Intro to Relational Databases

CS 234 N
Chapters 17++

Topics

- Review of relational database concepts
- Introduction to MS SQL Server
- Introduction to SQL
- Database application architecture in a .NET environment

Before We Begin

- Chapter 17 starts with a brief introduction to relational database concepts.
  - If you've taken 125D, lots of this will be review for you. If you've taken 275 or you have database experience, all of it will be review. In either case, skim the first part of the chapter.
  - If you don't have database experience – read the first sections carefully, plan on asking for help when writing SQL statements and buy yourself a good database book.
- I'll assume that folks know what I mean when I use database vocabulary.

Relational Databases

- Store data in a set of 2-dimensional tables
- Each table
  - Describes an entity or a thing - customer or a book or an order
  - Consists of a set of characteristics or attributes or fields related to that thing – customerid, name, address, city might be fields in the customer table
  - Contains one field that uniquely identifies each row or record in the table. That field is called the primary key.

Relational Databases

- A table is related to other tables
  - A customer places at least one but maybe lots of orders.
  - You physically create the relationship between two tables by putting the primary key from one table into another table.
    - In this example, the customer id would be included in the order to table. Why wouldn't you put the orderid in the customer table to create the relationship?
    - A pkey used in another table to create a relationship is called a foreign key.
  - The relationship between tables has a "cardinality".
    - One to many is most typical. There's a 1 to m relationship between customer and order.
    - One to one are sometimes used but often indicates that the 2 tables can be combined into 1 table.
    - Many to many relationships often indicate the need for a 3rd table between the 2 tables. A book is purchased on 1 to many orders and an order can contain 1 or more books. The relationship between book and order is many to many. Creating an order items table that lists the books in a specific order resolves the many to many relationship.

Relational Databases

- Lots of books describe formal rules for creating database structure. I'll use a couple rules of thumb to keep instead …
  - Avoid Redundant data
    - Wastes storage space
    - Creates possibility for inconsistency
    - Requires extra work when updating
  - Avoid "repeating groups" of data
    - Order table that contained fields like book1, book2, book3 would be an example. Why is that problematic?
  - Make sure field values are atomic. Why?
  - Use natural fields for the pkey. Why?
  - Avoid empty attributes.
    - Book table with a large text field for the book review that might be blank for lots of books would be example. Why is that problematic?

DB Vocabulary

- Do you know what each of these mean?
  - Relational database
  - File server database vs client server database
  - Table
  - Field
  - Primary key
  - Foreign key
  - Entity integrity
  - Referential integrity
  - 1 to 1 Relationship, 1 to many relationship, 1 to many relationship
  - Normalization

SQL

- SQL (Structured Query Language) is a special purpose programming language (as opposed to a general purpose language like C#) specifically designed for managing the structure of and data stored in a relational database.

SQL

- Simple data manipulation SQL statements allow you to do (CRUD) operations on data. Chapter 17 talked about (and you should know)
  - Select
  - Insert
  - Update
  - Delete

SQL

- SQL can also be used to create and/or databases and/or database objects. You should be able to read that kind of SQL statement at this point but not necessarily write it. CS 275 will give you plenty of practice with that! ☺

Database Products

- Are often distinguished by the size and scope of the database that they support.
  - Access is designed for use with a small group of concurrent users on a local area network.
    - MS introduced Access as the desktop database application "for the masses". It's easy to create both a database and a database application without knowing anything about a database.
    - Most often both the database data and the database application are stored in ONE file. Even if the developer knows how to separate the data from the application objects, all of the data in the database is stored in one file and that file is managed by the network file system as well as the "database engine". Because of that architecture, all kinds of "wacko" things happen with an Access database when you try to "scale" it.

Database Products

- MS SQL Server (and Oracle) were designed for large scale data sets and applications.
  - A server class application, the "database engine", is responsible for managing the database system which generally contains more than one user database and several system databases.
  - The data for a database as well as other database objects like indexes is stored in a complex set of files that are managed by the database engine. The files are also managed by a server class operating system.
  - Applications that use the data are completely separated from the database engine.
  - Administration of the database system is often the job of a highly skilled database professional, the dba.

