는 반면, OLTP 데이터베이스는 지속적인 쓰기 작업과 많은 양의 소규모 읽기 작업에 최적화되어 있습니다. 일반적으로 데이터 웨어하우스는 데이터 처리량에 대한 높은 요구 사항으로 인해 Star 스키마 및 Snowflake 스키마와 같은 비정형 스키마를 사용하는 반면, OLTP 데이터베이스는 트랜잭션 처리량에 대한 높은 요구 사항을 위해 고도로 정형화된 스키마를 사용합니다. Star 스키마는 많은 차원 테이블을 참조하는 몇 개의 대형 팩트 테이블로 구성되어 있습니다. Star 스키마의 확장인 Snowflake 스키마는 훨씬 더 정형화된 차원 테이블로 구성되어 있습니다.

소스 OLTP 또는 다른 소스 시스템을 통해 별도의 데이터 스토어로 관리되는 데이터 웨어하우스의 이점을 활용하려면 효율적인 데이터 파이프라인을 구축해야 합니다. 이러한 파이프라인은 소스 시스템에서 데이터를 추출하여 데이터 웨어하우징에 적합한 스키마로 변환한 후 이를 데이터 웨어하우스에 로드합니다. 다음 단원에서는 분석 파이프라인의 구성 요소와, 파이프라인의 아키텍처를 설계하는 데 사용할 수 있는 다양한 AWS 서비스에 대해 설명합니다.

분석 아키텍처

분석 파이프라인은 데이터베이스, 애플리케이션, 디바이스 등과 같이 다양한 소스로부터 유입되는 많은 양의 데이터 흐름을 처리하도록 설계되었습니다.

일반적인 분석 파이프라인은 다음과 같은 단계를 갖습니다.

1. 데이터 수집
2. 데이터 저장
3. 데이터 처리
4. 데이터 분석 및 시각화

아래 그림 1을 참조하십시오.

그림 1: 분석 파이프라인

데이터 수집

데이터 수집 단계에서는 트랜잭션 데이터, 로그 데이터, 스트리밍 데이터, 사물 인터넷(IoT) 데이터 등과 같은 다양한 유형의 데이터가 포함될 것을 고려해야 합니다. AWS는 이러한 각각의 데이터 유형에 대한 솔루션을 제공합니다.

트랜잭션 데이터

전자 상거래 구매 트랜잭션이나 금융 트랜잭션 등과 같은 트랜잭션 데이터는 일반적으로 관계형 데이터베이스 관리 시스템(RDBMS)이나 NoSQL 데이터베이스 시스템에 저장됩니다. 데이터베이스 솔루션을 선택할 때는 사용 사례와 애플리케이션 특성을 고려합니다. NoSQL 데이터베이스는 데이터가 정의된 스키마에 맞게 제대로 정형화되지 않았거나 스키마가 매우 자주 변경될 경우에 적합합니다. 반면에 RDBMS 솔루션은 복잡한 조인이 필요한 여러 테이블 행과 쿼리에서 트랜잭션이 발생할 경우에 적합합니다. Amazon DynamoDB는 완벽하게 관리되는 NoSQL 데이터베이스로서, 용도에 맞는 OLTP 스토어로 사용할 수 있습니다. Amazon RDS를 사용하면 원하는 용도의 SQL 기반 관계형 데이터베이스 솔루션을 구축할 수 있습니다.

로그 데이터

시스템에서 생성되는 로그를 확실히 캡처하면 로그에 저장된 정보를 사용하여 문제를 해결하고, 감사 및 분석을 수행하는 데 도움이 됩니다. Amazon S3(Amazon Simple Storage Service)는 로그 데이터와 같이 분석에 사용되는 비 트랜잭션 데이터용으로 널리 사용되는 스토리지 솔루션입니다. Amazon S3는 99.999999999퍼센트의 내구력을 제공하기 때문에 대표적인 아카이브 솔루션이기도 합니다.

스트리밍 데이터

웹 애플리케이션, 모바일 디바이스 및 많은 소프트웨어 애플리케이션과 서비스는 시간당 수 테라바이트에 이르는 엄청난 양의 스트리밍 데이터를 생성할 수 있으며 이러한 데이터는 지속적으로 수집 및 저장되고 처리되어야 합니다.[2] Amazon Kinesis 서비스를 사용하면 경제적인 비용으로 간단하게 이를 처리할 수 있습니다.

IoT 데이터

모든 디바이스와 센서는 지속적으로 메시지를 전송합니다. 오늘날 기업들은 이러한 데이터를 캡처하고 그로부터 인텔리전스를 얻어 내야 할 필요성이 많아지고 있습니다. AWS IoT를 사용하면 연결된 디바이스가 AWS 클라우드를 통해 간편하고도 안전하게 상호 작용할 수 있습니다. AWS IoT는 인프라스트럭처를 관리하지 않고도 AWS Lambda, Amazon Kinesis, Amazon S3, Amazon Machine Learning, Amazon DynamoDB와 같은 AWS 서비스를 사용하여 IoT 데이터를 수집, 처리, 분석 및 상호 작용하는 애플리케이션을 쉽게 개발할 수 있게 해줍니다.

데이터 처리

수집 프로세스는 유용한 정보를 포함할 수 있는 데이터를 제공합니다. 추출된 정보를 분석하여 비즈니스 성장에 도움이 될 인텔리전스를 얻을 수 있습니다. 예를 들어 이 인텔리전스는 사용자의 행동과 자사 제품의 상대적인 인기도를 알려 줄 수 있습니다. 이 인텔리전스를 수집하는 모범 사례는 원시 데이터를 데이터 웨어하우스에 로드하여 추가로 분석을 수행하는 것입니다.

이를 위해 배치(batch)와 실시간이라는 두 가지 유형의 처리 워크플로가 사용됩니다. 가장 일반적인 처리 방식인 온라인 분석 처리(OLAP)와 OLTP는 각각 이들 유형 중 하나를 사용합니다. 온라인 분석 처리(OLAP)는 일반적으로 배치 기반입니다. 반면에 OLTP 시스템은 실시간 처리 기반이므로 일반적으로 배치 기반 처리에는 적합하지 않습니다. OLTP 시스템에서 데이터 처리를 분리할 경우 데이터 처리가 OLTP 워크로드에 영향을 주지 않도록 해야 합니다.

