# Proje 2: Veritabanı Yedekleme ve Felaket Kurtarma

## 1. Proje Tanımı ve Amaç

Bu projenin amacı, SQL Server ortamında çalışan bir veritabanı için yedekleme stratejileri oluşturmak ve olası bir felaket durumunda veritabanının sorunsuz bir şekilde geri yüklenmesini sağlamaktır. Gerçek hayatta sistem çökmesi, veri kaybı, insan hatası veya kötü amaçlı yazılımlar nedeniyle veritabanlarında veri kaybı yaşanabilir. Bu nedenle, düzenli yedekleme almak ve alınan yedekleri gerektiğinde başarıyla geri yükleyebilmek, veri güvenliğinin temelini oluşturur.

Bu proje kapsamında:

* CompanyBackupTest2 adlı örnek bir veritabanı oluşturulmuş,
* İçerisine test verileri eklenmiş,
* Ardından **tam (full) yedekleme** ve **log (artımlı) yedekleme** işlemleri yapılmıştır.
* Daha sonra bu veritabanı manuel olarak silinmiş ve geri yükleme senaryosu uygulanmıştır.
* Süreç sonunda veriler başarıyla geri getirilmiş ve kayıtların bütünlüğü test edilmiştir.

Bu uygulama ile hem yedekleme mantığı, hem de restore sürecinde dikkat edilmesi gereken WITH NORECOVERY, WITH RECOVERY, WITH REPLACE gibi anahtar kelimelerin doğru kullanımı pratik edilmiştir.

## 2. Kullanılan Araçlar ve Ortam Bilgisi

| **Bileşen** | **Açıklama** |
| --- | --- |
| **Veritabanı Sunucusu** | Microsoft SQL Server 2019 |
| **Yönetim Aracı** | SQL Server Management Studio (SSMS) 18.x |
| **İşletim Sistemi** | Windows 10 / 11 (64-bit) |
| **Yedekleme Yolu** | C:\Backup\ klasörü (önceden elle oluşturulmuş olması gerekir) |
| **SQL Script Formatı** | T-SQL (Transact-SQL) |
| **Yedekleme Türleri** | Full Backup (.bak), Log Backup (.trn) |
| **Donanım (geliştirme ortamı)** | 8 GB RAM, Intel i5/7 veya eşdeğeri, SSD disk önerilir |

Bu ortamda geliştirilen örnek proje, gerçek bir senaryonun basit modellemesi olup ileride zamanlanmış yedekleme işleri, differential yedekleme ve Database Mirroring gibi ileri düzey konulara temel teşkil etmektedir.

## 3. Veritabanı Oluşturulması ve Test Verileri

Bu aşamada örnek senaryo için CompanyBackupTest2 adında bir veritabanı oluşturulmuştur. Veritabanı içinde Employees adında basit bir tablo tanımlanmış ve içine test verileri eklenmiştir. Bu veriler, yedekleme ve geri yükleme işlemlerinde takip edilebilirliği sağlamak amacıyla farklı zamanlarda eklenmiştir.

**Oluşturulan Yapılar:**

* **Veritabanı Adı:** CompanyBackupTest2
* **Tablo:** Employees
* **Kolonlar:**
  + ID (int, identity, primary key)
  + FullName (nvarchar(100))
  + Email (nvarchar(100))

**Eklenen Test Verileri:**

| **ID** | **FullName** | **Email** |
| --- | --- | --- |
| 1 | Ali Veli | ali@example.com |
| 2 | Ayşe Fatma | ayse@example.com |
| 3 | Zeynep Yılmaz | zeynep@example.com |
| 4 | Mehmet Karaca | mehmet@example.com |

Verilerin bir kısmı **tam yedekten önce**, bir kısmı ise **tam yedekten sonra** eklenmiştir. Bu sayede, log yedeğinin başarıyla çalışıp çalışmadığı geri yükleme işlemi sonrası kolayca test edilmiştir.

