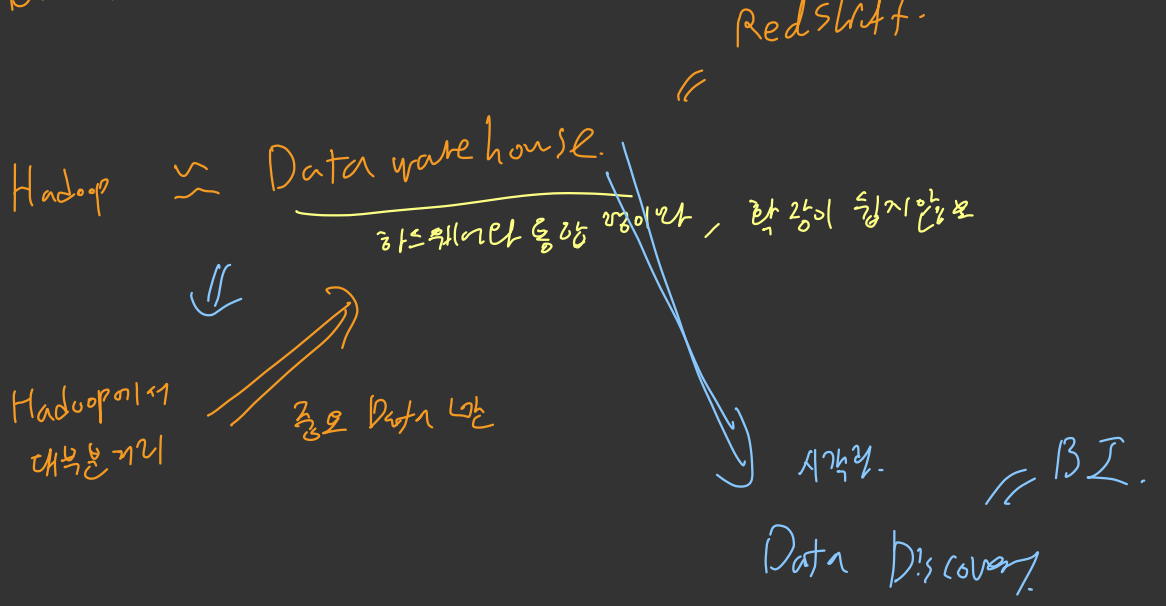
ch1.

Hadoop. - 다수의 컴퓨터에서 대용량의 데이터 처리

NoSQL ⇒ Hadoop
DB기술 분산처리



빅데이터의 특징은 DW와 다르게, 다수의 분산시스템을 조합하여 확장성에 따른 데이터 처리구조를 만듦

① 데이터수집.

① Stream → Stream 처리 → 시각화 DB등
↓ ↑
② Bulk. → 분산 Storage → 분산 데이터처리
S3 = 객체형저장소
Map Reduce.

배치처리 - 어느정도 정리된 데이터를 효율적으로 가공

데이터소스: table, log 등 원시 데이터

DW: 장기보존용으로 정리된 데이터.
장기 보존용.
소량 Data를 자주 읽고 쓰기엔 ...
scheme 정리....

DM: 분석용 축소 data.
분석에 쓰일것만 축소.

Data Lake: 수집 data를 그대로 보존.

BI 도구

① BI 도구에서 직접 Data 소스 접속
간편.
Data 시각화에 제한

② DM을 준비, BI에서 열기 =
오래걸림.
어떠한 Data로
가공할지

③ 웹형식의 BI 도구를
도입, CSV 파일로 출력.

CH 2.

Q 크로스 집계:

크로스 table = 사람들이 보기 쉬운 것.

transaction table = 데이터 추가 방식 table

크로스 집계

Data Lake

데이터 집계 → 데이터 마트 → 시각화.

시간이 오래 걸리므로 해결하기 위해,

1) 모든 데이터를 메모리에 올림

2) 압축과 분산 MPP

ㄴ가능한 Data를 작게 압축, 여러 디스크에 분산.

