STT-NF

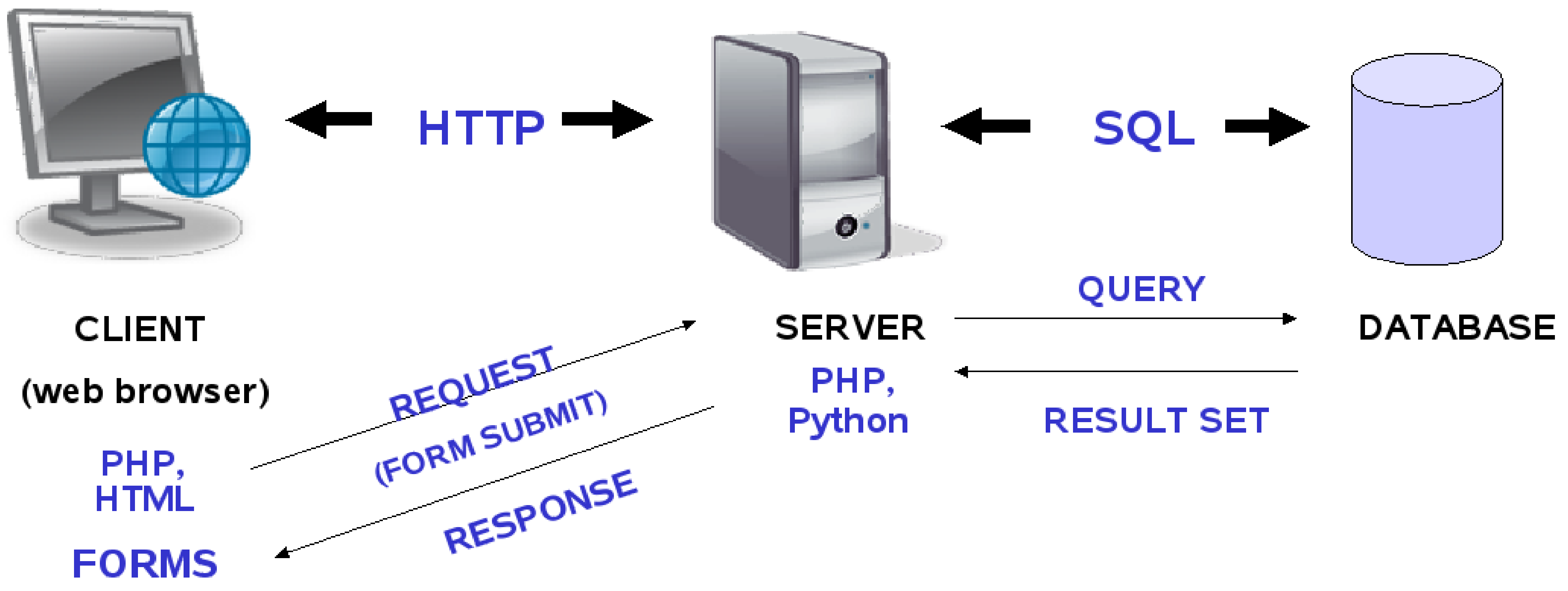
Pemrograman Web

Sirojul Munir | rojulman@nurulfikri.ac.id

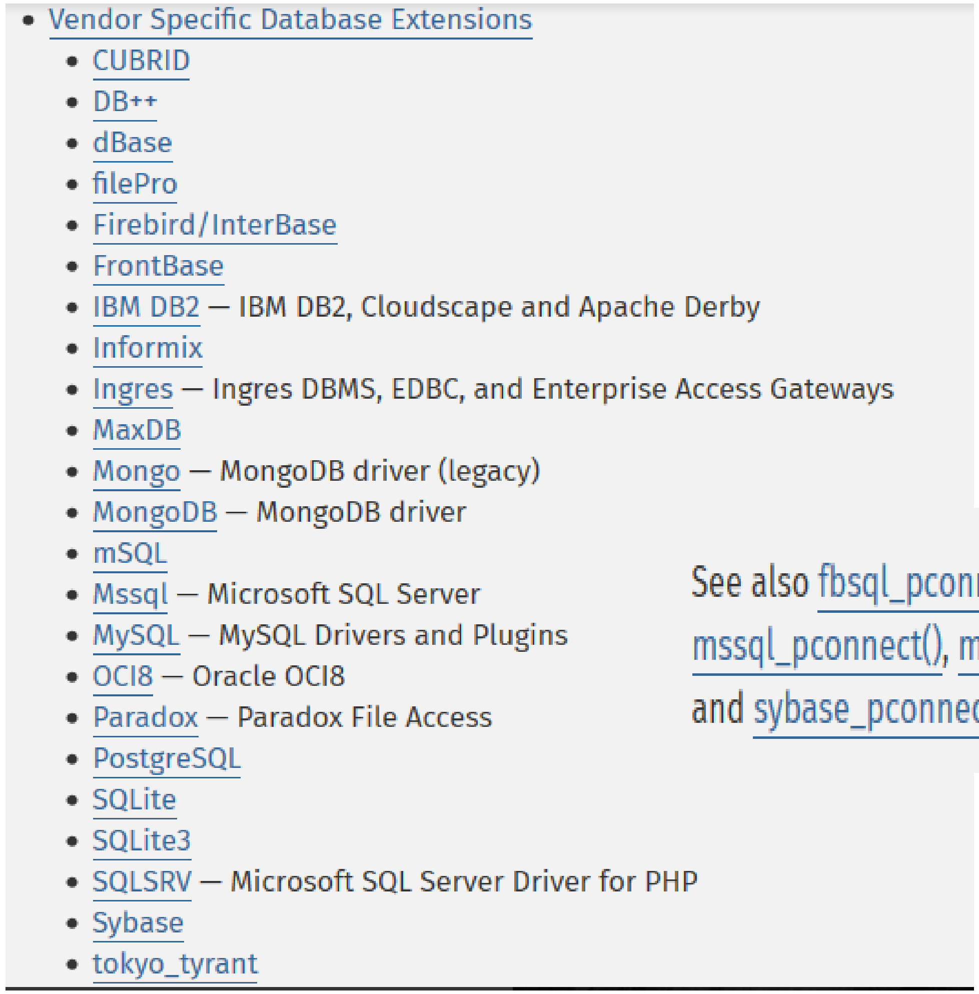
PHP - Database

Sirojul Munir | rojulman@nurulfikri.ac.id

Client – Server - Database

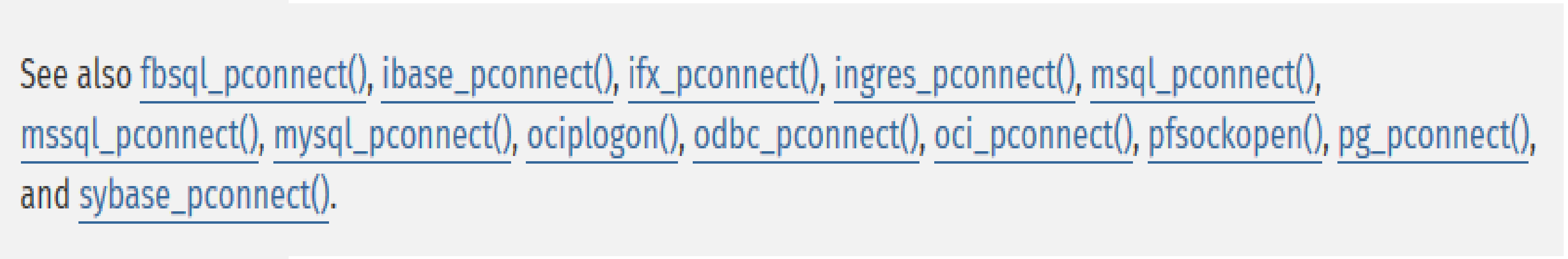


PHP – Database Support



- Vendor Specific Database Extensions
