Lakeside View Global Vision
CEGEP JOHN ABBOTT COLLEGE

Continuing Education / Formation Continue

Certificate Program in Internet Programming & Development (A.E.C. LEA.BN)
Full-Time Day Training Program

(420-P34-AB) Database 2

Assignment - Project

Deadline for submission:
First Part (Assignment) February 10th, 2021
Second Part (Team Project) February 19th, 2021
Late submissions will not be accepted.

Version 1

Student Name	Student ID

Purpose

To design and develop a database management system using recommended best practices covered in class. You will work to prepare final project based on a case problem. You should use all techniques covered in this course.

This project describes a case study project. You must follow all requested tasks and apply them to your own project.

You will work in team, with **3-4 students per team**, for the term project database system. Students are expected to form their own teams and send me an email with the team member names.

Although one person can maintain a small database management system, larger systems require groups of people filling a variety of roles. Of course, the line between these roles can be blurred, and many aspects of database design require collaboration to solve a problem.

Project Update Meeting: at the end of each class, you'll have about 15 minutes to discuss your project.

Begin working with your team members in class 1. It's often helpful if someone volunteers to be the "leader". Another group member should volunteer to post the group's SUMMARY message. Some groups divide up the work and each student has a smaller task. Other groups have each member research and then they combine the best of the answers together. It's up to you.

How to be successful in the group work:

- Determine which group you are in.
- Decide in the next class how to divide the work. Decide who will post the summary response.

Assignment includes

1. Creating database, schema, and table objects (Task 1).
2. Applying data integrity using constraints (Task 2).

Task 1

To ensure that you have a way to practice everything you learned in Database courses, you are going to design and complete a separate database for a similar business, **Aragon Pharmacy Canada**, which has slightly different data needs and requirements. Aragon Pharmacy is located in Montreal, Quebec (Canada).

This pharmacy has similar needs to those of **4Corners Pharmacy USA**, but because Canada has different requirements for the data it collects about employees and drugs, you will need to develop the database with these needs in mind. You will create this database and then do all subsequent work in your copy of the database. This approach makes it possible for you to develop a database from scratch and practice everything you learned in the database courses.

In this task you will:

I. Follow the Step-by-Step in Designing a database to create the **Aragon Pharmacy database** (see the **4Corners Pharmacy database** Case Study at the end of this document)
II. Create tables in the **Aragon Pharmacy database**
III. You will create at least 12 tables in the Aragon Pharmacy database using the skills you have learned so far.
    1. **Doctor table**: *DoctorID, lastName, firstName, middelInitial, phone, extension, cellphone, photograph, …*
    2. **Customer table**: *CustomerID, lastName, firstName, middelInitial, date of birth, gender, address, city, province, postalcode, phone, and so on*
    3. Create the rest of tables based on the business rules defined in this project.

Task 2

In this Task, you will:

1. Define and use all types of constraints to enforce business rules: Primary Key, Foreign Key, Default, Check, and Unique constraints to enforce data integrity
2. Create or modify the default constraint that makes QC (Quebec) the default for the province column in the customer table.
3. Create a phone number constraint to the **customer** table. You may use regular expression pattern.
4. Create all primary key constraints.
5. Write a script that adds all foreign key constraints.
6. Write a script that adds all default constraints.
7. Write a script that adds all check key constraints.
8. Write a script that adds all unique constraint

Introduction to the Company Database Case Study

The problems to be solved in this scenario are presented within the context of the company, 4Corners Pharmacy, which is so named because it is located in southwestern Colorado on the border of the four corners connecting the states of Arizona, Colorado, New Mexico, and Utah. Case scenarios involving 4Corners Pharmacy are used to provide real-world business examples to illustrate concepts of designing and developing a database. They are not based on real people or real events. You will be guided through the solutions to solve realistic business problems for various people working for this company. These "employees" work in a variety of business departments: accounting, finance, human resources, information systems, marketing, operations management, and sales. Context is an important factor to consider when solving the problems in this study case. The following background on 4Corners Pharmacy provides perspective on the situations that you will encounter throughout this case.

Vincent Ferrino opened 4Corners Pharmacy in 2014. Over the years, Vincent's business evolved from a small "mom-and-pop" corner drugstore to the busiest pharmacy in the area. Vincent's son, Paul Ferrino, has worked at the pharmacy in different capacities since 2015. After graduating with a degree in pharmacology and becoming licensed by the state of Colorado, Paul worked as a pharmacist in his father's store, and then took two years off to earn an MBA from Thunderbird University in Arizona. When Vincent retired, he sold the store to Paul, but continues to work as a part-time pharmacist.

Paul envisions expanding his father's business to take advantage of the growing pharmaceutical industry. Paul was encouraged recently by a study indicating that 47% of Americans take prescription drugs on a regular basis. He sees the trend increasing as life expectancy rises and as aging baby boomers need more prescriptions. Although he was a shrewd and successful businessperson, Vincent ran the day-to-day operations of the business with little help from computers, primarily because he was never trained to use them and never realized the benefits they could offer his business. Consequently, the pharmacy's recordkeeping, although meticulous and professional, is inefficient.

Maintaining the growing business using mostly manual systems is becoming more costly for two reasons. The pharmacy has had to hire additional employees to meet industry regulations regarding the Health Insurance Portability and Accountability Act (HIPAA) and because of federal and state regulations that affect the sale, storage, and dispensing of prescription drugs. Although Paul succeeded in automating some of the pharmacy's data management in Excel workbooks, he knows that a more substantial change is needed to properly maintain and store the company's data.

Key Players

Paul Ferrino – Pharmacist and Owner

Paul began working as a pharmacist at 4Corners Pharmacy in January 2015 and bought the store from his father in May 2018. Paul is the head pharmacist, and as such, he manages a dedicated and capable group of experienced pharmacists, pharmacy technicians, and sales assistants. Paul's vision for the pharmacy is to continue his father's lifelong pledge of providing excellent customer service and "giving back" to the community.

Vincent regularly sponsored and underwrote local events for kids and senior citizens, which resulted in several community service awards for Vincent and multiple wins in the "best pharmacy" category by a reader's poll sponsored by the local newspaper. The pharmacy also earned three awards from the local Chamber of Commerce. Although there are other big chain drugstores in the area, Vincent has managed to hang on to generations of customers because of his community involvement and excellent rapport with his customers. Paul is dedicated to continuing his father's traditions.

Donald Linebarger – Information Systems Director

After purchasing the store, Paul's first order of business was hiring someone who could overhaul the pharmacy's manual recordkeeping systems. Don Linebarger worked for many years as a systems analyst for a large business in the area. He was the perfect choice to work at 4Corners Pharmacy, not only because he is capable and experienced, but also because he has been a satisfied customer of 4Corners Pharmacy for many years. After developing the recordkeeping system for 4Corners Pharmacy, Don will train employees to use it, perform system maintenance, make any necessary changes, and maintain all of the records necessary for the business.

Maria Garcia – Human Resources Manager

Maria Garcia has been the human resources manager at 4Corners Pharmacy since 2017. She is responsible for interviewing and hiring employees; maintaining data about certifications for pharmacists and pharmacy technicians; monitoring attendance and sick days; and maintaining and administering company benefits, including life and health insurance, 401(k) programs, and bonuses.

Elaine Estes – Store / Operations Manager

Elaine Estes has worked at 4Corners Pharmacy since 2016. During this time, she has been the store's general manager and operations manager. Elaine maintains the inventory of prescription and nonprescription items, places orders for additional stock, and disposes of drugs when they are stocked past their expiration dates. Elaine also supervises the 4Corners Pharmacy staff and reports their achievements and any problems to Maria.

Company Goal: Expand Operations into Other Areas

The strength of 4Corners Pharmacy has always been Vincent's dedication to his customers and his community. Paul wants to continue this tradition by sponsoring local events and ensuring that his customers and their families never have a reason to seek another pharmacy. Paul also wants to open a second store in Colorado by the end of the year,
possibly by acquiring an existing business. With the addition of another store on the horizon, Don needs to ensure that the database created to run 4Corners Pharmacy is easily expandable and adaptable to run the operations at a second location.

How Is SQL Server Used at 4Corners Pharmacy?

Employees at 4Corners Pharmacy will use SQL Server in many ways to assist them with the daily operations of the pharmacy. Specific examples of how a database will improve store operations and customer service are as follows:

- **Accounting**

  As the store manager, Elaine needs reports that detail how much inventory remains for each prescription and nonprescription item in the store. She also needs to know the value of the existing inventory for insurance purposes. She can use SQL Server to compute information about sales and create a report from the information that is current and easy to read.

- **Finance**

  Elaine and Paul need to ensure the pharmacy's financial reserve is sufficient. As a result, they will be able to use SQL Server to produce reports that accurately detail the pharmacy's sales figures, salary commitments, and similar financial data to determine the financial health of the pharmacy.

- **Human Resources**

  As the human resources manager, Maria needs to manage all of the data about employees. Some of this data is descriptive, such as the employee's name and address. Other data is sensitive, such as the employee's Social Security number, salary, and information about job performance, and needs to be protected from unauthorized access. Maria can use SQL Server to manage data about employees and ensure that sensitive data is protected from fraud and abuse.

- **Information System**

  Don's goal for the new database at 4Corners Pharmacy is to produce a system that meets user needs, stores, and manages the correct information, displays information in the correct format, and is easy for employees to use. He will use the tools available in SQL Server to secure, back up, and maintain the database to ensure that it is properly protected from unauthorized access, loss, and failure. In addition, Don will ensure that the database will be expandable for future increased business.

- **Marketing**

  As the store's marketing contact, Elaine works with an outside advertising agency to produce the store's weekly ads that run in the local newspaper and on the company website. Elaine can use SQL Server to determine how store items are selling and determine which products to add to the store's advertising. She can also use SQL Server to produce reports about the buying behavior and demographics of 4Corners Pharmacy customers. Although this data exists within the pharmacy, it is not currently organized in a format that makes it easy to analyze for determining market trends and demographic buying behavior.

- **Operations Management**

  Elaine needs to be able to determine which drugs to order from suppliers and in what quantities. She frequently evaluates the store's inventory for low-drug volumes and expired drugs. Customer health is of great importance to the pharmacy. Although no one has ever been injured as a result of pharmacist negligence at 4Corners Pharmacy, increased federal regulations and legal liability issues mandate that pharmacy employees are properly certified and diligently inform customers of dosage instructions, possible side effects, and adverse interactions for each prescription sold. The pharmacy

already keeps meticulous records about pharmacist certifications, and pharmacists and pharmacy technicians diligently inform their customers about the drugs they dispense. But with the increase in hiring and the possible expansion of the company, Don can use SQL Server to put formal processes, to track and implement these requirements, into place.

