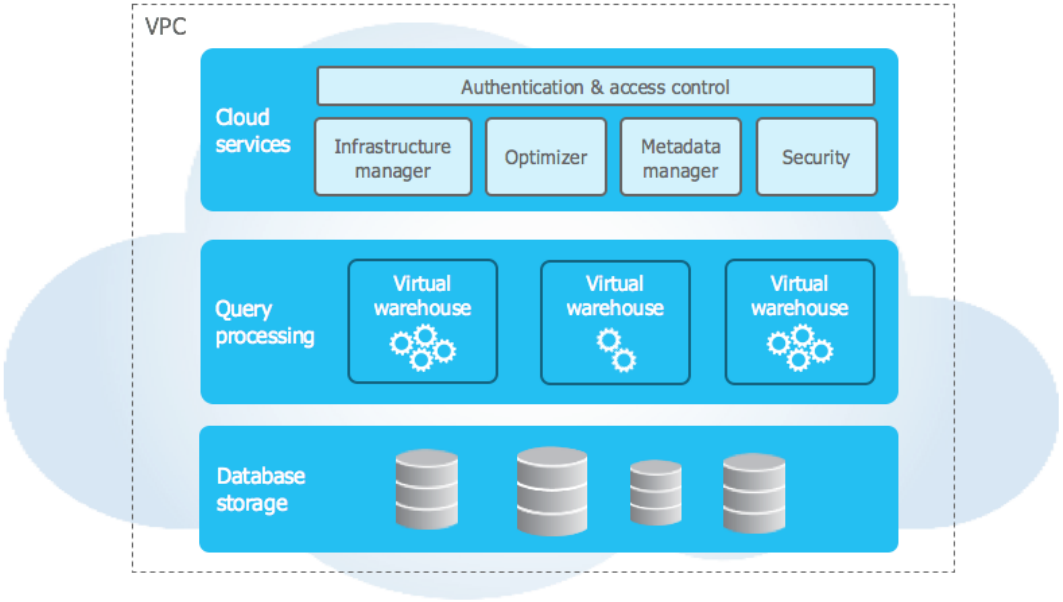
[Snowflake] 3-1. Data Storage Layer

💡 노션 웹 공유 링크 **(댓글 & 상세설명 참고)**

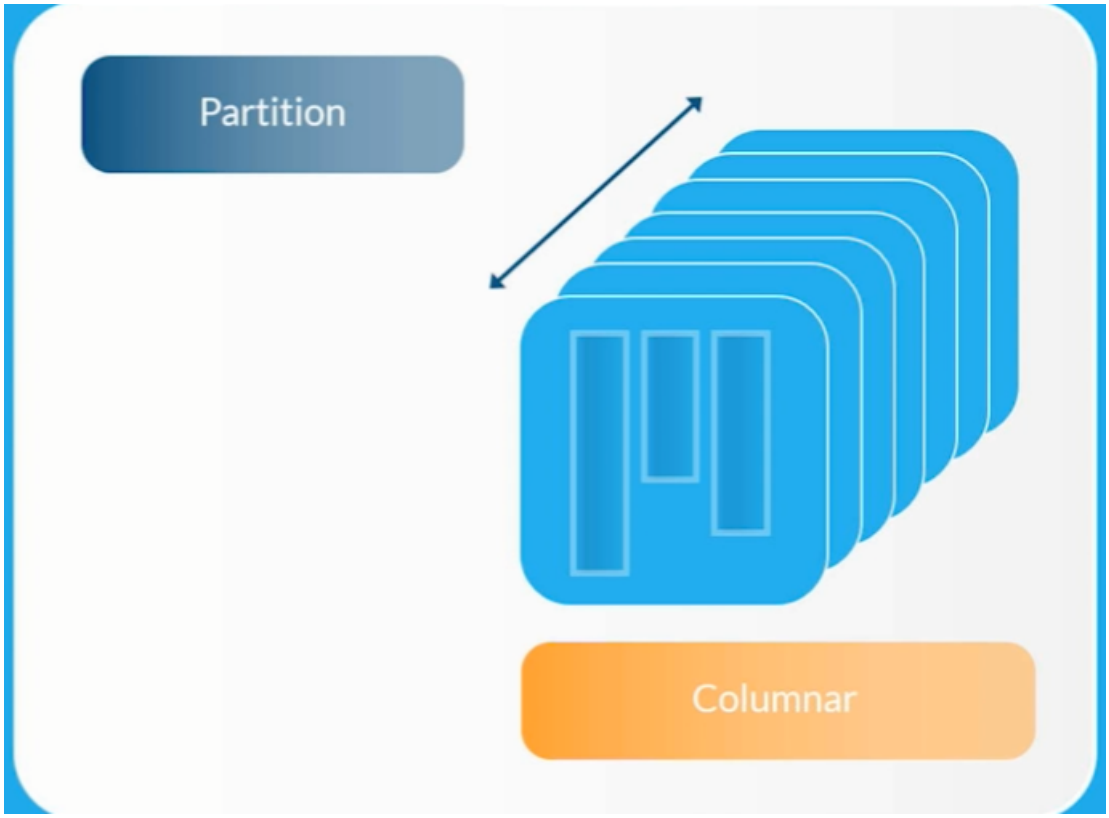
> **References**
>
> - **Snowflake Learn (SnowPro PREP-CORE Course) 3장 1강**
> - **Snowflake 설명서 (데이터베이스 저장소)**
> - **Blog (컬럼 기반 vs 행 기반 저장소)**
> - **Blog (DB 파티셔닝이란)**
> - **Snowflake 설명서 (마이크로 파티션 및 데이터 클러스터링)**
> - **참고 (Snowflake Best Practice)**