먼저 배치 처리에는 어떤 프로세스가 포함되는지 살펴보겠습니다.

ETL(Extract Transform Load)

ETL은 여러 소스로부터 데이터를 가져와 데이터 웨어하우징 시스템에 로드하는 프로세스입니다. ETL은 일반적으로 정의된 워크플로를 통해 지속적으로 진행되는 프로세스입니다. 이 프로세스를 통해 데이터는 하나 이상의 소스로부터 처음 추출됩니다. 추출된 데이터는 정리 및 보강되고 변형되어 데이터 웨어하우스에 로드됩니다. ETL 파이프라인에서 대량의 데이터에 대한 변형을 수행하기 위해 Apache Pig 및 Apache Hive와 같은 하둡 프레임워크 툴이 일반적으로 사용됩니다.

ELT(Extract Load Transform)

ELT는 ETL의 변종으로 추출된 데이터가 대상 시스템에 먼저 로드됩니다. 변형은 데이터가 데이터 웨어하우스에 로드된 후에 수행됩니다. ELT는 일반적으로 대상 시스템이 변형을 처리할 만큼 강력할 경우에 효과가 좋습니다. Amazon Redshift는 변형 수행에 있어 효율이 높기 때문에 ELT 파이프라인에 종종 사용됩니다.

OLAP(Online Analytical Processing)

OLAP 시스템은 취합된 이력 데이터를 다차원 스키마에 저장합니다. 데이터 마이닝에 많이 사용되는 OLAP 시스템을 통해 데이터를 추출하고 다차원적인 추이를 포착할 수 있습니다. OLAP는 패스트 조인(fast join)에 최적화되었기 때문에 Amazon Redshift는 OLAP 시스템을 개발하는 데 종종 사용됩니다.

이제 실시간 데이터 처리에는 어떤 프로세스가 포함되는지 살펴보겠습니다.

실시간 처리

위에서 스트리밍 데이터에 대해 설명하면서 스트리밍 데이터를 캡처하고 저장하는 솔루션으로 Amazon Kinesis를 언급했습니다. 이 데이터를 레코드별로 또는 유연한 시간 기간에 따라 연속하여 점차적으로 처리한 후, 처리된 데이터를 통해 상관관계, 취합, 필터링, 샘플링을 비롯하여 다양한 방법의 분석을 수행할 수 있습니다. 이러한 처리 유형을 실시간 처리라고 합니다. 실시간 처리를 통해 파생된 정보는 기업의 비즈니스와 고객 행동의 다양한 측면(예: 서비스 사용(사용량 계산 또는 요금 청구), 서버 동작, 웹 사이트 클릭 및 디바이스, 사람, 물리적 상품의 지리적 위치 등)에 대한 분석을 제공하여 새로운 상황에 신속하게 대응할 수 있게 해줍니다. 실시간 처리는 고도로 확장 가능한 동시 처리 계층이 필요합니다.

스트리밍 데이터를 실시간으로 처리하기 위해 AWS Lambda를 사용할 수 있습니다. Lambda는 AWS IoT 또는 Amazon Kinesis Streams에서 직접 데이터를 처리할 수 있습니다 Lambda는 서버를 프로비저닝하거나 관리하지 않고도 코드를 실행할 수 있습니다.

Amazon KCL(Kinesis Client Library)은 Amazon Kinesis Streams의 데이터를 처리하는 또 다른 방법입니다. KCL은 AWS Lambda보다 유연성이 좋아서 추가 처리가 필요한 유입 데이터를 일괄 처리할 수 있습니다. 또한 KCL을 사용하여 처리 로직에 광범위한 변형과 사용자 지정을 적용할 수도 있습니다.

Amazon Kinesis Firehose는 스트리밍 데이터를 AWS에 로드하는 가장 간편한 방법입니다. 스트리밍 데이터를 캡처하여 자동으로 Amazon Redshift에 로드함으로써 현재 사용 중인 기존의 비즈니스 인텔리전스(BI) 도구와 대시보드에서 실시간에 가까운 분석이 가능합니다. Firehose를 사용하여 배치(batch) 처리 규칙을 정의하면 이 규칙에 따라 확실하게 데이터가 배치 처리되어 Amazon Redshift에 로드됩니다.

데이터 스토리지

다음에서 설명한 대로 데이터를 데이터 웨어하우스나 데이터 마트에 저장할 수 있습니다.

데이터 웨어하우스

앞에서도 말했듯이 *데이터 웨어하우스*는 하나 또는 여러 데이터 소스로부터의 정보를 보관하는 중앙 저장소입니다. 데이터 웨어하우스를 사용하면 대량의 데이터에 대해 빠른 분석을 수행한 후 BI 도구를 활용함으로써 데이터에 숨겨져 있는 패턴을 찾아낼 수 있습니다. 데이터 과학자는 데이터 웨어하우스에 쿼리를 실행하여 오프라인 분석을 수행하고 추이를 발견합니다. 조직의 사용자들은 특별 SQL 쿼리, 정기 보고서, 대시보드를 통해 데이터를 사용하여 중요한 비즈니스 의사결정을 수행합니다.

데이터 마트

*데이터 마트*는 간단한 형태의 데이터 웨어하우스로, 특정 기능 영역이나 주제에 중점을 둡니다. 예를 들어 조직의 각 부서별로 데이터 마트를 두거나, 지역에 따라 여러 데이터 마트를 둘 수 있습니다. 대규모의 데이터 웨어하우스나 운영 스토어로부터 또는 이 둘 모두로부터 데이터 마트를 구축할 수 있습니다. 데이터 마트는 설계, 구축 및 관리가 간편합니다. 하지만 데이터 마트는 특정 기능 영역에 중점을 두기 때문에 기능 영역 전체에 걸쳐 쿼리를 실행하려면 배포 문제로 인해 복잡해질 수 있습니다.

Amazon Redshift를 사용하여 데이터 웨어하우스뿐 아니라 데이터 마트를 구축할 수 있습니다.

분석 및 시각화

데이터를 처리하여 분석할 수 있도록 만든 다음에는 그러한 데이터를 분석하고 처리하기 위한 적절한 도구가 필요합니다.

