- Ortaya çıkan Tablo 3'e dahil edilmek için, paylaşılmayan sütundaki (CUS_CODE) bir değer Tablo 2'deki her değerle ilişkilendirilmelidir.
- 123456, 234567 ve 567890 numaralı ürünlerin tamamıyla ilişkili tek müşteriler 10030 ve 12550 numaralı müşterilerdir.

Not

DIVIDE operatörü bölme sembolü 4 ile gösterilir. R ve S olmak üzere iki bağıntı verildiğinde, bunların BÖLÜNMESİ r 4 s olarak yazılacaktır.

3-5 Veri Sözlüğü ve Sistem Kataloğu

veri sözlüğü
Veriler hakkındaki meta verileri depolayan bir DBMS bileşenidir. Böylece, veri sözlüğü veri tanımını da içerir
özellikleri ve ilişkileri gibi. Bir veri
Sözlük, VTYSnin dışında bulunan verileri de içerebilir. *Bilgi kaynağı sözlüğü* olarak da bilinir. Ayrıca bkz. *aktif veri sözlüğü*, *meta veri* ve *pasif veri sözlüğü*.

Veri sözlüğü, kullanıcı ve tasarımcı tarafından oluşturulan veritabanındaki tüm tabloların ayrıntılı bir tanımını sağlar. Böylece veri sözlüğü, sistemdeki her tablo için en azından tüm öznitelik adlarını ve özelliklerini içerir. Kısacası, veri sözlüğü metadata- veri hakkında veri içerir. Şekil 3.3'te sunulan küçük veritabanını kullanarak, veri sözlüğünü Tablo 3.6'da gösterildiği gibi hayal edebilirsiniz.

Not

Tablo 3.6'daki veri sözlüğü, varlıkların, özniteliklerin ve ilişkilerin *insan* bakış açısına bir örnektir. Bu veri sözlüğünün amacı, veritabanı tasarım ve uygulama ekiplerinin tüm üyelerinin aynı tablo ve öznitelik adlarını ve özelliklerini kullanmasını sağlamaktır. VTYS'nin dahili olarak depolanan veri sözlüğü, ilişki türleri, varlık ve referans bütünlüğü kontrolleri ve uygulamaları ile dizin türleri ve bileşenleri hakkında ek bilgiler içerir. Bu ek bilgiler veritabanı uygulama aşamasında oluşturulur.

Veri sözlüğü bazen "veritabanı tasarımcısının veritabanı" olarak tanımlanır çünkü tablolar ve yapıları hakkındaki tasarım kararlarını kaydeder.

sistem kataloğu
Bir veritabanındaki tüm nesneleri tanımlayan ayrıntılı bir sistem veri sözlüğü.

Veri sözlüğü gibi, sistem kataloğu da meta veriler içerir. **Sistem kataloğu**, tablo adları, bir tablonun oluşturucusu ve oluşturulma tarihi, her tablodaki sütun sayısı, her sütuna karşılık gelen veri türü, dizin dosya adları, dizin oluşturucuları, yetkili kullanıcılar ve erişim ayrıcalıkları hakkındaki veriler dahil olmak üzere veritabanındaki tüm nesneleri tanımlayan ayrıntılı bir sistem veri sözlüğü olarak tanımlanabilir. Sistem kataloğu gerekli tüm veri sözlüğü bilgilerini içerdiğinden, *sistem kataloğu* ve *veri sözlüğü* terimleri genellikle birbirlerinin yerine kullanılır. Aslında, mevcut ilişkisel veritabanı yazılımı genellikle yalnızca tasarımcının veri sözlüğü bilgilerinin türetilebileceği bir sistem kataloğu sağlar. Sistem kataloğu aslında tabloları kullanıcı/tasarımcı tarafından oluşturulan veritabanı özelliklerini ve içeriklerini saklayan sistem tarafından oluşturulmuş bir veritabanıdır. Bu nedenle, sistem kataloğu tabloları tıpkı kullanıcı/tasarımcı tarafından oluşturulan herhangi bir tablo gibi sorgulanabilir.

eşsesli
Farklı nitelikleri etiketlemek için aynı adın kullanılması. Eşsesli isimlerden genellikle kaçınılmalıdır. Ayrıca bkz. *eşanlamlı*.

Gerçekte, sistem kataloğu otomatik olarak veritabanı dokümantasyonu üretir. Veritabanına yeni tablolar eklendikçe, bu dokümantasyon RDBMS'nin eşanlamlıları ve eşanlamlıları kontrol etmesini ve ortadan kaldırmasını da sağlar. Genel anlamda **eş** anlamlı sözcükler, *boar* ve *bore* gibi farklı anlamlara sahip benzer sesli sözcükler ya da *fair* gibi farklı anlamlara sahip sözcüklerdir (bazı bağlamlarda "adil", bazılarında ise "festival" anlamına gelir). Veritabanı bağlamında, eşsesli kelimesi farklı öznitelikleri etiketlemek için aynı adın kullanıldığını gösterir. Örneğin, CUSTOMER tablosundaki bir müşteri adı niteliğini etiketlemek için C_NAME kullanabilir ve CONSULTANT tablosundaki bir danışman adı niteliğini etiketlemek için C_NAME kullanabilirsiniz. Karışıklığı azaltmak için, veritabanı eşanlamlılarından kaçınmalısınız; veri sözlüğü bu konuda çok yararlıdır.

Tablo 3.6 Örnek Bir Veri Sözlüğü

Tablo Adı	Öznitelik Adı	İçindekiler	Tip	Biçim	Menzil	Gerekli	PK VEYA FK	FK Referanslı Tablo
MÜŞTERİ	CUS_CODE	Müşteri hesap kodu	CHAR(5)	99999	10000-99999	Y	PK	
	CUS_LNAME	Müşteri soyadı	VARCHAR(20)	Xxxxxxxx		Y		
	CUS_FNAME	Müşteri adı	VARCHAR(20)	Xxxxxxxx		Y		
	CUS_INITIAL	İlk müşteri	CHAR(1)	X				
	CUS_RENEW_DATE	Müşteri sigortası yenileme tarihi	TARİH	dd-mmm-yyyy				
	AGENT_CODE	Temsilci kodu	CHAR(3)	999			FK	AJAN
AJAN	AGENT_CODE	Temsilci kodu	CHAR(3)	999		Y	PK	
	AGENT_AREACODE	Acente alan kodu	CHAR(3)	999		Y		
	ACENTE_TELEFONU	Acente telefon numarası	CHAR(8)	999-9999		Y		
	AGENT_LNAME	Temsilci soyadı	VARCHAR(20)	Xxxxxxxx		Y		
	AGENT_YTD_SLS	Acente yılbaşından bugüne satışları	SAYI(9,2)	9,999,999.99				
FK	5 Yabancı anahtar							
PK	5 Birincil anahtar							
CHAR	5 Sabit karakter uzunluğunda veri (1- 255 karakter)							
VARCHAR	5 Değişken karakter uzunluğunda veri (1- 2.000 karakter)							
NUMARA	5 Sayısal veriler. NUMBER (9,2), ondalık basamağın sağındaki iki basamak da dahil olmak üzere en fazla dokuz basamaklı sayıları belirtmek için kullanılır. Bazı RDBMS'ler yerine NUMERIC veya DECIMAL kullanır ve diğerleri MONEY veya CURRENCY veri türünün kullanımına izin verir.							

Not

Telefon alan kodları her zaman 0-9 rakamlarından oluşur, ancak alan kodları aritmetik olarak kullanılmadığından, en verimli şekilde karakter verileri olarak saklanırlar. Ayrıca, alan kodları her zaman üç basamaktan oluşur. Bu nedenle, alan kodu veri tipi CHAR(3) olarak tanımlanır. Öte yandan, isimler standart bir uzunluğa uymaz. Bu nedenle, müşteri ilk adları VARCHAR(20) olarak tanımlanır ve adları saklamak için 20 karaktere kadar kullanılabileceğini gösterir. Karakter verileri sola hizalı olarak gösterilir.

eşanlamlı
Bir varlık, bir nitelik veya bir ilişki gibi aynı nesneyi tanımlamak için farklı adların kullanılması; eşanlamlılardan genellikle kaçınılmalıdır. Ayrıca bkz. *eşsesli*.

Bir veritabanı bağlamında, **eşanlamlı**, eşseslinin tersidir ve aşağıdakilerin kullanımını gösterir Aynı niteliği tanımlamak için farklı isimler. Örneğin, *car* ve *auto* aynı nesneye atıfta bulunur. Mümkün olduğunca eş anlamlı kelimelerden kaçınılmalıdır.

3-6 İlişkisel Veritabanındaki İlişkiler

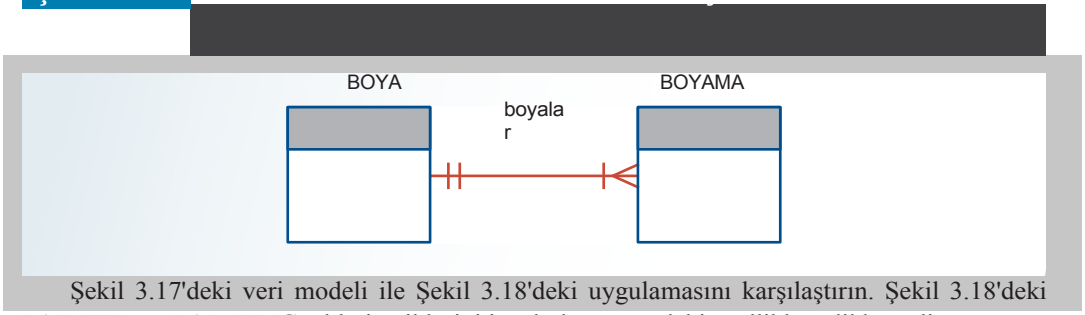
İlişkisel veritabanlarının tablolar arasındaki ilişkileri uygulamak için ortak nitelikler kullandığını zaten biliyorsunuz. Bu ortak nitelikler yabancı anahtarlar kullanılarak oluşturulur; yani bir tablonun birincil anahtarının başka bir tabloya yerleştirilmesi Sorun, hangi birincil anahtarın yabancı anahtar olarak kullanılacağıdır. Şekil 3.2'deki PRODUCT ve VENDOR arasındaki ilişkide, VENDOR'ın birincil anahtarı PRODUCT'a yabancı anahtar olarak yerleştirilmiştir. Ancak, bunun yerine PRODUCT'un birincil anahtarı VENDOR'a yerleştirilebilir miydi? İlişkisel modelde ortak bir nitelik oluşturmak için yabancı anahtar olarak hangi birincil anahtarın kullanılacağını belirlemek ilişkinin sınıflandırılmasına dayanır. İlişkilerin bire-bir (1:1), bire-çok (1:M) ve çoka-çok (M:N veya M:M) olarak sınıflandırıldığını hatırlayın. Bu bölüm, veritabanı tasarımları geliştirmeye başladığınızda bunları doğru şekilde uygulamanıza yardımcı olmak için bu ilişkileri daha ayrıntılı olarak incelemekte ve doğru yabancı anahtar yerleşimini nasıl belirleyeceğinizi göstermektedir. Bu bölüm aşağıdaki noktalara odaklanmaktadır:

- 1:M ilişkisi ilişkisel modelleme idealidir. Bu nedenle, bu ilişki türü herhangi bir ilişkisel veritabanı tasarımında norm olmalıdır.
- 1:1 ilişkisi herhangi bir ilişkisel veritabanı tasarımında nadiren görülmelidir.
- M:N ilişkileri ilişkisel modelde bu şekilde uygulanamaz. Bu bölümün ilerleyen kısımlarında, herhangi bir M:N ilişkisinin nasıl iki 1:M dönüştürülebileceğini göreceksiniz.

