

# ADO.NET

## **Accessing Databases**

# ActiveX Data Objects

- ADO.NET has a number of classes that :
  - Retrieve Data
  - Manipulate Data
  - Update Data

# ADO vs. ADO.NET

- ADO works great, but:
  - Requires COM and Windows
  - Recordsets don't travel well over the Internet
  - Connected behavior is hard to work with
  - Requires more code
- ADO.NET solves these problems
  - Uses XML under the covers for all data transport
  - No special code needed

# Disconnected?

- ADO.NET offers the capability of working with databases in a disconnected manner.
- An entire database table can be retrieved to a local computer/temp file if it is a network database.
- A connection could also be constant

# Web-Centric Applications

- Download the data and process it at a local level.
- If changes are made, the connection can be remade and the changes posted.
- The database could be LAN or Internet based.

# Data Providers

- MS SQL Server
- Oracle
- OLE DB (old SQL & Access- Jet 4.0)
- Open Database Connectivity (ODBC)

# 4 Core Classes of ADO.NET

- Connection - Connect to database
- Command - SQL statement to retrieve data
- DataReader - Sequential access to the data source
- DataAdapter - Populate a dataset & Update the database

# DataSet vs. DataReader

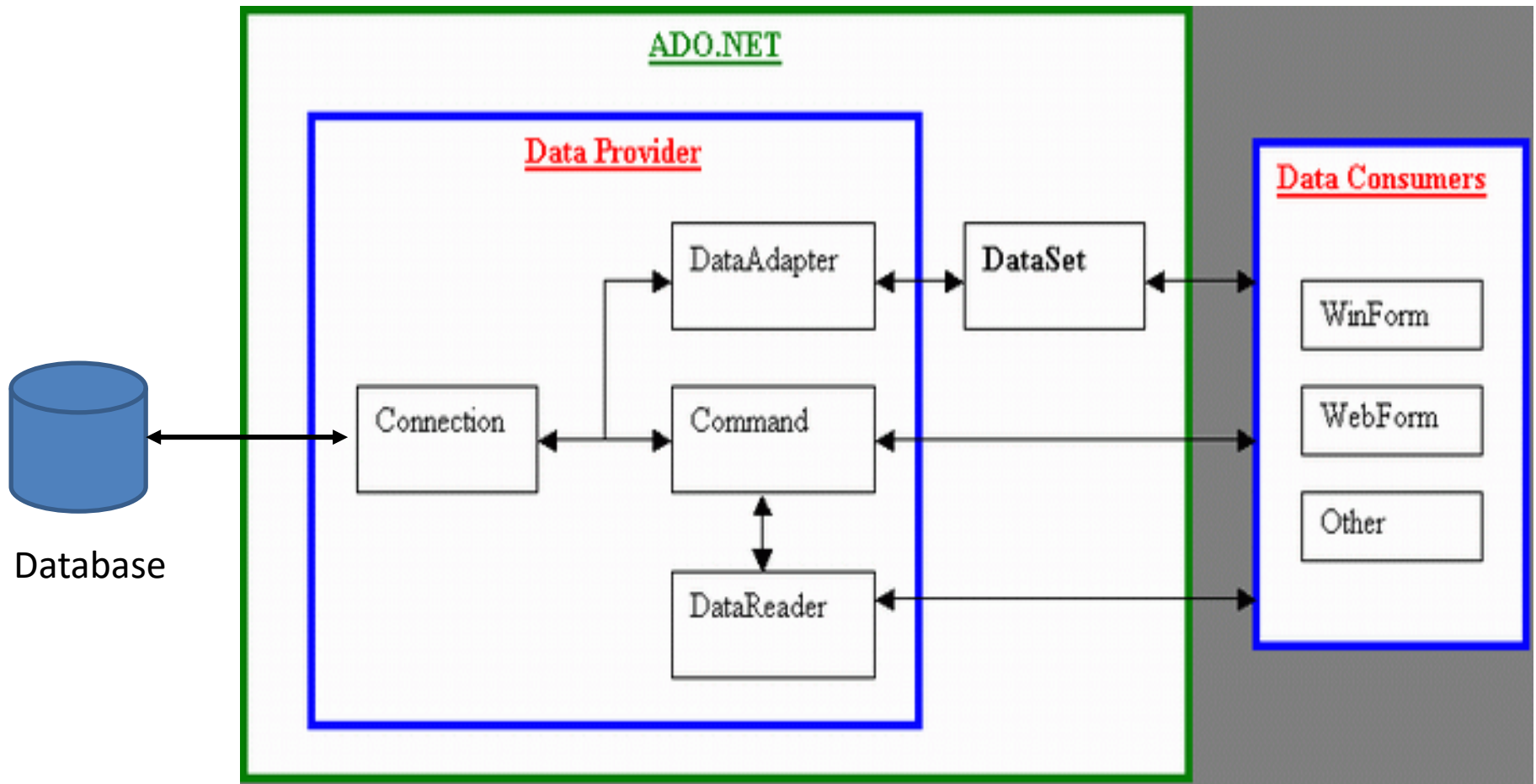
## DataSet:

- Like 2D array: `a[r][c]`
- Full dataset read from DB at query execution time
- Dataset cached locally
- Can disconnect from DB
- Implemented using `DataAdapter`
- + random data access
- - Limited scalability
- - initial read very slow

## DataReader: (standard)

- 1 row at a time
- No local memory storage
- Fetch each row from DB on demand
- “cursor” = current row
- Must stay connected to DB
- + scalability
- - each row slow
- - random access difficult

# ADO.NET Architecture



Examples?

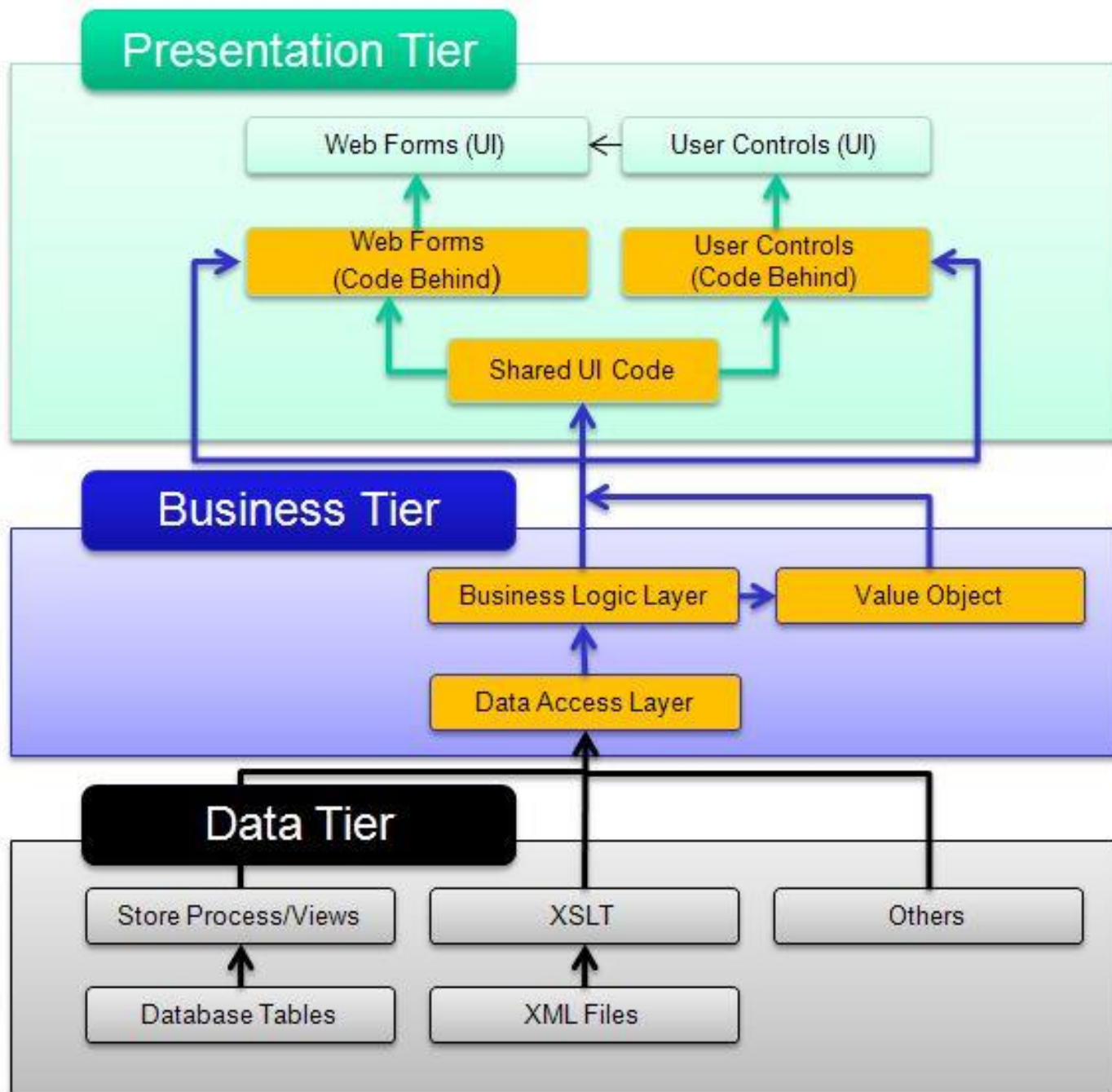
# DataReader with MS SQL DB

- Example?

# DataSet with MS SQL DB

- Example?

## **3- Tier Architecture in C#**



# **Crystal Reports in C#**