Uso dei data bases con PHP

Prof. Francesco Accaino
Iis Altiero Spinelli
Sesto Sa Giovanni

MDAC

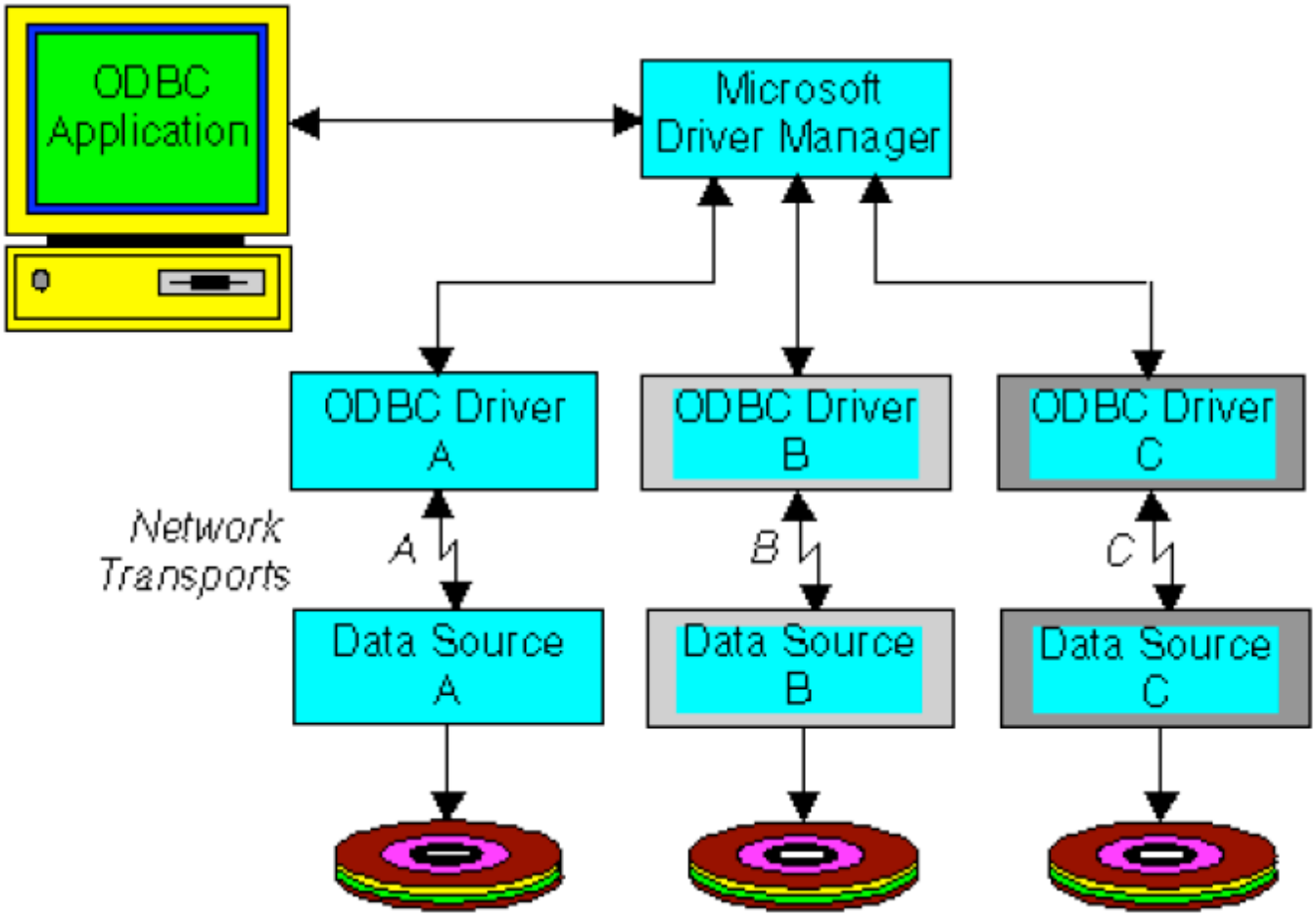
- MDAC è l'acronimo di Microsoft Data Access Component e fa parte
- della tecnologia Microsoft denominata Universal Data Access (UDA).
- Mette a disposizione una serie di componenti per l'accesso a svariate tipologie di dati.
- Di questa tipologia fanno parte:
  - **ADO**
  - **OLE DB**
  - **ODBC**

ODBC

- Open Database Connectivity (ODBC), È un'interfaccia composta da una serie di API che consentono l'accesso ai dati di diversi DBMS (Data Base Management System) fornendo un linguaggio comune d'interazione.

