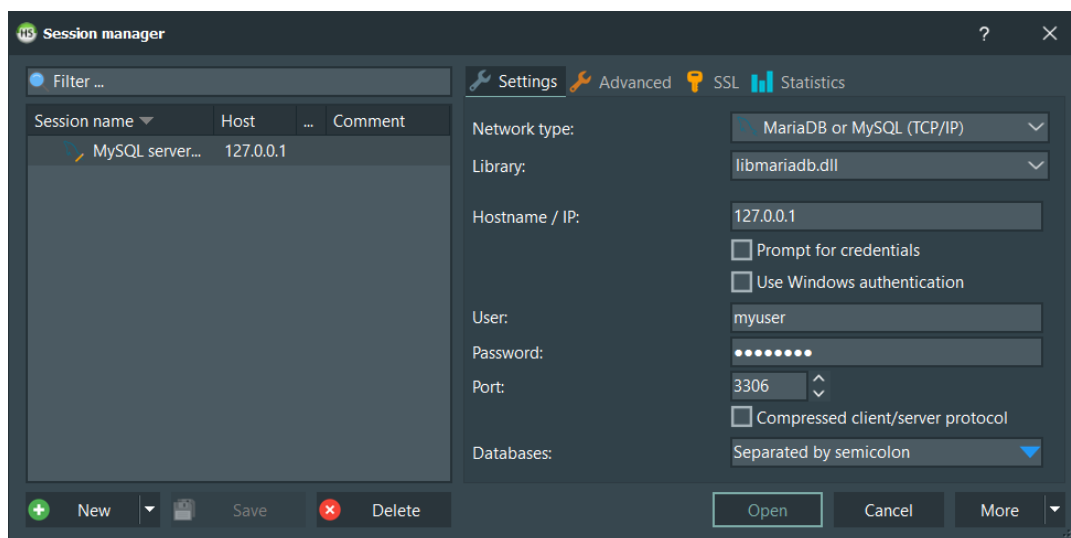


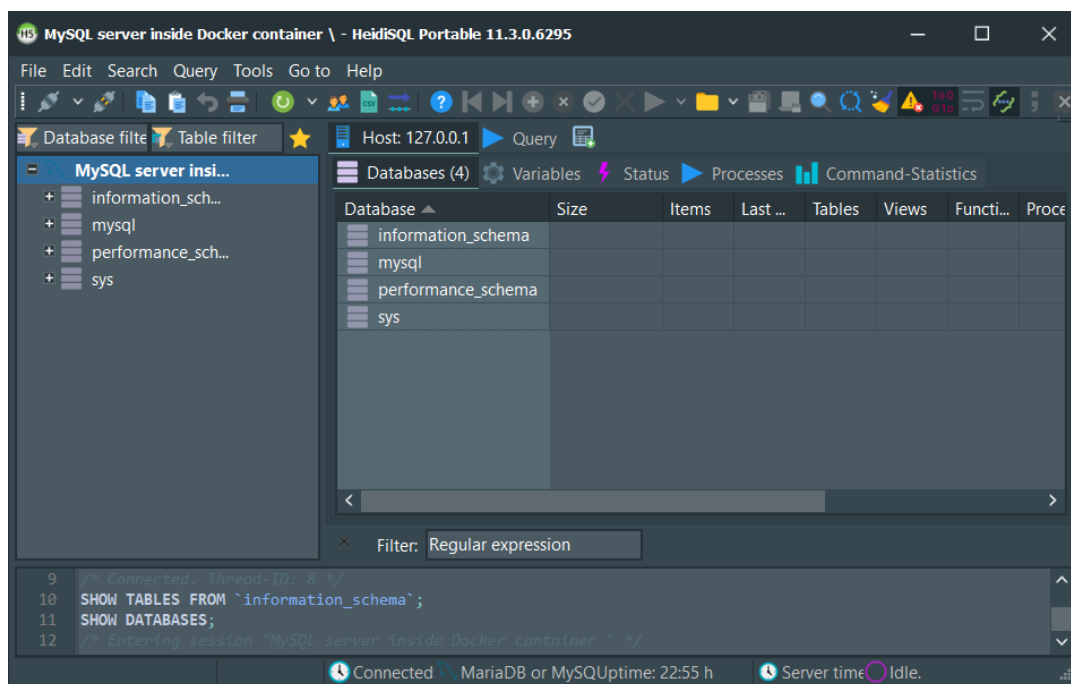
Самостоятельная работа №9.2. Работа с СУБД MySQL (Продвинутый уровень)

Соединение с базой данных посредством HeidiSQL

В самостоятельной работе №9.1 была произведена установка и настройка БД MySQL, создан пользователь для успешного дистанционного подключения к серверу БД MySQL, а в инструменте для управления базами данных HeidiSQL было создано подключение для доступа к серверу MySQL:

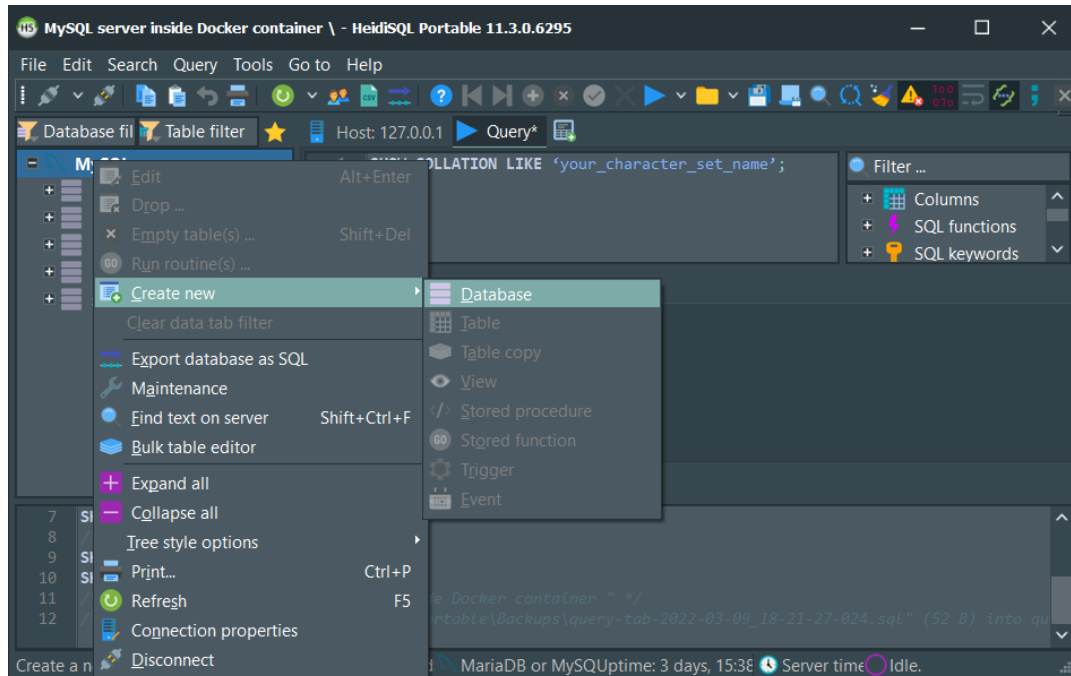


После нажатия на кнопку «Open» («Открыть») осуществляется соединение с базой данных:



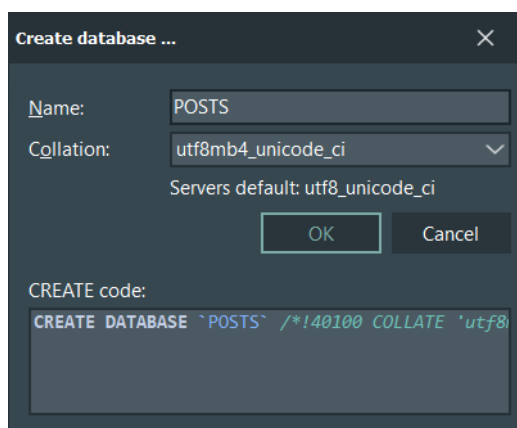
Создание базы данных

В левой части главного окна программы выбирается установленное соединение и нажимается ПКМ (правая кнопка мыши), в появившемся контекстном меню выбирается «Create new» -> «Database» («Создать» -> «База данных»):



В поля ввода открывшегося диалогового окна «Create database» («Создать базу данных») вносятся следующие данные:

