

Section 2

Creating and Linking Tables

By the end of this Section you should be able to:

Create Databases

Create Tables

Understand Data Types

Apply Primary Keys

Apply and Modify Relationships

Understand Different Relationship Types

Understand Referential Integrity

Create Linked Tables

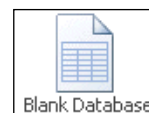
Exercise 7 - Creating a New Database


Guidelines:

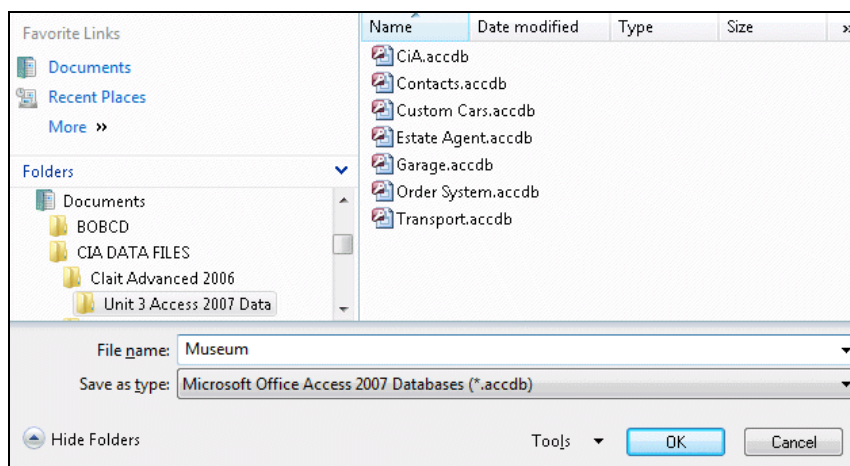
Unlike most applications, Access creates and saves a blank database before any objects have been created or any data has been added.

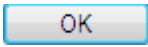
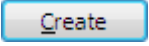
Actions:

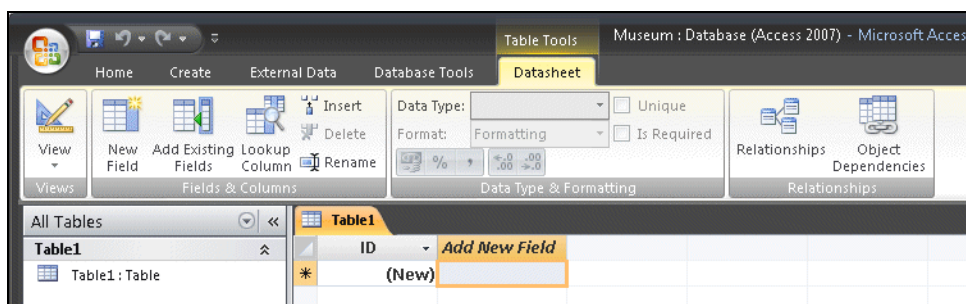
- From the Access **Getting Started** screen, click **Blank Database**. This displays a new panel on the right of the screen.



- Use the  button to browse for a location to put your database. Make sure the **Folders** pane is shown on the left and use it to locate the supplied data folder for this guide. Enter a **File name** of **Museum**.



- Click  in the **File New Database** dialog box.
- In the **Blank Database** panel, click .
- The **Access Database Window** is now displayed, with the database name (Museum) in the **Title Bar** and a default empty table **Table1** open.



- In order to demonstrate the general method of creating tables, close the default table, by right clicking on the **Table1** tab, and selecting **Close** from the shortcut menu. Leave the database open.

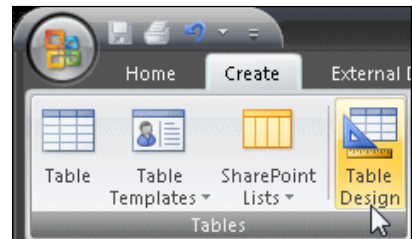
Exercise 8 - Creating Tables

Guidelines:

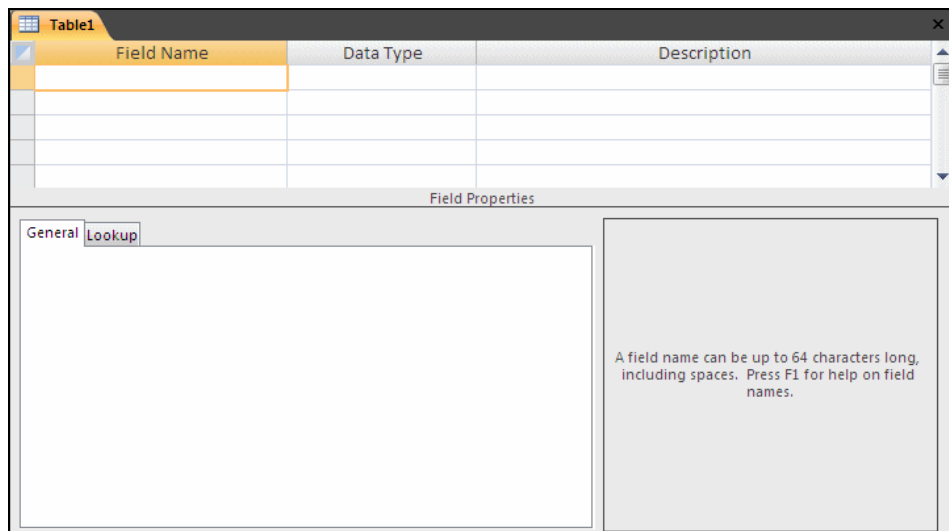
Tables are the fundamental objects of the database. However complex the database, all data is stored in tables. Tables can be created using a wizard which gives step by step instructions for the initial setting up and applies default formatting, such as date formats and number formats. Alternatively, a table can be created in **Design View** which allows a free hand for formatting.

