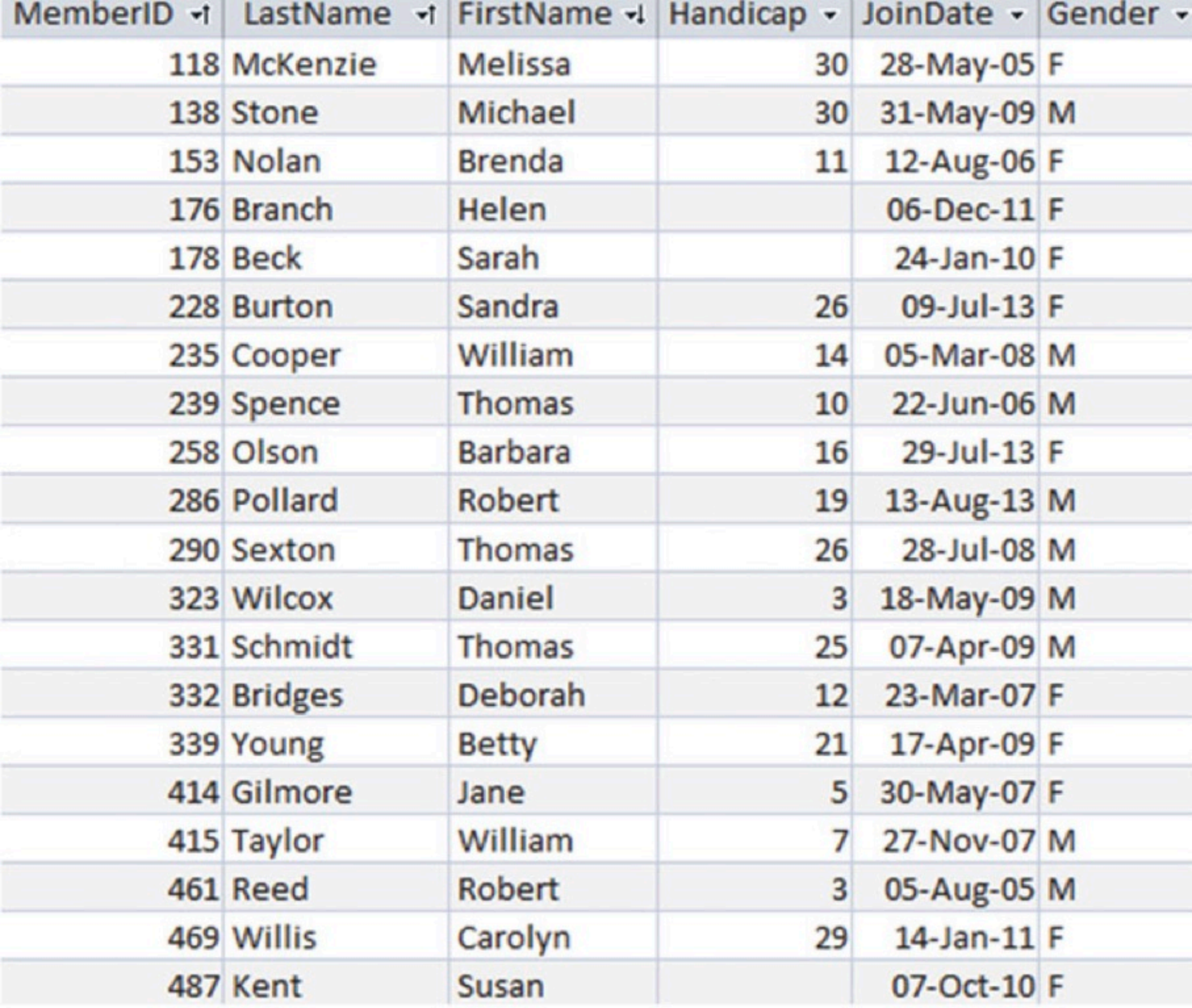
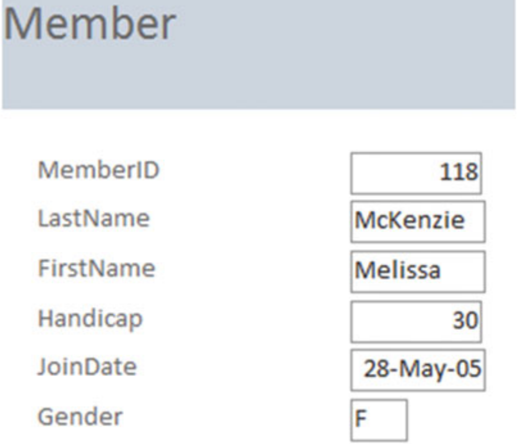


