

COMP1111

Week 13

Table Design Concepts

- <http://www.roma1.infn.it/people/barone/metinf/relational-databases-sandbox-handout.pdf>
-
- <https://support.office.com/en-US/Article/Database-design-basics-1eade2bf-e3a0-41b5-ae6-d2331f158280?ui=en-US&rs=en-US&ad=US>

Table Design Concepts

- A properly designed database provides you with access to up-to-date, accurate information.
- Because a correct design is essential to achieving your goals in working with a database, investing the time required to learn the principles of good design makes sense.
- In the end, you are much more likely to end up with a database that meets your needs and can easily accommodate change.
- We will discuss guidelines for planning a database. You will learn how to decide what information you need, how to divide that information into the appropriate tables and columns, and how those tables relate to each other.

- Certain principles guide the database design process.
- The first principle is that duplicate information (also called redundant data) is bad, because it wastes space and increases the likelihood of errors and inconsistencies.
- The second principle is that the correctness and completeness of information is important.
- If your database contains incorrect information, any reports that pull information from the database will also contain incorrect information.
- As a result, any decisions you make that are based on those reports will then be misinformed.

- A good database design is, therefore, one that:
 - Divides your information into subject-based tables to reduce redundant data.
 - Provides Access with the information it requires to join the information in the tables together as needed.
 - Helps support and ensure the accuracy and integrity of your information.
 - Accommodates your data processing and reporting needs.

The Design Process

- **Determine the purpose of your database**
 - This helps prepare you for the remaining steps.
- **Find and organize the information required**
 - Gather all of the types of information you might want to record in the database, such as product name and order number.
- **Divide the information into tables**
 - Divide your information items into major entities or subjects, such as Products or Orders. Each subject then becomes a table.
- **Turn information items into columns**
 - Decide what information you want to store in each table. Each item becomes a field, and is displayed as a column in the table. For example, an Employees table might include fields such as Last Name and Hire Date.

The Design Process

- **Specify primary keys**
 - Choose each table's primary key. The primary key is a column that is used to uniquely identify each row. An example might be Product ID or Order ID.
- **Set up the table relationships**
 - Look at each table and decide how the data in one table is related to the data in other tables. Add fields to tables or create new tables to clarify the relationships, as necessary.
- **Refine your design**
 - Analyze your design for errors. Create the tables and add a few records of sample data. See if you can get the results you want from your tables. Make adjustments to the design, as needed.
- **Apply the normalization rules**
 - Apply the data normalization rules to see if your tables are structured correctly. Make adjustments to the tables, as needed.

Determining the purpose of your database

- It is a good idea to write down the purpose of the database on paper — its purpose, how you expect to use it, and who will use it.
- For a small database for a home based business, for example, you might write something simple like "The customer database keeps a list of customer information for the purpose of producing mailings and reports."
- If the database is more complex or is used by many people, as often occurs in a corporate setting, the purpose could easily be a paragraph or more and should include when and how each person will use the database.
- The idea is to have a well developed mission statement that can be referred to throughout the design process. Having such a statement helps you focus on your goals when you make decisions.

Finding and organizing the required information

- To find and organize the information required, start with your existing information.
- For example, you might record purchase orders in a ledger or keep customer information on paper forms in a file cabinet.
- Gather those documents and list each type of information shown (for example, each box that you fill in on a form).
- If you don't have any existing forms, imagine instead that you have to design a form to record the customer information.
- What information would you put on the form? What fill-in boxes would you create?
- Identify and list each of these items. For example, suppose you currently keep the customer list on index cards. Examining these cards might show that each card holds a customers name, address, city, state, postal code and telephone number. Each of these items represents a potential column in a table.

Finding and organizing the required information

- As you prepare this list, don't worry about getting it perfect at first.
- Instead, list each item that comes to mind. If someone else will be using the database, ask for their ideas, too. You can fine-tune the list later.
- Next, consider the types of reports or mailings you might want to produce from the database.
- For instance, you might want a product sales report to show sales by region, or an inventory summary report that shows product inventory levels. You might also want to generate form letters to send to customers that announces a sale event or offers a premium.
- Design the report in your mind, and imagine what it would look like. What information would you place on the report? List each item.
- Do the same for the form letter and for any other report you anticipate creating.

Finding and organizing the required information

- A key point to remember is that you should break each piece of information into its smallest useful parts.
- In the case of a name, to make the last name readily available, you will break the name into two parts — First Name and Last Name.
- To sort a report by last name, for example, it helps to have the customer's last name stored separately. In general, if you want to sort, search, calculate, or report based on an item of information, you should put that item in its own field.

