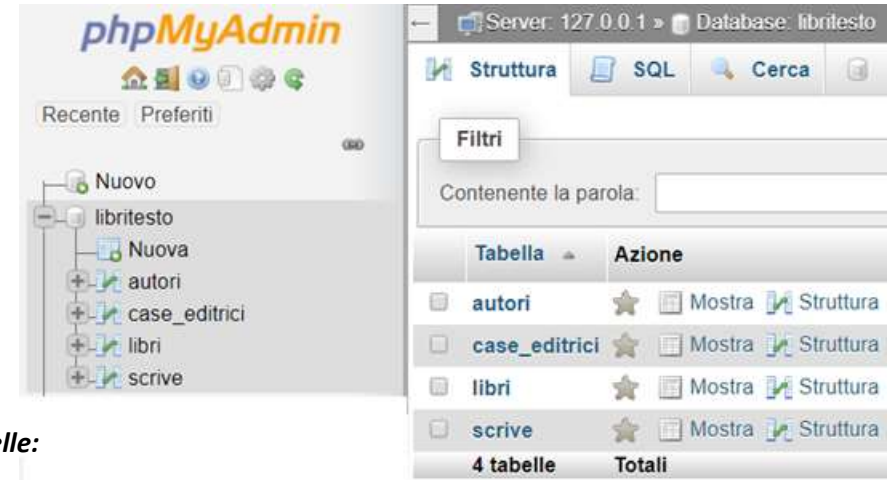
Esercizio: database LibriTesto – DBMS MySQL

III. Considerato il seguente Database

LIBRI (Id_LIBRO, TITOLO, NR_PAGINE, PREZZO, DATA_PUBBL, *COD_Ed*)
AUTORI (COD_AU, COGNOME, NOME)
CASE_EDITRICI (COD_CE, NOME, CITTA)
SCRIVE (*Id_LIBRI*, *Id_AUTORE*)

a) Rispondere alle query seguenti:

- Visualizzare per ciascuna casa editrice il numero di libri aventi prezzo inferiore a €30 e pubblicati nell'anno corrente
- Modificare il numero di pagine di un libro il cui codice è specificato in input.



Creiamo le tabelle:

```
CREATE TABLE Libri
(
Id_Libro int(11) NOT NULL AUTO_INCREMENT,
Cod_Ed INT NOT NULL,
Titolo varchar(255) NOT NULL,
Nr_Pagine INT(11),
prezzo double (5,2),
Data_Pubbl date,                          /* Format: YYYY-MM-DD */
FOREIGN KEY (Cod_Ed) REFERENCES Case_Editrici(Cod_Ce),
PRIMARY KEY (Id_Libro)
)

CREATE TABLE Autori
(
Cod_Au int(11) NOT NULL AUTO_INCREMENT,
Cognome varchar(255) NOT NULL,
Nome varchar(255) NOT NULL,
PRIMARY KEY (Cod_Au)
)

CREATE TABLE Case_Editrici
(
Cod_Ce int(11) NOT NULL AUTO_INCREMENT,
Città varchar(255) NOT NULL,
Nome varchar(255) NOT NULL,
PRIMARY KEY (Cod_Ce)
)
```

```
CREATE TABLE Scrive
(
id_libri int(11) NOT NULL,
id_autore int(11) NOT NULL,
FOREIGN KEY (id_libri) REFERENCES Libri(Id_Libro),
FOREIGN KEY (id_Autore) REFERENCES Autori(Cod_Au)
)
```

Popoliamo le tabelle utili per le query richieste

```
INSERT INTO Case_Editrici  (Città, Nome)  VALUES ('Bologna','Zanichelli');
INSERT INTO Case_Editrici  (Città, Nome)  VALUES ('Milano','Hoepli')
```

