
MySQL ja PHP - Perusteet

Tämän luvun esimerkeissä käytetään edellisessä luvussa *MySQL-esimerkkitietokannan luominen* luotua tietokantaa ja sen `customer`-taulua sisältöineen. Esimerkeissä käytetty `N1234` tarkoittaa JAMKin labraverkon käyttäjätunnustasi tai tilanteesta riippuen samannimistä tietokantaasi. Korvaa `N1234` omalla labraverkon tunnuksellasi.

Tietokantakyselyn tulos web-selaimelle

Kolmikerrosmallin mukaan web-palvelin PHP-ympäristöineen toimii palvelimena web-selaimelle ja asiakkaana tietokantapalvelimelle. Kun tietokantakyselyn tulos halutaan esittää web-selaimelle PHP-skriptin avulla, toistuvat seuraavat vaiheet eri muodoissaan:

- Web-selain lähettää pyynnön palvelimelle
- Käynnistetään PHP-skripti
- Avataan yhteys tietokantaan:

```
$db = new  
PDO('mysql:host=localhost;dbname=database_name;charset=utf8',  
    'db_username', 'db_user_password');
```
- Suoritetaan SQL-kysely käyttäen avattua yhteyttä:

```
$stmt = $db->query('SELECT * FROM db_table');
```
- Otetaan kyselyn tulostulokset seuraava tietue:

```
$row = $stmt->fetch(PDO::FETCH_ASSOC)
```
- Tulostetaan tietueen kenttien arvot halutulla tavalla Web-selaimelle:

```
echo "{$row['field_x']}, {$row['field_y']}<br>";
```

Esimerkki 2010: `mysql01.php`

Ensimmäisessä esimerkissä SQL-kyselyn tuloksena palautuu kolmen kentän (`id`, `name` ja `birth_date`) arvot kaikista viidestä rivistä. Kyselyn suorittamiseksi käytetään PDO-objektin `query()`-metodia.

```

1  <?php
2  // mysql01.php
3  $db = new PDO('mysql:host=mysql.labranet.jamk.fi;dbname=N1234
4          'N1234', 'se_pitka_db_salasana');
5
6  $results = $db->query('SELECT * FROM customer');
7
8  while($row = $results->fetch(PDO::FETCH_ASSOC)) {
9      echo "{$row['id']}: {$row['name']} {$row['birth_date']}<br>
10 }
11 ?>

```

Ohjelma tulostaa

```

1: Guru Ken 1955-05-05
2: Alainen Kim 1966-06-06
3: Ainen Sani 1977-07-07
4: Vainio Vilja 1988-08-08
5: Tana Ruu 1999-09-09

```

Esimerkki 2011: db-init.php + mysql02.php

- Tässä esimerkissä tietokanta-asetukset tallennetaan omaan tiedostoonsa /home/N1234/db-config/db-init.php pois web-palvelimen suoralta saanta public_html-kansion alikansioista. Yhteen paikkaan tallennettu asetustiedostoa voi sitten käyttää usea PHP-ohjelma
- mysql02.php-ohjelma sisällyttää tietokanta-asetukset käytettäväkseen.
- mysql02.php-ohjelmassa SQL-kyselyn tuokset muotoillaan HTML-
tauluktoon.

/home/N1234/db-config/db-init.php

```

1  <?php
2  // /home/N1234/db-config/db-init.php
3
4  $db = new PDO('mysql:host=mysql.labranet.jamk.fi;dbname=N1234
5          'N1234', 'se_pitka_db_salasana');

```

mysql02.php

```

1  <?php
    // mysql02.php

```

```

2  require_once("/home/N1234/db-config/db-init.php");
3  $results = $db->query('SELECT * FROM customer');
4
5  echo "<table border='1'>";
6
7  while($row = $results->fetch(PDO::FETCH_ASSOC)) {
8      echo "<tr><td>{$row['id']}</td><td>{$row['name']}</td></tr>";
9  }
10
11 echo "</table>";
12
13 ?>
14

```

Ohjelma tulostaa

1	Guru Ken
2	Alainen Kim
3	Ainen Sani
4	Vainio Vilja
5	Tana Ruu

INSERT, UPDATE ja DELETE: db->exec()

- PDO-objektin `exec()` -metodilla voidaan suorittaa INSERT-, UPDATE- ja DELETE-lauseita. Tätä voidaan käyttää silloin kun kyselyn osaksi ei tuoda ulkopuolista dataa.
- Kyselyn (query) tulosjoukon rivien lukumäärän palauttaa kyselyjoukon `rowCount()` -metodi.
- Esimerkki on itsekomentoiva. Kokeile ja muuta tarvittaessa

Esimerkki 2012: mysql03.php

```

1  <?php
2  // mysql03.php
3  require_once("/home/N1234/db-config/db-init.php");
4
5  //-- -- -- -- --
6  // Lisätään yksi tietue
7  $sql = "INSERT INTO customer VALUES(6, 'Mieli Kaino', '2011-01-01')";
8  $affected_rows = $db->exec($sql);