분산 Data를 읽어 들이려면 멀티코어 (병렬처리) MPP.

Redshift, Google BigQuery

Column Oriented.

- 열지향 (column-oriented)
  - 칼럼 압축 → Disk I/O 줄이기.

☆

- 행지향 → 데이터 추가가 효율적.
  - → 검색을 고속화하기 위해 index를 만든다.
    - → index가 없다면 저장되는 모든 데이터를 load해서 레코드를 찾아야 함.
      - → 많은 I/O.
  - → Data 분석에서는 어떤 column 사용될지 모르기 때문에 index는 별로...

☆ index에 의존하지 않는 고속화 기술이 필요.

- 열지향 - 미리 column 단위로 정리해 두어서, 필요한 column만을 로드하여 디스크 I/O을 한다.
  - → 집계는 고속이지만, 저장하는 시간이 오래...
  - → 압축 효율 좋음!

Redshift (하드웨어 일체형)

◎ MPP에선,
- 병렬화에 의해 멀티코어 활용하기가 가능
  - + 하나의 쿼리를 다수의 작은 task로 분해, 병렬로 실행.
    - → 병목 현상 발생하지 않기 위해, 데이터 고르게 분산.

대화형-
- 분산 스토리지에 보관.

시각화 도구

원본 DB에서 가져옴.

- Redash
  1. 데이터 소스 연결
  2. 쿼리 실행, 결과 그래프로 변환
  3. 그래프 대시보드 구축
- Superset
  - 파이썬으로 구현.
  - 시계열은 Druid.

◎ 시각화 도구 어떤 것을 사용할까?

1) 스프레드시트

2) 대시보드 도구 (Redash)

SQL

동일한 쿼리로, 현황파악용

3) Jupyter notebook 등

Table report

☆ 데이터 마트

~~① 전체 파악용~~

② 중요지표를 위한 높은 일관성 대시보드

③ 대화형 대시보드로 조건을 변경해서.

바뀐 데이터 등장하여, 대강 쿼리 반복 X 안정 ↑

☆ 데이터 마트의 기본구조 — 시각화에 필요한 Data만 모은 DM이 필요

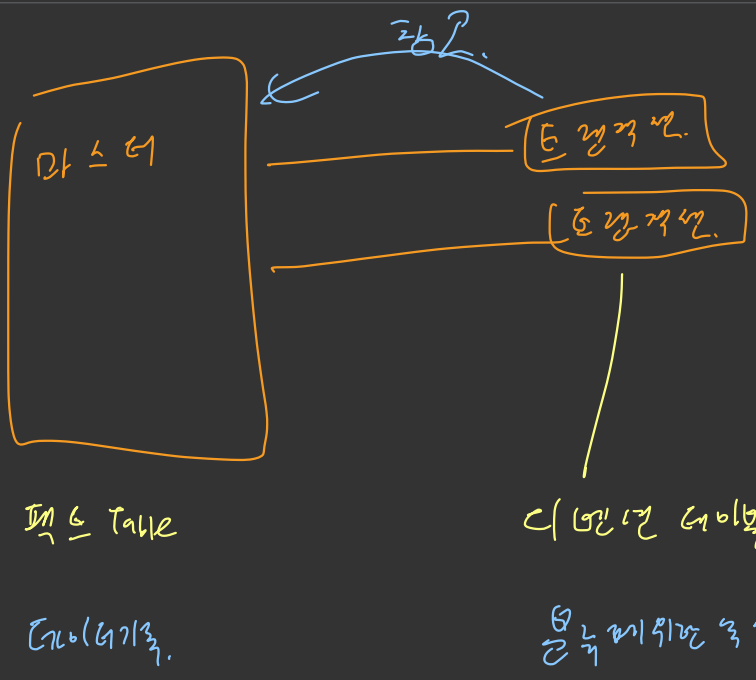
ㄴ 시각화에 적합한 DM 만들기 (OLAP)

ㄴ 다차원 데이터.

