

# Lesson 20: Recording supplier shipments

## 20.1 Introduction: Inflows and outflows of product over time

Your system—as it currently stands—is capable of subtracting items from inventory as you make shipments to your customers. However, it is clear that at some point, you must replenish your inventory with shipments from your suppliers. In order to have an accurate picture of your inventory level, you need to account for both incoming and outgoing flows of products.<sup>1</sup>

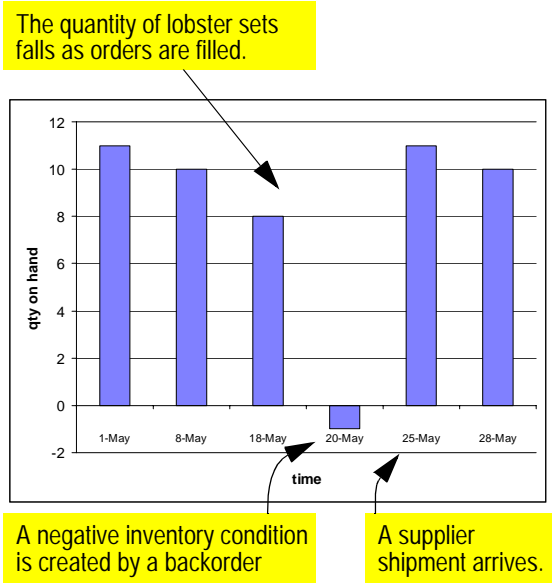
Consider the inventory level of a product over time. For example, the **QtyOnHand** value for ProductID = 74 6881 (“lobster set”) is shown in **Figure 20.1**. The initial inventory is 11 items; however, as customer orders are filled, the inventory drops and as supplier shipments are received, the inventory rises.

In this lesson, you are going to enhance your system to deal with inflows of product, such as

<sup>1</sup> In the real world, you would also have to account for “shrinkage”—loss of product due to breakage, theft, and so on. In the face of inventory shrinkage, the only way to keep your inventory record accurate is to periodically count the physical items in your warehouse and reconcile the physical count with the inventory levels in the database. To keep things simple in this lesson, we ignore shrinkage.

the shipment from one of your suppliers on May 25<sup>th</sup>. The issue of what to do when inventory drops below zero (i.e., backorders) is addressed in **Lesson 21**.

FIGURE 20.1: Changes in inventory level over time for product 74 6881





## 20.2 Learning objectives

- gain more experience creating tables, relationships, and queries
- gain more experience building multi-part forms
- create an event-driven procedure to update inventory levels on receipt of a shipment
- understand the difference between inventory tracking and inventory management

## 20.3 Exercises

A supplier shipment is an event that causes changes in a status field (**QtyOnHand** in the **Products** table). As such, you will want to record the details of “shipment” events in the same way that you record details of “customer order” events.

Given the fixed costs associated with shipping physical goods (e.g., a truck, a driver, and so on), a shipment is typically a nested transaction—that is, each shipment transaction consists of many shipment detail transactions.

Since you already have experience with nested transactions from **Orders** and **OrderDetails**, implementing the tables, forms, and event-driven procedures for recording supplier shipments uses skills you already possess.

### 20.3.1 Creating tables

➔ Create a **Shipments** table to record the critical attributes of a shipment event. The table should include the date of the shipment, the ID of the supplier, and whether the shipment has already been processed.



Think carefully about an appropriate primary key for the **Shipments** table. Although you could use a concatenated primary key such as **ShipDate** + **ShipTime** + **SupplierName**, it is probably a better idea to use a **surrogate primary key**, such as **ShipmentID**.

➔ Add a field named **ShipmentID** to your **Shipments** table and designate it as the primary key. It should be an AutoNumber data type.

Since each supplier shipment is likely to contain more than one item, you need a **ShipmentDetails** table.

