

# 데이터베이스 프로그래밍-PL/SQL 개요 및 주요특징

## 1 차시: PL/SQL 개요 및 주요 특징

### [학습목표]

PL/SQL 의 기본 개념 및 주요 특징에 대한 기본 개념을 익히고 이를 토대로 PL/SQL의 활용방안을 이해하는 과정

### [학습목차]

- 1-1 PL/SQL 개요
- 1-2 주요 특징 및 장점
- 1-3 실행구조 및 PL/SQL 엔진
- 1-4 개발 Tool
- 1-5 실행환경 및 디버깅 Tool

### 1-1 PL/SQL 개요

PL/SQL 은 데이터베이스 관련 어플리케이션 개발시

- (1) 생산성 (2) 효율성

을 가져다주며 PL + SQL 의 합성어

PL 은 절차적 언어(Procedural Language)의 약어.

SQL 기능에 PL(절차적) 기능을 추가한 언어가 PL/SQL 입니다.

SQL 의 4 가지 특징

- ① RDBMS( 관계형 데이터 베이스) 에 접근(사용되는)하는 유일한 언어
- ② ENGLISH LIKE (영어와 유사한 구조)
- ③ ANSI-SQL
- ④ 비절차적 언어(Non - Procedural Language)

SQL 의 주요 특징중 4 번째 항목인 비절차적 언어라는 특징 과 PL(절차적언어)이라는 용어가 서로 상충되는 의미

비절차적인 특징을 가진 SQL 에 절차적인 언어(Procedural Language)의 특성을 결합한 언어가 PL/SQL. SQL 과 PL/SQL 은 상충되는 이질적인 관계가 아닌 상호 보완적인 관계. 비절차적인 언어의 장점 과 절차적인 언어의 장점을

# 데이터베이스 프로그래밍-PL/SQL 개요 및 주요특징

결합하여 탄생한 언어가 바로 PL/SQL. 이런 연유로 해서 데이터베이스 관련 어플리케이션 개발시 생산성 과 효율성을 동시에 가질수 있게 된것.

<질문> 데이터베이스 관련 어플리케이션을 개발시를 생각해보면

JAVA + SQL , C + SQL , Delphi + SQL 등 2 가지 언어의 조합으로

개발을 해왔습니다. SQL 은 대표적인 비절차적 언어 이며

JAVA 나 C , Delphi 등의 여러분이 다루시는 일반적인 개발언어는

절차적 언어 입니다. 그동안 개발 해오셨던 개발언어의 조합도 분석을 해보면 절차적 언어 + 비절차적 언어의 조합인 것 입니다.

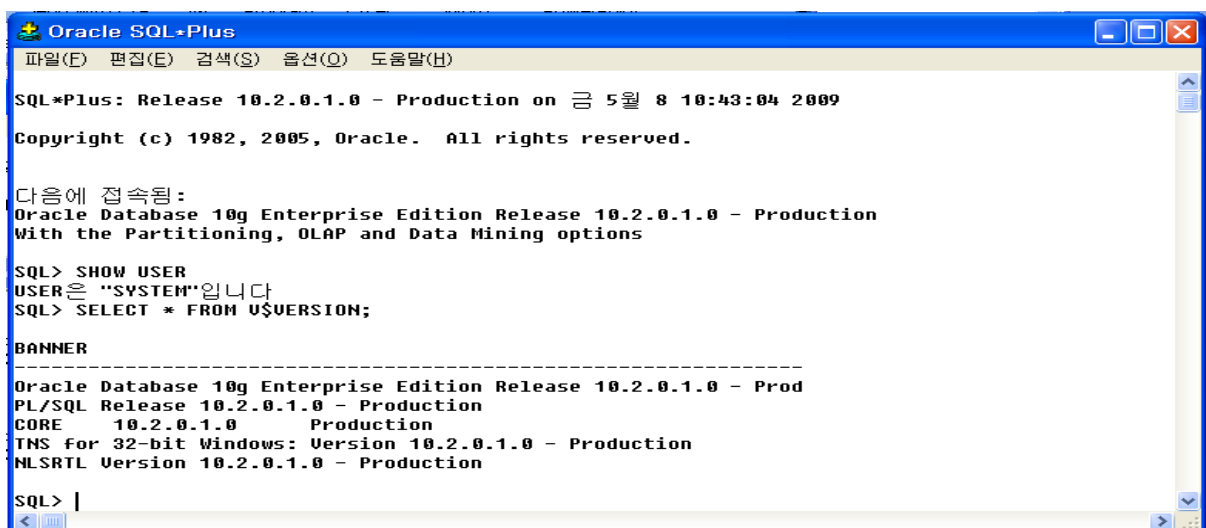
그렇다면 JAVA + SQL 개발 환경과 JAVA + PL/SQL 개발 환경은 개발 방법 및 생산성, 효율성 측면에서 어떤 차이가 있을까요?

