데이터베이스
강의 노트

제 16 회차
물리적 설계 단계 심화 및 DB 구현

❖ 학습목표

- 뷰(View) 활용의 장점을 나열할 수 있다.
- 디스크 용량 설계의 목적을 설명할 수 있다.
- 테이블 분산 방법에 대해 설명할 수 있다.
- DDL을 이용해서 DB를 생성할 수 있다.

❖ 학습내용

- 뷰와 디스크 용량 설계
- 분산 설계
- DB 구현 및 테스트 단계

뷰와 디스크 용량 설계

1. 뷰 설계
2. 오라클의 논리적 저장구조
3. 디스크 용량 설계

1. 뷰 설계

1) 뷰 설계

뷰(View)란?

- 하나 이상의 테이블에 포함된 데이터의 부분 집합으로 구성되는 논리적인 테이블 즉, 가상의 테이블(Virtual Table)
- 저장공간은 없지만 테이블과 거의 유사하게 사용할 수 있는 하나의 독립된 DB 객체*

테이블, 뷰, 인덱스, 시퀀스 등과 같이 서로 다른 정보를 포함하고 있는 데이터를 참조하기 위한 논리적 구조

2) 뷰의 특징

뷰를 사용하면 복잡한 테이블 구조를 단순화시켜서 효율적인 검색이 가능

하나 이상의 테이블을 기초로 뷰를 생성

테이블뿐만 아니라 다른 뷰를 기초로 생성

뷰 자체는 데이터를 직접 포함하지 않지만, 창문 역할을 하는 뷰를 통해서 데이터의 검색 및 수정이 가능

열 별칭을 사용해서 생성된 뷰에 대해서는 열 별칭을 사용한 조작만 가능

뷰는 기초가 되는 테이블의 인덱스를 사용하므로, 인덱스 칼럼은 함부로 가공해서는 안됨

1. 뷰 설계

3) 뷰 활용의 장점

보안 관리 지원

인증되지 않은 DB 접근을 방지하기 위해서 데이터 접근을 제한하고 모니터링하는 기능

- 사용자가 특정 테이블의 데이터 가운데 뷰로 정의된 특정 부분만 접근할 수 있도록 제한하여 보안 관리가 가능
- 조건에 따라 데이터에 접근하는 사용자 그룹을 분류해서, 각각 동일한 테이블의 다른 뷰를 기초로 데이터를 조작하도록 제한

사용 편의성 제공

- 다중 테이블을 기초로 뷰를 생성하면 테이블 조인이 불필요하게 되므로, 복잡한 질의를 단순한 질의로 변환

데이터 독립성 제공

- 테이블이 변경되어도 뷰는 그대로 유지할 수 있으므로, 임시 사용자와 응용 프로그램에 대한 데이터 독립성을 제공

4) 뷰의 종류

단순 뷰(Single View)

단 하나의 테이블만을 기초로 생성된 뷰

▶ 표현식 등에 의해 데이터가 조작된 경우를 제외하면, 뷰를 통한 모든 DML 연산의 수행이 가능

복합 뷰(Complex or Join View)

다중 테이블을 기초로 생성된 뷰

▶ 데이터 그룹핑 또는 그룹 함수를 사용해서 뷰를 생성
▶ 뷰를 통한 모든 DML이 항상 가능한 것은 아님

1. 뷰 설계

4) 뷰의 종류

뷰 생성

▶ 뷰 생성 방법

CREATE VIEW 명령문에 서브쿼리를 이용해서 생성하고, 뷰가 생성된 후 뷰 이름과 뷰 정의는 데이터 사전의 USER_VIEWS 테이블에 저장

▶ CREATE VIEW 명령의 형식

서브쿼리를 수행해서 가져온 열(Column)들만으로 뷰를 생성

```
CREATE  [FORCE | NOFORCE]
              VIEW  뷰이름  [(열별칭1[,  열별칭2,  . . .]) ]
 AS  서브쿼리
 [WITH CHECK OPTION [CONSTRAINT  제약이름]]
 [WITH READ ONLY];
```

