Chapter 16

How to design a database

Objectives

Applied

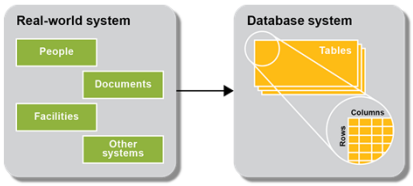
1. Given the specifications for a database modeled on a real-world system, design the database. That is, identify the tables, columns, keys, relationships, and indexes for the new database.
2. Given a diagram for an unnormalized database, normalize the structure to the third normal form.
3. Use MySQL Workbench to create database diagrams.

Objectives (continued)

Knowledge

1. Describe the process of designing a database in terms of tables, columns, keys, and relationships.
2. Describe referential integrity, and explain how enforcing it assures that the database isn't corrupted when rows are inserted, updated, or deleted.
3. Explain how normalizing a database to the third normal form improves database performance.
4. In general terms, describe the criteria for indexing a column.

A database system is modeled after a real-word system



The six basic steps for designing a data structure

Step 1: Identify the data elements
Step 2: Subdivide each element into its smallest useful components
Step 3: Identify the tables and assign columns
Step 4: Identify the primary and foreign keys
Step 5: Review whether the data structure is normalized
Step 6: Identify the indexes

An invoice that can be used to identify data elements

Acme Fabrication, Inc.

Custom Contraptions, Contrivances and Confabulations

1234 West Industrial Way East Los Angeles California 90022
800.555.1212 fax 562.555.1213 www.acmefabrication.com

Invoice Number:	I01-1088
Invoice Date:	10/05/17
Terms:	Net 30

Part No.	Qty.	Description	Unit Price	Extension
CUST345	12	Design service, hr	100.00	1200.00
457332	7	Baling wire, 25x3ft roll	79.90	559.30
50173	4375	Duct tape, black, yd	1.09	4768.75
328771	2	Rubber tubing, 100ft roll	4.79	9.58
CUST281	7	Assembly, hr	75.00	525.00
CUST917	2	Testing, hr	125.00	250.00
		Sales Tax		245.20

Your salesperson:	Ruben Goldberg, ext 4512
Accounts receivable:	Inigo Jones, ext 4901

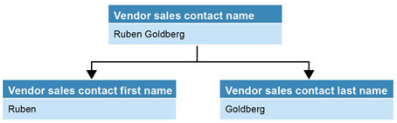
$7,557.83
PLEASE PAY THIS AMOUNT

Thanks for your business!

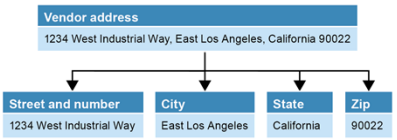
The data elements identified on the invoice

Vendor name	Invoice date	Item extension
Vendor address	Invoice terms	Vendor contact name
Vendor phone number	Item part number	Vendor contact ext.
Vendor fax number	Item quantity	Vendor AR contact name
Vendor web address	Item description	Vendor AR contact ext.
Invoice number	Item unit price	Invoice total

A name that's divided into first and last names



An address that's divided into street address, city, state, and zip code



Possible tables and columns for an A/P system

Vendors table

Vendor name	Vendor contact first name
Vendor address	Vendor contact last name
Vendor city	~~Vendor contact phone~~
Vendor state	~~Vendor AR first name~~
Vendor zip code	~~Vendor AR last name~~
Vendor phone number	~~Vendor AR phone~~
~~Vendor fax number~~	*Terms**
~~Vendor web address~~	*Account number**

- *Data elements that were added*
- *Data element related to two or more entities

Possible tables and columns for an A/P system (continued)

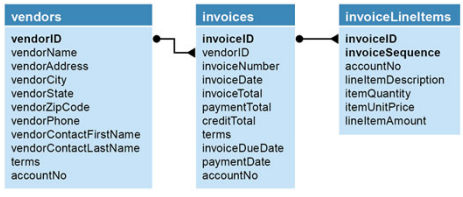
Invoices table

Invoice number*
Invoice date
Terms*
Invoice total
Payment date
Payment total
Invoice due date
Credit total
*Account number**