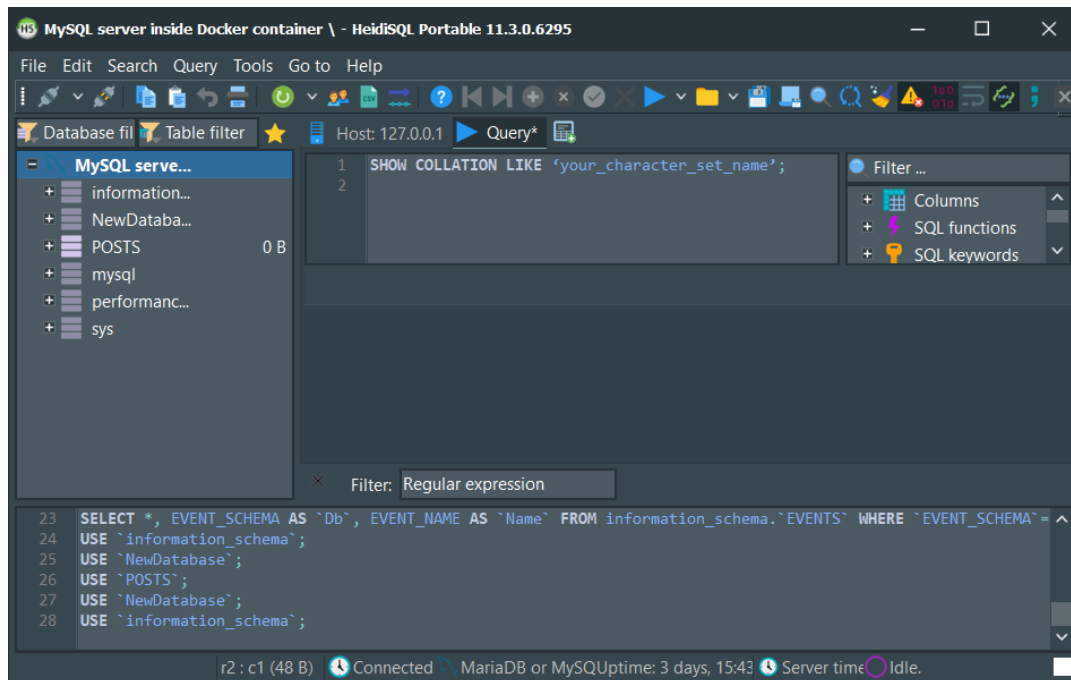
- **Name (Имя):** POSTS.
- **Collation (Сопоставление):** utf8mb4_unicode_ci.



Выбранные параметры формируют следующие команды SQL, создающие БД POSTS:

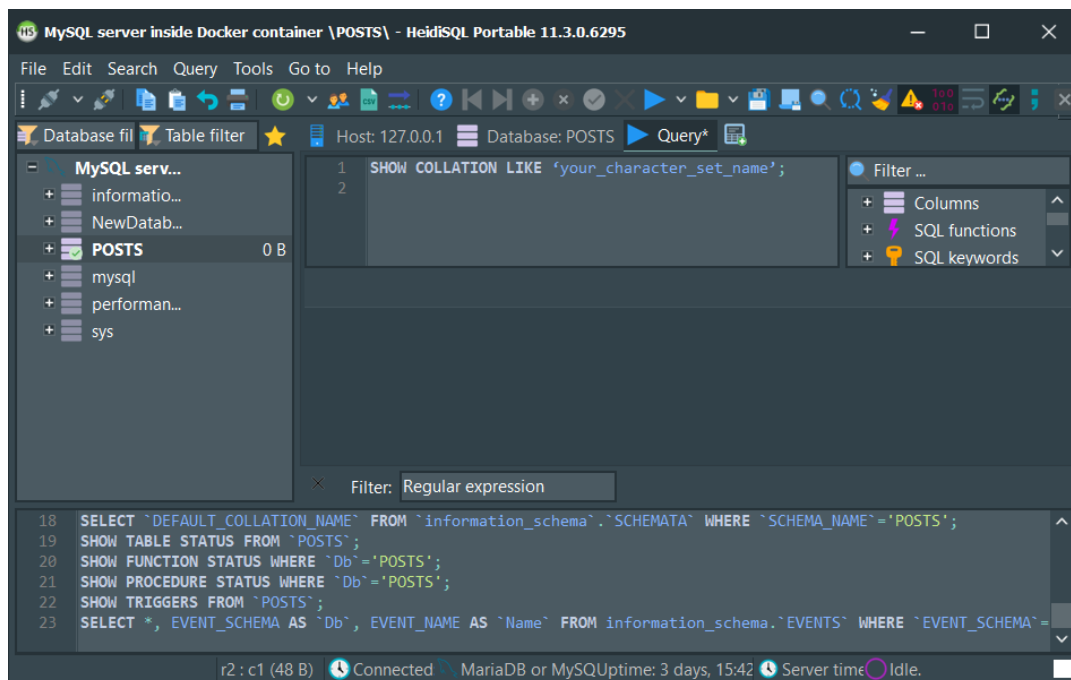
```
CREATE DATABASE IF NOT EXISTS `POSTS` /*!40100 COLLATE 'utf8mb4_unicode_ci' */
```

После нажатия кнопки «ОК», в левой части главного окна программы появляется новая БД POSTS:

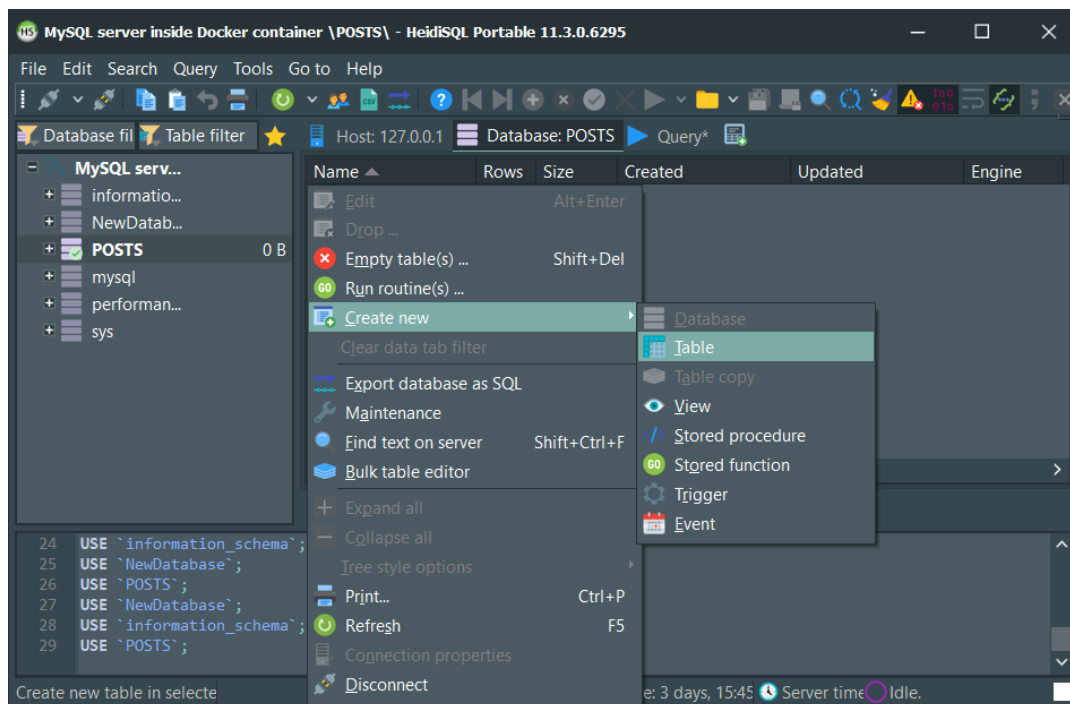


Создание новой таблицы

После нажатия ЛКМ (левой кнопки мыши) по названию БД POSTS, в правой части экрана откроется новая вкладка «Database: POSTS» («База данных: POSTS»):



Осуществляется переход в эту вкладку, на пустом месте в этой вкладке нажимается ПКМ (правая кнопка мыши), в появившемся контекстном меню выбирается «Create new» -> «Table» («Создать» -> «Таблица»).



В появившейся вкладке «Table» («Таблица») заполняются следующие поля ввода:

Name (Имя): last_news.

Comment (Комментарий): Таблица с последними новостями сайта.

