Project 건물주

Team Activity @ 4RG0S

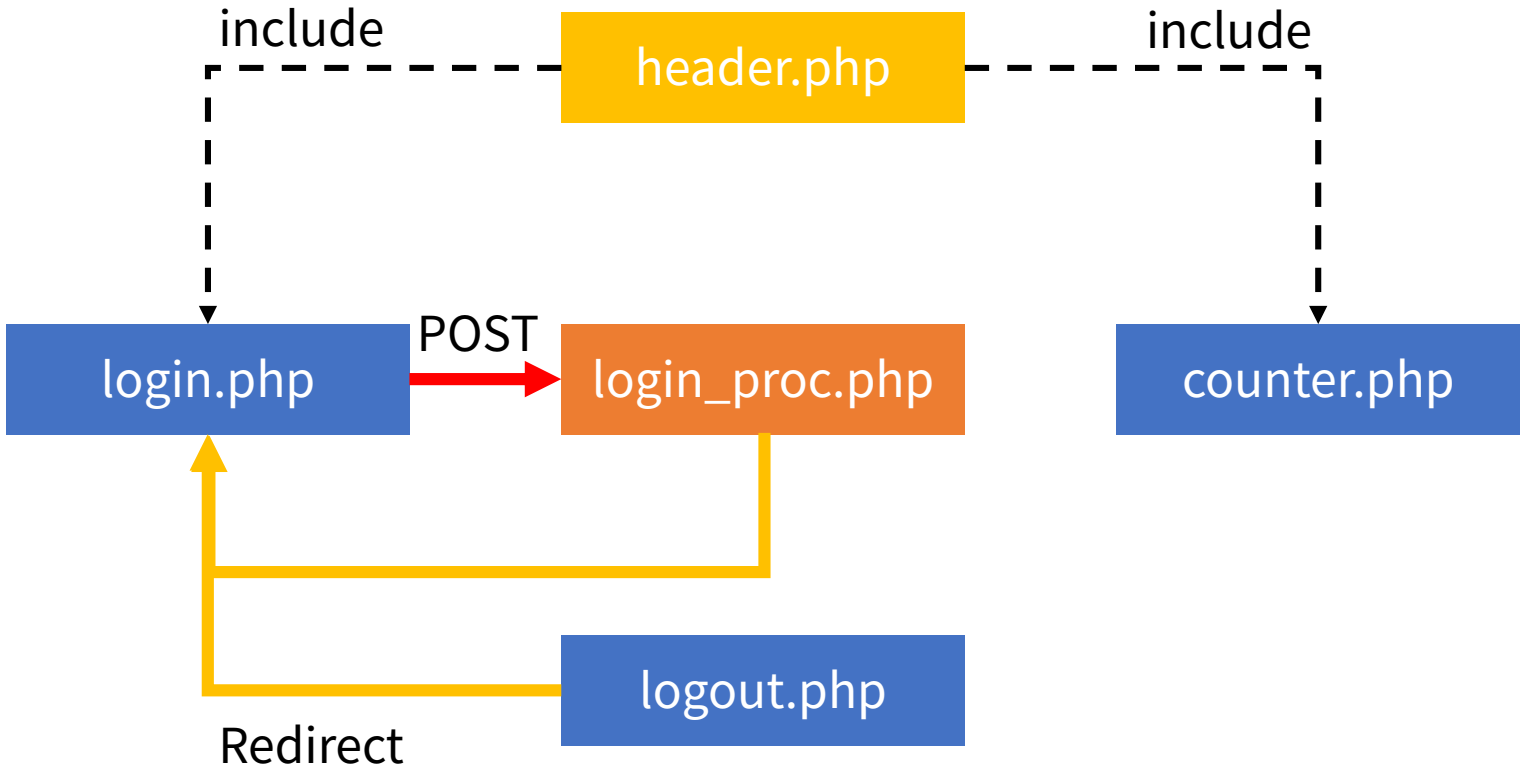
Week4 :: 데이터베이스와 SQL

CSE 17’ Kangjun Heo

Week3 :: Homework Review

세션을 이용한 간단 로그인 Application

https://201704150.islab.work/week3hw/login.php



생각해보기

로그아웃 하더라도 카운터를 유지할 순 없을까?

