**SQL Server ile ilgili (pek) bilinmeyenler**

1) **sp\_help** bir sistem stored procedure’ü olup nesneler hakkında ayrıntılı bilgi almak için kullanılır.

2) **Değer girilmemiş (NULL) olan ifadeler**, gruplamalı fonksiyonlar için bir istisna olarak ele alınır. Örneğin, SELECT MIN (price) FROM tblUrun sorgusu, en küçük ifade, fiyatı girilmemiş olan ürün gibi düşünmemize rağmen, NULL değeri döndürmez, girilmiş olan en küçük değeri döndürür.

3) **İki tablo birbiri ile birleştirilerek sorgulanıyorsa (join)** bu iki tabloyu farklı disklerde saklamak, seçme yapmak için geçen süreyi kısaltır.

4) **Bir veritabanı tarafından kullanılan bir dosya**, başka bir veritabanı tarafından daha kullanılamaz.

5) **Veri ve transaction log bilgileri** asla ortak bir dosyada yer alamaz.

6) **Sp\_spaceused sistem procedure’ü** kullanarak veritabanındaki dosyaların doluluk oranları gibi özet bilgilere erişilebilir.

7) **Transaction log dosyasının boyutu***,*Indeks ile ilgili değişiklikler yapıldığında ya da WRITETEXT,UPDATETEXT ile metin veya resim yazılırken, WITH LOG parametresi ile kullanılırsa artar.

8) **Bir veritabanının boyutunu azaltma** işini otomatik olarak SQL SERVER’a yaptırabiliriz (AUTO\_SHRINK özelliği ile)

|  |  |
| --- | --- |
| **1**  **2** | **ALTER DATABASE veritabani ismi**  **SET AUTO\_SHRINK ON** |

**9) Veri tabanını silmeden önce Master veri tabanının bir kopyasını almak, dah sonrası için yarar sağlayabilir.**

10) **IDENTITY fonksiyonu ile elde edilen değerin***,* her zaman tablo başına tekilliği garanti edilemez. Bu türden bir amaç için, Constraint ile ya da doğrudan unique index tanımlamak gerekir.

11) **IDENTITY fonksiyonu ile üretilen primary key için** üretimden sonra hata oluşursa bir sonraki seferde aynı sayı kullanılmaz. Bu nedenle, primary key’ler arasında açıklıklar kalabilir.Bu türden bir durumu toleransı olmayan durumlarda NEWID() fonksiyonu ile UNIQEIDENTIFIER türü ikilisinin kullanılması önerilir.

12) **Çok güncellenen veriler***X*ML olarak saklanmamalıdır.

13) **Bir tabloya, kayıt girildikten sonra sütun eklenecekse**, bir sütun NOT NULL ile tanımlanırken dikkatli olmak gerekir. Hesaplanmış sütun veya default değer tanımlanması olmadan bu işlem gerçeklenemez.

14) **Foreign key Constraint ile ilişkilendirilmiş iki tablodan***,*foreign key’in bulunduğu tablo üstünde bir kullanıcının değişiklik yapabilmesi için birincil tarafa en azından SELECT ve REFERENCES hakkında verilmesi gerekir*.*

15) **Foreign key ile sağlanan veri bütünlüğü***,*sadece bir veritabanı içerisinde tanımlanabilir. Farklı iki veritabanıdaki tablolar arasında veri bütünlüğü sağlayabilmek için Trigger kodlamak gerekir.

16) **Her zaman where koşuluna** sütunları kapsayan şartlar vermek zorunluluk değildir. Bir sorguda öylesine şartlarda verilebilir. (where 1=1 deyimi, arama yapmak üzere dinamik SQL oluşturulurken takip eden şartların sayısı bilinmediğinde ilerlemek için kullanılan bir yöntemdir.)

