

## **II Database basis, Isolation, ADO .Net, XML**

- Overzicht bestaande RDBMS, SQL, ACID  
<http://en.wikipedia.org/wiki/ACID>

- Isolation Models

<http://msdn2.microsoft.com/en-us/library/ms173763.aspx>

- o Read Committed

Specifies that statements can read rows that have been modified by other transactions but not yet committed.

- o Read Uncommitted

Specifies that statements cannot read data that has been modified but not committed by other transactions. This prevents dirty reads. Data can be changed by other transactions between individual statements within the current transaction, resulting in nonrepeatable reads or phantom data. This option is the SQL Server default.

- o Repeatable Read

Specifies that statements cannot read data that has been modified but not yet committed by other transactions and that no other transactions can modify data that has been read by the current transaction until the current transaction completes.

- o Serializable

Specifies the following:

- Statements cannot read data that has been modified but not yet committed by other transactions.
- No other transactions can modify data that has been read by the current transaction until the current transaction completes.

- Other transactions cannot insert new rows with key values that would fall in the range of keys read by any statements in the current transaction until the current transaction completes.

- Snapshot

Specifies that data read by any statement in a transaction will be the transactionally consistent version of the data that existed at the start of the transaction. The transaction can only recognize data modifications that were committed before the start of the transaction. Data modifications made by other transactions after the start of the current transaction are not visible to statements executing in the current transaction. The effect is as if the statements in a transaction get a snapshot of the committed data as it existed at the start of the transaction.

## - ADO .Net

### o Abstract Data Provider Classes

```
System.Data.DbConnection  
System.Data.DbCommand  
System.Data.DbParameter  
System.Data.DbDataAdapter  
System.Data.DbDataReader  
System.Data.DbTransaction
```

### o Specifiek per provider, bv.

```
System.Data.Odbc.OdbcConnection  
System.Data.OleDb.OleDbConnection  
System.Data.SqlClient.SqlConnection  
System.Data.OracleClient.OracleConnection
```

### o Connection

<http://www.connectionstrings.com/> (.Net)

#### Standaard:

```
Data Source=myServerAddress;Initial Catalog=myDataBase;User  
Id=myUsername;Password=myPassword;
```

#### Trusted:

```
Data Source=myServerAddress;Initial Catalog=myDataBase;Integrated  
Security=SSPI;
```

```
string connectionString = @" Data Source=(local);Initial  
Catalog=FishDb;Integrated Security=SSPI";
```

```
SqlConnection con = new SqlConnection(connectionString);  
con.Open();  
con.Close();
```

### o Transaction

```
SqlTransaction tcx = con.BeginTransaction(IsolationLevel);  
tcx.Commit();  
tcx.Rollback();
```

- **Command**

```
SqlCommand cmd = new SqlCommand("Select Name From Fishes", con, tcx);

DataReader dr = cmd.ExecuteReader();           // Select * from Emp;
object o = cmd.ExecuteScalar();              // Select Max(Age) from Emp;
cmd.ExecuteNonQuery();                       // Delete from Emp;
```

- **DataReader**

**Provides a way of reading a forward-only stream of rows from a SQL Server database.**

```
SqlDataReader rd = cmd.ExecuteReader();
while (rd.Read())
{
    Console.WriteLine(rd.GetString(1));
}
rd.Close();
```

- **DataSet**

**Represents an in-memory cache of data.**

```
SqlDataAdapter da = new SqlDataAdapter();
da.SelectCommand = cmd;
da.InsertCommand = cmd;
...
DataSet ds = new DataSet();
da.Fill(ds);

DataRow dr = da.Tables[0].Rows[0];
```

## - Xml, XPath

<http://www.w3schools.com/xml/>  
<http://www.w3schools.com/xpath/>

### o System.Xml.XmlDocument

```
XmlDocument doc = new XmlDocument();  
using (FileStream fs = new FileStream(docpath, FileMode.Open))  
{  
    doc.Load(fs);  
}
```

```
XmlTextWriter writer = new XmlTextWriter(docpath, Encoding.UTF8);  
writer.Formatting = Formatting.Indented;  
doc.Save(writer);
```

### o System.Xml.XmlNode

```
XmlNode node = doc.SelectSingleNode("//fishes");
```

```
XmlNode n2 = doc.CreateElement("fish");  
node.AppendChild(n2);
```

### o System.Xml.XmlNodeList

```
XmlNodeList nodelist = doc.SelectNodes("//Fish");  
Console.WriteLine(nodelist.InnerXml);
```

### o System.Xml.XmlAttribute

```
XmlAttribute attribute = n.Attributes["name"];  
Console.WriteLine(attribute.Value);
```

```
XmlAttribute a2 = doc.CreateAttribute("type");  
a2.Value = "Groot";  
node.Attributes.Append(a);
```