Progettazione Web
Applicazioni client-server

Sviluppo di un'applicazione Web integrando:

- HTML
- MySQL
- PHP
- Apache

Applicativi utilizzabili:

- gestione database: JDER, MySQL Workbench, HeidiSQL (Win), DBeaver (Linux), phpMyAdmin
- creazione pagine HTML e script PHP: Notepad++ (Win), gedit (Linux)

Destinatari e competenze

- Il progetto sarà sviluppato dagli studenti della classe quinta di un Istituto Tecnico
- **Competenza**: sviluppo di applicazioni informatiche per reti locali o servizi a distanza (rif. LL.GG. D.P.R. 15 marzo 2010)
- **Conoscenza**: gestione di database in rete, istruzioni PHP per la connessione e gestione di database
- **Abilità**: implementazione database remoti e pagine Web dinamiche

Prerequisiti

Conoscenze

Metodologia di sviluppo di software e fasi di sviluppo di un progetto

Sistema informatico e sistema informativo aziendale

Data Base Management System (DBMS)

Progettazione di Data Base

Linguaggio SQL, nello specifico MySQL

Linguaggi e strumenti di implementazione per il Web (tag principali di HTML e cenni di CSS)

Reti di computer e reti di comunicazione

Linguaggi di script lato server: PHP

Abilità

Esprimere procedimenti risolutivi attraverso algoritmi

Implementare algoritmi con diversi stili di programmazione e idonei strumenti software

Produrre la documentazione relativa alle fasi di progetto

Progettare e realizzare basi di dati in relazione alle esigenze aziendali

Progettare e realizzare pagine Web statiche

Pubblicare su Internet pagine Web statiche

Il progetto

Una palestra vuole informatizzare la gestione dei corsi offerti alla propria clientela commissionando la progettazione di un database e di un sito dinamico per accedere alle varie informazioni.

I vincoli:

- gli istruttori di un corso devono essere specializzati
- ogni corso ha un numero massimo di partecipanti
- corsi offerti in diversi orari e diversi giorni
- ogni cliente deve avere il certificato medico

Visualizzazioni e possibili evoluzioni

Le visualizzazioni richieste:

- elenco degli iscritti ad ogni corso
- numero dei posti liberi in ciascun corso
- elenco dei corsi tenuti da ogni istruttore
- elenco dei clienti con certificato medico scaduto

Possibili evoluzioni:

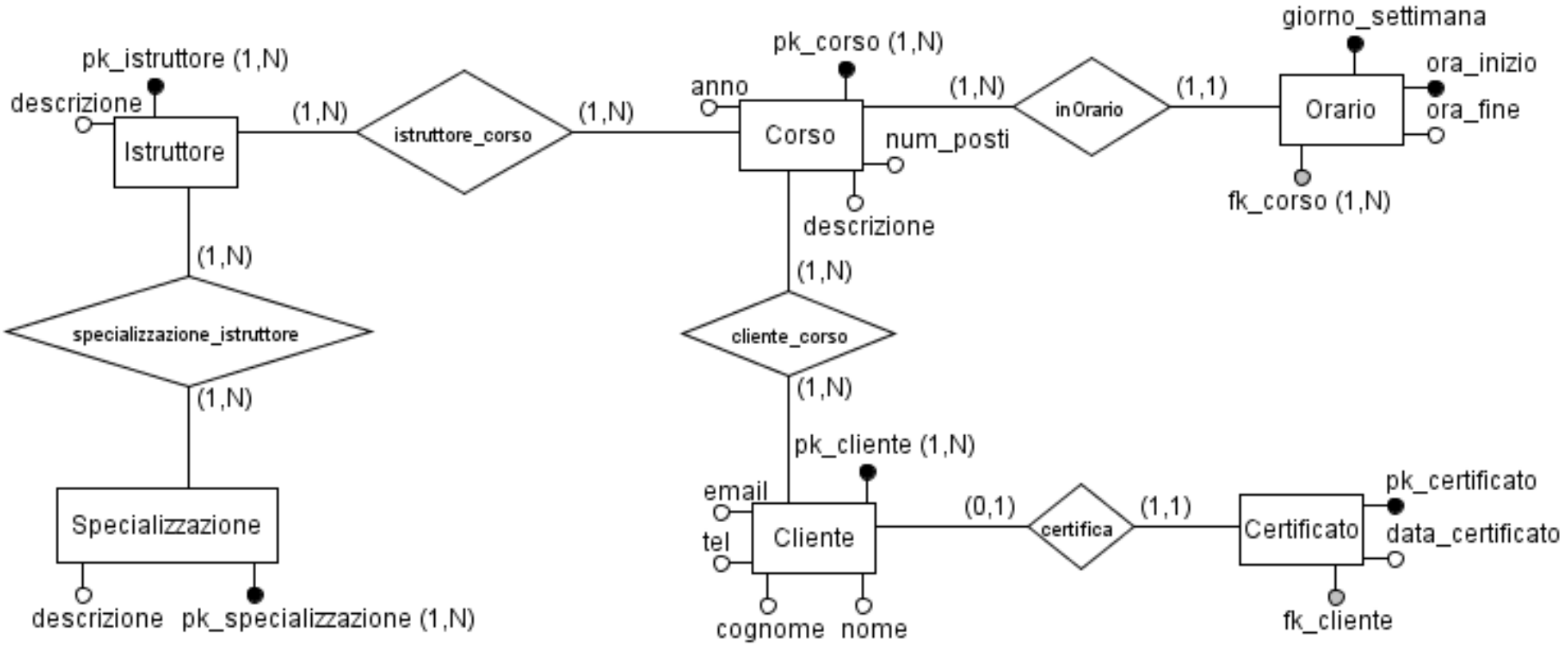
- inserimento di nuovi clienti
- inserimento di nuovi istruttori con le relative specializzazioni
- aggiunta di nuovi corsi
- visualizzazione della tabella settimanale dei corsi
- utilizzo dei fogli di stile