요즘엔 OLAP 없이

BI 도구 + MPP DB 조합 구조화

○ 데이터 비정규화.



팩트 Table

데이터 기록.

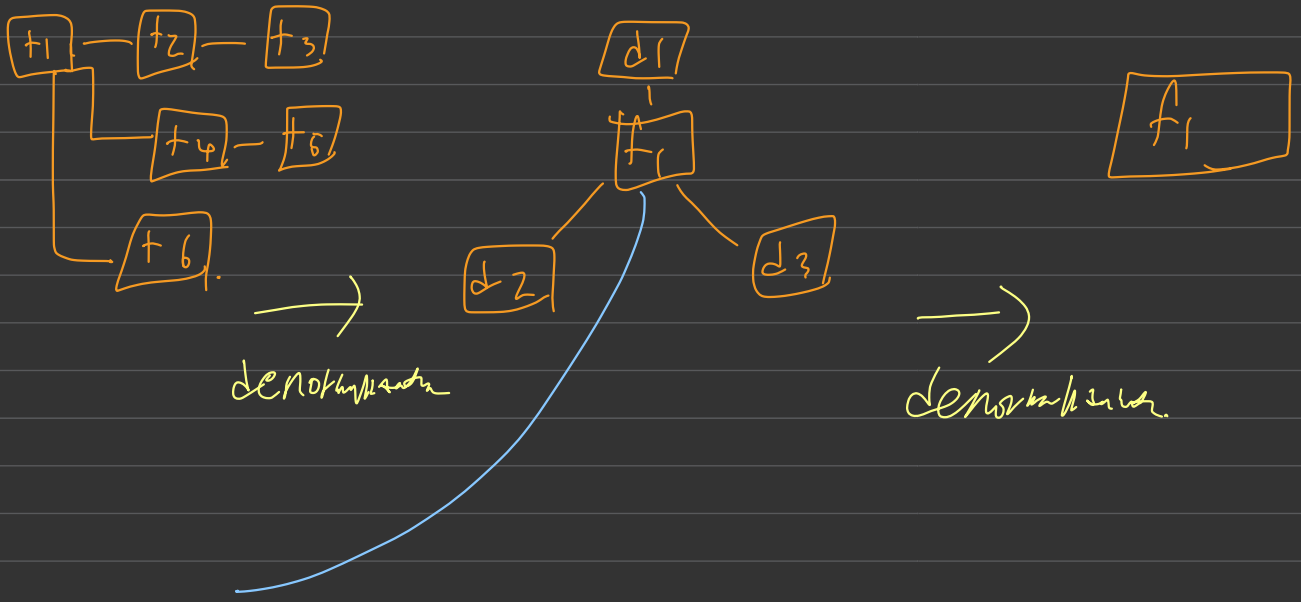
디멘션 데이터.

분석 위한 속성.

① 정규화 (normalize)

② 스타스키마

③ 비정규



① 스타스키마 과정 : 분석할 데이터 가져옴

② fact table을 중심으로 구성.

└ 키, FD 만족함.

★ 데이터 웨어하우스 구조로는 스타스키마가 유명.

① 다차원 모델 시각화에 대비하여 데이터를 측정 가능.

ㄴ 측정값 , 디멘션 구분.
Y X

CH3 대규모 분산처리 FRAMEWORK

데이터 구조.

