데이터베이스
강의 노트

제 15 회차
물리적 설계 단계

❖ 학습목표

- 릴레이션 스키마를 내부 스키마로 변환할 수 있다.
- 성능 향상을 위해 물리적 구조를 변경하는 방법에 대해 설명할 수 있다.
- 트랜잭션 분석 자료의 활용에 대해 설명할 수 있다.
- 인덱스를 설계하는 3단계 절차를 나열할 수 있다.

❖ 학습내용

- 물리적 모델링
- 트랜잭션 분석과 인덱스 설계

물리적 모델링

1. 물리적 모델링 개요
2. 내부 스키마 작성 방법
3. 성능 향상과 물리적 구조 변경

1. 물리적 모델링 개요

1) 물리적 모델링

물리적 모델링의 주요 업무

논리적 스키마(릴레이션 스키마)를 기초로, DBMS의 특성과 구현 환경 등을 고려,
내부 스키마(물리적 데이터 구조) 정의

속성 즉, 테이블 칼럼의 데이터 타입과 크기, 제약조건 등 정의

데이터 사용량의 예측을 통해서 역정규화 및 인덱스 설계 작업 수행
(역정규화는 논리적 모델링 단계에서 수행할 수 있음)

물리적 모델링의 특징

하나의 릴레이션이 물리적으로 하나 이상의 테이블이 될 수 있음

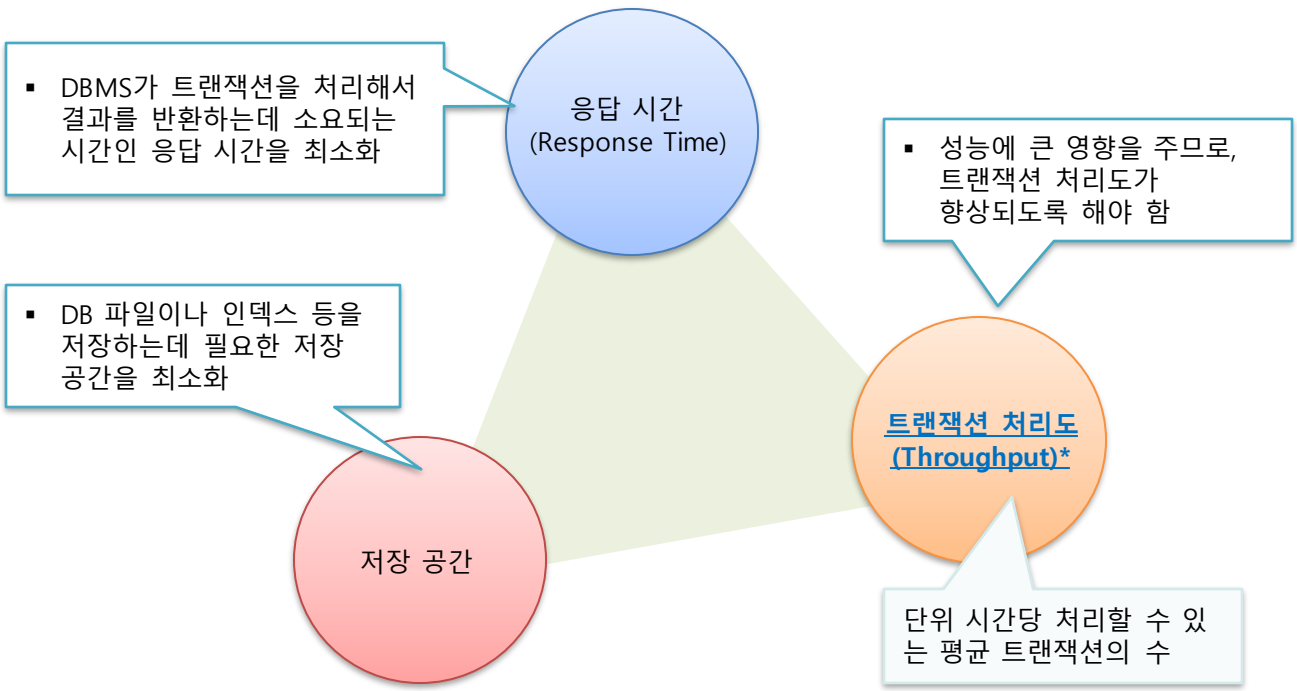
논리적 모델링을 기초로 시스템 환경(하드웨어, 운영체제, 디스크 용량, 네트워크, DBMS 제품 등)을 고려해서, 성능 향상을 목적으로 물리적 모델링을 수행

시스템 환경이 변경되면 물리적 모델링도 변경됨

성능을 고려해서 통계 테이블이 추가되거나, 백업이나 복제 용도의 테이블이 추가될 수 있음

1. 물리적 모델링 개요

2) 물리적 모델링 시 고려사항



1. 물리적 모델링 개요

2) 물리적 모델링 시 고려사항

여기서 잠깐!

