**VERİTABANI –II**

**1.TRANSACTION İŞLEMLERİ**

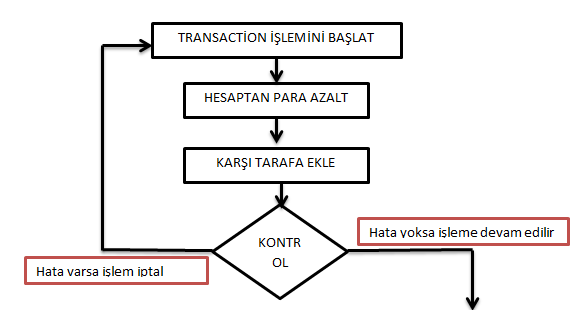
Transaction, bir veya birden fazla sql ifadesinin oluşturmuş olduğu en küçük işlem birimi olarak tanımlanmaktadır. Tansactionlar’ da bir grup işlem bir arada gerçekleşmesine bağlı bir kurallar bütünüdür. Bu kurallar doğrultusunda çalıştırılacak sql sorgularını tamamının başarılı olmasına bağlıdır. Aksi halde herhangi bir sorgunun hatalı çalışması bütün işlemlerin başarısız sayılıp geri alınmasına neden olacaktır. Bütün işlemlerin başarılı olması transaction işleminin onaylanmasını sağlayacaktır.

Transaction ‘a örnek olarak banka işlemleri verilebilir. Bir banka kullanıcısı kendi hesabından bir ödeme yapmak istediği zaman veri tabanında bulunan temel iki tablo üzerinde değişikliğin bir biri ile eş zamanlı yapılması gerekmektedir. Ödeme yapıldığımda hesap tablosu üzerinden ödenecek miktarın düşülmesi işlemi gereçekleştiğinde karşı tarafın hesabında bu artışın gerçekleşmesi gerekmektedir. Aksi taktirde bu iki işlemden birinin doğru çalışmaması hesaplar arasında bir tutarsızlığın oluşmasına neden olacaktır. İşte bu durumda transaction blokları devreye girmektedir. Hesaptan düşme işlemnin başarılı olması ve başka hesapta bu miktarın eklenmesi olayı başarısız olması durumunda bu iki işlemde geri alınır ve işlemlerin tekrar yapılması kullanıcıdan istenir.

*Update Banka\_Hesabı Set H\_bakiyesi=H\_bakiyesi-2000 Where HesapID=123*

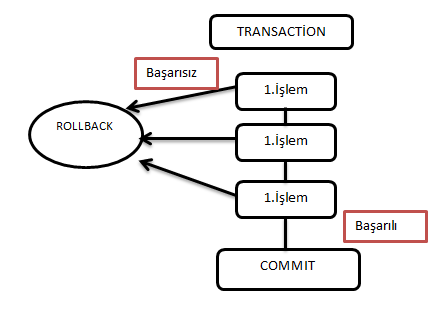
*Update Banka\_Hesabı Set H\_bakiyesi=H\_bakiyesi +2000 Where HesapID=1234*

Yukarıda gösterilen iki işlem bir işlemmiş gibi kabul edilmektedir.Bu işlemlerden birinin hatalı çalışması durumunda diğer işlemde hatalı sayılır ve geri alınma işlemi(rollback) devreye girer. Aksi durumda her iki işlemin de başarılı bir şekilde gerçekleşmesi durumunda değişiklikler kalıcı olarak sisteme kaydedilir(commit).



Transaction işlemleri yapılırken transaction log denilen dosya yapıları kullanılmaktadır.Bu yapılar ile transaction işlemi başladıktan sonra sql ifadesi için kullanılan veriler hafızaya yüklenir ve yapılacak değişiklikler hafızada bulunan bu bilgiler üzerinde gerçekleştirilir. İşlemler bittikten sonra hafızadaki bu bilgiler diskte bulunan dosyaların üzerine tekrardan yazılır.İşlemlerin hatasız bir şekilde yapılması durumunda veriler kalıcı olarak disk üzerinde bulann Transaction Log dosyaları üzerine yazılır ve dosya kapatılır.

Transaction işlemlerinin hatasız bir şekilde tamamlanması durumunda COMMİT ifadesi ile yapılan değişiklikler kalıcı bir halde kaydedilir. Aksi bir durumda transaction yığınında bulunan herhangi bir ifadenin hatalı çalışması durumda ROLLBACK ifadesi ile bütün değişiklikler geri alınır.



