Teaching Case
Adapting the Access Northwind Database to Support a Database Course

John N. Dyer
Camille Rogers
Department of Information Systems
Georgia Southern University
Statesboro, Georgia 30460, USA
jdyer@GeorgiaSouthern.edu, cfrogers@GeorgiaSouthern.edu

ABSTRACT

A common problem encountered when teaching database courses is that few large illustrative databases exist to support teaching and learning. Most database textbooks have small 'toy' databases that are chapter objective specific, and thus do not support application over the complete domain of design, implementation and management concepts across a single database. The Northwind Traders sample database is available by Microsoft for download and use in Microsoft Access, and illustrates transactional processing for a fictitious company that imports (purchases) and exports (sells) specialty foods from around the world. The database contains sample tables, queries, forms, reports, Macros, VBA Class Objects, functions and modules, and other database features. Although the primary purpose for the database is to serve as an illustrative design template for students and practitioners, unfortunately the database and business processes are largely undocumented. This paper attempts to more completely document the business processes, including establishing business rules, describing relationships and participations, and discusses some problems with the existing design. Following understanding of the database and associated business processes, this paper can be used as both a teaching tool and a guide for practitioners using the Northwind Traders sample database as a design template. Additionally, it has been used successfully to introduce the concept of business processes and mapping them in an underlying database in introductory ERP courses.

Keywords: Northwind database, Database Design & development, Data modeling, Database management systems (DBMS)

1. INTRODUCTION

The pedagogical literature is very sparse in all regards to database design, implementation and management. In fact, since 2004, only eight articles have appeared in the *Journal of Information Systems Education* regarding database. Of these, two are teaching cases (Green, 2005; Irwin, Wessel and Blackburn, 2012), while six cover topics loosely related to teaching various database concepts (Itri, 2012; Casterella and Vijayasarathy, 2013; Unch, 2009; Carpenter, 2008; Olsen and Hauser, 2007; Hsiang-Jui and Hui-Lien, 2006).

First and foremost, the scope of this paper is to more completely document the Northwind Traders database, including business processes, establishing business rules, describing relationships and participations, and discussing some problems with the existing design, hence allowing use of the database to enhance teaching and learning. Although the Northwind database has advanced design features related to data macros, embedded macros, Access Class Objects, functions and VBA Modules, these components are well outside the scope of most introductory IS database course and will receive less discussion in this paper. A more advanced database course covering triggers, stored procedures, and procedural code may better facilitate exploration of these components within the Northwind database, following an understanding of the content of this paper, that is, documentation of the database. Furthermore, it is assumed that students already be presented with the basic concepts of database design, including tables, queries, forms, reports, primary/foreign keys, table relationships, etc., so that this paper can be used illustratively, either concurrent to the concepts being learned, or post-concept teaching to bring it all together in a complete database solution. It is hoped that this paper can be used as a teaching and learning tool, and as a guide for practitioners using the Northwind sample database as a design template.

Yue (2013) recently presented a thorough overview of a common problem in database courses: the unavailability of large sample databases for teaching and learning database concepts. He provided evidence that large illustrative databases are scarce, while those provided by textbook publishers are too small, overly simplified, contain only basic tables, and use multiple databases across examples and exercises. He further recommended the Sakila database as a

solution, which is a large sample database that installs with MySQL (Sakila, 2013; MySQL, 2013). He also echoed the sentiment that these textbook databases would "under no circumstance prepare the students for the true feel and experience they would need to cope with once they graduate and work in the real world" (Jukić and Gray, 2008).

On the other hand, large and complex databases can create challenges to teaching and learning because of the complexity, schema, constraints, and other barriers that must be overcome prior to illustrating database concepts, such as design, implementation, and management. Yue (2013) further related the reality that students must overcome a steep learning curve for any large database before being able to successfully complete examples and assignments. As such, this paper more fully documents the semi-realistic database: Microsoft Access's Northwind Traders database. The documentation in this paper includes describing the fictitious company's business processes, including describing the database objects, establishing business rules, defining the tables, establishing the entity representation model (ERM), further describing the table relationships and participations, and discussing some errors within the existing design.

2. THE ACCESS DBMS AND THE NORTHWIND DATABASE

Microsoft Access is a desktop relational database with many design features of enterprise level systems, but with limitations on the database size, the number of objects (tables, queries, forms, and reports), the number of fields and queries per table, the number of concurrent users, and lack of concurrency control (among other differences) (Access 2010 specifications, n.d.). Although Access is primarily a desktop database, it can also be deployed for 255 concurrent users over a small network, or even over the Internet. Many individuals and small businesses use Access to create personal information systems, transaction processing systems, and accounting information systems. Rice (2005) related that Access is ideal in three scenarios: for rapid application development (RAD) where development time and money are an issue, as a front-end to an enterprise database (SQL Server) to put a "friendly" face on the application, and as a data source for interoperating with other Office applications (Rice, 2005). In spite of many limitations of Access, Chung (2012) provides a comprehensive overview of the strength and capabilities of Access within organizations. Additionally, Access (with reference to the Northwind Traders database) is referenced in scores of software applications courses, is often illustrated in introductory and advanced database textbooks, and is used in web examples (w3schools).

Access installs with many predesigned templates for personal and business use. Of particular interest is the sample database named Northwind, which can be installed complete with tables (containing sample data), queries, forms, reports, macros, and VBA object classes, functions and modules. The Northwind database has shipped with Access since the earliest versions, with every new release of Access up to Access 2007 providing an updated version of Northwind, with the exception of Access 2010, 2013, and 2016. Although not installed with these versions, an option to download the database is provided when Access is opened. Installation instructions for Northwind 2010 to Access 2013 are available from Chapple (2012), while a limited overview of the database (including objects and object navigation) is available at Best STL (An Essential Guide to Using the Northwind Database in Access 2010, 2010). Additionally, the database can be downloaded for SQL Server 2000 from Microsoft (Northwind and Pubs Sample Databases for SQL Server 2000, 2010) and for SQL Server 2005 and 2008 from Codeplex (Northwind Database, 2011). For those unfamiliar with Access, a great tutorial resource has been provided by GCF Global (Free Access 2013 Tutorial at GCFLearnFree, 2015). Although the Northwind database has existed for almost two decades, the versions have changed significantly from the initial release to the current version, and there has been little to no documentation ever provided by Microsoft or other sources to document the database in a way that is ready useful as a template, or as an effective teaching and learning tool.

The Northwind database is illustrative of a merchandising company that buys and sells products. Northwind provides a model on which to base tables, relationships, queries, forms, and reports for one's own database, and illustrates relational database concepts such as relationships, interoperability with other Office applications, table/query/form/report construction techniques, normalization, and VBA and data access and manipulation, etc. (Rice, 2005). Unfortunately, Microsoft provides very little documentation on the Northwind databases or business processes regarding the business rules, metadata, the objects (tables, queries, forms, reports, macros, and modules), table attributes, or relationships. With 20 tables, 22 table relationships, 27 queries, 34 forms, 15 reports, 2 macros, scores of embedded macros, 10 Access Object Classes, and 9 VBA modules, it can be overwhelming to decipher the undocumented process, structure, and design. Although Northwind is robust and rich in content, it can be difficult to illustrate due to the lack of even minimal documentation.

3. NORTHWIND OVERVIEW

The Northwind sample database is a simple transaction processing database, illustrating the recording, storing, retrieving, and editing of data related to only some of the procurement and fulfillment activities of a merchandising company. For a typical merchandising company, products the company sells are purchased from vendors, held in inventory, and sold to customers. The sequences of buying and selling activities are known as *business processes*, which can be broken down into the *procurement process* and the *fulfillment process*, as shown in Figures 3.1 and 3.2.

Figure 3.1. Procurement Process

Figure 3.2. Fulfillment Process

Procurement activities relate to acquiring inventory while fulfillment activities relate to selling inventory, including recording merchandise movement as well as financial transactions. It should be noted that the current Northwind database is not configured to record financial transactions because it does not provide the design or structure required to fulfill most financial or managerial accounting requirements, e.g., tables to record financial transactions, queries determining accounts payable/receivable, expenses, etc., or reports for income statements and balance sheets, etc.

The basic Northwind business processes include employees purchasing products from suppliers, placing them in inventory, and reselling to customers. To purchase products from suppliers, an employee submits a purchase order to the supplier. Following receipt of the ordered products, the products are added to inventory. To sell products to customers, an employee creates a customer order. If the products are not in inventory, an employee can initiate a purchase order to fill the customer order. Once the customer order is filled, the order is invoiced and shipped. The shipped products are removed from inventory. Throughout the remainder of the paper it is suggested that the reader open and view the Northwind database in Access.

4. NORTHWIND TABLES

The various tables in Northwind can be described as *Master*, *Supporting*, and *Transaction*. Master tables are those that record data that does not change frequently, like employees, vendors, products, customers, and shippers. By "does not change frequently" means that while new records may be appended to master tables, the existing records are seldom edited. A typical new record many include a new employee, supplier, or product, etc. A typical record edit might include editing an employee's address or a supplier's contact person. Supporting tables, also known as organizational tables, are those that typically support master or transaction tables, and like master tables, the data changes infrequently. Examples include lookup table links to master tables, like an employee's department, title, state of residence, or a customer's tax status or tax rate.