▶ 옵션 설명

- FORCE : 기본 테이블의 존재 여부와 무관하게 뷰를 생성
- NOFORCE : 기본 테이블이 존재할 때만 뷰를 생성
- 열별칭: 서브쿼리에 의해 선택된 열이나 표현식에 대한 별칭을 지정
- 서브쿼리 : 뷰에 포함될 데이터를 검색하는 SELECT 문을 작성
- WITH CHECK OPTION : 뷰에 의해 접근 가능한 행만 삽입 또는 수정될 수 있음을 명시
- WITH READ ONLY : 뷰에 대해서 SELECT 만 가능하고, 다른 DML 연산은 불가능함을 명시

1. 뷰 설계

5) 뷰 및 뷰 정의서 사례

뷰 사례

사원 테이블을 기초로 생성된 사원급여_뷰와 사원평가_뷰

사원 테이블

사원번호	부서번호	이름	직책	입사일	호봉	교육참여점수	상사평가점수	동료평가점수	주소	연락처
3214	100	김철수	팀장	1998/01/01	20	10	20	18	서울	010-1111-2222
2456	200	이영호	프로그래머	2010/03/01	8	20	18	19	천안	010-2222-3333
2456	300	이영호	설계자	2005/01/01	13	15	18	18	천안	010-3333-4444
4602	100	박민희	사원	2012/09/01	5	10	15	16	서울	010-4444-5555
3722	300	김철수	팀장	2000/03/01	18	12	19	20	천안	010-5555-6666

사원급여_뷰

사원번호	이름	입사일	호봉
3214	김철수	1998/01/01	20
2456	이영호	2010/03/01	8
2456	이영호	2005/01/01	13
4602	박민희	2012/09/01	5
3722	김철수	2000/03/01	18

```
CREATE  VIEW  사원급여_뷰
 AS  SELECT  사원번호, 이름, 입사일, 호봉
      FROM  사원;
```

1. 뷰 설계

5) 뷰 및 뷰 정의서 사례

뷰 사례

사원 테이블을 기초로 생성된 사원급여_뷰와 사원평가_뷰

사원평가_뷰

사원번호	부서번호	이름	직책	입사일	교육참여점수	상사평가점수	동료평가점수
3214	100	김철수	팀장	1998/01/01	10	20	18
2456	200	이영호	프로그래머	2010/03/01	20	18	19
2456	300	이영호	설계자	2005/01/01	15	18	18
4602	100	박민희	사원	2012/09/01	10	15	16
3722	300	김철수	팀장	2000/03/01	12	19	20

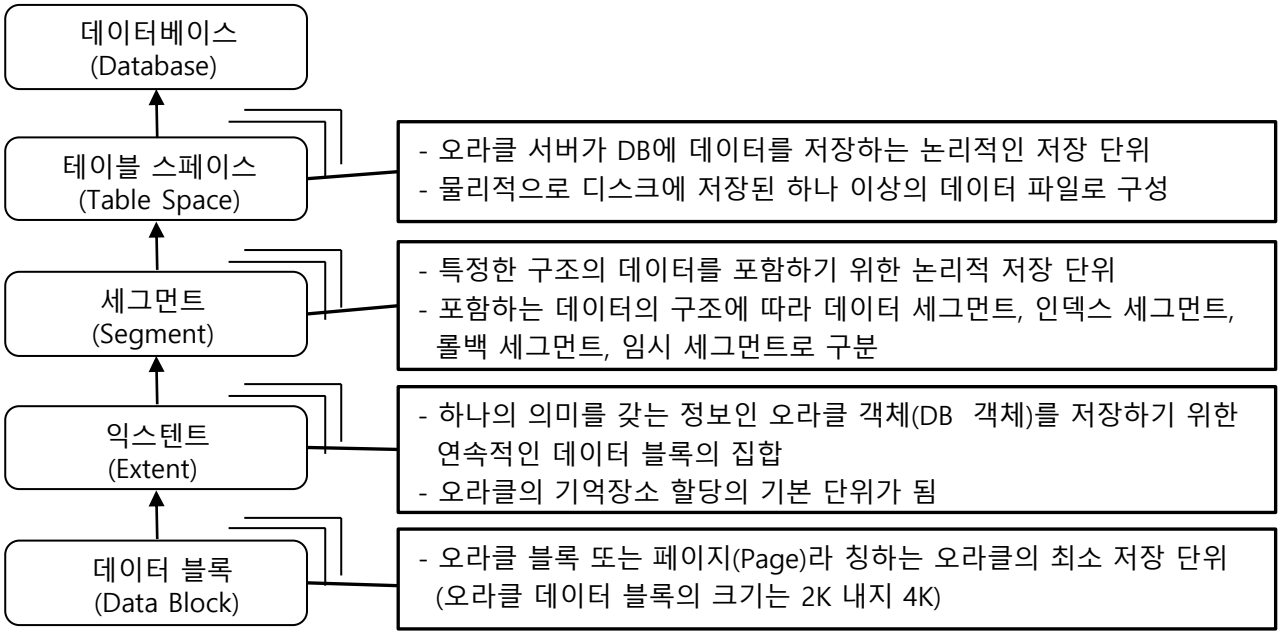
```
CREATE   VIEW  사원평가_뷰
  AS  SELECT  사원번호, 부서번호, 이름, 직책,
              교육참여점수, 상사평가점수, 동료평가점수
        FROM  사원;
```

