Bölüm 3

İleri Tasarım ve Uygulama

7 Yapılandırılmış Sorgu Diline (SQL) Giriş

8 Gelişmiş SQL

9 Veritabanı Tasarımı

Bölüm

7

Giriş Yapılandırılmış Sorgu Dili (SQL)

Öğrenme Hedefleri

Bu bölümü tamamladıktan sonra şunları yapabileceksiniz:

- 7-1 Bir veritabanından belirtilen veri sütunlarını alma
- 7-2 Tek bir SQL sorgusunda birden fazla tabloya katılma
- 7-3 Veri alımlarını karmaşık ölçütlerle eşleşen satırlarla kısıtlayın
- 7-4 Satır grupları arasında veri toplama
- 7-5 Diğer sorgulara dahil etmek üzere verileri önceden işlemek için alt sorgular oluşturma
- 7-6 Dize, sayı ve tarih manipülasyonu için çeşitli SQL fonksiyonlarını tanımlama ve kullanma
- 7-7 Bir SELECT sorgusu oluşturmanın temel ilkelerini açıklayın

Önizleme

Bu bölümde, Yapısal Sorgu Dili'nin (SQL) temellerini öğreneceksiniz. *S-Q-L* veya *devamı* şeklinde telaffuz edilen SQL, kullanıcıların veritabanı ve tablo yapıları oluşturmasını, çeşitli veri işleme ve veri yönetimi türlerini gerçekleştirmesini ve yararlı bilgileri ayıklamak için veritabanını sorgulamasını sağlayan komutlardan oluşur. Tüm ilişkisel DBMS yazılımları SQL'i destekler ve birçok yazılım satıcısı temel SQL komut setine uzantılar.

Oldukça kullanışlı ve güçlü olmasına rağmen SQL, uygulamalar arenasında tek başına durmak için tasarlanmamıştır. SQL ile veri girişi mümkündür ancak veri düzeltmeleri ve eklemelerinde olduğu gibi zordur. SQL'in kendisi menüler, özel rapor formları, kaplamalar, açılır pencereler veya son kullanıcıların genellikle beklediği diğer özellikleri oluşturmaz. Bunun yerine, bu özellikler satıcı tarafından sağlanan geliştirmeler olarak mevcuttur. SQL, veri tanımlama (tablo ve dizin oluşturma) ve veri manipülasyonuna (ekleme, değiştirme) odaklanır,
silme ve veri alma). SQL programcıları için en yaygın görev veri erişimidir. İş gereksinimlerini karşılamak için bir veritabanından veri alma yeteneği, veritabanı uzmanları için en kritik becerilerden biridir. Bu bölümde veri alma konusu ayrıntılı olarak ele alınmaktadır.

Veri Dosyaları ve Mevcut Formatlar

	MS Erişim	Oracle	MS SQL	MySQL
Ch07_SaleCo	Evet	Evet	Evet	Evet
Ch07_ConstructCo	Evet	Evet	Evet	Evet
Ch07_LargeCo	Evet	Evet	Evet	Evet
Ch07_Fact	Evet	Evet	Evet	Evet

Veri Dosyaları cengage.com adresinde mevcuttur

7-1 SQL Temelleri

İdeal olarak, bir veritabanı dili veritabanı ve tablo yapıları oluşturmanıza, temel veri yönetimi işlerini (ekleme, silme ve değiştirme) gerçekleştirmenize ve ham verileri yararlı bilgilere dönüştürmek için tasarlanmış karmaşık sorgular gerçekleştirmenize olanak tanır. Dahası, bir veritabanı dili bu tür temel işlevleri minimum kullanıcı çabasıyla yerine getirmeli ve komut yapısı ve sözdiziminin öğrenilmesi kolay olmalıdır. Son olarak, taşınabilir olmalıdır; , bazı temel standartlara uygun olmalıdır, böylece bir kişi bir RDBMS'den diğerine geçerken temel bilgileri yeniden öğrenmek zorunda kalmaz. SQL bu ideal veritabanı dili gereksinimlerini iyi bir şekilde karşılamaktadır.

SQL fonksiyonları birkaç geniş kategoriye ayrılır:

- Bir *veri manipülasyon dilidir (DML)*. SQL, veritabanı tablolarına veri eklemek, güncellemek, silmek ve almak için komutlar içerir. Bu bölümde öğreneceğiniz veri işleme komutları Tablo 7.1'de listelenmiştir. Bu bölümde, verileri ilginç şekillerde almak için kullanılan komutlara odaklanacaksınız.
- Bir *veri tanımlama dilidir (DDL)*. SQL, tablolar, dizinler ve görünümler gibi veritabanı nesneleri oluşturmak için komutların yanı sıra bu veritabanı nesnelerine erişim haklarını tanımlamak için komutlar içerir. Bölüm 8, Gelişmiş SQL'de öğreneceğiniz bazı yaygın veri tanımlama komutları Tablo 7.2'de listelenmiştir.
- Bir *işlem kontrol dilidir (TCL)*. SQL'deki DML komutları, iş kuralları tarafından tanımlandığı gibi bir veya daha fazla SQL deyiminden oluşan mantıksal bir iş birimi olan bir **işlem** bağlamında yürütülür (bkz. Bölüm 10, İşlem Yönetimi ve Eşzamanlılık Kontrolü). SQL, bu ifadelerin bölünmez bir iş birimi olarak işlenmesini kontrol etmek için komutlar sağlar. Bu komutlar, bir işlemi oluşturan DML komutlarını öğrendikten sonra Bölüm 8'de ele alınacaktır.
- Bu bir *veri kontrol dilidir (DCL)*. Veri kontrol komutları, bir kullanıcıya yalnızca ÜRÜN tablosunu görüntüleme izni vermek ve başka bir kullanıcıya ÜRÜN tablosundaki verileri değiştirme izni vermek gibi veri nesnelerine erişimi kontrol etmek için kullanılır. Yaygın TCL ve DCL komutları Tablo 7.3'te gösterilmektedir.

işlem
Bir veya daha fazla SQL deyiminden oluşan mantıksal bir çalışma birimi.

SQL'i öğrenmek nispeten kolaydır. Temel komut seti 100 kelimeden daha az bir kelime dağarcığına sahiptir. Daha da iyisi, SQL prosedürel olmayan bir dildir: sadece *ne* yapılacağını söylersiniz; *nasıl* yapılacağı konusunda endişelenmenize gerek yoktur. Örneğin, tek bir komut verileri başarılı bir şekilde depolamak ve işlemek için gereken karmaşık tablo yapılarını oluşturur; son kullanıcıların ve programcıların fiziksel veri depolama formatını veya bir SQL komutu yürütüldüğünde gerçekleşen karmaşık faaliyetleri bilmeleri gerekmez.

