Relational Databases & Queries

Slides by Robert Grauer, Keith Mast, Mary Anne Poatsy and Bill Wilson, Geoff Whale, Ashesh Mahidadia and Angela Finlayson

Objectives

- Design data
- Create tables
- Understand table relationships
- Share data with Excel
- Establish table relationships
- Create a single-table query
- Specify criteria for different data types
- Copy and run a query
- Use the Query Wizard
- Create a multi-table query
- Modify a multi-table query

Objectives (continued)

- Specify criteria for different data types
- Copy and run a query
- Use the Query Wizard
- Create a multi-table query
- Modify a multi-table query

Terminology

- **Table**: a relation, i.e. a collection of tuples, i.e. a collection of records
- **Attribute**: a column in the table
- **Database**: a collection of tables

  In Access, the queries, forms, reports, etc. are considered part of the database.

Table Design: Designing Data

- Table Definition – Revised
  - A table consists of records (tuples)
  - Records are made up of one or more fields (components of the tuples)
- Input vs. Output in Design
  - Designer needs to consider what data is available, and what outputs are needed
  - Then the designer has to connect the two…

Guidelines for Designing Fields

1. Include the necessary data
2. Design for now & the future
3. Store data in its smallest parts
4. Add calculated fields to a table
5. Design to accommodate date arithmetic
6. Link tables using common fields

Include Necessary Data

- Determine what data is necessary
- Create a rough draft of reports that may be needed
- Create tables based on fields necessary for reports

Design for Now and the Future

- Organizations evolve over time
- Databases should evolve with the organization
  - Anticipate future needs of the organization
  - Build flexibility into system to satisfy future needs
  - Design in such a way that later revisions don't require a massive reorganisation of the database

Store Data in Smallest Possible Pieces

- Creating a name field with the entire name in it ("Ms Jane Smith") can make sorting by surname difficult.
- Divide data into the smallest pieces that you're going to need to access
  - Example:

    Prefix, FirstName, LastName, Suffix

    Group Captain, Lionel, Mandrake, DFC

    Professor, John, Dwyer, OA
- Provide flexibility for the user

Calculated Fields in a Table

- Produce a value from an expression or function that references one or more existing fields
- Access 2010 allows the user to store calculated fields
  - Can be a benefit or a potential problem
  - Exercise caution when using calculated (derived) fields

Design to Accommodate Date Arithmetic

- Despite the advice about storing data in its smallest parts, dates are typically stored in a single field (9/11/2001).
- Microsoft represents dates as number of days since 1899-12-31, so this works out OK.
- Calculated fields can also create date/time data

Link Tables Using Common Fields

- Tables may be joined based on a common field
- Join lines are created
  - Manually by the user, or
  - Automatically by Access when two fields in separate tables share the same name between two related tables
- Avoid Data redundancy errors
  - The unnecessary storage of duplicate data in two or more tables

Multiple Tables

- What is wrong with the following table?

Order ID	Description	Quantity	Unit Price	Family Name	Given Name	Address
O001	Widget	5	$1.50	Doe	Jane	412 Anzac Pde, KINGSFORD 2032
O002	Widget	12	$1.50	Smith	John	3 Main St, YOUNG 2594
O003	Gadget	7	$2.05	Doe	Jane	412 Anzac Pde, KINGSFORD 2032
O004	Widget	14	$1.50	Doe	Jane	412 Anzac Pde, KINGSFORD 2032
O005	Gadget	4	$2.05	Smith	John	3 Main St, YOUNG 2594
O006	Gadget	23	$2.05	Smith	John	3 Main St, YOUNG 2594
O007	Widget	3	$1.50	Doe	Jane	412 Anzac Pde, KINGSFORD 2032

Multiple Tables

- **Parts**

Part ID	Description	Unit Price
P01	Widget	$1.50
P02	Gadget	$2.05

Multiple Tables

- **Customers**

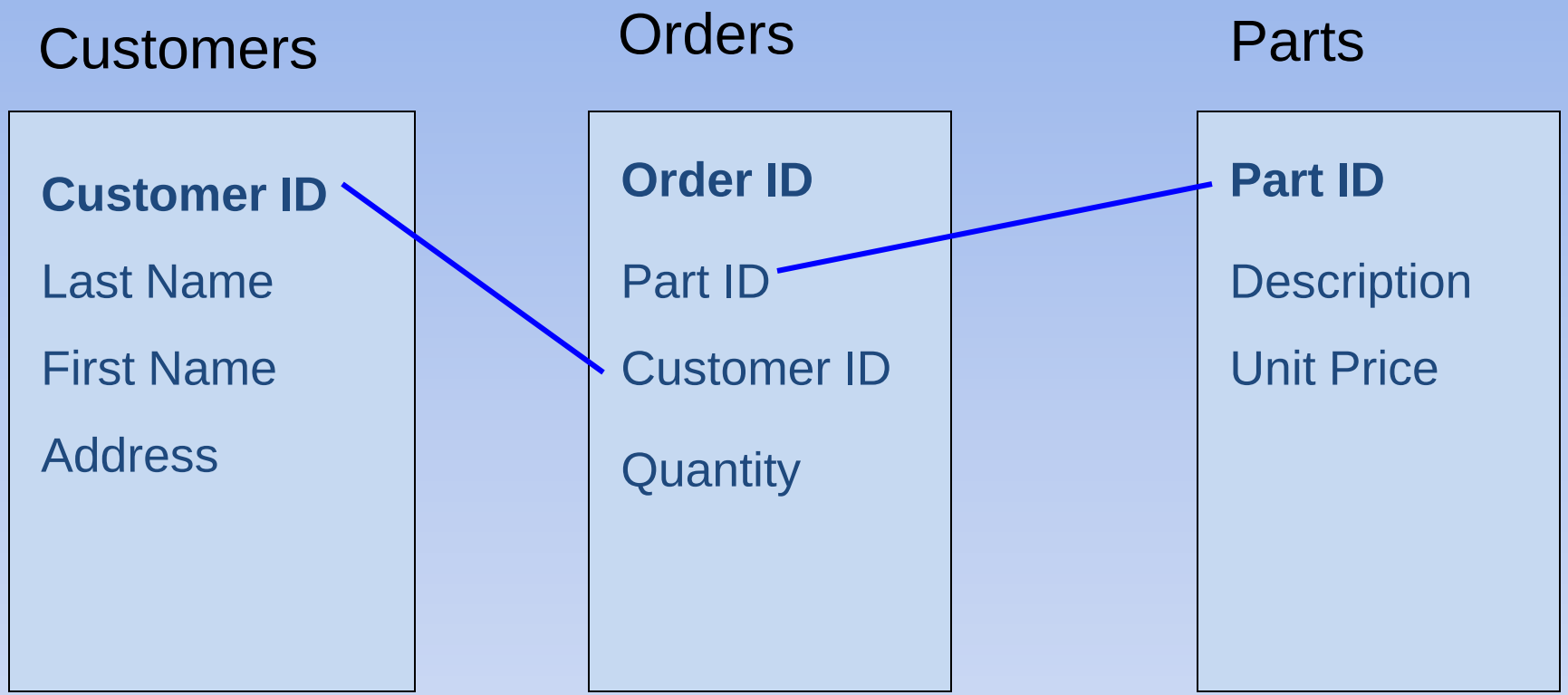
Customer ID	Family Name	Given Name	Address
C01A	Doe	Jane	412 Anzac Pde, KINGSFORD 2032
C02A	Smith	John	3 Main St, YOUNG 2594

Multiple Tables

- **Orders**

Order ID	Customer ID	Part ID	Quantity
O001	C01A	P01	5
O002	C02A	P01	12
O003	C01A	P02	7
O004	C01A	P01	14
O005	C02A	P02	4
O006	C02A	P02	23
O007	C01A	P01	3

Multiple Tables



Multiple tables

- The previous example illustrates some of the advantages of multiple tables. In a single table with multiple relationships,
  - Data is duplicated many times
  - Storage space is wasted
  - Likelihood of introducing errors increases
- In designing a database we need to **identify** the tables required and **create relationships** between them to show how the data they contain is related
- Relationships exist between a field in one table (usually a primary key) and a field in another table

Primary Keys and Foreign Keys

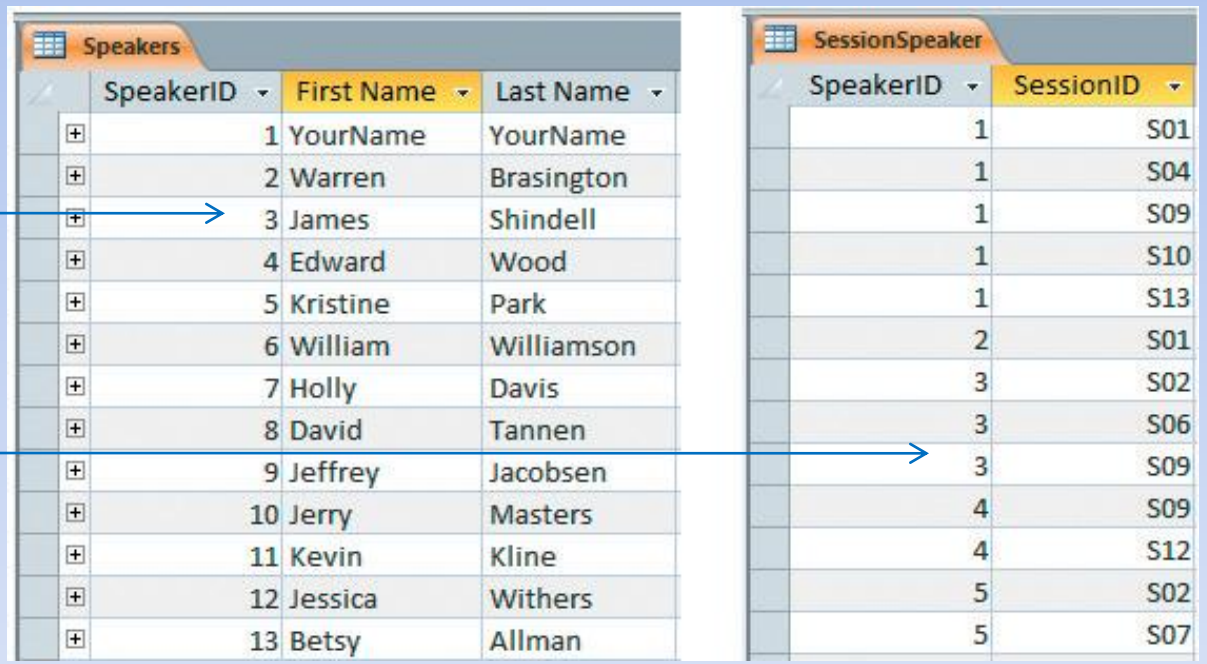
- Notice that *Customer ID* field occurs in both Customers table and Orders table.
- In Customers table it **uniquely identifies** each record
  - *Customer ID* is the **primary key** of the Customers table
- In the Orders table *Customer ID* field does **not** uniquely identify each record but serves to "link" to the Customers table
  - *Customer ID* is a **foreign key** in the Orders table

Foreign Key Example

A *foreign key* is a field in one table that is also a primary key of another table

SpeakerID is the primary key of the Speakers table

SpeakerID is a foreign key in the SessionSpeaker table (duplicates are allowed)



Speakers

SpeakerID	First Name	Last Name
1	YourName	YourName
2	Warren	Brasington
3	James	Shindell
4	Edward	Wood
5	Kristine	Park
6	William	Williamson
7	Holly	Davis
8	David	Tannen
9	Jeffrey	Jacobsen
10	Jerry	Masters
11	Kevin	Kline
12	Jessica	Withers
13	Betsy	Allman

SessionSpeaker

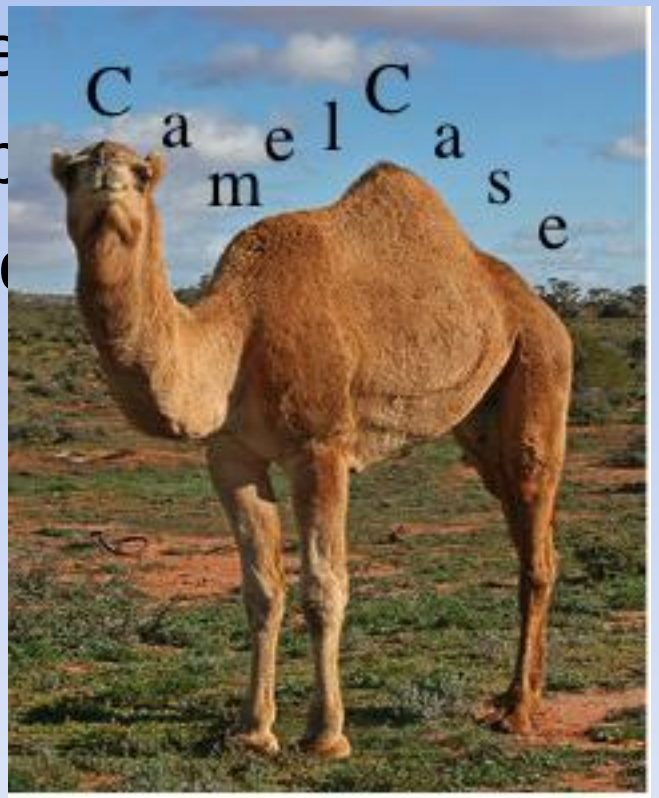
SpeakerID	SessionID
1	S01
1	S04
1	S09
1	S10
1	S13
2	S01
3	S02
3	S06
3	S09
4	S09
4	S12
5	S02
5	S07

Creating Tables

- Create fields in Design View
- Import data from another database or application
  - Examples: Excel spreadsheets or Word text files
- Enter data directly into rows in Datasheet view

Creating Fields in Tables

- Field names should be meaningful
- Rules for naming fields:
  - Length can be up to 64 characte
  - Can include letters, numbers, sp
  - Access uses **CamelCase** notati
    - Example: ProductCost



Field Data Types

- Every field has a data type
- Determines:
  - The type of data that can be entered
  - The operations that can be performed on that data
- Access recognizes 10 data types