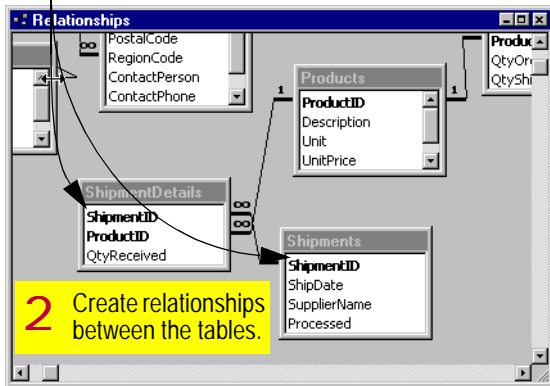
➔ Create a **ShipmentDetails** table. The foreign key corresponds to the **ShipmentID** field created above (recall [Section 5.4.2](#) when deciding on the correct data type for **ShipmentDetails.ShipmentID**).



- ➔ Create relationships between the new tables and existing tables, as shown in [Figure 20.2](#).

FIGURE 20.2: Relationships between the new shipment tables and the existing tables.

## 1 Create **Shipments** and **ShipmentDetails** tables



## 2 Create relationships between the tables.

- ➔ Ensure field properties such as **Caption** and **Default Value** are set appropriately for both tables.
- ➔ Populate the **suppliers** table you created in [Section 12.3.4](#) with a small number of records. You can either make up the supplier names or use the names shown on

the in the list of backorders and shipments (**BackShip.pdf**) in the project package.

### 20.3.2 Getting the right information for the shipment details subform

Your form for recording shipments is going to be very similar to the form shown in [Figure 14.1](#) for recording orders. To make the details of the shipment more readable, you will want to base the subform on a join query.

- ➔ Create a new query called **qryShipmentDetails** based on the **ShipmentDetails** and **Products** tables.
- ➔ Project the asterisk from the **ShipmentDetails** table and the **Description** field from the **Products** table.
- ➔ Save and close the query.

### 20.3.3 Creating a form for recording shipments

- ➔ Use the form wizard to create a new columnar form based on the **Shipments** table and save the form as **frmShipments**. If you are having difficulty remembering how to create a main form, review [Section 14.3.1](#).



- ➔ Create a new tabular form based on your `qryShipmentDetails` query and save the form as `sfrmShipmentDetails`. If you are having difficulty remembering how to create a subform, review [Section 14.3.2](#).



Since the form is based on a query, you should bring up the properties sheet for the form and ensure that the **RecordSource** property is `qryShipmentDetails`, not an *ad hoc* SQL statement.

- ➔ Replace the **SupplierID** textbox on the `frmShipments` form with a bound combo box. The combo box should only show the suppliers' names (review [Section 15.3.2](#) as required).
- ➔ Replace the **ProductID** textbox on the `sfrmShipmentDetails` form with a bound combo box. The combo box should show the **ProductID** and **ProductName** fields.
- ➔ Drag the subform onto the main form (review [Section 14.3.3](#) as required).



Ensure the master and child link properties for the subform control are set properly. Otherwise, the form and subform will not be synchronized.

### 20.3.4 Processing the shipment

Recall the procedure implemented in [Lesson 19](#) for processing an order:

1. The user enters the order information and the order details from the faxed-in order.
2. When the user is satisfied that the order has been entered correctly, she presses a button on the order form to process the order.
3. Pressing the button raises an **On Click** event, which causes a few lines of VBA code to be executed.
4. The VBA code executes a parameter action query that decrements the inventory the appropriate amount for each item in the order.

The procedure for processing a shipment is identical, except that items are *added* to inventory.

- ➔ Create a parameter action query called `pqryProcessShipmentDetails` to add the amount stored in **QtyReceived** to **QtyOnHand** (review [Section 17.3.3](#) as required).



Remember to use the value of **ShipmentID** on the main shipment form as a parameter. If you forget to do this, all shipment details (including those



associated with other shipments) will be processed against the **Products** table.

- ➔ Open **frmShipments** in edit mode and turn the control wizard off (recall [Figure 17.9](#)). This permits you to create the VBA code for the button from scratch.