< 참고: PL/SQL 의 버전>

DBMS 버전	PL/SQL 버전
ORACLE 6	1.0
ORACLE 7.x	2.X
ORACLE 8.X	8.X
ORACLE 9.X	9.X
ORACLE 10.X	10.X
ORACLE 10.X	11.X

PL/SQL 은 ORACLE DBMS 6.X 버전에서 처음 지원.

ORACLE DBMS 7.X 버전에서는 PL/SQL 은 2.X 버전으로 표기가 되어서 간혹 버전 표기 때문에 헷갈려 하는 경우도 있었지만 ORACLE DBMS 8.X 부터 PL/SQL 도 DBMS 와 동일한 버전으로 표기가 통일.



# 데이터베이스 프로그래밍-PL/SQL 개요 및 주요특징

CONN SYSTEM

SELECT \* FROM V\$VERSION;

## 1-2 주요 특징 및 장점

절차적 언어는 주요 특징

- (1) 변수 선언
- (2) 제어 구조(Control Structure)
- (3) 예외처리(Exception) 처리
- (4) 모듈화(Modular Programming)

### (1) 변수선언

선언부(DECLARE ~ BEGIN)에서 변수/상수/커서/사용자 정의 예외를 정의하여 사용

```
DECLARE
    V_EMPNO      NUMBER(4)      := 8888;  -- 변수선언 및 초기화
    V_DEPTNO     NUMBER(2);
    V_ENAME      VARCHAR2(10) := 'XMAN'; -- 변수선언 및 초기화
    V_JOB        VARCHAR2(9);
    V_SAL        NUMBER(7,2);
BEGIN
```

Table 컬럼 정의:	컬럼명	테이타 타입(데이터 길이)
Ex)	EMPNO	NUMBER(4)

변수 정의	:	변수명	데이터 타입(데이터 길이)
Ex)		V_EMPNO	NUMBER(4)

```
DECLARE
    CURSOR CUR_EMP IS
        SELECT EMPNO, JOB, SAL, COMM FROM EMP WHERE DEPTNO = 10;

    V_ENAME      VARCHAR2(10);
    V_JOB        VARCHAR2(9);
    V_SAL        NUMBER(7,2);
    V_COMM       NUMBER(7,2);
BEGIN
```

# 데이터베이스 프로그래밍-PL/SQL 개요 및 주요특징

## (2) 제어 구조

제어는 프로그램의 처리(실행) 흐름을 제어하는 조건문/반복문/GOTO 문

```
IF V_JOB IS NULL THEN
    V_JOB := '신입';
END IF;

IF V_JOB = '신입' THEN
    V_SAL := 2000;
ELSIF V_JOB IN ('MANAGER', 'ANALYST') THEN
    V_SAL := 3500;
ELSE
    V_SAL := 2500;
END IF;

WHILE(V_INDEX >= 0 )
LOOP
    DBMS_OUTPUT.PUT_LINE(' WHILE LOOP ['||TO_CHAR(V_INDEX)||' ]');
    V_INDEX := V_INDEX - 1;
END LOOP;
```

## (3) 예외처리(Exception)

JAVA 나 C++ 같은 최근의 젊은 언어(^^) 에서는 예외처리를 개발 언어 차원에서 지원.

```
public class exceptions{
    public static void main(String Args[]){
        int[] array = new int[3];
        try{
            for(int i=0;i<3;++i){
                array[i] = i;
            }
            array[0] = 2/0;
        }
        catch(ArrayIndexOutOfBoundsException e){
            System.out.println("Oops, we went to far, better go back to 0!");
        }
        catch(Exception e){
            System.out.println("An Unknown Error has Occured");
            e.printStackTrace();
        }
        catch(ArithmeticException e){
            System.out.println("Cannot Divide by Zero!");
            //method call to continue program
        }
        finally{
            System.out.println(array);
            //method call to continue program
        }
    }
}
```

