

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



2024 – 2025 BAHAR DÖNEMİ

FÖY-3 RAPORU

HAZIRLAYAN:
21060684 - AYŞEGÜL AKMAN

SAMSUN-2025

ÖZET

Bu raporda SQL Server üzerinde veri tabanı oluşturma, veri girişi yapma ve SQL sorguları ile istenilen bilgileri gerçekleştirme ele alınmaktadır. Öncelikle verilen diyagrama uygun bir veri tabanı oluşturulmuş tablolar arasındaki ilişkiler dikkatlice aktarılmıştır. Primary Key ve Foreign Key atamaları yapılmış çeşitli veri tipleri atanmıştır. Her bir tabloya istenilen veriler girilmiştir. Son olarak SQL sorguları ile veri tabanının doğruluğu incelenmiştir.

İÇİNDEKİLER

1. SORU 1

2. SORU 2

3. SORU 3

4. SORU 4

5. SORU 5

6. SORU 6

7. SORU 7

8. SORU 8

9. SORU 9

10. SORU 10

1. SORU 1

SQL Server’da SQL komutları kullanılarak föy içerisinde verilen diyagrama uygun bir veri tabanı oluşturuldu. Veri tabanı oluşturulurken öncelikle “birimler” tablosu “birim_ad” ve “birim_id” adlı sütunlar yer alacak şekilde tanımlandı. “calisanlar” tablosunda “ad”, “soyad”, “calisan_id”, “maas”, “katilmaTarihi” ve “calisan_birim_id” yer almaktadır. “calisanlar” tablosunun “calisan_birim_id” sütunu ve “birimler” tablosunun “birim_id” sütunu birincil-ikincil anahtar ilişkisi oluşturmaktadır. Bu ilişki ile her bir çalışanın hangi birimde çalıştığı belirtilmektedir.

“unvanlar” tablosunda “unvan_calisan_id”, “unvan_calisan” ve “unvan_tarih” bilgileri yer almaktadır. “calisanlar” tablosunun “calisan_id” sütunu ve “unvanlar” tablosunun “unvan_calisan_id” sütunu sayesinde tablolar birbirleri ile ilişki içindedirler. Bu ilişki ile her bir çalışanın sahip olduğu unvan belirtilmektedir.

