

**T.C.**  
**ONDOKUZ MAYIS ÜNİVERSİTESİ**  
**Bilgisayar Mühendisliği Bölümü**  
**Veri Tabanı Yönetim Sistemleri Laboratuvarı**



**2024 – 2025 BAHAR DÖNEMİ**

**FÖY-4 RAPORU**

**HAZIRLAYAN:**  
**21060684 - AYŞEGÜL AKMAN**

**SAMSUN-2025**

## ÖZET

Bu raporda MS SQL Server üzerinde bir veri tabanı oluşturarak bu veri tabanı üzerinde kısıtlayıcılar tanımlama ve silme; sorguları basitleştirme, sorgu süresini kısaltma için view yani sanal tablo kullanma, bu viewları tanımlama silme ve güncelleme; indeksler ile tablolalara eklemiş olduğumuz verileri daha performanslı bir şekilde aramayı öğreneceğiz.

## **İÇİNDEKİLER**

**1. GİRİŞ**

**2. SORU 1**

**3. SORU 2**

**4. SORU 3**

**5. SORU 4**

**6. SORU 5**

**7. SORU 6**

**8. SORU 7**

**9. SORU 8**

# 1. GİRİŞ

SQL Server’da SQL komutları kullanılarak föy içerisinde verilen diyagramlara uygun bir veri tabanı oluşturuldu. Veri tabanı oluşturulurken aynı zamanda uygun SQL komutları ile tablolara veriler eklendi. Tablolar oluşturulurken istenilen özellikler göz önüne alındı.

```
CREATE DATABASE Foy4;
GO

USE Foy4;
GO

--CREATE TABLE client_master (
    Client_no NVARCHAR(6) NOT NULL PRIMARY KEY,
    Name NVARCHAR(20),
    Address1 NVARCHAR(30),
    Address2 NVARCHAR(30),
    City NVARCHAR(15),
    State NVARCHAR(15),
    Pincode NVARCHAR(6),
    Bal_due DECIMAL(10, 2) CHECK (Bal_due >= 0)
);

INSERT INTO client_master VALUES ('0001', 'Ivan', NULL, NULL, 'Bombay', 'Maharashtra', 400054, 15000);
INSERT INTO client_master VALUES ('0002', 'Vandana', NULL, NULL, 'Madras', 'Tamilnadu', 780001, 0);
INSERT INTO client_master VALUES ('0003', 'Pranada', NULL, NULL, 'Bombay', 'Maharashtra', 400057, 5000);
INSERT INTO client_master VALUES ('0004', 'Basu', NULL, NULL, 'Bombay', 'Maharashtra', 400056, 0);
INSERT INTO client_master VALUES ('0005', 'Ravi', NULL, NULL, 'Delhi', NULL, 100001, 2000);
INSERT INTO client_master VALUES ('0006', 'Rukmini', NULL, NULL, 'Bombay', 'Maharashtra', 400050, 0);

--CREATE TABLE product_master (
    Product_no NVARCHAR(10) NOT NULL PRIMARY KEY,
    Description NVARCHAR(50),
    Profit_percent DECIMAL(5, 2),
    Unit_measure NVARCHAR(30),
    Qty_on_hand INT,
    Reorder_lvl INT,
    Sell_price DECIMAL(10, 2),
    Cost_price DECIMAL(10, 2)
);

INSERT INTO Product_master VALUES ('P00001', '1.44floppies', 5, 'piece', 100, 20, 525, 500);
INSERT INTO Product_master VALUES ('P03453', 'Monitors', 6, 'piece', 10, 3, 12000, 11200);
INSERT INTO Product_master VALUES ('P06734', 'Mouse', 5, 'piece', 20, 5, 1050, 500);
INSERT INTO Product_master VALUES ('P07865', '1.22 floppies', 5, 'piece', 100, 20, 525, 500);
INSERT INTO Product_master VALUES ('P07868', 'Keyboards', 2, 'piece', 10, 3, 3150, 3050);
INSERT INTO Product_master VALUES ('P07885', 'CD Drive', 2.5, 'piece', 10, 3, 5250, 5100);
INSERT INTO Product_master VALUES ('P07965', '540 HDD', 4, 'piece', 10, 3, 8400, 8000);
INSERT INTO Product_master VALUES ('P07975', '1.44 Drive', 5, 'piece', 10, 3, 1050, 1000);
INSERT INTO Product_master VALUES ('P08865', '1.22 Drive', 5, 'piece', 2, 3, 1050, 1000);

--CREATE TABLE salesman_master (
    Salesman_no NVARCHAR(6) NOT NULL PRIMARY KEY CHECK (Salesman_no LIKE 's%'),
    Sal_name NVARCHAR(20) NOT NULL,
    Address NVARCHAR(20) NOT NULL,
    City NVARCHAR(20),
    State NVARCHAR(20),
    Pincode NVARCHAR(6),