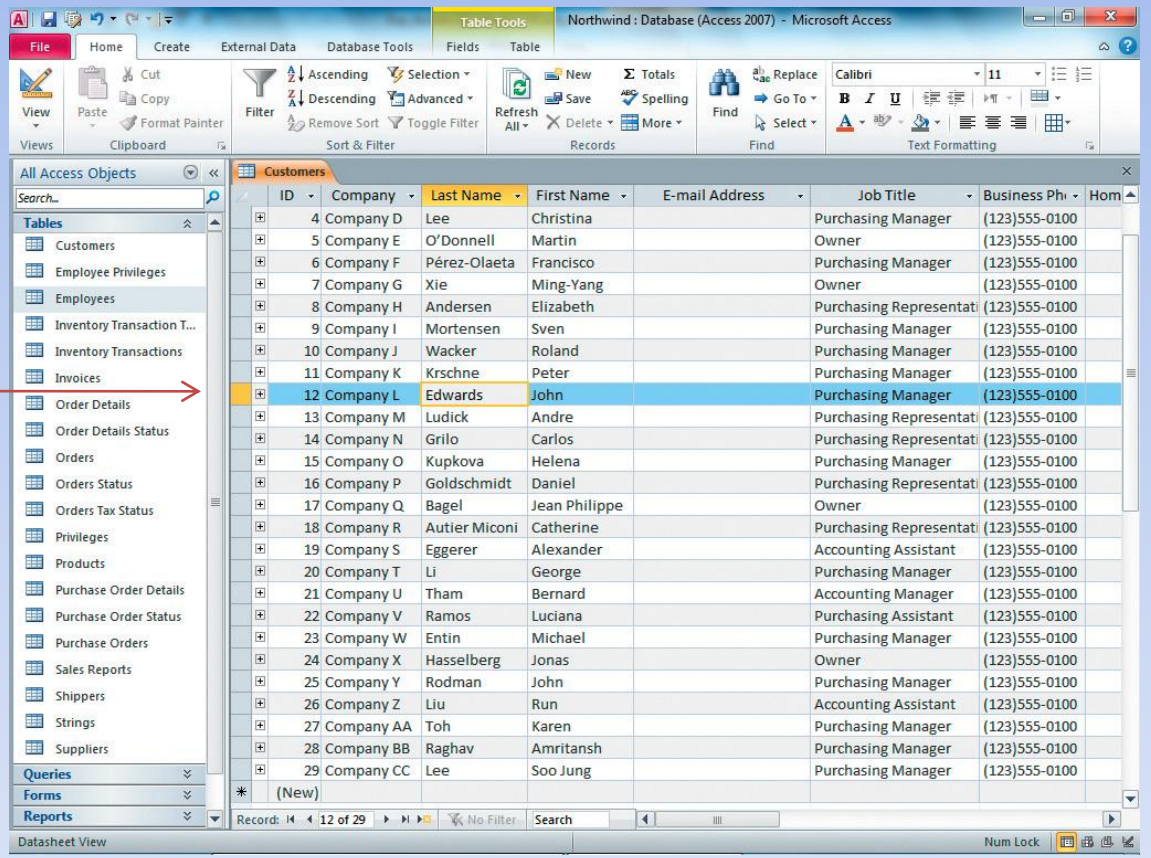
Access Data Types

- Number
- Text
- Memo
- Date/Time
- Currency
- Yes/No
- OLE (e.g. spreadsheets, pictures, sounds)
- AutoNumber
- Hyperlink
- Attachment (e.g. workbooks, documents, multiple images)

Using Table Views

Datasheet View

Active record



Using Table Views

- Design View

  Toggle between Datasheet and Design views by clicking Home > Views > View
- PivotTable
- PivotChart

Work with Field Properties

- Field properties
- Data type

  Text data type

  Number data type

  Etc.
- Caption property – a more readable field name
- Validation rule – to help check data is valid

Access Field Properties

Property Type	Description
Field Size	Max. no. characters (text field) or format (number field)
Format	How field is displayed or printed
Input Mask	E.g. slashes in a date
Caption	Alternate name for use in datasheets, forms, reports
Default Value	… for new records
Validation Rule	A rule that data in this field must conform to
Validation Text	A message to display if the validation rule is broken
Required	Does a value have to be entered for this field?
Allow Zero Length	Allows text or memo fields of zero length
Indexed	Increases efficiency of a search on this field. Indexes take up space…
Expression	Used for calculated fields.
Result Type	Format for calculated field results.

Table Design, Etc. Demo 1

- Start Access. Click Blank database. Find Browse icon near File Name box and select location. Type *a02h1bank_demo*.accdb in File Name box. Click OK, click Create.
- Click in the 2nd column and type B10, then tab. Type Mast, tab, Uptown.
- Type 3 tabs. You are now on a new row. Type B20 tab Esposito tab Eastern tab tab tab. Type B30 tab Amoako tab Western tab tab. Type B40 tab Singh tab Southern tab tab. Type B50 tab Hilmer tab Campus. Click in another row to save.
- Click Save. Type Branch, click OK.
- Click Views > View. Click ID field, Tools > Delete Rows, Yes, Yes
- Double-click the Field1 field name, type BranchID. Similarly Field2 becomes Manager, Field3 becomes Location.
- Click BranchID field. Click Tools > Primary Key.
- Click Save.

Table Design, Etc. Demo 2

- Click *BranchID* field name. Modify its properties: Field Size = 5, Caption = "Branch ID". Check Indexed is Yes (No Duplicates)
- Ditto *Manager*: Field Size = 30, Caption = "Manager's Name"
- Ditto *Location*: Field Size = 30, Caption = "Branch Location"
- Create a new field: click in the blank row below Location: type StartDate. Tab to Data Type column. Click arrow at right, select Date/Time. Tab to Description column, type "Date manager started working at this location".
- Click Format property, click arrow at right, select Short Date.
- Click Caption property, type "Manager's Start Date". Click Save, Yes.
- Right-click Branch tab, select Datasheet View. Enter 3/12/2007 for Mast, 18/6/2008, 13/3/2006, 15/9/2009, then click calendar icon for Hilmer, navigate to 1/1/2013. Click Close icon (x) for datasheet, below Ribbon.
- Double-click Branch table in Navigation pane. Check data. Check Print Preview, close it.
- Click File > Compact & Repair Database, then File > Save &Publish > double-click Back Up Database, accept name.

Indexing to Retrieve Data Quickly

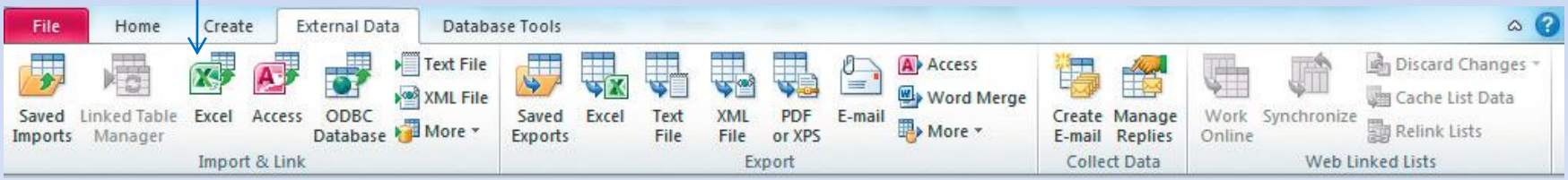
- Provides quick sorting based on the primary key
- Provides quick retrieval of data based on the primary key