  - CUBRID
  - DB++
  - dBase
  - filePro
  - Firebird/InterBase
  - FrontBase
  - IBM DB2 — IBM DB2, Cloudscape and Apache Derby
  - Informix
  - Ingres — Ingres DBMS, EDBC, and Enterprise Access Gateways
  - MaxDB
  - Mongo — MongoDB driver (legacy)
  - MongoDB — MongoDB driver
  - mSQL
  - Mssql — Microsoft SQL Server
  - MySQL — MySQL Drivers and Plugins
  - OCI8 — Oracle OCI8
  - Paradox — Paradox File Access
  - PostgreSQL
  - SQLite
  - SQLite3
  - SQLSRV — Microsoft SQL Server Driver for PHP
  - Sybase
  - tokyo_tyrant

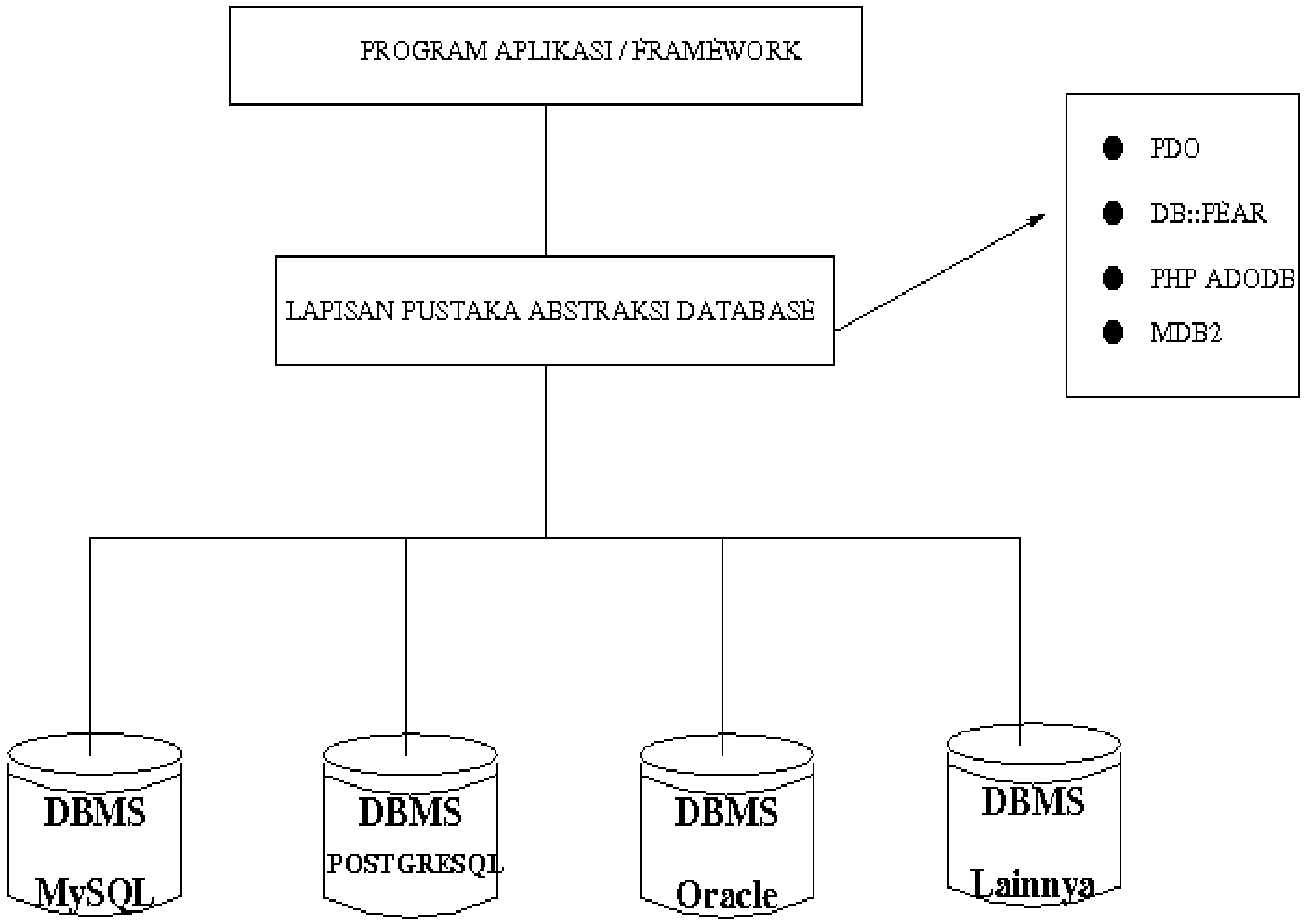
- Native Library
  - php-mysql
  - php-pgsql
  - php-oci
  - php-odbc
  - …

