Appendix D

Converting an ER Model into a Database Structure

Preview

Converting any ER model to a set of tables in a database requires following specific rules that govern the conversion. The application of those rules requires an understanding of the effects of updates and deletions on the tables in the database. Before discussing these rules in detail, let's briefly review a simple ER model and the SQL commands used to generate the tables.

Data Files and Available Formats

	MS Access	Oracle	MS SQL	My SQL	MS Access	Oracle	MS SQL	My SQL
Artist	✓							

Data Files Available on cengagebrain.com

D-1 The Artist Database

To illustrate the conversion of an ER model into a database structure, let's use the Artist database, located in the Premium Website for this book. The Artist database conforms to the following conditions:

- A painter might paint many paintings. To be considered a painter in the Artist database, the painter must have painted at least one painting. This business rule decrees that the cardinality is (1,N) in the relationship between PAINTER and PAINTING.
- Each painting is painted by one (and only one) painter.
- A painting might (or might not) be exhibited in a gallery; that is, GALLERY is an optional entity to the PAINTING entity.

Given that description, let's use a simple ER model and some matching tables for the Artist database, shown in Figure D.1.

FIGURE D.1 THE ARTIST DATABASE ERD AND TABLES

The Artist Database ERD

Database name: Artist

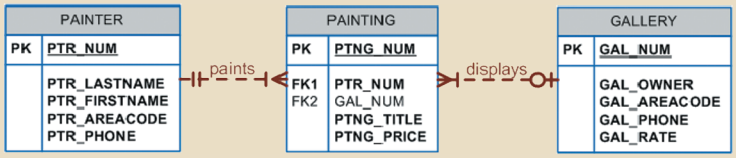

Table name: PAINTER

PTR_NUM	PTR_LASTNAME	PTR_FIRSTNAME	PTR_AREACODE	PTR_PHONE
123	Ross	Georgette	901	885-4567
126	Itero	Julio	901	346-1112
127	Geoff	George	615	221-4456

Table name: PAINTING

PTNG_NUM	PTNG_TITLE	PTNG_PRICE	PTR_NUM	GAL_NUM
1338	Dawn Thunder	245.5	123	5
1339	A Faded Rose	6723	123	
1340	The Founders	567.99	126	6
1341	Hasty Pudding Exit	145.5	123	
1342	Plastic Paradise	8328.99	126	6
1343	Roamin'	785	127	6
1344	Wild Waters	999	127	5
1345	Stuff 'n Such 'n Some	9800	123	5

Table name: GALLERY

GAL_NUM	GAL_OWNER	GAL_AREACODE	GAL_PHONE	GAL_RATE
5	L. R. Gilliam	901	123-4456	0.37
6	G. G. Waters	405	353-2243	0.45

The data dictionary in Table D.1 shows the characteristics of the attributes found in the three tables.

Given the information in Figure D.1 and Table D.1, note that

- PTR_NUM in the PAINTING table is the foreign key that references the PAINTER table. Because the relationship between PAINTER and PAINTING is mandatory, the PTR_NUM foreign key must be classified as NOT NULL.
- GAL_NUM in PAINTING is the foreign key that references the GALLERY table. Because the relationship between PAINTING and GALLERY is optional, the GAL_NUM foreign key may be null.

TABLE D.1

A DATA DICTIONARY FOR THE ARTIST DATABASE

TABLE NAME	ATTRIBUTE NAME	CONTENTS	TYPE	FORMAT	RANGE	REQUIRED	PK OR FK	FK REFERENCED TABLE
PAINTER	PTR_NUM	Painter number	CHAR(4)	9999	1000–9999	Y	PK	
	PTR_LASTNAME	Painter last name	VARCHAR(15)	Xxxxxxxxxxx		Y		
	PTR_FIRSTNAME	Painter first name	VARCHAR(15)	Xxxxxxxxxxx		Y		
	PTR_AREACODE	Painter area code	CHAR(3)	999				
	PTR_PHONE	Painter phone	CHAR(8)	999-9999				
GALLERY	GAL_NUM	Gallery number	CHAR(4)	9999	1000–9999	Y	PK	
	GAL_OWNER	Gallery owner	VARCHAR(35)	Xxxxxxxxxxx				
	GAL_AREACODE	Gallery area code	CHAR(3)	999		Y		
	GAL_PHONE	Gallery phone	CHAR(8)	999-9999		Y		
	GAL_RATE	Gallery commission rate (pct.)	NUMBER(4,2)	99.99	0.00–60.00	Y		
PAINTING	PTNG_NUM	Painting number	CHAR(4)	9999	1000–9999	Y	PK	
	PTNG_TITLE	Painting title	VARCHAR(35)	Xxxxxxxxxxx				
	PTNG_PRICE	Painting price	NUMBER(9,2)	99,999.99	10.00–99,999.99	Y		
	PTR_NUM	Painter number	CHAR(4)	9999	1000–9999	Y	FK	PAINTER
	GAL_NUM	Gallery number	CHAR(4)	9999	1000–9999		FK	GALLERY