Actions:

1. On the new **Database** screen, display the **Create** tab and click the **Table Design** button. A new table appears in the work area in **Design View**, which allows the structure of the table to be defined.



2. Notice that the window is divided into two areas. The upper half of the **Design View** screen allows fields for the table to be defined. The lower half has panels to specify more detailed properties of each field, as well as some explanatory text, specific to the property being defined.



3. In the first **Field Name** row, enter **Artefact**.
4. Leave the **Design** window open for the next exercise.

Exercise 9 - Data Types

Guidelines:

Each field in an Access table has a **Data Type** associated with it. This defines the type of data that can be contained within the field.

The following available data types are covered in this unit:

<u>Type</u>	<u>Description</u>
Text	Any combination of characters and numbers up to a maximum length of 255.
Memo	Same content as Text but up to 64,000 characters allowed.
Number	Numeric data only. Essential for calculations.
Date/Time	Will only accept dates or times. A variety of formats may be specified.
Currency	Numeric data related to currency. The currency symbol can be changed.
AutoNumber	A sequential number applied by Access to each new record in a table.
Yes/No	A logical field that represents yes/no or true/false values. Actually contains either a 1 or 0.

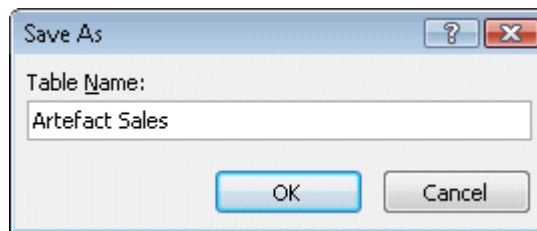
Actions:

1. In the table **Design** window open from the previous exercise, click in the **Data Type** field for the first record. The default type of **Text** is shown.
2. Press <Tab>. The cursor is now in the **Description** column; this column is optional. Enter **Name of Artefact** into this column, then press <Tab>.
3. Enter a second row with a **Field Name** of **Era**, **Data Type** of **Text** and **Description** of **Historical Period**.
4. For the next **Field Name**, enter **Date Acquired** and press <Tab>.
5. Click the drop down arrow in the **Data Type** field and select **Date/Time** from the list.
6. Press <Tab> to move into the **Description** column and type **Date the artefact was bought** then press <Tab> again.
7. Enter **Cost** as the fourth field name, <Tab>, then select **Currency** as the **Data Type** and enter **Buying price** as the **Description**.
8. The next **Field Name** is **Quantity**, the **Data Type** is **Number** and the **Description** is **Number of artefacts bought**.

continued over

Exercise 9 - Continued

9. For the next field enter **Supplier Name**, the **Data Type** is **Text** and the **Description** is **Name of Supplier**.
10. For the next field enter **Supplier Address**, the **Data Type** is **Text** and the **Description** is **Address of Supplier**.
11. For the next field enter **Information**, the **Data Type** is **Memo** and the **Description** is **History**.
12. It is decided to give each record a unique reference number. Click anywhere in the first field name (**Artefact**) and select **Insert Rows** from the **Design** tab.
13. A blank row is inserted. Enter a **Field Name** of **Artefact Reference**, select a **Data Type** of **AutoNumber**, and enter a **Description** of **Artefact Record ID**.
14. Right click on the **Table1** tab and click **Save**. Because the table has not been saved before, the **Save As** dialog box is displayed. Type **Artefact Sales** as the **Table Name**.



15. Click **OK**. A message is displayed regarding **Primary Keys**.



16. Although not essential for a single table database, click **Yes**. The **AutoNumber** field will be automatically selected to be the **Primary Key**.

Note: At present this is not an efficient database design. As the same suppliers provide different artefacts, it is inefficient to store the supplier name and address on each artefact record. It would be better to create another table containing supplier details and relate the relevant information to each artefact. This will be done in a later exercise.

17. Click **All Tables** in the **Navigation Pane** and select **Object Type** from the list. Close the table and the **Museum** database.

Exercise 10 - Applying a Primary Key

Guidelines:

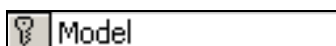
When creating databases which use more than one table, it is important to be able to uniquely identify individual records in a table if they are to be referenced from another table. Records can be identified by finding a field in the table which contains unique data for each record, e.g. a serial number or identification number. This is then defined as the **Primary Key**.

If there is no suitable field in the table, a new field can be added specifically for the purpose.

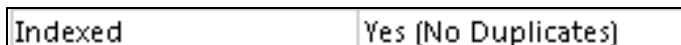
As well as enabling the linking of tables, use of a **Primary Key** prevents duplication of records in a table and also allows sorting and querying to be performed more efficiently.

Actions:

1. Open the **Custom Cars** database. This contains a table with vehicle details including registration number, and a table of repair details which also contains registration number.
2. Open the **Vehicles** table in **Design View** and look at the fields with a view to selecting a field to become the **Primary Key**.
3. Click in the **Model** field. Click the **Primary Key** button, on the toolbar. A **Primary Key** is then applied to the **Model** field.