## 데이터베이스 프로그래밍-PL/SQL 개요 및 주요특징

```
<<Nested_BLOCK_1>>
BEGIN
    INSERT INTO DEPT VALUES(76,'LOCAL_PART_1','Nested_Blk1');
    INSERT INTO DEPT VALUES(777,'LOCAL_PART_1','Nested_Blk1'); -- Run Time Error 발생
    --INSERT INTO DEPT VALUES(77,'LOCAL_PART_1','Nested_Blk1');
    INSERT INTO DEPT VALUES(78,'LOCAL_PART_1','Nested_Blk1');
    COMMIT;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        NULL;
    WHEN OTHERS THEN
        NULL;
END Nested_BLOCK_1;
```

JAVA 의 CATCH 와 PL/SQL 의 WHEN 절

예외처리의 목적: 안정적이고 오작동하지 않는 개발

### (4) 모듈화(Modular Programming)

프로그램 개발시 모듈단위로 나누어 개발 하거나 독립 모듈(ex 사용자 정의 함수/프로시저) 을 만들어 라이브러리(Library)화 시켜 재사용.

PL/SQL 은 BLOCK 구조화된 언어로 BLOCK 단위를 통해 모듈화.

```
DECLARE
    V_EMPNO          NUMBER(4)      := 0;      -- 변수선언 AND 초기값 할당
    V_ENAME           VARCHAR2(10);  -- 변수선언, 초기화를 하지 않은경우는?
    V_DEPTNO          NUMBER(2);

BEGIN
    V_EMPNO := 7778;                  -- 대입연산자: := , 비교연산자: =
    V_ENAME := 'PL/SQL';

    INSERT INTO EMP(EMPNO,DEPTNO,ENAME) VALUES(V_EMPNO,V_DEPTNO,V_ENAME);

    COMMIT;

EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('INSERT ERR :'||SQLERRM);
        ROLLBACK;

END;
/
```

BLOCK 안에 BLOCK 을 사용하는 중첩 BLOCK(Nested Block).

## 데이터베이스 프로그래밍-PL/SQL 개요 및 주요특징

```
<<MAIN_BLK>>
DECLARE
    V_DNAME          VARCHAR2(14);
    V_DEPTNO         NUMBER(2);
    V_LOC            VARCHAR2(13);

BEGIN
    V_DEPTNO := 77;
    V_DNAME := 'GLOBAL_PART';
    V_LOC := 'Main_Blk';

    <<LOCAL_BLOCK_1>>
    DECLARE
        V_DNAME          VARCHAR2(14);
        V_DEPTNO         NUMBER(2);

    BEGIN
        V_DEPTNO := 88;
        V_DNAME := 'LOCAL_PART_1';
        V_LOC := 'Nested_Blk1';
        INSERT INTO DEPT VALUES(V_DEPTNO,MAIN_BLK.V_DNAME,V_LOC);
    END LOCAL_BLOCK_1;

    <<LOCAL_BLOCK_2>>
    DECLARE
        V_DNAME          VARCHAR2(14);
        V_DEPTNO         NUMBER(2);

    BEGIN
        V_DEPTNO := 99;
        V_DNAME := 'LOCAL_PART_2';
        V_LOC := 'Nested_Blk2';
        INSERT INTO DEPT VALUES(V_DEPTNO,V_DNAME,V_LOC);
    END;

    INSERT INTO DEPT VALUES(V_DEPTNO,V_DNAME,V_LOC);
END MAIN_BLK;
/
```