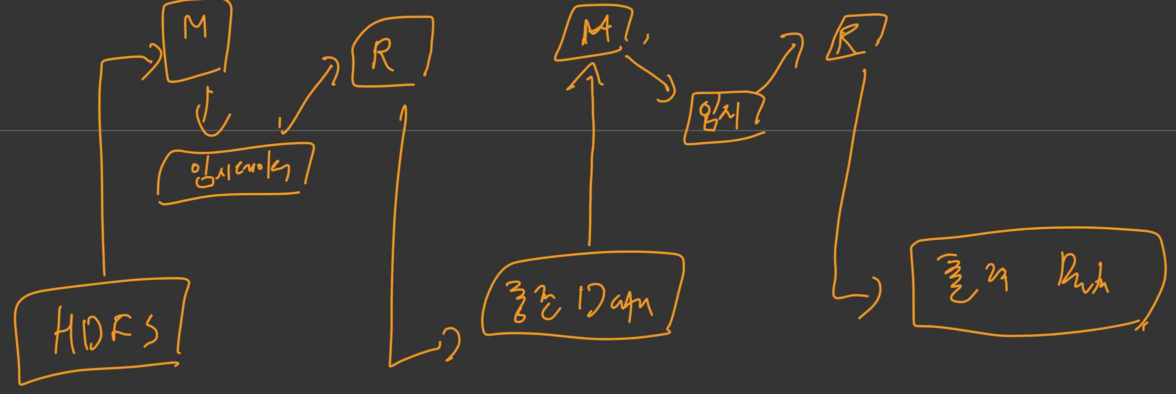
① 구조화 데이터 - scheme table
② 비구조화 - text, Binary, image
③ schemaless (반구조화) - CSV, JSON, XML.

★ 비구조화 데이터를 읽어들여 열지향 Storage로 변환 후 관리해서

(데이터 가공 / 압축을 위해 많은 Resource...

ㄴ 해결위해 Hadoop.

① Hadoop 구조
- 분산 파일 시스템 HDFS — 처리되는 Data는 분산 HDFS에 저장 (다수컴터에 파일복사)
- 리소스 관리 Yarn
  - ㄴ 애플리케이션의 리소스관리
- 분산 데이터처리 MapReduce



ㄴ 해결하기 위해 나온 것이 Tez. (Hive 고속화) 중간 Data 저장없음.

