데이터베이스관리시스템(DBMS)

- 정의기능(DDL)조작기능(DML)제어기능(DCL)

DDL

CREATE / ALTER / DROP

DML

CRUD

C: INSERT(데이터 생성) / create R: SELECT (데이터 조회) / read U: UPDATE(데이터 수정) /update D: DELETE(데이터 삭제) /delete



테이블 정의하기 (만들기)

create table member(

```
id varchar2(50) primary key, pwd varchar2(50), name varchar2(50), gender char(6), //남성, 여성 age number(3), birthday date, phone char(13)
```

member

| id | pwd | name | gender | age | birthday | phone |
|-------|------|------|--------|-----|------------|---------------|
| angel | 1234 | 홍길동 | 남성 | 18 | 2021-08-10 | 010-1111-2222 |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

Primary key설정(주키 설정)

CREATE TABLE dept6 AS

SELECT decode , dname FROM dept2 WHERE dcode IN (1000, 1001, 1002);

데이터형식

| character | 'Hong' |
|-----------|---------------------------|
| (문자) | 'A' |
| | '123' |
| numeric | 37 |
| (숫자) | 2.87 |
| Date (날짜) | 2012-02-12 |
| timestamp | 2009-10-23 10.11.23.00000 |

문짜형데이터

| CHAR | 고정길이 | CHAR(50) : 무조건 50 의미함 한글: 3byte |
|----------|-----------|------------------------------------|
| VARCHAR2 |) 가변길이 | VCHAR2(50) : 최대 50 의미함 |
| NCHAR | 고정길이 | 한글:2byte :한글저장시 char보다 더 효율적임 |
| CLOB | | 대용량데이터 (최대 4G) |

| NUMBER(4) | 9999 최대길이4인 숫자, |
|--------------|---|
| NUMBER(6,2) | 소수점 2자리를 포함하는 최대6자리 (소수점 2째자리에서 반올림) |
| NUMBER(6,-2) | 소수점 -2째 자리에서 반올림 하는 최대6자리 |

숫짜형데이터

| NUMBER(4) | 9999 최대길이4인 숫자, |
|--------------|---|
| NUMBER(6,2) | 소수점 2자리를 포함하는 최대6자리 (소수점 2째자리에서 반올림) |
| NUMBER(6,-2) | 소수점 -2째 자리에서 반올림 하는 최대6자리 |

날짜형데이터

| DATE | 년-월-일 |
|-----------|-------------|
| TIMESTAMP | 년-월-일:시:분:초 |

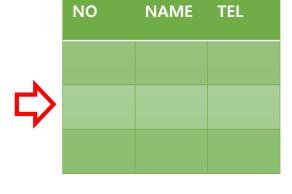
테이블 수정

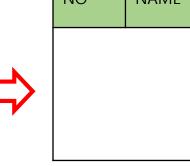
```
ALTER TABLE MEMBER MODIFY ID NVARCHAR2(50);
ALTER TABLE MEMBER ADD ( ADDR VARCHAR2(50) );
ALTER TABLE MEMBER RENAME COLUMN ID TO userID;
ALTER TABLE MEMBER DROP COLUMN ADDR;
```

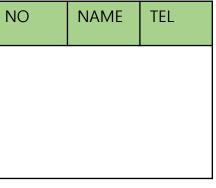
TRUNCATE 명령 TRUNCATE TABLE MEMBER;

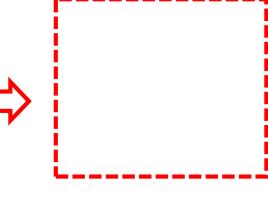
DROP 명령
DROP TABLE MEMBER;

| NO | NA ME | TEL |
|-----|----------|----------|
| 111 | AAA | 82/10/01 |
| 222 | BBB | 83/01/01 |
| 333 | CCC | 84/01/01 |









원본테이블

DELETE결과

TRUNCATE 결과

데이터를 전부 삭제하고 사용 하고 있던 공간을 반납 (테이블자체가 지워지지는 않 음)

DROP 결과

테이블 만들기 테이블 수정하기 테이블삭제하기

MEMBER테이블의 스키마가 어떻게 되니?

```
데이블만들기 (테이블구조 만들기), 구조(스키마)
CREATE TABLE MEMBER(
id varchar2(7),
name varchar2(10)
);
INSERT INTO MEMBER VALUES('aaa', 'test');
```

```
ALTER TABLE MEMBER MODIFY ID VARCHAR2(5);
ALTER TABLE MEMBER ADD (ADDR VARCHAR2(5));
ALTER TABLE MEMBER RENAME COLUMN ID TO user_id;
ALTER TABLE MEMBER DROP COLUMN ADDR;

SELECT * FROM MEMER; // 조회해 보면 id ->user_id 변경
```