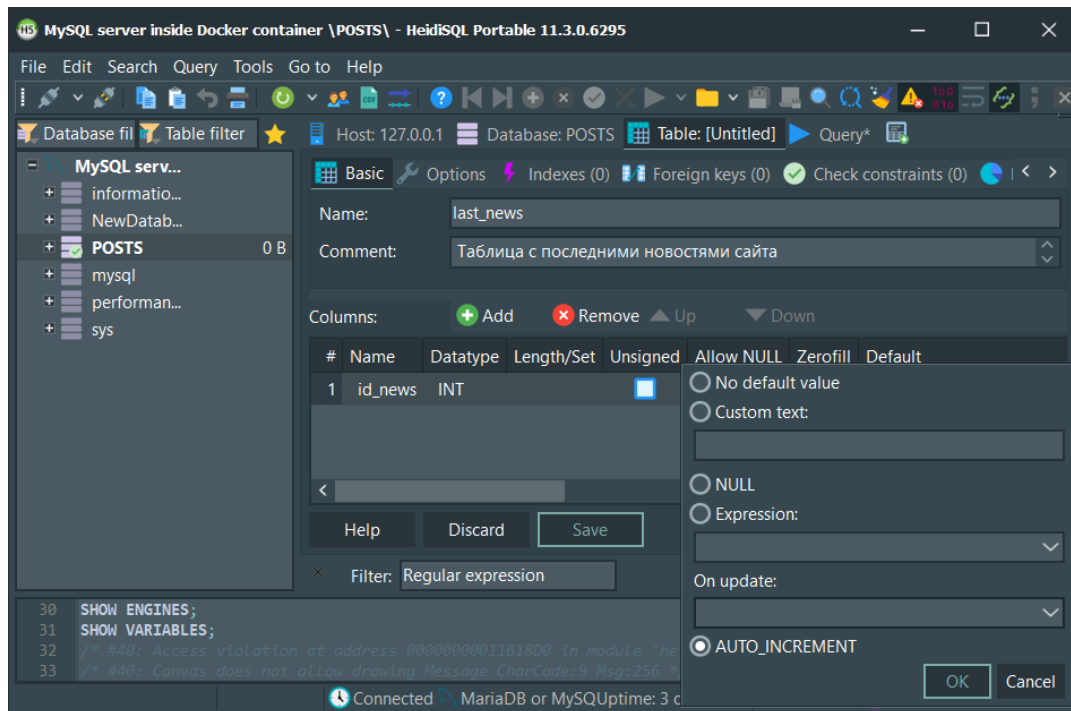
Столбец — это поле, предназначенное для хранения определенной информации о каждой записи в таблице. Для того, чтобы создать автоинкрементный столбец (column) в этой таблице, нажимается кнопка «Add» («Добавить»), затем указываются следующие параметры столбца:

- **Name (Имя):** id_news.
- **Datatype (Тип данных):** INT.
- **Default (По умолчанию):** AUTO_INCREMENT.

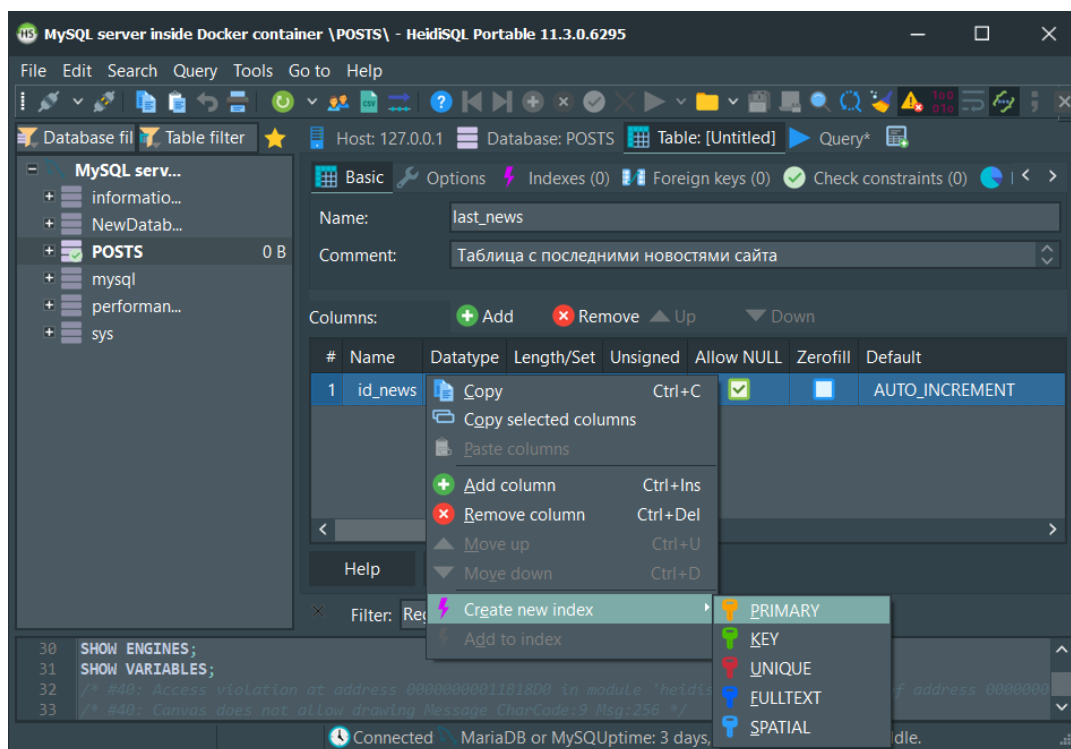
Первичный ключ (primary key) представляет собой один из примеров уникальных индексов и применяется для однозначной идентификации записей таблицы. Никакие из двух записей таблицы не могут иметь одинаковых значений первичного ключа. В реляционных базах данных практически всегда разные таблицы логически связаны друг с другом. Именно первичные ключи используются для однозначной организации такой связи.

Атрибут AUTO_INCREMENT позволяет автоматически генерировать уникальный ключ, если его тип является целочисленным. При вставке записи в базу данных

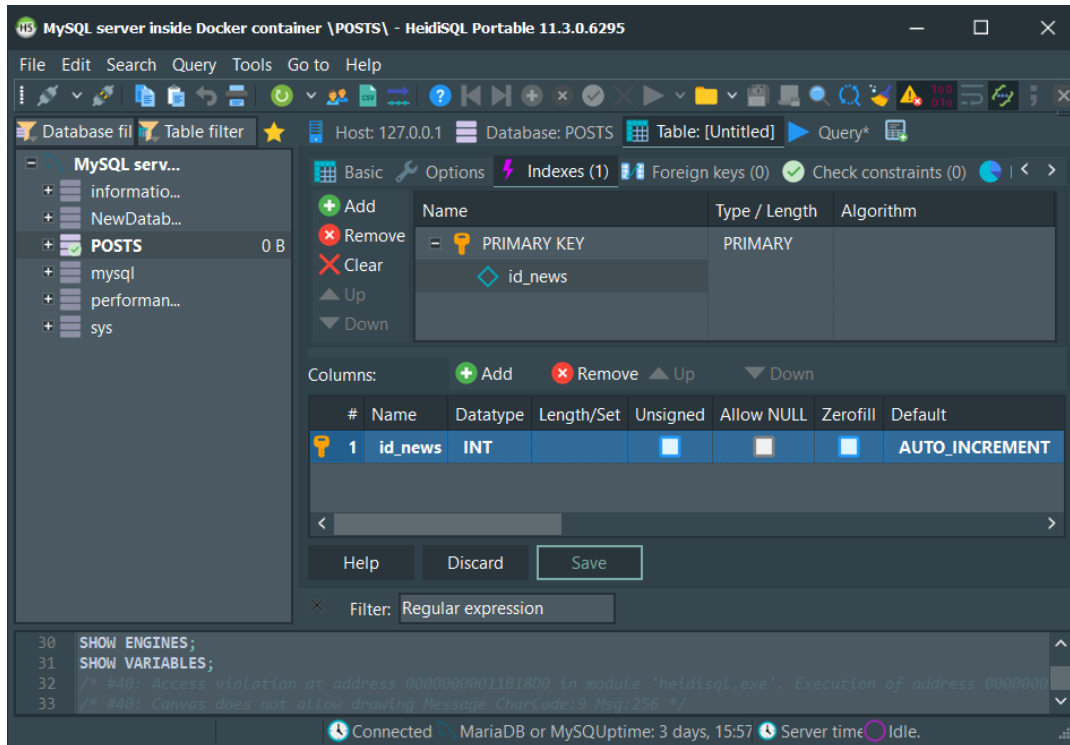
значение ключа выставляется равным нулю, MySQL автоматически вычисляет максимальный номер первичного ключа, увеличивает его на единицу и присваивает это значение первичному ключу новой записи.



Для того, чтобы сделать столбец id_news первичным ключом, на его названии нажимается ПКМ и выбирается «Create new index» («Создать новый индекс») -> «PRIMARY»:



После успешного создания уникального индекса рядом с названием столбца отобразится иконка ключа, а во вкладке «Indexes» («Индексы») появится его название:



Добавляются ещё три столбца с типом данных **VARCHAR** (строковый):

- **title** — заголовок новости;
- **author** — автор;
- **postcard** — путь к файлу иллюстрации.

Предпоследний столбец имеет тип данных **LONGTEXT** (столбец TEXT с максимальной длиной в 4,294,967,295 или 4GB ($2^{32}-1$) символов. Эффективная максимальная длина меньше, если значение содержит мультбайтные символы):

- **content** — текстовое содержимое новости.