- Grazie a ODBC è possibile far dialogare tra loro applicazioni e origini dati diversi, a un livello di astrazione tale che la tipologia di dati noninfluisca sulla modalità di interrogazione degli stessi.

Architettura ODBC



Accesso ai dati

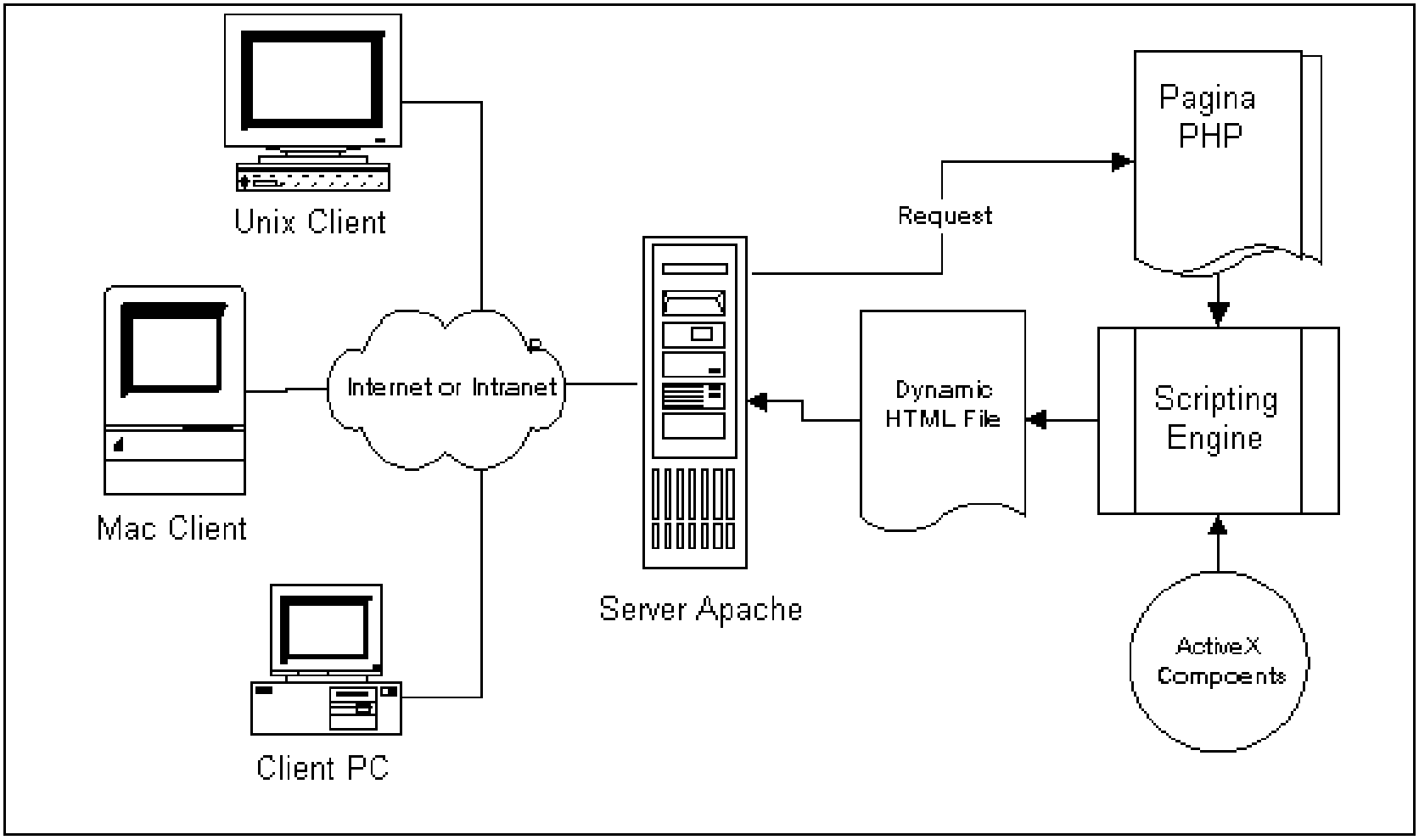
L'accesso ad un database (remoto) tramite ODBC richiede la cooperazione di 4 componenti:

