Ödev 1 - begin tran TL_F buraya isolation seviyesi belirtilebilir mi?

Ödev 2 - Bir transaction tanımlandığı anda mesela TL dedik orada isolation seviyesi tanımlanabilir mi? (Ödev 1-2 cevabı)
Cevap : Evet tanımlayabiliyoruz. MSSQL default levelı read comitted'dır. Eğer değiştirmek istersek SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED ya da SET TRANSACTION ISOLATION LEVEL Serializable gibi yazıp değiştirebiliriz.

Özet olarak
SET GLOBAL TRANSACTION transaction_name syntaxı ile tüm db transaction isolation levelını değiştiririz. SET SESSION TRANSACTION transaction_name ile sadece o oturumda değiştiririz. SET TRANSACTION transaction_name ile next transactiondaki levelı değiştiririz sadece.

Ödev 3 - Tüm isolation levelları localde nasıl değiştiriyoruz?
Cevap : SET GLOBAL TRANSACTION transaction_name syntaxı ile.

DBMS transactionlar için ACID özelliklerini sağlamak durumundadır.
Böylelikle istenen veri bütünlüğü sağlanmış olacaktır.

Ödev 4 - Isolation levelların handikapları nelerdir?
Cevap : Dirty read kullanılırsa yanlış veri alma riski var. Read Uncommitted levelı dirty read kullanıyor. En yavaş ama en güvenlikli seviye seriablizable.

Ödev 5 – ACID nedir?

ACID
Atomicity : Atomiklik
Consistency : Tutarlılık
Isolation : İzolasyon
Durability : Dayanıklılık

Atomicity -> Veritabanı işlemlerinde okuma/yazma işlemleri gibi bir bütün olarak gerçekleşmesi esastır. İşlemin bir adımında başarısız olma mümkündür, mesela elektrik kesintisi, bunun gibi başarısızlıklar olursa veritabanı yönetim sistemleri rollback işlemi gerçekleştirir.
rollback -> Değişiklikler otomatik geri sarılır.
Bu nedenle transactionlar mantıksal açıdan bölünemez yani atomiktir. commit, rollback transaction işlemini bitiren ifadelerdir ya da veritabanı kapanması transacition olayını sonlandırır.

Consistency-> begin try ile commit tran arasındaki kodalr bir bbütündür yani transaction başlangıç

ve comit bölümü aynı anda çalışır begin tran ve comit bölmesi, eğer sıra sıra kod çalışsaydı,
işlemler uzun sürerdi mesela atmden 30 dk da para çekilebilirdi.

Isolation -> Aynı anda meydana gelen işlemlerde bu işlemler birbirlerinin sonuçlarını etkilememelidir. Yani iki farklı transaction kesinlikle birbiriyle çakışmamalıdır. Her transaction işlemi veritabanının bütünlüğünü kendi içerisinde korumak zorundadır. Zaten belirttiğimiz ID olayı bunu sağlar. Sonuç olarak bir veritabanı transaction'ı
COMMIT'lenmeden bu işleme ait sonuç diğer kullanıcılar tarafından görülmez. Bu hesaplar arasındaki izolasyonu sağlar.

Isolations Levels

1) **Read Uncommitted :** En düşük isolasyon seviyesidir. Dirty read durumuna izin verir. 2 farklı transaction için transaction'ların herhangi birinde işlem commit edilmemiş data'yı gösterir. Transaction hata alıp rollback olsa dahil yanlış data'yı gösterilir. Yanlış veri çekme olasılığı olan seviyedir.

2) **Read Committed:** Mssql'in varsayılan varsayılan isolasyon seviyesidir.Phantom ve non-repetable read durumlarına uygundur izin vardır. Fakat dirty read durumuna izin yoktur ve dirty read olursa, timeout expired hatası alınır. Expired hatası verir. Özetlemek gerekirse bir transaction içerisinde yapılan insert-update işlemlerinin başka bir transaction işleminde gözükebilmesi için işlemin commit veya rollback yapılması gerekir.

3) **Repetable Read:** Sadece phantom-read durumuna izin verir.Diğer durumların oluşması durumunda timeout expired hatası alınır.

4) **Serializable:** Herhangi bir locking işleminin oluşmasına izin vermez. Özetlemek gerekirse eş zamanlı çalışan 2 transaction işlemi için insert-update işlemine izin vermez.Sadece select işlemine izin verir. Serializable en yavaş seviyedir, sıraya alıyo bekletiyo iş bittiğindee dataların tamamını gösterir en güvenli seviyedir.

5) **Snapshot :**

Özet -> Read Uncomitted dirty read kullanır. Read Uncomitted phantom ve non-repotable read'e izin verir. Repatable read sadece phantom read.

İzalasyon seviyesi değişmiş bir db çalışmışken ne olduğunu bunlar bilerek anlaşılabilir.

Mesela product tablosunu select ile çektiğimizde hiç birşey gözükmezse, o zaman isolation level 'ı db nin serializable'dır muhtemelen. Ya da read comitted. Yüksek olasalık serializable'dır.

DURABILITY -> x zamanda işletilen kodun cevabı aynı sonucu vermeli hep. Başarıyla uygulanmış bir transaction işlemi (COMMIT ile uygulanmış) geri alınamaz. Yani işleme ait tüm adımlar başarıyla uygulanmışsa ve kullanıcı da bundan haberdar edilmişse sonuç diskteki problemlere rağmen kalıcı olmalıdır. Diskte bir problem olsa bile bu durabilty özelliğini etkilememelidir.

READS

Dirty Read : Bir işlemin henüz işlenmemiş bir veriyi okuduğu durumdur. Örnek olarak elimizde T1 ve T2 diye iki işlemimiz olsun. T1 herhangi bir satırı güncelliyor fakat değişikliği onaylamadan işini bırakıyor. Bu arada T2 bu güncellenmiş veriyi okuyabiliyor. Eğer T1 bu işlemi geri alırsa T2 var olmamış veriyi okumaya devam edecektir.

Phantom Read : Eş zamanlı çalışan 2 transaction var mesela, 1. T tablodaki verileri okusun 2. T tabloya veri eklesin ve onaylasın (commit), 1. T okurken 2. T ekledigi ve commitledigi verileri hayalet olarak görür. Çünkü T1 işlemi sonladırmadan commit etmeden yeni okudugu tablonun yeni versiyonunu elde etmiş olur. Yeni eklenen satırlar onun için bir anda ortaya çıkan hayalet satırlar gibidir.

Non-Repeatable Reads : Yine iki eş zamanlı çalışan Transaction olsun. T'lerden biri yine veri çeksin. Özellikle çektiği belirli bir satır olabilir. Diğer Transaction' da bu belirli satırı veya başkalarını güncellesin ve işlemi onaylansın (Commit). Diğer T'nin işleminden sonra, diğer çalışmakta olan Transaction aynı satırlara yeniden baktığında verilerin yeni halini görür. İşte bu satırlardaki ani ve bilinmeyen değişiklikler nedeni ile, bu durum Non-Repeatable Read 'dir. Diğer T tarafından güncellenen ya da eklenen veriler phantomdaki gibi hayalet olarak gözükmez.

Phantom Read yapılıyorsa bir işlem var demektir ya da dirty read.