****

**Fırat Üniversitesi**

**Mühendislik Fakültesi Bilgisayar Mühendisliği**

**BMÜ329 VERİ TABANI SİSTEMLERİ**

**Proje Adı:** Otomotiv Sektöründe Müşteri Ve Stok Takip

**Proje Ekibi:**

* (210260302) Celal Can SAĞNAK
* (220260134) Mehmet Can AKTOĞ
* (220260162) Rümeysa ER

**1. Proje Özeti:**

“Otomotiv Sektöründe Müşteri Ve Stok Takip" projesi, küçük şehirlerde kurumsallaşamadan geleneksel defter tutma yöntemlerini kullanan otomotiv tamirat ve tadilat işletmelerinin işleyişlerini modernize etmeyi ve iyileştirmeyi amaçlayan kapsamlı bir projedir. Projenin temel amacı, işletmelerin stok takibi, müşteri kayıtları, maliyet hesaplamaları gibi temel iş süreçlerini etkin bir şekilde yönetmelerini ve gerektiğinde bu kayıtları düzenlemelerini sağlamaktır. Bu amaçla, kullanıcı dostu bir arayüzün yanı sıra arka planda karmaşık ilişkisel veri tabanı modelleri kullanılacaktır.

**2. Proje Ayrıntıları:**

**2.1. Proje Amaçları:**

* İşletmelerin stok takibi süreçlerini güçlendirmek ve stok yönetimini daha etkili hale getirmek.
* Müşteri kayıtlarını güvenli ve sistematik bir şekilde tutarak müşteri ilişkilerini yönetebilmek.
* Maliyet hesaplamalarını doğru bir şekilde yaparak işletmelerin karını artırmak.
* İşletmelerin tüm bu süreçleri kolay, anlaşılır ve etkili bir şekilde yönetmelerini sağlamak.

**2.2. Proje Süreci:**

* İşletmelerin stok takibi için veri tabanı oluşturulması ve ilişkisel modellerin tasarlanması.
* Müşteri kayıtlarını tutacak bir veri tabanı mimarisi oluşturulması.
* Maliyet hesaplamaları için gerekli veri tabanı yapılarının oluşturulması.
* Kullanıcı dostu bir arayüz geliştirme ve tüm bu süreçlerin entegrasyonu.
* Test aşamaları ve kullanıcı geri bildirimlerine dayalı iyileştirmeler.

**2.3. Kullanılan Teknolojiler:**

* SQL Server Management Studio 2022: Güçlü veri tabanı yönetim araçları ve işletme için verimli veri tabanı altyapısı sağlanacaktır.
* İlişkisel Veritabanı Modelleri: İşletmelerin karmaşık veri ilişkilerini yönetmek için uygun veri tabanı modelleri tasarlanacak ve uygulanacaktır.

**3. Proje Üyelerinin Görevleri:**

1. Celal Can: Veri tabanı tasarımı ve geliştirme sorumlusu olarak stok takibi ve maliyet hesaplamaları için veri tabanı mimarisini oluşturacak ve yönetecektir.
2. Mehmet Can: Arayüz geliştirme ve entegrasyon uzmanı olarak kullanıcı dostu bir arayüz tasarlayacak, tüm süreçlerin entegrasyonunu sağlayacak ve kullanıcı deneyimini optimize edecektir.
3. Rümeysa: Test ve geri bildirim koordinatörü olarak proje aşamalarını test edecek, kullanıcı geri bildirimlerini toplayacak ve bu geri bildirimlere dayalı olarak geliştirmeler yapılmasını sağlayacaktır.

**4. Proje Sonuçları:**

Bu proje tamamlandığında, otomotiv tamirat ve tadilat işletmeleri aşağıdaki sonuçları elde edecektir:

* Stok takibi süreçlerinin daha etkin ve verimli hale gelmesi.
* Müşteri kayıtlarının güvenli ve sistematik bir şekilde tutulması.
* Maliyet hesaplamalarının doğru ve sistematik bir şekilde yapılması.
* İşletmelerin rekabet gücünün artması ve karlılığının yükselmesi.

