Şekil P3.17 Ch03_TransCo Veritabanı Tabloları

Tablo adı: TRUCK **Veritabanı adı: Ch03_TransCo**
anahtar: TRUCK_NUM
Yabancı anahtar: BASE_CODE, TYPE_CODE

TRUCK_NUM	BASE_CODE	TYPE_CODE	TRUCK_MILES	TRUCK_SERIAL_NUM
1001	501	1	32123.5	AA-322-12212-W11
1002	502	1	76984.3	AC-342-22134-Q23
1003	501	2	12346.6	AC-445-78656-Z99
1004		1	2894.3	WQ-112-23144-T34
1005	503	2	45673.1	FR-998-32245-W12
1006	501	2	193245.7	AD-456-00845-R45
1007	502	3	32012.3	AA-341-96573-Z84
1008	502	3	44213.6	DR-559-22189-D33
1009	503	2	10932.9	DE-887-98456-E94

Tablo adı: BASE Birincil anahtar: BASE_CODE
Yabancı anahtar: yok

BASE_CODE	BASE_CITY	BASE_STATE	BASE_AREA_CODE	BASE_PHONE	BASE_MANAGER
501	Murfreesboro	TN	615	123-4567	Andrea D. Gallager
502	Lexington	KY	568	234-5678	George H. Delarosa
503	Cape Girardeau	MO	456	345-6789	Maria J. Talindo
504	Dalton	GA	901	456-7890	Peter F. McAvee

Tablo adı: TYPE Birincil anahtar: TYPE_CODE
Yabancı anahtar: yok

TYPE_CODE	TYPE_DESCRIPTION
1	Single box, double-axle
2	Single box, single-axle
3	Tandem trailer, single-axle

Problem 17-23'ü yanıtlamak için Şekil P3.17'de gösterilen veritabanını kullanın.

17. Her tablo için birincil anahtarı ve yabancı (lar)ı tanımlayın. Bir tablonun yabancı anahtarı yoksa, *Yok* yazın.
18. Tablolar varlık bütünlüğü sergiliyor mu? Evet veya hayır olarak yanıtlayın ve ardından açıklayın.
19. Tablolar referans bütünlüğü sergiliyor mu? Evet veya hayır olarak yanıtlayın ve ardından yanıtınızı açıklayın. Tabloda yabancı anahtar yoksa *NA* (Uygulanamaz) yazın.
20. TRUCK tablosunun aday anahtar(lar)ını tanımlayın.
21. Her tablo için bir üst anahtar ve bir ikincil anahtar tanımlayın.
22. Bu veritabanı için ERD oluşturun.
23. Bu veritabanı için ilişkisel diyagram oluşturun.

Şekil P3.24 Ch03_AviaCo Veritabanı Tabloları

Tablo adı: CHARTER **Veritabanı adı: Ch03_AviaCo**

CHAR_TRIP	CHAR_DATE	CHAR_PILOT	CHAR_COPILOT	AC_NUMBER	CHAR_DESTINATION	CHAR_DISTANCE	CHAR_HOURS_FLOWN	CHAR_HOURS_WAIT	CHAR_FUEL_GALLONS	CHAR_OIL_QTS	CUS_CODE
10001	05-Feb-22	104		2289L	ATL	936.0	5.1	2.2	354.1	1	10011
10002	05-Feb-22	101		2778V	BNA	320.0	1.6	0.0	72.6	0	10016
10003	05-Feb-22	105	109	4278Y	GNV	1574.0	7.8	0.0	339.8	2	10014
10004	06-Feb-22	106		1484P	STL	472.0	2.9	4.9	97.2	1	10019
10005	06-Feb-22	101		2289L	ATL	1023.0	5.7	3.5	397.7	2	10011
10006	06-Feb-22	109		4278Y	STL	472.0	2.6	5.2	117.1	0	10017
10007	06-Feb-22	104	105	2778V	GNV	1574.0	7.9	0.0	348.4	2	10012
10008	07-Feb-22	106		1484P	TYS	644.0	4.1	0.0	140.6	1	10014
10009	07-Feb-22	105		2289L	GNV	1574.0	6.6	23.4	459.9	0	10017
10010	07-Feb-22	109		4278Y	ATL	998.0	6.2	3.2	279.7	0	10016
10011	07-Feb-22	101	104	1484P	BNA	352.0	1.9	5.3	66.4	1	10012
10012	08-Feb-22	101		2778V	MOB	884.0	4.8	4.2	215.1	0	10010
10013	08-Feb-22	105		4278Y	TYS	644.0	3.9	4.5	174.3	1	10011
10014	09-Feb-22	106		4278Y	ATL	936.0	6.1	2.1	302.6	0	10017
10015	09-Feb-22	104	101	2289L	GNV	1645.0	6.7	0.0	459.5	2	10016
10016	09-Feb-22	109	105	2778V	MGY	312.0	1.5	0.0	67.2	0	10011
10017	10-Feb-22	101		1484P	STL	508.0	3.1	0.0	105.5	0	10014
10018	10-Feb-22	105	104	4278Y	TYS	644.0	3.8	4.5	167.4	0	10017

Varış noktaları standart üç harfli havaalanı kodları ile gösterilir. Örneğin, STL = St. MO ATL = Atlanta, GA BNA= Nashville, TN

Tablo adı: AIRCRAFT

AC_NUMBER	MOD_CODE	AC_TTAF	AC_TTEL	AC_TTER
1484P	PA23-250	1833.1	1833.1	101.8
2289L	C-90A	4243.8	768.9	1123.4
2778V	PA31-350	7992.9	1513.1	789.5
4278Y	PA31-350	2147.3	622.1	243.2

AC-TTAF= Uçağın toplam süresi, gövde (saat) AC-TTEL= Toplam süre, sol motor (saat) AC_TTER = Toplam , sağ motor (saat)

Tam gelişmiş bir sistemde, bu tür öznitelik değerleri CHARTER tablo girişleri yayınlandığında uygulama yazılımı tarafından güncellenecektir.

Tablo adı: MODEL

MOD_CODE	MOD_MANUFACTURER	MOD_NAME	MOD_SEATS	MOD_CHG_MILE
B200	Beechcraft	Super KingAir	10	1.93
C-90A	Beechcraft	KingAir	8	2.67
PA23-250	Piper	Aztec	6	1.93
PA31-350	Piper	Navajo Chieftain	10	2.35

Müşteriler, MOD_CHG_MILE oranı kullanılarak gidiş-dönüş mil başına ücretlendirilir. MOD_SEATS sütunu pilot ve yardımcı pilot koltukları dahil olmak üzere uçaktaki toplam koltuk sayısını listeler. Bu nedenle, bir pilot ve bir yardımcı pilot tarafından uçurulan bir PA31-350 seyahatinde sekiz yolcu koltuğu mevcuttur.

Şekil P3.24 Ch03_AviaCo Veritabanı Tabloları *(Devamı)*

Tablo adı: PILOT **Veritabanı adı: Ch03_AviaCo**

EMP_NUM	PIL_LICENSE	PIL_RATINGS	PIL_MED_TYPE	PIL_MED_DATE	PIL_PT135_DATE
101	ATP	ATP/SEL/MEL/Instr/CFII	1	20-Jan-22	11-Jan-22
104	ATP	ATP/SEL/MEL/Instr	1	18-Dec-21	17-Jan-22
105	COM	COMM/SEL/MEL/Instr/CFI	2	05-Jan-22	02-Jan-22
106	COM	COMM/SEL/MEL/Instr	2	10-Dec-21	02-Feb-22
109	COM	ATP/SEL/MEL/SES/Instr/CFII	1	22-Jan-22	15-Jan-22

PİLOT tablosunda gösterilen pilot lisansları ATP = Havayolu Nakliye Pilotu ve COM = Ticari Pilotu içerir. "Talep üzerine" hava hizmetleri işleten işletmeler, Federal Havacılık İdaresi (FAA) tarafından uygulanan Federal Hava Düzenlemelerinin (FAR'lar) 135. Bölümüne tabidir. Bu tür işletmeler "Bölüm 135 operatörleri" olarak bilinir. Bölüm 135 operasyonları, pilotların her altı ayda bir uçuş yeterlilik kontrollerini başarıyla tamamlamalarını gerektirir. "Part 135" uçuş yeterlilik kontrol tarihi PIL_PT135_DATE dosyasına kaydedilir. Ticari olarak uçmak için, pilotlar en az ticari lisansa ve 2. sınıf sağlık sertifikasına sahip olmalıdır (PIL_MED_TYPE= 2.)

PIL_RATINGS şunları içerir
SEL= Tek Motor, Kara
MEL= Çok Motorlu Kara
SES= Tek Motor (Deniz)
Enstrüman. = Enstrüman
CFI= Sertifikalı Uçuş Eğitmeni
CFII= Sertifikalı Uçuş Eğitmeni, Aletli

Tablo adı: EMPLOYEE

EMP_NUM	EMP_TITLE	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_DOB	EMP_HIRE_DATE
100	Mr.	Kolmycz	George	D	15-Jun-62	15-Mar-08
101	Ms.	Lewis	Rhonda	G	19-Mar-85	25-Apr-06
102	Mr.	Vandam	Rhett		14-Nov-78	18-May-13
103	Ms.	Jones	Anne	M	11-May-94	26-Jul-17
104	Mr.	Lange	John	P	12-Jul-91	20-Aug-10
105	Mr.	Williams	Robert	D	14-Mar-95	19-Jun-17
106	Mrs.	Duzak	Jeanine	K	12-Feb-88	13-Mar-18
107	Mr.	Diante	Jorge	D	01-May-95	02-Jul-16
108	Mr.	Wiesenbach	Paul	R	14-Feb-86	03-Jun-13
109	Ms.	Travis	Elizabeth	K	18-Jun-81	14-Feb-16
110	Mrs.	Genkazi	Leighla	W	19-May-90	29-Jun-10

Tablo adı: CUSTOMER

CUS_CODE	CUS_LNAME	CUS_FNAME	CUS_INITIAL	CUS_AREACODE	CUS_PHONE	CUS_BALANCE
10010	Ramas	Alfred	A	615	844-2573	0.00
10011	Dunne	Leona	K	713	894-1238	0.00
10012	Smith	Kathy	W	615	894-2285	896.54
10013	Olowski	Paul	F	615	894-2180	1285.19
10014	Orlando	Myron		615	222-1672	673.21
10015	O'Brian	Amy	B	713	442-3381	1014.56
10016	Brown	James	G	615	297-1228	0.00
10017	Williams	George		615	290-2556	0.00
10018	Farriss	Anne	G	713	382-7185	0.00
10019	Smith	Olette	K	615	297-3809	453.98

Problem 24-31'i yanıtlamak için Şekil P3.24'te gösterilen veritabanını kullanın. AviaCo, dört uçaktan oluşan bir filo kullanarak isteğe bağlı charter uçuş hizmetleri sağlayan bir uçak kiralama şirketidir. Uçaklar benzersiz bir kayıt numarası ile tanımlanır. Bu nedenle, uçak kayıt numarası AIRCRAFT tablosu için uygun bir birincil anahtardır. CHARTER tablosunun CHAR_COPILOT sütunundaki boşluklar, bazı charter seferleri veya bazı uçaklar için yardımcı pilot gerekmediğini gösterir. Federal Havacılık İdaresi (FAA) kuralları, jet uçaklarında ve brüt kalkış ağırlığı 12.500 poundun üzerinde olan uçaklarda bir yardımcı pilot gerektirir. UÇAK tablosundaki uçakların hiçbiri bu zorunluluğa tabi değildir; ancak bazı müşteriler sigorta nedenleriyle bir yardımcı pilotun bulunmasını isteyebilir. Tüm charter seyahatleri CHARTER tablosuna kaydedilir.