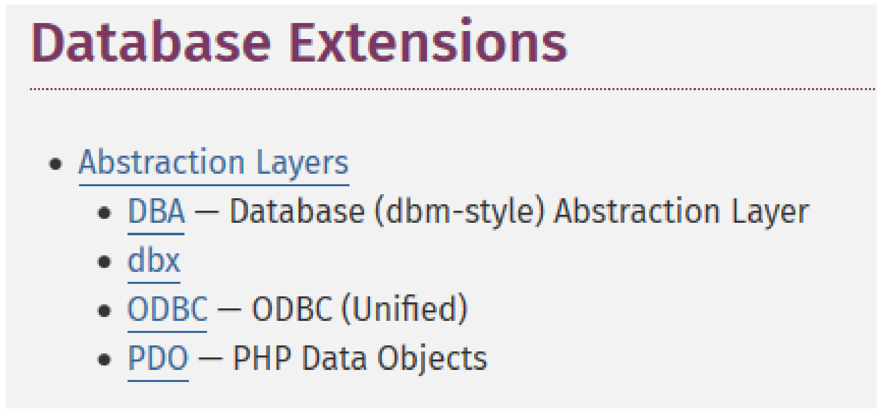


See also fbsql_pconnect(), ibase_pconnect(), ifx_pconnect(), ingres_pconnect(), msql_pconnect(), mssql_pconnect(), mysql_pconnect(), ociplogon(), odbc_pconnect(), oci_pconnect(), pfsockopen(), pg_pconnect(), and sybase_pconnect().

http://php.net/manual/en/refs.database.php

PHP – Database Abstraksi





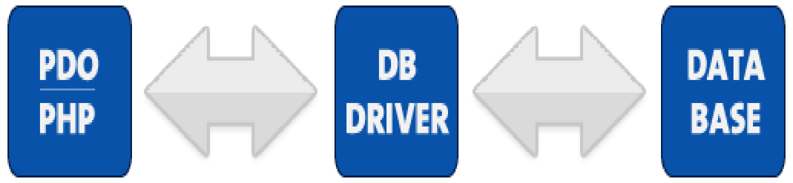
Database Extensions

- Abstraction Layers
  - DBA — Database (dbm-style) Abstraction Layer
  - dbx
  - ODBC — ODBC (Unified)
  - PDO — PHP Data Objects

http://php.net/manual/en/refs.database.php

PHP – PDO (PHP Data Object)

- Mulai PHP 5.0 , PDO menjadi Library default untuk koneksi/akses database



```
<?php
 print_r(PDO::getAvailableDrivers());
?>
```

