

PL SQL

PLSQL(Procedural Language)

Sql의 장점은 쿼리문 하나로 원하는 데이터를 검색 조작 할 수 있다는 것.

그런데 sql문 자체는 비 절차성 언어이기에 몇 개의 쿼리문 사이에 어떠한 연결 및 절차성 이 있어야 하는 경우 사용 할수 없음.

이점을 극복하기 위해 오라클 사에서 sql언어에 절차적인 프로그래밍 가능하도록 PLSQL을 만듦.

PLSQL(Procedural Language) 구조

Declare —선언(선택)

Begin

실행(필수)

End;

예

Begin

DBMS_output.put_line('heejung'); --출력

End;

PLSQL(Procedural Language)

1. 변수선언
2. 제어문
3. 프로시저(input, output, cursor)

변수선언

* 변수선언방법

DECLARE 변수이름 데이터타입;

ex) DECLARE NAME varchar2(10);

DECLARE 변수이름 데이터타입:=값;

ex) DECLARE NAME varchar2(10):= '이도연' ;

DECLARE 변수이름 데이터타입 DEFAULT 기본값;

ex) DECLARE NAME varchar2(10) DEFAULT '이도연' ;

* 변수를 한번에 여러개 선언방법

DECLARE NAME VARCHAR2(20) ; age NUMBER(2) ;

ddr VARCHAR2(50);

변수선언

* 데이터타입의 종류

- number(), varchar2(), char(), int, date,....
- 테이블이름.필드명%TYPE => 필드명과 같은 타입 선언
ex) DECLARE NAME emp.ename%TYPE ;
- 테이블이름%RowType => 테이블의 전체열을 갖는 타입(한 레코드)
ex) DECLARE data emp%ROWTYPE ;

* 변수 선언하여 사용하는 방법

```
DECLARE NAME VARCHAR2(20) := '이효리';  
BEGIN  
    DBMS_OUTPUT.PUT_LINE('이름은 ' || name); --화면에 출력  
END;
```

=> PLSQL문장은 반드시 BEGIN ~ end;안에서 사용함.

변수선언

ex) 테이블이름.필드명%type에 대한 예제

```
DECLARE e_name emp.ENAME%TYPE ;  
        e_sal emp.SAL%TYPE;  
BEGIN  
    SELECT sal, ename INTO e_sal, e_name FROM EMP WHERE empno='7788';  
    DBMS_OUTPUT.PUT_LINE('7788님의 이름은 ' || e_name || ' 급여는 ' || e_sal);  
END;
```

ex) 테이블이름%Rowtype에 대한 예제

```
DECLARE data emp%ROWTYPE;  
BEGIN  
    SELECT * INTO data FROM EMP WHERE empno = '7788';  
    DBMS_OUTPUT.PUT_LINE(data.ename || ', ' || data.job || ', ' || data.mgr || ', ' || data.deptno);  
END;
```

* 테이블로부터 가져온 필드의 값을 변수에 저장하기 위해서 SELECT절에서 INTO 사용함.

제어문의 종류

1. if문

```
IF 조건식 THEN 실행문장;  
ELSIF 조건식 THEN 실행문장;  
ELSIF 조건식 THEN 실행문장;  
....  
ELSE 실행문장;  
END IF;
```

2. for문

```
FOR 변수이름 IN 시작 .. 끝 loop  
    실행문장;  
END LOOP;
```



제어문의 종류

3. LOOP END 문

LOOP

실행문장;

증감식;

EXIT [WHEN 조건식] ; --조건식이 만족할 때 loop을 빠져나간다.

END loop;

=> EXIT 문이 사용되었을 경우 무조건 loop 빠져나간다.

=> EXIT WHEN 사용되었을 경우 WHEN절에 loop를 빠져 나가는 조건으로 제어할 수 있다.