Invoice line items table

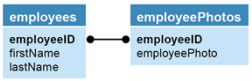
Invoice number*
~~Item part number~~
Item quantity
Item description
Item unit price
Item extension
*Account number**
Sequence number

The relationships between the tables in the accounts payable system

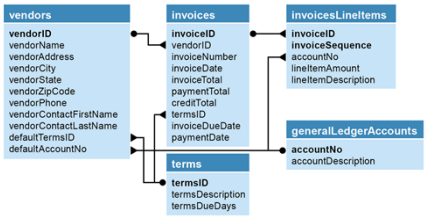


Two tables with a many-to-many relationship

Two tables with a one-to-one relationship



The accounts payable system in third normal form



Operations that can violate referential integrity

This operation...	Violates referential integrity if...
Delete a row from the primary key table	The foreign key table contains one or more rows related to the deleted row
Insert a row in the foreign key table	The foreign key value doesn't have a matching primary key value in the related table
Update the value of a foreign key	The new foreign key value doesn't have a matching primary key value in the related table
Update the value of a primary key	The foreign key table contains one or more rows related to the row that's changed

About indexes

- An *index* provides a way for a database management system to locate information more quickly.
- MySQL automatically creates an index for a primary key.
- You can create *composite indexes* of two or more columns.
- Because indexes must be updated each time you add, update, or delete a row, don't create more indexes than you need.

When to create an index

- When the column is a foreign key
- When the column is used frequently in search conditions or joins
- When the column contains a large number of distinct values
- When the column is updated infrequently

A table that contains repeating columns

vendorName	invoiceNumber	itemDescription_1	itemDescription_2	itemDescription_3
Cahners Publishing	112897	VB ad	SQL ad	Library directory
Zylka design	97/552	Catalogs	SQL flyer	*NULL*
Zylka design	97/553B	Card revision	*NULL*	*NULL*

A table that contains redundant data

vendorName	invoiceNumber	itemDescription
Cahners Publishing	112897	VB ad
Cahners Publishing	112897	SQL ad
Cahners Publishing	112897	Library directory
Zylka design	97/522	Catalogs
Zylka design	97/522	SQL flyer
Zylka design	97/533B	Card revisions

The seven normal forms

- First (1NF)
- Second (2NF)
- Third (3NF)
- Boyce-Codd (BCNF)
- Fourth (4NF)
- Fifth (5NF)
- Domain-key (DKNF) or Sixth (6NF)

The benefits of normalization

- Since a normalized database has more tables than an unnormalized database, and since each table has an index on its primary key, the database has more indexes. That makes data retrieval more efficient.
- Since each table contains information about a single entity, each index has fewer columns (usually one) and fewer rows. That makes data retrieval and insert, update, and delete operations more efficient.
- Each table has fewer indexes, which makes insert, update, and delete operations more efficient.
- Data redundancy is minimized, which simplifies maintenance and reduces storage.

The invoice data with a column that contains repeating values

vendorName	invoiceNumber	itemDescription
Cahners Publishing	112897	VB ad, SQL ad, Library directory
Zylka design	97/522	Catalogs, SQL Flyer
Zylka design	97/533B	Card revision

The invoice data with repeating columns

vendorName	invoiceNumber	itemDescription_1	itemDescription_2	itemDescription_3
Cahners Publishing	112897	VB ad	SQL ad	Library directory
Zylka design	97/552	Catalogs	SQL flyer	*NULL*
Zylka design	97/553B	Card revision	*NULL*	*NULL*

The invoice data in first normal form

vendorName	invoiceNumber	itemDescription
Cahners Publishing	112897	VB ad
Cahners Publishing	112897	SQL ad
Cahners Publishing	112897	Library directory
Zylka design	97/522	Catalogs
Zylka design	97/522	SQL flyer
Zylka design	97/533B	Card revisions