- PDO Database Driver Support:
- 1. PDO_DBLIB, support database FreeTDS / Microsoft SQL Server / Sybase
  2. PDO_FIREBIRD , Firebird/Interbase 6
  3. PDO_IBM , IBM DB2
  4. PDO_INFORMIX , IBM Informix Dynamic Server
  5. PDO_MYSQL , MySQL 3.x/4.x/5.x
  6. PDO_OCI , Oracle Call Interface
  7. PDO_ODBC , ODBC v3 (IBM DB2, unixODBC and win32 ODBC)
  8. PDO_PGSQL , PostgreSQL
  9. PDO_SQLITE , SQLite 3 and SQLite 2

PDO :: Data Source Name (DSN)

- Konfigurasi untuk akses database :
  - **`database driver, host, db (schema) name`** and **`charset`**, as well as less frequently used **`port`** and **`unix_socket`** go into **DSN**;
  - **`username`** and **`password`** go to constructor;
- all other options go into options array.

PDO :: Data Source Name (DSN)

- Konfigurasi untuk akses database :
  - **database driver, host, db (schema) name** and **charset**, as well as less frequently used **port** and **unix_socket** go into **DSN**;
  - **username** and **password** go to constructor;

```
<?php
  $host = '127.0.0.1';
  $db   = 'dblatihan';
  $user = 'root';
  $pass = '';
  $charset = 'utf8mb4';

  $dsn = "mysql:host=$host;dbname=$db;charset=$charset";
?>
```

PDO :: Data Source Name (DSN)

- all other options go into options array.

```
$opt = [
    PDO::ATTR_ERRMODE            => PDO::ERRMODE_EXCEPTION,
    PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
    PDO::ATTR_EMULATE_PREPARES   => false,
];
```

PDO Object :: new PDO()

- Create Connection :: PDO Instance Class

```
<?php
   $host = '127.0.0.1';
   $db   = 'dblatihan';
   $user = 'root';
   $pass = '';
   $charset = 'utf8mb4';

   $dsn = "mysql:host=$host;dbname=$db;charset=$charset";


$opt = [
    PDO::ATTR_ERRMODE            => PDO::ERRMODE_EXCEPTION,
    PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
    PDO::ATTR_EMULATE_PREPARES   => false,
];


$dbh = new PDO($dsn, $user, $pass, $opt);

?>
```

PDO Exception

```
<?php
try{
  // Database MySQL dengan PDO_MYSQL
  $host = '127.0.0.1';
  $dbname = 'dblatihan';
  $dbuser = 'root';
  $dbpass = '';
  $dbh=new PDO("mysql:host=$host;dbname=$dbname",$dbuser,$dbpass);
  $dbh->setAttribute(PDO::ATTR_ERRMODE,PDO::ERRMODE_EXCEPTION);
 }catch(PDOException $e){
     echo $e->getMessage();
}
?>
```

PDO Connection Database

```
// Database Postgresql dengan PDO_POSTGREQL
$dbh=new PDO("pgsql:host=$host;dbname=$dbname",$dbuser,$dbpass);

// Database SQLite
$dbh =new PDO("sqlite:my/database/path/database.db");

// Database Ms.Access
$dbh= new PDO('odbc:Driver={Microsoft Access Driver(*.mdb)};
                         DBQ=C:\database.mdb;Uid=Admin');
```

PDO::Function

- Fungsi : **exec( )**
- Digunakan untuk eksekusi perintah SQL, jika SQL sukses dilakukan akan mengembalikan nilai 0

```
$sql1 = " CREATE TABLE prodi( id integer auto_increment primary key,
kode varchar(2) UNIQUE,nama varchar(50) not null ) ";
$dbh->exec( $sql1 );


$sql2 = " INSERT INTO prodi (kode,nama) VALUES (‘TI’,’Informatika’) ";
$dbh->exec( $sql2 );
```

PDO::Function

- Fungsi : **query ( )**
- Digunakan untuk eksekusi perintah SQL dan mengembalikan hasil query berupa object ResultSet (kumpulan baris data/record)

```
$sql = " SELECT * FROM prodi ";
$rs = $dbh->query( $sql );

foreach($rs as $row) {
 echo '<br/>'$row['id'] . ' – ' . $row[‘nama’] ;
}
```

