Veri Tabanı Yönetim Sistemleri

SQL Server IDENTITY sütun örneği

- IDENTITY Bir tablo için kimlik sütunu oluşturmak için özelliği aşağıdaki gibi kullanırsınız :
- IDENTITY[(seed,increment)]
- Bu sözdiziminde:
  - Seed Tabloya yüklenen ilk satırın değeridir.
  - Bu, increment bir önceki satırın kimlik değerine eklenen artımlı değerdir.
- Seed ve'nin varsayılan değeri increment 1'dir, yani, (1,1). Bu, ilk satırın 1 değerine, ikinci satırın 2 değerine sahip olacağı ve bu şekilde devam edeceği anlamına gelir.
- İlk satırın kimlik sütununun değerinin 10 ve artımlı değerin 10 olmasını istiyorsanız, aşağıdaki sözdizimini kullanabilirsiniz:
  - IDENTITY (10,10)
- SQL Server'da her tablonun *yalnızca bir* kimlik sütunu vardır. Genellikle bu, tablonun birincil anahtar sütunudur .

SQL Server IDENTITY sütun örneği

```
CREATE SCHEMA hr;
```

```
CREATE TABLE hr.person (
    person_id INT IDENTITY(1,1) PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    gender CHAR(1) NOT NULL
);
```

```
INSERT INTO hr.person(first_name, last_name, gender)
OUTPUT inserted.person_id
VALUES('John','Doe', 'M');
```

person_id
1

SQL Server IDENTITY sütun örneği

İkinci olarak tabloya bir satır daha ekleyelim person:

```
INSERT INTO hr.person(first_name, last_name, gender)
OUTPUT inserted.person_id
VALUES('Jane','Doe','F');
```

Çıktı:

person_id
2

Kimlik sütunu değerinin sıfırlanması

- Kimlik sayacını sıfırlamak için DBCC CHECKIDENT yönetim komutunu kullanabilirsiniz:

```
DBCC CHECKIDENT ('[TableName]', RESEED, 0);
GO
```

Öncelikle tablodaki tüm satırları silelim `hr.person` :

```
DELETE FROM hr.person;
```

İkinci olarak kimlik sayacını sıfırlayalım:

```
DBCC CHECKIDENT ('hr.person', RESEED, 0);
GO
```

Kimlik sütunu değerinin sıfırlanması

Çıktı:

```
Checking identity information: current identity value '4'.
DBCC execution completed. If DBCC printed error messages, contact
```

Çıktı, mevcut kimlik değerinin 4 olduğunu gösterir. Değeri sıfırlar.

Üçüncüsü, tabloya yeni bir satır ekleyin `hr.person` :

```
INSERT INTO hr.person(first_name, last_name, gender)
OUTPUT inserted.person_id
VALUES('Jhoan','Smith','F');
```

Çıktı:

```
person_id
-----------
1

(1 row affected)
```

SQL Server ALTER TABLE ADD

Aşağıdaki ALTER TABLE ADD ifade tabloya yeni bir sütun ekler:

```
ALTER TABLE table_name
ADD column_name data_type column_constraint;
```

- Bu açıklamada:
  - Öncelikle yeni sütunu eklemek istediğiniz tablonun adını belirtin.
  - İkinci olarak, sütunun adını, veri türünü ve varsa kısıtlamayı belirtin.
- Tek bir ifadeyi kullanarak bir tabloya aynı anda birden fazla sütun eklemek istiyorsanız ALTER TABLE, aşağıdaki sözdizimini kullanırsınız:

SQL Server ALTER TABLE ADD

```
ALTER TABLE table_name
ADD
    column_name_1 data_type_1 column_constraint_1,
    column_name_2 data_type_2 column_constraint_2,
    ...,
    column_name_n data_type_n column_constraint_n;
```

SQL Server ALTER TABLE ADD sütun örnekleri

```
CREATE TABLE sales.quotations (
    quotation_no INT IDENTITY PRIMARY KEY,
    valid_from DATE NOT NULL,
    valid_to DATE NOT NULL
);
```

`description` Tabloya adlı yeni bir sütun eklemek için `sales.quotations` aşağıdaki ifadeyi kullanırsınız:

```
ALTER TABLE sales.quotations
ADD description VARCHAR (255) NOT NULL;
```

SQL Server ALTER TABLE ADD sütun örnekleri

Aşağıdaki ifade tabloya ve adında iki yeni sütun `amount` ekler
: `customer_name` `sales.quotations`

```
ALTER TABLE sales.quotations
    ADD
        amount DECIMAL (10, 2) NOT NULL,
        customer_name VARCHAR (50) NOT NULL;
```

