

Übungen

22.05.2025

Inhalt

1	Hinweise.....	4
1.1	Bitte beachten Sie.....	4
1.2	Kontakt.....	4
2	Übungen.....	5
2.1	XAMPP installieren / aktivieren.....	5
2.2	mysqli / php.....	5
2.2.1	Datenbank und Daten erzeugen (manuell / mysqli).....	5
2.2.2	Daten lesen (mysqli).....	6
2.2.3	Neue Datensätze (mysqli).....	7
2.2.4	Datensätze löschen (mysqli).....	8
2.2.5	Sicherheit.....	9
2.2.6	Base-Beispiel zu Laufen bringen (optional).....	9
2.3	PHP und ODBC.....	11
2.3.1	PHP/ODBC => Textdatei.....	11
2.3.2	PHP/ODBC => MySQL.....	11
2.4	Python und ODBC.....	12
2.4.1	Python/ODBC => MySQL.....	12
2.4.2	Python => SQLite.....	12
2.5	Suchmaschine / WebCrawler.....	12
2.5.1	WebCrawler (1) – Links einer Seite in Datenbank ablegen.....	13
2.5.2	WebCrawler (2) – Seiteninhalte in Datenbank ablegen.....	15
2.5.3	WebCrawler (3) – Crawlerdatenbank interaktiv erweitern.....	16
2.5.4	WebCrawler: Optionale Erweiterungen.....	16
2.6	MongoDB.....	18
2.6.1	MongoDB – MongoShell / Javascript.....	18
2.7	PDO.....	20
2.8	JDBC.....	21
2.8.1	JDBC – Betriebssystem-Speicherauslastung aufzeichnen / auslesen / auswerten.....	21
2.8.2	Java / JDBC (optional).....	21
2.8.3	JDBC - Metadaten.....	21
2.9	JAVA: Batch Abfrage.....	22
2.10	JAVA Servlet (ohne DB).....	22

2.11	JAVA Servlet / SQL Portal.....	22
2.12	JAVA: Prepared Statement.....	22
2.13	JAVA: Blätten im ResultSet.....	22
2.14	JAVA: Updatable ResultSet.....	23
2.15	Transaktionen.....	23
2.16	ADO.NET Mit C# und ODBC Datenbank auslesen.....	24
2.17	Mit C++ und ODBC auf Datenquellen zugreifen.....	25
2.18	REST / API finden.....	26
2.19	REST / REST API - Abfrage programmieren.....	26

1 Hinweise

1.1 Bitte beachten Sie

- Übungen sind nach Möglichkeit eigenständig oder in 2-er-Gruppen durchzuführen.
- Bei allen Übungen kommt es vorrangig darauf an, sich intensiv mit dem Thema auseinanderzusetzen.
- Seien Sie kreativ.
- Versuchen Sie sich auch an Entwicklungen, Experimenten und Ansätzen, die nicht in den Übungen vorgesehen sind.
- Erweitern Sie Ihren Horizont, in dem Sie die einschlägigen Internetquellen oder weiterführende Literatur nutzen.
- Trauen Sie sich, nachzufragen.
- Sprechen Sie mit Ihren Mitstudierenden.
- Bereiten Sie sich (mental) darauf vor, dass Sie das Arbeitsergebnis ggf. vorstellen.

1.2 Kontakt

Alexander Simianer
alexander.simianer@gmx.de
+49 (0) 157 79 21 22 98 (Tel/SMS/WApp)

2 Übungen

2.1 XAMPP installieren / aktivieren

Stellen Sie sicher, dass XAMPP installiert ist und...

- dass PHP verfügbar ist,
- dass MySQL verfügbar ist,
- dass in htdocs per index.php und Zugriff auf den lokalen WebServer in einem Internetbrowser „Hello world“ dargestellt werden kann.
(<http://localhost>)

Download XAMPP: <https://www.apachefriends.org/de/index.html>

Hinweis:

Sie können auch Alternativen zu XAMPP verwenden. Z.B. einen Dockercontainer.

Vgl.: <https://www.sitepoint.com/docker-php-development-environment/>

2.2 mysqli / php

2.2.1 Datenbank und Daten erzeugen (manuell / mysqli)

Erzeugen Sie eine MySQL-Datenbank, Datenbanktabellen und Datenbankinhalte

Datenbank: mydb

Tabelle 1: **user**

Attribute: id, firstname, lastname, age, id_car

Tabelle 2: **car**

Attribute: id, brand, country

Erzeugen Sie je Tabelle mindestens 5 Datensätze!

- (A) Nutzen Sie MySQL-Admin (o.ä.) zur Erstellung.
- (B) Erstellen Sie ein PHP-Programm, das per mysqli-API die Erzeugung durchführt.

Hinweise:

- Der PHP Interpreter befindet sich hier: <PFAD>\xampp\php\php.exe
- Aufruf: php -f <dateiname>.php
- MySQLi Handbuch: <http://php.net/manual/de/book.mysqli.php>
- SQL: <http://www.w3schools.com/sql/default.asp>

2.2.2 Daten lesen (mysqli)

Erstellen Sie ein PHP-Programm, das per mysqli-**API** die in der vorherigen Aufgabe erstellte Datenbank ausliest und beim Aufruf in Webbrowser eine Tabelle erzeugt, die die Spalten „Nachname“, „Alter“ und „Automarke“ enthält.