중첩 BLOCK 을 통해 모듈성을 높일수있다

실행시 에러가 발생할 때 마다 에러를 기록하는 프로시저로 NAMED BLOCK 은 이름을 가지고(EX WRITE\_LOG) DBMS 서버내에 저장되어 공용 사용

```
REM -----
REM EXCEPTION을 기록하는 WRITE_LOG PROCEDURE 생성
REM -----

CREATE OR REPLACE PROCEDURE
    WRITE_LOG(A_PROGRAM_NAME IN VARCHAR2,A_ERROR_MESSAGE IN VARCHAR2,A_DESCRIPTION IN VARCHAR2)
AS
BEGIN
    -- EXCEPTION을 LOG 테이블에 기록
    INSERT INTO EXCEPTION_LOG(PROGRAM_NAME,ERROR_MESSAGE,DESCRIPTION)
        VALUES(A_PROGRAM_NAME,A_ERROR_MESSAGE,A_DESCRIPTION);

    COMMIT;
EXCEPTION
    WHEN OTHERS THEN
        NULL;
END;
/
```

NAMED BLOCK 으로 함수,프로시저,패키지,트리거 생성

# 데이터베이스 프로그래밍-PL/SQL 개요 및 주요특징

## PL/SQL 의 장점

- (1) 이식성(Portability)
- (2) 통합성(Intergration)
- (3) 성능(Performance)

### (1)이식성(Portability)

이식성은 개발 생산성 및 유지보수/기능개선 비용을 줄여주는 특징.

ORACLE DBMS 는 약 70 여종의 OS 및 H/W 플랫폼에서 구동. ORACLE DBMS 가 지원하는 모든 플랫폼간에 PL/SQL 로 작성된 것은 그대로 호환 .

JAVA 언어의 장점중 하나인“Write once Run anywhere”와 동일한 개념.

NT 에서 운영되는 ORACLE DBMS 관련 프로젝트에서 개발된 PL/SQL 개발 산출물을 UNIX 상의 ORACLE DBMS 로 프로그래밍 변경 없이 이식(이관).

오늘날의 복잡한 H/W,S/W 개발환경 및 운영환경에서 이식성은 개발 생산성 및 유지 보수 비용을 줄여줄 수 있는 매우 중요한 장점.

예를 들면 이식성이 매우 낮은 프로젝트인

WINDOW NT 계열에서 VC++로 구축된 응용 시스템을

SUN SOLARIS 의 UNIX OS 상의 JAVA 로 이식(이관) 하는 프로젝트에

여러분이 참여한다면 여러분은 얼마나 많은 밤을 지새워야 할까요?

### (2) 통합성(Intergration)

통합성은 개발 생산성과 효율성을 주는 주요 특징.

PL/SQL 로 개발하는 Module 은 Anonymous Block 과 Named Block 2 가지.

Anonymous Block 은 Client 프로그램내에 저장 되어 있다가 실행시점에 DBMS 서버에 전달되어 실행.

Named Block 은 DBMS 서버내에 저장되어 있다가 DBMS 서버내에서 실행.

Named Block 은 DBMS 서버내에 통합되어 저장/실행/관리.

DATA 는 DBMS 서버내에 저장되어 있고 PL/SQL 의 주요 목적은 데이터 처리.

처리할 대상 데이터와 데이터 처리 로직이 동일한 위치에 통합되어 처리(실행).

### (3) 성능(Performance)

PL/SQL Module 의 이식성 및 통합성으로 인해성능 향상의 장점을 가짐.

## <참고>

“ PL/SQL groups SQL statements together within a single block and

# 데이터베이스 프로그래밍-PL/SQL 개요 및 주요특징

send the entire block to the server in a single call, therefore reducing network traffic “

PL/SQL 은 Client 프로그램과 DBMS 서버상에 주고 받아야 하는 여러 SQL 을 하나의 덩어리(BLOCK)단위로 처리 하므로 네트워크 트래픽을 줄여서 PL/SQL 을 사용하면 성능 향상이 된다 ??

여러 번 주고 받아야 하는 것을 묶어서 하나의 덩어리(BLOCK) 처리하면 네트워크 트래픽을 줄이는 효과는 나타나지만 PL/SQL 의 성능 향상은 데이터 와 데이터 처리 로직이 DBMS 서버내에 통합되어 처리되는 통합성에 의한 것.