Cod_Ce	Città	Nome
1	Bologna	Zanichelli
2	Milano	Hoepli

Libri di Zanichelli: 1 solo libro / Libri di Hoepli: 3

```
INSERT INTO Libri  (Cod_Ed, Titolo, Nr_Pagine, prezzo, Data_Pubbl)
VALUES (1, 'Titolo 1', 120, 29.99,  '2019-02-24' );
INSERT INTO Libri  (Cod_Ed, Titolo, Nr_Pagine, prezzo, Data_Pubbl)
VALUES (1, 'Titolo 2', 440, 49.67,  '2019-02-24' );
INSERT INTO Libri  (Cod_Ed, Titolo, Nr_Pagine, prezzo, Data_Pubbl)
VALUES (1, 'Titolo 3', 220, 30.99,  '2019-01-24' );
INSERT INTO Libri  (Cod_Ed, Titolo, Nr_Pagine, prezzo, Data_Pubbl)
VALUES (1, 'Titolo 4', 420,  24.99,  '2017-03-24' );

INSERT INTO Libri  (Cod_Ed, Titolo, Nr_Pagine, prezzo, Data_Pubbl)
VALUES (2, 'Titolo SIS', 220, 21.99,  '2019-03-24' );
INSERT INTO Libri  (Cod_Ed, Titolo, Nr_Pagine, prezzo, Data_Pubbl)
VALUES (2, 'Titolo SIS2', 220, 22.99,  '2019-02-24' );
INSERT INTO Libri  (Cod_Ed, Titolo, Nr_Pagine, prezzo, Data_Pubbl)
VALUES (2, 'Titolo SIS3', 220, 23.99,  '2019-01-24' );
```

Id_Libro ▲ 1	Cod_Ed	Titolo	Nr_Pagine	prezzo	Data_Pubbl
1	1	Titolo 1	120	29.99	2019-02-24
2	1	Titolo 2	440	49.67	2019-02-24
3	1	Titolo 3	220	30.99	2019-01-24
4	1	Titolo 4	420	24.99	2017-03-24
5	2	Titolo SIS	220	21.99	2019-03-24
6	2	Titolo SIS2	220	22.99	2019-02-24
7	2	Titolo SIS3	220	23.99	2019-01-24

Selezione di tutti i campi senza conteggio con condizioni richieste

```
SELECT *
FROM `case_editrici`, libri
WHERE COD_CE =COD_Ed
AND prezzo < 30
AND Year(Data_Pubbl) = Year(CURDATE())
```

Cod_Ce	Città	Nome	Id_Libro	Cod_Ed	Titolo	Nr_Pagine	prezzo	Data_Pubbl
1	Bologna	Zanichelli	1	1	Titolo 1	120	29.99	2019-02-24
2	Milano	Hoepli	5	2	Titolo SIS	220	21.99	2019-03-24
2	Milano	Hoepli	6	2	Titolo SIS2	220	22.99	2019-02-24
2	Milano	Hoepli	7	2	Titolo SIS3	220	23.99	2019-01-24

```
SELECT Nome, Titolo, Nr_Pagine, prezzo,  Year(Data_Pubbl) AS Anno_Pubbl
FROM `case_editrici`, libri
WHERE COD_CE =COD_Ed
AND prezzo < 30
```

Nome	Titolo	Nr_Pagine	prezzo	Anno_Pubbl
Zanichelli	Titolo 1	120	29.99	2019
Zanichelli	Titolo 4	420	24.99	2017
Hoepli	Titolo SIS	220	21.99	2019
Hoepli	Titolo SIS2	220	22.99	2019
Hoepli	Titolo SIS3	220	23.99	2019

· Visualizzare per ciascuna casa editrice il **numero di libri** aventi prezzo inferiore a €30 e pubblicati nell'anno corrente

```
Select Case_Editrici.Nome, count(*) AS numLibri
FROM `case_editrici`, libri
WHERE COD_CE =COD_Ed
AND prezzo < 30
AND Year(Data_Pubbl) = Year(CURDATE())
Group BY Case_Editrici.Nome;
```

Nome	numLibri
Hoepli	3
Zanichelli	1

Nb: *volendo estrarre tutti i libri pubblicati nel* ***mese corrente***

```
SELECT Titolo, Month(CurDATE()) AS mese
FROM `libri`
Where Month(Libri.Data_Pubbl) = Month(CurDATE())
```

Titolo	mese
Titolo 4	3
Titolo SIS	3

… o formattando

```
SELECT Titolo, DATE_FORMAT(CurDATE(),"%M")  AS mese
FROM `libri`
Where Month(Libri.Data_Pubbl) = Month(CurDATE())
```

Titolo	mese
Titolo 4	March
Titolo SIS	March

- Modificare il numero di pagine di un libro il cui codice è specificato in input.