User Name	User Age	Car Brand
Schmitt	45	BMW
Venus		VW
MUELLER	111	VW
Francis	92	Alfa Romeo
Moon		Toyota
Jupiter		Toyota
Maier-8	12	

Hinweise:

- Tabellendarstellung per HTML:
 - http://www.w3schools.com/html/html_tables.asp
 - http://www.w3schools.com/bootstrap/bootstrap_tables.asp

2.2.3 Neue Datensätze (mysqli)

Erstellen Sie ein HTML-Formular, das es ermöglicht, neue Benutzer (user) anzulegen.

Benutzer anlegen

Vorname:

Nachname:

Alter:

Hinweise:

- Sie können beispielsweise eine html-Formulardatei erstellen und parallel dazu eine php-Datei, die den Programmcode für den Datenbankzugriff enthält.
- Prüfen Sie, ob nun neu angelegte Benutzer tatsächlich in der Tabelle der vorhergehenden Aufgabe (Abschnitt 2.2.2) dargestellt werden!

2.2.4 Datensätze löschen (mysqli)

Erweitern Sie das Programm so, dass es möglich ist, Benutzer wieder aus der Datenbank zu löschen.

Exemplarisch:

Benutzer anlegen

Vorname:

Nachname:

Alter:

Benutzer löschen

Benutzer ID:

Benutzerliste

1 / Joe / Maier-8 / 12
2 / Tina / Schmitt / 45
3 / Anna / Moon /
4 / Tim / Jupiter /
5 / Lisa / Venus /
6 / Peter / Francis / 92
7 / KARL / MUELLER / 111
15 / Testuser / Testuser Nachname / 99

2.2.5 Sicherheit

Erstellen Sie ein PHP-Programm, das per http-Query-String (z.B.: `http://beispiel.de?lastname='maier'` oder `http://beispiel.de?id=4`) die Automarke des Nutzers zurückliefert.

Verhindern Sie - z.B. durch `if ($id == 3) {exit;} - dass auf bestimmte Datensätze zugegriffen werden kann.`

Minimieren Sie das Risiko eines SQL-Injection-Angriffs!

Tipps:

- Zugriff auf Argumente z.B. per: `$_GET['<ARGUMENT NAME>']`
- Ist z.B. die nachfolgende Abfrage sicher?
`$sql = "SELECT user.id, user.firstname, user.lastname, user.age, car.brand, car.id as car_id FROM user left JOIN car ON user.id_car=car.id WHERE user.id = $arg_id";`
- Nutzen Sie ggf. „prepared“ Statements.

2.2.6 Base-Beispiel zu Laufen bringen (optional)

Bringen Sie das nachfolgende Beispiel zum Laufen. Experimentieren Sie!

```
<?php
// Let's pass in a $_GET variable to our example, in this case
// it's aid for actor_id in our Sakila database. Let's make it
// default to 1, and cast it to an integer as to avoid SQL injection
// and/or related security problems. Handling all of this goes beyond
// the scope of this simple example. Example:
// http://example.org/script.php?aid=42
if (isset($_GET['aid']) && is_numeric($_GET['aid'])) {
    $aid = (int) $_GET['aid'];
} else {
    $aid = 1;
}

// Connecting to and selecting a MySQL database named sakila
// Hostname: 127.0.0.1, username: your_user, password: your_pass, db: sakila
$mysqli = new mysqli('127.0.0.1', 'your_user', 'your_pass', 'sakila');

// Oh no! A connect_errno exists so the connection attempt failed!
if ($mysqli->connect_errno) {
    // The connection failed. What do you want to do?
    // You could contact yourself (email?), log the error, show a nice page, etc.
    // You do not want to reveal sensitive information

    // Let's try this:
    echo "Sorry, this website is experiencing problems.";

    // Something you should not do on a public site, but this example will show you
    // anyways, is print out MySQL error related information -- you might log this
    echo "Error: Failed to make a MySQL connection, here is why: \n";
    echo "Errno: " . $mysqli->connect_errno . "\n";
    echo "Error: " . $mysqli->connect_error . "\n";

    // You might want to show them something nice, but we will simply exit
    exit;
}

// Perform an SQL query
$sql = "SELECT actor_id, first_name, last_name FROM actor WHERE actor_id = $aid";
if (!$result = $mysqli->query($sql)) {
    // Oh no! The query failed.
    echo "Sorry, the website is experiencing problems.";
}
```

```
// Again, do not do this on a public site, but we'll show you how
// to get the error information
echo "Error: Our query failed to execute and here is why: \n";
echo "Query: " . $sql . "\n";
echo "Errno: " . $mysqli->errno . "\n";
echo "Error: " . $mysqli->error . "\n";
exit;
}

// Phew, we made it. We know our MySQL connection and query
// succeeded, but do we have a result?
if ($result->num_rows === 0) {
    // Oh, no rows! Sometimes that's expected and okay, sometimes
    // it is not. You decide. In this case, maybe actor_id was too
    // large?
    echo "We could not find a match for ID $aid, sorry about that. Please try
again.";
    exit;
}

// Now, we know only one result will exist in this example so let's
// fetch it into an associated array where the array's keys are the
// table's column names
$actor = $result->fetch_assoc();
echo "Sometimes I see " . $actor['first_name'] . " " . $actor['last_name'] . " on
TV.";

// Now, let's fetch five random actors and output their names to a list.
// We'll add less error handling here as you can do that on your own now
$sql = "SELECT actor_id, first_name, last_name FROM actor ORDER BY rand() LIMIT 5";
if (!$result = $mysqli->query($sql)) {
    echo "Sorry, the website is experiencing problems.";
    exit;
}

// Print our 5 random actors in a list, and link to each actor
echo "<ul>\n";
while ($actor = $result->fetch_assoc()) {
    echo "<li><a href='" . $_SERVER['SCRIPT_FILENAME'] . "?aid=" .
$actor['actor_id'] . "'>\n";
    echo $actor['first_name'] . " " . $actor['last_name'];
    echo "</a></li>\n";
}
echo "</ul>\n";

// The script will automatically free the result and close the MySQL
// connection when it exits, but let's just do it anyways
$result->free();
$mysqli->close();
?>
```

