Multiplicity (Cardinality)

There are three types of relationships between data within separate tables: **one-to-one**, **one-to-many**, and **many-to-many**. To be able to identify these relationships, you need to examine the data and have an understanding of what business rules apply to the data and tables. One table is called the primary or parent table and the other is the related or child table.

One-to-One Relationship

A *one-to-one* (1:1) relationship means that each record in Table A relates to one, and only one, record in Table B, and each record in Table B relates to one, and only one, record in Table A. Look at the following example of tables from a company's Employees database:

PERSONAL

EmployeeID	FirstName	LastName	Address	City	State	Zip
EN1-10	Carol	Schaaf	2306 Palisade Ave.	Union City	NJ	07087
EN1-12	Gayle	Murray	1855 Broadway	New York	NY	12390
EN1-15	Steve	Baranco	742 Forrest St.	Kearny	NJ	07032
EN1-16	Kristine	Racich	416 Bloomfield St.	Hoboken	NJ	07030
EN1-19	Barbara	Zumbo	24 Central Ave.	Ritchfield Park	NJ	07660
EN1-20	Daniel	Gordon	2 Angelique St.	Weehawken	NJ	07087
EN1-22	Jacqueline	Rivet	3600 Bergeline Ave.	Union City	NJ	07087
EN1-23	Betsy	Rosyln	1800 Boulevard East	Weehawken	NJ	07086
EN1-25	Will	Strick	2100 91st St.	North Bergen	NJ	07047
EN1-26	Susan	Shipe	240 Fifth Ave.	New York	NY	10018

PAYROLL

EmployeeID	PayRate
EN1-10	$25.00
EN1-12	$27.50
EN1-15	$20.00
EN1-16	$19.00
EN1-19	$22.75
EN1-20	$23.00
EN1-22	$22.50
EN1-23	$19.50
EN1-25	$12.50
EN1-26	$14.00

Each record in the Personal table is about one employee. That record relates to one, and only one, record in the Payroll table. Each record in the Payroll table relates to one, and only one, record in the Personal table. **In a one-to-one relationship, either table can be considered to be the primary or parent table.**

One-To-Many Relationship

A *one-to-many* (1:N) relationship means a record in Table A can relate to zero, one, or many records in Table B. Many records in Table B can relate to one record in Table A. The potential relationship is what's important; for a single record in Table A, there might be no related records in Table B, or there might be only one related record, but there could be many. Look at the following tables about a company's Customers and Orders.

CUSTOMERS

CustomerID	CustomerName	Address	City	State	Zip
20151	Engel's Books	19 International Dr	Ryebrook	NY	10273-9764
20493	Jamison Books	396 Apache Ave	Fountain Valley	CA	92708-4982
20512	Gardening Galore	79 Gessner Pk	Houston	TX	77024-6261
20688	Books Abound	51 Ulster St	Denver	CO	80237-3386
20784	Book World	687 Mountain Rd	Stowe	VT	08276-3196
20926	The Corner Booksotre	36 N.Miller Ave	Syracuse	NY	13206-4976
20932	Allendale Books	512 Columbia Rd	Someville	NJ	08876-2987
21570	In Between the Covers	2008 Delta Ave	Cincinnati	OH	45208-4468
21587	Books and Beyond	51 Windsor St	Cambridge	MA	02139-2123
21965	Cover to Cover	12 Harbor St	Burlington	VT	04982-2977

ORDERS

OrderNum	CustomerID	OrderDate	ShipDate	Shipper
76654	20151	2/1/00	2/6/00	USPS
74432	20151	6/30/99	7/2/99	Federal Express
75987	20151	11/10/99	11/12/99	UPS
62922	20493	9/5/99	9/6/99	UPS
65745	20493	10/1/99	10/3/99	USPS
72212	20493	4/22/00	4/25/00	UPS
73547	20493	8/17/99	8/20/99	UPS
69211	21570	5/12/99	5/12/99	Federal Express
70343	21587	10/2/00	10/4/00	UPS
72833	21587	12/14/99	12/17/99	UPS

The Customers table holds a unique record for each customer. Each customer can place many orders. Many records in the Orders table can relate to only one record in the Customers table. This is a one-

to-many relationship (1:N) between the Customers table and the Orders table. **In a one-to-many relationship, the table on the one side of the relationship is the primary table and the table on the many side is the related table.**

Many-To-Many Relationship

Examine the sample data below. These tables hold data about employees and the projects to which they are assigned. Each project can involve more than one employee and each employee can be working on more than one project. This constitutes a **many-to-many** (N:N) relationship.

EMPLOYEES

EmployeeID	Last Name	First Name	ProjectNum
EN1-26	O'Brien	Sean	30-452-T3
EN1-26	O'Brien	Sean	30-457-T3
EN1-26	O'Brien	Sean	31-124-T3
EN1-33	Guya	Amy	30-452-T3
EN1-33	Guya	Amy	30-482-TC
EN1-33	Guya	Amy	31-124-T3
EN1-35	Baranco	Steven	30-452-T3
EN1-35	Baranco	Steven	31-238-TC
EN1-36	Roslyn	Elizabeth	35-152-TC
EN1-38	Schaaf	Carol	36-272-TC
EN1-40	Wing	Alexandra	31-238-TC
EN1-40	Wing	Alexandra	31-241-TC

PROJECTS

ProjectNum	ProjectTitle	EmployeeID
30-452-T3	Woodworking Around The House	EN1-26
30-452-T3	Woodworking Around The House	EN1-33
30-452-T3	Woodworking Around The House	EN1-35
30-457-T3	Basic Home Electronics	EN1-26
30-482-TC	The Complete American Auto Repair Guide	EN1-33
31-124-T3	The Sport Of Hang Gliding	EN1-26
31-124-T3	The Sport Of Hang Gliding	EN1-33
31-238-TC	The Complete Baseball Reference	EN1-35
31-238-TC	The Complete Baseball Reference	EN1-35
31-241-TC	Improving Your Tennis Game	EN1-40
35-152-TC	Managing Your Personal Finances	EN1-36
36-272-TC	Using Electronic Mail Effectively	EN1-38

Example: Mapping Many-to-Many

Friend-like relationship involves two users, but we can't use the user entity alone to map this relationship, so we need a `JUNCTION TABLE`.

1. Run the following code in your script:

```
CREATE TABLE friends (
       friend1 INTEGER REFERENCES users(id),
       friend2 INTEGER REFERENCES users(id),
       PRIMARY KEY (friend1, friend2) -- implicitly NOT NULL and UNIQUE
);

INSERT INTO friends (friend1, friend2) VALUES
(2, 7), (4, 5), (1, 5), (2, 1), (2, 5), (7, 4);
```

- `IN` keyword to in a `WHERE` clause to match any value in a set of values.
- A subquery can be supplied to provide a result set for a `SET` in a `WHERE` clause.
- The subquery acting as set should include only one column per row.
- `UNION` will combine multiple result sets into a single result.
- `UNION ALL` will include duplicate.

```
SELECT first_name, last_name FROM users WHERE id IN
(SELECT friend1 FROM friends WHERE friend2 = 7
UNION
SELECT friend2 FROM friends WHERE friend1 = 7);

SELECT * FROM users;
```

References

- Database Relationships