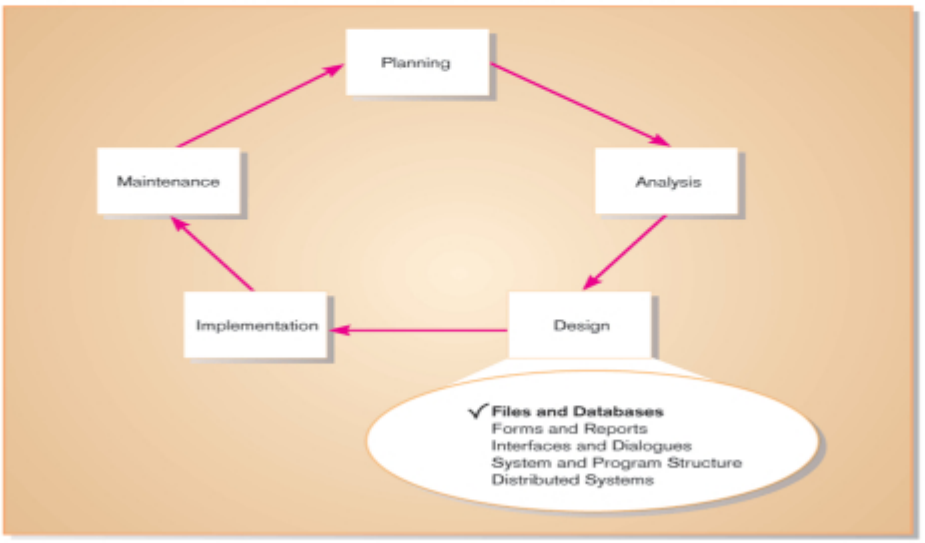
4. Design

4.1. Designing Databases

Introduction

Figure 10-1 Systems development life cycle with design phase highlighted



Database Design

- Database design occurs in Three stages:
  - Conceptual Design
    - Entities and their relationship are identified by analyzing input and output of the system
  - Logical Design
    - Attributes are identified
    - Primary keys are defines
    - ER diagram is used for logical representation
  - Physical Design
    - Convert ER diagram to database tables
    - Foreign keys are defined
    - Normalization

Database Design

Features	Conceptual Design	Logical Design	Physical Design
Entity Names	√	√	
Entity Relationships	√	√	
Attributes		√	√
Primary Keys		√	√
Foreign Keys			√
Table Names			√
Normalization			√

Database Design

- Generic steps of database design are:
  - Discover and organize entities and their attributes
  - Define primary keys
  - Define the relationships between different entities
  - Create ER diagram
  - Convert ER diagram to tables using sample data
  - Normalize the tables

Deliverables and Outcomes

- **Logical database design**
  - Relations are the primary deliverable.
  - Must account for every data element on a system input or output.
- Physical database design
  - Convert relations into database tables.
  - Programmers and database analysts code the definitions of the database.
  - Written in Structured Query Language (SQL).

Transforming E-R Diagrams into Relations

- It is useful to transform the conceptual data model (E-R) into normalized relations, then merge all in one final, consolidated set of relation which can be accomplished by the following steps:
  - Represent entities
  - Represent relationships
  - Normalize the relations
  - Merge the relations

Transforming E-R Diagrams into Relations

- Represent Entities
  - Each regular entity type in an E-R is transformed into a relation
  - The identifier of the entity type becomes the primary key of the corresponding relation
  - The primary key must satisfy the following two conditions
    - The key should be none-redundant, which mean no attribute in the key can be deleted without destroying its unique identification
    - The value of the key must uniquely identify every row in the relation
  - The entity type label is translates into a relation name.

Figure 9.10a Transforming an Entity Type to a Relation — E-R Diagram

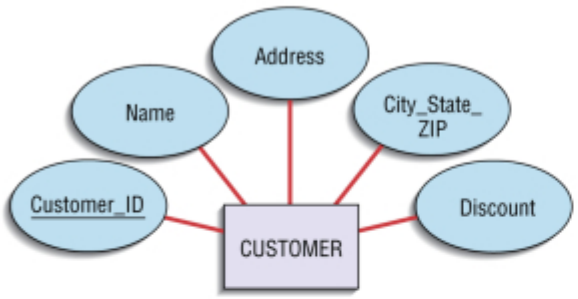


Figure 9.10b Transforming an Entity Type to a Relation — Relation

CUSTOMER

Customer_ID	Name	Address	City_State_ZIP	Discount
1273	Contemporary Designs	123 Oak St.	Austin, TX 28384	5%
6390	Casual Corner	18 Hoosier Dr.	Bloomington, IN 45821	3%

Transforming E-R Diagrams into Relations

- Represent Relationships
- Binary 1:N Relationships
  - Add the primary key attribute (or attributes) of the entity on the one side of the relationship as a foreign key in the relation on the many side
  - The one side *migrates* to the many side, see figure 10-11 where CUSTOMER ID which it is the primary key in CUSTOMER becomes a FK to the relation ORDER
- Binary or Unary 1:1
  - Three possible options
    - Add the primary key of A as a foreign key of B
    - Add the primary key of B as a foreign key of A
    - Both of the above

Figure 9.11a Representing a 1:N Relationship — E-R Diagram

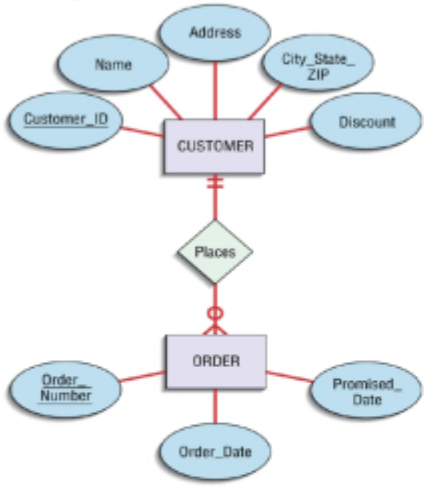


Figure 9.11b Representing a 1:N Relationship — Relations

CUSTOMER

Customer_ID	Name	Address	City_State_ZIP	Discount
1273	Contemporary Designs	123 Oak St.	Austin, TX 28384	5%
6390	Casual Corner	18 Hoosier Dr.	Bloomington, IN 45821	3%

ORDER

Order_Number	Order_Date	Promised_Date	Customer_ID
57194	3/15/0X	3/28/0X	6390
63725	3/17/0X	4/01/0X	1273
80149	3/14/0X	3/24/0X	6390

Transforming E-R Diagrams into Relations

- Represent Relationships (continued)
- Binary and Higher M:N relationships
  - Create another relation and include primary keys of all relations as primary key (composite) of new relation
- Unary 1:N Relationships
  - Relationship between instances of a single entity type
  - Utilize a recursive foreign key
    - A foreign key in a relation that references the primary key values of that same relation ex. EMPLOYEE(emp_id, Name, Birthdate, Manager_id)
- Unary M:N Relationships
  - Create a separate relation
  - Primary key of new relation is a composite of two attributes that both take their values from the same primary key ex. ITEM(item_no, name, Cost) and ITEM_BILL(item_no, component_id, Quantity)

Figure 9.12a Representing an M:N Relationship — E-R Diagram

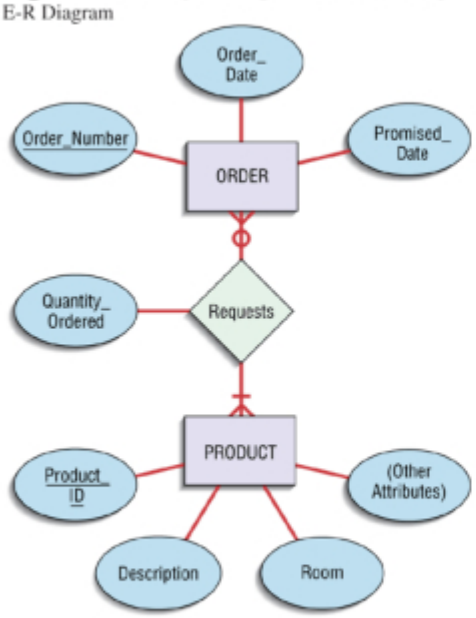


Figure 9.12b Representing an M:N Relationship — Relations

ORDER

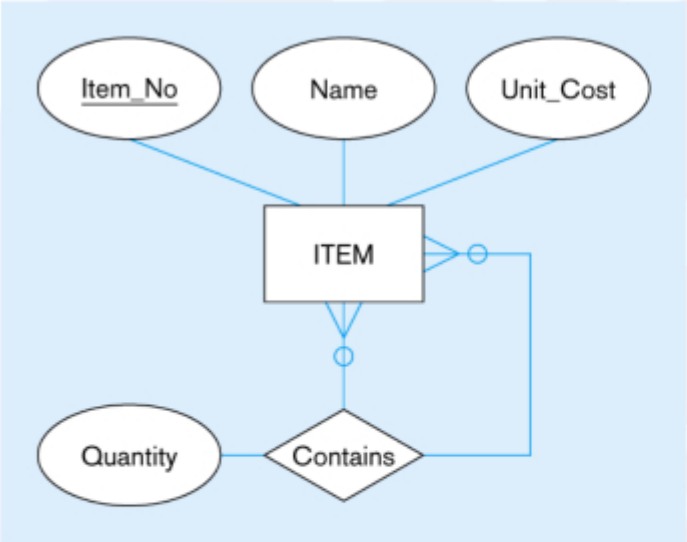
Order_Number	Order_Date	Promised_Date
61384	2/17/2002	3/01/2002
62009	2/13/2002	2/27/2002
62807	2/15/2002	3/01/2002

ORDER LINE

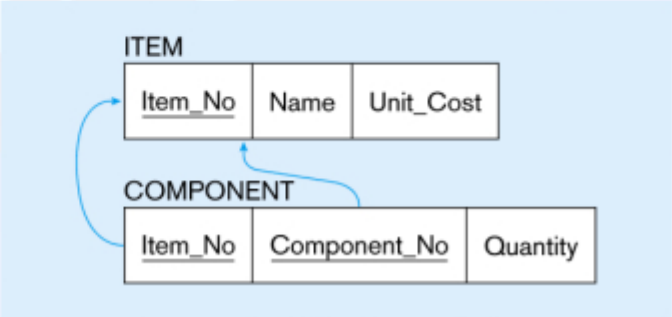
Order_Number	Product_ID	Quantity_ Ordered
61384	M128	2
61384	A261	1

PRODUCT

Product_ID	Description	(Other Attributes)
M128	Bookcase	—
A261	Wall unit	—
R149	Cabinet	—



Mapping a unary M:N relationship



Designing Fields

- Field
  - The smallest unit of named application data recognized by system software
  - Each attribute from each relation will be represented as one or more fields
- Choosing data types
  - Data Type
    - A coding scheme recognized by system software for representing organizational data
  - Four objectives
    - Minimize storage space
    - Represent all possible values of the field
    - Improve data integrity of the field
    - Support all data manipulations desired on the field
- Calculated field:
  - a field that can be derived from other database fields. It is common for an attribute to be mathematically related to other data
  - The calculate value is either stored or computed when it is requested.