    Sal_amt DECIMAL(8,2) NOT NULL CHECK (Sal_amt > 0),
    Tgt_to_get DECIMAL(6,2) NOT NULL CHECK (Tgt_to_get > 0),
    Ytd_sales DECIMAL(6,2) NOT NULL CHECK (Ytd_sales > 0),
    Remarks NVARCHAR(30)
);

--INSERT INTO salesman_master VALUES
('S00001', 'Kiran', 'A/14 worli', 'Bombay', 'Mah', '400002', 3000, 100, 50, 'Good'),
('S00002', 'Manish', '65,narim', 'Bombay', 'Mah', '400003', 3000, 200, 100, 'Good'),
('S00003', 'Ravi', 'P-7 Bandra', 'Bombay', 'Mah', '400032', 3000, 200, 100, 'Good'),
('S00004', 'Ashish', 'A/5 Juhu', 'Bombay', 'Mah', '400044', 3500, 200, 150, 'Good');
```

```

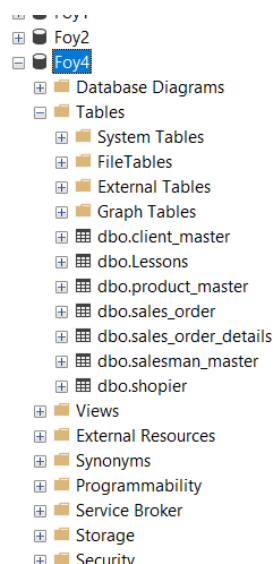
CREATE TABLE sales_order (
    S_order_no NVARCHAR(6) NOT NULL PRIMARY KEY,
    S_order_date DATE,
    Client_no NVARCHAR(25) NOT NULL,
    Dely_type CHAR(1) DEFAULT 'F' CHECK (Dely_type IN ('P', 'F')),
    Billed_yn CHAR(1) CHECK (Billed_yn IN ('Y', 'N')),
    Salesman_no NVARCHAR(6) NOT NULL,
    Dely_add NVARCHAR(6),
    Dely_date DATE,
    Order_status NVARCHAR(10) CHECK (Order_status IN ('in process', 'fulfilled', 'back order', 'canceled')),
    CONSTRAINT fk_salesman FOREIGN KEY (Salesman_no) REFERENCES salesman_master(Salesman_no),
    CONSTRAINT chk_delay CHECK (Dely_date >= S_order_date),
    CONSTRAINT chk_order_no CHECK (SUBSTRING(S_order_no, 1, 1) = '0')
);

INSERT INTO sales_order (S_order_no, S_order_date, Client_no, Dely_type, Billed_yn, Salesman_no, Dely_add, Dely_date, Order_status)
VALUES
('019001', '1996-01-12', '0001', 'F', 'N', 'S00001', NULL, '1996-01-20', 'in process'),
('019002', '1996-01-25', '0002', 'P', 'N', 'S00002', NULL, '1996-02-27', 'canceled'),
('016865', '1996-02-18', '0003', 'F', 'Y', 'S00003', NULL, '1996-02-20', 'fulfilled'),
('019003', '1996-04-03', '0001', 'F', 'Y', 'S00001', NULL, '1996-04-07', 'fulfilled'),
('046866', '1996-05-20', '0004', 'P', 'N', 'S00002', NULL, '1996-05-22', 'canceled'),
('010008', '1996-05-24', '0005', 'F', 'N', 'S00004', NULL, '1996-05-26', 'in process'),
('046865', '1996-01-12', '0001', 'F', 'N', 'S00001', NULL, '1996-01-20', 'in process');

CREATE TABLE sales_order_details (
    S_order_no NVARCHAR(6) NOT NULL,
    Product_no NVARCHAR(10) NOT NULL,
    Qty_ordered INT,
    Qty_disp INT,
    Product_rate DECIMAL(10,2),
    CONSTRAINT FK_Sales PRIMARY KEY (S_order_no, Product_no),
    FOREIGN KEY (S_order_no) REFERENCES sales_order(S_order_no),
    FOREIGN KEY (Product_no) REFERENCES product_master(Product_no)
);

INSERT INTO sales_order_details VALUES ('019001', 'P00001', 4, 4, 525);
INSERT INTO sales_order_details VALUES ('019001', 'P07965', 2, 1, 8400);
INSERT INTO sales_order_details VALUES ('019001', 'P07885', 2, 1, 5250);
INSERT INTO sales_order_details VALUES ('019002', 'P00001', 10, 0, 525);
INSERT INTO sales_order_details VALUES ('046865', 'P07868', 3, 3, 3150);
INSERT INTO sales_order_details VALUES ('046865', 'P07885', 10, 10, 5250);
INSERT INTO sales_order_details VALUES ('019003', 'P00001', 4, 4, 1050);
INSERT INTO sales_order_details VALUES ('019003', 'P03453', 2, 2, 1050);
INSERT INTO sales_order_details VALUES ('046866', 'P06734', 1, 1, 12000);
INSERT INTO sales_order_details VALUES ('046866', 'P07965', 1, 0, 8400);
INSERT INTO sales_order_details VALUES ('010008', 'P07975', 1, 0, 1050);
INSERT INTO sales_order_details VALUES ('010008', 'P00001', 10, 5, 525);