대부분의 경우 데이터 처리에 사용한 도구를 사용하여 데이터 분석을 수행할 수 있습니다. SQL Workbench와 같은 도구를 사용하여 Amazon Redshift에서 ANSI SQL로 데이터를 분석할 수 있습니다. 또한 Amazon Redshift는 시중에서 널리 사용되는 타사의 BI 솔루션과 잘 연동됩니다.

Amazon QuickSight는 클라우드 기반의 신속한 BI 서비스로서, 간편하게 시각화를 생성하여 특별 분석을 수행하고 데이터로부터 신속하게 비즈니스 통찰력을 확보할 수 있게 해줍니다. Amazon QuickSight는 Amazon Redshift와 통합되어 현재 프리뷰 버전이 제공되며, 2016년 말에 정식 출시될 예정입니다.

Amazon S3를 기본 스토리지로 사용할 경우 분석 및 시각화를 수행하는 대표적인 방법은 Amazon EMR(Amazon Elastic MapReduce)에서 Apache Spark 노트북을 실행하는 것입니다. 이 프로세스를 사용하면 SQL을 실행하거나, Python 및 Scala와 같은 언어로 작성된 사용자 지정 코드를 실행할 수 있습니다.

또 다른 시각화 방법은 오픈 소스 BI 솔루션인 Apache Zeppelin을 사용하는 것입니다. Amazon EMR에서 Apache Zeppelin을 실행하여 Spark SQL로 Amazon S3의 데이터를 시각화할 수 있습니다. Apache Zeppelin을 사용하여 Amazon Redshift의 데이터를 시각화할 수도 있습니다.

AWS 서비스를 사용한 분석 파이프라인

AWS는 완벽한 분석 플랫폼을 구축하기 위한 광범위한 서비스를 제공합니다. 그림 2에는 앞에서 설명한 서비스들이 나와 있으며 이러한 서비스들이 분석 파이프라인에서 어디에 적합한지를 보여 줍니다.

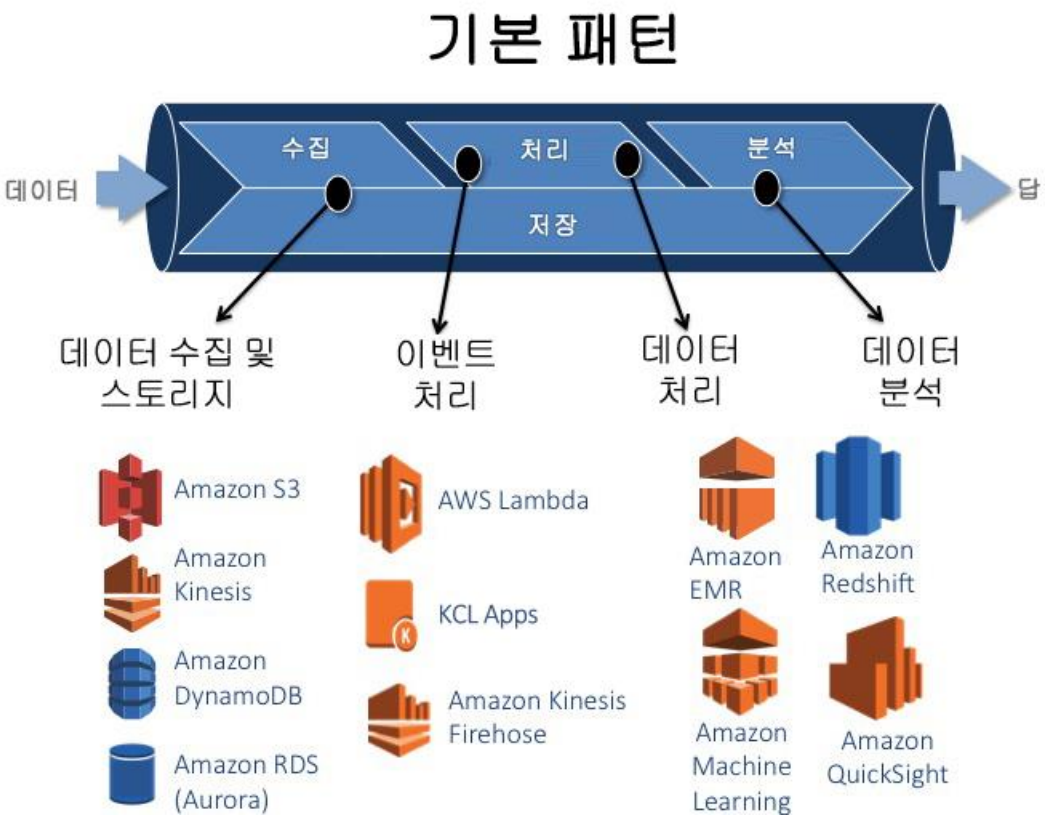

그림 2: AWS 서비스를 사용한 분석 파이프라인

데이터 웨어하우스 기술 옵션

이 단원에서는 데이터 웨어하우스, 즉 행 기반 데이터베이스, 열 기반 데이터베이스, 대량 병렬 처리(MPP) 아키텍처를 구축하는 데 사용할 수 있는 옵션에 대해 설명합니다.

행 기반 데이터베이스

행 기반 데이터베이스에는 일반적으로 물리적 블록의 전체 행이 저장됩니다. 읽기 작업을 위한 고성능은 보조 인덱스를 통해 실현됩니다. Oracle Database Server, Microsoft SQL Server, MySQL, PostgreSQL과 같은 데이터베이스는 행 기반 데이터베이스 시스템입니다. 이러한 시스템은 기존에 데이터 웨어하우징에 사용되어 왔지만, 분석용보다는 트랜잭션 처리(OLTP)에 더 적합합니다.

데이터 웨어하우스로 사용되는 행 기반 시스템의 성능을 최적화하기 위해 개발자들은 구체화된 뷰 만들기, 사전에 집계된 롤업 테이블 생성, 가능한 모든 조건부 조합에 대한 인덱스 생성, 데이터 파티셔닝을 구현하여 쿼리 최적화 프로그램을 통해 파티션 잘라내기 활용, 인덱스 기반 조인 수행 등과 같은 다양한 기법을 사용합니다.