4. Look at the **Field Properties** for this field.



5. Notice in the field properties that the **Indexed** property is automatically set to **Yes (No Duplicates)**. This means that there can be no duplicate **Model** in any two records within this table. This is obviously a bad choice for the **Primary Key** field as it is likely that there will be more than one record in the table for any particular model of vehicle.
6. Click in the **Reg No** field and click the **Primary Key** button again.
7. **Reg No** is now designated as the **Primary Key** field. This is an ideal choice, as a registration number is a unique identifier for any particular vehicle.

continued over

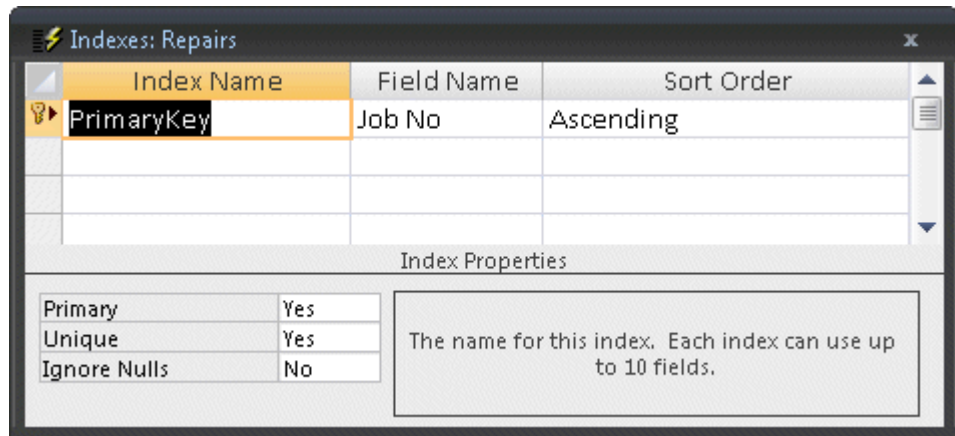
Exercise 10 - Continued

*Note: When **Reg No** was selected as the Primary Key, the indicator was automatically removed from the **Model** field.*

8. Save and close the table.
9. Open the **Repairs** table in **Design View**. A **Primary Key** for this table is already applied to the **Job No** field.

*Note: **Reg No** is not a suitable Primary Key for the **Repairs** table because there may well be more than one job record for the same vehicle.*

10. Click the **Indexes** button on the **Design** tab, to view the indexes applied to this table. An index in a table automatically maintains a sort sequence for the records in the table. The **Primary Key** field is automatically an index.



*Note: The **Index Properties** show that a **Primary Key** field is unique and that **Ignore Nulls** is set to **No**. This means that as well as being unique, a primary key field cannot be left blank in a record.*

11. Click in the **Unique** box under **Index Properties**, click the drop down arrow and select **No** from the list.
12. A message box explains why this change is not allowed. Read the text and click **OK**. All **Index Properties** for a **Primary Key** field are fixed.
13. Close the **Indexes** dialog box and the **Repairs** table.
14. Leave the **Custom Cars** database open with the **Navigation Pane** displayed.

Exercise 11 - Applying Relationships

Guidelines:

Once tables have been designed and primary keys applied, a **Relationship** may be applied between two or more tables to link them together. Once two or more tables are linked by a relationship, the data from all of the tables may be used to create a single query, form or report.

Relationships are applied between tables which contain a common field. Usually, the related field in the first table, containing the unique record, is the **Primary Key**. The related field in the second table, which is used to link to information from the first table, is known as the **Foreign Key**.

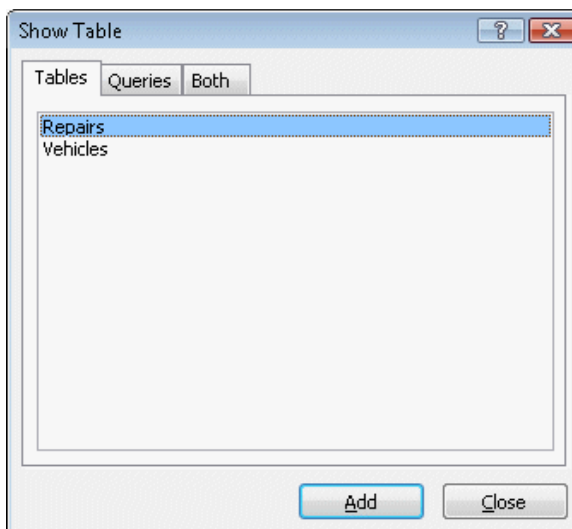
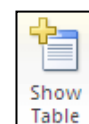
Applying relationships allows many smaller tables to be linked together to form an efficient database. In this exercise the **Vehicle** and **Repairs** tables are linked using the common field **Reg No**, so there is no need to have all vehicle details on every repair record. A query on the **Repairs** table will use **Reg No** to access all related data on the **Vehicles** table automatically. In this example, **Reg No** on the **Vehicles** table is the **Primary Key**, and **Reg No** on the **Repairs** table is the **Foreign Key**.

Actions:

1. With the **Custom Cars** database open, display the **Database Tools** tab and click the **Relationships** button. A blank relationship area is displayed and the **Design** tab is shown on the **Ribbon**.