On the other hand, transaction tables tend to involve frequent adding of new process transaction records, like recording procurement and fulfillment transactions related to movement of inventory and money. Examples include creating new purchase orders, recording new inventory, or posting customer payments. Transaction tables can also often experience editing of existing records, like changing an invoice payment status field from unpaid to paid. Often, a master table will also be a supporting table for a transaction table, which is a typical one-to-many (1:M) relationship between the master and transaction table. For example, a customer ID from the master table **CUSTOMERS** will be linked to the **ORDERS** transaction table to record the customer for which the order was placed.

Figure 4.1 reflects the complete entity relationship model (ERM) for the Northwinds database, including primary keys, while figures 4.2 and 4.3 reflect the entity relation diagrams (ERDs) for both the complete procurement and fulfillment processes, respectively. Note that Figure 4.1 omits two tables (**EMPLOYEE PRIVILEGES** and **PRIVILEGES**) for the sake of space constraints, while the ERD relating the two omitted tables to the **EMPLOYEES** table is shown in Figure 5.1. The ERM shown below has been modified for clarity by the authors (only in appearance) since the default ERM in Access is a near indecipherable entanglement of tables and relationships. Note also that, inasmuch as possible, the modified ERM places tables on the "1" side of a 1:M relationship on the left, and tables on the "Many" side of the relationship on the right, with the exception of the **PRODUCTS** table (on the far right).

The ERDs are more readily representative of the two primary processes without entanglement of the entire ERM. Even Best STL (An Essential Guide to Using the Northwind Database in Access 2010, 2010) related that the ERM "may give the impression of a tangle of tables and links resembling spaghetti," hence the ERM is broken into separate ERDs throughout this paper. Appendix A, Tables 1 and 2, describe the database tables in the Northwind database. Table 1 lists each database table, the primary key(s), as well as a table's relationship with other tables, like 1:1 or 1:M. Table 2 summarizes master, transaction, and supporting database tables.

Since purchasing and selling products are the company's primary activities, six very important transaction tables follow. Note that table names are formatted as UPPER letter-case and **bold** font-style, while attribute/field names are formatted in *italics*. The six transaction tables are then **PRODUCTS**, **PURCHASE ORDERS**, **PURCHASE ORDER DETAILS**, **ORDERS, ORDER DETAILS**, and **INVENTORY TRANSACTIONS**. The details for each product that can be purchased and sold are reflected in **PRODUCTS**. When the company purchases products, the purchase summary is recorded in **PURCHASE ORDERS** while the items ordered are recorded in **PURCHASE ORDER DETAILS**. When products are sold, the sales summary is recorded in **ORDERS** while the items ordered are recorded in **ORDER DETAILS**. When inventory is received or shipped, the transaction is recorded in

INVENTORY TRANSACTIONS. It should be noted that there is no direct end-user interaction with the tables, but instead, forms are used to add, update and delete records in underlying tables, via direct link to the tables or through multi-table queries.

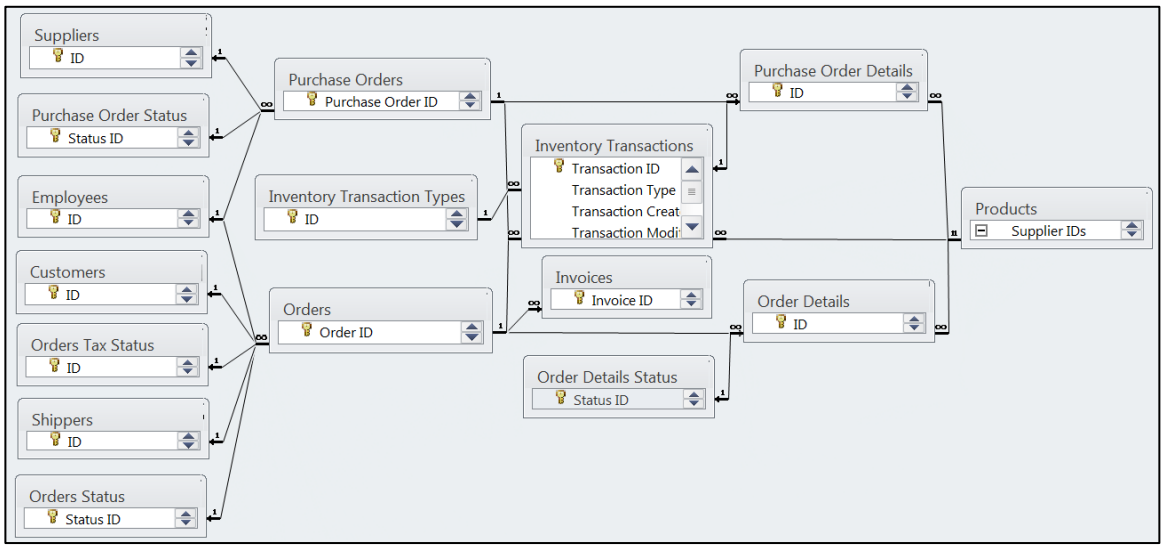

Figure 4.1. Northwind ERM

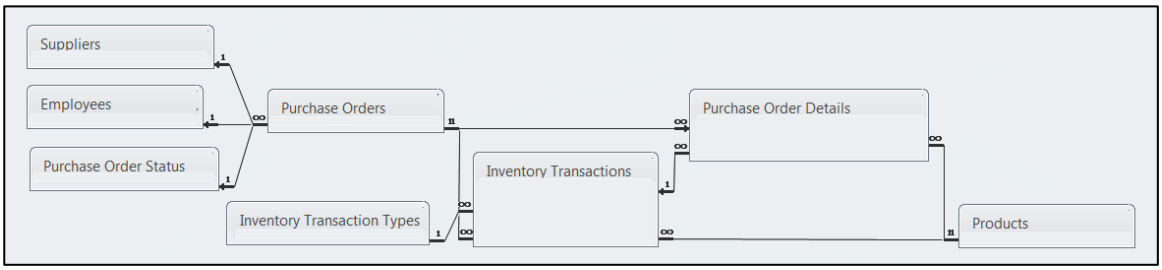

Figure 4.2. Procurement ERD

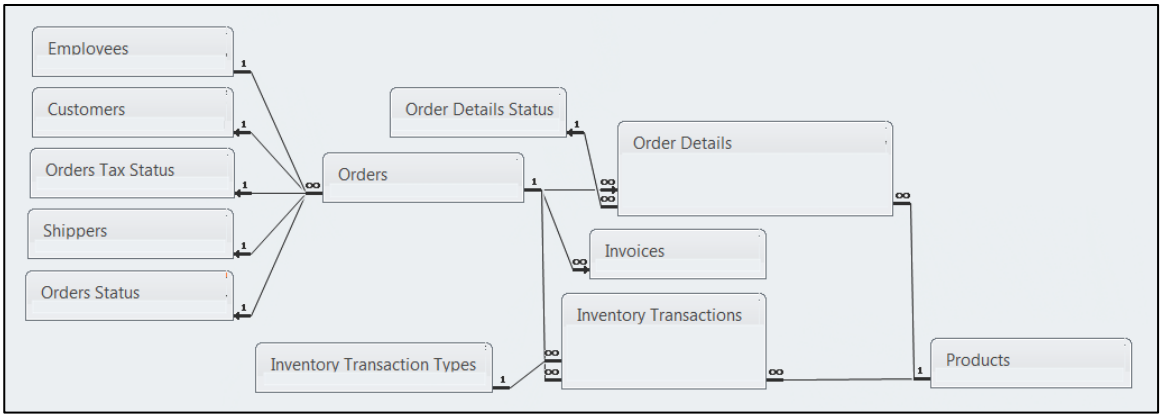

Figure 4.3. Fulfillment ERD

The following section describes the tables and relationships for the procurement process (Section 5) and the fulfillment (Section 6). Appendix A, Table 3, enumerates and describes the business rules, including relationships (1:1, 1:M, M:N) and participations; optional versus mandatory. Appendix A, Table 4, provides relevant notes where indicated in Table 3. Figures are also provided depicting the appropriate ERD connectivities for each business rule.

5. PROCUREMENT TABLES

5.1 Employee Privileges

EMPLOYEE PRIVILEGES is a bridge/composite entity recording an employee with a specific privilege, hence in the table an employee may have many privileges, and a privilege may be granted to many employees, but each employee-privilege composite will be a separate record in the table. Figure 5.1 reflects the ERD for business rule 1.

5.2 Purchase Orders

PURCHASE ORDERS reflects a summary of each purchase order, but not the products purchased. Instead, the details of each product on a single purchase order are recorded in one or more records in **PURCHASE ORDER DETAILS**. A single purchase order initiated by an employee may generate an order for many different products from a single supplier, with each product recorded as a separate record in **PURCHASE ORDER DETAILS**. Figure 5.2 reflects the ERD for business rules 2, 3 and 4.

5.3 Purchase Orders Details

PURCHASE ORDER DETAILS records every product ordered on every purchase order. For example, if a single purchase order has 3 products, **PURCHASE ORDER DETAILS** will have 3 records; one for each product ordered on the single purchase order. Figure 5.3 reflects the ERD for business rules 5 and 6.