MemberID	LastName	FirstName	Handicap	JoinDate	Gender
118	McKenzie	Melissa	30	28-May-05	F
138	Stone	Michael	30	31-May-09	M
153	Nolan	Brenda	11	12-Aug-06	F
176	Branch	Helen		06-Dec-11	F
178	Beck	Sarah		24-Jan-10	F
228	Burton	Sandra	26	09-Jul-13	F
235	Cooper	William	14	05-Mar-08	M
239	Spence	Thomas	10	22-Jun-06	M
258	Olson	Barbara	16	29-Jul-13	F
286	Pollard	Robert	19	13-Aug-13	M
290	Sexton	Thomas	26	28-Jul-08	M
323	Wilcox	Daniel	3	18-May-09	M
331	Schmidt	Thomas	25	07-Apr-09	M
332	Bridges	Deborah	12	23-Mar-07	F
339	Young	Betty	21	17-Apr-09	F
414	Gilmore	Jane	5	30-May-07	F
415	Taylor	William	7	27-Nov-07	M
461	Reed	Robert	3	05-Aug-05	M
469	Willis	Carolyn	29	14-Jan-11	F
487	Kent	Susan		07-Oct-10	F

Figure 1-1. *The Member table*

The code that follows shows the SQL code for creating the Member table shown in Figure 1-1. Each attribute has a name and type specified. In SQL, the keyword INT means an integer or non-fractional number, and CHAR(n) means a string of characters n long. The code also specifies that MemberID will be the primary key. Every table in a well-designed database should have a primary key clause.

```
CREATE TABLE Member (
MemberID INT PRIMARY KEY,
LastName CHAR(20),
FirstName CHAR(20),
Handicap INT,
JoinDate DATETIME,
Gender CHAR(1));
```



Member

MemberID	118
LastName	McKenzie
FirstName	Melissa
Handicap	30
JoinDate	28-May-05
Gender	F

Figure 1-2. *A form allowing entry and updating of data in the Member table*

```
INSERT INTO Member
VALUES (118, 'McKenzie', 'Melissa', '963270', 30, '05/10/1999', 'F')
```

If many of the data items are empty, we can specify which attributes will have values. If we had only the ID and last name of a member, we could insert just those two values as shown here:

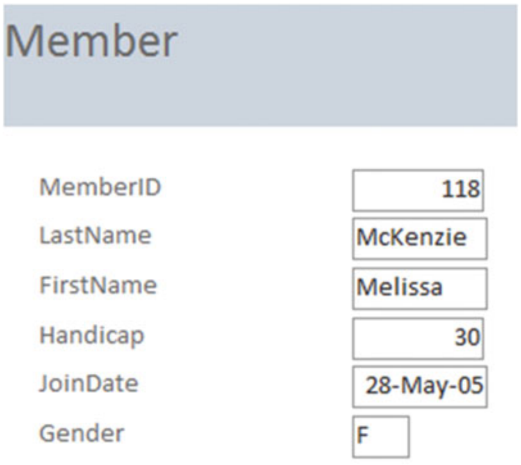
```
INSERT INTO Member (MemberID, LastName)
VALUES (258, 'Olson')
```

When adding a new row as just seen, we always have to provide a value for the primary key.

