07.

트리거

- ✓ 트리거 Before, After
- ✓ 트리거 Instead Of
- ✓ 연쇄트리거

트리거의 개념 및 형식

- 테이블 or 뷰에 변경연산이 실행되면 연동되어 실행되는 프로그램
  - 변경연산(Insert, Delete, Update) 시 자동실행. 논리적 무결성을 지원함.
    - 항상 실행대기(Background Process).
  - 트리거 설정 단위
    - 테이블이나 뷰 단위 : ON 테이블 or 뷰
    - 테이블이나 뷰의 컬럼 단위 : [OF 컬럼]

[트리거의 형식]

```
CREATE [or replace] TRIGGER 트리거이름
  {Before | After | Instead Of } { Insert | Delete | Update } [ OF 컬럼 ]
  ON 테이블or뷰
  [FOR EACH ROW]
BEGIN
   PL/SQL Statements
    ... ...
    -- 상태확인 변수 : Updating, Deleting, Inserting
END;
```

트리거 vs. (함수, 저장프로시저)

	트리거	함수, 저장 프로시저
실행방식	자동 실행(Background Process)	명시적 호출
실행시기	변경연산(삽입, 삭제, 갱신) 시	호출 명령을 실행했을 때
파라미터	없음	있음 - 함수 : 입력 - 프로시저 : IN, OUT, IN OUT
반환값	없음	- 함수 : 반환값 있음. - 프로시저 : out, in out 파라미터
부착	테이블 or 뷰	스키마. 단순 연산도 가능

After / Before / Instead Of

- BEFORE { insert or update or delete } on table
- AFTER { insert or update or delete } on table
- INSTEAD OF { insert or update or delete } on view
  - 이벤트가 발생하기 전에 작동
  - 시도된 변경 SQL은 무시됨.
    - 그 대신(instead of) 해당 뷰에 지정한 SQL이 작동함

Simple Trigger 예제

ORA.7 #1 (1).sql

```
CREATE or REPLACE TRIGGER T_Simple1
 AFTER INSERT
 ON 고객
 FOR EACH ROW
BEGIN
 DBMS_OUTPUT.PUT_LINE('After Insert 감지');
END;

CREATE or REPLACE TRIGGER T_Simple2
 AFTER DELETE
 ON 고객
 FOR EACH ROW
BEGIN
 DBMS_OUTPUT.PUT_LINE('After Delete 감지');
END;

SET SERVEROUTPUT ON;
INSERT INTO 고객 VALUES ('berry', 'JDK', 50, 'gold', '교수', 5000);
DELETE FROM 고객 WHERE 고객아이디 = 'berry';
```