```



## 2. SORU 1

salesman\_master tablosu üzerinde Tgt\_to\_get değeri 200'den büyük olanlar için bir view oluşturuldu. View oluşturmak için "CREATE VIEW" komutu kullanıldı.

salesman\_master tablosundan gerekli veriler çekildi.

```
CREATE VIEW salesman_view  
AS SELECT * FROM salesman_master  
WHERE Tgt_to_get > 200;
```

## 3. SORU 2

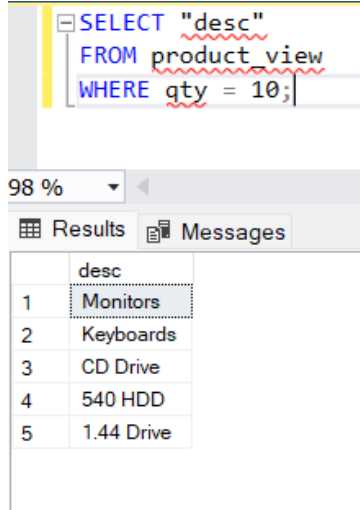
product\_master tablosu üzerinde product\_view isimli bir view oluşturarak sütun isimlerini sırasıyla pro\_no, desc, profit, Unit\_measure, qty olacak şekilde değiştireceğiz.

Verileri product\_master'dan çekiyoruz ve "AS" komutu ile tablo isimlerini istenildiği gibi değiştiriyoruz.

```
CREATE VIEW product_view  
AS SELECT product_no AS "pro_no",  
Description AS "desc",  
Profit_percent AS "profit",  
Unit_measure AS "Unit_measure",  
Qty_on_hand AS "qty"  
FROM product_master;
```