- L'**Applicazione** richiama le funzioni SQL per esequire query ed acquisire i risultati. E' trasparente rispetto al protocollo di rete, al sistema operativo ed al server DBMS utilizzati, poiché mascherati dal driver.
- Il **Driver Manager** è responsabile di caricare i driver richiesti dall'applicazione. Viene fornito da Microsoft e garantisce la gestione della corrispondenza tra i nomi e l'inizializzazione dei processi coinvolti.
- I **driver** sono responsabili di eseguire le funzioni ODBC, eseguendo le query SQL traducendole nella sintassi (e semantica) del proprio server di accesso. I driver sono anche responsabili della restituzione dei risultati alle applicazioni tramite meccanismi di buffer.
- La **fonte dei dati** (Data Source Name) rappresenta il database con il quale si vuole comunicare. I driver inviano le chiamate (opportunamente tradotto) al Data Source che le esegue sul server DBMS.

Oggetti utilizati

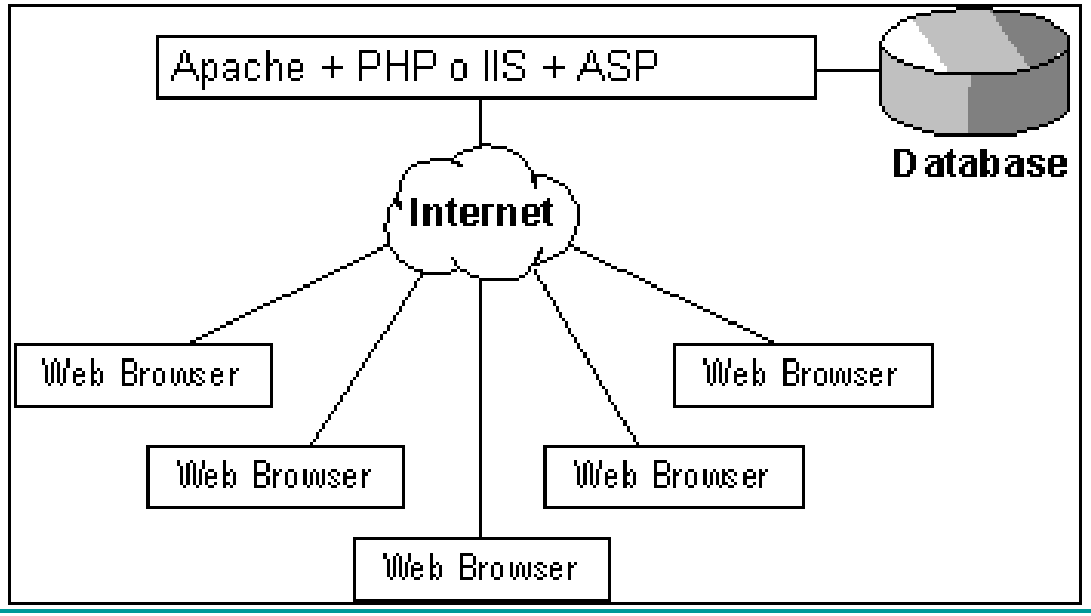
- Quando si creano oggetti con PHP si creano in pratica oggetti OLE (Object Linking and Embedding) o ActiveX Component. Tramite gli OLE gli utenti possono creare documenti "contenitori" che comprendono parti gestite da altri programmi, gli OLE si basano sul modello COM (Component Object Model).
- Tutta la specifica di ActiveX, essendo integrata nella specifica OLE, si basa su questo modello.
- COM fornisce l'architettura e il meccanismo per creare component che saranno condivisi tra applicazioni. Sulla base del modello COM è stato definito DCOM (Distribuited Component Object Model) che definisce una specifica e una mplementazione delle interfacce tra oggetti attraverso la rete.