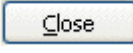


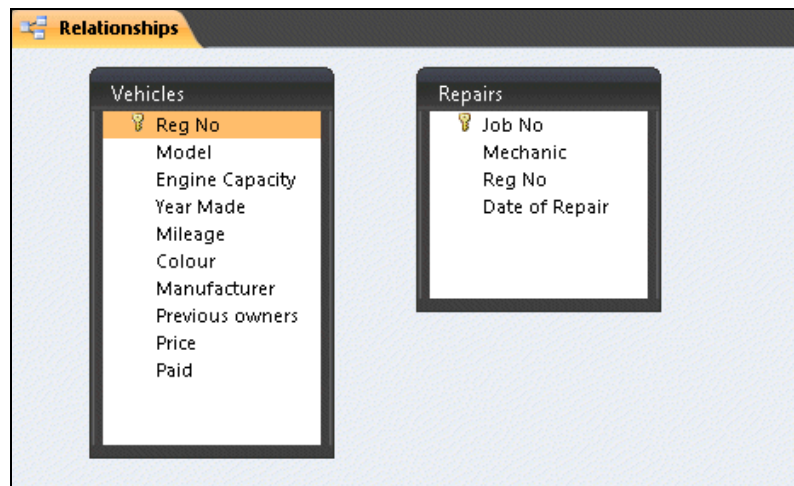
2. If the **Show Table** dialog box does not appear, click the **Show Table** button on the **Design** tab.



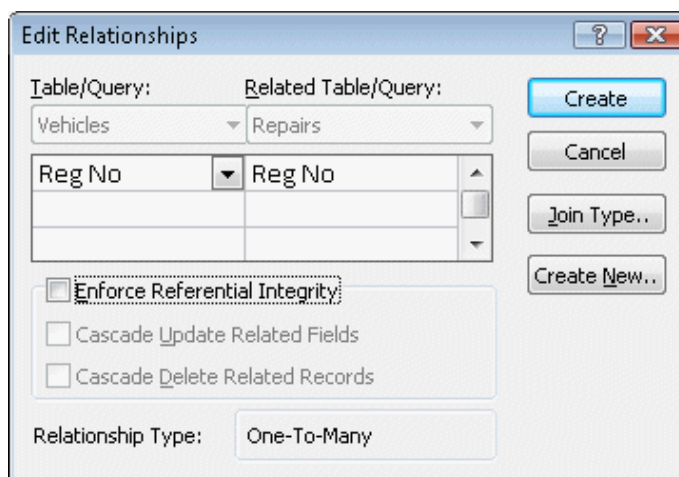
continued over

Exercise 11 - Continued

- With the **Vehicles** table highlighted, click  to place the table in the **Relationships** window.
- Click on the **Repairs** table and again add it to the window then click  to remove the **Show Table** dialog box.
- Resize the table boxes to see all of their fields. Notice the **Primary Keys** for each table are indicated by a **Key** symbol.



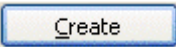
- Highlight the **Reg No** field in the **Vehicles** table.
- Drag the **Reg No** field, from the **Vehicles** table, over the **Reg No** field (the foreign key) in the **Repairs** table. Release the mouse when in position.

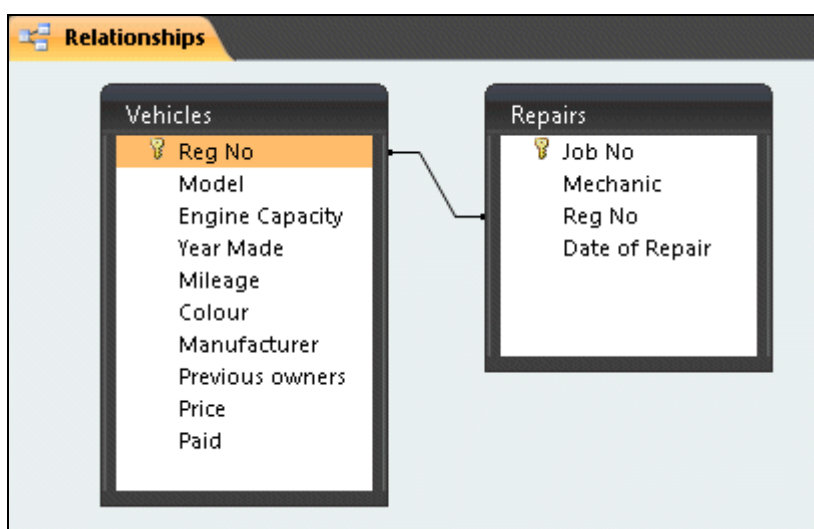


- In the **Edit Relationships** dialog box, note the relationship type is **One-To-Many** (one vehicle record can be linked to many repair records). This is the most common type of relationship.

continued over

Exercise 11 - Continued

9. Click  to create the relationship.
10. Notice how the relationship between the tables is now symbolised by a line, linking the same field, **Reg No**.



Note: The relationship is **One-To-Many**. This means that one record from **Vehicles** can have many related records in **Repairs**, i.e. one car may have many jobs. The fact that one of the fields in the link is a primary key and the other is not, defines the relationship as **One-To-Many**.

11. To delete the relationship, right click on the connecting line and select **Delete**. Click **Yes** to confirm the deletion.

Note: Any relationship can be modified by right clicking on the connecting line and selecting **Edit Relationship** from the shortcut menu.

12. Now click and drag **Mileage** in the **Vehicles** table to **Mechanic** in the **Repairs** table.
13. This is not a valid relationship. The relationship type is shown as **Indeterminate** in the **Edit Relationships** dialog box. Click **Cancel** then reapply the original relationship between **Reg No** in both tables.
14. Right click in a blank part of the **Relationships** window and select **Close** to close it. Select **Yes** when prompted to save the changes to the relationship.
15. Leave the database open for the next exercise.