**Kullanılan SQL Komutları (özet):**



## 4. Yedekleme Stratejileri

Veri kayıplarının önüne geçmek için aşağıdaki yedekleme stratejisi uygulanmıştır:

**4.1. Tam (Full) Yedekleme**

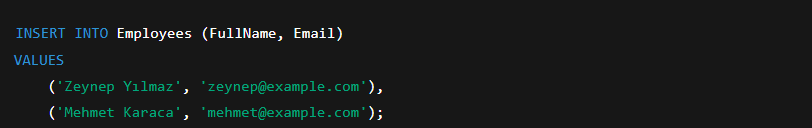
Veritabanının tamamı C:\Backup\CBT2\_FULL.bak konumuna .bak uzantılı bir dosya olarak yedeklenmiştir. Bu yedek, sistemin o anki bütün verilerini içerir ve geri yükleme işleminin temelini oluşturur.

**4.2. Recovery Model Değiştirme (Full Mode)**

SQL Server’da log yedeklemesi alabilmek için veritabanının recovery mode’unun FULL olması gerekir. Bu ayar aşağıdaki komutla yapılmıştır:

**4.3. Yeni Veri Girişi**

Full yedek alındıktan sonra birkaç yeni veri daha eklenmiştir. Bu veriler, sadece log yedeğiyle alınabildiği için geri yükleme testlerinde kritik rol oynamıştır.

**4.4. Log (Artımlı) Yedekleme**

Full yedekten sonra girilen verileri kapsayacak şekilde .trn uzantılı log yedeği alınmıştır. Bu işlem, veri tabanı içindeki değişiklikleri kurtarmak için kullanılır.



## 5. Felaket Senaryosu ve Veritabanı Silme

Gerçek hayat sistemlerinde çeşitli nedenlerle veri kaybı yaşanabilir. Örneğin:

* Kullanıcının yanlışlıkla veritabanını silmesi
* Sunucu çökmesi
* Donanım arızaları
* Fidye yazılımı saldırıları

Bu proje kapsamında böyle bir **felaket senaryosu simüle edilmiştir.**

Yedekleri aldıktan sonra CompanyBackupTest2 adlı veritabanı aşağıdaki komutla **kasıtlı olarak silinmiştir**:

Silme işlemi sonrası veritabanına erişim imkânsız hâle gelmiştir. Bu noktada, sadece **önceden alınan yedek dosyaları** sayesinde veritabanını kurtarmak mümkündür.

## 6. Yedekten Geri Yükleme (Restore) Süreci

Yedekten geri yükleme süreci dikkatli bir şekilde yürütülmelidir. Bu süreçte restore işlemlerinin doğru sırayla yapılması gerekir. Aksi takdirde hata alınabilir veya veri kayıpları oluşabilir.

**6.1. Full Backup Geri Yükleme**

İlk olarak alınan **tam yedek (CBT2\_FULL.bak)** geri yüklenmiştir. Bu işlem sırasında **WITH NORECOVERY** seçeneği kullanılmıştır. Bu seçenek, SQL Server’a geri yükleme işleminin henüz tamamlanmadığını ve başka yedeklerin de yükleneceğini belirtir.

Ayrıca, veritabanı önceden silinmiş olduğu için WITH REPLACE ifadesi ile overwrite (ezme) yetkisi verilmiştir.

**6.2. Log Backup Geri Yükleme**

Tam yedeğin ardından alınan **log yedeği (CBT2\_LOG.trn)** geri yüklenmiştir. Bu adım, full yedekten sonra yapılan veri eklemelerini (örneğin Mehmet ve Zeynep adlı kişilerin kayıtları) geri getirmeyi sağlar.

Bu işlemin sonunda WITH RECOVERY ifadesi kullanılarak restore süreci tamamlanmıştır:



**🔍 Kontrol ve Sonuç**

Restore işlemi tamamlandıktan sonra veritabanı tekrar kullanılabilir hâle gelmiştir. Geri yüklenen veritabanı üzerinde yapılan sorgularla tüm verilerin eksiksiz geldiği doğrulanmıştır:

Bu sorgu sonucunda hem tam yedekte yer alan kayıtlar hem de log yedeğiyle eklenen yeni kayıtlar eksiksiz olarak görüntülenmiştir.

