01. PL/SQL 개요 및 주요특징

목차

1. PL/SQL 개요

▣ 개요

- PL/SQL은 데이터베이스 관련 어플리케이션 개발시
  (1) 생산성 향상 (2) 효율성 증대

- PL/SQL = PL(Procedural Language: 절차적 언어) + SQL
  * SQL 기능에 PL(절차적) 기능을 추가한 언어

  * SQL의 4가지 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)

▣ 상충 (비절차적 언언 vs 절차적)

- 이질적인 관계가 아닌 상호 보완적인 관계
  비절차적인 언어의 장점 과 절차적 언어의 장점을 결합하여 탄생한 언어

1. PL/SQL 개요

* <참고> PL/SQL 버젼

DBMS version	PL/SQL version
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 19.X	19.X

- ORALCE DBMS 8.X 부터 PL/SQL도 DBMS와 동일한 버전으로 통일

* <참고> DBMS별 Procedural Language 호칭

DBMS	Naming
Oracle	PL/SQL
MySQL	Stored Program
MS SQL Server	T-SQL (Transact-SQL)
IBM DB2	PL/SQL

2. 주요 특징

▣ 절차적 언어의 특징

(1) 블록(Block) 구조

(2) 변수 선언

(3) 제어 구조(Control Structure)

(4) 예외처리(Exception) 처리

(5) 모듈화(Modular Programming)

2. 주요 특징

(1) 블록(Block) 구조

- PL/SQL is a block-structured language
- BLOCK은 ① 선언부 ② 실행부 ③ 예외처리부 로 구성

```
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;
/
```

① 선언부(Declare section)

- **변수,상수,커서,사용자정의 예외등을 정의(선언)**
- **선언부는 선택적 영역 (Optional section) 으로 선언(정의)할 대상이 없는 경우 생략 가능**

② 실행부(Execution section)

-선언부에서 정의한 변수에 값을 대입하거나 **연산을 실행하는 영역.**
- Block 내에서 **필수적 영역(Mandatory section)**
- BEGIN 으로 시작하여 END로 종료

<참고> PL/SQL에서 문장 종결자 세미콜론(;)
BLOCK 의 문장 종결자 세미콜론(;)

③ 예외처리부(Exception section)

- **예외(실행시간에 발생한 에러)를 처리하는 영역.**
- Block 내에서 **선택적 영역 (Optional section)** 으로 처리해야할 예외가 없는 경우 생략 가능.

2. 주요 특징

(2) 변수 선언

- 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
```

- 변수 선언
- 선언 및 초기화

```
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
```

- **커서(Cursor)** 선언(정의)

<참고> Table 컬럼 정의 Ex) EMPNO NUMBER(4),
변수 정의 Ex) V_EMPNO NUMBER(4);

2. 주요 특징

(3) 제어 구조(Control Structure)

- 프로그램의 처리(실행)흐름 제어(control flow) 하는 조건문/반복문/GOTO문

```
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;
```

- 조건문

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

- 반복문
  ex) while , for , loop

2. 주요 특징

(4) 예외처리(Exception) 처리

- JAVA or C++ 같은 최신언어 (?) 처럼 예외처리 기능 지원
- CATCH 와 WHEN절 , 예외처리 목적: ???

```
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
        }
    }
}
```

```
<<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;
```

2. 주요 특징

(5) 모듈화(Modular Programming)

- 프로그램 개발시 모듈단위로 나누어 개발 하거나 독립 모듈(Function / Procedure / Package) 를 만들어 라이브러리(Library)화 시켜 재사용.

Anonymous Block(= Client stored block)

```
<<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 LOAL_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;
/
```

Named Block (= Server Stored Block)

