PL/SQL 개념

- **PL/SQL의 개념**
  - Oracle에서 지원하는 프로그래밍 언어의 특성을 수용한
    SQL의 확장
  - PL/SQL Block내에서 SQL의 DML(데이터 조작어) 문과
    Query(검색어)문, 그리고 절차형 언어(IF, LOOP) 등을
    사용하여 절차적으로 프로그래밍을 가능하게 한
    강력한
    트랜잭션 언어

PL/SQL 의 장점

- PL/SQL은 SQL에서는 사용할 수 없는 절차적 프로그래밍 기능을 가지고 있습니다.
  다음은 SQL에 비교한 PL/SQL의 장점입니다.

- 프로그램 개발의 모듈화

Block내에서 논리적으로 관련된 문장들을 기술합니다.
강력한 프로그램을 작성하기 위해 Block내에 Sub Block들을 포함합니다.
복잡한 프로그램을 의미있고 잘 정의된 작은 Block들로 나눕니다.

- 변수선언

변수, 상수 등을 선언하여 SQL과 절차형 언어에서 사용합니다.
단일형(Scalar) 데이터 타입과 복합형(Composite) 데이터 타입을 선언합니다.
테이블과 칼럼의 데이터 타입을 기반으로 하는 유동적인 변수를 선언합니다.

PL/SQL 의 장점

- **절차형 언어의 사용**

IF문을 사용하여 조건에 따라 일련의 문장을 실행합니다.
LOOP문을 사용하여 일련의 문장을 반복적으로 실행합니다.
Explicit Cursor를 사용하여 여러 행을 검색합니다.

- **에러처리**

Exception 처리 루틴을 사용하여 Oracle 서버 에러를 처리합니다.
사용자 정의 에러를 선언하고 Exception 처리 루틴에서 처리합니다.

- **이식성**

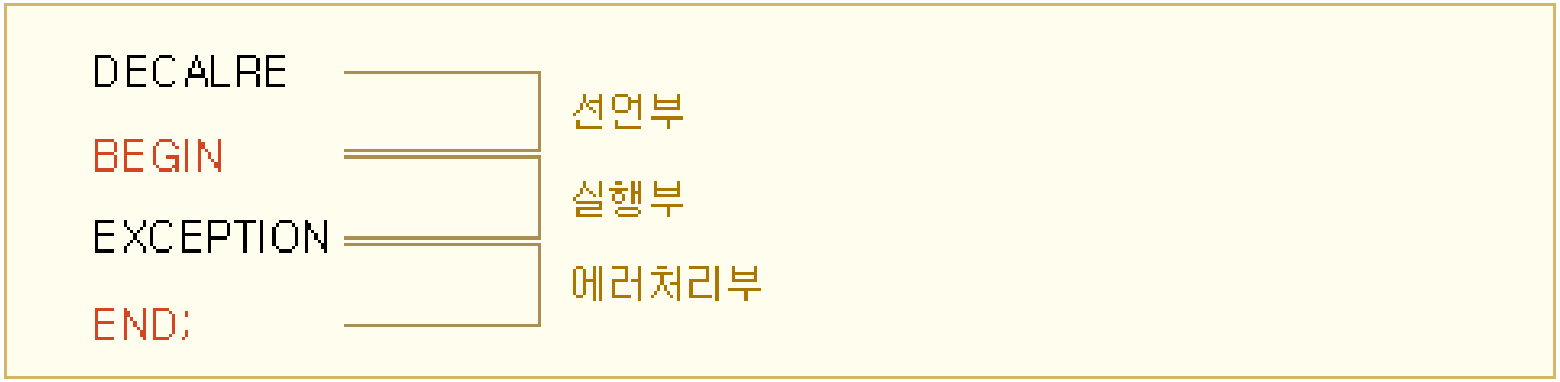
PL/SQL은 Oracle에 내장되어 있으므로 Oracle과 PL/SQL을 지원하는어떤 호스트로도 프로그램을 옮길 수 있습니다.

- **성능 향상**

PL/SQL은 응용 프로그램의 성능을 향상시킵니다.
PL/SQL은 여러 SQL 문장을 Block으로 묶고 한번에 Block 전부를 서버로 보내기 때문에 통신량을 줄일 수 있습니다.

PL/SQL 의 기본 Block

기본적인 PL/SQL 블럭 구조는 세 부분으로 구성됩니다.
블럭의 각 부분은 다음과 같습니다.