Options on External Data Tab

- Import & Link
- Export
- Collect Data
- Web Linked Lists

Click Excel to import spreadsheet data

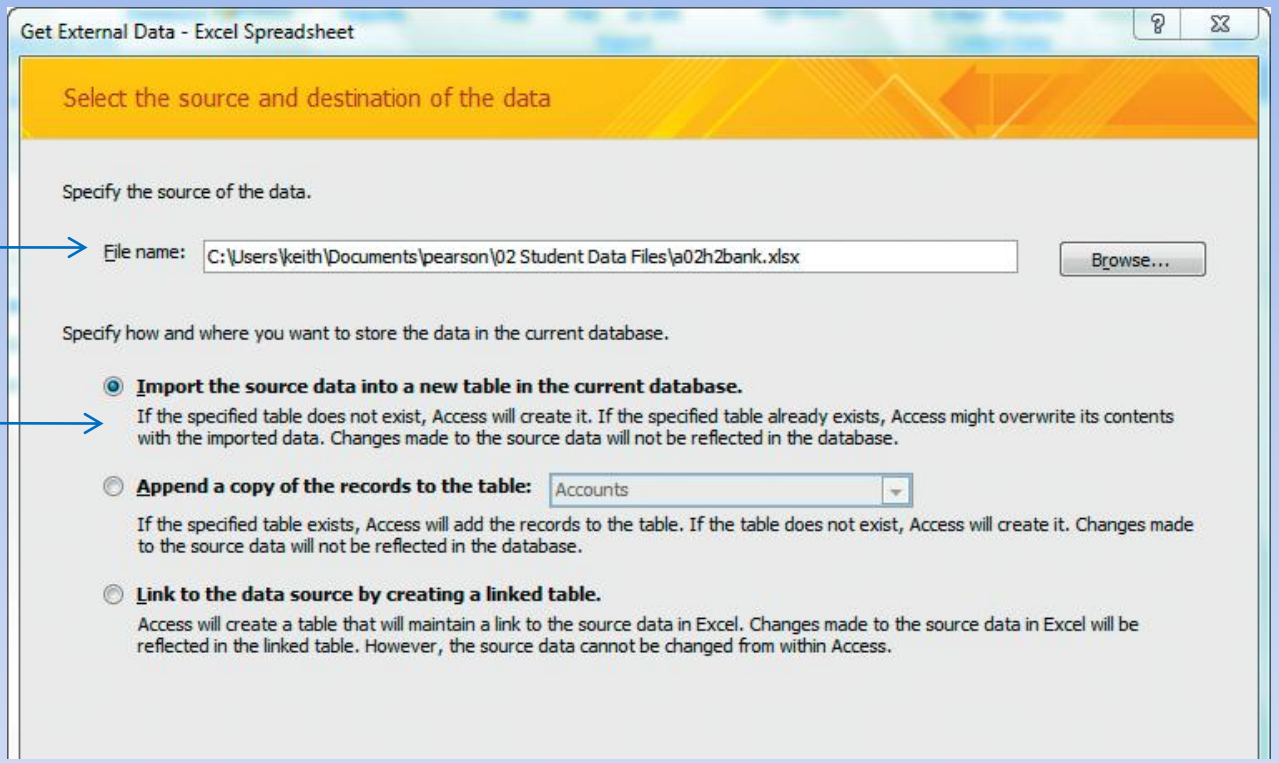
External Data Tab



Import Data from Excel

Click Browse to find a spreadsheet

Decide what you want to do with the data

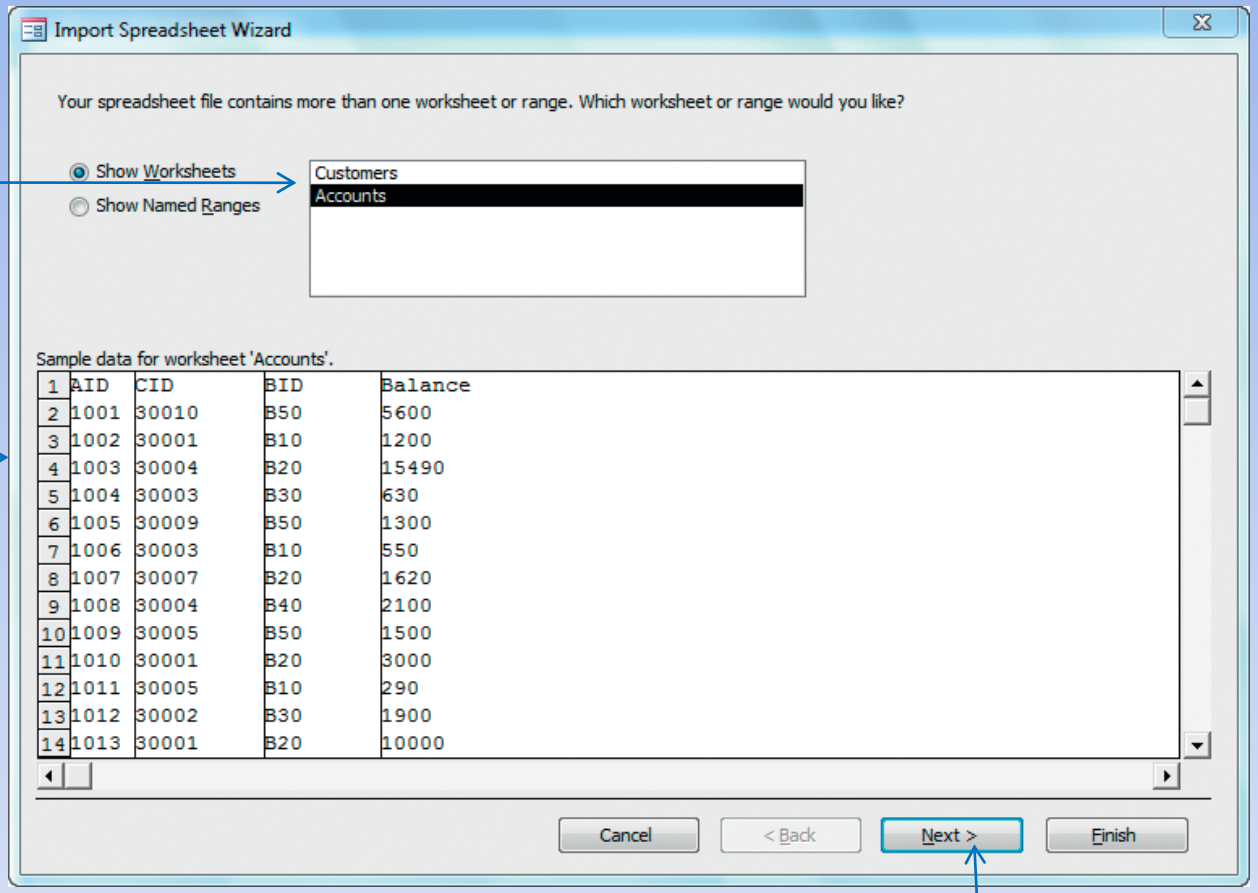


Import Data from Excel 2

Choose the worksheet to import

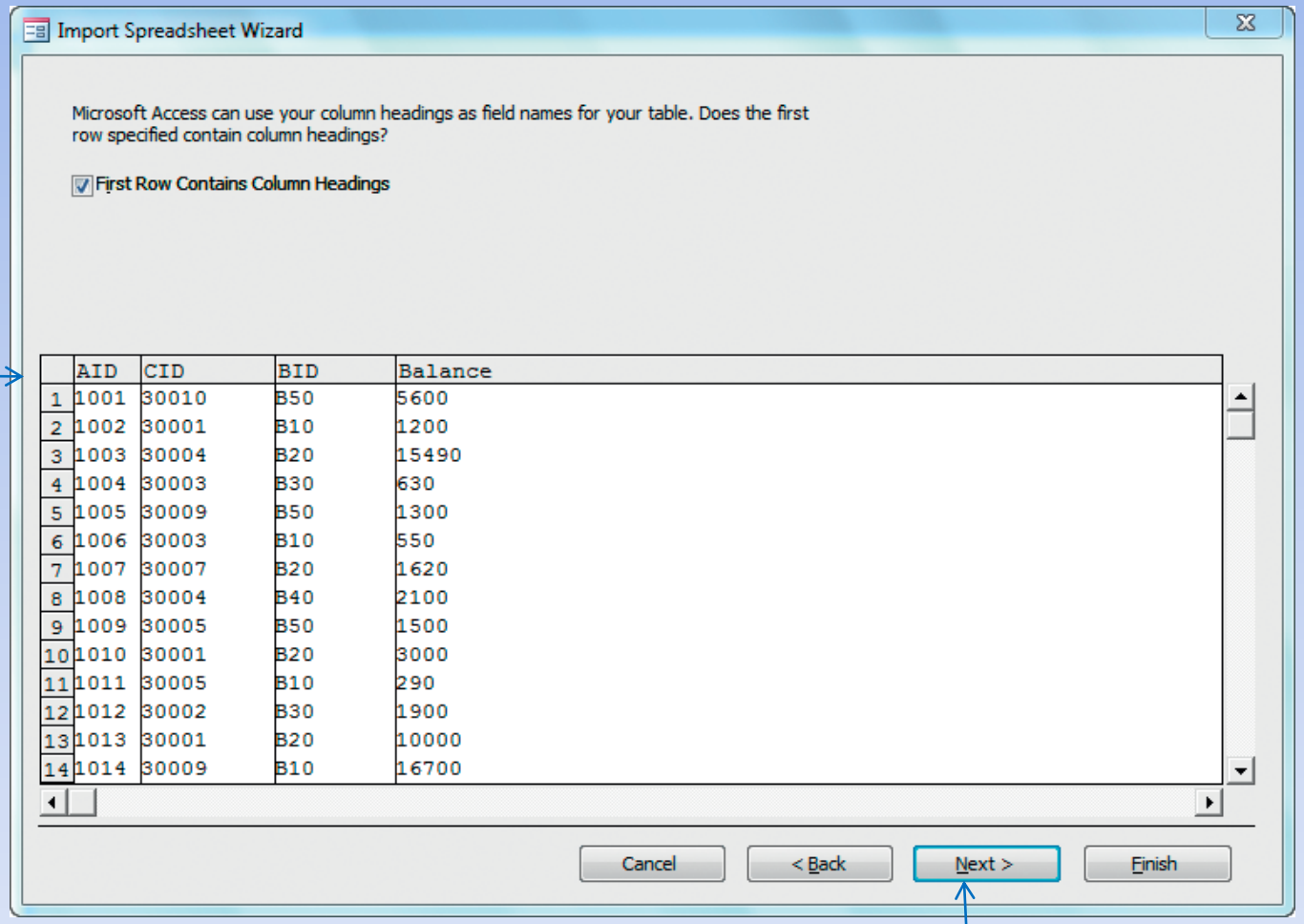
Preview of the worksheet data

Click Next to continue



Import Data from Excel 3

Column headings

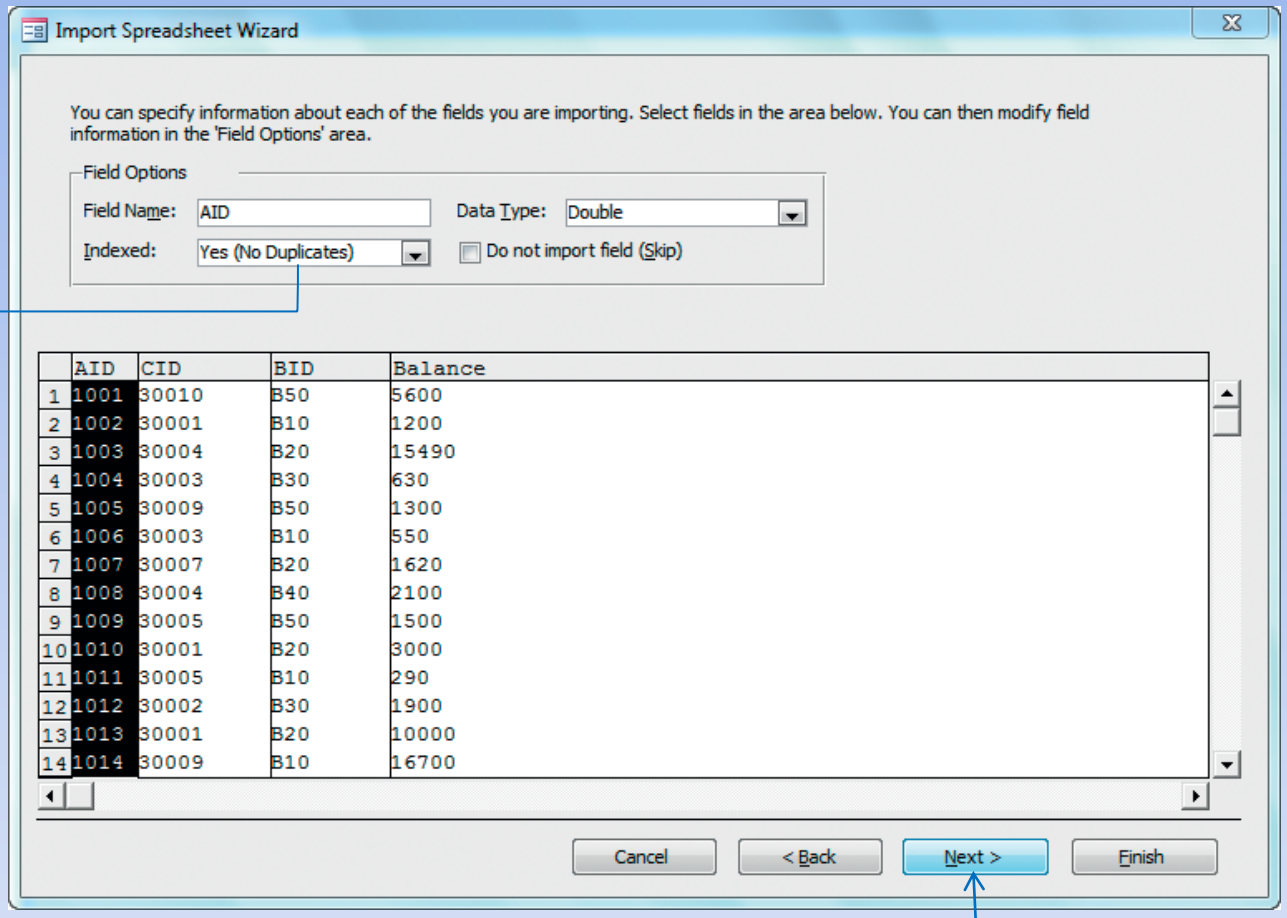


Click Next to continue

Import Data from Excel 4

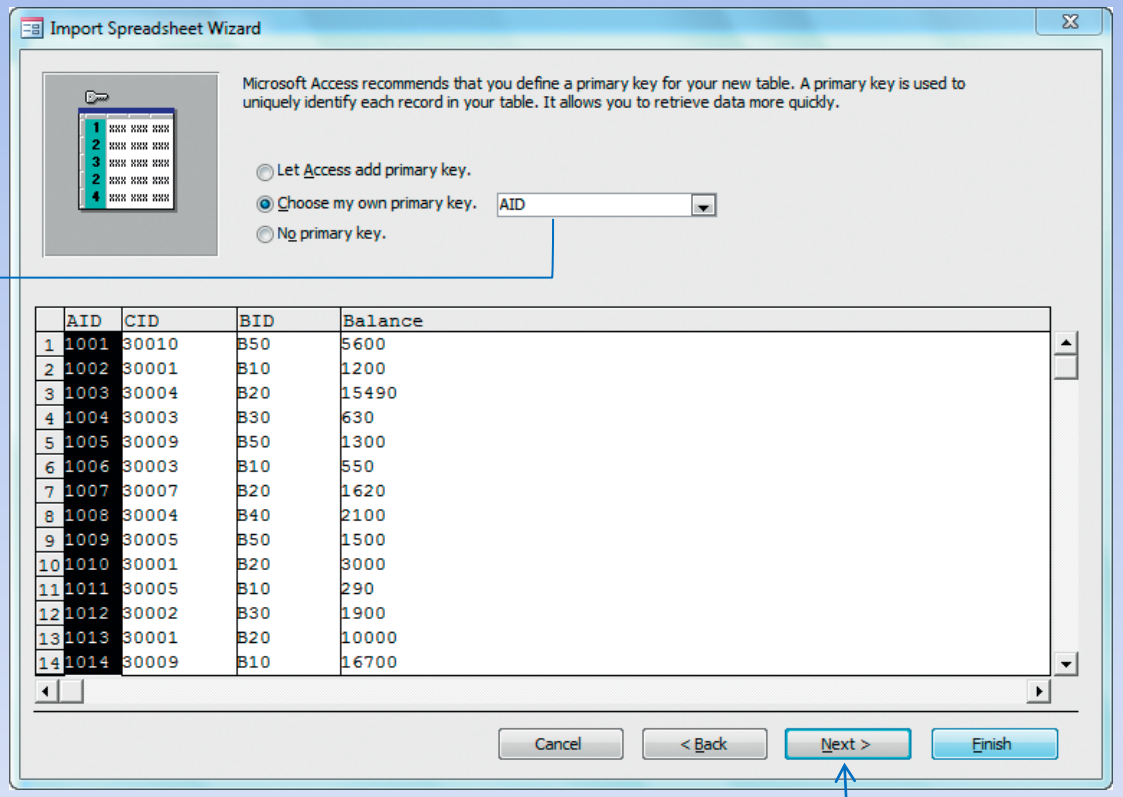
Change Indexed property to *Yes (No Duplicates)*