MS SQL Server

- We're going to use MS SQL Server 2014 as the database server in this class.
  - The Express edition is free. The "developer" edition, localdb, can be distributed with a database application for free.
  - It integrates easily with Visual Studio, C# and ASP.NET.
  - It's "easy" to use.
  - It's used by lots of local businesses, along with Oracle and MySQL, as the database engine for large scale application.

Use SQL to Create DB

- I've given you a SQL script to create the MMABooks in the starting files on moodle.
- Let's look at the code and see what we can figure out from reading it.
- Start the SQL Server service
- Let's use it and the MMC to create the database
  - Open the MMC. It should connect you to the master database which is a system database.
  - Open the file from the starting files.
  - Click the Execute! Button on the toolbar
  - Refresh the database tree. You should now see MMABooks AND all of the database objects in the database

SQL

- Simple data manipulation SQL statements allow you to do (CRUD) operations on data. Chapter 17 talked about (and you should know)
  - Select
  - Insert
  - Update
  - Delete

Simple Select

- Select *fieldlist* from *table*
  - Select * from album
  - Select artist_name from artist
  - Select artist_name, artist_id from artist

Where Clause

- Select *fieldlist* from *table* where *condition*
  - Select * from artist where artist_name = 'New Order'
  - Select * from artist where artist_id = 4
  - Select * from album where album_id != 2
  - Select * from album where album_name like 'Retro%'
  - Selete * from album where album_name > 'C' and album_name < 'M'

Order By Clause

- Select *fieldlist* from *table* order by *fieldlist*
  - Select * from artist order by artist_name
  - Select * from artist order by artist_name desc
  - Select * from track order by time, track_name

Lab 4

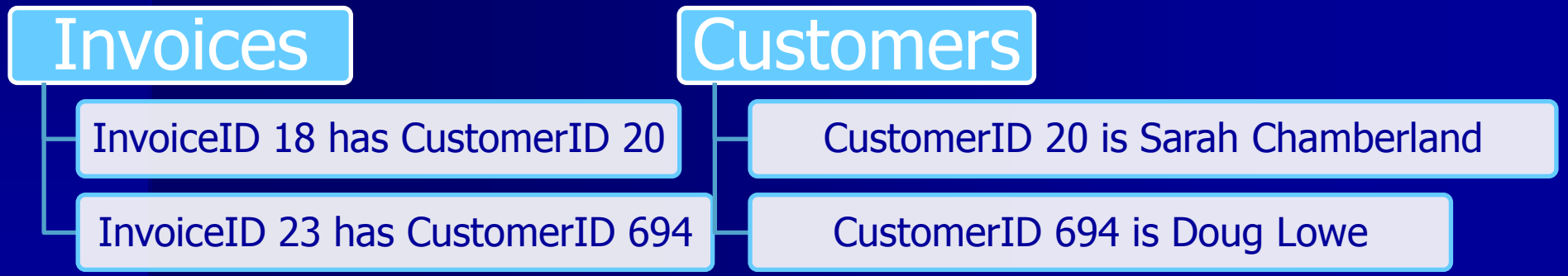
- The first part of lab 4 asks you to use the MMC and the MMABooks database to practice writing SQL Select statements.
  - You should be able to do 1 – 6 at this point

Joining tables

- Most of the time, the data that you need from the database is stored in a set of related tables. The sql statement that gets the data specifies how to "join" the tables together.
  - Most often you'll do an "inner" join (as opposed to an outer join). In an inner join (that involves 2 tables), the result set will include one copy of each record in the first table that has a matching record in the second table.

Joining tables

- Joining the Invoices table with Customers table in the MMABooks table yields 41 records because there are 41 invoices and each has ONE related customer record. Notice that there are 696 Customers but only the matching 41 appear in the result set.