We can also alter records that are already in the database with an update query. The following query will find the row for the member with ID 118 and then will update the phone number:

```
UPDATE Member
SET Phone = '875077'
WHERE MemberID = 118
```

This query specifies which rows are to be changed (the `WHERE` clause) and also specifies the field to be updated (the `SET` clause).



Member

MemberID	118
LastName	McKenzie
FirstName	Melissa
Handicap	30
JoinDate	28-May-05
Gender	F

Figure 1-2. *A form allowing entry and updating of data in the Member table*

It is possible to construct web forms or use mechanical readers, such as bar-code readers, that can collect data and insert it into a database. Data can also be added with bulk updates from files or be imported from other applications. Behind all the different mechanisms for updating data, SQL update queries are generated. We will see three types of queries for inserting or changing data just to get an idea of what they look like.

The code that follows shows the SQL to enter one complete row in our `Member` table. The data items are in the same order as specified when the table was created. Note that the date and string values need to be enclosed in single quotes.

```
INSERT INTO Member
VALUES (118, 'McKenzie', 'Melissa', '963270', 30, '05/10/1999', 'F')
```

If many of the data items are empty, we can specify which attributes will have values. If we had only the ID and last name of a member, we could insert just those two values as shown here:

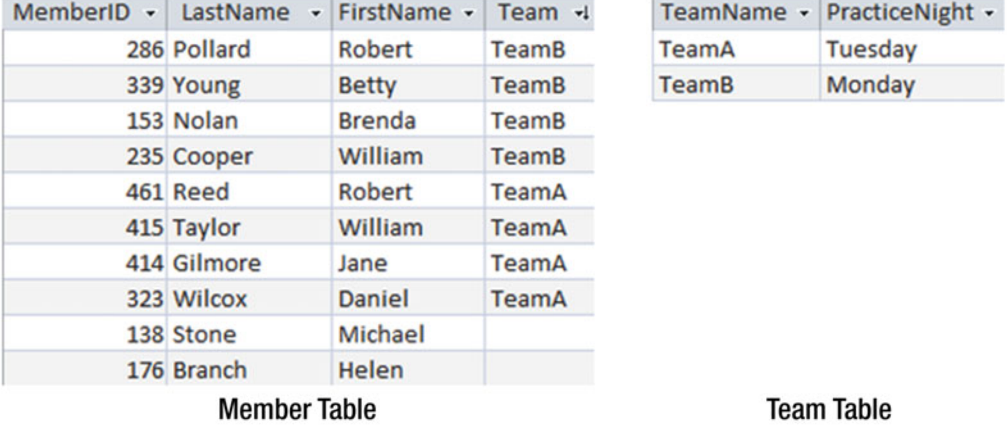
```
INSERT INTO Member (MemberID, LastName)
VALUES (258, 'Olson')
```

When adding a new row as just seen, we always have to provide a value for the primary key.

We can also alter records that are already in the database with an update query. The following query will find the row for the member with ID 118 and then will update the phone number:

```
UPDATE Member
SET Phone = '875077'
WHERE MemberID = 118
```

This query specifies which rows are to be changed (the `WHERE` clause) and also specifies the field to be updated (the `SET` clause).



MemberID	LastName	FirstName	Team
286	Pollard	Robert	TeamB
339	Young	Betty	TeamB
153	Nolan	Brenda	TeamB
235	Cooper	William	TeamB
461	Reed	Robert	TeamA
415	Taylor	William	TeamA
414	Gilmore	Jane	TeamA
323	Wilcox	Daniel	TeamA
138	Stone	Michael	
176	Branch	Helen	

Member Table

TeamName	PracticeNight
TeamA	Tuesday
TeamB	Monday

Team Table

Figure 1-4. *Member and Team tables*

This separation of information into two tables prevents the inconsistent data we had previously. The practice night for each team is stored only once. If we need to find out what night Brenda Nolan should be at practice, we now need to consult two tables: the `Member` table to find her team and then the `Team` table to find the practice night for that team. The bulk of this book is about how to do just that sort of data retrieval.

