**Trigger** :

İlişkisel [veri tabanı](http://tr.wikipedia.org/wiki/Veri_taban%C4%B1) yönetim sistemlerinde, bir tabloda belirli olaylar meydana geldiği zaman Triggerler otomatik olarak çalışıyorlar. Öyleyse, Triggerler ekleme, güncelleme, silme işlemlerinden biri gerçekleşmeden önce veya sonra çalışıp ve belirli işlemleri kodlandığı şekilde yerine getiriyorlar.

Bir Trigger oluşturmak için gerekli syntax yapısı aşağıdaki gibidir:

CREATE [OR REPLACE] TRIGGER Ttrigger\_name

{BEFORE|AFTER} Triggering\_event ON table\_name

[FOR EACH ROW]

[FOLLOWS another\_trigger]

[ENABLE/DISABLE]

[WHEN condition]

DECLARE

declaration statements

BEGIN

executable statements

EXCEPTION

exception-handling statements

END;

‘CREATE’ kelimesi yeni bir Trigger oluşturmak için ve ‘REPLACE’ kelimesi, daha önceden var olan bir Triggeri değiştirmek için kullanılır.

**Örnek**: Aşağıdaki Trigger, İNSERT olayı meydana geldikten sonra otomatik olarak çalışıp, belirli bir işlemi yapacaktır. Bu Trigger, BORC tablosuna yeni bir satır eklendikten sonra çalışıp ve ŞÜBE tablosunda Mevduat Birikimini güncelleyecektir. Bu iş için Mevduat birikiminden, alınan Borç mıktarını azaltacaktır.

CREATE OR REPLACE

TRIGGER BORCGUNCELLE

AFTER INSERT ON BORC

REFERENCING NEW AS YENI

FOR EACH ROW

BEGIN

NULL;

 UPDATE SUBE SET MEVDUAT\_BIRIKIMI=MEVDUAT\_BIRIKIMI-:YENI.BORC\_MIKTARI

WHERE SUBE\_ISIM=:YENI.SUBE\_ISIM;

END;

**Function:**

Daha önce SQL Developer içinde yer alan hazır fonksiyonları kullandık. Şimdi kendi fonksiyonlarımızı yaratmak ve onları sorgularımızda kullanmak istiyoruz. Bir fonksiyon tek bir değer döndürecektir. Fonksiyonlar, Stored Procudure ler gibi unique isim taşıması gereken yapılardır.

Bir fonksiyon oluşturabilmek için gerekli syntax yapısı :

CREATE [OR REPLACE] FUNCTION function\_name

(parameter list)

RETURN datatype

IS

BEGIN

<body>

RETURN (return\_value);

END;

**Örnek**: Aşağıdaki tanımlanan fonksiyon, EURO değerini parameter olarak alıp ve TL değerini döndürüyür.

CREATE OR REPLACE

FUNCTION EURO\_DAN\_TL\_YE ( EURO IN NUMBER ) RETURN NUMBER AS

BEGIN

 RETURN (EURO\*2.36);

END EURO\_DAN\_TL\_YE;

**Procedure:**

Bir ya da birden fazla işlem yapmak için yazılır. Bir Procedure, değer döndürmek zorunda değil ve değer döndürmeyebilir.

Bir Procedure oluşturabilmek için gerekli syntax yapısı :

CREATE OR REPLACE PROCEDURE name

[(parameter[, parameter, ...])]

AS

[local declarations]

BEGIN

executable statements

[EXCEPTION

exception handlers]

END [name];

**Örnek-1**: Aşağıdaki tanımlanan procedure, Bir FOR içerisinde HESAP tablosunda yer alan Mevduat Mıktarını değiştiriyor.

CREATE OR REPLACE
PROCEDURE FOR\_DENEME AS
BEGIN
    FOR i IN 1..10 LOOP
       UPDATE HESAP SET MEVDUAT\_MIKTARI=MEVDUAT\_MIKTARI+1;
    END LOOP;
    COMMIT;
END FOR\_DENEME;

**Örnek-2**: Aşağıdaki tanimlanan Procedure, İsim, Şübe, Mevduat ve Hesap numarasını parametre olarak alıp ve yeni bir müşteriye yeni bir hesap açıyor. Bunun için Müşteri, Hesap ve Mevduat\_sahibi tablolarına yeni satırlar ekleniyor.

CREATE OR REPLACE
PROCEDURE MEVDUAT\_EKLE (
   ISIM IN VARCHAR2,
   SUBE IN VARCHAR2,
   MEVDUAT IN NUMBER,
   HESAPNO IN NUMBER
) AS
BEGIN
   INSERT INTO MUSTERI (MUSTERI\_ISIM) VALUES(ISIM);
   INSERT INTO HESAP (HESAP\_NO, SUBE\_ISIM, MEVDUAT\_MIKTARI) VALUES(HESAPNO,
SUBE, MEVDUAT);
   INSERT INTO MEVDUAT\_SAHIBI (MUSTERI\_ISIM, HESAP\_NO) VALUES(ISIM,
HESAPNO);
END MEVDUAT\_EKLE;