Query parametrica con MySQL:

```
SET @a= 2;
UPDATE libri
SET Nr_pagine=310
WHERE Id_Libro = @a
```

Id_Libro	Cod_Ed	Titolo	Nr_Pagine	prezzo	Anno_Pubbl
1	1	Titolo 1	120	29.99	2019-02-24
2	1	Titolo 2	440	49.67	2019-02-24

Id_Libro	Cod_Ed	Titolo	Nr_Pagine	prezzo	Anno_Pubbl
1	1	Titolo 1	120	29.99	2019-02-24
2	1	Titolo 2	310	49.67	2019-02-24

In seguito, proponendo pagine php nell'interazione con DB remoto:

inviare da form il codice del libro le cui pagine devono essere modificate, recuperarlo lato server salvando in variabile che si accoda nella query di aggiornamento

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "nomeDB";

// Create connection MySQLi - OO o equivalente MySQLi - procedurale
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "UPDATE Libri SET Nr_pagine=330  WHERE Id_Libro = '".$_GET['dato']."'";
if ($conn->query($sql) === TRUE) {
    echo "Record updated successfully";
} else {
    echo "Error updating record: " . $conn->error;
}
$conn->close();
?>
```

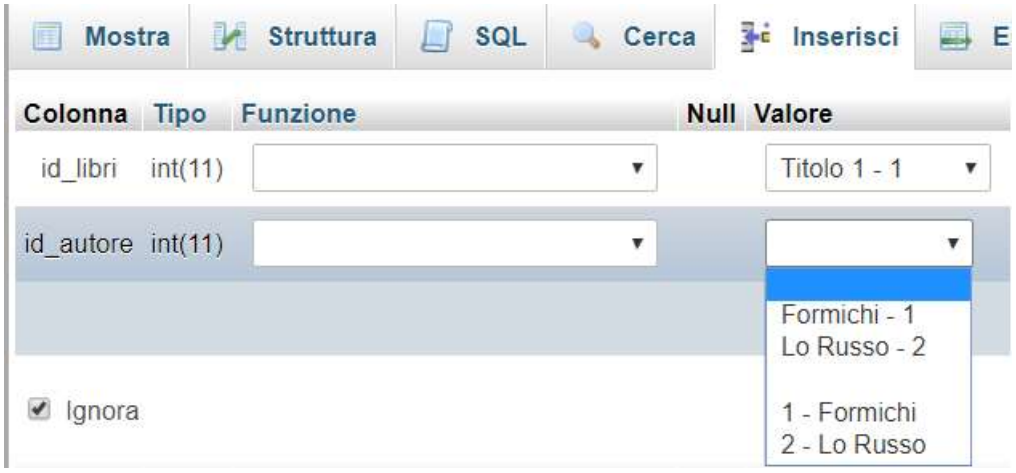
```
<form method="get" action ="update.php">
<p>inserisci codice libro: <input type="text" name ="dato" … ><p>
<p><input type = "submit" value = "invio"><p>
```

Id_Libro ▲ 1	Cod_Ed	Titolo	Nr_Pagine	prezzo	Data_Pubbl
1	1	Titolo 1	120	29.99	2019-02-24
2	1	Titolo 2	330	49.67	2019-02-24

Popoliamo le altre tabelle con uso 

Cod_Au	Cognome	Nome
1	Formichi	Fiorenzo
2	Lo Russo	Luigi

e, sfruttando inserimento facilitato:



si ottiene:

id_libri	id_autore
1	1
2	1
3	1
4	1
5	2
6	2
7	2

Visualizziamo informazioni tratte da tutte le tabelle