FK = Foreign key
PK = Primary key
CHAR = Fixed character length data, 1 to 255 characters
VARCHAR = Variable character length data, 1 to 2,000 characters. May also be labeled VARCHAR2.
NUMBER = Numeric data. NUMBER(9,2) is used to specify numbers with two decimal places and up to nine digits long, including the decimal places. Some RDBMSs permit the use of a MONEY or a CURRENCY data type.

D-1a The Effect of Foreign Key Constraints on Data Manipulation

Given the Artist database table structures, let's examine the effect of the following data manipulation events of the foreign key constraint actions:

1. *Adding a painter (row) to the PAINTER table.* Adding a painter does not cause any problems because the PAINTER table does not have any dependencies in other tables.
2. *Updating the PAINTER table's primary key.* Changing a PAINTER key causes problems in the database because some paintings in the PAINTING table may make reference to this key. The option is to use the UPDATE CASCADE. This option makes sure that a change in the PAINTER's PTR_NUM automatically triggers the required changes in the PTR_NUM foreign key found in other tables. This is the recommended option. This behavior (UPDATE CASCADE) is not supported by some RDBMS products, such as Oracle.
3. *Deleting a painter (row) from the PAINTER table.* If you delete a row (painter) from the PAINTER table, the PAINTING table may contain references to a painter who no longer exists, thereby creating a deletion anomaly. (A painting does not cease to exist just because the painter does.) Given this situation, it is wise to restrict the ability to delete a row from a table when there is a foreign key in another table that references the row. The restriction means that you can delete a painter from the PAINTER table only when there is no foreign key in another table related to this painter row. This behavior is enforced automatically by the RDBMS when using the FOREIGN KEY clause.
4. *Adding a gallery (row) to the GALLERY table.* Adding a new row does not affect the database because the GALLERY does not have dependencies in other tables.
5. *Updating the GALLERY table's primary key.* Changing a primary key value in a GALLERY row requires that all foreign keys making reference to it be updated as well. Therefore, you must use an UPDATE CASCADE clause. This option makes sure that a change in the GALLERY's GAL_NUM automatically triggers the required changes in the GAL_NUM foreign key found in other tables. This is the recommended option. This behavior (UPDATE CASCADE) is not supported by some RDBMS products, such as Oracle.
6. *Deleting a gallery (row) from the GALLERY table.* Deleting a GALLERY row creates problems in the database when rows in the PAINTING table make reference to the GALLERY row's primary key. Because GALLERY is optional to PAINTING, you may set all of the deleted gallery GAL_NUM values to null (DELETE SET NULL). Or you may want the database user to be alerted to the problem by specifying that the deletion of a GALLERY row is permitted only when there is no foreign key (GAL_NUM) in another table that requires the GALLERY row's existence. This behavior is enforced automatically by the RDBMS when using the FOREIGN KEY clause.

D-1b Transforming the ER Model into a Set of Tables

Armed with the results discussed in Section D-1a, you can now transform the ER model into a set of tables by using the following SQL commands:

1. Create the PAINTER table:

```
CREATE TABLE PAINTER (
PTR_NUM       CHAR(4)     NOT NULL     UNIQUE,
PTR_LASTNAME              CHAR(15)     NOT NULL,
PTR_FIRSTNAME             CHAR(15)     NOT NULL,
PTR_AREACODE              CHAR(3),
PTR_PHONE                 CHAR(8),
PRIMARY KEY (PTR_NUM));
```

2. Create the GALLERY table:

```
CREATE TABLE GALLERY (
GAL_NUM          CHAR(4)          NOT NULL UNIQUE,
GAL_OWNER        CHAR(35),
GAL_AREACODE     CHAR(3)          NOT NULL,
GAL_PHONE        CHAR(8)          NOT NULL,
GAL_RATE         NUMBER(4,2),
PRIMARY KEY (GAL_NUM));
```

3. Create the PAINTING table:

```
CREATE TABLE PAINTING (
PTNG_NUM         CHAR(4)          NOT NULL UNIQUE,
PTNG_TITLE       CHAR(35),
PTNG_PRICE       NUMBER(9,2),
PTR_NUM          CHAR(4)          NOT NULL,
GAL_NUM          CHAR(4),
PRIMARY KEY(PNTG_NUM),
FOREIGN KEY (PTR_NUM) REFERENCES PAINTER
ON UPDATE CASCADE,
FOREIGN KEY (GAL_NUM) REFERENCES GALLERY
ON UPDATE CASCADE);
```

After creating the database tables and entering their contents, you are now ready for data entry, queries, and reports. Note that the decisions made by the designer to govern data integrity are reflected in the foreign key rules. Implementation decisions vary according to the problem being addressed.