- **스토리지 계층 (Data Storage Layer)**



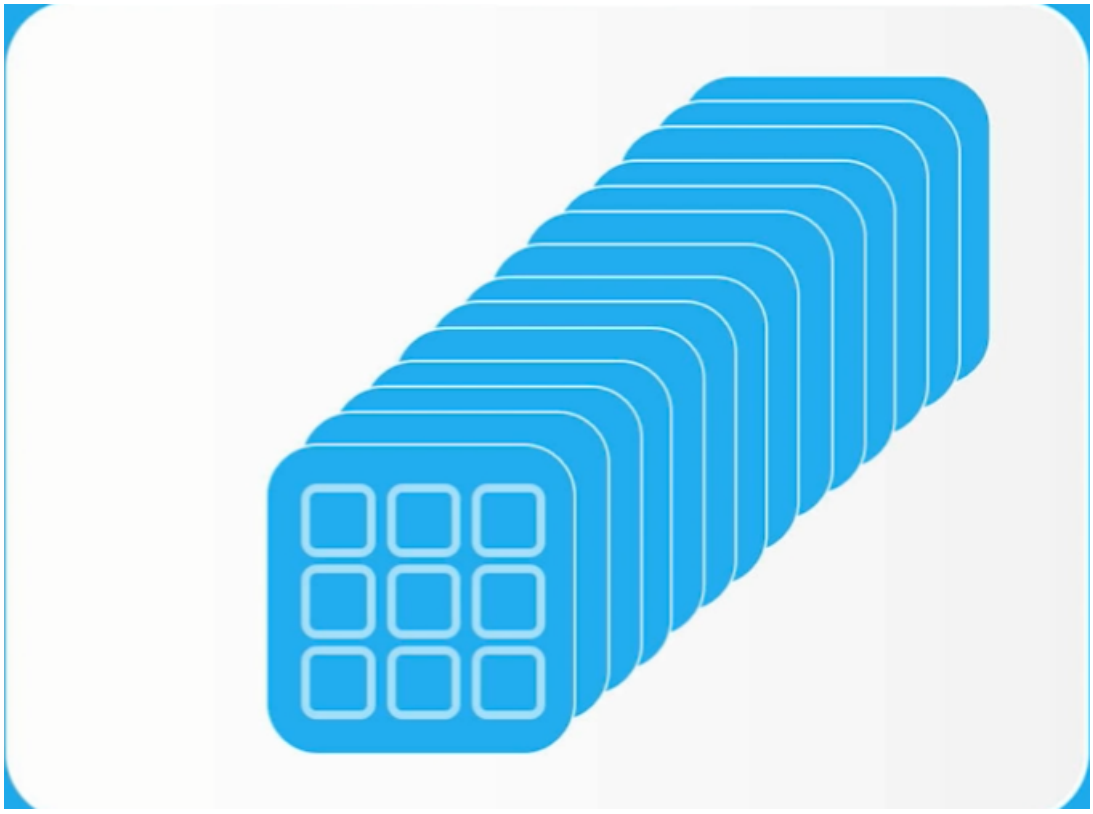
- Snowflake 는 클라우드 기반 스토리지를 사용한다.
  - 사용하는 클라우드 공급자 (AWS, GCP, Azure) 에 따라 다름
- Hybrid Columnar 저장 방식
  - 컬럼지향 저장방식과, 로우지향 저장방식을 혼합하여 사용
- 자동 마이크로 파티셔닝
  - 마이크로 파티셔닝은 테이블을 여러 개의 작은 파티션으로 분할하여 각 파티션에 대한 메타데이터와 통계를 저장
- 반 구조화 데이터 (Semi-Structured data) 지원
  - JSON, AVRO, ORC, XML, Parquet
- Snowflake 의 스토리지는 압축된 다음 과금 청구