Amerikan Ulusal Standartlar Enstitüsü (ANSI) standart bir SQL öngörmektedir. ANSI SQL standartları, 150'den fazla ülkenin ulusal standart kuruluşlarından oluşan bir konsorsiyum olan Uluslararası Standardizasyon Örgütü (ISO) tarafından da kabul edilmektedir. Buna rağmen

Tablo 7.1 SQL Veri Manipülasyon Komutları

Komut, Seçenek veya Operatör	Açıklama	Kapalı
SELECT	**Bir veya daha fazla tablo ya da görünümdeki satırlardan öznitelikleri seçer**	**Bölüm 7**
FROM	Verilerin alınacağı tabloları belirtir	Bölüm 7
WHERE	Satırların seçimini koşullu bir ifadeye göre kısıtlar	Bölüm 7
GROUP BY	Seçilen satırları bir veya daha fazla özniteliğe göre gruplar	Bölüm 7
HAVİNG	Gruplandırılmış satırların seçimini bir koşula göre kısıtlar	Bölüm 7
ORDER BY	Seçilen satırları bir veya daha fazla özniteliğe göre sıralar	Bölüm 7
INSERT	**Bir tabloya satır(lar) ekler**	**Bölüm 8**
UPDATE	**Bir tablonun bir veya daha fazla satırındaki bir özniteliğin değerlerini değiştirir**	**Bölüm 8**
DELETE	**Bir tablodan bir veya daha fazla satırı siler**	**Bölüm 8**
Karşılaştırma operatörleri		**Bölüm 7**
=,<,>,<=,>=,<>,!=	Koşullu ifadelerde kullanılır	Bölüm 7
Mantıksal operatörler		**Bölüm 7**
VE/VEYA/DEĞİL	Koşullu ifadelerde kullanılır	Bölüm 7
Özel operatörler	**Koşullu ifadelerde kullanılır**	**Bölüm 7**
BETWEEN	Bir öznitelik değerinin aralık içinde olup olmadığını kontrol eder	Bölüm 7
IN	Bir öznitelik değerinin değer listesindeki herhangi bir değerle eşleşip eşleşmediğini kontrol eder	Bölüm 7
LIKE	Bir öznitelik değerinin verilen bir dize kalıbıyla eşleşip eşleşmediğini kontrol eder	Bölüm 7
IS NULL	Bir öznitelik değerinin null olup olmadığını kontrol eder	Bölüm 7
EXISTS	Bir alt sorgunun herhangi bir satır döndürüp döndürmediğini kontrol eder	Bölüm 7
DISTINCT	Değerleri benzersiz değerlerle sınırlar	Bölüm 7
Toplu fonksiyonlar	**Sütunlarda matematiksel özetler döndürmek için SELECT ile kullanılır**	**Bölüm 7**
COUNT	Belirli bir sütun için boş olmayan değerlere sahip satır sayısını döndürür	Bölüm 7
MIN	Belirli bir sütunda bulunan minimum öznitelik değerini döndürür	Bölüm 7
MAX	Belirli bir sütunda bulunan maksimum öznitelik değerini döndürür	Bölüm 7
SUM	Belirli bir sütun için tüm değerlerin toplamını döndürür	Bölüm 7
AVG	Belirli bir sütun için tüm değerlerin ortalamasını döndürür	Bölüm 7

ANSI/ISO SQL standardına bağlılık genellikle ticari ve devlet sözleşmeli veritabanı spesifikasyonlarında gereklidir, birçok RDBMS satıcısı kendi özel geliştirmelerini ekler. Sonuç olarak, SQL tabanlı bir uygulamayı bazı değişiklikler yapmadan bir RDBMS'den diğerine taşımak nadiren mümkündür.

Bununla birlikte, birçok SQL "lehçesi" olmasına rağmen, aralarındaki farklar küçüktür. İster Oracle, Microsoft SQL Server, MySQL, IBM DB2, Microsoft Access ya da başka bir köklü RDBMS kullanın, bu bölümde sunulan materyali biliyorsanız, bir kullanıcı kılavuzu sizi hızlandırmak için yeterli olacaktır.

7-1a Veri Türleri

ANSI/ISO SQL standardı birçok veri türü tanımlar. Veri türü, bir öznitelikte saklanabilecek türleri hakkında bir belirtimdir. Veri türleri hakkında daha ayrıntılı bir tartışma, varlıkları ve öznitelikleri tablolar ve sütunlar olarak uygulamak için SQL komutlarını keşfedeceğiniz Bölüm 8'e kadar bekleyeceğiz. Ancak, verilerin nasıl alınacağını anlamak için veri türleri hakkında temel bir anlayışa ihtiyaç vardır. Veri türleri, veri alan sorguları etkiler çünkü SQL'in sözdizimindeki küçük farklılıklar ve sorgu sırasında nasıl davrandığı, alınan sütunun veri türüne bağlıdır. Şimdilik, üç temel veri türü olduğunu düşünün: karakter

Tablo 7.2 SQL Veri Tanımlama Komutları

Komut veya Seçenek	Açıklama	Kapalı
CREATE SCHEMA AUTHORIZATION	**Bir veritabanı şeması oluşturur**	**Bölüm 8**
CREATE TABLE	**Kullanıcının veritabanı şemasında yeni bir tablo oluşturur**	**Bölüm 8**
NOT NULL	Bir sütunun boş değerlere sahip olmamasını sağlar	Bölüm 8
UNIQUE	Bir sütunun yinelenen değerlere sahip olmamasını sağlar	Bölüm 8
PRIMARY KEY	Bir tablo için birincil anahtar tanımlar	Bölüm 8
FOREIGN KEY	Bir tablo için yabancı anahtar tanımlar	Bölüm 8
DEFAULT	Bir sütun için varsayılan değer tanımlar (değer verildiğinde)	Bölüm 8
CHECK	Bir öznitelikteki veriler doğrular	Bölüm 8
CREATE INDEX	**Bir tablo için dizin oluşturur**	**Bölüm 8**
CREATE VIEW	**Bir veya daha fazla tablodan satır ve sütunların dinamik bir alt kümesini oluşturur**	**Bölüm 8**
ALTER TABLE	**Bir tablonun tanımını değiştirir (öznitelikler veya kısıtlamalar ekler, değiştirir veya siler)**	**Bölüm 8**
CREATE TABLE AS	**Kullanıcının veritabanı şemasındaki bir sorguyu temel alarak yeni bir tablo oluşturur**	**Bölüm 8**
DROP TABLE	**Bir tabloyu (ve verilerini) kalıcı olarak siler**	**Bölüm 8**
DROP INDEX	**Bir dizini kalıcı olarak siler**	**Bölüm 8**
DROP VİEW	**Bir görünüm kalıcı olarak siler**	**Bölüm 8**

Tablo 7.3 Diğer SQL Komutları

Komut veya Seçenek	Açıklama	Kapalı
İşlem Kontrol Dili		
COMMIT	Veri değişikliklerini kalıcı olarak kaydeder	Bölüm 8
ROLLBACK	Verileri orijinal değerlerine geri yükler	Bölüm 8
Veri Kontrol Dili		
GRANT	Kullanıcıya bir sistem eylemi gerçekleştirme veya bir veri nesnesine erişme izni verir	Bölüm 16
REVOKE	Bir kullanıcıdan daha önce verilmiş bir izin kaldırır	Bölüm 16

veriler, sayısal veriler ve tarih verileri. *Karakter verileri* alfabetik değerler, rakamlar, noktalama işaretleri ve özel karakterler gibi yazdırılabilir karakterlerden oluşur. Karakter verileri genellikle "dize" olarak da adlandırılır çünkü değeri oluşturmak için bir araya getirilmiş bir karakter koleksiyonudur. *Sayısal veriler* rakamlardan oluşur, öyle ki verilerin belirli bir sayısal değeri vardır. *Tarih verileri tarih* ve bazen de zaman değerlerinden oluşur. Karakter verileri telefon numarası veya Sosyal Güvenlik numarası gibi rakamlar içerebilse de, DBMS bu rakamların sayısal değerini tanımaz.