Exercise 12 - Creating Other Relationships

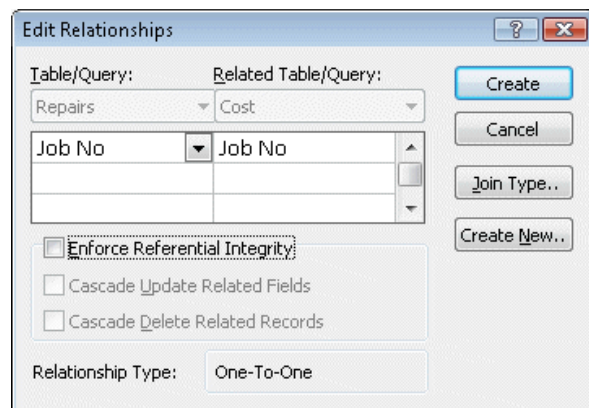
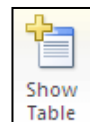
Guidelines:

A **One-to-One** relationship is used where one record in one table is linked to only one record in another. This can be used to split a table with many fields or if part of a table is removed for security reasons or if the second table contains optional data.

A **Many-to-Many** relationship is used when a record in the first table can have many matching records in the second, and vice versa. For example, a single product may have many orders and a single order may be for many products. A single **Many-to-Many** relationship cannot exist. An intermediate junction table must be created with **One-to-Many** links to the two original tables. It must contain two fields: the foreign keys from both tables.

Actions:

- To create a **One-to-One** relationship, using the **Custom Cars** database, create a table in **Design View** with two fields, **Job No** and **Charge**.
- The **Job No** field will have a data type of **Number** and the **Charge** field will be **Currency**.
- Make the **Job No** field the **Primary Key**.
- Save the table as **Cost**. Close it without adding any data.
- Open the **Relationships** window showing the existing relationship and use the **Show Table** button to display the dialog box.
- Select **Cost** then click **Add** to add it to the **Relationships** window. Close the **Show Table** box.
- Reposition the **Cost** table if necessary, then make the link between **Job No** in the **Repairs** table and **Job No** in the **Cost** table. Because each of these fields is a primary key, the **Relationship Type** is displayed as **One-To-One** in the **Edit Relationships** box.



continued over

Exercise 12 - Continued

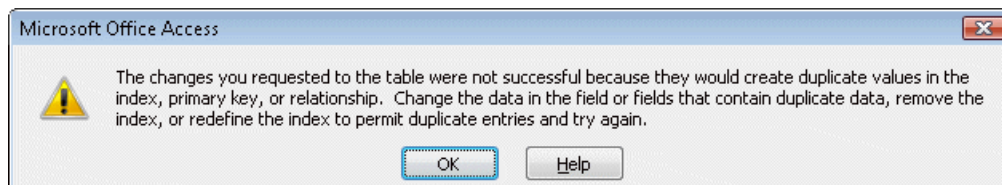
8. Click **Create** and close the **Relationships** window, saving at the prompt.
9. Open the **Vehicles** table and click the expand subdatasheet button next to the first record.


Vehicles			
	Reg No	Model	Engine
	379 CZ	QUATTRO	3500
	B123 FJK	250	2500

10. The related **Repairs** records are shown – notice that **Reg No**, the related field in the second table (the foreign key), is not shown.
11. There is also now an expand subdatasheet button in the **Repairs** table. Click on it. The **Cost** records are shown for this record.

Vehicles					
	Reg No	Model	Engine Capa	Year Made	Mileage
	379 CZ	QUATTRO	3500	1986	15082
	Job No		Mechanic	Date of Rep:	Add New Field
	11 Keith		03/01/2002		
	Charge		Add New Field		
	*				
	*	(New)			
	B123 FJK	250	2500	1983	57950

12. At the moment there is no data in the **Charge** field. Add a charge of **250** for this repair and press <Enter>.
13. Try adding another charge for this repair. Remember there is a **One-to-One** relationship in force, so only one charge should be allowed. A warning dialog box is displayed as below.

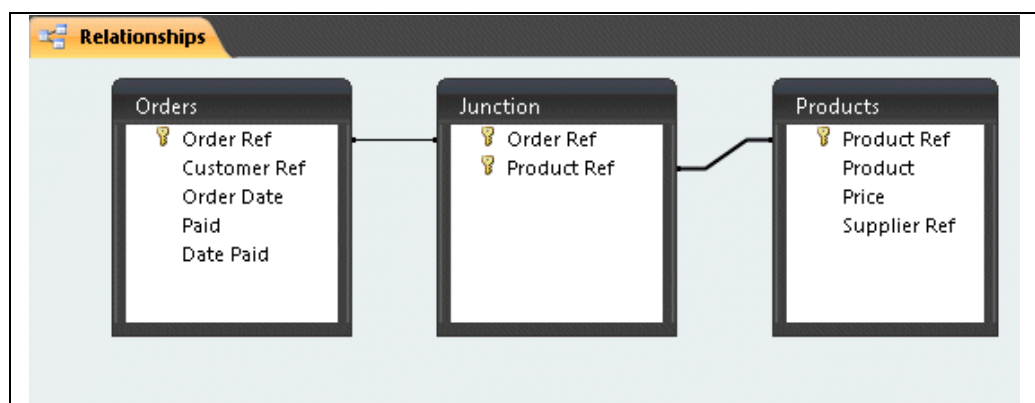


14. Click **OK** once you have read it. The only way to get rid of this useless extra record is to use the **Undo** button, . Click **Undo** now.
15. The extra entry is deleted and the subdatasheets can be collapsed.