```
SELECT `case_editrici`.Nome AS Editore , Titolo, Autori.Nome, Cognome
FROM `case_editrici`, libri, autori, scrive
WHERE COD_CE =COD_Ed
AND id_libri =Id_Libro
AND id_autore =Cod_Au
```

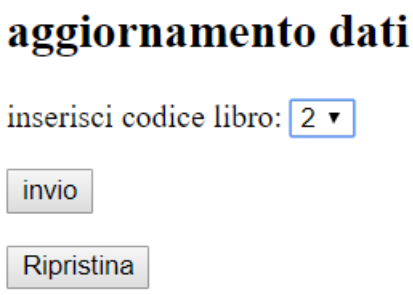
Editore	Titolo	Nome	Cognome
Zanichelli	Titolo 1	Fiorenzo	Formichi
Zanichelli	Titolo 2	Fiorenzo	Formichi
Zanichelli	Titolo 3	Fiorenzo	Formichi
Zanichelli	Titolo 4	Fiorenzo	Formichi
Hoepli	Titolo SIS	Luigi	Lo Russo
Hoepli	Titolo SIS2	Luigi	Lo Russo
Hoepli	Titolo SIS3	Luigi	Lo Russo

Codice pagina update.php

```php
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "libritesto";

// Create connection MySQLi procedurale
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

$sql = "UPDATE Libri SET Nr_pagine=330  WHERE Id_Libro = '".$_GET['dato']."'";

if (mysqli_query($conn, $sql)) {
    echo "Record updated successfully";
} else {
    echo "Error updating record: " . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

Codice semplice form (nella stessa directory)

```html
<html>
<head>
  <title>Scheda</title>
</head>
<body style ="background-color:#FFFFFF;color:#000000">
  <h2>aggiornamento dati</h2>
  <form method="get" action ="update.php">
    <p>inserisci codice libro:
     <select name ="dato">
      <option value="1">1</option>
      <option value="2">2</option>
      <option value="3">3</option>
      <option value="4">4</option>
      <option value="4">5</option>
      <option value="4">6</option>
      <option value="4">7</option>
     </select>
    <p><input type = "submit" value = "invio"><p>
    <p><input type = "reset"><p>
  </form>
</body>
</html>
```



Nb: alternativa all'uso di select (che evita errori di digitazione)
quando il numero di opzioni è alto e/o non prevedibile
è il *controllo sull'input che* determini la **correttezza e la validità** dei dati inseriti nei campi del form
(lato client con uso di JavaScript o lato server) - Semplici esempi con prova-codice

Esempio di gestione di un parametro GET *(solo controllo su **campo vuoto** con filtraggio del contenuto HTML in alternativa all'uso della funzione* PHP empy() *senza usare HTML5*[1]*)*

```php
<?php
/*
 * Controllo se è stato ricevuto il parametro "dato" tramite metodo GET.
 * Se è rilevato lo visualizzo; se non è rilevato avverto l'utente.
 */

if( isset( $_GET['dato'] ) ) {
        $a = $_GET['dato']; // Salvo nella variabile $a il parametro "dato"

        $a = htmlentities( $a ); // Questo "disinnesca" eventuali
                                 // contenuti HTML contenuti in $a

        echo $a; // Mostro a video
} else {
        echo "Nessun parametro di nome dato trovato. Devi inviarlo!";
}
?>
```

Da https://it.wikipedia.org/wiki/PHP#Esempio_connessione_con_database_MySql_usando_l'estensione_MySQLi

Volendo gestire **l'eliminazione di eventuali spazi iniziali e finali**:

trim($_GET['dato'])

infatti la funzione **trim()** è tra quelle che prendono come input una stringa e restituiscono la medesima eliminandone eventuali spazi.

Si possono poi usare alter funzioni PHP dedicate quali la funzione nl2br() per la gestione del ritorno a capo o altri accorgimenti di formattazione nell'interazione con MySQL (impostazione della variabile di configurazione del PHP: magic_quotes_gpc).