트리거의 삭제

- 트리거는 Background Process임.
  - 생성하면 그 때부터 시스템 부하가 증가함
  - 불필요한 트리거는 삭제하는 것이 좋음

- DROP TRIGGER 트리거 이름

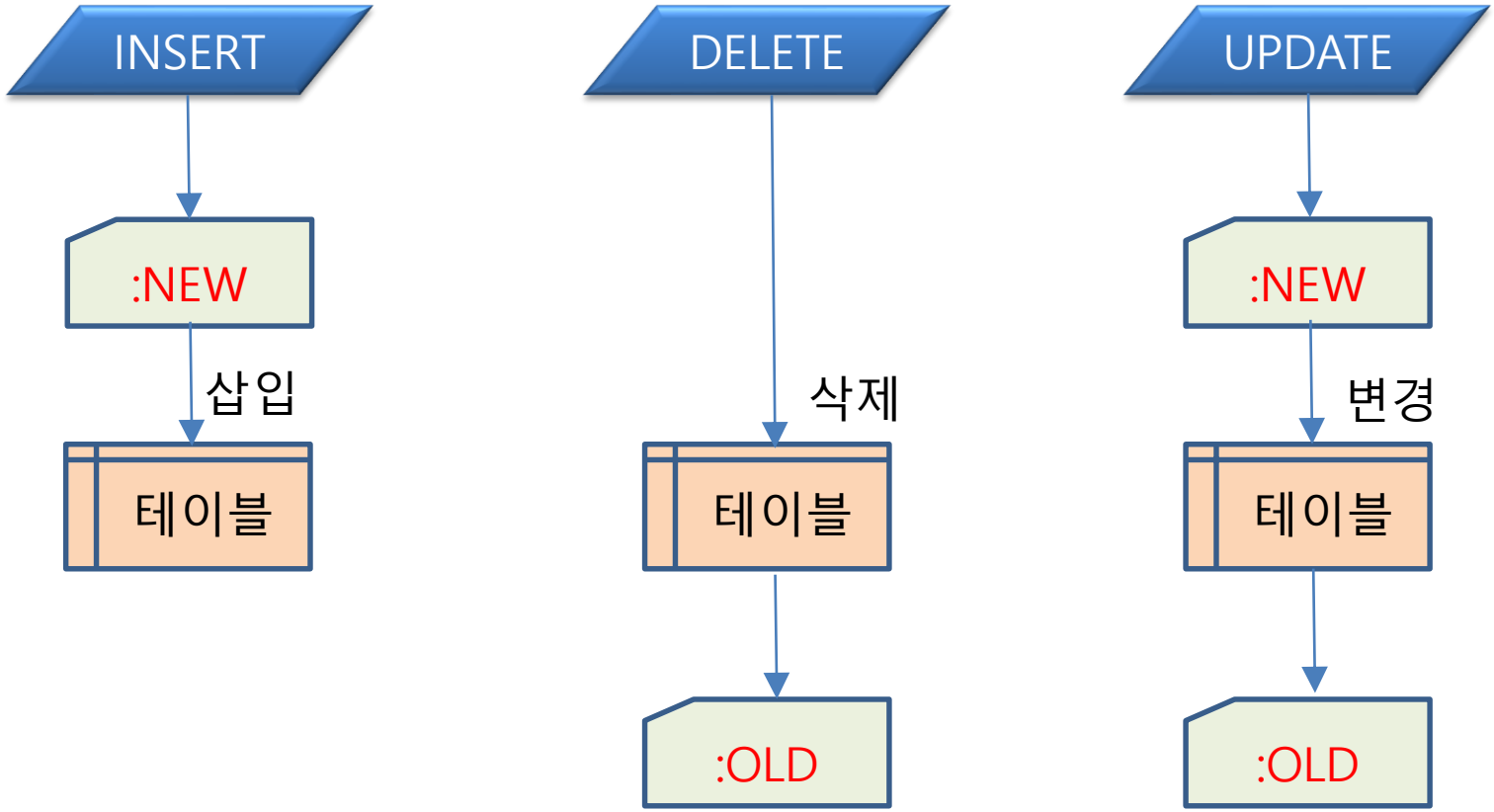
```
DROP TRIGGER T_Simple1;

DROP TRIGGER T_Simple2;
```

:OLD, :NEW

- 트리거에서 사용가능한 임시테이블
  - :OLD -- 변경(삽,삭,갱) 이전의 값을 가진 테이블
  - :NEW -- 변경 이후의 값을 가진 테이블
- Insert : 해당되는 이전 튜플이 없음. :OLD(x), :NEW(o)
- Delete : 해당되는 이후 튜플이 없음. :OLD(o), :NEW(x)
- Update : :OLD(o), :NEW(o) 둘다 유효
  - 예) 고객 테이블에 Update연산이 있고, 트리거가 부착되어 있을 경우
    - Update되기 전의 고객 이름을 접근하고자 한다면
      - 트리거 내에서는『 :OLD.고객이름 』을 사용
    - Update된 후의 고객이름을 접근하고자 한다면
      - 트리거 내에서는『 :NEW.고객이름 』을 사용