3-6a 1:M İlişkisi

1:M ilişkisi ilişkisel veritabanları için normdur. Böyle bir ilişkinin nasıl modellendiğini ve uygulandığını görmek için Şekil 3.17'de gösterilen PAINTER ve PAINTING örneğini ele alalım.

Şekil 3.17



Şekil 3.17'deki veri modeli ile Şekil 3.18'deki uygulamasını karşılaştırın. Şekil 3.18'deki PAINTER ve PAINTING tablo içeriklerini incelerken aşağıdaki özelliklere dikkat edin:

- Her resim bir ve yalnızca bir ressam tarafından yapılmıştır, ancak her ressam birçok resim yapmış olabilir. Ressam 123'ün (Georgette P. Ross) PAINTING tablosunda kayıtlı üç eseri olduğuna dikkat edin.
- PAINTER tablosundaki herhangi bir satır için PAINTING tablosunda yalnızca bir satır vardır, ancak PAINTER tablosundaki herhangi bir satır için PAINTING tablosunda birçok satır olabilir.

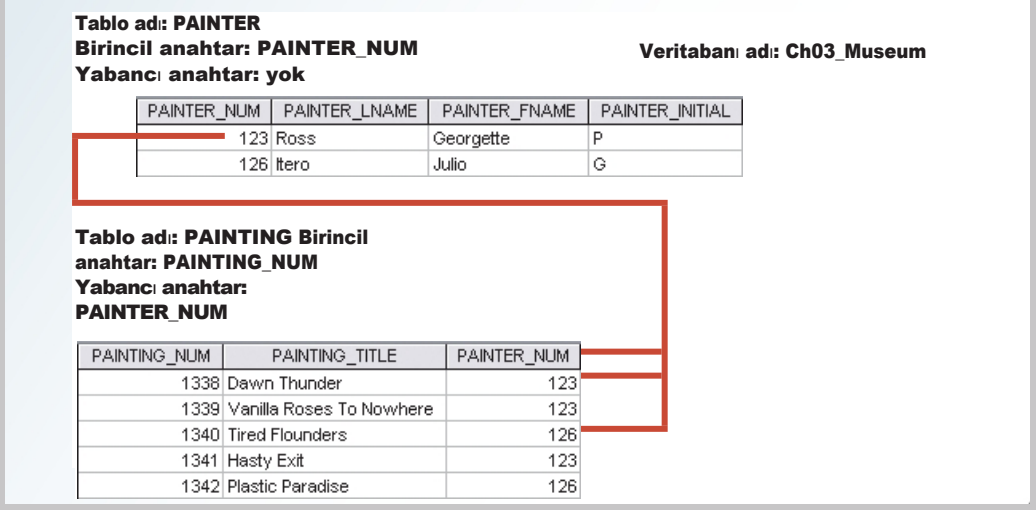
Yabancı anahtarın yerleşimi 1:M ilişkisinin etkili bir şekilde uygulanması için kritik öneme sahiptir. Bu bölümün başından itibaren, ilişkisel bir tablonun özelliklerinden birinin her hücrenin yalnızca tek bir değer içerebilmesi olduğunu hatırlayın. Her tablo yalnızca bir değerle ilişkilendirildiğinden

bir ressam, PAINTER_NUM'u PAINTING'e yerleştirmek, her resmin saklanacak bir ressam numarasına ve bir numaranın saklanacağı bir yere sahip olduğu bir senaryo ortaya çıkarır. Mükemmel! PAINTING_NUM öğesini PAINTER içine yerleştirmiş olsaydınız, 123 numaralı ressamın üç resim numarasını (1338, 1339 ve 1341) saklaması gerekirdi, ancak bunları saklamak için yalnızca bir hücre olurdu. Bu, ilişkisel tablolar için kuralları ihlal eder ve uygulanabilir bir çözüm olmazdı.

Not

Bir-çok (1:M) ilişkisi, ilişkisel modelde *"1" tarafının birincil anahtarının "çok" tarafının tablosuna yabancı anahtar olarak* yerleştirilmesiyle kolayca uygulanabilir.

Şekil 3.18 PAINTER ve PAINTING arasında Uygulanan 1:M İlişkisi

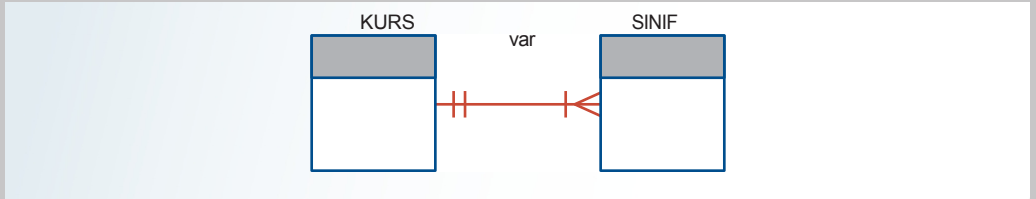


1:M ilişkisi herhangi bir veritabanı ortamında bulunur. Tipik bir kolej veya üniversitedeki öğrenciler, her KURS'un birçok SINIF oluşturabileceğini, ancak her SINIF'ın yalnızca bir KURS'a atıfta bulunduğunu keşfedeceklerdir. Örneğin, bir Muhasebe II dersi iki sınıf oluşturabilir: biri Pazartesi, Çarşamba ve Cuma günleri (MWF) sabah 10:00 ile 10:50 arasında ve diğeri Perşembe günleri (Th) akşam 18:00 ile 20:40 arasında:

- Her KURS'un birçok SINIFI olabilir, ancak her SINIF yalnızca bir KURS'a referans verir.
- CLASS herhangi bir satır için COURSE tablosunda yalnızca bir satır olacaktır, ancak COURSE tablosundaki herhangi bir satır için CLASS tablosunda birçok satır olabilir.

Şekil 3.19, COURSE ve CLASS arasındaki 1:M ilişkisi için varlık ilişki modelini (ERM) eşler.

Şekil 3.19 KURS ve SINIF arasındaki 1:M İlişkisi



Şekil 3.20 DERS ve SINIF arasında Uygulanan 1:M İlişkisi

Tablo adı: COURSE
Birincil anahtar:
CRS_CODE Yabancı anahtar: yok

Veritabanı adı: Ch03_TinyCollege

CRS_CODE	DEPT_CODE	CRS_DESCRIPTION	CRS_CREDIT
ACCT-211	ACCT	Accounting I	3
ACCT-212	ACCT	Accounting II	3
CIS-220	CIS	Intro. to Microcomputing	3
CIS-420	CIS	Database Design and Implementation	4
QM-261	CIS	Intro. to Statistics	3
QM-362	CIS	Statistical Applications	4

Tablo adı: CLASS Birincil anahtar: CLASS_CODE
Yabancı anahtar: CRS_CODE

CLASS_CODE	CRS_CODE	CLASS_SECTION	CLASS_TIME	CLASS_ROOM	PROF_NUM
10012	ACCT-211	1	MWF 8:00-8:50 a.m.	BUS311	105
10013	ACCT-211	2	MWF 9:00-9:50 a.m.	BUS200	105
10014	ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
10015	ACCT-212	1	MWF 10:00-10:50 a.m.	BUS311	301
10016	ACCT-212	2	Th 6:00-8:40 p.m.	BUS252	301
10017	CIS-220	1	MWF 9:00-9:50 a.m.	KLR209	228
10018	CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
10019	CIS-220	3	MWF 10:00-10:50 a.m.	KLR209	228
10020	CIS-420	1	W 6:00-8:40 p.m.	KLR209	162
10021	QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114
10022	QM-261	2	TTh 1:00-2:15 p.m.	KLR200	114
10023	QM-362	1	MWF 11:00-11:50 a.m.	KLR200	162
10024	QM-362	2	TTh 2:30-3:45 p.m.	KLR200	162

KURS ve SINIF arasındaki 1:M ilişkisi Şekil 3.20'de daha ayrıntılı olarak gösterilmiştir. Şekil 3.20'yi kullanarak, bazı önemli terminolojiyi gözden geçirmek için bir dakikanızı ayırın. CLASS tablosundaki CLASS_ CODE'un her satırı benzersiz bir şekilde tanımladığına dikkat edin. Bu nedenle, CLASS_CODE birincil anahtar olarak seçilmiştir. Ancak, CRS_CODE ve CLASS_SECTION kombinasyonu da sınıf tablosundaki her satırı benzersiz bir şekilde tanımlayacaktır. Başka bir deyişle, CRS_CODE ve CLASS_SECTION'dan oluşan *bileşik* bir aday anahtardır. Herhangi bir *aday anahtarın* not-null ve unique kısıtlamalarına sahip olması gerekir. (Bunun nasıl yapıldığını öğrendiğinizde göreceksiniz SQL Bölüm 8'de).

Örneğin, Şekil 3.18'de PAINTER tablosunun birincil anahtarı PAINTER_NUM'un yabancı olarak PAINTING tablosuna dahil edildiğine dikkat edin. Benzer şekilde, Şekil 3.20'de, COURSE tablosunun birincil anahtarı CRS_CODE, yabancı anahtar olarak CLASS tablosuna dahil edilmiştir.

3-6 b 1:1 İlişki

1:1 etiketinden de anlaşılacağı üzere, 1:1 ilişkisindeki bir varlık yalnızca bir başka varlıkla ilişkili olabilir ve bunun tersi de geçerlidir. Örneğin, bir bölüm başkanı -bir profesör- sadece bir departmana başkanlık edebilir ve bir departmanın sadece bir bölüm başkanı olabilir. PROFESÖR ve DEPARTMENT varlıkları bu nedenle 1:1 ilişki sergiler. (Tüm profesörlerin bir bölüme başkanlık etmediğini ve profesörlerden bir bölüme başkanlık etmelerinin *istenemeyeceğini* iddia edebilirsiniz. Yani, iki varlık arasındaki ilişki isteğe bağlıdır. Ancak, tartışmanın bu aşamasında, dikkatinizi temel 1:1 ilişkisine odaklamalısınız. İsteğe bağlı ilişkiler Bölüm 4'te ele alınacaktır). Temel 1:1 ilişkisi Şekil 3.21'de modellenmiştir ve uygulaması Şekil 3.22'de gösterilmiştir.

