

Bootcamp Bitirme Projesi Raporu

Tekstil Fabrikası veritabanı hazırladım. Bu veritabanına Users, UserTypes, Workers, WorkerTypes, Items, ItemTypes, Inventories, InventoryMovements, Logs isimli 9 tane tablo ekledim.

Users tablosu uygulamayı kullanabilen kişilerin datalarını tutmaktadır. UserTypes içerisinde bu dataların tipleri belirtilmektedir. İçerisine Director, Manager ve Team Lead eklenmiştir. UserTypes ve Users tabloları birbiri ile ilişkilendirilmiştir. Users tablosu çalıştırıldığında UserID gelmektedir. Bu sayede kullanıcının tür bilgisine ulaşılmaktadır.

Workers tablosuna fabrikada çalışanların dataları girilmiştir. Users tablosundaki gibi burada da bir tür tablosu yapılmıştır. Burada bu tablonun adı WorkerTypes olarak verilmiştir. Aynı şekilde bu tablolar birbiri ile ilişkilendirilmiştir.

Items tablosunda malzeme ve ürün olan datalar tutulmaktadır. Malzemeler, yapılacak ürün için gerekli olan datalardır. Ürünler de malzemeler birleştirilerek oluşturulan datalardır. Items tablosu içinde sütun olarak ItemID, ItemTypeID, CreateDate, LastUpdateDate, ItemDescription bulunmaktadır. ItemTypeID, ItemTypes tablosundan gelmektedir. Bu tabloya iki farklı veri eklenmiştir. Bunlar malzeme ve üründür. CreateDate item değerinin veritabanına girildiği tarihi gösteren data bilgisidir. LastUpdateDate bu malzeme ya da ürün üzerinde değişiklik yapıldığında bunun belli olmasını sağlayan sütundur. Son olarak ItemDescription alanı, item datalarının isimlerinin tutulduğu alanı göstermektedir.

Inventories tablosu envanterdeki dataları göstermektedir. Yani fabrikada bulunan malzemeleri, ürünleri gösteren tablodur. Burada InventoryID, ItemID, ItemTypeID ve Stock olarak sütunlar bulunmaktadır. ItemID, item değerini gösterir. ItemTypeID bu gelen item değerinin türünü belirtir. Stock ise bu item değerinden envanterde kaç tane malzeme ya da ürün olduğunu gösteren sütundur.

InventoryMovements tablosu içerisinde InventoryMovementID, InventoryID, UsedItemID, WorkerID ve Qty isimli sütun alanlarını tutmaktadır. Burada envanterdeki değişiklikler gösterilmiştir. InventoryID ile hangi üründe, malzemede değişiklik yapıldığını alırız. UsedItemID işleme giren item ID'yi getirmektedir. WorkerID bu envanter değişikliğinde rol oynayan çalışanı getirmektedir. Envanterdeki ürüne bir ekleme ya da azaltma işlemi yapılmışsa Qty sütununda gözükmemektedir. InventoryMovements tablosuna 100.000 data eklemesi yapılmıştır.

Logs tablosunda envanterdeki hareketler için çalıştırılmış olan stored prosedure dataları tutulmaktadır. Stored Procedure'da silme işlemi gerçekleştirildiğinde hangi data silinmiş bunu görüntüleyebilmek için bu tablo kullanılmaktadır.

Kullanılan Sorgular

Trigger Sorgusu

InventoryMovements tablosunun içerisine ChangeStock isimli bir trigger eklenmiştir. Bu trigger InventoryMovements tablosuna insert yapıldıktan sonra çalışmaktadır.

InventoryMovements tablosunda Qty isimli bir sütun vardır. Qty değişkeni envanterde bulunan adet alanını artırmak veya azaltmak için kullanılan bir alandır. Bu trigger insert atılırken Qty değişkenine atanan değeri Inventories tablosuna uygulamak için kullanılır.

```
Create trigger ChangeStock
on InventoryMovements
after insert
as
begin
Declare @Qty int
Declare @UsedItemID int
select @Qty = QTY , @UsedItemID = UsedItemID from inserted

update Inventories set Stock = Stock - ( @Qty * -1)
where @UsedItemID = Inventories.ItemID

end
```

Stored Prosedure Sorgusu

Bu prosedür envantere eklenmiş olan ürünün stoğunu düzeltmek için kullanılır. Qty değeri InventoryMovements tablosundan gelmektedir. Yazılan trigger sayesinde bu tabloda atılan insert'te Qty değeri Inventories tablosunu etkilemektedir. Bunu geri çekmek için kullanılır.

```
Create PROCEDURE sp_DeleteMovement

    @InventoryMovementId int
AS
BEGIN
Declare @ItemID int
Declare @qty int
select @ItemID = UsedItemID , @qty = qty from InventoryMovements where
InventoryMovementID = @InventoryMovementId
Update Inventories set Stock = Stock - @qty where ItemID = @ItemID
insert into Logs values (NewID(),@InventoryMovementId,@ItemID,@qty,'Fix from
sp_DeleteMovement')
delete from InventoryMovements where InventoryMovementID = @InventoryMovementId
END
GO
```

Yukarıdaki prosedürü çalıştırmak için aşağıdaki sql koduna benzer bir kod kullanılmalıdır.

```
exec sp_DeleteMovement
@InventoryMovementId = '52'
```

View Sorgusu

Eklenen view sorgusunda ItemDetail verilmiştir. Items, ItemTypes ve Inventories tabloları kullanılarak bir view yazılmıştır. Burada ItemTypeID, ItemTypeName, ItemID, ItemDescription ve Stock sütun alanları getirilmiştir.

```
create view ItemDetail
as
select
    Items.ItemTypeID,
    ItemTypes.ItemTypeName,
    Items.ItemID,
    Items.ItemDescription,
    Inventories.Stock
from
    Items
    inner join ItemTypes on Items.ItemTypeID = ItemTypes.ItemTypeID
    inner join Inventories on Items.ItemID = Inventories.ItemID
```

Yukarıdaki view alanını gösterebilmek için aşağıdaki sql sorgusu yazılmalıdır.

```
select * from ItemDetail
```

100.000 Datanın Giriş Sorgusu

Aşağıdaki sorguda while kullanılarak rastgele data girişi gerçekleştirilmiştir. 50.000 tane Qty değeri pozitif olan data, 50.000 tane Qty değeri negatif olan data eklenmiştir.

```
declare @counter int = 0

while @counter < 50000
begin
    declare @ItemID int = (select top 1 ItemId from Items order by NEWID())
    declare @WorkerID int = (select top 1 WorkerID from Workers order by NEWID())
    declare @qty int = (select FLOOR(RAND() * -10 -1+1)+1)
    declare @InventoryID int = (select top 1 InventoryID from Inventories where
ItemID = @ItemID)

    insert into InventoryMovements
    select @InventoryID,@ItemID,@WorkerID,@qty

    set @counter = @counter + 1
end
```

Hazırlayan Sena KARADENİZ