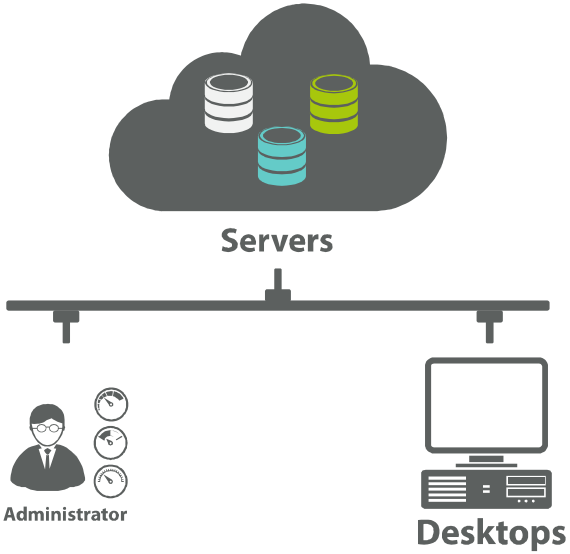
클라우드 기반 데이터레이크 및 분석

차세대 빅데이터 플랫폼 DataLake

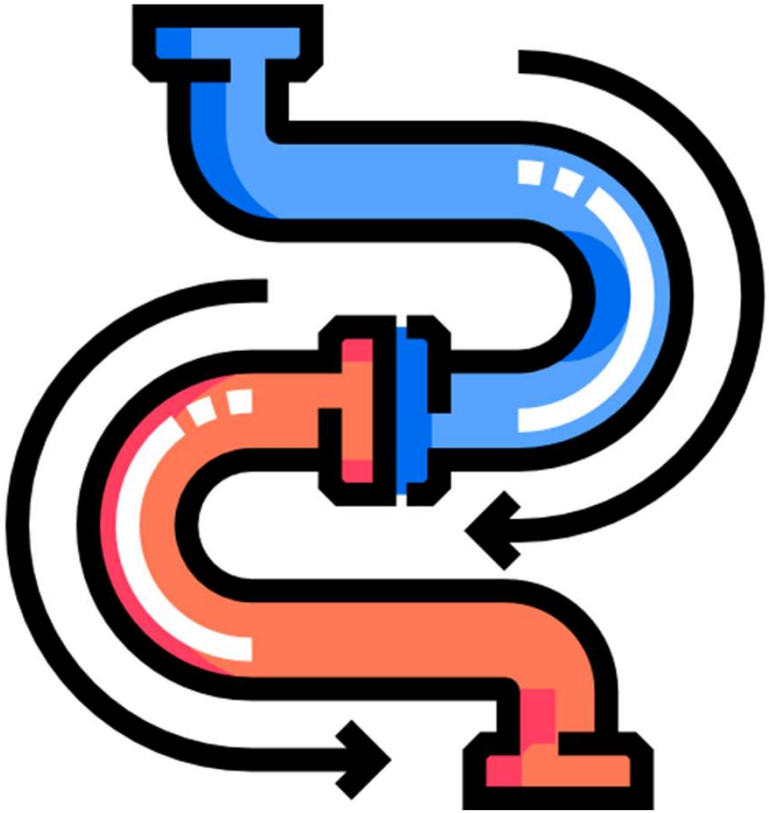
MEGAZONE CLOUD



Index

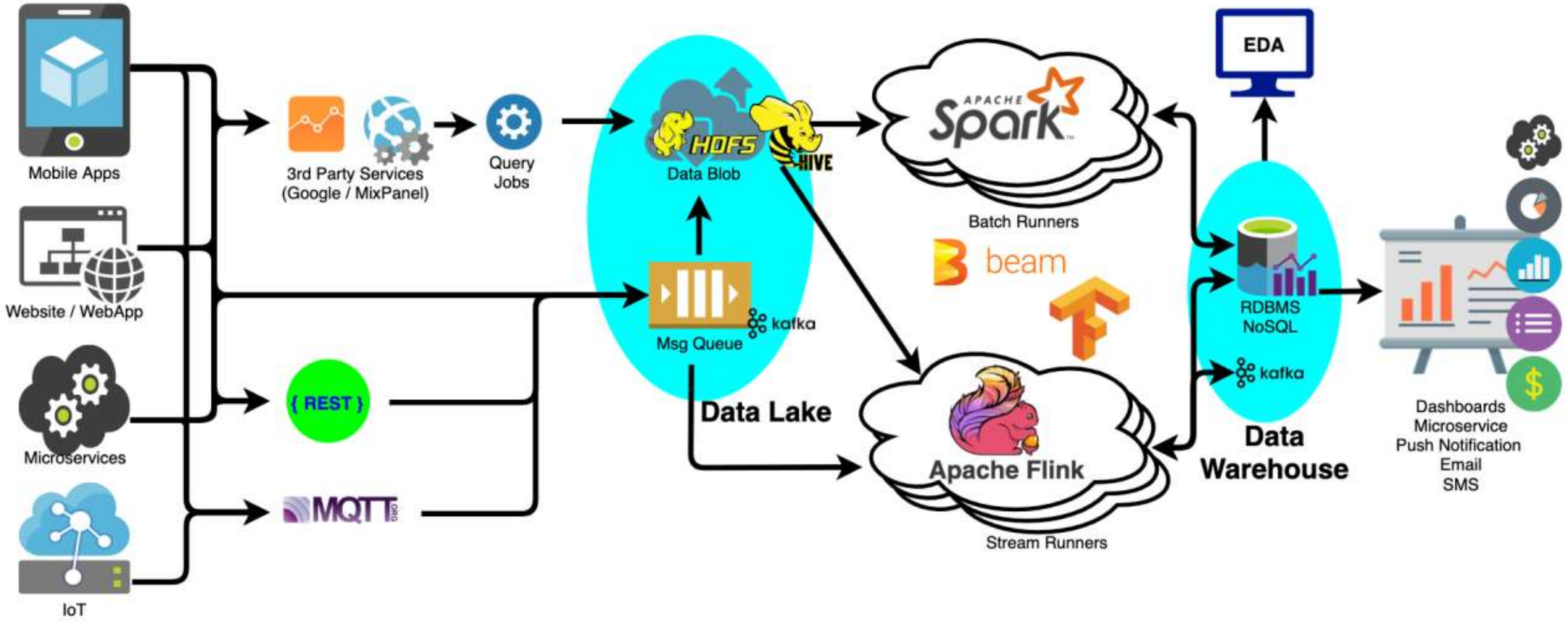
Data Pipeline

Data Pipeline



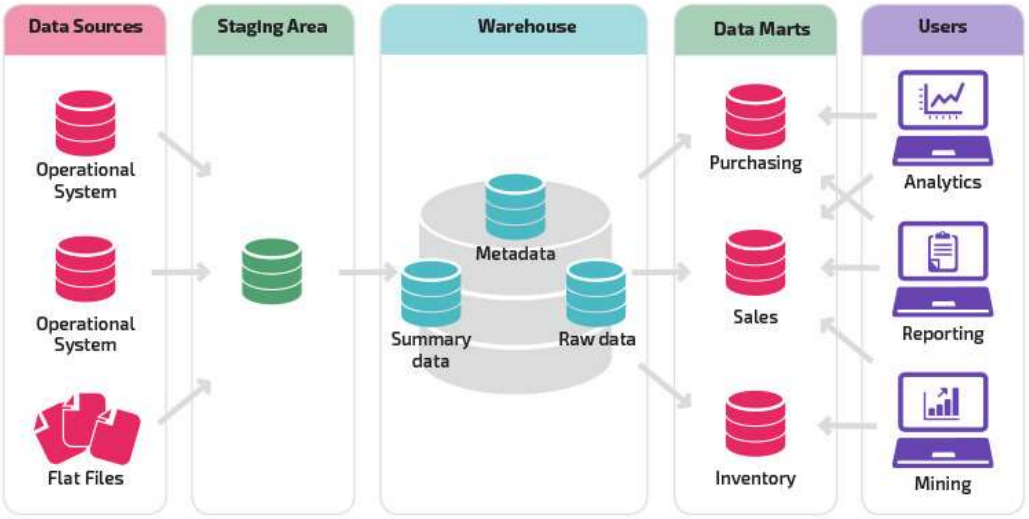
- 수집
- 처리
- 저장
- 분석
- 활용
- 거버넌스 & 모니터링

Data Pipeline (Cont.)



https://www.linkedin.com/pulse/data-pipeline-enterprise-strategy-rama-iyer

Data Pipeline (Cont.)