## 7. Veri Doğrulama ve Kayıt Kontrolü

Restore işlemi tamamlandıktan sonra veritabanının geri yüklenip yüklenmediğini ve tüm kayıtların eksiksiz olarak döndüğünü kontrol etmek için aşağıdaki sorgu çalıştırılmıştır:

**Beklenen Kayıtlar:**

| **ID** | **FullName** | **Email** |
| --- | --- | --- |
| 1 | Ali Veli | ali@example.com |
| 2 | Ayşe Fatma | ayse@example.com |
| 3 | Zeynep Yılmaz | zeynep@example.com |
| 4 | Mehmet Karaca | mehmet@example.com |

* **Kayıt 1 ve 2:** Tam (Full) yedekte yer alıyordu.
* **Kayıt 3 ve 4:** Log yedeğiyle alınmıştı.

Yukarıdaki sorgunun sonucunda tüm kayıtların başarıyla döndüğü görülmüştür. Bu da yedekleme ve geri yükleme işleminin eksiksiz ve doğru bir şekilde yapıldığını doğrulamaktadır.

## 8. Karşılaşılan Sorunlar ve Çözümler

Projeyi uygulama sürecinde aşağıdaki sorunlar ile karşılaşılmış ve uygun çözümlerle aşılmıştır:

**🔸 Sorun 1: Log yedeği alınırken hata**

* **Hata:** BACKUP LOG cannot be performed because there is no current database backup.
* **Sebep:** Full yedek alınmadan önce log yedeği alınmak istendi.
* **Çözüm:** Önce WITH INIT parametresi ile tam yedek alındı, ardından log yedeği başarıyla oluşturuldu.

**🔸 Sorun 2: Restore sırasında "backup set" hatası**

* **Hata:** The backup set holds a backup of a database other than the existing 'CompanyBackupTest2' database.
* **Sebep:** DROP işlemi sonrası eski veritabanı bellekte kalmıştı, yeni yedeği restore ederken çakışma yaşandı.
* **Çözüm:** WITH REPLACE ifadesi kullanılarak eski kalıntılar ezildi ve restore başarılı oldu.

**🔸 Sorun 3: Log zinciri hatası (gap in the log chain)**

* **Hata:** This log cannot be restored because a gap in the log chain was created.
* **Sebep:** Restore sıralaması ya da WITH NORECOVERY kullanımında hata yapıldı.
* **Çözüm:** Restore sıralaması yeniden gözden geçirildi ve NORECOVERY → NORECOVERY → RECOVERY sırası düzeltildi.

Bu sorunlar sayesinde SQL Server’da yedekleme ve geri yükleme işlemlerinde dikkat edilmesi gereken kurallar pratikte öğrenilmiş oldu.

## 9. Sonuç ve Değerlendirme

Bu proje kapsamında bir veritabanının nasıl yedekleneceği ve olası bir felaket durumunda nasıl geri yükleneceği detaylı bir şekilde uygulanmıştır. Aşağıdaki adımlar başarıyla tamamlanmıştır:

* CompanyBackupTest2 adlı örnek veritabanı oluşturulmuş ve test verileri eklenmiştir.
* Full (tam) yedekleme alınmış, ardından veri değişikliği yapılarak log yedeği alınmıştır.
* Felaket senaryosu doğrultusunda veritabanı silinmiş ve yedeklerden geri yükleme işlemi uygulanmıştır.
* Restore işlemi sonrası tüm verilerin eksiksiz şekilde geri geldiği doğrulanmıştır.
* Proje sürecinde karşılaşılan hatalar analiz edilmiş ve çözümleri uygulanarak işlem başarıyla tamamlanmıştır.

**Değerlendirme:**  
Bu çalışma, MSSQL’de yedekleme planlarının oluşturulması, log zincirlerinin takibi, ve veri bütünlüğünün korunması açısından oldukça faydalı ve öğretici olmuştur. Gerçek hayatta sıkça karşılaşılan senaryoların birebir simülasyonu sayesinde pratik tecrübe kazanılmıştır.