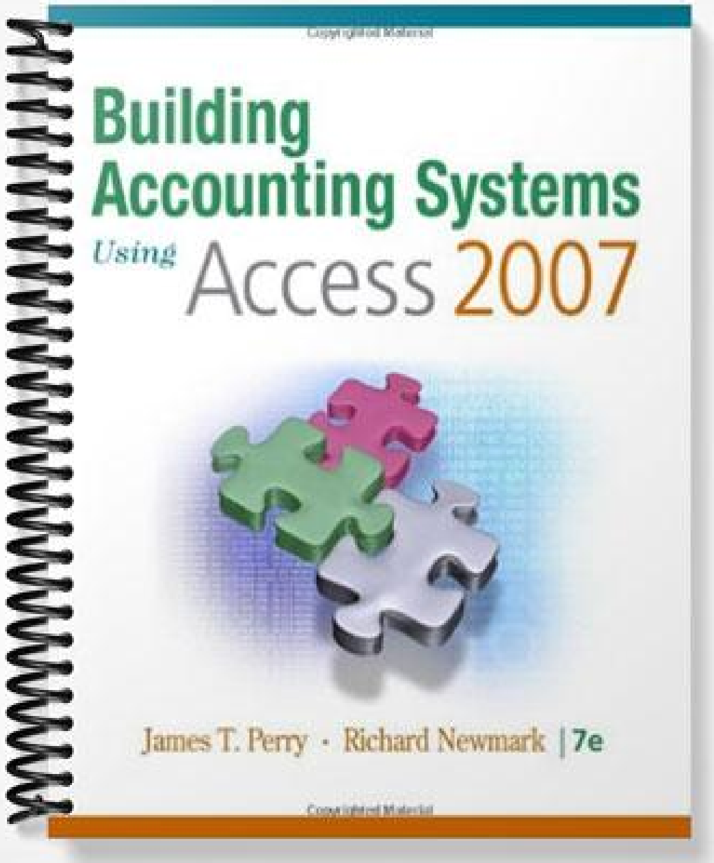
SOLUTIONS MANUAL



Building Accounting Systems Using Access 2007

James T. Perry · Richard Newmark | 7e

Chapter 2

Databases and Accounting Systems

Answers to Review Exercises

Multiple Choice Questions

1.	b	6.	c
2.	d	7.	a
3.	b	8.	b
4.	a	9.	d
5.	c	10.	b

Discussion Questions

The solutions presented here come from the textbook discussion. Your students may include other insight that is relevant but not presented in these solutions.

1. Double-entry bookkeeping provided an excellent way of recording transactions for many years. It helped accountants capture the essence of each transaction in a safe, effective manner. The double-entry nature of the approach also provided an automatic internal control feature in manual accounting systems. Since recording transactions with pen and paper was a time-consuming task, double-entry bookkeeping gave accountants a valuable tool that quickly identified essential elements of transactions. Today's computerized transaction processing systems have freed accountants from the need to capture only essential transaction elements and store them in a highly aggregated form. We can capture a wide variety of information about each transaction quite easily using bar code scanners and other electronic data entry devices. The costs of information storage have also dropped dramatically.

2. In manual systems, the dual nature of the accounting debit and credit model provides a built-in error correction mechanism. In automated systems, this same duality is inefficient and serves no control purpose. Manual double-entry bookkeeping systems can be very efficient; however, most computer implementations of double-entry bookkeeping use a flat file processing design instead of a relational database model. Database accounting systems only store data once and avoid the redundancies of flat-file systems. Database accounting systems use normalized tables to avoid anomalies. Anomalies can become very troublesome as the accounting system grows to a size that requires computers to handle transaction-processing volume.