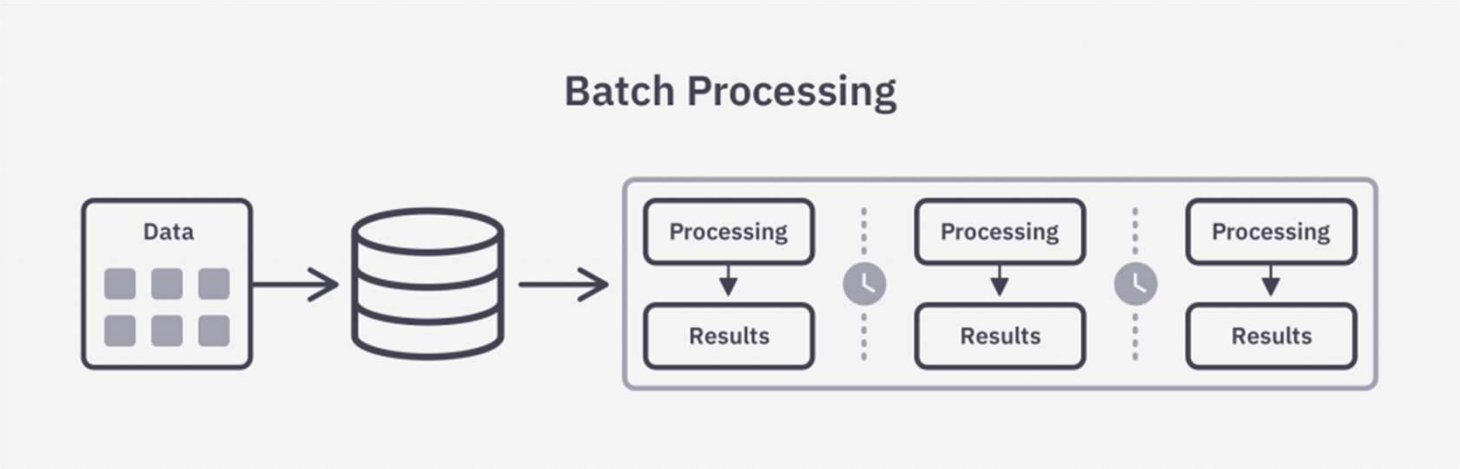


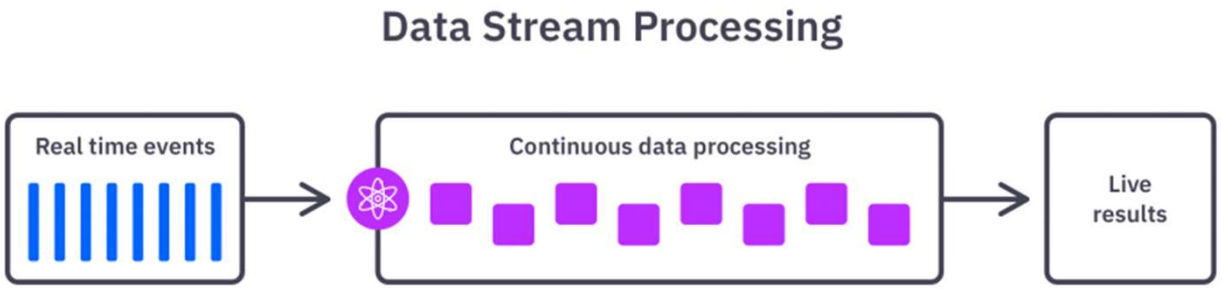
- Data Warehouse
  - Data를 정제해서 모아 두는 곳
- Data Lake
  - 여러가지 Data가 모여 있는 곳
- Data Mart
  - 목적별 Data

https://www.linkedin.com/pulse/data-pipeline-enterprise-strategy-rama-iyer

Data Processing

Data Processing

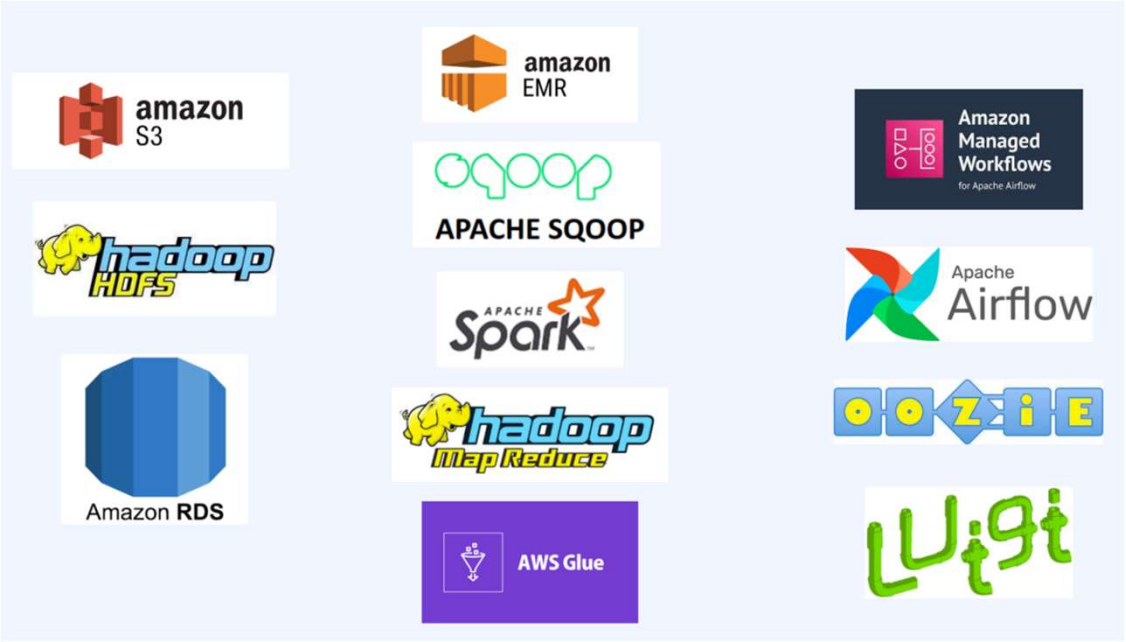




https://velog.io/@roo333/%EB%B0%B0%EC%B9%98-%ED%94%84%EB%A1%9C%EC%84%B8%EC%8B%B1-VS-%EC%8A%A4%ED%8A%B8%EB%A6%BC-%ED%94%84%EB%A1%9C%EC%84%B8%EC%8B%B1

Data Processing – Batch Processing

- 일괄처리
- 미리 설정된 시간 간격 동안 저장소에 데이터의 묶음(Batch)을 로드
- 일반적으로 사용량이 적은 업무 시간에 예약
- 대용량 데이터에 대한 작업으로 전체 시스템에 부담을 줄 수 있으므로 일괄처리를 통해 부담 최소화가 목적
- 즉시 분석할 필요가 없는 데이터
- 순차적 명령들의 워크플로우 형성
- ETL