Nel caso di controllo sulla **dimensione della variabile** al fine di evitare che un testo troppo lungo occupi troppo spazio rallentando il caricamento della pagina, possiamo usare la funzione PHP strlen()

if (strlen ($messaggio) < $dimensione_massima) { /* *continua nel controllo* */ }

Nel caso di controllo sul **tipo di dato**, utile ad esempio per recuperare un dato numerico per fare un certo calcolo, possiamo usare la funzione PHP (v.4) is_numeric() (eventualmente insieme ad altre funzioni simili come: is_integer(), is_real(), is_string() is_bool(), ecc per altri tipi di dati) in questo modo:

if (is_numeric($messaggio)) { /* *continua nel controllo* */ }

Ma non basta, nel caso di input numerici occorre controllare anche il **"range" di validità del numero** per evitare numeri troppo grossi o troppo piccoli che potrebbero fare andare in overflow il programma.
Quindi attenzione agli zeri! Una divisione per zero genera un errore.

Link interessanti (validazione di form) da w3schools, da Anna Campolo, da html.it, da Mr.Webmaster e falle nella sicurezza (articoli).

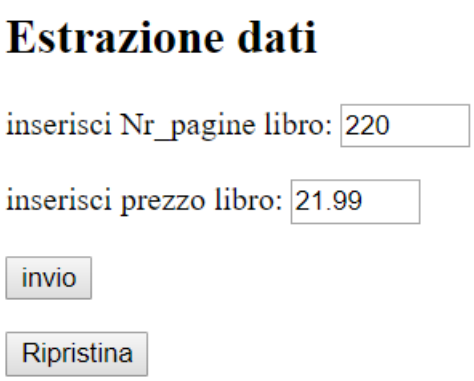
[1] Con HTML5 si può usare l'attributo "required" per indicare al browser quali campi siano obbligatori.

FAQ da alunno 5AI

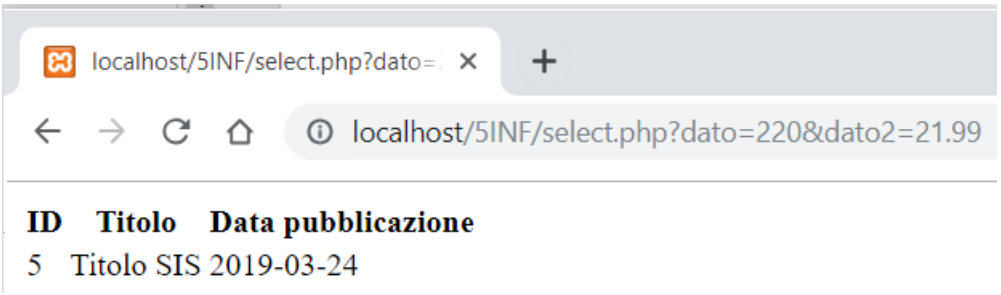
Esempio di estrazione da unica tabella Libri

- estraendo da form (come stringhe) volendo utilizzare tali informazioni come **dati numerici** nello script lato server
- impostando una condizione complessa con AND

Estrarre solo i libri con numero di pagine e prezzo impostati da form (senza controlli)



Nel caso unico libro con tali valori dei campi



Codice form

```
<html>
<head>
	<title>Scheda</title>
</head>
<body style ="background-color:#FFFFFF;color:#000000">
	<h2>Estrazione dati</h2>
	<form method="get" action ="select.php">
		<p>inserisci Nr_pagine libro: <input type="text" name ="dato" size ="5"><p>
		<p>inserisci prezzo libro: <input type="text" name ="dato2" size ="5"><p>
		<p><input type = "submit" value = "invio"><p>
		<p><input type = "reset"><p>
	</form>
</body>
</html>
```

Possibile codice PHP

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "libritesto[2]";

$dato2 = $_GET['dato2'];          // letto da form come stringa

$int = (int)$_GET['dato'];          // convertito int
$real = (float)$_GET['dato2'];      // convertito in float

// Create connection MySQLi – OO (si potrebbe usare estensione MySQLi procedurale nello stesso script)
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
   die("Connection failed: " . $conn->connect_error);
}
$sql = "SELECT * FROM Libri[3] WHERE Nr_Pagine = ".$_GET['dato']." AND prezzo = ".$dato2;