3. Problems of storing data in two or more places include the creation of outdated and redundant data. Years ago, it was common for departments and individuals to create and maintain their own

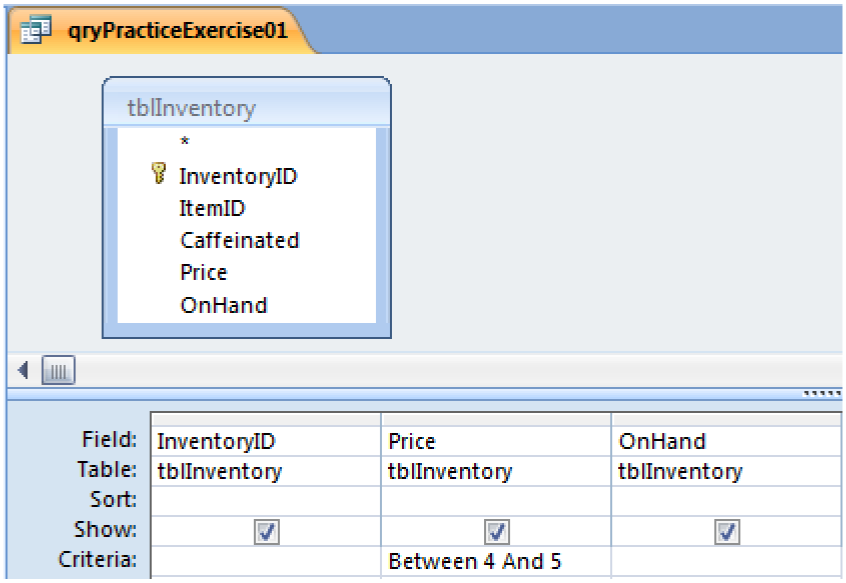
computer files that duplicated information kept in the master files. Duplicate files were maintained in order to access and examine the data with their own programs quickly, rather than incurring the long delays from the already overworked data processing department. Duplicate data files—files mirroring the master file data—leads to inevitable data redundancy and, soon, data inconsistency. For example, Marketing kept its own files, sorted by zip code, of its larger customers so that they could send them advertising pieces. The marketing department hired a bright young programmer to keep current the "best customer" file, and she would write programs to produce mailing labels from the customer file. Problems occurred when the independently maintained customer list fell out of date. While the master list of customers was kept current by the data processing department from each month's purchase orders, the marketing department did not have access to that data. Their customer list became so out of date—customers moved, and new customers came on board—that it was practically useless.

4. The primary key consists of one or more fields in each table that provides a unique identifier for each row in the table. Every table in a relational database (a table in a relational database is also called a relation) must have a primary key that is unique and that exists for each row in the table. The primary key can include one or more fields (columns). When the primary key includes more than one column it is called a composite primary key. The individual column values in a composite primary key need not be unique, but the combined column values must be unique. The primary key is important because it allows all of the information that has been broken apart to be stored in the individual tables to be pulled back together when a user needs the information. The primary keys (and the foreign keys) provide the connections among all of the tables in a database.

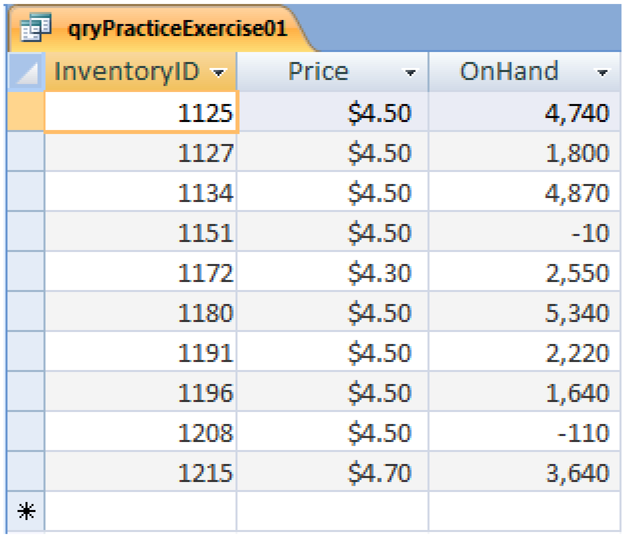
Practice Exercises

The solutions are included in the **Ch02-PracticeExercises.accdb** file on the Instructor web site.