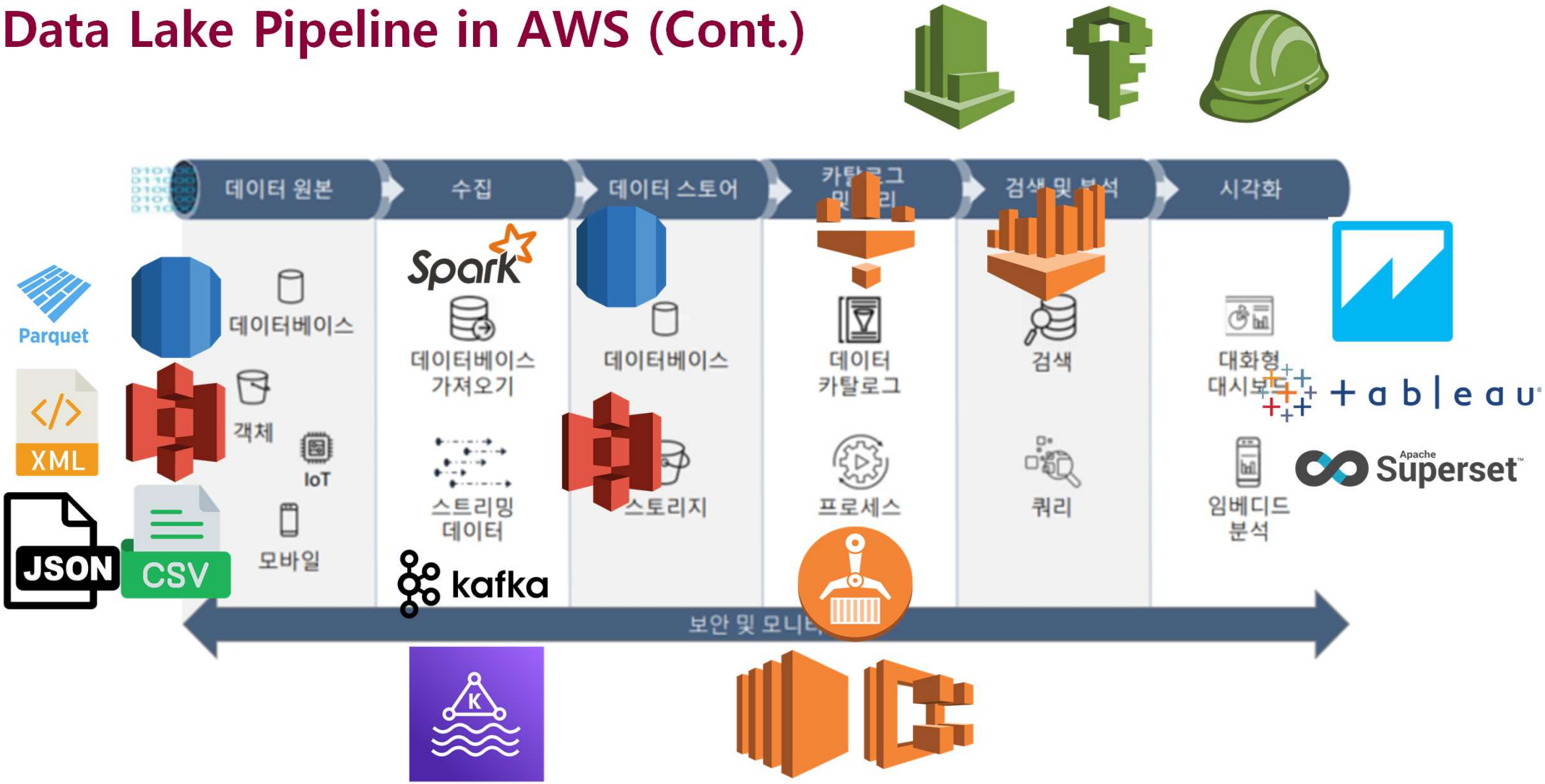
Data Processing – Data Stream Processing

- 실시간 데이터 처리
- 데이터를 지속적으로 업데이트해야 할 때
- 만일 App이나 POS 시스템은 제품의 재고와 판매 내용의 업데이트가 필요
- 제품 판매와 같은 단일 작업을 Event로 간주
- 결제에 항목 추가는 토픽 or 스트림으로 그룹화됨.
- Data Event는 발생한 직후에 처리됨
- 일괄처리에 비해 안정적인 것으로 간주되지 않음.
- 메시지 브로커가 필요

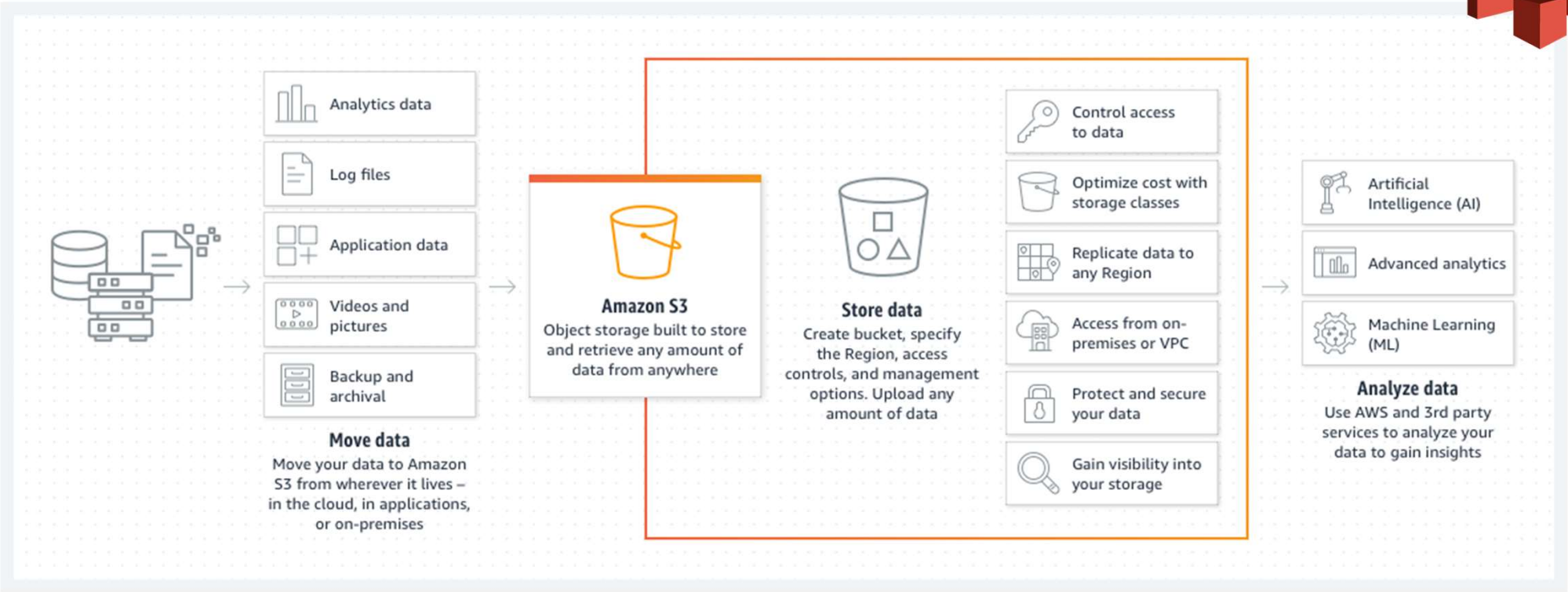
AWS DataLake Pipeline

Data Lake Pipeline in AWS

Data Lake Pipeline in AWS (Cont.)



AWS S3



- 어디서나 원하는 양의 데이터를 검색할 수 있도록 구축된 객체 스토리지

AWS S3 (Cont.) - 특징

- 추가적으로 시스템 관리 없이 많은 사용자가 데이터에 접근 가능
- 논리적으로 저장되는 파일 수에 제한이 없으며, 하나의 파일 크기는 최대 5TB까지 저장 가능
- 파일에 대한 접근 제한 기능 제공
- 데이터 손실을 걱정할 필요 없음.
- 버전관리를 통해 복원 가능
- 데이터의 중요도에 따라 스토리지 클래스별로 구분 저장 가능하면 이를 통해 비용 절감 가능

AWS S3 (Cont.) – 관련 용어

- 객체(Object)
  - S3에 저장되는 하나의 데이터(파일, 폴더 등)
- 버킷(Bucket)
  - 객체들을 보관하는 최상위 디렉토리
  - 버킷 단위로 다양한 보안 설정 가능하며, Region별로 생성
- 버전 관리
  - 객체의 여러 버전을 동일한 버킷에서 관리하기 위한 수단
  - 각 버전을 보존, 검색 및 복원 가능
- 스토리지 클래스
  - 데이터 액세스, 복원력 및 비용 요구 사항에 따라 선택할 수 있는 스토리지 클래스 제공
  - S3 Standard, S3 Standard-IA(Infrequent Access), S3 Glacier Flexible Retrieval)

Apache Spark

- https://spark.apache.org/
- Unified engine for large-scale data analytics.

Apache Spark (Cont.)

- 빅데이터 워크로드에 쓰이는 오픈 소스 고속 통합 분석 엔진
- 2009년 UC Berkeley에서 개발
- 데이터 처리 분야에서 가장 규모가 큰 오픈 소스 프로젝트
- Netflix, Yahoo, eBay와 같은 인터넷 대기업들이 대규모로 Spark를 사용하고, 8000개가 넘는 클러스터에서 PiB 규모의 데이터를 처리
- 현재 250개 이상의 조직에서 1000명 이상이 Contributor로 활동
- 인메모리 기반의 데이터 처리로 Hadoop MapReduce 대비 100배(디스크 사용시 10배) 빠름
- DAG 스케쥴러, 쿼리 최적화 도구, 물리적 실행 엔진을 사용하여 고성능을 제공
- 다양한 데이터 저장소와 생태계가 잘 구축되어 있음.
- 배치 및 실시간 처리 뿐만 아니라 머신 러닝 빌드 애플리케이션 지원(MLlib), 그래프 철(GraphX) 지원

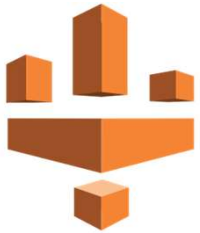
Apache Spark (Cont.)