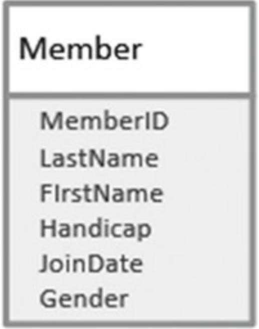


Figure 1-5. *UML representation of a Member class*

In a relational database, each class is represented as a table, the attributes are the columns, and each instance (in this case an individual club member) will be a row in the table.

The data model can also depict the way the different classes depend on each other. Figure 1-6 shows two classes, `Member` and `Team`, and how they are related.

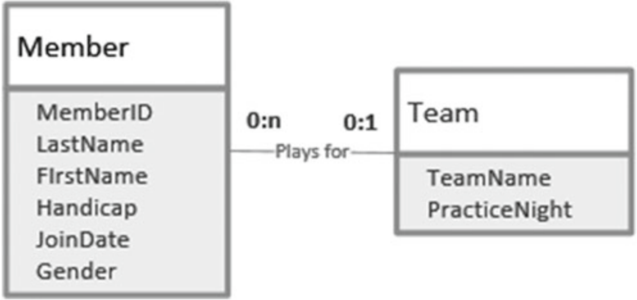
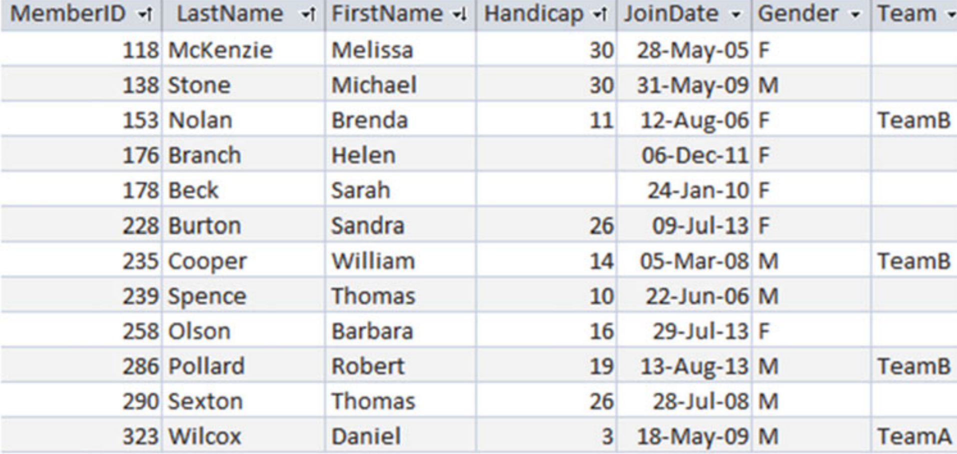


Figure 1-6. *A relationship between two classes*



MemberID	LastName	FirstName	Handicap	JoinDate	Gender	Team
118	McKenzie	Melissa	30	28-May-05	F	
138	Stone	Michael	30	31-May-09	M	
153	Nolan	Brenda	11	12-Aug-06	F	TeamB
176	Branch	Helen		06-Dec-11	F	
178	Beck	Sarah		24-Jan-10	F	
228	Burton	Sandra	26	09-Jul-13	F	
235	Cooper	William	14	05-Mar-08	M	TeamB
239	Spence	Thomas	10	22-Jun-06	M	
258	Olson	Barbara	16	29-Jul-13	F	
286	Pollard	Robert	19	13-Aug-13	M	TeamB
290	Sexton	Thomas	26	28-Jul-08	M	
323	Wilcox	Daniel	3	18-May-09	M	TeamA

Figure 1-7. *Member table with a foreign key column Team*

```
CREATE TABLE Member(
MemberID INT PRIMARY KEY,
LastName CHAR(20),
FirstName CHAR(20),
Phone CHAR(20),
Handicap INT,
JoinDate DATETIME,
Gender CHAR(1),
Team CHAR(20) FOREIGN KEY REFERENCES Team);
```

Because we need to compare the value in the foreign key column of the `Member` table with the primary key column of the `Team` table, these two columns must have the same domain or datatype.