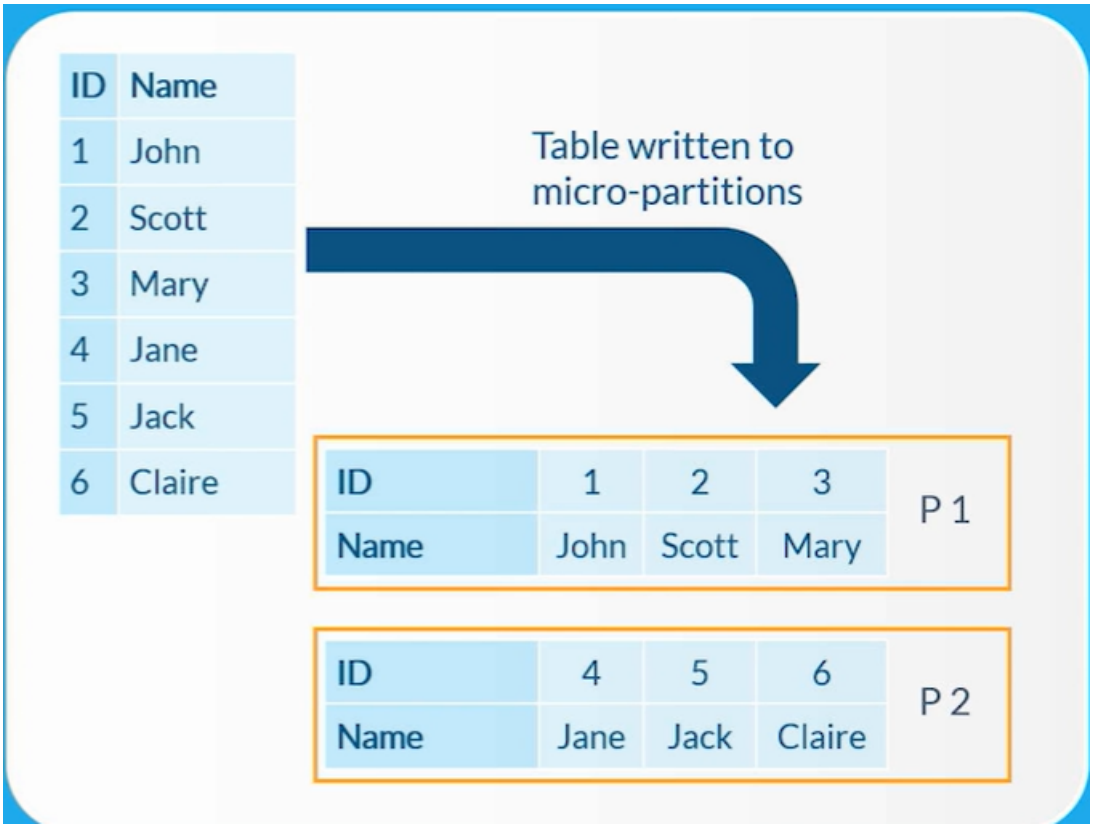
- **열 기반 저장소 (Columnar Compression)**



- 데이터베이스나 데이터 웨어하우스에서 사용되는 압축 기술 중 하나로 컬럼 단위로 데이터를 저장하고 압축
  - 컬럼 단위로 데이터를 저장함으로써 중복되는 값이 많은 경우에 효과적인 압축
- Columnar compression은 같은 값이 반복되는 경우 해당 값을 한 번만 저장하고, 나머지 데이터는 해당 값의 인덱스로 대체하여 저장 (데이터 크기 감소)
- 데이터를 로드하거나 캡처할때, 테이블이나 마이크로파티션에 로드할 때 자동으로 분석 및 압축
- 각 데이터 유형에 대한 최적의 압축방식을 찾아 효율적으로 압축
- 컬럼 단위로 압축을 하기 때문에 쿼리 처리 시 필요한 컬럼만 읽어오기 때문에 디스크 I/O 횟수가 줄어들어 처리 속도가 향상