1. One possible solution is shown below in Design view and Datasheet view.



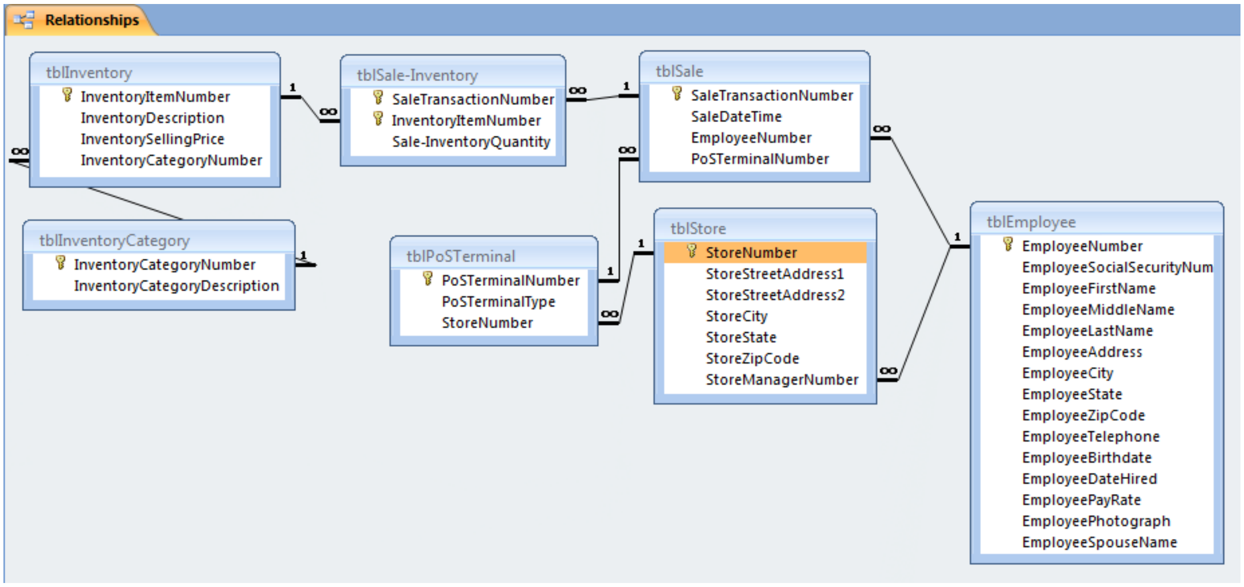
Field:	InventoryID	Price	OnHand
Table:	tblInventory	tblInventory	tblInventory
Sort:			
Show:	☑	☑	☑
Criteria:		Between 4 And 5	



qryPracticeExercise01

InventoryID	Price	OnHand
1125	$4.50	4,740
1127	$4.50	1,800
1134	$4.50	4,870
1151	$4.50	-10
1172	$4.30	2,550
1180	$4.50	5,340
1191	$4.50	2,220
1196	$4.50	1,640
1208	$4.50	-110
1215	$4.70	3,640

2. One possible solution to this exercise is the database shown in the following figure, which shows a screen capture of an Access Relationships window. In creating this file, we assumed that the grocery store does not track its customers' names or addresses and that it only accepts cash. The tblPoSTerminal stores information about each point-of-sale (PoS) terminal in each store. This database design is a useful example because it illustrates that not all businesses include customer tables in their sales/collection processes.



Relationships

tblInventory
InventoryItemNumber
InventoryDescription
InventorySellingPrice
InventoryCategoryNumber

tblSale-Inventory
SaleTransactionNumber
InventoryItemNumber
Sale-InventoryQuantity