## < 참고>

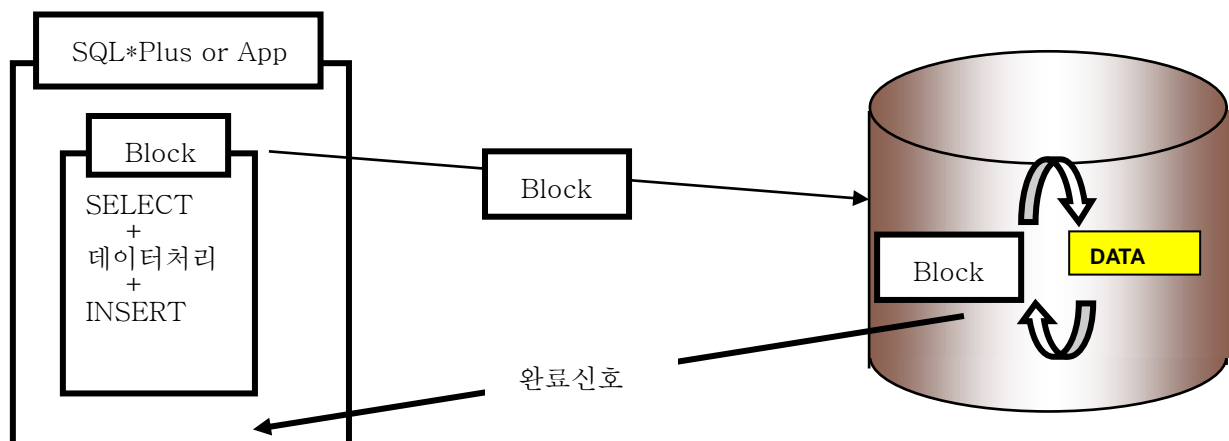
ORACLE DBMS 8i 이후 버전부터 ORACLE DBMS 를 설치하면 자바 가상 머신(JVM)이 설치가 되며 이를 통해 JAVA SOURCE 를 ORACLE DBMS 내에 데이터 디렉터리로 적재(LOAD)하여 PL/SQL 에서 JAVA 를 호출하여 사용하는 것이 가능.

## 1-3 실행구조 및 PL/SQL 엔진

PL/SQL 로 작성하는 BLOCK 은 ANONYMOUS BLOCK 과 NAMED BLOCK 이 있으며 개괄적인 관점에서 실행구조를 이해할 필요가 있다.

ANONYMOUS BLOCK 은 BLOCK 의 이름이 없고 Client Program 내에 존재하며 NAMED BLOCK(=STORED BLOCK)은 이름을 가지고 있고 DBMS 서버내에 존재 한다.

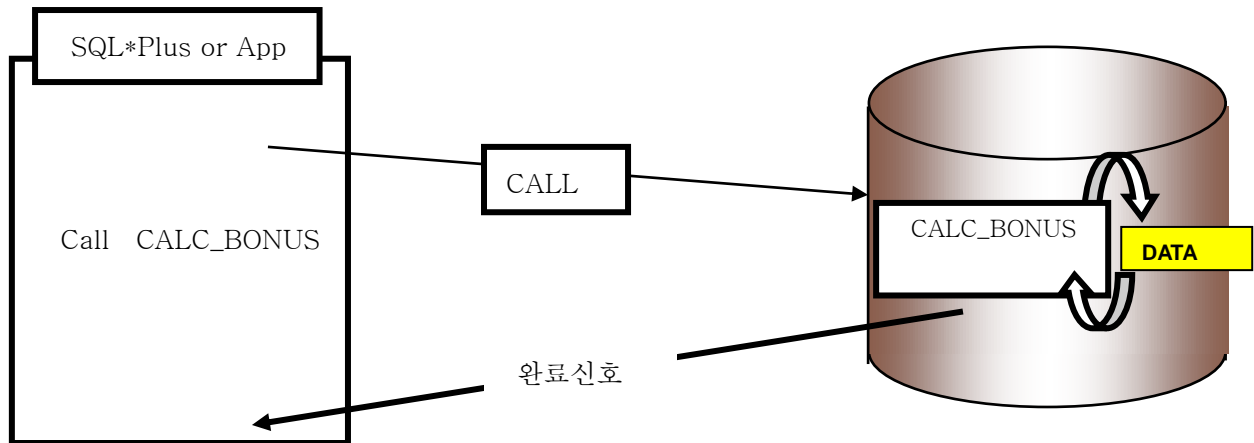
ANONYMOUS BLOCK 은 실행시점에 CLIENT 내에 있던 BLOCK 이 네트워크를 통해 DBMS 서버에게 전달되어 서버내에서 실행된후 완료 신호가 CLIENT 에게 전달되는 구조로 실행.