- 3주차 과제로 진행한 카운터 프로그램은 로그아웃하면 초기화 됨
- 세션에 저장하기 때문 – PHP 인터프리터가 동작할 때에만 유효
- 올려놓거나/내려놓은 카운터를 사용자 별로 유지해 볼 수는 없을까?

데이터베이스

Application의 영속성 보장을 위한 요소

- 영속성(Persistency) – 시스템의 Lifetime에 구애 받지 않는 데이터가 가지는 특성
  서버나 프로그램의 작동이 중지되더라도 사라지지 않음
- 파일이나 DBMS(Database Management System)를 이용함
- DB의 종류는 관계형 데이터베이스, 객체 데이터베이스가 있음
- 웹을 포함하여 수 많은 애플리케이션의 영속성은 관계형 데이터베이스를 통해 보장
  사용자 정보, 애플리케이션 데이터, 그 외 여러가지…
- “우리집”은 관계형 데이터베이스(MariaDB)를 이용하여 데이터를 저장

데이터베이스

많이 쓰이는 DBMS 들

- MySQL/MariaDB: 소규모 웹 시스템에 많이 사용, 주로 PHP와 많이 연동됨

- PostgreSQL: 좋은 성능, 높은 수준의 표준 구현, 최근 각광 받는 중

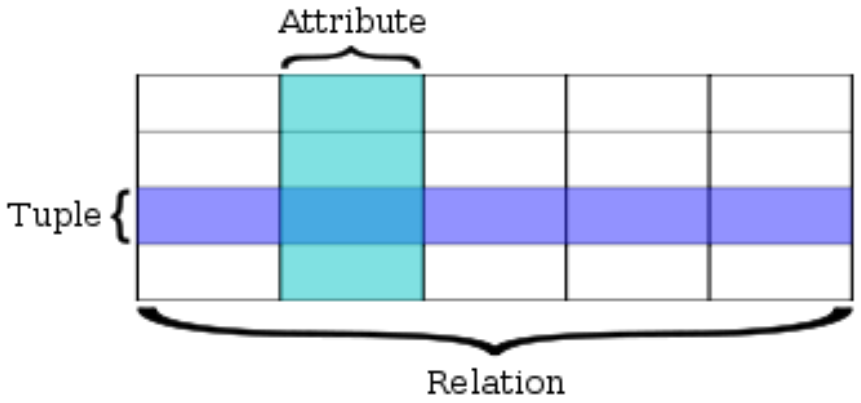
- Microsoft SQL Server: Microsoft에서 개발한 RDBMS, 주로 Windows 기반 서비스에서 연동됨