Joining tables

- The general syntax for doing a join is

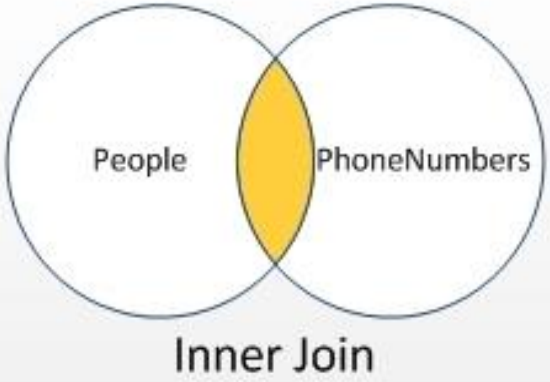
Select *fieldlist*
from *table1* inner join *table2*
On *table1.keyField = table2.keyField* …

Joining tables

- The previous example

```
Select *
    from Invoices inner join Customers
    On Invoices.CustomerID = Customers.CustomerID
    where ....
    order by …
```

All Kinds of Joins



Inner Join

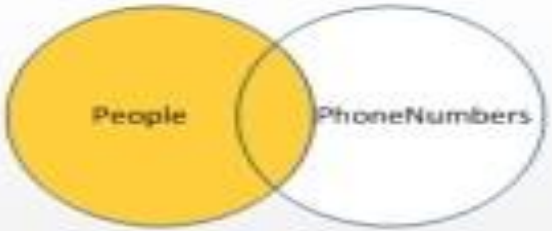
```
SELECT p.PeopleID, p.Name
      ,n.PhoneNumberID
      ,n.PeopleID, n.Number
  FROM dbo.People p
 INNER JOIN dbo.PhoneNumbers n
    ON p.PeopleID = n.PeopleID;
```

Select from two tables

```
SELECT PeopleID, Name
  FROM dbo.People;

SELECT PhoneNumberID
      ,PeopleID
      ,Number
  FROM dbo.PhoneNumbers;
```

All Kinds of Joins

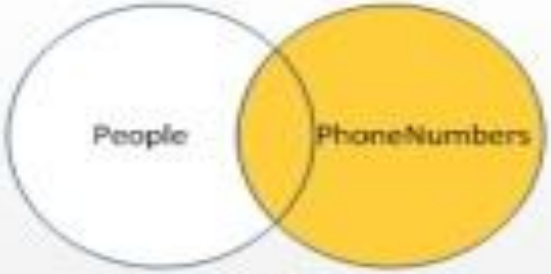


Left Outer Join

```
SELECT p.PeopleID, p.Name
     ,n.PhoneNumberID
     ,n.PeopleID, n.Number
  FROM dbo.People p
  LEFT JOIN dbo.PhoneNumbers n
    ON p.PeopleID = n.PeopleID;
```

PeopleID	Name	PhoneNumberID	PeopleID	Number
1	Steve	1	1	360 111-1234
2	Marcia	2	2	360 222-2222
3	Alex	3	3	360 333-4321
4	George	4	4	206 444-5432
5	Pete	5	5	206 555-9876
6	Ann	6	6	206 666-5634
7	Jack	NULL	NULL	NULL
8	Donna	NULL	NULL	NULL
9	Karen	NULL	NULL	NULL

All Kinds of Joins



Right Outer Join

```
SELECT p.PeopleID, p.Name
     ,n.PhoneNumberID
     ,n.PeopleID, n.Number
  FROM dbo.People p
 RIGHT JOIN dbo.PhoneNumbers n
    ON p.PeopleID = n.PeopleID;
```

PeopleID	Name	PhoneNumberID	PeopleID	Number
1	Steve	1	1	360 111-1234
2	Marcia	2	2	360 222-2222
3	Alex	3	3	360 333-4321
4	George	4	4	206 444-5432
5	Pete	5	5	206 555-9876
6	Ann	6	6	206 666-5634
NULL	NULL	7	NULL	206 777-8888

Lab 4

- The first part of lab 4 asks you to use the MMC and the MMABooks database to practice writing SQL Select statements.
  - You should be able to do 7 – 12 at this point

Group By clause

- When you want to summarize the data in a result set by adding, counting, averaging, finding the min, finding the max, and a couple of statistical functions, you use a group by clause.