Для сохранения внесенных в таблицу изменений нажимается кнопка «Save» («Сохранить»).

В результате формируются следующие команды SQL, создающие таблицу last_news:

```
USE `POSTS` ;

CREATE TABLE IF NOT EXISTS `last_news` (
  `id_news` int(11) NOT NULL AUTO_INCREMENT,
```

```

`title` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT
NULL,

`author` varchar(150) COLLATE utf8mb4_unicode_ci DEFAULT
NULL,

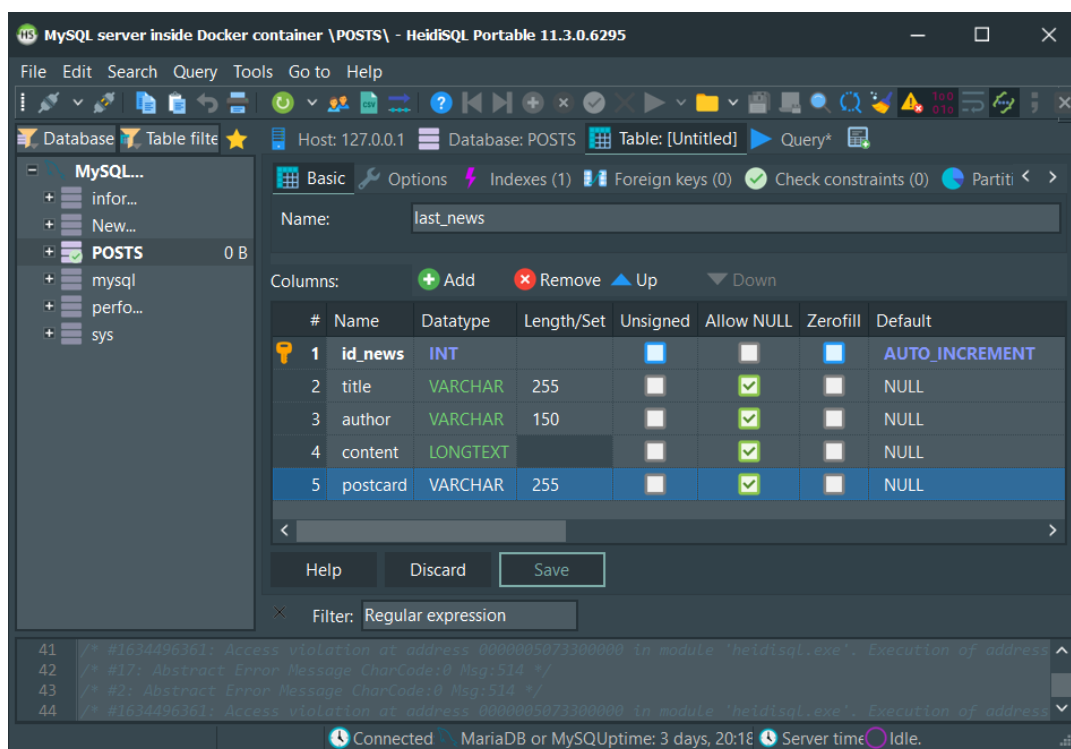
`content` longtext COLLATE utf8mb4_unicode_ci,

`postcard` varchar(255) COLLATE utf8mb4_unicode_ci
DEFAULT NULL,

PRIMARY KEY (`id_news`)

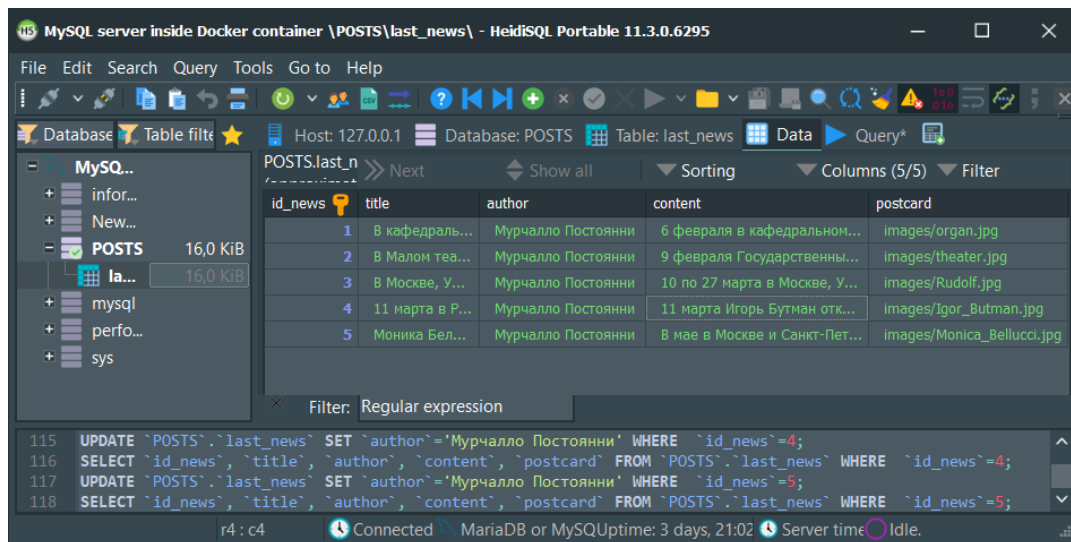
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci COMMENT='Таблица с последними
новостями сайта';

```



Добавление записей в таблицу БД

Открывается вкладка «Data» («Данные») и в верхнем меню нажимается кнопка «Insert row into table» («Вставьте строку в таблицу»), представляющая из себя белый знак «+» на фоне зелёного круга, затем в таблицу вносятся данные новой записи. Материалы (включая изображения) для записей взяты с сайта <https://www.culture.ru>. В столбце postcard указывается реальный путь к файлам изображений в будущем проекте. Всего в таблицу last_news вносится пять записей:



В результате формируются следующие команды SQL, вставляющие пять новых записей в таблицу last_news:

```
INSERT INTO `last_news` (`id_news`, `title`, `author`,
`content`, `postcard`) VALUES
```

```
(1, 'В кафедральном соборе Святых Петра и Павла...',
'images/organ.jpg'),
```

```
(2, 'В Малом театре состоится...',
'images/theater.jpg'),
```

```
(3, 'В Москве, Уфе и Казани пройдет...',
'images/Rudolf.jpg'),
```

```
(4, '11 марта в России открывается...',
'images/Igor_Butman.jpg'),
```

```
(5, 'Моника Беллуччи представит...',
'images/Monica_Bellucci.jpg');
```

Создание программы на языке PHP

Установка интерпретатора PHP 7.2 и связывание его с веб-сервером Apache 2 были выполнены в самостоятельной работе №1. Установка и настройка MySQL были произведены в самостоятельной работе №9.1.

Для входа в контейнер выполняется команда, запускающая в контейнере оболочку bash:

```
docker exec -it php_apache /bin/bash
```


Вход внутрь контейнера выполняется с использованием его имени, а не идентификатора (id). Внутри контейнера выполняется переход в директорию Apache 2:

```
cd /var/www/html/
```

Создание в ней поддиректории journal:

```
mkdir journal
```

Установка для этой дирекции полных прав доступа (каждый пользователь может читать, редактировать и запускать на выполнение):

```
chmod -R 777 journal
```

```
root@ubuntuserver:~# docker exec -it php_apache /bin/bash
root@3ace2e3bf10a:/# cd /var/www/html/
root@3ace2e3bf10a:/var/www/html# mkdir journal
root@3ace2e3bf10a:/var/www/html# chmod -R 777 journal
root@3ace2e3bf10a:/var/www/html# service mysql start; service ssh start; service
apache2 start
* Starting MySQL database server mysqld [ OK ]
* Starting OpenBSD Secure Shell server sshd [ OK ]
* Starting Apache httpd web server apache2
*
```

Запуск службы MySQL, демона SSH и веб-сервера Apache 2:

```
service mysql start; service ssh start; service apache2
start
```

На рабочей машине создаётся директория journal и производится переход в неё:

```
$ mkdir journal && cd journal
```