기존의 행 기반 데이터 스토어는 한 대의 머신에서 사용할 수 있는 리소스에 따라 제한을 받습니다. 데이터 마트는 기능적 샤딩을 사용하여 이 문제를 어느 정도 완화합니다. 데이터 웨어하우스를 각각 특정 기능 영역에 부합되는 여러 개의 데이터 마트로 분할할 수 있습니다. 하지만 시간이 지날수록 데이터 마트가 커지면 데이터 처리 속도가 느려집니다.

행 기반 데이터 웨어하우스에서 모든 쿼리는 선택하지 않은 열을 포함하여 쿼리 조건부를 충족하는 블록의 모든 행에 대한 모든 열을 읽어야 합니다. 이 방식은 테이블이 열이 많을 경우 데이터 웨어하우스에 상당한 성능 병목을 발생시키며, 실제로 쿼리에서 사용하는 열은 몇 개 되지 않습니다.

열 기반 데이터베이스

열 기반 데이터베이스는 전체 행을 블록에 넣는 것이 아니라 각 열을 자체적인 물리적 블록 집합에 구성합니다. 따라서 열 기반 데이터베이스는 디스크에서 쿼리가 액세스한 열만 읽으면 되기 때문에 읽기 전용 쿼리에 대한 I/O 효율이 높습니다. 이로 인해 데이터 웨어하우징에는 행 기반 데이터베이스보다 열 기반 데이터베이스를 선택하는 것이 더 좋습니다.

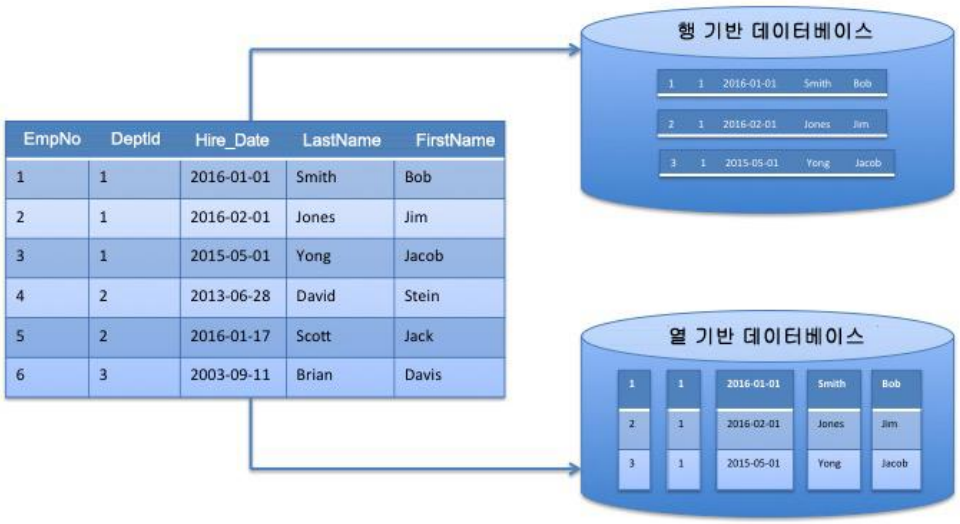

그림 3: 행 기반 데이터베이스와 열 기반 데이터베이스

위의 그림 3은 행 기반 데이터베이스와 열 기반 데이터베이스 간의 주요 차이점을 보여 줍니다. 행 기반 데이터베이스에서는 행이 자체 블록에 들어 있고, 열 기반 데이터베이스에서는 열이 자체 블록에 들어 있습니다.

I/O 속도의 향상 다음으로 열 기반 데이터베이스를 사용함에 따른 장점은 압축률의 향상입니다. 모든 열이 자체 블록 집합에 들어 있기 때문에 모든 물리적 블록은 동일한 데이터 형식을 포함합니다. 모든 데이터의 데이터 형식이 동일하면 데이터베이스에서 고도로 효율적인 압축 알고리즘을 사용할 수 있습니다. 따라서 행 기반 데이터베이스보다 스토리지가 적게 필요합니다. 또한 동일한 데이터가 더 적은 수의 블록에 저장되기 때문에 I/O도 상당히 감소합니다.

데이터 웨어하우징에 사용되는 열 기반 데이터베이스에는 Amazon Redshift, Vertica, Teradata Aster, Druid 등이 있습니다.

병렬 처리(MPP) 아키텍처

병렬 처리(MPP) 아키텍처에서는 클라우드에서 사용할 수 있는 모든 리소스를 데이터 처리에 사용할 수 있으므로 페타바이트 규모의 데이터 웨어하우스의 성능이 놀라울 만큼 향상됩니다. 병렬 처리(MPP) 데이터 웨어하우스는 클러스터에 단지 노드를 추가함으로써 성능을 향상시킬 수 있습니다. Amazon Redshift, Druid, Vertica, GreenPlum, Teradata Aster는 MPP 아키텍처를 기반으로 하는 데이터 웨어하우스입니다. 하둡이나 Spark와 같은 오픈 소스 프레임워크도 MPP를 지원합니다.

Amazon Redshift 심층 분석

컬럼 방식 MPP 기술을 사용하는 Amazon Redshift는 성능 기준에 맞는 경제적인 데이터 웨어하우징을 위해 효율적인 압축, I/O 감소, 스토리지 요구 사항 감소 등과 같은 중요한 이점을 제공합니다. 또한 ANSI SQL을 기반으로 하기 때문에 기존 쿼리를 거의 수정하지 않고 실행할 수 있습니다. 이러한 이유로 인해 Amazon Redshift는 오늘날 엔터프라이즈 데이터 웨어하우스 및 데이터 마트에 대표적으로 사용되는 제품이 되었습니다. 이 단원에서는 Amazon Redshift를 심층 분석하고 그 기능에 대해 자세히 알아보겠습니다.

Amazon Redshift는 컬럼 방식의 스토리지를 사용하고, 여러 노드에 대해 쿼리를 병렬로 실행하고 분산함으로써 데이터 크기에 상관없이 쿼리 및 I/O 성능이 우수합니다. 또한 데이터 웨어하우스에 대한 프로비저닝, 구성, 모니터링, 백업 및 보안과 관련된 일반적인 관리 작업을 대부분 자동화하므로 관리가 간편하면서도 비용이 적게 듭니다. 이러한 자동화는 기존의 온프레미스 구축 시 몇 주에서 몇 달까지 걸리던 페타바이트 규모의 데이터 웨어하우스를 단 몇 분만에 구축할 수 있게 해줍니다.

