Come realizzare delle select concatenate con PHP e jQuery?

In questo articolo tratteremo del **concatenamento di select** detto anche *select a cascata*.

Si tratta di un procedura che possiamo trovare spesso nel web e che consiste nel dare all'utente la possibilità di trovare un dato specifico attraverso alcuni passaggi: in pratica scegliendo una categoria nella prima select, la seconda verrà popolata con i dati relativi alla categoria scelta.

Si pensi ad esempio ad una prima select dove è possibile scegliere tra moto e automobili; una volta selezionata la categoria *automobili* si dovrà popolare la seconda select con le marche disponibili di automobili. Selezionando infine una marca, la terza select verrà popolata con i modelli disponibili per questa marca. In questo articolo vedremo un esempio molto semplice ma che fornirà le basi per comprendere la procedura da attuare anche in caso di cascate molto più complesse.

Quello che realizzeremo è visionabile in questa pagina di esempio. Premetto che l'argomento è piuttosto avanzato e presuppone delle conoscenze medio-avanzate di PHP e di MySql.

Come puoi vedere, l'esempio è molto minimalista. Una prima select che richiede la categoria, nella quale figurano: colori, fiori e attrezzi. Ed una seconda select, nella quale una volta selezionata la categoria, troveremo un elenco di voci riguardanti quella categoria.

Siccome nel caso reale i dati saranno molti di più e molto più complessi, immaginiamo ad esempio la classica cascata regione/provincia/comune, utilizzeremo un database MySql. A proposito di questa cascata: è così classica e richiesta, che abbiamo deciso di metterla a disposizione come risorsa gratuita. Il file da scaricare contiene l'esempio già pronto (come questo) e, molto utile, il database con le tre tabelle (regioni, province e comuni). Ma torniamo al nostro tutorial ed iniziamo a creare la struttura del database.

Creare la struttura del database

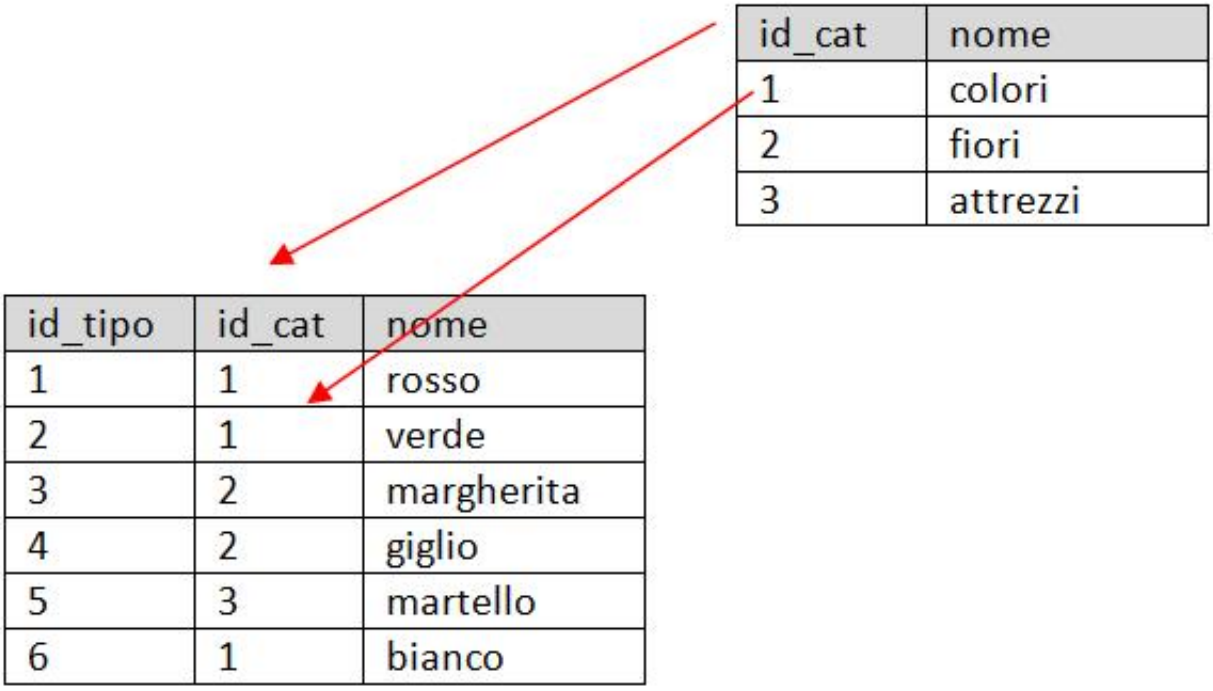
Creeremo dunque una tabella denominata *categorie* con i campi