:OLD, :NEW 개념도



Before, After>

- Before, After는 트리거 실행 시간만을 결정
- NEW, OLD와는 무관

예제) :OLD, :NEW ..... Insert시 New만 유효

```
CREATE OR REPLACE TRIGGER T_OLDNEW1
   BEFORE INSERT
   ON 고객
   FOR EACH ROW
BEGIN
        DBMS_OUTPUT.PUT_LINE('NEW 이름 ' || :NEW.고객이름);
        DBMS_OUTPUT.PUT_LINE('NEW 직업 ' || :NEW.직업);
        DBMS_OUTPUT.PUT_LINE('OLD 이름 ' ||:OLD.고객이름);
        DBMS_OUTPUT.PUT_LINE('OLD 직업 ' || :OLD.직업);
END;

INSERT INTO 고객 VALUES ('aaa', '도날드', NULL, 'silver', '학생', 300);

INSERT INTO 고객 VALUES ('aaa', '도날드', NULL, 'silver', '학생', 300);
```

```
NEW 이름 도날드
NEW 직업 학생
OLD 이름
OLD 직업


1 행 이(가) 삽입되었습니다.
```

```
NEW 이름 도날드
NEW 직업 학생
OLD 이름
OLD 직업


명령의 13 행에서 시작하는 중 오류 발생 -
INSERT INTO 고객 VALUES ('aaa', '도날드', NULL, 'silver', '학생', 300)
오류 보고 -
ORA-00001: unique constraint (HMART.SYS_C007297) violated
```

BEFORE vs. AFTER

❑ INSERT 시 검증하여 데이터를 변경하고자 할 때

- ▶ BEFORE : 트리거가 먼저 실행되어 튜플을 미리 검증한 뒤 입력
  - ▫ 검증하여 문제있으면 트리거에서 :NEW 값을 변경
    - – 변경된 :New가 자동적으로 Insert 됨

- ▶ AFTER : 튜플이 입력된 뒤 트리거 실행.
  - ▫ 검증하여 문제있으면 입력 취소 시킴
    - – Insert + 취소
  - ▫ 검증하여 문제있으면 값을 Update할 수 있을까?
    - – 불가능
    - – 트리거가 부착된 테이블의 이름을 트리거 BODY에서 사용할 수 없음.
      - ✓ :NEW or :OLD만 사용 가능

Before : Insert 전 데이터 변경

❑ 값을 변경해서 입력해야 하는 경우

- ▶ 지금부터 입력될 때 고객 이름을 김OO 형태로 입력하고자 함.
- ▶ Real DB에 입력되기 전에 :NEW값을 바꿈

```
CREATE OR REPLACE TRIGGER T_OLDNEW2
   before INSERT
   ON 고객
   FOR EACH ROW
BEGIN
    :NEW.고객이름 := SUBSTR(:NEW.고객이름, 1, 1) || 'OO';
END;

INSERT INTO 고객 VALUES ('aaa', '도날드', NULL, 'silver', '학생', 300);
```

1	apple	정소화	20	gold	학생	1000
2	banana	김선우	25	vip	간호사	2500
3	carrot	고명석	28	gold	교사	4500
4	orange	김용욱	22	silver	학생	0
5	melon	성원용	35	gold	회사원	5000
6	pear	채광주	31	silver	회사원	500
7	aaa	도OO	(null)	silver	학생	300

After : Insert 후 검증 & 입력 취소

❑ '신용불량자'이면 고객가입 취소

다른 예 :
주문 튜플을 입력할 때
물품 수량을 검증하여
모자라면 '품절'주문취소

```
CREATE OR REPLACE TRIGGER T_자격검사
   AFTER INSERT
   ON 고객
   FOR EACH ROW
BEGIN
    IF :NEW.직업 = '신용불량자' THEN
        DBMS_OUTPUT.PUT_LINE('자격이 맞지 않아요.');
        DBMS_OUTPUT.PUT_LINE('입력이 취소되었습니다.');
        RAISE_APPLICATION_ERROR(-20999,'자격검사 위배 입력 시도 발견 !!!'); -- 입력 취소
    END IF;
END;

INSERT INTO 고객 VALUES ('fruits', '신불자', NULL, 'silver', '신용불량자', 300);
```