Interazione Client Server

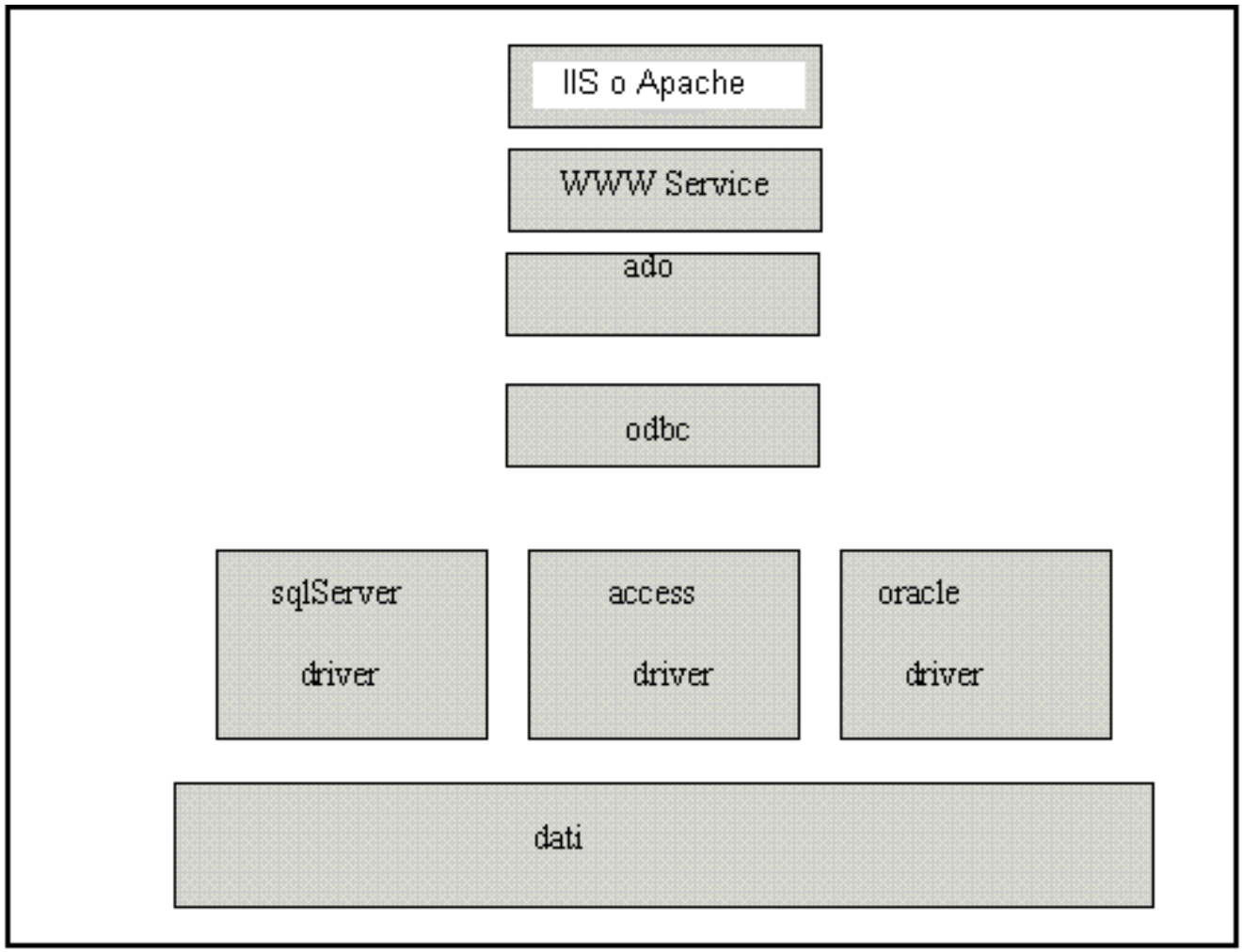


ADO

- **ActiveX Data Object Component:** Fornisce l'accesso a un database da un'applicazione web mediante Active database Object (ADO).
- ADO fornisce un semplice accesso per mezzo di ODBC standard a Micorsoft Access, SQL Server, Oracle, Informix , Sybase.