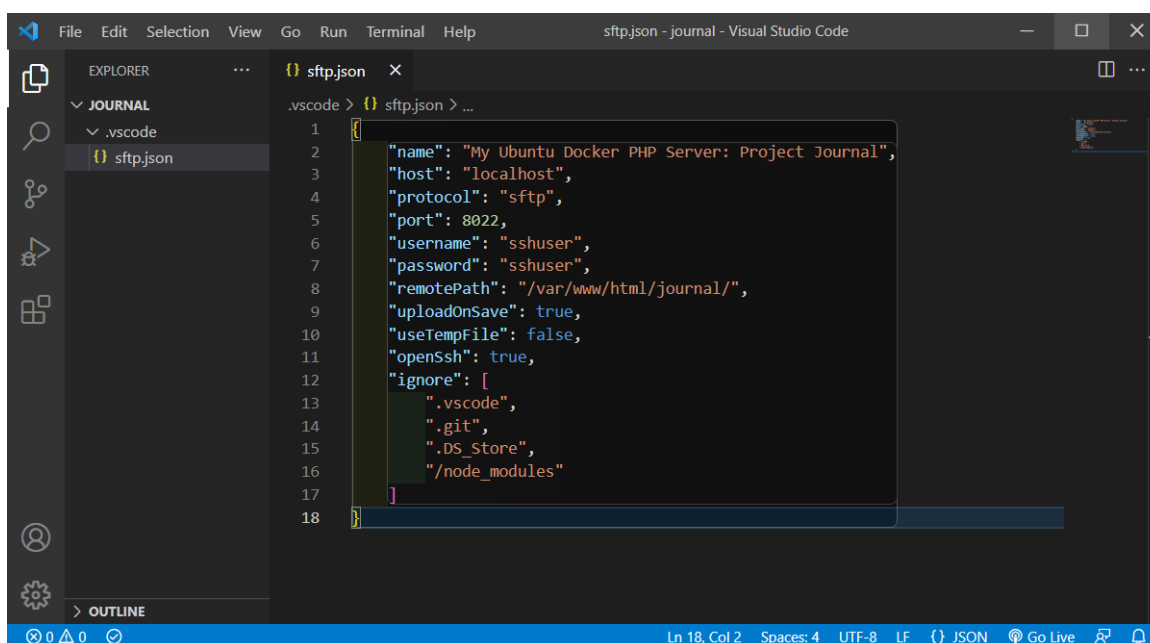
Запуск редактора Visual Studio Code в дирекции journal:

```
$ code .
```

```
ARTY@i7 MINGW64 /d
$ mkdir journal && cd journal
ARTY@i7 MINGW64 /d/journal
$ code .
```

Расширение SFTP от Natizyskunk для редактора кода Visual Studio Code было установлено в самостоятельной работе №1. Для того, чтобы синхронизировать с сервером директорию journal, в редакторе Visual Studio Code нажимается комбинация клавиш Ctrl+Shift+P и выполняется команда SFTP:Config. После этого расширение SFTP в дирекции PHP создаст поддиректорию .vscode с файлом конфигурации. В открывшемся файле sftp.json указываются данные для подключения по SSH к Docker контейнеру внутри Ubuntu Server:

- **name:** имя сервера;
- **host:** адрес сервера;
- **protocol:** протокол;
- **port:** порт (проброшен в виртуальной машине);
- **username** и **password:** имя пользователя и пароль для доступа к контейнеру;
- **remotePath:** абсолютный путь к директории на сервере (в контейнере), куда будут загружаться файлы;
- **uploadOnSave:** автоматическая загрузка файлов на сервер при их сохранении;
- **ignore:** директории, которые не предназначены для загрузки на сервер и выгрузки с него.



```

1  {
2    "name": "My Ubuntu Docker PHP Server: Project Journal",
3    "host": "localhost",
4    "protocol": "sftp",
5    "port": 8022,
6    "username": "sshuser",
7    "password": "sshuser",
8    "remotePath": "/var/www/html/journal/",
9    "uploadOnSave": true,
10   "useTempFile": false,
11   "openSsh": true,
12   "ignore": [
13     ".vscode",
14     ".git",
15     ".DS_Store",
16     "/node_modules"
17   ]
18 }

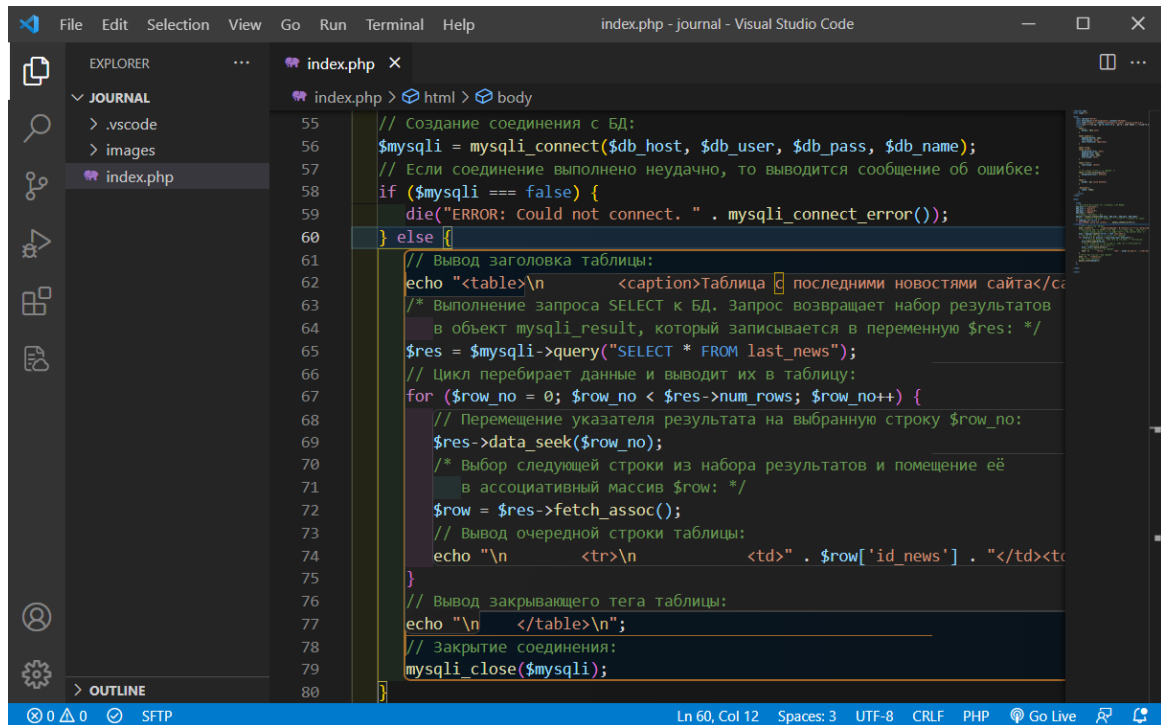
```

В боковом меню с файлами внутри редактора Visual Studio Code открывается локальная директория journal. Внутри неё производится нажатие правой кнопки мыши, выбор Sync Remote -> Local для того, чтобы синхронизировать удаленную директорию с локальной. Данная операция перезапишет всё, что есть в локальной директории journal! Нужно соблюдать осторожность, чтобы не перепутать её с командой Sync Local -> Remote — это затрет все файлы в Docker контейнере!

В боковом меню внутри редактора Visual Studio Code внутри директории journal производится нажатие правой кнопки мыши, выбор New Folder и задаётся имя новой поддиректории images. В эту поддиректорию переносятся все изображения, путь до которых был указан при добавлении записей в таблицу last_news.

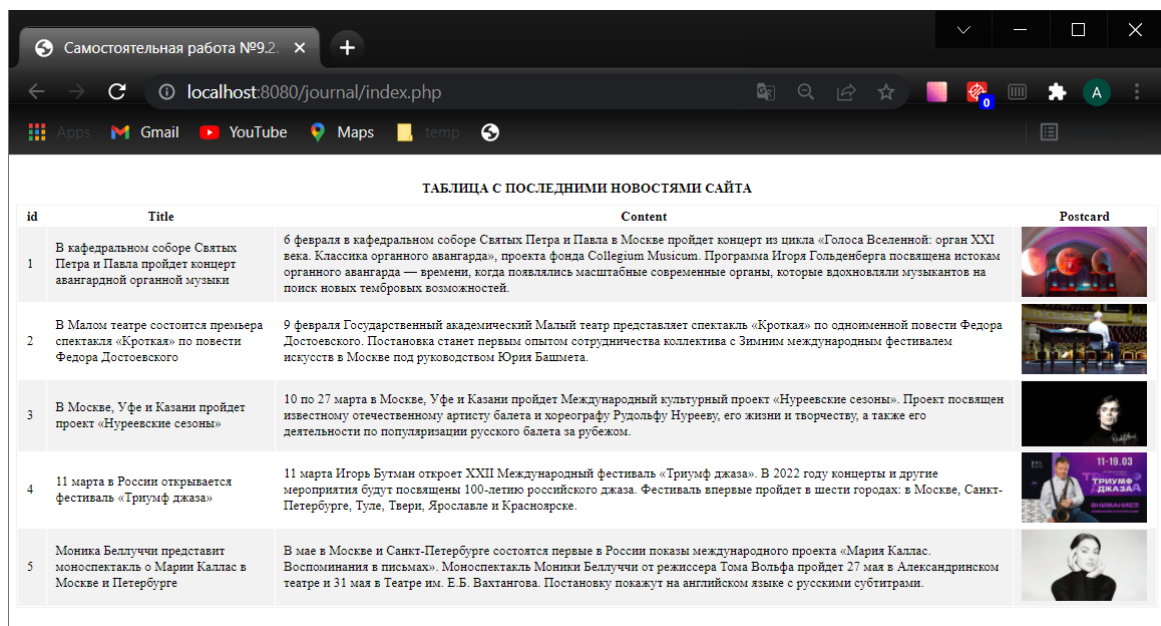
Для того, чтобы синхронизировать локальную директорию journal вместе с её новым содержимым с удаленной директорией в Docker контейнере, производится нажатие правой кнопки мыши, выбор Sync Local -> Remote.