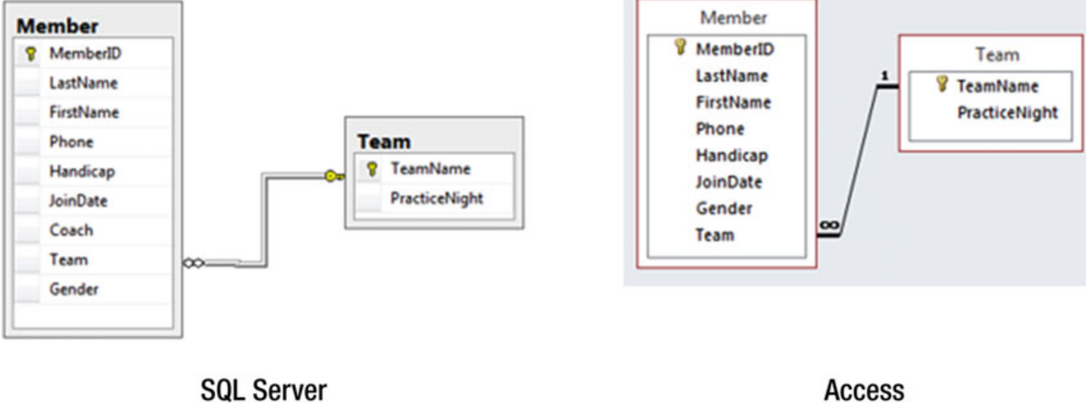
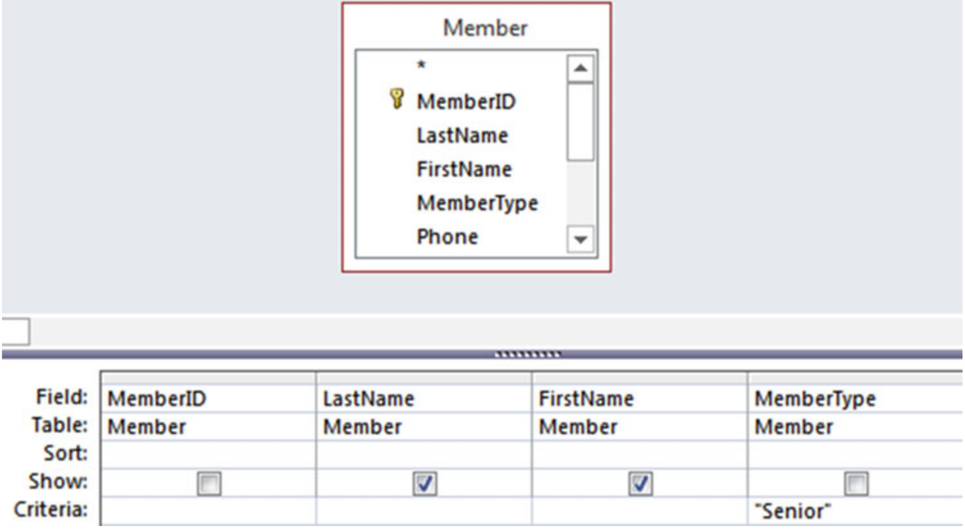