뷰 정의서 사례

뷰명	용도	기초 테이블	칼럼	데이터 타입
사원급여_뷰	급여관리 시스템에서 사용	사원	사원번호	CHAR(4)
			사원명	VARCHAR2(20)
			입사일	CHAR(10)
			호봉	NUMBER(2)
사원평가_뷰	인사고과 시스템에서 사용	사원	사원번호	CHAR(4)
			부서번호	CHAR(3)
			사원명	VARCHAR2(20)
			입사일	CHAR(8)
			직책	VARCHAR2(20)
			교육참여점수	NUMBER(3)
			상사평가점수	NUMBER(3)
			동료평가점수	NUMBER(3)

2. 오라클의 논리적 저장 구조

1) 논리적 저장 구조의 계층 관계



2) 물리적 저장 단위

데이터 파일

- 실제 데이터를 포함하면서 디스크에 저장되는 파일
- 각 데이터 파일을 별도의 디스크에 저장 가능

OS 블록

- 실제 입출력 단위인 물리적인 I/O 블록

2. 오라클의 논리적 저장 구조

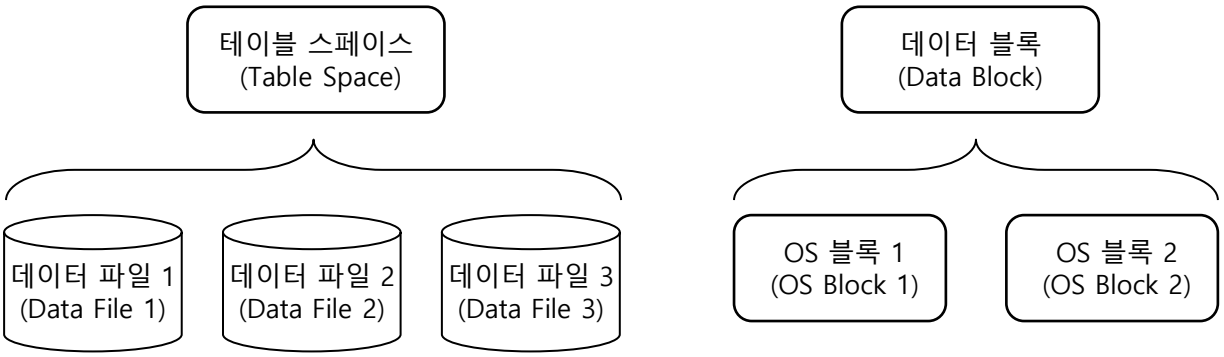
3) 논리적 저장 단위와 물리적 저장 단위의 관계

테이블 스페이스와 데이터 파일의 관계

하나 이상의 데이터 파일로 구성

데이터 블록과 OS 블록의 관계

하나 이상의 OS 블록으로 구성



4) 테이블 스페이스의 종류

시스템(System) 테이블 스페이스

- 오라클 DB가 처음 생성될 때 자동으로 생성되는, 오라클의 디폴트(Default) 테이블 스페이스
- 디폴트로 모든 사용자가 이 시스템 테이블 스페이스를 기본 테이블 스페이스 및 임시 테이블 스페이스로 사용하게 됨
- 주로 시스템 관리에 필요한 정보를 보관
- 모든 데이터 사전(Data Dictionary)의 정보와 저장 프로시저(Stored Procedure), 패키지(Package), 트리거(Trigger) 등의 정의를 저장
- 사용자 데이터도 포함할 수 있으나 바람직하지 않음