시스템 성능에 영향을 미치는 요소들

DBMS의 선정

업무의 특성에 맞는(즉, 개념적 모델의 특성을 가장 잘 표현할 수 있는 논리적 데이터 모델을 기반으로 하는) DBMS 제품을 선정해야 함
(일반적으로 업무의 특성보다는 가격이나 영업 전략에 따라 선정되는 경우가 많음)

하드웨어 자원

DB를 적용할 업무 시스템의 규모(사용자 수, 트랜잭션 양 등)에 적합한 하드웨어 자원(CPU, 디스크, 메모리 등)을 확보해야 함

데이터 전송량

통신을 통한 데이터 전송량이 많을수록 시스템 성능이 나빠짐
(통신량을 최소화할 수 있는 최적화된 프로그램 개발이 요구됨)

로깅(Logging)*

트랜잭션 처리 양과 형태, 위치 등을 고려한 로깅 간격 및 위치 결정에 따라 성능이 달라짐

기타

업무 프로그램의 설계 및 코딩, 데이터 접근 방식 등에 따라 성능이 달라짐

2. 내부 스키마 작성 방법

1) 내부 스키마

내부 스키마란?

논리적 스키마(릴레이션 스키마)와 무결성 제약조건 정의를 기초로 생성한 저장 레코드 양식

- 릴레이션을 테이블로 표현
- 속성은 칼럼으로 표현
- 관련성은 외부키로 표현

데이터 타입과 크기, 제약조건 등 포함

2) 내부 스키마 작성 사례

예) 학사 관리 DB를 위한 내부 스키마 작성

1 학사 관리 DB의 릴레이션 스키마와 무결성 제약조건 정의

교수(교수번호, 교수이름, 전공, 학과)
학생(학번, 이름, 주소, 학년, 교수번호(FK))
교과목(교과목번호, 교과목명, 학점, 교수번호(FK))
수강하다(학번(FK), 교과목번호(FK), 성적)

No.	무결성 제약조건	제약 유형	관련 릴레이션
1	교수이름, 전공, 학과는 널 값을 가질 수 없다.	NOT NULL	교수
2	이름과 학년은 널 값을 가질 수 없다	NOT NULL	학생
3	교과목명, 학점은 널 값을 가질 수 없다.	NOT NULL	교과목
4	교과목명은 유일한 값을 가져야 한다.	UNIQUE	교과목

2. 내부 스키마 작성 방법

2) 내부 스키마 작성 사례

예) 학사 관리 DB를 위한 내부 스키마 작성

2 내부 스키마 작성

교수

교수(교수번호, 교수이름, 전공, 학과)

No.	속성	칼럼이름	데이터타입	크기	NULL 허용	키
1	교수번호	prof_id	NUMBER	4	N	PK
2	교수이름	name	VARCHAR2	20	N	
3	전공	major	VARCHAR2	20	N	
4	학과	dept_num	NUMBER	3	N	

학생

학생(학번, 이름, 주소, 학년, 교수번호(FK))

No.	속성	칼럼이름	데이터타입	크기	NULL 허용	키
1	학번	std_id	VARCHAR2	8	N	PK
2	이름	name	VARCHAR2	20	N	
3	주소	addr	VARCHAR2	30		
4	학년	grade	VARCHAR2	1	N	
5	교수번호	prof_id	NUMBER	4		FK

2. 내부 스키마 작성 방법

2) 내부 스키마 작성 사례

예) 학사 관리 DB를 위한 내부 스키마 작성

2 내부 스키마 작성

교과목

교과목(교과목번호, 교과목명, 학점, 교수번호(FK))

No.	속성	칼럼이름	데이터타입	크기	NULL 허용	키	기타
1	교과목번호	sub_num	VARCHAR2	6	N	PK	
2	교과목명	sub_name	VARCHAR2	20	N		UNIQUE
3	학점	credit	NUMBER	1	N		DEFAULT 값 = 3
4	교수번호	prof_id	NUMBER	4		FK	

수강하다

수강하다(학번(FK), 교과목번호(FK), 성적)

No.	속성	칼럼이름	데이터타입	크기	NULL 허용	키	기타
1	학번	std_id	VARCHAR2	8	N	PK, FK	
2	교과목번호	sub_num	VARCHAR2	6	N	PK, FK	
3	성적	score	NUMBER	3			

3. 성능 향상과 물리적 구조 변경

1) 물리적 구조 변경 방법

테이블 재정의	칼럼 중복
① 성능향상을 위해 **역정규화**를 통해서 테이블 구조를 재정의 - 데이터 무결성이 저해되지 않아야 함 - 정규형 모델에 심각한 성능 저하를 유발하는 요건이 존재하는 경우에만 역정규화 함 ② 중복된 정보를 포함하는 **테이블 추가** ③ 테이블을 **수직 또는 수평 분할**	① 조인이 빈번하게 발생하는 경우, **조인 조건에 해당하는 속성을 중복시킴** - 자주 사용되는 검색 조건이 여러 테이블에 분산되어 있는 경우, 검색 조건에 포함되는 칼럼을 다른 테이블에 복사 ② 접근 경로를 단축하기 위해 부모 테이블(주개체)에 자식 테이블(종속 개체)의 **특정 칼럼의 집계(통계) 값을 저장하는 칼럼을 추가** ③ 접근 경로를 단축하기 위해 자식 테이블에 부모 테이블의 **칼럼을 복사해서 중복시킴** ④ 연산된 결과를 주로 사용하는 경우, **연산된 값을 저장하는 칼럼을 추가**

