**파티션 활용**

1. **파티션 개요**

- 테이블 또는 인덱스를 파티션 단위로 나누어 저장하며 파티션 키에 따라 물리적으로 별도의 세그먼트에 데이터 저장

- 보관 주기가 지난 데이터를 별도 장치에 백업하고 삭제하는 작업이 높은 효율

- 특정 파티션만 FULL SCAN으로 높은 효율 가능

**2. 파티션 유형**

1) RANGE 파티셔닝 : 파티션 키 값의 범위로 분할. 가장 일반적이며 주로 날짜칼럼 위주

2) HASH 파티셔닝 : 데이터가 고르게 분산되도록 DBMS가 관리하며 병렬 처리시 성능 효과 극대화되며 DML 경합 분산에 효과적이다.

3) LIST 파티셔닝 : 불연속적인 값의 목록을 각 파티션에 지정. 사용자가 정한 그룹핑 기준에 따라 데이터를 분할 저장

4) COMPOSITE 파티셔닝 : 파티션 내에 또 다른 서브 파티션 구성

**3. 파티션 PRUNING**

* 옵티마이저가 SQL의 대상 테이블과 조건절을 분석후 불필요한 파티션을 액세스 대상에서 제외하는 기능. 파티션 테이블 조회 또는 DML 수행 시 극적은 성능 개선을 줌
* DBMS가 지능적으로 수행하는 매커니즘이지만 파티션 키 칼럼 가공 및 데이터 타입이 묵시적 형 변환 시에도 정상적인 파티션 PRUNING이 수행되지 않는다.
  + 1. 정적 파티션 PRUNING : 액세스할 파티션을 컴파일 시점에 미리 결정하며, 파티션 키 칼럼을 상수 조건으로 조회하는 경우

2) 동적 파티션 PRUNING : 파티션을 실행 시점에 결정. 파티션 키 칼럼을 바인드 변수로 조회하는 경우.

**4. 인덱스 파티셔닝**

1) 로컬 파티션 / 글로벌 파티션 인덱스

- LOCAL 파티션 인덱스 : 테이블 파티션과 1:1 로 대응되도록 파티셔닝한 인덱스

- GLOBAL 파티션 인덱스 : 테이블 파티션과 독립적인 구성을 갖도록 파티셔닝한 인덱스

2) PREFIXED / NON-PREFIXED 파티션 인덱스

- PREFIXED : 파티션 키 칼럼을 인덱스 선두 컬럼에 위치

- NON-PREFIXED : 인덱스 선두 칼럼이 아닌 곳에 파티션 키 칼럼 위치 / 파티션 키 값이 인덱스에 없는 경우

- GLOBAL NON-PREFIXED는 존재하지 않는다.

3) 비파티션

- 파티션 PRUNING이 실행되지 않을 경우 파티션 액세스 할 때 비효율이 발생 (특히 OLTP)

- NL 조인에서 파티션 키에 대한 넓은 범위 검색 조건으로 INNER테이블 액세스 시 고려

- 파티션 인덱스를 이용하면 SORT ORDER BY 대체 효과 상실

- 테이블 파티션 이동, 삭제 등의 작업 시 UNUSABLE 되므로 적용 시 주의

4) GLOBAL PREFIXED

- 인덱스 경합 분산에 효과적

- 여러 로컬 인덱스 파티션을 액세스 하는 것이 비효율적일 때 대안으로 활용 가능

- 테이블 파티션 이동, 삭제 등의 작업 시 UNUSABLE 되므로 적용 시 주의

5) LOCAL NON-PREFIXED

- 테이블 파티션에 대한 추가, 삭제 등의 작업이 빈번할 때 유용

- 이력성 데이터를 주로 관리하는 DB 환경에 효과적

- 파티션 키 칼럼이 조건절에 사용될 때 유용 (파티션 PRUNING)

- 파티션 키 칼럼에 대한 검색조건이 없으면 인덱스 파티션 전체를 액세스 하는 비효율

- 범위검색 조간으로 자주 사용되면 이 방식이 유용 (인덱스 매칭도가 더 중요!)