비시스템(Non-System) 테이블 스페이스

- 보다 안정적인 DB 관리를 위해서 사용하는, 시스템 테이블 스페이스와 분리된 저장 공간
- 종류
  ① 사용자 테이블 스페이스(USERS) : 사용자의 데이터를 저장
  ② 임시 테이블 스페이스(TEMP) : 정렬 등을 위해서 사용되는 임시 데이터를 저장
  ③ 롤백 테이블 스페이스(RBS) : 롤백을 위한 트랜잭션을 저장
  ④ 응용 테이블 스페이스(APP_DATA) : DB 응용 프로그램의 데이터를 저장

2. 오라클의 논리적 저장 구조

5) 테이블 스페이스 생성 및 사용자에게 할당하는 방법

DB 관리자나 권한을 가진 사용자만이 테이블 스페이스를 생성하는 것이 가능

사용자 테이블 스페이스 생성 예

```
CREATE TABLESPACE  users
DATAFILE  'c:/oracle/oradata/orcl/test01.dbf'  SIZE 10M
DEFAULT STORAGE (INITIAL 2M
                 NEXT  1M
                 MINEXTENTS  1
                 MAXEXTENTS  121
                 PCTINCREASE  0);
```

임시 테이블 스페이스 생성 예

```
CREATE TEMPORARY TABLESPACE  temp
TEMPFILE 'c:/oracle/oradata/orcl/temp01.f'
BITMAP  ALLOCATION UNIFORM SIZE  16M;
```

테이블 스페이스 삭제 예

```
DROP TABLESPACE users;
```

사용자를 생성할 때 테이블 스페이스를 지정하는 방법

- DB 사용자를 생성하려면 CREATE USER 시스템 권한이 있어야 함
- 예 : ID가 koreatech이고, password가 sunflower인 새로운 사용자를 생성하는 명령문

```
CREATE USER   koreatech
IDENTIFIED BY  sunflower
DEFAULT TABLESPACE  users
TEMPORARY TABLESPACE  temp;
```

3. 디스크 용량 설계

1) 디스크 용량 설계

디스크 용량 설계란?

물리적 DB 설계 단계에서 데이터가 저장되는 디스크 공간을 정의하는 작업

디스크 용량 설계의 목적

- 정확한 데이터량을 산정해서 디스크 사용 효율을 향상
- 업무량이 집중되는 디스크를 분리해서 집중화된 I/O 부하를 분산
- 동일한 자원에 여러 프로세스가 동시에 접근할 때 발생하는 디스크 I/O 경합을 최소화해서 데이터 접근의 성능을 향상
- DB 객체를 위한 익스텐트(Extent)*의 추가를 감소

> 데이터 저장을 위해서 연속적으로 할당된 자유 데이터 블록(Free Data Block)의 집합

2) 용량 산정 방법

디스크 용량 산정 방법

각 테이블 별로 한 행의 길이와 초기 데이터 수 및 주기별로 데이터가 추가되는 건수, 데이터 증가율 및 데이터 보존 기간, 트랜잭션 양 등을 고려해서 각 테이블의 용량을 산정

인덱스 때문에 필요한 공간 등을 고려해서 DB가 저장될 디스크의 총 용량을 산정

테이블 용량 산정 방법

각 테이블 별로 한 행의 길이와 초기 데이터 수 및 주기별로 데이터가 발생(추가)하는 건수, 데이터 증가율 및 데이터 보존 기간 등을 고려해서 각 테이블의 용량을 산정

3. 디스크 용량 설계

2) 용량 산정 방법

테이블 용량 산정 표

개체명	테이블명	행 길이	초기 데이터 수	삽입 주기	평균 삽입 수	보존 기간	테이블 용량
직원	emp	120B	1000명	년	150명	10년	100M
부서	dept	80B	20개	년	1개	10년	30M
부품	item	300B	1500개	학기	100개	10년	500M
. . .							