Transaction işlemleri için kullanılan bazı ifadeler aşağıdaki şekildedir;

**Transaction Log:** Disk üzerinde bulunan bu dosya Transaction ile ilgili bir veri tabanında yapılan bütün değişiklikler hakkındaki bilgileri kaydetmektedir. Bu Dosya sıralı erişli bir dosya yapısına sahip olmakla birlikte verilerde bu dosyaya sıralı bir şekilde kaydedilmektedir.

**Commit:** Veritabanı üzerinde transaction yapısı kullanılarak gerçekleştirilen işlemlerin başarılı olması durumunda veritabanına kalıcı olarak kaydedilmesini sağlayan ifadedir.

**Rollback :** Transaction bloğu içinde bulunan herhangi bir sql kodunda meydana gelebilecek bir hata veya yanlışlık durumunda işlemlerin kalıcı olarak kaydedilmesini engelleyen ve yapılmış olan değişikliklerin geri alınmasını sağlayan ifadedir.

MS SQL server veritabanı sistemlerinde transaction işlemleri BEGIN TRANSACTION ifadesi ile başlar ve COMMIT veya ROLLBACK ifadesi ile sonlandırılır.

Örnek;

İşlem yapacağımız örnek veritabanı tablosunun aşağıdaki gibi olduğunu kabul edelim;

|  |  |  |
| --- | --- | --- |
| HesapID | Kullanici\_Adi | H\_bakiyesi |
| 123 | Taner UÇKAN | 3000 |
| 1234 | Faruk AYATA | 2500 |

*BEGIN TRANSACTION*

*Update Banka\_Hesabı Set H\_bakiyesi=H\_bakiyesi-2000 Where HesapID=123*

*Update Banka\_Hesabı Set H\_bakiyesi=H\_bakiyesi +2500 Where HesapID=1234*

*Select \* from Banka\_Hesabı Where HesapID=123*

*ROLLBACK*

Yukarıdaki işlemlerden görüldüğü üzere belirtilen kod bloğunun öncelikle 123 nolu HesapID ye sahip kullanıcının hesabından 2000 TL eksiltmesi ve ardından HesapID si 1234 olan kullanıcının hesabına 2500 TL eklemesi beklenmektedir.Son olarak Banka\_Hesabı tablosunun içeriğini listelmelidir. Bu işlem çalıştırıldığında üçüncü satıra geldiğinde tablonun son hali aşağıdaki şekilde olacaktır.

|  |  |  |
| --- | --- | --- |
| HesapID | Kullanici\_Adi | H\_bakiyesi |
| 123 | Taner UÇKAN | **1000** |
| 1234 | Faruk AYATA | **5000** |

Transactionun son satırında dikkat edilirse Rollback ifadesi ile sonlandırıldığı görülmektedir. Kod bloğunun bu ifade ile bitmesi öncesinde yapılan bütün değişikliklerin geri alınması anlamına gelmektedir. Son satırda Rollback ifadesi yerine Commit ifadesi olmuş olsaydı yapılan bütün değişiklikler kabul edilip kalıcı olarak veri tabanına kayıt yapılacaktı.

Veri tabanı sorguları yazarken sorgular çoğunlukla yukarıda verdiğimiz örnek gibi kısa olmayabilir. Bu tur durumlarda Transaction bloğunu savepoints adı verilen alt bölümlere ayırmak mümkündür. Bu sayede savepoint ler ile ayrılmış her bölüm için rollback ve commit işlemleri ana Transaction bloğundan ayrı bir şekilde yapılabilmektedir.