Click Next to continue



Import Data from Excel 5

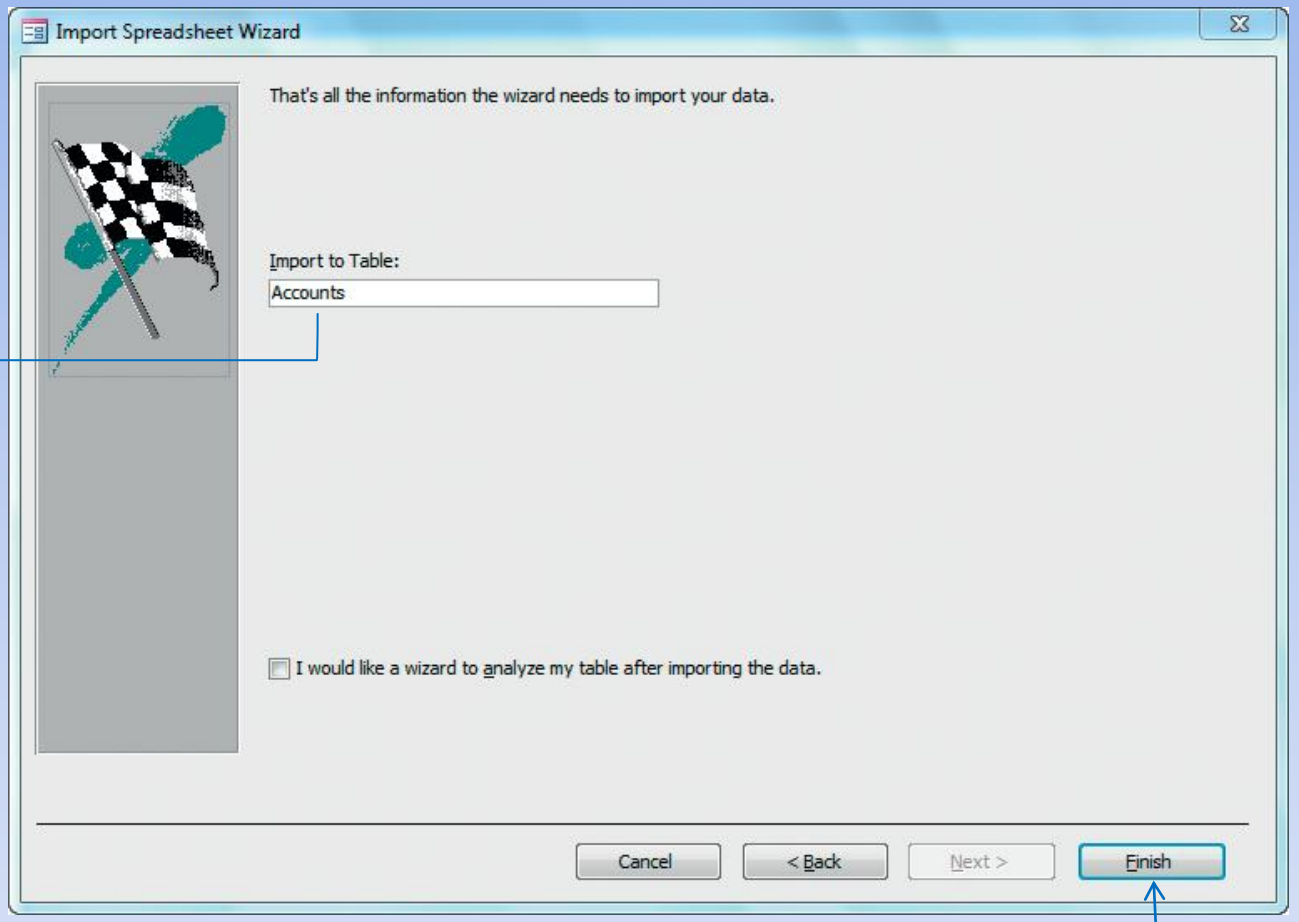
AID becomes the primary key



Click Next
to continue

Import Data from Excel 6

Accounts becomes the table name



Click Finish to import the data

Import from XML document

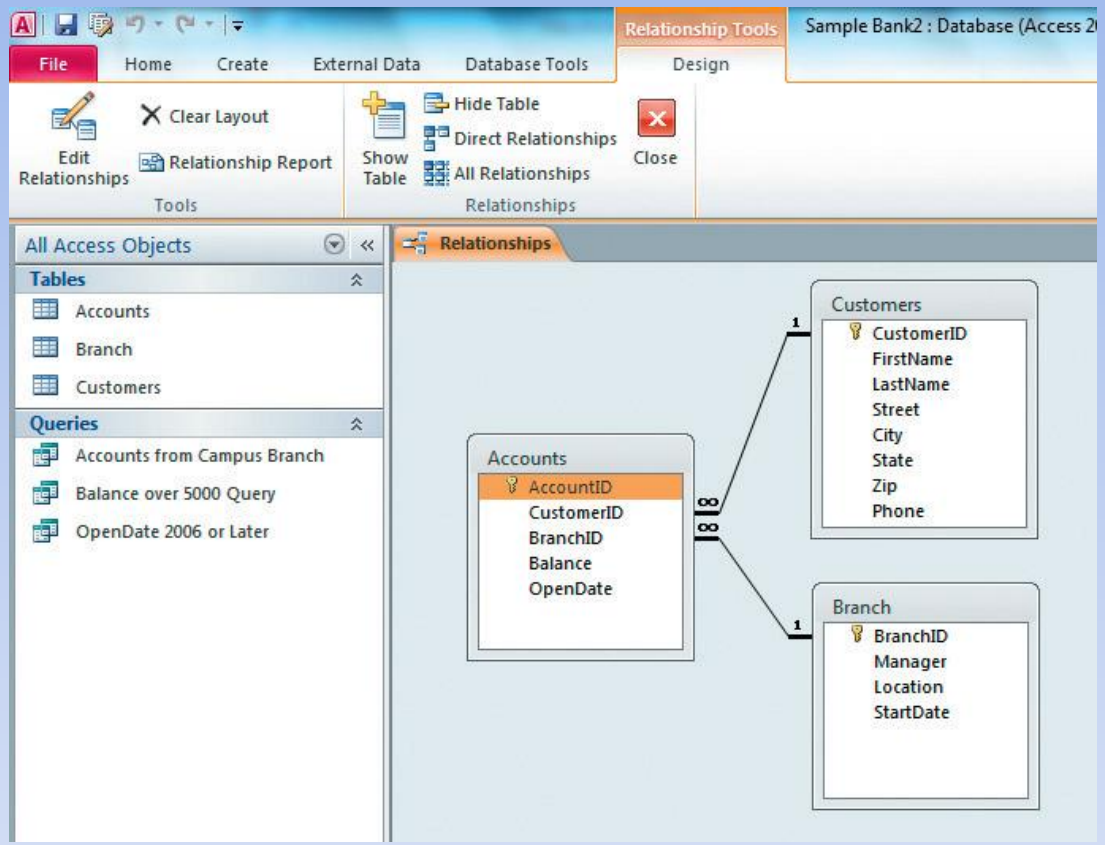
- XML (eXtensible Markup Language) is like HTML, but with arbitrary tag names

```
<course>
   <name>Introduction to Underwater Origami</name>
   <code>ORIG3001</code>
   <instructor>Michiko Suzuki</instructor>
   <UoC>6</UoC>
</course>
```

- Open a database.
- Click External Data > Import & Link > XML file, select file …

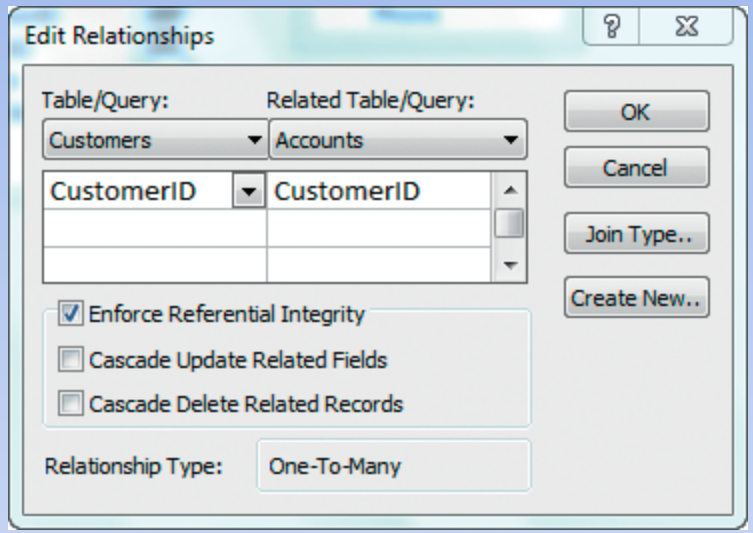
Understanding Table Relationships

- Efficiently combine data from related tables
- Create queries, forms, and reports



Establishing Referential Integrity

- Edit Relationships dialog box:
- Select *Enforce Referential Integrity* checkbox
  - stops user entering a foreign key in a related table unless that value already exists as a primary key in its home table.
  - e.g. an order for a customer that doesn't exist (yet).
  - user gets a message if they try



Set Cascade Options

- Cascade Update Related Fields

  If primary key is altered, related fields are automatically updated

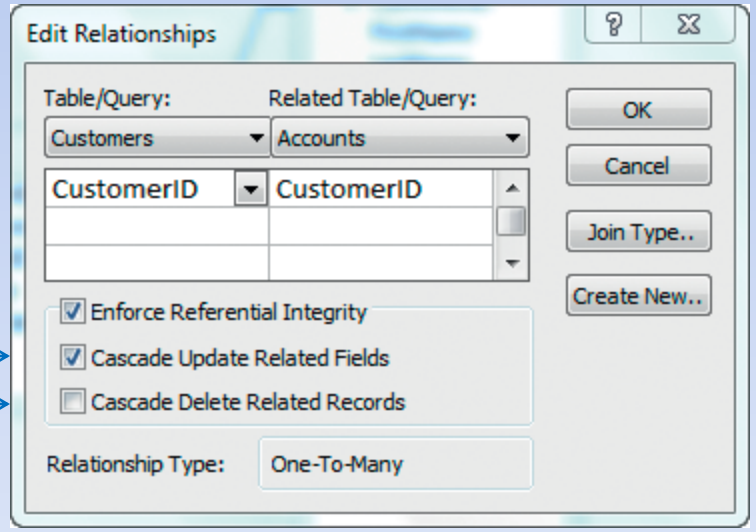
- Cascade Delete Related Records

  Dangerous – automatically deletes all records in related tables that refer to that primary key

Click Enforce Referential Integrity

Click Cascade Update in case the primary key changes

Probably best not to click Cascade Delete Related Records



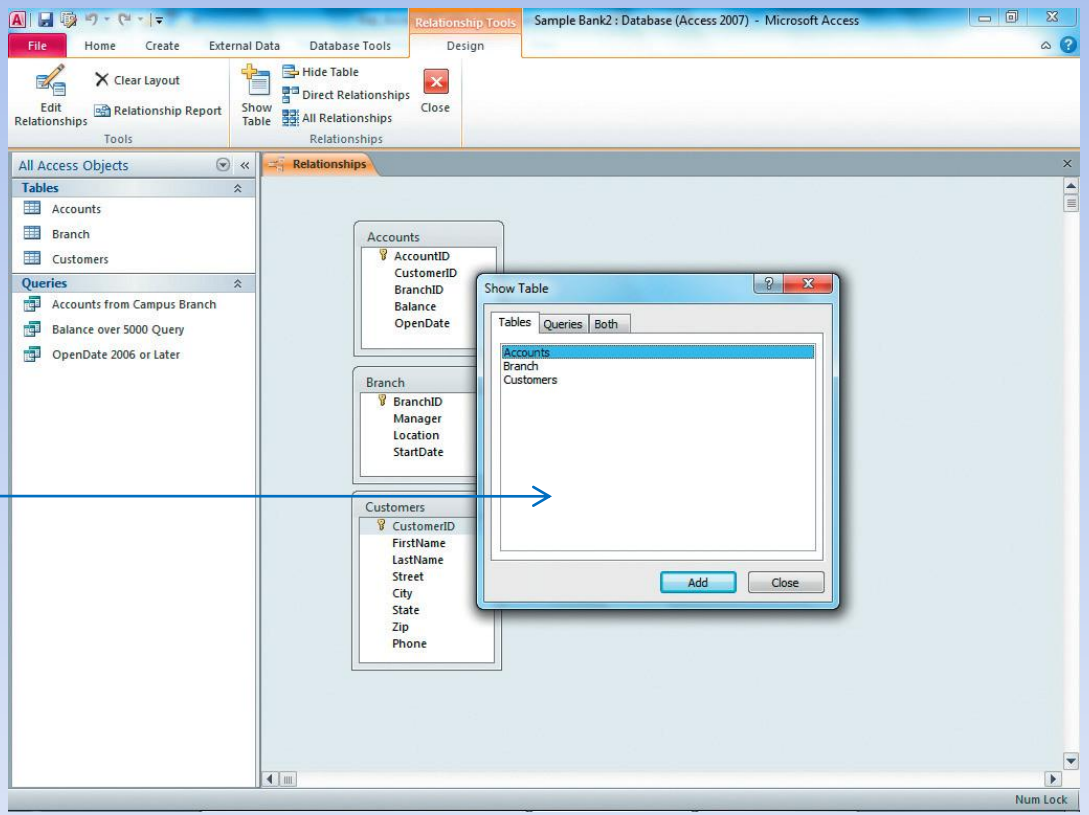
Types of Relationships

- **One-to-one relationship**
  - One person has one driver's license and one driver's license belongs to one person
- **One-to-many relationship**
  - One person may borrow many dvds and each dvd is borrowed (at any time) by one person.
- **Many-to-many relationship**
  - A student may take many subject
  - A subject may have many students