스크립트 출력 × | 질의 결과 ×

작업이 완료되었습니다.(0.05초)

```
Trigger T_자격검사이(가) 컴파일되었습니다.

자격이 맞지 않아요.
입력이 취소되었습니다.


명령의 13 행에서 시작하는 중 오류 발생 -
INSERT INTO 고객 VALUES ('fruits', '신불자', NULL, 'silver', '신용불량자', 300)
오류 보고 -
ORA-20999: 자격검사 위배 입력 시도 발견 !!!
ORA-06512: at "HMART.T_자격검사", line 5
ORA-04088: error during execution of trigger 'HMART.T_자격검사'
```

After : Error Case

- 데이터 삽입 후 취소를 하지 않고, 값을 Update하고자 하면...
  - 트리거의 컴파일 오류는 없지만, 삽입 시 오류
    - 트리거가 부착된 테이블의 이름을 트리거 BODY에서 사용할 수 없음.
      - 트리거 BODY에서는 :NEW or :OLD만 사용가능

```
CREATE OR REPLACE TRIGGER T_OLDNW3
   AFTER INSERT
   ON 고객
   FOR EACH ROW
BEGIN
    IF  :NEW.고객이름 = '도날드' THEN
      UPDATE  고객 SET 고객이름 = '도00' WHERE 고객이름 = '도날드';
    END IF;
END;

INSERT INTO 고객 VALUES ('aaa', '도날드', NULL, 'silver', '학생', 300);
```

스크립트 출력 | 질의 결과

작업이 완료되었습니다.(0.052초)

```
명령의 11 행에서 시작하는 중 오류 발생 -
INSERT INTO 고객 VALUES ('aaa', '도날드', NULL, 'silver', '학생', 300)
오류 보고 -
ORA-04091: table HMART.고객 is mutating, trigger/function may not see it
ORA-06512: at "HMART.T_OLDNEW2", line 3
ORA-04088: error during execution of trigger 'HMART.T_OLDNEW2'
```

After : 검증 후 다른 테이블 Update

- 데이터 삽입 후 취소를 하지 않고, 다른 테이블 Update
  - 가능. 트리거 내에서 다른 테이블에 대한 SQL은 적용 가능

```
CREATE OR REPLACE TRIGGER T_OLDNW4
   AFTER INSERT
   ON 고객
   FOR EACH ROW
BEGIN
    IF  :NEW.고객이름 = '도날드' THEN
      UPDATE  주문 SET 수량 = 99 WHERE 주문번호 = 'o08';
    END IF;
END;

INSERT INTO 고객 VALUES ('aaa', '도날드', NULL, 'silver', '학생', 300);
```

트리거 예제 : 탈퇴 고객의 정보는 어떡할까?

- 한 테이블에 적용된 트리거로 다른 테이블을 변경하는 경우
- '고객'테이블에서는 삭제한 후,'탈퇴고객'이라는 테이블에 삽입
  - 두 연산은 서로 연계되어 있음.
  - 그런 경우 고객 테이블에 AFTER DELETE 트리거를 생성
    - 그 트리거 Body는 삭제된 고객 정보를 '탈퇴고객'테이블에 기록하는 명령

- 트리거 테스트 시나리오
  - 먼저 탈퇴고객 테이블을 정의하고

⇒ 트리거 정의

⇒ 고객 삭제 & 탈퇴 고객 테이블 확인

ORA.7 #1 (1).sql

CASE : 탈퇴고객 백업

```
CREATE TABLE 탈퇴고객
(    "고객아이디" VARCHAR2(20 BYTE) ,
     "고객이름" VARCHAR2(10 BYTE),
     "나이" NUMBER(*,0),
     "등급" VARCHAR2(10 BYTE) ,
     "직업" VARCHAR2(20 BYTE),
     "적립금" NUMBER(*,0) DEFAULT 0,
     변경일 DATE,
     변경담당자 NCHAR(30)
);

CREATE OR REPLACE TRIGGER T_탈퇴고객관리
    AFTER DELETE
    ON 고객
    FOR EACH ROW
DECLARE
     -- 변수 선언부
BEGIN
    -- :OLD ? 변경전의 테이블
     INSERT INTO 탈퇴고객 VALUES( :OLD.고객아이디, :OLD.고객이름, :OLD.나이,
         :OLD.등급, :OLD.직업, :OLD.적립금, SYSDATE(), USER( ) );
END T_탈퇴고객관리;
```