Fasi di sviluppo del progetto

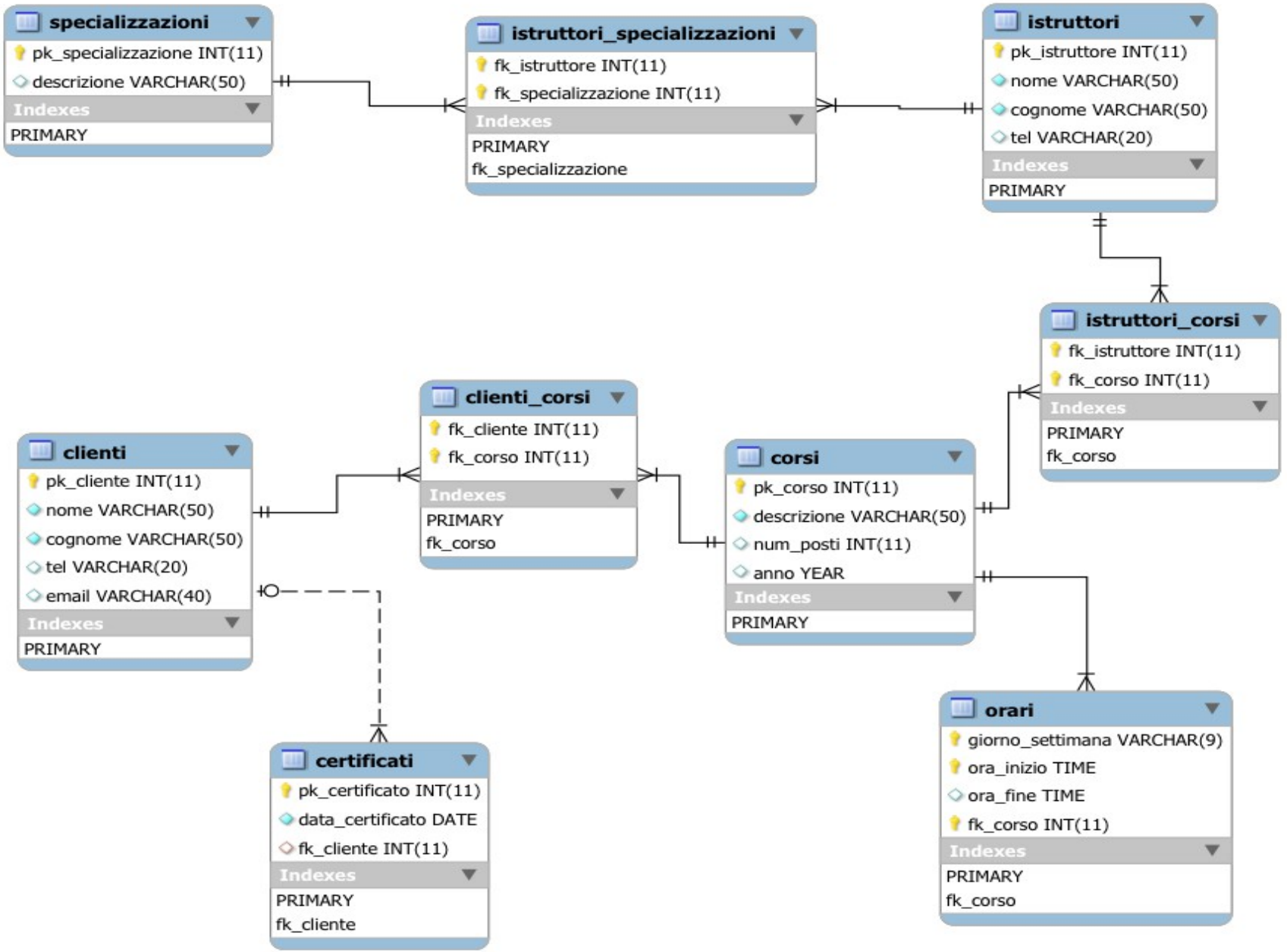
Il progetto sarà sviluppato per passi incrementali:

- analisi del caso e sviluppo del database
- progettazione e sviluppo delle pagine HTML
- progettazione e sviluppo degli script PHP
  - 1ª fase: script individuali per ogni query
  - 2ª fase: organizzazione integrata degli script parametrizzando la richiesta di visualizzazione
  - 3ª fase: ottimizzazione del codice, inserimento nuovi record in tabelle correlate, separazione codice da stile, esternalizzazione dello stile

Schema E-R del database



Schema logico del database



DDL del database

creiamo il database ...

```
--Creazione del database
create database db_palestra;
use db_palestra;
```

… e le tabelle, ad esempio ...

la tabella corsi, ...

```
--Creazione della tabella clienti_corsi
create table clienti_corsi(fk_cliente int, fk_corso int, primary key(fk_cliente, fk_corso),
foreign key(fk_cliente) references clienti(pk_cliente), foreign key(fk_corso) references
corsi(pk_corso))engine=innodb;
```

… la tabella clienti ...

```
--Creazione della tabella clienti
create table clienti(pk_cliente int auto_increment, nome varchar(50) not null, cognome
varchar(50) not null, tel varchar(20), email varchar(40), primary key(pk_cliente))engine=
innodb;
```

… e la tabella che le pone in relazione, clienti-corsi

```
--Creazione della tabella clienti_corsi
create table clienti_corsi(fk_cliente int, fk_corso int, primary key(fk_cliente, fk_corso),
foreign key(fk_cliente) references clienti(pk_cliente), foreign key(fk_corso) references
corsi(pk_corso))engine=innodb;
```