3. 성능 향상과 물리적 구조 변경

2) 테이블 추가

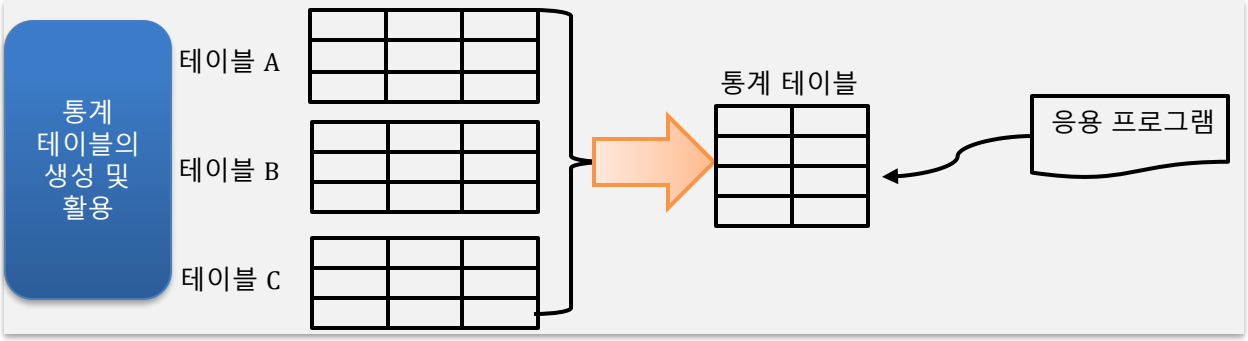
통계 테이블 추가

자주 사용되는 통계(집계) 데이터가 여러 테이블에 산재되어 있는 데이터를 기초로 생성되는 경우

▶ 별도의 통계 테이블을 생성하여 추가

기존의 응용 프로그램들이 통계 테이블을 활용할 수 있어야 함

추가한 통계 테이블을 기초로 생성된 통계 데이터를 포함하는 또 다른 통계 테이블이나 간단한 조인으로 생성할 수 있는 통계 테이블은 추가하지 않음



부분 테이블 추가

대용량 테이블의 특정 부분만 주로 사용되는 경우, 해당 부분만 복사해서 별도의 테이블을 생성

예)

접수

접수번호	고객번호	고객명	접수일자	접수방법	접수확인일자	접수확인자	납부일자	납부방법	. . .
. . .									
. . .									

접수확인

접수번호	고객번호	접수일자	접수방법	접수확인일자	접수확인자
. . .					
. . .					

3. 성능 향상과 물리적 구조 변경

2) 테이블 추가

이력 테이블 추가

하나의 업무 단위가 시간의 흐름에 따라 반복적으로 발생하거나 변경되는 이력 또는 진행 이력이 필요한 경우

▶ 이력 테이블 추가

3) 테이블 분할

수직 분할

특정 칼럼을 분리해서 별도의 테이블로 관리함으로써 물리적인 I/O 양을 감소시켜 검색 성능 향상시킴

1. 검색 위주의 칼럼과 갱신 위주의 칼럼으로 구분되는 경우
2. 특별히 빈번하게 검색되는 칼럼이 있는 경우
3. 특정 칼럼의 크기가 매우 큰 경우
4. 특정 칼럼에 보안이 필요한 경우

3. 성능 향상과 물리적 구조 변경

3) 테이블 분할

수직 분할

검색 위주의 칼럼과 갱신 위주의 칼럼으로 구분되는 경우

예) 회원정보 가운데 **접속 정보는 자주 갱신**되고, 나머지 정보는 **주로 검색**만 되므로 이들 칼럼을 수직 분할해서 2개의 테이블로 만듦

주로 검색만 됨

자주 갱신됨

회원

회원번호	회원명	주민등록번호	ID	PW	등급	사진	...	최종로그인일시	최종로그아웃일시
. . .									
. . .									

회원정보

회원번호	회원명	주민등록번호	ID	PW	등급	사진	...
. . .							
. . .							

회원접속정보

회원번호	최종로그인일시	최종로그아웃일시
. . .		
. . .		

특별히 빈번하게 검색되는 칼럼이 있는 경우

예) 회원이 로그인할 때마다 **회원번호와 id, 패스워드(pw)가 자주 검색**되므로 이들 칼럼을 수직 분할해서 2개의 테이블로 만듦

특히 자주 검색됨

회원정

회원번호	회원명	주민등록번호	ID	PW	등급	사진	...
. . .							
. . .							

회원정

회원번호	회원명	주민등록번호	등급	사진	...
. . .					
. . .					

회원인증정보

회원번호	ID	PW
. . .		
. . .		