```
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;
/
```

3. 장점

▣ 이식성(Portability)

이식성은 개발 생산성 및 유지보수/기능개선 비용을 줄여주는 특징.
ORACLE DBMS는 약 70여종의 OS 및 H/W 풀랫폼에서 구동. ORACLE DBMS가 지원하는 모든 플랫폼간에 PL/SQL로 작성된 Block은 그대로 호환 .
JAVA 언어의 장점중 하나인“Write once Run anywhere”와 동일한 개념.
NT에서 운영되는 ORACLE DBMS 관련 프로젝트에서 개발된 PL/SQL개발 산출물을UNIX상의 ORACLE DBMS로 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.

▣ 통합성(Intergration)

통합성은 개발 생산성과 효율성을 높혀주는 특징.
PL/SQL로 개발하는 Module은 Anonymous Block 과 Named Block 2가지.
Anonymous Block은 Client 프로그램내에 저장 되어 있다가 실행시점에 DBMS서버에 전달되어 실행.
Named Block은 DBMS 서버내에 저장되어 있다가 DBMS 서버내에서 실행.
Named Block은 DBMS 서버내에 통합되어 저장/실행/관리.

DATA는 DBMS 서버내에 저장되어 있고 PL/SQL의 주요 목적은 데이터 처리.
처리할 대상 데이터와 데이터 처리 로직이 동일한 위치에 통합되어 처리(실행).

3. 장점

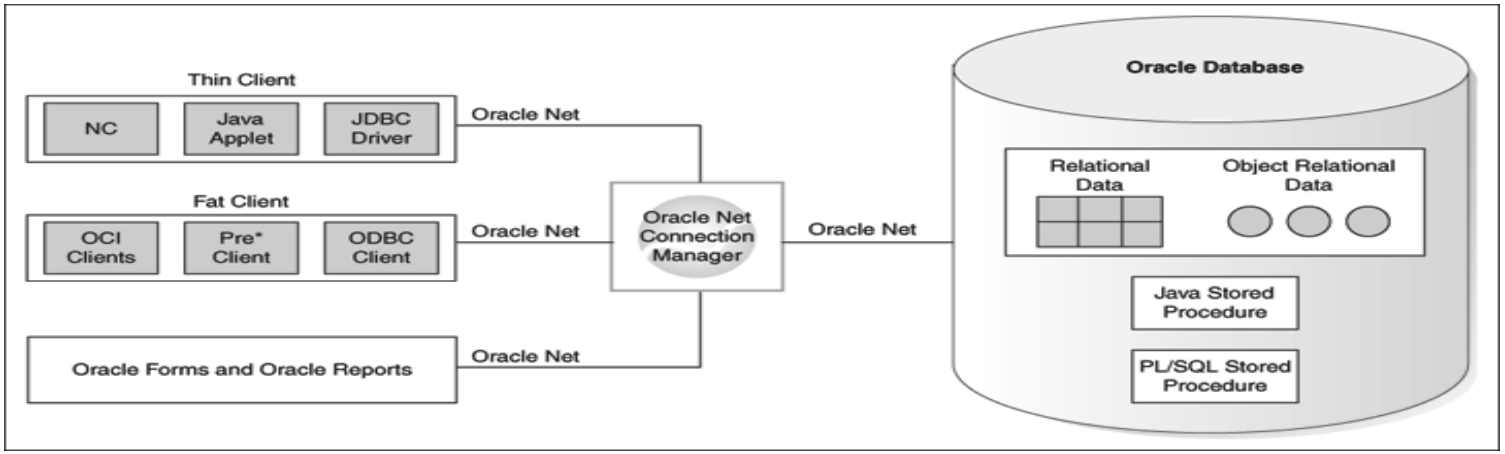
▣ 성능(Performance)

PL/SQL Module의 이식성 및 통합성으로 인해 성능 향상의 장점을 가짐.

“ PL/SQL groups SQL statements together within a single block and
send the entire block to the server in a single call, therefore reducing network traffic “
“ Reduce network traffic – Application과 MySQL Server 사이에 Network traffic을 줄여서 성능 향상된다 "