SQL Server ALTER TABLE ALTER COLUMN

- SQL Server, bir tablonun mevcut bir sütununda aşağıdaki değişiklikleri yapmanıza olanak tanır:
  - Veri türünü değiştirin
  - Boyutu değiştir
  - NOT NULL
- Sütunun veri türünü değiştir.

```
ALTER TABLE table_name
ALTER COLUMN column_name new_data_type(size);
```

- Yeni veri tipinin eskisi ile uyumlu olması gerekir, aksi takdirde sütunda veri olması durumunda dönüştürme işlemi başarısız olursa dönüştürme hatası alırsınız.
- Öncelikle veri türü tek sütunlu yeni bir tablo oluşturun INT :

```
CREATE TABLE t1 (c INT);
```

SQL Server ALTER TABLE ALTER COLUMN

İkinci olarak tabloya birkaç satır ekleyelim :

```
INSERT INTO t1
VALUES
    (1),
    (2),
    (3);
```

İkinci olarak, sütunun veri türünü `INT` şu şekilde değiştirin `VARCHAR` :

```
ALTER TABLE t1 ALTER COLUMN c VARCHAR (2);
```

SQL Server ALTER TABLE ALTER COLUMN

Üçüncüsü, karakter dizisi verisi içeren yeni bir satır ekleyin:

```
INSERT INTO t1
VALUES ('@');
```

Dördüncüsü, sütunun veri türünü şu şekilde `VARCHAR` değiştirin `INT` :

```
ALTER TABLE t1 ALTER COLUMN c INT;
```

SQL Server aşağıdaki hatayı verdi:

```
Conversion failed when converting the varchar value '@' to data type int.
```

Bir sütunun boyutunu değiştirmek

Aşağıdaki ifade, veri türü şu olan bir sütuna sahip yeni bir tablo oluşturulur VARCHAR(10):

```
CREATE TABLE t2 (c VARCHAR(10));
```

- t2 tablosuna bazı örnek veriler ekleyelim:

```
INSERT INTO t2
VALUES
    ('SQL Server'),
    ('Modify'),
    ('Column')
```

- Sütunun boyutunu aşağıdaki şekilde artırabilirsiniz:

```
ALTER TABLE t2 ALTER COLUMN c VARCHAR (50);
```

Bir sütunun boyutunu değiştirmek

- Ancak, sütunun boyutunu azalttığınızda, SQL Server yeni boyuta göre verileri dönüştürüp dönüştüremeyeceğini görmek için mevcut verileri kontrol eder. Dönüştürme başarısız olursa, SQL Server ifadeyi sonlandırır ve bir hata mesajı verir.
- Örneğin, sütunun boyutunu 5 karaktere düşürürseniz :

```
ALTER TABLE t2 ALTER COLUMN c VARCHAR (5);
```

SQL Server aşağıdaki hatayı verdi:

```
String or binary data would be truncated.
```

Boş bırakılabilir bir sütuna NOT NULL kısıtlaması eklemek

- Aşağıdaki ifade, null olabilen bir sütuna sahip
```
CREATE TABLE t3 (c VARCHAR(50));
```
ur:

- Aşa
```
INSERT INTO t3
VALUES
    ('Nullable column'),
    (NULL);
```
ya bazı satırlar ekler:

-
```
UPDATE t3
SET c = ''
WHERE
    c IS NULL;
```
IOT NULL kısıtlamasını eklemek
öncelikle NULL'u null olmayan bir
ellemelisiniz; örneğin:

Boş bırakılabilir bir sütuna NOT NULL kısıtlaması eklemek

- NOT NULL kısıtlamayı ekleyin:

```
ALTER TABLE t3 ALTER COLUMN c VARCHAR (20) NOT NULL;
```

SQL Server ALTER TABLE DROP COLUMN

- Bazen, bir tablodan kullanılmayan veya eski bir veya daha fazla sütunu kaldırmanız gerekir. Bunu yapmak için, ALTER TABLE DROP COLUMN ifadeyi aşağıdaki gibi kullanırsınız:

```
ALTER TABLE table_name
DROP COLUMN column_name;
```

