Ответы на вопросы:

1.

<?php

// хранение логина\пароля и остальных credentials из кода необходимо перенести

// в env файлы или вообще спрятать в специализированное хранилище, из которого

// доставать при сборке приложения для деплоя. env-файлы хороши в локальной

// разработке

$mysqli = new mysqli("localhost", "root", "", "eggheads");

// надо сделать приведение типа, если мы ждем int - надо переделать в int

// так же необходима валидация входящих параметров, чтобы в "холостую" не

// гонять скрипт дальше, если параметры не подходят

// плюс первая защита от SQL-инъекций

$id = (int) $\_GET[‘id’];

if ($id <= 0) {

die('id has not corrected value');

}

// напрямую подставлять данные даже после приведения типа выше плохая идея - надо

// использовать биндинг + если мы точно знаем, что запрос должен вернуть 1 запись,

// то надо добавить в него LIMIT

$stmt = $mysqli->prepare('SELECT \* FROM users WHERE id = ? LIMIT 1');

$stmt->bind\_param('i', $id);

$stmt->execute();

$res = $stmt->get\_result();

// необходима проверка на количество вернувшихся строк из БД

if (!$res->num\_rows) {

die('id not found');

}

$user = $res->fetch\_assoc();

var\_dump($user);

2.

возможно два варианта рефакторинга, всё зависит от размеров таблиц и передаваемых данных. В общем случае тут решение “проблемы N+1”, и решение (любое) будет лежать на стороне приложения, а не в БД.

2.1

Подводный камень в решении имеется в количестве user\_id которые надо подставить в where in. Можно достичь такого количества, что не пролезем в длину строки запроса.

// preparing

$catId = 1; // from somewhere

$result = [];

// getting all questions

$questionsQ = $mysqli->prepare('SELECT \* FROM questions WHERE catalog\_id = ?');

$questionsQ->bind\_param('i', $catId);

$questionsQ->execute();

// working with questions

$questions = $questionsQ->get\_result()->fetch\_all(MYSQLI\_ASSOC);

$questionsQ->free\_result();

// getting all question Ids in one array

$questionsIds = array\_map(fn($item) => $item['user\_id'], $questions);

$questionsIdsCnt = count($questionsIds);

if ($questionsIdsCnt > 0) {

// preparing for users

$placeholders = implode(',', array\_fill(0, $questionsIdsCnt, '?'));

$bindStr = str\_repeat('i', $questionsIdsCnt);

// getting users

$userQ = $mysqli->prepare("SELECT **id**, name, gender FROM users WHERE id IN ($placeholders)");

$userQ->bind\_param($bindStr, ...$questionsIds);

$userQ->execute();

// working with users

$users = $userQ->get\_result()->fetch\_all(MYSQLI\_ASSOC);

$userQ->free\_result();

// working with users and questions

foreach ($questions as $index => $question) {

$result[] = [

'question' => $question,

'user' => $users[array\_search($question['user\_id'], array\_column($users, 'id'))],

];

}

}

echo '<pre>';

print\_r($result);

echo '</pre>';

2.2

выбираем все возможные данные из таблиц одним запросом и отделяем вопросы от пользователей средствами приложения. Подводный камень в количестве возвращаемых данных, их может быть слишком много, в этом случае возможна временная задержка при передаче данных

// preparing

$catId = 1;

$result = [];

// getting all data (questions and users)

$allDataQ = $mysqli->prepare('SELECT q.\*, u.name as u\_name, u.gender as u\_gender FROM questions AS q INNER JOIN users as u ON u.id = q.user\_id WHERE q.catalog\_id = ?');

$allDataQ->bind\_param('i', $catId);

$allDataQ->execute();

// working with data

$allData = $allDataQ->get\_result()->fetch\_all(MYSQLI\_ASSOC);

$allDataQ->free\_result();

foreach ($allData as $index => $dataItem) {

// subtract user from data

$user = [

'name' => $dataItem['u\_name'],

'gender' => $dataItem[‘u\_gender'],

];

// unset unnecessary fields

unset($dataItem['u\_name'], $dataItem['gender']);

// combining result

$result[] = [

'question' => $dataItem,

'user' => $user,

];

}

echo '<pre>';

print\_r($result);

echo '</pre>';

3.

вариант 3.1: все пользователи, и с заказами и нет даже не сделавшие ни одного заказа

select u.name,

u.phone,

sum(o.subtotal) as total,

avg(o.subtotal) as average,

max(o.created) as last\_order\_date

from users as u

left join orders o on u.id = o.user\_id

group by u.name, u.phone;

вариант 3.1.1: чтобы не разбираться с null для пользователей без заказов на стороне приложения можно использовать функцию coalesce()

select u.name,

u.phone,

coalesce(sum(o.subtotal), 0) as total,

coalesce(avg(o.subtotal), 0) as average,

coalesce(max(o.created), 'no order date') as last\_order\_date

from users as u

left join orders o on u.id = o.user\_id

group by u.name, u.phone;

вариант 3.2: только пользователи имеющие хотя бы один заказ

select u.name,

u.phone,

coalesce(sum(o.subtotal), 0) as total,

coalesce(avg(o.subtotal), 0) as average,

coalesce(max(o.created), ‘no\_order\_date’) as last\_order\_date

from users as u

inner join orders o on u.id = o.user\_id

group by u.name, u.phone;

4.

честно говоря - делал “на ощупь”, я и фронтенд дело далёкое )) понимаю общий смысл и контекст, но не имею на нём практически никакого опыта. Могу описать свои действия:

1. загоняю всё это в файл html и создаю json-строку исходя из того что это должен быть массив (потому что foreach), и внутри сразу объект который должен иметь поле price. на всякий случай добавляю name, просто чтобы смотрелось хорошо )
2. вижу, что вместо сравнения используется присваивание - исправляем = на ===
3. вижу, что считается orderSubtotal но в результате идет обращение к несуществующей переменной total, объявляю var total; меняю orderSubtotal на total, проверяю - ошибок нет, но не работает. добавляю к total ноль как значение по-умолчанию, просто… а почему бы и нет - работает.
4. меняю логику работы проверки - если !== undefined только тогда делать подсчёт - становится чуть легче понимать код и не делается лишних созданий переменных, парсеру будет легче (наверное)
5. вижу непонятку в консоль.логе, меняю местами “бесплатно” и показ количества рублей, но не работает - вечно “бесплатно”. понимаю, что скорее всего не хватает скобок, потому что мы работаем в функции вывода и конкантенируем при этом, ну… возможно это не красиво, но не запрещено - добавляю скобки - вуаля - вижу цифры и руб, но исчезло “бесплатно” и нет строки до. окей, добавляю общие скобки - вуаля. jobs done.
6. вижу IDE ругается на var. варианты на замену (const и let). окей, лезу гуглить чем всё это друг от друга отличается, понимаю, что лично в текущем случае ничего страшного нет ни в одном ни в другом случае, меняю на let, просто чтобы не ругался, однако узнал что-то новое для себя ))

function printOrderTotal(responseString) {

let responseJSON = JSON.parse(responseString);

let total = 0;

responseJSON.forEach(function (item, index) {

if (item.price !== undefined) {

total += item.price;

}

});

console.log('Стоимость заказа: ' + ((total > 0) ? total + ' руб.' : 'Бесплатно'));

}

printOrderTotal('[{"name": "first", "price": 1000}, {"name":"second", "price": 500}]');

printOrderTotal('[{"name": "first", "price": 0}, {"name":"second", "price": 0}]');

printOrderTotal('[{"name": "first", "price": 10}, {"name":"second"}]');