Şekil 3.21 PROFESÖR ve BÖLÜM arasındaki 1:1 İlişki

Şekil 3.22 PROFESÖR ve BÖLÜM arasında Uygulanan 1:1 İlişki

Tablo adı: PROFESSOR
Birincil anahtar:
EMP_NUM Yabancı anahtar: DEPT_CODE

Veritabanı adı: Ch03_TinyCollege

EMP_NUM	DEPT_CODE	PROF_OFFICE	PROF_EXTENSION	PROF_HIGH_DEGREE
103	HIST	DRE 156	6783	Ph.D.
104	ENG	DRE 102	5561	MA
105	ACCT	KLR 229D	8665	Ph.D.
106	MKT/MGT	KLR 126	3899	Ph.D.
110	BIOL	AAK 160	3412	Ph.D.
114	ACCT	KLR 211	4436	Ph.D.
155	MATH	AAK 201	4440	Ph.D.
160	ENG	DRE 102	2248	Ph.D.
162	CIS	KLR 203E	2359	Ph.D.
191	MKT/MGT	KLR 409B	4016	DBA
195	PSYCH	AAK 297	3550	Ph.D.
209	CIS	KLR 333	3421	Ph.D.
228	CIS	KLR 300	3000	Ph.D.
297	MATH	AAK 194	1145	Ph.D.
299	ECON/FIN	KLR 284	2851	Ph.D.
301	ACCT	KLR 244	4683	Ph.D.
335	ENG	DRE 208	2000	Ph.D.
342	SOC	BBG 208	5514	Ph.D.
387	BIOL	AAK 230	8665	Ph.D.
401	HIST	DRE 156	6783	MA
425	ECON/FIN	KLR 284	2851	MBA
435	ART	BBG 185	2278	Ph.D.

PROFESÖR ilişkisini kullanan 1:M DEPARTMENT, PROFESSOR tablosuna DEPT_CODE yabancı anahtarının yerleştirilmesi yoluyla uygulanır.

1:1 PROFESÖR sandalye DEPARTMENT ilişkisi, DEPARTMENT tablosuna EMP_NUM yabancı anahtarının yerleştirilmesi yoluyla uygulanır.

Tablo adı: DEPARTMENT
Birincil anahtar:
DEPT_CODE Yabancı anahtar: EMP_NUM

DEPT_CODE	DEPT_NAME	SCHOOL_CODE	EMP_NUM	DEPT_ADDRESS	DEPT_EXTENSION
ACCT	Accounting	BUS	114	KLR 211, Box 52	3119
ART	Fine Arts	A&SCI	435	BBG 185, Box 128	2278
BIOL	Biology	A&SCI	387	AAK 230, Box 415	4117
CIS	Computer Info. Systems	BUS	209	KLR 333, Box 56	3245
ECON/FIN	Economics/Finance	BUS	299	KLR 284, Box 63	3126
ENG	English	A&SCI	160	DRE 102, Box 223	1004
HIST	History	A&SCI	103	DRE 156, Box 284	1867
MATH	Mathematics	A&SCI	297	AAK 194, Box 422	4234
MKT/MGT	Marketing/Management	BUS	106	KLR 126, Box 55	3342
PSYCH	Psychology	A&SCI	195	AAK 297, Box 438	4110
SOC	Sociology	A&SCI	342	BBG 208, Box 132	2008

Şekil 3.22'deki tabloları incelerken birkaç önemli özelliğe dikkat edin:

- Her profesör bir Tiny College çalışanıdır. Bu nedenle, profesör tanımlaması EMP_NUM aracılığıyla yapılır. (Ancak, tüm çalışanların profesör olmadığını unutmayın - başka bir isteğe bağlı ilişki daha vardır).
- 1:1 "PROFESSOR chairs DEPARTMENT" ilişkisi, DEPARTMENT tablosunda EMP_NUM yabancı anahtarına sahip olarak uygulanır. 1:1 ilişkisinin, "çok" tarafının tek bir oluşumla sınırlandırıldığı 1:M ilişkisinin özel bir durumu olarak ele alındığını unutmayın. Bu durumda, DEPARTMENT bir yabancı anahtar olarak EMP_NUM'u içerir ve bu da bir kürsüye sahip olan *departman* olduğunu gösterir.
- Ayrıca PROFESSOR tablosunun 1:M "DEPARTMENT employs PROFESSOR" ilişkisini uygulamak için DEPT_CODE yabancı anahtarını içerdiğine dikkat edin. Bu, iki varlığın aynı anda iki (hatta daha fazla) ilişkiye nasıl katılabileceğine dair iyi bir örnektir.

1:1 ilişki ile yabancı anahtar yerleşimi açısından, teoride her iki varlıktan birindeki birincil anahtar diğer varlıkta yabancı anahtar olarak kullanılabilir. Pratikte, bazı durumlar bize yabancı anahtarı bir yönde veya diğer yönde yerleştirme tercihi verecektir. Bu örnekte, aynı varlıklar arasında 1:M ilişkisinin bulunması, eşanlamlılıktan kaçınmak için yabancı anahtarın 1:1 ilişkisine yerleştirilmesini tercih etmemize neden olmuştur. İsteğe bağlı ilişkiler gibi diğer durumlar Bölüm 4'te ele alınmaktadır. Önceki "PROFES- SOR sandalyeleri DEPARTMAN" örneği uygun bir 1:1 ilişkisini göstermektedir. *Aslında, 1:1 ilişkisinin kullanılması, iki varlık kümesinin olmaması gerektiği halde aynı tabloya yerleştirilmemesini sağlar*. Ancak, 1:1 ilişkisinin varlığı bazen varlık bileşenlerinin doğru tanımlanmadığı anlamına gelir. Bu, iki varlığın aslında aynı tabloya ait olduğunu gösterebilir!

1:1 ilişkilerin nadiren kullanılması gerekse de, belirli koşullar kesinlikle kullanılmalarını gerektirir. Bölüm 5, Gelişmiş Veri Modelleme'de, genelleme hiyerarşisi olarak adlandırılan ve belirli koşullar altında veritabanı tasarımlarını geliştirerek boşların çoğalmasını önlemek için güçlü bir araç olan bir kavramı keşfedeceksiniz. Genelleme hiyerarşilerinin bir özelliği, 1:1 ilişkiler olarak uygulanmalarıdır.

Çevrimiçi İçerik

www.cengage.com , adresinde Ch03_ TinyCollege veritabanını açarsanızSTUDENT ve CLASS varlıklarının yabancı anahtar olarak hala PROF_NUM kullandığını görürsünüz. PROF_NUM ve EMP_NUM aynı özniteliğin etiketleridir, bu da eşanlamlıların kullanımına örnektir - yani, aynı öznitelik için farklı adlar. Bu eş anlamlı kelimeler gelecekte ortadan kaldırılacaktır Tiny College veri tabanı geliştirilmeye devam ederken bölümler.

Çevrimiçi İçerik

www. cengage.com adresindeki Ch03_ AviaCo veritabanına bakarsanız, 1:1 PILOT to EMPLOYEE ilişkisi. Bu ilişki, Bölüm 5'te öğreneceğiniz bir genelleme hiyerarşisine dayanmaktadır.

3-6c M:N İlişkisi

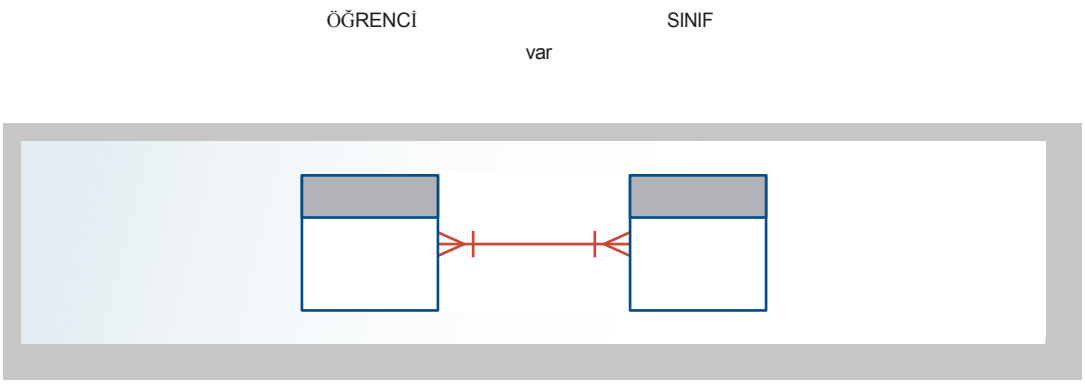
Çoktan çoka (M:N) ilişki ilişkisel ortamda doğrudan desteklenmez. Ancak, M:N ilişkileri, orijinal varlıklarla 1:M ilişkilerinde yeni bir varlık oluşturularak uygulanabilir.

M:N ilişkisini keşfetmek için tipik bir üniversite ortamını düşünün. Şekil 3.23'teki ER modeli bu M:N ilişkisini göstermektedir.

Şekil 3.23'te ERM'nin özelliklerine dikkat edin.

- Her SINIFIN birçok ÖĞRENCİSİ olabilir ve her ÖĞRENCİ birçok SINIF alabilir.
- STUDENT tablosundaki herhangi bir satır için CLASS tablosunda birçok satır olabilir ve CLASS tablosundaki herhangi bir satır için STUDENT tablosunda birçok satır olabilir.

Şekil 3.23 ERM'nin ÖĞRENCİ ve SINIF arasındaki M:N İlişkisi



Tablo 3.7 Örnek Öğrenci Kayıt Verileri

Öğrencinin Soyadı	Seçilmiş Sınıflar
Bowser	Muhasebe 1, ACCT-211, kod 10014 Mikrobilgisayara Giriş, CIS-220, kod 10018 İstatistiğe Giriş, QM-261, kod 10021
Smithson	Muhasebe 1, ACCT-211, kod 10014 Mikrobilgisayara Giriş, CIS-220, kod 10018 İstatistiğe Giriş, QM-261, kod 10021

M:N ilişkisini daha yakından incelemek için, her biri üç ders alan iki öğrencisi olan küçük bir üniversite düşünün. Tablo 3.7 iki öğrenci için kayıt verilerini göstermektedir.

Böyle bir veri ilişkisi ve Tablo 3.7'örnek veriler göz önüne alındığında, Şekil 3.24'te gösterildiği gibi, ilişkinin "çok" tarafına ilgili tablonun birincil anahtarına işaret eden bir yabancı anahtar ekleyerek bu M:N ilişkisini uygulayabileceğinizi yanlış bir şekilde varsayabilirsiniz.