DML del database – Popoliamo il database

ad esempio, la tabella corsi, ...

```
INSERT INTO `corsi` (`pk_corso`, `descrizione`, `num_posti`, `anno`) VALUES
    (1, 'aerobica', 5, '2013'),
    (2, 'pesi', 6, '2013'),
    (3, 'step', 4, '2013'),
    (4, 'yogaA', 6, '2013'),
    (5, 'yogaB', 5, '2013');
```

la tabella clienti ...

```
INSERT INTO `clienti` (`pk_cliente`, `nome`, `cognome`, `tel`, `email`) VALUES
    (1, 'Claudio', 'Borgogno', '333456789', 'claudio.borgogno@gmail.com'),
    (2, 'Flaviano', 'Monge', '334543213', 'flaviano.monge@libero.it'),
    (3, 'Maria Grazia', 'Maffucci', '324987098', 'mgm.mgm@gmail.com'),
    (4, 'Paola', 'Longobardi', '333432123', 'p_longobardi@yahoo.it'),
    (5, 'Marco', 'Bracco', '324987456', 'marco.bracco@hotmail.com'),
    (6, 'Jessica', 'Paschini', '344456456', 'jessica_p@libero.it'),
    (7, 'Nunzio', 'Ferrigno', '347898767', 'n_ferrigno@hotmail.com'),
    (8, 'Federico', 'Sarzotti', '324567234', 'sarzfed@yahoo.it');
```

DML del database – Popoliamo il database

… e la tabella che le mette in relazione, clienti_corsi

```
INSERT INTO `clienti_corsi` (`fk_cliente`, `fk_corso`) VALUES
(1, 1),
(3, 1),
(2, 2),
(6, 2),
(1, 3),
(4, 3),
(8, 3),
(3, 4),
(5, 4),
(8, 4),
(4, 5),
(6, 5),
(7, 5);
```

DML del database – Implementiamo le query

```
--Elenco degli iscritti ad ogni corso
select corsi.descrizione, clienti.cognome, clienti.nome from (corsi inner join clienti_corsi
on corsi.pk_corso=clienti_corsi.fk_corso) inner join clienti on clienti_corsi.fk_cliente=
clienti.pk_cliente order by corsi.descrizione asc, clienti.cognome asc;
--Numero dei posti liberi in ogni corso
select (corsi.num_posti - count(clienti_corsi.fk_cliente)) as 'posti_liberi', corsi.
descrizione, corsi.pk_corso from corsi inner join clienti_corsi on corsi.pk_corso=
clienti_corsi.fk_corso group by corsi.pk_corso;
--Elenco dei corsi tenuti da ogni istruttore
select istruttori.cognome, istruttori.nome, corsi.descrizione from (istruttori inner join
istruttori_corsi on istruttori.pk_istruttore=istruttori_corsi.fk_istruttore) inner join corsi
 on istruttori_corsi.fk_corso=corsi.pk_corso order by istruttori.cognome asc, corsi.
descrizione asc;
--Elenco dei clienti con certificato medico scaduto
select clienti.cognome, clienti.nome, certificati.data_certificato from (clienti inner join
certificati on clienti.pk_cliente=certificati.fk_cliente) where datediff(curdate(),
certificati.data_certificato)>365 order by certificati.data_certificato asc, clienti.cognome
asc;
```

Sviluppo del progetto – 1ª versione

index.html richiama **scelta.php** tramite il form, e gli passa come parametro il valore assunto dal radio button **scelta**

scelta.php assegna il valore del radio button alla variabile **$scelta** e in base al valore assunto da questa richiama la funzione necessaria per la gestione del caso

funzioni.php è lo script che contiene tutte la funzioni che gestiscono i 4 casi. Ogni singola funzione si occupa di effettuare la connessione al server MySQL, la connessione al database, l'interrogazione e la visualizzazione del risultato.
VANTAGGI: gli studenti si concentrano sulla risoluzione di un problema alla volta.
SVANTAGGI: ripetizione di parti comuni di codice nelle 4 funzioni.

function iscritti()
iscritti ad ogni corso

function liberi()
numero di posti liberi in ogni corso

function corsi()
corsi tenuti da ogni istruttore

function certificati()
clienti con certificato medico scaduto

La pagina HTML – index.html

Body&Fitness

Il modo migliore per tenersi in forma

Via Roma, 14 - 10134 Torino
Tel. 3758894523
e-mail: bodyefitness@esempio.it

Scegliere un'operazione:

- Elenco degli iscritti a ogni corso
- Numero di posti liberi in ciascun corso
- Elenco dei corsi tenuto da un istruttore
- Elenco dei clienti con certificato medico scaduto

Esegui Cancella

Il codice HTML – index.html

Crea il form e permette di passare allo script PHP i dati inseriti

Specifica la modalità di passaggio dei parametri allo script PHP

Crea il collegamento tra lato client (HTML) e lato server specificando lo script che riceve e gestisce i dati

```
<form name="feedback" method="post" action="PHPscript/scelta.php">
    <fieldset>
     <legend>Scegliere un'operazione:</legend>
     <input type="radio" name="scelta" value="iscritti"/>Elenco degli iscritti a ogni corso
     <br/>
     <input type="radio" name="scelta" value="liberi"/>Numero di posti liberi in ciascun
     corso<br/>
     <input type="radio" name="scelta" value="corsi_istruttori"/>Elenco dei corsi tenuto da
     un istruttore<br/>
     <input type="radio" name="scelta" value="certificato_scaduto"/>Elenco dei clienti con
     certificato medico scaduto<br/>
    </fieldset>
    <br/>
    <input type="submit" value="Esegui">
    <input type="reset" value="Cancella">
</form>
```

Passaggio dei parametri dal form

Il passaggio dei parametri dal form può avvenire secondo due modalità, GET e POST:

- GET prevede il passaggio dei parametri in una QUERY STRING, ovvero una stringa accodata all'URL che contiene i parametri. E' visibile dalla barra del browser. Lunghezza massima query string 256 caratteri.

- POST i parametri vengono passati direttamente tramite protocollo HTTP, non sono visibili.

Sono metodi equivalenti, ma il GET non viene usato in caso di password o in caso di molti parametri o passaggio di stringhe molto lunghe, in questi casi è consigliabile l'uso di POST.

Ogni metodo viene comunque trattato in modo diverso dal lato server

Gestione dei parametri in PHP

PHP può accedere ai parametri in tre modi, utilizzando tre diversi array globali, di tipo associativo:

- $_GET[nomepar] usato per il metodo GET
- $_POST[nomepar] usato per il metodo POST
- $_REQUEST[nomepar] è l'array globale delle richieste ed è utilizzabile per entrambi i metodi GET e POST, rendendo lo script PHP indipendente dal metodo usato nel form HTML

Dal form allo script - scelta.php (1ª versione)

```php
<?php
    require ("funzioni.php");
    $scelta=$_POST['scelta'];
    switch ($scelta) {
        case "iscritti":
            iscritti();
            break;
        case "liberi":
            liberi();
            break;
        case "corsi_istruttori":
            corsi();
            break;
        case "certificato_scaduto":
            certificati();
            break;
    }
?>
```

Dal form verrà passato il valore assunto dal radio button **scelta** che risulterà selezionato.

```
<input type="radio" name="scelta" value="iscritti"/>Elenco degli i
<br/>
<input type="radio" name="scelta" value="liberi"/>Numero di posti
corso<br/>
<input type="radio" name="scelta" value="corsi_istruttori"/>Elenco
un istruttore<br/>
<input type="radio" name="scelta" value="certificato_scaduto"/>Ele
```

Lo script scelta.php – Richiamo funzioni (1ª versione)