continued over

Exercise 12 - Continued

16. Open each subdatasheet in turn and add suitable **Charges** for all **Repairs**.
17. Close the table and the database.
18. Open the **CiA** database. To create a **Many-to-Many** relationship between the **Orders** table and the **Products** table (an order can include several products and a product can be on several orders) it is first necessary to create an intermediate, junction table.
19. Create a new table in **Design View** containing the fields **Order Ref (Number)** and **Product Ref (Text)**.
20. The **primary key** for this new table will be defined as the combination of both fields (neither individual field will be unique on the table). Click in the beige area at the left of **Order Ref**, hold down **<Ctrl>** and click at the left of **Product Ref**. Both fields will be selected.
21. Click the **Primary Key** button to apply .
22. Save the table as **Junction** and close it.
23. View the **Relationships** window and place the **Orders**, **Junction** and **Products** tables on to it.
24. Create a **One-to-Many** relationship between the **Orders** table and the **Junction** table using the **Order Ref** field.
25. Create a **One-to-Many** relationship between the **Products** table and the **Junction** table using the **Product Ref** field. The overall effect is that now a **Many-to-Many** relationship exists between the **Orders** and **Products** tables.



26. Close the **Relationships** window, saving at the prompt.
27. Close the database.

Exercise 13 - Referential Integrity

Guidelines:

Referential Integrity is a set of rules which can be applied to relationships, ensuring they are valid and that data is not accidentally deleted or changed. It may be applied when specific conditions are met: the matching field from the primary table is a primary key, the related fields are the same data types and both tables belong to the same database.

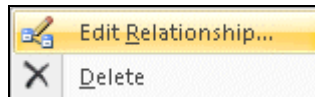
Enforcing referential integrity controls the updating of primary key data in the primary table and the deletion of any record from the primary table, if a related record exists elsewhere. A record cannot be added to a related table if a record does not exist in the primary table, e.g. there can be no job record without an associated vehicle record in the primary table.

Actions:

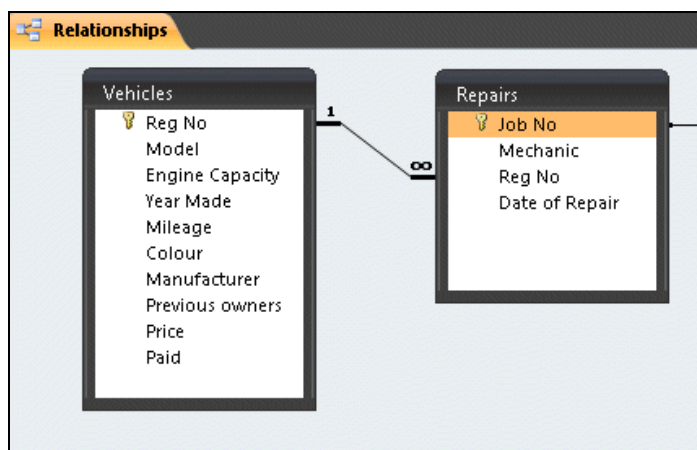
1. Open the database **Custom Cars** and click the **Relationships** button to display the relationships.



2. Right click with the mouse on the relationship line between **Vehicles** and **Repairs** and select **Edit Relationship** from the menu.



3. In the **Edit Relationship** dialog box, check the box for **Enforce Referential Integrity** and click **OK**.



Note: Enforcing referential integrity will change the relationship line to show the type of relationship, in this case, one to many.


4. Close the **Relationships** window and select **Yes** to save, if the prompt appears.
5. Leave the database open for the next exercise.

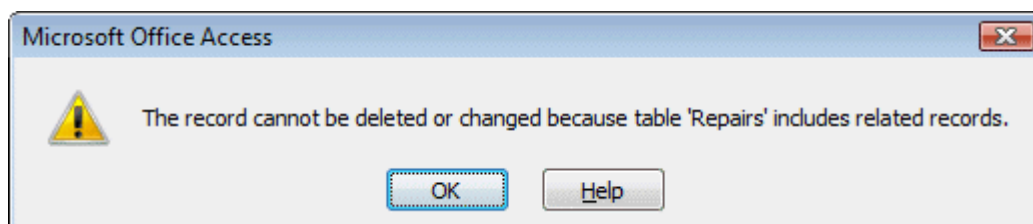
Exercise 14 - Cascade Options


Guidelines:

If referential integrity is enforced, deleting records from the primary table or changing a primary key is either prevented or controlled. Similarly, a record in a related table may not be created, if a matching record is not available in the primary table.

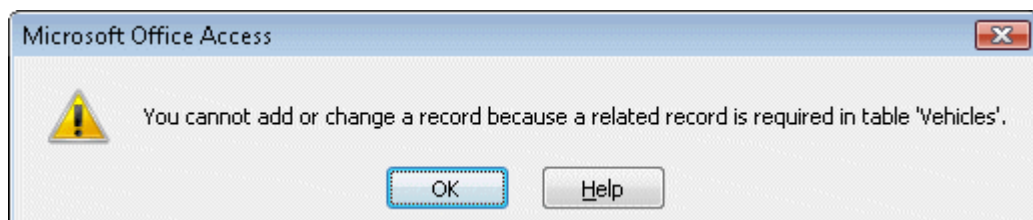
Actions:

1. Open the **Vehicles** table in **Datasheet View**.
2. Click in the record **R654 TFG** and click the drop down arrow on the **Delete** button, , from the **Records** group on the **Home** tab.
3. Select **Delete Record**. The record can not be deleted, as there are related records in the **Repairs** table.