분산 설계

1. 분산 설계 개요
2. 테이블 분산 방법

1. 분산 설계 개요

1) 분산 DB(Distributed DB)

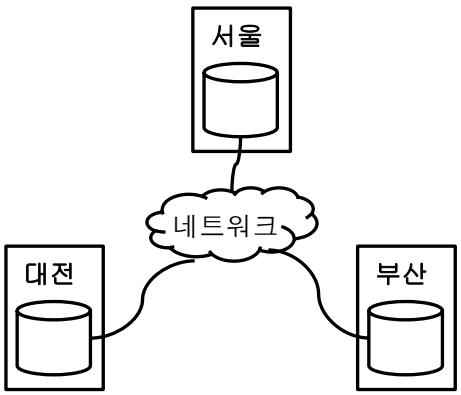
분산 DB(Distributed DB)란?

- 하나의 논리적인 DB를 물리적으로 여러 곳에 분산시킨 후, 하나의 가상 시스템으로 사용할 수 있도록 만든 DB
- DB를 연결하는 빠른 네트워크 환경을 이용해서 DB를 여러 지역에 위치시켜서 성능을 극대화시킨 DB

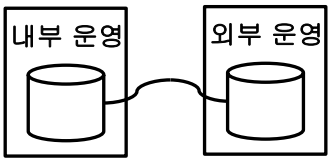
2) 분산 DB의 활용 방향

최근 업무가 매우 다양해지고 데이터 양이 기하급수적으로 증가하는 추세임

과거의 위치 중심 분산 보다는 업무의 특성을 반영해서 분산하는 것이 바람직함



[위치 중심의 분산 설계]



[업무 중심의 분산 설계]

3) 분산 DB의 장단점

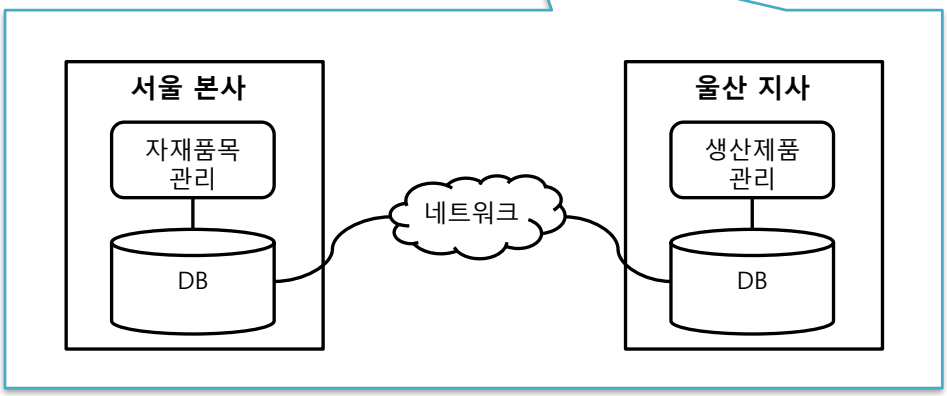
장점	단점
• 빠른 응답 속도와 통신 비용 절감 효과 • 데이터 가용성 및 신뢰성 증가 • 시스템 규모의 적절한 조절 가능 • 지역 사용자의 요구 수용 원할	• 설계 및 관리의 복잡성 및 비용 증가 • 응용 프로그램 개발 및 운용 비용 증가 • 불규칙한 응답 속도 • 통제의 어려움 • 데이터 무결성에 대한 위협 요인 증가

2. 테이블 분산 방법

1) 테이블 단위 위치 분산

테이블 구조를 변경하거나 테이블을 중복해서 생성하지 않고 테이블의 위치만 다르게 분산

위치에 따라 이용하는 테이블이 다른 경우, 테이블 별로 위치를 분산



테이블의 위치를 파악할 수 있도록 문서화

위치 \ 테이블	사원	부서	협력사	자재 품목	생산제품
서울 본사	O	O		O	
울산 지사			O		O
. . .					