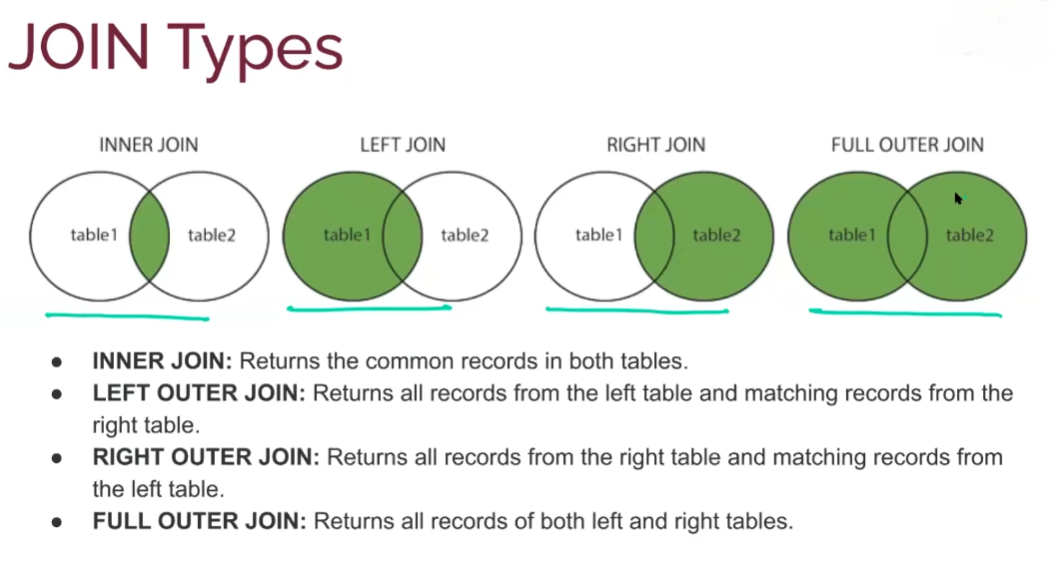
17) **Order By kullanılırken,** sıralamaya katılan sütunların seçilen sütunlardan olması gerekmez bunun aksine türetilmiş sütunlarda sıralamada kullanılabilir.

18) **Sütunlara verilen takma adlara** Order by tarafından erişilemez.

19) **Tarih alanları sorgularken** format problemi çekmemek için YYYY-AA-GG SS:DD:ss şeklinde string olarak istediğiniz tarihi vererek tarih kıyaslaması yapabilirsiniz.

20) **SQL server’da bir sorgunun sonucu**türetilmiş tablo olarak kullanılacaksa (sub query) hesaplanmış bütün sütunlara bir takma ad verilmelidir.

21) **Join ifadeleri***,*sorgu veya tablo sonuçlarını yatay olarak birleştirmek maksatlı kullanılır. Düşey birleştirme için UNION deyimi kullanılır.



22) **INNER ifadesi yer almadan** sadece JOIN ifadesi kullanılırsa, SQL Server bunu Inner Join olarak yorumlar.

23) **Cross join,** birinci tabloda yer alan her bir kaydı ikinci tabloda yer alan her bir kayıt ile ilişkilendirerek satırlar türetmede kullanılır.

24) **Union ifadesi ile birleştirilen iki resultset’ı** aynı sıradaki sütunların adları farklı ise, geçerli sütun adı, ilk resultset’teki sütun adıdır.

25) **Alt sonuçların varlığını kontrol etmeye yarayan**IN,NOT IN,EXIST ve NOT EXIST yardımı ile de tek sütunluk sorgular kesişim ve fark işlemine tabi tutulabilir.Ancak birden fazla sütun olması durumunda bu deyimler bir işe yaramayacaktır.Bu tür durumlarda INTERSECT ve EXCEPT deyimlerinin kullanılması gerekir.

26) **Bir sorguda, gruplamaya girmeden önce**elenecek satırları WHERE cümleciği ile tanımlamak gerekir. Gruplama neticesinde türetilen satırlar üstünde filtreleme yapılacaksa, bu filtreleme işlemi için HAVİNG kullanılır.

27) **Bir View’i silip yeniden oluşturduğumuzda***,*üstündeki izin ve engellemeleri yeniden ayarlamak gerekir. Ancak bir View Alter komutu ile değiştirilirse, hakların yeniden verilmesine gerek yoktur.

28) **Non-Clustered indekslerin performansları**, Clustered indekslerin performanslarından daha düşüktür.

29) **Ondalıklı değişken değerler için kullanılabilecek değişken tiplerinden** MONEY, DECIMAL gibi değişken tiplerinin tercih edilmesi önerilir, FLOAT değişken tipi kesin değer döndermediği için kullanılması pek tercih edilmez.

30) **Unicode değişken tipleri**, gereksiz yer tutması sebebiyle çok gerekmediği durumlarda kullanılması pek tercih edilmez (nvarchar, nchar gibi).