4. Click **OK**. Close the table.
5. Open the **Repairs** table in **Datasheet View**.
6. Click the **New Record** button,  from the **Home** tab and enter the following information starting with the **Mechanic** field (Access will add the next sequence number (22) to the AutoNumber **Job No** field):

David D325 ABC 04/01/02
7. Press **<Enter>** after the last entry. As there is no registration number for this job in the **Vehicles** table, a new job record cannot be created.

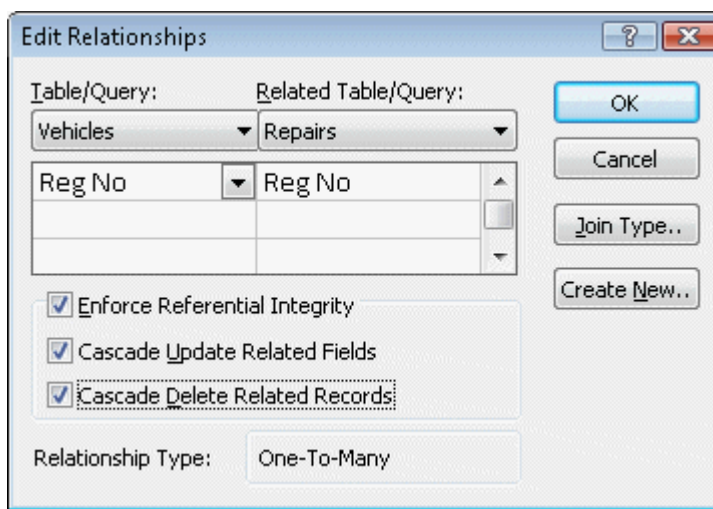


8. Click **OK**. Delete the information just entered using **Undo**. Close the table and return to the **Navigation Pane**.

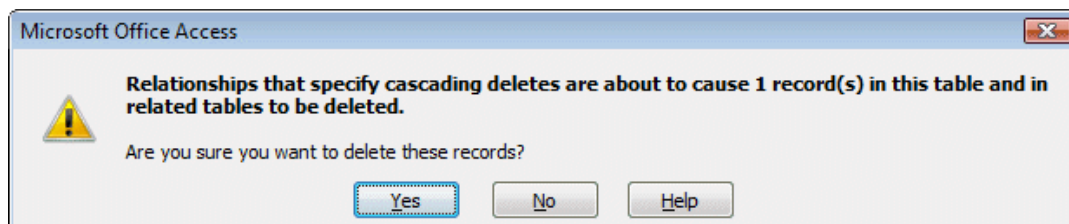
continued over

Exercise 14 - Continued

9. To allow editing or deletion of records, the relationship between the tables must be changed. Click the **Relationships** button.
10. Right click on the linking line between **Vehicles** and **Repairs** and select **Edit Relationship**.
11. Edit the relationship as follows: check the boxes for **Cascade Update Related Fields** and **Cascade Delete Related Records**.



12. Click **OK**. This means that now any changes made in one related table will be reflected in the others.
13. Close the **Relationships** window.
14. Open the **Vehicles** table in **Datasheet View** and delete record **8** (for the Hillman Imp, FSH 541F). There is a repair record for this vehicle on the **Repairs** table, record **14**.



15. Deletion is now allowed but there is a warning that records in related tables (**Repairs**) will also be deleted. Click **Yes** to continue.
16. In the **Vehicles** table, change the registration **D325 SJR** to **D325 SJP**.
17. Close the table and open the **Repairs** table in **Datasheet View**. The job for the Hillman Imp (job 14) is removed and the registration number change has been reflected in the table (job 12).
18. Close the table and the database.

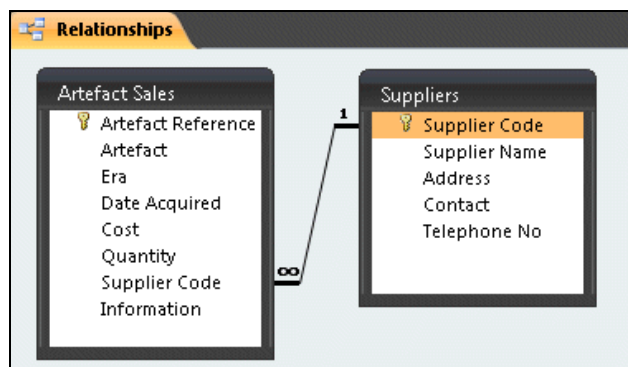
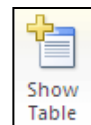
Exercise 15 - Creating Linked Tables

Guidelines:

Linking tables that already contain data can cause problems if the existing data is not consistent across all the tables. Normally, tables are linked as they are being designed, before any data is added. Integrity of the data can then be controlled by the relationships.

Actions:

1. Open the database **Museum** and the **Artefacts Sales** table.
2. In **Design View**, delete the **Supplier Name** and **Supplier Address** fields and replace them with a field named **Supplier Code**. The **Data Type** is **Text** and the **Description** is **Supplier Reference Code**.
3. Save and close the **Artefacts Sales** table and create a new table. Add the following fields: **Supplier Code**, **Supplier Name**, **Address**, **Contact**, **Telephone No**, all **Data Types** to be **Text**.
4. Apply **Primary Key** to the **Supplier Code** field, then save the table as **Suppliers** and close it.
5. Open the **Relationships** window and if necessary, use the **Show Table** button to display the dialog box.
6. Add both **Artefacts Sales** and **Suppliers** to the **Relationships** window. Close the **Show Table** box.
7. Create a **One-To-Many** relationship between the **Artefacts Sales** table and the **Suppliers** table using the **Supplier Code** field. Select **Referential Integrity** and all **Cascade** options and click **Create**.