- **Snowflake 의 마이크로 파티션 (Micro-Partitions) (사전 참고 - DB 파티셔닝)**



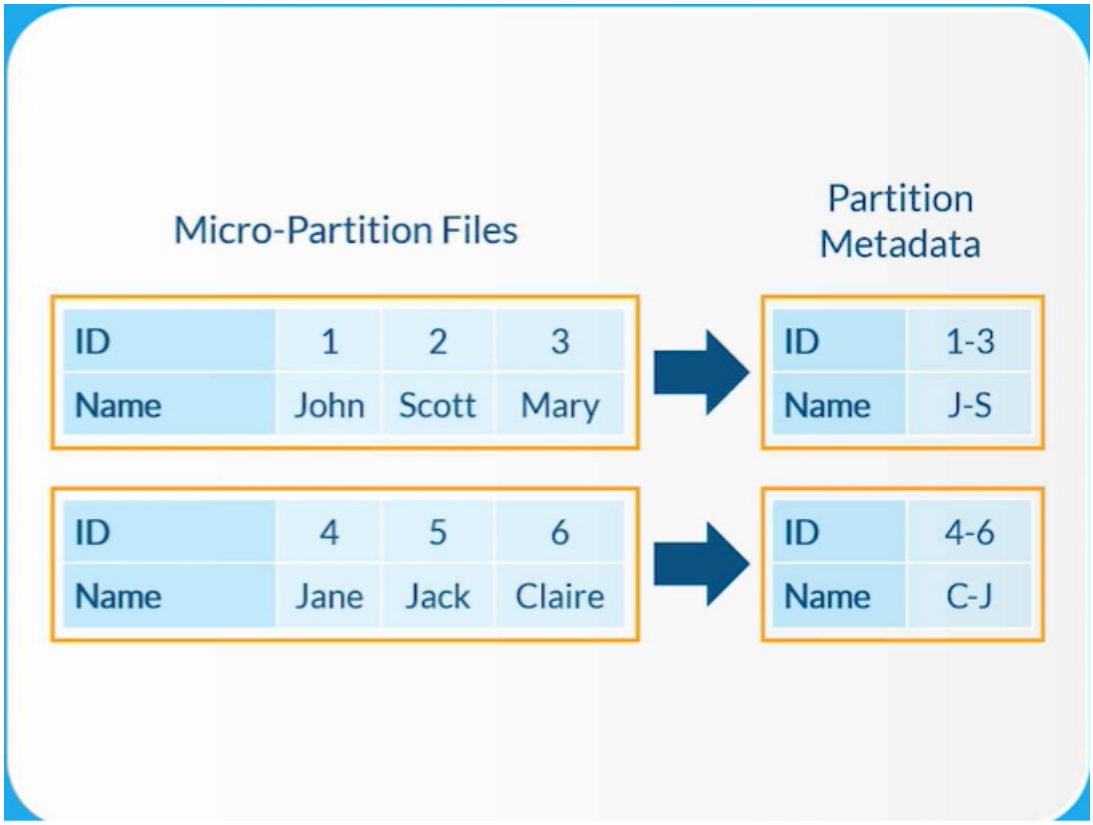


- Snowflake 테이블의 모든 데이터는 인접한 저장소 단위의 마이크로 파티션으로 자동분할
  - 각 마이크로 파티션은 50MB ~ 500MB의 압축되지 않은 데이터 포함 (압축 전)
  - 몇가지 예외가 있지만, 일반적으로 각 마이크로 파티션으로 변환 될 때 최대 약 16MB (압축 후)

- 각 테이블의 행들은 열 방식으로 구성 된 개별 마이크로 파티션에 매핑
- 마이크로 파티션은 크기가 작아 효율적인 DML 및 더 빠른 쿼리를 위한 세분화된 정리 가능

→ 이 구조는 수백만, 수억개의 마이크로 파티션으로 구성될 수 있는 테이블을 세분화된 정리가 가능하게 함

- 마이크로 파티션은 Immutable (불변), 편집 불가능
  - 데이터를 로드하고 마이크로파티션에 이동하면 업데이트, 삭제 이러한 조작으로 테이블 레코드에 변경이 생기면 새 마이크로 파티션이 작성 되어야함
  - 버전으로 관리되며, 클라우드 서비스 계층에 아키텍처에서 추적
    - 테이블도 버전으로 관리되며, Time Travel 기능과 관련