Quelle: (https://www.php.net/manual/pt_BR/mysqli.examples-basic.php)

2.3 PHP und ODBC

2.3.1 PHP/ODBC => Textdatei

Richten Sie eine ODBC-Datenquelle ein, die eine Schnittstelle zu Textdateien anbietet. Verwenden Sie hierzu einen Microsoft Text Treiber.

Erstellen Sie eine ";"-separierte Textdatei mit Dateiname user.txt, die eine Tabelle mit den Attributen: id, firstname, lastname und age und einige Datensätze enthält.

Erstellen Sie ein **PHP**-Programm, das die so angelegte Datentabelle ausliest und weitere Datensätze einfügt.

Prüfen Sie, ob sich die Inhalte der Textdatei tatsächlich verändern!

Wie verhält sich das System, wenn Sie den Lese-/Schreib-Zugriff auf die Textdatei blockieren?

2.3.2 PHP/ODBC => MySQL

Installieren Sie zunächst die auf Ihrem Übungssystem ggf. noch notwendigen ODBC-Treiber für den Zugriff auf **MySQL**.

Richten Sie anschließend eine Datenquelle ein, die den Zugriff auf die in den vorherigen Aufgaben erstellte Datenbank ermöglicht. Prüfen Sie die Verbindung zur Datenquelle.

Schreiben Sie ein **PHP**-Programm, das sich mit der nun eingerichteten ODBC-Datenquelle verbindet, per SQL die Benutzertabelle ausliest und die Benutzernamen in einem Webbrowser auflistet.

Hinweis:

- Ggf. ist auf Ihrem Übungssystem bereits ein passender ODBC Treiber installiert. In diesem Fall verzichten Sie auf die Neuinstallation des Treibers.

Hilfe:

- Treiber: <http://dev.mysql.com/downloads/connector/odbc/> bzw. <http://dev.mysql.com/downloads/file/?id=453005> (64bit ff)
- PHP und ODBC: <http://php.net/manual/de/intro.uodbc.php>

Optional (15min):

Prüfen Sie, ob weitere ODBC-Treiber auf Ihrem Übungssystem vorhanden sind. Versuchen Sie unter Verwendung eines solchen ODBC-Treibers weitere Datenquellen einzurichten und zu nutzen.

2.4 Python und ODBC

2.4.1 Python/ODBC => MySQL

Schreiben Sie ein **Python**-Programm, das sich mit der in der vorherigen Aufgabe eingerichteten ODBC-Datenquelle verbindet und per SQL eine Benutzertabelle mit den Attributen id, firstname, lastname und age erstellt und die Tabelle mit einigen Datensätzen befüllt.

Schreiben Sie ein weiteres Python-Programm, das die Benutzertabelle ausliest und die Benutzernamen in der der Konsole auflistet.

2.4.2 Python => SQLite

Schreiben Sie ein **Python**-Programm, das eine SQLite-Datenbank erstellt und per SQL eine Benutzertabelle mit den Attributen id, firstname, lastname und age erstellt und die Tabelle mit einigen Datensätzen befüllt.

Schreiben Sie ein weiteres Python-Programm, das die Benutzertabelle ausliest und das Durchschnittsalter der Benutzer errechnet und auf der Konsole ausgibt.

Mögliche Quelle:

<https://www.ionos.de/digitalguide/websites/web-entwicklung/sqlite3-python/>

<https://www.sqlite.org/index.html>

<https://sqlite.org/fiddle/index.html>

<https://sqlitebrowser.org/>

2.5 Suchmaschine / WebCrawler

Realisieren Sie unter Verwendung der Programmiersprache PHP, Python oder JavaScript mit node.js einen Web-Crawler, der Internetseiten hinsichtlich deren Inhalte und Verweise (Links) durchsucht und die zugehörigen Daten serverseitig **in einer relationalen Datenbank** speichert.