8. Now more information can be held for each supplier in the **Suppliers** table but all that is ever held on the **Artefacts Sales** table is the field **Supplier Code**.
9. Save and close the **Relationships** window and close the database.

Exercise 16 - Revision: Creating and Linking Tables

Assessment

The relevant assessment criteria for this section are to: create a relational database with at least 3 tables, use at least 4 data types, use primary and foreign keys, produce hard copies of tables and relationships.

1. Create a new blank database and save it as **Time Sheet**.
2. Create a table containing the following fields and save it as **Staff**:

Field Name	Type	Description
Staff No	Text	Primary Key
First Name	Text	
Surname	Text	
Department	Text	Choose from list
Date of Birth	Date/Time	
Cost	Currency	Calculated cost to the company
Extension	Number	Internal telephone number

Note: For this and all subsequent tables, accept the default properties for every field.

3. Create a table containing the following fields and save it as **Project**:

Field Name	Type	Description
Project Code	Text	Primary Key
Project Name	Text	
Project Type	Text	Choose from list
Active	Yes/No	

4. Create a table containing the following fields and save it as **Time Sheet Header**:

Field Name	Type	Description
Time Sheet Ref	AutoNumber	Primary Key
Staff No	Text	Foreign Key (link to Staff table)
Week No	Number	

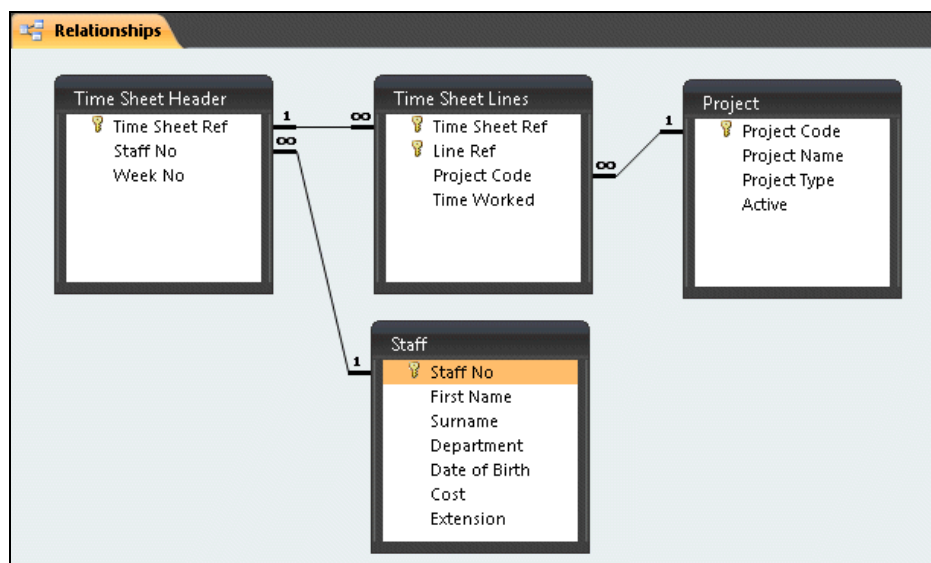
continued over

Exercise 16 - Continued

5. Create a table containing the following fields and save it as **Time Sheet Lines**:

Field Name	Type	Description
Time Sheet Ref	Number	Primary Key
Line Ref	AutoNumber	Primary Key
Project Code	Text	Foreign Key (link to Project table)
Time Worked	Number	

6. The **Primary Key** for this table is **Time Sheet Ref** and **Line Ref**. Highlight both lines in the table design window before clicking the **Primary Key** button. **Time Sheet Ref** is also a **Foreign Key** linking back to the **Time Sheet Header** table.
7. Create the following relationships between all the tables. Ensure that **Referential Integrity** and both **Cascade** options are specified for each link.



8. Start *Word* (or any other word processing application) and create a document to contain the evidence required for this task. Add a title **Timesheet Database System** and save the document as **Revision Evidence**.
9. Create a footer for the document and add **CANDIDATE NAME** followed by your name and the **CENTRE NUMBER** followed by the reference number.
10. In *Access*, display the **Staff** table in **Design View** and take a screen dump of the window.

continued over

Exercise 16 - Continued

11. Paste the screen dump into the **Revision Evidence** document and add a suitable caption.
12. Repeat for the other three tables. This covers **objectives 1c, 1d, 1e** and part of **1i** of the evidence checklist.
13. Paste a screen image of the **Relationships** window into the document (**objective 1d** and part of **1i**).
14. For this and subsequent revision exercises, annotate the various printouts and screen dumps with the relevant objective reference to aid assessment.
15. Save the **Revision Evidence Word** document and close it.
16. Close the **Time Sheet** database, saving any unsaved objects.

*Note: Completing this section of the guide has allowed the following elements of the **Evidence Checklist** to be completed. In your checklist, the ticks must be replaced by the page numbers of your portfolio where the relevant evidence is located. This task may have to be delayed until the portfolio is complete and page numbers have been applied.*

1	create a relational database using advanced design features.	Page Number	
1c	create a relational database of 3 tables with a minimum of 100 records	✓ ✓ ✓	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
1d	use primary key and foreign key	✓	<input type="checkbox"/>
1e	use at least 4 data types	✓ ✓ ✓ ✓	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
1i	produce hard copy evidence of: <ul style="list-style-type: none"> • tables • relationships between tables • modification of field characteristics • validation • linking with other applications 	✓ ✓	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>

Note: The required data records for the tables will be added in a later exercise.