ㄴ 사용할 수 없는 Resource 사용

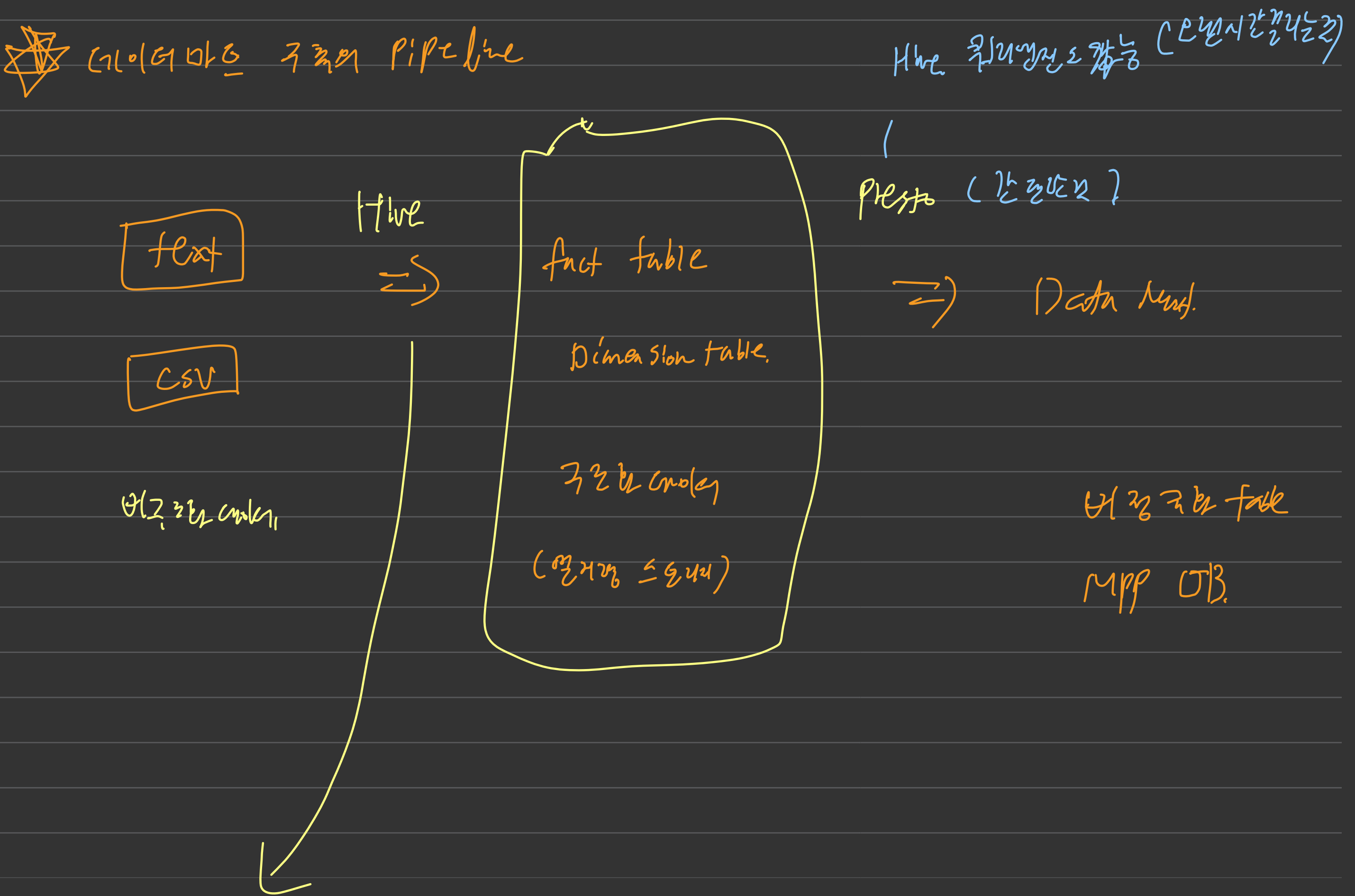
Yarn X

① Spark

ㄴ 가능한 많은 Data를 Ram에 올려놓음.

ㄴ MR을 대체.

★ 데이터 마트 구축의 Pipeline



① 초기단계에서 fact table을 작게하기.

ㄴ 서브 쿼리로 미니멀하게 fact table 작게하기

② Data 저장 크기를 골고루게 분산.

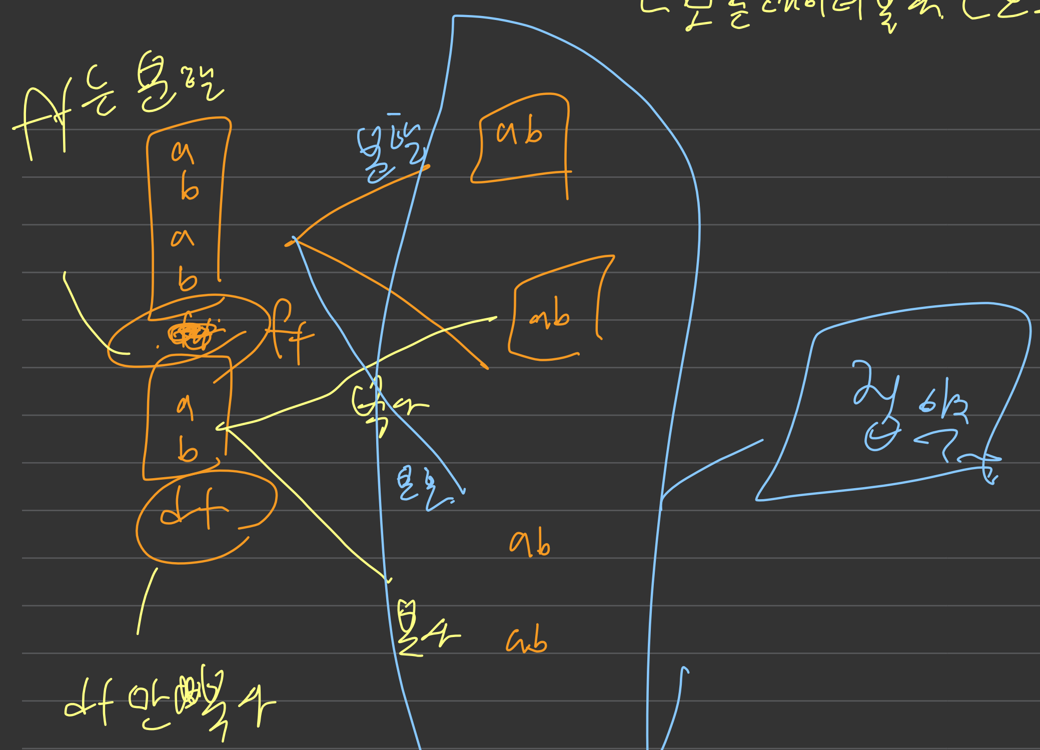
③ Presto.

