

제 8 장

저장 프로시저와 저장 함수



목차

8.1 저장 프로시저

8.2 저장 함수

8.3 예외 처리

8.1 저장 프로시저

```
create [or replace] procedure procedure-name  
  (argument1 [mode] data_type,  
    (argument2 [mode] data_type,  
      ... ..)  
is  
  variable data_type,  
    ... ..  
begin  
  ... ..  
exception  
  ... ..  
end;
```

```
drop procedure procedure-name
```

8.1 저장 프로시저의 생성 및 실행

➤ 과목 테이블(subject)에 한 과목 삽입

```
create procedure test1
(v_sub_name in subject.sub_name%type,
 v_sub_prof in subject.sub_prof%type,
 v_sub_grade in subject.sub_grade%type,
 v_sub_dept in subject.sub_dept%type)
is
  v_sub_no subject.sub_no%type;
begin
  select max(sub_no)
    into v_sub_no
    from subject;
  v_sub_no := to_number(v_sub_no)+1;

  insert
    into subject
    values(v_sub_no,v_sub_name,v_sub_prof,v_sub_grade,
          v_sub_dept);
  commit;
end test1;
/
```

8.1 저장 프로시저의 생성 및 실행

➤ 과목 테이블(subject)에 한 과목 삽입

```
SQL> @test1.sql
```

```
SQL> select * from subject;
```

```
SQL> execute test1( '컴퓨터구조' , '강주봉' ,2, '컴퓨터정보' );
```

```
SQL> select * from subject;
```

8.1 저장 프로시저

- 학번, 학년을 입력으로 학생의 학년을 수정

```
create or replace procedure test2  
  (v_stu_no in student.stu_no%type,  
    v_stu_grade in student.stu_grade%type)  
  is  
  begin  
    update student  
      set stu_grade = v_stu_grade  
      where stu_no = v_stu_no;  
end test2;
```

8.1 저장 프로시저

- 학번, 학년을 입력으로 학생의 학년을 수정

```
select *  
from student  
where stu_no = 20153075;
```

```
exec test2(20153075,2);
```

```
select *  
from student  
where stu_no = 20153075;
```

8.1 저장 프로시저

- 학번을 입력으로 학생의 이름을 검색

```
create or replace procedure test3  
  (v_stu_no in student.stu_no%type,  
   v_stu_name out student.stu_name%type)  
is  
begin  
  select stu_name  
    into v_stu_name  
  from student  
    where stu_no = v_stu_no;  
end test3;
```


8.1 저장 프로시저

- 학번을 입력으로 학생의 이름을 검색

```
variable d_stu_name varchar2(12);
```

```
exec test3(20153075,:d_stu_name);
```

```
print d_stu_name;
```

8.1 저장 프로시저

- 학생의 점수를 임의 점수만큼 올려주는 프로시저

```
create or replace procedure test4  
  (v_sub_no in enrol.sub_no%type,  
   v_stu_no in enrol.stu_no%type,  
   v_enr_grade in out enrol.enr_grade%type)  
is  
begin  
  update enrol  
    set enr_grade = enr_grade + v_enr_grade  
  where stu_no = v_stu_no  
    and sub_no = v_sub_no;  
  
  select enr_grade  
    into v_enr_grade  
  from enrol  
    where stu_no = v_stu_no  
      and sub_no = v_sub_no;  
  
end test4;
```

8.1 저장 프로시저

- 학생의 점수를 임의 점수만큼 올려주는 프로시저

```
variable d_enr_grade number  
begin :d_enr_grade := 10; end;
```

```
exec test4(101,20131001,:d_enr_grade);
```

```
print d_enr_grade;
```

8.1 저장 프로시저

- 과목 추가, 과목번호는 과목이 입력되는 순서대로 부여
- 시퀀스 seq1, 201부터 999까지 한 번에 1씩 증가

```
create sequence seq1  
increment by 1  
start with 201  
maxvalue 999;
```

8.1 저장 프로시저

- 과목 추가, 과목번호는 과목이 입력되는 순서대로 부여

```
create procedure test5  
(v_sub_name in subject.sub_name%type,  
  v_sub_prof in subject.sub_prof%type,  
  v_sub_grade in subject.sub_grade%type,  
  v_sub_dept in subject.sub_dept%type)  
is  
begin  
  insert into subject  
    values(seq1.nextval, v_sub_name,  
              v_sub_prof, v_sub_grade, v_sub_dept);  
  
  commit;  
end test5;
```