5.4 Inventory Transactions (for purchase orders)

INVENTORY TRANSACTIONS records all products purchased and sold, as well as products on-hold to fill customer backorders, and products that are not available for sale due to being waste. When the products from a single purchase order are received into inventory, each individual product is recorded in **INVENTORY TRANSACTIONS**, and is assigned a *Transaction ID*, the *Transaction Type* ('Purchased') from **INVENTORY TRANSACTION TYPE**, and a *Product ID* from **PRODUCTS**. Note that in the illustrative table the *Purchase Order ID* attribute is blank for all records, but new purchase orders entered using the appropriate form will record the *Purchase Order ID*. This is not necessarily an error, but likely due to Microsoft simply not including the data when populating the table, as is the case in several other tables. Figure 5.4.1 reflects the ERM for business rules 7, 8 and 9, while Figure 5.4.2 reflects the ERD for business rule 10.

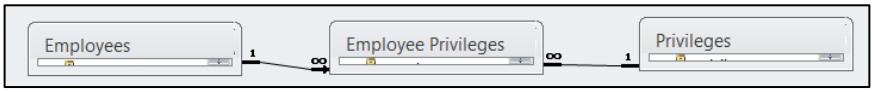

Figure 5.1. Employee Privileges

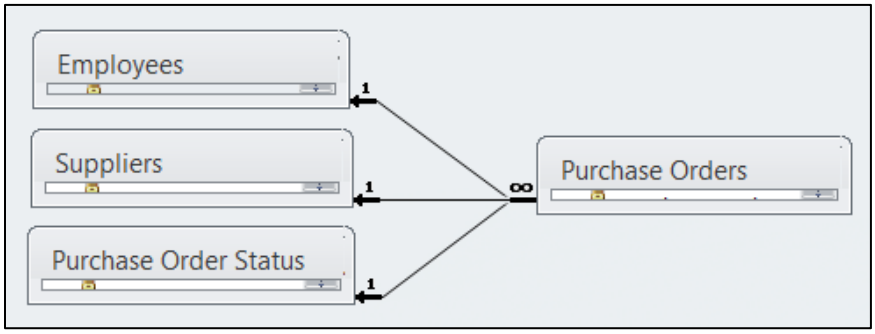

Figure 5.2. Purchase Orders

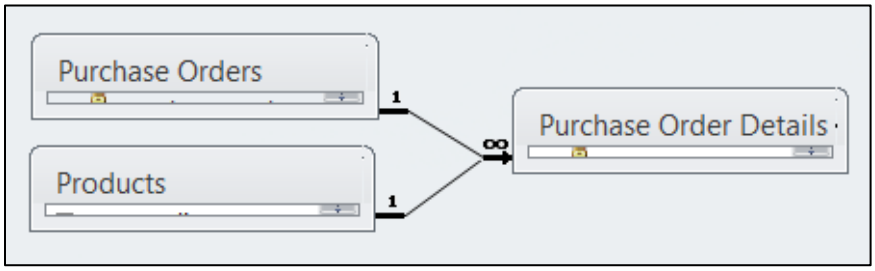

Figure 5.3. Purchase Order Details

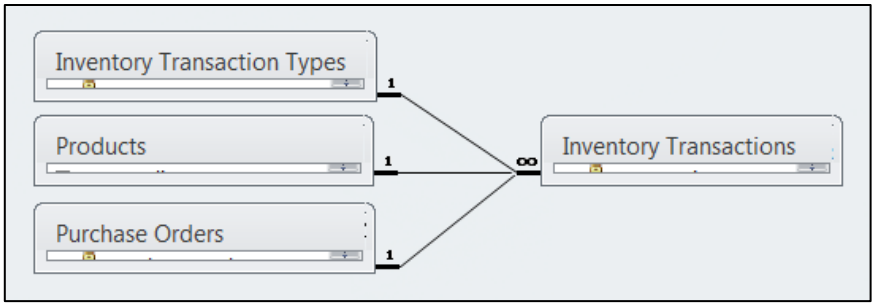

Figure 5.4.1. Inventory Transactions

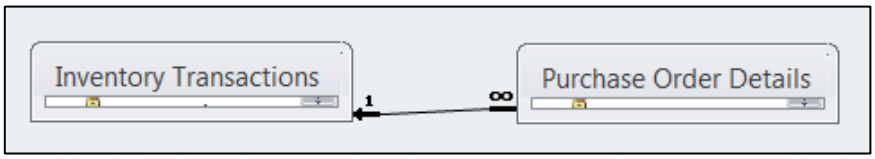

Figure 5.4.2. Purchase Order Detail

6. FULFILLMENT TABLES

6.1 Orders

ORDERS reflects a summary of a customer order, but not the products ordered. Instead, the details of each product on a single customer order are reflected in one or more records in **ORDER DETAILS**. A single customer order.

6.2 Order Details

ORDER DETAILS reflects every product ordered on every customer order. For example, if a single customer order has three products, **ORDER DETAILS** will have 3 records; one for each product ordered on the single customer order. Figure 6.2 reflects the ERD for business rules 16, 17 and 18.

6.3 Invoices

INVOICES reflects every customer order that has been invoiced. The table records only those orders that have been invoiced (using the appropriate form). **INVOICES** will record an *Invoice ID* and *Invoice Date*, and the related *Order ID* from **ORDERS**. Note that **INVOICES** has additional fields for *Due Date*, *Tax*, *Shipping* (fee) and *Amount Due*, but all fields are null in the table and are never updated with any form or query. Note, one would assume that an invoice is similar to a packing slip and that it would only be generated prior to order shipping. Yet, there are several orders in **ORDERS** that have shipped but are not recorded in **INVOICES**, and there are several unpaid and unshipped orders that do have an invoice. This is likely another error in populating the database with illustrative data. Figure 6.3 reflects the ERD for business rule 19.

6.4 Inventory Transactions (for orders)

For customer orders, **INVENTORY TRANSACTIONS** records all products 'sold,' as well as products 'on-hold' to fill customer backorders. When a customer order is filled, each product on the order is removed from inventory, and is assigned a *Transaction ID* and *Transaction Type* ('Sold'). The *Transaction ID* is also recorded in **ORDER DETAILS**. Figure 6.4 reflects the ERD for business rules 20, 21 and 22.