7-1b SQL Sorguları

SQL'in kalbinde sorgu yer alır. Bölüm 1, Veritabanı Sistemleri'nde, bir sorgunun anlık bir soru olduğunu öğrendiniz. Aslında SQL ortamında *sorgu* kelimesi hem soruları hem de eylemleri kapsar. Çoğu SQL sorgusu aşağıdaki gibi soruları yanıtlamak için kullanılır: "Şu anda envanterde tutulan hangi ürünlerin fiyatı 100 doların üzerindedir ve bu ürünlerin her biri için eldeki miktar nedir?" veya "1 Ocak 2020'den bu yana şirketin her bir departmanı tarafından kaç çalışan işe alındı?" Bununla birlikte, birçok SQL sorgusu tablo satırları eklemek silmek ya da tablolardaki öznitelik değerlerini değiştirmek gibi eylemleri gerçekleştirmek için kullanılır. Yine de diğer SQL

sorgular yeni tablolar veya dizinler oluşturur. Bir VTYS için *sorgu*, basitçe yürütülmesi gereken bir SQL deyimidir. Veritabanı ile ilgili işlerin çoğunda, veri almak açık ara en yaygın görev türüdür. Veritabanı uzmanları yalnızca veritabanından nasıl veri alınacağını bilmek zorunda değildir, aynı zamanda neredeyse tüm uygulama programcıları da bu beceriye ihtiyaç duyar.

Veri alma işlemi SQL'de SELECT sorgusu kullanılarak yapılır. Bir tablo üzerinde SELECT komutu çalıştırdığınızda, RDBMS ilişkisel bir tablo ile aynı özelliklere sahip veya daha fazla satırdan oluşan bir küme döndürür. Bu SQL komutlarının çok önemli bir özelliğidir. Varsayılan olarak, çoğu SQL veri işleme komutu tüm bir tablo (ilişki) üzerinde çalışır, bu nedenle SQL komutlarının küme *yönelimli* komutlar olduğu söylenir. Küme **odaklı** bir SQL komutu bir dizi satır üzerinde çalışır. Küme, bir veya daha fazla sütun ve bir veya daha fazla tablodan sıfır veya daha fazla satır içerebilir. Bir SELECT sorgusu hangi verilerin alınacağını ve bu verilerin nasıl filtreleneceğini, toplanacağını ve görüntüleneceğini belirler. Bir SELECT sorgusu, Tablo 7.1'de gösterildiği gibi birçok potansiyel cümleye veya parçaya sahiptir. Bir SELECT sorgusu oluşturmak, yapı taşlarıyla nesneler oluşturmaya benzer. Veritabanı programcısı her bir yapı taşının (cümle) ne yaptığını ve blokların birbirine nasıl uyduğunu anlamalıdır. Daha sonra hangi blokların kullanılacağını planlayabilir ve istenen sonucu üretmek için bu blokları nasıl bir araya getireceğini belirleyebilir.

yönelimli ayarla
Kümeler veya gruplarla ilgilenme veya ilgili olma. İlişkisel modelde, SQL operatörleri küme odaklıdır çünkü aynı anda tüm satır ve sütun kümeleri üzerinde çalışırlar.

Çevrimiçi İçerik
Şekil 7.1'deki veritabanı modeli, www adresinde bulunan Microsoft Access Ch07_SaleCo veritabanında uygulanmıştır. .cengage.com. (Bu veritabanı Şekil 7.1'de yansıtılmayan birkaç ek tablo içermektedir. Bu tablolar yalnızca tartışma amacıyla kullanılmaktadır). Oracle, MySQL ve SQL Server'da bu tabloları oluşturmak için komut dosyaları şunlardır www adresinde de mevcuttur .cengage.com.

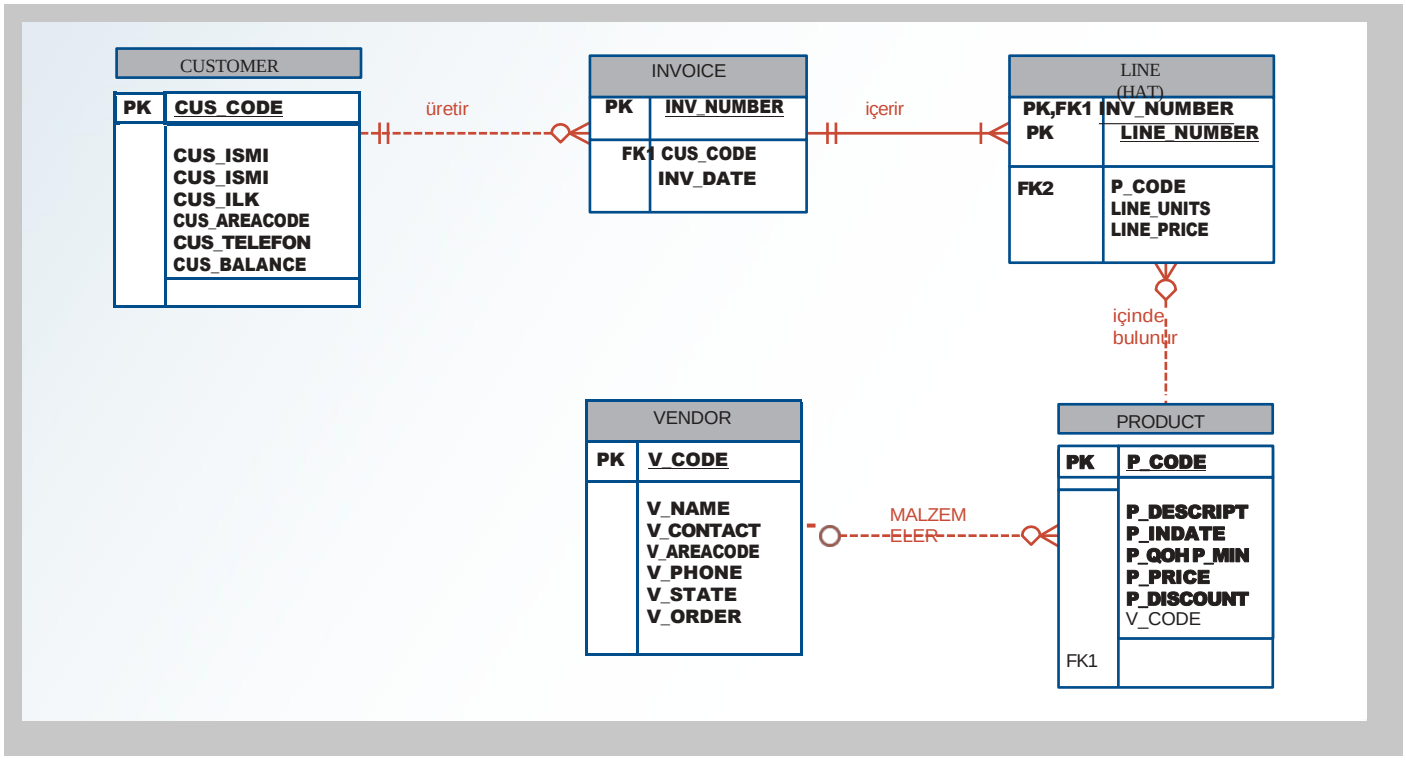
7-1c Veritabanı Modeli

Bu bölümdeki SQL komutlarını göstermek için aşağıdaki tablolardan oluşan basit bir veritabanı kullanılmıştır: MÜŞTERİ, FATURA, HAT, ÜRÜN ve SATICI. Bu veritabanı modeli Şekil 7.1'de gösterilmektedir.