3. 성능 향상과 물리적 구조 변경

3) 테이블 분할

수직 분할

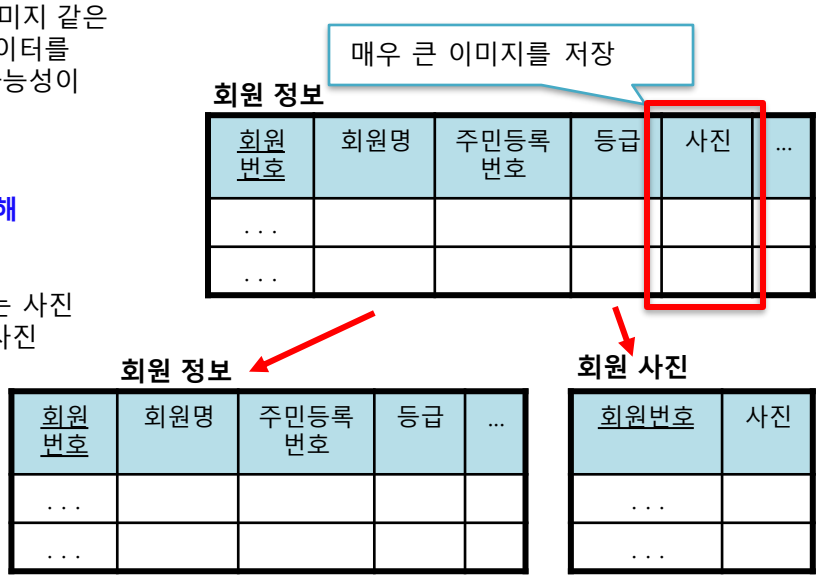
특정 칼럼의 크기가 매우 큰 경우

- 테이블이 매우 큰 문자열이나 이미지 같은 LOB(Large Object)* 형태의 데이터를 포함하는 경우, 성능이 저하될 가능성이 있으므로 분할함

- 특히 **백업(Backup)이나 복원(Recovery)의 효율성을 위해 분할**해서 관리함

예) 회원의 사진 이미지를 저장하는 사진 칼럼만 분리해서 별도로 회원사진 테이블에 저장해서 관리함

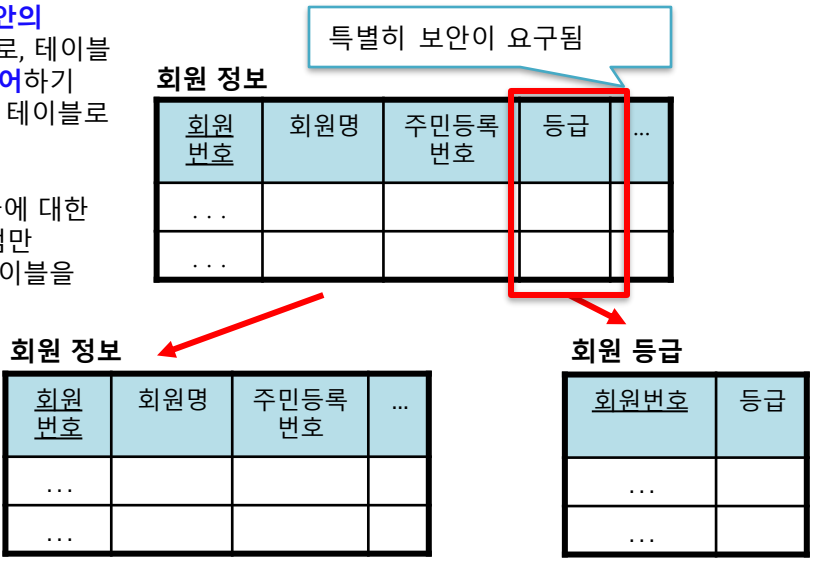
LOB(Large Object) : 대용량 데이터를 저장하고 관리하기 위해서 오라클이 제공하는 데이터 타입으로, 4GB까지 지원함



특정 칼럼에 보안이 필요한 경우

- DBMS에서 제공하는 **데이터 보안의 단위가 객체(Object) 단위**이므로, 테이블 내 **특정 칼럼에 대한 보안을 제어**하기 위해서 칼럼을 분리해서 별도의 테이블로 관리함