В редакторе Visual Studio Code в директории PHP создается файл `index.php`, в который добавляется HTML страница с программой на языке PHP. Программа соединяется с БД MySQL и выводит всё содержимое таблицы `last_news` на страницу. Файл `index.php` сохраняется и автоматически загружается на сервер (в контейнер):



```
55 // Создание соединения с БД:
56 $mysqli = mysqli_connect($db_host, $db_user, $db_pass, $db_name);
57 // Если соединение выполнено неудачно, то выводится сообщение об ошибке:
58 if ($mysqli === false) {
59     die("ERROR: Could not connect. " . mysqli_connect_error());
60 } else {
61     // Вывод заголовка таблицы:
62     echo "<table>\n    <caption>Таблица с последними новостями сайта</caption>\n";
63     /* Выполнение запроса SELECT к БД. Запрос возвращает набор результатов
64     в объект mysqli_result, который записывается в переменную $res: */
65     $res = $mysqli->query("SELECT * FROM last_news");
66     // Цикл перебирает данные и выводит их в таблицу:
67     for ($row_no = 0; $row_no < $res->num_rows; $row_no++) {
68         // Перемещение указателя результата на выбранную строку $row_no:
69         $res->data_seek($row_no);
70         /* Выбор следующей строки из набора результатов и помещение её
71         в ассоциативный массив $row: */
72         $row = $res->fetch_assoc();
73         // Вывод очередной строки таблицы:
74         echo "\n    <tr>\n        <td>" . $row['id_news'] . "</td><td>" . $row['title'] . "</td><td>" . $row['content'] . "</td><td>" . $row['postcard'] . "</td>\n";
75     }
76     // Вывод закрывающего тега таблицы:
77     echo "\n    </table>\n";
78     // Закрытие соединения:
79     mysqli_close($mysqli);
80 }
```

По умолчанию, сервер Apache работает на 80-м порту, но при запуске контейнера порт 80 был проброшен на порт 8080. Страница с загруженным файлом `index.php` доступна из браузера по адресу `http://localhost:8080/journal/index.php`



Как видно на снимке экрана, при открытии этой страницы в браузере выводится список всех записей таблицы `last_news`.

Декомпозиция файла проекта согласно шаблону MVC

MVC (Model-View-Controller, «Модель-Представление-Контроллер», «Модель-Вид-Контроллер») — это шаблон программирования, который позволяет разделить логику приложения на три компонента: модель, представление и контроллер. Разделение выполняется таким образом, что модификация каждого компонента может осуществляться независимо:

- Model (модель). Получает от контроллера, выполняет необходимые операции и передаёт их в представление.
- View (представление или вид). Получает данные от модели и отображает их пользователю.
- Controller (контроллер). Интерпретирует действия пользователя, проверяет полученные данные и передаёт их модели.

Необходимо выполнить декомпозицию файла и построить следующую структуру:

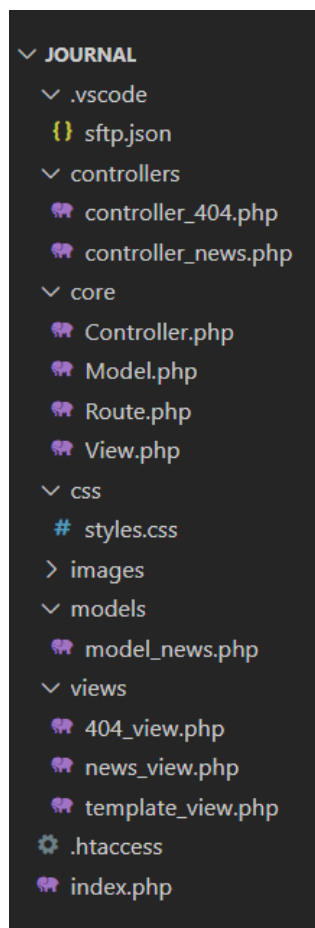


Если говорить более конкретно, то компоненты приложения будут следующие:

- Представление (или вид) — интерфейс.
- Контроллер — обработчик событий, инициируемых пользователем (нажатие на кнопку, переход по ссылке, отправка формы).
- Модель — метод, который запускается обработчиком и выполняет все основные операции (получение записей из базы данных, проведение вычислений).

Модель, контроллер и представление у каждой страницы наследуют методы и свойства от соответствующих общих классов: Модель, Контроллер и Представление.

Проект приводится к следующему виду:



Файлы в директории journal:

- **index.php** — подключение общих классов модели, представления, контроллера, а также подключение и запуск маршрутизатора.
- **.htaccess** (сокращение от "hypertext access") — файл дополнительной конфигурации веб-сервера Apache. Название файла начинается с точки. Это означает, что файл — служебный и не относится непосредственно к файлам сайта, а используется для настроек web-сервера, является частью конфигурации web-сервера Apache.

Файлы в директории journal/core:

- **model.php** — описание общего класса для модели (Model), в котором производится подключение к БД и создаётся общий метод исполнения запросов.
- **controller.php** — описание общего класса для контроллера (Controller). Он умеет при инициализации создавать экземпляр объекта View.
- **controller.php** — описание общего класса для представления (View). Он умеет при инициализации создавать экземпляр объекта View.
- **route.php** — описание общего класса для маршрутизатора (Router).

Файлы в директории journal/css:

- **styles.css** — файл со стилями пользователя.

Файлы в директории journal/models:

- **model_news.php** — описание класса модели последних новостей сайта.

Файлы в директории `journal/views`:

- **template_view.php** — общий шаблон представлений.
- **news_view.php** — шаблон представлений последних новостей сайта.
- **404_view.php** — шаблон представлений ошибки 404. Ошибка 404 или Not Found («не найдено») — стандартный код ответа по протоколу HTTP о том, что клиент был в состоянии общаться с сервером, но сервер не может найти данные согласно запросу. Пользователи наиболее часто сталкиваются с ошибкой 404 при посещении так называемых «битых» или «мёртвых ссылок», когда браузеру не удалось обнаружить на сервере указанный URL.

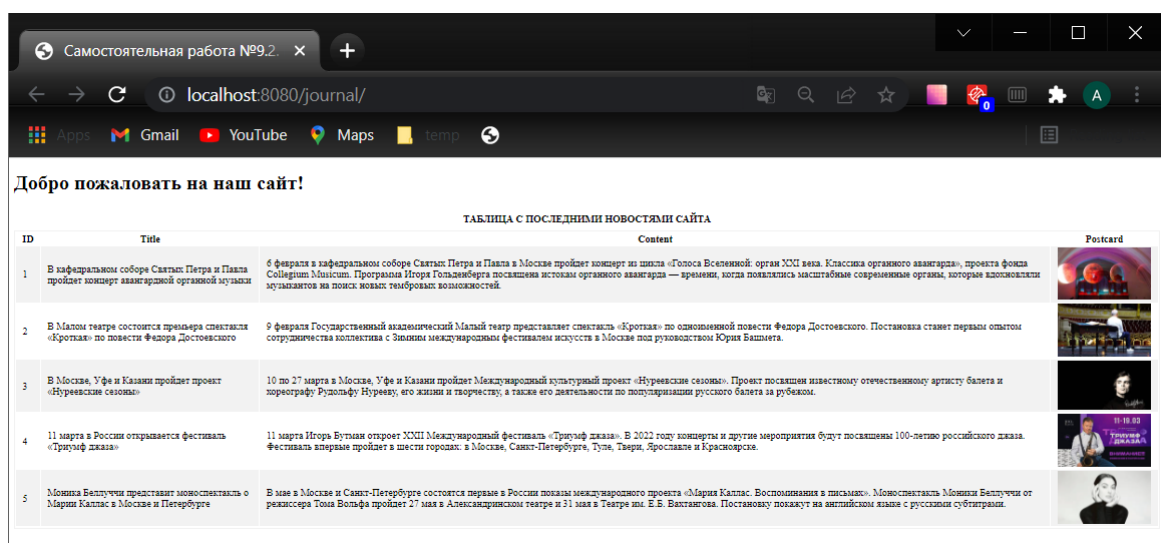
Файлы в директории `journal/controllers`:

- **controller_news.php** — описание класса контроллера для реализации страницы с последними новостями сайта.
- **controller_404.php** — описание класса контроллера для реализации страницы, сообщающей об ошибке 404.