“ikramiyeler” tablosunda “ikramiye_calisan_id”, “ikramiye_ucret” ve “ikramiye_tarih” bilgileri yer almaktadır. “ikramiyeler” tablosunun “ikramiye_calisan_id” sütunu ve “calisanlar” tablosunun “calisan_id” sütunu ilişkilidir. Bu ilişki ile ikramiye alan çalışanların kimler oldukları belirtiliyor.

```

CREATE DATABASE Salary_Database;
GO

USE Salary_Database;
GO

CREATE TABLE birimler (
    birim_id INT PRIMARY KEY,
    birim_ad NVARCHAR(25)
);

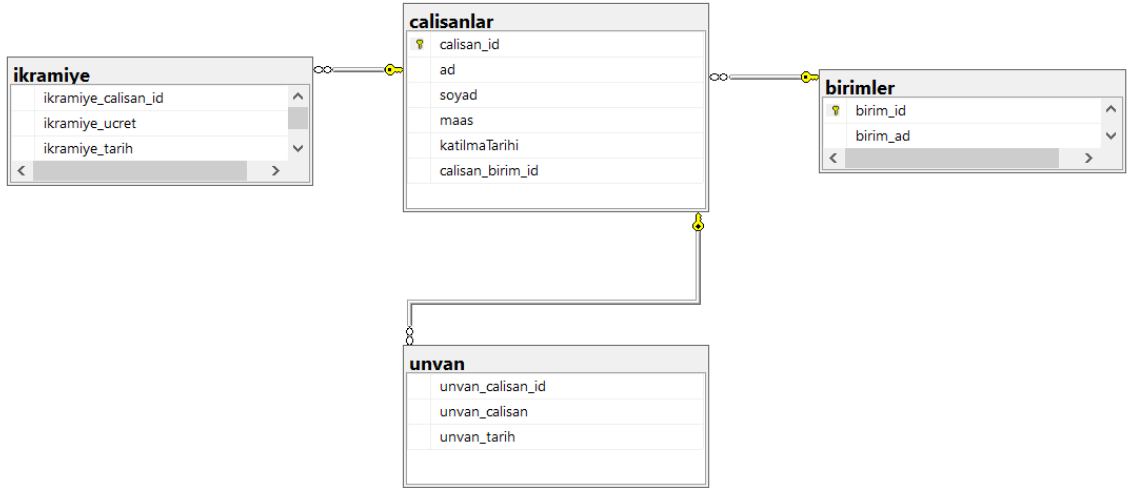
CREATE TABLE calisanlar (
    calisan_id INT PRIMARY KEY,
    ad NVARCHAR(25) NOT NULL,
    soyad NVARCHAR(25) NOT NULL,
    maas INT NOT NULL,
    katilmaTarihi DATETIME NOT NULL,
    calisan_birim_id INT,
    FOREIGN KEY (calisan_birim_id) REFERENCES birimler(birim_id)
);

CREATE TABLE unvan (
    unvan_calisan_id INT,
    unvan_calisan NVARCHAR(25) NOT NULL,
    unvan_tarih DATETIME NOT NULL,
    FOREIGN KEY (unvan_calisan_id) REFERENCES calisanlar(calisan_id)
);

CREATE TABLE ikramiye (
    ikramiye_calisan_id INT,
    ikramiye_ucret INT NOT NULL,
    ikramiye_tarih DATETIME NOT NULL,
    FOREIGN KEY (ikramiye_calisan_id) REFERENCES calisanlar(calisan_id)
);

```

- Salary_Database
 - Database Diagrams
 - Tables
 - System Tables
 - FileTables
 - External Tables
 - Graph Tables
 - dbo.birimler
 - dbo.calisanlar
 - dbo.ikramiye
 - dbo.unvan
 - Views
 - External Resources
 - Synonyms
 - Programmability
 - Service Broker
 - Storage
 - Security



2. SORU 2

Diyagrama uygun veri tabanı yapıldıktan sonra bu veri tabanına “INSERT INTO” komutu ile ilgili veri girişleri yapıldı.

```
USE Salary_Database;
GO

INSERT INTO birimler (birim_id, birim_ad)
VALUES
(1, 'Yazılım'),
(2, 'Donanım'),
(3, 'Güvenlik');

INSERT INTO calisanlar (calisan_id, ad, soyad, maas, katilmaTarihi, calisan_birim_id)
VALUES
(1, 'İsmail', 'İşeri', 100000, '2014-02-20 00:00:00.000', 1),
(2, 'Hami', 'Satılmış', 80000, '2014-06-11 00:00:00.000', 1),
(3, 'Durmuş', 'Şahin', 300000, '2014-02-20 00:00:00.000', 2),
(4, 'Kağan', 'Yazar', 500000, '2014-02-20 00:00:00.000', 3),
(5, 'Meryem', 'Soysaldı', 500000, '2014-06-11 00:00:00.000', 3),
(6, 'Duygu', 'Akşehir', 200000, '2014-06-11 00:00:00.000', 2),
(7, 'Kübra', 'Seyhan', 75000, '2014-01-20 00:00:00.000', 1),
(8, 'Gülcan', 'Yıldız', 90000, '2014-04-11 00:00:00.000', 3);

INSERT INTO ikramiye (ikramiye_calisan_id, ikramiye_ucret, ikramiye_tarih)
VALUES
(1, 5000, '2016-02-20 00:00:00.000'),
(2, 3000, '2016-06-11 00:00:00.000'),
(3, 4000, '2016-02-20 00:00:00.000'),
(1, 4500, '2016-02-20 00:00:00.000'),
(2, 3500, '2016-06-11 00:00:00.000');

INSERT INTO unvan (unvan_calisan_id, unvan_calisan, unvan_tarih)
VALUES
(1, 'Yazılım', '2016-02-20 00:00:00.000'),
(2, 'Donanım', '2016-06-11 00:00:00.000'),
(3, 'Güvenlik', '2016-02-20 00:00:00.000'),
(1, 'Yazılım', '2016-02-20 00:00:00.000'),
(2, 'Donanım', '2016-06-11 00:00:00.000');

91 %
Messages

(3 rows affected)

(8 rows affected)

(5 rows affected)

(8 rows affected)

Completion time: 2025-03-20T16:35:39.3714818+03:00
```

```

SELECT * FROM birimler;
SELECT *FROM calisanlar;
SELECT *FROM ikramiye;
SELECT *FROM unvan;

