Table Relationships

Access is a *Relational Database* meaning that it can work with data that is held in separate tables and relate the data in one table to that in another. The ability of a relational database to understand how separate sets of data are related makes Access a very powerful tool for the storage and retrieval of information.

What is a Relationship?

A relationship is a definition that declares a link between two tables based upon a common field. Each table is said to be on one or other *side* of the relationship. The ability to define relationships allows related data to be stored in separate tables, allowing greater freedom and efficiency in database design and reducing the need to duplicate data. Many tables can be linked together in this way.

Consider a database of invoices. Each invoice must contain details of the customer (their name, delivery address etc.) and the goods ordered (description, price and quantity). If all this information were to be kept in one table it would be totally impractical to store the details of each invoice as a single record. The only practical way to store the data in a single table would be to create a separate record for each item on the invoice. But this would require much duplication of data, because each record would also have to contain all the details of the customer too. Imagine the complexity if details of the salesperson were to be added to the table.

With linked tables, the data can be stored in different places and then linked together through matching data held in common fields. Access can display a graphical representation of the relationships between tables using the *Relationships* window (*Fig. 1*).

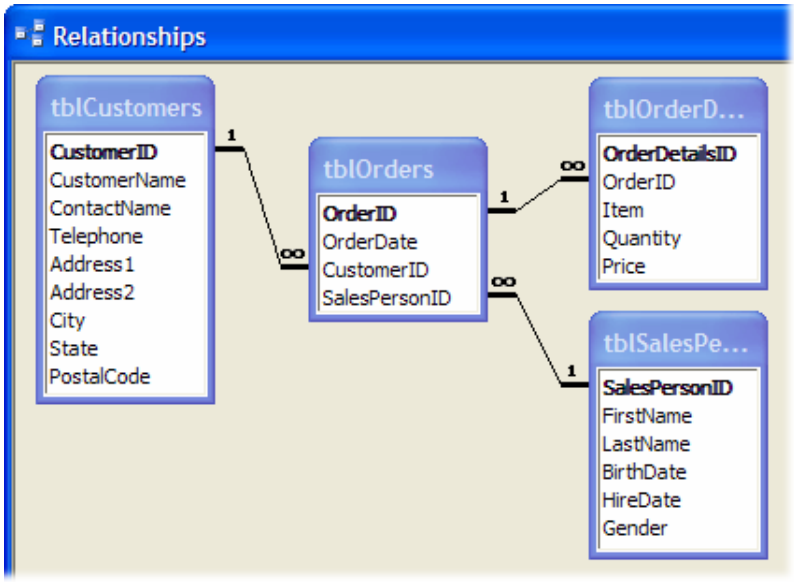


Fig. 1 The relationships window.

Whilst the relationships window might at first appear complex, it helps explain where data is kept in a database and how data in different tables relate to one another. If Access were not a relational database, all the fields shown in the illustration (*Fig. 1*) would have to be stored in a single table. Imagine how much information would have to be duplicated simply to document a single order for half-a-dozen items!

Organising Your Data

Before you can create a relationship between tables you must ensure that the tables on each side of the relationship contain fields of the same data type and that they will contain matching data. If a country is referred to in the country field of one table as *United Kingdom* the relationship will be meaningless if in the other table it is referred to as *UK*. There must be an exact match for the relationship to work. Combo boxes in forms and lookup fields in tables help to ensure that discrepancies like this do not occur. Also, it is helpful, but not essential, to give the fields on each side of a relationship the same name.

Most relationships are based on a link between the *Primary Key* field of one table and a field in another table containing matching data (and usually given the same name). The latter is sometimes referred to as the *Foreign Key* because it contains data which is Primary Key data in the table it is linked to.

How to Create Relationships Between Tables

Having created the tables in your database, open the *Relationships Window* by clicking the **Relationships** button on the toolbar (*Fig. 2*) or going to **Tools > Relationships**. If this is the first time relationships have been created in the database the *Show Table* dialog box will appear. If not, display it by clicking the **Show Table** button (*Fig. 3*).

Fig. 2 Relationships button.

Fig. 3 Show table button.