- Spark 장점
  - 속도
    - 여러 개의 병렬 작업에 걸쳐 데이터를 메모리에 캐시하여 빠른 실행 속도를 자랑,
    - Hadoop MapReduce 대비 최고 100배 빠르고 디스크에서 처리시 10배 빠름.
  - 실시간 스트림 처리
    - 실시간 스트리밍을 처리하기도 하고, 다른 프레임워크와 통합 가능
  - 통합 엔진(여러 워크로드 지원)
    - SQL 쿼리, 스트리밍 데이터, 머신러닝과 그래프 처리 지원 및 높은 수준의 라이브러리 패키지 제공을 통해 생산성 향상 및 복잡한 워크플로 구현
  - 사용 편리성 증가
    - Java, Scala, Python, R 여러가지 프로그래밍 언어를 지원, 데이터 변환을 위한 100개 이상의 연산자 컬렉션과 반구조화된 데이터 조작에 흔히 사용

AWS Glue

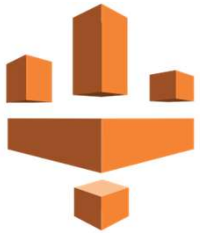
- 분석 사용자가 여러 소스의 데이터를 쉽게 검색, 준비, 이동, 통합할 수 있도록 하는 서버리스 데이터 통합 서비스
- 작성, 작업 실행, 비즈니스 워크플로 구현을 위한 추가 생산성 및 데이터 운영 도구 제공
- 70개 이상의 다양한 데이터 소스 연결 지원
- 중앙 집중식 데이터 카탈로그에서 데이터 관리
- ETL 파이프라인을 시각적으로 생성, 실행, 모니터링 가능
- DataLake에 데이터를 로드하거나 Athena, EMR, Redshift Spectrum을 사용하여 카탈로그화된 데이터를 즉시 검색하고 쿼리 가능
- ETL, ELT, Streaming과 같은 모든 워크로드를 하나의 서비스에서 유연하게 지원

AWS Glue (Cont.)



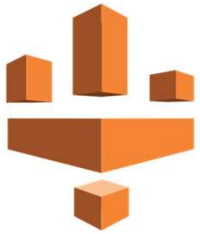
- AWS Glue 용어
  - AWS Glue Data Catalog
    - AWS Glue의 영구적 메타데이터 스토어.
    - 테이블 정의, 작업 정의 및 기타 관리 정보를 포함하여 AWS Glue 환경을 관리.
    - AWS 계정의 Region당 하나
  - Classifier
    - 데이터 스키마를 결정.
    - CSV, JSON, AVRO, XML 등과 같은 일반 파일 형식에 대한 분류자 뿐만 아니라 JDBC 연결을 사용한 일반 관계형 데이터베이스 관리 시스템을 위한 분류자를 제공
  - Connection
    - 특정 데이터 스토어에 연결하는 데 필요한 속성을 포함하는 Data Catalog 객체
  - Crawler
    - 데이터 스토어(소스 또는 대상)에 연결하는 프로그램
    - 분류자의 우선 순위 지정 목록을 통해 데이터의 스키마를 결정한 다음 AWS Glue Data Catalog에 메타데이터 테이블을 생성

AWS Glue (Cont.)



- Event-driven ETL
  - AWS Glue를 사용하면 새 데이터가 도착하는 대로 추출, 전환, 적재 작업을 실행할 수 있다.
  - 예를 들어 S3에서 새 데이터를 사용할 수 있게 되는 즉시 실행할 ETL 작업을 시작하도록 AWS Glue를 구성할 수 있다.
- Data Catalog
  - 데이터 카탈로그를 사용하면 데이터를 이동하지 않고도 여러 AWS 데이터 세트 전체에서 신속하게 데이터를 검색할 수 있다.
  - 일단 데이터가 카탈로그에 저장되면 Amazon Athena, Amazon EMR, Amazon Redshift Spectrum에서 즉시 검색 및 쿼리에 데이터를 사용할 수 있다.

AWS Glue (Cont.)

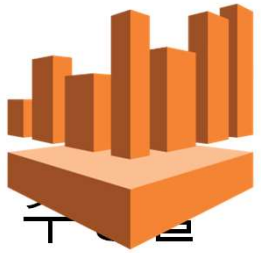


- No-code ETL jobs
  - AWS Glue Studio를 사용하면 AWS Glue ETL 작업을 시각적으로 간편하게 생성, 실행 및 모니터링할 수 있다.
  - Drag & Drop 방식의 편집기를 사용하여 데이터를 이동 및 변환하는 ETL 작업을 구축할 수 있으며, AWS Glue가 자동으로 코드를 생성한다.
- Data Preparation
  - AWS Glue DataBrew를 사용하면 Amazon S3, Amazon Redshift, AWS Lake Formation, Amazon Aurora 및 Amazon RDS를 비롯한 DataLake, Data Warehouse 및 Database에서 직접 데이터를 탐색하고 데이터로 실험할 수 있다.
  - DataBrew의 사전 구축된 250여 개의 변환 중에서 선택하여 이상 항목 필터링, 형식 표준화, 잘못된 값 수정 등의 데이터 준비 작업을 자동화할 수 있다.

Amazon Athena

- 표준 SQL를 사용하여 Amazon S3에 있는 데이터를 직접 간편하게 분석할 수 있는 대화형 쿼리 서비스
- AWS Management Console에서 몇 가지 작업을 수행하면 Athena에서 S3에 저장된 데이터를 저장하고 표준 SQL을 사용하여 Adhoc Query를 실행하여 몇 초 안에 결과를 얻을 수 있음.
- Athena는 Serverless 서비스이므로 설정하거나 관리할 인프라가 없음.
- 비용은 실행한 쿼리에 대해서만 과금됨
- Athena는 자동으로 확장되어 쿼리를 병렬로 실행하여 대규모 데이터 집합과 복잡한 쿼리에서도 빠르게 결과를 얻을 수 있음.
- 일반적으로 비정형, 반정형 및 정형 데이터를 분석하는 데 도움(ex. CSV, JSON, Parquet, ORC)
- 다양한 데이터 시각화 도구와 연결을 지원.