Group By clause

- The general syntax adds the aggregate function call to the field list and the group by clause after the where clause

  Select *fieldToGroupOn,*

  aggregate(field), aggregate(field) …

  from … where …

  group by *fieldToGroupOn*

Group By clause

Select *count(*)* from Customers

```
select Invoices.InvoiceID,
    sum(InvoiceLineItems.Quantity) as QtyOfItems
    From Invoices inner join InvoiceLineItems
    On Invoices.InvoiceID = InvoiceLineItems.InvoiceID
    Group by Invoices.InvoiceID
    Order By Invoices.InvoiceID
```

Lab 4

- The first part of lab 4 asks you to use the MMC and the MMABooks database to practice writing SQL statements.
  - You should be able to do 13 – 16 at this point but you'll need some help.

Insert

- Insert into *tablename* values (*valuelist*);
  - Insert into artist values (7, ‘Barry Adamson’);
- Insert into *tablename* (fieldlist) values (*valuelist*);
  - Insert into album (artist_id, album_id, album_name);
  - Insert into played (artist_id, album_id, track_id) values (7, 1, 2), (7, 1, 3), (7, 1, 4)

Insert

- Insert into *tablename* (fieldlist) *select statement that yields a valuelist;*
  - insert into played (track_id, artist_id, album_id) select track_id, artist_id, album_id from track where artist_id = 2;

Delete

- Delete from *tablename* where *condition*
  - Delete from artist where artist_id = 3

Update

- Update *tablename* set *field = value*, *field2 = value* where *condition*
  - Update artist set artist_name = upper(artist_name)

Lab 4

- There are no practice problems for insert, update or delete at this point. You'll get a chance to practice with those a little in labs 5 and 6.
  - Submit the script for all of the sql statements you just wrote in a text file. There is no peer evaluation for lab 4.

Lab 4

- The second part of lab 4 asks you to participate in the design of the CourseEvaluation database system.
  - Let me describe what I want to be able to do with the database and the application(s) that use the database.
  - In groups, design the database.
    - Identify entities
    - Identify attributes for each entity
    - Identify relationships between entities
    - Identify primary keys, foreign keys.
  - We'll review your designs when you're done.