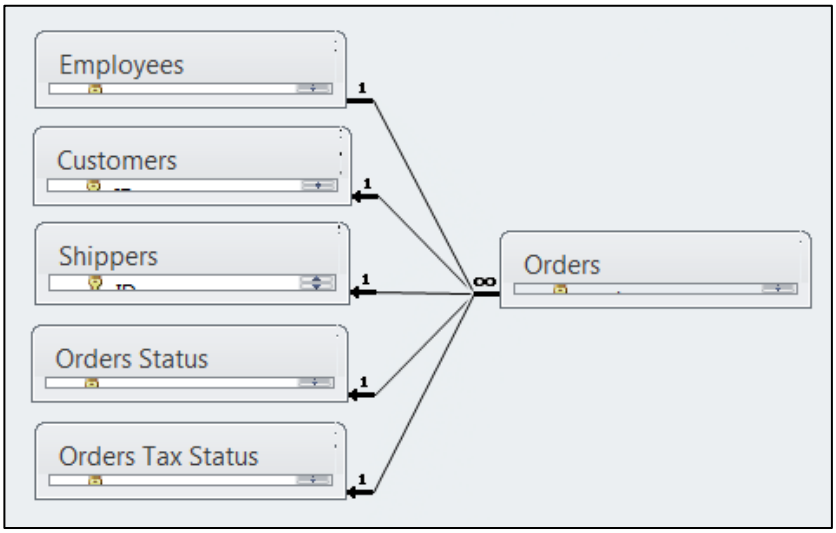

Figure 6.1. Orders

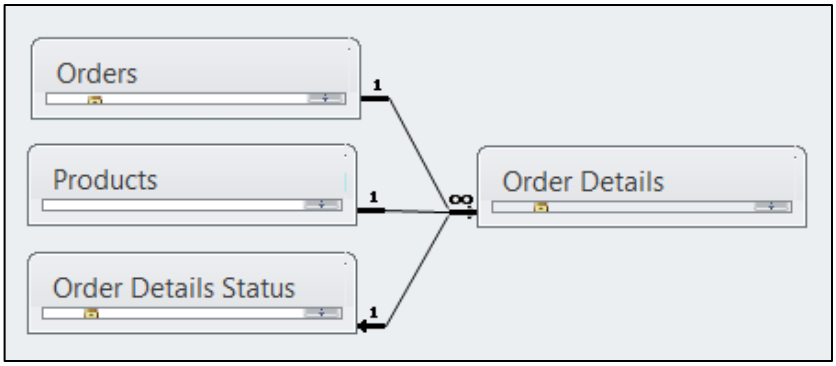

Figure 6.2. Order Details

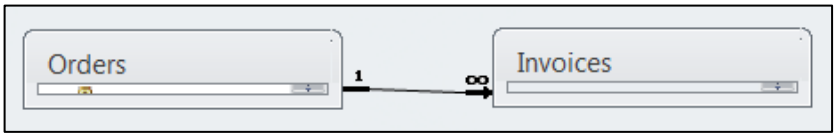

Figure 6.3. Invoices

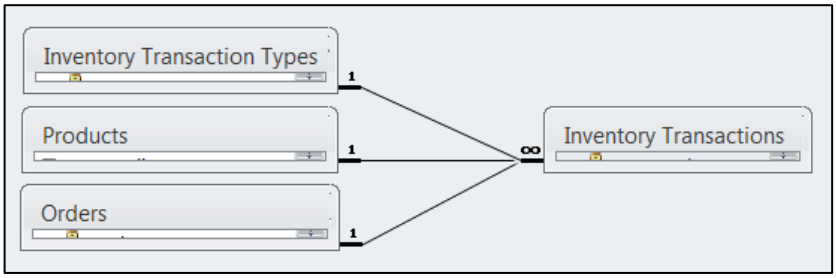

Figure 6.4. Inventory Transactions

7. NAVIGATING NORTHWIND

A ***Startup Screen*** form is loaded when the database is opened, prompting the user to click the Security Warning ***Enable Content*** button. The ***Login Dialog*** form (Figure 7.1) appears after clicking the ***Enable Content*** button. The form requires selecting an employee name from the combo box (drop-down list) and clicking the ***Login*** button. It is important to note that after pressing the button an embedded macro runs that captures the selected employee's ID and privileges and sets it to a temporary variable which is used throughout the processes to allow privileges and to auto-fill form fields with the employee data. Note that the employee named 'Andrew Cencini' has purchase approval privileges according to **EMPLOYEE PRIVILEDGES**, but none of the other eight employees have been assigned privileges, perhaps a design error. The records in **EMPLOYEE** reflect that all employees are associated with sales. Following login, another embedded macro opens the ***Home*** form (Figure 7.2) providing access to objects such as hyperlinks to create a *New Customer Order* or *New Purchase Order*, *Quick Links* to tables, queries, forms and reports, and embedded query results within subforms, such as ***Active Orders*** and ***Inventory to Reorder***. The object navigator is also available in the left pane, providing access to all database objects, although it is collapsed when the database opens.

Figure 7. 1. Login Dialog Form

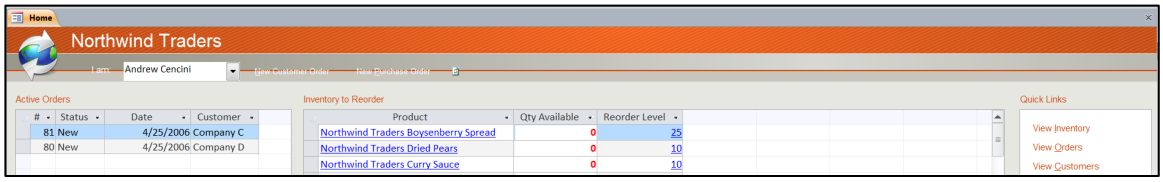

Figure 7.2. Home Form

The following sections will postulate assumed business processes for the Northwind procurement and fulfillment processes, based on the previously established business rules, relationships and participations. It is recommended that the reader become familiar with each object (tables, queries, forms, reports, macros, modules) involved in every process component, as well as the object relationships and dependencies. All business processes are completed using forms and subforms, which depend on underlying tables, queries, embedded macros and VBA modules. The user seldom interacts directly with objects other than forms and reports. Instead, form data is read from and written to tables and/or queries, hence updating related and dependent objects. Additionally, embedded macros and VBA subroutines are directly related to many form actions, e.g., assigning an employee ID (based on login) to forms, or triggering a purchase requisition creation/submission when a customer order for a product exceeds inventory quantity.

Access provides a tool for examining how objects interact with (or depend on) other objects. The tool is available in the menu tab 'Database Tools' inside the 'Relationships' group, by clicking on 'Object Dependencies.' Becoming familiar with database objects can make it easier to perform a wide variety of tasks, such as entering data into a form, adding or removing tables, finding and replacing data, and running queries.

To better understand object dependencies the interested reader should see the reference, Learn the Structure of an Access Database (2007). For example, by clicking on **INVOICES** and using the 'Object Dependencies' tool, one can determine that **INVOICES** interacts with (and depends on) data from **ORDERS**, while the ***SALES ANALYSIS*** query depends on **INVOICES**. Similarly, the ***Purchase Orders Detail*** form depends on tables **PURCHASE ORDER STATUS** and **PURCHASE ORDERS**, queries ***EMPLOYEES EXTENDED***, ***PURCHASE DETAILS EXTENDED***, and ***SUPPLIERS EXTENDED***, and forms ***Employee Details***, ***Purchase Subform for Purchase Order Details***, ***Receiving Subform for Purchase Order Details***, and ***Supplier Details***. The objects that depend on the ***Purchase Orders Detail*** form are the forms ***Home*** and ***Purchase Order List***. Unfortunately, the tool does not show dependencies on macros or VBA modules. Additionally, one can use the built-in documenter tool from the menu tab 'Database Tools' inside the 'Relationships' group, by clicking on 'Database Documenter.' This will provide the description and schema of each object in the database. The reader is encouraged to examine object dependencies for all Northwind objects.

8. PROCUREMENT PROCESS NAVIGATION

Although the actual Northwind procurement process is undocumented, the assumed process is described in the following procurement Steps, 8.1 to 8.7 (below). Note that form and subform names, as well as form tabs, buttons and links, are Proper letter-case, and **bold** and *italicized* font style. Attribute/field names are *italicized*, field data values are surrounded in 'single quotes,' table names are as aforementioned, while query names are UPPER letter-case, and **bold** and *italicized* font-style. An overview of the Northwind procurement process steps is shown in Figure 8.1 below.

8.1 Trigger

A purchase order requisition is triggered by a reorder level requirement or insufficient inventory (view the table **PRODUCTS**, and the two forms ***Inventory List*** and ***Inventory to Reorder Subform for Home***). This might occur after filling a customer order that depleted current inventory below the reorder level, or a new customer order for which current inventory is insufficient. Following login, the ***Home*** form ***Inventory to Reorder Subform for Home*** subform displays query results showing which products have been triggered to be reordered. Clicking on a hyperlink in the subform opens the ***Product Details*** form to determine the appropriate supplier and reorder quantity. Note that most of the product's attribute values can be updated directly in the form.

Additionally, the ***Order/Purchase History*** tab reflects the sales and purchases transactions of the product using the subform ***Product Transactions Subform for Product Details***. One can also view other products by clicking the ***Go to Product*** drop-down, as well as save changes and create a new product using the ***Save and New*** form hyperlink. The form does not show the actual quantity that needs to be ordered. Instead, the ***Inventory List*** form shows the exact quantity to order for every product. The ***Inventory List*** form is available in the left navigation pane.

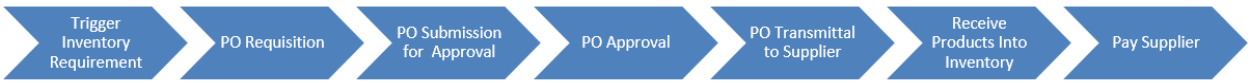

Figure 8.1 Procurement Process Steps

8.2 Purchase Order Requisition

Using the ***Purchase Order Details*** form, an employee creates a new purchase order requisition. This form is available from the ***Home*** form hyperlink named ***New Purchase Order***. For existing orders that have been created, submitted or approved, the Form Header section *Status* field is populated from an embedded 'Select' query set as the Record Source (see the form Property Sheet). For new orders, the *Created By* employee ID field is populated using the default value of the employee ID (set on login), while the *Creation Date* field is a default value using a built-in Now() function. The form requires selecting a *Supplier* from the drop-down list, populated from the ***SUPPLIERS EXTENDED*** query. After a supplier is selected and products are added, a new *Purchase Order ID* (*AutoNumber*) is created in **PURCHASE ORDERS**. The form data are inserted into **PURCHASE ORDERS**, including *Supplier ID*, *Created By* employee ID, *Creation Date*, and a *Status ID* set to 'New' (which is the default value).

After selecting the subform tab ***Purchase Details*** (***Purchase Subform for Purchase Order Details*** subform), the subform requires selecting a *Product* from a drop-down list, populated from **PRODUCTS**, and entering an order quantity (*Qty*). The subform creates a new record in **PURCHASE ORDER DETAILS** with in a new *ID* (*AutoNumber*) and inserts the corresponding *Purchase Order ID* and the *Product ID* into **PURCHASE ORDER DETAILS**. Note that ***PURCHASE DETAILS EXTENDED*** is based off of **PURCHASE ORDER DETAILS** and **PURCHASE ORDERS**, and the subform fields *Unit Cost* and *Total Price* are populated from calculated fields in ***PURCHASE DETAILS EXTENDED***. Additionally, the *Unit Cost* value can be changed to reflect a *Standard Cost* that is different from the value stored in **PRODUCTS**. It should be noted that once a product is selected, the *Product* can be changed, as well as the *Qty* and *Unit Cost*, but the product cannot be deleted. Instead, the purchase order can be canceled after submitting for approval (Step 8.3).