```

90 %

Results Messages

	birim_id	birim_ad
1	1	Yazilim
2	2	Donanim
3	3	Güvenlik

	calisan_id	ad	soyad	maas	katilmaTarihi	calisan_birim_id
1	1	Ismail	Iseri	100000	2014-02-20 00:00:00.000	1
2	2	Hami	Satilmis	80000	2014-06-11 00:00:00.000	1
3	3	Durmus	Sahin	300000	2014-02-20 00:00:00.000	2
4	4	Kagan	Yazar	500000	2014-02-20 00:00:00.000	3
5	5	Meryem	Soysaldi	500000	2014-06-11 00:00:00.000	3
6	6	Duygu	Aksehir	200000	2014-06-11 00:00:00.000	2
7	7	Kübra	Seyhan	75000	2014-01-20 00:00:00.000	1
8	8	Gülcan	Yildiz	90000	2014-04-11 00:00:00.000	3

	ikramiye_calisan_id	ikramiye_ucret	ikramiye_tarih
1	1	5000	2016-02-20 00:00:00.000
2	2	3000	2016-06-11 00:00:00.000
3	3	4000	2016-02-20 00:00:00.000
4	1	4500	2016-02-20 00:00:00.000
5	2	3500	2016-06-11 00:00:00.000

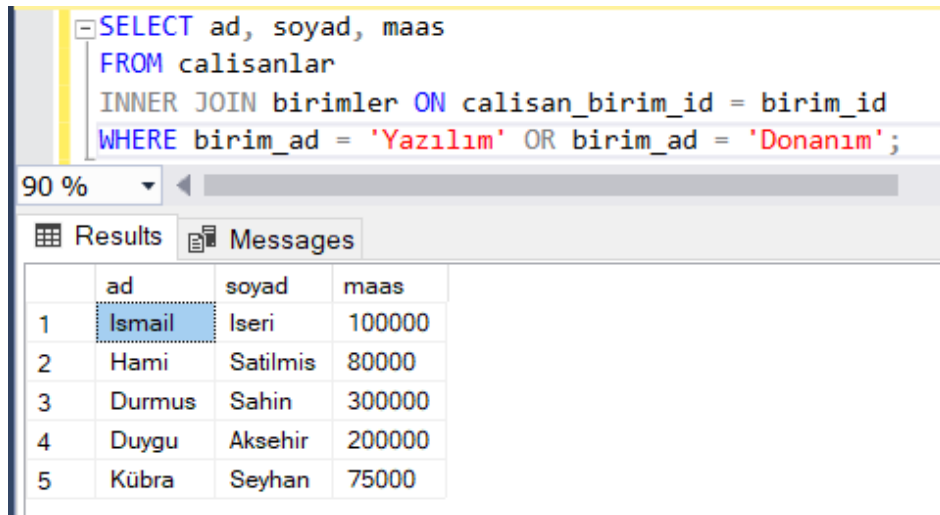
	unvan_calisan_id	unvan_calisan	unvan_tarih
1	1	Yönetici	2016-02-20 00:00:00.000
2	2	Personel	2016-02-20 00:00:00.000
3	8	Personel	2016-02-20 00:00:00.000
4	5	Müdür	2016-02-20 00:00:00.000
5	4	Yönetici Yard...	2016-02-20 00:00:00.000
6	7	Personel	2016-02-20 00:00:00.000
7	6	Takim Lideri	2016-02-20 00:00:00.000
8	3	Takim Lideri	2016-02-20 00:00:00.000