- **Sales**
  Customers are also data consumers at the pharmacy. They receive data from the pharmacists detailing drug information and potential interactions. Some customers participate in flexible spending accounts (FSAs) that allow them to pay for prescriptions with pre-tax dollars. Participating customers frequently request a list of the prescriptions ordered by their households on a quarterly or annual basis. Elaine and Paul can use SQL Server to manage these requests by creating reports of all drugs ordered by a single customer or household.

All of these business areas of the pharmacy are related, just as the tables in the database are related. For example, when a customer's prescription is filled, different parts of the database are affected, as shown in the following Figure 1.

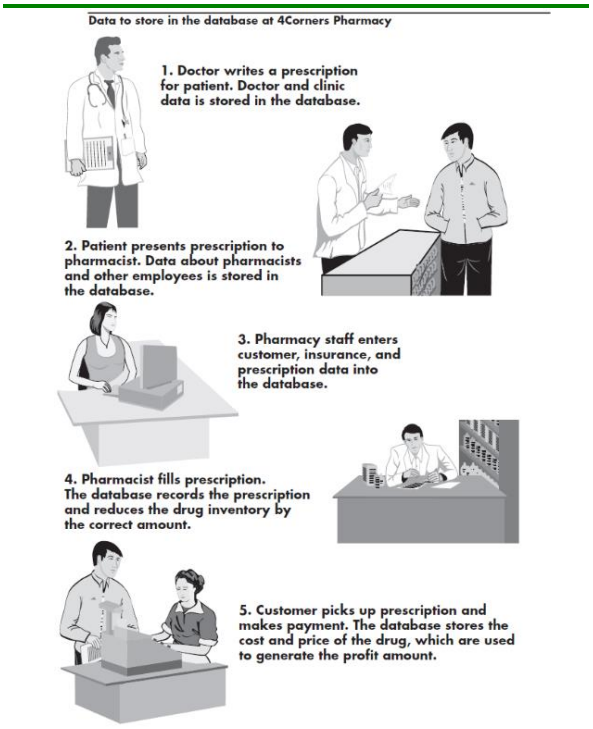


Figure 1 - Different parts of the database

The customer requests the prescription, either by presenting a written order from a doctor to a pharmacist or asking for a refill of an existing prescription. The pharmacist adds this request to the system by getting the required information to fill it, including information about the drug, customer, customer's insurance plan, and so on. The prescription also affects the inventory that is tracked by operations and the profile of buying behavior that is tracked for marketing purposes. Eventually, the prescription affects the pharmacy's financial health by contributing revenue and profit to the store.

Managing Data for the Organization

All organizations must deal with how best to collect, store, and manage their data. There are several problems at 4Corners Pharmacy. First, not all data is gathered for managers to make informed business

decisions. For example, marketing data is scarce and dispersed throughout the organization. Several departments import and export data to each other and then maintain that data separately, resulting in errors and redundancy of data.

Also, the operations department manages inventory based on data entered by the pharmacists. However, this data is not updated on a regular basis, resulting in orders placed too early, too frequently, or not frequently enough. Moreover, redundancy occurs in the customer list as each customer is listed with his own address, despite the fact that several people might live at the same address. Therefore, if one person submits a change of address or phone number, this data is not updated for people living in the same household. This problem leads to erroneous data that might result in undelivered mail sent to customers from 4Corners Pharmacy or the pharmacist not being able to reach a customer because of an incorrect phone number.

The business motivation for gathering all the data used in 4Corners Pharmacy and storing it in one relational database is to save time and money and to prevent errors and redundancy. But, why is using a database program such as SQL Server a superior method for organizing data? Why not use Excel to manage this data? After all, it is possible to sort, filter, and manipulate a significant amount of data in an Excel worksheet.

There are many benefits to using a relational database to address the data deficiencies at 4Corners Pharmacy. First, the pharmacy produces too much data to store effectively in an Excel workbook. 4Corners Pharmacy gathers data about employees, customers, drugs, health plans, and doctors, each of which is its own unique category and should have its own worksheet. It would be difficult to list all the pharmacy's employees, customers, and prescribing doctors in one Excel worksheet, much less all the drug and health plan data.

Excel is limited in the structure of its data; you cannot easily manage all of this data in one workbook, but you can do so in one database. A database might contain dozens of tables, just as an Excel workbook might contain dozens of worksheets. However, through a process of relating tables to each other, you can join together multiple tables in SQL Server if they share common data, and then query those tables as though they were one big table, presenting a huge advantage over managing data in multiple worksheets in Excel.

For example, 4Corners Pharmacy will use three tables to store and manage data about employees, customers, and doctors; a table to store and manage data about prescriptions that pharmacists fill for customers; and a table to store and manage data about the drugs prescribed, such as the drug's name, dosage, expiration date, cost, and so on. All of these tables have a relationship that connects them to the others, as shown in Figure 2.

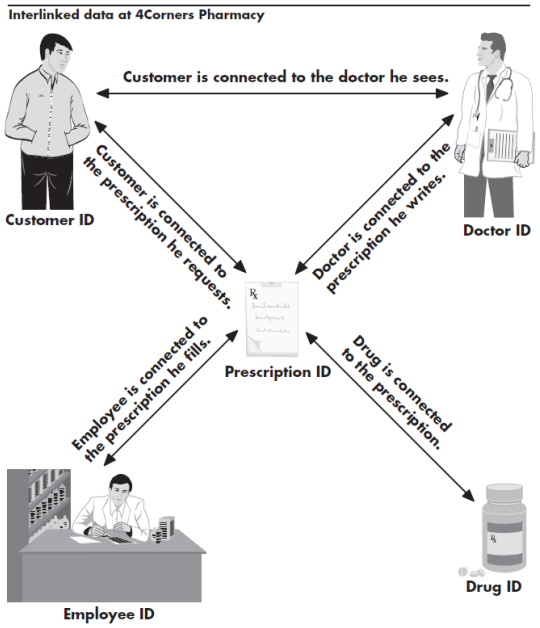


Figure 2 - Interlinked data at the Pharmacy

The above Figure shows that a customer brings in a prescription, and then a pharmacist fills the prescription. In the database, the customer's data is stored in a table that stores all customer information. The prescription table contains data about prescriptions and a reference to the customer table (such as a customer identification number) to indicate the customer who purchased the prescription. The prescription table contains a reference (such as a prescription identification number) to the table that stores data about pharmacists, a reference to which pharmacist filled the prescription, and a reference (such as a drug identification number) to the table that stores data about drugs. Finally,
the doctor table contains a reference to the prescription table, indicating which doctor wrote the prescription.

The benefit of designing tables and then creating relationships between them is that you are able to retrieve the data stored in these tables easily and quickly. For example, you can select any prescription and find out which doctor wrote it, which pharmacist filled it, and which customer purchased it. This type of querying simply isn't possible in Excel.

Problem Soving

The case study presents three levels of problem solving with SQL Server. Level 1 deals with basic problems or analyses that require the application of one or more database tools, focusing on the implementation of those tools. However, problem solving not only requires you to know how to use a tool, but, more importantly, why or when to use which tool. So, with Level 2 (more understanding of the case study) the problems and analyses increase in complexity. By the time you reach Level 3 (complete understanding of the problem), the complexity increases further, providing you with opportunities for more advanced critical thinking and problem solving. Figure 3 illustrates this approach to problem solving.

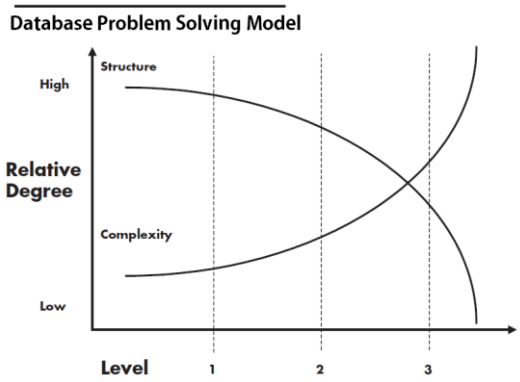


Figure 3 - Problem solving model

The goal is to increase your problem-solving skills while moving you toward an environment that is more like the real business world you will encounter during internships and upon graduation from college.

In addition, while the complexity increases, the structure of the problem to be solved decreases.
However, to ensure that you have a way to practice everything you learned in Database courses, you are going to design and complete a separate database for a similar business, **Aragon Pharmacy**, which has slightly different data needs and requirements. Aragon Pharmacy is located in Montreal, Quebec (Canada).

This pharmacy has similar needs to those of 4Corners Pharmacy, but because Canada has different requirements for the data it collects about employees and drugs, you will need to develop the database with these needs in mind. You will create this database and then do all subsequent work in your copy of the database. This approach makes it possible for you to develop a database from scratch and practice everything you learned in the database courses.

Because the database has slightly different requirements, you will be faced with decisions and problems that are unique to that database.

Discovery Phase

The first step, the discovery phase, includes gathering all existing data, researching missing and incomplete data, and talking with users about their data output needs. Subsequent steps in the process include putting data into groups, called tables; identifying unique values for each record in those tables; and designing a database to produce the desired output.
It is then imperative to test the new database prior to its implementation. Designing a database requires knowledge of database concepts, which are introduced in the Database I course.
Figure 4 illustrates the steps followed in the database design process and highlights the first step, the discovery phase.

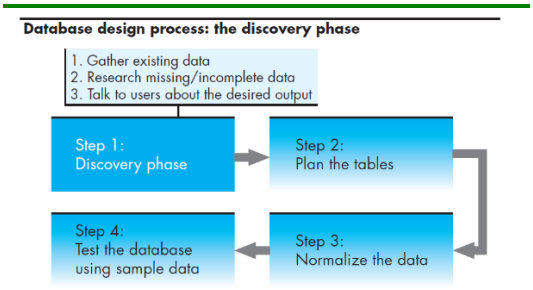


Figure 4 - Database design process: the discovery phase

Creating a database is a significant undertaking that requires careful planning, organization, and management. In many cases, it takes several weeks or even months to identify the data that an organization needs to collect and then determine how managers and employees will enter and use that data to assist them in the organization's day-to-day operations and long-term planning. 4Corners Pharmacy is an existing business, and you have already learned that Vincent Ferrino's system for managing the prescription, inventory, employee, and other data for the pharmacy is lacking in several respects. Paul Ferrino's vision for the pharmacy, which he now owns, is to convert all of the existing paper and spreadsheet systems to a database. By doing so, Paul will realize many benefits, such as more accurate and consistent data and the ability to track inventory and customer buying habits.