	고객아이디	고객이름	나이	등급	직업	적립금
1	apple	정소화	20	gold	학생	1000
2	banana	김선우	25	vip	간호사	2500
3	carrot	고명석	28	gold	교사	4500
4	orange	김용욱	22	silver	학생	0
5	melon	성원용	35	gold	회사원	5000
6	peach	오형준	(null)	silver	의사	300
7	pear	채광주	31	silver	회사원	500

DELETE FROM 고객 WHERE 고객아이디 = 'peach'; -- 트리거 실행. 탈퇴고객 테이블에 삽입
DELETE FROM 고객 WHERE 고객아이디 = 'apple'; -- 오류 : 참조무결성 위배

	고객아이디	고객이름	나이	등급	직업	적립금	변경일	변경담당자
1	peach	오형준	(null)	silver	의사	300	20/08/17	HMART

CASE : PK-FK Update CASCADE 구현

❑ 오라클에서는 외래키 제약 조건

ORA.7 #1 (1).sql

Foreign Key 주문고객 References 고객(고객아이디)
on delete {no action, cascade, set null} -- 지원
on update -- 미지원. 보통 Trigger로 구현.

▶ Parent테이블의 'PK'를 변경하면, Child 테이블의 'FK'도 변경

▫ 고객테이블의 'apple'을 'samsung'으로 변경하면

– apple을 참조하는 주문테이블의 속성도 모두 samsung으로 변경

변경연산은 부착된 트리거까지 실행된 후
FK 등의 Constraint 위배 여부 Check함

```
create trigger T_OnUpdateCascade
  AFTER UPDATE OF 고객아이디 ON 고객
  FOR EACH ROW
BEGIN
  UPDATE 주문 SET 주문고객 = :NEW.고객아이디 WHERE 주문고객 = :OLD.고객아이디;
END;

UPDATE 고객 SET 고객아이디 = 'samsung' WHERE 고객아이디 = 'apple';
```

INSTEAD OF

ORA.7 #1 (2).sql

- View에 적용됨
- 필요성 - 조인 뷰에 삽입 연산이 수행되지 않음

```
CREATE or REPLACE VIEW 학생학과정보
AS
        SELECT 학번, 이름, 학생.전화번호, 주소, 학과.학과번호, 학과명, 사무실
        FROM 학생, 학과
        WHERE 학생.학과번호 = 학과.학과번호;

INSERT INTO 학생학과정보
VALUES ('20301-006', '박문수', '200-2000', '부산', 303, '컴공', '917호');
```

- 이 문제를 Instead Of 트리거로 해결

INSTEAD OF(2)

❑ Insert를 대신할 트리거 생성

```
CREATE OR REPLACE TRIGGER 뷰삽입
   INSTEAD OF INSERT  -- 삽입작업 대신에 작동 작동하도록 지정
   ON 학생학과정보  -- 뷰에 장착
   FOR EACH ROW
BEGIN
   INSERT INTO 학과(학과번호, 학과명, 사무실)
     VALUES (:NEW.학과번호, :NEW.학과명, :NEW.사무실);
   INSERT INTO 학생(학번, 이름, 학과번호, 주소, 전화번호)
     VALUES (:NEW.학번, :NEW.이름, :NEW.학과번호, :NEW.주소, :NEW.전화번호);
END;
```

- 트리거 내에서 개별 테이블로 삽입하는 연산으로 대체

- 아래 삽입연산 대신 트리거가 실행됨

```
INSERT INTO 학생학과정보
VALUES ('20301-006', '박문수', '200-2000', '부산', 303, '컴공', '917호');
```