Şekil 7.1'deki veritabanı modeli aşağıdaki iş kurallarını yansıtmaktadır:

- Bir müşteri çok sayıda fatura oluşturabilir. Her fatura bir müşteri tarafından oluşturulur.
- Bir fatura, bir veya daha fazla fatura satırı içerir. Her fatura satırı bir ile ilişkilendirilir.
- Her fatura satırı bir ürüne referans verir. Bir ürün birçok fatura satırında bulunabilir. (Birden fazla müşteriye birden fazla çekiç satabilirsiniz).

Şekil 7.1 Veritabanı Modeli



- Bir satıcı birçok ürün tedarik edebilir. Bazı satıcılar henüz ürün tedarik etmemektedir. Örneğin, bir satıcı listesi potansiyel satıcıları içerebilir.
- Bir ürün satıcı tarafından tedarik ediliyorsa, yalnızca tek bir satıcı tarafından tedarik edilir.
- Bazı ürünler bir satıcı tarafından tedarik edilmez. Örneğin, bazı ürünler kurum içinde üretilebilir veya açık piyasadan satın alınabilir.

Belirtilenler dışında, bölümün geri kalanındaki sorgular için Şekil 7.1'de gösterilen veritabanı modeli kullanılacaktır. Bir ERD veritabanı olarak uygulandığında, her bir varlığın veritabanında bir tablo haline geldiğini ve bir varlık içindeki her bir niteliğin bu tabloda bir sütun haline geldiğini hatırlayın.

NOT

Bu bölüm, tablolardan veri almak için SELECT sorgularına odaklanmaktadır. Bölüm 8, bu tabloların nasıl oluşturulacağını ve verilerin bunlara nasıl yükleneceğini açıklayacaktır. Bu, çoğu giriş seviyesi veritabanı pozisyonunun deneyimini yansıtmaktadır. Veritabanlarıyla çalışan yeni bir çalışan olarak, yeni tablolar oluşturmaya ve verileri değiştirmeye başlamadan önce muhtemelen zaten var olan tablolardan veri almak için oldukça fazla zaman harcayacaksınız.

7-2 Temel SELECT Sorguları

SELECT sorgusundaki her bir cümle belirli bir işlevi yerine getirir. Her bir cümlenin işlevini anlamak, kullanıcıların raporlama ihtiyaçlarını karşılayacak sorgular oluşturma becerilerini geliştirmenin anahtarıdır. Bu bölümde aşağıdaki yan tümceler ele alınacaktır (bu sırada olmasa da).

- SELECT-sorgu tarafından döndürülecek öznitelikleri belirtir
- FROM-verilerin alınacağı tablo(lar)ı belirtir
- WHERE-Veri satırlarından sağlanan kriterlere göre filtreler
- GROUP BY-bir veya daha fazla öznitelikte aynı değerleri paylaşmaya dayalı olarak veri satırlarını koleksiyonlar halinde gruplandırır
- HAVING - GROUP BY cümlesinde oluşturulan gruplar sağlanan kriterlere göre filtreler
- ORDER BY - nihai sorgu sonucu satırlarını bir veya daha fazla özniteliğin değerlerine göre artan veya azalan sırada sıralar.

SQL komutları tek bir satırda bir araya getirilebilse de, karmaşık komut dizileri en iyi şekilde SQL komutu ve komutun bileşenleri arasında boşluk bırakılarak ayrı satırlarda gösterilir. Bu biçimlendirme kuralını kullanmak SQL deyimlerinin bileşenlerini görmeyi çok daha kolay hale getirir, bu da SQL mantığını izlemeyi ve gerekirse düzeltmeler yapmayı kolaylaştırır. Girintilemede kullanılan boşluk sayısı size bağlıdır. Bir SELECT sorgusunun veritabanından veri alabilmesi için en azından bir SELECT sütun listesi ve bir FROM cümlesi gerekir. **SELECT** sütun listesi, Bölüm 3 İlişkisel Veritabanı Modeli'nde anlatıldığı gibi ilişkisel projeksiyonu belirtir. Sütun listesi, sorgu tarafından hangi sütunların alınacağını ve bunların hangi sırayla döndürüleceğini belirtmenize olanak tanır. Yalnızca sütun listesinde belirtilen sütunlar sorgu sonucunda görünür. **FROM** cümlesi, verilerin alınacağı tabloyu belirtmek için kullanılır. Sorguların, Bölüm 3'te tartışıldığı gibi, birden fazla tablodan veri alması yaygındır. Ancak, FROM cümlesi seçeneklerine geçmeden önce sütun listesiyle yapılabilecek şeylere odaklanacaksınız.

SEÇİNİZ
Bir tablodaki tüm satırların veya satırların bir alt kümesinin değerlerini veren bir SQL komutu. SELECT deyimi tablolardan veri almak için kullanılır.

FROM
Verilerin alınacağı tablo veya tabloları belirten bir SQL .

7-3 SELECT İfadesi Seçenekleri

SELECT sorgusu, alınacak sütunları bir sütun listesi olarak belirtir. Bir tablodan veri alan temel SELECT sorgusunun sözdizimi şöyledir:

```
SELECT      sütun listesi
FROM        Masa listesi;
```

Sütun listesi, virgülle ayrılmış bir veya daha fazla özniteliği temsil eder. Programcı tüm sütunların döndürülmesini istiyorsa, yıldız (*) joker karakteri kullanılabilir. **Joker karakter**, diğer karakterler veya komutlar için genel bir ikame olarak kullanılabilen bir semboldür. Yıldız işareti joker karakteri "tüm sütunlar" anlamına gelir. Örneğin, aşağıdaki sorgu ÜRÜN tablosundaki tüm verileri döndürür (bkz. Şekil 7.2).

joker karakter
Genel bir ikame olarak kullanılabilecek bir sembol:
(1) Bir SELECT deyiminin öznitelik listesinde kullanıldığında tüm sütunlar (*) veya
(2) sıfır veya daha fazla karakter bir SQL LIKE cümlesi koşulunda (% ve _).

```
SELECT      *
FROM        ÜRÜN(PRODUCT);
```