Struttura Di accesso ai dati

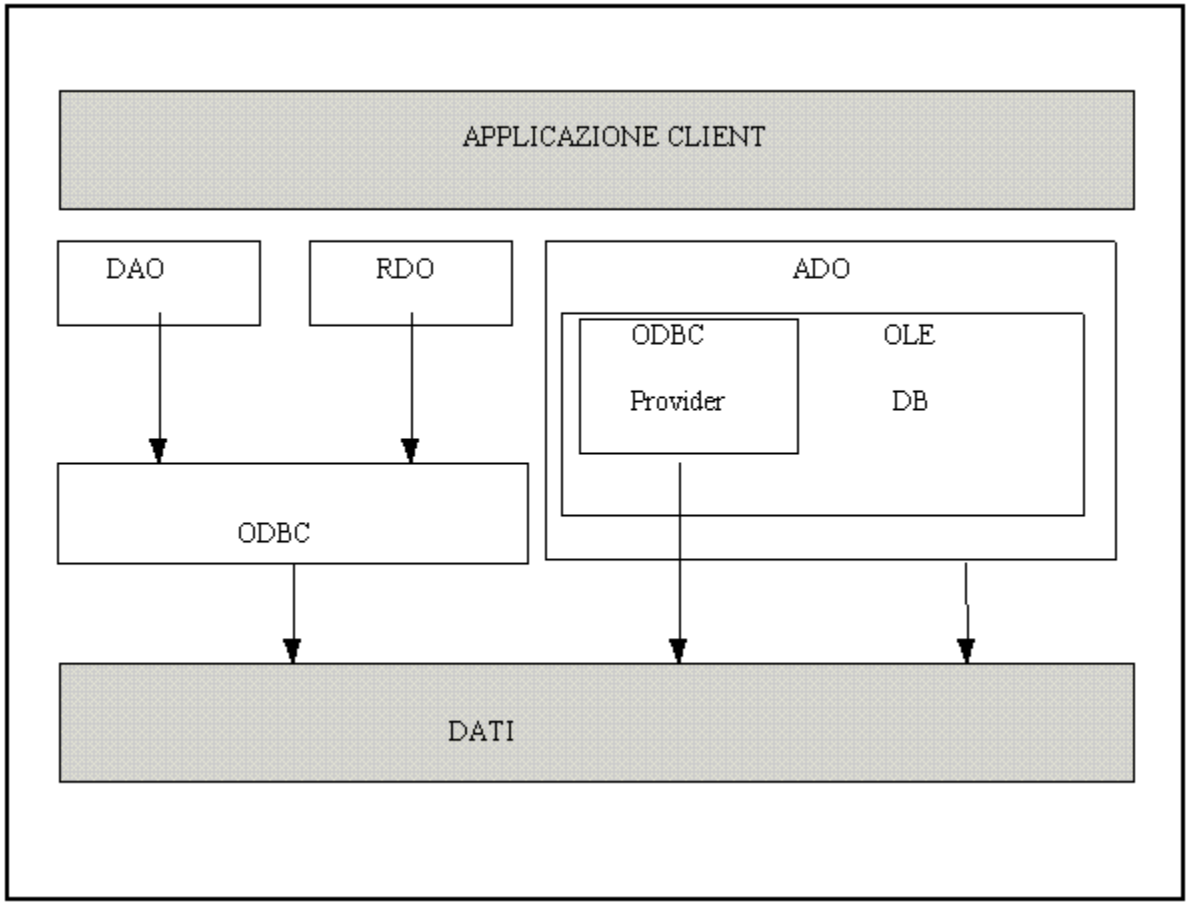


OLE DB

- OLE DB è l'evoluzione di ODBC. Sono delle interfacce COM che consentono di scrivere applicazioni senza tener conto della struttura dei dati.
- I componenti OLE DB sono formati da:
  - Data Providers: consentono l'accesso alle informazioni usando una struttura astratta comune.
  - Service components: elaborano e trasportano le informazioni.
  - Data consumers: usano le informazioni.

Accesso ai dati

OLE DB accedono ai dati in maniera diretta e molto più rapidamente di DAO (Data Acess Object) o RDO (Remote Data Object) che invece passano attraverso il livello ODBC.