Файлы в директории `images` — все изображения, путь до которых был указан при добавлении записей в таблицу `last_news`.

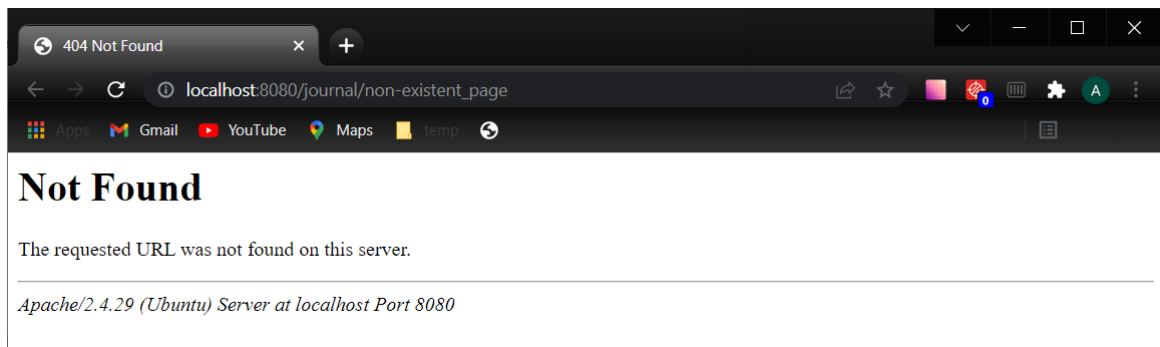
Нужно синхронизировать локальную директорию `journal` вместе с её новым содержимым с удаленной директорией в Docker контейнере. Для этого в боковом меню редактора Visual Studio Code внутри директории `journal` производится нажатие правой кнопки мыши, выбор `Sync Local -> Remote`.

Страница с загруженным файлом декомпозированным файлом `index.php` доступна из браузера по адресу `http://localhost:8080/journal/`.



Как видно на снимке экрана, при открытии этой страницы в браузере все так же успешно выводится список всех записей таблицы `last_news`.

Если открыть страницу на сервере по «мёртвой ссылке», например `http://localhost:8080/journal/non-existent_page`, то вместо реализованной в проекте страницы, сообщающей об ошибке 404, появляется стандартная страница 404 от Apache:



Для того, чтобы вместо стандартной страницы выводилась страница, реализованная в проекте, внутри контейнера нужно, во-первых, включить модуль `mod_rewrite` для Apache 2. Этот инструмент позволяет переписывать URL-адреса и создавать чистые ссылки, преобразовывая сложные пути в понятные и читабельные ссылки:

```
a2enmod rewrite
```

Во-вторых, нужно включить `AllowOverride`. Для этого в текстовом редакторе `nano` открывается конфигурационный файл Apache 2:

```
nano /etc/apache2/apache2.conf
```

```
root@3ace2e3bf10a:/var/www/html# a2enmod rewrite
Enabling module rewrite.
To activate the new configuration, you need to run:
  service apache2 restart
root@3ace2e3bf10a:/var/www/html# nano /etc/apache2/apache2.conf
```

В конфигурационном файле находится два параметра «`AllowOverride`» и их значения изменяются с `None` на `All`:

```
<Directory />
...
    AllowOverride All
...

<Directory /var/www/>
...
    AllowOverride All
```



```
GNU nano 2.9.3 /etc/apache2/apache2.conf Modified
# the latter may be used for local directories served by the web server. If
# your system is serving content from a sub-directory in /srv you must allow
# access here, or in any related virtual host.
<Directory />
    Options FollowSymLinks
    AllowOverride All
    Require all denied
</Directory>

<Directory /usr/share>
    AllowOverride None
    Require all granted
</Directory>

<Directory /var/www/>
    Options Indexes FollowSymLinks
    AllowOverride All
    Require all granted
</Directory>

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line
```

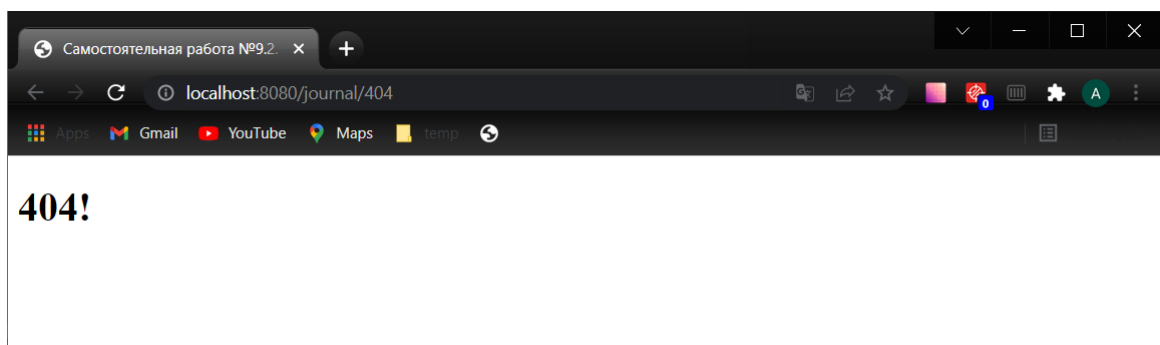
После сохранения файла конфигурации выполняется перезапуск веб-сервера Apache 2:

```
service apache2 restart
```

```
root@3ace2e3bf10a:/var/www/html# service apache2 restart
* Restarting Apache httpd web server apache2
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.17.0.2. Set the 'ServerName' directive globally to suppress this message

[ OK ]
```

Теперь если открыть страницу на сервере по «мёртвой ссылке», например http://localhost:8080/journal/non-existent_page, то выполнится переход на реализованную в проекте страницу, сообщающую об ошибке 404:



Добавление страницы со списком авторов статей

Через инструмент для управления базами данных HeidiSQL осуществляется соединение с БД MySQL. После нажатия ЛКМ (левой кнопки мыши) по названию БД POSTS, в правой части экрана откроется новая вкладка «Database: POSTS» («База данных: POSTS»). Осуществляется переход в эту вкладку, на пустом месте в этой вкладке нажимается ПКМ (правая кнопка мыши), в появившемся контекстном меню выбирается «Create new» -> «Table» («Создать» -> «Таблица»).

В появившейся вкладке «Table» («Таблица») заполняются следующие поля ввода:

Name (Имя): authors.

Comment (Комментарий): Таблица содержащая информацию об авторах статей.

Для того, чтобы создать автоинкрементный столбец (column) в этой таблице, нажимается кнопка «Add» («Добавить»), затем указываются следующие параметры столбца:

- **Name (Имя):** id — идентификационный номер автора в рамках БД.
- **Datatype (Тип данных):** INT.
- **Default (По умолчанию):** AUTO_INCREMENT.

Для того, чтобы сделать столбец id первичным ключом, на его названии нажимается ПКМ и выбирается «Create new index» («Создать новый индекс») -> «PRIMARY».

Добавляются ещё два столбца с типом данных **VARCHAR** (строковый):

- **fio** — фамилия, имя и отчество автора;
- **location** — географическое местоположение автора.

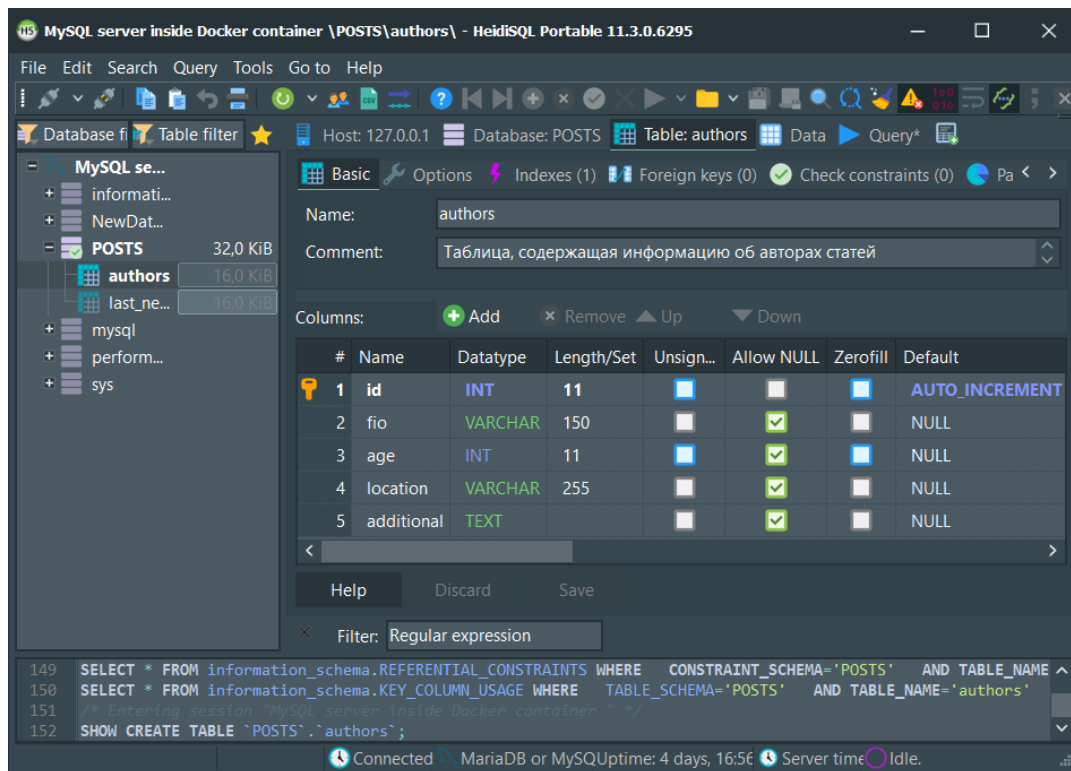
Третий по счёту столбец с типом данных **INT** (числовой):