성능

Amazon Redshift는 컬럼 방식 스토리지와 데이터 압축 및 영역 지도(zone map)를 사용하여 쿼리를 실행하는 데 필요한 I/O 수를 줄입니다. 인터리브 정렬 기능은 인덱스나 프로젝션 관리에 따른 부담 없이 성능을 향상시킵니다.

Amazon Redshift는 대량 병렬 처리(MPP) 아키텍처를 사용하여 SQL 작업을 병렬 처리하고 분산함으로써 사용 가능한 리소스를 모두 활용합니다. 기반 하드웨어는 CPU와 드라이브 간의 처리량을 최대화하는 로컬 연결 스토리지와, 노드 간의 처리량을 최대화하는 10GigE 메시 네트워크를 사용하여 고성능 데이터 처리에 맞게 설계되었습니다. 성능은 데이터 웨어하우징 요구 사항에 따라 조정할 수 있습니다. AWS는 SSD(Solid State Drive) 기반의 DC(Dense Compute)를 제공하며 DS(Dense Storage) 옵션도 제공합니다. 소프트웨어 업그레이드를 지속적으로 배포하여 사용자의 개입 없이 성능 향상을 계속 이뤄 냅니다.

지속성과 가용성

데이터의 지속성과 가용성을 최대화하기 위해 Amazon Redshift는 데이터 웨어하우스 클러스터에서 장애가 발생한 노드를 자동으로 검출하여 교체합니다. 대체 노드는 바로 사용할 수 있으며, 가장 자주 액세스하는 데이터가 제일 먼저 로드되기 때문에 최대한 빠르게 데이터에 대한 쿼리를 재개할 수 있습니다. Amazon Redshift는 클러스터 전체에서 데이터를 미러링하기 때문에 다른 노드의 데이터를 사용하여 장애가 발생한 노드를 재구축합니다. 클러스터는 대체 노드가 프로비저닝되어 클러스터에 추가될 때까지 읽기 전용 모드이며, 일반적으로 이러한 재구축 작업은 단 몇 분밖에 걸리지 않습니다.

Amazon Redshift 클러스터는 한 개의 가용 영역에 들어 있습니다.[3] 하지만 Amazon Redshift에 대해 다중 가용 영역(Multi-AZ)을 설정하려면 미러를 생성한 후 복제와 장애 조치를 스스로 관리하면 됩니다.

Amazon Redshift Management Console에서 간단한 작업을 통해 강력한 재해 복구(DR) 환경을 구축할 수 있습니다. 여러 AWS 리전에 백업 복사본을 둘 수 있습니다. 한 AWS 리전에서 서비스 중단이 발생하면 다른 AWS 리전의 백업에서 클러스터를 복구할 수 있습니다. 복구 작업이 시작되면 단 몇 분 내에 클러스터에 대해 읽기/쓰기 액세스 권한을 얻을 수 있습니다.

확장성과 탄력성

성능이나 용량에 대한 요구 사항에 따라 콘솔에서 간단한 작업을 수행하거나, API 호출을 사용하여 데이터 웨어하우스의 노드 수와 유형을 손쉽게 변경할 수 있습니다.[4] Amazon Redshift를 사용하면 최소 한 개의 160GB 노드로 시작한 후, 많은 노드를 사용하는 1페타바이트 이상의 압축된 사용자 데이터까지 확장할 수 있습니다. 자세한 내용은 *Amazon Redshift Cluster Management Guide*에서 About Clusters and Nodes를 참조하십시오.[5]

규모 조정 과정에서 Amazon Redshift는 기존의 클러스터를 읽기 전용 모드로 설정하고 선택한 크기의 새 클러스터를 프로비저닝한 후 구 클러스터의 데이터를 새 클러스터로 동시 복사합니다. 이 경우 활성 Amazon Redshift 클러스터에 대해서만 요금이 청구됩니다. 새 클러스터가 프로비저닝되는 동안에는 구 클러스터에서 계속 쿼리를 실행할 수 있습니다. 데이터가 새 클러스터로 모두 복사되면, Amazon Redshift는 자동으로 쿼리를 새 클러스터로 리디렉션하고 구 클러스터를 제거합니다.

Amazon Redshift API 동작을 사용하여 클러스터 시작, 클러스터 확장, 백업 생성, 백업 복원 등과 같은 작업을 프로그래밍 방식으로 수행할 수 있습니다. 이 방법을 통해 이러한 API 동작을 기존 자동화 스택에 통합하거나, 요구 사항에 맞는 자동화를 사용자 지정하여 구축할 수 있습니다.

인터페이스

Amazon Redshift에는 사용자 지정 JDBC(Java Database Connectivity) 드라이버와 ODBC(Open Database Connectivity) 드라이버가 있으며, 이러한 드라이버는 콘솔의 **Connect Client** 탭에서 다운로드할 수 있습니다. 이처럼 널리 사용되는 다양한 SQL 클라이언트를 사용할 수 있습니다. 또한 표준 PostgreSQL JDBC 드라이버와 ODBC 드라이버를 사용할 수도 있습니다. Amazon Redshift 드라이버에 대한 자세한 내용은 *Amazon Redshift Database Developer Guide*에서 Amazon Redshift and PostgreSQL을 참조하십시오.[6]

대표적인 BI 및 ETL 공급업체와의 검증된 통합에 대한 많은 사례도 찾아볼 수 있습니다.[7] 이러한 통합에서 로드(load) 및 언로드(unload)는 각 컴퓨팅 노드에서 동시에 실행되어 Amazon S3, Amazon EMR, Amazon DynamoDB 등과 같은 여러 리소스에 대해 데이터를 수집하거나 내보내는 속도를 최대한 향상시킵니다. Amazon Kinesis Firehose를 사용하여 Amazon Redshift로 스트리밍 데이터를 손쉽게 로드할 수 있으며, 기존의 BI 도구와 대시보드를 통해 거의 실시간으로 분석을 수행할 수 있습니다. 콘솔이나 Amazon CloudWatch API 연산을 사용하여 Amazon Redshift 데이터 웨어하우스 클러스터에 대해 컴퓨팅 사용률, 메모리 사용률, 스토리지 사용률, 읽기/쓰기 트래픽 측정치를 확인할 수 있습니다.

