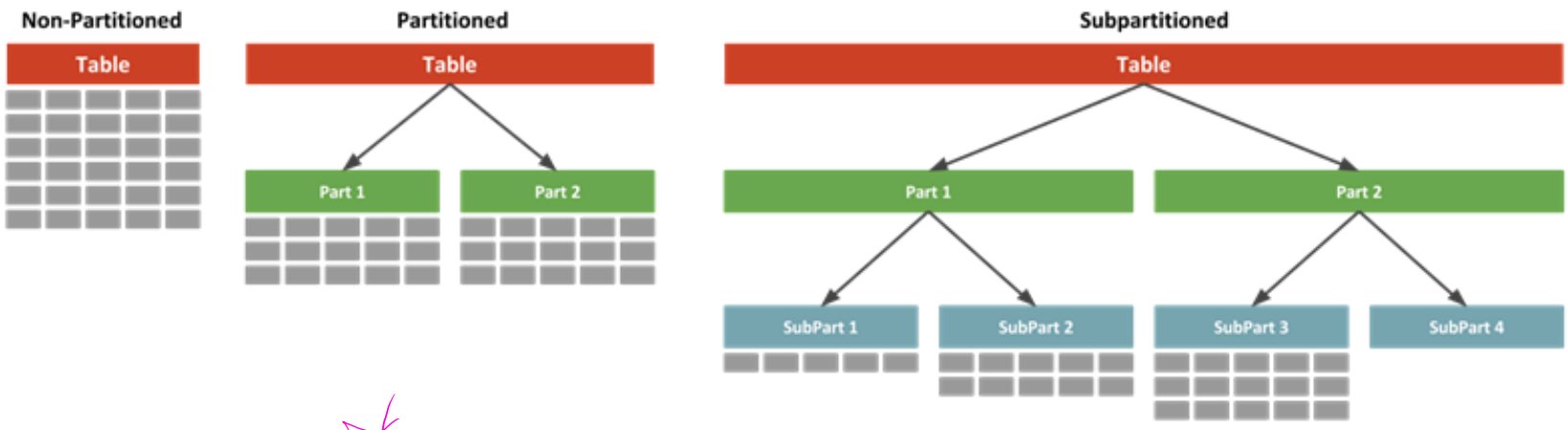
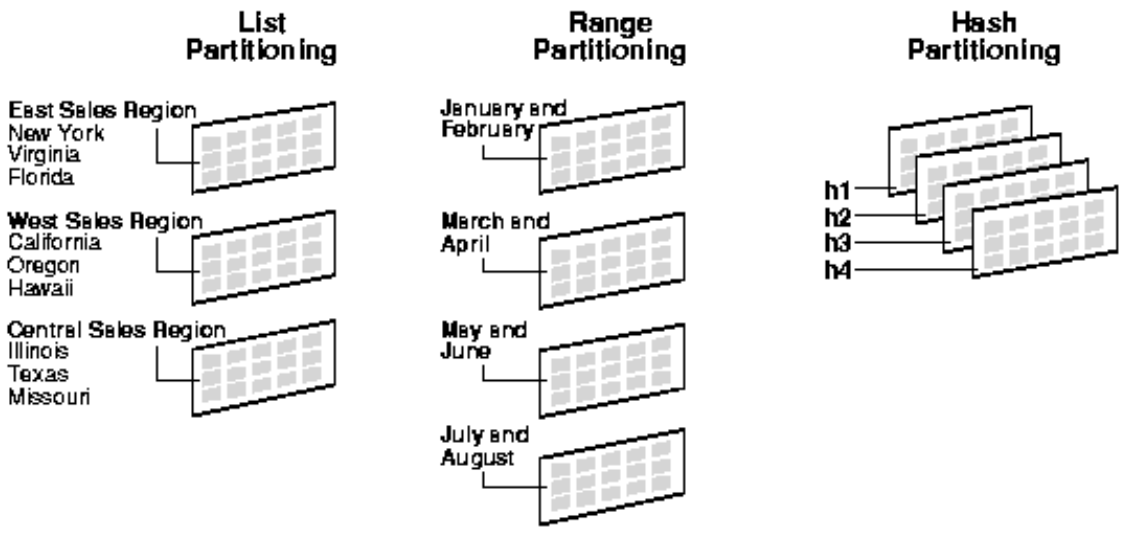
테이블 파티셔닝