Methods of Controlling Data Integrity

- Default Value
  - A value a field will assume unless an explicit value is entered for that field
- Range Control
  - Limits range of values which can be entered into field
- Referential Integrity
  - An integrity constraint specifying that the value of an attribute in one relation depends on the value of the same attribute in another relation
- Null Value
  - A special field value, distinct from 0, blank, or any other value, that indicates that the value for the field is missing or otherwise unknown

Database Design

- Normalization
- the process of converting complex data structures into simple, stable data structures.
  - 1 NF
    - Rule1: Each cell in a table should contain a single value
    - Rule 2: Each record must be unique
  - 2 NF
    - Rule 1: Be in 1 NF
    - Rule 2: Primary key must be single column
  - 3 NF
    - Rule 1: Be in 2 NF
    - Rule 2: There should be no transitive functional dependencies

Database Design

- Normalization
  - BCNF
    - Rule 1: It should be in the **Third Normal Form**.
    - Rule 2: And, for any dependency A → B, A should be a **super key**.
  - 4 NF
    - Rule 1: It should be in the **BCNF**.
    - Rule 2: the table should not have any **Multi-valued Dependency**.
  - 5 NF
    - Rule 1: Be in 2 NF
    - Rule 2: There should be no transitive functional dependencies
  - 6 NF
    - Not yet standardized

Database Design

- Normalization: Example

Full Names	Physical Address	Movies rented	Salutation	Category
Janet Jones	First Street Plot No 4	Pirates of the Caribbean, Clash of the Titans	Ms.	Action, Action
Robert Phil	3rd Street 34	Forgetting Sarah Marshal, Daddy's Little Girls	Mr.	Romance, Romance
Robert Phil	5th Avenue	Clash of the Titans	Mr.	Action

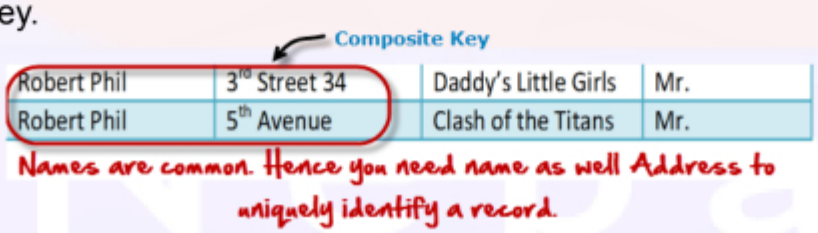
Here you see Movies Rented column has multiple values, so it can be split into multiple rows to convert it to 1 NF form

Database Design

- Normalization: Example

Full Names	Physical Address	Movies Rented	Salutation
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

Above table does not still have unique identification. We can combine name and address to represent primary key.



Database Design

- Normalization: Example

- Moving on to 2 NF, we must have single valued primary key
- It is clear that we can't move forward to make our simple database in 2nd Normalization form unless we partition the table above.

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3rd Street 34	Mr.
3	Robert Phil	5th Avenue	Mr.

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Database Design

- Normalization: Example
- To move our 2NF table into 3NF, we again need to again divide our table because we have transitive functional dependency
- Transitive Functional Dependency:
  - A transitive dependency is an indirect relationship between values in the same table that causes a functional dependency
  - Transitive dependency occurs when two or more non-key values are dependent
  - In above table, address and salutation are dependent on name

Database Design

- Normalization: Example
- To remove the transitive dependency between name an d salutation, the table can be split as below:

Membership ID	Full Names	Physical Address	Salutation ID
1	Janet Jones	First Street Plot No 4	2
2	Robert Phil	3rd Street 34	1
3	Robert Phil	5th Avenue	1

Membership ID	Movies Rented
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Salutation ID	Salutation
1	Mr.
2	Ms.
3	Mrs.
4	Dr.

4.2. Designing Forms and Reports

Designing Forms and Reports (Cont.)

- **Report**:
- A business document that contains only predefined data
- It is a passive document used solely for reading or viewing data.
- A report typically contains data from many unrelated records or transactions.

Designing Forms and Reports (Cont.)

- Common Types of Reports:
  - ***Scheduled****:* produced at predefined time intervals for routine information needs*Scheduled*
  - ***Key-indicator***: provides summary of critical information on regular basis
  - ***Exception***: highlights data outside of normal operating ranges
  - ***Drill-down***: provides details behind summary of key-indicator or exception reports
  - ***Ad-hoc***: responds to unplanned requests for non-routine information needs

Process of Designing Forms and Reports

- Is a user-focused activity.
- Follows a prototyping approach.
- First steps are to gain an understanding of the intended user and task objectives by collecting initial requirements during requirements determination.

Process of Designing Forms and Reports

- Requirements determination:
  - Who will use the form or report?
  - What is the purpose of the form or report?
  - When is the report needed or used?
  - Where does the form or report need to be delivered and used?
  - How many people need to use or view the form or report?

Process of Designing Forms and Reports

- Prototyping
  - Initial prototype is designed from requirements.
  - Users review prototype design and either accept the design or request changes.
  - If changes are requested, the construction-evaluation-refinement cycle is repeated until the design is accepted.

Process of Designing Forms and Reports

- A coding sheet is an “old” tool for designing forms and reports, usually associated with text-based forms and reports for mainframe applications.
- Visual Basic and other development tools provide computer-aided GUI form and report generation.

Process of Designing Forms and Reports

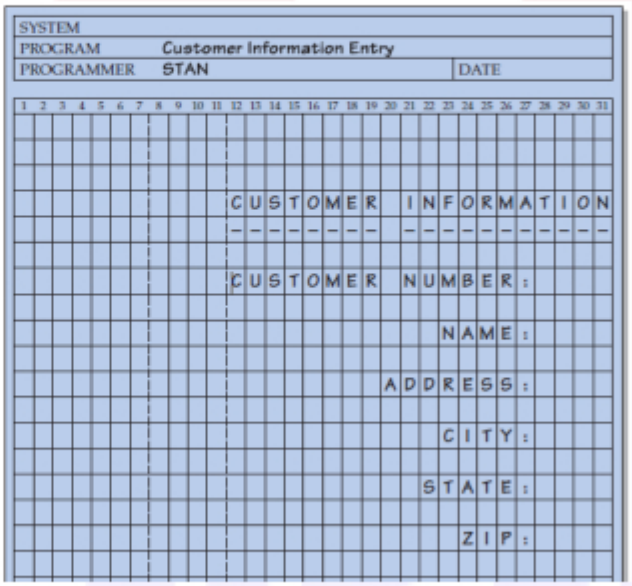


FIGURE 10-2
The layout of a data input form using a coding sheet

Process of Designing Forms and Reports

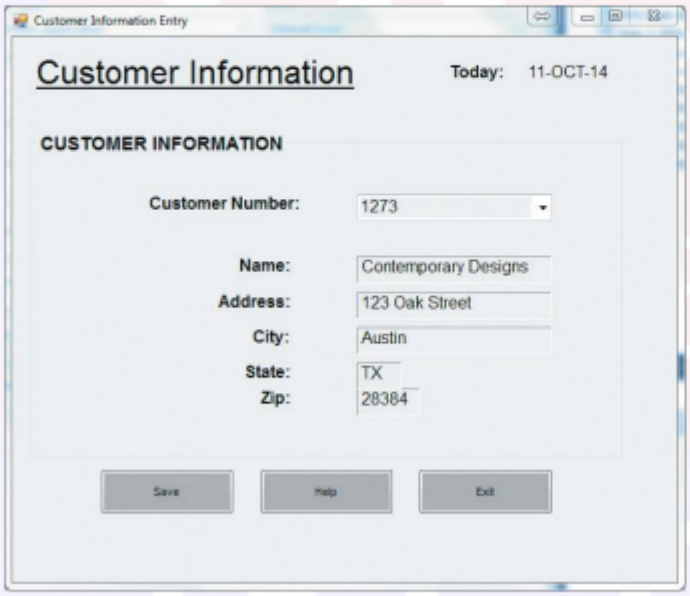


FIGURE 10-3
A data input screen designed in Microsoft's Visual Basic .NET (*Source:* Microsoft Corporation.)

Deliverables and Outcomes

- Design specifications are the major deliverables and inputs to the system implementation phase.

Deliverables and Outcomes

- Design specifications have three sections:
  - ***Narrative overview***: characterizes users, tasks, system, and environmental factors
  - ***Sample design***: image of the form (from coding sheet or form building development tool)
  - ***Testing and usability assessment***: measuring test/usability results (consistency, sufficiency, accuracy, etc.)

Formatting Forms and Reports

- *Meaningful titles* — use clear, specific, version information, and current date
- *Meaningful information* — include only necessary information, with no need to modify
- *Balanced layout* — use adequate spacing, margins, and clear labels
- *Easy navigation system* — show how to move forward and backward, and where you are currently

Formatting Forms and Reports

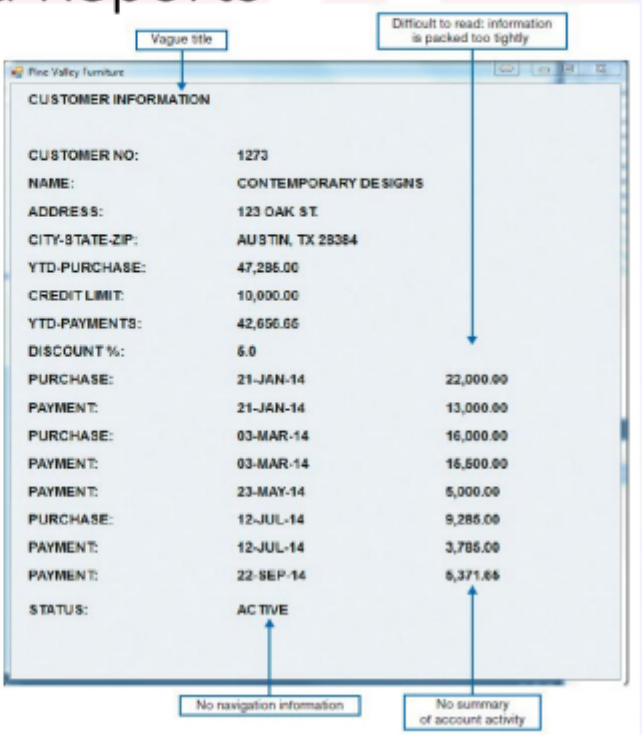


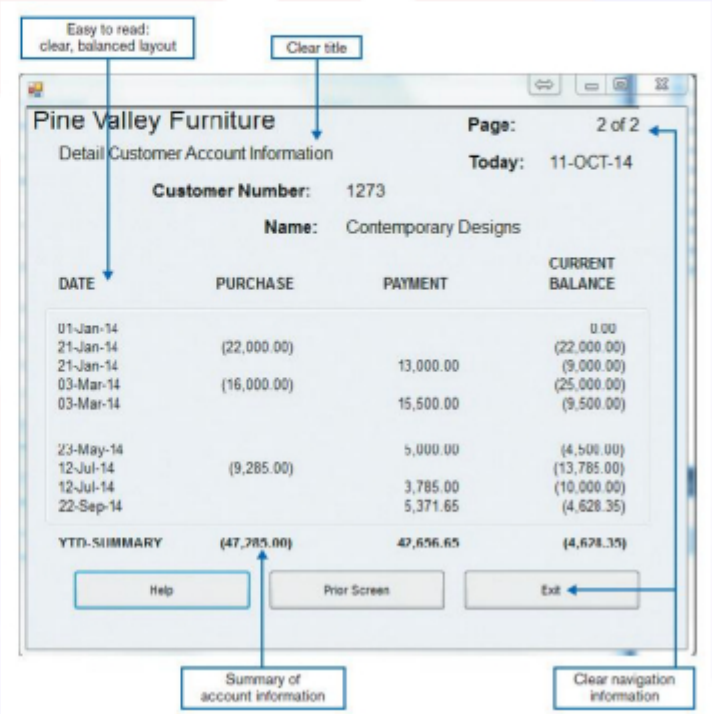
FIGURE 10-5
Contrasting customer information forms (Pine Valley Furniture) (*Source:* Microsoft Corporation.)