Discovering and Evaluating Sources of Existing Data

One of the first tasks of creating a database is to identify the information that the organization needs to manage and organize. This task might take several days to several weeks and might involve interviewing department heads and other key employees to understand the data they collect and the way they use it. You might find that some departments manage their data in paper files or in computerized records. In larger organizations, data might be stored in different computerized systems. Regardless of the current data storage method, it is important to take the time to understand not only what data is collected, but also how that data is used.

As you collect information from the organization's key players, you might begin to see patterns that indicate how to organize the data. For example, you might see that the organization manages data about customers, employees, and products. Different departments might use this data in different ways and employees might need different levels of access to this data, depending on the departments in which they work and their positions in the company. For example, a manager of an Order Department needs information about products and customers, but not about employees; a human resources manager needs information about employees, but not about products; a customer service representative needs information about products and orders, but not about employees; and so on. In addition to needing different kinds of data, some employees might need more detailed information than others. An assistant in the Human Resources Department might need a list of employees working in the organization, but only the human resources manager should be able to view their salaries.

You can use a database management system, or DBMS, to manage data. There are several DBMSs, including Oracle, MySQL, and Microsoft Access. Each of these DBMS programs has specific advantages that benefit different organizations, depending on the type and amount of data they store. For example, very large organizations will benefit from the power of Oracle to manage large amounts of data on a network or on a Web site. Other businesses might choose MySQL, an open source program. An open source program permits licensed users to use, change, improve, and redistribute the software in modified or unmodified formats. This type of software is called "free" software, in which the term free typically refers to freedom that the license

gives users. Free software might or might not be available at no charge. MySQL, however, requires programming expertise in Structured Query Language (SQL), the language used by most DBMSs. Oracle and MySQL are conceptually equivalent. Both are server based, both scale to large amounts of data, and both require knowledge of SQL.
Some midsized and smaller organizations might choose to use Microsoft Access because this DBMS, part of the Microsoft Office suite of programs, is fairly easy to use and quite powerful, but does not require extensive programming knowledge. Access can also be differentiated because it is an integrated DBMS, which includes both forms and reports along with storing the data. In addition, Access integrates very well with the other Microsoft Office programs, such as Excel. Access uses the SQL language but makes using it a simpler task compared to other programs that require programming expertise.

Because 4Corners Pharmacy is a middle-to-large business, the company's information systems director, Don Linebarger, selected **Microsoft SQL Server** as the DBMS to manage the pharmacy's database taking in consideration that the business grows in the future.

Before he gets started, Don needs to determine how the pharmacy collects and stores data about prescription transactions, drug inventories, customers, and employees. Don also needs to determine if there are other data needs at the pharmacy that are currently going unfulfilled and determine how the data will be collected. After extensive consulting with Vincent and Paul Ferrino, Don learns that data is managed in many ways and realizes that entering customer data at 4Corners Pharmacy is a manual and time-consuming process. When new customers visit the pharmacy to fill a prescription, they complete a customer information form by hand. A pharmacy technician enters the information into an Excel worksheet. Figure 5 shows the pharmacy's customer information form.

Customer information form used by 4Corners Pharmacy

CUSTOMER INFORMATION

Thank you for choosing 4Corners Pharmacy. Please fill out this form so that we can assist you promptly.

Name:	Birth Date:
Address:	
Social Security Number:	
Home Phone Number:	Fax Number:
Email Address:	
Gender: Male Female	Marital Status (circle one): Single Married Widowed Divorced
Employer:	Occupation:
Primary Care Physician:	Physician's Phone Number:
Emergency Contact:	Relationship to Patient:
Emergency Contact's Phone Number:	Spouse's Name:
Allergies:	
Prefer childproof caps? Y N	

HEAD OF HOUSEHOLD

(Complete this section only if someone other than the customer is financially responsible.)

Name:	Social Security Number:
Address:	
Home Phone Number:	Work Phone Number:

INSURANCE INFORMATION

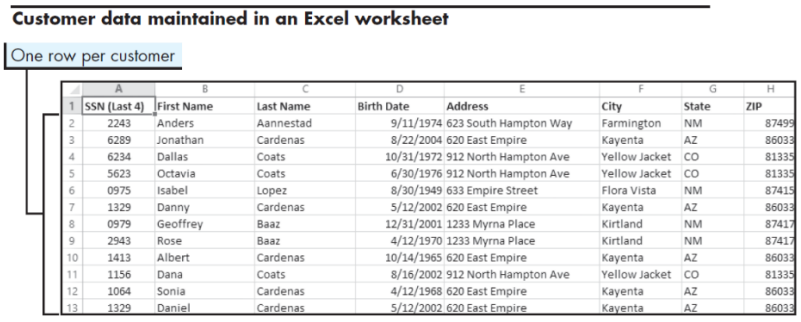
Insurance Plan:	
Insurance Plan Phone Number:	Insurance ID Number:
Subscriber's Name:	Subscriber's Employer:

I authorize the release of any medical information necessary to process my claim and payment of benefits to 4Corners Pharmacy.

Signature of Customer/Responsible Party **Date**

Figure 5 - Customer information form used by 4Corners Pharmacy

After entering the data from the customer information form into the worksheet, the technician stores the form in a filing cabinet in one of 26 file folders arranged by the first letter of the customer's last name. This system makes it difficult to find individual customer information later when forms require an update, such as a change of address, a change in insurance information, or a new drug allergy. Vincent's original objective was to have one row in the worksheet for each customer, as shown in Figure 6.



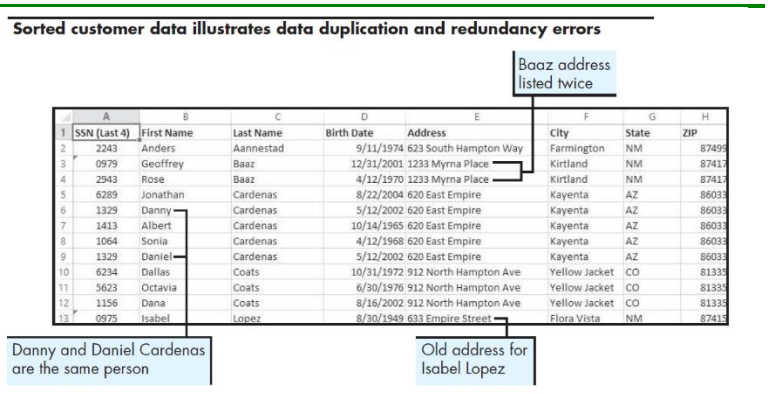
	A	B	C	D	E	F	G	H
1	SSN (Last 4)	First Name	Last Name	Birth Date	Address	City	State	ZIP
2	2243	Anders	Aannestad	9/11/1974	623 South Hampton Way	Farmington	NM	87499
3	6289	Jonathan	Cardenas	8/22/2004	620 East Empire	Kayenta	AZ	86033
4	6234	Dallas	Coats	10/31/1972	912 North Hampton Ave	Yellow Jacket	CO	81335
5	5623	Octavia	Coats	6/30/1976	912 North Hampton Ave	Yellow Jacket	CO	81335
6	0975	Isabel	Lopez	8/30/1949	633 Empire Street	Flora Vista	NM	87415
7	1329	Danny	Cardenas	5/12/2002	620 East Empire	Kayenta	AZ	86033
8	0979	Geoffrey	Baaz	12/31/2001	1233 Myrna Place	Kirtland	NM	87417
9	2943	Rose	Baaz	4/12/1970	1233 Myrna Place	Kirtland	NM	87417
10	1413	Albert	Cardenas	10/14/1965	620 East Empire	Kayenta	AZ	86033
11	1156	Dana	Coats	8/16/2002	912 North Hampton Ave	Yellow Jacket	CO	81335
12	1064	Sonia	Cardenas	4/12/1968	620 East Empire	Kayenta	AZ	86033
13	1329	Daniel	Cardenas	5/12/2002	620 East Empire	Kayenta	AZ	86033

Figure 6 - Customer data maintained in an Excel worksheet

Don knows from experience that the existing recordkeeping system is error-prone, as it is possible to store two records for the same customer without realizing the mistake. For example, if a customer has a formal name and a nickname or goes by a middle name, it might be possible to store two separate records for the same individual without realizing that these records are for the same person. This problem might also occur if the person who entered the record misspelled a customer's name on one or more occasions. This phenomenon is known as **data duplication** and it is undesirable, not only because it takes up additional space in the database, but because it often leads to inconsistent and inaccurate data. For example, a pharmacist might indicate that the customer named John W. Jackson has a drug allergy, but the record for J. William Jackson—the same person but with a different version of his name—might not indicate this important fact.

This situation might not seem important for billing or correspondence issues, but a serious error occurs when the pharmacist accesses the duplicate customer record that does not include the important information about the customer's drug allergy and gives the customer a prescription to which he is allergic. Don's first order of business is to work to avoid data duplication in the database.

In addition to deleting duplicate records for customers, Don also notes that he needs to find a way to group and store the address and phone information for people in the same household. This repetition is known as **data redundancy** and is to be avoided.
When one member of a household reports a new address or change of insurance that also affects other people in the household, the database must update the records for everyone affected by the change. Figure 7 shows the customer data shown in Figure 6, but it is sorted alphabetically by last name.



	A	B	C	D	E	F	G	H
1	SSN (Last 4)	First Name	Last Name	Birth Date	Address	City	State	ZIP
2	2243	Anders	Aannestad	9/11/1974	623 South Hampton Way	Farmington	NM	87499
3	0979	Geoffrey	Baaz	12/31/2001	1233 Myrna Place	Kirtland	NM	87417
4	2943	Rose	Baaz	4/12/1970	1233 Myrna Place	Kirtland	NM	87417
5	6289	Jonathan	Cardenas	8/22/2004	620 East Empire	Kayenta	AZ	86033
6	1329	Danny	Cardenas	5/12/2002	620 East Empire	Kayenta	AZ	86033
7	1413	Albert	Cardenas	10/14/1965	620 East Empire	Kayenta	AZ	86033
8	1064	Sonia	Cardenas	4/12/1968	620 East Empire	Kayenta	AZ	86033
9	1329	Daniel	Cardenas	5/12/2002	620 East Empire	Kayenta	AZ	86033
10	6234	Dallas	Coats	10/31/1972	912 North Hampton Ave	Yellow Jacket	CO	81335
11	5623	Octavia	Coats	6/30/1976	912 North Hampton Ave	Yellow Jacket	CO	81335
12	1156	Dana	Coats	8/16/2002	912 North Hampton Ave	Yellow Jacket	CO	81335
13	0975	Isabel	Lopez	8/30/1949	633 Empire Street	Flora Vista	NM	87415

Figure 7 - Customer data is sorted alphabetically by last name.