Not

Bölümün başlarında, eşanlamlı kelimelerden ve eşanlamlı sözcüklerden kaçınmanız talimatı verilmişti. Bu problemde, hem pilot hem de yardımcı pilot PILOT tablosunda listelenmiştir, ancak EMP_NUM CHARTER tablosunda her ikisi için de kullanılamaz. Bu nedenle, CHARTER tablosunda CHAR_PILOT ve CHAR_COPILOT eşanlamlıları kullanılır.

Çözüm bu durumda işe yarasa da, çok kısıtlayıcıdır ve yardımcı pilot gerekmediğinde boşluklar oluşturur. Daha da kötüsü, mürettebat gereksinimleri değiştikçe bu tür boşluklar çoğalır. Örneğin, AviaCo charter şirketi büyür ve daha büyük uçaklar kullanmaya başlarsa, mürettebat gereksinimleri uçuş mühendisleri ve yükleme ustalarını içerecek şekilde artabilir. Bu durumda CHARTER tablosunun ek mürettebat atamalarını içerecek şekilde değiştirilmesi gerekecektir; CHARTER tablosuna CHAR_FLT_ENGINEER ve CHAR_LOADMASTER gibi niteliklerin eklenmesi gerekecektir. Bu değişiklik göz önüne alındığında, daha küçük bir uçak daha büyük uçaklarda gereken sayıda mürettebat üyesi olmadan bir charter seferini her uçurduğunda, eksik mürettebat üyeleri CHARTER tablosunda ek boşluklar oluşturacaktır.

Problem 27'de bu tasarım eksikliklerini düzeltme şansınız olacak. Bu problem iki önemli noktayı göstermektedir:

1. Eşanlamlı kelimeler kullanmayın. Tasarımınız eş anlamlı kelimelerin kullanılmasını gerektiriyorsa, tasarımı gözden geçirin!
2. Mümkün , veritabanını, veritabanı tablolarında yapısal değişiklikler gerektirmeden büyümeye uyum sağlayacak şekilde tasarlayın. İleriyi planlayın ve değişimin veritabanı üzerindeki etkilerini öngörmeye çalışın.

24. Her bir tablo için, mümkün olduğunda aşağıdakilerin her birini tanımlayın:
    a. Birincil anahtar
    b. Birsüper anahtar
    c. Bir aday anahtar
    d. Yabancı anahtar(lar)
    e. İkincil bir anahtar
25. ERD'yi oluşturun. (*İpucu:* Tablo içeriğine bakın. Bir UÇAK'ın birçok KARTER seyahati uçurabileceğini ancak her KARTER seyahatinin bir UÇAK tarafından uçurulduğunu, bir MODEL'in birçok UÇAK'a referans verdiğini ancak her UÇAK'ın tek bir MODEL'e referans verdiğini vb. keşfedeceksiniz).
26. İlişkisel diyagramı oluşturun.
27. Eş anlamlı sözcüklerin kullanımının yarattığı sorunları ortadan kaldırmak için Problem 25'te oluşturduğunuz ERD'yi değiştirin. (*İpucu*: CHARTER tablo yapısını CHAR_PILOT ve CHAR_COPILOT niteliklerini kaldırarak değiştirin; ardından CHARTER ve EMPLOYEE tablolarını bağlamak için CREW adında bir bileşik tablo oluşturun. Uçuş görevlileri gibi bazı mürettebat üyeleri pilot olmayabilir. Bu nedenle EMPLOYEE tablosu bu ilişkiye girer).
28. Problem 27'de revize ettiğiniz tasarım için ilişkisel diyagramı oluşturun.

Robert Williams (çalışan numarası 105) veya Elizabeth Travis (çalışan numarası 109) tarafından pilot veya yardımcı pilot olarak uçulan charter seferlerine ilişkin verileri görmek istiyorsunuz, ancak her tarafından uçulan charter seferlerini değil. Bu bilgiyi bulmak için Problem 29-31'i tamamlayın.

29. CHARTER tablosuna SELECT ve PROJECT ilişkisel operatörlerini uygulayarak 105 veya 109 numaralı çalışan tarafından uçulan charter seferleri için yalnızca CHAR_TRIP, CHAR_PILOT ve CHAR_COPILOT özniteliklerini döndüren tabloyu oluşturun.
30. CHARTER tablosuna SELECT ve PROJECT ilişkisel operatörlerini uygulayarak hem 105 hem de 109 numaralı çalışan tarafından uçulan charter seferleri için yalnızca CHAR_TRIP, CHAR_PILOT ve CHAR_COPILOT özniteliklerini döndürmek için ortaya çıkacak tabloyu oluşturun.
31. Problem 29'daki sonucunuza Problem 30'daki sonucunuzdan bir FARK ilişkisel operatörü uygulandığında ortaya çıkacak tabloyu oluşturun.

Bölüm

4 Varlık İlişkisi (ER) Modellemesi

Öğrenme Hedefleri

Bu bölümü tamamladıktan sonra şunları yapabileceksiniz:

- **4-1** Varlık ilişkisi bileşenlerinin temel özelliklerini tanımlama
- **4-2** Varlıklar arasındaki ilişkilerin nasıl tanımlandığını, rafine edildiğini ve veritabanı tasarım sürecine nasıl dahil edildiğini açıklamak
- **4-3** ERD bileşenlerinin veritabanı tasarımını ve uygulamasını nasıl etkilediğini açıklayabilme
- **4-4** Gerçek dünya veritabanı tasarımının genellikle çelişen hedeflerin uzlaştırılmasını nasıl gerektirdiğini açıklayın

Önizleme

Bu bölüm, veritabanı tasarımının veri modelleme yönünün kapsamını genişletmektedir. Veri modelleme, veritabanı tasarım yolculuğunun ilk adımıdır ve gerçek dünya nesneleri ile bilgisayarda uygulanan veritabanı modeli arasında bir köprü görevi görür.
Bu nedenle, varlık ilişki diyagramları (ERD'ler) aracılığıyla grafiksel olarak ifade edilen veri modelleme ayrıntılarının önemi abartılamaz.

Varlık ilişki modelinde (ERM) kullanılan temel kavram ve tanımların çoğu Bölüm 2, Veri Modelleri'nde tanıtılmıştır. Örneğin, varlıkların ve ilişkilerin temel bileşenleri ve bunların gösterimi artık size tanıdık gelmelidir.
Bu bölüm, varlıklar arasındaki ilişkilerin grafiksel tasvirini analiz ederek ve bu tasvirlerin başarılı bir tasarım uygulamak için gereken veri zenginliğini özetlemenize nasıl yardımcı olduğunu göstererek çok daha derine iniyor.

Son olarak bu bölüm, veritabanı tasarımında çelişen hedeflerin nasıl bir zorluk oluşturabileceğini ve tasarımdan ödün verilmesini gerektirebileceğini göstermektedir.

Veri Dosyaları ve Mevcut Formatlar

	MS Erişim	Oracle	MS SQL	MySQL
Ch04_TinyCollege	Evet	Evet	Evet	Evet
Ch04_TinyCollege_Alt	Evet	Evet	Evet	Evet
Ch04_ShortCo	Evet	Evet	Evet	Evet
Ch04_Clinic	Evet	Evet	Evet	Evet
Ch04_PartCo	Evet	Evet	Evet	Evet
Ch04_CollegeTry	Evet	Evet	Evet	Evet

Veri Dosyaları cengage.com adresinde mevcuttur

Not

Bu kitap genellikle ilişkisel modele odaklandığından, ERM'nin yalnızca ilişkisel bir araç olduğu sonucuna varabilirsiniz. Aslında, ERM gibi kavramsal modeller bir kuruluşun veri gereksinimlerini anlamak ve tasarlamak için kullanılabilir. Bu nedenle, ERM veritabanı türünden bağımsızdır. Kavramsal modeller veritabanlarının kavramsal tasarımında kullanılırken, ilişkisel modeller veritabanlarının mantıksal tasarımında kullanılır. Ancak, ilişkisel modele önceki bölümden aşina olduğunuz için, ilişkisel model bu bölümde ER yapılarını ve bunların veritabanı tasarımları geliştirmek için nasıl kullanıldığını açıklamak için kapsamlı bir şekilde kullanılmıştır.

4-1 Varlık İlişki Modeli

Bölüm 2, Veri Modelleri ve Bölüm 3, İlişkisel Veritabanı Modeli'nden varlık ilişki modelinin (ERM) bir ERD'nin temelini oluşturduğunu hatırlayın. ERD, son kullanıcı tarafından görüldüğü şekliyle kavramsal veritabanını temsil eder. ERD'ler veritabanının ana bileşenlerini tasvir eder: varlıklar, nitelikler ve ilişkiler. Bir varlık gerçek dünyadaki bir nesneyi temsil ettiğinden, *varlık* ve *nesne* kelimeleri genellikle birbirinin yerine kullanılır. Bu nedenle, bu bölümde geliştirilen Tiny College veritabanı tasarımının varlıkları (nesneleri) öğrenciler, sınıflar, öğretmenler ve derslikleri içerir. Bölümde ERD bileşenlerinin ele alınma sırası, başarılı veritabanı tasarımı ve uygulaması için temel oluşturabilecek ERD'lerin geliştirilmesi için modelleme araçlarının kullanılma şekline göre belirlenir.

Bölüm 2'de, ERD'lerle kullanılan çeşitli gösterimleri de öğrendiniz - orijinal Chen gösterimi ve daha yeni olan Crow's Foot ve UML gösterimleri. İlk iki gösterim bu bölümün başında bazı temel ER modelleme kavramlarını tanıtmak için kullanılmıştır. Bazı gerçek veritabanı modelleme kavramları yalnızca Chen notasyonu kullanılarak ifade edilebilir. Ancak, veritabanlarının *tasarımı ve uygulanması üzerinde* durulduğu için, son Tiny College ER diyagramı örneği için Crow's Foot ve UML sınıf diyagramı notasyonları kullanılmıştır. Uygulamaya yaptığı vurgu , Karga Ayağı gösterimi yalnızca uygulanabilecek olanları temsil edebilir. Başka bir deyişle:

- Chen notasyonu kavramsal modellemeyi desteklemektedir.
- Crow's Foot notasyonu daha uygulama odaklı bir yaklaşımı tercih eder.
- UML notasyonu hem kavramsal hem de uygulama modellemesi için kullanılabilir.