2. 테이블 분산 방법

2) 수평 분할 분산

한 테이블을 특정 칼럼의 값을 기준으로 행(Row)을 분리해서 분산

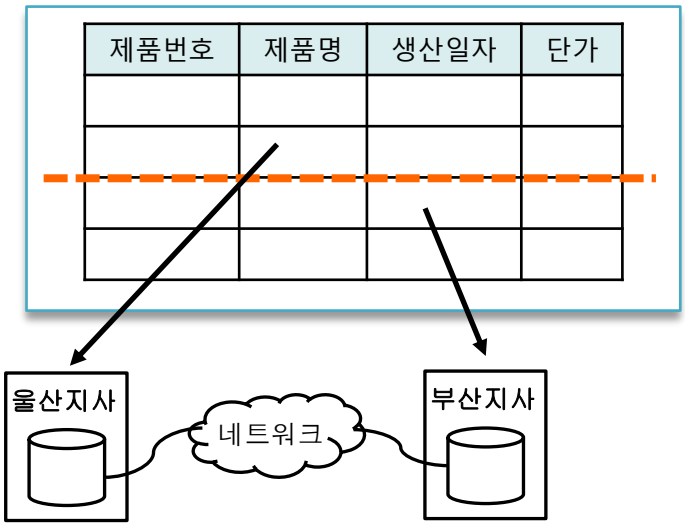
필요한 경우 다시 하나의 테이블로 통합해도 데이터 중복이 발생하지 않아야 함

한 테이블에 속하는 데이터라도 위치에 따라 접근하는 데이터 그룹이 특정 칼럼 값을 기준으로 확연하게 구분되는 경우, 한 테이블을 수평 분할해서 분산

수평 분할 분산 예

생산 제품 가운데 울산 지사에서는 '생산일자' 칼럼 값이 2012년 6월 이전인 제품을 관리하고, 부산 지사에서는 '생산일자' 칼럼 값이 2012년 6월 이후인 제품을 관리하는 경우

▶생산일자 칼럼 값을 기준으로 수평 분할함



2. 테이블 분산 방법

3) 수직 분할 분산

한 테이블의 칼럼(Column)들을 분리해서 위치를 분산(단, 기본 키 칼럼은 중복됨)

필요한 경우 다시 하나의 테이블로 통합해도 데이터 중복이 발생하지 않아야 함

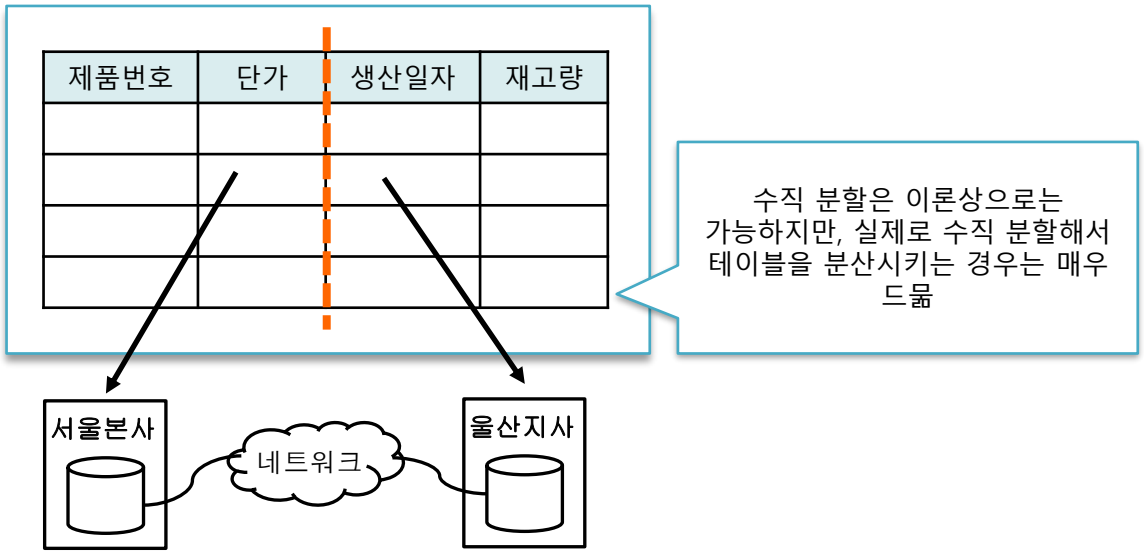
분산된 테이블을 조인해야 처리되는 업무가 있는 경우에는 수직 분할하지 않아야 함