- Bu sözdiziminde:
  - Öncelikle sütunu silmek istediğiniz tablonun adını belirtin.
  - İkinci olarak silmek istediğiniz sütunun adını belirtin.
- Silmek istediğiniz sütunda bir CHECK kısıtlama varsa, sütunu kaldırmadan önce kısıtlamayı silmeniz gerekir. Ayrıca, SQL Server, PRIMARY KEY veya FOREIGN KEY kısıtlaması olan bir sütunu silmenize izin vermez.
- Birden fazla sütunu aynı anda silmek istiyorsanız, aşağıdaki sözdizimini kullanırsınız:

```
ALTER TABLE table_name
DROP COLUMN column_name_1, column_name_2,...;
```

SQL Server ALTER TABLE DROP COLUMN örnekleri

- Yeni bir tablo oluşturalım .sales.price_lists

```
CREATE TABLE sales.price_lists(
    product_id int,
    valid_from DATE,
    price DEC(10,2) NOT NULL CONSTRAINT ck_positive_price CHECK(price >= 0
    discount DEC(10,2) NOT NULL,
    surcharge DEC(10,2) NOT NULL,
    note VARCHAR(255),
    PRIMARY KEY(product_id, valid_from)
);
```

SQL Server ALTER TABLE DROP COLUMN örnekleri

- Aşağıdaki ifade note sütunu tamamen siler. price_lists:

```
ALTER TABLE sales.price_lists
DROP COLUMN note;
```

- Fiyat sütununda bir CHECK kısıtlama var, bu nedenle onu silemezsiniz. Aşağıdaki ifadeyi yürütmeyi denerseniz bir hata alırsınız:

```
ALTER TABLE sales.price_lists
DROP COLUMN price;
```

- Hata mesajı:

```
The object 'ck_positive_price' is dependent on column 'price'.
```

- Fiyat sütunu silmek için öncelikle CHECK sabitini silin:

```
ALTER TABLE sales.price_lists
DROP CONSTRAINT ck_positive_price;
```

SQL Server ALTER TABLE DROP COLUMN örnekleri

- Price sütunu silin:

```
ALTER TABLE sales.price_lists
DROP COLUMN price;
```

- Aşağıdaki örnek, indirim ve ek ücret olmak üzere iki sütunu aynı anda siler:

```
ALTER TABLE sales.price_lists
DROP COLUMN discount, surcharge;
```

SQL Server Hesaplanan Sütunlar

```
CREATE TABLE persons
(
    person_id  INT PRIMARY KEY IDENTITY,
    first_name NVARCHAR(100) NOT NULL,
    last_name  NVARCHAR(100) NOT NULL,
    dob        DATE
);
```

SQL Server ALTER TABLE DROP COLUMN örnekleri

- Ve tabloya iki satır ekleyelim persons :

```
INSERT INTO
    persons(first_name, last_name, dob)
VALUES
    ('John','Doe','1990-05-01'),
    ('Jane','Doe','1995-03-01');
```

SQL Server ALTER TABLE DROP COLUMN örnekleri

- Persons tablosundaki kişilerin tam adlarını sorgulamak için CONCAT() fonksiyonunu veya + operatörünü aşağıdaki gibi kullanırsınız:

```
SELECT
    person_id,
    first_name + ' ' + last_name AS full_name,
    dob
FROM
    persons
ORDER BY
    full_name;
```

person_id	full_name	dob
2	Jane Doe	1995-03-01
1	John Doe	1990-05-01

SQL Server ALTER TABLE DROP COLUMN örnekleri

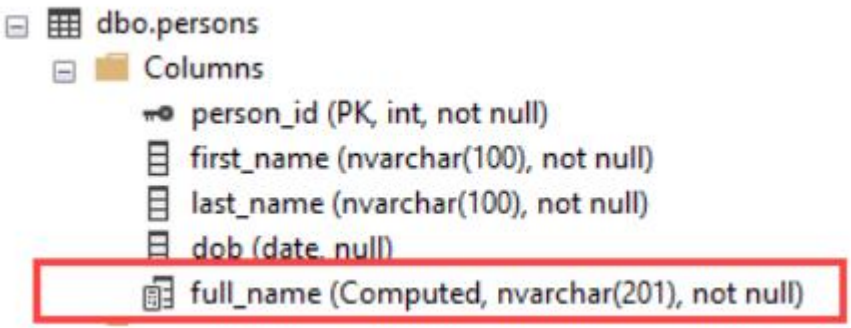
- Aşağıdaki şekilde kullanarak tablova full_name sütun eklevebilirsiniz :persons ALTER TA

```
ALTER TABLE persons
ADD full_name AS (first_name + ' ' + last_name);
```

- Persons tablosundan her veri sorguladığınızda, SQL Server, first_name + ' ' + last_name ifade ...unun değerini hesaplar ve sonucu

```
SELECT
    person_id,
    full_name,
    dob
FROM
    persons
ORDER BY
    full_name;
```

SQL Server ALTER TABLE DROP COLUMN örnekleri



- **Yeni bir tablo oluştururken hesaplanan sütunları tanımlama sözdizimi**

```
CREATE TABLE table_name(
    ...,
    column_name AS expression [PERSISTED],
    ...
);
```

SQL Server Geçici (Temporary) Tablolar

- Geçici tablolar, SQL Server'da geçici olarak bulunan tablolardır.