Çevrimiçi İçerik

Modelleme yazılımı yardımıyla ER diyagramlarının nasıl oluşturulacağını öğrenmek için www.cengage.com :adresine gidin

- *Ek A, Lucidchart ile Veritabanı Tasarımı*
- *Ek H, Birleşik Modelleme Dili (UML)*

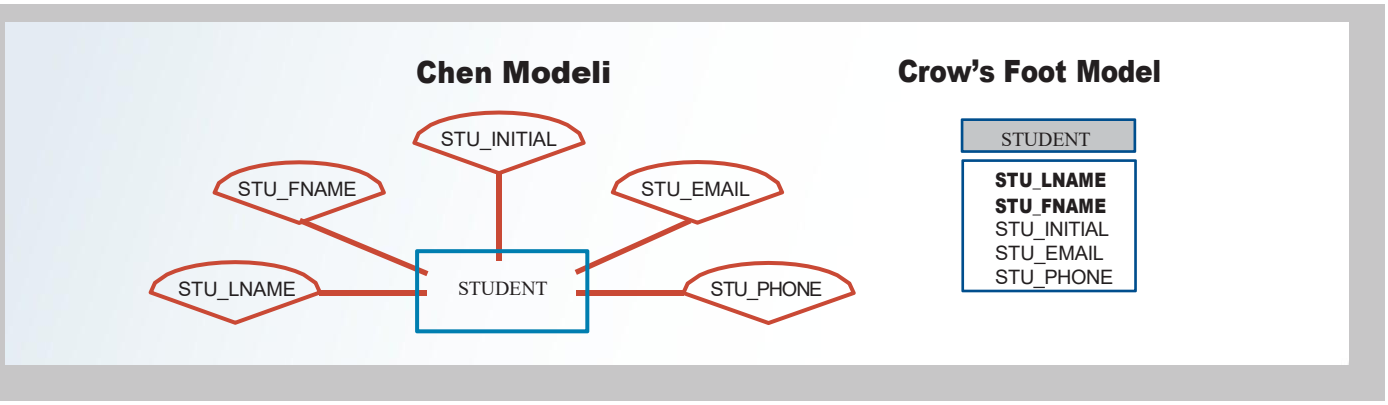
4-1a Tüzel Kişiler

Varlık, son kullanıcının ilgilendiği bir nesnedir. Bölüm 2'de, ER modelleme düzeyinde, bir varlığın aslında tek bir varlık oluşumuna değil, *varlık kümesine* atıfta bulunduğunu öğrendiniz. Başka bir deyişle, ERM'deki bir *varlık*, ilişkisel ortamdaki bir satıra değil, bir tabloya karşılık gelir. ERM, bir tablo satırını bir *varlık örneği* veya *varlık oluşumu* olarak ifade eder. Chen, Crow's Foot ve UML notasyonlarında bir varlık, varlığın adını içeren bir dikdörtgenle temsil edilir. Bir isim olan varlık adı genellikle büyük harflerle yazılır.

4-1b Nitelikler

Öznitelikler varlıkların özellikleridir. Örneğin, STUDENT varlığı diğerlerinin yanı sıra STU_LNAME, STU_FNAME ve STU_INITIAL özniteliklerini içerir. Orijinal Chen gösteriminde, nitelikler ovallerle temsil edilir ve varlık dikdörtgenine bir çizgi ile bağlanır. Her oval temsil ettiği özniteliğin adını içerir. Crow's Foot notasyonunda, öznitelikler varlık dikdörtgeninin altındaki öznitelik kutusuna yazılır. (Bkz. Şekil 4.1.) Chen gösterimi daha fazla yer kapladığından, yazılım satıcıları Karga Ayağı öznitelik gösterimini .

Şekil 4.1 ÖĞRENCİ Varlığının Nitelikleri: Chen ve Karga Ayağı



gerekli öznitelik
ER modellemesinde, bir değere sahip olması gereken bir nitelikBaşka bir deyişle, boş bırakılamaz.

isteğe bağlı nitelik
ER modellemede, bir değer gerektirmeyen bir niteliktir; bu nedenle boş bırakılabilir.

etki alanı
Belirli bir öznitelik için olası değerler kümesi.

Gerekli ve İsteğe Bağlı Özellikler

Gerekli bir nitelik, bir değere sahip olması gereken bir niteliktir; başka bir deyişle, boş bırakılamaz. Şekil 4.1'de gösterildiği gibi, Crow's Foot gösterimindeki iki koyu renkli nitelik, veri girişinin gerekli olacağını gösterir. STU_LNAME ve STU_FNAME veri girişi gerektirir çünkü tüm öğrencilerin bir soyadı ve bir adı olduğu varsayılır. Ancak, öğrencilerin bir göbek adı olmayabilir ve belki de henüz bir telefon numaraları ve e-posta adresleri . Bu nedenle, bu nitelikler varlık kutusunda kalın harflerle gösterilmez. **İsteğe bağlı** bir **nitelik,** değer gerektirmeyen bir niteliktir; bu nedenle boş bırakılabilir.

Etki Alanları (Domains)

Özniteliklerin bir etki alanı vardır. **Etki** alanı, belirli bir öznitelik için olası değerler kümesidir. Örneğin, bir not ortalaması (GPA) özniteliğinin etki alanı (0,4) olarak yazılır çünkü mümkün olan en düşük GPA değeri 0 ve mümkün olan en yüksek değer 4'tür. Bir cinsiyet özniteliğinin etki alanı yalnızca iki olasılıktan oluşur: M veya F (veya eşdeğer başka bir kod). Bir şirketin işe alım tarihi özniteliğinin etki alanı, bir aralığa uyan tüm tarihlerden oluşur (örneğin, şirketin başlangıç tarihinden geçerli tarihe kadar).

Öznitelikler bir etki alanını paylaşabilir. Örneğin, bir öğrenci adresi ve bir profesör adresi tüm olası adreslerin aynı etki alanını paylaşır. Aslında veri sözlüğü, aynı öznitelik adının kullanılması halinde yeni bildirilen bir özniteliğin mevcut bir özniteliğin özelliklerini devralmasına izin verebilir. Örneğin, PROFESÖR ve ÖĞRENCİ varlıklarının her biri ADRES adında bir özniteliğe sahip olabilir ve bu nedenle bir etki alanını paylaşabilir.

Tanımlayıcılar (Birincil Anahtarlar)(PK)

ERM, her bir varlık örneğini benzersiz bir şekilde tanımlayan bir veya daha fazla öznitelik **olan tanımlayıcılar** kullanır. İlişkisel modelde, varlıklar tablolarla eşleştirilir ve varlık tanımlayıcısı tablonun birincil anahtarı (PK) olarak eşleştirilir. ERD'de tanımlayıcıların altı çizilir. Anahtar niteliklerin altı, tablo yapısı için sıkça kullanılan ve **ilişkisel şema** olarak adlandırılan ve aşağıdaki formatı kullanan bir steno gösteriminde de çizilir:

TABLO ADI (**ANAHTAR_ÖZNITELIK 1**, ÖZNITELIK 2, ÖZNITELIK 3, ... ÖZNITELIK K)

Örneğin, bir CAR varlığı şu şekilde temsil edilebilir

CAR (**CAR_VIN**, MOD_CODE, CAR_YEAR, CAR_COLOR)

Her araç benzersiz bir araç kimlik numarası veya CAR_VIN ile tanımlanır.

tanımlayıcı
Her bir varlık örneğini benzersiz bir şekilde tanımlayan bir veya daha fazla öznitelik.

ilişkisel **şema** İlişkisel bir veritabanının organizasyonu veritabanı yöneticisi tarafından tanımlanır.

Bileşik Tanımlayıcılar

İdeal olarak, bir varlık tanımlayıcısı yalnızca tek bir öznitelikten oluşur. Örneğin, Şekil 4.2'deki tablo CLASS_CODE adında tek öznitelikli bir birincil anahtar kullanır. Ancak, birden fazla öznitelikten oluşan bir birincil anahtar **olan bileşik bir tanımlayıcı** kullanmak mümkündür. Örneğin, Tiny College veritabanı yöneticisi her bir CLASS varlık örneğini (oluşumunu) CLASS_CODE yerine CRS_CODE ve CLASS_SECTION bileşik birincil anahtarını kullanarak tanımlamaya karar verebilir. Her iki yaklaşım da her bir varlık örneğini benzersiz bir şekilde tanımlar. Şekil 4.2'de gösterilen CLASS tablosunun yapısı göz önüne alındığında, CLASS_CODE birincil ve CRS_CODE ve CLASS_SECTION kombinasyonu uygun bir aday anahtardır. CLASS_CODE özniteliği CLASS varlığından silinirse, aday anahtar (CRS_CODE ve CLASS_SECTION) kabul edilebilir bir bileşik birincil anahtar haline gelir.

bileşik tanımlayıcı
ER modellemesinde, birden fazla öznitelikten oluşan bir anahtar.

Şekil 4.2 SINIF Tablosu (Varlık) Bileşenleri ve İçeriği

Veritabanı adı: Ch04_TinyCollege

CLASS_CODE	CRS_CODE	CLASS_SECTION	CLASS_TIME	ROOM_CODE	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
10025	MATH-243	1	Th 6:00-8:40 p.m.	DRE155	325

Not

Bölüm 3'ün KURS ve arasında genel kabul görmüş bir ayrım yaptığını hatırlayın. Bir SINIF, bir KURS teklifinin belirli bir zamanını ve yerini oluşturur. Bir sınıf, dersin tanımı, zamanı ve yeri ya da bölümü ile tanımlanır. Veritabanı I, Bölüm 2; Veritabanı I, Bölüm 5; Veritabanı I, Bölüm 8; ve Spread- sheet II, Bölüm 6 derslerini veren bir profesör düşünün. Profesör iki ders (Veritabanı I ve Elektronik Tablo II), ancak dört sınıf veriyor. Tipik olarak, KURS teklifleri bir kurs kataloğunda basılırken, SINIF teklifleri her dönem için bir sınıf programında basılır.

Şekil 4.2'deki CLASS_CODE birincil anahtar olarak kullanılırsa, CLASS varlığı şu şekilde olabilir steno formunda aşağıdaki gibi temsil edilir:

CLASS (**CLASS_CODE**, CRS_CODE, CLASS_SECTION, CLASS_TIME, ROOM_CODE, PROF_NUM)

Öte yandan, CLASS_CODE silinirse ve bileşik birincil anahtar CRS_CODE ve CLASS_SECTION'ın birleşimi olursa, CLASS varlığı aşağıdaki gibi gösterilebilir:

CLASS (**CRS_CODE**, **CLASS_SECTION**, CLASS_TIME, ROOM_CODE, PROF_NUM)

Varlık gösteriminde *her iki* anahtar niteliğin de altının çizili olduğuna dikkat edin.

Bileşik ve Basit Öznitelikler

Bileşik öznitelik Ek öznitelikler elde etmek için daha da alt bölümlere ayrılabilen bir öznitelik. Örneğin, 615-898-2368 gibi bir telefon numarası bir alan kodu (615), bir santral numarası (898) ve dört basamaklı bir koda (2368) bölünebilir. *Basit özellik* ile karşılaştırın.