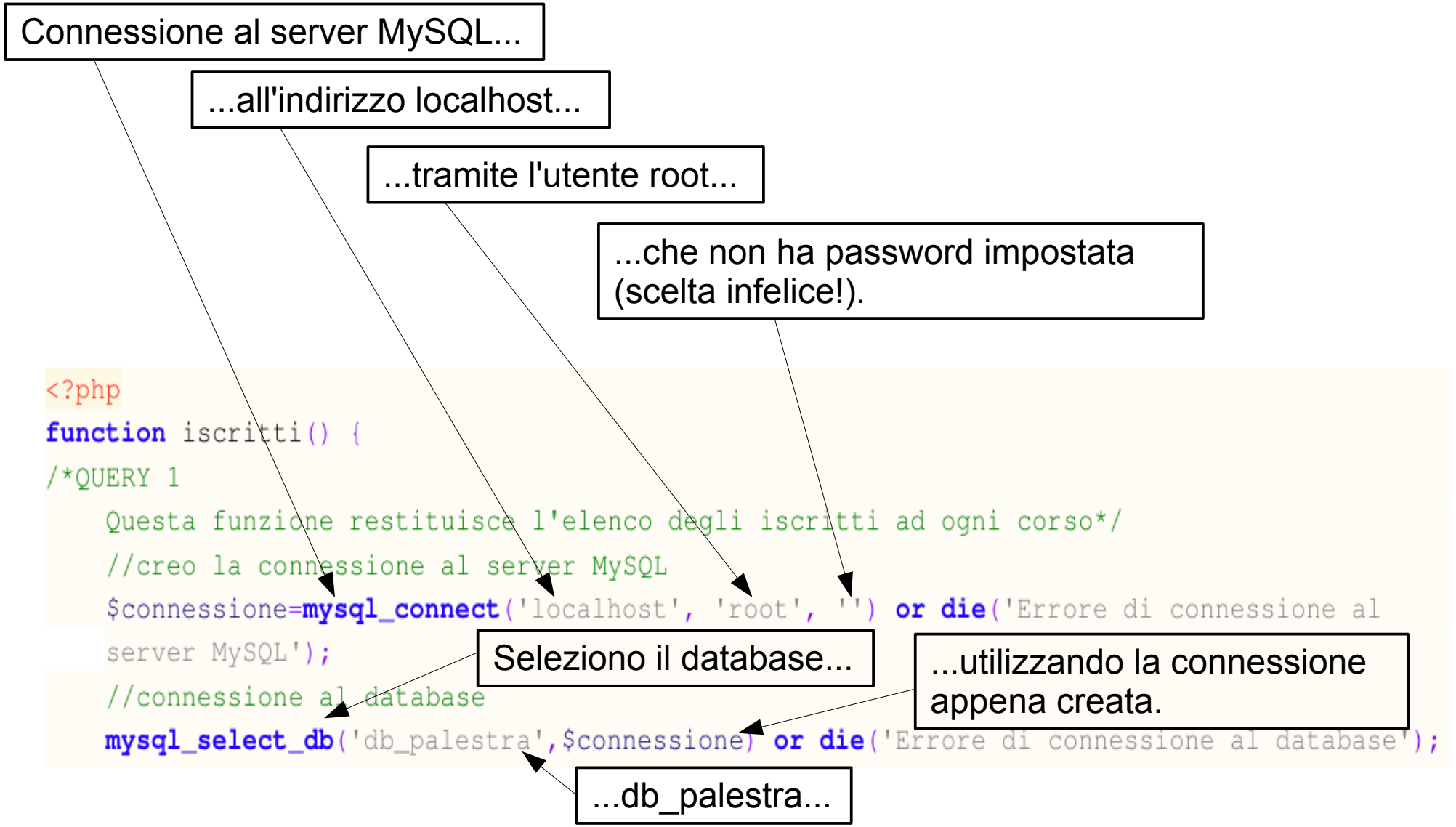
```php
<?php
    require ("funzioni.php");
    $scelta=$_POST['scelta'];
    switch ($scelta) {
        case "iscritti":
            iscritti();
            break;
        case "liberi":
            liberi();
            break;
        case "corsi_istruttori":
            corsi();
            break;
        case "certificato_scaduto":
            certificati();
            break;
    }
?>
```

Gli script di gestione sono memorizzati in **funzioni.php**

Lo script **scelta.php** richiamerà la funzione di gestione corrispondente in base al valore che assumerà la variabile **$scelta**.