- id_cat
- nome

Ed un'altra tabella denominata *tipo* con i seguenti campi

- id_tipo
- id_cat
- nome

In questo modo possiamo stabilire la corretta relazione tra le tabelle



id_cat	nome
1	colori
2	fiori
3	attrezzi

id_tipo	id_cat	nome
1	1	rosso
2	1	verde
3	2	margherita
4	2	giglio
5	3	martello
6	1	bianco

Crea dunque il database *selectExample* ed esegui la query che ho preparato per facilitarti il compito. Oltre a creare le tabelle necessarie, inserisce anche i dati che ci serviranno per l'esercizio.

Preparazione della pagina principale

Passiamo ora a preparare il file principale ovvero *select.php*

Come vedi ho incluso jQuery. Infatti andremo a popolare la seconda select grazie ad una chiamata ajax.

Ho dichiarato un form contenente la prima *select* senza nessuna *option* e la seconda select contenente unicamente un'*option*, ovvero "*scegli…*". Infine abbiamo un un elemento con id *result* nel quale alla fine inseriremo la scelta che è stata effettuata.

Creare il file con i parametri di connessione

Creaimo ora il file *db_config.php* che conterrà i parametri di connessione al database.

Naturalmente dovrai assegnare i parametri del tuo database.

Il file php per la gestione delle richieste

Iniziamo a questo punto a creare la classe PHP che avrà il compito di gestire l'elaborazione delle informazioni e di restituire i risultati. Il file si chiamerà *select.class.php.*

Iniziamo con il dichiarare la classe, in seguito la proprietà *$conn* che conterrà la risorsa di connessione al database, ed infine il metodo costruttore, che per il momento lasciamo vuoto.

Ora scriviamo il metodo per la connessione al database.

Come vedi, dopo avere incluso i parametri procedo con la connessione e quindi con la selezione del database.

Ora procediamo con la creazione del metodo ShowCategory() che sarà responsabile di creare le ***option*** **della prima *select***. Questo metodo verrà eseguito direttamente al caricamento della pagina, vedremo in seguito come.

Come puoi vedere eseguo una query che mi seleziona **tutte le righe** della tabella categorie. In seguito inserisco nella variabile *$category* la prima *option*, contenente un messaggio che invita l'utente ad effettuare una scelta. Infine aggiungo alla variabile *$category* tante *option* quante le voci presenti nella tabella, avendo cura di indicare come *value* l'id della categoria e come contenuto del tag *option* il nome stesso della categoria.

Ora passiamo a scrivere l'ultimo metodo che si occuperà di intercettare il valore passato dalla prima *select* (l'id della categoria) e di popolare la seconda in base al valore passato dalla prima

In questo caso tramite la query andremo a selezionare le righe della tabella *tipo* che hanno come *id_cat* il valore passato tramite POST (questo valore sarà passato con una chiamata ajax, ma non l'abbiamo ancora scritta, poi vedremo).

Il seguito è del tutto simile al metodo precedente.

Ora, visto che qualunque operazione svolga questa classe è necessaria la connessione al

database, possiamo invocare il metodo DbConnect() **direttamente nel metodo costruttore**.