Öznitelikler basit veya bileşik olarak sınıflandırılır. Bileşik **öznitelik**, bileşik anahtar ile karıştırılmamalıdır, ek öznitelikler elde etmek için daha da alt bölümlere ayrılabilen bir özniteliktir. Örneğin, ADRES özniteliği sokak, şehir, eyalet ve posta kodu olarak alt bölümlere ayrılabilir. Benzer şekilde, PHONE_NUMBER özniteliği alan kodu ve santral numarası olarak alt bölümlere ayrılabilir. **Basit** bir **öznitelik**, alt bölümlere ayrılamayan bir **özniteliktir**. Örneğin, yaş, cinsiyet ve medeni durum basit öznitelikler olarak sınıflandırılabilir. Ayrıntılı sorguları kolaylaştırmak için, bileşik öznitelikleri bir dizi basit özniteliğe dönüştürmek akıllıca olacaktır.

basit nitelik Anlamlı bileşenlere ayrılamayan bir nitelik. *Bileşik öznitelik* ile karşılaştırın.

Veritabanı tasarımcısı her zaman bileşik öznitelikler için tetikte olmalıdır. İş kurallarının politikaları basitleştirmek için bileşik öznitelikler kullanması yaygındır ve kullanıcılar genellikle çevrelerindeki varlıkları bileşik kullanarak tanımlar. Örneğin, Tiny College'daki bir kullanıcının bir öğrencinin adını, adresini ve telefon numarasını bilmesi gerekebilir. Tasarımcı bunların bileşik öznitelikler olduğunu kabul etmeli ve bileşiği basit özniteliklere ayırmanın doğru yolunu belirlemelidir.

Tek Değerli Öznitelikler

tek değerli öznitelik Yalnızca bir değere sahip olabilen bir **öznitelik**.

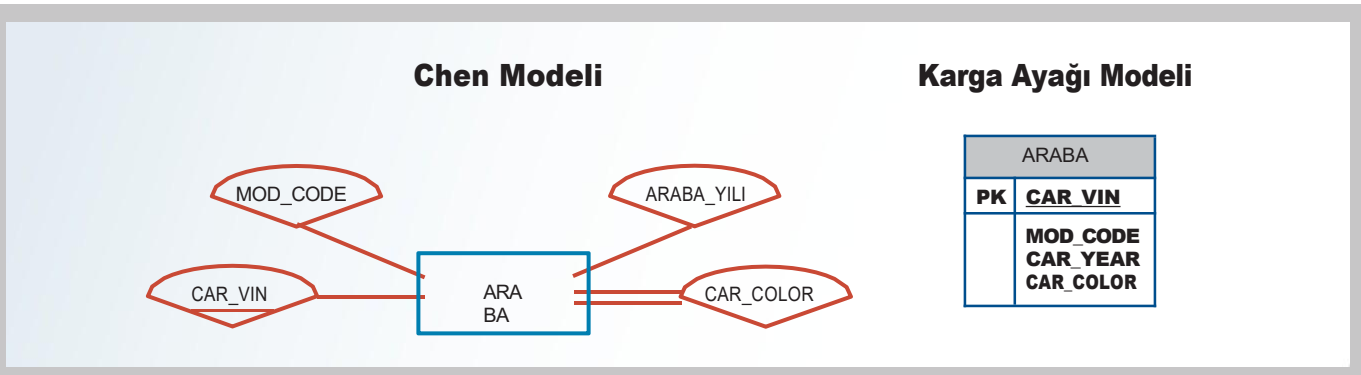
Tek değerli bir öznitelik, yalnızca tek bir değere sahip olabilen bir özniteliktir. Örneğin, bir kişinin yalnızca bir Sosyal Güvenlik numarası olabilir ve üretilen bir parçanın yalnızca bir seri numarası olabilir. *Tek değerli bir özniteliğin mutlaka basit bir öznitelik olması gerekmediğini unutmayın*. Örneğin, bir parçanın seri numarası (SE-08-02-189935 gibi) tek değerlidir, ancak parçanın üretildiği bölge (SE), bu bölge içindeki tesis (08), tesis içindeki vardiya (02) ve parça numarası (189935) olarak alt bölümlere ayrılabildiği için bileşik bir özniteliktir.

Çok Değerli Öznitelikler

çok değerli öznitelik Tek bir varlık oluşumu için birçok değere sahip olabilen bir öznitelik. Örneğin, bir EMP_DEGREE özniteliği "BBA, MBA, PHD" dizesini saklayabilir sahip olunan üç farklı dereceyi belirtmek için.

Çok değerli öznitelikler, birçok değere sahip olabilen özniteliklerdir. Örneğin, bir kişinin birkaç üniversite diploması olabilir ve bir evde her biri kendi numarasına sahip birkaç farklı telefon olabilir. Benzer şekilde, bir arabanın rengi tavan, gövde ve trim için birçok renge bölünebilir. Chen ERM'de çok değerli nitelikler, niteliği varlığa bağlayan çift çizgi ile gösterilir. Karga Ayağı notasyonu çok değerli öznitelikleri tanımlamaz. Şekil 4.3'teki ERD, şimdiye kadar tanıtılan tüm bileşenleri içerir; CAR_VIN'in birincil anahtar olduğuna ve CAR_COLOR'un CAR varlığının çok değerli bir özniteliği olduğuna dikkat edin.

Şekil 4.3 Bir Varlıktaki Çok Değerli Öznitelik



Not

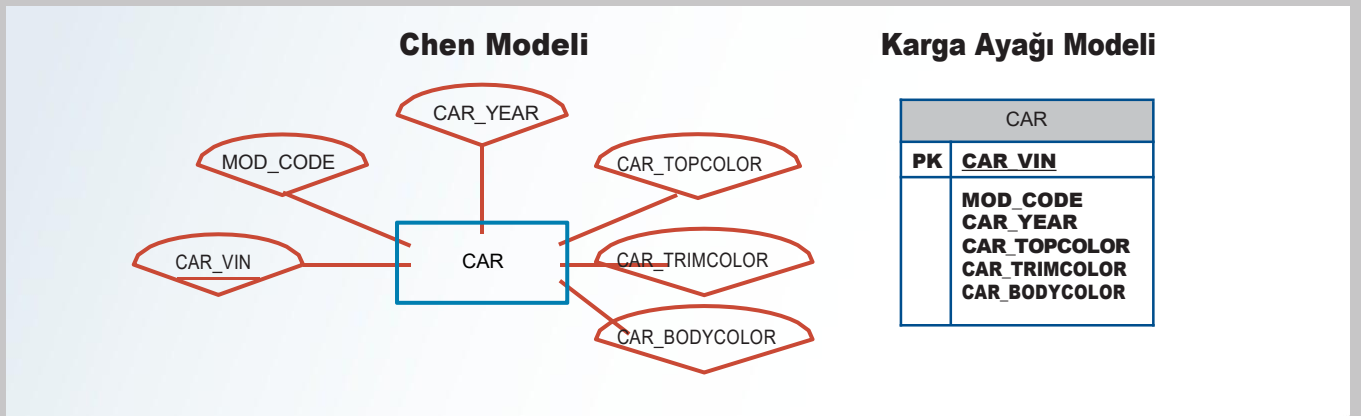
Şekil 4.3'teki ERD modellerinde, CAR varlığının yabancı anahtarı (FK) MOD_CODE olarak yazılmıştır. Bu nitelik varlığa manuel olarak eklenmiştir. Aslında, veritabanı modelleme yazılımının doğru kullanımı, ilişki tanımlandığında FK'yı otomatik olarak üretecektir. Buna ek olarak, yazılım FK'yı uygun şekilde etiketleyecek ve FK'nın uygulama ayrıntılarını bir veri sözlüğüne yazacaktır. (Bunun nasıl çalıştığını Ek A'da görebilirsiniz, Lucidchart ile Veritabanı Tasarlama: Bir Öğretici, www.cengage.com.)

Çok Değerli Özniteliklerin Uygulanması

Kavramsal model M:N ilişkilerini ve çok değerli nitelikleri işleyebilse de, *bunları RDBMS'de uygulamamalısınız*. Bölüm 3'ten, ilişkisel tabloda her sütun ve satır kesişiminin tek bir veri değerini temsil ettiğini hatırlayın. Dolayısıyla, eğer çok değerli nitelikler mevcutsa, tasarımcı iki olası hareket tarzından birine karar vermelidir:

1. Orijinal varlık içinde, orijinal çok değerli özniteliğin her bileşeni için bir tane olmak üzere birkaç yeni öznitelik oluşturun. Örneğin, CAR varlığının CAR_COLOR özniteliği bölünerek yeni CAR_TOPCOLOR, CAR_BODYCOLOR ve CAR_TRIMCOLOR özniteliklerini oluşturulabilir ve bunlar daha sonra CAR varlığına atanır. (Bkz. Şekil 4.4.)

Şekil 4.4 Çok Değerli Özniteliği Yeni Özniteliklere Bölme



Bu çözüm işe yarıyor gibi görünse de, benimsenmesi tabloda büyük yapısal sorunlara yol açabilir. Yalnızca her örnek çok değerli öznitelik için aynı sayıda değere sahip olacaksa ve hiçbir örnek daha fazla değere sahip olmayacaksa kabul edilebilir. Ancak, bu durumda bile

Ortamdaki yeni değişikliklerin hiçbir zaman bir örneğin eskisinden daha fazla değere sahip olacağı bir durum yaratmayacağına dair bir kumardır. Örneğin, bazı araçlar için logo rengi gibi ek renk bileşenleri eklenirse, tablo yapısının yeni renk bölümünü barındıracak şekilde değiştirilmesi gerekir. Bu durumda, bu tür renk bölümlerine sahip olmayan araçlar, mevcut olmayan bileşenler için boş değerler oluşturur veya bu bölümler için renk girişleri "uygulanamaz" anlamına gelmek üzere N/A olarak girilir. (Şekil 4.4'teki çözüm çok değerli bir niteliği yeni niteliklere bölmektir, ancak bu tür bir çözümün çalışanların derece ve sertifikalarını içeren bir çalışan varlığına uygulandığında neden olacağı sorunları hayal edin. Bazı çalışanların 10 derecesi ve sertifikası varken çoğunun daha az derecesi ve sertifikası varsa ya da hiç yoksa, derece/sertifika özniteliklerinin sayısı 10 olacaktır ve bu öznitelik değerlerinin çoğu çoğu çalışan için null olacaktır). Kısacası, çözüm 1'in uygulandığını görmüş olsanız da, bu her zaman kabul edilebilir değildir.

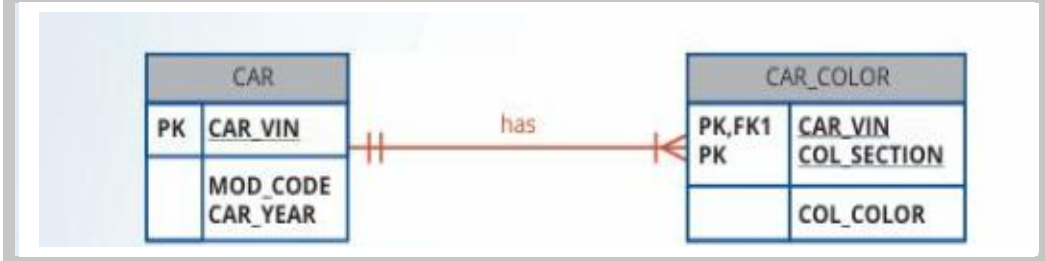
2. Orijinal çok değerli niteliğin bileşenlerinden oluşan yeni bir varlık oluşturun. Bu yeni varlık, tasarımcının otomobilin farklı bölümleri için renk tanımlamasına olanak tanır (bkz. Tablo 4.1). Daha sonra, bu yeni CAR_COLOR varlığı orijinal CAR varlığıyla 1:M ilişkisi içinde ilişkilendirilir.