- Oracle: 전 세계에서 가장 많이 쓰이는 RDBMS, 가장 좋은 성능, 안정성 보장

데이터베이스

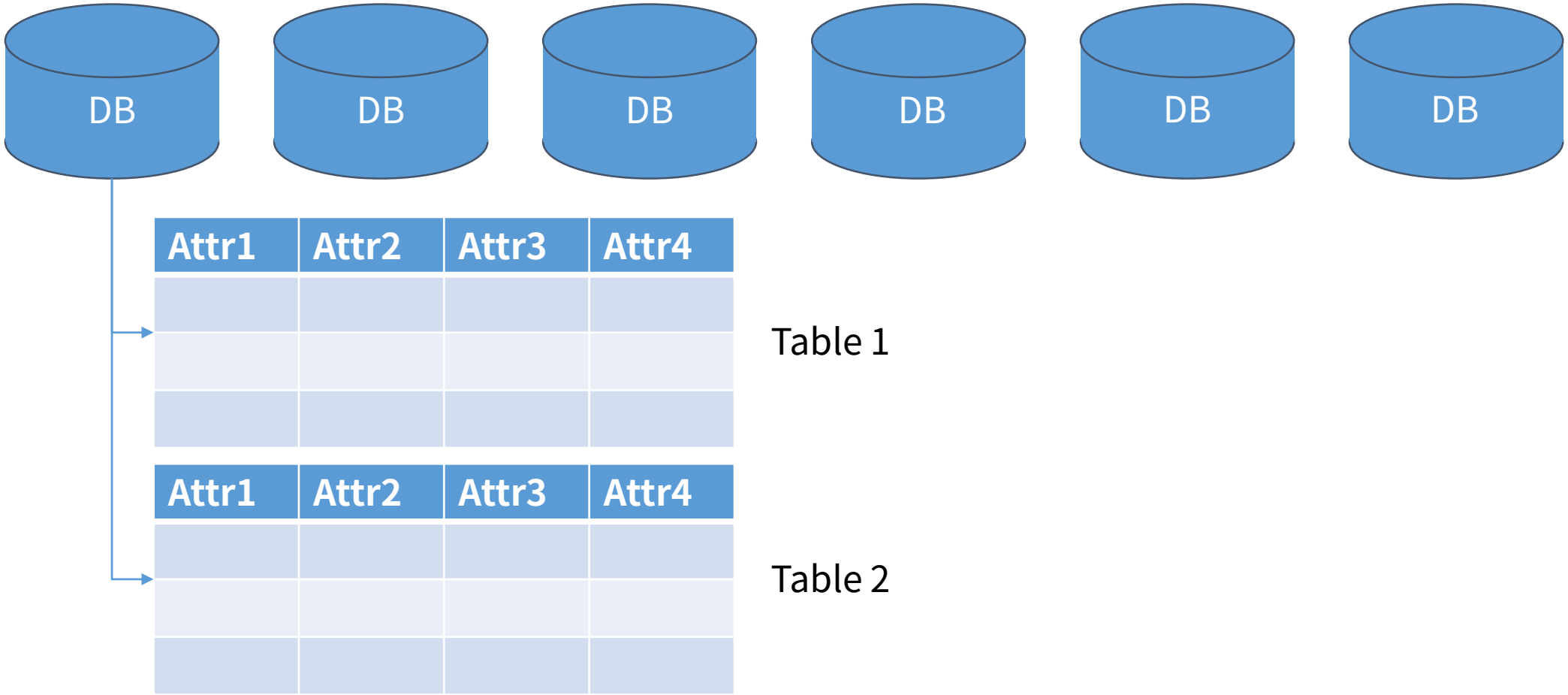
관계형 데이터베이스

- Relation, Attribute, Tuple 등으로 데이터를 구성하고 그 관계를 정의하는 데이터베이스
- Tuple: “Row” 혹은 “Record”라고도 부르며 하나의 데이터 항목을 표현하는 단위
  Table의 행에 해당
- Attribute: “Column” 혹은 “Field”라고 부르며 각 Tuple이
  가질 수 있는 속성, Table의 열에 해당
- Relation: “Table” 이라고도 부르며 같은 Attribute를
  가진 Tuple의 집합



데이터베이스

관계형 데이터베이스



Structured Query Language (SQL)

관계형 데이터베이스를 조작하는 언어

- DDL (Data Definition Language)
  데이터를 정의하는 언어
  CREATE, ALTER, DROP를 이용하여 테이블, DB를 생성, 수정, 삭제함

- DML (Data Manipulation Language)
  데이터를 조회/수정하는 언어
  SELECT, DELETE, UPDATE, INSERT를 이용하여 데이터를 조회, 수정, 삭제, 삽입함

- DCL (Data Control Language)
  데이터를 제어하는 언어
  GRANT, REVOKE를 이용하여 DBMS 내 객체들에 대한 권한을 제어함
  이번 활동에서는 다루지 않음

DDL: CREATE

테이블의 생성

```
CREATE TABLE  table_name (
    attribute_name        datatype PRIMARY KEY,
    attribute_name2       datatype NOT NULL,
    attribute_name3       datatype DEFAULT default_value,
    …,
    constraints…
);
```

- 테이블을 생성하는 SQL 구문, 각 데이터 타입 별로 Attribute를 구성함
- 많이 사용되는 데이터타입은 INT, VARCHAR(n), DATETIME
- PRIMARY KEY, NOT NULL, DEFAULT 등을 이용하여 각 Attribute의 제약사항, 기본값 등을 설정할 수 있음

DDL: DROP

테이블의 삭제

```
DROP TABLE table_name;
```

- 사용하지 않을 테이블을 삭제하려는 경우 DROP 문을 사용함

DML: SELECT

데이터의 조회

```
SELECT columns [FROM table_name [WHERE conditions]]
               [ORDER BY column <ASC/DESC>]
               [LIMIT offset, count]

SELECT id, password FROM user WHERE name='건물주';
```

- Attribute, Table을 지정한 후 조건에 따라 정렬, 오프셋 등을 지정하여 데이터를 조회
- SQL은 개행을 고려하지 않으므로 여러 줄에 걸쳐 써도 무방
- 조건은 AND, OR, 괄호 등을 이용하여 조합할 수 있음