# 데이터베이스 프로그래밍-PL/SQL 개요 및 주요특징

NAMED BLOCK(=STORED BLOCK)은 DBMS 서버내에 저장되어 있으며 CLIENT에서는 서버에 저장된 해당 BLOCK을 호출(CALL)하여 실행한후 완료신호가 CLIENT에게 전달되는 구조로 실행.



## 1-3-2 PL/SQL 엔진

PL/SQL BLOCK을 보면 변수선언 및 IF 조건은 PL/SQL의 기능이며 INSERT는 SQL 기능.

```
DECLARE
    V_EMPNO      NUMBER(4)      := 8888;  -- 변수선언 및 초기화
    V_DEPTNO     NUMBER(2);
    V_ENAME      VARCHAR2(10) := 'XMAN';  -- 변수선언 및 초기화
    V_JOB        VARCHAR2(9);
    V_SAL        NUMBER(7,2);
BEGIN
    V_DEPTNO := 20;  -- 변수에 값을 대입

    IF V_JOB IS NULL THEN
        V_JOB := '신입';
    END IF;

    IF V_JOB = '신입' THEN
        V_SAL := 2000;
    ELSIF V_JOB IN ('MANAGER', 'ANALYST') THEN
        V_SAL := 3500;
    ELSE
        V_SAL := 2500;
    END IF;

    INSERT INTO EMP(DEPTNO, EMPNO, ENAME, SAL, JOB)
    VALUES(V_DEPTNO, V_EMPNO, V_ENAME, V_SAL, V_JOB);

    COMMIT;
END;
```

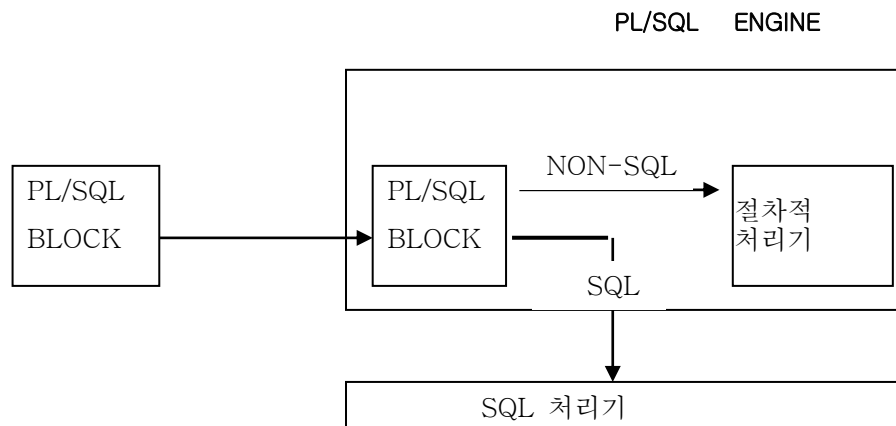
위의 ANONYMOUS BLOCK이 실행되면 해당 BLOCK은 DBMS 서버에 전달됩니다. DBMS 서버내에 PL/SQL 엔진에게 PL/SQL BLOCK이 전달됩니다.

PL/SQL 엔진에서는 PL/SQL BLOCK을 분석하여

PL/SQL 기능은 절차적 처리기 (Procedural Statement Executor)에게 보내어 실행하고

SQL 기능은 SQL 처리기 (SQL Statement Executor)에게 보내어 실행 합니다.

# 데이터베이스 프로그래밍-PL/SQL 개요 및 주요특징



## 1-4 개발 Tool

ORACLE 社 에서 기본으로 제공되는 SQL\*PLUS, SQL Developer TOOL 이외에도 SQL Navigator , Toad, PL/SQL Developer 등 PL/SQL 을 개발하는데 사용될수 있는 여러 상용툴이 있습니다. 상용 Tool 의 장점으로 문법 CHECK 및 디버깅의 편리성등이 제공 되어 생산성이 좋아지만 TOOL 에 의존적이게 되므로 기본적으로 제공되는 SQL\*PLUS 를 사용하여 진행.