Şekil 3.24 ÖĞRENCİ ve SINIF arasındaki M:N İlişkisinin Yanlış Uygulanması

Tablo adı: STUDENT Birincil anahtar: STU_NUM
Yabancı anahtar: yok
Veritabanı adı: Ch03_CollegeTry

STU_NUM	STU_LNAME	CLASS_CODE
321452	Bowser	10014
321452	Bowser	10018
321452	Bowser	10021
324257	Smithson	10014
324257	Smithson	10018
324257	Smithson	10021

Tablo adı: CLASS Birincil anahtar: CLASS_CODE
Yabancı anahtar: STU_NUM

CLASS_CODE	STU_NUM	CRS_CODE	CLASS_SECTION	CLASS_TIME	CLASS_ROOM	PROF_NUM
10014	321452	ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
10014	324257	ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
10018	321452	CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
10018	324257	CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
10021	321452	QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114
10021	324257	QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114

Ancak, M:N ilişkisi iki iyi nedenden dolayı Şekil 3.24'te gösterildiği gibi uygulanmamalıdır:

- Tablolar birçok fazlalık yaratır. Örneğin, STU_NUM değerlerinin STUDENT tablosunda birçok kez yer aldığına dikkat edin. Gerçek dünyadaki bir durumda, adres, sınıflandırma, ana dal ve ev telefonu gibi ek öğrenci öznitelikleri de STU- DENT tablosunda yer alacak ve bu öznitelik değerlerinin her biri burada gösterilen kayıtların her birinde tekrarlanacaktır. Benzer şekilde, CLASS tablosu da çok sayıda yineleme içerir: dersi alan her öğrenci bir CLASS kaydı oluşturur. Eğer CLASS tablosu kredi saatleri ve kurs açıklaması gibi nitelikleri de içeriyorsa sorun daha da kötüleşecektir. Bu fazlalıklar Bölüm 1'de tartışılan anomalilere yol açar.
- İki tablonun yapısı ve içeriği göz önüne alındığında, ilişkisel işlemler çok karmaşık hale gelir ve muhtemelen sistem verimliliği hatalarına ve çıktı hatalarına yol açar.

bileşik varlık
Bir M:N ilişkisini iki 1:M ilişkisine için tasarlanmış bir varlık. Bileşik varlığın birincil anahtarı, en azından bağladığı varlıkların birincil anahtarlarını içerir. *Köprü varlık* veya *ilişkisel varlık* olarak da bilinir. Ayrıca bkz. *bağlantı tablosu*.

köprü kuruluşu
Bkz. *bileşik varlık*.

ilişkisel varlık
Bkz. *bileşik varlık*.

Neyse ki, M:N ilişkisinin doğasında bulunan sorunlar, bir **bileşik varlık** (**köprü varlık** veya **ilişkisel varlık** olarak da adlandırılır) oluşturularak kolayca önlenebilir. Böyle bir tablo başlangıçta bir M:N ilişkisinde ilişkili olan tabloları birbirine bağlamak için kullanıldığından, bileşik varlık yapısı yabancı anahtarlar olarak en *azından* bağlanacak tabloların birincil anahtarlarını içerir. Veritabanı tasarımcısının bileşik tablonun birincil anahtarını tanımlarken iki ana seçeneği vardır: bu yabancı anahtarların kombinasyonunu kullanmak veya yeni bir birincil anahtar oluşturmak.

ERM'deki her bir varlığın bir tablo tarafından temsil edildiğini unutmayın. Bu nedenle, CLASS ve STU- DENT tablolarını birbirine bağlamak için Şekil 3.25'te gösterilen bileşik ENROLL tablosunu oluşturabilirsiniz. Bu örnekte, ENROLL tablosunun birincil anahtarı, CLASS_CODE ve STU_NUM yabancı anahtarlarının birleşimidir. Ancak tasarımcı, her ENROLL tablo satırını benzersiz bir şekilde tanımlamak için farklı bir satır değeri kullanarak ENROLL_LINE gibi tek özellikli yeni bir birincil anahtar oluşturmaya karar verebilirdi. (Microsoft Access kullanıcıları bu tür satır değerlerini otomatik olarak oluşturmak için Autonumber veri türünü kullanabilir).

Şekil 3.25 M:N İlişkisini İki 1:M İlişkisine Dönüştürme

Veritabanı adı: Ch03_CollegeTry2

Tablo adı: STUDENT
Birincil anahtar: STU_NUM Yabancı anahtar: yok

STU_NUM	STU_LNAME
321452	Bowser
324257	Smithson

Tablo adı: ENROLL
Birincil anahtar: CLASS_CODE + STU_NUM
Yabancı anahtar: CLASS_CODE, STU_NUM

CLASS_CODE	STU_NUM	ENROLL_GRADE
10014	321452	C
10014	324257	B
10018	321452	A
10018	324257	B
10021	321452	C
10021	324257	C

Tablo adı: CLASS Birincil anahtar: CLASS_CODE
Yabancı anahtar: CRS_CODE

CLASS_CODE	CRS_CODE	CLASS_SECTION	CLASS_TIME	CLASS_ROOM	PROF_NUM
10014	ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
10018	CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
10021	QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114

bağlantı tablosu
İlişkisel modelde, M:N ilişkisi uygulayan bir tablo. Ayrıca bkz. *bileşik varlık*.

Şekil 3.25'teki ENROLL tablosu STUDENT ve CLASS adlı iki tabloyu birbirine bağladığından, bu tablo aynı zamanda bir **bağlantı tablosu** olarak da adlandırılır. Başka bir deyişle, bir bağlantı tablosu bir bileşik varlığın uygulanmasıdır.

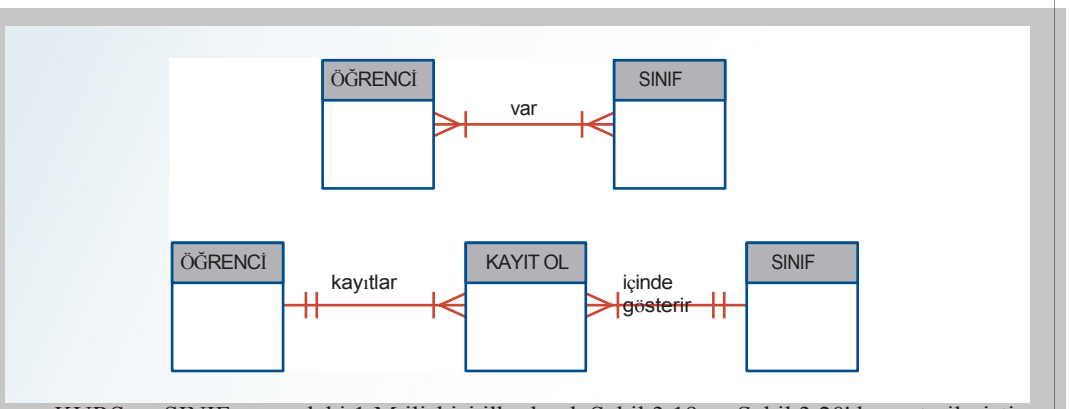
Not

Bağlantı özniteliklerine ek olarak, bileşik ENROLL tablosu kursta kazanılan not gibi ilgili öznitelikleri de içerebilir. Aslında, bileşik bir tablo, tasarımcının izlemek istediği herhangi bir sayıda öznitelik içerebilir. Bileşik varlığın, *gerçek bir tablo olarak uygulanmasına rağmen*, *kavramsal* olarak bir amaç için yaratılmış mantıksal bir varlık olduğunu unutmayın: orijinal M:N ilişkisindeki çoklu potansiyelini ortadan kaldırmak.

Şekil 3.25'te gösterilen ENROLL tablosu gerekli M:N - 1:M dönüşümünü sağlar. ENROLL tablosu tarafından temsil edilen bileşik varlığın, bağlayıcı olarak hizmet ettiği CLASS ve STUDENT tablolarının en azından birincil anahtarlarını (sırasıyla CLASS_CODE ve STU_NUM) içermesi gerektiğini gözlemleyin. Ayrıca STUDENT ve CLASS tablolarının artık varlık başına yalnızca bir satır içerdiğini unutmayın. ENROLL tablosu, yabancı anahtar değerlerinin birden fazla oluşumunu içerir, ancak bu kontrollü fazlalıklar, referans bütünlüğü uygulandığı sürece anomalilere neden olamaz. Gerektiğinde ek nitelikler atanabilir. Bu durumda, ENROLL_GRADE bir raporlama gereksinimini karşılamak için seçilmiştir. Ayrıca ENROLL_GRADE'in bileşik birincil anahtara tamamen bağımlı olduğunu unutmayın. Doğal olarak, dönüşüm ERM'ye de yansıtılır. Revize edilmiş ilişki Şekil 3.26'da gösterilmektedir.

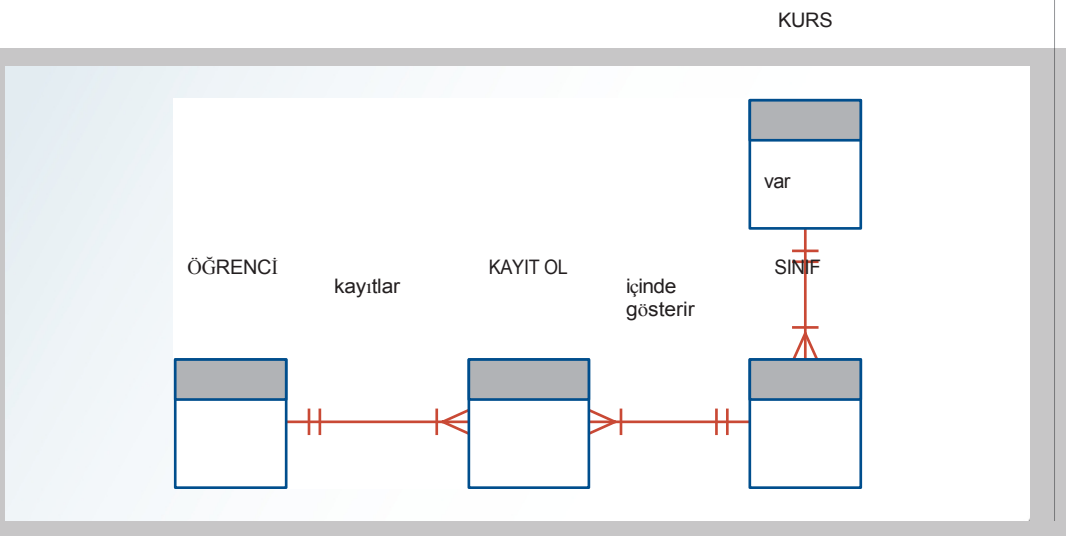
Şekil 3.26'yı incelerken, ENROLL adlı bileşik varlığın STUDENT ve CLASS arasındaki bağlantı tablosunu temsil ettiğine dikkat edin.