Figure 1-8. *Diagrams for implementing 1–Many relationships using foreign keys*

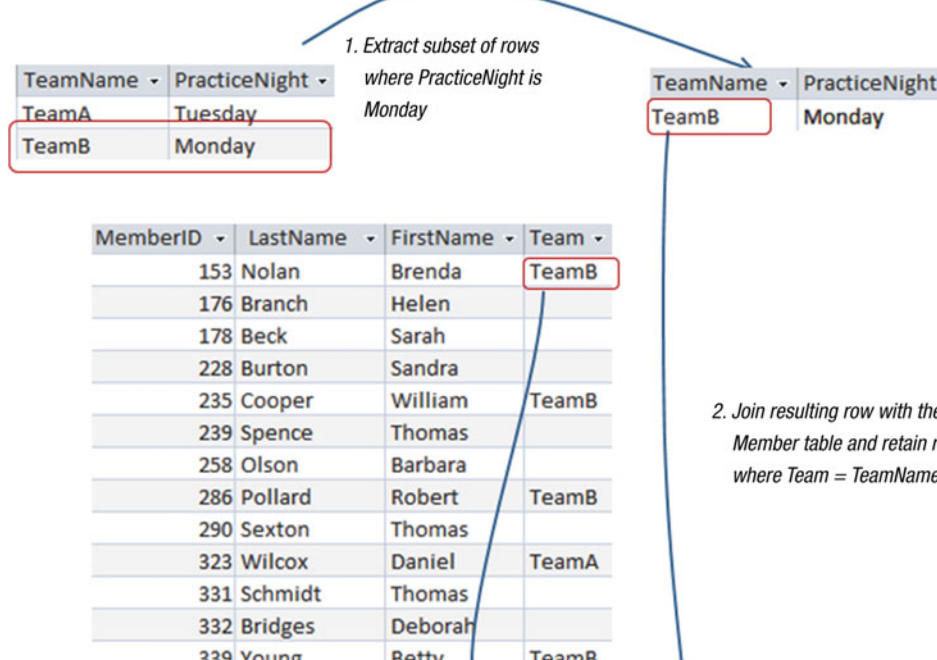


Field:	MemberID	LastName	FirstName	MemberType
Table:	Member	Member	Member	Member
Sort:				
Show:	☐	✓	✓	☐
Criteria:				"Senior"

Figure 1-9. *Access interface for a simple query on the Member table*

The application will take the information from the graphical interface and construct an SQL query. Most applications will show you the SQL that is generated, and you can amend it or write it from scratch yourself. The SQL equivalent to the query depicted in Figure 1-9 is:

```
SELECT FirstName, LastName
FROM Member
WHERE MemberType = 'Senior';
```



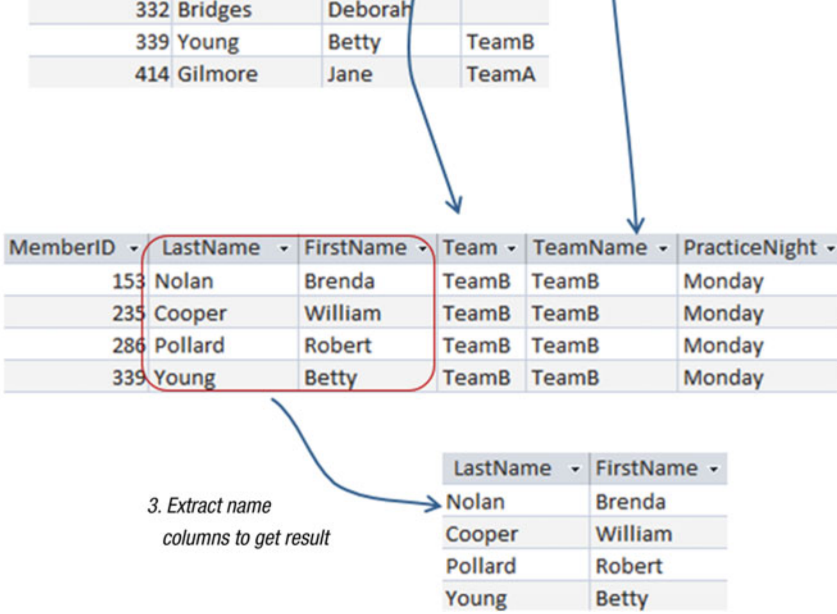
TeamName	PracticeNight
TeamA	Tuesday
TeamB	Monday