SQL\*PLUS 상에서도 PL/SQL BLOCK 을 개발/테스트/디버깅이 가능 합니다.

개발 ~ SQL 문장 작성하듯이 PL/SQL BLOCK 을 작성.

테스트 ~ SQL\*PLUS 명령어중 VARIABLE 과 PRINT 를 통해서 테스트 수행

디버깅 ~ 가장 간단하면서도 직관적인 DBMS\_OUTPUT 을 사용

## 1-5 실습환경 및 디버깅 Tool

### 1-5-1 실습환경 생성

실습 TOOL	: SQL*PLUS
실습 계정	: SCOTT 계정
실습 환경 생성 스크립트	: DEMOBLD.SQL

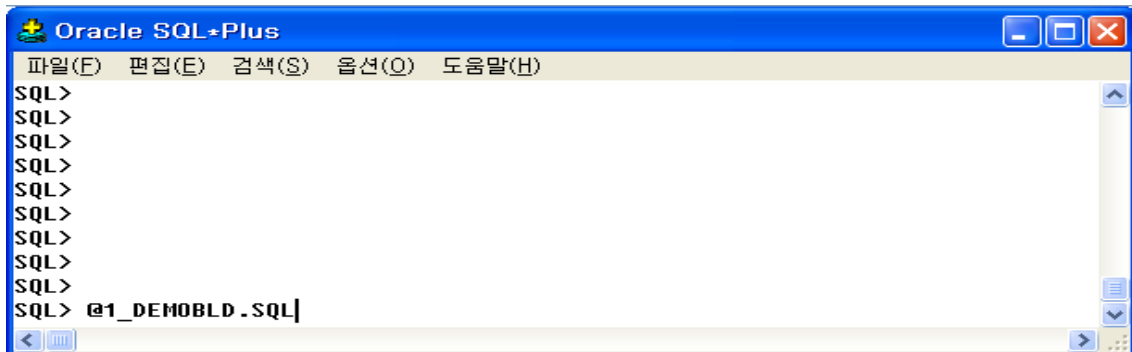
실습 환경 생성

1. DEMOBLD.SQL FILE Download

# 데이터베이스 프로그래밍-PL/SQL 개요 및 주요특징

2. SQL\*PLUS 를 통해 실습 계정으로 LOGIN

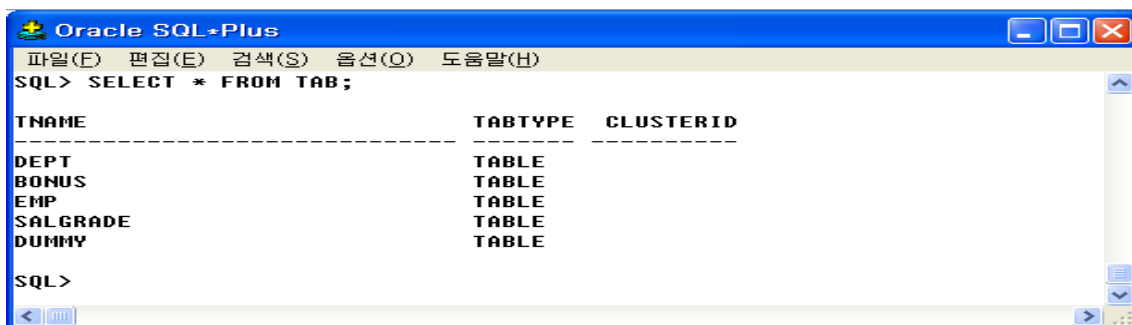
3. 1\_ DEMOBLD.SQL 실행



DEMOBLD.SQL을 실행하면 실습환경 생성후 SQL\*PLUS를 자동으로 종료.

4. 실습환경 생성 확인

SQL\*PLUS로 LOGIN하여 실습환경 생성 여부를 확인합니다.