ㄴ Hive Meta store에서 table 크기 파악가능!

ㄴ 쿼리분석에서 최적의 실행 계획을 생성

~~Hive~~ 분산처리(MR) 및 브로드캐스트 결합도 지원.

ㄴ 모든 데이터 복사 (노드에)



DA 프레임워크 선택하기

① MPP DB - 완성한 비정규화 테이블의 고속집계.

- └ Storage, 계산노드의 일체로 └ Redshift.
- ↗ ETL만 하면 된다. (열 지향)

MPP - BI 연결.

(TEZ)

② Hive - 높은 처리량.

- └ 데이터양에 좌우되지 않는
- └ Text 처리, 무거운 Data.

③ Presto - 속도 중시.

- └ Hadoop 뿐만 아닌 많은 DB와 대응.
- └ Text 처리는 좀...

④ Spark!!!!!

- └ Hive 에서는 data 구조화. → presto 실행

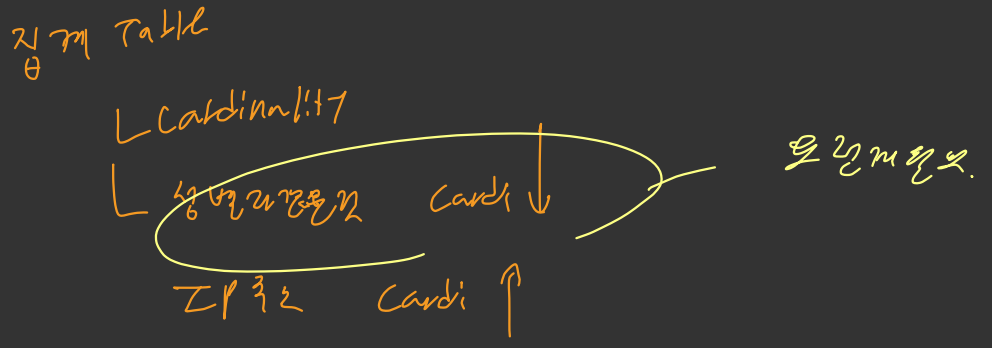
하나의 script로 가능!!~~

즉! 열지향 스토리지 변환 + SQL 집계 결과를 보내는 등 가능!

@ DM 구축!

Fact table 구축

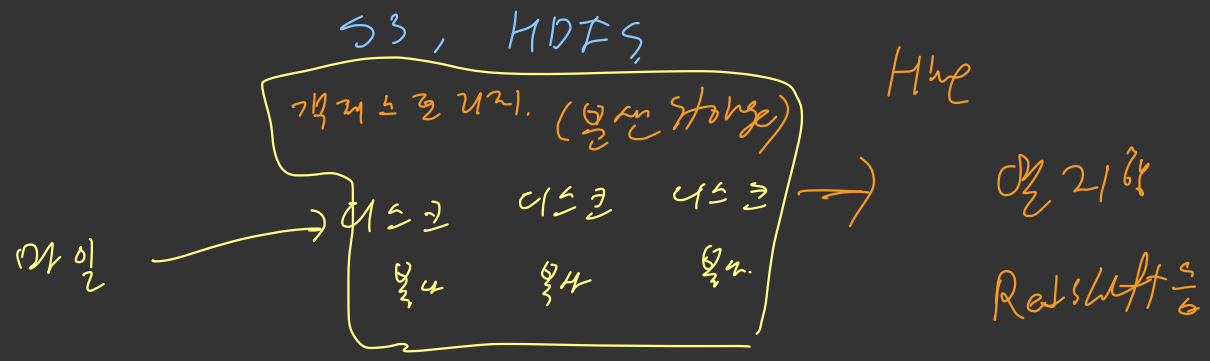
L append? Replace?