Establishing a One-to-Many Relationship

- Open Relationships window
- Add tables
- Establish relationships

Show Table window

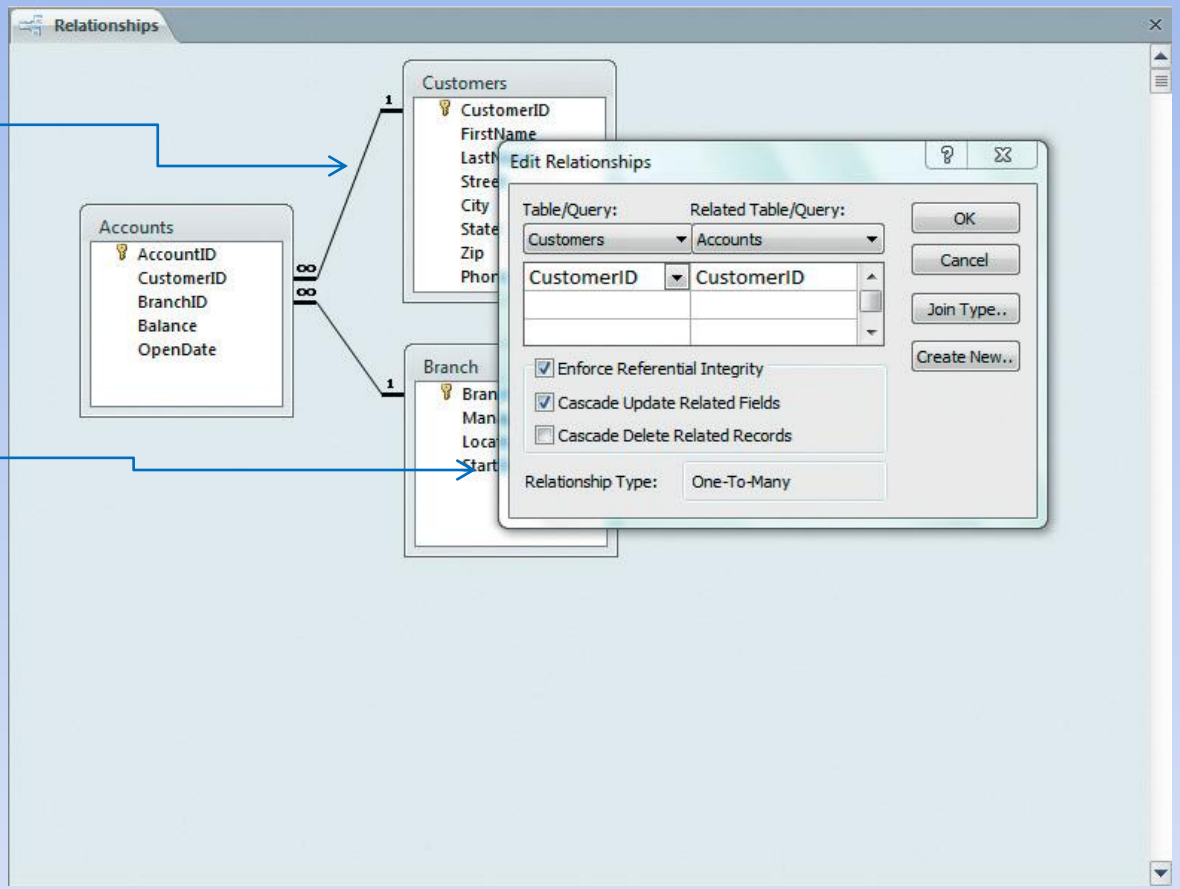


Relationships Between Tables

One-to-many relationships

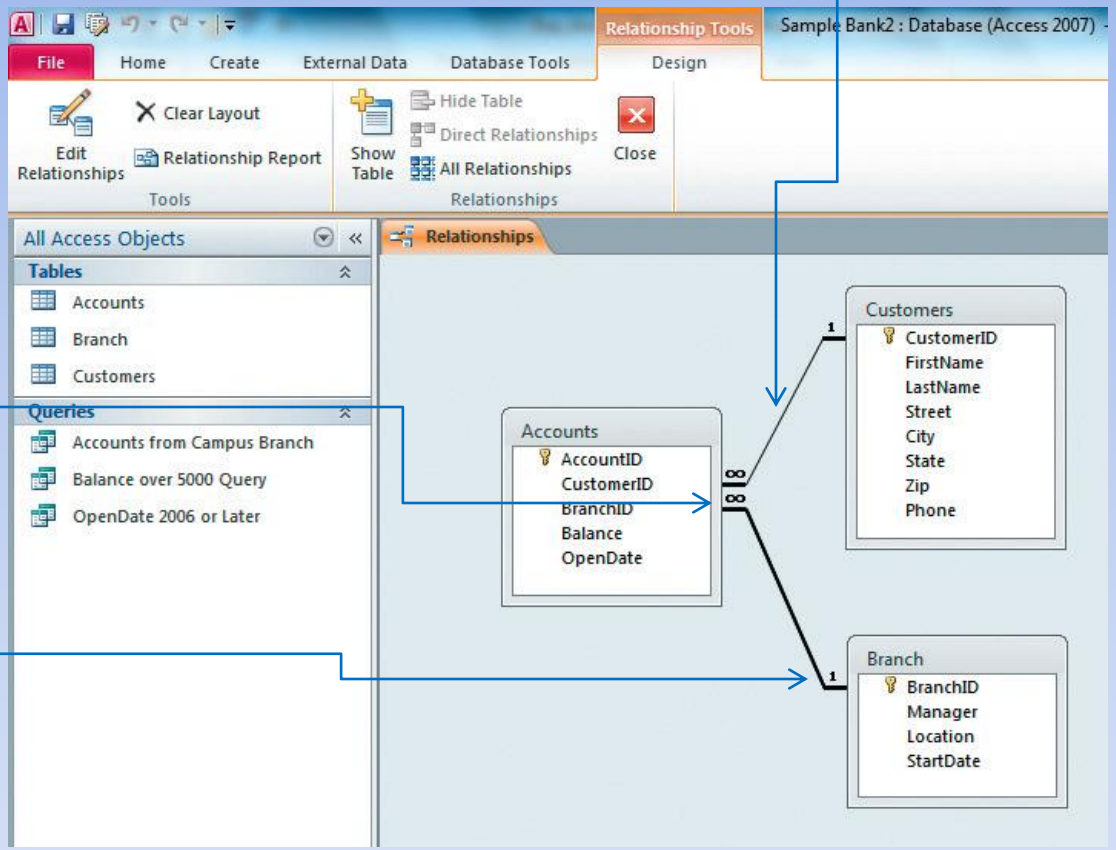
Edit Relationships dialog box

Right-click on join lines to edit the relationship



Relationships Window

Join lines



Many side: the "∞" (infinity symbol)

1 side

Multiple-table Databases Demo 1

- Open a02h1bank_demo.accdb – click File > Save Database As, rename it. Click Enable Content.
- Click External Data > Import & Link > Excel. *Select Import the source data in to a new table in the current database*, if necessary.
- Click Browse, find *a02h2bank.xlsx*, open it and start the Import Spreadsheet Wizard.
- Worksheet names are shown in the upper window. Select Accounts, click Next. Ensure "First Row contains column headings" is checked, click Next.
- Ensure AID is selected, click Indexed arrow, select Yes (No Duplicates), click Next.
- Click *Choose my own primary key*, select AID, click Next.
- "Accounts" is a good name, click Finish, then click Close.
- Open Accounts table, check it out, close it.

Multiple-table Databases Demo 2

- Next we import the other worksheet.
- Click External Data > Import & Link > Excel. *Select Import the source data in to a new table in the current database*, if necessary.
- Click Browse, find *a02h2bank.xlsx* again, open it and start the Import Spreadsheet Wizard.
- Customers worksheet is selected, click Next. Ensure "First Row contains column headings" is checked, click Next.
- Ensure CID is selected, click Indexed arrow, select Yes (No Duplicates), click Next.
- Click *Choose my own primary key*, select CID, click Next.
- "Customers" is a good name, click Finish, then click Close.
- Open Customers table, check it out, close it.

Multiple-table Databases Demo 3

- We need to adjust the imported tables.
- Double click the Accounts table. Change to Design view.
- Change AID to AccountID. Change Field Size property to Long Integer. Type Account ID in the Caption box.
- Change CID to CustomerID, Field Size: Long Integer, Caption: Customer ID.
- Change BID to Branch ID, Field Size is 5, Caption is Branch ID.
- Change Data Type of Balance field from Number to Currency.
- Type a new field name, OpenDate under the Balance field name, Data Type is Date/Time, then add Short Date format in Field Properties.
- Change to Datasheet View. Read messages, click Yes twice.
- Add OpenDate values. Right-click Customers table in Navigation pane, select Design View. Change CID field name to Customer ID. Long Integer, Caption. Text fields sizes 20 or State 2. Zip & Phone Text, length 15.
- Remove @ symbol from Format properties.
- Click Phone field name, click Input Mask in Field Properties. Click Build on right side. Click Yes to save table, then Yes again. Click Finish to apply default phone number Input Mask.
- Click Save to save the design changes to Customers table. Read warning, Yes.

Multiple-table Databases Demo 4

- Add a new record to the Customers table: switch to Datasheet view.
- Click the Customer ID field in the record after 30010. Type 30011. Fill in the rest of the data using you as the customer
- Close Customers, open Accounts, add an account for the new customer.
- Now we establish the relationships between the tables.
- Click Database Tools > Relationships. Box opens
- Double-click each table in the box. Close box. Resize table boxes if necessary.
- Drag BranchID field in Branch box to BranchID field in Accounts box. Click Enforce Referential Integrity and Cascade Update Related Fields. Click Create.
- Drag CustomerID field in Customerbox to CustomerID field in Accounts box. Click Enforce Referential Integrity and Cascade Update Related Fields. Click Create.
- Possible problems: types might be different. Values of e.g. Branch IDs in Branch table might not correspond to Branch ID values in Accounts table.
- Click Save.

Multiple-table Databases Demo 5

- Test Referential Integrity checking.
- Double click Accounts to open it in Datasheet view.
- Add a new record: 1022 30003 B60 4000 13/4/2013
- Warning message appears.
- Click OK. Check out Branch table. No branch B60 ☹
- Back to Accounts table. Change B60 to B50. Hit tab three times.
- Close Accounts table. Reopen it and check for new account.
- Close any open tables.
- Compact and Repair Database. Then File > Save&Publish > Back Up Database. Accept name and Save.

Queries

- Once you have a large amount of data stored in a database, you need a quick and easy way to access that information
- Queries allow you to access data that conforms to certain criteria
- Queries also allow you to update and modify data in tables

- Let's initially consider queries on single database tables …

Queries

- When a query is performed, it locates the records in an MS Access table satisfying the stated criteria
- The records selected in this way are stored in a **dynaset** - a dynamic subset of a table
- The dynaset exists temporarily to show the results of the query
- Changes made to the dynaset will be reflected in the orginal table

Types of Queries

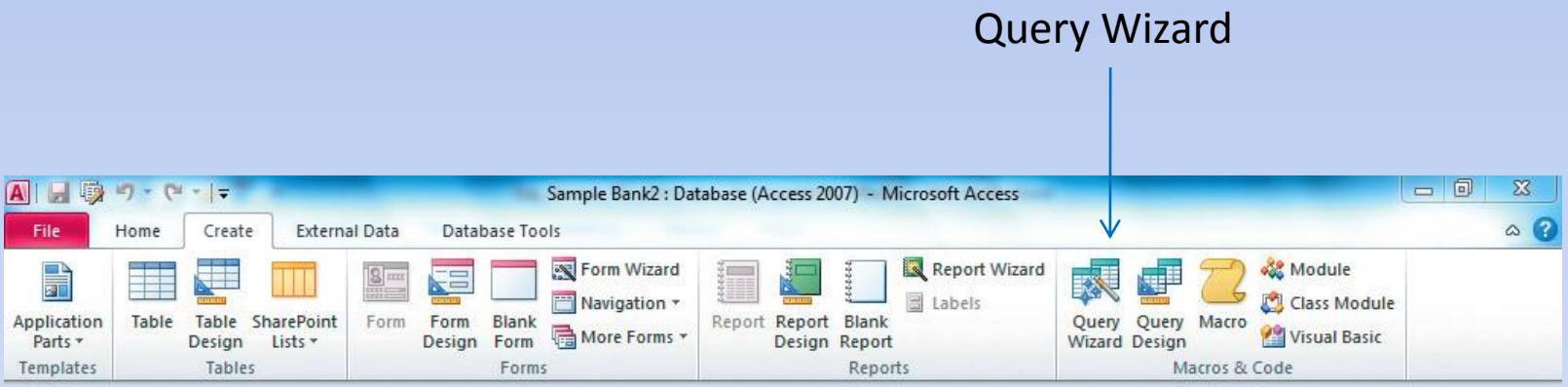
- **Select** query
  - Most basic form of query, returning records that satisfy specified criteria
- **Total** query
  - performs calculations
- **Action** query
  - Updates/modifies database
- **Specialised** queries
  - Parameter Queries
  - Crosstab Queries
  - Find Unmatched Records query
  - Find Duplicate Records query

Creating Queries

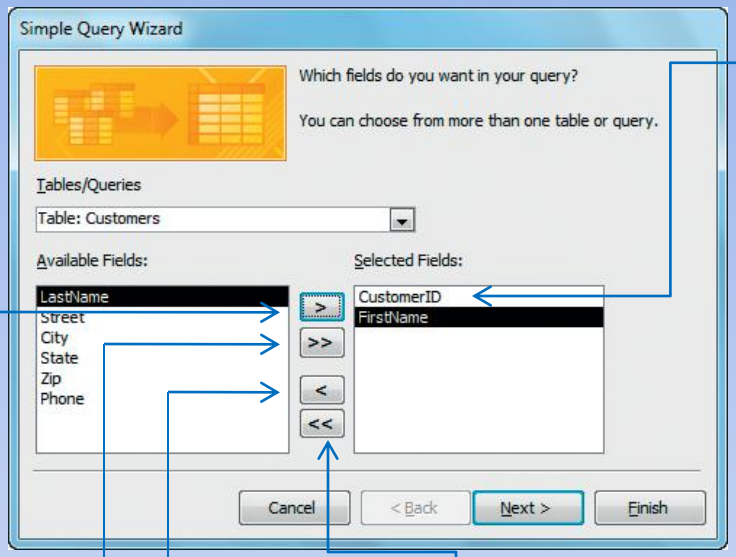
- MS Access provides a number of ways of designing queries
  - Query wizard
  - Design View
  - SQL (A lot of computer scientists pronounce this as “sequel”) - Structured Query Language

Using the Query Wizard

- Launch Query Wizard
- Modify Query in Design view



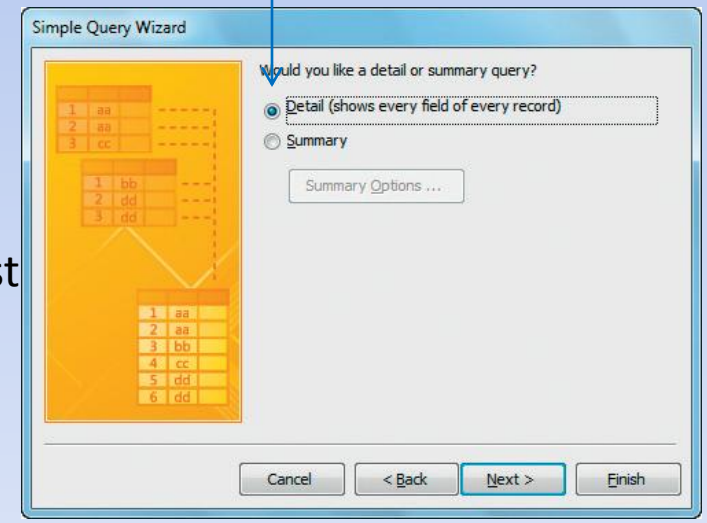
Query Wizard



Fields already moved to the Selected Fields list

Select a Table or Query

Select detail or summary data



Remove all fields from the Selected Fields list

Remove a single field from the Selected Fields list

Move all fields to the Selected Fields list

Move a single field to the Selected Fields list

Design View

- A more flexible way of designing queries is in Design View
- The following steps will guide you in designing a basic Select Query
  - Select tables containing fields that you would like to view or use in specifying criteria
  - Select fields in the order you wish to view them
  - Set any sort criteria
  - Select/De-select fields that will be displayed by the query
  - Set criteria

Datasheet View

- For tables, Datasheet view can be used to add, edit and delete records in the tables
- Changes made to the table in Datasheet view cannot be reversed
- For queries, Datasheet view can be used to view the results of performing a query
- Datasheet displays the dynaset
- Again, **changes** made in Datsheet view **will be reflected** in the original table

Running a Query

- Once designed a query can be performed by:
  - Clicking on the Run icon (!) in the menu bar or from the Query menu
  - Selecting Datasheet view
- Results of query will be displayed in Datasheet just as you would view the contents of a database table
- Results of query are temporarily stored in a dynaset

