# 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 tulosjoukosta 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.

```php
<?php
// mysql01.php
$db = new PDO('mysql:host=mysql.labranet.jamk.fi;dbname=N1234
               'N1234', 'se_pitka_db_salasana');

$results = $db->query('SELECT * FROM customer');

while($row = $results->fetch(PDO::FETCH_ASSOC)) {
    echo "{$row['id']}: {$row['name']} {$row['birth_date']}<b
}
?>
```

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-taulukkoon.

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

```php
<?php
// /home/N1234/db-config/db-init.php

$db = new PDO('mysql:host=mysql.labranet.jamk.fi;dbname=N1234
               'N1234', 'se_pitka_db_salasana');
```

### mysql02.php

```php
<?php
// mysql02.php
```

```php
require_once("/home/N1234/db-config/db-init.php");
$results = $db->query('SELECT * FROM customer');

echo "<table border='1'>";

while($row = $results->fetch(PDO::FETCH_ASSOC)) {
  echo "<tr><td>{$row['id']}</td><td>{$row['name']}</td></tr>
}

echo "</table>";

?>
```

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 itsekommentoiva. Kokeile ja muuta tarvittaessa

## Esimerkki 2012: mysql03.php

```php
<?php
// mysql03.php
require_once("/home/N1234/db-config/db-init.php");

//-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
// Lisätään yksi tietue
$sql ="INSERT INTO customer VALUES(6, 'Mieli Kaino', '2011-01
$affected_rows = $db->exec($sql);
```

```php
echo "<br>" . $affected_rows . " riviä lisättiin:<br>";

print_customers($db);

//-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
// Päivitetään yksi tietue
$sql ="UPDATE customer SET birth_date = '2019-01-09' WHERE nam
$affected_rows = $db->exec($sql);
echo "<br>" . $affected_rows . " riviä muutettiin:<br>";

print_customers($db);

//-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
// Poistetaan yksi tietue
$sql ="DELETE FROM customer WHERE name = 'Mieli Kaino'";
$affected_rows = $db->exec($sql);
echo "<br>" . $affected_rows . " riviä poistettiin:<br>";

print_customers($db);

//-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
// Tulostetaan customer-taulu HTML-taulukkona
function print_customers($db) {
  $result = $db->query('SELECT * FROM customer');
  $row_count = $result->rowCount();
  echo "Näytetään " . $row_count. " riviä:<br>";

  echo "<table border='1'>";
  while($row = $result->fetch(PDO::FETCH_ASSOC)) {
    echo "<tr><td>{$row['id']}</td><td>{$row['name']}</td><td
  }
  echo "</table>";
}
?>
```

## 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.

```php
<?php
//          /home/N1234/db-config/db-init.php


$db = new PDO('mysql:host=mysql.labranet.jamk.fi;dbname=N1234
               'N1234', 'se_pitka_db_salasana');


//$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_SILENT);
//$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_WARNING);
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

### Varsinainen poikkeusten käsittely: mysql04.php

```php
<?php
// mysql04.php

try {
  require_once("/home/N1234/db-config/db-init.php");
  $result = $db->query('SELECT * FROM XcustomerX');
} catch(PDOException $ex) {
  echo "ErrMsg to enduser!<hr>";
  echo "CatchErrMsg: " . $ex->getMessage() . "<hr>";
}
```

```php

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

## 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 ymmärttääksesi sen toiminnan!*

### db-column-names.php

```php
<?php
// db-column-names.php
require_once("/home/N1234/db-config/db-init.php");

$sql = <<<EOSql
SELECT id,
  name as nimi,
  DATE_FORMAT(created_at, "%d.%m.%Y %H:%i:%s") AS luotu
FROM customer
EOSql;

$results = $db->query($sql);

$printColumnNames = TRUE;
echo "<table border='1'>";
while($row = $results->fetch(PDO::FETCH_ASSOC)) {
    if ($printColumnNames) {
        echo "<tr>";
        foreach ($row as $avain => $arvo) {
            echo "<th>" . ucfirst($avain) . "</th>";
        }
```

```
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 |