INSTEAD OF(3) : 오류 사례

❑ 뷰를 새롭게 생성 : 단, 학과인 PK를 포함하지 않음

```
CREATE or REPLACE VIEW 학생학과정보
AS
            SELECT 학번, 이름, 학생.전화번호, 주소, 학과명, 사무실
            FROM 학생, 학과
            WHERE 학생.학과번호 = 학과.학과번호;
CREATE OR REPLACE TRIGGER 뷰삽입
   INSTEAD OF INSERT
   ON 학생학과정보
   FOR EACH ROW
BEGIN
   INSERT INTO 학과(학과명, 사무실)
     VALUES (:NEW.학과명, :NEW.사무실);
   INSERT INTO 학생(학번, 이름, 주소, 전화번호)
     VALUES (:NEW.학번, :NEW.이름, :NEW.주소, :NEW.전화번호);
END;
```

- **아래 삽입연산 대신 트리거가 실행될 때 오류 발생.**
  - **개체 무결성 위배(PK없는 입력은 불가능함)**

```
INSERT INTO 학생학과정보
VALUES  ('20301-006', '박문수', '200-2000', '부산', '컴공', '917호');
```

INSTEAD OF(4) : 오류 사례

❑ Instead Of 트리거 오류 사례 2

```
CREATE OR REPLACE TRIGGER 뷰삽입
   INSTEAD OF INSERT  -- 삽입작업 대신에 작동 작동하도록 지정
   ON 학생학과정보  -- 뷰에 장착
   FOR EACH ROW
BEGIN
   INSERT INTO 학생(학번, 이름, 학과번호, 주소, 전화번호)
     VALUES (:NEW.학번, :NEW.이름, :NEW.학과번호, :NEW.주소, :NEW.전화번호);
   INSERT INTO 학과(학과번호, 학과명, 사무실)
     VALUES (:NEW.학과번호, :NEW.학과명, :NEW.사무실);
END;
```

- **삽입 순서가 바뀜 :**
  - 자식 테이블에 먼저 삽입
- **아래 삽입연산 대신 트리거가 실행될 때 오류 발생.**
  - 참조 무결성 위해

```
INSERT INTO 학생학과정보
VALUES ('20301-006', '박문수', '200-2000', '부산', 303, '컴공', '917호');
```

다중 트리거, 연쇄 트리거

ORA.7 #1 (3).sql

❑ 다중 트리거

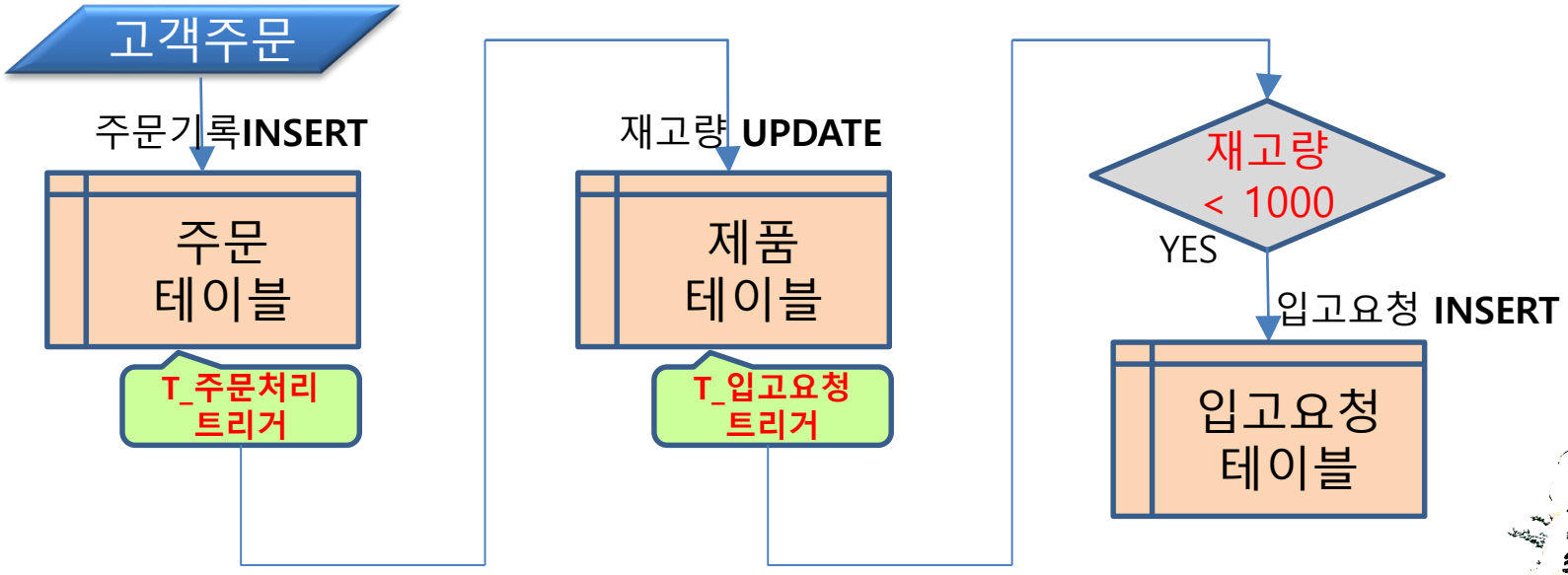
- ▶ 하나의 테이블이나 뷰에 트리거가 여러 개 부착되는 경우
  - ▫ 예) 고객 테이블에 Before Insert / After Insert 트리거 부착
  - ▫ 예) 주문 테이블에 After Insert / After Insert 트리거 부착