**5. Sonuç:**

“Otomotiv Sektöründe Müşteri Ve Stok Takip" projesi, geleneksel işletme yöntemlerinden modern ve veri tabanı tabanlı bir işleyişe geçişi destekleyen önemli bir adımdır. Proje ekibi, süreç boyunca işbirliği içinde çalışarak proje hedeflerine ulaşmayı hedeflemektedir. Projenin başarısı, küçük işletmelerin daha etkili ve verimli bir şekilde iş yapmalarına katkı sağlayacaktır. Elde edilen deneyimler ve sonuçlar, benzer dönüşüm projelerinin geliştirilmesi ve uygulanmasında da rehberlik sağlayacaktır.

**6.SQL Sorguları:**

**CREATE Komutu;**

**CREATE TABLE musteri\_bilgi (**

**musteri\_id INT PRIMARY KEY,**

**ad\_soyad VARCHAR(255),**

**arac\_id INT**

**);**

**SELECT komutu:**

**SELECT \* FROM musteri\_bilgi**

**WHERE musteri\_id = 1;**

**DELETE Komutu;**

**DELETE FROM musteri\_bilgi**

**WHERE musteri\_id = 1;**

**UPDATE Komutu;**

**UPDATE musteri\_bilgi SET**

**ad\_soyad = 'Mehmet Öz' WHERE**

**musteri\_id = 1;**

**FUNCTION Komutu;**

**CREATE FUNCTION**

**get\_total\_cost(arac\_id INT)**

**RETURNS INT**

**BEGIN DECLARE total\_cost INT;**

**SELECT SUM(maliyet) INTO**

**total\_cost**

**FROM parca\_bilgi**

**WHERE arac\_bilgi\_id = arac\_id;**

**RETURN total\_cost;**

**END;**

**STORED PROCEDURE Komutu;**

**CREATE PROCEDURE**

**AddNewPart(IN \_parca\_adi**

**VARCHAR(255), IN \_maliyet**

**DECIMAL(10,2))**

**BEGIN**

**INSERT INTO parca\_bilgi**

**(parca\_adi, maliyet) VALUES**

**(\_parca\_adi, \_maliyet);**

**END;**

**TRIGGER Komutu:**

**CREATE TRIGGER after\_part\_insert**

**AFTER INSERT ON parca\_bilgi FOR**

**EACH ROW**

**BEGIN**

**UPDATE stok\_bilgi**

**SET parca\_adet = parca\_adet + 1**

**WHERE parca\_id = NEW.parca\_id;**

**END;**

**TRANSACTION Komutu:**

**START TRANSACTION;**

**INSERT INTO musteri\_bilgi**

**(musteri\_id, ad\_soyad, arac\_id)**

**VALUES (1, 'Ahmet Yılmaz', 101);**

**INSERT INTO arac\_bilgi (arac\_id,**

**marka, model) VALUES (101,**

**'Toyota', 'Corolla');**

**COMMIT;**

**INDEX Komutu:**

**CREATE INDEX idx\_musteri ON**

**musteri\_bilgi (ad\_soyad);**

**VIEW Komutu;**

**CREATE VIEW arac\_musteri\_view**

**AS**

**SELECT mb.musteri\_id,**

**mb.ad\_soyad, ab.marka, ab.model**

**FROM musteri\_bilgi mb**

**JOIN arac\_bilgi ab ON mb.arac\_id**

**= ab.arac\_id;**

**7 Karmaşık Sorgular ve İşlemler:**

**Belirli bir müşterinin tüm bilgilerini getirmek:**

**SELECT mb.\*, ib.adres, ib.telefon**

**FROM musteri\_bilgi mb**

**JOIN iletisim\_bilgi ib ON mb.musteri\_id = ib.musteri\_id**

**WHERE mb.musteri\_id = 1;**

**Müşterinin toplam ödeme bilgisini hesaplamak:**

**SELECT mb.musteri\_id, mb.ad\_soyad, SUM(ob.taksit\_sayisi \* tb.iscilik\_maliyet) AS toplam\_odeyecegi**

**FROM musteri\_bilgi mb**

**JOIN arac\_bilgi ab ON mb.arac\_id = ab.arac\_id**

**JOIN tamirat\_bilgi tb ON ab.arac\_id = tb.arac\_bilgi\_id**

**JOIN odeme\_bilgi ob ON mb.musteri\_id = ob.odeme\_id**

**GROUP BY mb.musteri\_id;**

**Personelin alınan sipariş başına ortalama