Şekil 7.2 Tüm Tabloyu Seçme

P_CODE	P_DESCRIPT	P_INDATE	P_QOH	P_MIN	P_PRICE	P_DISCOUNT	V_CODE
11QER/31	Power painter, 15 psi., 3-nozzle	03-Nov-21	8	5	109.99	0.00	25595
13-Q2/P2	7.25-in. pwr. saw blade	13-Dec-21	32	15	14.99	0.05	21344
14-Q1/L3	9.00-in. pwr. saw blade	13-Nov-21	18	12	17.49	0.00	21344
1546-QQ2	Hrd. cloth, 1/4-in., 2x50	15-Jan-22	15	8	39.95	0.00	23119
1558-QW1	Hrd. cloth, 1/2-in., 3x50	15-Jan-22	23	5	43.99	0.00	23119
2232/QTY	B&D jigsaw, 12-in. blade	30-Dec-21	8	5	109.92	0.05	24288
2232/QWE	B&D jigsaw, 8-in. blade	24-Dec-21	6	5	99.87	0.05	24288
2238/QPD	B&D cordless drill, 1/2-in.	20-Jan-22	12	5	38.95	0.05	25595
23109-HB	Claw hammer	20-Jan-22	23	10	9.95	0.10	21225
23114-AA	Sledge hammer, 12 lb.	02-Jan-22	8	5	14.40	0.05	
54778-2T	Rat-tail file, 1/8-in. fine	15-Dec-21	43	20	4.99	0.00	21344
89-WRE-Q	Hicut chain saw, 16 in.	07-Feb-22	11	5	256.99	0.05	24288
PVC23DRT	PVC pipe, 3.5-in., 8-ft	20-Feb-22	188	75	5.87	0.00	
SM-18277	1.25-in. metal screw, 25	01-Mar-22	172	75	6.99	0.00	21225
SW-23116	2.5-in. wd. screw, 50	24-Feb-22	237	100	8.45	0.00	21231
WR3/TT3	Steel matting, 4'x8'x1/6", .5" mesh	17-Jan-22	18	5	119.95	0.10	25595

Bu sorguda, sütun listesinin tüm sütunların (ve varsayılan olarak tüm satırların) döndürülmesi gerektiğini belirtir. FROM cümlesi, ÜRÜN tablosundaki verilerin kullanılmasını belirtir. Bölüm 3'ten projeksiyonun döndürülen satırları sınırlamadığını hatırlayın. Döndürülen satırları sınırlamak için ilişkisel seçim (veya kısıtlama) kullanılmalıdır. Sütun listesi, bir sonraki sorguda gösterildiği gibi hangi sütunların döndürüleceğini belirtmenize olanak tanır (bkz. Şekil 7.3).

```
SELECT      P_CODE, P_DESCRIPT, P_PRICE, P_QOH
FROM        ÜRÜN;
```

Bu sorgu, verilerin ÜRÜN tablosundan gelmesi gerektiğini ve yalnızca ürün kodu, açıklama, fiyat ve eldeki miktar sütunlarının dahil edilmesi gerektiğini belirtir. Yalnızca istenen sütunların döndürüldüğüne ve sütunların sorguda listelendikleri gibi çıktıda da aynı sırada olduğuna dikkat edin. Sütunları farklı bir sırada görüntülemek için sütun listesindeki sütunların sırasını değiştirmeniz yeterlidir.

Şekil 7.3 Sütun Listesi ile SEÇ

P_CODE	P_DESCRIPT	P_PRICE	P_QOH
11QER/31	Power painter, 15 psi., 3-nozzle	109.99	8
13-Q2/P2	7.25-in. pwr. saw blade	14.99	32
14-Q1/L3	9.00-in. pwr. saw blade	17.49	18
1546-QQ2	Hrd. cloth, 1/4-in., 2x50	39.95	15
1558-QW1	Hrd. cloth, 1/2-in., 3x50	43.99	23
2232/QTY	B&D jigsaw, 12-in. blade	109.92	8
2232/QWE	B&D jigsaw, 8-in. blade	99.87	6
2238/QPD	B&D cordless drill, 1/2-in.	38.95	12
23109-HB	Claw hammer	9.95	23
23114-AA	Sledge hammer, 12 lb.	14.40	8
54778-2T	Rat-tail file, 1/8-in. fine	4.99	43
89-WRE-Q	Hicut chain saw, 16 in.	256.99	11
PVC23DRT	PVC pipe, 3.5-in., 8-ft	5.87	188
SM-18277	1.25-in. metal screw, 25	6.99	172
SW-23116	2.5-in. wd. screw, 50	8.45	237
WR3/TT3	Steel matting, 4'x8'x1/6", .5" mesh	119.95	18

7-3a Sütun Takma Adlarını Kullanma

Bir varlık içindeki özniteliğin tabloda bir sütun olarak uygulandığını hatırlayın. Öznitelik adı bu sütunun adı olur. sütun bir sorguda alındığında, öznitelik adı varsayılan olarak sorgu çıktısında etiket veya sütun başlığı olarak kullanılır. Çıktıda etiket olarak farklı bir adın kullanılmasını istiyorsanız, yeni bir belirtebilirsiniz. Yeni reklam, **takma** reklam olarak adlandırılır. Örneğin, takma adlar aşağıdaki sorguda kullanılır (bkz. Şekil 7.4).

diğer reklam
SQL deyimindeki bir sütun veya tablo için alternatif bir reklam.

```
SEÇİNİZ     P_CODE, P_DESCRIPT AS DESCRIPTION, P_PRICE AS "Birim Fiyat",
            P_QOH QTY
FROM        ÜRÜN;
```

Bu sorguda ve Şekil 7.4'teki çıktısında, P_DESCRIPT özelliğine DESCRIPTION takma adı, P_PRICE özelliğine Unit Price takma adı ve P_QOH özelliğine QTY takma adı verilmiştir. Bu takma adların kullanımıyla ilgili birkaç ilginç nokta vardır:

- Bir sorgudaki tüm sütunların takma ad kullanması gerekmez
- AS isteğe bağlıdır, ancak önerilir
- Boşluk içeren takma adlar bir sınırlayıcı (tırnak) içinde olmalıdır

AS anahtar sözcüğü gerekli değildir, ancak önerilir. Sütun adı ile takma ad arasında bir boşluk varsa, DBMS takma adı doğru şekilde yorumlayacaktır. Ancak, sütun listesi içine formüller ve fonksiyonlar yerleştirmek mümkündür ve genellikle üretilen sütunlar için bir takma ad istersiniz. Bu durumlarda, AS anahtar sözcüğüne sahip olmak sorguyu okumayı ve diğer reklamın formülün bir parçası değil yalnızca bir diğer reklam olduğunu anlamayı çok daha kolay hale getirir. Son olarak, DBMS bir diğer reklamın tek bir sözcük olarak görünmesini bekler. Diğer reklam boşluk içeriyorsa, diğer reklamın nerede başlayıp nerede bittiğini belirtmek için bir sınırlayıcı kullanmanız gerekir. Şekil 7.4'te, boşluk içerdiği için Birim Fiyatı takma adının etrafında çift tırnaklı bir sınırlayıcı kullanılmıştır. Çoğu DBMS ürünü, bir sütun diğer adının etrafında çift tırnak işaretine izin verir.

