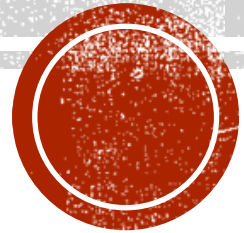
STORED PROSEDÜR



STORED PROSEDÜR NEDİR ?

- Stored Procedure(Saklı Yordam) defalarca kullanılabilen, değişken alabilen, hızlı çalışan, veri tabanına kayıt edilmiş SQL ifadeleridir.
- Sık kullanılan SQL ifadeleriniz varsa bunları stored procedure olarak kaydetmek zaman ve performans açısından kazançlı olacaktır.
- Fayadaları:
- Güvenlik, saklı yordamları hangi kullanıcıların görebileceğini ve çalıştırabileceğini kısıtlayabilirsiniz.
- Maintainability, Stored prosedürler başka stored prosedürleri çağırabildiklerinden karmaşık ifadeleri yönetilebilir parçalalara ayırmanıza olanak tanır.
- Hız, bir stored prosedür ilk kez çalıştığında yürütme planı oluşturulur ve bu plan önbellekte saklanır. Sonraki işlemlerde plan kullanılarak derleme gerçekleştirilmez.

STORED PROSEDÜR KULLANIMI

- Parametresiz Prosedür,

```
CREATE OR REPLACE PROCEDURE parametresiz()
language plpgsql
as $$
begin
raise notice 'Bu Bir Parametresiz Procedure ';
end; $$
```

- Prosedürü çağırma,

```
Call parametresiz()
```

- "Create or replace" komutu ile başlarsak ilerleyen zamanlarda prosedür içerisinde değişiklik yapabilme olanağı tanır

PARAMETRELİ STORED PROSEDÜRLER

- Parametreli Prosedür
- CREATE OR REPLACE PROCEDURE parametreli(p1 integer,p2 varchar,p3 integer,p4 integer)
- Language plpgsql
- As $$
- begin
- insert into dersler (ogrencino,derskodu,derskredi,dersnotu) values (p1,p2,p3,p4);
- end; $$
- $$$$Prosedür Çağırma
- call fakulte_ekle(5,'Besyo')

VİEW

- View sanal bir tablo olmasına rağmen veri tabanında gerçek bir tablo da olduğu gibi satır ve sütunlara sahiptir.
- Sorgunun tüm karmaşıklığını basit bir şekilde soyutlar ve tek bir tabloymuş gibi sorgulamamıza izin verir. Ancak veriyi sorgu zamanında dinamik oluşturur.
- Birden fazla join, iç içe sorgu, COUNT, SUM vb. Hesaplamaları içeren sorguları uygulama katmanında yazmak yerine view oluşturup bu verileri veri tabanı katmanından direkt alabilirisniz.

VİEW KULLANIMI

- Join içeren bir sorgu yazalım:
- Create view view1
- As
- Select o.ogrencino, d.derskodu, d.derskredi,d.dersnotu from dersler d
- inner join ogrenci o on d.ogrencino = o.ogrencino;
- Yazdığımız View'i çağıralım
- Select * from view1

VİEW GÜNCELLEME

- Burada dikkat edilmesi gereken konu eğer view güncellenmek isteniyorsa ilk önce bulunan view silinmeli sonra güncellenecek kısım yazılmalıdr.
- Drop view if exist view1;
- Create view view1
- As
- Select o.ogrencino, d.derskodu, d.derskredi,d.dersnotu from dersler d
- inner join ogrenci o on d.ogrencino = o.ogrencino inner join bolum b on b.ogrencino=o.ogrencino;

TRİGGERS

- Trigger (Tetikleyici), en genel anlamda kayıt ekleme, silme ve güncelleme sonucunda otomatik olarak devreye giren özel bir tür Stored Procedure olarak kabul edilir.
- Tetikleyici (Trigger) yapısı, ilişkisel veri tabanı yönetim sistemlerinde, bir tabloda belirli olaylar meydana geldiğinde veya gelmeden önce otomatik olarak çalışan özel bir store procedure türüdür.

TRİGGER OLUŞTURMA SENARYOSU

- Öncelikle istediğimiz değişim karşısında kontrol(ana) tablomuzu oluşturalım.
- CREATE TABLE UserLogs (
- log_id SERIAL PRIMARY KEY,
- user_id INT,
- action_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP
-);

TRİGGER OLUŞTURMA SENARYOSU

- Bu adımda ise ise işlem yapacak olan fonksiyon oluşturulur.
- CREATE OR REPLACE FUNCTION LogUserInsert()
- RETURNS TRIGGER AS $$
- BEGIN
- INSERT INTO UserLogs (user_id)
- VALUES (NEW.id); -- NEW: Eklenen satırın verisine erişim sağlar
- RETURN NEW;
- END;
- $$ LANGUAGE plpgsql;

TRİGGER OLUŞTURMA SENARYOSU

- Son olarak ise trigger oluşturulur.
- CREATE TRIGGER AfterUserInsert
- AFTER INSERT ON Users
- FOR EACH ROW
- EXECUTE FUNCTION LogUserInsert();

TRİGGERS DEĞİŞKEN TANIMLAMA

- Burada önemli olan komut declare ile değişken tanımlayıp içine değer atama

```
create or replace function test1()
 returns trigger
 as
 $$
 declare
 uzunluk integer;
 begin
 uzunluk:=(select length(ad) from fakülte order by id desc limit 1);
 update toplamfakulte2 set sayi=sayi+uzunluk;
 return new;
 end;
 $$
 language plpgsql;
```

TRİGGERS DEĞİŞKEN TANIMLAMA

- create Trigger Test1trg
- after insert
- on fakülte
- for each row
- execute procedure test1();
- ----------
- Veri ekleme
- Insert into fakülte(id,ad) values(değerler)