(a) Poorly designed form

Formatting Forms and Reports

FIGURE 10-5 (continued)

(b) Improved design for form



Highlighting Information

- Notify users of errors in data entry or processing.
- Provide warnings regarding possible problems.
- Draw attention to keywords, commands, high-priority messages, unusual data values.

Highlighting Information

Highlighting can include use of

- upper case
- bold
- italics
- underlining
- boxing
- size and color differences
- all capital letters
- blinking
- reverse video
- audible tones
- intensity differences
- offsetting nonstandard information

Highlighting Information

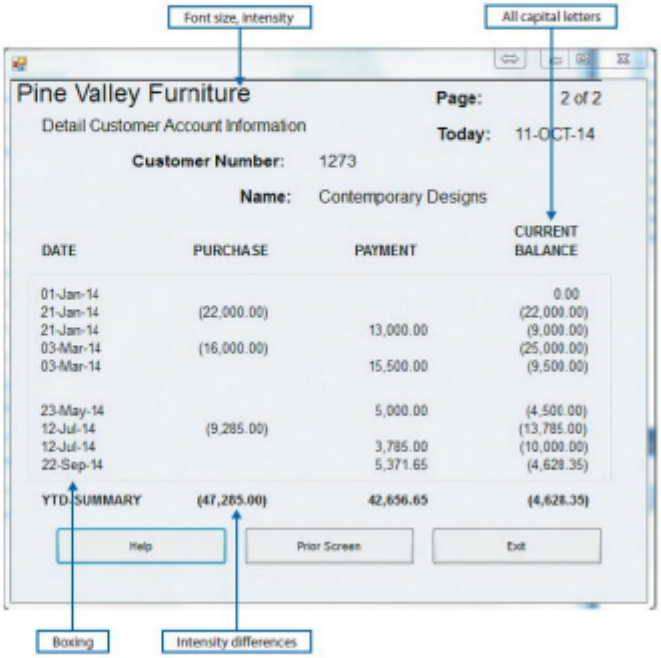


FIGURE 10-6
Customer account status display using various highlighting techniques
(Pine Valley Furniture)

(*Source:* Microsoft Corporation.)

Color vs. No Color

- Benefits — Color:
  - Soothes or strikes the eye.
  - Accents an uninteresting display.
  - Facilitates subtle discriminations in complex displays.
  - Emphasizes the logical organization of information.
  - Draws attention to warnings.
  - Evokes more emotional reactions.

Color vs. No Color

- Problems from Using Color
  - Color pairings may wash out or cause problems for some users.
  - Resolution may degrade with different displays.
  - Color fidelity may degrade on different displays.
  - Printing or conversion to other media may not easily translate.

Displaying Text

- **Case**: display in mixed upper and lower case, use conventional punctuation
- **Spacing**: use double spacing if possible, otherwise blank lines between paragraphs
- **Justification**: left justify text, ragged right margins
- **Hyphenation:** don't hyphenate words between lines
- **Abbreviations**: use only when widely understood and significantly shorter than full text

Displaying Text

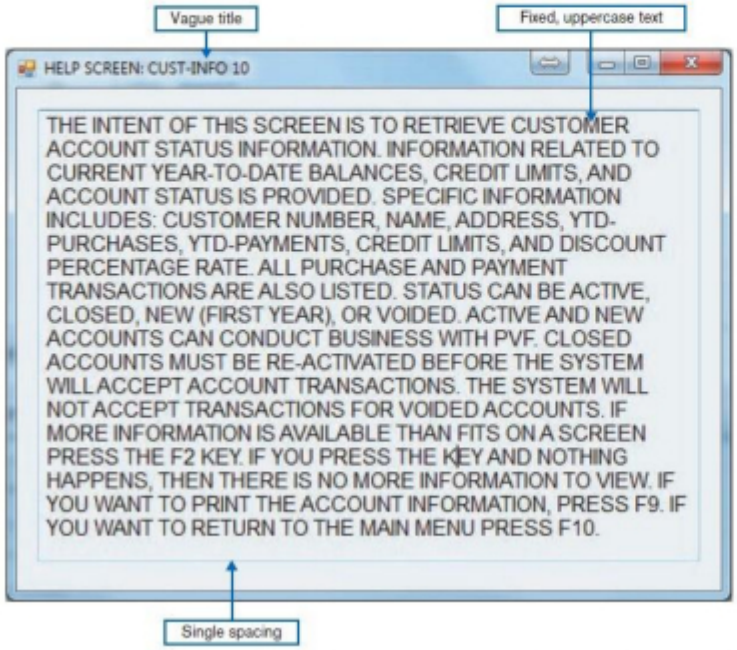


FIGURE 10-7
Contrasting the display of textual help information (*Source:* Microsoft Corporation.)

(a) Poorly designed help screen with many violations of the general guidelines for displaying text

Displaying Text

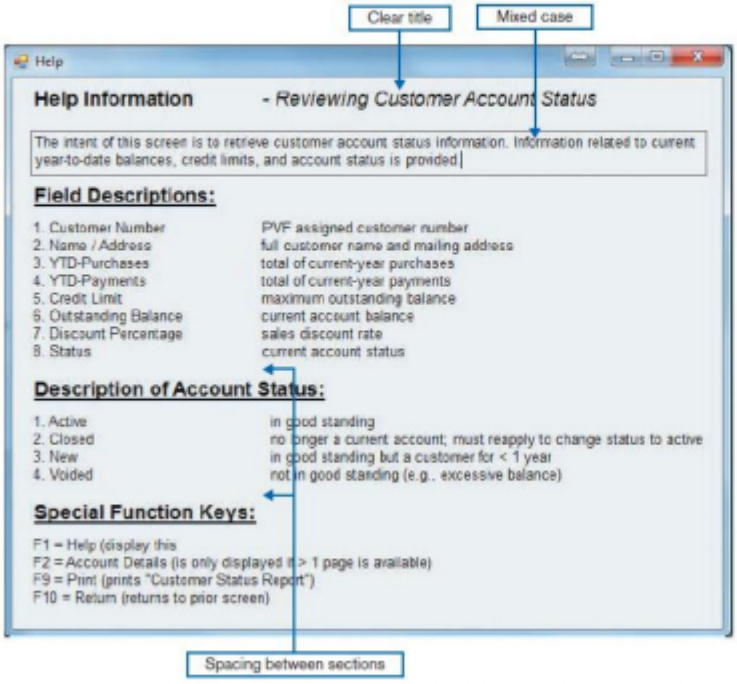


FIGURE 10-7 (continued)

(b) An improved design for a help screen

Designing Tables and Lists

- Labels
  - All columns and rows should have meaningful labels.
  - Labels should be separated from other information by using highlighting.
  - Redisplay labels when the data extend beyond a single screen or page.

Designing Tables and Lists

- Formatting columns, rows and text:
  - Sort in a meaningful order.
  - Place a blank line between every five rows in long columns.
  - Similar information displayed in multiple columns should be sorted vertically.
  - Columns should have at least two spaces between them.
  - Allow white space on printed reports for user to write notes.
  - Use a single typeface, except for emphasis.
  - Use same family of typefaces within and across displays and reports.
  - Avoid overly fancy fonts.

Designing Tables and Lists

- Formatting numeric, textual and alphanumeric data:
  - Right justify numeric data and align columns by decimal points or other delimiter.
  - Left justify textual data. Use short line length, usually 30 to 40 characters per line.
  - Break long sequences of alphanumeric data into small groups of three to four characters each.

Designing Tables and Lists

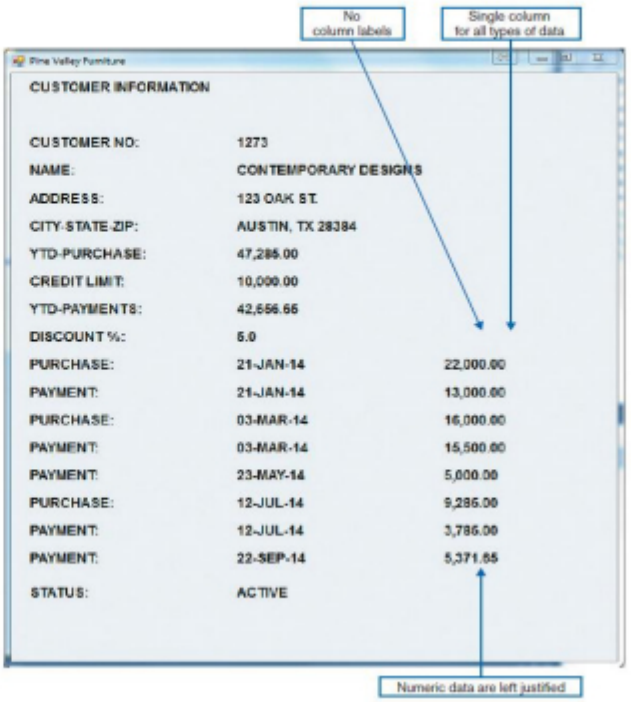


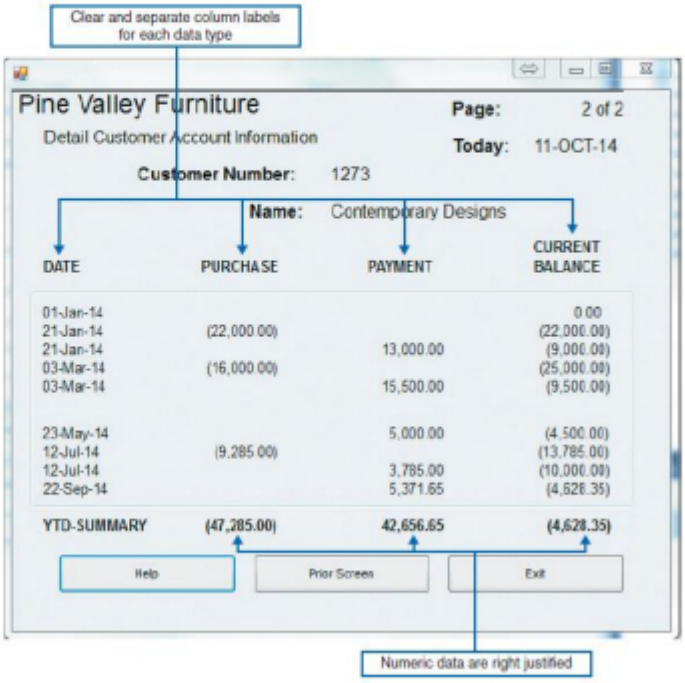
FIGURE 10-8
Contrasting the display of tables and lists (Pine Valley Furniture) (*Source:* Microsoft Corporation.)