논리적으로는 테이블로 접근하지만 물리적으로는 테이블 내의 각각의 파티션으로 접근한다. 각각의 파티션은 세그먼트에 해당한다. 세그먼트는 테이블과 1:1을 갖는다. 테이블 파티셔닝은 테이블 데이터를 일정 기준으로 나누어 저장하는 것이다. 예를 들어 서점에서 '오라클'관련 책을 찾기 위해 '데이터베이스' 코너로 가서 오라클을 찾을 수 있을것이다. 이 처럼 파티션은 데이터를 찾을 때 조금 더 쉽고 빠르게 찾을 수 있도록 도와주는 것이다. 단일 파티션으로 되어있는 경우에는 특정 영역에 해당하는 데이터를 삭제하려면 DELETE를 사용해야하지만 파티셔닝의 경우는 해당 파티션을 DDL로 삭제를 할 수 있다.

파티셔닝을 하는 방법은 크게 3가지가 있다.

- 레인지(Range) 파티션
- 리스트(List) 파티션
- 해시(Hash) 파티션



파티션 활용

SQL 고급 활용 및 튜닝 | 고급 SQL 튜닝 | 파티션 활용 | 작성자 admin | 작성일 2021-02-15 13:01 | 조회 2870

1. 파티션 개요

파티셔닝(Partitioning)은 테이블 또는 인덱스 데이터를 파티션(Partition) 단위로 나누어 저장하는 것을 말한다. 테이블을 파티셔닝하면 파티션 키에 따라 물리적으로는 별도의 세그먼트에 데이터를 저장하며, 인덱스도 마찬가지다. 파티셔닝이 필요한 이유를 관리적 측면과 성능적 측면으로 나누어 볼 수 있다.

- 관리적 측면 : 파티션 단위 백업, 추가, 삭제, 변경
- 성능적 측면 : 파티션 단위 조회 및 DML 수행, 경합 및 부하 분산

파티셔닝은 우선 관리적 측면에서 많은 이점을 제공한다. 보관주기가 지난 데이터를 별도 장치에 백업하고 지우는 일은 데이터베이스 관리자들의 일상적인 작업인데, 만약 파티션 없이 대용량 테이블에 이런 작업들을 수행하려면 시간도 오래 걸리고 비효율적이다. 대용량 테이블에 인덱스를 새로 생성하거나 재생성할 때도 파티션 기능을 이용하면 효과적이다. 성능적 측면의 효용성도 매우 높다. 데이터를 빠르게 검색할 목적으로 데이터베이스마다 다양한 저장구조와 검색 기법들이 개발되고 있지만, 인덱스를 이용하는 방법과 테이블 전체를 스캔하는 두 가지 방법에서 크게 벗어나지는 못하고 있다. 인덱스를 이용한 Random 액세스 방식은 일정량을 넘는 순간 Full Table Scan보다 오히려 성능이 나쁘다. 그렇다고 초대용량 테이블을 Full Scan 하는 것은 매우 비효율적이다. 이런 경우 테이블을 파티션 단위로 나누어 관리하면, Full Table Scan이라 하더라도 일부 세그먼트만 읽고 작업을 마칠 수 있다. 테이블이나 인덱스를 파티셔닝하면 DBMS는 내부적으로 2개 이상(생성 초기에 하나일 수는 있으나 계속 하나를 유지한다면 파티셔닝은 불필요)의 저장영역을 생성하고, 그것들이 논리적으로 하나의 오브젝트임을 메타정보로 관리한다. 파티션되지 않은 일반 테이블일 때는 테이블과 저장영역(Oracle의 세그먼트)이 1:1 관계지만 파티션 테이블일 때는 1:M 관계다. 인덱스를 파티셔닝할 때도 마찬가지다.