DROP TABLE MEMBER;

테이블삭제하기

데이터 활용 데이터 조작(DML)

INSERT#

```
데이터 삽입:
```

INSERT INTO 테이블명(필드명1, 필드명2...) VALUES (값1, 값2 ..);

데이터 삽입:

INSERT INTO 테이블명 VALUES (값1, 값2 ..); 주의 !! 테이블의 컬럼의 순서와 개수에 맞 춰 값을 정해줘야 함 INSERT INTO member (user_id , name)
VALUES('testid' , '홍길동') ;

INSERT INTO member VALUES('testid2', '홍길순');

데이터조회(Read, SELECT)

SELECT * FROM 테이블명

SELECT 컬럼명1, 컬럼명2 FROM 테이블명

SELECT * FROM 테이블명 WHERE 컬럼명=조건

| 연산자 종류 | 설명 |
|---------------------|---|
| = | 비교 대상에서 같은 조건을 검색(<mark>같은것</mark>) |
| != , <> | 비교 대상에서 같지 않은 조건을 검색 (<mark>같지않은것</mark>) |
| > | 비교 대상에서 큰 조건을 검색 (<mark>큰것</mark>) |
| >= | 비교 대상에서 크거나 같은 조건을 검색(이상) |
| < | 비교 대상에서 작은 조건을 검색 (작은것) |
| <= | 비교 대상에서 작거나 같은 조건을 검색 (이하) |
| BETWEEN a AND b | a와 b 사이에 있는 범위 값을 모두 검색(a에서b까지) |
| IN(a,b,c) | a이거나 b이거나 c인 조건을 검색(<mark>괄호안에 있는것만</mark>) |
| LIKE | 특정패턴을 가지고 있는 조건을 검색('% , _) % :글자수에 제한이 없음, _ : 한글자만 |
| IS NULL/IS NOT NULL | NULL값을 검색/ NULL이 아닌 값을 검색 |
| A AND B | 조건이 여러 개 일때 둘 다 만족하는 값 검색 |
| A OR B | A조건이나 B조건 중 한가지라도 만족하는 값을 검색 |
| NOT A | A가 아닌 모든 조건을 검색(<mark>조건부정</mark>) |

조건절에서 사용되는 연산자

조회연습문제

- 1. 모든 정보조회 (* : 모든컬럼을 조회하는 것으로 정함)
- 2. 이름과 생일만 조회
- 3. 이름이 황가온인 학생 정보조회
- 4. 몸무게가 70키로 이상인 학생정보 조회
- 5. 키가 175 초과인 학생정보 조회
- 6. 몸무게가 65에서 70인 학생 정보 (65~70)
- 7. 주소가 텍사스, 뉴욕인 학생정보
- 8. 김씨 성을 가진 학생정보 조회
- 9. 김씨 성을 가지고 키가 177이상인 학생정보 조회
- 10. 김씨 성을 가지거나 키가 177이사인 학생정보 조회
- 11. 김씨 성을 가진 학생이 아닌 학생정보 조회
- 12. 몸무게가 NULL 인 학생정보 조회

단일행함수

1. 문자함수

```
LOWER()함수
UPPER()함수
LENGTH()함수
LENGTHB()함수
CONCAT()함수 || 연산자와 동일
SUBSTR()함수
SUBSTRB()함수
INSTR()함수
LPAD()함수
RPAD()함수
LTRIM()함수
RTRIM()함수
REPLACE()함수
```

SUBSTR(): 문자열에서 특정길의의 문자만 골라낼 때 사용 (문자의 일부만을 활용할 때 사용)

SELECT SUBSTR('abcde', 3, 2) FROM dual;

=>3번째에서 문자열길이 2만큼 (위치는 1부터 시작함)

SELECT SUBSTR('abcde', -3,2) FROM dual;

=> 마이너스(-)위치는 오른쪽에서 시작함

결과값: cd , cd

SUBSTRB(): (추출할 바이트수를 지정함)