한 테이블에 속하는 데이터라도 위치에 따라
접근하는 칼럼이 분명하게 구분되는 경우, 한 테이블을 수직 분할해서 분산

수직 분할 분산 예

생산 제품의 제품번호와 단가 칼럼 값은 서울 본사 업무에만 필요하고, 각 제품의 생산일자와 재고량 칼럼 값은 울산 지사 업무에만 필요한 경우

▶ 수직 분할해서 테이블을 분산시킴



2. 테이블 분산 방법

4) 테이블 요약(Summarization) 분산

분산 요약

여러 위치(지점)에 있는 데이터를 한 곳(본사)에서 통합해서 전체적인 요약 데이터를 생성

여러 위치의 테이블을 조인하는 오버헤드로 인해
업무에 장애가 발생할 수 있으므로 사용 시 유의

통합 요약

각 위치 별로 산출한 요약 정보를 한 곳에 취합해서 통계 정보를 쉽게 제공

여러 위치의 테이블을 조인할 필요가 없으며,
주로 야간을 이용해 데이터를 복사해서 요약 정보를 한 곳에 통합

DB 구현 및 테스트 단계

1. 구현 단계의 주요 업무
2. 테스트 단계의 주요 업무
3. DB 생성 및 초기 데이터 삽입

1. 구현 단계의 주요 업무

설계된 DB 구조를 DDL로 작성

내부 스키마를 기초로 목표 DBMS의 DDL로 DB 스키마를 작성하고 컴파일해서 공백(Empty) DB 파일을 생성

기존 DB를 새로운 DB로 변환

기존 DB가 있는 경우 변환 루틴이나 유틸리티를 사용해서 기존 데이터 파일을 변환해서 초기 데이터를 입력

DB에 초기 데이터 적재(loading)

INSERT 명령문을 활용해서 초기 데이터를 직접 삽입

트랜잭션 처리용 응용 프로그램 작성

DML 명령문을 사용해서 트랜잭션 처리용 응용 프로그램 코드를 작성

유지보수를 위한 문서화 작업

추후 유지보수를 위해 자세한 문서화 작업을 수행

2. 테스트 단계의 주요 업무

생성된 DB 구조가 설계된 DB 구조와 일치하는지 확인

응용 프로그램과 DB의 연동이 원활한지 확인

트랜잭션 유형별로 테스트 케이스를 선정하여
데이터 조작이 원활한지 확인

3. DB 생성 및 초기 데이터 삽입

1) DB 생성

CREATE TABLE 명령문을 사용해서 빈 테이블을 생성

예) 다음 내부 스키마를 기초로 교수 테이블을 생성

교수(prof) 테이블

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		
4	학과	dept_num	NUMBER	3	N	FK

DDL로 생성한 DB 스키마

```
CREATE TABLE prof (
prof_id          NUMBER(4),
name             VARCHAR2(20),
major            VARCHAR2(20),
dept_num         NUMBER(3),
CONSTRAINT   prof_id_pk PRIMARY   KEY(prof_id),
CONSTRAINY   dept_num_fk   FOREIGN   KEY(dept_num)
                  REFERENCES   dept(dept_id);
```

dept_num은 dept 테이블의 dept_id를 참조하는 외래 키임을 정의함

3. DB 생성 및 초기 데이터 삽입

2) 초기 데이터 삽입

INSERT 명령문을 사용해서 초기 데이터를 삽입

예) 앞에서 생성한 prof 테이블에 초기 데이터를 삽입

```
INSERT INTO prof (1101, '김철수', '컴퓨터', 112);
INSERT INTO prof (1108, '이영미', '전자', 113);
INSERT INTO prof (1205, '박남수', '컴퓨터', 112);
INSERT INTO prof (1228, '최영우', '전기', 115);
INSERT INTO prof (1320, '장민이', '기계', 117);
. . .
```

쉼터 닫기

너무 어렵게 살지 말자

- 관허스님

너무 어렵게 이야기하며 살지 말자
사랑하면 사랑한다고
보고 싶으면 보고 싶다고
좋아하면 좋아한다고
있는 그대로만 이야기하고 살자

너무 어렵게 셈하며 살지 말자
하나를 주었을 때 몇 개가 돌아올까
두 개를 주었을 때 몇 개를 손해 볼까
계산 없이 주고 싶은 만큼은 주고 살자

너무 어렵게 등 돌리며 살지 말자
등 돌릴 만큼 외로운 게 사람이니
등돌린 힘까지 내어 사람에게 걸어가자
절망 끝에서 건져 올린 희망이 되어

너무 인색하게 살지 말자
하늘에 떠 있는 구름도 자유롭게 지나가듯이
순리대로 인생의 흐름을 받아들이면서
잘 살아가자