Amazon Athena - Features

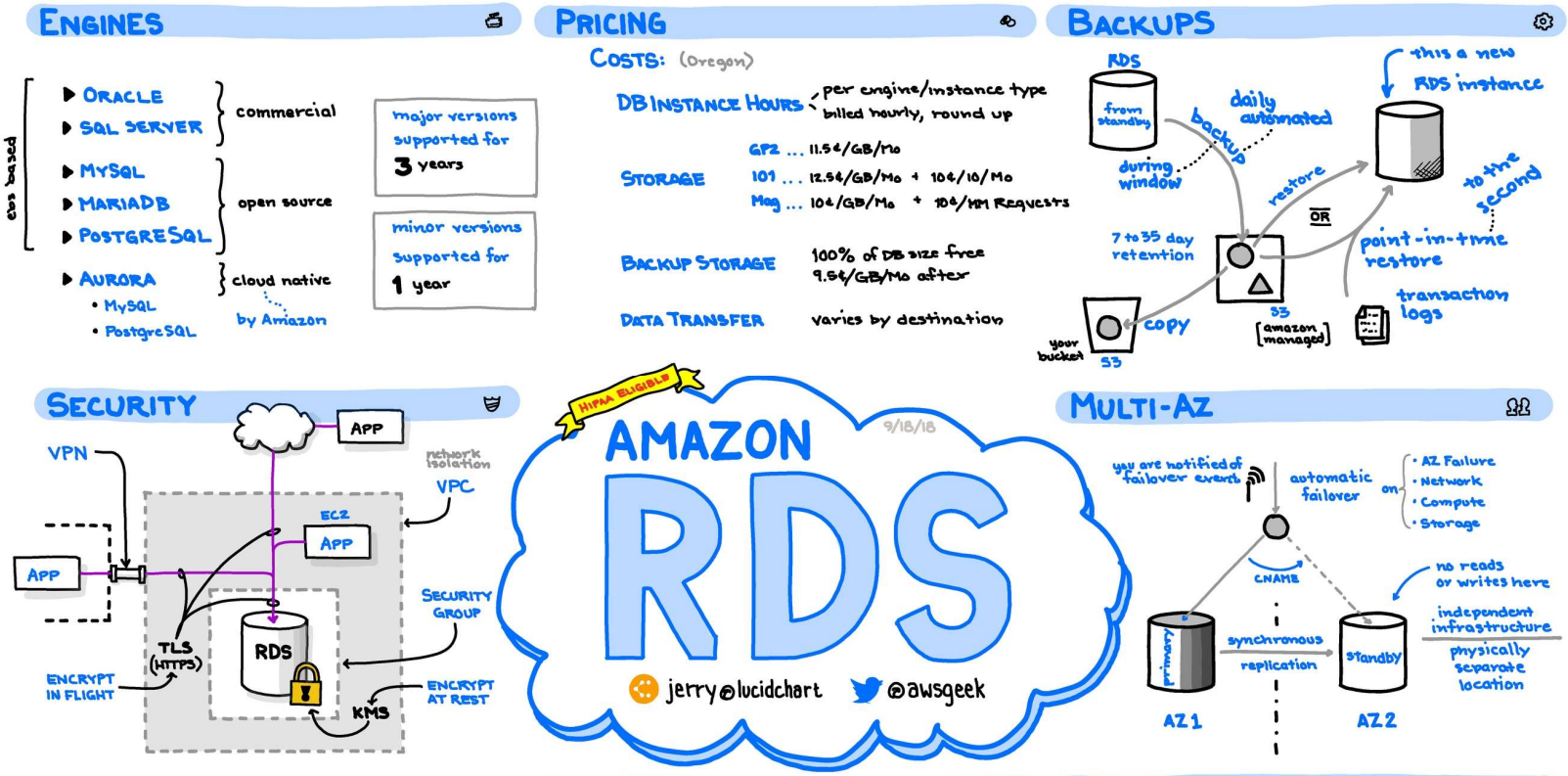


- Athena는 PiB 규모의 데이터에 대해 표준 SQL문에 기반한 질의를 수 [illegible] 수 있다.
- S3를 스토리지로 사용하기 때문에 99.999999999%에 달하는 S3의 내구성이 그대로 데이터에 적용
- 데이터 소스에 대응하는 테이블 메타 정보만 생성하면 바로 쿼리를 수행할 수 있으며, 쿼리 수행 속도 또한 매우 빠르다.
- S3에서 스캔하는 데이터 1TB당 5달러로 매우 저렴한 가격(매번 쿼리를 수행할 때 스캔하는 데이터의 양에 따라 과금되며, 미리 서버를 준비할 필요가 없어 고정 비용이 발생하지 않음)
- Presto, Hive 크게 두 가지의 오픈 소스 기술이 적용되어 있음.
  - Presto
    - In-Memory 분석 쿼리 엔진으로 ANSI-SQL 호환
  - Hive
    - DDL 관련 기능을 처리하는 것을 담당, 복잡한 데이터 타입, 여러 포맷, 데이터 파티셔닝, 테이블 생성 등

Amazon RDS

- AWS에서 관계형 데이터베이스를 더 쉽게 설치, 운영 및 확장할 수 있는 웹서비스
- 산업 표준 관계형 데이터베이스를 위한 경제적이고 크기 조절이 가능한 용량을 제공하고 공통 데이터베이스 관리 작업을 관리