부 분	설 명	포 함
선언부 (DECLARE)	실행부에서 참조할 모든 변수, 상수, CURSOR, EXCEPTION을 선언	옵션
실행부 (BEGIN)	데이터베이스의 데이터를 처리할 SQL문과 블럭안의 데이터를 처리할 PL/SQL문을 기술	필수
에러 처리부 (EXCEPTION)	실행부에서 에러와 비정상적인 조건이 발생했을 때 수행될 문장을 기술	옵션

PL/SQL 프로그램 작성요령

- PL/SQL 블럭내에서는 한 문장이 종료할 때마다 세미콜론(;)을 씁니다.
- END뒤에 ;을 사용하여 하나의 블럭이 끝났다는 것을 명시합니다.
- PL/SQL 블럭의 작성은 편집기를 통해 파일로 작성할 수도 있고 SQL 프롬프트에서 바로 작성할 수도 있습니다.
- SQL *Plus환경에서는 DECLARE나 BEGIN이라는 키워드로 PL/SQL 블록이 시작하는 것을 알 수 있습니다.
- CREATE 명령이 실행되기 위해서는 / 가 필요합니다.

PL/SQL 프로그램의 종류

프로그램의 종류	
SUBPROGRAM	FUNCTION : Return a Value PROCEDURE : Do Action
자동 실행 PROGRAM	TRIGGER
복합 프로그램	PACKAGE

FUNCTION의 구조

- 실행환경에 반드시 하나의 값을 돌려줘야 되는 경우에 Function을 생성합니다.
  Function 정의의 Header에 RETURN 될 데이터 타입을 선언하고
  PL/SQL Block에서 RETURN문에 의해 RETURN되는 값을 정의합니다.

```
CREATE [OR REPLACE] FUNCTION function명
[ (parameter1 parameter타입 parameter데이터타입 ,
    parameter2, ...) ]
  RETURN 데이터타입
IS
  변수선언
BEGIN
  RETURN(값);
EXCEPTION
END ;
```

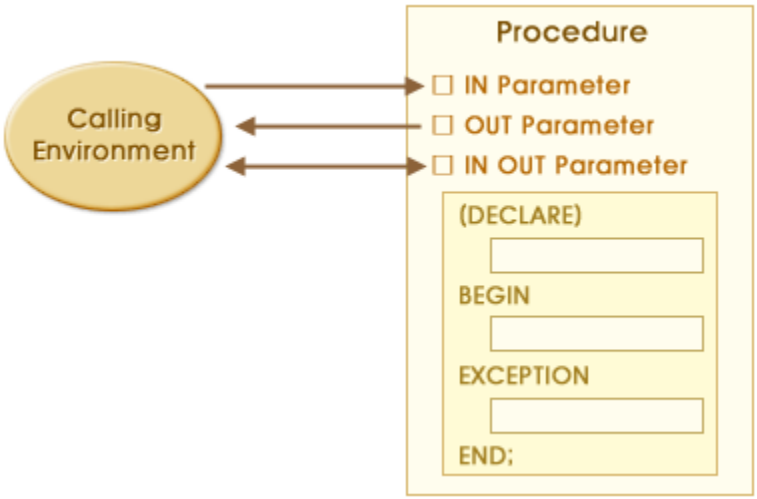
- HEADER의 IS Keyword 다음에 DECLARE는 기본 Block 구조에서 생략되어야 합니다.
  따라서 변수선언이 필요한 경우에는 IS와 BEGIN 사이에 선언합니다.

PROCEDURE의 구조

- 나중에 실행할 일련의 작업을 위해 PL/SQL Procedure를 작성합니다.