4. while문

WHILE 조건식 LOOP

실행문장;

증감식;

END LOOP;

제어문의 종류

ex) emp테이블에서 empno가 7788인 레코드의 empno, ename, comm을 검색하여 comm의 값이 0보다 크면 'ename의 커미션은 ~입니다.' 출력하고
아니면 'ename은 커미션을 받지않습니다.'출력

ex) emp테이블에서 empno가 7788인 레코드의 deptno가 10이면 'Accounting' 20이면 'Research' 30이면 'sales', 40이면 'operation' 을 변수(dname)에 담아서 if문 끝난 다음에 출력해주세요.

ex) emp테이블에서 empno가 7788인 레코드의 sal의 값이 4000이상이면 고액년봉,
sal 3000 ~ 4000미만이면 일반년봉, sal 2000 ~ 3000미만이면 저소득년봉
sal이 2000미만이면 소외계층 년봉 출력

제어문의 종류

ex) 1~10 까지 출력

ex) 1~10사이의 짝수만 출력

ex) 2단의 구구단 출력

ex) 사용자 입력(단수)를 받아 구구단 출력

[힌트]

DECLARE vdan NUMBER(1) :=&dan; -- &는 사용자입력값받을때 사용하는 키워드

ex) 구구단을 2중 반복문


ex) 1~10까지 총 합계를 구하기(loop)

프로시저

기능을 미리 만들어 놓고 호출하여 사용하는것 (재사용성)

* 프로시저생성방법 (인수 없는 경우)

```
CREATE PROCEDURE 프로시저이름
IS
[
    변수이름 데이터타입; --프로시저내에서 사용할 변수선언
    변수이름 데이터타입;
    ....
]
BEGIN
    기능 구현;
END;
```



프로시저

* 프로시저생성방법 (인수 있는 경우)

```
CREATE PROCEDURE 프로시저이름(  
    변수이름 IN 데이터타입 , 변수이름 IN 데이터타입 , ....  
)  
IS  
[  
    변수이름 데이터타입; --프로시저내에서 사용할 변수선언  
    변수이름 데이터타입;  
    ....  
]  
BEGIN  
    기능 구현;  
END;
```

=> 인수의 타입 선언부분정리
변수이름 IN VARCHAR2 ; --인수선언할때 byte수 지정안함.
변수이름 IN 테이블이름.컬럼명%TYPE;
변수이름 IN 테이블이름.컬럼명%TYPE :=값;
변수이름 IN 테이블이름.컬럼명%TYPE DEFAULT 값;

* 인수의 타입정의할때 IN 키워드 생략가능함.

프로시저

프로시저 호출방법

EXECute 프로시저이름; --인수 없는 경우 호출

EXECute 프로시저이름(값, 값,...) ; --인수 있는 경우 호출



프로시저

ex) 메시지를 출력하는 프로시저 작성

```
CREATE PROCEDURE p_test
IS
BEGIN
  DBMS_OUTPUT.PUT_LINE('Oracle 시험을 잘 볼수있겠죠~~ ㅎㅎ');
END;
```

--프로시저 호출

```
EXEC p_test;
```

ex) 이름을 인수로 받아 ~ 님 정보처리 합격하셨습니다. 출력한 프로시저작성

```
CREATE OR REPLACE PROCEDURE p_test(NAME IN varchar2)
IS
BEGIN
  DBMS_OUTPUT.PUT_LINE(NAME || '님 정보처리 합격하셨습니다.');
```

--프로시저호출

```
EXEC p_test('장희정');
```

```
EXEC p_test ; --오류발생(프로시저에 in이 있으면서 기본값이 없기때문에 반드시 인수값 필요함)
```

프로시저

Ex) 인수 아이디, 이름, 나이, 주소를 입력받아 userlist테이블에 insert한 후
인수의 값들을 출력하는 프로시저작성한다(인수에 기본값 지정해본다)

```
CREATE OR REPLACE PROCEDURE P_userInsert(  
  id IN userlist.id%TYPE := 'kkk',  
  NAME IN userlist.name%TYPE DEFAULT '이쁜이',  
  age IN userlist.age%TYPE := 10,  
  addr IN userlist.addr%TYPE := null  
)  
IS  
BEGIN  
  INSERT INTO USERLIST VALUES(id, NAME, age, addr);  
  DBMS_OUTPUT.PUT_LINE('insert정보는 ' || id || NAME || age || addr);  
END;
```

--프로시저실행

```
EXEC P_userInsert('bb', '이효리', 10, '서울');  
EXEC P_userInsert; --인수 넣지않으면 기본값으로 들어감.
```

```
EXEC P_userInsert(id=>'장동건', age=>50);  
* 기본값을 정했지만 실행할때 원하는 인수에만 값을 넣고 나머지는 default로 인자가  
전달되게 하기 위해서는  
EXEC 프로시저이름(인수이름=>값, 인수이름=>값, ...);
```


프로시저

저장프로시저 찾기

```
SELECT * FROM user_objects  
WHERE LOWER(object_type)='procedure'
```

프로시저가 작성된 쿼리문 검색

```
SELECT text FROM user_source  
WHERE LOWER(NAME) = ' 프로시저이름'
```

프로시저 output

프로시저를 실행한 후 특정 결과값을 out변수에 저장하여 보낼 수 있다.