Tablo 4.1'de gösterilen yaklaşımı kullanarak, bir yan fayda bile elde edersiniz: artık tablo yapısını değiştirmek zorunda kalmadan gerektiği kadar renk atayabilirsiniz. Şekil 4.5'te gösterilen ERM, Tablo 4.1'de listelenen bileşenleri yansıtır. Bu, çok değerli niteliklerle başa çıkmak için tercih edilen yoldur. Orijinal varlık ile 1:M ilişkisi içinde yeni bir varlık oluşturmak çeşitli faydalar sağlar: daha esnek, genişletilebilir bir çözümdür ve ilişkisel model ile uyumludur!

Tablo 4.1 Çok Değerli Özniteliğin Bileşenleri

Bölüm	Renk
Üst	Beyaz
Vücut	Mavi
Döşeme	Altın
İç Mekan	Mavi

Şekil 4.5 Çok Değerli Bir Özniteliğin Bileşenlerinden Oluşan Yeni Bir Varlık Kümesi



Not

Microsoft Access tarafından üretilenler gibi ilişkisel *diyagramlara* bakmaya alışkınsanız, *ilişkisel diyagramdaki* ilişki çizgisinin PK'dan FK'ya doğru çizildiğini görmeyi beklersiniz. Ancak, ilişkisel diyagram konvansiyonunun ERD'de yansıtılması gerekmez. Bir ERD'de odak noktası, bu ilişkilerin grafiksel olarak nasıl bağlandığından ziyade varlıklar ve ilişkilerdir. Hem yatay hem de dikey olarak yerleştirilmiş varlıkları içeren karmaşık bir ERD'de, ilişki çizgilerinin yerleşimi büyük ölçüde tasarımcının okunabilirliğini artırma kararı tarafından belirlenir. (ERD'nin tasarımcılar ve son kullanıcılar arasındaki iletişim için kullanıldığını unutmayın).

Türetilmiş Öznitelikler

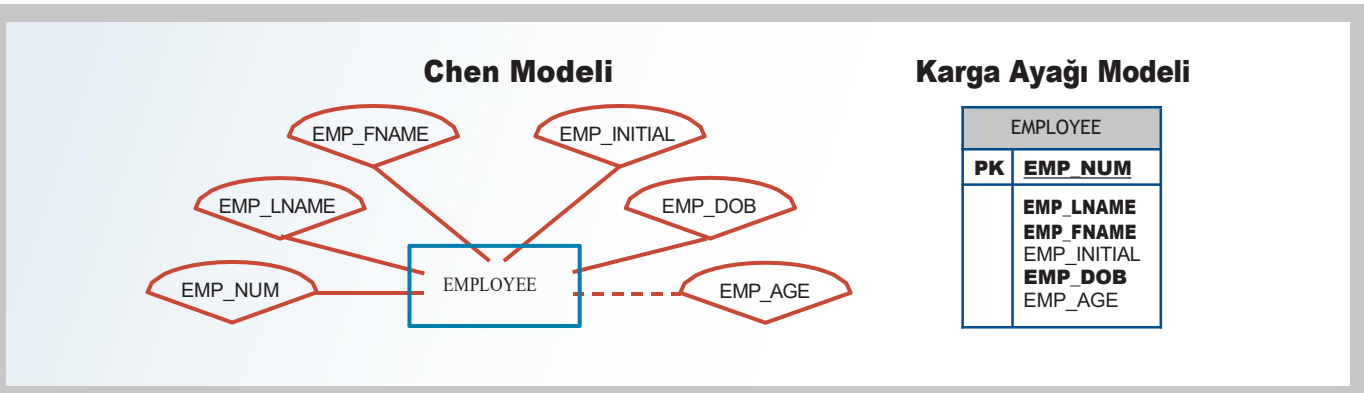
türetilmiş öznitelik
Varlık içinde fiziksel olarak bulunmayan ve bir algoritma aracılığıyla türetilen bir öznitelik. Örneğin, Yaş özniteliği doğum tarihinin geçerli tarihten çıkarılmasıyla elde edilebilir.

Son olarak, **türetilmiş** bir **öznitelik**, değeri diğer özniteliklerden hesaplanan (türetilen) bir özniteliktir. Türetilen özniteliğin veritabanında fiziksel olarak saklanması gerekmez; yerine, bir algoritma kullanılarak türetilebilir. Örneğin, bir çalışanın yaşı, EMP_AGE, geçerli tarih ile EMP_DOB arasındaki farkın tamsayı değeri hesaplanarak bulunabilir. Microsoft Access kullanıyorsanız, INT((DATE() - EMP_DOB)/365) formülünü kullanırsınız. Microsoft SQL Server'da, DATEDIFF("DAY", EMB_DOB, GETDATE())/365 formülünü kullanırsınız; burada DATEDIFF, tarihler arasındaki farkı hesaplayan bir işlevdir. Oracle kullanıyorsanız, TRUNC((SYSDATE - EMP_DOB)/365,0) kullanırsınız.

Benzer şekilde, bir siparişin toplam maliyeti, sipariş edilen miktarın birim fiyatla çarpılmasıyla elde edilebilir. Ya da tahmini ortalama hız, yolculuk mesafesinin yolda harcanan süreye bölünmesiyle elde edilebilir. Türetilmiş bir nitelik Chen gösteriminde, nitelik ile varlığı birbirine bağlayan kesikli bir çizgi ile gösterilir. (Bkz. Şekil 4.6.) Crow's Foot notasyonunda türetilmiş niteliği diğer ayırmak için bir yöntem yoktur.

Türetilmiş öznitelikler bazen *hesaplanmış öznitelikler* olarak da adlandırılır. Türetilmiş bir özniteliğin hesaplanması, aynı satırda bulunan iki öznitelik değerinin toplanması kadar basit olabilir veya birçok tablo satırında (aynı tablodan veya farklı tablolardan) bulunan değerlerin toplamının bir sonucu olabilir. Türetilmiş öznitelikleri veritabanı tablolarında saklama kararı, işlem gereksinimlerine ve belirli bir uygulamaya getirilen kısıtlamalara bağlıdır. Tasarımcı, tasarımı bu tür kısıtlamalara uygun olarak dengeleyebilmelidir. Tablo 4.2, türetilmiş özniteliklerin veritabanında saklanmasının (veya) avantaj ve dezavantajlarını göstermektedir.

Şekil 4.6 Türetilmiş Bir Özniteliğin Tasviri



Tablo 4.2 Türetilmiş Öznitelikleri Saklamanın Avantaj ve Dezavantajları

	Türetilmiş Öznitelik	
	Depolandı	**Saklanmıyor**
Avantaj	CPU işlem döngülerinden tasarruf sağlar Veri erişim süresinden tasarruf sağlar Veri değeri kolayca elde edilebilir Geçmiş verileri takip etmek için kullanılabilir	Depolama alanından tasarruf sağlar Hesaplama her zaman mevcut değeri verir
Dezavantaj	Türetilen değerin güncel olmasını sağlamak için sürekli bakım gerektirir, özellikle hesaplamada kullanılan herhangi bir değer değişirse	CPU işlem döngülerini kullanır Veri erişim süresini artırır Sorgulara kodlama karmaşıklığı ekler

Not

Modern veritabanı yönetim sistemleri, hesaplanmış veya hesaplanan verileri desteklemek için yeni veri türü tanımları sağlar. Örneğin, MS Access'te Hesaplanmış veri türünü kullanabilirsiniz. SQL Server, Oracle ve MySQL de türetilmiş veya hesaplanmış niteliklerin tanımlanmasını destekler.

4-1c İlişkiler

Katılımcılar
Bir ilişkiye katılan varlıklar için bir ER terimi. Örneğin, "PROFESÖR SINIF ÖĞRETİR" ilişkisinde, *öğretir* PROFESÖR ve SINIF katılımcılarına dayanır.

Bölüm 2'den bir ilişkinin varlıklar arasındaki bir birliktelik olduğunu hatırlayın. Bu varlıklar Bir ilişkiye katılanlar aynı zamanda **katılımcılar** olarak da bilinir ve her ilişki, ilişkiyi tanımlayan bir adla tanımlanır. İlişki adı aktif veya pasif bir fiildir; örneğin, bir ÖĞRENCİ bir SINIF *alır,* bir PROFESÖR bir SINIF *öğretir,* bir BÖLÜM bir PROFESÖR *istihdam eder,* bir BÖLÜM bir ÇALIŞAN *tarafından yönetilir* ve bir UÇAK bir MÜRETTEBAT *tarafından uçurulur.*

Varlıklar arasındaki ilişkiler her zaman her iki yönde de çalışır. CUSTOMER ve INVOICE adlı varlıklar arasındaki ilişkiyi tanımlamak için şunu belirtirsiniz:

- Bir MÜŞTERİ çok sayıda FATURA oluşturabilir.
- Her FATURA bir MÜŞTERİ tarafından oluşturulur.

MÜŞTERİ ve FATURA arasındaki ilişkinin her iki yönünü de bildiğiniz için, bu ilişkinin 1:M olarak sınıflandırılabileceğini görmek kolaydır.

İlişkinin sadece bir tarafını biliyorsanız, ilişki sınıflandırmasını belirlemek zordur. Örneğin, şunu belirtirseniz:

Bir BÖLÜM, bir ÇALIŞAN tarafından yönetilir.

İlişkinin 1:1 mi yoksa 1:M mi olduğunu bilmiyorsunuz. Bu nedenle, "Bir çalışan birden fazla bölümü yönetebilir mi?" sorusunu sormalısınız. Cevap evet , ilişki 1:M'dir ve ilişkinin ikinci kısmı şu şekilde yazılır:

Bir ÇALIŞAN birçok BÖLÜMÜ yönetebilir.

Bir çalışan birden fazla bölümü yönetemiyorsa, ilişki 1:1'dir ve ilişkinin ikinci kısmı şu şekilde yazılır:

Bir ÇALIŞAN yalnızca bir BÖLÜMÜ yönetebilir.

Çevrimiçi İçerik Çünkü tam ve eksiksiz tanımının dikkatli bir şekilde doğru iş kuralları iyi veritabanı tasarımı için çok önemlidir, bunların türetilmesi şu bölümde ayrıntılı olarak incelenmiştir *Ek B, Üniversite Laboratuvarı: Kavramsal Tasarım*. Bu bölümde öğrenmekte olduğunuz modelleme becerileri şunlardır gerçek bir veritabanı tasarımının uygulanır Ek B'de yer almaktadır. Ek B'de gösterilen ilk tasarım daha sonra *Ek C, Üniversite Laboratuvarı*'nda değiştirilmiştir: *Kavramsal Tasarım Doğrulaması, Mantıksal Tasarım ve Uygulama*. (Her iki eke de www.cengage.com .)adresinden ulaşılabilir Varlıklar arasındaki ilişkinin sınıflandırılması. Sınıflandırmalar 1:1, 1:M ve M:N'yi içerir.