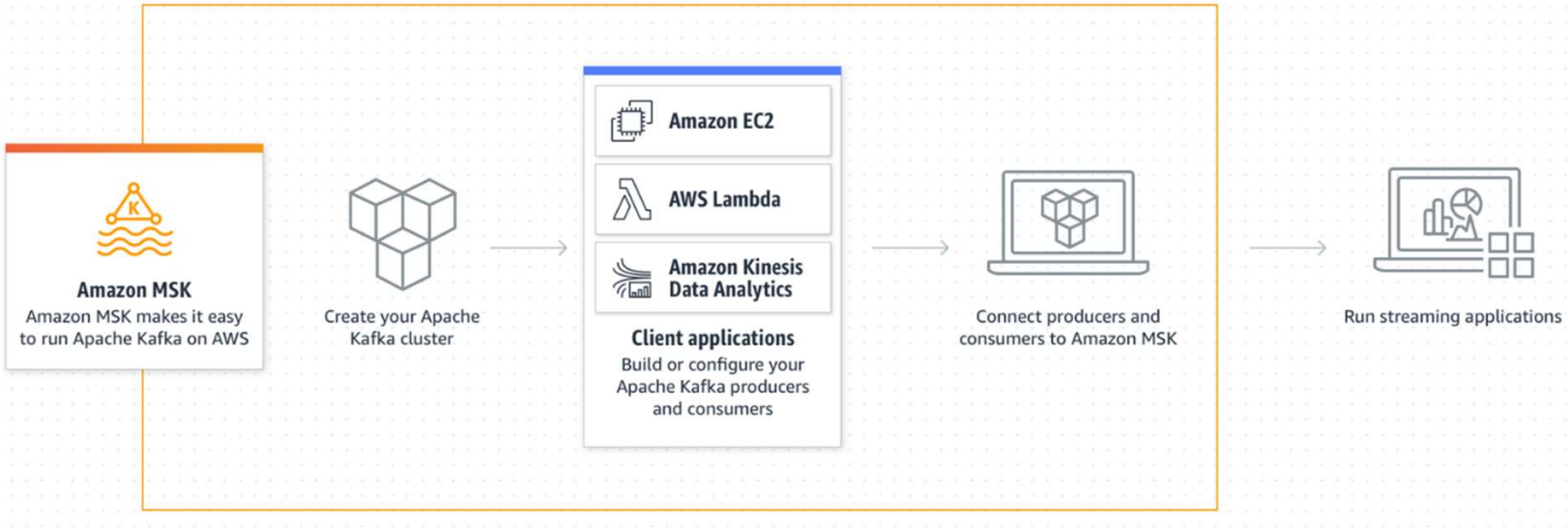
Amazo



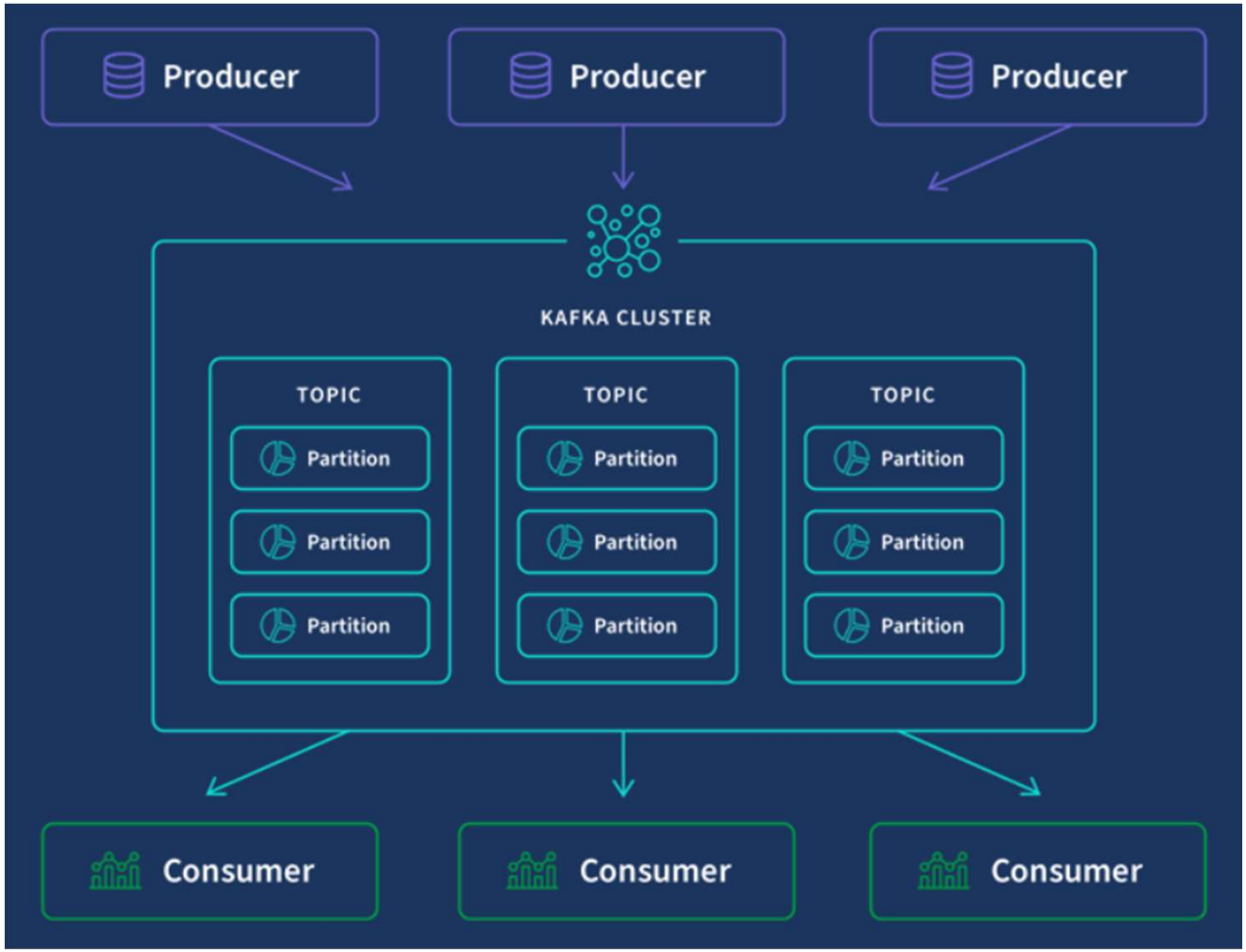
https://www.awsgeek.com/Amazon-RDS/

Amazon MSK

- Kafka 인프라와 운영을 관리하는 AWS 스트리밍 데이터 서비스
- Kafka 운영 관련 전문 지식이 없는 개발자 및 Devops 관리자도 손쉽게 AWS에서 Apache Kafka 애플리케이션과 Kafka Connect 커넥터를 실행할 수 있도록 지원
- Amazon MSK는 Apache Kafka 클러스터를 운영, 유지 관리, 크기 조정하고, 즉시 사용 가능한 엔터프라이즈급 보안 기능을 제공하며, 스트리밍 데이터 애플리케이션 개발 속도를 높여 주는 내장 AWS 통합 기능 제공



Amazon MSK (Cont.)



https://www.qlik.com/us/streaming-data/apache-kafka

Amazon MSK (Cont.) - 용어

- Topic
  - 메시지를 전송하기 위해 사용되는 파티션의 집합
- Partition
  - 메시지를 병렬적으로 처리하기 위한 분산 저장 단위
  - 파티션 내부는 Queue로 되어 있어 순서를 보장하지만, 파티션 간 순서는 보장하지 않음.
- Record
  - 파티션에 들어가는 Byte 배열
  - Key/Value/Timestamp로 구성
- Offset
  - 파티션의 각 레코드를 식별할 수 있는 유일한 값
- Replica
  - 레코드의 복제본으로 이벤트 유실 방지

Amazon MSK (Cont.) -용어

- Broker
  - Kafka Client와 데이터를 주고받기 위해 사용하는 주체
  - 데이터를 분산 저장하여 장애가 발생하더라도 안전하게 사용할 수 있도록 도와주는 요소
- Producer
  - 데이터를 Kafka로 전송하는 주체
- Consumer
  - 데이터를 Kafka로부터 소비하는 주체
- ZooKeeper
  - Cluster의 설정 정보 관리, 동기화 등 Cluster 서버들이 공유하는 데이터 관리

Amazon MSK (Cont.) – Producer

- Broker에게 Record를 전송하는 Client Application
- Record를 Topic으로 보내는 역할을 수행
- Record를 어떤 Topic에 어떤 방식으로 넣을 것인가를 구현
- **acks** 옵션을 통해 Broker에 정상 전송여부 확인
- **Serializer**를 통해 직렬화 전략 선택
- 다양한 **Partitioner**를 지원하여 파티셔닝을 지원
  - Record를 받아서 Partition 번호를 반환하는 역할을 함
  - Record는 Partitioner에 의해 어떤 Partition으로 보내질 것이지가 결정됨
  - Key가 존재할 경우 : UniformStickyPartitioner
  - Key가 존재하지 않는 경우 : **RoundRobinPartitioner**

Amazon MSK (Cont.) – Producer

- **bootstrap.servers**
  - Kafka Cluster에 대한 초기 연결 설정을 위한 Host/Port List
- acks
  - Data Durability(데이터 내구성)
  - 0, 1, all(-1)
  - **acks=0**
    - Producer는 Record 전송 후, Broker로부터 수신 응답을 대기하지 않음
    - Speed 높음, 유실율 높음
  - **acks=1**
    - Producer는 자기가 보낸 Record에 대해 Kafka의 Reader가 메시지는 받았는지 대기
    - Reader가 확인을 보내고, Follower에게 복제가 되기 전에 Reader가 down되면 Message 손실됨.
    - Speed 중간, 유실율 중간
  - **acks=all**
    - Producer는 자기가 보낸 Record에 대해 Kafka의 Reader와 Follower까지 받았는지 기다림.
    - 최소 하나의 복제본까지 처리된 것까지 확인하기 때문에 Message가 손실될 이유가 거의 없음
    - Speed 낮음, 유실율 낮음.

Amazon MSK (Cont.) – Producer

- **serializer**
  - Record의 Key, Value는 지정한 Serializer에 의해서 Byte 배열로 변환
  - key.serializer
    - Key를 Serializer하는 방법에 대한 전략
  - value.serializer
    - Value를 Serializer하는 방법
- **send()**
  - Publish a message to a topic.
- flush()
- close()

Amazon MSK (Cont.) – Consumer

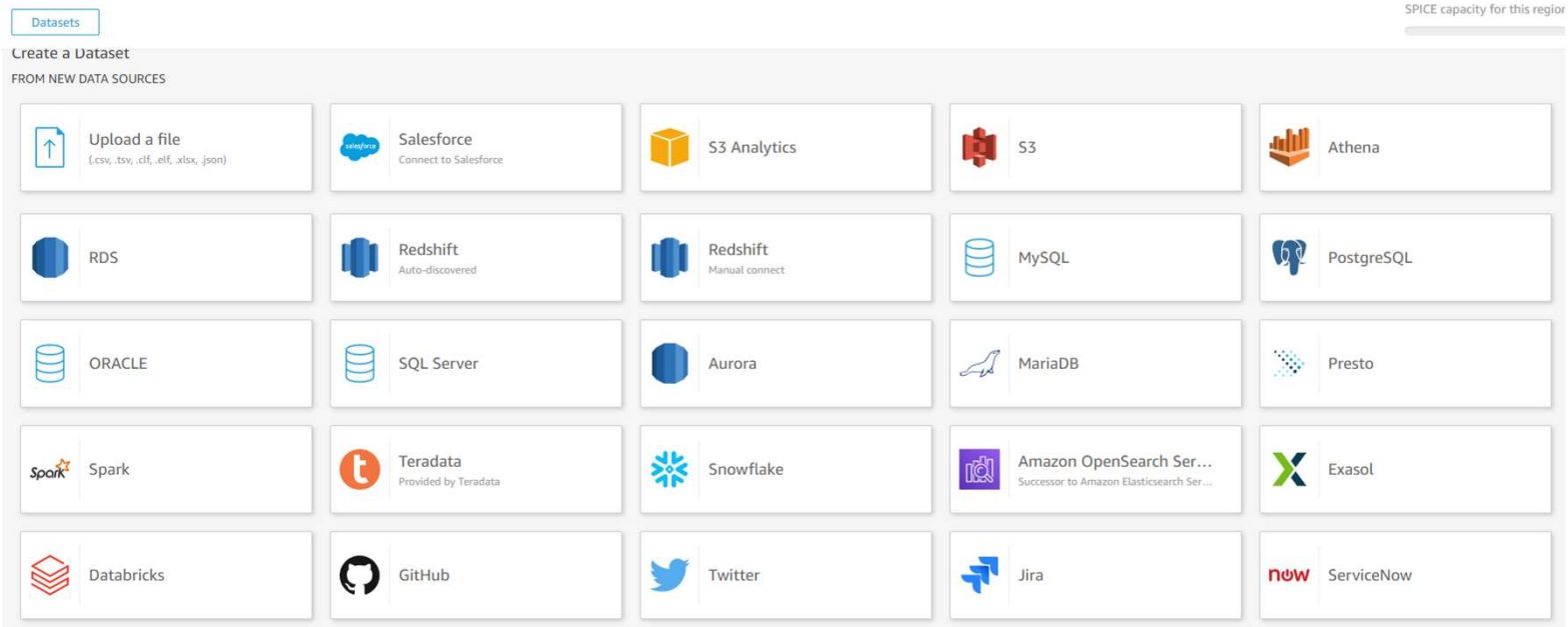
- Topic에서 Message를 소비하는 Client Application
- Partition을 지정하여 Data를 소비할 수 있음
- Consumer Group이 같은 Consumer들은 동일한 Message를 소비하지 않음.
- Consumer Group에 새로운 Consumer가 추가되면 Rebalancing이 일어나게 되고 Partition이 Consumer에 골고루 분배됨
- **Deserializers**
  - Key 및 Value의 Deserialize 수행
  - Byte array를 Object로 변환
- **group.id**
  - 고유한 Consumer Group ID
  - Client가 Message를 소비하기 위해 subscribe()를 사용하거나 Offset 관리 기능을 사용하는 경우 필요