For each product selected in the subform, the ***INVENTORY ON ORDER*** (query fields (*Product* and/or *Quantity on Order*) are updated, although the purchase order has not yet been submitted or approved. The ***INVENTORY*** field *Qty on Order* is also updated, hence affecting the *Qty Below Target*, *Current Level*, and *Qty To Reorder* fields. The form can be closed at this point, or immediately submitted for approval. Before submitting for approval, either or both the *Product* and *Qty* can be changed, and the fields are correctly updated in the related tables and queries.

Once the requisition is created, closed and reopened, the existing form fields are populated from **PURCHASE ORDERS**, while the subform is populated from ***PURCHASE DETAILS EXTENDED.*** Neither the form nor subform show the total price for the purchase order, but ***PURCHASE PRICE TOTALS*** (based on ***PURCHASE DETAILS EXTENDED***) sums the individual extended prices (*Price Total*) grouped by *Purchase Order ID*. As previously mentioned, prior to submission for approval, the existing *Product*(s), *Qty*, and *Unit Cost* fields can be changed, and/or new products added. The supplier can also be changed, but doing so deletes the existing products. Note that the form allows an error of selecting products not available from the specified supplier, and allows entering negative-integer and non-integer quantities, e.g., quantity of -1 or 3.245. Furthermore, **PURCHASE ORDERS** has fields for *Shipping Fee* and *Taxes*, but due to apparent design errors these fields are not available on the ***Purchase Details*** subform, and all values in the tables are '$0.00.' The other query affected by this step is ***PURCHASE SUMMARY***, which summaries the *Order Total* for each *Purchase Order ID* (based on **PURCHASE ORDERS** and ***PURCHASE PRICE TOTALS***).

8.3 Purchase Order Requisition Submission for Approval

Using the ***Purchase Order Details*** form an employee can click the ***Submit for Approval*** form hyperlink, and in turn the *Submitted By* employee ID and *Submitted Date* fields are populated based on the same method in Step 8.2. The form data are inserted into **PURCHASE ORDERS** (and updated in ***PURCHASE DETAILS EXTENDED***), including the *Submitted By* employee ID, the *Submitted Date*, and with the *Status ID* set to 'Submitted.' Prior to submitting for approval, the existing product details can be changed and/or new products added. Following submission, the ***Submit for Approval*** form hyperlink is disabled. Alternatively, instead of clicking ***Submit for Approval***, the employee can click the ***Cancel Purchase*** form hyperlink which will permanently delete the purchase order and details from **PURCHASE ORDERS** and **PURCHASE ORDER DETAILS**. The other query affected by this step is ***PURCHASE SUMMARY***, wherein the *Submitted By* and *Submitted Date* fields are updated.

8.4 Purchase Order Approval

Using the ***Purchase Order Details*** form an employee with approval privileges approves or cancels the requisition using the ***Approve Purchase*** or ***Cancel Purchase*** form hyperlinks. If approved, the purchase order requisition becomes a purchase order. The *Approved By* and *Approved Date* fields are populated based on the same method in Step 8.2. The form data are inserted into **PURCHASE ORDERS** (and updated in ***PURCHASE DETAILS EXTENDED***), including the *Approved By* employee ID, the *Approved Date*, and the *Status ID* set to 'Approved.' Prior to submitting for approval, the existing product details can be changed and/or new products added. Following approval, the ***Approve Purchase*** form hyperlink is disabled, and product details cannot be changed. Due to an error in design, the supplier can be changed, but it deletes all product details, and allows only one product line.

Alternatively, instead of clicking ***Submit for Approval***, the employee can click the ***Cancel Purchase*** form hyperlink which will permanently delete the purchase order and details as described in Step 8.3. Note that the *Expected Date* field is optional in all steps. The other query affected by this step is ***PURCHASE SUMMARY***, wherein the *Approved By* and *Approved Date* fields are updated.

8.5 Send Purchase Order to Supplier

The purchase order is assumed to be sent to the supplier via the web, email, fax, phone, etc.

8.6 Receive Products from Supplier

Using the ***Purchase Order Details*** form an employee will receive inventory from the supplier. After clicking on the subform tab ***Inventory Receiving*** (***Receiving Subform for Purchase Order Details*** subform), the subform data are populated from ***PURCHASE DETAILS EXTENDED***, and the subform requires selecting a *Date Received* from the integrated form calendar and clicking the *Add to Inventory* check box. The form data are inserted into **PURCHASE ORDER DETAILS** (and updated in ***PURCHASE DETAILS EXTENDED***). Additionally, checking *Add to Inventory* creates a new record in **INVENTORY TRANSACTIONS** with a *Transaction ID* (*AutoNumber*), *Transaction Type* set to 'Purchased,' the *Transaction Created* date, the *Transaction Modified Date*, the *Product ID*, the *Quantity*, and the *Purchase Order ID*. Other queries affected by this step include ***INVENTORY***, ***INVENTORY ON ORDER***, ***PURCHASE SUMMARY***, ***INVENTORY PURCHASED***, and ***PRODUCT PURCHASES***.

8.7 Pay Supplier

Using the ***Purchase Order Details*** form an employee will pay the supplier for the purchase order. After clicking on the subform tab ***Payment Information***, the subform requires selecting *Payment Type* from a drop-down list, and selecting a *Payment Date* from the integrated calendar. The *Notes* field is optional and should not be used. The form data are inserted in **PURCHASE ORDERS**. Partial payments are not allowed. Note that **PURCHASE ORDERS** has a field for *Payment Amount*, but because of a design error this field is not available on the ***Payment Information*** subform, and all values are '$0.00' in **PURCHASE ORDERS** (even after payment is posted). Additionally, for existing purchase orders that have been paid, the subform fields are populated from **PURCHASE ORDERS**.

9. FULFILLMENT PROCESS NAVIGATION

As with the procurement process, the actual Northwind fulfillment process is undocumented, and the assumed process is described in the following procurement steps. An overview of the Northwind fulfillment process steps is shown in Figure 9.1 below.

9.1 Receive Customer Order

Using the ***Order Details*** form, an employee creates a new customer order. This form is available from the ***Home*** form hyperlink ***New Customer Order***. The *Salesperson* employee ID field is populated using the default value of the employee ID (set on login), while the *Order Date* field is a default value using the built-in Now() function. The form requires selecting a *Customer* from the drop-down list which is populated from the ***CUSTOMERS EXTENDED*** query, based on **CUSTOMERS**. Selecting a customer also results in customer's e-mail address being populated by ***CUSTOMERS EXTENDED***. Note that there are no actual E-mail addresses in **CUSTOMERS**.

After a customer is selected, a new *Order ID* (*AutoNumber*) is created in **ORDERS**. The form data are inserted into **ORDERS,** including the *Salesperson* employee ID, *Customer* customer ID, and *Order Date* from the form data, and a *Status ID* set to 'New.' Additionally, the fields in the ***Shipping Information*** subform tab (starting with *Ship Name*) are populated from **CUSTOMERS**. The form can be closed without adding products. Additionally, the employee can click the ***Delete Order*** form hyperlink, which in turn deletes the record from **ORDERS**.

After selecting the subform tab titled ***Order Details*** (***Order Subform for Order Details*** subform), the subform requires selecting *Product(s)* from a drop-down list, and entering the order quantity (*Qty*) and any *Discount* allowed. The product drop-down shows the description and quantity available, populated from a 'select query' of *Product* and *Quantity Available* from **INVENTORY**. Note that the subform allows an error of entering negative-integer and non-integer quantities, e.g., quantity of -1 or 3.245. Furthermore, **ORDERS** has a field for *Taxes*, but by design error the field is not available in the ***Order Details*** subform, and all values in **ORDERS** are '$0.00.' Other queries affected by this step include ***INVENTORY*** and ***INVENTORY ON HOLD***.

The product quantity ordered may or may not have sufficient inventory to fill the order, but only complete orders can be invoiced and shipped. The two cases of sufficient verses insufficient inventory are described in subsections 9.1.a and 9.1.b below.

9.1.1 Product quantity is sufficient to fill the order: If quantity of a product in inventory is sufficient, a new *Transaction ID* is created in **INVENTORY TRANSACTIONS**, setting the *Transaction Type* to 'On Hold', inserting the *Transaction Created* date, *Transaction Modified Date*, *Product ID*, *Quantity*, and *Customer Order ID*. Additionally, a new *ID* is created in the **ORDER DETAILS**, inserting the customer *Order ID*, *Product*, *Quantity*, *Unit Price*, *Discount*, *Inventory ID*, and setting the *Status ID* to 'Allocated.' **INVENTORY TRANSACTIONS** is updated with a new *Transaction ID*, *Transaction Type* set to 'On Hold', the *Transaction Created* date, the *Transaction Modified Date*, the *Product ID*, the *Quantity*, and the *Customer Order ID*. Additionally, a product can be deleted from the subform tab by highlighting the product and pressing the 'Delete' key.