tblSale
SaleTransactionNumber
SaleDateTime
EmployeeNumber
PoSTerminalNumber

tblInventoryCategory
InventoryCategoryNumber
InventoryCategoryDescription

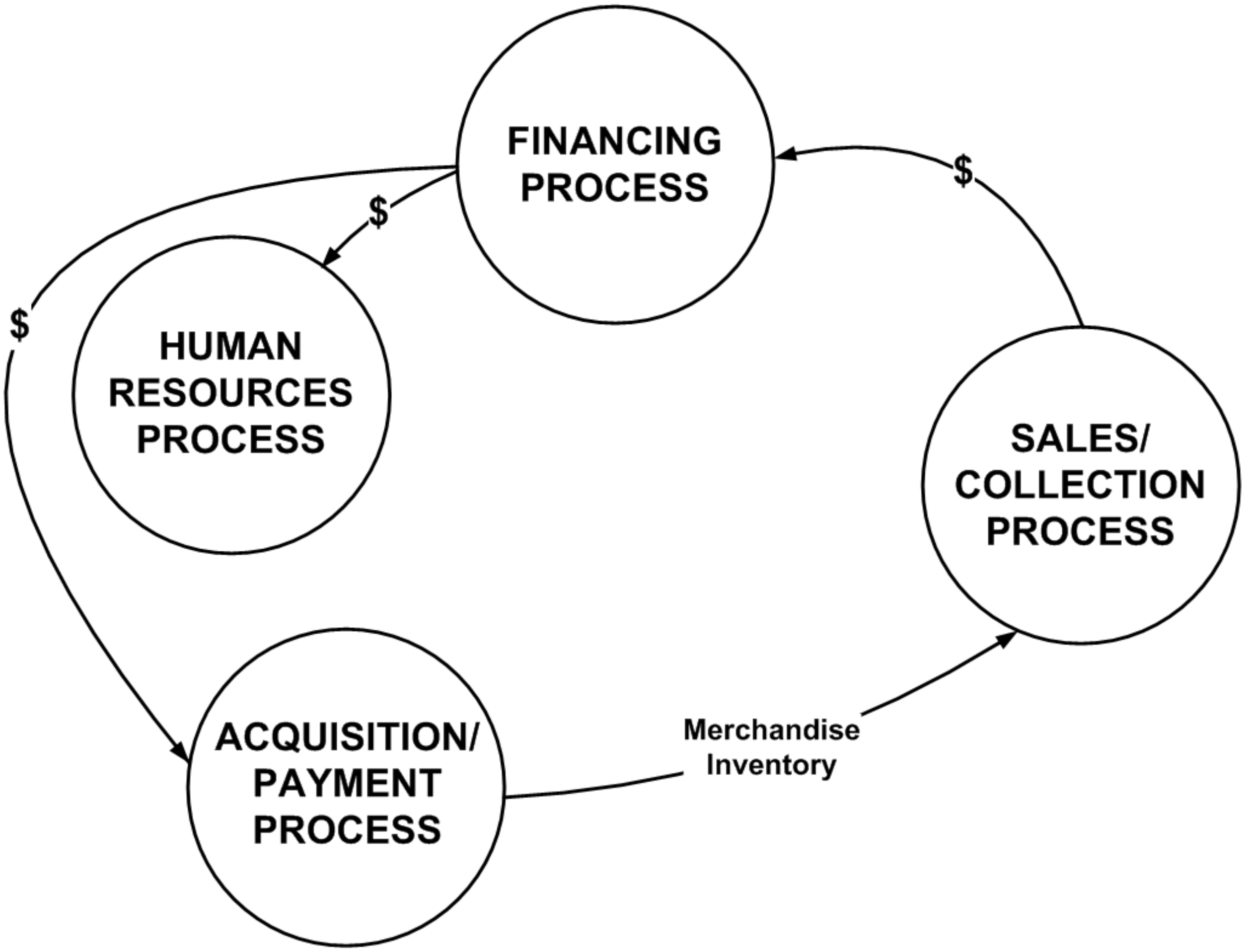
tblPoSTerminal
PoSTerminalNumber
PoSTerminalType
StoreNumber

tblStore
StoreNumber
StoreStreetAddress1
StoreStreetAddress2
StoreCity
StoreState
StoreZipCode
StoreManagerNumber

tblEmployee
EmployeeNumber
EmployeeSocialSecurityNum
EmployeeFirstName
EmployeeMiddleName
EmployeeLastName
EmployeeAddress
EmployeeCity
EmployeeState
EmployeeZipCode
EmployeeTelephone
EmployeeBirthdate
EmployeeDateHired
EmployeePayRate
EmployeePhotograph
EmployeeSpouseName

3. The additional entities would be best modeled in two separate tables, an Office table and a Department table. The primary keys of these two tables would each appear as a foreign key in the Time Worked table.