보안

데이터 보안을 제공하려면 Amazon VPC(Amazon Virtual Private Cloud) 서비스 기반의 가상 프라이빗 클라우드 내에서 Amazon Redshift를 실행하면 됩니다. VPC의 소프트웨어 정의 네트워킹 모델을 사용하면 구성한 규칙에 따라 트래픽을 제한하는 방화벽 규칙을 정의할 수 있습니다.[8] Amazon Redshift는 클라이언트 애플리케이션과 Amazon Redshift 데이터 웨어하우스 클러스터 간에 SSL 연결을 지원하므로 데이터를 전송할 때 암호화할 수 있습니다.

Amazon Redshift 컴퓨팅 노드에는 데이터가 저장되지만 이 데이터는 클러스터의 리더 노드에서만 액세스할 수 있습니다. 이러한 분리를 통해 보안이 한층 강화됩니다. Amazon Redshift는 AWS CloudTrail과 통합되어 모든 Amazon Redshift API 호출에 대해 감사를 수행할 수 있습니다.[9] 데이터 보안을 보장하기 위해 Amazon Redshift는 디스크에 각 블록이 기록될 때마다 하드웨어 가속 AES-256 암호화를 사용하여 각 블록을 암호화합니다. 이 암호화는 I/O 서브 시스템의 하위 수준에서 이루어지며, I/O 서브 시스템은 중간 쿼리 결과를 포함하여 디스크에 기록되는 모든 것을 암호화합니다. 블록은 그대로 백업되므로 백업 역시 암호화됩니다. 기본적으로 Amazon Redshift 키 관리를 수행하지만, 귀사에서 소유한 하드웨어 보안 모듈(HSM)을 사용하여 키를 관리하거나, AWS Key Management Service를 통해 키를 관리할 수도 있습니다.[10],[11]

비용 모델

Amazon Redshift는 장기 약정이나 선납 요금이 필요 없습니다. 따라서 자본 비용이 들지 않으며 데이터 웨어하우스 용량을 사전에 복잡하게 계획하고 구입할 필요도 없습니다. 요금은 클러스터의 노드 크기와 수에 따라 청구됩니다.

프로비저닝한 데이터베이스 스토리지의 100%까지 추가 비용 없이 백업 스토리지를 제공합니다. 예를 들어 총 4TB 스토리지에 대해 XL 노드가 2개인 활성 클러스터가 하나 있을 경우 AWS는 추가 비용 없이 Amazon S3에서 최대 4TB의 백업 스토리지를 제공합니다. 프로비저닝한 스토리지 크기를 초과하는 백업 스토리지와, 클러스터가 만료된 후에 저장된 백업에 대해서는 표준 Amazon S3 요율이 적용됩니다.[12] Amazon S3와 Amazon Redshift 간의 통신에 대해서는 데이터 전송 요금이 부과되지 않습니다. 자세한 내용은 Amazon Redshift 요금을 참조하십시오.[13]

적합한 사용 패턴

Amazon Redshift는 기존 BI 도구를 사용한 OLAP(Online Analytical Processing)에 적합합니다. 조직에서는 Amazon Redshift를 사용하여 다음과 같은 작업을 수행하고 있습니다.

- 엔터프라이즈 BI 및 보고 실행
- 여러 제품에 대한 전체 판매 데이터 분석
- 주식 거래 데이터 저장
- 광고 노출 수 및 클릭 횟수 분석
- 게임 데이터 집계
- 소셜 트렌드 분석
- 의료 분야의 임상적 품질, 작업 효율, 재무 성과 측정

비적합 패턴

Amazon Redshift는 다음 사용 패턴에는 적합하지 않습니다.

- **작은 데이터 세트** – Amazon Redshift는 클러스터 전체에 대해 병렬 처리를 수행하기 위한 솔루션입니다. 데이터 세트가 100기가바이트 미만일 경우 Amazon Redshift가 제공하는 모든 기능을 활용할 수 없으므로 Amazon RDS가 더 적합한 솔루션이 될 수 있습니다.
- **OLTP** – Amazon Redshift는 경제적이면서도 속도가 뛰어난 분석 기능을 제공해야 하는 데이터 웨어하우징 워크로드용으로 설계되었습니다. 속도가 빠른 트랜잭션 시스템이 필요할 경우 Amazon RDS 기반의 기존 관계형 데이터베이스 시스템이나, Amazon DynamoDB와 같은 NoSQL 데이터베이스를 선택할 수 있습니다.
- **비정형 데이터** – Amazon Redshift의 데이터는 정의된 스키마를 통해 정형화되어야 합니다. Amazon Redshift는 각 행에 대해 임의의 스키마 구조를 지원하지 않습니다. 데이터가 비정형화 상태일 경우, Amazon EMR에서 추출, 변형 및 로드(ETL)를 수행하여 Amazon Redshift에 로드할 수 있는 데이터를 얻을 수 있습니다. JSON 데이터의 경우 키 값 페어를 저장한 후 쿼리에 네이티브 JSON 함수를 사용할 수 있습니다.[14]

- **BLOB 데이터** – 디지털 비디오, 이미지, 음악 등과 같은 BLOB(Binary Large Object) 파일을 저장하려는 경우, 데이터를 Amazon S3에 저장한 후 Amazon Redshift에서 그 위치를 참조하는 방법을 고려해 볼 수 있습니다. 이 시나리오에서 Amazon Redshift는 이진 객체에 대한 메타데이터(항목 이름, 크기, 생성일, 소유자, 위치 등)를 기록하지만, 대형 객체 자체는 Amazon S3에 저장됩니다.

Amazon Redshift로 마이그레이션

기존 데이터 웨어하우스를 Amazon Redshift로 마이그레이션할 경우 다음과 같은 몇 가지 요소를 기준으로 마이그레이션 전략을 선택해야 합니다.