예) 회원 정보 가운데 회원의 등급에 대한 보안이 필요한 경우, 등급 칼럼만 분리해서 별도의 회원등급 테이블을 만듦



3. 성능 향상과 물리적 구조 변경

3) 테이블 분할

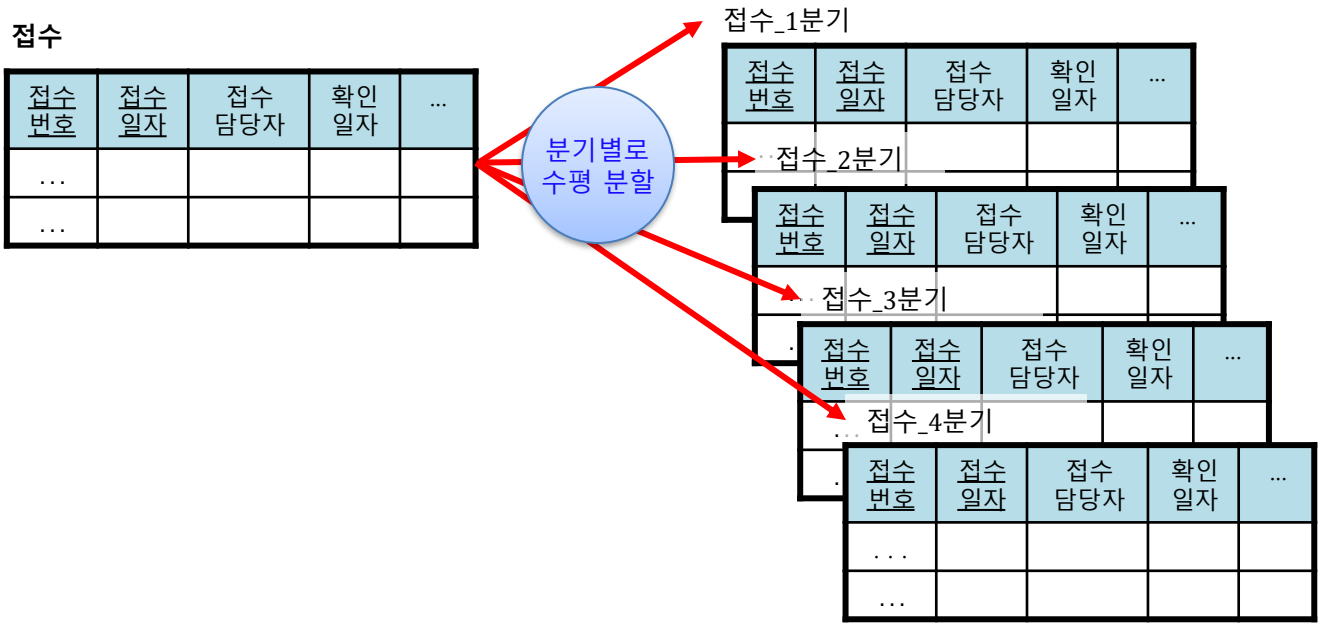
수평 분할

테이블의 행을 기준으로 분할해서 별도의 테이블로 관리함으로써 물리적인 I/O 양을 감소시켜 검색 성능 향상시킴

1. 한 테이블에 데이터가 너무 많고, 특정 레코드 그룹에만 자주 접근하는 경우 적용
2. 분할된 각 테이블은 다른 디스크에 위치시켜서 디스크의 효율성을 극대화
3. DBMS가 제공하는 테이블 파티셔닝(Partitioning)* 기능을 이용

수평 분할 예시

예) 접수에 대한 처리가 **기본 키(접수번호와 접수일자)의 구간(분기별)**에 따라 큰 차이가 있는 경우, 기본 키에 따라 수평 분할해서 별도의 테이블로 분리해서 관리함



3. 성능 향상과 물리적 구조 변경

4) 칼럼 중복

동일 칼럼 추가

검색 성능을 향상시키기 위해 자주 검색되는 칼럼을 중복시킴

데이터 무결성을 보장하기 위해서, 중복되는 칼럼은 변경이 거의 일어나지 않아야 함

예) 학생이 소속된 학과이름을 자주 검색해야 하는 경우, 학과 테이블과의 조인을 통해서 알 수 있지만 검색 효율을 위해서 학생 테이블에 '학과명' 칼럼을 중복시킴

학생

학번	이름	주민등록번호	...	학과번호 (FK)	학과명
...					
...					

동일 칼럼 중복

학과

학과번호	학과명	전화번호	사무실	...	학과장 (FK)
...					
...					

파생(Derived) 칼럼 추가

다른 칼럼 값들을 기초로 계산된 합계, 평균, 개수 등의 파생된 값이 많이 활용되는 경우

▶ 파생 칼럼을 추가하면 검색 성능이 향상됨

데이터 무결성을 보장하기 위해서 파생된 값의 기초가 되는 원래 칼럼의 값이 변경되는 경우

▶ **파생된 값도 연쇄적으로 변경되도록 정의**

예) 학생의 평점은 수강한 각 교과목의 성적을 기초로 계산할 수 있지만, 검색 효율을 위해서 '평점'이라는 파생 칼럼을 추가함

학생

학번	이름	주민등록번호	...	학과번호 (FK)	평점
...					
...					

파생 칼럼 추가

수강

학번 (FK)	교과목번호 (FK)	이수학기	성적	...
...				...
...				...

3. 성능 향상과 물리적 구조 변경

4) 칼럼 중복

기본 키의 일부에 해당하는 칼럼 추가

기본 키가 여러 가지 정보를 포함하고 있는 경우

▶ 그 일부가 자주 사용되면 분리해서 별도의 칼럼을 추가

예) 학번에 학과 정보가 포함되어 있지만, 학과별로 처리할 일이 많은 경우 '학과번호' 칼럼을 별도로 추가함

학생

학번	이름	주민등록번호	...	학과번호(FK)	학과번호
…					
…					

기본 키의 일부에 해당하는 칼럼 추가

학번구성

2012 36 185

입학년도(4자리) 학과(2자리) 개인번호 (3자리)

트랜잭션 분석과 인덱스 설계

1. 트랜잭션 분석
2. 인덱스 설계
3. 인덱스 설계 절차

1. 트랜잭션 분석

1) 트랜잭션

트랜잭션(Transaction)이란?

- 하나 이상의 SQL문으로 구성된 하나의 논리적인 작업 단위
- 하나의 트랜잭션은 하나 이상의 DML 명령문 또는 하나의 DDL이나 DCL 명령문으로 구성

트랜잭션 분석 대상

DB의 검색, 추가, 삭제, 변경 작업이 발생하는 단위 프로세스를 중심으로 트랜잭션의 양과 빈도수

2) 트랜잭션 분석 자료의 활용

인덱스 설계의 참조 자료

인덱스는 DB 성능에 큰 영향을 주므로, 트랜잭션 분석을 통해서 **과부하가 발생하는 칼럼을 파악**해서 인덱스 생성 여부를 결정

예) 온라인 쇼핑몰을 위한 DB 구축 단계에서 인덱스 생성을 위해 트랜잭션을 분석함
- 고객의 상품 주문을 처리하는 업무가 가장 중요하므로 관련 트랜잭션을 분석함
- 한 명의 고객이 평균 5개의 상품을 주문하며, 일일 평균 500건의 주문이 있다고 가정함

고객이 상품을 주문할 때 '상품명'을 선택하게 되며, 따라서 상품 테이블에서 상품명으로 검색하는 횟수가 1일 약 2,500번 발생하므로 '상품명' 칼럼에 인덱스 생성을 고려함