Finding and organizing the required information

- Think about the questions you might want the database to answer.
- For instance, how many sales of your featured product did you close last month? Where do your best customers live? Who is the supplier for your best-selling product? Anticipating these questions helps you zero in on additional items to record.
- After gathering this information, you are ready for the next step.

Dividing the information into tables

- To divide the information into tables, choose the major entities, or subjects.
- For example, after finding and organizing information for a product sales database, the preliminary list might look like this:



Customers	Products
Name	Product Name
Address	Price
City, State, Zip	Units in Stock
Send E-mail	Units on Order
Salutation	
E-mail Address	
Suppliers	Orders
Company Name	Order Number
Contact Name	Salesperson
Address	Order Date
City, State, Zip	Product
	Quantity
	Price
	Total

Dividing the information into tables

- The major entities shown here are the products, the suppliers, the customers, and the orders. Therefore, it makes sense to start out with these four tables: one for facts about products, one for facts about suppliers, one for facts about customers, and one for facts about orders.
- Although this doesn't complete the list, it is a good starting point. You can continue to refine this list until you have a design that works well.
- When you first review the preliminary list of items, you might be tempted to place them all in a single table, instead of the four shown in the preceding illustration. You will learn here why that is a bad idea.

Turning information items into columns

- To determine the columns in a table, decide what information you need to track about the subject recorded in the table.
- For example, for the Customers table, Name, Address, City-State-Zip, Send e-mail, Salutation and E-mail address comprise a good starting list of columns.
- Each record in the table contains the same set of columns, so you can store Name, Address, City-State-Zip, Send e-mail, Salutation and E-mail address information for each record.
- For example, the address column contains customers' addresses. Each record contains data about one customer, and the address field contains the address for that customer.

Turning information items into columns

- Once you have determined the initial set of columns for each table, you can further refine the columns.
- For example, it makes sense to store the customer name as two separate columns: first name and last name, so that you can sort, search, and index on just those columns.
- Similarly, the address actually consists of five separate components, address, city, state, postal code, and country/region, and it also makes sense to store them in separate columns.
- If you want to perform a search, filter or sort operation by state, for example, you need the state information stored in a separate column.

Turning information items into columns

- You should also consider whether the database will hold information that is of domestic origin only, or international, as well.
- For instance, if you plan to store international addresses, it is better to have a Region column instead of State, because such a column can accommodate both domestic states and the regions of other countries/regions.
- Similarly, Postal Code makes more sense than Zip Code if you are going to store international addresses.

Turning information items into columns

- **Don't include calculated data**

- In most cases, you should not store the result of calculations in tables. Instead, you can have Access perform the calculations when you want to see the result.
- The subtotal itself should not be stored in a table, your forms/reports could handle this information, as could a calculated field in a query.

- **Store information in its smallest logical parts**

- You may be tempted to have a single field for full names, or for product names along with product descriptions.
- If you combine more than one kind of information in a field, it is difficult to retrieve individual facts later.
- Try to break down information into logical parts; for example, create separate fields for first and last name, or for product name, category, and description.

Turning information items into columns

Customers Name Address City, State, Zip Send E-mail Salutation E-mail Address	Products Product Name Price Units in Stock Units on Order
Suppliers Company Name Contact Name Address City, State, Zip	Orders Order Number Salesperson Order Date Product Quantity Price Total

Customers Name Address City Region Postal Code Country Send E-mail Salutation E-mail address	Products Product Name Unit Price Units in Stock Units on Order Quantity per Unit
Suppliers Company Name Contact Name Address City Region Postal Code Country Phone	Orders Order Number Salesperson Order Date Product Quantity Price

Specifying primary keys

- Each table should include a column or set of columns that uniquely identifies each row stored in the table.
- This is often a unique identification number, such as an employee ID number or a serial number.
- In database terminology, this information is called the **primary key** of the table.
- Access uses primary key fields to quickly associate data from multiple tables and bring the data together for you.

Specifying primary keys

- If you already have a unique identifier for a table, such as a product number that uniquely identifies each product in your catalog, you can use that identifier as the table's primary key — but only if the values in this column will always be different for each record.
- You cannot have duplicate values in a primary key. For example, don't use people's names as a primary key, because names are not unique. You could easily have two people with the same name in the same table.
- A primary key must always have a value. If a column's value can become unassigned or unknown (a missing value) at some point, it can't be used as a component in a primary key.