❑ 연쇄 트리거

- ▶ 트리거가 또 다른 트리거를 연쇄적으로작동 시킴

연쇄 트리거

- 한빛마트에서 주문을 하면, 아래 업무를 연쇄적으로 해야 함.
  - 주문 : 주문테이블에 INSERT
  - ▶ 주문량만큼 제품 테이블의 재고량을 차감해야 함
  - 재고량참감 : 제품테이블을 UPDATE
  - ▶ 재고량이 1000개 미만으로 떨어지만 입고를 요청해야 함
  - 입고요청 : 입고요청 테이블에 INSERT



```
CREATE TABLE 입고요청
   ("입고요청번호" NUMBER,              "제품번호" CHAR(3 BYTE),
    "제조업체" VARCHAR2(20 BYTE),      "요청일" DATE,
    "요청자" NCHAR(15),                PRIMARY KEY ("입고요청번호")    );
```

```
create or replace TRIGGER T_주문처리
  AFTER INSERT ON 주문
  FOR EACH ROW
BEGIN
  UPDATE 제품 SET 재고량 = 재고량 - :NEW.수량 WHERE 제품번호 = :NEW.주문제품;
END;

CREATE SEQUENCE 입고SEQ;
create or replace TRIGGER T_입고요청
  AFTER UPDATE ON 제품
  FOR EACH ROW
DECLARE
   V_주문량 NUMBER;
   V_재고량 NUMBER;
BEGIN
  SELECT :OLD.재고량 - :NEW.재고량, :NEW.재고량
         INTO V_주문량, V_재고량 FROM DUAL;
   DBMS_OUTPUT.PUT_LINE('주문량 : ' || V_주문량 || ', 재고량 : ' || V_재고량 );
   IF V_재고량 < 1000 THEN
    INSERT INTO 입고요청 VALUES(입고SEQ.NEXTVAL, :NEW.제품번호, :NEW.제조업체,
SYSDATE, USER() );
   END IF;
END;
```

중첩트리거 : 주문 + 연쇄 작업

초기 상태

주문

주문번호	주문고객	주문제품	수량	배송지	주문일자
1 o01	apple	p03	10	서울시 마포구	19/01/01
2 o02	melon	p01	5	인천시 계양구	19/01/10
3 o03	banana	p06	45	경기도 부천시	19/01/11
4 o04	carrot	p02	8	부산시 금정구	19/02/01
5 o05	melon	p06	36	경기도 용인시	19/02/20
6 o06	banana	p01	19	충청북도 보은군	19/03/02
7 o07	apple	p03	22	서울시 영등포구	19/03/15
8 o08	pear	p02	99	강원도 춘천시	19/04/10
9 o09	banana	p04	15	전라남도 목포시	19/04/11
10 o10	carrot	p03	20	경기도 안양시	19/05/22