Use the *Show Table* dialog (*Fig. 4*) to display the field lists for the tables for which you want to create a relationship. Add each field list to the Relationships Window by selecting its name and clicking the **Add** button.

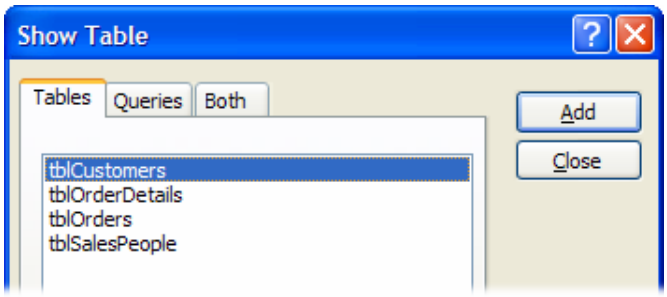


Fig. 4 The Show Table dialog.

Having added the field lists to the Relationships Window you can define the relationships between the tables. There are two ways to do this:

Method 1: Using the Mouse

Use the mouse to point to the name of the chosen field in one field list, then drag it to the other field list and drop it on to the name of the matching field in the other table (*Fig. 5*).

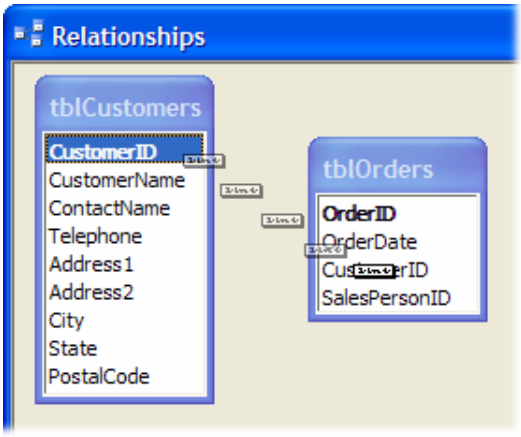


Fig. 5 Creating a table by dragging a field with the mouse

When you release the mouse the *Edit Relationships* dialog opens with the tables and fields you chose already entered (*Fig. 7*). Finally click the **Create** button to close the dialog and create the relationship.

Method 2: Using a Dialog Box

Double-click on an empty space in the Relationships Window or go to **Relationships > Edit Relationships** to open the *Edit Relationships* dialog then click the **Create New** button. This opens another dialog in which you can pick from lists of tables and fields (referred to here as "columns") to specify which fields you want to link (*Fig. 6*).

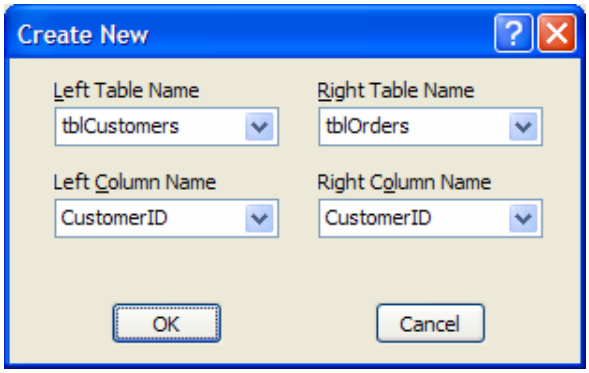


Fig. 6 Specifying related fields using the dialog box

Click the **OK** button to close the dialog box and return to the *Edit Relationships* dialog where your choices of tables and fields will have been entered (*Fig. 7*). Finally click the **Create** button to close the dialog and create the relationship.

The new relationship is indicated by a line joining the tables in the Relationships window (*Fig. 1*). Note that the one and infinity (*1* and *∞*) symbols representing the *One* and *Many* sides of the relationship are only displayed when *Referential Integrity* has been enforced (see below).

Relationships can be edited or deleted. In the *Relationships* window, right-click on the join line and choose **Delete** to remove the relationship, or choose **Edit Relationship** to open the *Edit Relationships* dialog.

What Kind of Relationship Is It?