8.1 저장 프로시저

- 과목 추가, 과목번호는 과목이 입력되는 순서대로 부여

```
select * from subject order by 1;
```

```
exec test5('앱', '홍길동', 3, '컴퓨터정보');
```

```
select * from subject order by 1;
```

8.2 저장함수

```
create [or replace] function function-name  
  (argument1 [mode] data_type,  
   argument2 [mode] data_type,  
   .....  
   return data type  
  is  
   variable data_type;  
  .....  
begin  
  .....  
   return variable;  
exception  
  .....  
end;
```

```
drop function function-name
```

8.2 저장함수

● 성적 환산 함수

```
create or replace function test6  
  (v_enr_grade in number)  
  return char  
is  
  enr_score char;  
begin  
  if v_enr_grade >= 90 then enr_score := 'A';  
  elsif v_enr_grade >= 80 then enr_score := 'B';  
  elsif v_enr_grade >= 70 then enr_score := 'C';  
  elsif v_enr_grade >= 60 then enr_score := 'D';  
  else enr_score := 'F';  
  end if;  
  return (enr_score);  
end test6;
```


8.2 저장함수

● 성적 환산 함수

```
variable d_score char;
```

```
execute :d_score := test6(95);
```

```
print d_score;
```

```
select enr_grade, test6(enr_grade) as score  
from enrol  
where stu_no = 20131001;
```

8.3 예외처리

●예외처리(EXCEPTION) : 에러발생, 예외적인 상황 발생

➤에러처리

처리 조건명	설 명
NO_DATA_FOUND	검색문 사용 후 결과가 있는지 여부 판단
NOT_LOGGED_ON	데이터베이스에 연결상태를 판단
TOO_MANY_ROWS	SELECT문에 INTO절을 사용한 경우 SELECT절의 결과가 복수행일 경우
VALUE_ERROR	변수의 길이보다 큰 값을 저장하는 경우
ZERO_DEVIDE	열의 값을 0 값으로 나누는 경우
INVALID_CURSOR	커서 선언의 SELECT문에 대한 연산이 부적절한 경우
DUP_VAL_ON_INDEX	UNIQUE INDEX가 설정된 열에 중복 값이 입력하는 경우

8.3 예외처리

●학번에 의한 학생 이름 검색

```
create or replace procedure test7  
(v_stu_no in student.stu_no%type)  
is  
    v_stu_name student.stu_name%type;  
begin  
    select stu_name  
        into v_stu_name  
        from student  
        where stu_no = v_stu_no;  
    dbms_output.put_line(v_stu_name);  
exception  
    when no_data_found then  
        dbms_output.put_line('해당 데이터가 없습니다.');  
end test7;
```

8.3 예외처리

●과목 수강자(학번)과 성적 검색

```
exec test7(20153088)
```

이태연

```
exec test7(20153089)
```

해당 데이터가 없습니다.

8.3 예외처리

●예외처리

create or replace procedure test

.....

is

예외조건명 EXCEPTION;

begin

.....

if(condition) then

RAISE 예외조건명;

exception

when 예외조건명 then

.....

end test;.

8.3 예외처리

●과목 수강자 수 검색

```
create or replace procedure test8  
(v_sub_no in enrol.sub_no%type)  
is  
    v_cnt number;  
    cnt_error exception;  
begin  
    select count(stu_no) into v_cnt from enrol  
                                where sub_no = v_sub_no;  
    dbms_output.put_line(v_sub_no||' 과목 수강자는 '||v_cnt||'명입니다.');  
    if v_cnt = 0 then  
        raise cnt_error;  
    end if;  
exception  
    when cnt_error then  
        dbms_output.put_line('수강자가 없습니다.');  
end test8;
```

8.3 예외처리

●과목 수강자(학번)과 성적 검색

exec test8(102)

102 과목 수강자는 2명입니다.

=====

exec test8(120)

120 과목 수강자는 0명입니다.
수강자는 없습니다.