Once a product and quantity are selected, and the form closed and reopened, the quantity value cannot be changed. An error also exists in the form as follows. If the product is deleted after reopening the form, the record is deleted in **ORDER DETAILS**, but the record in **INVENTORY TRANSACTIONS** is not deleted, hence the affected quantities in the related queries (***INVENTORY*** and ***INVENTORY ON HOLD***) are not updated to their original values. As a result, a record still exists in **INVENTORY TRANSACTIONS** for a product no longer in **ORDER DETAILS**. Consequently, the ***INVENTORY*** fields *Qty Purchased*, *Qty On Hand*, *Qty Available*, *Qty On Hold*, and *Current Level* are understated, while *Qty On Hold*, *Qty Below Target Level*, and *Qty To Reorder* are overstated.

9.1.2 Product quantity is insufficient to fill the order: If quantity of a product in inventory is insufficient, a message box prompts the employee to create a purchase order. If the employee clicks 'No,' **ORDER DETAILS** is affected as in Step 1.a above, but the *Status ID* is set to 'No Stock' and the

Figure 9.1 Fulfillment Process Steps

related *Purchase Order ID* is recorded in **ORDER DETAILS**. If 'Yes' is clicked, a purchase requisition will be automatically created and submitted (Procurement Steps 8.1, 8.2 and 8.3), with the *Notes* field in **PURCHASE ORDERS** automatically set to 'Purchase generated based on Order #...'. Step 1.a is completed as shown above, but the *Status ID* is set to 'On-Order' and the related *Purchase Order ID* is recorded in **ORDER DETAILS**. The customer order cannot be invoiced and shipped until Procurement Steps 8.4, 8.5 and 8.6 have been completed.

Once Procurement Step 8.6 is completed, a new record is created in **INVENTORY TRANSACTIONS** with a new *Transaction ID*, *Transaction Type* set to 'Purchased,' the *Transaction Created* date, the *Transaction Modified Date*, the *Product ID*, the *Quantity*, and the *Purchase Order ID*. The employee is also prompted to fill the customer order. After filling the order, **INVENTORY TRANSACTIONS** is updated with a new *Transaction ID*, *Transaction Type* set to 'On Hold', the *Transaction Created* date, the *Transaction Modified Date*, the *Product ID*, the *Quantity*, and the *Comments* field set to 'Fill back ordered product, Order #.' Additionally, a new record is created in **ORDER DETAILS**, with a new *ID*, inserting the customer *Order ID*, *Product*, *Quantity*, *Unit Price*, *Discount*, *Purchase Order ID*, *Inventory ID*, and setting the *Status ID* to 'Allocated.'

Several problems exist with the form and data in this step. For example, if a product has non-zero quantity, but quantity is insufficient to fill the customer order, the quantity to order populated in the automatic purchase order is set to the total, not the required quantity (although the quantity can be changed in the purchase order). Additionally, if the related purchase order is cancelled, the **ORDER DETAILS** *Status* field remains set to 'On Order,' although it is no longer 'On Order.'

9.2 Prepare Shipping Information

Using the ***Order Details*** form, an employee chooses the subform tab ***Shipping Information***. The *Shipping Company* field is selected from a drop-down list, the *Ship Date* selected from the embedded calendar, and a *Shipping Fee* entered. **ORDERS** is automatically updated from the subform in the *Ship Via*, *Shipped Date*, and *Shipping Fee* fields. There are no changes to any other tables.

9.3 Create Invoice

Using the ***Order Details*** form, an employee clicks the ***Create Invoice*** form hyperlink. **INVENTORY TRANSACTIONS** is updated with *Transaction Type* set to 'Sold'. In **INVOICES**, a new record is created with an *Invoice ID* (*AutoNumber*), inserting the *Order ID* as the *Invoice #*, and the *Invoice Date* is set using the built-in Now() function. The *Invoice ID* field is hidden in the table. Note the *Shipping* fee and *Amount Due* fields are '$0.00.' In **ORDER DETAILS** the *Status ID* is set to 'Invoiced', and a report named 'Invoice' is created, detailing the order. There are no changes to any other tables. An error exists in this step, in that **ORDERS** *Status ID* remains set to 'New' although it should update to 'Invoiced.'

9.4 Ship Order

Using the ***Order Details*** form, an employee clicks the form hyperlink named ***Ship Order***. The **ORDERS** *Status ID* field is set to 'Shipped.' There are no changes to any other tables. Note another design error, that in **ORDER DETAILS** the *Status ID* remains set to 'Invoiced' although one would expect it to be updated to 'Shipped.'

9.5 Receive Payment

Using the ***Order Details*** form, an employee clicks the subform tab ***Payment Information***. The subform requires selecting a *Payment Type* from a drop-down list, and selecting a *Payment Date* from the integrated calendar. The *Payment/Order Notes* field is optional. The data is automatically updated in **ORDERS** in the *Payment Type* and *Paid Date* fields. There are no changes to any other tables.

9.6 Complete and Close Order

Using the ***Order Details*** form, an employee clicks the form hyperlink named ***Complete Order***. The **ORDERS** *Status ID* field is set to 'Closed.' There are no changes to any other tables.

10. SUPPORTING DATABASE OBJECTS

As previously related, forms are the primary object for navigating and executing components of the Northwind business processes. Forms are also used to add new records, as well as edit and delete existing records in underlying tables and queries. Many of the illustrative forms in the database are very sophisticated, including execution of embedded data macros and VBA modules to bind data from tables and queries, as well as add, edit, update and delete records in underlying tables and queries. Additionally, many embedded data macros and VBA modules are used to create, view and print reports from forms, send emails from forms, open forms from other forms, trigger new processes, etc.

The reports in the Northwind database are reasonably self-explanatory by name, and most have an underlying table or query as the source data. All of the reports can be opened directly from the navigation pane, with the exception of the ***Invoice*** report. Instead, the ***Order Details*** form allows the user to click a link labeled "Create Invoice," which in turn executes an Event Procedure bound to an Access Class Object named **Form_Order Details**. The result is the ***Invoice*** report that opens for viewing and printing. Unfortunately, an Invoice cannot be created once a customer order is closed.

Although a complete description and discussion of every database object in Northwind is outside of the scope of this paper, the reader is encouraged to further examine the embedded macros associated with all event procedures, as

well as the VBA Class Objects, modules, functions, subroutines and procedures. As of Access 2007, two types of macros are available: stand-alone and embedded macros. In general, a macro is a tool that allows automation tasks and add functionality to forms, reports, and controls. The interested reader is encouraged to reference sources regarding creating data macros and for using embedded macro and VBA modules (Introduction to Macros. (n.d.)), (Create a Data Macro, 2010), and (Introduction to Access Programming, 2010). A brief description of some of the important supporting objects is provided in the following subsections.

10.1 Additional Tables

SALES REPORTS: The table contains five records of SQL used to create five sales reports: by Category, Country, Customer, Employee, and Product. The SQL statements are used in the ***Sales Reports Dialog*** form, and after the user selects the sales report type and the period (Monthly, Quarterly, Yearly), the user clicks the link labeled "Preview" to display the report. A VBA module then creates and displays the report. Unfortunately, the form does not work properly in that the Year drop-down allows only years 2003–2005, for which there is no data.

STRINGS: The table contains 62 records of various messages displayed in Message Boxes that result from a user action. As with the previous table, these messages are queried in a VBA module and displayed in a message box object.

10.2 Additional Queries

EMPLOYEES EXTENDED: Based on **EMPLOYEES**, the query adds two calculated fields (File As and Employee Name) concatenating employee first and last name for use in updating various tables, queries, forms and reports (see Object Dependencies).

CUSTOMERS EXTENDED: Based on **CUSTOMER**, the query adds two calculated fields (File As and Contact Name) concatenating first and last name for use in updating various tables, queries, forms and reports (see Object Dependencies).

10.3 Forms

As previously noted, forms are the primary means of navigating the Northwind's transactional processes, but other forms (not enumerated below) are used to add and edit data regarding shippers and suppliers, including ***Shipper Details***, ***Shipper List***, ***Supplier Details***, and ***Supplier List***. The following are the primary forms used in the database.

Startup Screen: Provides a welcome message and instructs the user to '***Enable Content***' in the ***Message Bar***. If the content is not enabled, all macros and VBA modules are disabled for security reasons. Content must be enabled for full database functionality. After enabling content, the *Login* dialog form is opened.

Login Dialog: Requires selecting an employee name from a drop-down list, populated from a 'Select' query based on ***EMPLOYEES EXTENDED***. Following selection, an embedded macro (Property Sheet, Event tab, After Update property) assigns the selected Employee ID as a temporary variable, hence identifying the employee participating in sales and purchasing processes, and recording employee information in related transaction tables and queries. The Employee ID variable is used to auto-fill forms throughout various purchase order and sales processes. After login, the Home form is opened.

Home: Provides form hyperlinks for new customer and new purchase orders, a refresh button, as well as subforms displaying active orders, inventory to reorder, and quick links (Figure 7.2). Note the 'I am:' drop-down allows changing the employee so the sample user does not have to close and reopen the database to login as a different employee each time various process components are transacted, e.g., create, submit, approve a purchase order, etc. Hyperlinks and subforms embedded in the Home form include:

1. ***New Customer Order*** (form hyperlink): Opens the ***Order Detail***s form.

2. ***New Purchase Order*** (form hyperlink): Opens the ***Purchase Order Details*** form.