DML: INSERT

데이터의 삽입

```
INSERT INTO table_name (column_list) VALUES (…)
```

- 테이블에 데이터를 삽입하는 구문
- column_list는 없을 경우 전체 column에 대해 삽입함
- column_list에 지정된 순서에 따라 VALUES에 지정된 Tuple이 삽입됨
- VALUES 뒤에 나오는 Tuple은 ,(콤마) 로 연결하여 여러 Tuple을 한번에 삽입할 수 있음

DML: UPDATE/DELETE

데이터의 수정과 삭제

UPDATE table_name SET column=value, column2=value2, …
[WHERE conditions]

- 테이블 내의 데이터를 수정하려고 하는 경우 UPDATE문을 사용함
- SET 뒤에 (수정하고자 하는 Attribute의 이름)=값을 ,로 연결하려 서술

DELETE from table_name [WHERE conditions]

- 테이블 내의 데이터를 삭제하고자 하는 경우 DELETE문을 사용
- **절대 주의!** UPDATE/DELETE문은 WHERE 절을 사용하지 않으면 **전체 데이터**에 영향을 줌!

PHP와 데이터베이스

PDO (PHP Data Objects)를 이용한 PHP의 DB 연동

- PHP에서 MySQL/MariaDB 를 사용하기 위해서는…
  mysql_ 시리즈 함수들 (보안문제, 사용하면 Error 발생!)
  mysqli_ 시리즈 함수들 (사용 가능하지만 곧 삭제된다는 얘기가 있음)

- PHP에서 다른 DBMS를 사용하기 위해서는…
  거기에 맞는 extension을 설치하고 API를 공부해야함

- 따라서 표준화 된 인터페이스를 제공하는 PDO를 사용하는 것이 권장됨
  필요한 드라이버만 추가하면 같은 코드를 재활용 할 수 있음

PHP와 데이터베이스

PDO (PHP Data Objects)를 이용한 PHP의 DB 연동

- 데이터베이스에 접속하기 위한 필요한 정보들: 호스트, 사용자명, 비밀번호, DB 이름

- 호스트: DBMS가 구동되고 있는 서버 (때로는 포트번호도 필요, 그러나 대부분 불필요)
- 사용자명: DBMS에서 사용할 사용자 이름
- 비밀번호: DBMS에서 사용할 비밀번호
- DB 이름: 데이터 CRUD를 수행할 DB의 이름

- PDO는 DBMS에 연결할 때 DSN(Data Source Name)를 사용함
- DSN에는 호스트, DB이름이 명시되어야 함

PHP와 데이터베이스

PDO (PHP Data Objects)를 이용한 PHP의 DB 연동

```php
<?
  // DB 접속 정보
  $host     = "localhost";
  $username = "201704150";
  $password = "password";
  $dbname   = "db_201704150";

  // PDO 객체 생성
  $dsn  = "mysql:dbname=$dbname;host=$host;charset=utf8";
  $conn = new PDO($dsn, $username, $password);

  // PDO 옵션 설정
  // 여기선 에러 모드를 Exception으로 설정함
  $conn->setAttribute(PDO::ATTR_ERRMODE, PDO_ERRMODE_EXCEPTION);

  // 쿼리 실행
  $stmt = $conn->prepare("SELECT * FROM Woorizip_Users");
  $stmt->execute();

  // SELECT 문으로 조회한 데이터 가져오기
  $data = $stmt->fetchAll(PDO::FETCH_ASSOC);
```

4주차 도전과제: DB와 연동된 로그인 App

3주차로 부터 이어지는 업그레이드 과제

- 3주차에서 만든 로그인 프로그램을 데이터베이스와 연동
- 사용할 사용자의 스키마는 다음 슬라이드에

- 데이터베이스 접속 서버: localhost, 아이디: 본인 학번, 비밀번호: 공지한 대로

- 가능한 기존에 만든 프로그램을 재사용 해볼 것
- 추가 도전과제: 3주차 카운터 프로그램을 구현했다면 사용자 별로 카운트 값을 저장해볼 것!

4주차 도전과제: DB와 연동된 로그인 App

3주차로 부터 이어지는 업그레이드 과제

```
CREATE TABLE Woorizip_Users (
      user_id         VARCHAR(32)    PRIMARY KEY,
      password        VARCHAR(64)    NOT NULL,
      user_name       VARCHAR(32)    NOT NULL
);
```

EOF