kazancını hesaplamak:**

**SELECT personel\_id, (aylik\_brut\_maas / kayit\_alinan\_siparisler)**

**AS ortalama\_kazanc**

**FROM personel\_bilgi;**

**Araç için gerekli olan parçaların maliyetini hesaplamak:**

**SELECT SUM(pb.maliyet) AS toplam\_maliyet**

**FROM parca\_bilgi pb**

**WHERE pb.arac\_bilgi\_id = 1;**

**İskonto uygulanmış fiyatı hesaplamak:**

**SELECT parca\_id, maliyet, maliyet - (maliyet \* ib.yuzdelik\_oran / 100)**

**AS iskontolu\_fiyat**

**FROM parca\_bilgi pb**

**JOIN iskonto\_bilgi ib ON pb.arac\_bilgi\_id = ib.iskonto\_id;**

**8. Örnekler İle kompleks SQL sorguları:**

**1.Araçlar için toplam maliyet hesaplama:**

**SELECT**

**a.arac\_id,**

**a.marka,**

**a.model,**

**SUM(p.maliyet) AS toplam\_parca\_maliyeti,**

**SUM(t.iscilik\_maliyet) AS toplam\_iscilik\_maliyeti,**

**(SUM(p.maliyet) + SUM(t.iscilik\_maliyet)) AS toplam\_maliyet**

**FROM**

**arac\_bilgi a**

**LEFT JOIN parca\_bilgi p ON a.arac\_id = p.arac\_bilgi\_id**

**LEFT JOIN tamirat\_bilgi t ON a.arac\_id = t.arac\_bilgi\_id**

**GROUP BY**

**a.arac\_id;**

**2.İndirim oranına göre satış fiyatı hesaplama**:

**SELECT**

**s.stok\_id,**

**s.parca\_adi,**

**p.maliyet,**

**i.yuzdelik\_oran,**

**(p.maliyet - (p.maliyet \* i.yuzdelik\_oran / 100)) AS indirimli\_fiyat**

**FROM**

**stok\_bilgi s**

**JOIN parca\_bilgi p ON s.parca\_numarasi = p.parca\_id**

**JOIN iskonto\_bilgi i ON s.stok\_id = i.iskonto\_id;**

**4.Her müşterinin toplam borç bilgisini öğrenme**:

**SELECT**

**mb.musteri\_id,**

**mb.ad\_soyad,**

**SUM(ob.taksit\_sayisi \* tb.iscilik\_maliyet) AS toplam\_borc**

**FROM**

**musteri\_bilgi mb**

**JOIN odeme\_bilgi ob ON mb.musteri\_id = ob.odeme\_id**

**JOIN tamirat\_bilgi tb ON mb.arac\_id = tb.arac\_bilgi\_id**

**GROUP BY**

**mb.musteri\_id;**

**3.Personel başına düşen ortalama sipariş maliyeti**:

**SELECT**

**personel\_id,**

**ad\_soyad,**

**(aylik\_brut\_maas / kayit\_alinan\_siparisler) AS ortalama\_maliyet**

**FROM**

**personel\_bilgi**

**WHERE**

**kayit\_alinan\_siparisler > 0;**

**6.Müşteriler ve onların araçlarına yapılan son tamirat bilgileri**:

**SELECT**

**mb.musteri\_id,**

**mb.ad\_soyad,**

**a.marka,**

**a.model,**

**t.son\_tamirat**

**FROM**

**musteri\_bilgi mb**

**INNER JOIN (**

**SELECT**

**arac\_bilgi\_id,**

**MAX(tt\_suresi) AS son\_tamirat**

**FROM**

**tamirat\_bilgi**

**GROUP BY**

**arac\_bilgi\_id**

**) t ON mb.arac\_id = t.arac\_bilgi\_id**

**JOIN arac\_bilgi a ON mb.arac\_id = a.arac\_id;**

**5.Her marka için stoktaki parça sayısı ve toplam değeri**:

**SELECT**

**a.marka,**

**COUNT(p.parca\_id) AS parca\_sayisi,**

**SUM(s.parca\_adet \* p.maliyet) AS toplam\_stok\_degeri**

**FROM**

**arac\_bilgi a**

**JOIN parca\_bilgi p ON a.arac\_id = p.arac\_bilgi\_id**

**JOIN stok\_bilgi s ON p.parca\_id = s.parca\_numarasi**

**GROUP BY**

**a.marka;**

**8.Stoktaki parçaların toplam maliyeti ve ortalama maliyeti**:

**SELECT**

**parca\_adi,**

**SUM(parca\_adet) AS