#### 4. SORU 3

product\_view isimli view'dan Qty\_on\_hand değeri '10' olan product isimlerini getiren sorguyu yazacağız. Bunun için “SELECT” komutuyla istenilen sütunu ve “FROM” komutuyla bu sütunun nereden çekileceğini belirtiyoruz. Sonrasında “WHERE” komutuyla bizden istenilen bilgiyi yani “qty=10” işlemini çağırıyoruz.



```
SELECT "desc"
FROM product_view
WHERE qty = 10;
```

	desc
1	Monitors
2	Keyboards
3	CD Drive
4	540 HDD
5	1.44 Drive

#### 5. SORU 4

Sipariş tarihi 10 gün geçen siparişleri müşteri isimleri ve ürün isimleri olarak listeleyen bir SQL sorgusu yazacağız. Burada kullanılacak olan tablolar “sales\_order, client\_master, product\_master ve sales\_order\_details” tablolarıdır.

“Client\_no” sütunu “client\_master” ve “sales\_order” tablolarını, “S\_order\_no” sütunu “sales\_order” ve “sales\_order\_details” tablolarını, “Product\_no” sütunu “sales\_order\_details” ve “product\_master” tablolarını birbirine bağlamaktadır.

“WHERE” komutuyla “S\_order\_date” sütununda yer alan sipariş tarihlerinden 10 gün gecikmiş olanları çekerek ekrana yazdırıyoruz.

“DATEADD” belli bir tarihe veya zamana ekleme veya çıkarma yapmak için kullanılan bir fonksiyondur. Burada biz “GETDATE()” ile günümüzü kastederek bu tarihten 10 gün çıkartıp işlem sonucunda elde ettiğimiz tarihten daha önceki tarihte kalan siparişleri ekrana yazdırıyoruz. Yani sipariş tarihleri üzerinden 10 gün ve daha fazla geçen siparişler ekrana yazdırılıyor.

```

SELECT c.Name, p.Description
FROM sales_order s
JOIN client_master c ON s.Client_no = c.Client_no
JOIN sales_order_details o ON s.S_order_no = o.S_order_no
JOIN product_master p ON o.Product_no = p.Product_no
WHERE s.S_order_date < DATEADD(DAY, -10, GETDATE());

```

98 %

Results Messages

	Name	Description
1	Ravi	1.44floppies
2	Ravi	1.44 Drive
3	Ivan	1.44floppies
4	Ivan	CD Drive
5	Ivan	540 HDD
6	Vandana	1.44floppies
7	Ivan	1.44floppies
8	Ivan	Monitors
9	Ivan	Keyboards
10	Ivan	CD Drive
11	Basu	Mouse
12	Basu	540 HDD

## 6. SORU 5

sales\_order tablosunu kullanarak günlük siparişleri listelemeye yarayan bir view oluşturacağız. Bu view her çalıştığında sistem tarihi baz alınarak o güne ait siparişler listelenecektir.

Öncelikle “CREATE VIEW” komutuyla bir view oluşturuyoruz ve “sales\_order” tablosundan tüm sütunları çekiyoruz. “WHERE” koşuluyla “CONVERT” fonksiyonu kullanılarak tabloda “S\_order\_date” sütununda sistem tarihine eşit olan tarihlerdeki bilgiler çağırılıyor.

```

CREATE VIEW daily_orders AS
SELECT * FROM sales_order
WHERE CONVERT(DATE, S_order_date) = CONVERT(DATE, GETDATE());

```



## 7. SORU 6

Herhangi bir tablo oluşturup anlatılmış olan kısıtlayıcılardan en az ikisinin kullanımına dair bu tabloda örnekler vereceğiz.

Öncelikle “CREATE TABLE” ile bir tablo oluşturuyoruz.

Oluşturulan tabloda birden fazla Primary Key olduğunu sonrasında belirtmek istediğimiz zaman “ADD CONSTRAINT” fonksiyonunu ve “PRIMARY KEY” kısıtlayıcısı olduğunu belirtiriz. “ALTER TABLE” ise sonradan ekleme yapılacağı zaman kullandığımız bir yapıdır ve hangi tabloya ekleme yapılacağını belirtmeye yarar.

“ADD CHECK” fonksiyonu ile kontrol kısıtlayıcısı ekleriz. Burada eklediğimiz kısıtlayıcı “shop\_id”, ‘W’ ile başlayacak demek istemektedir.