Don sees some serious problems with the current process of logging prescriptions using this system. First, there is no method to control the duplication of data. If a customer fills three prescriptions on the same day, there will be three rows in the worksheet—one row for each prescription filled. The name, address, and phone number can vary in each of these three rows because there is no built-in method in a worksheet to prevent this problem from occurring. It is possible to have similar variances in the doctor's name, address, clinic affiliation, and phone number. Second, data is difficult to track and gather together. For example, because the pharmacy creates new worksheets for each day's prescriptions, pharmacists would need to know the original fill date for a prescription to find out how many refills the doctor authorized. Aggregating, or gathering, data by customer, doctor, or drug would also be difficult.

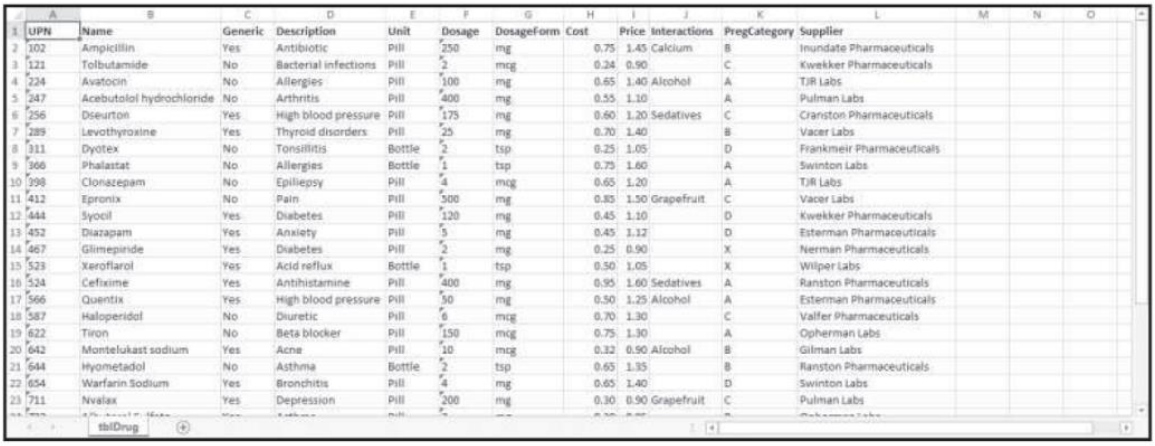
After finishing his exploration of the current system, Don asks Paul for other changes that he wants to make and types of data that he wants to add to the database that is currently not in the system. Paul tells Don that in the new database system, he would like to be able to do the following:

- Eliminate paper registration forms
- Make it easier to update customer information
- Have quick access to a customer's prescription history (without searching a paper file)
- Print a list of drug names
- Create a way to ensure that the person entering a customer's information inquired about the customer's allergies
- Verify that the pharmacist explained interactions with other drugs to a customer
- Prove that the pharmacist provided counseling about using the drug to the customer
- Print a list of doctors and their contact information

In talking with the pharmacy staff, Don also learns that in the past it has been difficult to generate items for the marketing department. Elaine sends notices to customers about pharmacy events, drug recalls, and local health fairs. She needs to be able to obtain addresses of customers and physicians to create mailing labels. Don also learns from the pharmacy technicians that some customers have requested year-end reports identifying all of the prescriptions and their total cost for each family member. The customers would like to have the total prescription drug costs to use when reconciling with flexible spending accounts (FSAs) provided by their employers. Because the current system has no way to account for this data, this need is going unmet.

After examining the pharmacy's records and discovering all of the problems with data entry errors and data duplication, Don moves on to look at how Paul manages the pharmacy's inventory of prescription drugs. He talks with Paul and Vincent about their method for tracking the value and number of each item in inventory. He also discusses the system they use for reordering out-of-stock items, evaluating items that do not sell well and discontinuing them and making sure that items on sale are stocked in appropriate quantities before the sale is advertised. Figure 8 shows the worksheet that Paul uses to track prescription drug inventory.

Prescription drug inventory data



UPN	Name	Generic	Description	Unit	Dosage	DosageForm	Cost	Price	Interactions	PregCategory	Supplier
102	Ampicillin	Yes	Antibiotic	Pill	250	mg	0.75	1.45	Calcium	B	Inundate Pharmaceuticals
121	Tolbutamide	No	Bacterial infections	Pill	2	mcg	0.24	0.90		C	Kwekker Pharmaceuticals
224	Avatocin	No	Allergies	Pill	100	mg	0.65	1.40	Alcohol	A	TJR Labs
247	Acebutolol hydrochloride	No	Arthritis	Pill	400	mg	0.55	1.10		A	Pulman Labs
256	Dseurton	Yes	High blood pressure	Pill	175	mg	0.60	1.20	Sedatives	C	Cranston Pharmaceuticals
289	Levothyroxine	Yes	Thyroid disorders	Pill	25	mg	0.70	1.40		B	Vacer Labs
311	Dyotex	No	Tonsillitis	Bottle	2	tsp	0.25	1.05		D	Frankmeir Pharmaceuticals
366	Phalastat	No	Allergies	Bottle	1	tsp	0.75	1.60		A	Swinton Labs
398	Clonazepam	No	Epiliepsy	Pill	4	mcg	0.65	1.20		A	TJR Labs
412	Epronix	No	Pain	Pill	500	mg	0.85	1.50	Grapefruit	C	Vacer Labs
444	Syocil	Yes	Diabetes	Pill	120	mg	0.45	1.10		D	Kwekker Pharmaceuticals
452	Diazapam	Yes	Anxiety	Pill	5	mg	0.45	1.12		D	Esterman Pharmaceuticals
467	Glimepiride	Yes	Diabetes	Pill	2	mg	0.25	0.90		X	Nerman Pharmaceuticals
523	Xeroflarol	Yes	Acid reflux	Bottle	1	tsp	0.50	1.05		X	Wilper Labs
524	Cefixime	Yes	Antihistamine	Pill	400	mg	0.95	1.60	Sedatives	A	Ranston Pharmaceuticals
566	Quentix	Yes	High blood pressure	Pill	50	mg	0.50	1.25	Alcohol	A	Esterman Pharmaceuticals
587	Haloperidol	No	Diuretic	Pill	6	mcg	0.70	1.30		C	Valfer Pharmaceuticals
622	Tiron	No	Beta blocker	Pill	150	mcg	0.75	1.30		A	Opherman Labs
642	Montelukast sodium	Yes	Acne	Pill	10	mcg	0.32	0.90	Alcohol	B	Gilman Labs
644	Hyometadol	No	Asthma	Bottle	2	tsp	0.65	1.35		B	Ranston Pharmaceuticals
654	Warfarin Sodium	Yes	Bronchitis	Pill	4	mg	0.65	1.40		D	Swinton Labs
711	Nvalax	Yes	Depression	Pill	200	mg	0.30	0.90	Grapefruit	C	Pulman Labs

tblDrug

Figure 8 - Prescription drug inventory data

Because the data in this worksheet is not connected to the systems that the pharmacists and cashiers use to sell prescription drugs and other items, sales volume cannot be calculated electronically. Indeed, Vincent has always performed a bimonthly hand count of inventory and then built temporary columns in Excel to determine costs and profits. Paul has continued to use the same method for tracking inventory. Don suspects that it would be difficult to use this worksheet to determine how much inventory exists for any given item in the store or prescription drug in the pharmacy, and imagines that it is unlikely that Vincent could accurately account for any product in stock. Thus, not only is it difficult to determine the quantity or volume of drugs in stock and the overall value of the inventory, there is no way to determine which items sell well and which sell poorly. Consequently, Don guesses that reordering occurs only when a pharmacist or other employee notices that a product's inventory is low or nonexistent. Don wants the database to address all of these problems by providing timely and accurate information about sales volume, inventory levels, and drug expiration dates. Furthermore, he wants to incorporate a list of interactions, pregnancy risks, and suppliers of each drug in the inventory, as presently these must be cross-referenced with data found online or in proprietary databases.

Vincent explains to Don that his system has worked fine for the nearly two decades in which he has been in business. However, Don knows that Paul wants to automate all of the business processes that must occur to run the pharmacy. Paul knows that Vincent's systems do not work well enough to ensure timely and accurate information. Paul also knows that his father's systems contain many errors and as the owner of the pharmacy, Paul wants to increase the efficiency of these systems.

Don continues his research by talking with Maria Garcia, who manages all of the pharmacy's employment records, including applications, employee reviews, benefits, certifications, training, and salary and tax-related information. Maria tells Don that as the pharmacy grows, it is becoming increasingly difficult to manage employee records.
Currently, she maintains employee data in paper forms and in three programs. As shown in Figure 9, Maria uses Microsoft Outlook's contacts feature to maintain employee address and telephone information.

Employee contact information in Outlook

Figure 9 - Employee contact information in Outlook

Maria uses Excel for tracking key employee dates, such as date of birth, date of hire, termination date, and date of the employee's last personnel review. Maria also uses Excel for recording salary data. Because this worksheet contains sensitive information—in addition to salaries, the table lists employee Social Security numbers—Maria keeps this file in its own password-protected workbook that only she has access to. Figure 10 shows Maria's Excel worksheets.

Employee date and salary data in Excel worksheets

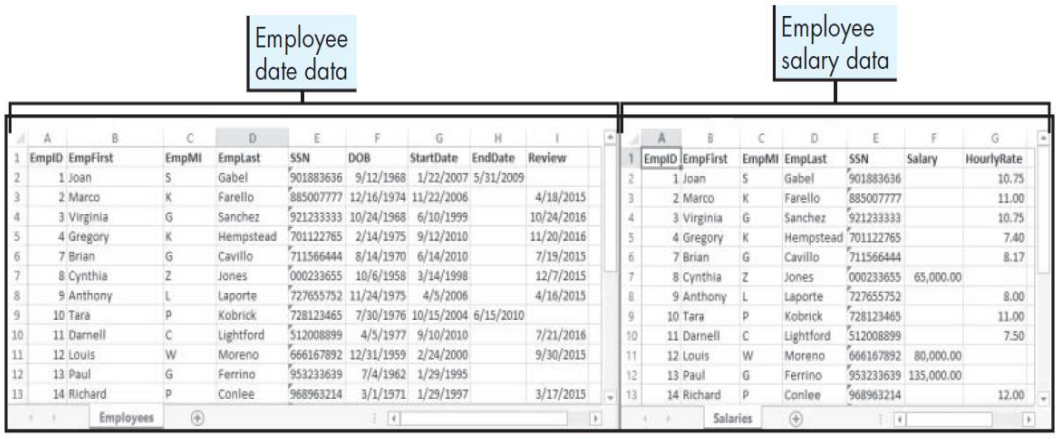


Figure 10 - Employee date and salary data in Excel worksheets

In addition to using Outlook and Excel, Maria keeps several employee records in Word, including a list of bilingual employees at 4Corners Pharmacy. If a bilingual employee is not on duty, pharmacists can refer to this list and call these employees when Spanish-speaking customers request assistance. She also keeps a Word document to track absenteeism.