Out이 있는 프로시저 작성법

```
CREATE PROCEDURE 프로시저이름(  
    변수이름 IN 테이터타입 , 변수이름 IN 테이터타입 ,....  
    변수이름 out 테이터타입 ,변수이름 out 테이터타입 ,....  
)  
IS  
[  
    변수이름 테이터타입; --프로시저 내에서 사용할 변수선언  
    변수이름 테이터타입;  
    ....  
]  
BEGIN  
    기능 구현;  
END;
```

프로시저 output

프로시저를 실행한 후 특정 결과값을 out변수에 저장하여 보낼 수 있다.

Out이 있는 프로시저 작성법

```
CREATE PROCEDURE 프로시저이름(  
    변수이름 IN 테이터타입 , 변수이름 IN 테이터타입 ,....  
    변수이름 out 테이터타입 ,변수이름 out 테이터타입 ,....  
)  
IS  
[  
    변수이름 테이터타입; --프로시저 내에서 사용할 변수선언  
    변수이름 테이터타입;  
    ....  
]  
BEGIN  
    기능 구현;  
END;
```

프로시저 output

ex) 이름과 나이를 output해주는 프로시저 작성

```
CREATE OR REPLACE PROCEDURE p_outTest(  
    NAME OUT VARCHAR2 , age OUT varchar2  
)  
IS  
BEGIN  
    NAME := '이나영';  
    age := 20;  
    DBMS_OUTPUT.PUT_LINE('out을 이용한 프로시저 완료');  
END;
```

--out이 있는 프로시저 호출방법

VARIABLE 변수이름 데이터타입; --메모리에 만들어지는 변수

ex) 바인드 변수 선언

VARIABLE v_name VARCHAR2(20);

VARIABLE v_age NUMBER(3);

EXEC p_outTest(:v_name , :v_age); --프로시저를 실행 한후에 out을 받을 변수지정.

print 변수이름 ; -- 출력하기

프로시저 cusor 활용

ex) job을 인수로 받아 해당하는 사원의 이름을 검색(프로시저작성)

```
--SELECT  ename FROM EMP WHERE job='연예인'
```

```
CREATE OR REPLACE PROCEDURE p_job_emp(e_job IN emp.job%type)
IS
NAME emp.ENAME%TYPE ;
BEGIN
    SELECT  ename INTO name FROM EMP WHERE job=e_job;
    DBMS_OUTPUT.PUT_LINE(NAME || '님 담당업무는 ' || e_job);
END;
```

--프로시저호출해본다.

```
EXEC p_job_emp('총무부'); --실행완료(총무부에 해당하는 레코드결과는 한행이다.)
EXEC p_job_emp('연예부'); --오류발생(실행결과의 레코드 수가 1개이상이므로 오류발생함)
EXEC p_job_emp('연예부2'); --오류발생(연예부2에 해당하는 레코드가 없으므로 오류발생)
```

*프로시저안에서 결과의 레코드가 여러개를 반환 할 경우는 결과값을
cursor에 저장하여 커서를 통해 가져와야한다.

프로시저 cursor 활용

--cursor 활용

=> 커서를 이용하여 질의 수행결과 반환되는 여러 행을 처리 할수 있다.

=> 커서 사용 순서

1. 커서 선언

CURSOR 커서이름 IS select문장;

2. 커서 열기

OPEN 커서이름;

3. 커서로부터 데이터 읽기 (LOOP end의 반복문을 활용한다)

FETCH 커서이름 INTO 저장할 로컬변수

4. 커서닫기

CLOSE 커서이름

=> 커서 속성

%FOUND -- PL/SQL코드가 마지막으로 얻은 커서의 결과 set에 레코드가 있다면 참.