Erweitern Sie den Crawler so, das die Datenbasis durch den Anwender um weitere explizit zu durchsuchende Quellen (URIs) erweitert werden kann.

Realisieren Sie eine passende Benutzerschnittstelle, die (a) bei der Erweiterung der Datenbasis unterstützt und (b) eine interaktive Suche ermöglicht.

Beachten Sie, dass eine Suchmaschine typischerweise aus zwei Programmen besteht:

(1) Es gibt ein Suchprogramm (Crawler), das kontinuierlich das Internet bzw. die konfigurierte Datenquelle durchsucht, relevante Informationen extrahiert und für die Suchanfragen in einer (lokalen) Datenbank speichert.

(2) Darüber hinaus gibt es ein Suchprogramm, das eine Benutzerschnittstelle zur Verfügung stellt, über die eine explizite Suche in der (lokalen) Datenbank möglich ist und in der die Suchergebnisse per GUI angezeigt werden.

Möglicher Pseudocode für den Crawler:

Algorithmus worker:

```
while true do
  foreach link in link-list from database do
    if age of latest linkvisit > 1 day then
      crawl(link) // find and append links to db
      update time_stamp latest linkvisit
    end if
  end foreach
end while
```

Algorithmus crawl:

```
databaseUpdateWords(url)
links = getLinks(url);
foreach link in links do
  databaseAppendLink(link)
end foreach
```

Die Aufgabe kann in drei Schritten (1 bis 3) umgesetzt werden:

2.5.1 WebCrawler (1) – Links einer Seite in Datenbank ablegen

Zunächst sollen alle verlinkten Seiten einer Website einschließlich der Seite selbst mit ihren URIs und optional auch mit ihrer Überschrift in einer **Datenbank** aufgezeichnet werden.

Ein Codebeispiel eines rudimentären Crawlers ist gegeben. Er findet aber nur Links auf einer einzelnen Seite.

Crawler (PHP-Code):

```
<?php
class Crawler {

    protected $markup = '';
    public $base = '';

    public function __construct($uri) {
        $this->base = $uri;
        $this->markup = $this->getMarkup($uri);
    }

    public function getMarkup($uri) {
        return file_get_contents($uri);
    }

    public function get($type) {
        $method = "_get_{$type}";
        if (method_exists($this, $method)){
            return call_user_func(array($this, $method));
        }
    }

    protected function _get_images() {
        if (!empty($this->markup)){
            preg_match_all('/<img([^\>]+)\>/i', $this->markup, $images);
            return !empty($images[1]) ? $images[1] : FALSE;
        }
    }

    protected function _get_links() {
        if (!empty($this->markup)){
            //preg_match_all('/<a([^\>]+)\>(.*?)\</a\>/i', $this->markup, $links);
        }
    }
}
```

```
preg_match_all('/href=\"(.*)\"/i', $this->markup, $links);
return !empty($links[1]) ? $links[1] : FALSE;
}
}
}

$crawl = new Crawler('http://www.dhbw-heidenheim.de');
$images = $crawl->get('images');
$links = $crawl->get('links');
?>
<html>

<body>
<h2>webcrawler</h2>
<?php

foreach($links as $l) {

    if (substr($l,0,7)!='http://')
        echo "<br>Link: $crawl->base/$l";

}

?>
</body>
</html>
```

Skizze / Pseudocode für eine rekursive Implementierung:

```
crawl ( $URL )
{
    speichere $URL in die DB als besuchte Seite
    mit Angaben wie title-Inhalt, und Schlagworte.
    für alle Links $link in der mit URL adressierten Seite
    {
        wenn ($link in der Form "/...")
            füge $link links "http://<hostname_von_$URL>" an
        wenn ($link in der Form "...")
            füge $link links
            "http://<hostname_von_$URL>/<dir_von_$URL>/" an
        wenn ($link in der DB nicht als besuchte Seite
            gespeichert ist)
            crawl ( $link );
    }
}
```

Tipps:

- Das Durchsuchen einer Webseite zunächst ohne Datenbankanbindung ablauffähig machen.
- Vereinheitlichung der unterschiedlichen URI-Formate implementieren
- Datenbanktabelle zur Speicherung der gefundenen URIs anlegen
- Algorithmus so modifizieren, dass gefundene URIs gespeichert werden.

Hilfen:

- Connectoren zu mysql: <https://dev.mysql.com/downloads/>
- Reguläre Ausdrücke online testen: <https://regex101.com/>
- Infos zu preg_match_all(): <http://php.net/manual/de/function.preg-match-all.php>
- PHP und MySQL: z.B.: <http://php.net/manual/de/book.mysql.php>
...oder auch: <http://php.net/manual/de/book.uodbc.php>
- Verwendung von node.js mit mysql:
https://www.w3schools.com/nodejs/nodejs_mysql.asp

2.5.2 WebCrawler (2) – Seiteninhalte in Datenbank ablegen

Der Web-Crawler soll dahingehend erweitert werden, dass er neben den Links auch sämtliche Worte aus den Seiten herausliest und in der Datenbank aufzeichnet.

