1. 서브프로그램이란

프로그램 내의 다른 루틴(routine)들을 위해서 특정한 기능을 수행하는 부분적 프로그램을 말한다. 서브프로그램(subprogram)은 한 프로그램에서 동일한 처리가 반복되거나, 여러 프로그램에서 공통으로 처리되는 내용으로 작성하고, 필요한 위치나 블록에서 서브프로그램을 호출한다.

서브 프로그램은 메인 블록에서 반복되거나, 여러 블록에서 공통으로 처리되는 명령문을 추출하여 별개의 서브프로그램으로 작성하여 사용하면, 프로그램 해독이 쉽고, 블록에서 호출이 가능하기 때문에 효율성, 재사용성, 유지 보수성, 호환성 측면에서 장점이 많다.

예제) 익명의 블록의 처리 작업으로 데이타 저장하기.

2. 서브프로그램 작성

PL/SQL은 프로시저(Procedure)와 함수(Function)라는 두 종류의 서브프로그램을 지원한다. 익명의 블록에 작성되는 서브프로그램은 선언절에 기술되고, 실행절에서 호출한다. 서브프로그램은 실행할 때마다 컴파일(complie)되기 때문에 실행 속도가 느리고, 다른 블록에서 호출할 수 없는 단점을 가지고 있다.

문법	DECLARE
	.
	.
	[서브프로그램 선언]
	BEGIN
	.
	.
	[서브프로그램 호출]
	.
	.
	END;
	/

2.1. 프로시저(Procedure) 생성 구문

문법	Procedure 프로시저명 ([형식인자1, …]) IS [지역변수선언; …] BEGIN 처리 명령문1; …; 처리 명령문N; [Exception] [예외처리문; …] END [프로시저명];

	/

※ 기술 방법

- 프로시저명 : 생성하려는 프로시저명, 객체 이름 짓기 제한에 따른다.
- 형식인자 : 값을 주고받는 인자로, 다음과 같은 형식으로 선언한다.

형식인자명 [IN | OUT | IN OUT] 데이터타입 [{ := | DEFAULT } 값]

☞ IN : 호출되는 서브프로그램에 값이 전달되는 것을 지정하고, 생략 가능. 기본 값을 지정할 경우 리터럴, 변수, 식 등이 올 수 있다.

☞ OUT : 프로시저가 호출 프로그램에게 값을 반환하는 것을 지정. 기본 값을 지정할 경우 변수만 올 수 있다.

☞ IN OUT : 호출되는 서브프로그램에 값을 전달하고, 실행 후 호출 프로그램에게 값을 반환하는 것을 지정한다.

☞ 데이터타입 : 형식인자에 대한 데이터타입을 기술한다.

☞ := 또는 Default : 형식인자에 대한 기본 값을 지정하는 예약어

- 지역변수 선언 : 프로시저 서브프로그램에서 필요한 변수나 상수, 다른 프로시저와 함수 등을 기술한다.
- 처리명령문 : 프로시저 서브프로그램에서 실행할 명령문을 기술한다.
- 예외처리문 : 예외처리에 관한 명령문을 기술한다.

2.2. 프로시저(Procedure) 호출 방법

선언절에 기술된 프로시저 서브프로그램을 실행절에서 호출한다.

호출방법	프로시저명 (실인자1, 실인자2, ····);

※ 기술 방법

- 프로시저명 : 호출하려는 프로시저명을 기술한다.
- 실인자 : 호출하는 프로시저 서브프로그램에게 전달할 값, 실인자는 리터럴, 칼럼명, 변수명 등으로 기술한다.

☞ 실인자와 형식인자와의 관계

- 실인자와 형식인자의 수가 같아야 한다.
- 실인자와 형식인자의 데이터타입이 같아야 한다.

•주고받는 값은 실인자와 형식인자의 기술된 순서로 결정된다.

예제) Patient 테이블에 프로시저 서브프로그램으로 데이타를 insert해 보자.

2.3. 함수(Function) 생성 구문

문법	FUNCTION 함수명 (형식인자1, …·) RETURN 데이터타입 IS [지역변수선언 ;] BEGIN 처리 명령문1; ·····; 처리 명령문N; RETURN 변수명; [EXCEPTION] [예외처리문] END [함수명]; /

※ 기술 방법

• 함수명 : 생성하려는 함수명을 기술한다.

• 형식인자 : 값을 주고받는 인자로, 다음과 같은 형식으로 선언한다.

형식인자명 [IN | OUT | IN OUT] 데이터타입 [{ := | DEFAULT } 값]

☞ IN : 호출되는 서브프로그램에 값이 전달되는 것을 지정하고, 생략 가능, 기본 값을 지정할 경우 리터럴, 변수, 식 등이 올 수 있다.

☞ OUT : 프로시저가 호출 프로그램에게 값을 반환하는 것을 지정, 기본 값을 지정할 경우 변수만 올 수 있다.

☞ IN OUT : 호출되는 서브프로그램에 값을 전달하고, 실행 후 호출 프로그램에게 값을 반환하는 것을 지정한다.