%NOTFOUND -- %FOUND의 반대

%ROWCOUNT -- 커서에서 얻은 레코드수 반환

%ISOPEN -- 커서가 열렸고 아직 닫히지 않은 상태라면 참

|

프로시저 cusor 활용

ex) job을 인수로 받아 해당하는 사원의 이름을 검색(프로시저작성 - CURSOR 이용함.)

```
CREATE OR REPLACE PROCEDURE p_job_emp(e_job IN emp.job%type)
IS
NAME emp.ENAME%TYPE ;
CURSOR c_name IS SELECT  ename FROM EMP WHERE job=e_job; -- 1.커서선언.
BEGIN
    OPEN c_name; --2. 커서열기
    DBMS_OUTPUT.PUT_LINE('=====');
    LOOP
        FETCH c_name INTO name;--3. 커서로부터 데이터 읽기
        EXIT WHEN c_name%NOTFOUND; --커서에서 데이터를 찾을수 없으면 반복문 빠져나가라.
        DBMS_OUTPUT.PUT_LINE(NAME || '님 담당업무는 ' || e_job);
    END LOOP;
    DBMS_OUTPUT.PUT_LINE('결과 레코드수 => ' || c_name%ROWCOUNT);
    CLOSE c_name ;--4. 커서 닫기
END;
```

--프로시저실행해본다.

```
EXEC p_job_emp('총무부');
EXEC p_job_emp('연예인');
EXEC p_job_emp('연예부2');
```

프로시저 cusor 활용

ex) 커서의 데이터에 여러 열을 가져오기

job을 인수로 받아 해당하는 사원의 이름을 검색(프로시저작성 - CURSOR 이용함.)

```
--SELECT empno, ename, sal FROM EMP WHERE job='연예인'
```

```
CREATE OR REPLACE PROCEDURE p_job_emp(e_job IN emp.job%type)
```

```
IS
```

```
NAME emp.ENAME%TYPE ;
```

```
empno emp.EMPNO%TYPE;
```

```
sal emp.SAL%TYPE;
```

```
CURSOR c_name IS SELECT empno, ename, sal FROM EMP WHERE job=e_job; -- 1. 커서선언.
```

```
BEGIN
```

```
OPEN c_name; --2. 커서열기
```

```
DBMS_OUTPUT.PUT_LINE('=====');
```

```
LOOP
```

```
FETCH c_name INTO empno, NAME, sal ;--3. 커서로부터 데이터 읽기
```

```
EXIT WHEN c_name%NOTFOUND; --커서에서 데이터를 찾을수 없으면 반복문 빠져나가라.
```

```
DBMS_OUTPUT.PUT_LINE(NAME || '님 ' || empno || ' , ' || sal || ' , ' || e_job);
```

```
END LOOP;
```

```
DBMS_OUTPUT.PUT_LINE('결과 레코드수 => ' || c_name%ROWCOUNT);
```

```
CLOSE c_name ;--4. 커서 닫기
```

```
END;
```


프로시저 cursor 활용

ex) 커서에 테이블의 모든 컬럼 가져오기

job을 인수로 받아 해당하는 사원의 이름을 검색(프로시저작성 - CURSOR 이용함.)

```
--SELECT * FROM EMP WHERE job='연예인'
```

```
CREATE OR REPLACE PROCEDURE p_job_emp(e_job IN emp.job%type)
IS
data emp%ROWTYPE;
CURSOR c_name IS SELECT * FROM EMP WHERE job=e_job; -- 1.커서선언.
BEGIN
OPEN c_name; --2. 커서열기
DBMS_OUTPUT.PUT_LINE('=====');
LOOP
FETCH c_name INTO data ;--3. 커서로부터 데이터 읽기
EXIT WHEN c_name%NOTFOUND; --커서에서 데이터를 찾을수 없으면 반복문 빠져나가라.
DBMS_OUTPUT.PUT_LINE(data.ename || '님 ' || data.deptno||' , ' || data.sal || ' , ' || e_job);
END LOOP;
DBMS_OUTPUT.PUT_LINE('결과 레코드수 => ' || c_name%ROWCOUNT);
CLOSE c_name ;--4. 커서 닫기
END;
```

프로시저 cursor 활용 문제

1번) 특정 과목을 인수로 받아 그 과목을 강의하는 강사 정보 검색을 검색하는 프로시저 작성.

2번) 특정 과목을 수강하고 있는 학생의 정보를 검색하는 프로시저 작성.