4-1 d Bağlanabilirlik ve Kardinalite

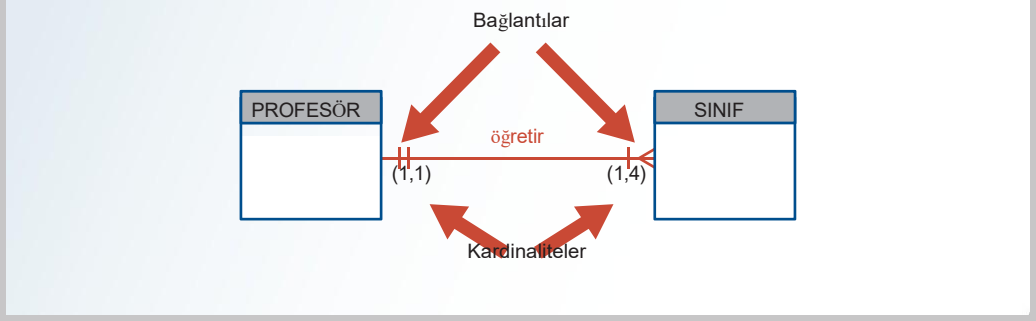
Bölüm 2'de varlık ilişkilerinin bire-bir, bire-çok veya çoka-çok olarak sınıflandırılabileceğini öğrendiniz. Ayrıca bu tür ilişkilerin Chen ve Crow's Foot gösterimlerinde nasıl tasvir edildiğini de öğrendiniz. **Bağlanabilirlik** terimi, ilişki sınıflandırmasını tanımlamak için kullanılır.

kardinalite
Bağlanabilirliğe belirli bir değer atayan ve ilgili varlığın tek bir ilişkili izin verilen varlık oluşumları aralığını ifade eden bir özellik.

Kardinalite, ilgili varlığın bir oluşumu ile ilişkilendirilen minimum ve maksimum varlık oluşum sayısını ifade eder. ERD'de kardinalite, (x,y) formatı kullanılarak varlıkların yanına uygun sayılar yerleştirilerek gösterilir. İlk değer ilişkili varlıkların minimum sayısını temsil ederken, ikinci değer ilişkili varlıkların maksimum sayısını temsil eder. Crow's Foot modelleme notasyonunu kullanan birçok veritabanı tasarımcısı, ER diyagramı üzerinde belirli kardinaliteleri göstermez çünkü kardinaliteler tarafından tanımlanan belirli sınırlar veritabanı tasarımı aracılığıyla doğrudan uygulanamaz. Buna paralel olarak, bazı Crow's Foot ER modelleme araçları sayısal kardinalite aralığını diyagrama yazdırmaz; bunun yerine, gösterilmesini istiyorsanız metin olarak ekleyebilirsiniz. Crow's Foot gösteriminde belirli kardinaliteler diyagrama dahil edilmediğinde, kardinalite Şekil 4.7'de gösterilen ve bağlantıyı ve katılımı (daha sonra ele alınacaktır) tanımlayan sembollerin kullanımıyla ima edilir.

Varlık oluşumlarının minimum ve maksimum sayısını bilmek uygulama yazılımı düzeyinde çok kullanışlıdır. Örneğin Tiny College, en az 10 öğrencinin kayıtlı olmadığı bir sınıfta ders verilmemesini sağlamak isteyebilir. Benzer şekilde, sınıf yalnızca 30 öğrenci alabiliyorsa, uygulama yazılımı sınıfa kayıtları sınırlamak için bu kardinaliteyi kullanmalıdır. Ancak, DBMS'nin tablo düzeyinde kardinalitelerin uygulanmasını gerçekleştiremeyeceğini unutmayın; bu yetenek uygulama yazılımı veya tetikleyiciler tarafından sağlanır. Tetikleyicilerin nasıl oluşturulacağını ve çalıştırılacağını Bölüm 8, Gelişmiş SQL'de öğreneceksiniz.

Şekil 4.7 ERD'de Bağlanabilirlik ve Kardinalite



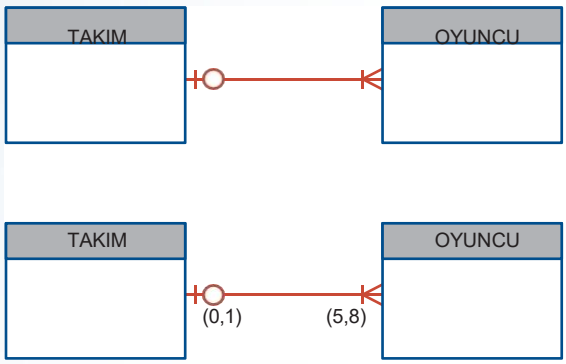
Şekil 4.7'deki Crow's Foot diyagramını incelerken, kardinalitelerin *ilgili* varlıktaki oluşum sayısını temsil ettiğini unutmayın. Örneğin, "PROFESSOR teaches CLASS" ilişkisinde CLASS varlığının yanındaki kardinalite (1,4), her profesörün en fazla dört sınıfa ders verdiğini gösterir; bu da PROFESSOR tablosunun birincil anahtar değerinin CLASS tablosunda yabancı anahtar değeri olarak en az bir kez ve en fazla dört kez yer aldığı anlamına gelir. Kardinalite (1,N) olarak yazılmış olsaydı, bir profesörün verebileceği ders sayısı için bir üst sınır olmazdı. Benzer şekilde, PROFESSOR varlığının yanındaki (1,1) kardinalitesi, her sınıfın bir ve yalnızca bir profesör tarafından verildiğini gösterir. Yani, her CLASS varlık oluşumu PROFESSOR'da bir ve yalnızca bir varlık oluşumu ile ilişkilidir.

Bağlanabilirlik ve maksimum kardinalitenin benzer kavramlar olduğunu unutmayın. Her ikisi de bir tablodaki ilgili tablodaki bir satırla ilişkilendirilebilecek maksimum satır sayısını ele alır. Bağlanabilirlik bu soruya verilen yarı belirsiz bir yanıttır, maksimum kardinalite aynı soruya verilen belirli bir yanıttır. Bağlanabilirlik yarı belirsizdir, çünkü "1" bağlanabilirliği spesifiktir - bir sayısı anlamına gelir. "Çok" bağlantısı ise belirsizdir - birden büyük herhangi bir sayı anlamına gelir. Doğal olarak, eğer bağlantı "1" ise, maksimum kardinalite 1 olacaktır. Eğer bağlantı "çok" ise, o zaman maksimum kardinalite tam olarak kaç tane olduğunu gösterecektir.

Katılım ve minimum kardinalite arasında da benzer bir ilişki vardır, ancak bir tablodaki kaç satırın ilgili tablodaki bir satırla ilişkili olabileceği sorusunu ele alırlar. Katılım da yarı belirsiz bir cevaptır. İlişkili olması gereken en az satır sıfır ise, katılım isteğe bağlıdır. İlişkili olması gereken en az satır sıfırdan büyükse, katılım zorunludur. Katılım, en az satır sayısının tam olarak ne belirtmez, sadece sıfırdan fazla olduğunu belirtir. Minimum kardinalite belirli bir sayı sağlar.

Bir gençlik basketbol ligi örneğini düşünün. Oyuncular oynamak için kayıt yaptırır. Bir takım oluşturmak için yeterli sayıda oyuncu mevcut olduğunda, takım oluşturulur ve oyuncular bu takıma atanır. Bir takımın en az beş oyuncusu olmalıdır; , en az beş oyuncu bir takıma atanmaya hazır olana kadar bir takım oluşturulmayacaktır. Bir takımda en fazla sekiz oyuncu olabilir. Bir oyuncu sadece bir takıma atanabilir. Şekil 4.8'deki modelin üst versiyonunda gösterildiği gibi, Oyuncu tablosunda Takım tablosundaki tek bir satırla ilişkilendirilebilecek en az satır sayısı beştir ve bu sayı sıfırdan büyüktür, bu nedenle Oyuncudan Takıma katılım zorunludur. Oyuncu tablosunda Takım tablosundaki tek bir satırla ilişkilendirilebilecek en büyük satır sayısı sekizdir ve bu sayı birden büyüktür, dolayısıyla bağlantı çoktur. Şekildeki modelin ikinci versiyonu, belirli beş ve sekiz sayılarını gösteren kardinaliteler ekler.

Şekil 4.8 TAKIM ve OYUNCUNUN İki Temsili



Açıkçası, kardinaliteler katılım ve bağlanabilirlikten daha fazla bilgi sağlar. O halde, kardinalite daha üstün görünürken neden katılım ve bağlanabilirliği kullandığımızı merak etmek doğru olacaktır. Sorun, DBMS'nin önceki örnekteki beş ve sekiz gibi belirli değerleri zorlamasının bir yolu olmamasıdır. Bu tür özel sayısal gereklilikleri uygulamak için uygulama mantığı kodlanmalıdır. Bu kod ön uç uygulamasına yazılabilir ya da saklı yordam veya tetikleyici olarak veritabanına gömülebilir; ancak VTYS bunu sadece normal, ilişkisel bütünlük denetimi biçimleriyle uygulayamaz. VTYS'nin normal, ilişkisel bütünlük denetimi yoluyla uygulayabileceği gereksinim türleri, daha belirsiz katılım ve bağlantı ifadeleriyle temsil edilebilir. Takım ve Oyuncu arasındaki bağlantıların 1:M olduğunu bilmek size yabancı anahtarı nasıl yerleştireceğinizi söyler. Katılımın zorunlu olduğunu bilmek size bir NOT NULL kısıtlamasının (Bölüm 7, Yapısal Sorgu Diline Giriş'te tanıtılmıştır) gerekli olduğunu söyler. Katılım ve bağlantının size ilgili veritabanı tasarım bilgilerini vermesi ve okunması kolay bir gösterimle modele kolayca dahil edilmesi, bunları veri modellerindeki baskın veri parçaları haline getirir.

Not

Bu bölümde bağlantılar, katılımlar ve kardinaliteler ele alınmaktadır. Diğer ortamlarda bir ilişkinin tüm bu özelliklerinin ele alınmayabileceğini unutmayın. Uygulamada, belirli minimum ve maksimum kardinalitelerin atlanması yaygındır. Bu gibi durumlarda, katılımlara ve bağlantılara sırasıyla minimum ve maksimum kardinalite olarak atıfta bulunmak yaygındır. Bunda yanlış bir şey yoktur. "Kardinalite" terimi temelde "kaç tane" ifadesinin bir ifadesidir. Kaç tane sorusunun belirli sayı yanıtlarının yokluğunda, kaç tane sorusunun daha genel yanıtlarını kardinalite olarak adlandırmak uygundur. Veritabanı tasarımcıları bağlanabilirliği "maksimum kardinalite" ve katılımı "minimum kardinalite" olarak adlandırırsa işinizde şaşırmayın. Yanlış değiller, sadece bu bölümde yaptığımız gibi modellenebilecek olası verilerin tamamını dikkate almıyorlar.