Şekil 7.4 Sütun Takma Adları ile SELECT

P_CODE	DESCRIPTION	Unit Price	QTY
11QER/31	Power painter, 15 psi., 3-nozzle	109.99	8
13-Q2/P2	7.25-in. pwr. saw blade	14.99	32
14-Q1/L3	9.00-in. pwr. saw blade	17.49	18
1546-QQ2	Hrd. cloth, 1/4-in., 2x50	39.95	15
1558-QW1	Hrd. cloth, 1/2-in., 3x50	43.99	23
2232/QTY	B&D jigsaw, 12-in. blade	109.92	8
2232/QWE	B&D jigsaw, 8-in. blade	99.87	6
2238/QPD	B&D cordless drill, 1/2-in.	38.95	12
23109-HB	Claw hammer	9.95	23
23114-AA	Sledge hammer, 12 lb.	14.40	8
54778-2T	Rat-tail file, 1/8-in. fine	4.99	43
89-WRE-Q	Hicut chain saw, 16 in.	256.99	11
PVC23DRT	PVC pipe, 3.5-in., 8-ft	5.87	188
SM-18277	1.25-in. metal screw, 25	6.99	172
SW-23116	2.5-in. wd. screw, 50	8.45	237
WR3/TT3	Steel matting, 4'x8'x1/6", .5" mesh	119.95	18

NOT

Diğer reklam boşluk içermediğinde bile sütun diğer reklamlarla sınırlayıcıların kullanılması başka amaçlara hizmet edebilir. Bazı DBMS'ler, sütun takma adı bir sınırlayıcı içine yerleştirilmezse, otomatik olarak büyük harflere dönüştürülür. Bu gibi durumlarda, sınırlayıcı kullanmak programcının sütun takma adının büyük harfle yazılmasını kontrol etmesini sağlar. Sınırlayıcıların kullanılması ayrıca bir sütun diğer adının "1" gibi özel bir karakter veya "SELECT" gibi bir SQL anahtar sözcüğü içermesine olanak tanır. Genel olarak, sütun diğer adlarında özel karakterlerin ve SQL anahtar sözcüklerinin kullanılması önerilmez, ancak bu mümkündür.

NOT

MySQL, bu bölümün ilerleyen kısımlarında ele alınan ORDER BY cümlesi gibi, sorgu içinde başka bir yerde bu takma ada başvurmak istiyorsanız, sütun takma adları için sınırlayıcı olarak özel bir sınırlayıcı olan "` " (genellikle standart bir klavyede 1 rakamının solunda bulunur) arka işaretini kullanır.

7-3b Hesaplanmış Sütunları Kullanma

Hesaplanmış bir sütun (hesaplanmış olarak da adlandırılır), Bölüm 4, Varlık İlişkisi Modellemesi'nde açıklandığı gibi türetilmiş bir niteliği temsil eder. Bölüm 4'ten türetilmiş bir niteliğin veritabanında saklanabileceğini ya da saklanamayabileceğini hatırlayın. Türetilmiş özniteliğin saklanmamasına karar verilirse, özniteliğin ihtiyaç duyulduğunda hesaplanması gerekir. Örneğin, şu anda envanterde tutulan her bir ürünün toplam değerini belirlemek istediğinizi. Mantıksal olarak, bu belirleme her ürünün eldeki miktarının geçerli fiyatıyla çarpılmasını gerektirir. Bu görevi aşağıdaki komutla gerçekleştirebilirsiniz:

```
SELECT      P_DESCRIPT, P_QOH, P_PRICE, P_QOH * P_PRICE
FROM        ÜRÜN;
```

SQL komutunun girilmesi Şekil 7.5'te gösterilen çıktıyı oluşturur.

Şekil 7.5 Hesaplanmış Sütun içeren SELECT Deyimi

P_DESCRIPT	P_QOH	P_PRICE	Expr1
Power painter, 15 psi., 3-nozzle	8	109.99	879.92
7.25-in. pwr. saw blade	32	14.99	479.68
9.00-in. pwr. saw blade	18	17.49	314.82
Hrd. cloth, 1/4-in., 2x50	15	39.95	599.25
Hrd. cloth, 1/2-in., 3x50	23	43.99	1011.77
B&D jigsaw, 12-in. blade	8	109.92	879.36
B&D jigsaw, 8-in. blade	6	99.87	599.22
B&D cordless drill, 1/2-in.	12	38.95	467.40
Claw hammer	23	9.95	228.85
Sledge hammer, 12 lb.	8	14.40	115.20
Rat-tail file, 1/8-in. fine	43	4.99	214.57
Hicut chain saw, 16 in.	11	256.99	2826.89
PVC pipe, 3.5-in., 8-ft	188	5.87	1103.56
1.25-in. metal screw, 25	172	6.99	1202.28
2.5-in. wd. screw, 50	237	8.45	2002.65
Steel matting, 4'x8'x1/6", .5" mesh	18	119.95	2159.10

SQL, hesaplanan sütunlardaki tüm geçerli ifadeleri (veya formülleri) kabul eder. Bu formüller, SELECT deyiminin FROM cümlesinde belirtilen tabloların herhangi birindeki özniteliklere uygulanan geçerli matematiksel operatörleri ve fonksiyonları içerebilir. Farklı DBMS ürünleri, hesaplanan sütun için görüntülenen sütun başlıklarına göre değişiklik gösterir.

NOT

MS Access, bir takma ad belirtilmediğinde tüm hesaplanan sütunlara otomatik olarak bir Expr etiketi ekler. (İlk hesaplanan sütun Expr1; ikincisi Expr2 olarak etiketlenir ve bu devam eder). Oracle, hesaplanan sütun için etiket olarak gerçek formül metnini kullanır. Diğer DBMS'ler sütunu başlık etiketi olmadan döndürür.

Çıktıyı daha okunabilir hale getirmek için, hesaplanan alanlar için genellikle bir takma ad kullanılır. Örneğin, önceki SQL ifadesini aşağıdaki gibi yeniden yazabilirsiniz:

```
SEÇİNİZ     P_DESCRIPT, P_QOH, P_PRICE, P_QOH * P_PRICE AS TOTVALUE
FROM        ÜRÜN;
```

Komutun çıktısı Şekil 7.6'da gösterilmektedir.

7-3c Aritmetik Operatörler: Öncelik Kuralı

Öncelik kuralları İşlemlerin gerçekleştirilme sırasını belirleyen temel cebirsel **kurallar.** Örneğin, parantez içindeki işlemler önce, bu nedenle 2 1 (3 3 5) denkleminde çarpma kısmı önce hesaplanır ve doğru cevap 17 olur.

Önceki örnekte gördüğünüz gibi, aritmetik işleçleri Tablo ile birlikte bir sütun listesinde veya koşullu bir ifadede kullanabilirsiniz. Aslında, SQL komutları genellikle Tablo 7.4'te gösterilen aritmetik operatörlerle birlikte kullanılır.

Çarpma sembolünü (*) MS Access gibi bazı SQL uygulamaları tarafından kullanılan joker karakter sembolü ile karıştırmayın. Joker karakter sembolü yalnızca dize karşılaştırmalarında kullanılırken, çarpma sembolü matematiksel prosedürlerde kullanılır.