트랜잭션	순서	관련 테이블	관련 칼럼	DML 유형	DML 수행 횟수	트랜잭션 발생 주기
고객이 상품을 주문한다.	1	고객	고객번호, 이름	SELECT	500	일
	2	주문	주문번호, 고객번호, 주문일자	INSERT	500	
	3	상품	상품번호, 상품명, 단가, 재고량	SELECT	2,500	
	4	주문목록	주문번호, 상품번호, 주문수량	INSERT	2,500	

1. 트랜잭션 분석

2) 트랜잭션 분석 자료의 활용

DB 용량 산정의 기초 자료

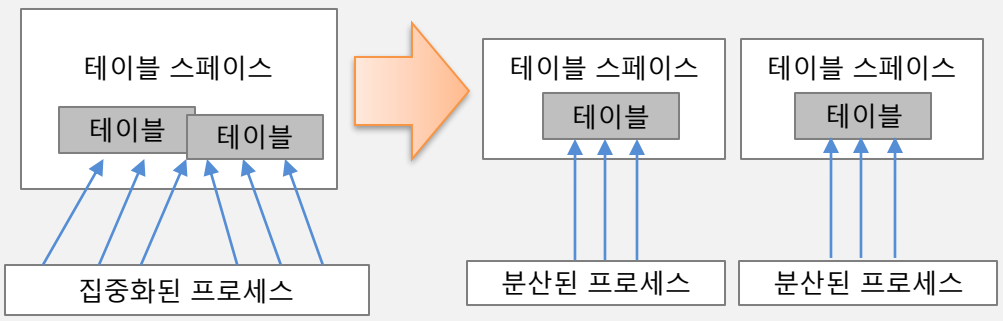
테이블과 인덱스, 테이블 당 **트랜잭션 양과 보존 기간 등을 고려**해서 DB가 저장될 디스크의 용량을 산정

예) 직원, 승진, 급여 테이블의 용량 산정하기

개체명	테이블명	행 길이	초기 데이터 수	트랜잭션 발생 주기	평균 발생 수	보존 기간	테이블 용량
직원	emp	120B	1000명	년	150명	20년	200M
승진	promo	80B	20개	학기	20	10년	20M
급여	salary	200B	10,000개	월	1000개	10년	500M
. . .							

디스크 구성 전략의 근거 자료

트랜잭션이 과도하게 집중되는 테이블이 있으면 각각 다른 **테이블 스페이스(Table Space)***에 배치해서 디스크 I/O를 분산시킴



2. 인덱스 설계

1) 인덱스

인덱스(Index)란?

포인터를 사용해서 행(Row)의 검색을 촉진할 수 있는 DB 객체

인덱스가 DB 성능에 큰 영향을 주므로, **트랜잭션의 양과 데이터 분포, 사용 빈도 등을 고려**해서 적절한 수의 인덱스를 설계

2) 인덱스의 특징

1. 테이블 행에 대한 직접적이고 빠른 접근을 제공
2. 인덱스는 행을 식별할 수 있도록 칼럼의 값과 행의 논리적인 주소(RowId)로 구성되며, 별도의 저장공간에 저장
3. 인덱스는 DBMS에 의해서 자동으로 생성되거나, 사용자에 의해 명시적으로 생성될 수 있음
4. 인덱스를 생성하면 일반적으로 물리적인 디스크 I/O가 감소되지만, 인덱스를 너무 많이 생성하면 오히려 DML 처리 효율을 저하시킴
5. 인덱스는 DBMS에 의해서 자동으로 사용되고 유지됨
6. 인덱스는 테이블과는 논리적, 물리적으로 독립적임
7. 기본 테이블에 영향을 주지 않고 생성하거나 제거할 수 있음
8. 기본 테이블을 제거하면, 인덱스도 자동으로 제거됨

2. 인덱스 설계

3) 인덱스 선정 시 고려사항

데이터 양이 적은 테이블은 인덱스를 생성하는 것보다 전체 테이블 스캔(Full Table Scan: FTS)*이 더 효율적임

인덱스가 많으면 검색 속도는 향상되지만 갱신 처리 시에 오버헤드(Overhead)가 발생

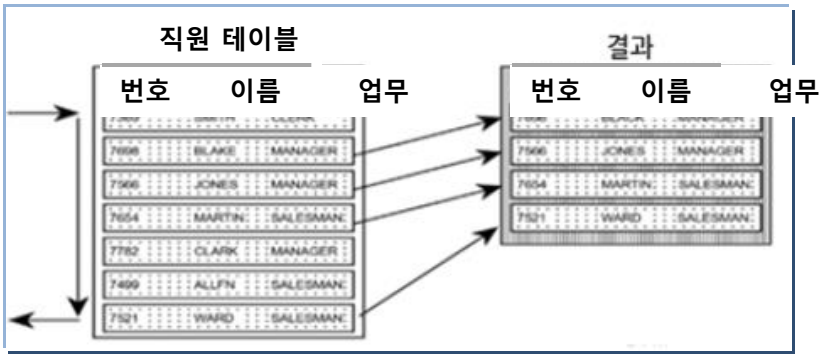
검색(SELECT문)과 나머지 DML(INSERT, DELETE, UPDATE)의 비용은 1 : 4 정도이므로 검색 효율만을 위해서 지나치게 많은 인덱스를 생성하는 것은 좋지 않음