SELECT SUBSTR('홍길동', 1,2) from dual; => 홍길

SELECT SUBSTRB('홍길동, 1,3) from dual; => 홍

LPAD() 함수 : 왼쪽을 기준으로 특정값으로 채운다

SELECT LPAD('star', 10,'*') FROM DUAL;

*****star

RPAD() 함수 : 오른쪽 기준

SELECT RPAD('star', 10,'*') FROM DUAL;

star*****

2.숫자함수

1) ROUND() 함수 :반올림

문 법: ROUND(숫자 , 출력을 원하는 자리 수)

2) TRUNC() 함수: 버림

이 함수는 ROUND 와 사용법은 동일하며 차이점은 무조건 버림을 한다는 것입니다. 문 법: TRUNC(숫자, 원하는 자리수)

3) MOD(), CEIL(), FLOOR() 함수

3. 날짜함수
SYSDATE , SYSTIMESTAMP :현재날짜를 얻어 오는 함수 , 괄호없음 -주의
MONTHS_BETWEEN()함수



SELECT TO_CHAR(SYSDATE, 'YYYY') FROM DUAL;

YYYY -연도를 4자리로 표현 (예: 2014)

YY - 연도의 끝 2자리를 표현

MM : 월을 숫자2자리로 표현 MON : OCT :영문3자리 표현

MONTH: 월을 뜻하는 이름 전체 표현

DD : 일을 숫자2자리로 표시

DAY: 요일에 해당하는 명칭 표시

SELECT 주문번호, 주문수량 FROM 주문테이블 WHERE <mark>주문일자</mark> BETWEEN

'2021-08-01'

AND

'2021-08-31';

8월 한달동안의 주문 내역 조회

SELECT 주문번호, 주문수량 FROM 주문테이블 WHERE 주문일자 BETWEEN

TO_DATE('2021-08-01', 'YYYY-MM-DD')

AND

TO_DATE('2021-08-31', 'YYYY-MM-DD');

```
TO_CHAR( 1234, '99999') => 1234
TO_CHAR( 1234, '099999') => 001234
TO_CHAR( 1234, '$9999') => $1234
TO_CHAR( 1234, '9999.99') => 1234.00
TO_CHAR( 12345, '99,999') => 12,345
```

TO_NUMBER()함수 숫자가 아닌 숫자처럼 생긴 문자를 숫자로 바꾸어 주는 함수입니다.

SELECT TO_NUMBER('5') from dual;

Dual 테이블의 정의

- 1. 오라클 자체에서 제공되는 테이블
- 2. 간단하게 함수를 이용해서 계산 결과값을 확인 할 때 사용하는 테이블

함수에 대한 쓰임을 알고 싶을때 특정 테이블을 생성할 필요없이 dual 테이블을 이용하여 함수의 값을 리턴(return)받을 수 있다.

TO_DATE()함수

:날짜처럼생긴 문자를 날짜로 바꿔 주는 함수이다.

SELECT TO_DATE('2014/05/31') from dual

일반함수

NVL()함수

NULL값을 만나면 다른 값으로 치환해서 출력하는 함수.

SELECT name, salary , nvl(comm , 0) FROM emp ;

NVL2()함수

NULL값이 아닐 경우 출력할 값을 지정할 수 있다.

comm이 null이 아니면 comm+400 null이면 100 출력함

SELECT name, salary, nvl2(comm, comm+400, 100) FROM emp;

DECODE()함수

일반언어의 if문을 오라클에서 가져온 개념

SELECT name, DECODE(gender, 1, '남성', 2, '여성') FROM customer;

| id | name | gender |
|------|------|--------|
| a001 | 김혜겸 | 1 |
| b001 | 김세정 | 2 |
| c001 | 아이유 | 2 |

CASE문

SELECT name, tel , CASE SUBSTR(tel, 1, INSTR(tel, ')') -1)

WHEN

WHEN '02' THEN 'SEOUL'

'031' THEN 'GYEONGGGI'

WHEN '051' THEN 'BUSAN'

WHEN '052' THEN 'ULSAN'

WHEN '055' THEN 'GYEONGNAM'

대상

END LOC

FROM student

02)6255-9875 02)312-9839 051)738-1234



SEOUL SEOUL BUSAN select m_name, case substr(m_tel,1, instr(m_tel , ')') -1) when '02' then 'seoull' when '031' then 'GYEONGGG' end loc from member_tbl_12;

```
CREATE TABLE score_tbl(
   id varchar2(5) not null primary key,
   name varchar2(20)
   total number(3)
);
```