Gli Oggetti ADO

- ADO espone tre oggetti principali:
  - Connection
  - Recordset
  - Command
- Tali oggetti espongono altri componenti come Fields, Property …..

Come connettersi ad un database access

ADODB è il tramite necessario per operare nelle pagine PHP o ASP , sia in lettura che in scrittura, su una fonte di dati ODBC.

L'oggetto ActiveX che si occupa di fornire la connettività verso un database è **Connection**, che va dichiarato come un qualsiasi COM

```
$connessione = new COM("ADODB.Connection");
```

Tramite il riferimento **$connessione** diventa possibile accedere a metodi e proprietà dell'oggetto Connection.

Oggtto Connection

L'oggetto **Connection** possiede numerose proprietà e metodi, tra i quali più importanti sono:

Proprietà:

ConnectionString (Contiene le informazioni utilizzate per stabilire una connessione a una fonte dati)

Metodi:

Open (Apre una connessione a una fonte dati)
Execute (Esegue la particolare istruzione SQL passata al metodo mediante un parametro stringa)
Close (Chiude un oggetto aperto)

Operazioni tipiche

Si crea un oggetto **Connection**:

$connessione = new COM("ADODB.Connection")

Si aprire il database:

$connessione->Open($stringa_di_connessione);

passandogli un parametro **$stringa_di_connessione** opportuno

Si eseguono operazioni SQL sul database:

$recordset = $connessione->execute($interrogazione);

passandogli un parametro **sql** opportuno

Si chiude il database:

$connessione->close();

Oggetto Connection

La stringa (stringa di connessione) che si deve passare al metodo open in :

$connessione = new COM("ADODB.Connection")

è costituita da una serie di coppie **chiave=valore**, separate tra di loro con un punto e virgola.

Nel caso tipico è necessario inserire in tale stringa almeno due chiavi, che indicano:

- il formato di database utilizzato **("driver=…")**
- la sua locazione fisica nel file system del server **("dbq=…")**

Esempio

```
$percorso_database = realpath("studenti.mdb");

$connessione = new COM("ADODB.Connection");

$stringa_di_connessione = "DRIVER={Microsoft Access
    Driver(*.mdb)};DBQ=".$percorso_database;

$connessione->Open($stringa_di_connessione);
```

Oggetto Recordset

Il metodo Execute, come visto nell'esempio:

$recordset = $connessione->execute($interrogazione);

restituisce un riferimento ad un oggetto di tipo **Recordset** (che viene quindi creato automaticamente)**, che** rappresenta una **collezione di record** del database.

Ad esempio, con

$ris = $connessione.Execute("SELECT * FROM nominativi");

l'oggetto **ris** contiene tutti i campi di tutti i record presenti nella tabella **nominativi** del database.

E' possibile creare un oggetto RecordSet direttamente:

```
$recordset = new COM("ADODB.Recordset");
$recordset->Open($interrogazione,$connessione);
```

Oggetto Recordset

I Recordset possono essere immaginati come vere e proprie tabelle di dati, dove ogni riga corrisponde ad un record, un po' come nella rappresentazione visuale di Access.

I valori dei singoli campi del record evidenziato dal cursore sono leggibili tramite l'utilizzo della sintassi:

$recordset->fields['IdStudente']->value

Per poter visualizzare i dati della tabella è quindi possibile sfruttare i metodi e le proprietà degli oggetti Recordset.

Esempio di utilizzo dei metodi eof e movenext

```
while (!$recordset->eof) {
    echo'<tr>';
    echo'<td> '.$recordset->fields['IdStudente']->value.'</td>';
    echo'<td> '.$recordset->fields['Cognome']->value.'</td>';
    echo'<td> '.$recordset->fields['Nome']->value.'</td>';
    echo'<td> '.$recordset->fields['Telefono']->value.'</td>';
    echo'</tr>';
    $recordset->movenext();
}
```

Ogetto Recordset metodi principali

M**etodi** principali:

- AddNew
- Close
- Delete
- Find
- Move
- MoveFirst - MoveLast
- MoveNext - MovePrevious
- Open
- Seek
- Update

Proprietà principali:

- BOF (primo record del recordset)
- EOF (ultimo record del recordset)
- Index

Utilizzo di Recordset

Per quel che riguarda le operazioni di ricerca, inserimento, modifica e cancellazione ci sono due "filosofie":

1. Si usa una stringa SQL nel metodo Execute dell'oggetto Connection per eseguire tutte le operazioni; si usa il RecordSet ottenuto da una ricerca solo per la visualizzazione.