(a) Poorly designed form

Designing Tables and Lists

FIGURE 10-8 (continued)

(b) Improved design for form



Designing Tables and Lists

- Use tables for reading individual data values.
- Use graphs for:
  - Providing quick summary.
  - Displaying trends over time.
  - Comparing points and patterns of variables.
  - Forecasting activity.
  - Simple reporting of vast quantities of information.

Designing Tables and Lists

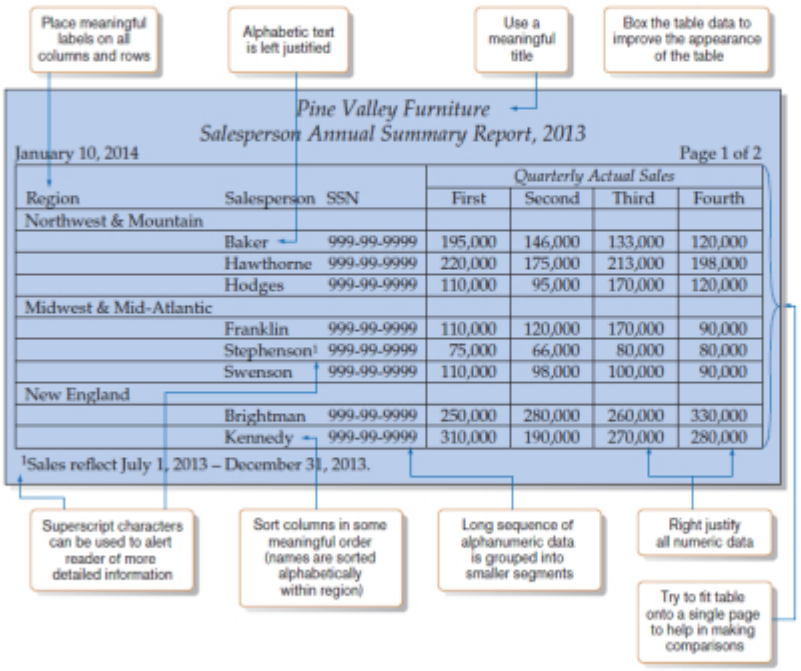


FIGURE 10-9
Tabular report illustrating numerous design guidelines

(Pine Valley Furniture)

Designing Tables and Lists

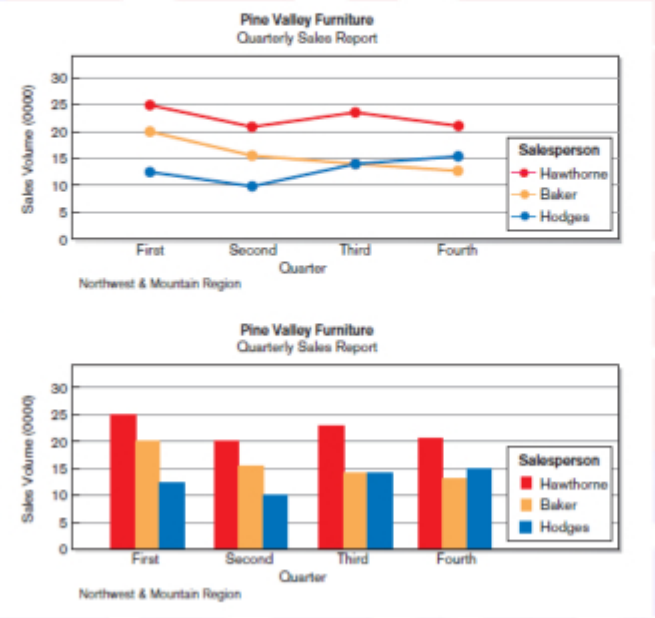


FIGURE 10-10
Graphs for comparison

(a) Line graph

(b) Bar graph

Assessing Usability

- Objective for designing forms, reports and all human-computer interactions is usability.
- There are three characteristics:
  - *Speed* — Can you complete a task efficiently?
  - *Accuracy* — Does the output provide what you expect?
  - *Satisfaction* — Do you like using the output?

Assessing Usability (Cont.)

- **Usability**: an overall evaluation of how a system performs in supporting a particular user for a particular task

Usability Success Factors

- **Consistency** — of terminology, formatting, titles, navigation, response time
- **Efficiency** — minimize required user actions
- **Ease** — self-explanatory outputs and labels
- **Format** — appropriate display of data and symbols
- **Flexibility** — maximize user options for data input according to preference

Usability Success Factors

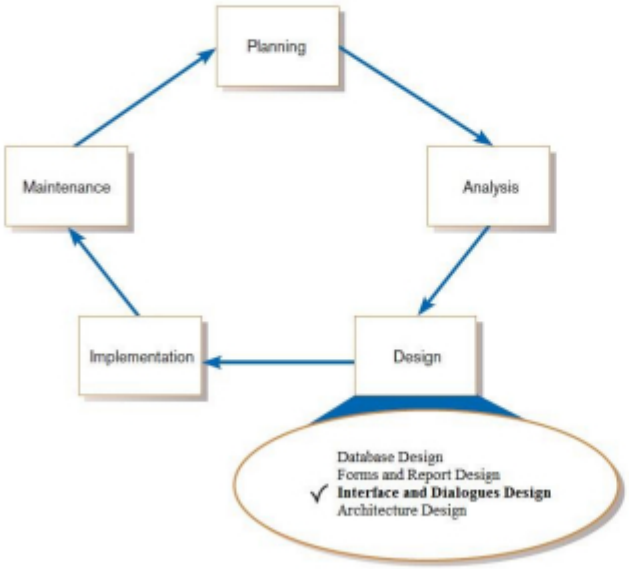
- Characteristics for consideration:
  - **User**: experience, skills, motivation, education, personality
  - **Task**: time pressure, cost of errors, work durations
  - **System**: platform
  - **Environment**: social and physical issues

Measures of Usability

- Time to learn
- Speed of performance
- Rate of errors
- Retention over time
- Subjective satisfaction
- Consistency of layout

4.3. Designing Interfaces and Dialogues

Designing Forms and Reports



Systems development life cycle with logical design phase highlighted

Interface/Dialogue Design

- Layout (of widgets, text, and table data)
- Structuring data entry (tab order)
- Controlling data input (validation and format controls)
- Feedback (prompting, status, warning, and error messages)
- Dialogue sequencing

Deliverables and Outcomes

Figure 12-2 Specification outline for the design of interfaces and dialogues

Design Specification

1. Narrative overview
   a. Interface/Dialogue Name
   b. User Characteristics
   c. Task Characteristics
   d. System Characteristics
   e. Environmental Characteristics

2. Interface/Dialogue Designs
   a. Form/Report Designs
   b. Dialogue Sequence Diagram(s) and Narrative Description

3. Testing and Usability Assessment
   a. Testing Objectives
   b. Testing Procedures
   c. Testing Results
      i) Time to Learn
      ii) Speed of Performance
      iii) Rate of Errors
      iv) Retention over Time
      v) User Satisfaction and Other Perceptions

Interface Methods

- Interface: the method by which a user interacts with the information system
- Common interaction methods
  - Command line
  - Menu
  - Form
  - Object-based
  - Natural language

Command Line Interaction

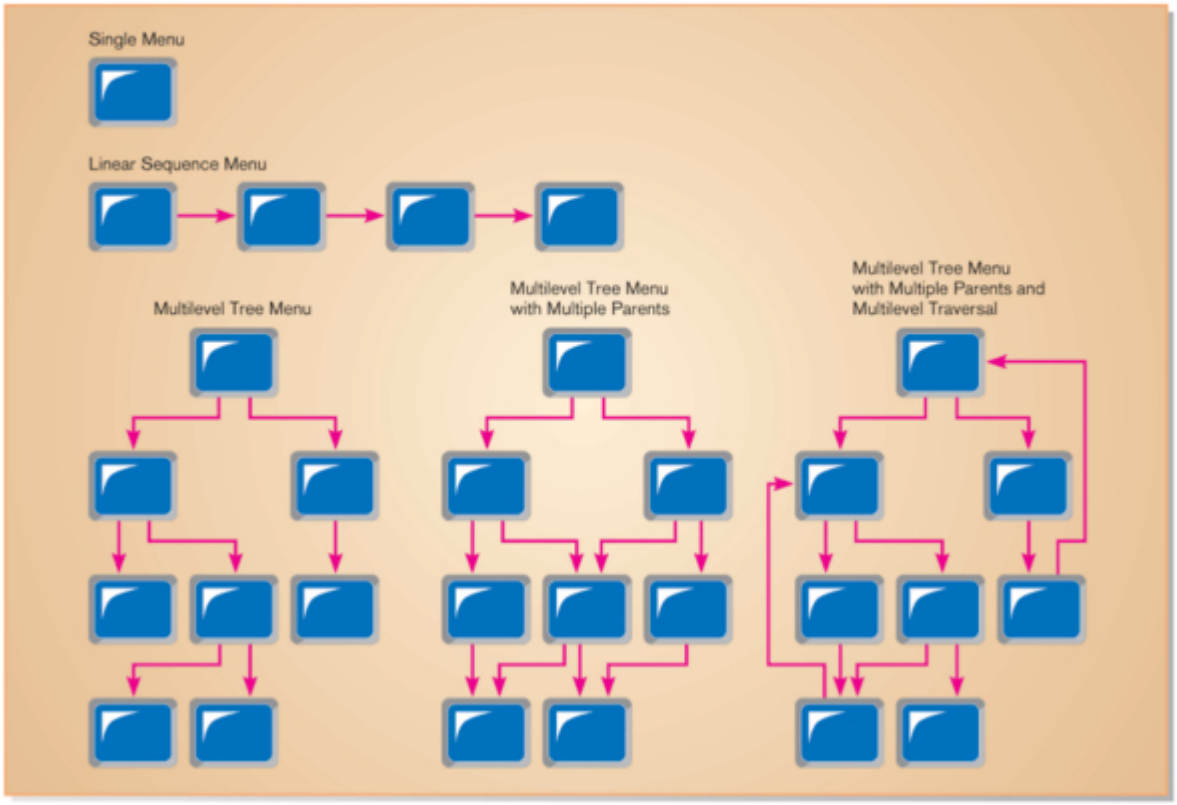
- Users enter explicit statements into a system to invoke operations
- Example from MS DOS:
  - COPY C:PAPER.DOC A:PAPER.DOC
  - This copies a file from the C: drive to the A: drive
- Includes keyboard shortcuts and function keys
- Experienced users and for rapid interaction with a system
- User interface standards

Menu Interaction

- A list of system options is provided and specific command is invoked by user selection of a menu option
- Two common menu types:
  - Pop-up: menu placed near current cursor position
    - List of commands or possible values
  - Drop-down: access point to menu placed at top line of display, menu drops down when access point clicked

Figure 12-4 Single-level menu (along left edge of screen) from SnagIt® by TechSmith.