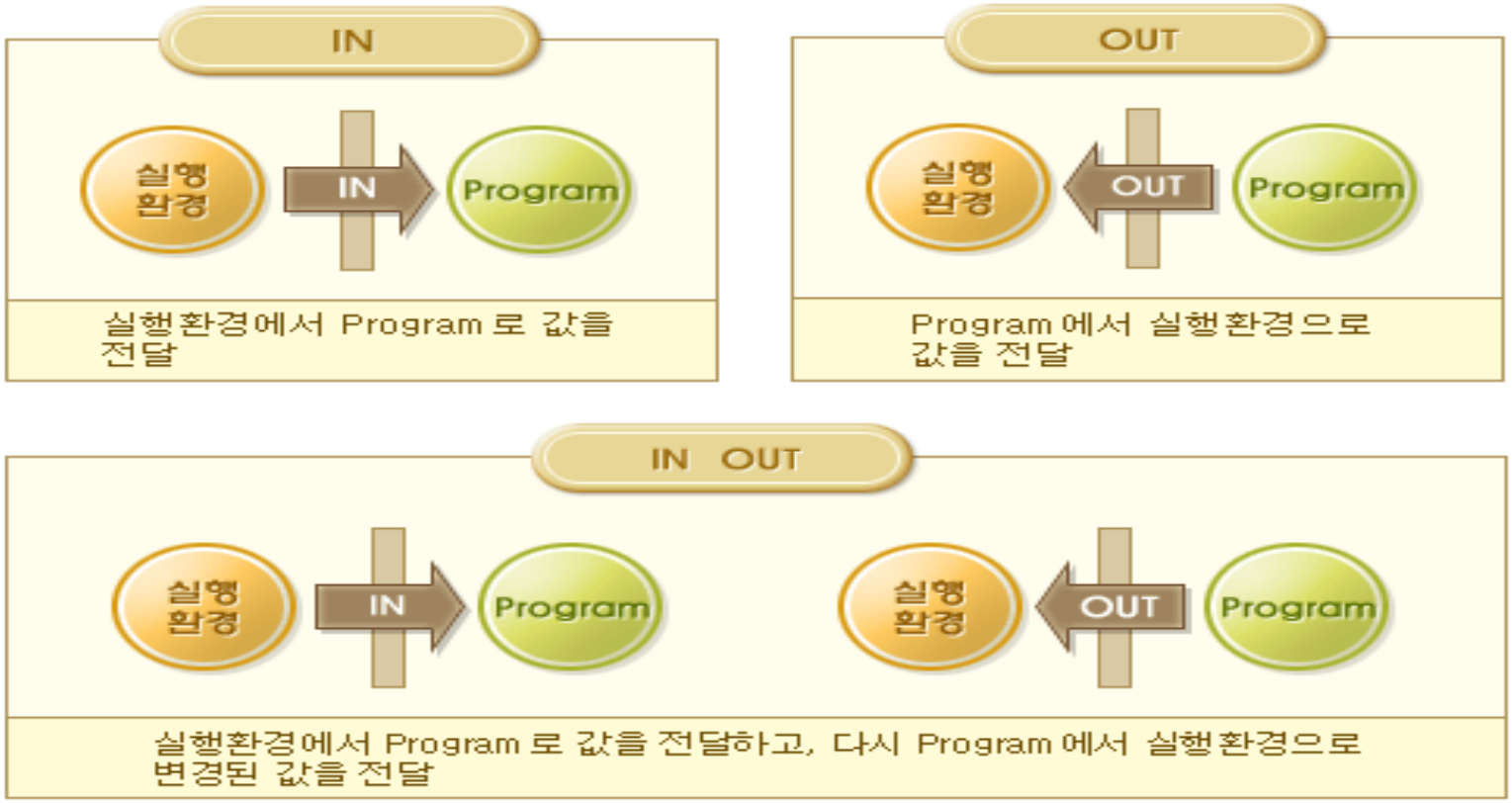
```
CREATE [OR REPLACE] PROCEDURE procedure명
    [ (parameter1 parameter타입 parameter데이터타입 ,
    parameter2, ...) ]
IS
  변수선언
BEGIN

EXCEPTION

END ;
```

Parameter 타입 및 선언방법



- Parameter는 실행환경과 Program사이에 값을 주고 받는 역할을 합니다.
  또한 Block 안에서는 변수와 똑같이 일시적으로 값을 저장하는 역할을 합니다.

Parameter 타입 및 선언방법



- Procedure나 Function은 실행할 때 사용할 Parameter가 없거나 여러 개를 가질 수도 있으며, 사용되는 Parameter에는 실행환경에서 Function/Procedure로 값을 전달하는 IN이 있고, F/P에서 실행환경으로 값을 전달하는 OUT, 양쪽으로 전달하는 IN OUT등이 있다

Parameter 타입 및 선언방법

[예제1] 특정한 수에 세금을 7%로 계산하는 Function을 작성하면 다음과 같습니다.

```
CREATE OR REPLACE FUNCTION tax
(v_num in   number)
RETURN number
IS
  v_tax       number ;
BEGIN
v_tax := v_num * 0.07 ;
  RETURN( v_tax ) ;
END ;
/
```

PL/SQL 프로그램 실행

- Function의 실행 예
  SQL>EXECUTE :a := tax(100)
- Procedure의 실행 예
  SQL>EXECUTE p_tax(100, :a)

[예제3]

- SQL>VARIABLE a NUMBER

  SQL>EXECUTE :a := TAX(100)

- PL/SQL 처리가 정상적으로 완료되었습니다.

  SQL>PRINT a

```
         A
--------
         7
```

  SQL>SELECT name, salary , TAX(salary)
  2 FROM s_emp ;

결과

NAME	SALARY	TAX(SALARY)
박구곤	5000	350
손명성	3000	210
. . .		
이용호	1100	77

25 개의 행이 선택되었습니다.