The invoice data in first normal form with keys added

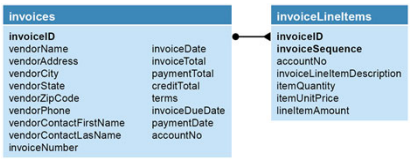
invoiceID	vendorName	invoiceNumber	invoiceSequence	itemDescription
1	Cahners Publishing	112897	1	VB ad
1	Cahners Publishing	112897	2	SQL ad
1	Cahners Publishing	112897	3	Library directory
2	Zylka design	97/522	1	Catalogs
2	Zylka design	97/522	2	SQL flyer
3	Zylka design	97/533B	1	Card revision

The invoice data in second normal form

invoiceNumber	vendorName	invoiceID
11287	Cahners Publishing	1
97/522	Zylka design	2
97/533B	Zylka design	3

invoiceID	invoiceSequence	itemDescription
1	1	VB ad
1	2	SQL ad
1	3	Library directory
2	1	Catalogs
2	2	SQL flyer
3	1	Card revision

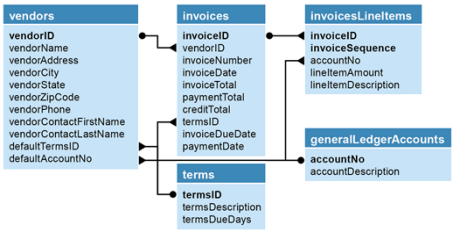
The A/P system in second normal form



Questions about the structure

1. Does the vendor information depend only on the invoiceID column?
2. Does the terms column depend only on the invoiceID column?
3. Does the accountNo column depend only on the invoiceID column?
4. Can the invoiceDueDate and lineItemAmount columns be derived from other data?

The A/P system in third normal form



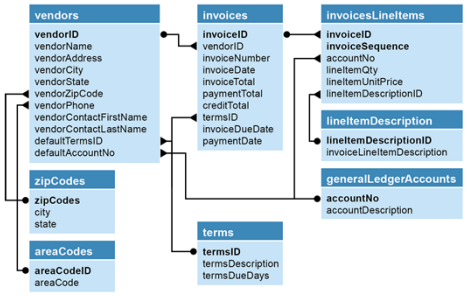
MySQL Workbench…

- Lets you create and edit diagrams.
- Lets you define the tables, columns, and indexes for a database.
- Lets you define the relationships between the tables in a database.
- Lets you generate a diagram from a SQL creation script.
- Lets you generate a SQL creation script from a diagram.

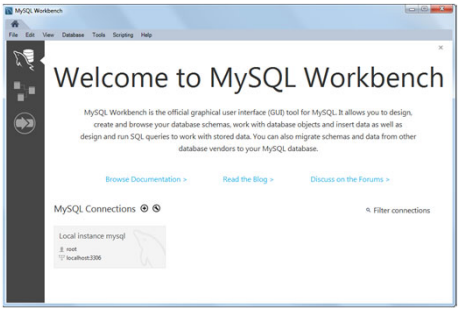
How to install MySQL Workbench

1. Go to the MySQL Workbench web site at:
   `http://www.mysql.com/products/workbench`
2. Download the version for your system.
3. Run the installer or setup file and respond to the prompts.

The accounts payable system in fifth normal form



The Welcome tab of the Home page



When to denormalize

- When a column from a joined table is used repeatedly in search criteria, you should consider moving that column to the primary key table if it will eliminate the need for a join.
- If a table is updated infrequently, you should consider denormalizing it to improve efficiency. Because the data remains relatively constant, you don't have to worry about data redundancy errors once the initial data is entered and verified.
- Include columns with derived values when those values are used frequently in search conditions. If you do that, you need to be sure that the column value is always synchronized with the value of the columns it's derived from.

MySQL Workbench