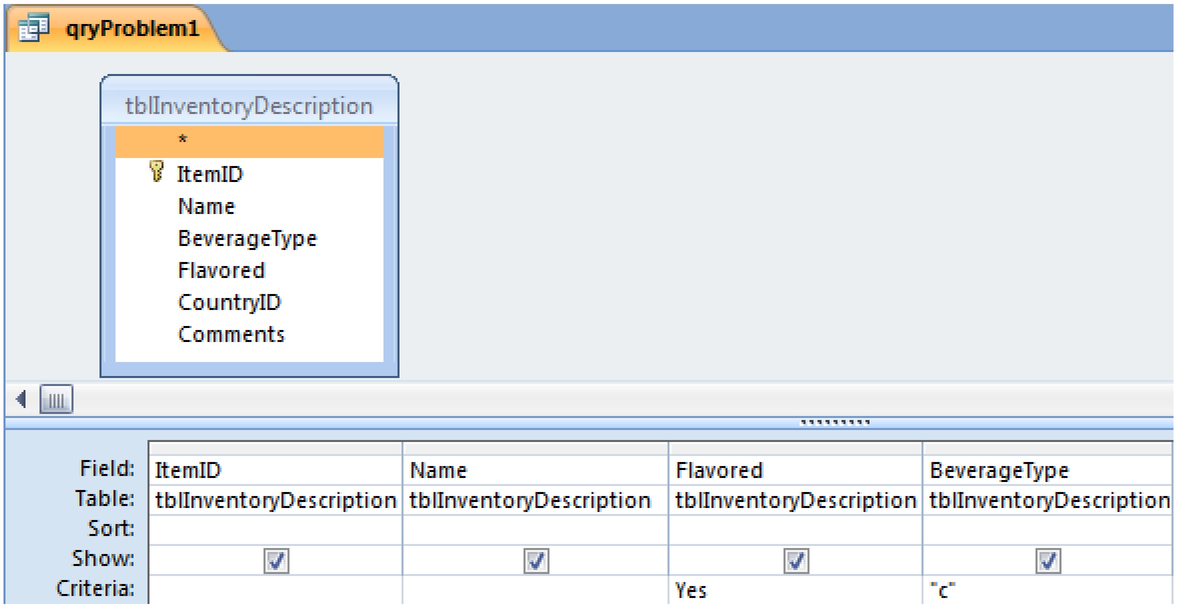
4. Omit the conversion process bubble and the related resource flow arrows from Figure 2.2. Also, add a resource flow line for merchandise inventory from the acquisition/payment process to the sales/collection process to yield the correct answer. See below.



5. The problem is that age is a value that changes each year. The solution is to store a person’s date of birth. That never changes. Follow the general rule “never store any value in a table that can be derived from other values in the table or the external environment.” Age is such a value—it can be calculated from date of birth and the current date, which is stored in the computer’s clock.

Problems

1. One possible solution is the object qryProblem01, which is included in the **Ch03Problem01.mdb** database on the Instructor’s CD. The figures below show that query in Design view and in Datasheet view.