D-2 General Rules Governing Relationships Among Tables

Given the experience with the simple Artist database in the previous section, here is a general set of rules to help you create any database table structure that will meet the required integrity constraints.

1. All primary keys must be defined as NOT NULL and UNIQUE. If your applications software does not support the NOT NULL option, you should enforce the condition by using programming techniques. This is another argument for using DBMS software that meets ANSI SQL standards.
2. Define all foreign keys to conform to the following requirements for binary relationships.

1:M Relationships Create the foreign key by putting the primary key of the "one" in the table of the "many." The "one" side is referred to as the parent table, and the "many" side is referred to as the dependent table. Observe the following foreign key rules:

If both sides are MANDATORY:

Column constraint:	NOT NULL
FK constraint:	Default behavior (on delete restrict) ON UPDATE CASCADE

If both sides are OPTIONAL:

Column constraint:	NULL ALLOWED
FK constraint:	ON DELETE SET NULL ON UPDATE CASCADE

If one side is OPTIONAL and one side is MANDATORY:

a. If the "many" and the mandatory components of the relationship are on the same side in the ER diagram, a NULL ALLOWED condition must be defined for the dependent table's foreign key. The foreign key rules should be:

Column constraint:	NULL ALLOWED
FK constraint:	ON DELETE SET NULL or default behavior: ON DELETE RESTRICT ON UPDATE CASCADE

b. If the "many" and the mandatory components of the relationship are not on the same side in the ER diagram, a NOT NULL condition must be defined for the dependent table's foreign key. Deletion and update in the parent table of the foreign key should be subject to default behavior (on delete restrict) and UPDATE CASCADE restrictions.

Weak Entities

a. Put the key of the parent table (the strong entity) in the weak entity. The key of the weak entity will be a composite key composed of the parent table key and the weak entity candidate key, if any. (The designer may decide to create a new unique ID for the entity.)

b. The weak entity relationship conforms to the same rules as the 1:M relationship, except for the following foreign key restrictions:

Column constraint:	NOT NULL
FK constraint:	ON DELETE CASCADE ON UPDATE CASCADE

M:N Relationships Convert the M:N relationship to a composite (bridge) entity consisting of (at least) the parent tables' primary keys. Thus, the composite entity primary key is a composite key that is subject to the NOT NULL restriction.

1:1 Relationships If both entities are in a mandatory participation in the relationship and they do not participate in other relationships, it is most likely that the two entities should be part of the same entity.

Table D.2 summarizes the ramifications of the foreign key actions that could be used to represent multiple cases of 1:1, 1:M, and M:N relationships.

TABLE D.2

SUMMARY OF FOREIGN KEY RULES

				FOREIGN KEY ACTIONS	
RELATIONSHIP	**FOREIGN KEY LOCATION**	**THE ENTITIES PARTICIPATING IN THE RELATIONSHIPS ARE...**	**FOREIGN KEY ATTRIBUTE CONSTRAINT**	**DELETE**	**UPDATE**
M:N	New entity composite primary key	Both mandatory Both optional One mandatory, one optional If FK located on mandatory side If FK located on optional side	NN NN NN NN	R C C R	C C C C
1:M	Many side	Both mandatory Both optional One mandatory, one optional If FK located on mandatory side If FK located on optional side	NN NA NA NN	R SN or R SN or R R	C C C C
1:1	Foreign key placement is a matter of informed choice.* Put the FK in the ERDs optional side, the strong entity, the most frequently accessed side, or the side dictated by the case semantics. *Do not put the FK in both sides.*	Both mandatory Both optional One mandatory, one optional If FK located on mandatory side If FK located on optional side	NN NA NA NN	R SN SN R	C C C C
Weak	Weak entity		NN**	C	C
Multivalued Attributes	Create a set of new tables in 1:M relationships. Conform to the weak entity rules.		NN	C	C

NN = Not Null NA = Null Allowed R = Restrict ** = Inherited from parent entity
SN = Set to Null C = Cascade * = See Chapter 5, Advanced Data Modeling, for a detailed discussion of the 1:1 relationship.