Specifying primary keys

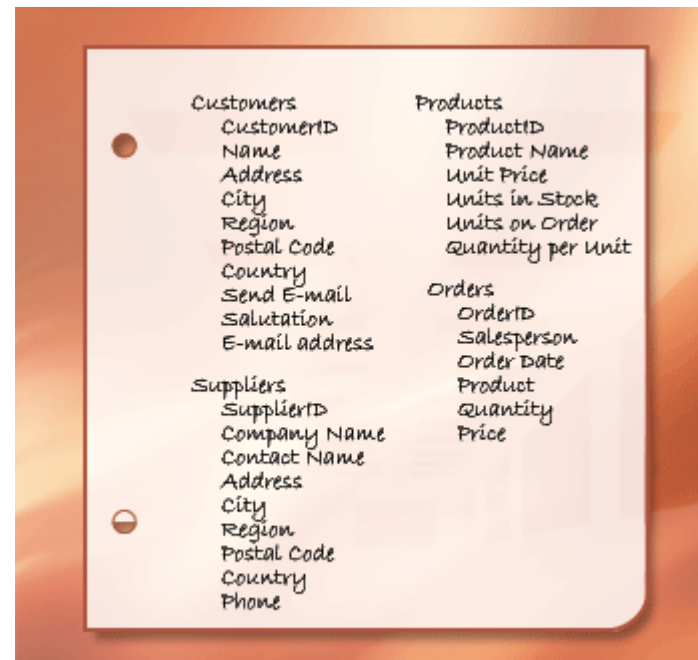
- You should always choose a primary key whose value will not change.
- In a database that uses more than one table, a table's primary key can be used as a reference in other tables.
- If the primary key changes, the change must also be applied everywhere the key is referenced.
- Using a primary key that will not change reduces the chance that the primary key might become out of sync with other tables that reference it.

Specifying primary keys

- Often, an arbitrary unique number is used as the primary key.
- For example, you might assign each order a unique order number. The order number's only purpose is to identify an order. Once assigned, it never changes.
- If you don't have in mind a column or set of columns that might make a good primary key, consider using a column that has the AutoNumber data type.
- When you use the AutoNumber data type, Access automatically assigns a value for you.
- Such an identifier is factless; it contains no factual information describing the row that it represents.
- Factless identifiers are ideal for use as a primary key because they do not change. A primary key that contains facts about a row — a telephone number or a customer name, for example — is more likely to change, because the factual information itself might change.

Specifying primary keys

- For the product sales database, you can create an AutoNumber column for each of the tables to serve as primary key: ProductID for the Products table, OrderID for the Orders table, CustomerID for the Customers table, and SupplierID for the Suppliers table.



A graphic of a notepad with a light beige background and a dark orange border. The notepad has two binder rings on the left side. It contains a list of database tables and their fields, organized into four columns. The tables are: Customers, Products, Suppliers, and Orders. The fields are listed under each table name.

Customers	Products	Suppliers	Orders
CustomerID	ProductID	SupplierID	OrderID
Name	Product Name	Company Name	Salesperson
Address	Unit Price	Contact Name	Order Date
City	Units in Stock	Address	Product
Region	Units on Order	City	Quantity
Postal Code	Quantity per Unit	Region	Price
Country		Postal Code	
Send E-mail		Country	
Salutation		Phone	
E-mail address			

Creating the table relationships

- Now that you have divided your information into tables, you need a way to bring the information together again in meaningful ways.
- For example, the following form includes information from several tables.

Creating the table relationships

1. Information in this form comes from the Customers table...
2. ...the Employees table...
3. ...the Orders table...
4. ...the Products table...
5. ...and the Order Details table.

Access is a relational database management system. In a relational database, you divide your information into separate, subject-based tables. You then use table relationships to bring the information together as needed.

Orders

Bill To: Alfreds Futterkiste
Obere Str. 57
Berlin 12209
Germany

Ship To: Alfreds Futterkiste
Obere Str. 57
Berlin 12209
Germany

Salesperson: Suyama, Michael

Ship Via: ☒ Speedy ☐ United ☐ Federal

Order ID: 10643 **Order Date:** 25-Aug-1997 **Required Date:** 22-Sep-1997

Products	Unit Price	Quantity	Discount	Extended Price
Spegesild	\$12.00	2	25%	\$18.00
Chartreuse verte	\$18.00	21	25%	\$283.50
Rossle Sauerkraut	\$45.60	15	25%	\$513.00
*			0%	