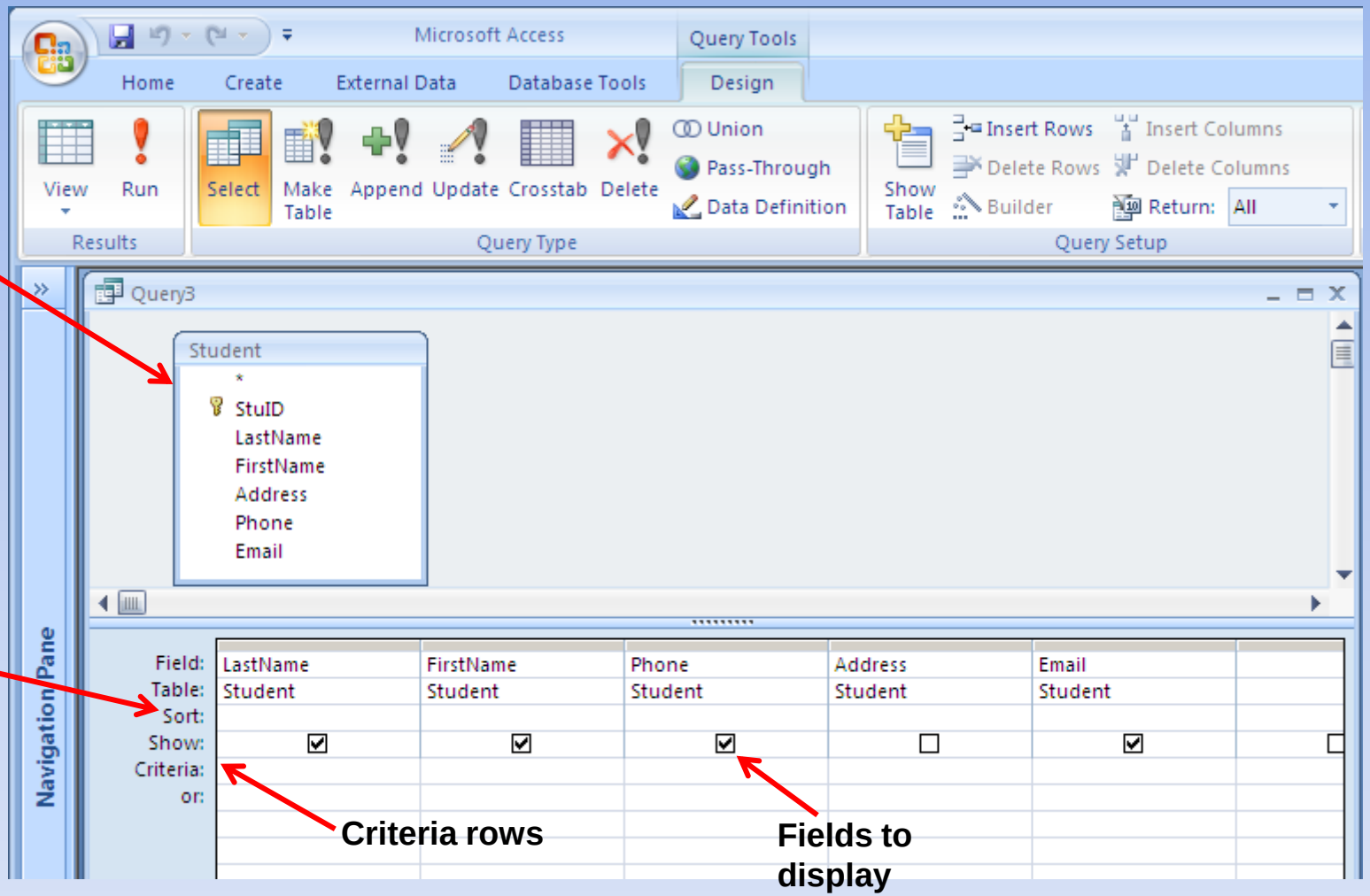
Select Query

- The simplest Select Query simply lists certain chosen fields from among those in the table
- No criteria are specified so all records are displayed but only the selected fields are shown for each record
- The steps involved:
  - Select tables
  - Select fields to display
  - Impose any sorting order

Select Query

Tables Involved in query

Sort order



Sorting Query Results

- Results of a query can be **sorted** on certain fields by specifying a sort order
  - Ascending or Descending
- Sort order is applied on fields from **left-to-right**

Specifying Selection Criteria

- Can reduce the number of records displayed by specifying criteria
- Each record whose fields match the specified criteria will be displayed (but note that only the selected fields are shown)
- Several criteria rows can be specified
  - In any given criteria row, a record must match **every** field with a criterion
  - If there are multiple criteria rows, the record must match **at least one** row to be displayed

Specifying Criteria for Different Data Types

- Delimiters

  text fields – enclose in quotes "Smith"

  date/time – enclose in # signs: #14/10/2012#

  numeric – no delimiters needed

- Examples of Criteria

  "Smith"

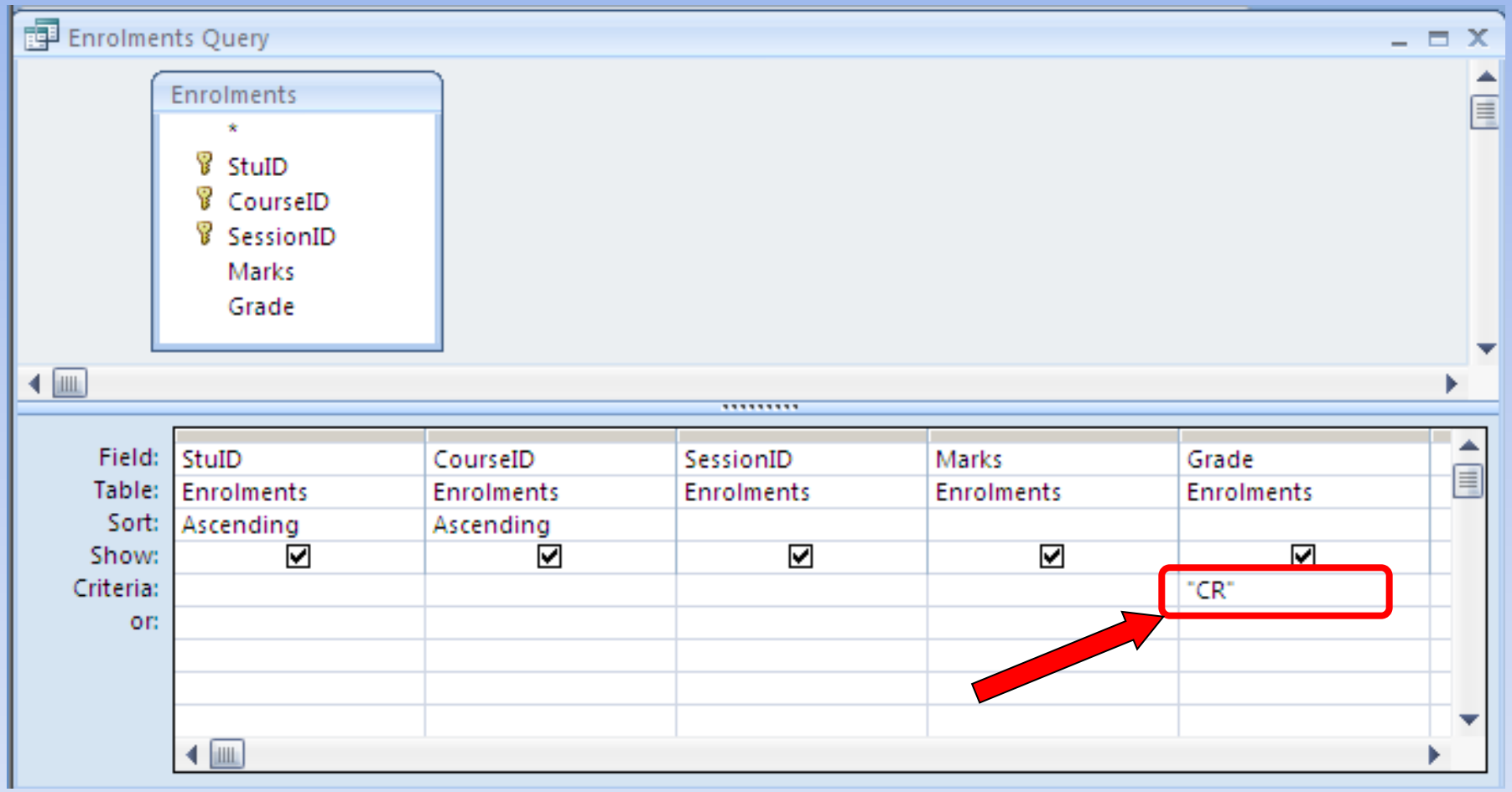
  >= #14/10/2012#

  <= 5000

Text Criteria

- Can specify text in double quotes:
  - "*search text*"
  - Double quotes only necessary if search text contains space, otherwise Access will put them in for you anyway.
- Again, only records that have the text in that field will match
- Text is not case sensitive
- Can use wildcards for matching more sophisticated patterns – see slide 71

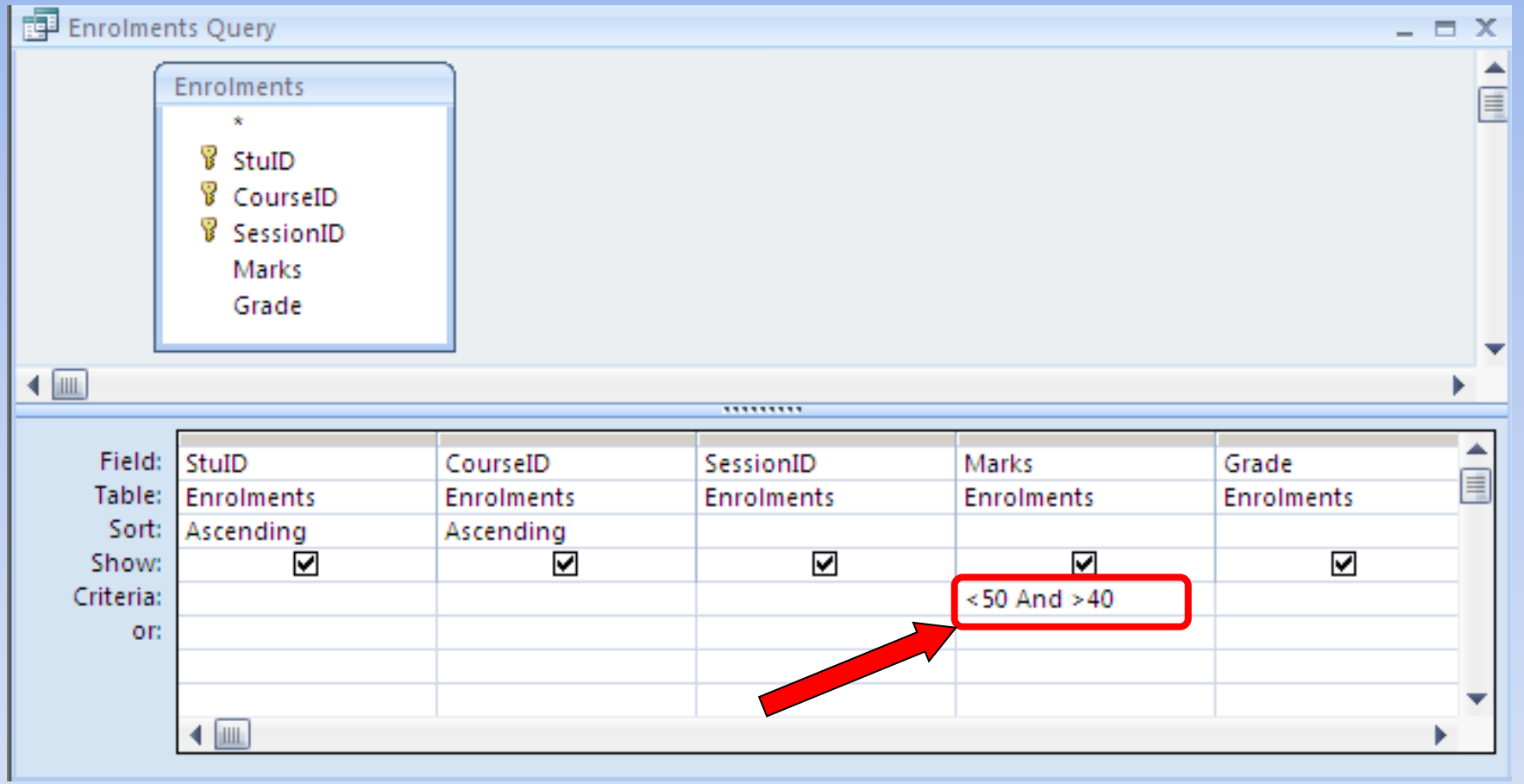
Text Criteria



Numeric Criteria

- For numeric fields:
  - Entering a single number in the criteria row of a field means that only records containing that value in the field will match
  - Can prefix number with logical criteria
    = ,<, <=, >, >=, <>
    - = can usually be omitted

Numeric Criteria

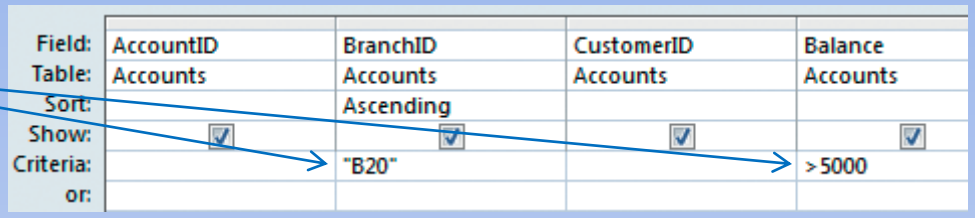


AND OR and Not

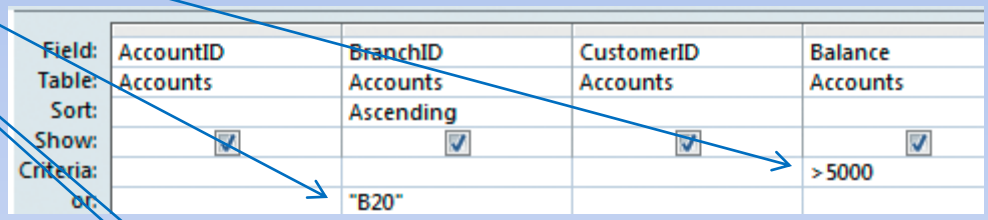
- In a given criteria row all fields must match criteria (AND)
- If multiple criteria are specified, record can match any of them (OR)
- Can also use logical operators to specify criteria in a single field
  - E.g., >=110 And <130
  - E.g., “HD” Or “DN”
  - E.g., Not “HD”