Although absenteeism is not a large problem at 4Corners Pharmacy, Maria is aware that Paul plans to expand the company and documentation should be in place as the employee base grows. Figure 11 shows both of these documents.

Bilingual employees and absentee report

Bilingual Employees

Name	Language	Proficiency
Joan Gabel	Spanish	Fluent
Maria Garcia	Spanish	Fluent
Darnell Lightford	Spanish & Chinese	Fluent

Absentee/Tardy

Name	Date	Absent/Tardy
Dominique Latour	3/4/2016	Tardy
Shayla Jackson	10/15/2016	Absent
Gregory Hempstead	11/14/2016	Tardy

Figure 11 - Bilingual employees and absentee report

Maria is also responsible for monitoring staff development, a time-consuming task. Each pharmacist and pharmacy technician must maintain the proper license and certification as mandated by Colorado state law. Maria is responsible for ensuring that employees take the required classes to maintain their certifications. In addition, she must manage the employee performance reviews, which occur every 90 days, 6 months, or 12 months, depending on their job titles and the number of years they have worked at the pharmacy. She also serves as the pharmacy's benefits coordinator and manages requests for sick days and vacation, extended leaves of absence, pension benefits, and life insurance benefits.

Although Maria's various documents and worksheets seem to manage the information she needs, Don worries that these systems will not work well if the pharmacy grows as expected and adds additional stores and employees. Don sees that Maria could also benefit from storing and managing personnel data in a database.

After talking extensively with Vincent, Paul, and Maria, Don took a few days to talk to other employees in the store to understand and view the data that they use and need to perform their jobs. Upon concluding his discovery phase of the database design process, Don learned many things. First, there are many paper-based and handwritten methods of gathering data at 4Corners Pharmacy that use various forms, faxes, and memos. Second, many employees copy and paste data from one worksheet to another or import and export data from one program to another, thereby creating redundant data. This process occasionally leads to errors. Finally, Don suspects that the pharmacy is not collecting data that it needs to produce reports for outside consultants, such as the value of inventory in the pharmacy for insurance purposes or the value of business capital for local tax authorities. Don also sees that some data collected by the customer information form (Figure 2), such as insurance information, is never recorded in detail. As the business grows, it will be important to automate all of the processes at the pharmacy. Don knows that a DBMS will serve this function better than

hiring a large staff of people to manage the growing data needs of 4Corners Pharmacy and that using a DBMS will make Paul's business run more smoothly and more efficiently.

Concluding the first part of the discovery phase, Don details his findings to Paul. Table 1 lists the expectations of the database as Don understands them.

Table 1 - Preliminary database expectations for 4Corners Pharmacy

Preliminary database expectations for 4Corners Pharmacy

Department	Expectation
Pharmacy	• Eliminate paper forms • Provide quick access to prescription history • Print list of drug names • Ensure pharmacist consulted with customer about dosage instructions, allergies, and drug interactions • Print doctor contact information
Marketing	• Print customer mailing labels • Print physician mailing labels • Use customer data to gather data information about customer buying habits
Operations	• Track inventory • Identify poor-selling items
Human Resources	• Monitor staff development • Confirm certifications • Create employee performance reviews
Customers	• Print a year-end report of all prescriptions purchased by an individual or household
Accounting and Finance	• Determine profit and loss for individual sale items • Value the store's inventory • Track the business profit

Identifying Sources of Missing Data

Don's findings have identified the data collected by 4Corners Pharmacy and have given him a chance to evaluate how employees and managers use it. Another important part of the discovery phase is determining data that is missing and identifying the sources you can use to obtain that missing data. This part of the process is often difficult because you must ask the right questions of the right people to get the right answers. In interviews with key employees and managers, it is best to ask questions such as "What else would you like to know about the data?" and similarly worded inquiries. You might find that employees need to know certain things about the data that the current system cannot provide.

In his conversations with employees, Don learned a great deal about the kind of data that the DBMS must collect, organize, and manage. He also learned about data that various departments and employees want to collect but are not collecting presently. For example, Maria needs to obtain certain data about employee training and classes but has no way to do so. She also wants to have an electronic record of the classes that employees have taken and need to take, a way to determine which employees are bilingual, and scanned images of important employment documents such as the driver's licenses of employees who are authorized to make pharmacy deliveries.

Dividing the Existing and Missing Data into Tables

Now that the discovery phase is over, Don will create the tables to be used in the 4Corners Pharmacy database. Figure 12 details the four steps involved in this process.

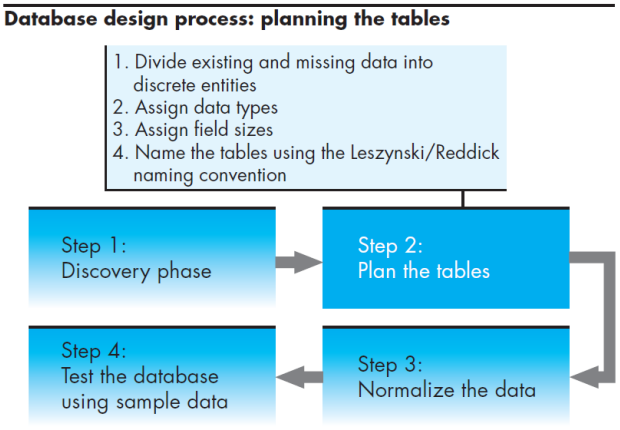


Figure 12 - Database design process: planning the tables

Don focuses first on the employee data that Maria collects because it is currently the most fragmented. Recall that Maria keeps personnel records in Outlook, Excel, Word, and on paper, with much of the data, such as the employee's name, appearing in all of these sources. Don lists the type of data that Maria identified about employees in the discovery phase:

- Name
- Age
- Years with company
- Address
- Position
- Job description
- Pay rate
- Annual review
- Training classes attended
- Other information (such as whether the employee is bilingual and attendance information)
- The type of prescription that can be filled by a pharmacist or a pharmacy technician

After listing the items Maria provided, Don analyzes the list provided by Maria and finds five problem areas.

1. **Missing Field for Telephone Numbers**
2. **Primary Key Field Needed**
3. **Fields Listed Are Too Broad**
4. **Age and Years with Company Should be Calculated Fields**
5. **Data Redundancy**

Don sees a similar issue with the Position field because multiple employees can share the same job title. To fix this, he creates a table to store all of the job descriptions at 4Corners Pharmacy. He also needs a way to identify which employee filled which prescription, so he creates another table to store information about prescriptions.

Don will complete this brainstorming process for each table that he creates, making sure to include all necessary fields, identify primary keys, split broad fields into smaller discrete components, and avoid data redundancy. He will create as many tables as necessary to avoid repeating any information in the database. The result of Don's brainstorming is shown in Table 2, with the fields for the Employee table and their corresponding data types.

Table 2 - Employee table

Field names and data types for the Employee table

Field Name	Data Type
Employee ID	Number
First Name	Short Text
Middle Initial	Short Text
Last Name	Short Text
Social Security Number	Short Text
Date of Birth	Date/Time
Start Date	Date/Time
End Date	Date/Time
Address	Short Text
City	Short Text
State	Short Text
ZIP	Short Text
Comments	Long Text
Phone	Short Text
Cell	Short Text
Salary	Currency
Hourly Rate	Currency
Review Date	Date/Time

Identifying and Eliminating Database Anomalies by Normalizing Data

By now you are familiar with database concepts and how to create tables and relationships between them. You have learned that good database design seeks to avoid data redundancy and inconsistent data. You also learned the specific rules for ensuring good database design using a process called **normalization**. Figure 1.30 details the third step of the database design process and illustrates how normalization fits into the plan. Normalization requires a series of steps, each one building on the previous. These steps are called normal forms, or first, second, and third normal forms, sometimes abbreviated as 1NF, 2NF, and 3NF. Notice in Figure 13 that 3NF is the final step necessary before the database is created and tested.

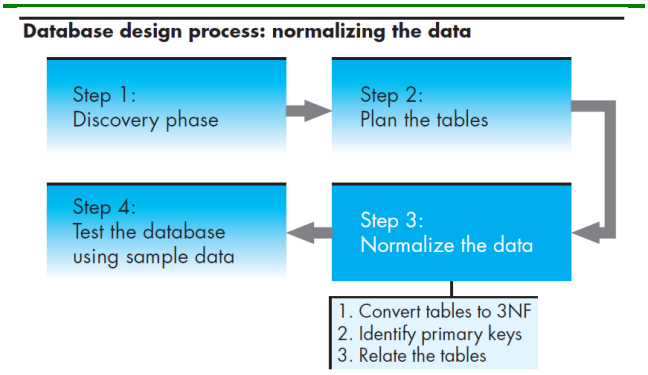


Figure 13 - Normalizing the Tables in the Database

Building the Database

Creating tables, entering data, verifying data, relating tables, documenting the database objects, backing up the database, repairing the database, and securing data.

Don needs to begin work on the database for 4Corners Pharmacy by creating the database in SQL Server and then creating the tables that will store the data the pharmacy needs to track **customers**, **prescriptions**, **drugs**, **employees**, **training classes**, **health insurance companies**, **doctors**, and **clinics**. The pharmacy's owner, Paul Ferrino, worked as a cashier, pharmacy technician, and pharmacist in his father's pharmacy before purchasing the pharmacy upon his father's retirement. Vincent Ferrino still works part time at the pharmacy as a pharmacist. Although Vincent's business was successful for more than two decades, his system for managing data about the pharmacy is obsolete. Paul wants to automate many of the processes at the pharmacy so he can better evaluate and operate the business.

Paul's top priority was hiring Don, who will create the system for the pharmacy, train users (that is, the employees) to use the system, and maintain and expand the system over time.