Lo script funzioni.php – Connessione (1ª versione)

Ogni singola funzione gestisce indipendentemente la connessione al server MySQL, la connessione al database, la relativa interrogazione e la visualizzazione del risultato. Vediamo un esempio di connessione:




```php
<?php
function iscritti() {
/*QUERY 1
    Questa funzione restituisce l'elenco degli iscritti ad ogni corso*/
    //creo la connessione al server MySQL
    $connessione=mysql_connect('localhost', 'root', '') or die('Errore di connessione al
    server MySQL');
    //connessione al database
    mysql_select_db('db_palestra',$connessione) or die('Errore di connessione al database');
```

Lo script funzioni.php – Interrogazione (1ª versione)

Per effettuare l'interrogazione utilizziamo, per comodità, una variabile che conterrà la stringa della query, esattamente come fu scritta nel DML visto in precedenza:

Imposto la variabile **$temp_str** con la query da eseguire.

Eseguo la query...

…memorizzata nella veriabile **$temp_str**...

…utilizzando la connessione creata in precedenza

```
//imposto la stringa ed eseguo le query
$temp_str="select corsi.descrizione, clienti.cognome, clienti.nome from (corsi inner
join clienti_corsi on corsi.pk_corso=clienti_corsi.fk_corso) inner join clienti on
clienti_corsi.fk_cliente=clienti.pk_cliente order by corsi.descrizione asc,
clienti.cognome asc";
$iscritti=mysql_query($temp_str, $connessione) or die('Errore nella query per
visualizzare gli iscritti ad ogni corso');
```

… e memorizzo il risultato della query nella variabile **$iscritti**.

Lo script funzioni.php – Visualizzazione (1ª versione)

```
    $riga = mysql_fetch_assoc($iscritti);
    $corso= "";
    while($riga != NULL) {
        if($corso != $riga['descrizione']) {
            if($corso != "") {
                echo "</table><br/>";
            }
            $corso = $riga['descrizione'];
            echo "<h4>$corso</h4>";
            echo <<< ISCRITTI_DUE
            <table border="2">
            <tr align='center'>
                <td><b>Cognome</b></td>
                <td><b>Nome</b></td>
            </tr>
ISCRITTI_DUE;
        }
            echo <<< ISCRITTI_TRE
            <tr align='center'>
                <td>{$riga['cognome']}</td>
                <td>{$riga['nome']}</td>
            </tr>
ISCRITTI_TRE;
        $riga = mysql_fetch_assoc($iscritti);
    }
```

Il risultato dell'interrogazione, memorizzato nella variabile **$iscritti**, è un array di array, e le sigole righe vengono estratte con la funzione **mysql_fetch_assoc()** ...

...e assegnate, una per volta, alla variabile **$riga**.

La variabile **$riga** è un array associativo, cioè le singole posizioni possono essere indicizzate tramite il nome assegnato alla colonna, invece di usare un indice numerico.

Il codice HTML viene generato direttamente dallo script PHP.

Recupero risultato query

Abbiamo tre costrutti diversi per accedere alle righe (array) del risultato di una query, e “catturare” così la riga su cui lavorare:

- **mysql_fetch_row()**: utilizzato quando la riga catturata viene gestita come un array indicizzato numericamente, ad es. **$riga[3]**;
- **mysql_fetch_assoc()**: utilizzato quando la riga catturata viene gestita come un array associativo, indicizzato tramite un sostantivo, ad es. **$riga['descrizione']**;
- **mysql_fetch_array()**: utilizzato quando la riga catturata viene gestita sia come un array associativo che come un array indicizzato numericamente.

Lo script funzioni.php – Chiusura connessione (1ª versione)

Una volta concluse le operazioni di gestione del risultato della query, la connessione al server MySQL...

```
//chiudo la connessione al database
mysql_close($connessione);
```

...deve essere chiusa.