AND, OR, and NOT Criteria

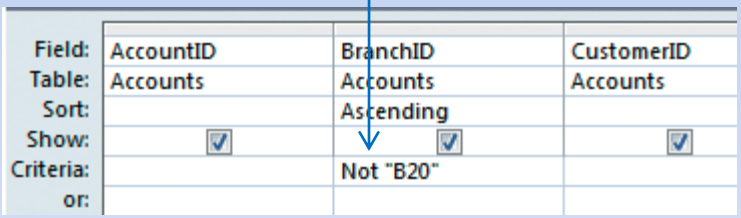
- And operator
- Or operator
- Not operator



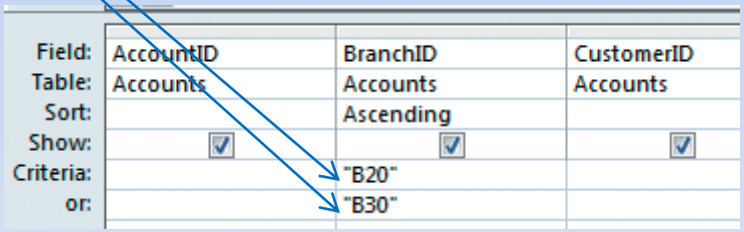
BranchID is B20 **and** Balance > 5000



BranchID is B20 **or** Balance > 5000



BranchID is not B20



BranchID is B20 or B30

BETWEEN, IN, LIKE

- **Between** - specifying ranges
  - Between 110 And 130
    - Same as >=110 And <= 130
- **In** - lists of values
  - In(“two”, “four”)
    - Same as “two” Or “four”
- **Like** - searching patterns using wildcards
  - Like “*z”
    - Strings that end in letter ‘z’

Wildcards

- Wildcards can be used to specify patterns in a text query criterion
  - * - matches 0 or more characters
  - ? - matches any single character
  - [] - match any single character specified inside the brackets
  - [!] - match any single character except those specified inside the brackets []
  - - match a range of characters (e.g., [a – z])
  - # - matches any single numeric character
  - To search for literal wildcard characters place them in brackets (e.g., [?] matches the character '?')

Wildcard Examples

- The question mark (?)
  - *"H?LL"* would match *Hall*, *Hill*, and *Hull*
- The asterisk (*)
  - *"S*nd"* would match *Sand*, *Stand*, and *StoryLand*
- The hash (#)
  - "????####" would match *COMP1000*, MATH1313 etc
- The brackets ([])
  - "[aeiou]*s" would match anything starting with a vowel and ending in s

Dates and Times

- Can specify dates and times by surrounding them with # signs
  - E.g.,
    - #26/4/05#
    - #April 26, 2005#
    - #26-Apr-2005#
    - #1:15PM#,
    - #13:15PM#
  - Date()
    - gets today's date
  - Between #1/1/2013# and #31/3/2013#

Null Criteria

- Criterion to select fields where no value has been entered
  - Null
  - Is Not Null
- Use key word **Null** in Criteria row
  - Take care not to quote the string “Null” otherwise you will be looking for the text string composed of those four letters

Expressions and Calculated Fields

- Assuming we have a field called HourlyRate and HoursWorked we can use them to create an expression such as
  - [HourlyRate] * [HoursWorked]
- We can use an expression as a criteria
  - [HourlyRate] * [HoursWorked] > 500
- We can use expressions to create computed fields
  - EmployeePay: [HourlyRate] * [HoursWorked]

Saving a Query Containing Calculated Fields

- When in Design View, saving queries saves the query design
- When in Datasheet View, saving updates underlying data
- Calculated fields automatically updated when components of expression are updated
- Use Zoom (Shift+F2) to view long expressions

Query with Calculated Fields Demo

- Open *a03h1property* and save a copy (*a03h1property_demo*)

Create and save a query

- Click Create tab > Queries > Query Design
- Select Agents & Properties tables, double-click FirstName & LastName fields from Agents, and DateListed, ListPrice, SqFeet, and Sold fields from Properties.
- Click Run, admire the 303 results.
- Click View. Type No in Criteria row of Sold field.
- Select Ascending from the Sort row of the ListPrice field.
- Click Run, 213 results now. Save query as TargetHouses.

Query with Calculated Fields Demo 2

Create calculated field

- Click View. Click File tab > Save Object As, type PricePerSqFt, click OK. This saves a copy of the query with the new name.
- Click File tab again. Find first blank column in query grid.
- Click in top row of 1st blank column. Type PricePerSqFt:ListPrice/SqFeet, press Enter. Widen the field. Click Run.
- Click View. Click in the PricePerSqFt field cell, then click Show/Hide group > Property Sheet.
- Click the Format property arrow, select Currency. Click in the Caption box and type Price Per Sq Ft. Close Property Sheet. Click Run.

Verify Results

- Launch Excel. Return to Access. Select first 10 rows of the query result, Copy.
- Return to Excel, click cell A1, paste the 10 rows. Enter =E2/D2 in cell H2 and press Enter. Copy the formula down the column using the fill handle.
- Results match column G. Exit Excel without saving.
- Return to Access. Click Save.

Query with Calculated Fields Demo 3

Recover from an Error in a Formula for a Calculated Field

- Click View. Find first blank field, type WrongPricePerSqFt: xListPrice/xSqFeet (deliberate error).
- Click Run. An Enter Parameter Value dialogue box appears. Enter 100000 into it, click OK. Another Parameter box comes up. Enter 1000, click OK.
- Check results for WrongPricePerSqFt. All results are 100.
- Return to Design View. Press Shift+F2 to view the formula conveniently. Edit out the extra *x*s.
- Run and Save the query.
- File > Compact & Repair Database
- File > Save & Publish > double click Back Up Database, check name and Save.

Expression Builder, Functions, and Date Arithmetic

- Expression builder:
  - Helps to create expressions
  - Provides access to built-in Access functions

    Functions work similarly to Excel
  - Allows you to perform date arithmetic

    Either directly with +/–, or using functions like DateDiff()
  - Avoids typos in field names

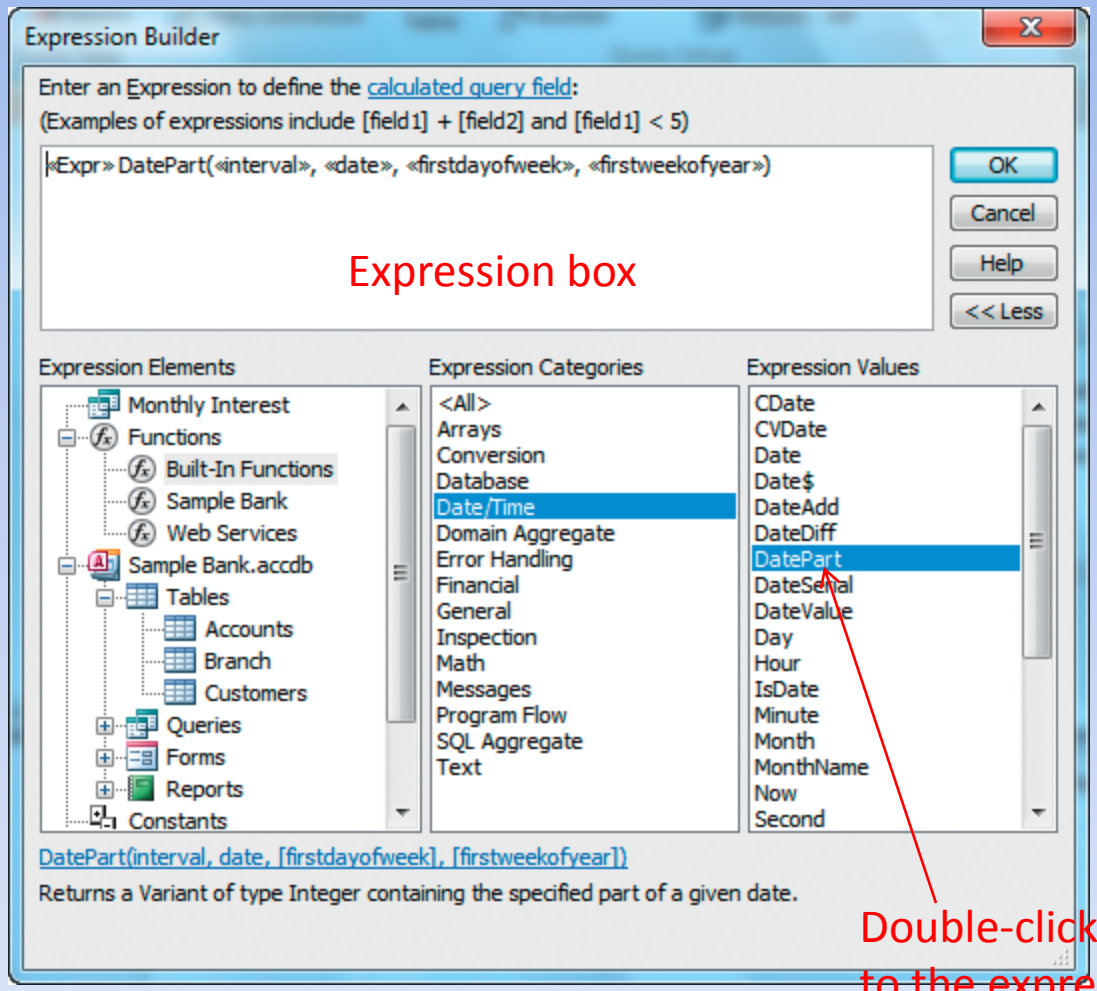
Creating Expressions with the Expression Builder

To launch:

- Open a query in Design view;
- Click Design Tab > Query Setup > Builder

Or:

- Right-click a cell in the field row, select Build…

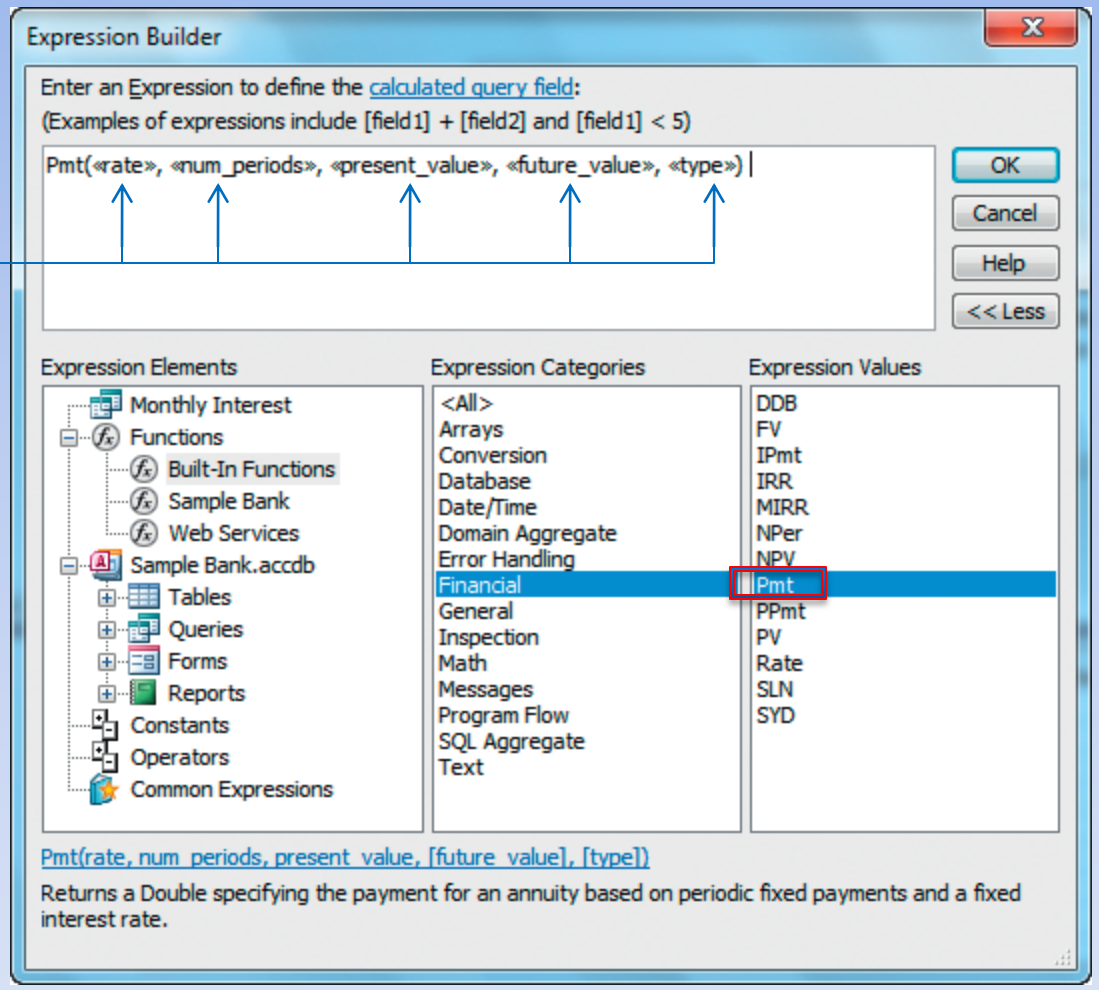


Pmt Example

- Pmt Function has five arguments, the last two optional as in Excel

- Google:
  "Access 2010 Pmt function" for more info.

Substitute the function you are interested in here…



Conditional Output with the IIF Function

- Function syntax
  - IIF (condition, then-part, else-part)
  - Examples
    - IIf([IsRenewal], [StandardRate]+[Bonus], [StandardRate])
    - IIf([State]="CA", "CA", "Out of State")
    - PropertyStatus:IIf(Date() – [DateListed] <= 30, "New Listing", "For Sale")
- Nested IIF statement
  - Example

```
IIf(Date() – [DateListed]<=30,
    "New Listing",
    IIf(Date() – [DateListed]>=180,
        "Stagnant",
        "For Sale"))
```

condition
then-part
else-part

condition
then-part
else-part

Date Arithmetic

- Date()

  returns the current date
- DatePart("yyyy", [Employees]![HireDate])

  returns the year when an employee was hired.
- DateDiff("yyyy", *date1*, *date2*)

  returns the number of full years between *date1* and *date2*

  Google "Access 2010 DateDiff function" for more.

