

第 11 回 データベース接続 (UPDATE)

今回は新しいメソッドなどは登場しません。前回までの知識を用いて、データベースの更新処理を行いましょう。

■配布ファイル (フォーマット)

- ・ kadai11_1.php ※編集画面
- ・ kadai11_2.php ※DB への更新処理、エラー時の結果表示画面
(更新成功時は、kadai08_1.php の一覧画面へ遷移する)
- ・ kadai11_3.php ※削除確認画面
- ・ kadai11_4.php ※DB からレコードの削除処理、結果表示画面
(一覧・検索画面へ戻るボタン付き)

■既に配布済みのファイル (今回使用します)

- ・ def.php ※6 回目で配布したものを使用。各種定数が設定されている。
- ・ kadai08_1.php ※kadai11_1.php (更新)、kadai11_3.php (削除) に遷移するリ

ンクを追加します。

更新

★事前準備（ファイル名：kadai08_1.php）

一覧画面（kadai08_1.php）に各データの「編集」「削除」リンクを追加しましょう。

サーバーサイドスクリプト演習 1

データベース検索

新規登録

NAME	商品番号	商品名	カテゴリ	価格	編集	削除
	0901	エビマヨネーズ	ピザ	1600	編集	削除
	0902	アスパラベーコン	ピザ	1800	編集	削除
	2901	ビール	ドリンク	500	編集	削除
	2902	レモンチューハイ	ドリンク	350	編集	削除
	2903	ウイスキー	ドリンク	3400	編集	削除
	0903	プレーンピザ	ピザ	890	編集	削除
	9999	特価ドリンク	ドリンク	999	編集	削除
	1234	試作ドリンク	ドリンク	500	編集	削除

★仕様

部品	動作
編集リンク	<p>編集画面（kadai11_1.php）に遷移する。</p> <p>該当行の product_no は GET 形式で送信。</p> <p>番号の数字だけでなく、「product_no=該当行の商品番号」の形式で送る。</p>
削除リンク	<p>削除画面（kadai11_3.php）に遷移する。</p> <p>該当行の product_no は GET 形式で送信。</p> <p>番号の数字だけでなく、「product_no=該当行の商品番号」の形式で送る。</p>

実装できたら、リンクを押下時にそれぞれのページへ遷移する、かつ遷移先のページで GET データが「**product_no=該当行の商品番号**」の形式で送られているかを確認してください。

★課題 1 1 – 1 （ファイル名:kadai11_1.php）

編集用の画面を作成しましょう。編集データは DB から検索します。

★仕様

タイミング	動作
ページ表示時	GET で受け取った product_no をキーとし、DB から検索し、結果を画面表示する。 ※GET データがない場合は、一覧画面（kadai08_1.php）に遷移する。
「更新」ボタン押下時	更新処理（kadai11_2.php）に POST 形式で入力データを送信する。

★画面

サーバーサイドスクリプト演習 1

データベース更新

商品番号 ※商品番号は変更不可

9999

カテゴリ

ドリンク

価格

999

商品名

特価ドリンク

更新

戻る

★課題 1 1 - 2 第 1 段階（ファイル名：kadai11_2.php）

DB 更新処理を行きましょう。

① 入力画面より送られたデータを変数に格納する。

* 今回は \$postData 配列に格納しましょう。

② 入力値の trim 処理を行う。

* \$postData は配列なので、ループで trim 処理が可能ですね。

③ 入力値のチェックをし、エラーの場合、\$errMsg にエラーメッセージを追加していく。

入力値	エラー
商品名	空のとき
価格	<p>数値以外。</p> <p>* ただし、下記のように filter_input で INT フィルターを掛けたとき、 price の中が「12abc」であれば、 filter_input での戻り値、すなわち、\$postData["price"] に代入された値は どうなるでしょうか。</p> <p>ここは DB 接続の処理を作成する前に、echo など各自確認してください。</p> <p>ヒント：エラー条件がシンプルになります。</p> <p>【コード例】</p> <pre>\$postData["price"] = filter_input(INPUT_POST, "price", FILTER_VALIDATE_INT);</pre>