Bağlantılar ve kardinaliteler, Bölüm 2'de tanıtılan ve iş kuralları olarak bilinen özlü ifadelerle belirlenir. Bir kuruluşun veri ortamının kesin ve ayrıntılı bir tanımından türetilen bu kurallar, ERM'nin varlıklarını, niteliklerini, ilişkilerini, bağlantılarını, kardinalitelerini ve kısıtlamalarını da belirler. İş kuralları ERM'nin bileşenlerini tanımladığından, tüm uygun iş kurallarının tanımlandığından emin olmak, bir veritabanı tasarımcısının işinin önemli bir parçasıdır.

Not

ER diyagramında kardinalitelerin yerleşimi geleneksel bir konudur. Chen notasyonu kardinaliteleri ilgili varlığın yanına yerleştirir. Karga Ayağı ve UML diyagramları, kardinaliteleri uygulandıkları varlığın yanına yerleştirir.

4-1e Varoluş Bağımlılığı

Bir varlık, veritabanında yalnızca ilgili başka bir varlık oluşumuyla ilişkilendirildiğinde var olabiliyorsa varlığa **bağımlı** olduğu söylenir. Uygulama açısından, bir varlık zorunlu bir yabancı anahtara, yani boş olamayan bir yabancı anahtar özniteliğine sahipse varlığa bağımlıdır. Örneğin, bir çalışan vergi stopajı amacıyla bir veya daha fazla bakmakla yükümlü olduğu kişiyi talep etmek isterse, "EMPLOYEE claims DEPENDENT" ilişkisi uygun olacaktır. Bu durumda, DEPENDENT varlığı açıkça EMPLOYEE varlığına bağlıdır çünkü bağımlı kişinin veritabanında EMPLOYEE'den ayrı olarak var olması mümkün değildir.

Bir varlık, ilgili tüm **varlıklarından** ayrı olarak var olabiliyorsa, o zaman **varlıktan bağımsızdır** ve **güçlü varlık** veya **normal varlık** olarak adlandırılır. Örneğin, XYZ Corporation'ın ürünlerini üretmek için parçalar kullandığını varsayalım. Ayrıca, bu parçalardan bazılarının şirket içinde üretildiğini ve diğer parçaların satıcılardan satın alındığını varsayalım. Bu senaryoda, parçaların en azından bir kısmı bir satıcı tarafından tedarik edilmediğinden, bir PARÇA'nın "PARÇA, SATICI tarafından tedarik edilir" ilişkisinde bir SATICI'dan bağımsız olarak var olması oldukça mümkündür. Bu nedenle, PARÇA SATICI'dan bağımsız olarak var olabilir.

varoluşa bağlı
Varlığı bir veya daha fazla başka varlığa bağlı olan bir varlığın özelliği. Böyle bir ortamda, varlığa bağlı anahtar henüz var olmayan bir tabloya referans veremeyeceğinden, önce varlıktan bağımsız tablo oluşturulmalı ve yüklenmelidir.

varlıktan bağımsız Bir veya daha fazla ilgili varlıktan ayrı olarak var olabilen bir **varlığın** özelliği. Böyle bir tablo oluşturulmalıdır varlığa bağlı bir tabloya başvururken ilk önce.

güçlü varlık
Varoluştan bağımsız bir varlıktır, yani ilgili tüm varlıklardan ayrı olarak var olabilir.

düzenli varlık
Bkz. *güçlü varlık*.

Not

İlişki gücü kavramı orijinal ERM'nin bir parçası değildir. Bunun yerine, bu kavram doğrudan Crow's Foot diyagramları için geçerlidir. Crow's Foot diyagramları ilişkisel veritabanlarını tasarlamak için yaygın olarak kullanıldığından, veritabanı uygulamasını etkilediği için ilişki gücünü anlamak önemlidir. Chen ERD notasyonu kavramsal modellemeye yöneliktir ve bu nedenle zayıf ve güçlü ilişkiler arasında ayrım yapmaz.

4-1f İlişki Gücü

İlişki gücü kavramı, ilgili bir varlığın birincil anahtarının nasıl tanımlandığına dayanır. Bir ilişkiyi uygulamak için, bir varlığın birincil anahtarı (ana varlık, genellikle bire çok ilişkinin "bir" tarafında yer alır) ilgili varlıkta (çocuk varlık, çoğunlukla bire çok ilişkinin "çok" tarafında yer alır) yabancı anahtar olarak görünür. Bazen, yabancı anahtar aynı zamanda ilgili varlıkta birincil anahtar bileşenidir. Örneğin, Şekil 4.5'te CAR varlığı birincil anahtarı (CAR_VIN) CAR_COLOR varlığında hem birincil anahtar bileşeni hem de yabancı anahtar olarak görünür. Bu bölümde, çeşitli ilişki gücü kararlarının veritabanı tasarımında birincil anahtar düzenlemesini nasıl etkilediğini öğreneceksiniz.

Zayıf (Tanımlayıcı Olmayan) İlişkiler

Tanımlayıcı olmayan ilişki olarak da bilinen **zayıf ilişki**, ilgili varlığın birincil anahtarının ana varlığın birincil anahtar bileşenini içermemesi durumunda mevcuttur. Varsayılan olarak, ilişkiler üst varlığın birincil anahtarının ilgili varlıkta (alt varlık olarak da bilinir) yabancı anahtar (FK) olarak görünmesiyle kurulur. Örneğin, COURSE ve CLASS arasındaki 1:M ilişkisinin şu şekilde tanımlandığını varsayalım:

zayıf (tanımlayıcı olmayan) ilişki
İlgili birincil anahtarının ana kuruluşun birincil anahtar içermediği bir ilişki.

COURSE (**CRS_CODE**, DEPT_CODE, CRS_DESCRIPTION, CRS_CREDIT)

CLASS (**CLASS_CODE**, CRS_CODE, CLASS_SECTION, CLASS_TIME, ROOM_CODE, PROF_NUM)

Bu örnekte, CLASS birincil anahtarı bir birincil anahtar bileşenini COURSE varlığı. Bu durumda, CRS_CODE (ana birincil anahtarı) CLASS varlığında yalnızca bir yabancı anahtar olduğundan, COURSE ve CLASS arasında zayıf bir ilişki vardır. Şekil 4.9, Karga Ayağı gösteriminin, varlıklar arasına kesikli bir ilişki çizgisi yerleştirerek zayıf bir ilişkiyi nasıl tasvir ettiğini göstermektedir. ERD'nin altında gösterilen tablolar aşağıdakilerin nasıl yapıldığını göstermektedir

böyle bir ilişki uygulanmaktadır.

güçlü (tanımlayıcı) ilişki
İki varlık birbirine bağımlı olduğunda ortaya çıkan bir ilişki; veritabanı tasarımı açısından bakıldığındailgili varlığın birincil anahtarı ana birincil anahtarını içerdiğinde bu ilişki mevcuttur.

Güçlü (Tanımlayıcı) İlişkiler

İlgili varlığın birincil anahtarı ana varlığın birincil anahtar bileşenini içeriyorsa **güçlü (tanımlayıcı) bir ilişki** mevcuttur. Örneğin, COURSE ve CLASS arasındaki 1:M ilişkisinin şu şekilde tanımlandığını varsayalım:

COURSE (**CRS_CODE**, DEPT_CODE, CRS_DESCRIPTION, CRS_CREDIT)

CLASS (**CRS_CODE, CLASS_SECTION**, CLASS_TIME, ROOM_CODE, PROF_NUM)

Bu durumda, CLASS varlık birincil anahtarı CRS_CODE ve CLASS_SECTION öğelerinden oluşur. Bu nedenle, COURSE ve CLASS arasında güçlü bir ilişki vardır çünkü CRS_CODE (ana varlığın birincil anahtarı)

Şekil 4.9 Arasında Zayıf (Belirleyici Olmayan) Bir İlişki KURS ve SINIF

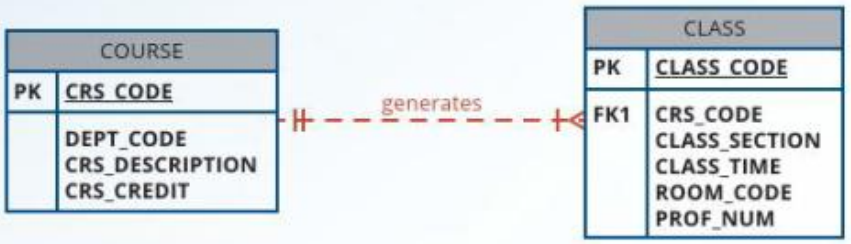


Table name: COURSE **Database name: Ch04_TinyCollege**

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
MATH-243	MATH	Mathematics for Managers	3
QM-261	CIS	Intro. to Statistics	3
QM-362	CIS	Statistical Applications	4

Table name: CLASS

CLASS_CODE	CRS_CODE	CLASS_SECTION	CLASS_TIME	ROOM_CODE	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
10025	MATH-243	1	Th 6:00-8:40 p.m.	DRE155	325

CLASS varlığı. Başka bir deyişle, CLASS birincil anahtarı COURSE varlığından bir birincil anahtar bileşeni devralmıştır. (CLASS'taki CRS_CODE'un *aynı zamanda* COURSE varlığının FK'sı olduğuna dikkat edin).

Karga Ayağı gösterimi, Şekil 4.10'da gösterildiği gibi, varlıklar arasındaki güçlü (tanımlayıcı) ilişkiyi düz bir çizgi ile tasvir eder.

Şekil 4.10'u incelerken, CLASS varlığının yanındaki O sembolünün ne anlama geldiğini merak edebilirsiniz. Bu kardinalitenin anlamını Bölüm 4-1h, İlişki Katılımı'nda keşfedeceksiniz.

Özet olarak, COURSE ve CLASS arasındaki ilişkinin güçlü veya zayıf olması CLASS varlığının birincil anahtarının nasıl tanımlandığına bağlıdır. İlişkinin niteliğinin genellikle, hangi ilişki türünün ve gücünün veritabanı işlemine, verimliliğine ve bilgi gereksinimlerine en uygun olduğunu belirlemek için profesyonel muhakeme kullanması gereken veritabanı tasarımcısı tarafından belirlendiğini unutmayın. Bu nokta ayrıntılı olarak vurgulanacaktır!