Subtotal: \$814.50
Freight: \$29.46
Total: \$843.96

[Print Invoice](#)

one-to-many relationship

- Consider this example: the Suppliers and Products tables in the product orders database.
- A supplier can supply any number of products.
- It follows that for any supplier represented in the Suppliers table, there can be many products represented in the Products table.
- The relationship between the Suppliers table and the Products table is, therefore, a one-to-many relationship.

one-to-many relationship

The image shows two database tables in Microsoft Access:

- Suppliers Table:** Contains 'Supplier ID' (primary key) and 'Company Name'. The first record is '1 Exotic Liquids'.
- Products Table:** Contains 'Product ID', 'Product Name', and 'Supplier' (foreign key). It lists three products: '1 Chai', '2 Chang', and '3 Aniseed Syrup', all supplied by '1'.

A red arrow indicates the relationship from the 'Supplier ID' field in the Suppliers table to the 'Supplier' field in the Products table.

Supplier ID	Company Name
1	Exotic Liquids

Product ID	Product Name	Supplier
1	Chai	1
2	Chang	1
3	Aniseed Syrup	1

one-to-many relationship

- To represent a one-to-many relationship in your database design, take the primary key on the "one" side of the relationship and add it as an additional column or columns to the table on the "many" side of the relationship.
- In this case, for example, you add the Supplier ID column from the Suppliers table to the Products table.
- Access can then use the supplier ID number in the Products table to locate the correct supplier for each product.

one-to-many relationship

The Supplier ID column in the Products table is called a foreign key. A foreign key is another table's primary key. The Supplier ID column in the Products table is a foreign key because it is also the primary key in the Suppliers table.

You provide the basis for joining related tables by establishing pairings of primary keys and foreign keys. If you are not sure which tables should share a common column, identifying a one-to-many relationship ensures that the two tables involved will, indeed, require a shared column.



A notepad with a light pink background and a dark orange border, featuring two punch holes on the left. It contains a list of database tables and their columns, organized into four sections: Customers, Suppliers, Products, and Orders.

Customers	Suppliers	Products	Orders
CustomerID	SupplierID	ProductID	OrderID
Name	Company Name	Product Name	Salesperson
Address	Contact Name	Unit Price	Order Date
City	Address	Units in Stock	Product
Region	City	Units on Order	Quantity
Postal Code	Region	Quantity per Unit	Price
Country	Postal Code	SupplierID	
Send E-mail	Country		
Salutation	Phone		
E-mail address			

one-to-one relationship

- Another type of relationship is the one-to-one relationship.
- For instance, suppose you need to record some special supplementary product information that you will need rarely or that only applies to a few products.
- Because you don't need the information often, and because storing the information in the Products table would result in empty space for every product to which it doesn't apply, you place it in a separate table.
- Like the Products table, you use the ProductID as the primary key. The relationship between this supplemental table and the Product table is a one-to-one relationship.
- For each record in the Product table, there exists a single matching record in the supplemental table. When you do identify such a relationship, both tables must share a common field.

one-to-one relationship

- Determining the relationships between tables helps you ensure that you have the right tables and columns.
- When a one-to-one or one-to-many relationship exists, the tables involved need to share a common column or columns.

Refining the Design

- Once you have the tables, fields, and relationships you need, you should create and populate your tables with sample data and try working with the information: creating queries, adding new records, and so on.
- Doing this helps highlight potential problems — for example, you might need to add a column that you forgot to insert during your design phase, or you may have a table that you should split into two tables to remove duplication.

Refining the Design

- See if you can use the database to get the answers you want.
- Create rough drafts of your forms and reports and see if they show the data you expect.
- Look for unnecessary duplication of data and, when you find any, alter your design to eliminate it.

Refining the Design

- Did you forget any columns? If so, does the information belong in the existing tables?
- Are any columns unnecessary because they can be calculated from existing fields?
- Are you repeatedly entering duplicate information in one of your tables? If so, you probably need to divide the table into two tables that have a one-to-many relationship.
- Do you have tables with many fields, a limited number of records, and many empty fields in individual records? If so, think about redesigning the table so it has fewer fields and more records.
- Has each information item been broken into its smallest useful parts? If you need to report, sort, search, or calculate on an item of information, put that item in its own column.
- Does each column contain a fact about the table's subject? If a column does not contain information about the table's subject, it belongs in a different table.
- Are all relationships between tables represented, either by common fields or by a third table? One-to-one and one-to-many relationships require common columns. Many-to-many relationships require a third table.

GCFLearnFree

- Information from <http://www.gcflearnfree.org/access2016> and other web resources