1. Extract subset of rows where PracticeNight is Monday

TeamName	PracticeNight
TeamB	Monday

MemberID	LastName	FirstName	Team
153	Nolan	Brenda	TeamB
176	Branch	Helen	
178	Beck	Sarah	
228	Burton	Sandra	
235	Cooper	William	TeamB
239	Spence	Thomas	
258	Olson	Barbara	
286	Pollard	Robert	TeamB
290	Sexton	Thomas	
323	Wilcox	Daniel	TeamA
331	Schmidt	Thomas	
332	Bridges	Deborah	
339	Young	Betty	TeamB

2. Join resulting row with the Member table and retain r where Team = TeamName

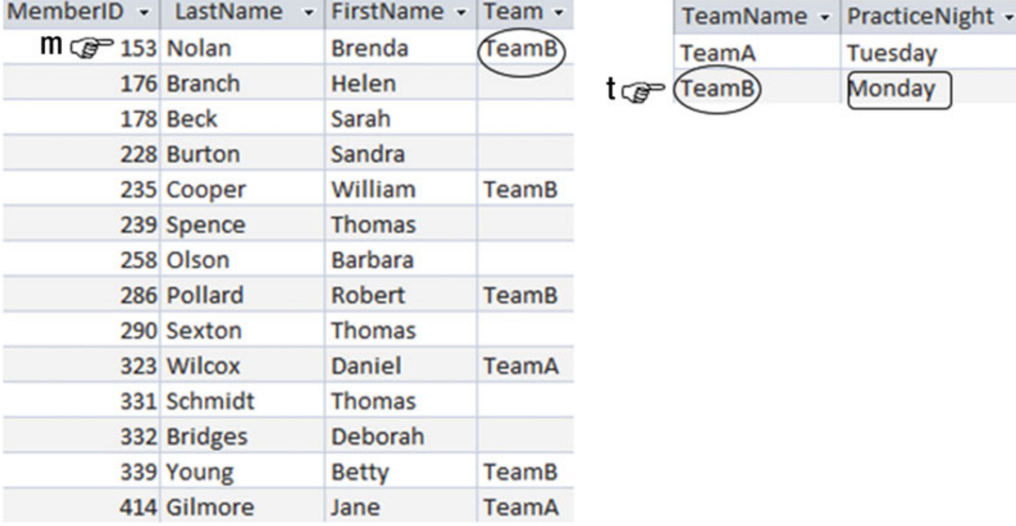


332	Bridges	Deborah	
339	Young	Betty	TeamB
414	Gilmore	Jane	TeamA

MemberID	LastName	FirstName	Team	TeamName	PracticeNight
153	Nolan	Brenda	TeamB	TeamB	Monday
235	Cooper	William	TeamB	TeamB	Monday
286	Pollard	Robert	TeamB	TeamB	Monday
339	Young	Betty	TeamB	TeamB	Monday

LastName	FirstName
Nolan	Brenda
Cooper	William
Pollard	Robert
Young	Betty

Figure 1-10. *The process approach: thinking of a query as a sequence of operations*



MemberID	LastName	FirstName	Team
153	Nolan	Brenda	TeamB
176	Branch	Helen	
178	Beck	Sarah	
228	Burton	Sandra	
235	Cooper	William	TeamB
239	Spence	Thomas	
258	Olson	Barbara	
286	Pollard	Robert	TeamB
290	Sexton	Thomas	
323	Wilcox	Daniel	TeamA
331	Schmidt	Thomas	
332	Bridges	Deborah	
339	Young	Betty	TeamB
414	Gilmore	Jane	TeamA

TeamName	PracticeNight
TeamA	Tuesday
TeamB	Monday

Figure 1-11. *Considering if the row m satisfies the criteria for the query.*