Amazon MSK (Cont.) – Consumer

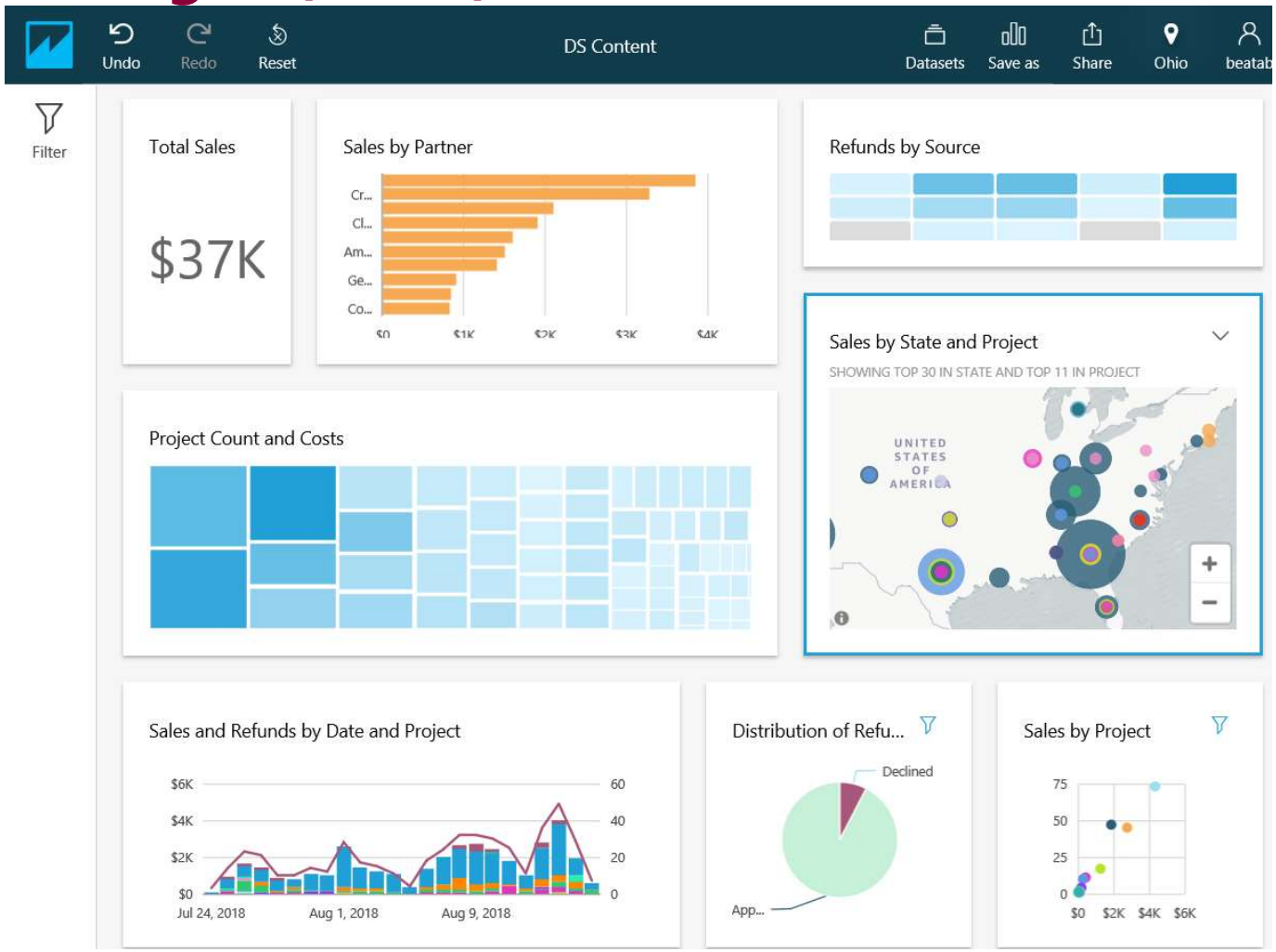
- **auto_offset_reset**
  - A policy for resetting offsets on OffsetOutOfRange errors
  - 'earliest' : move to the oldest available message.
  - '**latest**' : move to the most recent.
  - Default : **latest**
- **enable_auto_commit**
  - True : The consumer's offset will be periodically committed in the background.
  - Default: **True**.
- **consumer_timeout_ms**
  - Number of milliseconds to block during message iteration
- **poll()**
  - Fetch data from assigned topics / partitions.
- **subscribe()**

Amazon QuickSight

- Amazon에서 제공하는 BI 서비스
- 다양한 Source의 Data를 결합
- 단일 Data Dashboard에서 AWS Data, 일반 Data, BigData, Spreadsheet, SaaS Data, B2B 등 조회
- 배포하거나 관리할 Infra없이 사용자 10에서 10000명까지 확장 가능
- Dataset에 대해 매번 조회하지 않고 **SPICE** 저장소(Cache 기능)를 활용해 빠른 조회 속도 제공



Amazon QuickSight (Cont.)

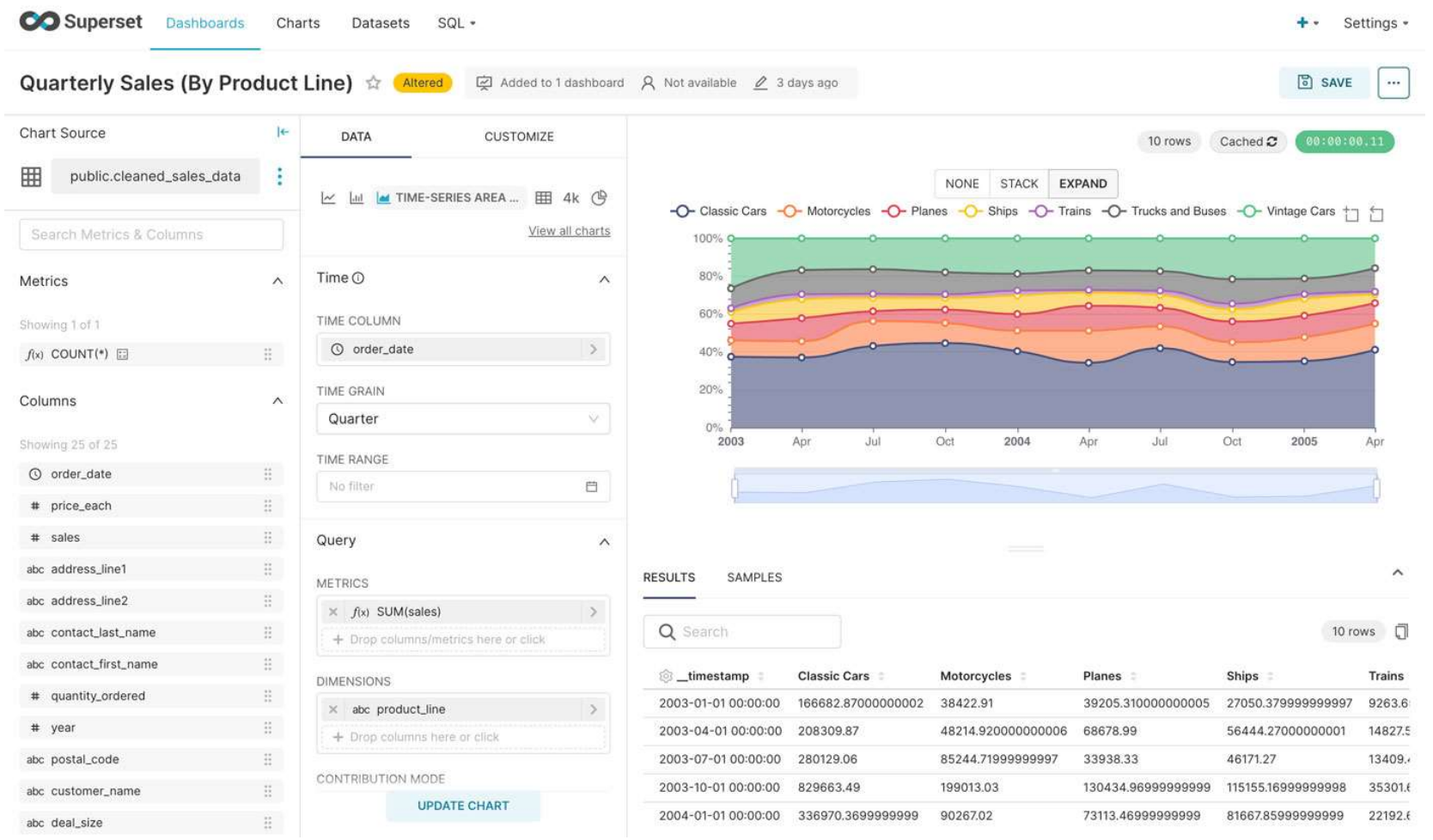


https://www.allthingsdistributed.com/2016/11/amazon-quicksight-generally-available.html