Figure 12-5 Various types of menu configurations (Adapted from Schneiderman and Plaisant, 2004)



Guidelines for Menu Design

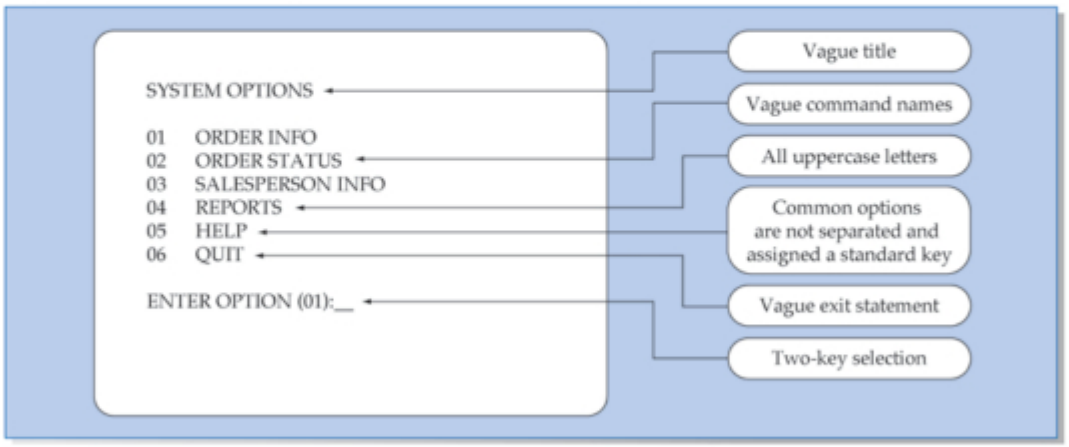
- Wording: meaningful titles, clear command verbs, mixed upper/lower case
  - Quit → prior menu or exit program?
- Organization: consistent organizing principle
  - Related options grouped together
  - Same option should have the same wording
- Length: all choices fit within screen length
  - Use submenus to break up exceedingly long menus

Guidelines for Menu Design

- Selection: consistent, clear and easy selection methods
  - How to select and the consequences of each option – will another menu appear?
- Highlighting: only for selected options (check mark) or unavailable options (dimmed text)
- Use menu building tools

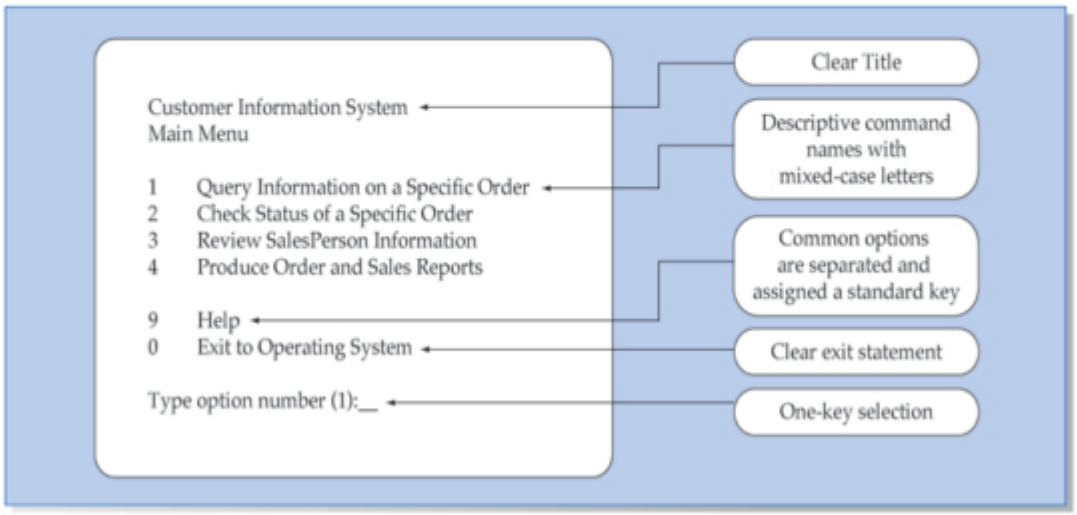
Poor Menu Design

Figure 12-7a Contrasting menu designs - Poor menu design



Good Menu Design

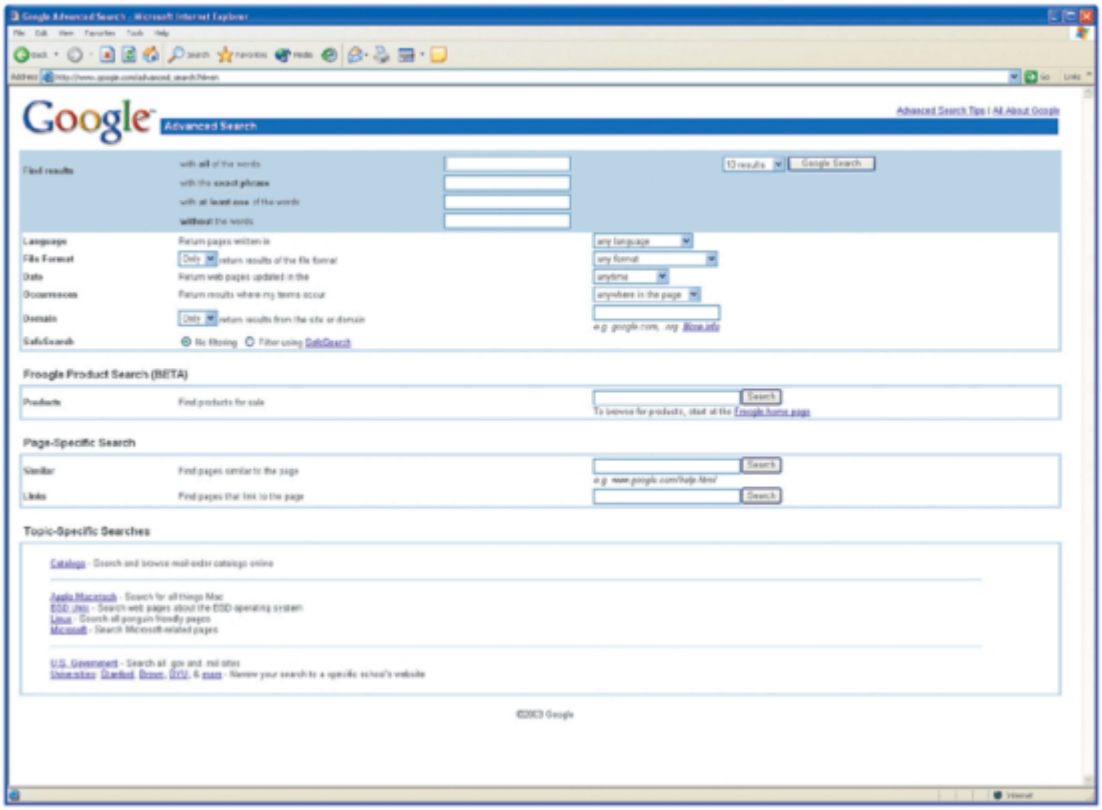
Figure 12-7b Contrasting menu designs - Improved menu design



Form Interaction

- Allows users to fill in the blanks when working with a system
- Measures of an effective design:
  - Self-explanatory title and field headings
  - Fields organized into logical groupings
  - Distinctive boundaries
  - Default values
  - Displays appropriate field lengths
  - Minimizes the need to scroll windows

Figure 12-9 Example of form interaction from the Google Advanced Search Engine



Object Interaction

- Symbols are used to represent commands or functions.
- Icons:
  - Graphic symbols that look like the processing option they are meant to represent
  - Use little screen space
  - Can be easily understood by users

Natural Language Interaction

- Inputs to and outputs from system are in a conventional speaking language like English
- Based on research in artificial intelligence
- Current implementations are tedious and difficult to work with, not as viable as other interaction methods

Hardware Options Interaction

- Keyboard
- Mouse
- Joystick
- Trackball

Make selection during logical design!!!

- Touch Screen
- Light Pen
- Voice

Different interfaces require different devices

Usability Problems

- Visual Blocking: extent to which device blocks display when using
- User Fatigue: potential for fatigue over long use
- Movement Scaling: extent to which device movement translates to equivalent screen movement
- Durability: lack of durability or need for maintenance (e.g., cleaning) over extended use

Usability Problems

- Adequate feedback: extent to which device provides adequate feedback for each operation
- Speed: cursor movement speed
- Pointing accuracy: ability to precisely direct cursor

Usability Problems with Hardware Devices

- Visual Blocking
  - touch screen, light pen
- User Fatigue
  - touch screen, light pen
- Movement Scaling
  - keyboard, mouse, joystick, trackball, voice
- Durability
  - trackball, touch screen

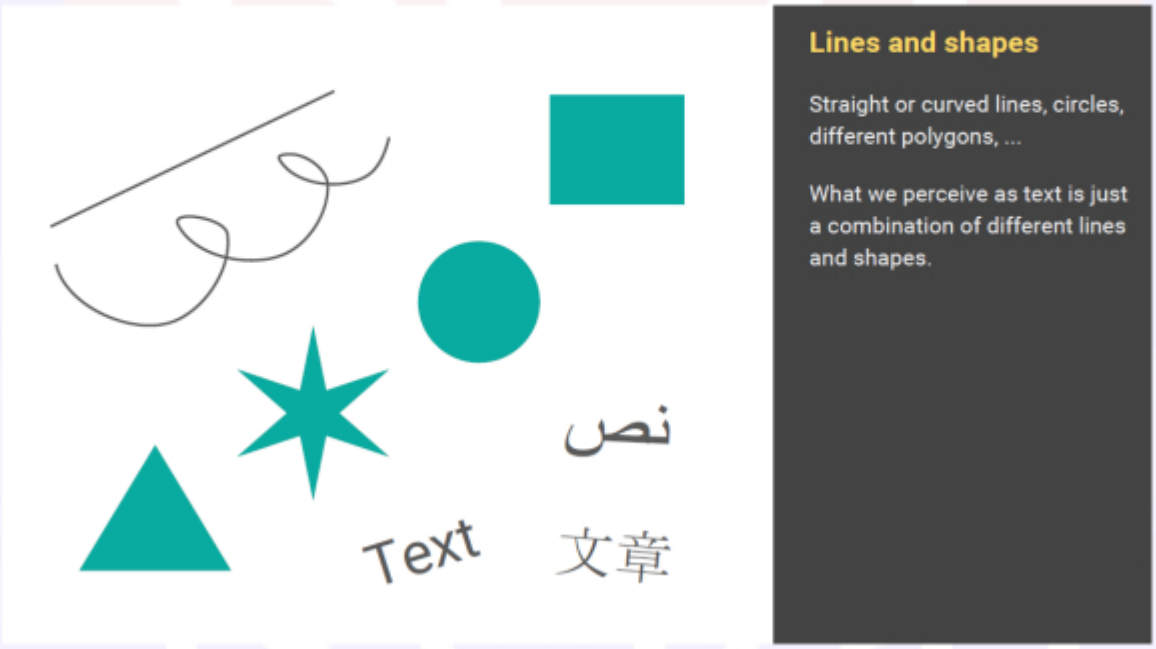
- Adequate Feedback
  - keyboard, mouse, joystick, trackball, voice
- Speed
  - keyboard
- Pointing Accuracy
  - joystick, touch screen, light pen, voice

Table 12-4 Summary of General Conclusions from Experimental Comparisons of Input Devices in Relation to Specific Task Activities

Task	*Most Accurate*	*Shortest Positioning*	*Most Preferred*
Target Selection	trackball, graphics tablet, mouse, joystick	touch screen, light pen, mouse, graphics tablet, trackball	touch screen, light pen
Text Selection	mouse	mouse	—
Data Entry	light pen	light pen	—
Cursor Positioning	—	light pen	—
Text Correction	light pen, cursor keys	light pen	light pen
Menu Selection	touch screen	—	keyboard, touch screen

(*Source:* Adapted from Blattner & Schultz, 1988.)
Key:
Target Selection = moving the cursor to select a figure or item
Text Selection = moving the cursor to select a block of text
Data Entry = entering information of any type into a system
Cursor Positioning = moving the cursor to a specific position
Text Correction = moving the cursor to a location to make a text correction
Menu Selection = activating a menu item
— = no clear conclusion from the research

Designing Interfaces: Building Blocks



Lines and shapes

Straight or curved lines, circles, different polygons, ...