2. Si crea un oggetto RecordSet e su di esso si fanno tutte le operazioni volute.

Usare l'oggetto RecordSet o SQL?

Nel primo caso si possono fare operazioni complesse (ricerca tutti i record dove …, cancella il record nel quale il nome è ...) con la scrittura di un'espressione SQL.

Nel secondo caso si utilizzano le normali strutture di programmazione per la ricerca sul RecordSet, gestito in modo analogo ad un array di record.

Noi utilizzeremo il primo metodo.

SQL

Vediamo quindi come si eseguono ricerche, inserimenti, modifiche o cancellazioni su un database, utilizzando una stringa SQL nel metodo Execute dell'oggetto Connection.

Dopo aver dichiarato un oggetto Connection **$connesione**, si utilizza il metodo Execute:

$ris = $connesione->Execute(sql);

passandogli una stringa **sql**, contenente un'istruzione SQL da eseguire. Cioè all'interno di questa stringa viene **dichiarato** al metodo Execute cosa fare.

SQL

SQL (Structured Query Language) è un linguaggio che rappresenta lo standard per la definizione, l'interrogazione e la modifica dei database.

La conoscenza dell'SQL esula dagli scopi di questo corso, ma per poter realizzare qualche applicazione è necessario vederne almeno qualche esempio.

SQL

Recupero dati o interrogazioni (*query*) in SQL

SELECT [<lista-colonne> | *****]
FROM <lista-tabelle>
[**WHERE** <condizione>]

La **condizione** della clausola **WHERE** può essere una condizione semplice o composta, mediante gli operatori **AND OR** e **NOT**, da predicati semplici, in cui ciascun predicato rappresenta un confronto tra due valori.

Esempi di ricerca:

```
SELECT * FROM nominativi

SELECT * FROM nominativi WHERE id > 3

SELECT * FROM nominativi WHERE id > 3 AND id < 15

SELECT * FROM nominativi WHERE nome = 'Carlo'

SELECT nome,cognome FROM nominativi WHERE
nome LIKE 'carlo'

SELECT * FROM nominativi WHERE nome LIKE
'%carlo'

SELECT * FROM nominativi WHERE nome LIKE
'%carlo%'

SELECT * FROM nominativi WHERE nome LIKE
'%carlo%' ORDER BY cognome

SELECT * FROM nominativi WHERE nome LIKE
'%carlo%' ORDER BY cognome DESC
```

Operazioni di aggiornamento in SQL

INSERT INTO <nome-tabella> [(<listanomi-colonne>)]
VALUES (<listavalori-colonne>)

Es:
INSERT INTO STUDENTI VALUES
('Anna','Turchese','355773','Via Piave, 7 - Treviso')

DELETE FROM <nome-tabella>
WHERE <condizione>
Es:
DELETE FROM STUDENTI WHERE Telefono='02-333878'

Operazioni di aggiornamento in SQL

UPDATE <nome-tabella> SET <nome-attributo> = [<espressione> {, <nome-attributo> = <espressione>] [WHERE <condizione>]

Es:

```
UPDATE STUDENTI
SET Codice = 'IN'
WHERE Matricola > 400001
```

```
UPDATE nominativi
SET [e-mail] = 'carlo@tin.it'
WHERE nome = 'Carlo' AND cognome = 'Rossi'
```

ESERCITAZIONE

Progetto proposto:

RUBRICA TELEFONICA / E-MAIL (realizzata con database) - o un altro db a piacere

nb.: si dovrebbe realizzare una pagina base (HTML) che consenta di scegliere l'operazione desiderata (lista completa, ricerca, modifica, inserimento nuovi dati e cancellazione), attivando quindi la pagina (HTML) con un modulo che richiami una pagina asp opportuna:

ESERCITAZIONE