Inoltre possiamo, per comodità, instanziare la classe direttamente in questo file.

Ed ecco dunque la classe completa:

Ora torniamo a lavorare sul file principale (*select.php*)

Al momento che questo file viene caricato dovrà essere popolata la prima *select*. Lo faremo inserendovi il metodo ShowCategory() in questo modo:

Come vedi, includiamo la classe già istanziata nell'oggetto $opt. In seguito, tra i tag *select*, stampiamo il risultato del metodo ShowCategory(), che saranno le *option* correttamente popolate con i valori presi dal database.

Puoi provare lo script e constatare che la prima *select* si popola correttamente.

Implementare le funzionalità ajax

Ora, quello che dobbiamo fare tramite jQuery è intercettare quanto verrà scelto in questa select ed inviare il dato (l'id della categoria) al metodo ShowType() che ci restituirà le *option* da inserire nella seconda *select*. Il tutto naturalmente con una chiamata ajax. Il codice (come sempre con jQuery è molto semplice) è il seguente (che va inserito nella funzione $(document).ready).

Cosa significa? Al verificarsi dell'evento *change* nella *select* con id *categorie*, esegui questa funzione che:

- Valorizza la variabile *id* con l'attributo *value* dell'opzione selezionata.
- Invia tramite POST la variabile *id* al file *select_type.php* (che non abbiamo ancora scritto, ma lo faremo subito).
- Quanto restituito da questa chiamata inseriscilo nella *select* con id *tipo*.

Creare un file di servizio

Ovviamente il file *select_type.php* sarà un file di servizio che semplicemente invocherà il metodo ShowType().

A questo punto puoi provare lo script. Ora selezionando una categoria, la seconda *select* si popolerà correttamente.

Migliorare l'usabilità

Adesso facciamo una piccola aggiunta. Ricordate la questione sull'usabilità esposta nel precedente articolo su ajax? Bene, credo che la seconda select vada disabilitata al caricamento della pagina. All'inizio della chiamata si potrebbe sostituire "scegli" con "attandere...". Una volta che la chiamata è stata eseguita, la select può essere riabilitata. Questo é corretto dal punto di vista dell'usabilità. Modifichiamo dunque il codice in questo modo:

Adesso non ci resta che stabilire **cosa succede quando si preme su *invia***. Nel nostro caso scriveremo la scelta effettuata nell'elemento *#result*. Dovremo anche tenere conto (usabilità) di cosa dovrebbe succedere se il bottone *invia* viene premuto **prima che siano state fatte tutte le scelte**.

Questo è il codice che in seguito spiegherò

Al verificarsi dell'evento *submit*, prendiamo il valore dell'attributo *value* delle due *select*. Se questi valori sono entrambi maggiori a zero (quindi è stata fatta una scelta nelle due select), prendiamo il contenuto della *select* con id *tipo* e lo stampiamo nell'elemento *#result*.

Se invece non sono state fatte le due scelte, nell'elemento *#result* stampiamo il messaggio di errore.

Ecco *select.php* completo con tutte le modifiche.

Qui puoi scaricare un esempio dello script applicato alle regioni, province e comuni, uno dei casi più comuni di utilizzo di select concatenate.

Conclusione

In questo articolo abbiamo visto un'applicazione di ajax molto utile e richiesta. Trattandosi di una funzionalità avanzata è chiaro che può risultare complessa a chi non ha una certa padronanza di php, ma d'altra parte **non si può fare più semplicemente di così**.

Bisogna comunque dire che, per una corretta applicazione dei criteri di accessibilità, dovremo prevedere una soluzione che permetta l'utilizzo di questo sistema, anche senza l'utilizzo di

javascript (e quindi di jQuery).

E tu cosa ne pensi? Hai trovato questo articolo troppo complesso? Utilizzerai questi principi nei tuoi siti?