3. ***Active Orders*** (subform): Displays the ***Active Orders Subform*** for ***Home*** form as the Source Object, binding only data for the selected employee in the Property Sheet, by setting the Data tab Link Master Fields property to the current employee and the Link Child Fields property to the Employee ID. See Chung (2012) for information on subform properties for Master and Child Link Fields properties.

4. ***Active Orders Subform for Home*** (form): Used as the subform for (3) above. The bound data Record Source (Property Sheet, Data tab) is a 'Select' query based on **ORDERS**.

5. ***Inventory to Reorder*** (subform): Displays the ***Inventory to Reorder*** subform for ***Home*** form as the Source Object

6. ***Inventory to Reorder Subform for Home*** (form): Used as the subform for (5) above. The bound data Record Source (Property Sheet, Data tab) is a 'Select' query based on **INVENTORY**. Clicking on any product field hyperlink executes an embedded macro (Property Sheet, Event tab, On Click property) that opens the ***Product Details*** form for the selected product.

7. ***Quick Links*** (rectangle control): Displays 7 Command Buttons to view various forms. Clicking on any button executes an embedded macro (Property Sheet, Event tab, On Click property) that opens the associated form (Inventory List, Orders List, Customers List, Purchase Orders List, Suppliers List, Employees List, Shippers List, Sales Reports Dialog).

8. ***Sales Analysis Subform*** for Home Chart

10.4 Stand-Alone Macros

Autoexec: Automatically runs when database opens. If user does not click the ***Enable Content*** button, then the ***Startup Screen*** form is displayed. If user clicks the ***Enable Content*** button, then the ***Login Dialog*** form is displayed.

Delete All Data: Deletes all data in the transaction tables.

10.5 VBA Class Objects, Modules, Functions, Subroutines and Procedures

Many Microsoft Office Applications software, including Access, Excel and Word include Visual Basic for Applications (VBA) that is a tool allowing development of programs within the application. As such, a developer can write programs that may automate some aspect of the software, perform calculations, or respond to events (among many other tasks). The programs are written using the Visual Basic Editor (VBE) inside of the application. As with any object-oriented language, VBA allows one to define an object by code in a class module, then add the module to an application within the software, and then add the property and method code to a module before using the object that the class module defines.

Although writing code in VBA is well beyond the scope of this paper, the end result of the modules in Northwind is that actions are performed based on triggers, like opening an Access form from another form's button or hyperlink. The trigger is clicking on a button or hyperlink, while the action is opening the form. A good tutorial on VBA in Access can be found at SourceDaddy (Object-Oriented Programming with VBA. (n.d.)).

11. CONCLUSIONS

There are few large illustrative databases available for teaching and learning database design, and implementation concepts. The Access Northwind database is one such large illustrative database that can be used as teaching database concepts. Although the database is robust, it is largely undocumented, and unfortunately has several errors and omissions. For documentation purposes, this paper first provided an overview of the company's business processes, then established process business rules, described and defined the primary objects (including tables, queries, forms and reports), described relationships and participations, provided a navigational overview of the database as it relates to procurement and fulfillment processes, mapped the transactional processes to the underlying tables and queries, and discussed some problems with the existing design.

Additionally, the navigation overview related the effect of transactional processes on underlying tables and queries. Several errors were also discussed and are most likely due to Microsoft publishing an incomplete illustrative template. As such, the database is made much more usable as a teaching tool as well as a template for database design. Although not provided in the paper, an instructional set with screen-shots is available from the author illustrating each step of both the procurement and fulfillment processes. The Teaching Notes specifies the target audience, the topic coverage, prerequisite knowledge, and some learning outcomes associated with the case, and also provides suggestions for additional student activities.

12. REFERENCES

Access 2010 Specifications. (n.d.). Retrieved October 2, 2015, from https://support.office.com/en-us/article/Access-2010-specifications-1e521481-7f9a-46f7-8ed9-ea9dff1fa854?ui=en-US&rs=en-US&ad=US&fromAR=1

An Essential Guide to Using the Northwind Database in Access 2010. (2010). In *Best STL*. Retrieved September 30, 2012, from microsofttraining.net: http://www.microsofttraining.net/article-1262-essential-guide-using-northwind-database-in-access-2010.html

Carpernet, D. A. (2008). Clarifying Normalization. *Journal of Information Systems Education*, 19(4), 379-382.

Casterella, G. I. & Vijayasarathy, L. (2013). An Experimental Investigation of Complexity in Database Query Formulation Tasks. *Journal of Information Systems Education*, 24(3), 211-221.

Chapple, M. (2012). *Installing the Northwind Sample Database in Microsoft Access 2010*. Retrieved September 30, 2012, from About.com: http://databases.about.com/od/sampleaccessdatabases/ht/Installing-The-Northwind-Sample-Database-In-Microsoft-Access-2010.htm

Chung, L. (2012). *Database Evolution: Microsoft Access within an Organization's Database Strategy*. Retrieved October 29, 2012, from FMSInc.com: http://www.fmsinc.com/MicrosoftAccess/Strategy/index.asp

Create a Data Macro. (2010). In *Microsoft*. Retrieved Oct 17, 2012, from Office.com: http://office.microsoft.com/en-us/access-help/create-a-data-macro-HA010378170.aspx

Free Access 2013 Tutorial at GCFLearnFree. (2015). In *GCF Global*. Retrieved October 5, 2015, from http://www.gcflearnfree.org/access2013

Green, G. C. (2005). Greta's Gym. A Teaching Case for Term-Long Database Projects. *Journal of Information Systems Education*, 16(4), 387-390.

Hsiang-Jui, K. & Hui-Lien, T. (2006). An Alternative Approach to Teaching Database Normalization: A Simple Algorithm and an Interactive e-Learning Tool. *Journal of Information Systems Education*, 17(3), 315-325.

Introduction to Access Programming. (2010). Retrieved October 17, 2012, from Office.Com: http://office.microsoft.com/en-us/access-help/introduction-to-access-programming-HA010341717.aspx

Introduction to Macros. (n.d.). Retrieved October 6, 2015, from https://support.office.com/en-us/article/Introduction-to-macros-a39c2a26-e745-4957-8d06-89e0b435aac3

Irwin, G., Wessel, L., & Blackburn, H. (2012). The Animal Genetic Resource Information Network (AnimalGRIN) Database: A Database Design & Implementation Case, *Journal of Information Systems Education*, 23(1), 19-27.

Itri, M. (2012). Applying Analogical Reasoning Techniques for Teaching XML Document Querying Skills in Database Classes. *Journal of Information Systems Education*, 23(4), 385-394.

Jukić, N. & Gray, P. (2008). Using Real Data to Invigorate Sudent Learning. *SIGCSE Bulletin*, 40(2), 6-10.

Learn the Structure of an Access Database. (2007). In *Office*. Retrieved October 17, 2012, from Office.Com: http://office.microsoft.com/en-us/access-help/learn-the-structure-of-an-access-database-HA001213954.aspx

MySQL. (2013). Retrieved December 17, 2014, from http://www.mysql.com/

Northwind and Pubs Sample Databases for SQL Server 2000. (2010). In *Microsoft*. Retrieved December 17, 2014, from http://www.microsoft.com/en-us/download/details.aspx?id=23654

Northwind Database. (2011). In *Codeplex*. Retrieved December 12, 2014, from https://northwinddatabase.codeplex.com/

Object-Oriented Programming with VBA. (n.d.). Retrieved October 5, 2015, from sourceDaddy: http://sourcedaddy.com/ms-access/defining-objects-with-class-modules.html

Olsen D. & Hauser, K. (2007). Teaching Advanced SQL Skills: Text Bulk Loading. *Journal of Information Systems Education*, 18(4), 399-402.

Rice, F. C. (2005). *Using Northwind Traders can Save Yourself a lot of Time*. Retrieved October 15, 2012, from frice's WebLog: http://blogs.msdn.com/b/frice/archive/2005/10/16/481710.aspx

Sakila. (2013). *Sakila Sample Database*, Retrieved December 14, 2014, from http://dev.mysql.com/doc/sakila/en/

Unch, J. M. (2009). An Approach to Reducing Cognitive Load in the Teaching of Introductory Database Concepts. *Journal of Information Systems Education*, 20(3), 269-275.

Yue, K. (2013). Using a Semi-Realistic Database to Support a Database Course. *Journal of Information Systems Education*, 24(4), 327-333.

AUTHOR BIOGRAPHIES

John N. Dyer is a Professor of Information Systems and Certified SAP Business Consultant at Georgia Southern University, with degrees in statistics (Ph.D., MS,), business (MBA) and information systems (MMIS). He has over twenty-two years of teaching, research and consulting experience in the fields of ERP, information systems, database, statistics and finance, as well as private industry experience as a statistician and 13 years of military service.

Camille Rogers is an Associate Professor and Chair of the Department of Information Systems at Georgia Southern University. She is the founding Director of the online Graduate ERP/SAP program and her teaching areas include enterprise information systems, SAP certification, ABAP programming, and Java development. Her research areas are in application development, STEM education, online teaching, and ethical issues in IS. She frequently conducts SAP training for other universities and Fortune 500 companies. Prior to academia, Dr. Rogers served in the USAF as an avionics technician and a database programmer.