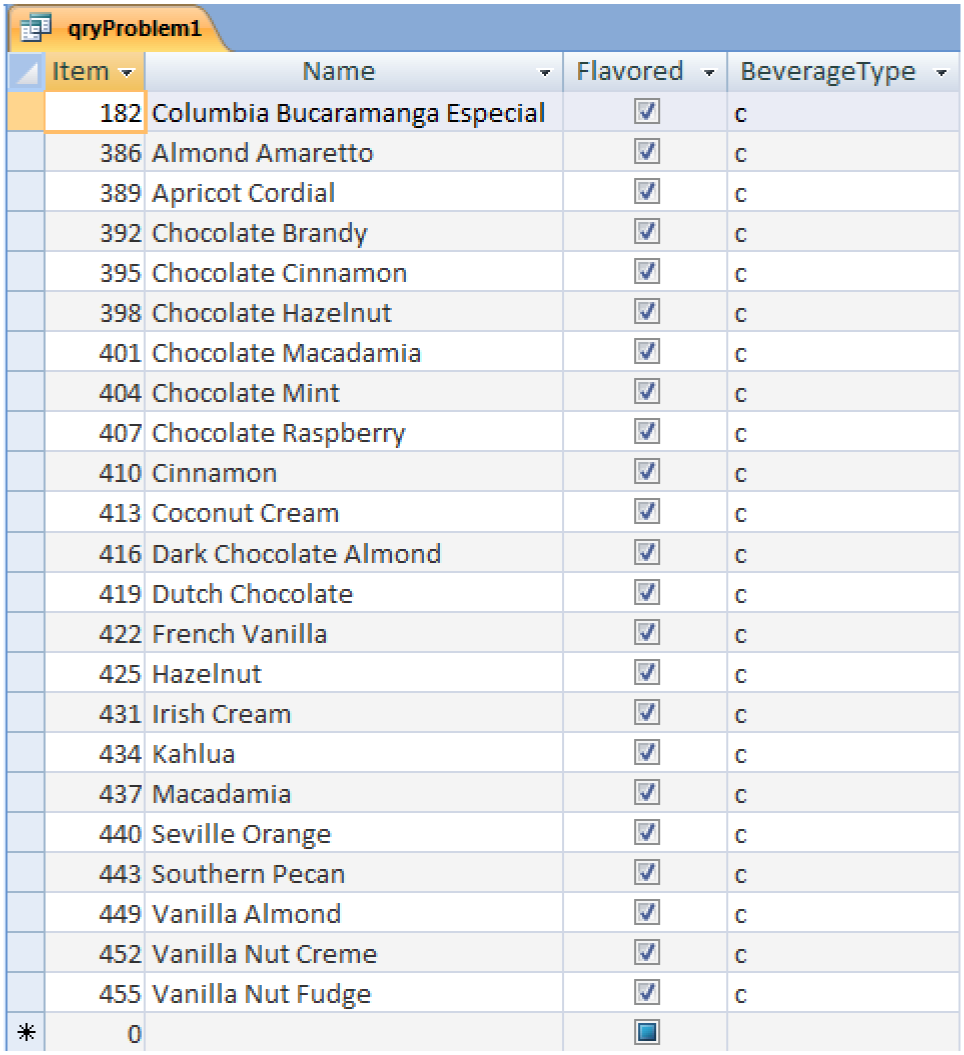


qryProblem1

tblInventoryDescription

- *
- ItemID
- Name
- BeverageType
- Flavored
- CountryID
- Comments

Field:	ItemID	Name	Flavored	BeverageType
Table:	tblInventoryDescription	tblInventoryDescription	tblInventoryDescription	tblInventoryDescription
Sort:				
Show:	☑	☑	☑	☑
Criteria:			Yes	"c"