- 데이터베이스와 그 테이블의 크기
- 소스 서버와 AWS 간의 네트워크 대역폭
- AWS로의 마이그레이션 및 전환을 한 단계로 수행할지 또는 점차적인 일련의 단계로 수행할지 여부
- 소스 시스템의 데이터 변경률
- 마이그레이션 중의 변형
- 마이그레이션 및 ETL에 사용하려는 파트너 도구

한 단계 마이그레이션

한 단계 마이그레이션은 지속적인 작업이 필요하지 않은 작은 데이터베이스에 적합한 선택입니다. 기존 데이터베이스를 CSV(쉼표로 구분된 값) 파일로 추출한 후 AWS Import/Export Snowball과 같은 서비스를 사용하여 Amazon S3에 데이터 세트를 전송하고 Amazon Redshift에 로드합니다. 그런 다음 대상 Amazon Redshift 데이터베이스에 대해 소스와의 데이터 일관성을 검사합니다. 모든 확인에서 이상이 없으면 데이터베이스가 AWS로 전환됩니다.

두 단계 마이그레이션

두 단계 마이그레이션은 일반적으로 모든 크기의 데이터베이스에 사용됩니다.

1. **최초 데이터 마이그레이션:** 소스 데이터베이스에서 데이터가 추출됩니다. 사용량이 적은 시간에 수행하여 영향을 최소화합니다. 추출된 데이터는 앞에서 설명한 한 단계 마이그레이션 방법에 따라 Amazon Redshift로 마이그레이션됩니다.
2. **변경된 데이터 마이그레이션:** 최초 데이터 마이그레이션 후에 소스 데이터베이스에서 변경된 데이터는 전환 전에 대상으로 전파됩니다. 이 단계에서 소스와 대상 데이터베이스를 동기화합니다. 변경된 데이터가 모두 마이그레이션되면, 대상 데이터베이스의 데이터에 대해 유효성을 검사하고 필요한 테스트를 수행하여 모든 테스트를 통과하면 Amazon Redshift 데이터 웨어하우스로 전환할 수 있습니다.

데이터베이스 마이그레이션용 도구

데이터 마이그레이션을 위해 다양한 도구와 기술을 사용할 수 있습니다. 이러한 도구의 몇 가지를 서로 대체하여 사용할 수 있으며, 시중에서 구할 수 있는 타사의 도구나 오픈 소스 도구를 사용할 수도 있습니다.

1. AWS Database Migration Service는 앞에서 설명한 한 단계 마이그레이션 프로세스와 두 단계 마이그레이션 프로세스를 모두 지원합니다.[15] 두 단계 마이그레이션 프로세스를 수행하려면 보충 로깅을 활성화하여 소스 시스템에 대한 변경을 캡처해야 합니다. 보충 로깅은 테이블 수준이나 데이터베이스 수준에서 활성화할 수 있습니다.
2. 그 외의 통합 파트너 도구에는 다음과 같은 것이 있습니다.
   - Attunity
   - Informatica
   - SnapLogic
   - Talend
   - Bryte

데이터 통합 및 컨설팅 파트너에 대한 자세한 정보는 Amazon Redshift 파트너를 참조하십시오.[16]

데이터 웨어하우징 워크플로 설계

앞 단원에서 데이터 웨어하우징에 적합한 Amazon Redshift의 기능을 살펴보았습니다. Amazon Redshift를 사용하여 데이터 웨어하우징 워크플로를 설계하는 방법을 이해하기 위해 이번에는 가장 일반적인 설계 패턴과 사용 사례를 알아보겠습니다.

매장이 천 곳이 넘고, 백화점과 할인 매장을 통해 특정 의류 상품을 판매하며, 온라인 매장을 가지고 있는 다국적 의류 기업을 예로 들어 보겠습니다. 기술적 관점에서 이러한 세 가지 채널은 현재 독립적으로 운영되며, 관리진, POS(Point-of-Sale) 시스템, 회계 부서도 각각 다릅니다. 관련 데이터 세트를 모두 취합하여 CEO에게 전체 사업에 대한 포괄적인 분석 자료를 제공하는 단일 시스템이 없습니다.

CEO는 이러한 채널에 대해 전사적인 관점에서 파악할 수 있는 자료를 원하며, 다음과 같은 특별 분석을 수행할 수 있기를 바랍니다.

- 전체 유통 채널의 추세는 어떠한가?
- 전체 채널에 걸쳐 어느 지역이 실적이 좋은가?
- 회사의 광고와 프로모션이 얼마나 효과가 있는가?
- 각 의류 상품의 추세는 어떠한가?
- 회사의 매출에 어떤 외부적인 요인이 영향을 주는가? (예: 실업률, 기후 조건 등)
- 매장의 특성이 판매에 어떤 영향을 주는가? (예: 직원 및 관리진의 근속 기간, 길가의 개방형 매장과 건물 내 매장의 비교, 매장 내 상품의 위치, 프로모션, 진열대 및 전시 공간, 판매 권유, 매장 내 디스플레이)

엔터프라이즈 데이터 웨어하우스는 이 문제의 답을 알려 줍니다. 엔터프라이즈 데이터 웨어하우스는 세 채널의 다양한 시스템 각각으로부터 그리고 날씨 및 경제 보고서와 같은 공개 자료로부터 데이터를 수집합니다. 각 데이터 소스는 데이터 웨어하우스에서 사용하는 데이터를 매일 전송합니다. 각 데이터 소스는 서로 다른 구조를 가지고 있을 수 있으므로 추출, 변형 및 로드(ETL) 프로세스를 수행하여 데이터를 공통된 구조로 다시 정형화합니다. 그런 다음 모든 소스의 데이터에 대해 동시에 분석을 수행할 수 있습니다. 이 프로세스를 위해 다음과 같은 데이터 흐름 아키텍처가 사용됩니다.

그림 4: 엔터프라이즈 데이터 웨어하우스 워크플로

1. 이 프로세스의 첫 단계는 여러 소스의 데이터를 Amazon S3로 모으는 것입니다. Amazon S3는 다양한 소스의 데이터를 매우 경제적인 비용으로 동시에 쓸 수 있으며, 지속성이 우수하고 경제적이면서도 확장 가능한 스토리지 플랫폼을 제공합니다.

2. Amazon EMR은 소스 형식의 데이터를 변형하고 정리하여 대상 형식으로 만드는 데 사용됩니다. Amazon EMR에는 Amazon S3이 통합되어 있어서 Amazon EMR 클러스터의 각 노드에서 Amazon S3와 주고받는 작업을 병렬 스레드로 동시에 처리합니다.