After careful analysis and preparation, Don formulated the database design for Paul's approval. Don reviews the database design with Paul to ensure that it is correct and meets the needs of the pharmacy. To make the design easier to understand, Don begins by presenting some of the table designs to Paul. Figure 14 shows the table that stores customer data, including each customer's name, address, phone number, date of birth, gender, account balance, preference for childproof caps, health plan, household information, and allergies. The table that stores customer information is linked to the table that stores details about the customer's prescriptions, to the table that stores details about the customer's health plan, and to the table that stores the details about the customer's household. These links are made by creating foreign keys in the related tables.

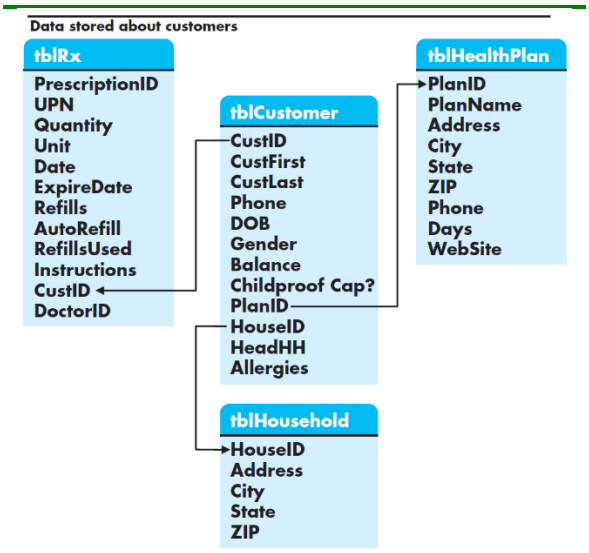


Figure 14 - Data stored about customers

"**Rx**" (℞) is often used as a short form for **prescription drug** in North America.

Don used the Leszynski/Reddick guidelines for naming database objects and created four tables: **tblCustomer**, **tblRx**, **tblHousehold**, and **tblHealthPlan**. Because tblCustomer is linked to the related tables tblRx, tblHousehold, and tblHealthPlan, Don can retrieve detailed information about each customer and the prescriptions they have filled, the household to which they belong, and the health plan that covers them.

These related tables will also enable him to list all the customers within a household and determine which drugs are being prescribed to which customers.

Figure 15 shows the table that stores prescription data (Rx), including the prescription's identification number, UPN, quantity ordered, unit of measurement, fill date, expiration date, refills authorization, preference for automatic refills, number of refills used, and the prescribing instructions.

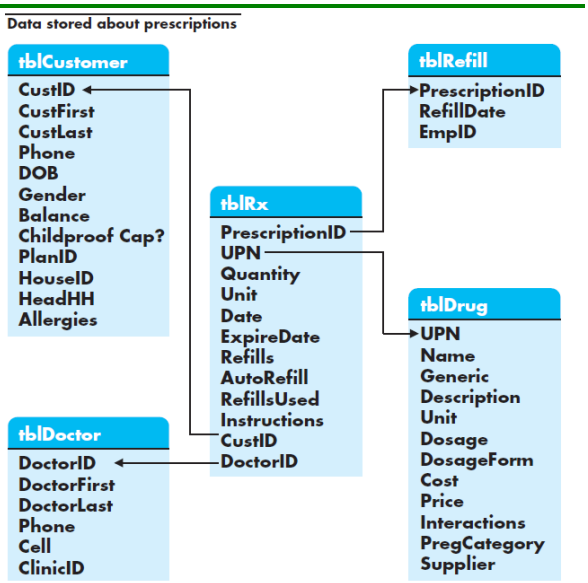


Figure 15 - Data stored about prescriptions

In addition, tblRx stores the identification numbers of the customer and the prescribing doctor. The table that stores prescription information is linked to the table that stores the details about the prescribing doctor, to the table that stores details about the customer, to the table that stores details about the drug, and to the table that stores information about refills using foreign keys in the related tables.

Figure 16 shows the table that stores data about employees at the pharmacy, including the employee's identification number, first name, middle initial, last name, Social Security number, date of birth, start date, termination date (if applicable), address, city, state, ZIP code, job identification number, phone number, cell phone number, salary or hourly rate of compensation, and next personnel review date. In addition, there is a field named Memo to store miscellaneous information about the employee, such as being part time or bilingual. The table that stores employee information is linked to the table that stores the details about the employee's training, to the table that stores information about the refills that the employee has filled, and to the table that identifies the different job titles at the pharmacy.

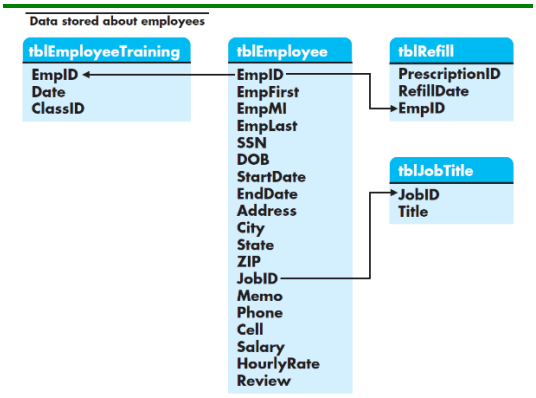


Figure 16 - Data stored about employees

Figure 17 shows the table that stores data about doctors, including the doctor's identification number, first and last names, phone number, cell phone number, and the identification number of the clinic at which the doctor works. The table that stores doctor data is linked to the table that stores details about the clinic at which the doctor works and to the table that stores details about prescriptions the doctor has written.

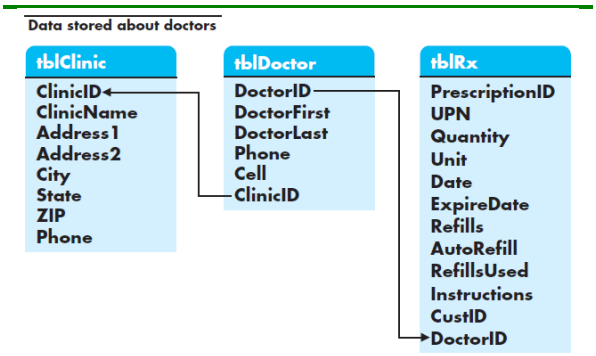


Figure 17 - Data stored about doctors

Figure 18 shows the table that stores data about the classes that pharmacy employees must take to maintain their professional certifications and other classes of interest. The table stores data about classes, including the class identification number, description, cost, renewal requirement (in years), and provider. In addition, there is a field to indicate whether the class is required. The table that stores data about classes is linked to the table that stores details about the employees who took the classes.

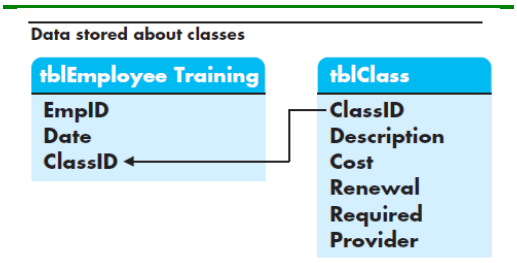


Figure 18 - Data stored about classes

As is customary in the design process, Paul and Don agreed to limit the scope of the new system to managing data about customers, prescriptions, drugs, employees, training classes, doctors, and clinics. As the pharmacy's staff and managers begin using the system, Paul and Don might reevaluate it and add additional functionality, such as managing the inventory of nonprescription items. If Paul opens additional pharmacy locations in the future, he and Don might plan for the system to go on a network so it is possible for employees and managers at each location to share data an and other resources across a network. The entire database design appears in Figure 18.

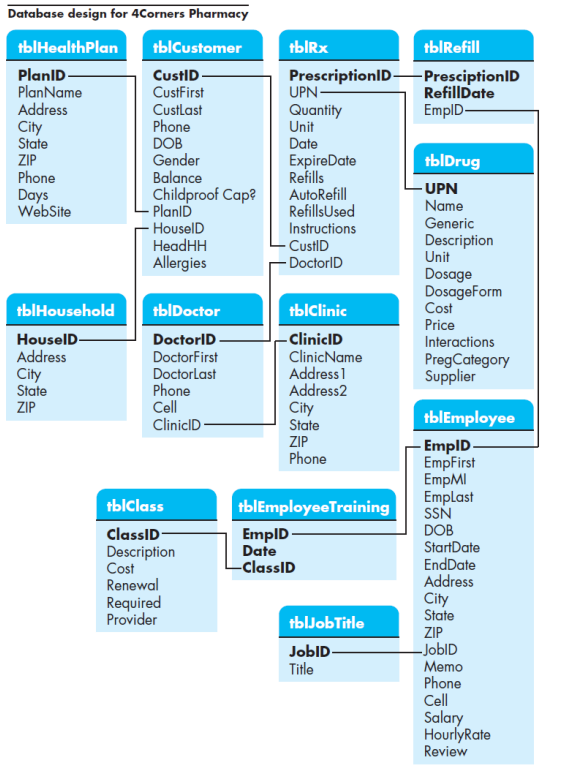


Figure 19 - Database design for 4Corners Pharmacy

Aragon Pharmacy database

Don just received a request from Paul to develop a database for a new pharmacy that Paul is opening in Montreal, Quebec, Canada. Because the data collected about customers, employees, and drugs is slightly different in Canada, Don decides to create the databases simultaneously, but separately, so he can account for the individual differences in each version. Don wants you to create the database for Aragon Pharmacy. (**You will create 12 tables in this exercise using the skills you have learned so far)**. He will provide you additional information, so you know how to account for the differences in the Canadian version of the database.

Level 1 – Complete the following:

1. Create a new SQL script named **1-create-aragon-pharmacy.sql**.

2. Create the following tables in the Aragon database using the method you prefer. create another SQL script named **2-create-aragon-schemas.sql**, and then another script named **3-create-aragon-tables.sql**

tblClass, tblCustomer, tblHealthPlan, and tblHousehold. Be certain that each table includes a primary key and the necessary fields. Also, be certain to set the field properties as necessary to collect the correct data. These tables have the same requirements as the ones in the 4Corners database, with the following exceptions:

- Canadian addresses include a street address, city, province, and postal code. Provinces are abbreviated using two uppercase letters. Add properties to the Province field so that the data entered by the user is converted to uppercase letters. Also, make sure that users cannot enter digits or any other characters except letters into the Province field.
- Postal codes in Canada have the following format: uppercase letter, number, uppercase letter, space, number, uppercase letter, number (for example, T6H 8U7). Add properties to the PostalCode field to ensure that data entry into this field is accurate and correctly formatted. You might want to consider using a pattern, such as >L0>L\0>L0.