SELECT INTO ifadesini kullanarak geçici tablolar oluşturma

```
SELECT
    select_list
INTO
    temporary_table
FROM
    table_name
....
```

- Geçici tablonun adı bir karma simgesiyle (#) başlar.

```sql
SELECT
    product_name,
    list_price
INTO #trek_products --- temporary table
FROM
    production.products
WHERE
    brand_id = 9;
```

CREATE TABLE ifadesini kullanarak geçici tablolar oluşturma

- Bu ifadenin söz dizimi, normal bir tablo oluşturma ile aynıdır. Ancak, geçici tablonun adı # simgesiyle başlar

```
CREATE TABLE #haro_products (
    product_name VARCHAR(MAX),
    list_price DEC(10,2)
);
```

- Geçici tabloyu oluşturduktan sonra, bu tabloya normal bir tablo gibi veri ekleyebilirsiniz:

```
INSERT INTO #haro_products
SELECT
    product_name,
    list_price
FROM
    production.products
WHERE
    brand_id = 2;
```

- Elbette, mevcut oturum içerisinde buna karşı veri sorgusu yapabilirsiniz:

```
SELECT
    *
FROM
    #haro_products;
```

product_name	list_price
Haro Flightline One ST - 2017	379.99
Haro Flightline Two 26 Plus - 2017	549.99
Haro Shift R3 - 2017	1469.99
Haro SR 1.1 - 2017	539.99
Haro SR 1.2 - 2017	869.99
Haro SR 1.3 - 2017	1409.99
Haro Downtown 16 - 2017	329.99
Haro Shredder 20 - 2017	209.99
Haro Shredder 20 Girls - 2017	209.99
Haro Shredder Pro 20 - 2017	249.99

- Ancak başka bir bağlantı açıp yukarıdaki sorguyu denerseniz aşağıdaki hatayı alırsınız:

```
Invalid object name '#haro_products'.
```

- Bunun nedeni, geçici tablolara yalnızca onları oluşturan oturum içerisinden erişilebilmesidir.

Global geçici tablolar

- Bazen, bağlantılar arasında erişilebilen geçici bir tablo oluşturmak için global geçici tablolar kullanılabilir.
- Global geçici tablonun adı ## simgesiyle başlar.

```
CREATE TABLE ##heller_products (
    product_name VARCHAR(MAX),
    list_price DEC(10,2)
);

INSERT INTO ##heller_products
SELECT
    product_name,
    list_price
FROM
    production.products
WHERE
    brand_id = 3;
```

Geçici tabloları silme

Otomatik silme

- SQL Server, bağlantıyı kapattığınızda geçici bir tabloyu otomatik olarak siler.
- SQL Server, onu oluşturan bağlantı kapandığında ve diğer bağlantılar üzerinden bu tabloya yönelik sorgular tamamlandığında genel geçici tabloyu siliyor.

Manuel silme

```
DROP TABLE ##table_name;
```

SQL Server Eş Anlamlısı (Synonym)

- SQL Server'da eşanlamlı (synonym), tablo, view, stored procedure, kullanıcı tanımlı fonksiyon ve sıra gibi bir veri tabanı nesnesi için bir takma ad veya alternatif addır. Eşanlamlı (synonym), düzgün kullanırsanız size birçok avantaj sağlar.