Erstellen Sie ein passendes ER-Diagramm und programmieren Sie die Erstellung der Tabellen der relationalen Datenbank.

Realisieren Sie eine für Suchplattformen typische Eingabemöglichkeit im Webbrowser und eine angemessene Darstellung der Suchergebnisse aus der Datenbank.

Tipp:

- Lassen Sie den Erstellungsverlauf per Ausgabe im Webbrowser oder in einer Protokolldatei oder auch in der Datenbank selbst geeignet protokollieren.
- Löschen Sie Datenbankinhalte ggf. von Zeit zu Zeit, um beim Test mit konsistenten Daten arbeiten zu können.

2.5.3 WebCrawler (3) – Crawlerdatenbank interaktiv erweitern

Erweitern Sie den Crawler so, dass in einem Formular URIs eingegeben werden können, deren Analyseergebnisse dann umgehend in Ihrer Datenbank eingefügt werden.

GUI-Skizze (Entwicklermodus):

localhost/VL-DHBW-DB/040-PHP-CRAWL-WORI

Suchen

DHBW Add URL to Database:

(no http, only www.xyz.de)

Daten absenden

DHBW Search

Heidenheim

Daten absenden

Results

- Heidenheim has id: 3738
 - wordlink found. The id_link is 16
Link: <http://www.dhbw-heidenheim.de><
 - wordlink found. The id_link is 17
Link: <http://www.dhbw-heidenheim.de/><
 - wordlink found. The id_link is 27
Link: <http://www.dhbw-heidenheim.de/Home.135.0.html><
 - wordlink found. The id_link is 28
Link: <http://www.dhbw-heidenheim.de/Kontakt.12.0.html><
 - wordlink found. The id_link is 29
Link: <http://www.dhbw-heidenheim.de/Anfahrt.133.0.html><
 - wordlink found. The id_link is 30
Link: <http://www.dhbw-heidenheim.de/Impressum.134.0.html><
 - wordlink found. The id_link is 31
Link: <http://www.dhbw-heidenheim.de/Sitemap.11.0.html><
 - wordlink found. The id_link is 33
Link: <http://www.dhbw-heidenheim.de/Studienangebote.92.0.html><
 - wordlink found. The id_link is 34
Link: <http://www.dhbw-heidenheim.de/Firmenpartner.893.0.html><

2.5.4 WebCrawler: Optionale Erweiterungen

Google arbeitet an seiner Suchmaschine bereits seit über 20 Jahren. Auch der im Rahmen dieser Übungen erstellte Crawler und die Suchfunktion haben noch Erweiterungspotential:

- **Bildersuche:** Erweitern Sie den Crawler so, dass auch Bilder (img) gesucht und gefunden werden.
 - Legen Sie gefundene Bilder als „Thumbnails“ in der Datenbank ab.

- Legen Sie Abbildungstexte (alt=...) ebenfalls in der Datenbank ab. Überlegen Sie sich auch ein Verfahren, das mit fehlenden alt-Texten umgehen kann.
- Erweitern Sie den Suchalgorithmus um eine Bildersuche.
- Erweitern Sie das GUI der Suchfunktion um eine Bildersuche.
- Stellen Sie als Suchergebnis die gefunden Bilder dar.
- **Stoppwortliste:** Verwenden Sie eine Stoppwortliste, um Trivialergebnisse zu vermeiden und die Datenbankauslastung zu reduzieren.
Vorschlag:
"and", "the", "of", "to", "einer", "eine", "eines", "einem", "einen", "der",
"die", "das", "dass", "daß", "du", "er", "sie", "es", "was", "wer", "wie",
"wir", "und", "oder", "ohne", "mit", "am", "im", "in", "aus", "auf", "ist",
"sein", "war", "wird", "ihr", "ihre", "ihres", "ihnen", "ihrer", "als", "für",
"von", "mit", "dich", "dir", "mich", "mir", "mein", "sein", "kein", "durch",
"wegen", "wird", "sich", "bei", "beim", "noch", "den", "dem", "zu", "zur",
"zum", "auf", "ein", "auch", "werden", "an", "des", "sein", "sind", "vor",
"nicht", "sehr", "um", "unsere", "ohne", "so", "da", "nur", "diese", "dieser",
"diesem", "dieses", "nach", "über", "mehr", "hat", "bis", "uns", "unser",
"unserer", "unserem", "unsers", "euch", "euers", "euer", "eurem", "ihr",
"ihres", "ihrer", "ihrem", "alle", "vom"
- **Background-Crawler mit Statusansicht:** Lassen Sie den Crawler selbstständig im „Hintergrund“ laufen. Stellen Sie eine Benutzerschnittstelle zur Verfügung, über die der Status des Crawlers eingesehen werden kann.
- **Seitentitel:** Erweitern Sie die Darstellung des Suchergebnisses um die Anzeige des Seitentitels.
- **Update:** Zeichnen Sie die Zeitpunkte der Seitenbesuche (Timestamps) in der Datenbank auf und erlauben Sie eine wiederholte Crawleranalyse, wenn die Seite seit mindestens 1 Tag nicht mehr vom Crawler besucht worden ist.
- **Ranking:** Zeichnen Sie in der Datenbank auf, von welcher Seite eine Seite verlinkt wird und ranken Sie die Seiten z. B. entsprechend der Häufigkeit ihrer Verlinkung und des Rankings der auf sie verlinkenden Seite. Sortieren Sie die Suchergebnisse entsprechend des Seitenrankings, so dass der Anwender Ihrer Suchmaschine die relevantesten Ergebnisse ganz oben findet.
- **Kontext:** Erweitern Sie die Darstellung des Suchergebnisses um die Anzeige der Textumgebung des gefundenen Wortes.