Ch 4 빅데이터의 특징

① 배치 방식 스트리밍 방식 Data 특성

- 객체스토리지와 데이터특징 - 분산스토리지 데이터웨어하우징.



☆ 스트리밍 방식 Data 특징.

☆

파일 → 메시지브로커 → 분산스토리지.

임시스토리지.

Kafka

Amazon Kinesis

실시간처리 - 특징

속도측정 - 초.

① 시계열 Data의 최적화.

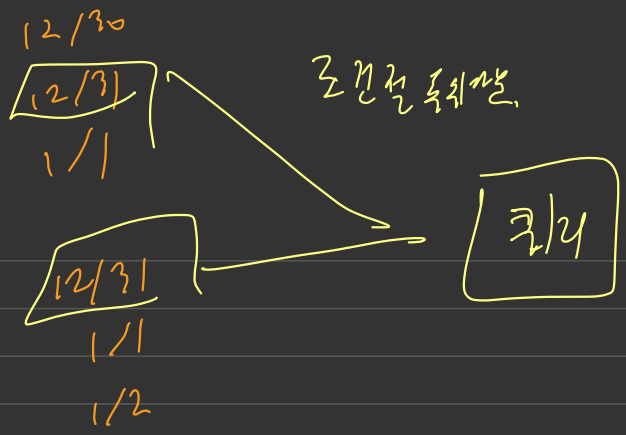
/

① 시계열 index.

└ 이벤트 시간에 index.

② 슬라이딩 윈도우.

시간 틀으로 group 화



1/1

1/2

1/3

③ 이벤트 시간에 의한 물결 — 위와 같이 시계열 table 새로 생성

④ DM을 이벤트 시간으로 정렬. — DM 시계열 만들기

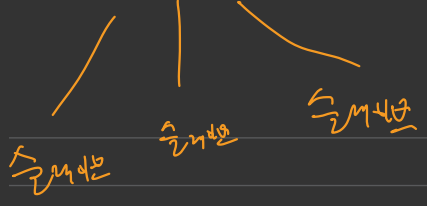
NoSQL DB

① 분산 KVS.

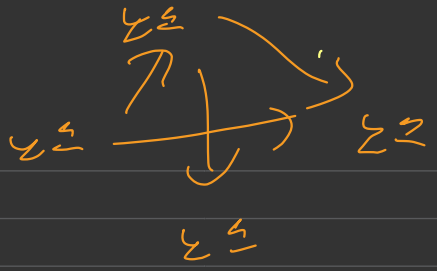
- 모든 데이터를 키-value 형태로 저장하는 구조
- 노드간의 부하를 균등하게 분산.

client

① 마스터



② P2P

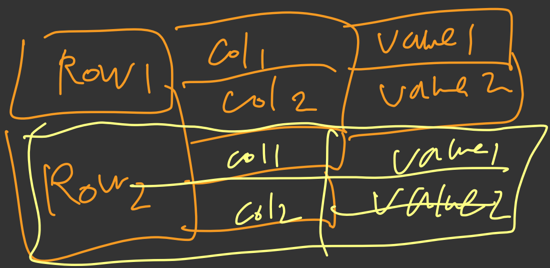


DynamoDB - P2P 형의 분산 아키텍처

- 사용자가 늘어나면 비용은?, 0원!

② Wide column store

ㄴ 분산 KVS 확장판, 2개 이상의 칼럼에 데이터 저장.



③ Document store.

ㄴ 데이터 저장 유형.

예) 몽고DB 등

2 계층 등...

CHB 워크플로 개념.