```

```

 9  echo "<br>" . $affected_rows . " riviä lisättiin:<br>";
10
11  print_customers($db);
12
13  //-- -- -- -- --
14  // Päivitetään yksi tietue
15  $sql = "UPDATE customer SET birth_date = '2019-01-09' WHERE na
16  $affected_rows = $db->exec($sql);
17  echo "<br>" . $affected_rows . " riviä muutettiin:<br>";
18
19  print_customers($db);
20
21  //-- -- -- -- --
22  // Poistetaan yksi tietue
23  $sql = "DELETE FROM customer WHERE name = 'Mieli Kaino'";
24  $affected_rows = $db->exec($sql);
25  echo "<br>" . $affected_rows . " riviä poistettiin:<br>";
26
27  print_customers($db);
28
29  //-- -- -- -- --
30  // Tulostetaan customer-taulu HTML-taulukkona
31  function print_customers($db) {
32      $result = $db->query('SELECT * FROM customer');
33      $row_count = $result->rowCount();
34      echo "Näytetään " . $row_count . " riviä:<br>";
35
36      echo "<table border='1'>";
37      while($row = $result->fetch(PDO::FETCH_ASSOC)) {
38          echo "<tr><td>{$row['id']}</td><td>{$row['name']}</td><td>
39      }
40      echo "</table>";
41  }
42  ?>

```

Poikkeusten käsittely: try...catch

PDOssa on kolme tilaa virheiden käsittelyyn

- **PDO::ERRMODE_SILENT** => jokainen tulos katsottava, sitten yksityiskohdat metodilla `$db->errorInfo()`

- **PDO::ERRMODE_WARNING** => heittää PHP-varoitukset
- **PDO::ERRMODE_EXCEPTION** => heittää PDO-poikkeuksen; suositeltavin sovellusten tuotantoversioissa.

Virheidenkäsittelytila voidaan asettaa mm. PDO-objektin `setAttribute()` -metodilla seuraavan esimerkin tapaan.

Kokeile seuraavaa esimerkkiä jokaisella virheidenkäsittelytilalla (poistamalla kommentit halutusta vaihtoehdosta) asettamalla samalla SQL-kysely virheelliseksi esim. taulun nimen osalta `SELECT * FROM XcustomersX`

Virheiden käsittelyn tila db-init.php-tiedostoon

Haluttu virheiden käsittelyn tila voidaan asettaa db-init.php-tiedostossa. Huomaa kommentoidut mahdollisuudet.

```
1  <?php
2  //          /home/N1234/db-config/db-init.php
3
4
5  $db = new PDO('mysql:host=mysql.labranet.jamk.fi;dbname=N1234
6              'N1234', 'se_pitka_db_salasana');
7
8
9  //$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_SILENT);
10 //$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_WARNING);
11 $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

Varsinainen poikkeusten käsittely: mysql04.php

```
1  <?php
2  // mysql04.php
3
4  try {
5      require_once("/home/N1234/db-config/db-init.php");
6      $result = $db->query('SELECT * FROM XcustomerX');
7  } catch(PDOException $ex) {
8      echo "ErrMsg to enduser!<hr>";
9      echo "CatchErrMsg: " . $ex->getMessage() . "<hr>";
10 }
```

```

11
12 while($row = $result->fetch(PDO::FETCH_ASSOC)) {
13     echo "{$row['id']}, {$row['name']}<br>";
14 }
15 ?>

```

Sarakkeiden läpikäynti ja pitkän SQL-kyselyn muotoilu

Esimerkissä havainnollistetaan kuinka

- tietokantakyselyn tulosjoukon kaikki sarakkeet voidaan käydä läpi muuttamatta ohjelmakoodia vaikka SQL-kyselyä muutettaisiin.
- halutut sarakeotsikot saadaan käyttöliittymään SQL-kyselystä `AS`-avainasanalla.
- kuinka pitkä SQL-kysely voidaan muotoilla siististi luettavaksi **HereDoc**-syntaksilla
- kuinka MySQL-aikaleima voidaan muotoilla haluttuun formaattiin `DATE_FORMAT`-funktiolla

Tutki, muuta ja kokeile ohjelmakoodia ymmärtääksesi sen toiminnan!

db-column-names.php

```

1  <?php
2  // db-column-names.php
3  require_once("/home/N1234/db-config/db-init.php");
4
5  $sql = <<<EOSql
6  SELECT id,
7      name as nimi,
8      DATE_FORMAT(created_at, "%d.%m.%Y %H:%i:%s") AS luotu
9  FROM customer
10 EOSql;
11
12 $results = $db->query($sql);
13
14 $printColumnNames = TRUE;
15 echo "<table border='1'>";
16 while($row = $results->fetch(PDO::FETCH_ASSOC)) {
17     if ($printColumnNames) {
18         echo "<tr>";
19         foreach ($row as $avain => $arvo) {
20             echo "<th>" . ucfirst($avain) . "</th>";
21         }

```

```

22         echo "</tr>";
23         $printColumnNames = FALSE;
24     }
25     echo "<tr>";
26     foreach ($row as $arvo) {
27         echo "<td>$arvo</td>";
28     }
29     echo "</tr>";
30 }
31 echo "</table>";
32
33 ?>

```

Ohjelman tulostus

Id	Nimi	Luotu
1	Guru Ken	22.09.2019 15:31:36
2	Alainen Kim	22.09.2019 15:31:36
3	Ainen Sani	22.09.2019 15:31:36
4	Vainio Vilja	22.09.2019 15:31:36
5	Tana Ruu	22.09.2019 15:31:36