검색 대상이 넓은 경우, 즉 전체 데이터 가운데 상당수의 데이터를 검색하는 경우, 인덱스를 사용하는 것은 오히려 비효율적임

전체 테이블 스캔과 인덱스 사용 접근의 차이점

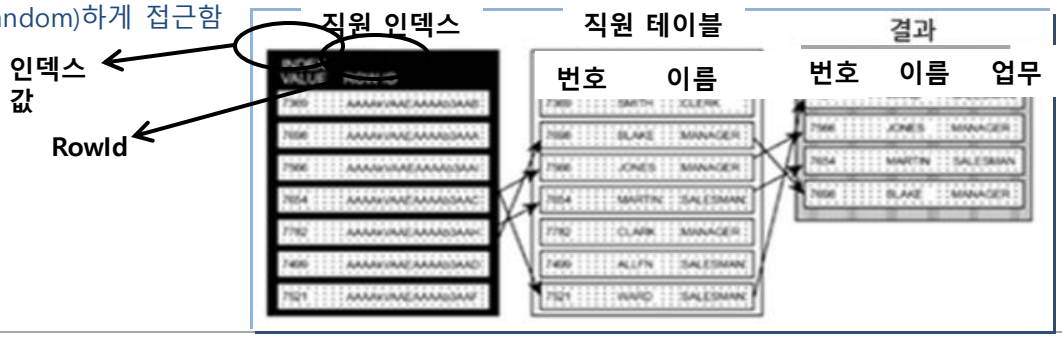
1) 전체 테이블 스캔 (Full Table Scan: FTS)

- 테이블에 존재하는 모든 행을 **순차적**으로 접근함



2) 인덱스 사용 접근

- 먼저 인덱스를 읽어서 인덱스의 논리적인 주소(RowId) 값을 가져와서 테이블에 랜덤(Random)하게 접근함



2. 인덱스 설계

4) 인덱스 선정 기준

분포도가 좋은 칼럼은 인덱스를 단독으로 생성해서 활용도를 향상시킴

SQL문의 WHERE절, ORDER BY절, GROUP BY절에 자주 사용되는 칼럼에 인덱스 생성

기본 키에 대해서는 인덱스가 자동으로 생성되므로 별도로 생성할 필요없음

자주 조합되어 사용되는 칼럼의 경우, 결합 인덱스 생성

테이블 간의 조인 조건으로 자주 사용되는 인덱스 생성

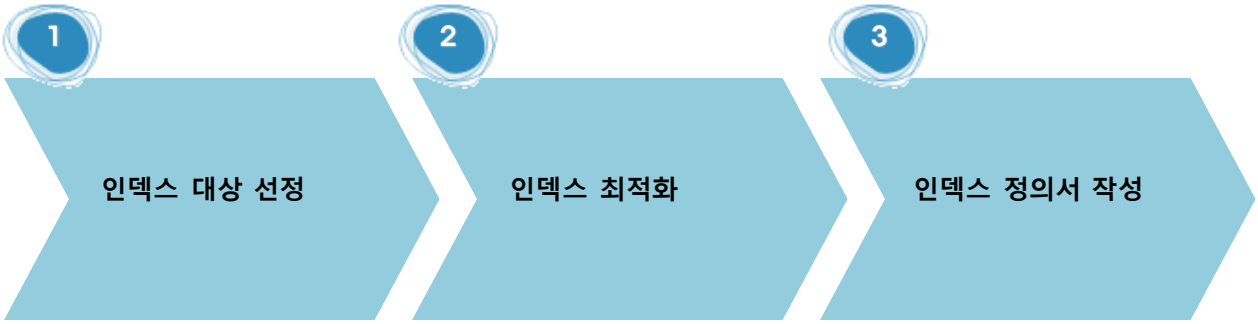
실제로 사용될 접근을 기초로 인덱스 선정

5) 인덱스로 부적합한 대상

1. 수정이 자주 발생되는 칼럼
2. 분포도가 좋지 않은 칼럼 (즉, 데이터 종류가 많지 않은 칼럼)
3. 검색되는 행이 전체 행의 대부분에 해당하는 경우
4. 검색 조건으로 자주 사용되지 않는 칼럼

3. 인덱스 설계 절차

1) 인덱스 설계의 3단계 절차



2) 인덱스 대상 선정

먼저, 대상 테이블을 선정한 다음,

1. SQL문에서 조인 조건(WHERE절)으로 자주 사용되거나, ORDER BY절, GROUP BY절 등에서 자주 사용되는 칼럼을 선정

2. 자주 사용되는 칼럼 가운데 평균 분포도(Selectivity)가 10~15% 이내인 칼럼을 선정

분포도(%) = 1 / 칼럼 값의 종류 * 100
= 해당 칼럼 값을 갖는 행의 평균 수/ 테이블의 총 행의 수 * 100

3. 자주 조합되어 사용되는 칼럼의 경우, 결합 인덱스를 생성

3. 인덱스 설계 절차

2) 인덱스 대상 선정

분포도(Selectivity)

(1) 분포도란?