// estrazione e visualizzazione in forma tabellare MySQLi – OO (correzione di svista in w3schools (tre campi)
$result = $conn->query($sql);
if ($result->num_rows > 0) {
  echo "<table><tr><th>ID</th><th>Titolo</th><th>Data pubblicazione</th></tr>";

   // output data of each row
   while($row = $result->fetch_assoc()) {
         echo "<tr><td>". $row['Id_Libro'] . "</td><td>" . $row['Titolo'] ."</td><td>" . $row['Data_Pubbl'] . "</td></tr>";
   }
   echo "</table>";
}
else {
         echo "nessun risultato";
}
$conn->close();
?>
```

Lettura interessante per creare script sicuri per login difendendosi dai seguenti attacchi:

- SQL Injections
- Session Hijacking
- Network Eavesdropping
- Cross Site Scripting
- Brute Force Attacks

Condiviso dal prof. Massaro manualetto che introduce anche Sessioni o Esempio da testo Atlas *(da aggiornare relativamente all'uso dell'estensione mysqli o chiara guida PHP (aggiornata): gestire sessioni html.it)*

---

[2] Con account-free Altervista l'unico DB è *my_nomeUtente*

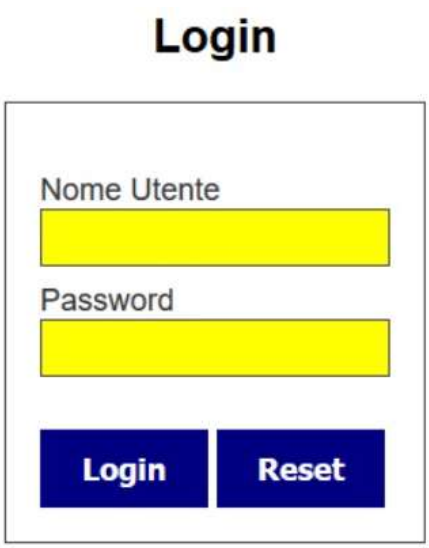
[3] In ambiente phpMyAdmin su Altervista necessaria attenzione: *Libri* causa errore se tabella è *libri* (case sensitive)

Interessanti esempi scaricabili

Login con AJAX e md5() (aggiornato lavoro classe 5BI – Itis Alessandrini di Teramo 2015/16)

La tecnologia AJAX è qui utilizzata per realizzare un sistema di login in cui viene evitato il refresh della pagina.

Gli strumenti utilizzati sono: la libreria jQuery lato client, il linguaggio Php lato server e il DB MySql per la gestione della tabella degli utenti che effettuano il login.



La tabella "utenti" ha una struttura molto semplice

- id - int(5), primary key, auto increment
- nomeutente - varchar(16)
- password - varchar(32)

Il campo "password", che ha una lunghezza di 23 caratteri, contiene la password utente crittografata tramite la funzione php md5().

Scarica la soluzione
Pagina Downloads del prof. Mauro De Berardis
Prova la soluzione

Select a cascata con MySql e Php (aggiornato lavoro classe 5BI – Itis Alessandrini di Teramo 2014)

Il problema affrontato è quello delle cosidette "select a cascata" o "select concatenate" utilizzate per gestire, in questo caso, le tabelle di province e comuni: selezionando una provincia nella prima select, la seconda select viene popolata con i comuni appartenenti alla provincia selezionata.

Il problema viene risolto in "modo tradizionale", utilizzando Php e pochissimo codice JavaScript: la pagina viene ricaricata e processata da Php ogni volta che si effettua una selezione della provincia e una selezione del comune.

In alternativa si potrebbe utilizzare AJAX: in questo caso la pagina verrebbe aggiornata senza essere ricaricata.

La soluzione fa riferimento a due tabelle MySql "province" e "comuni" molto datate e con nomi di comuni non più esistenti. Gli studenti che vogliono approfondire l'argomento possono scaricare tabelle MySql più aggiornate ed adattare facilmente il codice.

Scarica la soluzione
Pagina Downloads del prof. Mauro De Berardis
Prova la soluzione → *Scarica il codice*