Şekil 3.26 M:N İlişkilerinin İki 1:M İlişkisine Değiştirilmesi

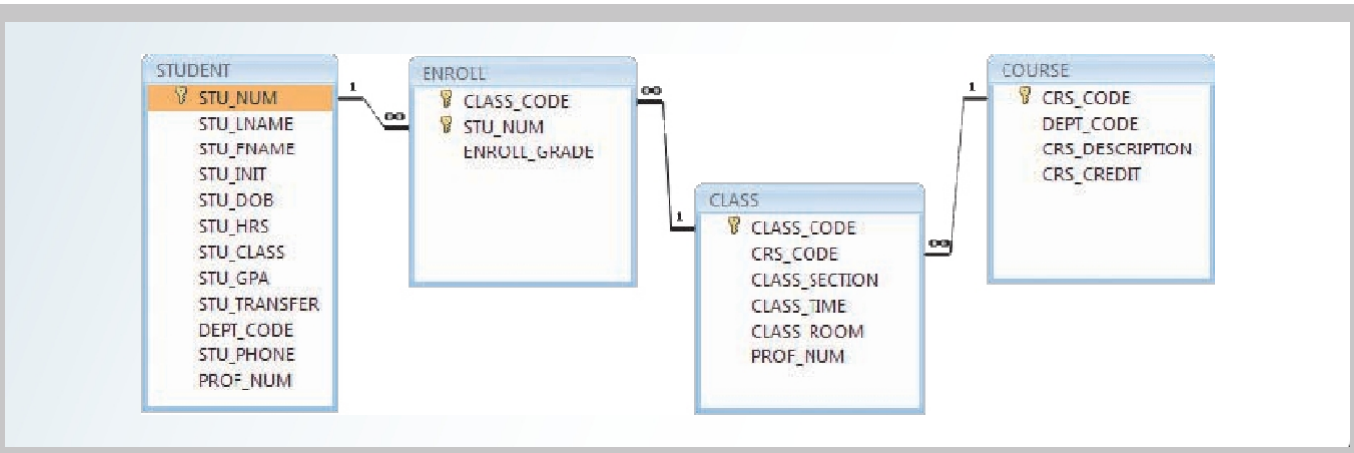


KURS ve SINIF arasındaki 1:M ilişkisi ilk olarak Şekil 3.19 ve Şekil 3.20'de gösterilmiştir. Veritabanının fazlalıklarını kontrol ederken bile mevcut bilgi miktarını artırabilirsiniz. Böylece Şekil 3.27, Şekil 3.19'da gösterilen COURSE ve CLASS arasındaki 1:M ilişkisini de içeren genişletilmiş ERM'yi göstermektedir. Modelin, her bir SINIF için ortak olan tüm KURS verilerinin KURS tablosunda tutulmasını sağlayarak fazlalıkları kontrol ederken bir SINIF'ın birden fazla bölümünü işleyebileceğine dikkat edin.

Şekil 3.27 Genişletilmiş ER Modeli



Şekil 3.28 Ch03_TinyCollege Veritabanı için İlişkisel Diyagram



Şekil 3.27'deki ERM'ye karşılık gelen ilişkisel diyagram Şekil 3.28'de gösterilmektedir. ERM, daha karmaşık veritabanları tasarlamak için nasıl kullanıldığını göstermek amacıyla Bölüm 4'te daha ayrıntılı olarak incelenecektir. ERM ayrıca Ek B ve C'de bir üniversite bilgisayar laboratuarının gerçekçi bir veritabanı tasarımını geliştirmek ve uygulamak için temel olarak kullanılacaktır.

Bu eklere www.cengage.com adresinden ulaşılabilir.

3-7 Veri Yedekliliği Yeniden Değerlendirildi

Bölüm 1'de, veri fazlalığının veri anormalliklerine yol açtığını ve bunun da veritabanının etkinliğini yok edebileceğini öğrendiniz. Ayrıca ilişkisel veritabanının, yabancı anahtarlar olarak adlandırılan tablolar tarafından paylaşılan ortak nitelikleri kullanarak veri fazlalıklarını kontrol etmeyi mümkün kıldığını da öğrendiniz.

Yabancı anahtarların doğru kullanımı veri fazlalığını kontrol etmek için çok önemlidir, ancak yabancı anahtar değerleri birçok kez tekrarlanabileceğinden sorunu tamamen ortadan kaldırmazlar. Ancak, yabancı anahtarların doğru kullanımı veri fazlalıklarını ve yıkıcı veri anormalliklerinin ortaya çıkma olasılığını *en aza indirir*.

Not

Gereksizliğin gerçek testi, belirli bir özniteliğin kaç kopyasının saklandığı *değil, bir özniteliğin ortadan kaldırılmasının bilgiyi ortadan kaldırıp kaldırmayacağıdır*. Bu nedenle, bir özniteliği silerseniz ve orijinal bilgi ilişkisel cebir yoluyla hala üretilebiliyorsa, bu özniteliğin dahil edilmesi gereksiz olacaktır. Bu görüşü göz önüne alındığında, uygun yabancı anahtarlar bir tabloda birden fazla bulunmalarına rağmen açıkça gereksiz değildir. Ancak, bu daha az kısıtlayıcı fazlalık görüşünü kullandığınızda bile, *kontrollü* fazlalıkların genellikle işlem hızını ve/veya bilgi gereksinimlerini sağlamak için sistemin bir parçası olarak tasarlandığını unutmayın.

Bölüm 4'te veritabanı tasarımcılarının genellikle birbiriyle çelişen üç gereksinimi uzlaştırması gerektiğini öğreneceksiniz: tasarım zarafeti, işlem hızı ve bilgi gereksinimleri. Ayrıca, Bölüm 13, İş Zekası ve Veri Ambarları'nda doğru veri ambarı tasarımının düzgün çalışması için dikkatlice tanımlanmış ve kontrol edilmiş veri fazlalıkları gerektirdiğini öğreneceksiniz. Veri fazlalıklarını nasıl tanımladığınızdan bağımsız olarak, zarar verme potansiyeli doğru uygulama ve dikkatli kontrol ile sınırlandırılır.

Veri fazlalığını kontrol etmek ne kadar önemli olsa da, bazen veritabanının önemli bilgi amaçlarına hizmet etmesi için veri seviyesinin gerçekten artırılması gerekir. Bu tür fazlalıklar hakkında Bölüm 13'te bilgi edineceksiniz. Ayrıca, veri fazlalıkları bazen verilerin tarihsel doğruluğunu korumak için varmış *gibi görünür*. Örneğin, küçük bir faturalama sistemi düşünün. Sistem, bir veya daha fazla ÜRÜN satın alabilen ve böylece bir FATURA oluşturan MÜŞTERİ'yi içerir. Bir müşteri aynı anda birden fazla ürün satın alabileceğinden, bir fatura, her biri satın alınan ürün hakkında ayrıntılar sağlayan birkaç fatura SATIRI içerebilir. Faturada görünen her ürün için tutarlı bir fiyatlandırma girdisi sağlamak amacıyla ÜRÜN tablosu ürün fiyatını içermelidir. Böyle bir sistemin parçası olan tablolar Şekil 3.29'da gösterilmektedir. Sistemin ilişkisel diyagramı Şekil 3.30'da gösterilmektedir.

Şekil 3.29 Küçük Bir Faturalama Sistemi

Tablo adı: CUSTOMER
Birincil anahtar: CUS_CODE
Yabancı anahtar: yok

Veritabanı adı: Ch03_SaleCo

CUS_CODE	CUS_LNAME	CUS_FNAME	CUS_INITIAL	CUS_AREACODE	CUS_PHONE
10010	Ramas	Alfred	A	615	844-2573
10011	Dunne	Leona	K	713	894-1238
10012	Smith	Kathy	W	615	894-2285
10013	Olowski	Paul	F	615	894-2180
10014	Orlando	Myron		615	222-1672
10015	O'Brian	Amy	B	713	442-3381
10016	Brown	James	G	615	297-1228
10017	Williams	George		615	290-2556
10018	Farriss	Anne	G	713	382-7185
10019	Smith	Olette	K	615	297-3809

Tablo adı: INVOICE Birincil anahtar: INV_NUMBER
Yabancı anahtar: CUS_CODE

INV_NUMBER	CUS_CODE	INV_DATE
1001	10014	08-Mar-22
1002	10011	08-Mar-22
1003	10012	08-Mar-22
1004	10011	09-Mar-22

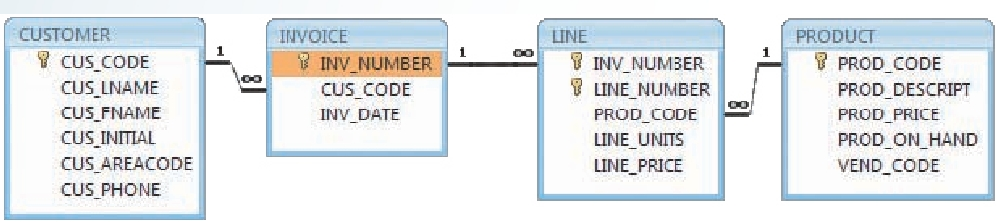
Tablo adı: LINE
Birincil anahtar: INV_NUMBER + LINE_NUMBER
Yabancı anahtar: INV_NUMBER, PROD_CODE

INV_NUMBER	LINE_NUMBER	PROD_CODE	LINE_UNITS	LINE_PRICE
1001	1	123-21UUY	1	189.99
1001	2	SRE-657UG	3	2.99
1002	1	QER-34256	2	18.63
1003	1	ZZX/3245Q	1	6.79
1003	2	SRE-657UG	1	2.99
1003	3	001278-AB	1	12.95
1004	1	001278-AB	1	12.95
1004	2	SRE-657UG	2	2.99

Tablo adı: PRODUCT
Birincil anahtar: PROD_CODE
Yabancı anahtar: yok

PROD_CODE	PROD_DESCRIPT	PROD_PRICE	PROD_ON_HAND	VEND_CODE
001278-AB	Claw hammer	12.95	23	232
123-21UUY	Houselite chain saw, 16-in. bar	189.99	4	235
QER-34256	Sledge hammer, 16-lb. head	18.63	6	231
SRE-657UG	Rat-tail file	2.99	15	232
ZZX/3245Q	Steel tape, 12-ft. length	6.79	8	235

