



프로시저 vs 트리거 vs 함수

SI2팀 엄예지

목 차

Chapter 01 | 프로시저

Chapter 02 | 트리거

Chapter 03 | 함수

Chapter 04 | 토론

프로시저

01

CHAPTER

프로시저란?

- 오라클에서 프로시저는 PL/SQL을 통해 만들어진다.
- 자주 사용하는 SQL을 프로시저로 만든 뒤 필요할때마다 호출하여 사용
- 특정한 로직을 처리하기만 하고 결과 값은 반환하지 않는 서브 프로그램 (이 아니다.)



구성

- DECLARE : 프로시저의 명칭, 변수, 인수, 데이터 타입을 정의하는 선언부
- BEGIN / END : 프로시저의 시작과 종료를 의미
- CONTROL : 조건문 또는 반복문이 삽입되어 순차적으로 처리
- SQL : DML, DCL이 삽입되어 데이터 관리를 위한 작업
- EXCEPTION : BEGIN~END 안의 구문 실행시 예외 처리
- TRANSACTION : 수행된 데이터 작업들을 DB에 저장할지 취소할지 결정



구성

```
CREATE OR REPLACE PROCEDURE TEST_PR
```

```
//지역변수 선언
```

```
( P_GMNCOD IN VARCHAR2,  
  P_GNNAME IN VARCHAR2,  
  P_BIRDAY IN VARCHAR2,  
  P_TELNUM IN VARCHAR2  
)
```

```
IS
```

```
BEGIN
```

```
INSERT INTO GMNMST (GMNCOD, GNNAME, BIRDAY, TELNUM)  
VALUES(P_GMNCOD, P_GNNAME, P_BIRDAY, P_TELNUM);
```

```
END;
```



구성

파라미터

- IN : 호출 프로그램이 프로시저에게 값을 전달할때 사용
- OUT : 프로시저가 호출 프로그램에게 값을 전달할때 사용
- INOUT : 호출 프로그램이 프로시저에게 값을 전달하고, 프로시저 실행후 호출 프로그램에게 값을 반환할때 사용
- 자료형 : 변수의 자료형을 지정

프로시저 실행

- EXECUTE TEST_PR;
- EXEC TEST_PR;
- CALL TEST_PR;



트리거

02

CHAPTER

트리거란?

- INSERT, UPDATE, DELETE 등의 이벤트가 발생할 때마다 관련 작업이 자동으로 수행되는 절차형 SQL
- 트리거는 테이블에 저장되는 것이 아니라 별도로 오라클 데이터베이스 자체에 저장
- 뷰에 대해서는 동작하지 않고 테이블 자체에 대해서만 정의 가능
- 데이터 변경 및 무결성 유지 로그 메시지 출력 등



구성

- DECLARE : 트리거의 명칭, 변수 및 상수, 데이터 타입을 정의
- EVENT : 트리거가 실행되는 조건
- BEGIN / END : 트리거의 시작과 끝
- CONTROL : 조건문, 반복문이 삽입되어 순차적으로 처리
- SQL : DML문이 삽입되어 데이터 관리를 위한 작업
- EXCEPTION : 예외 처리



구성

```
CREATE OR REPLACE TRIGGER TEST1234
동작시기옵션/동작옵션 ON GMNMST
BEGIN
DBMS_OUTPUT.PUT_LINE('관리인 등록 완료! 축하 ');
END;
```

동작시기옵션 : 트리거가 실행될 때를 지정

- AFTER : 테이블이 변경된 후
- BEFORE : 테이블이 변경되기 전

동작옵션 : 트리거가 실행되게 할 작업의 종류를 지정

- INSERT, DELETE , UPDATE

NEW, OLD : 트리거가 적용될 테이블의 별칭을 지정

- NEW : 추가되거나 수정에 참여할 테이블을 의미
- OLD : 수정되거나 삭제 전 대상이 되는 테이블을 의미

FOR EACH ROW : 각 행마다 트리거를 적용한다는 의미



함수

03

CHAPTER

함수란?

- 호출한 곳으로 반드시 하나의 값을 리턴해줘야 되는 PL/SQL Stored Program
- 프로시저는 PL/SQL문으로 실행하지만, 함수는 식의 일부로서 사용
- 함수 작성시 권한 필요
- 자신의 스키마 CREATE PROCEDURE
- 다른 사용자의 스키마
- CREATE ANY PROCEDURE



구성

```
CREATE OR REPLACE FUNCTION TEST_FN  
RETURN datatype  
IS  
BEGIN  
RETURN 변수;  
END;
```

DATATYPE

- 해당 변수의 데이터 타입을 지정
- 테이블명.컬럼명%type을 입력하면 해당 테이블의 해당 컬럼의 type 참조

RETURN

- 리턴할 값의 데이터타입 지정



비교

04

CHAPTER

프로시저

- CREATE PROCEDURE 문법 사용
 - 소스코드와 실행코드 생성
 - EXCUTE 명령어로 실행
- COMMIT, ROLLBACK 실행 가능

트리거

- CREATE TRIGGER 문법 사용
 - 소스코드와 실행코드 생성
 - 생성 후 자동실행
- COMMIT, ROLLBACK 실행 안됨

프로시저

- 특정 작업을 수행
- 리턴값을 가질수도 안가질수도 있음
- 리턴값을 여러 개 가질수 있음
- 서버(DB)단에서 기술
- 수식내에서 사용 불가
- 단독으로 문장 구성 가능

함수

- 특정 계산을 수행
- 리턴값을 반드시 가져야함
- 리턴값을 오직 하나만 가질수 있음
- 화면(Client)단에서 기술
- 수식내에서만 사용 가능
- 단독으로 문장 구성 불가



토리

Thank You!
감사합니다