Тестовое задание GorodRabot.ru

Задача #1. MySQL

a)

```
---
--- Структура таблицы `shop`
---
CREATE TABLE IF NOT EXISTS `shop` (
`id` int(11) NOT NULL AUTO_INCREMENT,
`article` char(10) NOT NULL,
`dealer` char(10) NOT NULL,
`price` float NOT NULL,
PRIMARY KEY (`id`))CHARSET=utf8 AUTO_INCREMENT=1;

---
--- Дамп данных таблицы `shop`
---
INSERT INTO `shop` (`id`, `article`, `dealer`, `price`) VALUES
(1, '0001', 'B', 3.99),
(2, '0002', 'A', 10.99),
(3, '0003', 'C', 1.69),
(4, '0004', 'D', 19.95),
(5, '0005', 'E', 21.05),
(6, '0001', 'A', 3.32),
(7, '0002', 'D', 10.99);
```

Задание:
Для всех артикулов выведите поставщика или поставщиков (артикулы и поставшики на выводе могут повторяться) с самой высокой ценой.

b)

```
---
--- Структура таблицы `user`
---
CREATE TABLE user(
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    email VARCHAR(255) NOT NULL,
    name VARCHAR(255),
    created_at DATETIME NOT NULL
);

---
--- Структура таблицы `order`
---
CREATE TABLE order (
    id BIGINT PRIMARY KEY AUTO_INCREMENT,
    user_id INT NOT NULL,
    total DECIMAL(10,2) NOT NULL,
    status ENUM('new', 'paid', 'cancelled') NOT NULL,
    created_at DATETIME NOT NULL
);
```

Связь между таблицами user.id -> order.user_id

Задание:
Напишите запрос, который вернёт количество оплаченных товаров больше 100 по каждому клиенту за вчера.
Создайте индекс(ы), который(ые) ускорит(ят) этот запрос.

Напишите запрос, который вернёт email клиентов, которые не оплатили вчера месяц назад и чьи аккаунты были созданы 2 недели назад.
Создайте индекс(ы), который(ые) ускорит(ят) этот запрос.

Обоснуйте выбор индекса(ов) для каждого случая.
Не нашли ли вы ошибок при создании таблиц для использования созданных вами индексов?

Задача #2. PHP

Необходимо получить информацию о районе города, ближайшим станция метро, улице и дому по адресу, введённого пользователем в текстовом поле.
Пользователь может ввести всё что угодно, но интересны только адреса Москвы.
Результат выполнения выводить на этой же странице в любом читаемом виде.
Если результатов несколько, то выводить первые 5.
В качестве API для геокодирования можно использовать, например, бесплатную версию https://dadata.ru/. Либо любой другой сервис.
Приложение должно сохранять запросы пользователя (только уникальные) в БД (MySQL).
В качестве фреймворка для приложения разрешается использовать Slim. Либо вы можете использовать «чистый» PHP.
Для быстрого развёртывания ожидается увидеть файл docker-compose.yml с необходимыми сервисами.
Итоговой код можно разместить в репозитории и прислать ссылку на репозиторий.

Задача #3. Manticoresearch

Для поискового движка manticoresearch добавьте БД и plain-индекс vacancy, который будет собираться на основе таблицы MySQL с вымышленными данными.
Поля таблиц Manticoresearch: title, salary_from, salary_to, country_id, region_id, city_id, created_at.
Язык морфологии: русский.
Допускается не использовать wordforms и exceptions в конфигурационном файле создания plain-индекса.
Для быстрого развёртывания ожидается увидеть файл docker-compose.yml с необходимыми сервисами.