어떤 칼럼 값을 기준으로 검색했을 때 전체 데이터 가운데 얼마 만큼의 데이터가 선택되는지를 나타내는 것으로, 분포도(%) 값이 작을수록 분포도가 좋다고 함

(2) 분포도 계산식

분포도(%) = 1 / 칼럼 값의 종류 * 100
= 해당 칼럼 값을 갖는 행의 평균 수/ 테이블의 총 행의 수 * 100

(3) 분포도 계산 및 분석 예제

1) 기준 칼럼 : 학번(기본 키), 학생 테이블의 총 행 수 : 1,000개
→ 분포도 = 1 / 1000 * 100 (학번 칼럼 값의 종류는 1,000개)
= 1 / 1000 * 100 = 0.1%

- 기본 키는 모든 데이터가 다른 값을 가지므로, 하나의 기본 키 값에 대해 1,000개 데이터 가운데 어떤 데이터가 선택될 확률이 0.1%이므로 분포도가 아주 좋다고 함

2) 기준 칼럼 : 성별, 학생 테이블의 총 행 수 : 1,000개
→ 분포도 = 1 / 2 * 100 (성별 칼럼 값의 종류는 단 2개)
= 500 / 1000 * 100 = 50%

- 성별 칼럼은 남, 여 둘 가운데 하나의 값을 가지므로, 한 값에 대해 어떤 데이터가 선택될 확률이 50%이므로 분포도가 좋지 않다고 함

3) 기준 칼럼 : 전공, 학생 테이블의 총 행 수 : 1,000개
→ 분포도 = 1 / 20 * 100 (전공 칼럼 값의 종류는 10개)
= 100 / 1000 * 100 = 10%

- 전공 칼럼은 10개의 값 가운데 하나를 가지므로, 하나의 전공 값에 대해 1,000개 데이터 가운데 어떤 데이터가 선택될 확률이 10%이므로 분포도가 좋은 편이며, 전공으로 검색하는 일이 많은 경우 전공 칼럼에 인덱스를 생성하는 것이 효율적임

3. 인덱스 설계 절차

3) 인덱스 최적화

1 인덱스의 효율성 검토

① 인덱스를 생성한 칼럼은 수정이 자주 발생하지 않아야 함

② 평균 분포도가 10~15% 이내로 양호해도 분포가 일정하지 않으면 인덱스를 설정하지 않아야 함

③ 한 테이블에 인덱스 수가 5개를 초과하는 경우, 입력이나 수정, 삭제가 자주 발생하면 인덱스를 삭제해야 함

2 인덱스의 데이터타입 검토

① 칼럼에 업무 처리에 적합한 도메인이 지정되었어도, 적절한 인덱스 활용을 위해서 칼럼의 데이터 타입을 변경할 수 있음
예) 칼럼의 데이터 타입이 DATE인 경우, 실제로 시·분·초까지 처리하지 않고 날짜 단위로 처리한다면, DATE 보다 VARCHAR2(8)* 형식으로 변경함
오라클의 가변 길이 데이터 타입

② 데이터 길이가 변경되는 칼럼은 오라클의 VARCHAR2()같은 가변길이 데이터타입을 사용함

3 인덱스 정렬 및 결합 인덱스 검토

① 인덱스는 정렬 상태에 따라 순정렬(Ascending) 인덱스와 역정렬(Descending) 인덱스가 있으므로 특히 역정렬 인덱스가 필요한지 검토함

② 여러 칼럼이 동시에 사용될 때는 결합 인덱스를 생성함

③ 결합 인덱스의 경우 인덱스를 구성하는 칼럼의 순서가 성능에 큰 영향을 미치므로 **앞쪽에 오는 칼럼이 검색 대상의 범위를 줄일 수 있는 칼럼이 오도록 설계**해야 함

4 클러스터링(Clustering) 검토

① 검색 성능을 향상시킬 필요가 있지만 인덱스를 사용할 수 없는 경우, 클러스터링을 고려함

② 클러스터링의 경우 인덱스와는 반대로 분포도가 넓은 것이 좋음

③ 클러스터링도 검색 성능은 향상되지만 나머지 DML의 성능이 저하되므로 신중히 결정함

④ 대량의 데이터를 처리하는 트랜잭션이 많은 경우 클러스터링을 사용하지 않는 것이 좋음

3. 인덱스 설계 절차

4) 인덱스 정의서 작성

인덱스에 대한 정보를 포함하는 일종의 인덱스 명세서를 작성

예) 학사 관리 DB의 인덱스 정의서

학사 관리 DB의 인덱스 정의서							
개체 타입명	테이블명	인덱스명	칼럼명	데이터 타입	인덱스 유형 (UNIQUE 여부)	정렬	키 여부
교수	prof	prof_id_idx	prof_id	NUMBER(4)	Yes	ASC	PK
		prof_name_idx	name	VARCHAR2(20)	No	ASC	
학생	student	std_id_idx	std_id	VARCHAR2(8)	Yes	ASC	PK
		std_name_idx	name	VARCHAR2(20)	No	ASC	
교과목	subject	sub_num_idx	sub_num	VARCHAR2(6)	Yes	ASC	PK
		sub_name_idx	sub_name	VARCHAR2(20)	No	ASC	
. . .							

쉼터

코이 잉어

일본산 '코이 잉어'는 독특한 특징을 가진 물고기이다.

작은 어항에 넣어두면 5~8 cm가량 자라는 물고기인데,
수족관이나 연못에 넣어두면 15~25 cm까지 자란다고 한다.
심지어 강물에 방류하면 90~120 cm까지 자랄수도 있다고 한다.

한계를 어디에 두느냐에 따라 어디까지 성장할 수 있는지가
제한되는 사람의 모습과 비슷하다고 할 수 있다.