Şekil 3.30 Faturalama Sistemi için İlişkisel Diyagram



İki şekildeki tabloları ve ilişkileri incelerken, aşağıdakileri takip edebileceğinizi unutmayın tipik satış bilgilerinden oluşur. Örneğin, dört tablo arasındaki ilişkileri izleyerek, 10014 numaralı müşterinin (Myron Orlando) 8 Mart 2022 tarihinde 1001 numaralı faturaya yazılan iki ürünü satın aldığını keşfedersiniz: bir adet 16 inç çubuklu Houselite zincirli testere ve üç adet fare kuyruğu dosyası. Başka bir deyişle, CUSTOMER tablosundaki 10014 numaralı CUS_CODE değerini INVOICE tablosundaki eşleşen CUS_CODE değerine kadar izleyin. Ardından, INV_NUMBER 1001'i LINE tablosundaki ilk iki satıra kadar izleyin. Son olarak, LINE'daki iki PROD_CODE değerini PRODUCT'taki PROD_CODE değerleriyle eşleştirin. Uygulama yazılımı, her bir fatura satır kaleminin SATIR_BİRİMLERİ ile SATIR_FİYATI çarpılarak, sonuçlar toplanarak ve uygun vergiler uygulanarak doğru faturayı yazmak için kullanılacaktır. Daha sonra, diğer uygulama yazılımları aynı tekniği kullanarak satışları hafta, ay veya yıl bazında takip eden ve karşılaştıran satış raporları yazabilir. Şekil 3.29'daki satış işlemlerini incelerken, müşteriye fatura edilen ürün fiyatının PRODUCT tablosundan türetildiğini düşünebilirsiniz çünkü ürün verileri burada saklanmaktadır. *Ancak aynı ürün fiyatı neden LINE tablosunda tekrar ortaya çıkıyor? Bu bir veri fazlalığı değil midir?* Kesinlikle öyle görünüyor, ancak bu kez, görünürdeki fazlalık sistemin başarısı için çok önemlidir. Ürün fiyatının PRODUCT tablosundan LINE tablosuna kopyalanması *işlemlerin tarihsel doğruluğunu* korur. Örneğin, LINE tablosuna LINE_PRICE değerini yazmadığınızı ve satış gelirini hesaplamak için PRODUCT tablosundaki PROD_PRICE değerini kullandığınızı varsayalım. Şimdi, fiyatların sıklıkla yaptığı gibi PRODUCT tablosunun PROD_PRICE değerinin değiştiğini varsayalım. Bu fiyat değişikliği, sonraki tüm satış geliri hesaplamalarına uygun şekilde yansıtılacaktır. Ancak, geçmiş satış gelirlerinin hesaplamaları da işlem gerçekleştiğinde yürürlükte olmayan yeni ürün fiyatını yansıtacaktır! Sonuç , tüm geçmiş işlemler için gelir hesaplamaları hatalı olacak ve böylece zaman içinde uygun satış karşılaştırmaları yapma olasılığı ortadan kalkacaktır. Öte yandan, fiyat verileri PRODUCT tablosundan kopyalanır ve LINE tablosunda işlemle birlikte saklanırsa, bu fiyat *her* zaman *o sırada* gerçekleşen işlemi doğru bir şekilde yansıtacaktır. Keşfedeceksiniz

Bu tür planlı "fazlalıklar" iyi veritabanı tasarımında yaygındır.

Son olarak, Şekil 3.29'daki LINE tablosunda neden LINE_NUMBER özniteliğinin kullanıldığını merak edebilirsiniz. INV_NUMBER ve PROD_CODE kombinasyonu yeterli bir bileşik birincil anahtar olmaz mıydı ve bu nedenle LINE_NUMBER gereksiz değil mi? Evet, öyle, ancak bu fazlalık, genellikle bu tür satır numaralarını otomatik olarak oluşturan faturalama yazılımlarında yaygın bir uygulamadır. Bu durumda, fazlalık gerekli değildir, ancak otomatik olarak oluşturulduğu göz önüne alındığında, fazlalık bir anormallik kaynağı değildir. LINE_NUMBER'ın dahil edilmesi başka bir fayda da sağlar: alınan faturalama verilerinin sırası her zaman verilerin girildiği sırayla eşleşecektir. Birincil anahtarın bir parçası olarak ürün kodları kullanılıyorsa, fatura tamamlanır tamamlanmaz ve veriler depolanır depolanmaz indeksleme bu ürün kodlarını düzenleyecektir. Bir müşteri arayıp "Faturamdaki ikinci kalemin fiyatı yanlış" dediğinde ve siz de satırları müşterinin kopyasındakilerden farklı bir sıra gösteren bir faturaya baktığınızda yaşanabilecek olası karışıklığı tahmin edebilirsiniz!

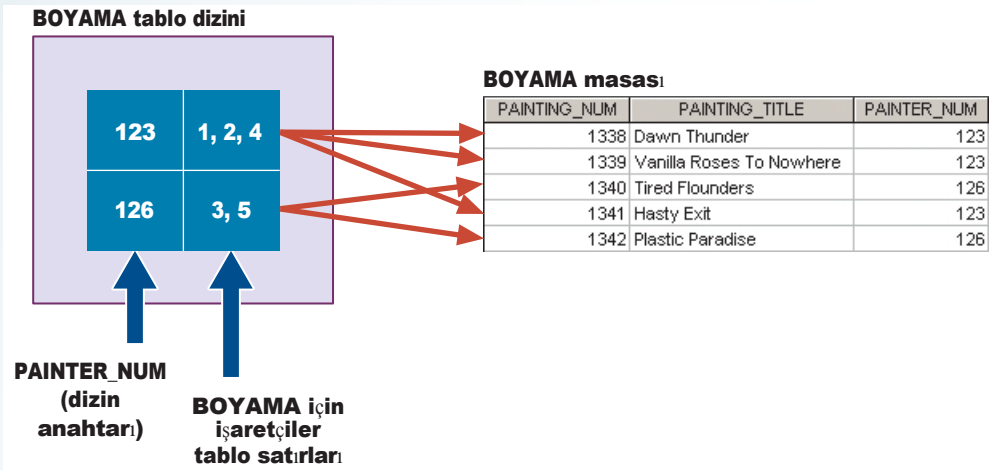
3-8 İndeksler

Bir kütüphanede bir kitabı bulmak istediğinizi varsayalım. İstediğiniz kitabı bulana kadar her kitaba bakmak mantıklı mı? Elbette hayır; başlık, konu ve yazara göre indekslenen kütüphane kataloğunu kullanırsınız. Dizin (manuel ya da bilgisayarlı kütüphane kataloğunda) sizi kitabın bulunduğu yere yönlendirir ve böylece kitaba hızlı ve basit bir şekilde ulaşmanızı sağlar. **Dizin**, bir tablodaki satırlara mantıksal olarak erişmek için kullanılan düzenli bir düzenlemedir.

indeks
Dizin anahtar değerleri ve satır kimliği değerlerinden (işaretçiler) oluşan sıralı bir dizi. Dizinler genellikle veri alımını hızlandırmak ve kolaylaştırmak için kullanılır. *Dizin anahtarı* olarak da bilinir.

Ya da bu kitapta *ER modeli* gibi bir konu bulmak istediğinizi varsayalım. Konuya rastlayana kadar her sayfayı okumak mantıklı mı? Elbette hayır; kitabın dizinine gitmek, *ER modeli* ifadesini aramak ve sizi uygun sayfa(lar)a yönlendiren referansları okumak çok daha basittir. Her durumda, ihtiyaç duyulan bir öğeyi hızlı bir şekilde bulmak için bir dizin kullanılır.

Şekil 3.31 Bir Endeksin Bileşenleri



İlişkisel veritabanı ortamındaki indeksler, önceki paragraflarda açıklanan indeksler gibi çalışır. Kavramsal açıdan bakıldığında, bir dizin bir dizin anahtarı ve bir dizi işaretçiden oluşur. **İndeks anahtarı** aslında indeksin referans noktasıdır. Daha resmi olarak, bir dizin anahtarların ve sıralı bir düzenlemesidir. Her anahtar, anahtar tarafından tanımlanan verinin konumuna işaret eder.

dizin anahtarı
Dizine bakınız.

Örneğin, Şekil 3.18'deki Ch03_Museum veritabanında belirli bir ressam tarafından oluşturulan tüm tabloları aramak istediğinizi varsayalım. Bir dizin olmadan, PAINTING tablosundaki her satırı okumanız ve PAINTER_NUM'un istenen ressamla eşleşip eşleşmediğine bakmanız gerekir. Ancak, PAINTER tablosunu indeksler ve PAINTER_NUM indeks anahtarını kullanırsanız, sadece indekste uygun PAINTER_NUM'a bakmanız ve eşleşen işaretçileri bulmanız gerekir. Kavramsal olarak konuşmak gerekirse, dizin Şekil 3.31'deki sunuma benzeyecektir.

Şekil 3.31'i incelerken, ilk PAINTER_NUM dizin anahtarı değerinin (123) PAINTING tablosunun 1, 2 ve 4 numaralı kayıtlarında bulunduğuna dikkat edin. İkinci PAINTER_NUM dizin anahtarı değeri (126) PAINTING tablosunun 3 ve 5 numaralı kayıtlarında bulunur.

DBMS'ler indeksleri birçok amaç için kullanır. Az önce bir dizinin verileri daha verimli bir şekilde almak için kullanılabileceğini öğrendiniz, ancak dizinler bir DBMS tarafından belirli bir özniteliğe veya özniteliklere göre sıralanmış verileri almak için de kullanılabilir. Örneğin, bir müşterinin soyadı üzerinde bir dizin oluşturmak, müşteri verilerini müşterinin soyadına göre alfabetik olarak almanızı sağlayacaktır. Ayrıca, bir dizin anahtarı bir veya daha fazla öznitelikten oluşabilir. Örneğin, Şekil 3.29'da, VEND_CODE ve PROD_CODE üzerinde bir dizin oluşturarak PRODUCT tablosundaki satıcıya göre sıralanmış ve satıcı içinde ürüne göre sıralanmış tüm satırları alabilirsiniz.

Dizinler, birincil anahtarların uygulanması için DBMS'lerde önemli bir rol oynar. Bir tablonun birincil anahtarını tanımladığınızda, DBMS otomatik olarak birincil anahtar sütun(lar)ı üzerinde benzersiz bir dizin oluşturur. Örneğin, Şekil 3.29'da CUS_CODE'u CUSTOMER tablosunun birincil anahtarı olarak bildirdiğinizde, VTYS otomatik olarak bu nitelik üzerinde benzersiz bir dizin oluşturur. **Benzersiz** bir **dizinde**, adından da anlaşılacağı gibi, dizin anahtarının kendisiyle ilişkili yalnızca bir işaretçi değeri (satırı) olabilir. (Şekil 3.31'deki dizin benzersiz bir dizin değildir çünkü PAINTER_NUM ile ilişkili birden fazla işaretçi değeri vardır. Örneğin, ressam numarası 123, PAINTING tablosunda üç satıra (1, 2 ve 4) işaret eder).

benzersiz dizin
Dizin anahtarının yalnızca bir ilişkili işaretçi değerine (satır) sahip olabileceği bir dizin.

Bir tablonun birçok dizini olabilir, ancak her dizin yalnızca bir tabloyla ilişkilendirilir. Dizin anahtarı birden fazla özniteliğe sahip olabilir (bileşik dizin). Bir dizin oluşturmak kolaydır. Bölüm 8'de basit bir SQL komutunun gerekli herhangi bir dizini oluşturduğunu öğreneceksiniz.