• RETURN 데이터타입 : 호출 프로그램에게 반환되는 값의 데이터타입을 기술한다.

• 지역변수 선언 : 함수 서브프로그램에서 필요한 변수나 상수, 다른 프로시저와 함수 등을 기술한다.

• 처리명령문 : 함수 서브프로그램에서 실행할 명령문을 기술한다.

• 예외처리문 : 예외처리에 관한 명령문을 기술한다.

2.4. 함수(Function) 호출 방법

호출방법	변수명 := 함수명 (실인자1, 실인자2, ·····);

※ 기술 방법

- 함수명 : 호출하려는 함수명을 기술한다.
- 실인자 : 호출하려는 함수 서브프로그램에게 전달할 값을 기술한다.
- 변수명 : 실행한 함수의 결과 값을 저장할 변수명을 기술한다.

예제) 화씨온도(℉)를 섭씨온도(℃)로 변환하는 함수 서브프로그램을 작성하고, 화씨온도 100.2℉를 변환해보자.

2.5. 프로시저와 함수 서브프로그램의 차이점

- 프로시저는 형식인자로 데이터를 전달받을 수도 있고, 받지 않을 수도 있다. 실행 후 프로시저는 호출한 프로그램에게 값을 반환할 수도 있고, 안 할 수도 있다.
- 함수는 형식인자로 데이터를 전달받을 수도 있고, 받지 않을 수도 있다. 그러나 실행 후 반드시 하나의 값을 반환한다.
- 기술 방법과 호출 방법이 다르다.

3. 저장된 서브프로그램 작성

블록에 선언하지 않고, 별도로 저장된 프로시저(Stored Procedure) 서브 프로그램이나 저장된 함수(Function) 서브프로그램을 작성하여 오라클 데이터베이스에 내장되는 PL/SQL 블록을 작성하면, 다음과 같은 장점이 있다.

- 블록의 단순화

  서브프로그램을 별도로 작성하여 프로그램이 단순해지고, 또한 프로그램 해독의 용이하게 된다.

- 효율성

  클라이언트/서버 컴퓨팅 환경에서, 클라이언트 애플리케이션은 데이터베이스 버서에게 SQL의 요청을 건네주는데, 사용자 수가 증가하면 SQL의 요청 건수도 증가하게 된다. 이렇게 되면 서버에 부하가 많이 걸리게 되고, 네트워크의 트래픽도 증가한다. 저장된 프로시저나 함수 서브프로그램을 사용하게 되면 호출한 여러 개의 SQL문이 데이터베이스에 저장된 하나의 호출문으로 실행하게 되어 서버의 부하를 줄이고, 네트워크 트래픽도 감소하게 된다.

- 재사용성

  저장된 서브프로그램을 생성하면, SQL*PLUS 스크립트, 데이터베이스 트리거, 클라이언트 애플리케이션 로직 등에서 호출하여 사용할 수 있다.

- 유지보수성

  저장된 프로시저나 함수 서브프로그램은 SQL*PLUS 스크립트, 데이터베이스 트리거, 클라이언트 애플리케이션 로직 등에서 동일한 저장된 프로시저나 함수 서브프로그램을 호출할 수 있기 때문에 프로그램 관리가 용이하고, 유지 보수비도 줄일 수 있다.

3.1. 저장된 프로시저 생성

오라클 데이터베이스에 저장되는 PL/SQL 서브프로그램을 말한다. 블록에 선언되어 사용되는 프로시저 서브프로그램과 유사하나, 별도로 생성되고, "CREATE [OR REPLACE] PROCEDURE문"으로 작성한다.

문법	CREATE [OR REPLACE] PROCEDURE 프로시저명 (형식인자1, ……) IS [지역변수선언;] BEGIN 처리 명령문1; ……; 처리 명령문N; [EXCEPTION] [예외처리문;] END [프로시저명]; /

☞ 저장된 프로시저 서브프로그램 호출

EXECUTE 명령어로 다음과 같이 호출할 수 있다. 실인자란 서브프로그램에 데이터를 전달할 상수, 변수, 수식을 말한다.

호출방법	EXECUTE 프로시저명 (실인자1, 실인자2, ……)

☞ 저장된 프로시저 서브프로그램 삭제

호출방법	DROP PROCEDURE 프로시저명;

예) 섭씨/ 화씨 온도를 계산하는 프로시저를 생성해보고, 환자번호가 'YN0005', 환자 체온이 43 ℉일 때의 값을 만들어진 프로시저를 이용하여 Patient 테이블에 섭씨온도로 변환하여 실행한 후 확인해보자. 그리고 삭제해보자.

3.2. 저장된 함수 서브프로그램 생성

저장된 프로시저 서브프로그램과 같이 오라클 데이터베이스에 함수로 저장되는 PL/SQL 서브프로그램이다.