Lab 4

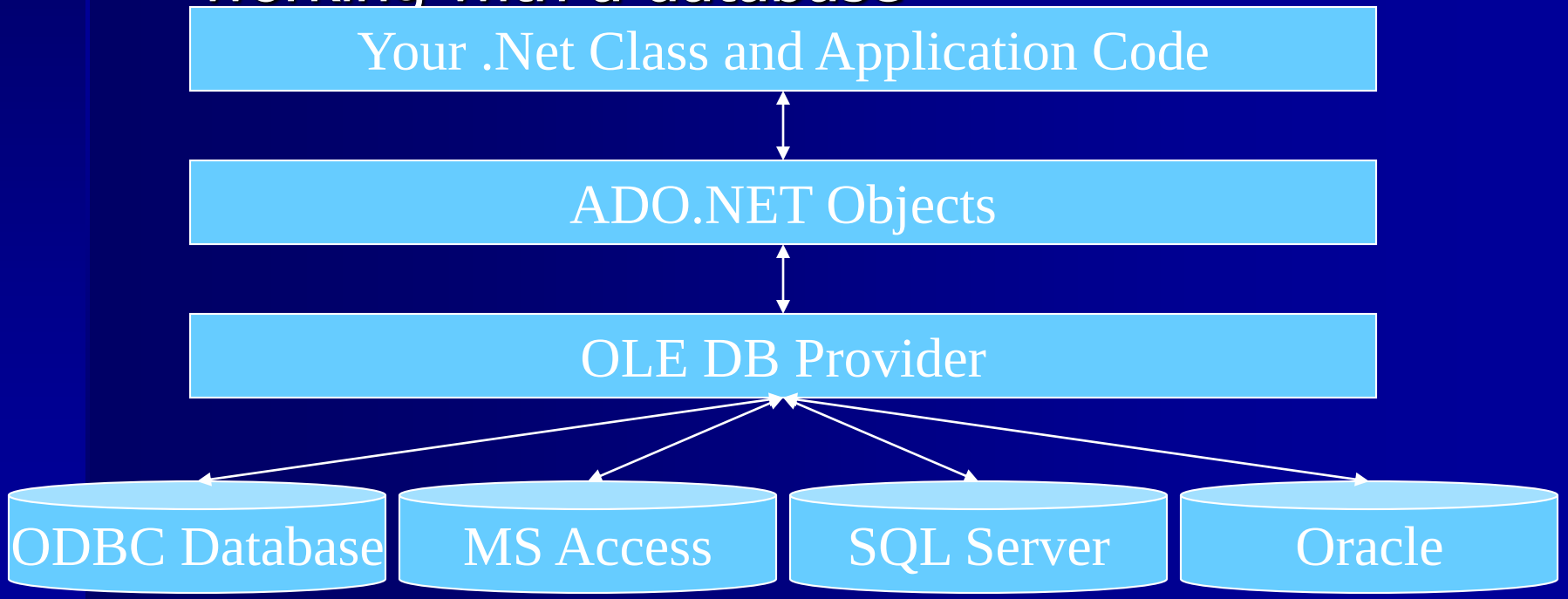
- The third part of lab 4 asks you to use the MMC GUI to create the CourseEvaluation database system.
  - Let me create one table with you and in the process, let's decide on naming conventions etc.
  - Let's divide the rest of the tables up among groups. Each group will create a table and add data to it. I'll show you how to script the result. You'll post your script on the forum for the lab and I'll integrate all of them, including adding foreign key contraints as necessary.

ADO.Net Architecture

- ADO.Net is the name used to refer to the set of technologies and classes that are part of .Net and allow you to interact with all kinds of data sources – particularly relational databases.
  - These classes are not the same as the ADO classes that were popularized in earlier versions of VBA and VB.
  - In particular, they're optimized to work well in n-tier application components that have to be "scalable" for the web and the cloud
    - Data processing has traditionally relied primarily on a connection-based, two-tier model.
    - As data processing increasingly uses multi-tier architectures, programmers are switching to a disconnected approach to provide better scalability for their applications. (Ask me about the potato chip analogy!)