Note that in the example illustrated here (*Fig. 7*) the *Edit Relationships* dialog defines the *Relationship Type* as *One-To-Many*. This is because Access has detected that a *primary key* field has been related to a field that is not a primary key. This kind of relationship is known as a *One-To-Many* relationship because one side can only have one occurrence of a piece of data (because it is a primary key field and duplicates are not permitted) whereas the corresponding field can potentially have many occurrences of that piece of data.

In this example a customer's *CustomerID* can occur only once in the *tblCustomers* table but if that customer had placed several orders their *CustomerID* would occur several times in the *tblOrders* table.

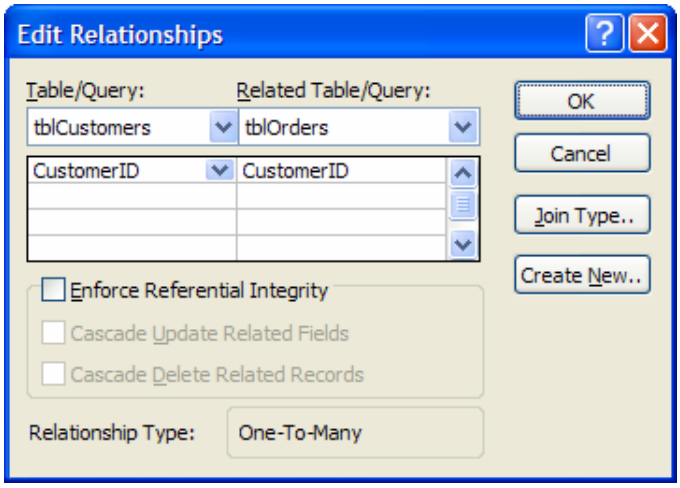


Fig. 7 The Edit Relationships dialog

There are also *One-To-One* relationships, where two primary key fields are linked, and *Many-To-Many* relationships where neither field is a primary key.

Enforcing Referential Integrity

Referential Integrity is a powerful tool. When it is enforced (by putting a tick in the box marked **Enforce Referential Integrity** in the *Edit Relationships* dialog (*Fig. 8*)) Access prevents you from making entries in the table on the *Many* side of a *One-To-Many* relationship if there is not *already* a corresponding entry in the table on the *One* side of the relationship.

This ensures that you don't get "orphan" records in your tables. In this example, it would prevent you from entering orders from a customer who didn't have an account (i.e. who didn't have an entry in the *tblCustomers* table).

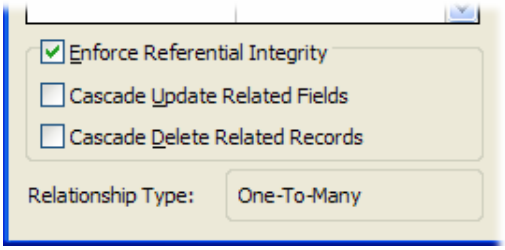


Fig. 8 Enforcing referential integrity

Once it is set up, Access works to maintain referential integrity. If you attempt to delete a record from the table on the *One* side of a relationship Access will prevent you from doing so if related records exist on the *Many* side. In the example, you would be prevented from deleting a customer record from the *tblCustomers* table if there were orders from that customer in the *tblOrders* table. Access displays a message informing you of this restriction (*Fig. 9*).

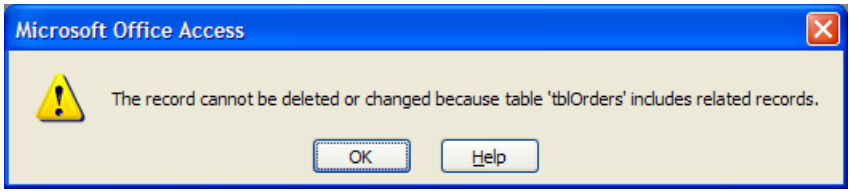


Fig. 9 Referential integrity being enforced

If you attempt to enforce referential integrity when creating a relationship between tables that already contain data, Access checks to see if there are already any records which violate the rules (i.e. records on the *many* side of the relationship which do not have corresponding records on the *one* side). If records are found Access displays a message explaining the problem (*Fig. 10*).