3-9 Codd'un İlişkisel Veritabanı Kuralları

1985 yılında Dr. E. F. Codd ilişkisel bir veritabanı sistemini tanımlamak için 12 kuraldan oluşan bir liste yayınlamıştır.[1] Codd bu listeyi, birçok satıcının minimum ilişkisel standartları karşılamamasına rağmen ürünlerini "ilişkisel" olarak pazarlamasından duyduğu endişe nedeniyle yayınlamıştır. Tablo 3.8'de gösterilen Dr. Codd'un listesi, gerçekten ilişkisel bir veritabanının ne olması gerektiğine dair bir referans çerçevesidir. Baskın veritabanı satıcılarının bile 12 kuralın tümünü tam olarak desteklemediğini unutmayın.

Tablo 13.8 Dr. Codd'un 12 İlişkisel Veritabanı Kuralı

Kural	Kural Adı	Açıklama
1	Bilgi	İlişkisel bir veritabanındaki tüm bilgiler mantıksal olarak tablolardaki satırlarda sütun değerleri olarak temsil edilmelidir.
2	Garantili erişim	Bir tablodaki her değere tablo adı, birincil anahtar değeri ve sütun adı kombinasyonuyla erişilebileceği garanti edilir.
3	Boşların sistematik olarak ele alınması	Null'lar, veri türünden bağımsız olarak sistematik bir şekilde temsil edilmeli ve ele alınmalıdır.
4	İlişkisel modele dinamik çevrimiçi katalog	Meta veriler sıradan veriler gibi, yani veritabanı içindeki tablolarda saklanmalı ve yönetilmelidir; bu veriler standart veritabanı ilişkisel dilini kullanan yetkili kullanıcılar tarafından kullanılabilir olmalıdır.
5	Kapsamlı veri alt dili	İlişkisel veritabanı birçok dili destekleyebilir; ancak, iyi tanımlanmış, bildirimsel bir dilin yanı sıra veri , görünüm tanımı, veri manipülasyonu (etkileşimli ve program tarafından), bütünlük kısıtlamaları, yetkilendirme ve işlem yönetimini (başlatma, işleme ve geri alma) desteklemelidir.
6	Güncellemeyi görüntüle	Teorik olarak güncellenebilir olan herhangi bir görünüm, sistem aracılığıyla güncellenebilir olmalıdır.
7	Üst düzey ekleme, güncelleme ve silme	Veritabanı set düzeyinde ekleme, güncelleme ve silme işlemlerini desteklemelidir.
8	Fiziksel veri bağımsızlığı	Uygulama programları ve geçici tesisler, fiziksel erişim yöntemleri veya depolama yapıları değiştirildiğinde mantıksal olarak etkilenmez.
9	Mantıksal veri bağımsızlığı	Orijinal tablo değerlerini koruyan tablo yapılarında değişiklik yapıldığında (sütunların sırasını değiştirme veya sütun ekleme) uygulama programları ve geçici tesisler mantıksal olarak etkilenmez.
10	Dürüstlük bağımsızlık	Tüm ilişkisel bütünlük kısıtlamaları ilişkisel dilde tanımlanabilir olmalı ve uygulama düzeyinde değil sistem kataloğunda saklanmalıdır.
11	Dağıtım bağımsızlığı	Son kullanıcılar ve uygulama programları veri konumundan (dağıtık yerel veritabanları) habersizdir ve bundan etkilenmez.
12	Dönüşümsüzlük	Sistem verilere düşük seviyeli erişimi destekliyorsa, kullanıcıların veritabanının bütünlük kurallarını atlamasına izin verilmemelidir.
13	Kural sıfır	Önceki tüm kurallar, bir veritabanının ilişkisel olarak kabul edilebilmesi için ilişkisel olanaklarını yalnızca yönetim için kullanması gerektiği fikrine dayanmaktadır.

[1]Codd, E., "DBMS'niz Gerçekten İlişkisel mi?" ve "DBMS'niz Kurallara Göre mi Çalışıyor?" *Computerworld*, 14 ve 21 Ekim 1985.

Özet

- Tablolar, ilişkisel bir veri tabanının temel yapı taşlarıdır. Varlık kümesi olarak bilinen ilgili varlıkların bir gruplaması bir tabloda saklanır. Kavramsal olarak, ilişkisel tablo kesişen satırlardan (tuples) ve sütunlardan oluşur. Her satır tek bir varlığı temsil eder ve her sütun varlıkların özelliklerini (niteliklerini) temsil eder.
- Anahtarlar, ilişkisel tabloların kullanımının merkezinde yer alır. Anahtarlar işlevsel bağımlılıkları tanımlar; , diğer nitelikler anahtara bağlıdır ve bu nedenle anahtar değeri biliniyorsa bulunabilir. Bir anahtar bir üst anahtar, aday anahtar, birincil anahtar, ikincil anahtar veya yabancı anahtar olarak sınıflandırılabilir.
- Her tablo satırının bir birincil anahtarı olmalıdır. Birincil anahtar, herhangi bir satırda bulunan diğer tüm öznitelikleri benzersiz bir şekilde tanımlayan bir öznitelik veya öznitelik kombinasyonudur. Bir birincil anahtarın benzersiz olması gerektiğinden, varlık bütünlüğü korunacaksa null değerlere izin verilmez.

- Tablolar bağımsız olmalarına rağmen, ortak niteliklerle birbirlerine bağlanabilirler. Böylece, bir tablonun birincil anahtarı, bağlı olduğu başka bir tabloda yabancı anahtar olarak görünebilir. Referans bütünlüğü, yabancı anahtarın ilgili tablodaki birincil anahtarla eşleşen değerler içermesi veya boş değerler içermesi gerektiğini belirtir.
- İlişkisel model, SELECT, PROJECT, JOIN, INTERSECT, UNION, DIFFERENCE, PRODUCT dahil olmak üzere çeşitli ilişkisel birleştirme işlevlerini destekler,
  ve DIVIDE. Bu fonksiyonların temel matematiksel formlarının anlaşılması, veri manipülasyon seçeneklerinin daha geniş bir şekilde anlaşılmasını sağlar.
- İlişkisel bir veritabanı, veri işleme işinin çoğunu perde arkasında gerçekleştirir. Örneğin, siz bir veritabanı oluşturduğunuzda, RDBMS otomatik olarak için bir veri sözlüğü barındıracak bir yapı . Veritabanında her yeni tablo oluşturduğunuzda, RDBMS veri sözlüğünü günceller ve böylece veritabanı belgelerini sağlar.
- İlişkisel veritabanlarının temellerini öğrendikten sonra tasarıma odaklanabilirsiniz. İyi tasarım, uygun varlıkları ve bunların niteliklerini ve ardından varlıklar arasındaki ilişkileri belirleyerek başlar. Bu ilişkiler (1:1, 1:M ve M:N) ERD'ler kullanılarak gösterilebilir. ERD'lerin kullanımı basit mantıksal tasarımlar oluşturmanızı ve değerlendirmenizi sağlar. 1:M ilişkisi iyi bir tasarıma en kolay şekilde dahil edilebilir; sadece "1 "in birincil anahtarının "çok" tablosuna dahil edildiğinden emin olun.

Anahtar Terimler

associative entity
attribute domain
bridge entity
candidate key
closure
composite entity
composite key
data dictionary
dependent
determinant
determination
DIFFERENCE
DIVIDE
etki alanı
varlık bütünlüğü
equijoin
bayraklar
yabancı anahtar (FK)
tam fonksiyonel bağımlılık
fonksiyonel bağımlılık
eşanlamlısı
dizin dizin
anahtarı iç
birleştirme
INTERSECT
JOIN
sütunlara katıl
anahtar
anahtar
öznitelik sol
dış
birleştirme
bağlantı
tablosu doğal
birleştirme
null
outer join yüklem
mantığı birincil
anahtar (PK)
ÜRÜN PROJESI
referans bütünlüğü
ilişkisel cebir relvar
KISITLAMA
sağ dış birleştirme
ikincil anahtar
SELECT
küme teorisi
süper anahtar
eşanlamlı
sistem kataloğu
teta join tuple
BİRLİK
sendika uyumlu
benzersiz dizin

İnceleme Soruları

1. Bir veritabanı ile bir tablo arasındaki fark nedir?
2. Bir veritabanının hem varlık bütünlüğü hem de referans bütünlüğü gösterdiğini söylemek ne anlama gelir?
3. Bir veritabanında varlık bütünlüğü ve referans neden önemlidir?

Çevrimiçi İçerik

Sorularda ve problemlerde kullanılan tüm veritabanları www.cengage.com adresinde mevcuttur. Veritabanı adları şekillerde gösterilen veritabanı adlarıyla eşleşmektedir.

4. İki ilişkinin sendika uyumlu olarak kabul edilmesi için karşılaması gereken gereklilikler nelerdir?

5. Birlik uyumlu olmayan bir tablo çiftine hangi ilişkisel cebir operatörleri uygulanabilir?
6. Veri sözlüğüne neden bazen şu adların verildiğini açıklayın "veritabanı tasarımcısının veritabanı."
7. Bir veritabanı kullanıcısı manuel olarak "Dosya iki yüz kayıt içeriyor, her kayıt dokuz alan içeriyor." şeklinde not düşer. Bu ifadeyi "çevirmek" için uygun ilişkisel veritabanı terminolojisini kullanın.

Soru 8-12'yi yanıtlamak için Şekil Q3.8'i kullanın.

8. STUDENT ve PROFESSOR tablolarını kullanarak, doğal birleştirme, eşit birleştirme ve dış birleştirme arasındaki farkı gösterin.
9. $\pi_{(stu\_code)}$ (öğrenci) ile sonuçlanacak tabloyu oluşturun.
10. π'den kaynaklanacak tabloyu oluşturun $_{stu\_code,\ dept\_code}$ (öğrenci⋈ profesör).
11. Şekil Q3.8'de gösterilen veritabanı için temel ERD'yi oluşturun.
12. Şekil Q3.8'de gösterilen veritabanı için ilişkisel diyagramı oluşturun.

Şekil Q3.8 Ch03_CollegeQue Veritabanı Tabloları

Veritabanı adı: Ch03_CollegeQue

Tablo adı: ÖĞRENCİ

STU_CODE	PROF_CODE
100278	
128569	2
512272	4
531235	2
531268	
553427	1

Tablo adı: PROFESÖR

PROF_CODE	DEPT_CODE
1	2
2	6
3	6
4	4

Soru 13-17'yi yanıtlamak için Şekil Q3.13'ü kullanın.