Apache Superset

- Airbnb사에서 Hackathon Project로 시작되어 OpenSource化
- 직관적인 대화형 시각화 BI 플랫폼
- 무료
- SQL을 사용하여 Data를 분석하고 Chart 및 Dashboard를 쉽게 작성 가능
- 다양한 데이터베이스 지원

Apache Superset (Cont.)



https://superset.apache.org/

Tableau

- BI를 위한 Data 시각화 소프트웨어 및 회사
- 2019년 Salesforce에 인수
- 통합 플랫폼으로 폭넓고 심층적인 분석을 지원(AI/ML 기능, 거버넌스 및 데이터 관리, 시각적 스토리텔링, 협업 등)
- 손쉬운 사용법

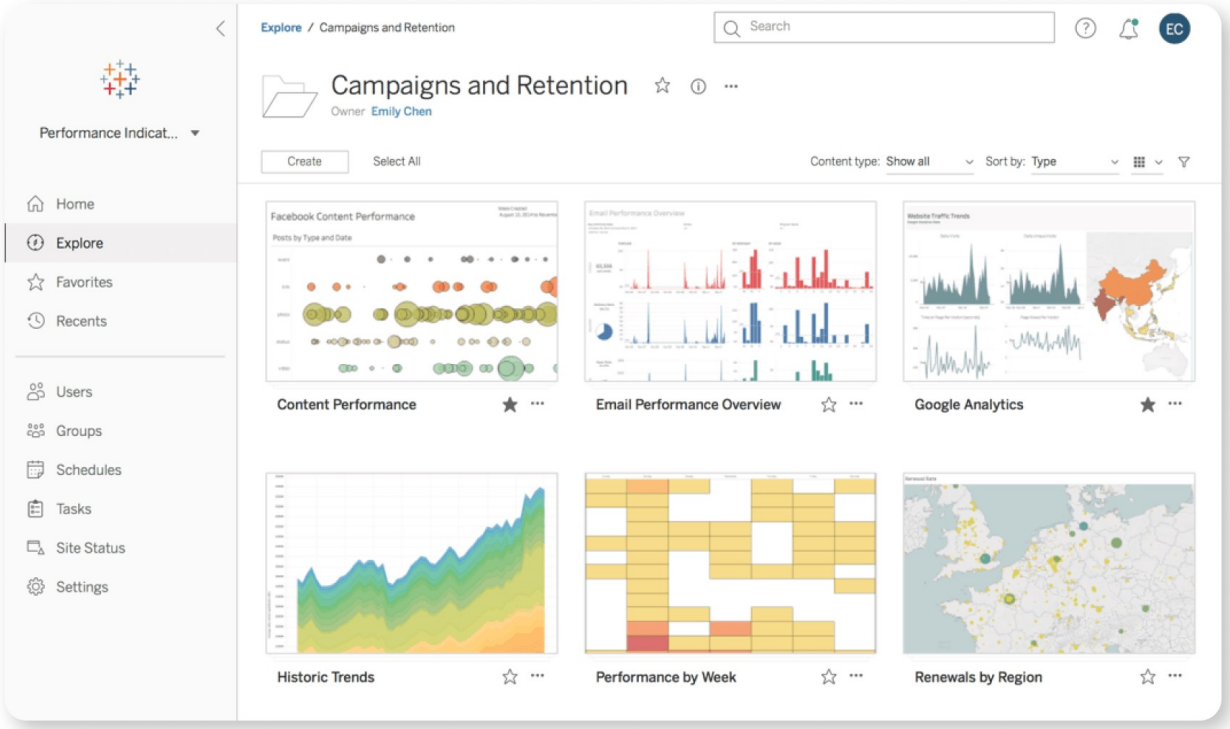
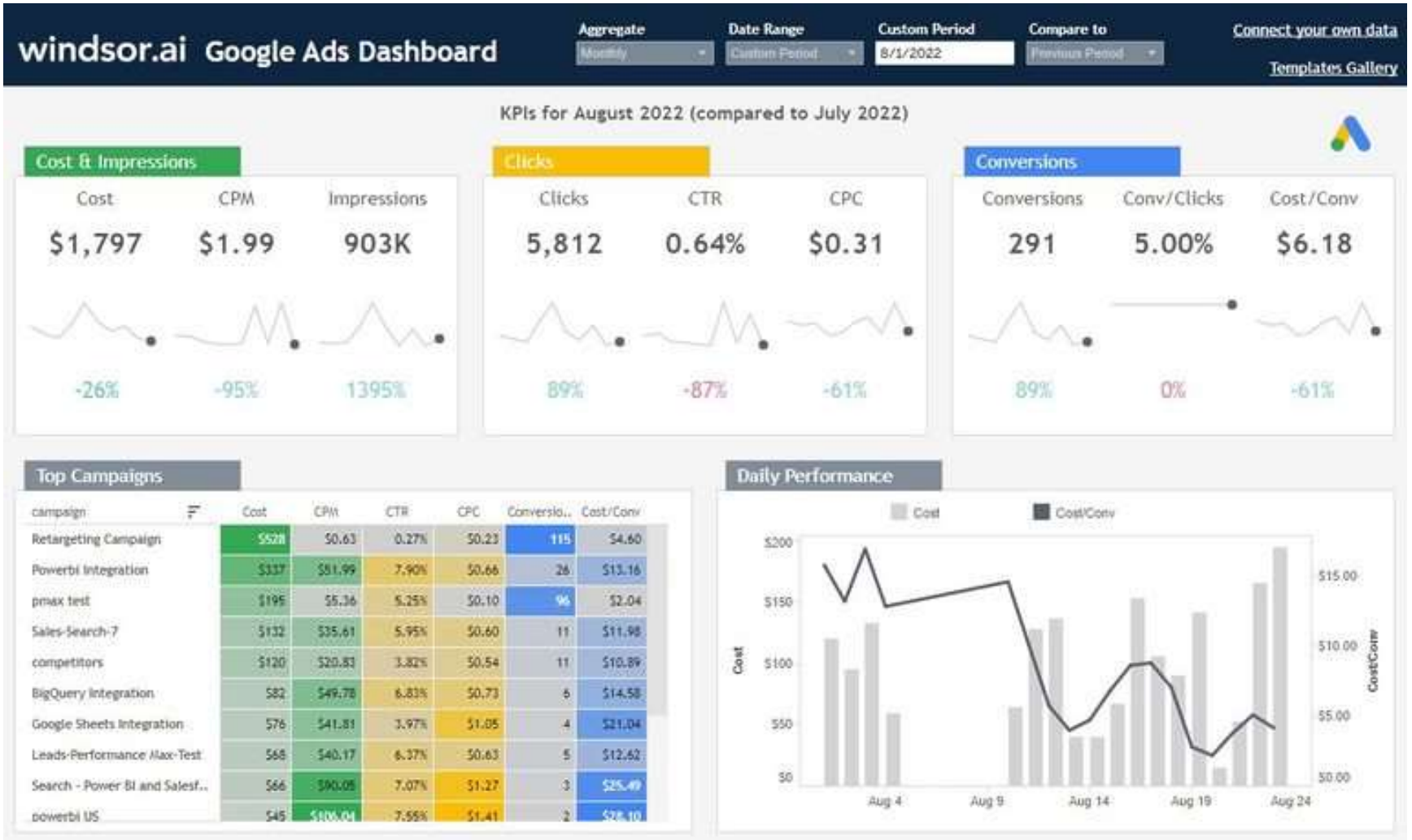


Tableau (Cont.)



https://windsor.ai/tableau-google-ads-dashboard-template/

Tableau (Cont.)

기능	아파치 슈퍼셋	타블로	루커	파워 BI
사용 편의성	보통	높음	높음	높음
사용자 지정	높음	높음	보통	보통
데이터 소스 호환성	높음	높음	높음	높음
비용	무료	$$	$$	$
오픈 소스	예	아니요	아니요	아니요
커뮤니티 지원	높음	높음	높음	높음
교육 리소스	보통	높음	높음	높음
클라우드 기반 옵션	예	예	예	예
온프레미스 옵션	예	예	예	예
시각적 매력	보통	높음	보통	높음
협업 기능	보통	높음	높음	높음

https://docs.kanaries.net/ko/articles/apache-superset-vs-tableau