INSERT INTO score_tbl values('G001', '홍길동' ,80);

```
SELECT name, CASE WHEN total >= 90 THEN '수' WHEN total >=80 THEN '우' WHEN total >=70 THEN '미' ELSE '가' END GRADE
```

FROM score_tbl;

복수행함수 (그룹함수)

COUNT()함수 SUM()함수 AVG()함수 MAX()함수 MIN()함수

GROUP BY절 (특정 조건을 주고 해당 조건에 맞는 결과들을 모아서 조금 더 구체적인 결과 얻음)

부서별 월급 합계, 또는 부서별 월급 평균

```
CREATE TABLE t_emp(
eid
            VARCHAR2(13) NOT NULL PRIMARY KEY,
            VARCHAR2(20),
 ename
            NUMBER(5) ,
salary
deptno
            NUMBER(5) ,
             NUMBER(5)
comm
COMMIT;
INSERT INTO t_emp VALUES ('e001', '신동엽', 280,10,100);
INSERT INTO t_emp VALUES ('e002', '유재석', 250,20,50);
INSERT INTO t_emp VALUES ('e003', '전현무', 190,30,0);
INSERT INTO t_emp VALUES ('e004', '양세영', 300,20,0);
INSERT INTO t emp VALUES ('e005', '양세찬', 290,40,100);
CREATE TABLE t_dept(
deptno
             NUMBER(5) NOT NULL PRIMARY KEY,
             VARCHAR2(20)
dname
COMMIT;
INSERT INTO t_dept VALUES (10, '영업팀');
INSERT INTO t_dept VALUES (20, '기획팀');
INSERT INTO t dept VALUES (30, '관리팀');
INSERT INTO t dept VALUES (40, '마케팅팀');
INSERT INTO t_dept VALUES (50, '재무팀');
```

View/서브쿼리 연습테이블정보

VIEW

가상 테이블 View에는 데이터가 없고 원본 테이블에 가서 데이터를 불러오는 sql query만 저장되어 있음

(보안과 사용자의 편의성을 위해 사용함)

- 1. 단순뷰 (하나의 테이블로 부터 쿼리)
- 2. 복합뷰 (여러 개의 테이블로 부터 쿼리)
- 3. 인라인뷰(FROM절에 뒤에 나오는 서브쿼리)

왜 부를 사용하는가?

- 1. 불필요한 정보를 노출하고 싶지 않을 때 (가리고 싶은 내용이 있을 때)
- 2. 업무적으로 사용하는 단위의 테이블을 만들고 싶을 때 (매번 복작입 join 쿼리를 만들지 않아도 됨) 쿼리의 복잡도를 낮출 수 있음

단순뷰 예제

```
CREATE VIEW v_emp
AS
SELECT ename, deptno
FROM t_emp;
```

SELECT * FROM v_emp;

t_emp

| eid | ename | salary | deptno | comm |
|------|-------|--------|--------|------|
| e001 | 신동엽 | 280 | 10 | 100 |
| e002 | 유재석 | 250 | 20 | 50 |
| e003 | 전현무 | 190 | 30 | |
| e004 | 양세영 | 300 | 20 | |
| e005 | 양세찬 | 290 | 40 | 100 |

JOIN된 쿼리로 뷰만들기

CREATE VIEW v_emp2
AS

SELECT e.ename, d.dname
FROM t_emp e, t_dept d
WHERE e.deptno = d.deptno;

SELECT * FROM v_emp2;

| ename | dname |
|-------|-------|
| 신동엽 | 영업팀 |
| 유재석 | 기획팀 |
| 전현무 | 관리팀 |
| 양세영 | 기획팀 |
| 양세찬 | 마케팅 |

| eid | ename | salary | deptno | comm |
|------|-------|--------|--------|------|
| e001 | 신동엽 | 280 | 10 | 100 |
| e002 | 유재석 | 250 | 20 | 50 |
| e003 | 전현무 | 190 | 30 | |
| e004 | 양세영 | 300 | 20 | |
| e005 | 양세찬 | 290 | 40 | 100 |

| deptno | dname |
|--------|-------|
| 10 | 영업팀 |
| 20 | 기획팀 |
| 30 | 관리팀 |
| 40 | 마케팅 |
| 50 | 재무팀 |

부서 테이블(t_dept)

t_emp

(사원)