Şekil Q3.13 Ch03_VendingCo Veritabanı Tabloları

Veritabanı adı: Ch03_VendingCo

Masa adı: BOOTH

BOOTH_PRODUCT	BOOTH_PRICE
Chips	1.5
Cola	1.25
Energy Drink	2

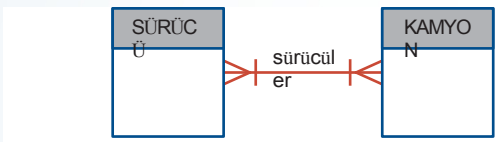
Tablo adı: MACHINE

MACHINE_PRODUCT	MACHINE_PRICE
Chips	1.25
Chocolate Bar	1
Energy Drink	2

13. Şekil Q3.13'te gösterilen tablolara UNION ilişkisel operatörünü uygulamak için ilişkisel cebir formülünü yazınız.
14. Şekil Q3.13'te gösterilen tablolara UNION ilişkisel operatörü uygulandığında ortaya çıkan tabloyu oluşturun.
15. Şekil Q3.13'te gösterilen tablolara bir INTERSECT ilişkisel işleci uygulamak için ilişkisel cebir formülünü yazınız.
16. Şekil Q3.13'te gösterilen tablolara bir INTER- SECT ilişkisel operatörü uygulandığında ortaya çıkan tabloyu oluşturun.
17. Şekil S3.13'teki tabloları kullanarak, MAKİNE FARKI TABLOSU'ndan kaynaklanan tabloyu oluşturun.

Soru 18'i yanıtlamak için Şekil Q3.18'i kullanın.

Şekil Q3.18 DRIVER ve TRUCK için Karga Ayağı ERD



Belirli bir zaman aralığında, bir SÜRÜCÜ birçok KAMYON'u sürebilir ve herhangi bir KAMYON birçok SÜRÜCÜ tarafından sürülebilir

18. Şekil Q3.18'de gösterilen ERD'ye sahip olduğunuzu varsayalım. Bu modeli yalnızca 1:M ilişkilerini gösteren bir ERM'ye nasıl dönüştürürsünüz? (Revize edilmiş ERD'yi oluşturduğunuzdan emin olun).
19. Eşanlamlılar ve eşanlamlılar nedir ve veritabanı tasarımında neden bunlardan kaçınılmalıdır?
20. İki tablodan oluşan bir veri tabanında l:M ilişkisini nasıl uygularsınız? Bir örnek veriniz.

Soru 21'i yanıtlamak için Şekil Q3.21'i kullanın.

Şekil Q3.21 Ch03_NoComp Veritabanı EMPLOYEE Tablosu

Tablo adı: EMPLOYEE **Veritabanı adı: Ch03_NoComp**

EMP_NUM	EMP_LNAME	EMP_INITIAL	EMP_FNAME	DEPT_CODE	JOB_CODE
11234	Friedman	K	Robert	MKTG	12
11238	Olanski	D	Delbert	MKTG	12
11241	Fontein		Juliette	INFS	5
11242	Cruazona	J	Maria	ENG	9
11245	Smithson	B	Bernard	INFS	6
11248	Washington	G	Oleta	ENGR	8
11256	McBride		Randall	ENGR	8
11257	Kachinn	D	Melanie	MKTG	14
11258	Smith	W	William	MKTG	14
11260	Ratula	A	Katrina	INFS	5

21. Şekil Q3.21'de gösterilen tablonun bileşenlerini doğru terminolojiyi kullanarak tanımlayın ve açıklayın. Tablonun olası yabancı anahtar(lar)ını tanımlamak için adlandırma kuralları bilginizi kullanınız.

Soru 22-27'yi yanıtlamak için Şekil Q3.22'de gösterilen veritabanını kullanın.

Şekil Q3.22 Ch03_Theater Veritabanı Tabloları

Veritabanı adı: Ch03_Theater

Tablo adı: DİREKTÖR

DIR_NUM	DIR_LNAME	DIR_DOB
100	Broadway	12-Jan-65
101	Hollywoody	18-Nov-53
102	Goofy	21-Jun-62

Tablo adı: PLAY

PLAY_CODE	PLAY_NAME	DIR_NUM
1001	Cat On a Cold, Bare Roof	102
1002	Hold the Mayo, Pass the Bread	101
1003	I Never Promised You Coffee	102
1004	Silly Putty Goes To Washington	100
1005	See No Sound, Hear No Sight	101
1006	Starstruck in Biloxi	102
1007	Stranger In Parrot Ice	101

22. Birincil anahtarları tanımlayın.
23. Yabancı anahtarları tanımlayın.
24. ERM'yi oluşturun.
25. DIRECTOR ve PLAY arasındaki ilişkiyi göstermek için ilişkisel diyagram oluşturun.
26. Belirli bir yönetmen tarafından yönetilen tüm oyunların bir listesini almak için hızlı arama özelliği istediğinizi varsayalım. Hangi tablosu INDEX tablosunun temeli olur ve indeks anahtarı ne olur?
27. Soru 26'da açıklanan INDEX tablosunun kavramsal görünümü ne olurdu? Kavramsal INDEX tablosunun içeriğini tasvir .

Problemler

Problem 1-9'u yanıtlamak için Şekil P3.1'de gösterilen veritabanını kullanın.

Şekil P3.1 Ch03_StoreCo Veritabanı Tabloları

Tablo adı: EMPLOYEE **Veritabanı adı: Ch03_StoreCo**

EMP_CODE	EMP_TITLE	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_DOB	STORE_CODE
1	Mr.	Williamson	John	W	21-May-84	3
2	Ms.	Ratula	Nancy		09-Feb-89	2
3	Ms.	Greenboro	Lottie	R	02-Oct-81	4
4	Mrs.	Rumpersfro	Jennie	S	01-Jun-91	5
5	Mr.	Smith	Robert	L	23-Nov-79	3
6	Mr.	Renselaer	Cary	A	25-Dec-85	1
7	Mr.	Ogallo	Roberto	S	31-Jul-82	3
8	Ms.	Johnsson	Elizabeth	I	10-Sep-88	1
9	Mr.	Eindsmar	Jack	W	19-Apr-75	2
10	Mrs.	Jones	Rose	R	06-Mar-86	4
11	Mr.	Broderick	Tom		21-Oct-92	3
12	Mr.	Washington	Alan	Y	08-Sep-94	2
13	Mr.	Smith	Peter	N	25-Aug-84	3
14	Ms.	Smith	Sherry	H	25-May-86	4
15	Mr.	Olenko	Howard	U	24-May-84	5
16	Mr.	Archialo	Barry	V	03-Sep-80	5
17	Ms.	Grimaldo	Jeanine	K	12-Nov-90	4
18	Mr.	Rosenberg	Andrew	D	24-Jan-91	4
19	Mr.	Rosten	Peter	F	03-Oct-88	4
20	Mr.	Mckee	Robert	S	06-Mar-90	1
21	Ms.	Baumann	Jennifer	A	11-Dec-94	3

Tablo adı: STORE

STORE_CODE	STORE_NAME	STORE_YTD_SALES	REGION_CODE	EMP_CODE
1	Access Junction	1003455.76	2	8
2	Database Corner	1421987.39	2	12
3	Tuple Charge	986783.22	1	7
4	Attribute Alley	944568.56	2	3
5	Primary Key Point	2930098.45	1	15

Tablo adı: BÖLGE

REGION_CODE	REGION_DESCRIPT
1	East
2	West

1. Her tablo için birincil anahtarı ve yabancı (lar)ı tanımlayın. Bir tablonun yabancı anahtarı yoksa, *Yok* yazın.
2. Tablolar varlık bütünlüğü sergiliyor mu? Evet veya hayır olarak yanıtlayın ve ardından açıklayın.
3. Tablolar referans bütünlüğü sergiliyor mu? Evet veya hayır olarak yanıtlayın ve ardından yanıtınızı açıklayın. Tabloda yabancı anahtar yoksa *NA* (Uygulanamaz) yazın.

4. MAĞAZA ve BÖLGE arasındaki ilişki(ler)in tür(ler)ini tanımlayın.
5. STORE ve REGION arasındaki ilişkiyi göstermek için ERD'yi oluşturun.
6. STORE ve REGION arasındaki ilişkiyi göstermek için ilişkisel diyagramı oluşturun.
7. ÇALIŞAN ile MAĞAZA arasındaki ilişki(ler)in tür(ler)ini tanımlayın. (*İpucu:* Her mağaza, biri mağazayı yöneten çok sayıda çalışan istihdam etmektedir).
8. EMPLOYEE, STORE ve REGION arasındaki ilişkileri göstermek için ERD'yi oluşturun.
9. ÇALIŞAN, MAĞAZA ve BÖLGE arasındaki ilişkileri göstermek için ilişkisel diyagramı oluşturun.

Şekil P3.10 Ch03_BeneCo Veritabanı Tabloları

Veritabanı adı: Ch03_BeneCo

Tablo adı: EMPLOYEE

EMP_CODE	EMP_LNAME	JOB_CODE
14	Rudell	2
15	McDade	1
16	Ruellardo	1
17	Smith	3
20	Smith	2

Tablo adı: BENEFIT

EMP_CODE	PLAN_CODE
15	2
15	3
16	1
17	1
17	3
17	4
20	3

Tablo adı: JOB

JOB_CODE	JOB_DESCRIPTION
1	Clerical
2	Technical
3	Managerial

Tablo adı: PLAN

PLAN_CODE	PLAN_DESCRIPTION
1	Term life
2	Stock purchase
3	Long-term disability
4	Dental

Problem 10-16 üzerinde çalışmak için Şekil P3.10'da gösterilen veritabanını kullanın. Veritabanının bu ilişkileri yansıtan dört tablodan oluştuğunu unutmayın

- Bir ÇALIŞAN yalnızca bir İŞ_KODU'na sahiptir, ancak bir İŞ_KODU birçok ÇALIŞAN tarafından tutulabilir.
- Bir ÇALIŞAN birçok PLAN'a katılabilir ve herhangi bir PLAN birçok ÇALIŞAN'a atanabilir.

Ayrıca, M:N ilişkisinin, BENEFIT tablosunun bileşik veya köprü varlık olarak hizmet verdiği iki 1:M ilişkisine bölündüğüne dikkat edin.

10. Veritabanındaki her tablo için birincil anahtarı ve yabancı (lar)ı tanımlayın. Eğer bir tablonun yabancı anahtarı yoksa, *None* yazın.
11. EMPLOYEE ve JOB arasındaki ilişkiyi göstermek için ERD'yi oluşturun.
12. EMPLOYEE ve JOB arasındaki ilişkiyi göstermek için ilişkisel diyagram oluşturun.
13. Tablolar varlık bütünlüğü sergiliyor mu? Evet veya hayır olarak yanıtlayın ve ardından açıklayın.
14. Tablolar referans bütünlüğü sergiliyor mu? Evet veya hayır olarak yanıtlayın ve ardından yanıtınızı açıklayın. Tabloda yabancı anahtar yoksa *NA* (Uygulanamaz) yazın.
15. EMPLOYEE, BENEFIT, JOB ve PLAN arasındaki ilişkileri göstermek için ERD'yi oluşturun.
16. EMPLOYEE, BENEFIT, JOB ve PLAN arasındaki ilişkileri göstermek için ilişkisel diyagramı oluşturun.