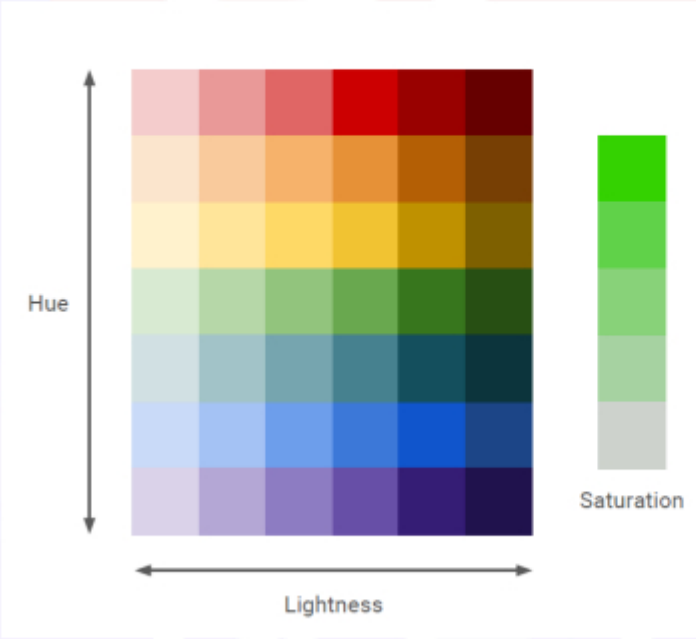
What we perceive as text is just a combination of different lines and shapes.

Designing Interfaces: Building Blocks

Size

All shapes can vary in size, width and height. Lines can be thicker or thinner. We usually assign more importance to bigger things.

Designing Interfaces: Building Blocks



Color

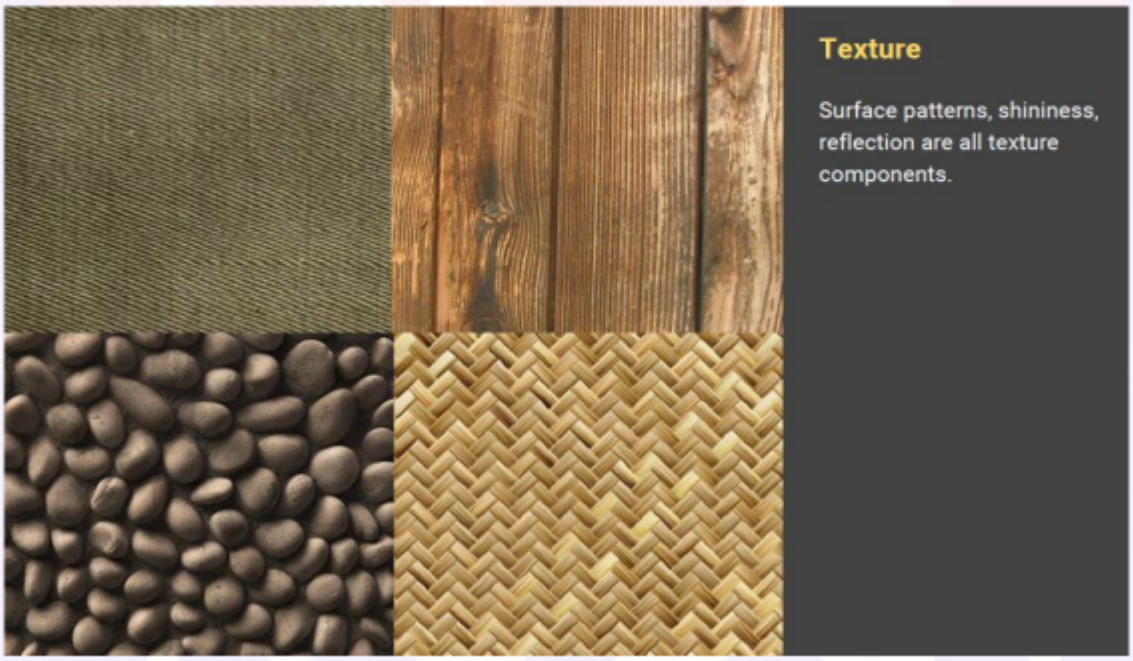
Color can be evaluated on three dimensions: hue, lightness, and saturation.

Hue is what most people mean by color and what we have a clear name for: red, green, yellow, etc.

Lightness, often called value, is how light or dark a color is.

Saturation is how vivid and intense a color is. Desaturating a color leads to gray.

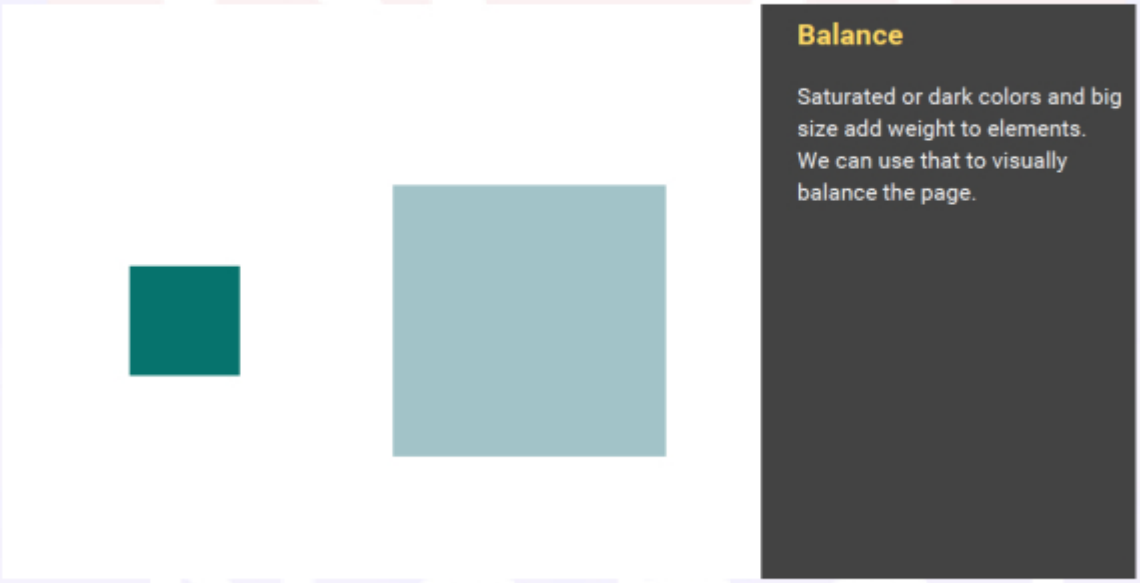
Designing Interfaces: Building Blocks



Texture

Surface patterns, shininess, reflection are all texture components.

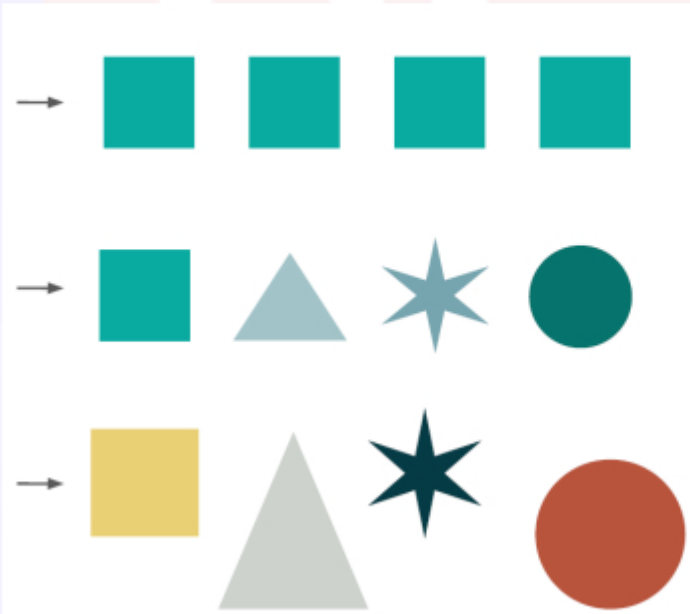
Designing Interfaces: Basic Principles



Balance

Saturated or dark colors and big size add weight to elements. We can use that to visually balance the page.

Designing Interfaces: Basic Principles

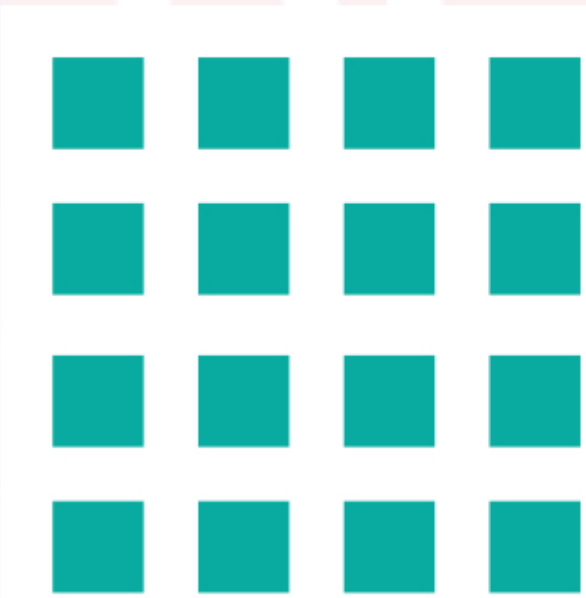


Repetition

If similar things are repeated, we see them as belonging together; they form a group.

Repeated element don't have to share all characteristics. But the more they do, the easier it is for us to group them.