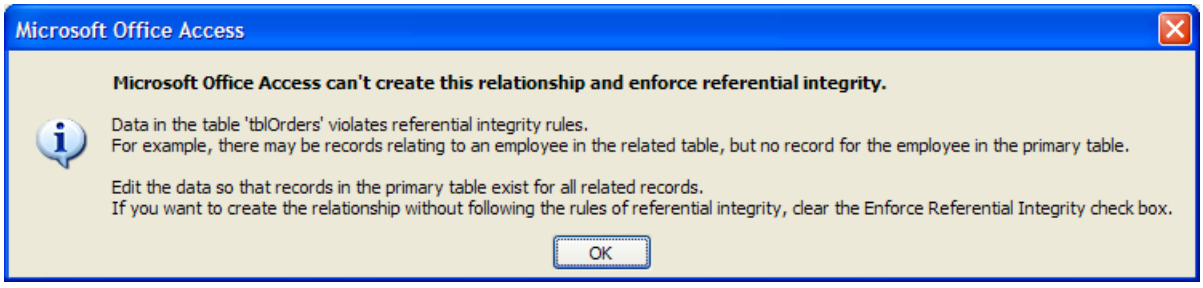


Fig. 10 Access can not enforce referential integrity when orphan records are already present.

Enforcing referential integrity offers additional options to allow changes and deletions whilst still maintaining referential integrity…

Cascade Update Related Fields

Choosing this option causes Access to automatically *update* the data in the field on the *Many* side of the relationship if the corresponding field on the *One* side of the relationship is changed. In the example, if a customer's *CustomerID* is changed all the occurrences of that *CustomerID* in the *tblOrders* table will be changed too.

Cascade Delete Related Records

Choosing this option causes Access to automatically *delete* records from the *Many* side of the relationship if the corresponding record on the *One* side of the relationship is deleted. In the example, if a customer's record is deleted from the *tblCustomers* table then *all* that customer's orders are deleted from the *tblOrders* table.

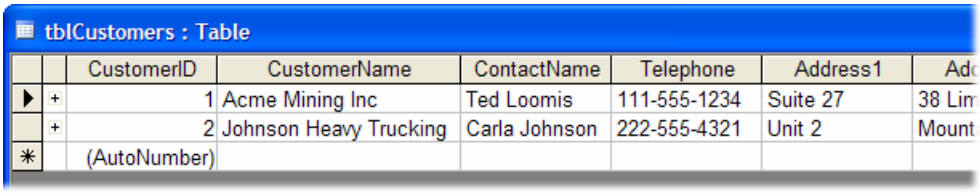
Such automatic deletions can be very useful but are potentially dangerous because they can cause changes or deletions to "cascade" through series of related tables.

In the example, deleting a customer record from the *tblCustomers* table will result in all that customer's orders being deleted from the *tblOrders* table, which in turn will result in all the details of those orders being deleted from the *tblOrderDetails* table. The cascade will *not* affect any records in the related *tblSalesPeople* table because that table is on the *One* side of a relationship.

If referential integrity is enforced but cascading deletes are *not* enabled, Access will prevent you from deleting a record on the *One* side of the relationship if records exist on the *Many* side.

Working in Related Tables

If you open a table that is on the *one* side of a *one-to-many* relationship you will see a column of plus signs (+) to the left of the first field (*Fig. 11*).