*以降④～⑥は入力値にエラーがなければ行う。 * * * * *

④ DB 接続処理

⑤ SQL 文の準備と実行

■注意

SQL 文の区切りの空白がなく、SQL エラーになることがよくあります。

SQL 文の連結の際に、空白を入れ忘れることが原因です。

(例：

```
SELECT * FROM OLDPRODUCTWHERE pname = :pnameAND category = :category)
```

DB 接続を行う前に、prepare でセットする予定の SQL 文だけを表示してみましょう。

また、その際は prepare や execute などは一応コメントアウトして行いましょう。

⑥ DB 切断処理

⑦ 更新成功の場合、一覧画面 (kadai08_1.php) に遷移しましょう。

入力値に不備がある or 更新失敗した場合、エラーメッセージを表示しましょう。

* エラーメッセージ画面は、後述の「★画面例」参照。

*以降⑧はエラーがあった場合のみ

⑧ 「戻る」リンク押下時、kadai11_1.php に遷移。

★仕様

部品	動作
戻るリンク	編集画面 (kadai11_1.php) に遷移する。 ★kadai11_1.php の仕様を確認ください。 編集データを表示するには、何が必要ですか。

★画面例 (エラーメッセージ表示)

データベース更新結果

商品名を入力してください。
価格を半角数字で入力してください。

戻る

削除

★課題 1 1 - 3 (ファイル名:kadai11_3.php)

確認用の画面を作成しましょう。編集データは DB から検索します。

★仕様

タイミング	動作
ページ表示時	GET で受け取った product_no をキーとし、DB から検索し、結果を画面表示する。 ※GET データがない場合は、一覧画面 (kadai08_1.php) に遷移する。
「削除」 ボタン押下時	削除処理 (kadai11_3.php) に POST 形式で商品番号データを送信する。* 削除レコードは主キーのみで判別可能なので、全データを送る必要はない。
「戻る」 ボタン押下時	一覧・検索画面 (kadai08_1.php) に遷移する。

★画面

サーバーサイドスクリプト演習 1

データベース削除

下記のデータを削除します。

商品番号

9999

カテゴリ

ピザ

価格

1200

商品名

プレーンピザ

削除

戻る

★課題 1 1 - 4 第 1 段階 (ファイル名 : kadai11_4.php)

DB レコード削除処理を行いましょう。

⑨ POST 形式でデータが送られてきていなければ、kadai08_1.php に戻る。

⑩ 入力画面より送られた商品番号データを変数に格納する。

エラーメッセージ格納用の変数も用意しておく。

⑪ DB 接続処理

DB の設定で、SQL エラーの詳細を Throw するようにしましょう。

ヒント：setAttribute メソッドを使用。

⑫ SQL 文の準備と実行 *今回は削除処理です。

DB 接続を行う前に、prepare でセットする予定の SQL 文だけを表示してみましょう。

また、その際は prepare や execute などは一応コメントアウトして行いましょう。

実行してエラーがなければ、commit しましょう。

例外 (PDOException) は catch して、\$errMsg にエラーメッセージを入れましょう。

⑬ DB 切断処理

⑭ エラーメッセージがある場合は、エラーメッセージを、

エラーがない場合は、「データを削除しました」の文言を、表示しましょう。

⑮ 「一覧・検索画面へ戻る」リンク押下時、kadai08_1.php に遷移。

★仕様

部品	動作
一覧・検索画面へ戻る	一覧・検索画面 (kadai08_1.php) に遷移する。

★画面例 (データ削除成功)

データベース削除結果

データを削除しました。

[一覧・検索画面へ戻る](#)

★画面例（データ削除失敗）

データベース削除結果

DBエラー : SQLSTATE[42S22]: Column not found: 1054 Unknown column
'PRODUCT_NO' in 'where clause'

[一覧・検索画面へ戻る](#)

Extra 課題

一覧画面 → 新規登録・更新・削除の設計

学習用なので、下記は簡易動作としています。

- － 入力値チェック
- － 画面遷移（結果を表示するのみ）
- － DB 設計

***実務では、下記のようにすることが多いです。課題完成後は、下記「設計を改修」してみてください。**

【入力チェック】

- ・ フロント側
- ・ サーバーサイド側
- ・ DB の設定

【画面遷移】

ユーザーが使い易い画面遷移にすること！

以上です。