제품

제품번호	제품명	재고량	단가	제조업체
1 p01	그냥만두	5000	4500	대한식품
2 p02	매운쫄면	2500	5500	민국푸드
3 p03	쿵떡파이	3600	2600	한빛제과
4 p04	맛난초콜릿	1250	2500	한빛제과
5 p05	얼큰라면	2200	1200	대한식품
6 p06	통통우동	1000	1550	민국푸드
7 p07	달콤비스킷	1650	1500	한빛제과

입고요청

입고요...	제품번호	제조업체	요청일	요청자

```
INSERT INTO 주문 VALUES ('o11', 'apple', 'p01', 50, '동의대', SYSDATE);
```

주문번호	주문고객	주문제품	수량	배송지	주문일자
1 o01	apple	p03	10	서울시 마포구	19/01/01
2 o02	melon	p01	5	인천시 계양구	19/01/10
3 o03	banana	p06	45	경기도 부천시	19/01/11
4 o04	carrot	p02	8	부산시 금정구	19/02/01
5 o05	melon	p06	36	경기도 용인시	19/02/20
6 o06	banana	p01	19	충청북도 보은군	19/03/02
7 o07	apple	p03	22	서울시 영등포구	19/03/15
8 o08	pear	p02	99	강원도 춘천시	19/04/10
9 o09	banana	p04	15	전라남도 목포시	19/04/11
10 o10	carrot	p03	20	경기도 안양시	19/05/22
11 o11	apple	p01	50	동의대	20/08/19

제품번호	제품명	재고량	단가	제조업체
1 p01	그냥만두	4950	4500	대한식품
2 p02	매운쫄면	2500	5500	민국푸드
3 p03	쿵떡파이	3600	2600	한빛제과
4 p04	맛난초콜릿	1250	2500	한빛제과
5 p05	얼큰라면	2200	1200	대한식품
6 p06	통통우동	1000	1550	민국푸드
7 p07	달콤비스킷	1650	1500	한빛제과

입고요...	제품번호	제조업체	요청일	요청자

```
INSERT INTO 주문 VALUES ('o12', 'banana', 'p06', 90, '부산시 진구', SYSDATE);
```

주문번호	주문고객	주문제품	수량	배송지	주문일자
1 o01	apple	p03	10	서울시 마포구	19/01/01
2 o02	melon	p01	5	인천시 계양구	19/01/10
3 o03	banana	p06	45	경기도 부천시	19/01/11
4 o04	carrot	p02	8	부산시 금정구	19/02/01
5 o05	melon	p06	36	경기도 용인시	19/02/20
6 o06	banana	p01	19	충청북도 보은군	19/03/02
7 o07	apple	p03	22	서울시 영등포구	19/03/15
8 o08	pear	p02	99	강원도 춘천시	19/04/10
9 o09	banana	p04	15	전라남도 목포시	19/04/11
10 o10	carrot	p03	20	경기도 안양시	19/05/22
11 o11	apple	p01	50	동의대	20/08/19
12 o12	banana	p06	90	부산시 진구	20/08/19

제품번호	제품명	재고량	단가	제조업체
1 p01	그냥만두	4950	4500	대한식품
2 p02	매운쫄면	2500	5500	민국푸드
3 p03	쿵떡파이	3600	2600	한빛제과
4 p04	맛난초콜릿	1250	2500	한빛제과
5 p05	얼큰라면	2200	1200	대한식품
6 p06	통통우동	910	1550	민국푸드
7 p07	달콤비스킷	1650	1500	한빛제과

입고요청번호	제품번호	제조업체	요청일	요청자
1	1 p06	민국푸드	20/08/19	HMART