3. Change the PlanID field in tblCustomer to a lookup field that displays the PlanID and PlanName values from tblHealthPlan but stores only the PlanID values. Sort values in ascending order by PlanID and adjust the lookup column widths as necessary to display the values that will be stored in these fields

TROUBLESHOOTING: In order to complete this task, you need to think about the record(s) that might already be entered in the table and decide whether it is okay to change the data integrity. Typically, you would want the data integrity to be updated.

4. Use Datasheet view and the data shown in Table 3 to create tblJobTitle. JobIDs are whole numbers and the Title field stores 30 characters. Use the Description property to document each field.

Table 3 Data for tblJobTitle

Data for tblJobTitle

JobID	Title
1	Owner
2	Pharmacist
3	Technician
4	Cashier
5	Manager

5. Examine the 4Corners database tblDoctor table shown in Figure 19. The fields for the doctor's first name and last name should store 30 characters each. The fields that store phone numbers should store 15 characters each and format values with an input mask that formats values with the following format: (###) ###-####. Note that these pound signs simply indicate numeric values. The ClinicID field is a foreign key to tblClinic.

6. Create tblDrug shown in Figure 18. Make sure that the table includes a primary key and the necessary fields. Also, make sure that you set the field properties as indicated in the following list to collect the correct data. Fields should include appropriate names.

- Canadian drugs do not have UPNs. Instead, drugs are uniquely identified using a Drug Identification Number (DIN). The DIN is a unique, eight-digit value that identifies each drug sold in Canada. DINs are not used in calculations.
- Drug names do not exceed 30 characters.
- Generic drugs are indicated by selecting a check box.

- The Description field collects data about the drug, such as contraindications, generic equivalents, and recommended dosage.
- The Unit field stores information about the unit of measure for a drug, such as pill or bottle, and requires 10 characters.
- The Dosage field stores information about the drug's strength and requires 10 characters. The Dosage field is not used in calculations.
- The DosageForm field stores information about the unit of measure for the drug strength, such as mg (for milligrams) or mcg (for micrograms). Dosage abbreviations do not exceed 20 characters.
- The Cost and Price fields store the cost and price, respectively, for one unit of the drug. The Canadian government regulates the prices that pharmacies can charge for drugs so pharmacies cover their overhead costs by charging a separate dispensing fee of $10 to $12 per prescription. Dispensing fees are determined by the pharmacy and set individually for each drug.
- The Interactions field stores information about possible drug interactions and possible reactions.
- Canada does not track pregnancy risk categories like pharmacies in the United States.
- The Supplier field identifies the drug company or manufacturer from which a drug was purchased.

7. Use the following information to create tblEmployee shown in Figure 19. Make sure that the table includes a primary key and the necessary fields. Also, make sure that you set the field properties where indicated in the following list to collect the correct data. Fields should include appropriate names.

- EmpIDs are assigned by the pharmacy using unique numbers. EmpIDs are used to relate tables in the database.
- The EmpFirst and EmpLast fields do not exceed 30 characters each. The EmpMI field stores up to two characters.
- Canada issues Social Insurance numbers (SINs) instead of Social Security numbers. An SIN is a nine-digit number displayed with the following format: ###-###-###.
- The DOB (date of birth), StartDate, EndDate, and Review fields should store two-digit months and days and four-digit years in the format ##/##/####.
- Make sure that the table stores a province and postal code for employees instead of a state and ZIP code.
- JobID is a foreign key in tblJobTitle. This field stores values as numbers.
- Memo field stores other information about employees that might exceed 255 characters.
- The Phone and Cell fields should store 15 characters in the format of (###) ###-####. Area code entry is required.
- Employees are paid an annual salary or an hourly rate.
- There is an annual review date for each employee based on his or her hire date.

8. Use the following information to create tblRx shown in Figure 15. Make sure that the table includes a primary key and the necessary fields. Also, make sure that you set the field properties as indicated in the following list to collect the correct data. Fields should include appropriate names.

- PrescriptionIDs are assigned as records are added to the table.
- The DIN is a unique, eight-digit value that identifies each drug sold in Canada. DINs are not used in calculations.
- The Quantity field stores the amount of medication dispensed and it is a numeric field that might contain decimal places.
- The Unit field stores information about the unit of measure for a drug, such as mg or ml, and requires 10 characters.
- The Date and ExpireDate fields store the date of the prescription and the prescription's expiration date, respectively. Both fields should store two-digit months and days and four-digit years in the format ##/##/####.

- The Refills field indicates the number of refills authorized by the prescribing doctor.
- The AutoRefill field indicates the customer's preference (yes or no) for automatic refills. The default value is not to order auto refills unless requested by the customer.
- The RefillsUsed field stores the number of refills a customer has used.
- The Instructions field stores medication directions and does not exceed 50 characters.
- The CustID field is a foreign key in tblCustomer
- The DoctorID field is a foreign key in tblDoctor. Both fields store numbers.

9. Create a new SQL script named **4-create-data-integrity.sql**. Create the relationships between all tables.

While you were working on creating the database and tables for the Aragon Pharmacy, Don created the tblJobTitle, tblDoctor, tblDrug, tblEmployee, and tblRx tables in the 4Corners database.

Populating the Database Tables

The 4Corners database now contains eight tables. The next step is to load the tables with data, also known as populating the database. Don could enter records into each table by typing in Datasheet view. However, because much of the data he needs to load is stored in sources found during the discovery phase, Don can import this data into the tables.
There are several ways to import data into a database. Don will use two of them, which includes copying and pasting records from another database table and importing data from an Excel workbook.

Importing Data from an Excel Workbook and CSV files
In some cases, you might need to import data stored in an Excel workbook into a table. If you have not yet created the table in the database, you can import the data and create the table at the same time. In the discovery phase, Don received some Excel files from Paul containing data that he needs to store in the database. One of those files, tblClinic.xlsx, contains data about the clinics that employ the doctors who write prescriptions for 4Corners Pharmacy customers. Don decides to create tblClinic by importing the data from the Excel file.

Level 2 – Complete the following

In the Level 1, you created the database for Aragon Pharmacy and created the nine tables: tblClass, tblCustomer, tblDoctor, tblDrug, tblEmployee, tblHealthPlan, tblHousehold, tblJobTitle, and tblRx. The only table that contains data at this point is tblJobTitle. Don asks you to create the remaining tables in the database (tblClinic, tblEmployeeTraining, and tblRefill) and to populate the database with data using various sources that he has provided to you. You also need to relate the appropriate tables to each other.

1. Populate the existing database tables. The files you need are saved in the Data Source folder. Remember that you might need to make adjustments as necessary for importing to work.
    a. Use the file hbpClass.xlsx to populate tblClass.
    b. Use the file hbpHhold.xlsx to populate tblHousehold.
    c. Use the file hbpCust.xlsx to populate tblCustomer.
    d. Use the file hbpHPlan.xlsx to populate tblHealthPlan.
    e. Use the file hbpDrug.xlsx to populate tblDrug.
    f. Use the file hbpRx.xlsx to populate tblRx.
    g. Use the file hbpEmp.xlsx to populate tblEmployee.
    h. Import records in the Doctor table from the hbpDoc.accdb database into tblDoctor in the Aragon.accdb database.

2. Import data from the file hbpClin.xlsx into tblClinic table. Be certain the table meets the following requirements:
- Add a ClinicID field and make it the table's primary key.
- The ClinicName cannot exceed 50 characters.

- The Address1, Address2, and City fields cannot exceed 40 characters.
- The default value for the Province field is "QC" and all values entered into this field must be uppercase and two characters in length.
- Values in the PostalCode field must appear in the following format: uppercase letter, number, uppercase letter, space, number, uppercase letter, number (for example, T6H 8U7).
- The Phone field stores 15 characters, displays values in the format of (###) ###-####, and area code entry is required.
- All fields must have appropriate captions and field descriptions.

3. Create tblEmployeeTraining using the following information:
- The EmpID and ClassID fields store numbers with no decimal places and are foreign keys.
- The Date field stores the date of the training session. Display the date using the format ##/##/####.
- Set the composite primary key, add captions to appropriate fields, and add appropriate field descriptions to all fields.
- Use the hbpEmpTr.xlsx file to populate the table.

4. Create tblRefill using the following information:
- The PrescriptionID and the EmpID fields store numbers with no decimal places and are foreign keys.
- The RefillDate field stores the date the prescription was refilled. Display the date using the format ##/##/####.
- Set the composite primary key, add captions to appropriate fields, and add appropriate field descriptions to all fields.
- Use the hbpRefl.xlsx file to populate the table.

5. Create an index on the CustLast field in tblCustomer and on the Name field in tblDrug.
6. Create the appropriate relationships in the database.

Maintaining and Securing a Database

The 4Corners database now contains 12 related tables that store the data needed to dispense prescriptions to customers of the pharmacy. Don populated the tables with the existing data he gathered during the discovery phase. It might be tempting to think that the database is "finished" at this point, but it is not. Don knows that there are other tasks he needs to complete before bringing the database online for users. These tasks include securing the database, backing up and restoring the database, archiving data, and compacting and repairing the database. He also needs to document the database design for himself and other database users. Collectively, these tasks are known as maintaining a database. Most organizations assign these tasks to **database administration**, also known as **DBA**. In many organizations, DBA is a group that is responsible for designing, maintaining, and securing the database. In other organizations, DBA is assigned to an individual who is charged with these tasks; sometimes, this individual is called the **database administrator**. At 4Corners Pharmacy, Don is the database administrator. The database administrator sets the security and other features of a database in addition to setting options for individual users and groups of users.

Backing Up a Database

Create a backup copy. **Backing up** a database creates a copy of the database that you can restore in the event of a loss. A loss might be caused by a power failure or hard disk crash, a user who maliciously or accidentally deletes database objects, or any other situation that affects the records and objects in the database. Most database administrators perform regular database backups for each day the database is used. A good rule of thumb is to schedule database backups based on the amount of data loss that you can manage. If you cannot manage reentering all of the transactions completed in a week's time, you should schedule backups more frequently than once a week. It is not uncommon for an organization to back up a database nightly, after all users have completed their work.

After creating a backup copy of the database, it is important to store the copy in a fireproof location, preferably at a location outside the pharmacy. In most cases, a backup is created on external media, such as a CD, DVD, USB Flash Drive, or external hard drive. Many organizations store multiple backup copies of their databases in multiple locations. This protects them from losing their business data in the event that a fire or other disaster destroys their company headquarters.

Human Resources

Maria Garcia is the human resources manager for 4Corners Pharmacy. Her responsibilities involve hiring and firing employees and making sure they complete the necessary employment paperwork. In addition, she manages employee training, schedules periodic job reviews, and helps analyze compensation and other employee information.