문법	CREATE [OR REPLACE] Function 함수명 (형식인자1, ……) RETURN 데이터타입 IS [지역변수선언;] BEGIN 처리 명령문1; ……; 처리 명령문N; RETURN 변수명; [EXCEPTION] [예외처리문;] END [함수명]; /

☞ **저장된 함수 호출 방법**

호출방법	함수명 (실인자1, 실인자2, ……);

☞ **저장된 함수 삭제**

호출방법	DROP FUNCTION 함수명;

예) 화씨온도(℉)를 섭씨온도(℃)로 변환하는 함수를 저장된 함수(Stored Function)서브프로그램을 작성하고, 만들어진 함수를 호출하여 Patient 테이블의 환자 체온을 섭씨로 변환하여 출력해보자. 그리고 만들어진 함수를 삭제해보자.

4. 패키지(Package)

오라클 데이터베이스에 저장된 애플리케이션 관련 PL/SQL 프로시저와 함수의 집합체이다. PL/SQL 패키지는 패키지 명세(Package Specification)와 패키지 본문(Package Body)으로 구성되어 있다. 또한 패키지 본문에는 타입, 변수 커서 등이 포함될 수 있다.

☞ **패키지를 사용하는 이유?**

- 응용 프로그램을 보다 효율적인 모듈 단위로 구성할 수 있다. 애플리케이션 단위로 패키지를 구성하면 이해하기도 쉽고, 패키지 간의 인터페이스도 간단명료하게 된다.
- 권한을 효율적으로 허가할 수 있다.
- 패키지의 public 변수와 커서는 세션이 열려 있는 동안 지속되어 이 환경에서 실행되는 모든 커서와 프로시저의 공유가 가능하다.
- 프로시저와 함수의 오버로드(overload)를 줄일 수 있다.

- 여러 개의 객체를 한 번에 메모리에 로드하기 때문에 성능이 향상된다.
- 저장된 프로시저와 함수를 갖고 있는 라이브러리를 사용하여 코드 재사용을 향상시키고, 불필요한 코딩량을 줄일 수 있다.

4.1. 패키지 명세

패키지명과 인수의 이름, 데이터타입을 public으로 선언하는 부분이다.

문법	CREATE [OR REPLACE] PACKAGE 패키지명 IS [프로시저 선언절;] [함수 선언절;] · · END [패키지명]; /

※ 기술 방법

- 패키지명 : 작성할 패키지명을 기술한다.
- 프로시저 선언절 : 패키지 본문에 나타나는 프로시저 선언절을 기술한다.
- 함수 선언절 : 패키지 본문에 나타나는 함수 선언절을 기술한다.

예) 이전에 만든 Body_Temp_Change_F 프로시저와 Temp_Change_C 함수를 포함하는 패키지 명세를 생성해보자.

4.2. 패키지 본문

패키지 명세에서 선언한 Public 객체의 본문과 타입, 상수, 변수, 커서 등을 정의한다.

문법	CREATE [OR REPLACE] PACKAGE BODY 패키지명 IS [타입 선언;] [상수, 변수 선언;] [커서 선언;] [프로시저 본문;] [함수 본문;] · · END [패키지명]; /

※ 기술 방법

- 패키지명 : 작성할 패키지 본문의 패키지명을 기술한다.
- 타입 선언 : 패키지에 포함할 데이터타입을 기술한다.
- 상수, 변수 선언 : 패키지에 포함할 상수, 변수를 기술한다.
- 커서 선언 : 패키지에 포함할 커서를 기술한다.
- 프로시저 본문 : 프로시저 서브프로그램을 기술한다.
- 함수 본문 : 함수 서브프로그램을 기술한다.

4.3. 패키지 호출

호출방법	패키지명.타입 패키지명.변수 패키지명.상수 패키지명.프로시저명 패키지명.함수명

4.4. 패키지 서브프로그램 삭제

문법	DROP PACKAGE 패키지명;

예) 상수, 변수, 레코드 타입과 저장된 프로시저와 함수를 이용하여 패키지 본문을 만들고 호출해 보자. 그리고 삭제해보자.

※※ 패키지를 생성할 경우에는 반드시 패키지 명세를 먼저 생성해야 한다. 패키지 명세가 없는 패키지 본문은 생성할 수 없다.

[간단한 연습문제]

1. 회원관리(EC_Member) 테이블의 주민등록번호(Regist_No)를 전달받아 성별을 추출하여 '남자' 혹은 '여자'로 출력하는 저장된 함수(Stored Function) "Member_Gender"를 작성하고, 이 저장된 함수를 이용하여 각 회원의 회원ID(USERID), 회원명(NAME), 주민등록번호(REGIST_NO), 성별을 출력하시오.

2. 회원관리(EC_Member) 테이블의 회원의 가입일자(Timestamp)를 입력받아, 가입 연수와 개월 수

를 구하는 저장된 함수 "Member_YYMM"을 작성하고, 이 저장된 함수를 이용하여 회원 ID(USERID), 회원명(NAME), 가입일자(Timestamp), 가입기간을 출력하시오.

3. 주문처리(EC_Order) 테이블의 회원의 주문자ID(ORDER_ID)를 입력받아, 주문 금액의 합계를 구하는 저장된 함수 'user_order_sum()'을 작성하고, 'supark' 주문자에 대한 주문 금액의 합계를 출력하시오. < 출력 데이타 : ORDER_ID, USER_ORDER_SUM(ORDER_ID) >