Sviluppo del progetto – 2ª versione

index.html richiama **scelta.php** tramite il form, e gli passa come parametro il valore assunto dal radio button **scelta**.

scelta.php assegna il valore del radio button alla variabile **$scelta**, effettua la conessione al server MySQL, la connessione al database e, in base al valore assunto dalla variabile **$scelta** esegue la query corrispondente. La funzine di gestione richiamata è unica e il suo comportamento dipenderà dai valori dei parametri passati.

funzioni.php è lo script che contiene la **funzione operazione($s,$q)** che gestisce la visualizzazione del risultato della query. Il parametro **$s** contiene la scelta effettuata nel form, il parametro **$q** contiene il risultato della query.
E' il primo tentativo di ottimizzazione del codice.

Lo script scelta.php - Differenze (2ª versione)

La connessione al server MySQL e al database viene effettuata una sola volta all'interno dello script **scelta.php**.

Connessione al server MySQL...

```php
<?php
    require ("funzioni.php");
    $scelta=$_POST['scelta'];
    //creo la connessione al server MySQL
    $connessione=mysql_connect('localhost', 'root', '') or die('Errore di connessione al
    server MySQL');
    //connessione al database
    mysql_select_db('db_palestra',$connessione) or die('Errore di connessione al database');
```

...e selezione del database db_palestra.

Lo script scelta.php - Differenze (2ª versione)

L'impostazione della variabile **$temp_str**, con la query da eseguire, verrà effettuata direttamente nello script **scelta.php**.

...in base alla scelta effettuata.

Imposto la variabile **$temp_str** con la query da eseguire...

```
//imposto la stringa ed eseguo le query
switch ($scelta) {
    case "iscritti":
    //Elenco degli iscritti ad ogni corso
        $temp_str="select corsi.descrizione, clienti.cognome, clienti.nome from (corsi
        inner join clienti_corsi on corsi.pk_corso=clienti_corsi.fk_corso) inner join
        clienti on clienti_corsi.fk_cliente=clienti.pk_cliente order by
        corsi.descrizione asc, clienti.cognome asc";
        $errore='Errore nella query per visualizzare gli iscritti ad ogni corso';
        break;
    case "liberi":
    //Elenco dei posti liberi in ogni corso
        $temp_str="select (corsi.num_posti - count(clienti_corsi.fk_cliente)) as
```

Lo script scelta.php - Differenze (2ª versione)

L'esecuzione della query viene effettuata all'interno dello script **scelta.php** e il risultato sarà passato come argomento alla funzione di gestione della pagina da restituire all'utente. Infine verrà chiusa la connessione al server MySQL.

Esecuzione della query, ...

```
    $query=mysql_query($temp_str, $connessione) or die($errore);
    operazione($scelta,$query);
    mysql_close($connessione);
    echo "<a href='../index.html'>Torna alla home page</a></body></html>";
?>
```

...richiamo della funzione **operazione($scelta, $query)** alla quale vengono passati come parametri la scelta effettuata dall'utente e il risultato della query...

...e chiusura della connessione al server MySQL una volta concluse tutte le operazioni.

Lo script funzioni.php – Differenze (2ª versione)

Nello script **funzioni.php** verrà gestita, tramite la funzione **operazione($s,$q)**, solo la generazione della pagina HTML da restituire all'utente, in base alla scelta effettuata nel form.

La funzione **operazione($s,$q)**...

...in base alla scelta fatta dall'utente.

...popolata con il risultato della query, passatagli come parametro, ...

... si occupa esclusivamente di generare la pagina HTML...

```
<?php
function operazione($s,$q) {
    switch ($s) {
        case "iscritti":
        //Elenco degli iscritti ad ogni corso
            echo <<< ISCRITTI_UNO
            <html><head><title>Elenco degli iscritti ad ogni corso
            </title></head><body bgcolor="#FFFFCC">
            <h3>Elenco degli iscritti ad ogni corso</h3>
ISCRITTI_UNO;
            $riga = mysql_fetch_assoc($q);
            $corso= "";
            while($riga != NULL) {
                if($corso != $riga['descrizione']) {
                    if($corso != "") {
                        echo "</table><br/>";
                    }
```

Andiamo a vedere il codice

- index.html
- scelta.php e funzioni.php (1ª versione)
- scelta.php e funzioni.php (2ª versione)