## Задача №1

Имеется база со следующими таблицами:

CREATE TABLE `users` (

`id` INT(11) NOT NULL AUTO\_INCREMENT,

`name` VARCHAR(255) DEFAULT NULL,

`gender` INT(11) NOT NULL COMMENT '0 - не указан, 1 - мужчина, 2 - женщина.',

`birth\_date` INT(11) NOT NULL COMMENT 'Дата в unixtime.',

PRIMARY KEY (`id`)

);

CREATE TABLE `phone\_numbers` (

`id` INT(11) NOT NULL AUTO\_INCREMENT,

`user\_id` INT(11) NOT NULL,

`phone` VARCHAR(255) DEFAULT NULL,

PRIMARY KEY (`id`)

);

Оптимизируйте структуру таблиц и напишите запрос, возвращающий имена и количества телефонных номеров девушек в возрасте от 18 до 22 лет.

Файлы create\_tables.sql, request.sql

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `users` (

`id` **int** **unsigned** **NOT** **NULL** **AUTO\_INCREMENT**,

`name` **varchar**(255) **DEFAULT** **NULL**,

`gender` **tinyint** **NOT** **NULL** **COMMENT** '0 - не указан, 1 - мужчина, 2 - женщина.',

`birth\_date` **int** **unsigned** **NOT** **NULL** **COMMENT** 'Дата в unixtime.',

**PRIMARY** **KEY** (`id`),

**KEY** `gender\_birth\_date` (`gender`,`birth\_date`)

) **ENGINE**=**InnoDB** **AUTO\_INCREMENT**=4 **DEFAULT** **CHARSET**=**utf8mb4** **COLLATE**=utf8mb4\_0900\_ai\_ci;

**CREATE** **TABLE** **IF** **NOT** **EXISTS** `phone\_numbers` (

`id` **int** **unsigned** **NOT** **NULL** **AUTO\_INCREMENT**,

`user\_id` **int** **unsigned** **NOT** **NULL**,

`phone` **varchar**(255) **DEFAULT** **NULL**,

**PRIMARY** **KEY** (`id`),

**KEY** `user\_id` (`user\_id`)

) **ENGINE**=**InnoDB** **AUTO\_INCREMENT**=9 **DEFAULT** **CHARSET**=**utf8mb4** **COLLATE**=utf8mb4\_0900\_ai\_ci;

**Запрос**

**SELECT**

u.**name** **AS** user\_name,

**COUNT**(p.phone) **AS** phone\_count

**FROM**

users u

**LEFT JOIN**

phone\_numbers p **ON** u.id = p.user\_id

**WHERE**

u.gender = '2'

**AND** **TIMESTAMPDIFF**(**YEAR**, **FROM\_UNIXTIME**(u.birth\_date), **CURDATE**()) **BETWEEN** 18 **AND** 22

**GROUP** **BY**

u.id, u.**name**;

## Задача №2

Напишите код в парадигме ООП, соответствующий следующей структуре.

**Сущности:** Пользователь, Статья.

**Связи:** Один пользователь может написать несколько статей. У каждой статьи может быть только один пользователь-автор.

**Функциональность:**

* возможность для пользователя создать новую статью;
* возможность получить автора статьи;
* возможность получить все статьи конкретного пользователя;
* возможность сменить автора статьи.

Если вы применили какие-либо паттерны при написании, укажите какие и с какой целью.

Код, реализующий конкретную функциональность, не требуется, только общая структура классов и методов. Код должен быть прокомментирован в стиле PHPDoc.

## Использовал паттерн репозиторий для лучшей структурированности кода и абстрагирования от источника данных. В класс ArticleRepository можно было и другие методы, но задача была сильно абстрактная.

## Файл для демонстрации classes-pattern.php

## Задача №3

Проведите рефакторинг, исправьте баги и продокументируйте в стиле PHPDoc код, приведённый ниже (таблица users здесь аналогична таблице users из задачи №1).

*Примечание: код написан исключительно в тестовых целях, это не "жизненный пример" :)*

function load\_users\_data($user\_ids) {

$user\_ids = explode(',', $user\_ids);

foreach ($user\_ids as $user\_id) {

$db = mysqli\_connect("localhost", "root", "123123", "database");

$sql = mysqli\_query($db, "SELECT \* FROM users WHERE id=$user\_id");

while($obj = $sql->fetch\_object()){

$data[$user\_id] = $obj->name;

}

mysqli\_close($db);

}

return $data;

}

// Как правило, в $\_GET['user\_ids'] должна приходить строка с номерами пользователей через запятую, например: 1,2,17,48

$data = load\_users\_data($\_GET['user\_ids']);

foreach ($data as $user\_id=>$name) {

echo "<a href=\"/show\_user.php?id=$user\_id\">$name</a>";

}

Плюсом будет, если укажете, какие именно уязвимости присутствуют в исходном варианте (если таковые, на ваш взгляд, имеются), и приведите примеры их проявления.

Уязвимости и проблемы:

* SQL Инъекции: SQL-запрос без использования параметризованных запросов
* Соединение с бд в цикле - нерационально
* Отсутствие обработки ошибки при соединении с БД
* Нет проверки данных в $\_GET['user\_ids']

Файл task3.php. От параметризованных запросов я отказался, т.к. массив фильтруется.