  Access has built-in help, but google seems better at finding Access help, often at office.microsoft.com

  See e.g. the top hit from this Google search.

Functions on Dates and Times cont…

- DateDiff can find how many years “yyyy” or months “m” or days “d” between 2 dates.
  - DateDiff ("yyyy", #15/10/1998#, #22/11/2003#) would return 5
  - DateDiff ("m", #15/10/2003#, #22/11/2003#) would return 1
  - DateDiff ("d", #15/10/2003#, #22/11/2003#) would return 38

Builder Demo 1

Use Expression Builder to Modify a Field

- Click in PricePerSqFt field, click Query Setup > Builder.
- Change PricePerSqFt to PricePerBR.
- Double-click the [SqFeet] term, Delete.
- Under Expression Elements, click the + next to the *a03h2property_demo* database name, click Tables, click Properties, double-click Beds.
- Select [Properties]! in the expression, Delete, click OK, Run.
- Click View, click the PricePerBR field, click Property Sheet, change Format to Currency, type Price Per Bedroom in the Caption box. Run, inspect changes, click View, save query.

Builder Demo 2

Use Expression Builder to Add a Field

- Select the whole PricePerBR expression, right-click it, then select Copy. Right-click in next blank field, click Paste.
- Click the new field, click Query Setup > Builder.
- Change the expression to

  PricePerRoom: [ListPrice]/([Beds]+[Baths]+3)
- Click OK, switch to Design view, click PricePerRoom, Property Sheet, Format = Currency.
- Run, inspect, Save, close the query.

Builder Demo 3

Work with Date Arithmetic, Add Criteria

- Click in the top row of the first blank column.
- Open Query Setup / Builder, double-click Functions, click Built-in Functions, Financial, double-click Pmt. Set

  <<rate>> = 0.06/12 (i.e. 0.005)

  <<num_periods>> = 360

  <<present_value>> = –[ListPrice]*0.9
- Delete last two parameters. Click OK. Run the query, widen the column.
- Use Property Sheet for Payment, set Format=Currency.
- Run, Save query.

Builder Demo 4

Work with Date Arithmetic, Add Criteria

- Click View. Click top row of 1st blank column, Shift+F2
- Type
  - DaysOnMarket: #7/5/2014# - [DateListed]
- Type
  - <100000 in Criteria row of ListPrice.
- Run, Save.
- Click View. Click top row of 1st blank column, Shift+F2.
- Type
  - MonthsOnMarket: (#7/5/2014#-[DateListed])/30, or
  - MonthsOnMarket: DateDiff("m",[DateListed],Date())
- Click OK. Run, Save.

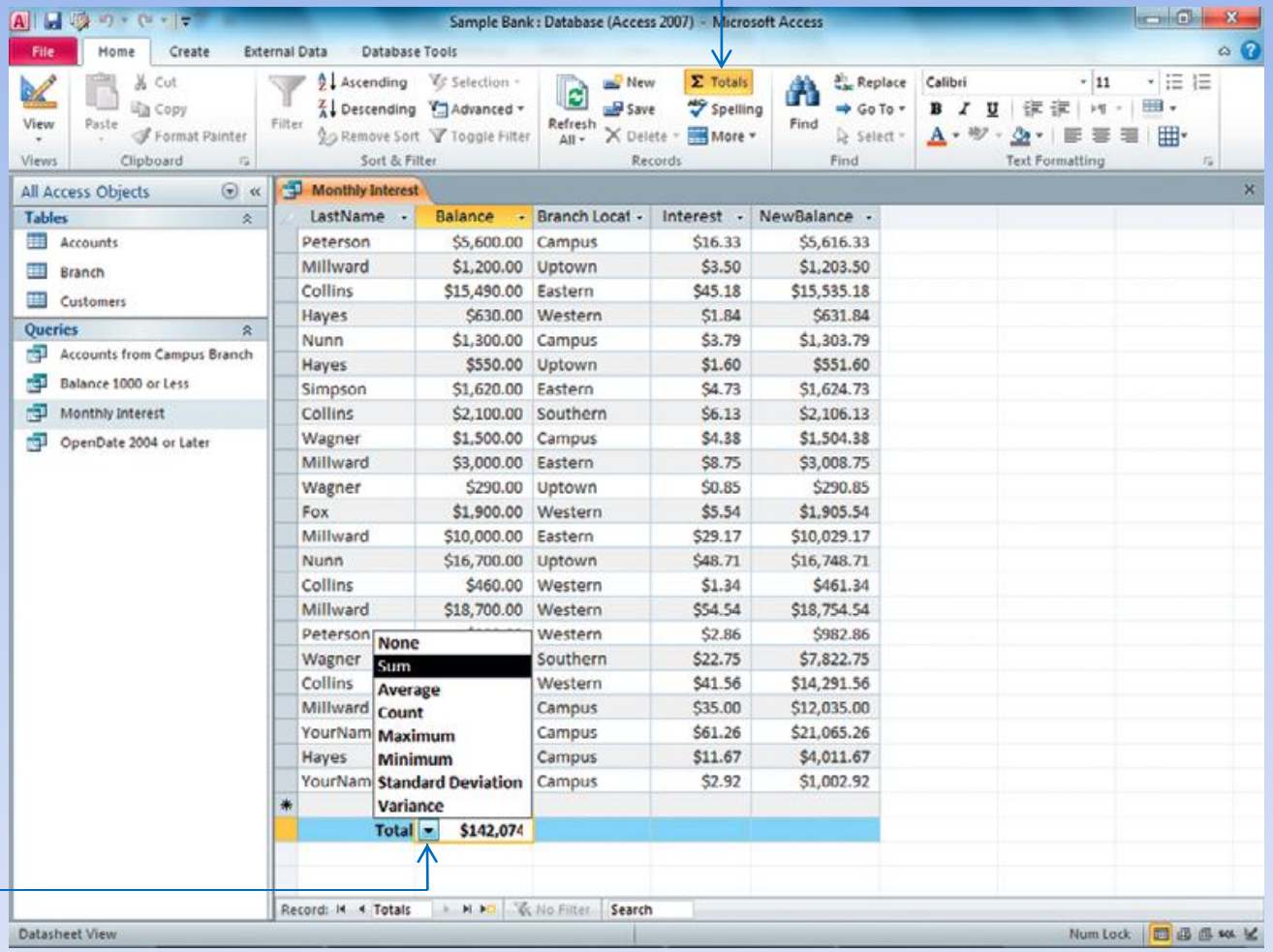
Aggregate Functions

- Perform calculations on an entire column of data and return a single value
- Examples:
  - Average
  - Count
  - Maximum
  - Minimum
  - Standard Deviation
  - Sum
  - Variance

Adding Aggregate Functions to *Datasheets*

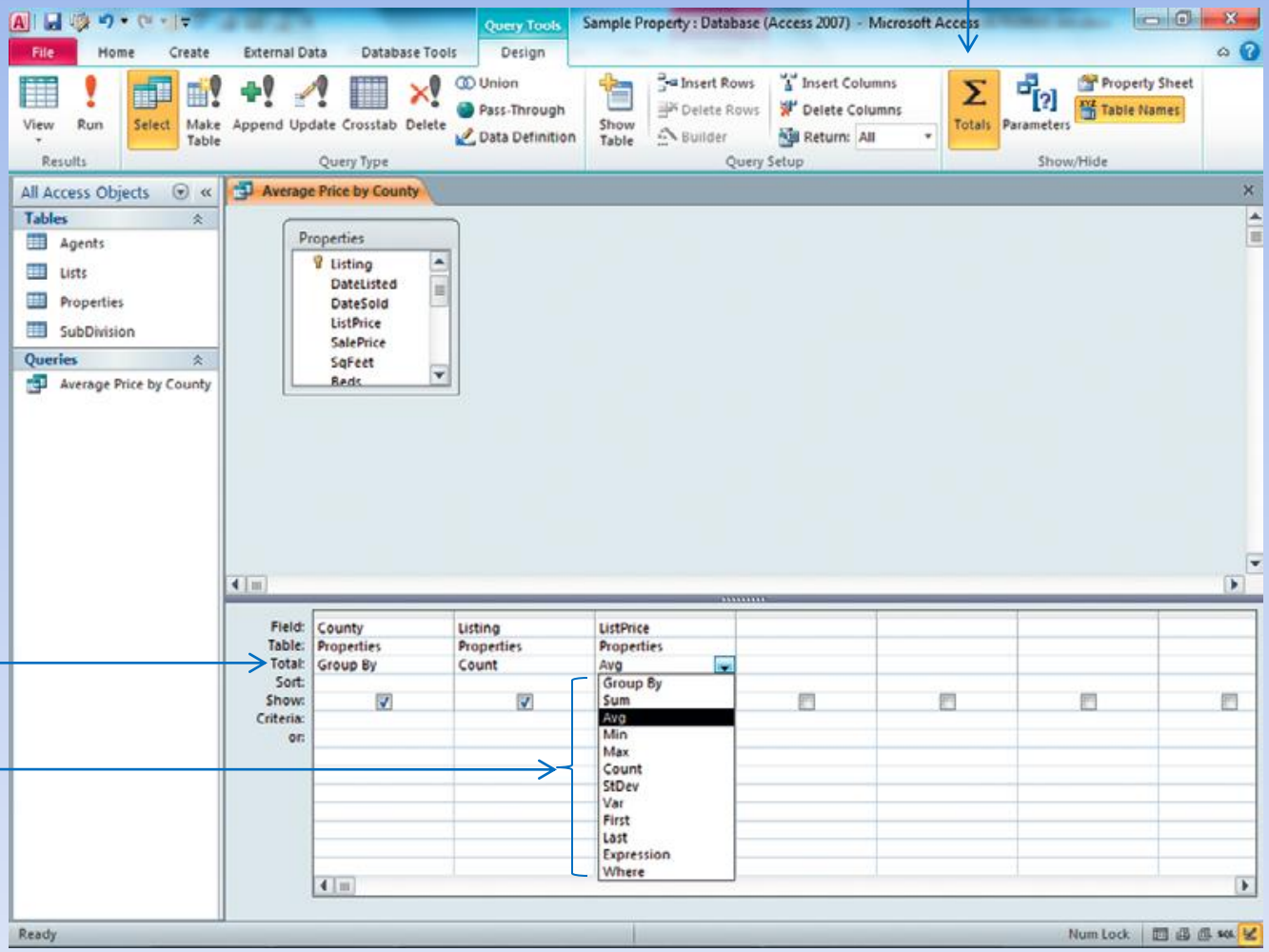
Click Totals to add the totals row

Click the Total row arrow to show the choice of aggregate functions



Adding Aggregate Functions to *Queries*

Click Totals to add query totals



Total row

Aggregate Functions

Aggregate Functions Demo 1

- Open *a03h2property_demo*, save as *a03h3property_demo*

Add Total Row to Datasheet View

- Right-click MortgagePayments query, select Design view, drag Listing field from Properties table to fifth column.
- Click View, click Home > Records > Totals (Σ), scroll to end of table.
- Click Total row in ListPrice column, click arrow, select Sum
- Click Total row in Listing column, click arrow, select Count
- Click Total row in Price Per Sq Ft column, click arrow, select Average
- Save and close the query.

Aggregate Functions Demo 2

Create a Totals Query based on a Select Query

- Click Create tab, click Queries > Query Design.
- Add Properties table and Lists table, close Show Table dialogue box.
- Add NameOfList field from Lists table, SalePrice and Sold fields from Properties table to the design grid.
- Click Totals in the Show/Hide group.
- Change the Total row to Avg in the SalePrice column.
- Change Total row to Where in Sold column, type Yes in Criteria row.
- Click in SalePrice field, then click Property Sheet in the Show/Hide group. Change the SalePrice format to Currency. Close Property sheet, run query.
- Click View. Add Listing field from Properties table to fourth column. Change Total row for Listing to Count. Run the query.
- Click View. Change caption of Listing column to "Number Sold", run query, widen. Save query as ListResults.

Aggregate Functions Demo 3

Add a Calculated Field to a Totals Query

- Click File tab, click Save Object As, save the query as ListResultsRevised.
- Click File tab. Click Totals in the Records group, then add Sum to the Number Sold column.
- Click Views Group > View. In the first blank column, type DaysOnMarket: [DateListed]-#12/1/2011#. Change Total row to Avg.
- Right-click DaysOnMarket field, Properties > Format > Fixed, close.
- Run query. Save and close query.

Exercise

- Using the the Student Database and the enrollments table
  - **Write a query that displays average marks for each course offering (in the Enrolment table). In other words, if a course is offered in more than one session, you need to calculate average marks for each session separately.**
  - **Write a query that displays min, max and average marks for every student. You should display records in the descending order of the average marks.**

Filter by Form vs Queries

What is the difference between a filter and a query?

- Filter by Selection is less powerful than Filter by Form, so let's compare Queries with Filter by Form.
- Filter by Form does the same sort of thing as queries – it allows selection of records satisfying multiple criteria constructed using **and** and **or**.
- However, Filter by Form does not allow you to save the selection specification for reuse, and it is restricted to a single table.
- This can be important with a complicated query that you want to run regularly – say each month.
- As we shall see later, the *selection* queries that we've been studying can be converted into *action* "queries", which can update or delete records.

Multi-Table Queries

- Uses data from more than one table
- It makes sense to use *related* tables
- Similar to creating a single-table query
  - However, changes to results in datasheet view do not change the underlying tables (unlike single-table queries)
- Tables need to be linked properly to ensure the query works properly

Summary

- This section introduced the concept of tables and query design.
- Tables and forms are used to input data, and to create queries and reports to extract information from the database in an organized and useful way.
- The information to be extracted, though, is dependent on the quality of the underlying tables.
- you learned how to create a calculated field in a query, and create expressions using the Expression Builder.
- You learned how to use built-in functions in Access, perform date arithmetic.
- You learned how to use aggregate functions in datasheets and queries.