Nitelikler üzerinde matematiksel işlemler gerçekleştirirken, matematiksel öncelik kurallarını unutmayın. Adından **da** anlaşılacağı gibi, **öncelik** kuralları öznitelikler arasındaki

Şekil 7.6 Hesaplanmış Sütun ve Diğer Ad İçeren SELECT Deyimi

P_DESCRIPT	P_QOH	P_PRICE	TOTVALUE
Power painter, 15 psi., 3-nozzle	8	109.99	879.92
7.25-in. pwr. saw blade	32	14.99	479.68
9.00-in. pwr. saw blade	18	17.49	314.82
Hrd. cloth, 1/4-in., 2x50	15	39.95	599.25
Hrd. cloth, 1/2-in., 3x50	23	43.99	1011.77
B&D jigsaw, 12-in. blade	8	109.92	879.36
B&D jigsaw, 8-in. blade	6	99.87	599.22
B&D cordless drill, 1/2-in.	12	38.95	467.40
Claw hammer	23	9.95	228.85
Sledge hammer, 12 lb.	8	14.40	115.20
Rat-tail file, 1/8-in. fine	43	4.99	214.57
Hicut chain saw, 16 in.	11	256.99	2826.89
PVC pipe, 3.5-in., 8-ft	188	5.87	1103.56
1.25-in. metal screw, 25	172	6.99	1202.28
2.5-in. wd. screw, 50	237	8.45	2002.65
Steel matting, 4'x8'x1/6", .5" mesh	18	119.95	2159.10

Tablo 7.4 Aritmetik Operatörler

Operatör	Açıklama
+	ADD (Ekle)
–	SUBTRACT (Çıkarma)
*	MULTIPLY (Çarpma)
/	DIVIDE (Bölmek)
^	Raise to the power of (some applications use ** instead of ^) yükselt (bazı uygulamalar^ yerine ** kullanır)

hesaplamaların tamamlanma sırası. Örneğin, aşağıdaki hesaplama dizisinin sırasına dikkat edin:

1. Parantez içindeki işlemleri gerçekleştirin.
2. Güç işlemlerini gerçekleştirin.
3. Çarpma ve bölme işlemlerini gerçekleştirin.
4. Toplama ve çıkarma işlemlerini gerçekleştirin.

Öncelik kurallarının uygulanması boyut 8 + 2 *5 = 8 +10 = 18 olduğunu söyler, ancak (8 + 2) * 5 = 10 *5 = 50. Benzer şekilde, 4 + 5^2 *3 = 4 + 25 *3 = 79, ancak (4 + 5)^2 *3 = 81 *3 = 243, (4 + 5^2) * ile ifade edilen işlem ise (4 + 25) *3 = 29 *3 = 87 cevabını verir.

7-3d Tarih Aritmetiği

Sütun listesindeki tarih verileri, hesaplanan alanlarda kullanıldığında ilginç olabilir. Dahili olarak, DBMS bir tarih değerini sayısal bir formatta saklar. Ayrıntılar karmaşık olsa da, temelde bir tarih bir gün sayısı, yani tanımlanmış bir noktadan bu yana geçen gün sayısı olarak saklanır. Tarihteki bu noktanın tam olarak ne olduğu bir VTYS'den diğerine değişir. Ancak, değerler gün sayısı olarak saklandığından, bir sorguda tarih aritmetiği gerçekleştirmek mümkündür. Örneğin, bazı DBMS'lerde bugünün tarihi "250.000" gün sayısı ise, yarın 250.001" ve dün "249.999" olacaktır. Tarih veri türünde depolanan bir tarihe bir sayı eklendiğinde ya da çıkarıldığında, belirtilen sayı kadar tarih döndürülür.

verilen tarihten itibaren günler. Bir tarih değerinin diğerinden çıkarılması gün sayısını verir bu tarihler arasında.

Bir yöneticinin tüm ürünlerin, alındıkları tarihlerin ve garanti sona erme tarihinin (ürünün alınmasından itibaren 90 gün) bir listesini istediğini varsayalım. Bu listeyi oluşturmak için aşağıdaki sorguyu yaparsınız:

```
SEÇİNİZ     P_CODE, P_INDATE, P_INDATE 1 90 AS EXPDATE
FROM        ÜRÜN;
```

Bu sorgu, tek bir sorguda bir takma ad ve tarih aritmetiği ile hesaplanmış bir sütun kullanır. DBMS'nin ayrıca veritabanı sunucusundaki geçerli tarihi döndüren bir işlevi vardır, bu da sorgunun içeriğini her gün değiştirmek zorunda kalmadan geçerli tarihe başvuran sorgular yazmayı mümkün kılar. Örneğin, sırasıyla MS Access, SQL Server ve MySQL'deki DATE(), GETDATE() ve CURDATE() işlevleri ve Oracle'daki SYSDATE anahtar sözcüğünün tümü geçerli tarihi alır. Bir yönetici ürünlerinin ve garanti bitiş tarihlerinin bir listesini istiyorsa, Oracle'daki sorgu şöyle olacaktır:

```
SEÇİNİZ     P_CODE, P_INDATE, SYSDATE - 90 AS CUTOFF
FROM        ÜRÜN;
```

Bu sorguda, çıktı geçerli tarihe göre değişecektir. Bu fonksiyonları bir tarih değişmezinin beklendiği her yerde kullanabilirsiniz.

7-3e Benzersiz Değerleri Listeleme

Şu anda ÜRÜN tablosunda kaç *farklı* satıcı temsil ediliyor? Tablo birkaç bin satır içeriyorsa ve satıcı kodlarını manuel olarak elemek zorundaysanız, basit bir listeleme (SELECT) çok kullanışlı değildir. Neyse ki, SQL'in **DISTINCT** cümlesi yalnızca birbirinden farklı olan değerlerin bir listesini oluşturur. Örneğin, komut

DISTINCT
Birbirinden farklı değerlerin bir listesini üreten bir SQL cümlesi.

```
SEÇİNİZ     DISTINCT V_CODE
FROM        ÜRÜN;
```

Şekil 7.7'de gösterildiği gibi, PRODUCT tablosunda yalnızca farklı satıcı kodlarını (V_CODE) verir. DISTINCT anahtar sözcüğü sorguda yalnızca bir kez, SELECT anahtar sözcüğünün hemen ardından görünür. İlk çıktı satırının bir null gösterdiğine dikkat edin. Ürün şirket içinde geliştirildiyse veya doğrudan üreticiden satın alındıysa, satırlar V_CODE özniteliği için bir null içerebilir. Bölüm 3'te tartışıldığı gibi, null'lar sorunlu olabilir çünkü iş ortamında null'un ne anlama geldiğini bilmek zordur. Null'lar SQL kodu yazarken de sorun yaratabilir. Farklı operatörler ve fonksiyonlar null'ları farklı şekilde ele alır. Örneğin, DISTINCT anahtar sözcüğü null'u bir değer olarak kabul eder ve tüm null'ları aynı değer olarak kabul eder. İlerleyen bölümlerde, null'ları yok sayan fonksiyonlarla karşılaşacak ve tüm null'ları farklı olarak değerlendiren karşılaştırmalar göreceksiniz. Bir SQL geliştiricisi olarak, null değerlerin yazdığınız kod tarafından nasıl ele alınacağını anlamanız gerekir.