“ADD CONSTRAINT” fonksiyonu ile eklenmiş olan ve “DEFAULT” kısıtlayıcısı kullanılan bu komutta yapılmaya çalışılan “shop\_id” kısmının boş olmasını istemediğimiz için herhangi bir değer girmeyi unuttuğumuzda sistem otomatik olarak atama yapсын diyerdir.

```
CREATE TABLE shopier (  
    shop_id INT NOT NULL,  
    sum_worker INT,  
    sum_place INT,  
    principial_name NVARCHAR(25),  
    principial_lastname NVARCHAR(25),  
    principial_id INT NOT NULL,  
    shop_date DATE,  
);
```

```
ALTER TABLE shopier  
ADD CONSTRAINT PK_shop PRIMARY KEY (shop_id, principial_id);
```

```
ALTER TABLE shopier  
ADD CHECK(shop_id LIKE 'W%');
```

```
ALTER TABLE shopier  
ADD CONSTRAINT DF_shop_id  
DEFAULT 'W00000' FOR shop_id;
```

## 8. SORU 7

Herhangi bir tablo oluşturacağız ve anlatılan indeks yapılarından en az iki sorgu çalıştırıp bu tablo üzerinde göstereceğiz.

Önce bir tablo oluşturuyoruz ve sonrasında bu tabloya veriler ekliyoruz.

“CREATE UNIQUE INDEX” komutu tek indeks komutudur. 'Lessons' tablosunda yer alan 'name' sütununda aynı isimde başka bir verinin olmasını engeller.

“CREATE INDEX “ komutu 'Lessons' tablosundaki 'lesson\_id' sütunu için bir indeks oluşturur. Bu indeks “ASC” fonksiyonu sayesinde artan sırada verileri sıralar.

```
CREATE TABLE Lessons (  
    lesson_id INT NOT NULL PRIMARY KEY,  
    name NVARCHAR(25),  
    teacher NVARCHAR(60),  
    class NVARCHAR(10),  
    sum_student INT,  
);
```

```
INSERT INTO Lessons (lesson_id, name, teacher, class, sum_student)  
VALUES ('001', 'Biology', 'Ali', 'A11', 300);
```

```
INSERT INTO Lessons (lesson_id, name, teacher, class, sum_student)  
VALUES ('002', 'Physics', 'Veli', 'B12', 400);
```

```
INSERT INTO Lessons (lesson_id, name, teacher, class, sum_student)  
VALUES ('003', 'Math', 'Ayşe', 'A02', 450);
```

```
INSERT INTO Lessons (lesson_id, name, teacher, class, sum_student)  
VALUES ('004', 'Chemistry', 'Fatma', 'A01', 350);
```

```
INSERT INTO Lessons (lesson_id, name, teacher, class, sum_student)  
VALUES ('005', 'Music', 'Aslı', 'B01', 500);
```

```
CREATE UNIQUE INDEX UQ_Lesson_name ON Lessons(name);
```

```
CREATE INDEX IX_Lessons_lesson_id ON Lessons(lesson_id ASC);
```

## 9. SORU 8

İndeks oluşturmada dikkat edilecek unsurlar nelerdir?

- Farklı değerin fazla olmasına dikkat edilmeli. (Çeşitlilik)
- Sık kullanılan sütunlar tercih edilmeli.
- Çok sık güncellenen sütunlar olmamalı.
- İndex performansı düşmemesi için seçilen sütunda veri tekrarı az olmalı.
- İndeksler zamanla parçalanabilir ve performansını kaybedebilir. Bu nedenle düzenli bakım yapılması gerekir.
- Gereksiz yere çok indeks koyulmamalı.

Bir veri tabanı tasarımında hangi sütunlar üzerinde indeks oluşturulur?

- Sık kullanılan sorgulardaki sütunlar için kullanılır.
- Primary Key ve Foreign Key sütunlarında kullanılır.
- Çok farklı değerler içeren sütunlarda kullanılır.
- Sıralama ve gruptama yapılan sütunlar kullanılır.