3. SORU 3

“Yazılım” ve “Donanım” birimlerinde çalışanların ad, soyad ve maaş bilgilerini listeleyen SQL sorgusu yazılacaktır. Bu sorgu ile “calisanlar” tablosundan ilgili birimlere ait çalışanlar seçilmektedir. Öncelikle “birimler” tablosunda “Yazılım” ve “Donanım” birimlerinin ID’leri belirleniyor. Daha sonra “calisanlar” tablosunda bu birim ID’ye sahip çalışanlar listelenir.

Bu SQL kodu sayesinde “calisanlar” ve “birimler” tablosu birleşerek istenilen birimlerde çalışanların ad, soyad, ve maaş bilgileri listelenecek. Bu birleşim INNER JOIN kullanılarak sağlanacak. “calisan_birim_id” sütunu ve “birim_id” sütunu eşleştirilecek.

WHERE koşulu ile filtreleme yapılacak ve sadece istenilen bilgilere sahip olanlar listelenecek.



The screenshot shows a SQL query editor with a query window and a results window. The query window contains the following SQL code:

```
SELECT ad, soyad, maas
FROM calisanlar
INNER JOIN birimler ON calisan_birim_id = birim_id
WHERE birim_ad = 'Yazılım' OR birim_ad = 'Donanım';
```

The results window shows a table with 5 rows and 4 columns: ad, soyad, and maas. The first row is highlighted.

	ad	soyad	maas
1	Ismail	Iseri	100000
2	Hami	Satilmis	80000
3	Durmus	Sahin	300000
4	Duygu	Aksehir	200000
5	Kübra	Seyhan	75000

4. SORU 4

Maaşı en yüksek olan çalışanların ad, soyad ve maaş bilgilerini listeleyen SQL sorgusu yazılacaktır. “calisanlar” tablosundan bilgi çekilecektir. WHERE koşulu ile maaşı diğer çalışanlara göre en yüksek olan kişi veya kişiler listelenecektir. Alt sorgu kullanılarak gerçekleştirilen bu sorgulama işleminde MAX(maas) fonksiyonu en yüksek maaşı bulmaktadır.

```
SELECT ad, soyad, maas
FROM calisanlar
WHERE maas = (
    SELECT MAX(maas)
    FROM calisanlar
);
```

90 %

Results Messages

	ad	soyad	maas
1	Kagan	Yazar	500000
2	Meryem	Soysaldi	500000

5. SORU 5

Birimlerin her birinde kaç adet çalışan olduğunu ve birimlerin adlarını listeleyen SQL sorgusu yazılacaktır. Bu sorguda bir grupta bir işleme de vardır. Grupta bir işleme sayesinde her birimde kaç adet çalışan olduğunu belirleriz.

Sorguda her bir birim için çalışan kişi sayısını belirlemede sayma işlemi için COUNT kullanılır. LEFT JOIN kullanılarak birimler tablosu temel alınır ve calisanlar tablosu ile eşleştirme yapılır. Böylece çalışanı olmayan birimler de listelenmiş olur. GROUP BY ile birim adlarına göre grupta bir işleme yapılır.

```
SELECT birim_ad, COUNT(calisan_id) FROM birimler
LEFT JOIN calisanlar ON birimler.birim_id = calisanlar.calisan_birim_id
GROUP BY birim_ad;
```

91 %

Results Messages

	birim_ad	(No column name)
1	Donanim	2
2	Güvenlik	3
3	Yazilim	3

6. SORU 6

Birden fazla çalışana ait unvanların isimlerini ve o unvan altında çalışanların sayısını listeleyen SQL sorgusu yazılacaktır.