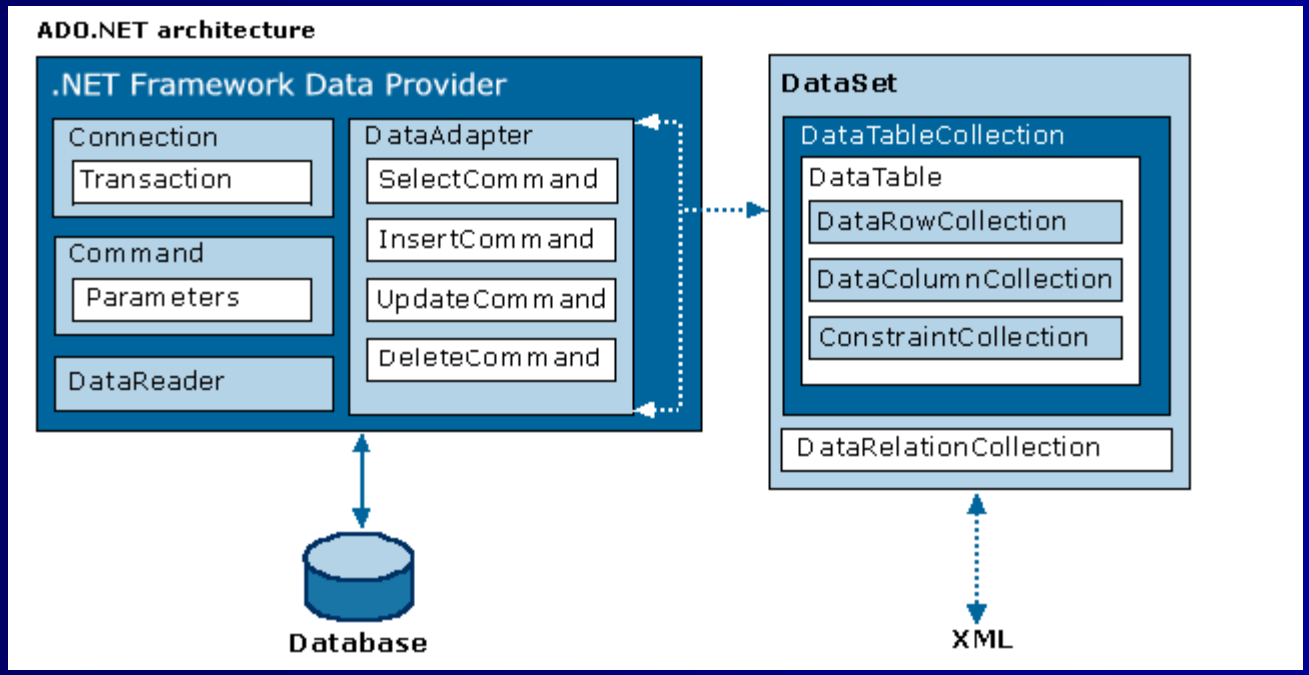
Intro to ADO.NET

- This figure illustrates the relationship between the .NET code you'll be writing and the other software components involved in working with a database