- Eş anlamlı bir sözcük oluşturmak için şu ifadeyi kullanabilirsiniz:

```
CREATE SYNONYM [ schema_name_1. ] synonym_name
FOR object;
```

- Nesne şu biçimdedir:

```
[ server_name.[ database_name ] . [ schema_name_2 ]. object_name
```

SQL Server CREATE SYNONYM ifadesi örnekleri

A) Aynı veritabanı örneği içerisinde eşanlamlı oluşturma

- Aşağıdaki örnek, sales.orders tablosu için bir eşanlamlı oluşturmak üzere CREATE SYNONYM ifadesini kullanır:

```
CREATE SYNONYM orders FOR sales.orders;
```

- Örneğin, aşağıdaki sorgu sales.orders tablosu yerine orders eşanlamlısını kullanır:

```
SELECT * FROM orders;
```

B) Başka bir veritabanındaki bir tablo için eşanlamlı oluşturma

- Öncelikle test adında yeni bir veritabanı oluşturun ve mevcut veritabanını test olarak ayarlayın:

```
CREATE DATABASE test;
GO

USE test;
GO
```

- Ardından test veritabanının içinde purchasing adında yeni bir şema oluşturun:

```
CREATE SCHEMA purchasing;
GO
```

- Ardından, test veritabanının purchasing şemasında yeni bir tablo oluşturun:

```
CREATE TABLE purchasing.suppliers
(
    supplier_id   INT
    PRIMARY KEY IDENTITY,
    supplier_name NVARCHAR(100) NOT NULL
);
```

- Ardından, BikeStores veritabanından, test veritabanındaki purchasing.suppliers tablosu için bir eşanlamlı oluşturun:

```
CREATE SYNONYM suppliers
FOR test.purchasing.suppliers;
```

- Son olarak, BikeStores veritabanından, suppliers eşanlamlısını kullanarak test.purchasing.suppliers tablosuna başvurun:

```
SELECT * FROM suppliers;
```

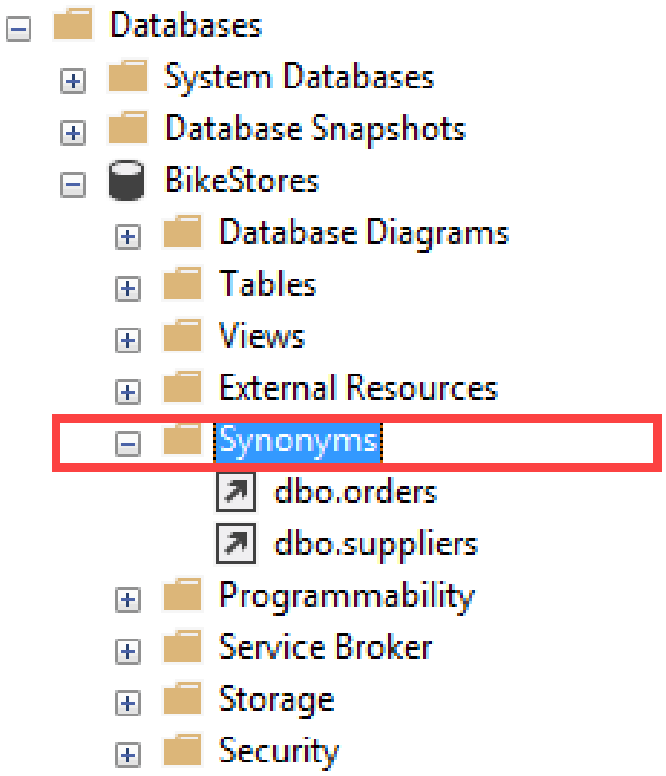
Bir veritabanının tüm eş anlamlılarını listeleme

A) Transact-SQL komutunu kullanarak eş anlamlıları listeleme

```
SELECT
    name,
    base_object_name,
    type
FROM
    sys.synonyms
ORDER BY
    name;
```

name	base_object_name	type
orders	[sales].[orders]	SN
suppliers	[test].[purchasing].[suppliers]	SN

B) SQL Server Management Studio'yu kullanarak eş anlamlıları listeleme



Bir eşanlamlıyı silme

- Bir eşanlamlıyı silmek için, aşağıdaki sözdizimi kullanılır:

```
DROP SYNONYM [ IF EXISTS ] [schema.] synonym_name
```

- Aşağıdaki örnekte, sipariş eşanlamlısını silmek için DROP SYNONYM ifadesi kullanılmıştır:

```
DROP SYNONYM IF EXISTS orders;
```

Eş anlamlılar ne zaman kullanılır?

- Nesne adlarını basitleştirmek.
- Sorunsuz nesne adı değişikliklerini etkinleştirmek.

Eş anlamlıların faydaları

- Temel nesnelerin üzerine bir soyutlama katmanı sağlamak.
- Uzun adı kısaltmak, basitleştirilmiş bir takma adla değiştirmek.
- Tablolar, view, stored procedure, kullanıcı tanımlı işlevler ve diziler gibi veritabanı nesnelerini yeniden adlandırdığınızda mevcut uygulamalar için geriye dönük uyumluluğa izin vermek.