Bu SQL sorgusu ile “unvan” tablosundan unvanlar alınarak çalışanların sayısı hesaplanacaktır. Daha sonra HAVING ifadesi ile sadece birden fazla çalışana sahip olan unvanlar seçilecektir.

```
SELECT unvan_calisan, COUNT(calisan_id) FROM unvan
LEFT JOIN calisanlar ON unvan.unvan_calisan_id = calisanlar.calisan_id
GROUP BY unvan_calisan
HAVING COUNT(calisan_id) > 1;
```

91 %

Results Messages

	unvan_calisan	(No column name)
1	Personel	3
2	Takim Lideri	2

7. SORU 7

Maaşları “50000” ve “100000” arasında değişen çalışanların ad, soyad ve maaş bilgilerini listeleyen sorguyu yazacağız.

“calisanlar” tablosundan veri çekilecektir. Bunun için SELECT ile ad, soyad ve maaş bilgilerinin döndürüleceği bilgisi verilir. FROM ile hangi tablodan veri çekileceği belirtilir. WHERE maas BETWEEN 50000 AND 100000 koşuluyla hangi sütun olduğu ve hangi koşulu istediği belirtilir.

```
SELECT ad, soyad, maas FROM calisanlar WHERE maas BETWEEN 50000 AND 100000;
```

91 %

Results Messages

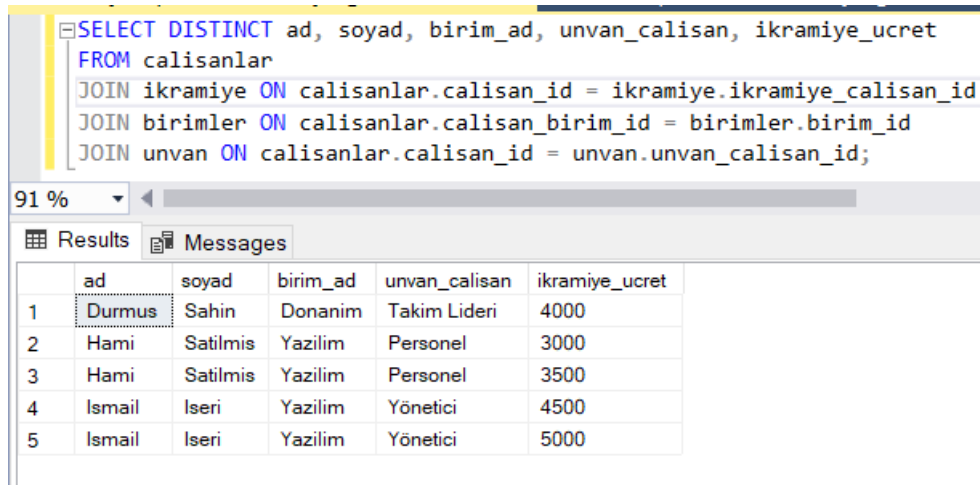
	ad	soyad	maas
1	Ismail	Iseri	100000
2	Hami	Satilmis	80000
3	Kübra	Seyhan	75000
4	Gülcan	Yildiz	90000

8. SORU 8

İkramiye hakkına sahip çalışanlara ait ad, soyad, birim, unvan ve ikramiye ücret bilgilerini listeleyen SQL sorgusu yazılacak.

Bu sorguda “calisanlar” tablosundan ad ve soyad bilgileri; “birimler” tablosundan birim (birim_ad); “unvan” tablosundan unvan (unvan_ad) ve “ikramiye” tablosundan ikramiye (ikramiye_ucret) alınacaktır.

Her bir tablo birleştirilerek sorgulama yapılacaktır.



The screenshot shows a SQL query in a query editor and its results in a table. The query is a JOIN query that combines data from four tables: calisanlar, ikramiye, birimler, and unvan. The results table has five columns: ad, soyad, birim_ad, unvan_calisan, and ikramiye_ucret. There are five rows of data.