<질문>
PL/SQL은 Client 프로그램과 DBMS 서버상에 주고 받아야 하는 여러 SQL을 하나의 덩어리(BLOCK)단위로 처리 하므로 네트웍 트래픽을 줄여서 PL/SQL을 사용하면 성능 향상 된다 ??
여러 번 주고 받아야 하는 것을 묶어서 하나의 덩어리(BLOCK) 처리하면 네트웍 트래픽을 줄이는 효과는 나타나지만 PL/SQL의 성능 향상은 데이터 와 데이터 처리 로직이 DBMS 서버내에 통합되어 처리되는 통합성에 의한 것.

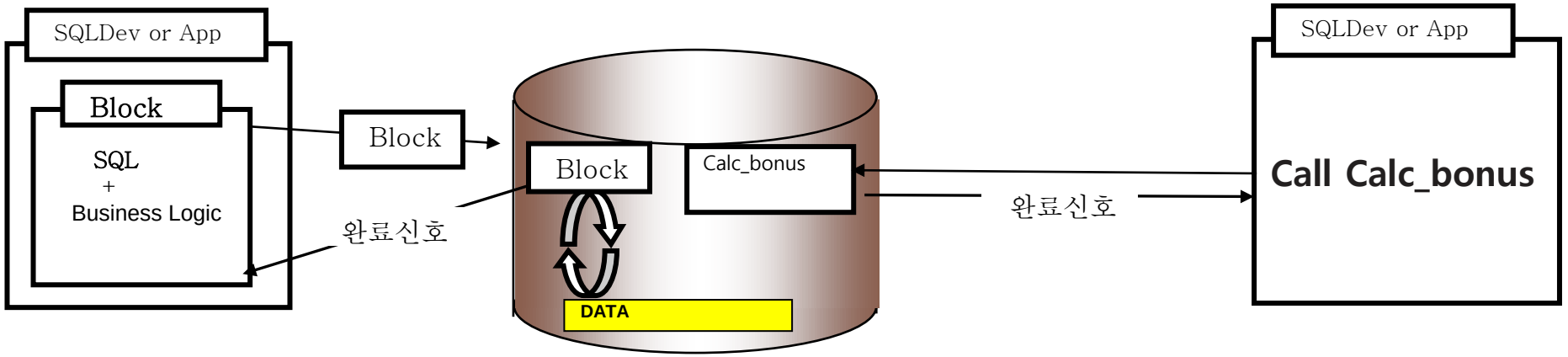
* <참고>
Oracle JVM



4. 실행구조 및 PL/SQL 엔진

▣ 실행구조

	Anonymous Block	Named Block
Block 이름	X	O
Source 저장위치	**Client program or SQL Script file**	DBMS Server.**Data dictionary**
주용도	PL/SQL을 사용하여 효율(개발 및 실행)적인 데이터처리 구현	재사용 가능한 모듈 (공통 Library) ex) function, procedure,package
실행구조	실행시점에 Client내에 있는 Anonymous Block이 네트웍을 통해 DBMS 서버에 전달되어 compile & execution	DBMS 서버내에 저장되어 있으며 Client에서는 서버에 저장된 해당 Block을 호출(CALL)하여 execution



4. 실행구조 및 PL/SQL 엔진

▣ PL/SQL 엔진

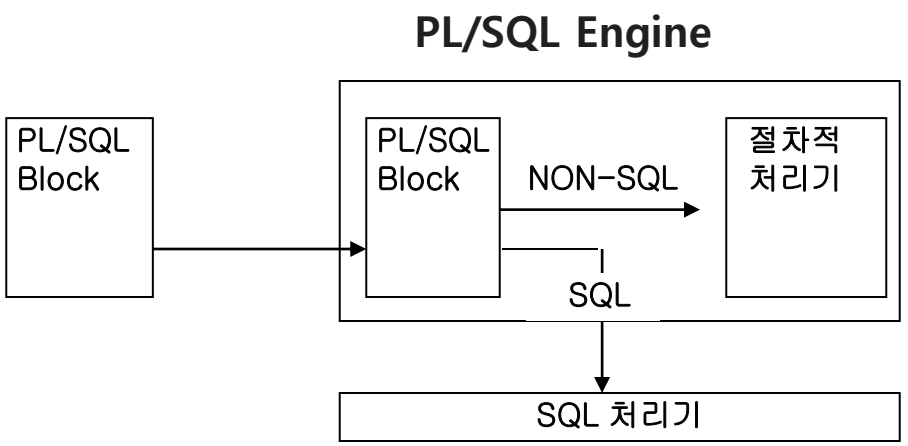
```
DECLARE
        V_EMPNO         NUMBER(4)    := 8888;   -- 변수선언 및 초기화
        V_DEPTNO        NUMBER(2);
        V_ENAME         VARCHAR2(10) := 'XMAN'; -- 변수선언 및 초기화
        V_JOB           VARCHAR2(9);
        V_SAL           NUMBER(7,2);
BEGIN
        V_DEPTNO := 20;         -- 변수에 값을 대입