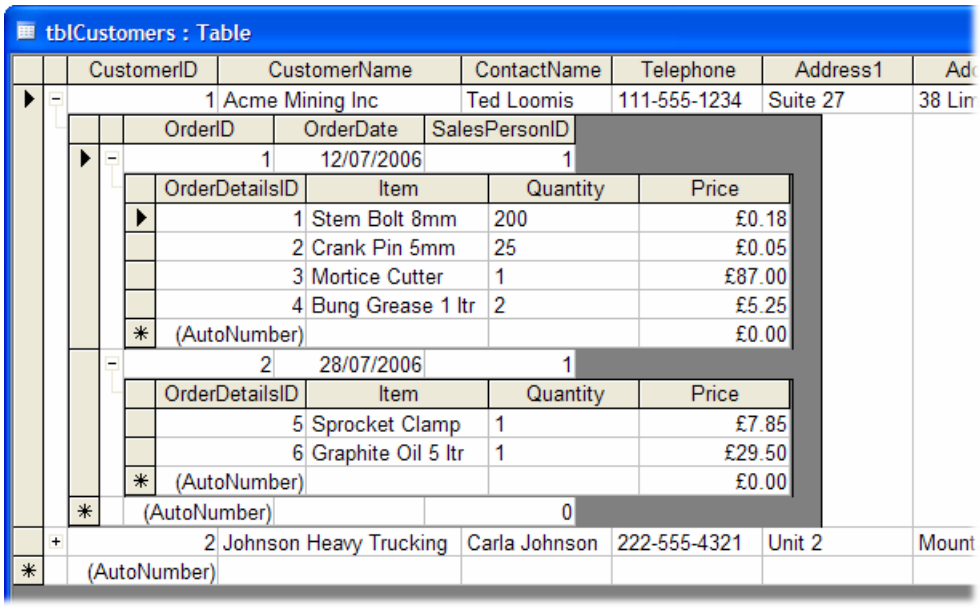


tblCustomers : Table

	CustomerID	CustomerName	ContactName	Telephone	Address1	Ad
+	1	Acme Mining Inc	Ted Loomis	111-555-1234	Suite 27	38 Lin
+	2	Johnson Heavy Trucking	Carla Johnson	222-555-4321	Unit 2	Mount
*	(AutoNumber)					

Fig. 11 The plus signs indicate that there are related records in another table.

Clicking the plus sign adjacent to a record opens the linked table showing just the records related to the one you chose. If this second table is itself on the *one* side of another *one-to-many* relationship it will also display plus signs against each record allowing the view of the data to be further expanded (*Fig. 12*).



tblCustomers : Table

	CustomerID	CustomerName	ContactName	Telephone	Address1	Ad
−	1	Acme Mining Inc	Ted Loomis	111-555-1234	Suite 27	38 Lin

	OrderID	OrderDate	SalesPersonID
−	1	12/07/2006	1

OrderDetailsID	Item	Quantity	Price
1	Stem Bolt 8mm	200	£0.18
2	Crank Pin 5mm	25	£0.05
3	Mortice Cutter	1	£87.00
4	Bung Grease 1 ltr	2	£5.25
(AutoNumber)			£0.00

	OrderID	OrderDate	SalesPersonID
−	2	28/07/2006	1

OrderDetailsID	Item	Quantity	Price
5	Sprocket Clamp	1	£7.85
6	Graphite Oil 5 ltr	1	£29.50
(AutoNumber)			£0.00

	OrderID	OrderDate	SalesPersonID
*	(AutoNumber)		0

+	2	Johnson Heavy Trucking	Carla Johnson	222-555-4321	Unit 2	Mount
*	(AutoNumber)					

Fig. 12 The table's datasheet view can display related records.

Displaying Linked Data on Forms and Reports

Defining relationships in the Relationships Window has many advantages. Since Access knows about the relationships between tables it is able to make use of this when you are building forms and reports.

A table's data can be displayed on a form with the related data displayed on one or more subforms on that form. The form shown below (*Fig. 13*) has two subforms. The main form displays information from the *tblCustomers* table. The two subforms display the related data from the *tblOrders* and *tblOrderDetails* tables.

