

## **Lezione IV: accesso ai Database con la tecnologia ADO<sup>1</sup>**

Questa lezione, riveste un ruolo importante, perché introduce i concetti chiave per quanto riguarda l'accesso ai dati in generale e ai Database in particolare.

Una delle funzioni più interessanti di ASP in ambiente Windows è la capacità di accedere ai Database. Si tratta di una funzionalità essenziale per costruire applicazioni Web come *e-commerce*, *e-government* e altre.

ADODB (*ActiveX Data Object DB*)<sup>2</sup> è un superoggetto che contiene gli oggetti COM per manipolare i Database compatibili ODBC (*Open DataBase Connectivity*).

ODBC è uno standard che consente alle applicazioni di accedere ai Database indipendentemente dal particolare prodotto utilizzato, servendosi di un *driver* di interfaccia ODBC, implementato dal costruttore del Database, e indirizzando il Database con un nome logico detto DSN (*Data Source Name*; memorizza un collegamento a un Database con le seguenti informazioni: indirizzo IP del server o directory remota, username, password e driver ODBC da usare).

Per i nostri scopi, gli esempi che seguono utilizzano il Database relazionale Access per la sua semplicità e diffusione, anche se in ambiente server Web si usano potenti Database relazionali di classe enterprise. I Database Access sono contenuti in un singolo file con estensione *.mdb*.

In Windows la procedura per creare un DSN è molto semplice. In Windows 2000 Professional: **Start**→**Pannello di Controllo**→**Strumenti di amministrazione**→**Origine dati (ODBC)**. Dalla scheda **DSN di sistema** si seleziona **Aggiungi**, quindi si sceglie il driver per Access.

I componenti *ADO* forniscono gli oggetti e i metodi per la connessione ai Database e per l'accesso ai dati in esso contenuti.

---

<sup>1</sup> Abbreviazione di *ActiveX Data Object* che indica l'interfaccia ad alto livello di Microsoft per l'accesso ai dati e per la connettività ai Database. ADO rimpiazza ed estende gli obsoleti DAO e RDO, disegnati per accedere soltanto ai Database relazionali.

<sup>2</sup> ActiveX è una tecnologia sviluppata da Microsoft per espandere altri due sottosistemi già esistenti e usati dai programmatori conosciuti col nome di OLE (*Object Linking and Embedding*: è il sistema sviluppato da Microsoft per consentire il collegamento e l'inserimento di oggetti all'interno di altre applicazioni) e COM (*Component Object Model*). In realtà oggi non esiste più una definizione rigorosa del termine "ActiveX", perché ingloba al suo interno una serie di altre tecnologie ed è talmente vasta che è difficile distinguere cosa è ActiveX da cosa non lo è. In pratica si può dire che ActiveX non è più un termine puramente tecnico, ma solo un'etichetta di mercato venduta da Microsoft ai programmatori.

In particolare sono disponibili gli oggetti predefiniti:

- **ADODB.Connection**, per stabilire una connessione con l'origine dei dati, cioè con il Database residente sul server;
- **ADODB.RecordSet**, per conservare l'insieme delle righe della tabella ottenute come risultato di un'interrogazione oppure le righe sulle quali si effettuano le operazioni di manipolazione.

Ho sottolineato il fatto che un *RecordSet* in pratica è una tabella temporanea costruita eseguendo una *query* su un Database. Le righe corrispondono ai record, le colonne ai campi. È dotato di una variabile detta *RecordPointer* che punta al record desiderato.

Il programmatore di pagine ASP può utilizzare questi oggetti predefiniti creando prima di tutto un'*istanza* dell'oggetto e assegnando ad essa un nome, con il metodo **CreateObject**:

```
Set miaconnessione = Server.CreateObject("ADODB.Connection")  
Set miorecordset = Server.CreateObject("ADODB.RecordSet")
```

I RecordSet saranno popolati effettuando una query su un Database, poi si faranno le elaborazioni sul RecordSet: ricerche, inserimenti, modifiche, cancellazioni, visualizzazioni. Per ultima cosa si chiuderà il RecordSet per liberare la memoria centrale, applicando il metodo **Close**: *miorecordset.Close*.

Se sul server Web a disposizione è stata configurata l'interfaccia per i dati *OLE DB*, l'apertura della connessione richiede di specificare, come parametri, le caratteristiche del Database (*provider* dell'origine dati) e il nome del Database.

Per i nostri scopi, l'accesso al Database *Access* richiede la seguente stringa di connessione (per comodità di lettura, l'assegnazione è effettuata usando due righe):

```
DataBase = Server.MapPath ("Magazzino.mdb")  
strconn = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & DataBase & ";Persist Security Info=False"
```

Con queste impostazioni, la connessione viene poi aperta con l'istruzione:

```
miaconnessione.Open strconn
```

utilizzando il metodo **Open** dell'oggetto **Connection**.

I dati contenuti in una tabella sono resi disponibili con il metodo **Open** dell'oggetto **RecordSet**:

```
miorecordset.Open nometabella, miaconnessione
```

dove *nometabella* indica la tabella del Database e *miaconnessione* il nome della connessione.

Per spostarsi da un record all'altro basta applicare al RecordSet il metodo **MoveNext**. Occorre controllare che non si superi l'ultimo record; per questo motivo si controlla la proprietà *EOF* (*End Of File*) del RecordSet:

```
If Not miors.EOF Then
  Do While Not miors.EOF
    Response.Write(miors.Fields("Codice").Value & " - " & miors.Fields("Descr").Value & "<BR>")
    miors.MoveNext
  Loop
Else
  Response.Write("Nessun record trovato ...")
End If
```

Esistono anche i metodi **MovePrevious**, **MoveFirst**, **MoveLast**, **Move**.

Altri metodi per lavorare sui RecordSet sono:

- **AddNew**: aggiunge un nuovo record al RecordSet;
- **Delete**: cancella il record corrente.

Inoltre è possibile accedere al valore di un campo del RecordSet corrente in modo molto semplice:

**miors ("nome\_campo")**.

Si può quindi leggere, testare, modificare, cancellare ogni valore contenuto nei campi del RecordSet corrente.