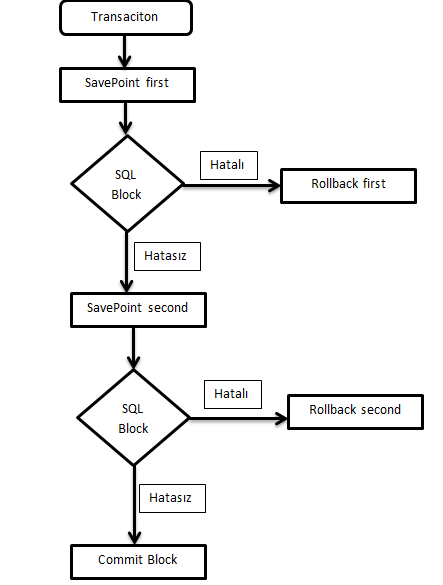
Alt transaction bloğu aşağıdaki gibi tanımlanmaktadır;

*SAVE TANSACTION save\_point\_adi*

Tanımlanan alt transaction bölgeleri için Rollback veya Commit işlemleri aşağıdaki şekilde tanımlanmaktadır;

ROLLBACK veya COMMIT TRANSACTION save\_point\_adi

Alt Transaction işlemlerinin çalışma mantığı aşağıda ki şekilde gösterilmektedir.



Örnek; Daha önceki örneklerde de kullanmış olduğumuz tablolar üzerinde savepoint işlemini aşağıdaki gibi gerçeleştirelim;

*BEGIN TRANSACTION*

*SAVE TRANSACTION first*

*Update Banka\_Hesabı Set H\_bakiyesi=H\_bakiyesi-2000 Where HesapID=123*

*Select \* from Banka\_Hesabı*

*ROLLBACK TRANSACTION first*

*Select \* from Banka\_Hesabı*

*Update Banka\_Hesabı Set H\_bakiyesi=H\_bakiyesi +2500 Where HesapID=123*

*Select \* from Banka\_Hesabı*

*COMMIT*

*Select \* from Banka\_Hesabı*

Yukarıda belirtilen Transaction bloğuna bakıldığında ilk olarak first adında bir savepoint oluşturulmakta ve yapılması istnilen sql kodları yazılmakta.Bloğun sonunda işlemlerin veritabanına kalıcı olarak kaydedilmesini veya tamamen geri alınmasını isteyebiliriz.Yukarıda bellirtilen alt blokta işlemler tamamen geri alınmakta ve HesapID si 123 olan kullanıcının hesabında herhangi bir değişiklik olmamaktadır.İşlem devam ettiğinde aynı hesap ID sine sahip kullanıcı hesabında ekleme işlemi yapılmakta ve ana Transaction bloğu Commit ifadesi ile sonlandırılmaktadır.Bu ifade , alt transaction bloğundan sonra yapılan bütün değişikliklerin veritabanına kalıcı olarak yansıtılmasını sağlamaktadır.Bu işlemler sonucunda tablodaki bütün bilgiler listelenmekmektedir.

Yapılan işlemlerin sonucunda tabloda meydana gelen değişiklikler aşağıda adım adım gösterilmektedir.

Tablonun ilk Hali;

|  |  |  |
| --- | --- | --- |
| HesapID | Kullanici\_Adi | H\_bakiyesi |
| 123 | Taner UÇKAN | 3000 |
| 1234 | Faruk AYATA | 2500 |

İlk Select ifadesinin sonucu;

|  |  |  |
| --- | --- | --- |
| HesapID | Kullanici\_Adi | H\_bakiyesi |
| 123 | Taner UÇKAN | **500** |
| 1234 | Faruk AYATA | 2500 |

İkinci seelect ifadesinin sonucunda yapılan değişikliklerin geri alındığı görülmektedir.

|  |  |  |
| --- | --- | --- |
| HesapID | Kullanici\_Adi | H\_bakiyesi |
| 123 | Taner UÇKAN | **3000** |
| 1234 | Faruk AYATA | 2500 |

Üçüncü select ifadesine bakıldığında 123 nolu hesap bakiyesine ekleme yapıldığı görülmekredir;

|  |  |  |
| --- | --- | --- |
| HesapID | Kullanici\_Adi | H\_bakiyesi |
| 123 | Taner UÇKAN | **5500** |
| 1234 | Faruk AYATA | 2500 |

Tablonun son halinden görüldüğü üzere hesaptan eksiltme işlemi sonucu Rollback ifadesi ile geri alındığından hesap tutarına yansıma olmamış devamında ise hesaba ekleme işlemi Commit ifadesi ile sonlandırıldığından veritabanın da bulunan tabloya kalıcı kayıt olarak yansıtılmıştır.

|  |  |  |
| --- | --- | --- |
| HesapID | Kullanici\_Adi | H\_bakiyesi |
| 123 | Taner UÇKAN | **5500** |
| 1234 | Faruk AYATA | 2500 |