   일반적으로 데이터 웨어하우스에는 야간에 새 데이터를 받습니다. 한밤중에는 분석 기능이 필요하지 않기 때문에 이 변형 프로세스의 유일한 요구 사항은 CEO와 다른 업무 관계자들이 보고서와 대시보드를 액세스해야 하는 아침까지 완료하는 것입니다. 따라서 이때 Amazon EC2 스팟 시장을 사용하여 ETL 비용을 더 낮출 수 있습니다.[17] 좋은 스팟 전략은 자정에 매우 낮은 가격으로 입찰을 시작한 후 용량이 부여될 때까지 점차적으로 가격을 올리는 것입니다. 마감 시간이 다가오는데 스팟 입찰이 아직 낙찰되지 않았을 경우 온디맨드 가격으로 후퇴하여 완료 시간 요건을 맞출 수 있습니다. 각 소스는 Amazon EMR에서 서로 다른 변형 프로세스를 가질 수 있지만, AWS 선불형 종량 요금제를 사용하면 각 변형에 대해 별도의 Amazon EMR 클러스터를 생성한 후 정확한 용량이 되도록 조정하여 다른 작업의 리소스와 경합 없이 모든 데이터 변형 작업을 완료할 수 있습니다.

3. 각 변형 작업은 형식을 변환하여 정리한 데이터를 Amazon S3에 로드합니다. Amazon S3를 여기서 다시 사용하는 이유는 Amazon Redshift가 각 클러스터 노드의 여러 스레드를 사용하여 Amazon S3로부터 동시에 데이터를 로드할 수 있기 때문입니다. 또한 Amazon S3는 기록 레코드를 제공하고, 시스템 간에 형식이 지정된 SOT(Source of Truth) 역할을 합니다. 시간이 지남에 따라 추가적인 요구 사항이 발생할 경우 Amazon S3의 데이터를 다른 분석용 도구에서 사용할 수 있습니다.
4. Amazon Redshift는 데이터를 테이블에 로드하고, 정렬 및 분산, 압축하여 분석 쿼리가 효율적으로 동시에 실행될 수 있도록 합니다. 시간이 지남에 따라 데이터 크기가 커지고 비즈니스가 확장되면 노드를 추가하여 손쉽게 용량을 늘릴 수 있습니다.
5. 분석을 시각화하려면 Amazon QuickSight 또는 ODBC나 JDBC를 사용하여 Amazon Redshift에 연결하는 파트너 사의 다양한 시각화 플랫폼 중 하나를 사용하면 됩니다. 이 시점에서 CEO와 경영진은 보고서, 대시보드, 차트를 봅니다. 이제 경영진은 데이터를 통해 회사 리소스에 대해 보다 나은 의사결정을 할 수 있으며, 더 나아가서 수익과 주주 가치를 향상시킬 수 있습니다.

이 아키텍처는 유연하므로 사업이 확장되고, 신규 채널이 개설되고, 고객용 모바일 앱이 출시되며, 데이터 소스가 늘어날 경우 손쉽게 확장할 수 있습니다. 이러한 확장 작업은 Amazon Redshift Management Console에서 몇 번의 클릭으로 수행하거나, 몇 개의 API 호출을 사용하여 수행할 수 있습니다.

결론

기업에서 분석 데이터베이스와 솔루션을 온프레미스 솔루션에서 클라우드로 마이그레이션하여 클라우드의 간편함과 성능 및 경제성을 활용함에 따라 AWS는 데이터 웨어하우징의 전략적 변화를 예상하고 있습니다. 본 백서에서는 AWS 기반의 데이터 웨어하우징의 현재 상태를 포괄적으로 다룹니다. AWS는 엔터프라이즈 데이터 웨어하우징을 클라우드에 손쉽게 구축하고 실행할 수 있는 다양한 서비스와 강력한 파트너 에코시스템을 제공합니다. 따라서 AWS의 글로벌 인프라를 기반으로 귀사의 비즈니스를 확장할 수 있는 경제적이고도 성능이 뛰어난 분석 아키텍처를 구축할 수 있습니다.

기고자

다음은 이 문서의 작성에 도움을 준 개인 및 조직입니다.

- Babu Elumalai, Amazon Web Services의 솔루션 아키텍트
- Greg Khairallah, Amazon Web Services의 수석 BDM
- Pavan Pothukuchi, Amazon Web Services의 수석 제품 관리자
- Jim Gutenkauf, Amazon Web Services의 시니어 테크니컬 라이터
- Melanie Henry, Amazon Web Services의 시니어 테크니컬 에디터
- Chander Matrubhutam, Amazon Web Services 제품 마케팅

참고 문헌

자세한 내용은 다음을 참조하십시오.

- Apache Hadoop 소프트웨어 라이브러리[18]
- Amazon Redshift 모범 사례[19]
- Lambda 아키텍처[20]

참고

1 https://www.forrester.com/report/The+Forrester+Wave+Enterprise+Data+Warehouse+Q4+2015/-/E-RES124041

2 http://aws.amazon.com/streaming-data/

3 http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html

4 http://docs.aws.amazon.com/redshift/latest/APIReference/Welcome.html

5 http://docs.aws.amazon.com/redshift/latest/mgmt/working-with-clusters.html#rs-about-clusters-and-nodes

6 http://docs.aws.amazon.com/redshift/latest/dg/c_redshift-and-postgres-sql.html

7 http://aws.amazon.com/redshift/partners/

8 https://aws.amazon.com/vpc/

9 https://aws.amazon.com/cloudtrail/

10 http://docs.aws.amazon.com/redshift/latest/mgmt/working-with-HSM.html

11 https://aws.amazon.com/kms/

12 http://aws.amazon.com/s3/pricing/

13 http://aws.amazon.com/redshift/pricing/

14 http://docs.aws.amazon.com/redshift/latest/dg/json-functions.html

15 https://aws.amazon.com/dms/

16 https://aws.amazon.com/redshift/partners/

17 http://aws.amazon.com/ec2/spot/

18 https://hadoop.apache.org/

19 http://docs.aws.amazon.com/redshift/latest/dg/best-practices.html

20 https://en.wikipedia.org/wiki/Lambda_architecture