- **Zusammenfassung per KI:** Erstellen Sie eine KI-gestützte Zusammenfassung der Suchergebnisse, die den Anwender auf einen Blick informieren, ob er seine Suche erfolgreich war.
- **Weitere Suchvorschläge per KI:** Erstellen Sie KI-gestützt weitere Suchvorschläge und stellen Sie diese erweiterte Suche per „Klick“ zur Verfügung.

2.6 MongoDB

2.6.1 MongoDB – MongoShell / Javascript

(a) Stellen Sie zunächst sicher, dass MongoDB auf Ihrem Übungssystem installiert ist.

Varianten:

- **Hochschulrechner**
Auf den Hochschulrechner ist die MongoDB möglicherweise vorinstalliert. Starten Sie per `mongod` den entsprechenden Prozess aus der Kommandozeile.
- **Direkte Installation**
Installieren Sie MongoDB auf Ihrem Rechner.
Installationsquelle:
<https://www.mongodb.com/download-center/community>
- **Dockercontainer**
Alternativ zur direkten Installation kann auch eine passender Dockercontainer verwendet werden. Hierzu kann man per Kommandozeile den Dockercontainer aktivieren. Ggf. wird die aktuelle Mongo-Version heruntergeladen:

```
docker run -d -p 27017:27017 --name example-mongo mongo:latest
```

James Walker erläutert in [<https://www.howtogeek.com/devops/how-to-run-mongodb-in-a-docker-container/>] das Kommando wie folgt:

„Damit erhalten Sie einen Live-Server, auf dem die neueste Version von MongoDB läuft. Er verwendet das offizielle Image, das auf Docker Hub verfügbar ist. Das Flag `-d` (detach) bedeutet, dass der Container getrennt von Ihrem Shell-Prozess im Hintergrund ausgeführt wird.

Der Container-Port 27017, der MongoDB-Standard, ist an Port 27017 auf Ihrem Host gebunden. Sie können sich mit Ihrer Mongo-Instanz über `localhost:27017` verbinden. Wenn Sie die Portnummer ändern möchten, ändern Sie den ersten Teil des Flags `-p`, z. B. `9000:27017`, um

localhost:9000 zu verwenden."

Anschließend kann auch die Mongo-Shell genutzt werden:

```
docker exec -it example-mongo mongo
```

- **Online-Übungen mit der Mongo-Shell (ohne Installation)**

Einfache Online-Übungen mit der Mongo-Shell können auch über die Webseite <https://onecompiler.com/mongodb> durchgeführt werden.

(b) Erstellen Sie ein serverseitiges JavaScript-Programm, das eine MongoDB-Datenbank öffnet, die Collection "geoprims" erzeugt und im Anschluss folgende grafische Primitive anlegt:

Kreis1

radius: 12
x: 0
y: 5

Kreis2

radius: 4
x: 3
y: -5

Linie1

x1: 1
y1:-4
x2: 3
y2:15

(c) Öffnen Sie interaktiv die Mongo-Shell in der Eingabeaufforderung und lassen Sie sich auf diesem Weg alle grafischen Primitive auflisten.

(d) Erweitern Sie das in (b) erstellte Programm so, dass es in der Eingabeaufforderung alle grafischen Objekte auflistet, die sich geometrisch vollständig unterhalb von der x-Achse befinden.

Hinweise:

- Tutorial: <https://docs.mongodb.org/manual/reference/method/>
- Ggf. muss der Mongo-Server manuell gestartet werden: mongod
- Aufruf von Mongo Scripts: mongo.exe <dateiname>.js
- Aufruf der Mongo Shell: mongo.exe
=> help liefert weitere Kommandos.

2.7 PDO

Erstellen Sie ein PHP-Programm, das unter Verwendung der Abstraktionsebene PDO (PHP Data Objects) schrittweise Attributinhalt eines Datensatzes ändert (z.B. durch mehrfachen Aufruf des SQL-Kommandos „UPDATE“) und die Änderungen im Anschluss per Commit durchschreibt.

Prüfen Sie hierbei zunächst, ob die Daten tatsächlich erst bei einem Commit und nicht bereits vorher in die Datenbank geschrieben werden.

Führen Sie nun zusätzlich ein Rollback durch.

Prüfen Sie, ob ein Rollback das Durchschreiben der Daten wirklich verhindert.