        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;
/
```



PL/SQL 엔진에서는 PL/SQL Block을 분석하여

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

- SQL 기능은 SQL 처리기(SQL Statement Executor)에게 보내어 실행

5. 개발 Tool

▣ 도구

- ORACLE社에서 기본으로 제공되는 SQL Developer 이외에도 SQL Navigator , Toad, PL/SQL Developer 등 PL/SQL을 개발하는데 사용될수 있는 여러 상용툴
- 상용 Tool의 장점으로 문법 CHECK 및 디버깅의 편리성등이 제공 되어 개발 생산성이 향상되지만 Tool 의존적이게 되므로 기본적으로 제공되는 도구 사용

▣ SQLDEV

개발	SQL 문장 작성하듯이 PL/SQL block 작성
테스트	VARIABLE , PRINT
디버깅	DBMS_OUTPUT

5. 개발 Tool

```
REM   DEFAULT SIZE  1M bytes

SET  SERVEROUTPUT ON                          // SET  SERVEROUTPUT OFF

BEGIN
            FOR I IN 1..10
            LOOP
                        DBMS_OUTPUT.PUT_LINE(TO_CHAR(I)||' processed');
            END LOOP;
END;
/
```

▣ DBMS_OUTPUT

① ORACLE社에서 **디버깅 도구**로 제공하는 Oracle definded package
   ex) desc dbms_output

② PL/SQL block이 실행되는 동안에 메모리상에 출력 결과를 저장해 두었다가 **block 실행 종료후**
   메모리상의 결과를 SQL*PLUS or SQLDEV의 화면에 출력
   <참고> DBMS_OUTPUT에 대한 오해는 실행 시점에 실시간으로 결과를 출력한다고 생각하는 경우가 많지만
            Block 실행 종료후 해당 내용을 화면에 일괄적으로 출력

③ 환경변수 인 SERVEROUTPUT을 ON 설정해야 출력 결과를 저장한후 화면에 출력

④ Default size : 1M , maxsize : Unlimited (10g R2 이후)
      ex) show serveroutput or show all

5. 개발 Tool

```
REM   DEFAULT SIZE  1M bytes

SET  SERVEROUTPUT ON  SIZE 2000
BEGIN
        DBMS_OUTPUT.PUT_LINE(rpad('X',100,'X'));          // line 길이 32767 bytes
END;
/

BEGIN
  FOR I IN 1..20                                          // FOR I IN 1..100
  LOOP
        DBMS_OUTPUT.PUT_LINE(rpad('X',100,'X'));
  END LOOP;
END;
/
```

```
오류 보고 -
ORA-20000: ORU-10027: buffer overflow, limit of 2000 bytes
```

```
>SHOW SERVEROUTPUT
  SET SERVEROUTPUT ON
```