APPENDIX A

Master Tables	**Supporting & Related Tables**
EMPLOYEES	EMPLOYEE PRIVILEGES (supported by PRIVILEGES)
CUSTOMERS	na
SUPPLIERS	na
PRODUCTS	na
SHIPPERS	na
Transaction Tables	**Supporting & Related Tables**
PURCHASE ORDERS	EMPLOYEES, SUPPLIERS, PURCHASE ORDER STATUS
PURCHASE ORDER DETAILS	PURCHASE ORDERS, PRODUCTS
ORDERS	EMPLOYEES, CUSTOMERS, ORDERS, TAX STATUS, ORDERS STATUS
ORDER DETAILS	ORDERS, PRODUCTS, ORDER DETAIL STATUS
INVOICES	ORDERS
INVENTORY TRANSACTIONS	PURCHASE ORDERS, PURCHASE ORDER DETAILS, ORDERS, PRODUCTS, INVENTORY TRANSACTION STATUS

Table 1. Northwind Database Tables

Table	**Primary Key(s)**	**Foreign Key in Related (1:M) Table(s)**
EMPLOYEES	ID	PURCHASE ORDERS, ORDERS, EMPLOYEE PRIVILEGES
PRIVILEGES	Privilege ID	EMPLOYEE PRIVILEGES
EMPLOYEE PRIVILEGES	ID, Privilege ID	na
SUPPLIERS	ID	PURCHASE ORDERS
CUSTOMERS	ID	ORDERS
SHIPPERS	ID	ORDERS
PRODUCTS	ID	PURCHASE ORDER DETAILS, ORDER DETAILS, INVENTORY TRANSACTIONS
PURCHASE ORDER STATUS	Status ID	PURCHASE ORDERS
PURCHASE ORDERS	Purchase Order ID	PURCHASE ORDER DETAILS, INVENTORY TRANSACTIONS
PURCHASE ORDER DETAILS	ID	na
ORDER STATUS	Status ID	ORDERS
ORDERS TAX STATUS	ID	ORDERS
ORDERS	Order ID	ORDER DETAILS, INVENTORY TRANSACTIONS, INVOICES (1:1)
ORDER DETAILS STATUS	Status ID	na
ORDER DETAILS	ID	na
INVENTORY TRANSACTION TYPES	ID	INVENTORY TRANSACTIONS

Table 2. Tables, Primary Keys and Foreign Keys

Rule	Business Rule Description
1	An **EMPLOYEE** may be granted many **PRIVILEGES**, and a **PRIVILEGE** can be granted to many **EMPLOYEES**; *3*, *O*. A **PRIVILEGE** may be assigned to many employees in **EMPLOYEE PRIVILEGE**; *2*, *O*. An **EMPLOYEE** may be granted many privileges in **EMPLOYEE PRIVILEGE**; *2*, *O*.
2	An **EMPLOYEE** can make many product **PURCHASE ORDERS**, but a **PURCHASE ORDER** can be made by one and only one **EMPLOYEE**; *2*, *O*.
3	A **SUPPLIER** can supply products in **PURCHASE ORDERS**, but a **PURCHASE ORDER** can be supplied by one and only one **SUPPLIER;** *2*, *O*.
4*	A **PURCHASE ORDER STATUS** can be assigned to many **PURCHASE ORDERS**, but a **PURCHASE ORDER** will be assigned one and only one **PURCHASE ORDER STATUS**; *2*, *O*.
5	A **PURCHASE ORDER** can generate a purchase for many products in **PURCHASE ORDER DETAILS**, but a product in **PURCHASE ORDER DETAIL** is generated by one and only one **PURCHASE ORDER**; *2*, *M*.
6	A **PRODUCT** can be purchased many times in **PURCHASE ORDER DETAILS**, but a **PURCHASE ORDER DETAIL** is for one and only one **PRODUCT;** *2*, *O*.
7*	An **INVENTORY TRANSACTION TYPE** can be assigned to many **INVENTORY TRANSACTIONS**, but an **INVENTORY TRANSACTION** is assigned one and only one **INVENTORY TRANSACTION TYPE**; *2*, *M*.
8	A **PRODUCT** from a purchase order can appear in many **INVENTORY TRANSACTIONS**, but an **INVENTORY TRANSACTION** will be for one and only one purchased **PRODUCT**; *2*, *O*.
9*	A **PURCHASE ORDER** can be assigned to many **INVENTORY TRANSACTIONS**, but an **INVENTORY TRANCTION** will be for one and only one **PURCHASE ORDER**; *2*, *M*.
10*	An **INVENTORY TRANSACTION** can be assigned to many **PURCHASE ORDER DETAILS**, but a **PURCHASE ORDER DETAIL** is associated with one and only one **INVENTORY TRANSACTION**; *2*, *M*.
11	An **EMPLOYEE** can process many customer **ORDERS**, but a customer **ORDER** can be processed by one and only one **EMPLOYEE**; *2*, *O*.
12	A **CUSTOMER** can place many **ORDERS**, but an **ORDER** can be placed by one and only one **CUSTOMER**; *2*, *O*.
13	A **SHIPPER** can ship many customer **ORDERS**, but an **ORDER** can be shipped by one and only one; *2*, *O*.
14*	An **ORDERS STATUS** can be assigned to many customer **ORDERS**, but an ORDER can be assigned one and only one **ORDERS STATUS**; *2*, *M*.
15*	An **ORDERS TAX STATUS** ('Tax Exempt,' 'Taxable') can be assigned to many **ORDERS**, but a customer **ORDER** can be assigned one and only one **ORDERS** TAX **STATUS**; *2*, *M*.
16*	An **ORDER** can generate a customer purchase for many products in **ORDER DETAILS**, but a product in ORDER DETAIL is generated by one and only one customer **ORDER**; *2*, *M*.
17	A **PRODUCT** can be purchased many times in **ORDER DETAILS**, but an **ORDER DETAIL** is for one and only one **PRODUCT**; *2*, *O*.
18	An **ORDER DETAIL STATUS** can be assigned to many **ORDER DETAILS**, but an **ORDER DETAIL** will be assigned one and only one **ORDER DETAIL STATUS**; *2*, *M*.
19*	An **ORDER** may generate one **INVOICE**, and an **INVOICE** is generated by one and only one **ORDER**; *1*, *O*.
20	An **INVENTORY TRANSACTION TYPE** can be assigned to many **INVENTORY TRANSACTIONS**, but an **INVENTORY TRANSACTION** is assigned one and only one **INVENTORY TRANSACTION TYPE**; *2*, *M*.
21	A **PRODUCT** from a customer order can appear in many **INVENTORY TRANSACTIONS**, but an **INVENTORY TRANSACTION** will be for one and only one sold **PRODUCT**; *2*, *O*.
22	An **ORDER** can be assigned to many **INVENTORY TRANSACTIONS**, but an **INVENTORY TRANCTION** will be for one and only one **ORDER**; *2*, *O*.

* see note in Table 4. **Relationships**: *1*=1-to-1, *2*=1-to-Many, *3*=Many-to-Many. **Participation**: *O*=Optional, *M*=Mandatory

Table 3. Business Rules

Rule	Notes
4	**PURCHASE ORDER STATUS**: New, Submitted, Approved, Closed
7	**INVENTORY TRANSACTION TYPE**: 'purchased' for purchase orders.
9	When products from a specific purchase order are received and added to inventory, the originating *Purchase Order ID* is recorded in **INVENTORY TRANSACTIONS**. Note that in the illustrative **INVENTORY TRANSACTIONS** the *Purchase Order ID* attribute is blank for all records, but new purchase orders entered using the appropriate form will record the *Purchase Order ID*.
10	Note that a single purchase order can result in many inventory transactions, e.g., a purchase order for two products will result in two records in **INVENTORY TRANSACTIONS**. When inventory is received to fill a purchase order, the *Purchase Order ID* in **PURCHASE ORDER DETAILS** will be updated with the *Inventory ID* from **INVENTORY TRANSACTIONS**. Likewise, in **INVENTORY TRANSACTIONS**, each of the two separate *Transaction ID* records will be updated with the same *Purchase Order ID*. This helps associate each product **PURCHASE ORDER DETAILS** with an individual inventory receiving transaction, and associate each inventory receiving transaction in **INVENTORY TRANSACTIONS** with the purchase order that initiated the transaction. Note that in the illustrative **PURCHASE ORDER DETAILS** that many *Inventory ID* attribute values are null, but inventory items received and added to inventory using the appropriate form will record the *Inventory ID*.
14	**ORDERS STATUS**: New, Invoiced, Shipped, Closed
15	**ORDERS TAX STATUS**: Tax Exempt, Taxable. Note that the *Tax Status* attribute values are null for all records in **ORDERS**, most likely assuming all orders are generated from tax exempt customers.
19	Based on the database design, only full orders are invoiced and shipped, hence the *1-to-1* relationship. If a customer order were split into two or more shipments, then an order may generate many invoices, but this is not the case since the design assumes shipping all or nothing on a customer order. The participation is *optional* since a customer order may be canceled, hence never invoiced.
20	**INVENTORY TRANSACTION TYPE**: 'sold', or 'on-hold' for customer orders.

Table 4. Business Rule Notes

STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the Journal of Information Systems Education have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.

ISSN 1055-3096