인라인뷰 예제

뷰를 따로 만들지 않고 서브쿼리로 작성시 뷰가 만들어져 사용된다

```
SELECT e.deptno, d.dname, e.maxsal FROM(
SELECT deptno, MAX(salary) maxsal FROM t_emp
GROUP BY deptno
) e, t_dept d
WHERE e.deptno = d.deptno;
```

서브쿼리의 결과

| deptno | maxsal |
|--------|--------|
| 10 | 280 |
| 20 | 300 |
| 30 | 190 |
| 40 | 290 |

t_dept (부서)

| dcode | dname |
|-------|-------|
| 10 | 영업팀 |
| 20 | 기획팀 |
| 30 | 관리팀 |
| 40 | 마케팅 |
| 50 | 재무팀 |

t_emp(사원) 주키



| eid | ename | salary | deptno | comm |
|------|-------|--------|--------|------|
| e001 | 신동엽 | 280 | 10 | 100 |
| e002 | 유재석 | 250 | 20 | 50 |
| e003 | 전현무 | 190 | 30 | |
| e004 | 양세영 | 300 | 20 | |
| e005 | 양세찬 | 290 | 40 | 100 |

쿼리결과

| deptno | dname | maxsal |
|--------|-------|--------|
| 10 | 영업팀 | 280 |
| 20 | 기획팀 | 300 |
| 30 | 관리팀 | 190 |
| 40 | 마케팅 | 290 |

<u></u> 세 빌 뤰 리

신동엽 사원보다 급여가 더 많은 사원이름과 상여금을 조회하시오

| eid | ename | salary | deptno | comm |
|------|-------|--------|--------|------|
| e001 | 신동엽 | 280 | 10 | 100 |
| e002 | 유재석 | 250 | 20 | 50 |
| e003 | 전현무 | 190 | 30 | |
| e004 | 양세영 | 300 | 20 | |
| e005 | 양세찬 | 290 | 40 | 100 |

괄호안의 쿼리가 먼저 수행됨

단일행 쿼리

```
SELECT ename, salary from t_emp
Where salary > (select salary from t_emp where ename='신동엽');
```

```
SELECT ename, salary from t_emp Where salary > 280;
```

다중행 서브쿼리

SELECT eid, ename, deptno FROM t_emp WHERE deptno IN (SELECT deptno FROM t_dept WHERE dname='기획팀' or dname='마케팅팀');

SELECT eid, ename, deptno FROM t_emp WHERE deptno IN (20, 40)

사원테이블(t_emp)

| eid | ename | salary | deptno | comm |
|------|-------|--------|--------|------|
| e001 | 신동엽 | 280 | 10 | 100 |
| e002 | 유재석 | 250 | 20 | 50 |
| e003 | 전현무 | 190 | 30 | |
| e004 | 양세영 | 300 | 20 | |
| e005 | 양세찬 | 290 | 40 | 100 |

부서 테이블(t_dept)

| deptno | dname |
|--------|-------|
| 10 | 영업팀 |
| 20 | 기획팀 |
| 30 | 관리팀 |
| 40 | 마케팅팀 |
| 50 | 재무팀 |

인라인뷰 예제

뷰를 따로 만들지 않고 서브쿼리로 작성시 뷰가 만들어져 사용된다

SELECT e.deptno , d.dname , e.salary FROM(

SELECT deptno , salary FROM t_emp

) e , t_dept d

WHERE e.deptno = d.deptno ;

t_emp(사원)

| eid | ename | salary | deptno | comm |
|------|-------|--------|--------|------|
| e001 | 신동엽 | 280 | 10 | 100 |
| e002 | 유재석 | 250 | 20 | 50 |
| e003 | 전현무 | 190 | 30 | |
| e004 | 양세영 | 300 | 20 | |
| e005 | 양세찬 | 290 | 40 | 100 |

서브쿼리의 결과

| deptno | salary |
|--------|--------|
| 10 | 280 |
| 20 | 300 |
| 30 | 190 |
| 40 | 290 |

| deptno | dname | salary |
|--------|-------|--------|
| 10 | 영업팀 | 280 |
| 20 | 기획팀 | 300 |
| 30 | 관리팀 | 190 |
| 40 | 마케팅 | 290 |

부서 테이블(dept)

| deptno | dname |
|--------|-------|
| 10 | 영업팀 |
| 20 | 기획팀 |
| 30 | 관리팀 |
| 40 | 마케팅 |
| 50 | 재무팀 |