toplam\_stok\_adeti,**

**SUM(parca\_adet \* maliyet) AS toplam\_maliyet,**

**AVG(maliyet) AS ortalama\_maliyet**

**FROM**

**stok\_bilgi s**

**JOIN parca\_bilgi p ON s.parca\_numarasi = p.parca\_id**

**GROUP BY**

**parca\_adi;**

**7.Müşteriye göre yapılan ödemelerin toplamı ve ortalama ödeme tutarı:**

**SELECT**

**mb.musteri\_id,**

**mb.ad\_soyad,**

**COUNT(o.odeme\_id) AS toplam\_odeme\_sayisi,**

**SUM(o.taksit\_sayisi) AS toplam\_taksit\_sayisi,**

**AVG(o.taksit\_sayisi) AS ortalama\_taksit\_sayisi**

**FROM**

**musteri\_bilgi mb**

**LEFT JOIN odeme\_bilgi o ON mb.musteri\_id = o.odeme\_id**

**GROUP BY**

**mb.musteri\_id;**

**10.Her aracın modeline göre gereken parça sayısı ve toplam maliyeti**:

**SELECT**

**a.model,**

**COUNT(p.parca\_id) AS gerekli\_parca\_sayisi,**

**SUM(p.adet) AS toplam\_gerekli\_adet,**

**SUM(p.maliyet) AS toplam\_maliyet**

**FROM**

**arac\_bilgi a**

**LEFT JOIN parca\_bilgi p ON a.arac\_id = p.arac\_bilgi\_id**

**GROUP BY**

**a.model;**

**9.Her personelin çalıştığı dönemde aldığı toplam siparişler ve ortalama maaş**:

**SELECT**

**p.personel\_id,**

**p.ad\_soyad,**

**COUNT(s.kayit\_alinan\_siparisler) AS toplam\_siparis\_sayisi,**

**DATEDIFF(NOW(), p.baslama\_tarihi) / 30 AS calisma\_ayi,**

**p.aylik\_brut\_maas,**

**(p.aylik\_brut\_maas / (DATEDIFF(NOW(), p.baslama\_tarihi) / 30)) AS ortalama\_aylik\_maas**

**FROM**

**personel\_bilgi p**

**LEFT JOIN satis\_bilgi s ON p.personel\_id = s.personel\_id**

**GROUP BY**

**p.personel\_id;**

**9. Normalizasyon Aşamaları:**

**1. Normal Form (1NF):**

Her sütunun atomik değerleri olmalı (yani, her sütun tek bir değer içermelidir). Her sütunda veri tipi aynı olmalıdır. Her satır benzersiz olmalıdır.

**2. Normal Form (2NF):**

1NF'de olmalıdır. Her sütun tam olarak birincil anahtara bağlı olmalıdır (yani, kısmi bağımlılıklar olmamalıdır).

**3. Normal Form (3NF):**

2NF'de olmalıdır. Herhangi bir sütun diğer sütunlara transitif olarak bağımlı olmamalıdır (yani, birincil anahtara dolaylı bağımlılıklar olmamalıdır).

**9.1. Normalizasyon Düzenlemeleri:**

**musteri\_bilgi Tablosu**

arac\_id burada yabancı anahtar olarak kalmamalı, çünkü bir müşterinin birden fazla araca sahip olabileceğini varsayarsak, bu bir kısmi bağımlılık yaratır. Bunun yerine, musteri\_arac isimli bir ara tablo oluşturulmalı.

**iletisim\_bilgi Tablosu**

musteri\_id burada yabancı anahtar olarak kalmalı. Bir müşterinin birden fazla iletişim bilgisi olabilir, bu durumda iletişim bilgisi için ayrı birincil anahtar eklenmeli.

**parca\_bilgi ve tamirat\_bilgi Tabloları**

Her iki tabloda da arac\_bilgi\_id yabancı anahtar olarak kalmalı. Ancak, bir aracın birden fazla parçaya veya tamirata ihtiyacı olabileceğinden, bu tabloların kendi birincil anahtarları olmalı.

**stok\_bilgi Tablosu**

parca\_numarasi veri tekrarını azaltmak ve veri bütünlüğünü sağlamak için parca\_bilgi ile ilişkilendirilmeli ve burada birincil anahtar olarak kullanılmamalıdır.