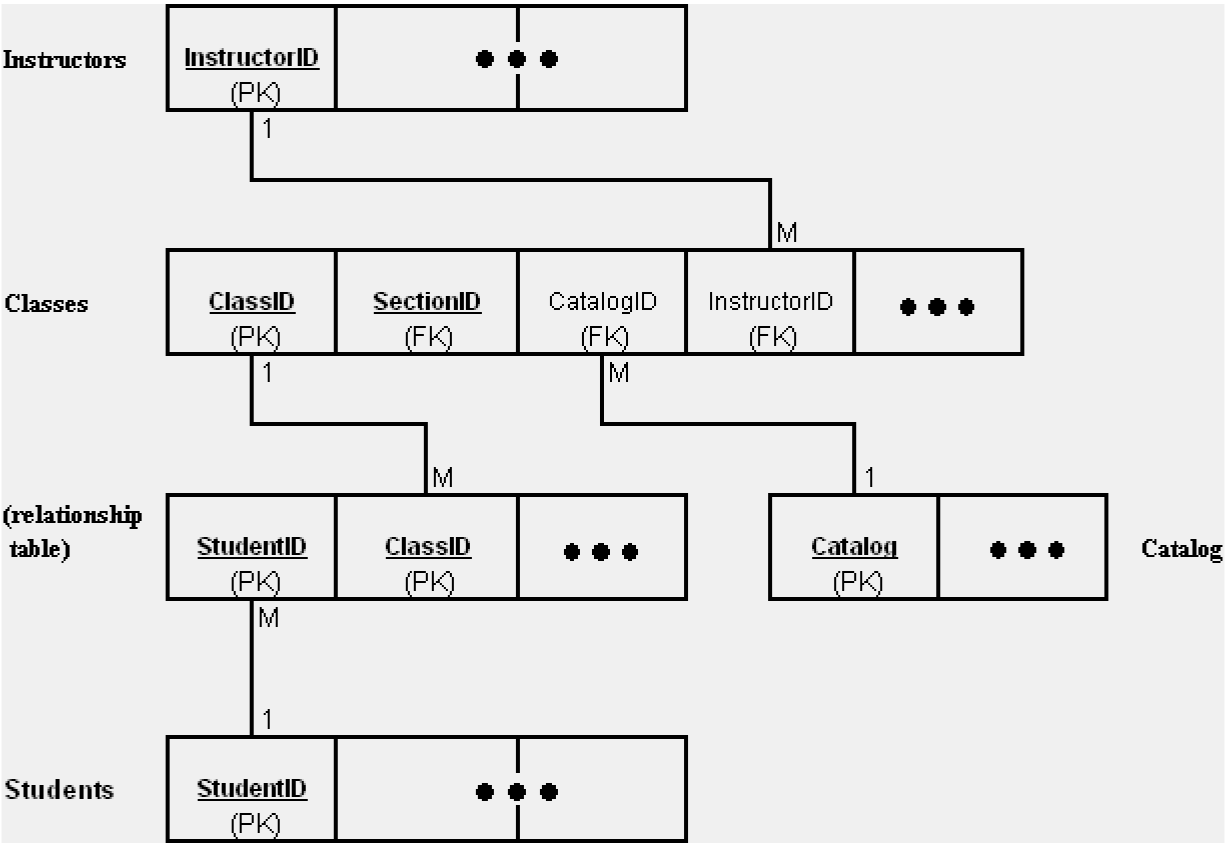


qryProblem1

Item	Name	Flavored	BeverageType
182	Columbia Bucaramanga Especial	☑	c
386	Almond Amaretto	☑	c
389	Apricot Cordial	☑	c
392	Chocolate Brandy	☑	c
395	Chocolate Cinnamon	☑	c
398	Chocolate Hazelnut	☑	c
401	Chocolate Macadamia	☑	c
404	Chocolate Mint	☑	c
407	Chocolate Raspberry	☑	c
410	Cinnamon	☑	c
413	Coconut Cream	☑	c
416	Dark Chocolate Almond	☑	c
419	Dutch Chocolate	☑	c
422	French Vanilla	☑	c
425	Hazelnut	☑	c
431	Irish Cream	☑	c
434	Kahlua	☑	c
437	Macadamia	☑	c
440	Seville Orange	☑	c
443	Southern Pecan	☑	c
449	Vanilla Almond	☑	c
452	Vanilla Nut Creme	☑	c
455	Vanilla Nut Fudge	☑	c
0		■	

2. The Invoice Line table is a relationship table. All relationship tables have a composite primary key that includes two or more individual fields (the primary key fields of the entity tables that the relationship table joins). In this case, tblInvoiceLine joins the Invoice table and the Inventory table. Thus, the primary key of tblInvoiceLine includes the InvoiceID field and the InventoryID field. The extension is not included in the table because it is a calculated field. A printed invoice would be produced by creating a report based on a query. That query would calculate the extension.

3. The relationship between the Instructor table and a Classes table is one-to-many (1—M). Each instructor teaches many classes (unless he or she is a researcher and teaches only one class per semester—atypical of most instructors). The relationship between Students table and the Classes table is many-to-many (M—M). That is, a given student can take several classes. Any particular class can have several students enrolled in it. The following shows a typical relationship between the Catalog, Classes, Instructors, and Students. Notice a relationship table exists when necessary to convert M—M relationships to two 1—M relationships.



Instructors
InstructorID
(PK)
● ● ●
1
M
Classes
ClassID
(PK)
SectionID
(FK)
CatalogID
(FK)
InstructorID
(FK)
● ● ●
1
M
M
1
(relationship table)
StudentID
(PK)
ClassID
(PK)
● ● ●
Catalog
(PK)
● ● ●
Catalog
M
1
Students
StudentID
(PK)
● ● ●