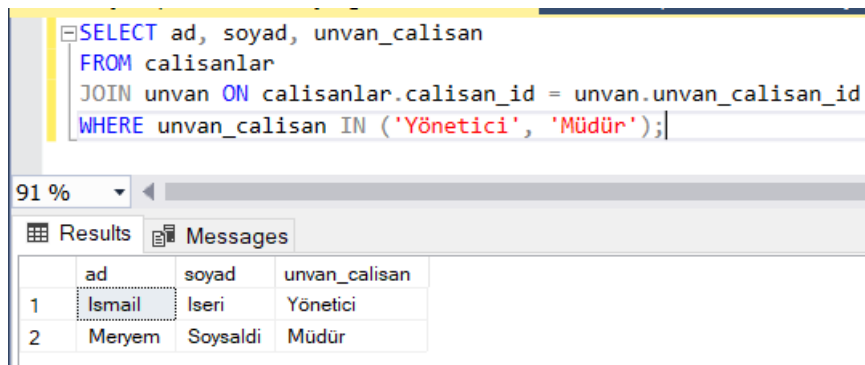
```
SELECT DISTINCT ad, soyad, birim_ad, unvan_calisan, ikramiye_ucret
FROM calisanlar
JOIN ikramiye ON calisanlar.calisan_id = ikramiye.ikramiye_calisan_id
JOIN birimler ON calisanlar.calisan_birim_id = birimler.birim_id
JOIN unvan ON calisanlar.calisan_id = unvan.unvan_calisan_id;
```

	ad	soyad	birim_ad	unvan_calisan	ikramiye_ucret
1	Durmus	Sahin	Donanim	Takim Lideri	4000
2	Hami	Satilmis	Yazilim	Personel	3000
3	Hami	Satilmis	Yazilim	Personel	3500
4	Ismail	Iseri	Yazilim	Yönetici	4500
5	Ismail	Iseri	Yazilim	Yönetici	5000

9. SORU 9

Unvanı “Yönetici” ve “Müdür” olan çalışanların ad, soyad ve unvanlarını listeleyen SQL sorgusunu yazacağız.

Burada “calisanlar” ve “unvan” tabloları “calisan_id” ve “unvan_calisan_id” eşleşmesi sayesinde ilgili bilgileri ekrana yazdıracaktır.



The screenshot shows a SQL query in a query editor and its results in a table. The query is a JOIN query that combines data from two tables: calisanlar and unvan. The results table has three columns: ad, soyad, and unvan_calisan. There are two rows of data.

```
SELECT ad, soyad, unvan_calisan
FROM calisanlar
JOIN unvan ON calisanlar.calisan_id = unvan.unvan_calisan_id
WHERE unvan_calisan IN ('Yönetici', 'Müdür');
```

	ad	soyad	unvan_calisan
1	Ismail	Iseri	Yönetici
2	Meryem	Soysaldi	Müdür

10. SORU 10

Her bir birimde en yüksek maaş alan çalışanların ad, soyad ve maaş bilgilerini listeleyen SQL sorgusunu yazacağız.

En yüksek maaşı belirlemek için bir alt sorgu kullanacağız. Öncelikle “calisanlar” ve “birimler” tablosunu JOIN ile birleştiriyoruz. WHERE koşulu içerisinde alt sorgu açarak “calisan_birim_id” ile her bir birimdeki max maaş belirlenir. Maksimum maaş için MAX(maas) ifadesi kullanılır.

Birleştirme işlemi her çalışanın bulunduğu birimin max maaşı ile eşlesen bir “calisan_birim_id” si olmasını sağlar.

```
SELECT ad, soyad, maas
FROM calisanlar
JOIN birimler ON calisanlar.calisan_birim_id = birimler.birim_id
WHERE calisanlar.maas = (
    SELECT MAX(maas)
    FROM calisanlar
    WHERE calisanlar.calisan_birim_id = birimler.birim_id
);
```

91 %

Results Messages

	ad	soyad	maas
1	Kagan	Yazar	500000
2	Meryem	Soysaldi	500000
3	Durmus	Sahin	300000
4	Ismail	Iseri	100000

GİTHUB LİNK:

<https://github.com/aysegulnamka/Veri-Lab.git>