Şekil 7.7 ÜRÜN Tablosundaki Farklı V_CODE Değerlerinin Bir Listesi

V_CODE
21225
21231
21344
23119
24288
25595

7-4 FROM Cümle Seçenekleri

Sorgunun cümlesi, verilerin alınacağı tablo veya tabloları belirtir. Aşağıdaki sorguda, veriler yalnızca ÜRÜN tablosundan alınmaktadır.

```
SEÇİNİZ     P_CODE, P_DESCRIPT, P_INDATE, P_QOH, P_MIN, P_PRICE,
            P_DISCOUNT, V_CODE
FROM        ÜRÜN;
```

Sorgunun geri kalanında yalnızca FROM cümlesinde belirtilen tablodaki sütunlar kullanılabilir. Bu, sorgunun yazıldığı veri modelini anlamanın önemli olduğu anlamına gelir. Örneğin, bir kullanıcının Şekil 7.1'de gösterilen veri modelinden tüm satın alımlar için fatura numarasını, ürün kodunu ve satın alınan birim sayısını bilmek istediğini düşünün. İlk başta, bu isteği karşılamak için INVOICE, PRODUCT ve LINE tablolarına ihtiyacınız olduğunu düşünebilirsiniz. Ancak, veri modelinin dikkatli bir şekilde incelenmesi, gerekli tüm verilerin (inv_num, p_code ve line_units) yalnızca LINE tablosu kullanılarak elde edilebileceğini ortaya koymaktadır. Bu nedenle, sorgu aşağıda gösterildiği gibi yazılabilir ve sonuçlar Şekil 7.8'de gösterildiği gibi olur.

```
SEÇİNİZ     INV_NUM, P_CODE, LINE_UNITS
FROM        LINE;
```

INV_NUMBER	P_CODE	LINE_UNITS
1001	13-Q2/P2	1
1001	23109-HB	1
1002	54778-2T	2
1003	2238/QPD	1
1003	1546-QQ2	1
1003	13-Q2/P2	5
1004	54778-2T	3
1004	23109-HB	2
1005	PVC23DRT	12
1006	SM-18277	3
1006	2232/QTY	1
1006	23109-HB	1
1006	89-WRE-Q	1
1007	13-Q2/P2	2
1007	54778-2T	1
1008	PVC23DRT	5
1008	WR3/TT3	3
1008	23109-HB	1

FROM cümlesindeki tablo, sorgunun geri kalanı için temel oluşturur. Sorgunun geri kalanı için kullanılabilir olacak verileri tanımlar. Gelecek bölümde, sorgu tarafından döndürülen satırları sınırlama yöntemleri ele alınacaktır. Döndürülen satırları sınırlamak için herhangi bir ölçüt de FROM içinde belirtilen tablodaki sütunlarla kısıtlanır. Örneğin, kullanıcı önceki sorgunun sonuçlarının yalnızca %50 $'dan daha yüksek bir fiyatla (line_price) satılan ürünleri içerecek şekilde sınırlandırılmasını isterse, bu kısıtlamayı önceki sorguya dahil edebilirsiniz. Ancak, kullanıcı önceki sorgunun sonuçlarının yalnızca geçerli fiyatı (p_price) $50'dan yüksek olan ürünleri içerecek şekilde sınırlandırılmasını isterse, FROM cümlesini değiştirmeden bu kısıtlamayı dahil edemezsiniz çünkü kısıtlama için gerekli öznitelik (p_price) LINE tablosunda bulunmamaktadır.

Birçok iş sorusu, tek bir tablo kullanılarak alınan veriler kullanılarak yanıtlanabilir FROM cümlesinde. Ancak, diğer sorular birden fazla tablo içeren daha karmaşık bir FROM cümlesi gerektirir. FROM cümlesine birden fazla tablo dahil etmek, FROM içinde tabloların bir listesini sağlamak kadar basit değildir. Tabloların bir listesini eklemek, Bölüm 3'te tartışıldığı gibi bir Kartezyen çarpım oluşturacaktır. Bölüm 3'te de belirtildiği gibi, bir Kartezyen çarpım bir iş sorusu için gereken doğru sonuçları neredeyse hiçbir zaman üretmeyecektir. Bunun yerine, Tablolar Bölüm 3'te de ele alınan bir tür JOIN işlemi kullanılarak birleştirilmelidir. Hatırlayacağınız gibi, veritabanlarında birçok birleştirme türü kullanılır ve doğru birleştirme türünü seçmek ve bunun için sözdizimini yazmak biraz göz korkutucu olabilir. JOIN işlemlerinin kullanımıyla ilgili daha ileri düzeydeki konuların tam bir açıklaması bu bölümün ilerleyen kısımlarında yer almaktadır.

7-5 ORDER BY Cümle Seçenekleri

ORDER BY

Çıktıyı sıralamak için yararlı olan bir SQL cümlesi bir SELECT sorgusunun (örneğin, artan veya azalan sırada).

ORDER BY cümlesi özellikle listeleme sırası sizin için önemli olduğunda kullanışlıdır. Sözdizimi

şöyledir: SEÇİNİZ *sütun listesi*
FROM *tablelist*
[ORDER BY *sütun listesi* [ASC | DESC]];

Sıra türünü (artan veya azalan) bildirme seçeneğiniz olmasına rağmen, varsayılan sıra artan sıradır. Örneğin, ÜRÜN tablosunun içeriğinin P_PRICE'a göre artan sırada listelenmesini istiyorsanız, aşağıdaki komutu kullanın:

```
SEÇİNİZ     P_CODE, P_DESCRIPT, P_QOH, P_PRICE
FROM        ÜRÜN
ORDER BY    P_PRICE;
```

Çıktı Şekil 7.9'da gösterilmektedir. ORDER BY'nin artan bir fiyat listesi verdiğine dikkat edin. Şekil 7.9'daki listeleme ile daha önce Şekil 7.9'da gösterilen gerçek tablo içeriği karşılaştırıldığında 7.2, Şekil 7.9'da ilk olarak en düşük fiyatlı ürünün listelendiğini, ardından bir sonraki en düşük fiyatlı ürünün geldiğini ve bu şekilde ettiğini görebilirsiniz. Ancak, ORDER BY sıralanmış bir çıktı üretse de, gerçek tablo içeriği ORDER BY işleminden etkilenmez.

Şekil 7.9 Fiyata Göre Artan Sıralı Ürünler

P_CODE	P_DESCRIPT	P_QOH	P_PRICE
54778-2T	Rat-tail file, 1/8-in. fine	43	4.99
PVC23DRT	PVC pipe, 3.5-in., 8-ft	188	5.87
SM-18277	1.25-in. metal screw, 25	172	6.99
SW-23116	2.5-in. wd. screw, 50	237	8.45
23109-HB	Claw hammer	23	9.95
23114-AA	Sledge hammer, 12 lb.	8	14.40
13-Q2/P2	7.25-in. pwr. saw blade	32	14.99
14-Q1/L3	9.00-in. pwr. saw blade	18	17.49
2238/QPD	B&D cordless drill, 1/2-in.	12	38.95
1546-QQ2	Hrd. cloth, 1/4-in., 2x50	15	39.95
1558-QW1	Hrd. cloth, 1/2-in., 3x50	23	43.99
2232/QWE	B&D jigsaw, 8-in. blade	6	99.87
2232/QTY	B&D jigsaw, 12-in. blade	8	109.92
11QER/31	Power painter, 15 psi., 3-nozzle	8	109.99
WR3/TT3	Steel matting, 4'x8'x1/6", .5" mesh	18	119.95
89-WRE-Q	Hicut chain saw, 16 in.	11	256.99