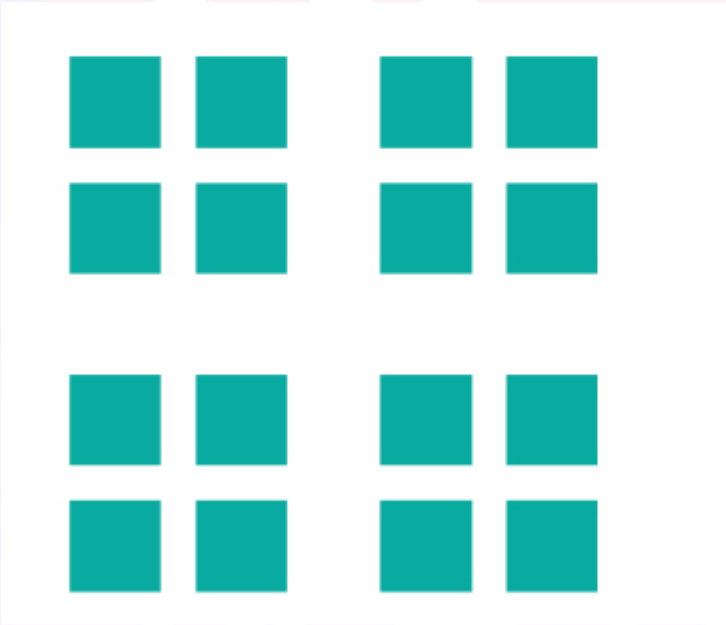
Designing Interfaces: Basic Principles



Repetition

Here we see one group of sixteen elements.

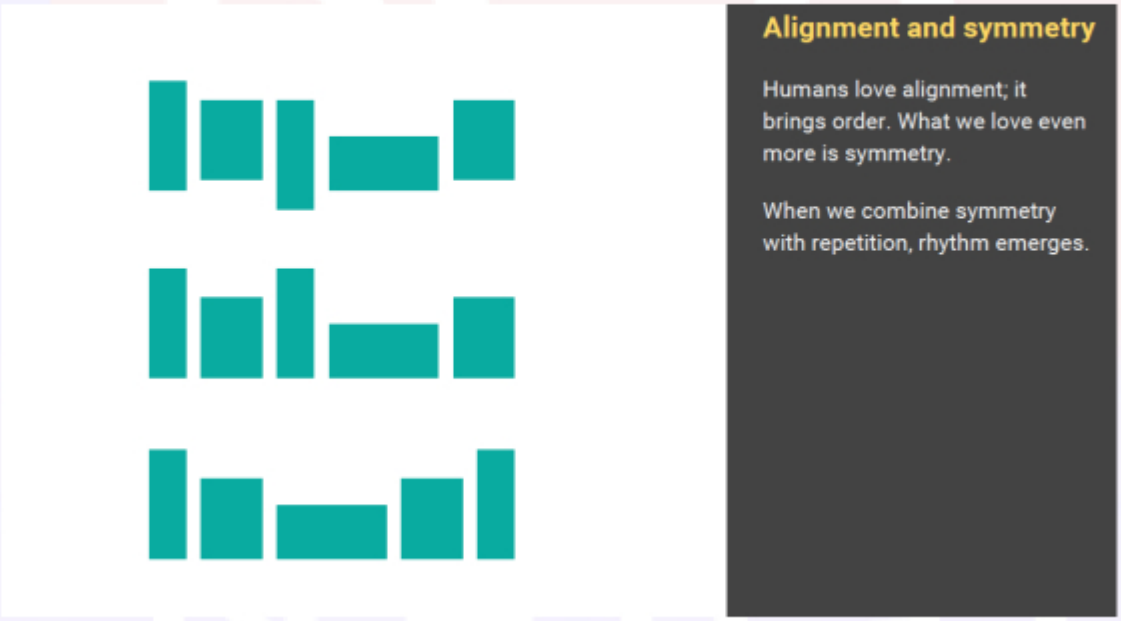
Designing Interfaces: Basic Principles



Repetition

If we change the position of some elements, we suddenly see four groups. It's almost impossible to perceive all elements as one group, even though they are exactly the same.

Designing Interfaces: Basic Principles



Alignment and symmetry

Humans love alignment; it brings order. What we love even more is symmetry.

When we combine symmetry with repetition, rhythm emerges.

Designing Interfaces: Basic Principles

This text has high contrast
and is easy to read.

This text has low contrast
and is harder to read.

Contrast

Contrast shows how different something is from its surroundings.

Feedback Messages

- ***Status information***: keep user informed of what's going on, helpful when user has to wait for response
- ***Prompting cues***: tell user when input is needed, and how to provide the input
  - Should be specific
- ***Warning or Error***: inform user that something is wrong, either with data entry or system operation:
  - Specific, no jargon, don't scold user
  - Same format, placement

Providing Help

- Place yourself in user's place when designing help
- Guidelines:
  - Simplicity
    - Help messages should be short and to the point
    - Use complete sentences and words that enable understanding
  - Organize
    - Information in help messages should be easily absorbed by users – bulleted or ordered list
  - Show
    - It is useful to explicitly show users how to perform an operation
- Use tools to design system help
  - HTML help environment – text editor to construct help pages that can be easily linked

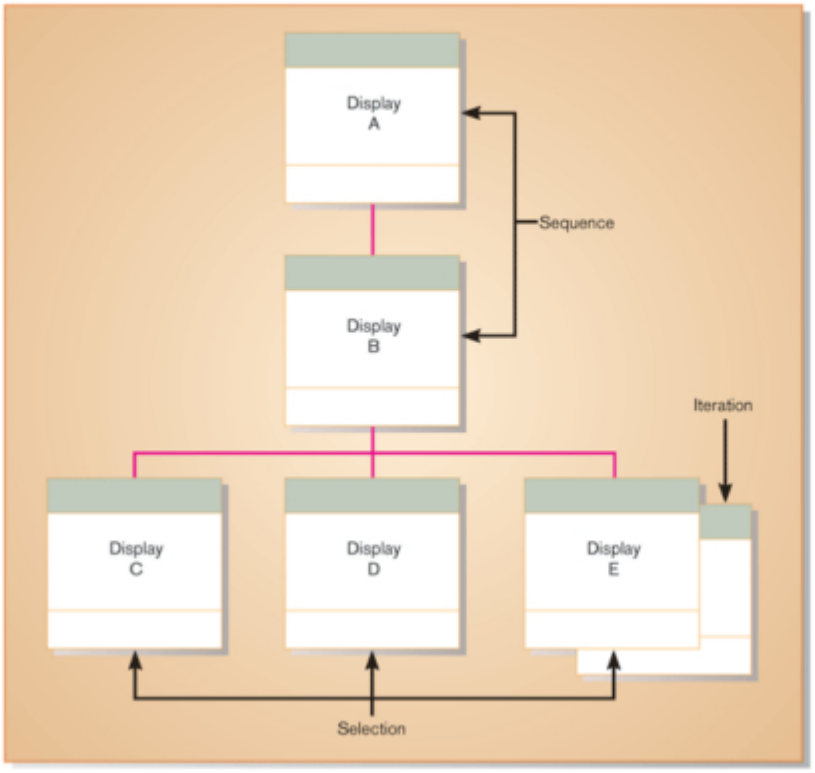
What is a Dialogue?

- A sequence of interactions between the system and a user
- Dialogue design involves:
  - Designing a dialogue sequence
  - Building a prototype
  - Assessing usability

Guidelines for Dialogue Design

- Consistency
- Shortcuts and Sequence (natural)
- Feedback for every user action (confirm add, delete)
- Closure – there are no more screens
- Error Handling – detect, report, make suggestions
- Reversal – undo a delete
- Control
- Ease of use

Figure 12-18 Dialogue diagram illustrating sequence, selection, and iteration



Dialogue diagrams depict the sequence, conditional branching, and repetition of dialogues

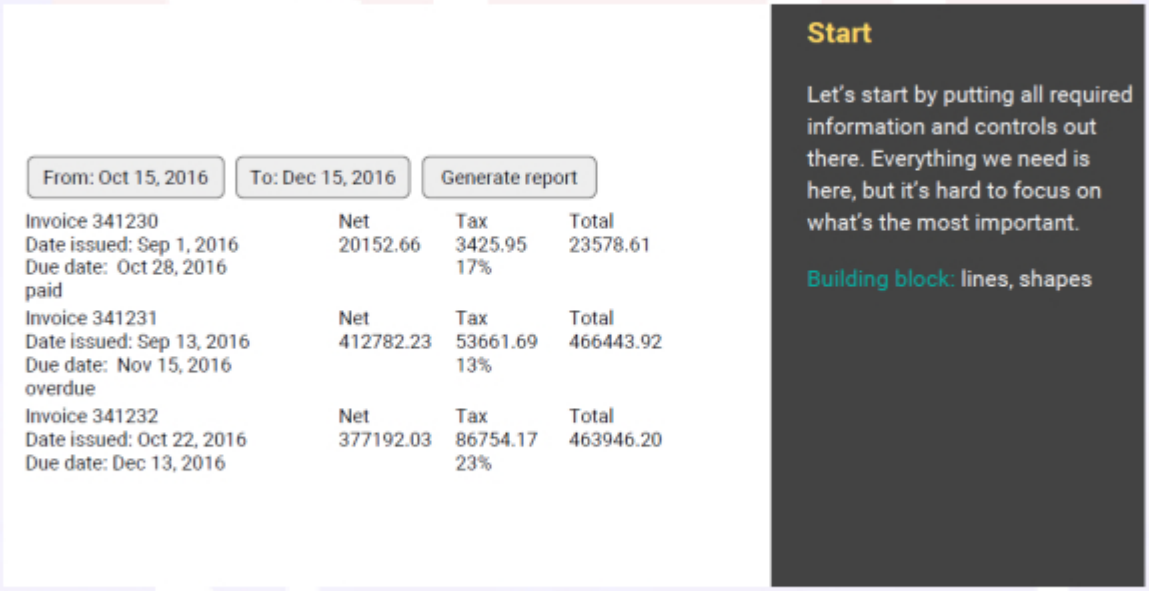
Designing Interfaces: Example

Design **a report** which is a part of a broader financial suite.

The report lists **invoices due in a given period with tax breakdown**.

It's important to mark which invoices are **overdue** and which are already **paid**.

Designing Interfaces: Example

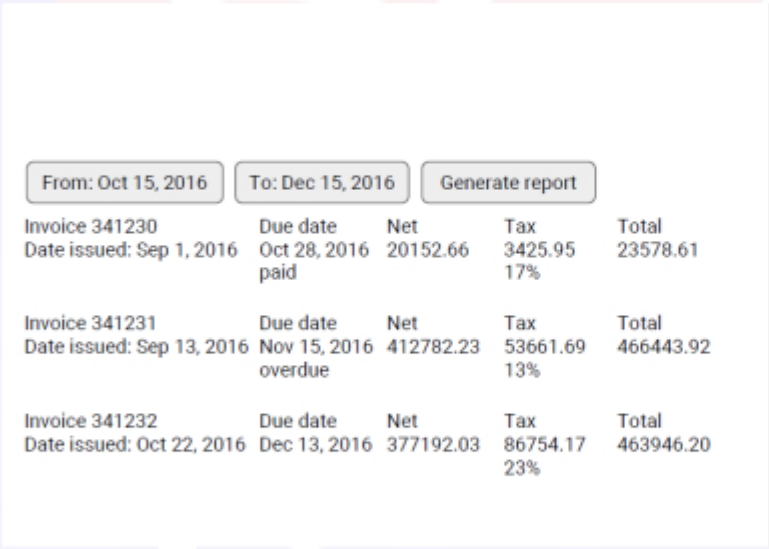


Start

Let's start by putting all required information and controls out there. Everything we need is here, but it's hard to focus on what's the most important.