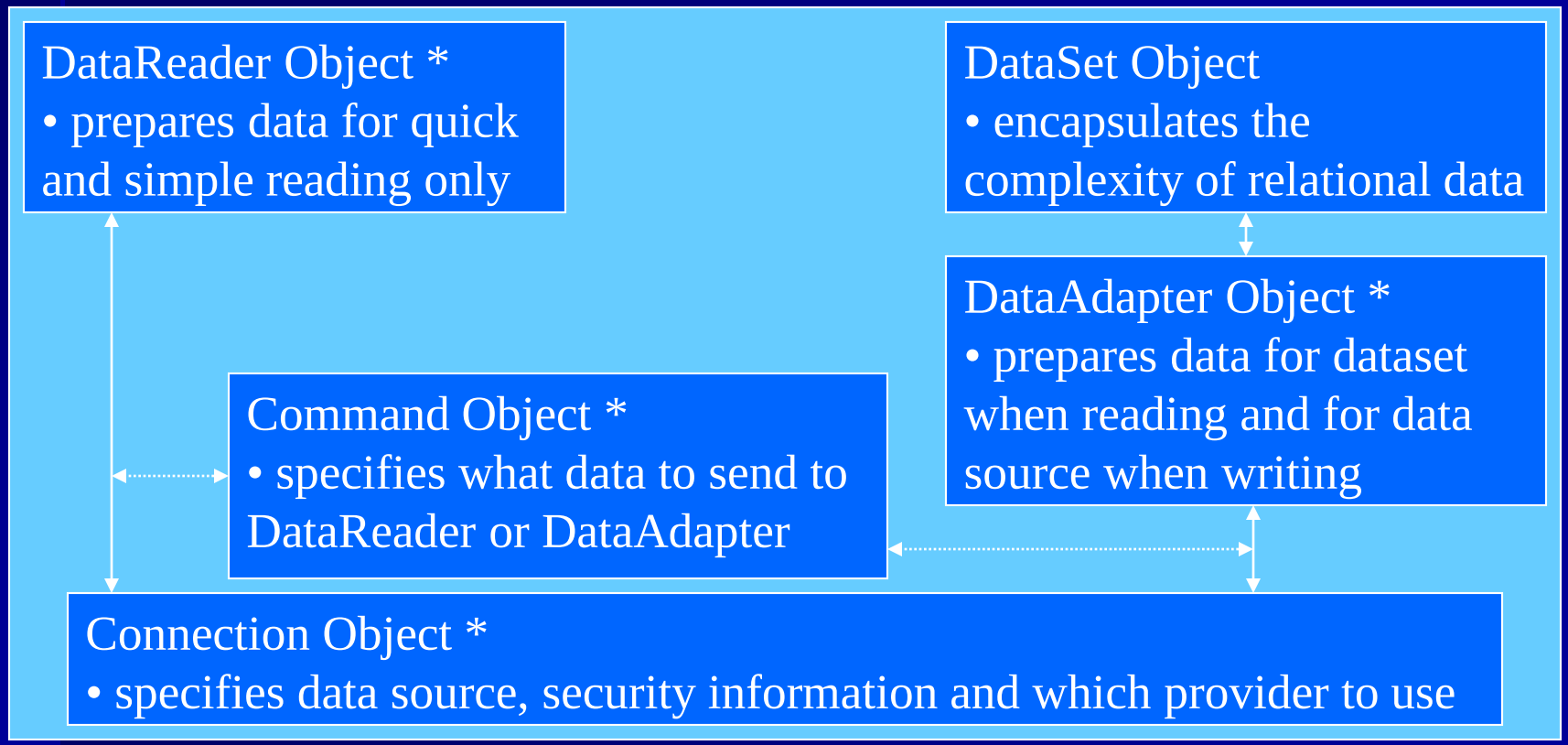
Intro to ADO.NET

- One illustration of the classes involved



Intro to ADO.NET

- And another …



Intro to ADO.NET

- The * on the previous slide mark classes that come in different "flavors" optimized for working with different databases
  - SqlDataReader, SqlCommand, SqlDataAdapter and SqlConnection are optimized for use with SQL Server. SqlClient namespace.
  - OleDbDataReader, OleDbCommand, OleDbDataAdapter and OleDbConnection can be used with SQL Server, but are recommended for all other data sources including Access. OleDb namespace.
  - OracleDataReader… to work with Oracle. OracleClient namespace.
  - ODBCDataReader… to work with other databases that are ODBC compliant but not OLEDB compliant. ODBC namespace. Requires an ODBC dsn.

Intro to ADO.NET

- The data provider classes were explicitly designed for data manipulation and fast, forward-only, read-only access to data.
  - The Connection object provides connectivity to a data source.
  - The Command object enables access to database commands to return data, modify data, run stored procedures, and send or retrieve parameter information. This class also provides the functionality necessary to perform transactions.
  - The DataReader provides a high-performance, read only, forward access only, stream of data from the data source.
  - The DataAdapter provides the bridge between the DataSet object and the data source. The DataAdapter uses Command objects to execute SQL commands at the data source to both load the DataSet with data, and reconcile changes made to the data in the DataSet back to the data source.

Intro to ADO.NET

- DataSet class is an in-memory representation of complex data
  - It contains a collection of one or more DataTable objects.
    - Each DataTable is made up of rows and columns of data, as well as primary key, foreign key, constraint, and relation information about the data.
  - It is explicitly designed for data access independent of any data source… even xml because it's a tree like structure rather than a 2 dimensional recordset like structure.
  - Because it is so versatile, it's also complex to use and very resource intensive! You'll want to use it when that kind of complexity is necessary and avoid it when you can do the work a simpler way. The alternative is to
    - Use a datareader and build a datatable yourself when retrieving data
    - Use a command object and stored procedures with parameters when editing data

Next Time

- Chapter 20 – Creating DB classes using ADO.NET
  - Reading Quiz 6
  - Lab 5
- Lab 6 (40 points) – DB Classes for Course Evaluation
- Quiz 2 - Chapters 17 and 20 and labs 4 – 6