Most of the training sessions that employees of 4Corners Pharmacy attend involve mandatory certification for cardiopulmonary resuscitation (CPR) and the use of defibrillators. Maria needs a way to make sure all employees attend required classes and receive annual certification, as necessary.

Until now, Maria has maintained employee and training information on printed forms that she stores in a filing cabinet. However, it is often inconvenient and time consuming to access and update these forms, especially to answer management questions that require her to analyze data. Maria plans to automate the records by using the 4Corners.accdb database to track employee and training information.

Paul asks Maria to create some views:

1. Paul Ferrino, the owner of 4Corners Pharmacy, periodically reviews the wages he pays to his employees so he can budget for upcoming pay raises. He has recently increased the salaries of the two pharmacists at 4Corners, but he needs to check the hourly rates he pays to the other employees. He asks Maria to list the wages for pharmacy technicians and cashiers, ranked from highest to lowest so he can quickly see the full range of pay for non-salaried employees. Maria creates a view and saves it as **HourlyRateAnalysisView.**

2. Maria has another problem she wants to work on next. Paul asked Maria to look into the possibility of scheduling at least one employee in the pharmacy who is fluent in Spanish for all shifts. She thinks she can create a query that uses broader criteria than the exact match criteria she's used so far to help with Paul's scheduling request.
   Maria is ready to turn to Paul's scheduling request. Because many of the pharmacy's clientele speak Spanish, he wants to schedule at least one Spanish-speaking employee for each shift. If an employee speaks Spanish (or any other language besides English and French), it is noted in the Comments field of tblEmployee. Because the Memo field data type allows for lengthy text or combinations of text and numbers, it is often best using wildcards in criteria that involve comments fields. Maria **creates a view and saves it as SpeakSpanishView**

3. Paul asks Maria to work on a view that returns the hourly rate summary that returns the minimum, and maximum of hourly rate related to the job position. Maria creates a view and saves it as **HourlyRateSummaryView.**

Evaluating Data Using Special Types of Views

Now that Maria Garcia is familiar with the 4Corners database, she wants to use it to automate many of her human resources tasks. To conserve gasoline and other vehicle maintenance costs, employees have asked Maria to help them coordinate carpools. One of her tasks will be to help employees determine how they can share rides to work. She will create a carpool list identifying those employees who live in the same city.

Find Duplicates values in the fields you select.

For example, you might have accidentally entered the same employee or customer name twice with two different IDs. Newly transferred data from a spreadsheet or other source to a database table also often contains duplicate information, as does a database that was not set up correctly. In these cases, you can use SQL Statements to identify duplicate records. You can then edit the data in the table as necessary.
For example, you can find customers who live in the same city so you can coordinate sales calls, compare costs of vendors who supply the same products, or alert employees enrolled in the same training session of a schedule change.

Maria wants to identify employees who live in the same city so they can create carpools and share rides to work. Because 4Corners Pharmacy is located in southwestern Colorado and borders four states (Arizona, Colorado, New Mexico, and Utah), employees live in many surrounding cities. Maria can create a list of employees who live in the same city. She can post this list so that employees can form carpools.

Avoiding Duplicate Records
If you expect duplicates to be a problem, the best way to solve the problem is to use good database design. For example, the 4Corners database allows only one person to be listed as the head of household in the HeadHH field in tblCustomer, and addresses are listed only once in a separate table named tblHousehold. The tblCustomer and tblHousehold tables share a common field, HeadHH which allows the two tables to be used in queries. By setting a criterion that HeadHH must be Yes, the problem of duplicates is eliminated.

Using Queries to Find Unmatched Records
Another useful query is the find unmatched records. This type of select query compares the records in two specified tables or recordsets and finds all the records in one table or query that have no related records in a second table or query. For example, you could find all customers who have not ordered any products, or vendors who provide services you no longer use. Identifying these unmatched records means you can then contact inactive customers to solicit business or delete records for vendors who no longer serve your needs.

At 4Corners Pharmacy, Paul Ferrino has set a store policy that all employees must maintain certifications in adult, infant, and child CPR and in defibrillator use. One of Maria's new responsibilities is to monitor training to make sure employees are enrolling in the required certification classes. This information is stored in the tblEmployeeTraining table. However, Maria is concerned that some employees may not have completed any training, and therefore would not have a record in tblEmployeeTraining. To identify these employees, she can use the unmatched values between two tables.

Using Parameter Values
When you need to run a query multiple times with changes to the criteria, you can enter a parameter value. A parameter value is a phrase, usually in the form of a question or instruction, such as @jobID. The parameter value serves as a prompt to the user to enter a value.

Maria decides to create a user-define function (UDF) with parameter by specifying a parameter value in the @JobID field. She begins creating the function using tblEmployee. She chooses EmpLast, EmpFirst, Phone, Cell, JobID, and EndDate as fields for the query. She sorts the data in alphabetical order by last name. She remembers to include the EndDate criterion Is Null. She saves the query as **getSubstituteListFn.**

Analyzing Data from More Than One Table
powerful advantage of creating SQL objects like views and functions is that they let you combine records from two or more tables to display only the information you need. For example, Maria needs data from both tblEmployee and tblEmployeeTraining to answer questions about employee certification training.

Maria wants to start investigating employee training by producing a list of employees who have taken certification classes, including the date and class ID. To do so, she needs fields from two tables: tblEmployee and tblEmployeeTraining. She starts creating a query, selecting the field lists for tblEmployee and tblEmployeeTraining. She adds the EmpLast and EmpFirst fields from tblEmployee and the Date and ClassID fields from tblEmployeeTraining. She notes that the tables have a one-to-many relationship, and that they are

linked by the common EmpID field. Maria saves the query as EmployeeClassesView, and then runs it. The results list employees who have attended training classes. Because some employees have taken more than one class, they are listed more than once.

Maria can use this list herself, but she needs a similar list that includes the class description instead of the class ID. She'll post this list on the employee bulletin board as a reminder about the certification classes. Now that she's created EmployeeClassesView, she can save time by using it as the basis for a new query that lists class descriptions. The EmployeeClassesView query already has fields from tblEmployee and tblEmployeeTraining, including ClassID. However, neither table contains the Description field, which provides a description or title of each class. Because only tblClass contains that field, she needs to include the tblClass table in the new query to list class descriptions instead of class IDs.

Maria starts creating a view and adds thw EmployeeClassesView and tblClass to the query. Because she wants to list employee names, the date they attended a class, and the class description, she inserts EmpLast, EmpFirst, and Date from the EmployeeClassesView and Description from the tblClass. Then, she and sorts the EmpLast field in ascending order. She saves the query as EmployeeClassesDescriptionView, and then runs it. The results now include the descriptions of the classes instead of the ClassIDs, which is more meaningful to the employees who will view the list.

The EmployeeTrainingView produces a list that includes employees and the classes they've attended and employees who have not attended any classes. Maria wants to base a new query on EmployeeTrainingView to see if employees taking required classes are up to date on their certifications in Adult CPR, Child/Infant CPR, and Defibrillator Use.
Each type of certification needs to be renewed at different intervals, so she needs to set the criteria carefully to produce the results she needs.

Maria decides to save EmployeeTrainingView as UpToDateView and then modify it. The view already includes field lists for tblEmployee, tblEmployeeTraining, and tblClass, with outer joins specified so that all employees are listed in the results, even if they have not attended a class. The EmpLast and EmpFirst fields from tblEmployee and the Description field from tblClass already appear in the query. To determine whether an employee's certification is up to date, she needs the Date field from tblEmployeeTraining and she inserts this field between the EmpFirst and Description fields.

To determine whether a particular class is required for certification, she needs to include the Required field from tblClass. She also decides to include the ClassID field from tblClass to make setting up the criteria easier—all she will have to do is specify the ID of the class rather than the long description. She adds the three fields—Date, Required, and ClassID—to the view.

Next, she will specify the criteria for selecting information about only the classes required for certification. Pharmacy employees must take the five classes listed in Table 4.

Table 4 - Required classes for pharmacy employees

Required classes for pharmacy employees

ClassID	Description	Renewal in Years
1	Adult CPR	0
2	Child/Infant CPR	0
3	Adult CPR Recertification	1
5	Defibrillator Use	1
6	Child/Infant CPR Recertification	1

The first two classes—Adult CPR and Child/Infant CPR—are the comprehensive classes employees take to receive CPR certification for the first time. Employees complete these comprehensive classes only once, and do not need to renew them. Instead, they need to complete classes with IDs 3 and 6, which are refresher courses for recertification. ClassID 5 provides certification for defibrillator use and must be taken every year.

First, Maria adds a criterion to determine which employees are current in their Adult CPR certification. These employees would have completed the Adult CPR or the Adult CPR Recertification classes (ClassIDs 1 or 3) in the past year. She can use the logical operator OR to select employees who have completed ClassID 1 or 3.

To narrow the criteria and select only those employees who have taken these courses in the past year, Maria can use the Between…And comparison operator, which you use to specify two Date fields. She needs to specify the time period from January 1, 2019 to December 31, 2019, so she types Between '1/1/2019' And '12/31/2019' in the Where clause for the Date field.

Calculating Statistical Information

Recall that you can use the Summary (Aggregate) Functions in a query to calculate statistical information such as totals and averages in query results. You can group, or aggregate, the records so that your results are for records in each group that meet the selection criteria. Table 5 lists the aggregate functions available in SQL Server.

Table 5 - Aggregate functions

Aggregate functions

Aggregate Function	Operation
Avg	Average of the field values for records meeting the criteria within each group
Count	Number of records meeting the criteria in each group
Max	Highest field value for selected records in each group
Min	Lowest field value for selected records in each group
Sum	Total of the field values for the selected records
StDev	Standard deviation for the selected records
Var	Statistical variance for the selected records
First	Returns the value from the first row encountered in the group; results may be unpredictable because the result depends on the physical sequence of the stored data
Last	Returns the value for the last row encountered for the group; results may be unpredictable because the result depends on the physical sequence of the stored data

Maria needs to calculate the minimum, maximum, and average hourly rates for each job ID. She creates a view, adding the JobID and HourlyRate fields from the tblEmployee table and setting criteria to display only records for technicians and cashiers, so she adds the criteria 3 or 4 in the JobID field. Because she wants to calculate three types of statistics on the HourlyRate field, she adds that field to the design grid two more times. She names the view MaxMinAvgHourlyRateView, and then runs it.

Maria needs to calculate how long each employee has worked for 4Corners Pharmacy. Paul will use this information as he reviews retirement account plans and sets eligibility rules. Maria needs to calculate the years of service each employee has provided 4Corners and decides to create a function with parameter. She saves this function as YearsOfServiceFn.