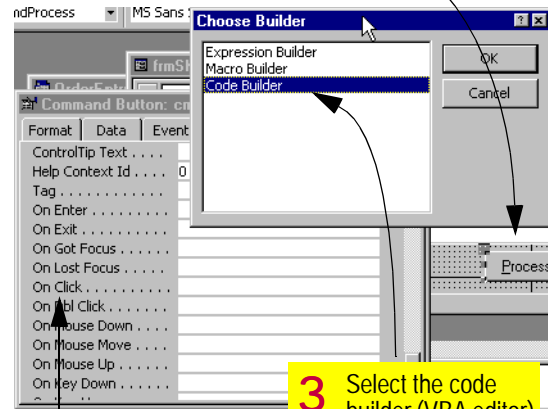
Alternatively, you can leave the control wizard on and press the **Cancel** button as soon as the wizard appears.

- ➔ Add a button to the main shipment form and change its **Name** property to **cmdProcess**.
- ➔ Bring up the properties sheet for the button, move to the **On Click** event, and invoke the VBA editor by selecting “event procedure”, as shown in [Figure 20.3](#).
- ➔ Enter the following code into the VBA editor (this is similar to the code in [Figure 19.9](#)):

```
NL If Me!Processed Then
NL     MsgBox "This shipment has already
NL         been processed"
NL Else
NL     DoCmd.SetWarnings False
NL     DoCmd.OpenQuery
NL         "pqryProcessShipmentDetails"
NL     MsgBox "The shipment has been
NL         processed"
```

FIGURE 20.3: Create an event handler for the **On Click** event of the new button.

- 1 Create a new command button named **cmdProcess**.



- 2 Find the **On Click** event for the button.

- 3 Select the code builder (VBA editor) to write an event handler.

```
NL DoCmd.SetWarnings True
NL Me!Processed = True
NL End If
```

- ➔ Close the module and test the update button. The final shipment form is shown in [Figure 20.4](#).



FIGURE 20.4: A form for recording supplier shipments.

The main form contains information about the shipment

The subform contains information about the products in the shipment

ProductID	Description	Quantity received
74 6308	Wok 14" steel with handles	24
82 25162	Nutmeg mill	24
88 3113196015	Salad plate, ocre	12
74 4539	Meat tenderizing hammer	12
74 4539	Meat tenderizing hammer	124
74 6083	Spring form pan, 9" non stick	48
74 6084	Spring form pan, 10" non stick	48
74 6102	Deluxe measuring spoon set	
74 6109	Colander, s.s., 5 qt.	
74 6191	Potato ricer, timed	
74 6245	Pastry blender	
74 6308	Wok 14" steel with handles	

The product description is shown to help the user recognize data entry errors.

Once the shipment has been entered, it is processed (added to inventory)

A combo box showing the product number and the product description makes it easy to enter items in the shipment.



Remember, you can use your rollback feature to return the **QtyOnHand** values for all products to their original values.

- the expected lead time from your suppliers (i.e., how long after placing an order do you expect to receive the shipment)
- the fixed cost of placing an order
- the cost of holding inventory
- the cost of stocking out

## 20.4 Discussion: Tracking versus optimizing

Determining how much to order (the reorder quantity) and when to order (the reorder point) requires additional information not explicitly stored in your database:

In addition, you need to be able to accurately forecast the demand for each product during its lead time. Although you do not store demand forecasts in your database explicitly, information from past orders (which are stored in the database) could be a starting point for



forecasting methods such as seasonally-adjusted moving averages, and so on.

In principal, the inventory levels for each product can be monitored and a supplier order can be automatically generated whenever the reorder points are reached. The mathematical techniques for determining the optimal inventory management policy can be found in any production management text and the data requirements are not particularly onerous. However, you are not expected to implement this functionality in this lesson.

### 20.5 Application to the assignment

Once the shipment form is working, you should take a few moments to refine it.

- ➔ Lock and disable the **Processed** check box.
- ➔ Lock and disable the product description in the subform.
- ➔ Ensure the tab order is correct for both the main form and the subform.