Hilfe: <http://php.net/manual/de/book.pdo.php>

2.8 JDBC

2.8.1 JDBC – Betriebssystem-Speicherauslastung aufzeichnen / auslesen / auswerten

Erstellen Sie mit Eclipse eine javabasierte **Datenbankanwendung**, die JDBC verwendet, um auf MySQL zugreifen zu können.

Aufgabe der Anwendung ist es zunächst, in einer MySQL-Datenbanktabelle „log“ bei jedem Programmaufruf das aktuelle **Datum**, die aktuelle **Uhrzeit** sowie die aktuelle **Speicherauslastung** einzutragen.

Das Programm soll bei jedem Aufruf in der Console ausgeben, wie viele Sekunden seit dem letzten Aufruf vergangen sind, wie groß die mittlere Aufrufhäufigkeit des Programms (Messhäufigkeit) ist, und um wie viel Prozent die aktuelle Speicherauslastung von der mittleren Speicherauslastung (gemittelt über eine Stunde) abweicht.

Mögliche Ausgabe:

Aktuelle Speicherauslastung: 12348 MB (+23% gegenüber Durchschnitt)

Mittlere Speicherauslastung: 10029 MB gemittelt über 1h.

Mittlere Messhäufigkeit: 3 Messungen pro Stunde gemittelt über 3 Tage.

Tipps:

- Nutzen Sie JDBC und einen MySQL JDBC-Treiber.
- Ermittlung der aktuellen Speicherauslastung z.B. so:
`Runtime rte = Runtime.getRuntime();`
`current_load = rte.totalMemory() - rte.freeMemory();`

Wenn noch Zeit: Erweitern Sie das Programm so, dass es die Messungen selbstständig wiederholt.

2.8.2 Java / JDBC (optional)

Erstellen Sie mit Eclipse in JAVA eine Datenbankanwendung, die JDBC verwendet, um auf MySQL zuzugreifen.

Variante 1: Wiederholen sie die Entwicklungen der bisherigen Übungen nun mit Java und JDBC.

Variante 2: Erstellen Sie eine weitere Datenbankanwendung Ihrer Wahl.

2.8.3 JDBC - Metadaten

Erstellen Sie eine Java-Anwendung, die beim Aufruf per JDBC die **Metadaten** einer beliebigen MySQL Datenbanktabelle ausliest, und die Inhalte in eine HTML5-Textdatei so exportiert, dass diese in einem WebBrowser geöffnet werden kann.

Folgende Daten sollen angezeigt werden:

- Erstellungszeit

- Tabellennamen
- Tabellenüberschriften
- Zeilennummern
- Inhalte
- Optional: Prozentualer zahlenmäßiger Anteil (z.B. 97%) von Zellen mit Inhalt.

2.9 JAVA: Batch Abfrage

Erstellen Sie eine Java Demonstrations-Anwendung Ihrer Wahl, in der erkennbar wird, wie SQL-Abfragen in „Batches“ zusammengefasst werden können.

2.10 JAVA Servlet (ohne DB)

Realisieren Sie ein JAVA Servlet, das bei Aufruf per WebBrowser die Quadratzahlen der Zahlen 1 bis 10 erscheinen lässt.

(Eclipse, Tomcat, XAMPP)

2.11 JAVA Servlet / SQL Portal

Erstellen Sie unter Verwendung eines Servlets ein SQL-Portal, das beliebige SQL Abfragen entgegen nimmt und das Abfrageergebnis tabellarisch darstellt.

http://localhost:8080/MyServlet004/SQLRequester2

Servlet zur Eingabe beliebiger SQL-Anfragen

Ergebnis der Abfrage:
*SELECT * FROM USER WHERE firstname='Joe'*

id	firstname	lastname	id_car
1	Joe	Francis	null

SQL-Code:

```
SELECT * FROM USER WHERE firstname='Joe'
```

Absenden Löschen

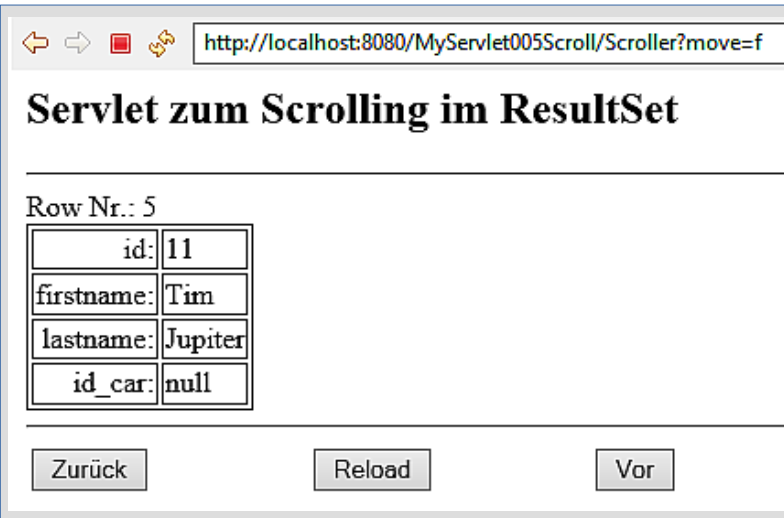
2.12 JAVA: Prepared Statement

(a) Erstellen Sie eine Java Anwendung, die per Loop alle Datensätze einer Tabelle mit den Datensatz-IDs von 1 bis 100 abfragt. Verwenden Sie hierbei „Prepared Statements“.