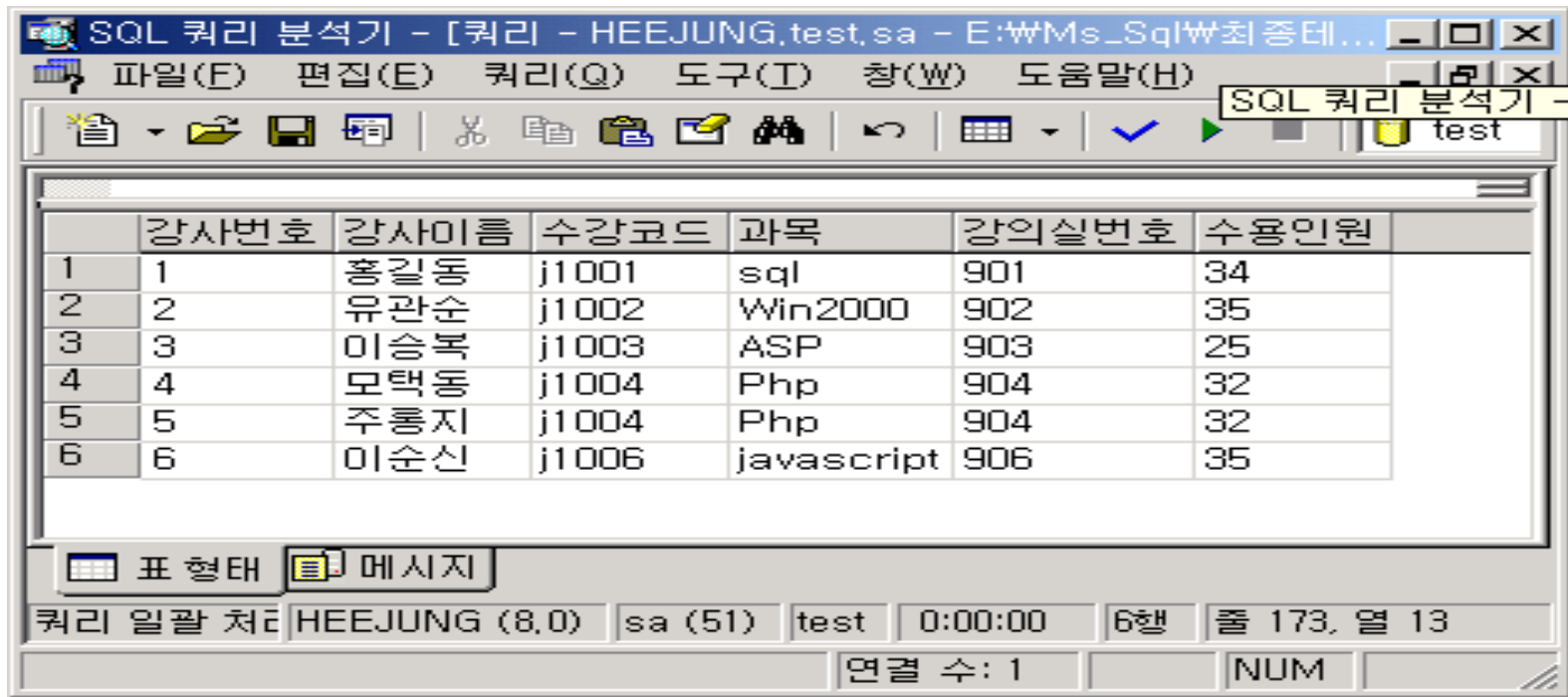
3번) 1~100 사이를 반복하되 10의 배수만 출력하는 프로시저 작성.

4번) member 테이블(id, name, age, addr)에 데이터를 insert한 후 최종 모든 레코드를 검색하는 프로시저작성 (insert 데이터는 인수로 받는다 , cursor 이용, 출력 후 최종 전체 레코드수를 출력한다.)

5번) id, name, age를 인수로 받아 id에 해당하는 레코드의 값을 인수로 들어온 name, age로 update 하고 난 후 commit 하고 변경된 레코드의 결과를 출력하는 프로시저를 작성하시오.

프로시저 cusor 활용 문제

6번) 강사가 담당하는 과목과 그 과목이 어느 강의실에서 진행되며 총 수용 인원이 몇 명인지 검색하는 프로시저 작성.(아래그림참조) -(Join이용 , cursor 이용)



	강사번호	강사이름	수강코드	과목	강의실번호	수용인원
1	1	홍길동	j1001	sql	901	34
2	2	유관순	j1002	Win2000	902	35
3	3	이승복	j1003	ASP	903	25
4	4	모택동	j1004	Php	904	32
5	5	주홍지	j1004	Php	904	32
6	6	이순신	j1006	javascript	906	35

꿈

사

함

니

다

!