Şekil 4.10 KURS Arasında Güçlü (Belirleyici) Bir İlişki ve SINIF

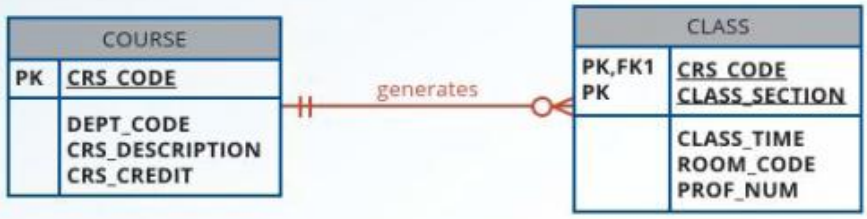


Table name: COURSE **Database name: Ch04_TinyCollege_Alt**

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
MATH-243	MATH	Mathematics for Managers	3
QM-261	CIS	Intro. to Statistics	3
QM-362	CIS	Statistical Applications	4

Table name: CLASS

CRS_CODE	CLASS_SECTION	CLASS_TIME	ROOM_CODE	PROF_NUM
ACCT-211	1	MWF 8:00-8:50 a.m.	BUS311	105
ACCT-211	2	MWF 9:00-9:50 a.m.	BUS200	105
ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
ACCT-212	1	MWF 10:00-10:50 a.m.	BUS311	301
ACCT-212	2	Th 6:00-8:40 p.m.	BUS252	301
CIS-220	1	MWF 9:00-9:50 a.m.	KLR209	228
CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
CIS-220	3	MWF 10:00-10:50 a.m.	KLR209	228
CIS-420	1	W 6:00-8:40 p.m.	KLR209	162
MATH-243	1	Th 6:00-8:40 p.m.	DRE155	325
QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114
QM-261	2	TTh 1:00-2:15 p.m.	KLR200	114
QM-362	1	MWF 11:00-11:50 a.m.	KLR200	162
QM-362	2	TTh 2:30-3:45 p.m.	KLR200	162

Not

Tabloların oluşturulma ve yüklenme sırasının çok önemli olduğunu unutmayın. Örneğin, "COURSE generates CLASS" ilişkisinde, COURSE tablosu CLASS tablosundan önce oluşturulmalıdır. Sonuçta, CLASS tablosunun yabancı anahtarının henüz var olmayan bir COURSE tablosuna başvurması kabul edilemez. Aslında, ilişkilerin zayıf veya güçlü olmasına bakılmaksızın, *referans bütünlüğü hataları olasılığını önlemek için 1:M ilişkisinde önce "1" tarafının verilerini yüklemelisiniz*.

4-1g Zayıf Varlıklar

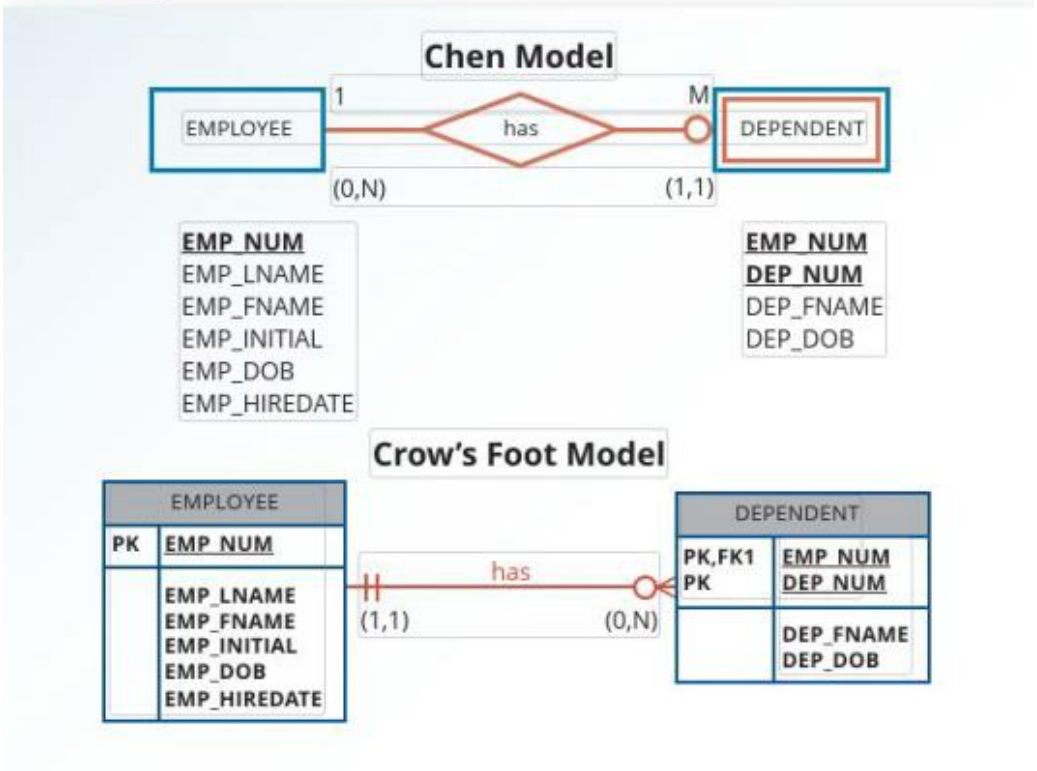
zayıf varlık
Varlık bağımlılığı gösteren ve ana varlığının birincil anahtarını devralan bir varlık. Örneğin, bir BAĞIMLI, bir ÇALIŞAN'ın varlığını gerektirir.

Bölüm 4-1f'de bahsedilen güçlü veya düzenli varlığın aksine, **zayıf** bir **varlık** iki koşulu karşılayan bir **varlıktır**:

1. Varlık varoluşa bağımlıdır; ilişki içinde olduğu varlık olmadan var olamaz.
2. Varlığın, ilişkideki ana varlıktan kısmen veya tamamen türetilmiş bir birincil anahtarı vardır.

Örneğin, bir şirket sigorta poliçesi bir çalışanı ve bakmakla yükümlü olduğu kişileri sigortalar. Bir sigorta poliçesini tanımlamak amacıyla, bir ÇALIŞAN'ın bir BAĞIMLISI olabilir veya olmayabilir, ancak BAĞIMLI bir ÇALIŞAN ile ilişkilendirilmelidir. Dahası, BAĞIMLI, ÇALIŞAN olmadan var olamaz; , bir kişi bir çalışanın bakmakla yükümlü olduğu kişi olmadığı sürece bağımlı olarak sigorta kapsamına alınamaz. BAĞIMLI, "ÇALIŞAN'ın BAĞIMLISI vardır" ilişkisindeki zayıf varlıktır. Bu ilişki Şekil 4.11'de gösterilmektedir.

Şekil 4.11 ERD'de Zayıf Bir Varlık



Şekil 4.11'deki Chen gösteriminin çift duvarlı bir varlık dikdörtgeni kullanarak zayıf varlığı tanımladığına dikkat edin. Visio Professional tarafından oluşturulan Karga Ayağı gösterimi, ilgili varlığın zayıf olup olmadığını belirtmek için ilişki çizgisini ve PK/FK tanımlamasını kullanır. Güçlü (tanımlayıcı) bir ilişki, ilgili varlığın zayıf olduğunu gösterir. Böyle bir ilişki, zayıf varlık tanımı için her iki koşulun da karşılandığı anlamına gelir - ilgili varlık varlığa bağımlıdır ve ilgili varlığın PK'sı ana varlığın bir PK bileşenini içerir.

Zayıf varlığın birincil anahtarının bir kısmını güçlü muadilinden miras aldığını unutmayın. Örneğin, Şekil 4.11'de gösterilen DEPENDENT varlığının anahtarının en azından bir kısmı EMPLOYEE varlığından miras alınmıştır:

EMPLOYEE (**EMP_NUM**, EMP_LNAME, EMP_FNAME, EMP_INITIAL, EMP_DOB, EMP_HIREDATE)

BAĞIMLI (**EMP_NUM, DEP_NUM**, DEP_FNAME, DEP_DOB)

Şekil 4.12'de zayıf varlık (DEPENDENT) ile onun ebeveyni veya güçlü muadili (EMPLOYEE) arasındaki ilişkinin uygulanması gösterilmektedir. DEPENDENT'ın birincil anahtarının EMP_NUM ve DEP_NUM olmak üzere iki öznitelikten oluştuğunu ve EMP_NUM'ın EMPLOYEE'den miras alındığını unutmayın.

Şekil 4.12 Güçlü Bir İlişkide Zayıf Bir Varlık

Table name: EMPLOYEE **Database name: Ch04_ShortCo**

EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_INITIAL	EMP_DOB	EMP_HIREDATE
1001	Callifante	Jeanine	J	12-Mar-64	25-May-97
1002	Smithson	William	K	23-Nov-70	28-May-97
1003	Washington	Herman	H	15-Aug-68	28-May-97
1004	Chen	Lydia	B	23-Mar-74	15-Oct-98
1005	Johnson	Melanie		28-Sep-66	20-Dec-98
1006	Ortega	Jorge	G	12-Jul-79	05-Jan-02
1007	O'Donnell	Peter	D	10-Jun-71	23-Jun-02
1008	Brzenski	Barbara	A	12-Feb-70	01-Nov-03

Table name: DEPENDENT

EMP_NUM	DEP_NUM	DEP_FNAME	DEP_DOB
1001	1	Annelise	05-Dec-97
1001	2	Jorge	30-Sep-02
1003	1	Suzanne	25-Jan-04
1006	1	Carlos	25-May-01
1008	1	Michael	19-Feb-95
1008	2	George	27-Jun-98
1008	3	Katherine	18-Aug-03

Bu senaryo göz önüne alındığında ve bu ilişkinin yardımıyla şunu belirleyebilirsiniz:

Jeanine J. Callifante, Annelise ve Jorge adında iki bakmakla yükümlü olduğu kişi

olduğunu iddia etmektedir.

Veritabanı tasarımcısının genellikle bir varlığın zayıf olarak tanımlanıp tanımlanamayacağını iş kurallarına göre belirlediğini unutmayın. Şekil 4.9'u CLASS'ın COURSE için zayıf bir varlık olduğu sonucuna varabilirsiniz. Sonuçta, bir SINIF'ın bir KURS olmadan var olamayacağı açıktır, bu nedenle varoluş bağımlılığı vardır. Örneğin, bir öğrenci ACCT-211 dersi olmadığı sürece ACCT-211, Bölüm 3 (CLASS_CODE 10014) Muhasebe I dersine kaydolamaz. Ancak, CLASS tablosunun birincil anahtarının CLASS_CODE olduğunu ve bunun COURSE üst varlığından türetilmediğini unutmayın. , CLASS şu şekilde temsil edilebilir:

CLASS (**CLASS_CODE**, CRS_CODE, CLASS_SECTION, CLASS_TIME, ROOM_CODE, PROF_NUM)

İkinci zayıf varlık şartı karşılanmamıştır; bu nedenle, tanım gereği, Şekil 4.9'daki CLASS varlığı zayıf olarak sınıflandırılamaz. Öte yandan, CLASS varlığının birincil anahtarı CRS_CODE ve CLASS_SECTION kombinasyonundan oluşan bir bileşik anahtar olarak tanımlanmış olsaydı, CLASS şu şekilde temsil edilebilirdi:

CLASS (**CRS_CODE, CLASS_SECTION**, CLASS_TIME, ROOM_CODE, PROF_NUM)

Bu durumda, Şekil 4.10'da gösterildiği gibi, CRS_CODE, COURSE tablosunun birincil anahtarı olduğu için CLASS birincil anahtarı kısmen COURSE'den türetilir. Bu karar göz önüne alındığında,