SELECT \* FROM TAB; 를 실행하여 EMP,BONUS,DEPT 3개의 테이블이 조회되면 실습환경 생성

## 1-5-2 디버깅 TOOL

**DBMS\_OUTPUT**은 ORACLE社에서 디버깅 Tool 용도로 제공하는 **PACKAGE**.

① ORACLE社에서 디버깅 Tool용도로 제공하는 PACKAGE

② PL/SQL BLOCK이 실행되는 동안에는 **메모리상에 출력 결과를 저장해 두었다가**

PL/SQL BLOCK의 실행이 종료된후 **메모리의 결과를 SQL\*PLUS의 화면상에 출력**

<참고> DBMS\_OUTPUT에 대한 오해는 실행 시점에 실시간으로 결과를 출력한다고

생각하는 경우가 많지만 PL/SQL 실행이 종료된후 해당 내용을 화면에 일괄적으로 출력

③ SQL\*PLUS의 환경변수 인 SERVEROUTPUT 을 설정해야 출력 결과가 나타난다.

**SET SERVEROUTPUT ON**

# 데이터베이스 프로그래밍-PL/SQL 개요 및 주요특징

1\_DBMS\_OUTPUT\_1.SQL

```
REM  DEFAULT SIZE  20000  BYTES.
SET  SERVEROUTPUT ON

BEGIN
    FOR I IN 1..10
    LOOP
        DBMS_OUTPUT.PUT_LINE('[' ||TO_CHAR(I)||' ] PROCESSED');
    END LOOP;
END;
/

SET SERVEROUTPUT OFF
BEGIN
    -- OFF가 되어서 SQL*PLUS화면상에 출력되지 않고 메모리상에도 기록하지 않는다.
    DBMS_OUTPUT.PUT_LINE('OK...');
END;
/
```

1\_DBMS\_OUTPUT\_2.SQL

```
REM -----
REM  출력 버퍼의 크기 조절
REM  - DEFAULT SIZE 20000 BYTES
REM  - MAX SIZE   UNLIMITED
REM -----

SET  SERVEROUTPUT ON SIZE 2000

BEGIN
    FOR I IN 1..40
    LOOP
        DBMS_OUTPUT.PUT_LINE('[' ||TO_CHAR(I)||' ] '||
                               '12345678901234567890123456789012345678901234567890');
    END LOOP;
END;
/
```

- 출력 SIZE를 초과하는 경우 buffer overflow에 에러를 유발합니다.

<참고> DBMS\_OUTPUT을 사용할 때 버퍼의 길이

라인의 출력 길이 :

10G R2 이전	255 Bytes
10G R2 이후부터	32767 Bytes

출력 버퍼 크기

10G R2 이전	1000000 Bytes
10G R2 이후부터	Unlimited

# 데이터베이스 프로그래밍-PL/SQL 개요 및 주요특징

학습 정리

1. PL/SQL 은 SQL 의 장점에 PL(Procedural Language)의 기능을 추가한 언어입니다.

2. PL/SQL은

절차적 언어 특성인 변수선언,제어구조,예외처리, 모듈화 가 가능하며  
장점으로는 이식성, 통합성,성능 이다.

3. NAMED BLOCK 은 이름을 가지고(EX WRITE\_LOG) DBMS 서버내에 저장되어 공용으로 사용할수 있습니다. 즉 모듈화된 공용 LIBRARY!

```
REM -----
REM EXCEPTION을 기록하는 WRITE_LOG PROCEDURE 생성
REM -----

CREATE OR REPLACE PROCEDURE
  WRITE_LOG(A_PROGRAM_NAME IN VARCHAR2,A_ERROR_MESSAGE IN VARCHAR2,A_DESCRIPTION IN VARCHAR2)
AS
BEGIN
  -- EXCEPTION을 LOG 테이블에 기록
  INSERT INTO EXCEPTION_LOG(PROGRAM_NAME,ERROR_MESSAGE,DESCRIPTION)
    VALUES(A_PROGRAM_NAME,A_ERROR_MESSAGE,A_DESCRIPTION);
  COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    NULL;
END;
/
```

NAMED BLOCK 으로 함수,프로시저,패키지,트리거를 작성할수 있습니다.

4.PL/SQL 엔진의 주요 기능은

PL/SQL BLOCK을 분석하여 SQL은 SQL처리기로 보내고

PL/SQL 기능은 절차적 처리기에 보내어

처리하게 한다.

5.SQL\*PLUS상에서 PL/SQL BLOCK을

개발/테스트/디버깅이 가능하다.