PDO::Function

- Fungsi : **prepare() & execute ( )**
- Digunakan untuk eksekusi perintah SQL menggunakan prepareStatement

```
$sql = " INSERT INTO prodi (kode,nama) VALUES (?,?) ";
$statement1 = $dbh->prepare( $sql );
$ar_data = ['TE', 'Teknik Elektro ']; // array
$statement1->execute( $ar_data );

$statement2 = $dbh->prepare(" DELETE FROM prodi WHERE id=? " );
$statement2->execute( array(2) );
```

PDO::Function

- Fungsi : **fetch( )**
- Digunakan untuk eksekusi perintah SQL menggunakan prepareStatement yaitu untuk mengambil satu baris hasil query

```
$sql = " SELECT * FROM prodi WHERE id=? ";
$statement1 = $dbh->prepare( $sql );
$statement1->execute( array(2) );
$row = $statement1->fetch();

echo 'ID : '. $row['id']. ' -- ' . $row['nama'];
```

PDO::Function

- Fungsi : **fetch( )**
- Dapat memiliki opsi argumen
  - **PDO::FETCH_NUM** returns enumerated array
  - **PDO::FETCH_ASSOC** returns associative array
  - **PDO::FETCH_BOTH** - both of the above
  - **PDO::FETCH_OBJ** returns object
  - **PDO::FETCH_LAZY** allows all three (numeric associative and object) methods without memory overhead.

```
$statement = $dbh->prepare("SELECT * FROM produk");
$row = $statement1->fetch(PDO::FETCH_OBJ);
echo $row->id . ` - ` . $row->nama;
```

PDO::Function

- Fungsi : **fetchAll( )**
- Digunakan untuk eksekusi perintah SQL menggunakan prepareStatement yaitu untuk mengambil kumpulan baris hasil query (resulset)

```
$sql = " SELECT * FROM prodi ";
$statement1 = $dbh->prepare( $sql );
$statement1->execute( );
$rows = $statement1->fetchAll();

echo 'ID : '. $row['id']. ' -- ' . $row['nama'];
```

PDO::Function

- Fungsi : **rowCount( )**
- Digunakan untuk mendapakan jumlah baris dari hasil query (affected rows) dari perintah SQL : INSERT, UPDATE atau DELETE

```
$sql = " DELETE FROM prodi ";
$statement1 = $dbh->prepare( $sql );
$statement1->execute( );
$jml = $statement->rowCount();

echo 'Jumlah Data Yang DIHAPUS : '. $jml;
```

PDO::Function

- Fungsi : **fetchColumn( )**
- Dapat digunakan untuk mengambil data dari fungsi aggregate : COUNT, MAX, MIN, AVG pada perintah query

```
$sql = "SELECT COUNT(id) FROM prodi";
$jumlah = $dbh->query($sql)->fetchColumn();
echo 'Jumlah Data : ' . $jumlah ;
```

PDO :: Transaction

- Kumpulan query dapat di eksekusi dalam block transaction
- Pada transaction harus dipastikan perintah query tidak terjadi kesalahan (error exception)
- Berikut method untuk transaction menggunakan PDO
  - beginTransaction() to start a transaction
  - commit() to commit one
  - rollback() to cancel all the changes you made since transaction start.

PDO :: Transaction

```
try {
   $dbh->beginTransaction();
   $stmt = $dbh->prepare("INSERT INTO users (name) VALUES (?)");
   foreach (['Indra','Rio‘, 'Edo'] as $name)
   {
      $stmt->execute([$name]);
   }
   $dbh->commit();
}catch (Exception $e){
   $dbh->rollback();
   throw $e;
}
```

Referensi

- http://php.net/manual/en/book.pdo.php
- https://phpdelusions.net/pdo