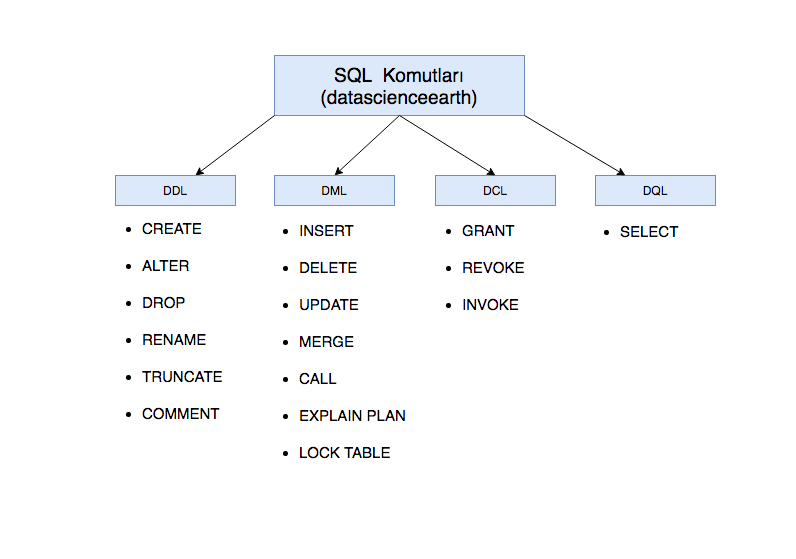
31) **TEXT değişken tipi** 2gb’a kadar yazıları tutabilir, adres değerini tuttuğu için Update, Insert işlemleri yaptığımız zaman işlem yapmaz, onun için ayrı bir kod yazmak gerekir.

32) **Data Defination Language (DDL)** – Veri Tanımlama Dili: CREATE DROP ve ALTER komutlarından oluşan DDL, veritabanı nesnelerini oluşturma, değiştirme ve silme işlemleri için kullanılır. Yani objectler ile yaptığımız işlemlerdir.

33) **Data Manipulation Language (DML)** – Veri İşleme Dili : INSERT, DELETE, UPDATE gibi komutlarla oluşturulan ifadeler, kayıt ekleme, silme ve güncelleme işlemleri için kullanılır. Yani Data ile ilgili yaptığımız işlemlerdir.

34) **Data Control Language (DCL)** – Veri Kontrol Dili : GRANT, INVOKE, REVOKE komutları izin ve yetkilendirme işleri için kullanılır.

35) **Data Query Language (DQL)** – Veri Sorgulama Dili : SELECT komutu ile oluşturulan ifadeler, verilerin listelenmesi için kullanılır.



36) **Yarattığımız Stored Procedure’ün değerini değiştirmek istediğimizde**artık ALTER TABLE etmeliyiz, aksi taktirde Create ifadesi ile zaten oluşturduğumuz stored procedure’ü tekrar oluşturmaya çalıştığımız için hata alırız.

**[](https://i0.wp.com/koddit.com/wp-content/uploads/2014/08/uniqure.jpg)**

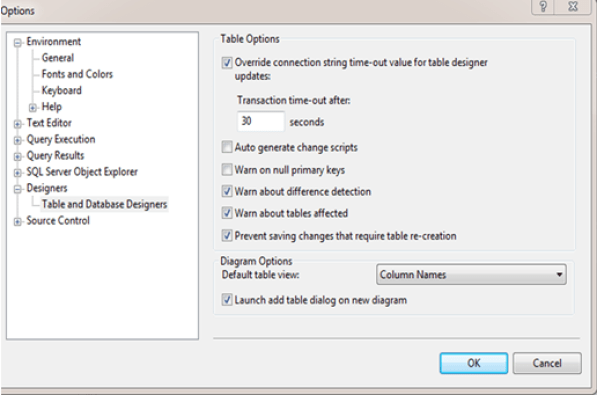
37) **Tablo üzerinde yapılacak her türlü fiziksel değişiklik**ALTER TABLE ile yapılır.

38) **SELECT TOP 0 \* INTO** komutu tablonun direkt kendisini yaratır; ama keyleri kaybederek yaratır.

39) **Veri tabanında oluşturduğumuz bir tabloda üzerinde herhangi bir değişiklik yapıp** sonra da kaydetmek istediğimizde, sonradan yapacağımız değişikliklere izin verilmez.

Bu durumu aşmak için,

**Tools** > **Options** > **Prevent saving changes that require table re-creation** checkbox’ında seçili olarak gelen işareti kaldırıp save edebiliriz.

**[](https://i0.wp.com/koddit.com/wp-content/uploads/2014/08/1.png)**

40) **Bir SP’nin bittiği yerde go deyimini kullanmak,**sp’nizin bittiği yeri tam olarak belirtmenizi sağlayacağından istemediğiniz kodların sp ile birlikte derlenmesine ve istenmeyen hatalar meydana getirmesine engel olur.