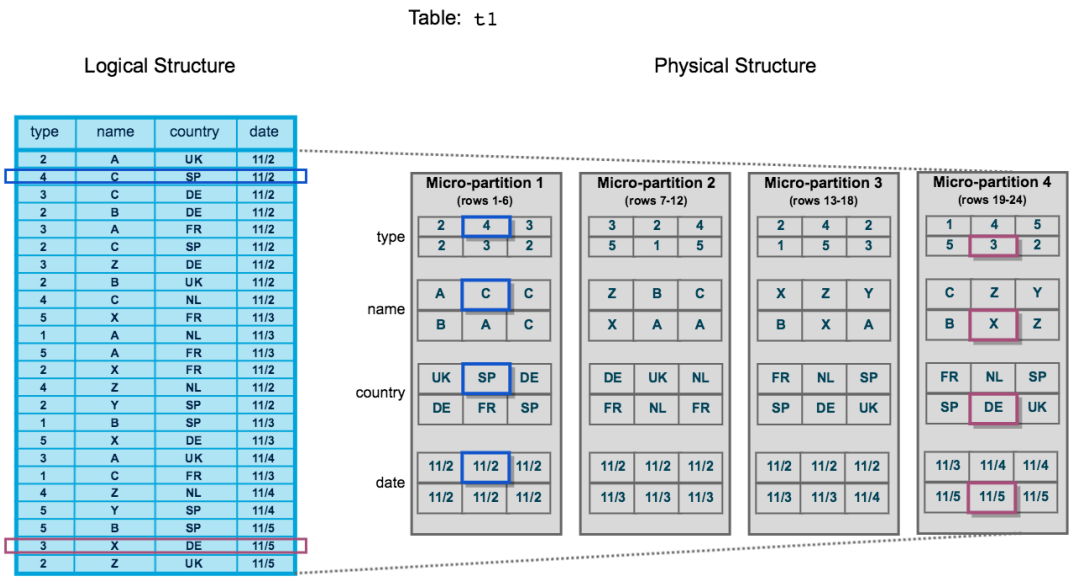
- 각 마이크로 파티션에는 모든 행에 대한 메타데이터 저장
  - 각 열에 대한 값 범위
  - 고유 값 수
  - 최적화 및 효율적인 쿼리 처리에 사용되는 추가 속성

  → 이 메타데이터들은 자동으로 수집하고 유지
- Snowflake 의 마이크로 파티션은 기존의 정적 파티셔닝과 달리 자동으로 생성
  - 명시적으로 사전 정의, 사용자 유지관리 필요가 없음
- 마이크로 파티션은 값 범위에서 겹칠 수 있으며, 균일하게 작은 크기와 결합, 왜곡을 방지하는데 도움

- 열을 열 저장소라고 하는 마이크로 파티션 내 독립적으로 저장, 이를 통해 열을 효율적으로 스캔
  - 열은 마이크로 파티션 내에서 개별적 압축
- 각 테이블에 대해 클러스터링 키를 지정, 특정 테이블에서 클러스터링 사용 가능

- **마이크로 파티션의 영향 - 참고**
  - Snowflake 의 모든 DML 작업 (Delete, Update, Merge 등)은 기본 마이크로 파티션 메타데이터를 활용하여 테이블 유지관리를 용이하고 단순화

- **데이터 클러스터링이란**



날짜별로 정렬된 Snowflake 테이블 예시

- 일반적으로 테이블에 저장된 데이터는 날짜 또는 지리적 리전에 따라 정렬
- Snowflake 에서는 데이터가 테이블에 삽입/로드 되면서 클러스터링 메타데이터가 수집되고, 프로세스 중 생성 된 각 마이크로 파티션에 기록
- 클러스터링 정보를 활용하여 불필요한 스캔을 방지 & 쿼리 성능 가속화
- 클러스터링 메타데이터
  - 테이블을 구성하는 총 마이크로 파티션 수
  - 지정된 테이블 열 하위 세트에서 서로 겹치는 값을 포함하는 마이크로 파티션의 수
  - 겹치는 마이크로 파티션의 깊이

- **데이터 스토리지 요금**

- 고객은 스토리지 사용량에 따라 요금 부과
- 월 요금 부과
- 스토리지 청구 단위는 테라바이트 단위로, 일일 평균을 구하고 전일평균 계산, 그 값을 사용하여 월별 평균을 내서 청구
- On-demand, Pre-Purchase Capacity 방식이 있으며, 청구용량은 다름