(b) Überlegen Sie, ob man mit prepared Statements, das SQL-Portal sicherer machen könnte. Welche andere Möglichkeiten wären einsetzbar, um das Portal vor Injection Angriffen zu schützen?

2.13 JAVA: Blätten im ResultSet

Erstellen Sie ein Java Servlet, das in einem Scrolling-Portal das Blätten in einem ResultSet ermöglicht.



Servlet zum Scrolling im ResultSet

Row Nr.: 5

id:	11
firstname:	Tim
lastname:	Jupiter
id_car:	null

Zurück Reload Vor

2.14 JAVA: Updatable ResultSet

Erstellen Sie eine Java Anwendung Ihrer Wahl, die zeigt, wie **Datenbankinhalte über die Manipulation von ResultSets** verändert werden können.

2.15 Transaktionen

Erstellen Sie ein JAVA Servlet, in dem Transaktionen

- begonnen
- committet und
- abortet werden.

Prüfen Sie ob die Operationen zum erwarteten Ergebnis führen. Versuchen Sie Inkonsistenzen zu erzeugen.

Vorschlag:

- Sie könnten beispielsweise die Scroll-Anwendung (siehe oben) um Transaktionen erweitern.

2.16 ADO.NET Mit C# und ODBC Datenbank auslesen

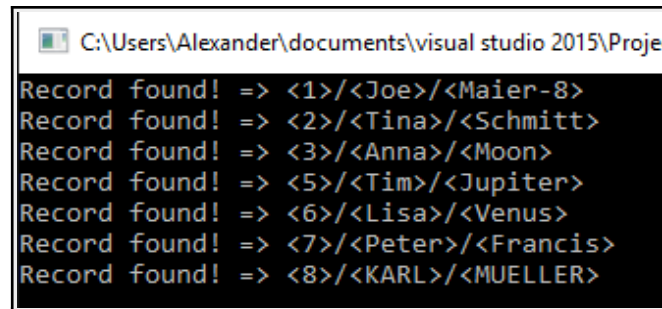
Erstellen Sie eine Anwendung, die unter Verwendung des .NET Frameworks und ADO.NET die Inhalte der ersten drei Spalten einer MySQL-Tabelle in der Konsole ausgibt.

Hinweise

- Nutzen Sie „System.Data.Odbc“ mit den Klassen:
 - „OdbcConnection“
 - „OdbcCommand“
 - „OdbcDataReader“

2.17 Mit C++ und ODBC auf Datenquellen zugreifen

Erstellen Sie mit MS VisualStudio eine C++-Konsolenanwendung, die unter Verwendung einer ODBC Datenquelle ihrer Wahl (MySQL, Excel, Oracle, ...) und der ODBC-Schnittstellenfunktion „**SQLExecDirect**“ (aus ODBC V1.0) Daten aus der Datenbank ausliest und tabellarisch per Konsole ausgibt.



```
C:\Users\Alexander\documents\visual studio 2015\Projekte\...>
Record found! => <1>/<Joe>/<Maier-8>
Record found! => <2>/<Tina>/<Schmitt>
Record found! => <3>/<Anna>/<Moon>
Record found! => <5>/<Tim>/<Jupiter>
Record found! => <6>/<Lisa>/<Venus>
Record found! => <7>/<Peter>/<Francis>
Record found! => <8>/<KARL>/<MUELLER>
```

Tipps:

- Versuchen Sie zunächst durch geeignete Internet-Recherche zu einer Lösung zu kommen.
- Nutzen Sie hierzu **ODBC-Schnittstellenfunktionen**, wie etwa `SQLAllocHandle()`, `SQLSetEnvAttr()`, `SQLSetConnectAttr()`, `SQLConnect()`, `SQLExecDirect()`, `SQLFetch()`, `SQLGetData()`, `SQLFreeHandle()`, `SQLDisconnect()`.

2.18 REST / API finden

Ermitteln Sie ein kostenfreies RESTful API, das Sie über http ansprechen können. Prüfen Sie, ob das API wirklich RESTful ist.

2.19 REST / REST API - Abfrage programmieren

Programmieren Sie unter Verwendung einer Serverprogrammiersprache Ihrer Wahl eine Formularabfrage (Portal) an ein freies http-erreichbares RESTful API!

Hinweise:

- Falls Sie kein API finden können, so verwenden Sie ersatzweise folgendes:
<https://restcountries.com/>
Programmieren Sie hiermit eine Abfrage, die nach Eingabe eines Hauptstadtnamens im Formular das zugehörige Land ermittelt.
Beispiel:
Eingabe „Berlin“, Ausgabe „Germany“
- Wenn Sie sich für keine Serverprogrammiersprache entscheiden können, dann verwenden Sie einfach PHP.