**Yeni Ara Tablolar:**

**Müşteri ve Araç İlişkisi:**

CREATE TABLE musteri\_arac (

musteri\_id INT,

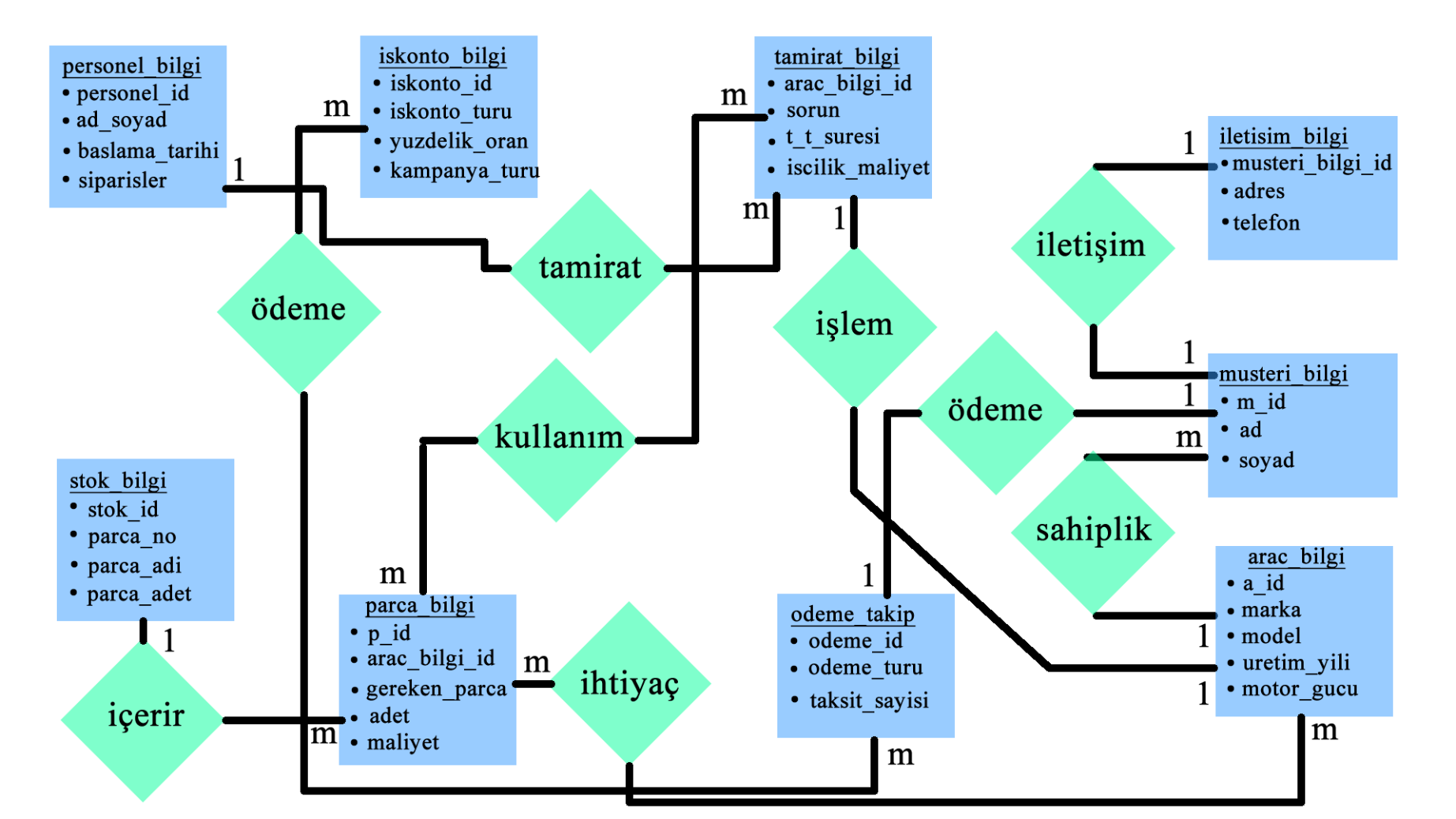
arac\_id INT,

PRIMARY KEY (musteri\_id, arac\_id),

FOREIGN KEY (musteri\_id) REFERENCES musteri\_bilgi(musteri\_id),

FOREIGN KEY (arac\_id) REFERENCES arac\_bilgi(arac\_id)

);

**10. ER Diyagramı:**