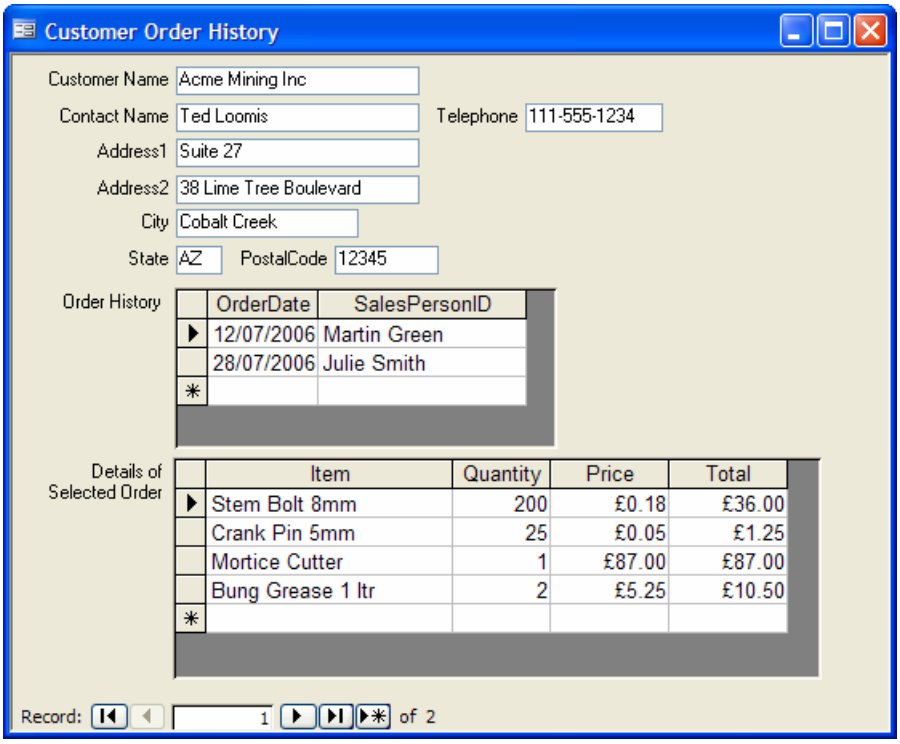


Fig. 13 A form with two linked subforms.

Because Access knows how these tables are related to each other it can display the data logically allowing you to create powerful forms like the one above and reports like the one below (*Fig. 14*):

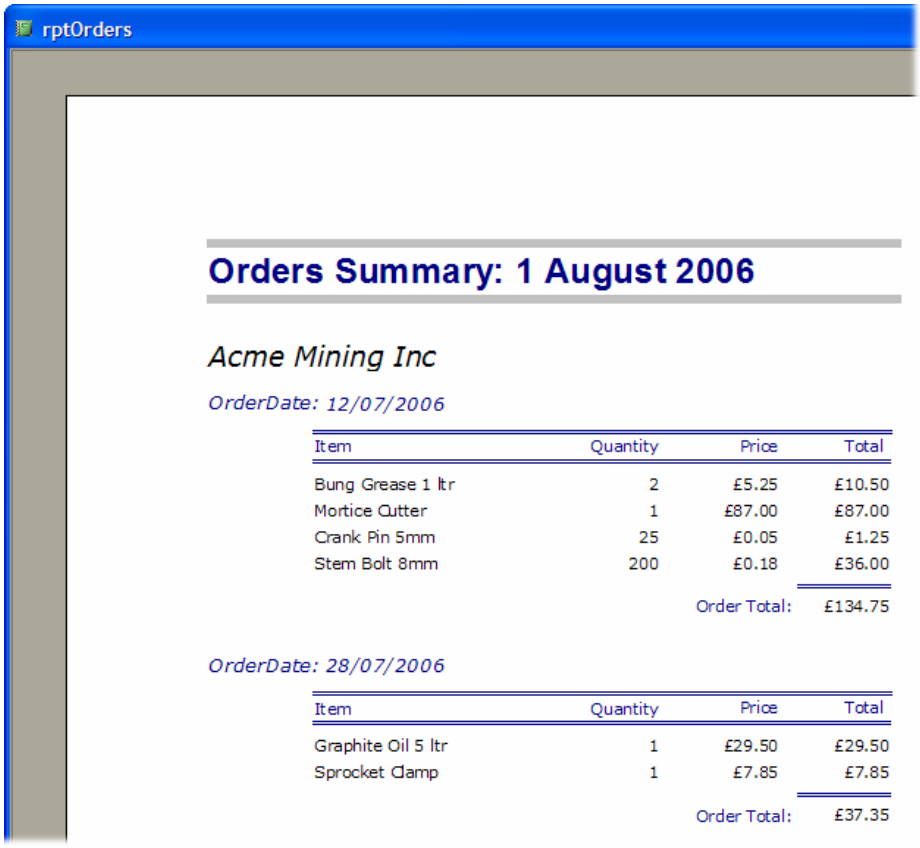


Fig. 14 A report displaying data from three linked tables.