Building block: lines, shapes

Designing Interfaces: Example



Spread things around

Since "due date" is important, we moved it to a dedicated column. This will make it more prominent and balance everything on the screen.

These rows look very much like a table, so let's do just that.

Principle: balance

Designing Interfaces: Example

From: Oct 15, 2016 | To: Dec 15, 2016 | Generate report

Invoice ID and date issued	Due date	Net	Tax	Total
Invoice 341230 Sep 1, 2016	Oct 28, 2016 paid	20152.66	3425.95 17%	23578.61
Invoice 341231 Sep 13, 2016	Nov 15, 2016 overdue	412782.23	53661.69 13%	466443.92
Invoice 341232 Oct 22, 2016	Dec 13, 2016	377192.03	86754.17 23%	463946.20

Move to a table

Redundant labels are removed. A new header row is created to explain data in the table. Lines are drawn between rows to create a clear boundary.

Building block: line
Principle: grouping

Designing Interfaces: Example

From: Oct 15, 2016 | To: Dec 15, 2016 | Generate report

Invoice ID and date issued	Due date	Net	Tax	Total
Invoice 341230 Sep 1, 2016	Oct 28, 2016 paid	20152.66	3425.95 17%	23578.61
Invoice 341231 Sep 13, 2016	Nov 15, 2016 overdue	412782.23	53661.69 13%	466443.92
Invoice 341232 Oct 22, 2016	Dec 13, 2016	377192.03	86754.17 23%	463946.20

Align numbers

Numbers are much easier to scan and compare if they're aligned to the right.

Principle: alignment

Designing Interfaces: Example

From: Oct 15, 2016 | To: Dec 15, 2016 | Generate report

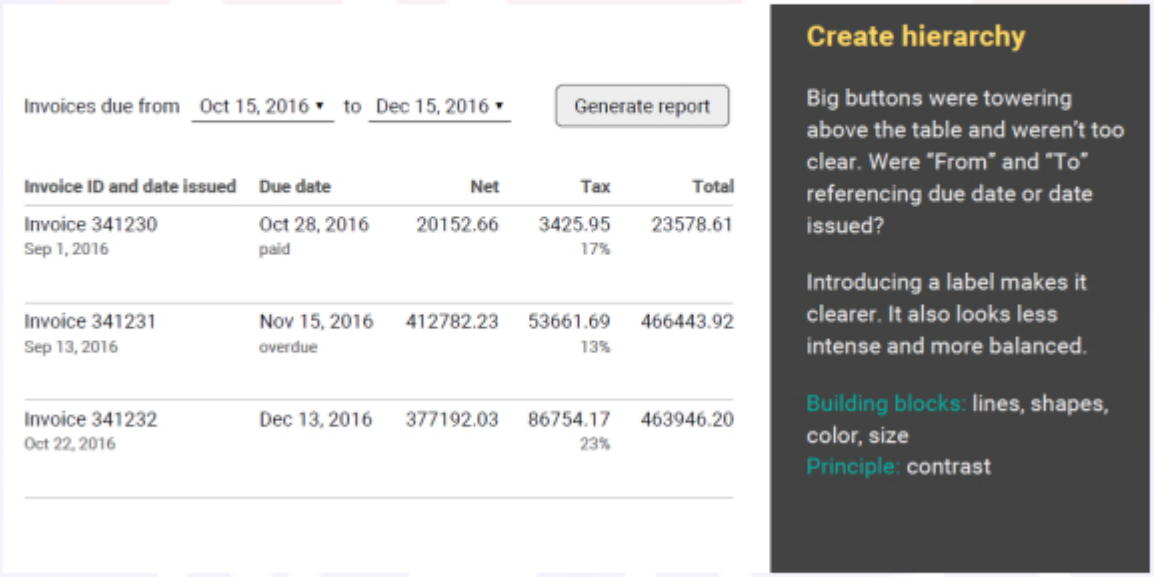
Invoice ID and date issued	Due date	Net	Tax	Total
Invoice 341230 Sep 1, 2016	Oct 28, 2016 paid	20152.66	3425.95 17%	23578.61
Invoice 341231 Sep 13, 2016	Nov 15, 2016 overdue	412782.23	53661.69 13%	466443.92
Invoice 341232 Oct 22, 2016	Dec 13, 2016	377192.03	86754.17 23%	463946.20

Create hierarchy

We need to show what information is the most important. For example, headers, issue dates, and tax percentages aren't. By decreasing them in size and changing color to gray, we decreased the contrast between them and the background. The lines between rows were also too strong and they got a similar treatment.

Building blocks: color, size
Principle: contrast

Designing Interfaces: Example



Create hierarchy

Big buttons were towering above the table and weren't too clear. Were "From" and "To" referencing due date or date issued?

Introducing a label makes it clearer. It also looks less intense and more balanced.

Building blocks: lines, shapes, color, size
Principle: contrast

Designing Interfaces: Example

Invoices due from Oct 15, 2016 ▾ to Dec 15, 2016 ▾ Generate report

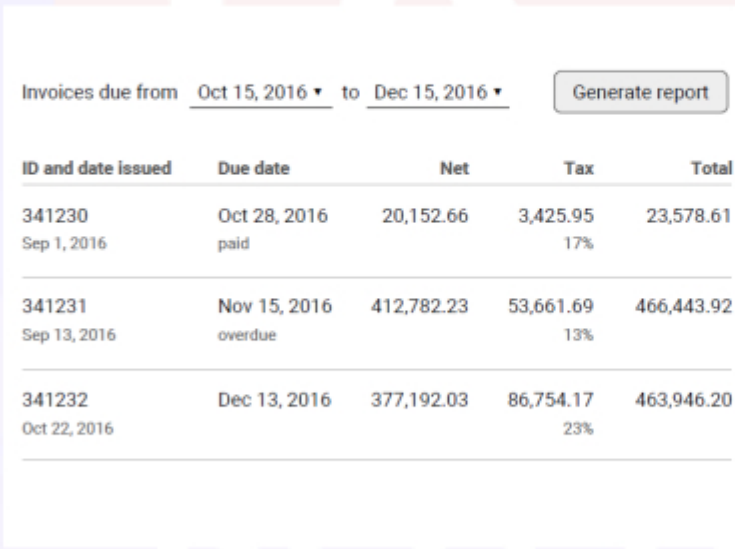
ID and date issued	Due date	Net	Tax	Total
341230 Sep 1, 2016	Oct 28, 2016 paid	20152.66	3425.95 17%	23578.61
341231 Sep 13, 2016	Nov 15, 2016 overdue	412782.23	53661.69 13%	466443.92
341232 Oct 22, 2016	Dec 13, 2016	377192.03	86754.17 23%	463946.20

Remove redundancy

The first column contained too many words "invoice". We removed them and got more space for other columns.

As a general rule, removing redundant and unnecessary elements always improves clarity and focuses on what is important.

Designing Interfaces: Example



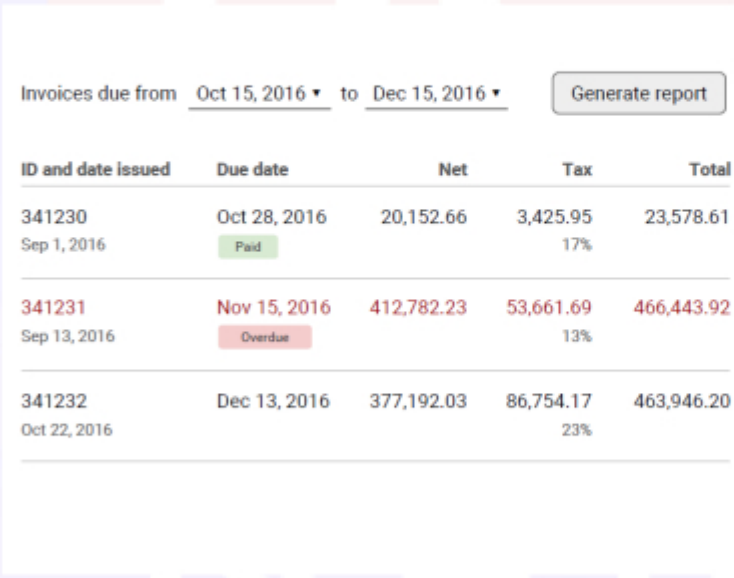
Group

Long numbers are hard to read. We formatted monetary values by creating groups of three digits. This makes them easy to scan and compare.

The same principle is used for grouping credit card and phone numbers.

Principle: grouping

Designing Interfaces: Example

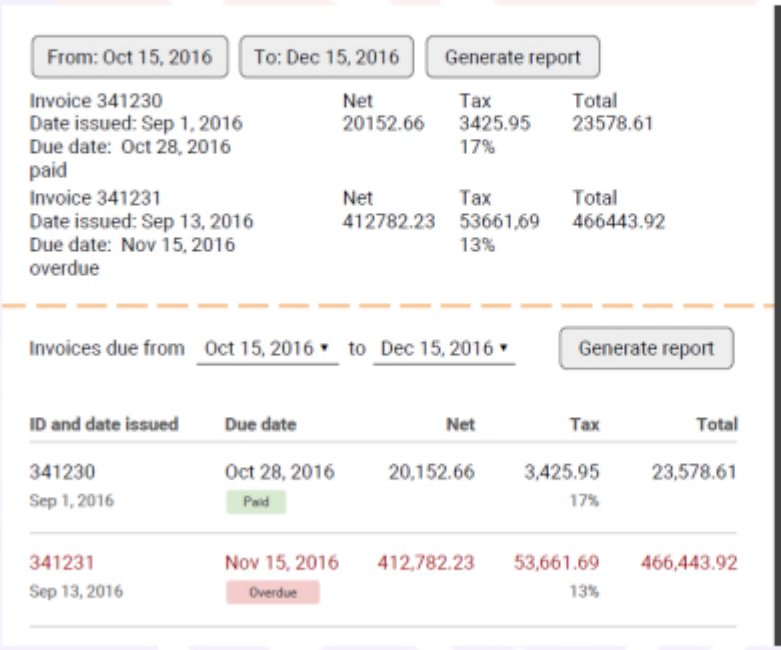


Highlight details

We added color to critical elements. This way they're different enough from the rest to be immediately noticeable. Even if someone has a type of color blindness that would prevent them from discerning red from green, the text and the width of "Paid" and "Overdue" labels are different enough to be recognized.

Building blocks: shape, color
Principle: contrast

Designing Interfaces: Example



Before and after

Each step was simple and logical; it didn't seem like a big change. But many steps together led to a complete transformation.

Visual design shouldn't look like magic anymore. It is a chain of deliberate decisions following basic principles.