- **age** — возраст автора.

Последний столбец имеет тип данных **TEXT**:

- **additional** — дополнительная информация.

Для сохранения внесенных в таблицу изменений нажимается кнопка «Save» («Сохранить»):



В результате формируются следующие команды SQL, создающие таблицу authors:

```
USE `POSTS` ;
```

```
CREATE TABLE IF NOT EXISTS `authors` (
```

```
  `id` int(11) NOT NULL AUTO_INCREMENT,
```

```
  `fio` varchar(150) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
```

```
  `age` int(11) DEFAULT NULL,
```

```
  `location` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
```

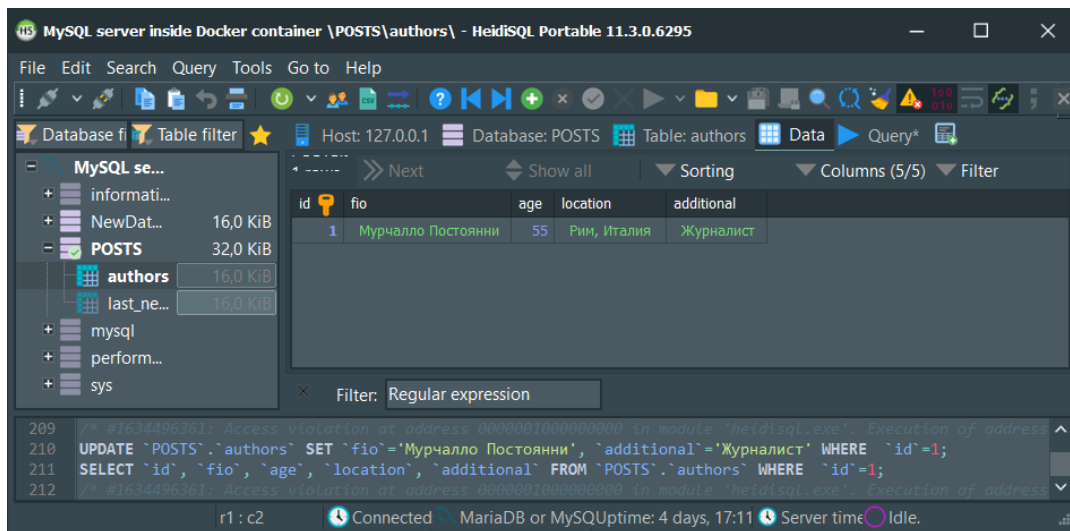
```
  `additional` text COLLATE utf8mb4_unicode_ci,
```

```
  PRIMARY KEY (`id`)
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
```

```
COLLATE=utf8mb4_unicode_ci COMMENT='Таблица, содержащая информацию об авторах статей';
```

Открывается вкладка «Data» («Данные») и в верхнем меню нажимается кнопка «Insert row into table» («Вставьте строку в таблицу»), представляющая из себя белый знак «+» на фоне зелёного круга, затем в таблицу вносятся данные новой записи. В таблицу authors вносится запись об одном авторе:



В результате формируются следующие команды SQL, вставляющие одну новую запись в таблицу authors:

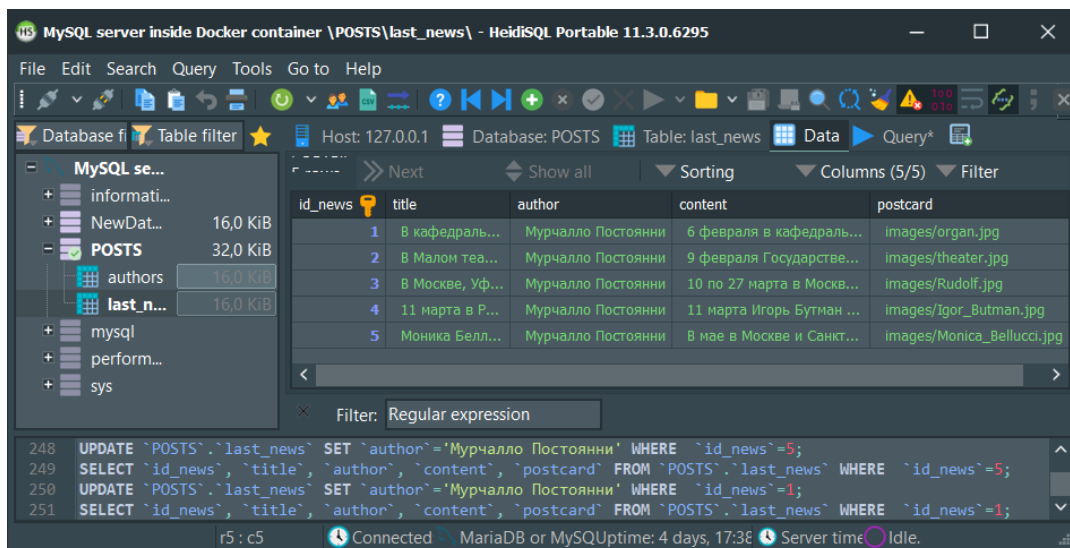
```

INSERT INTO `authors` (`id`, `fio`, `age`, `location`,
`additional`) VALUES

(1, 'Мурчалло Постоянни', 55, 'Рим, Италия',
'Журналист') ;

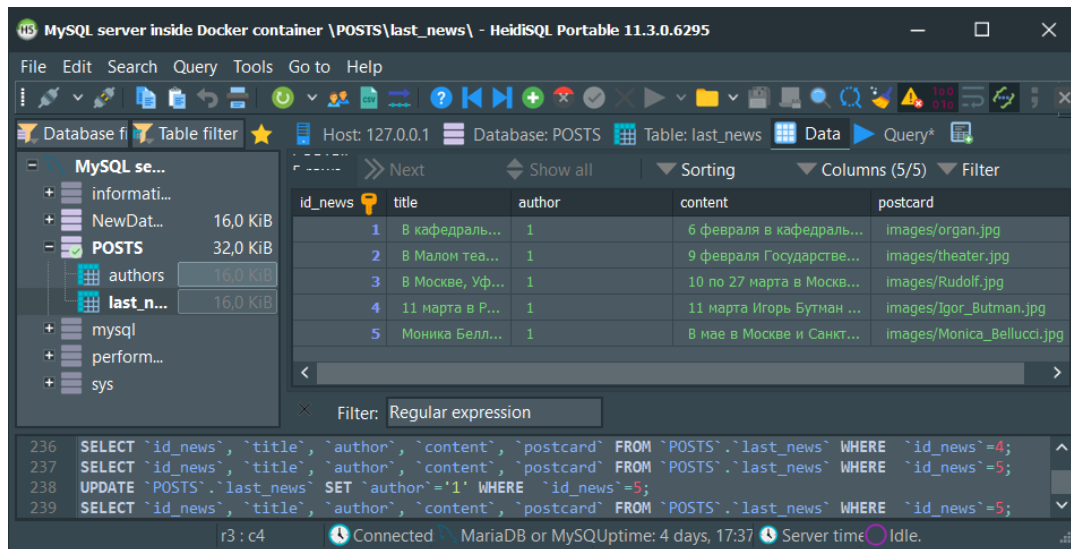
```

В левой части главного окна программы выбирается БД POSTS, затем таблица last_news, после этого открывается вкладка «Data» («Данные»). Как видно, в созданной ранее таблице last_news несколько раз идёт повторение одной и той же строки «Мурчалло Постоянни»:

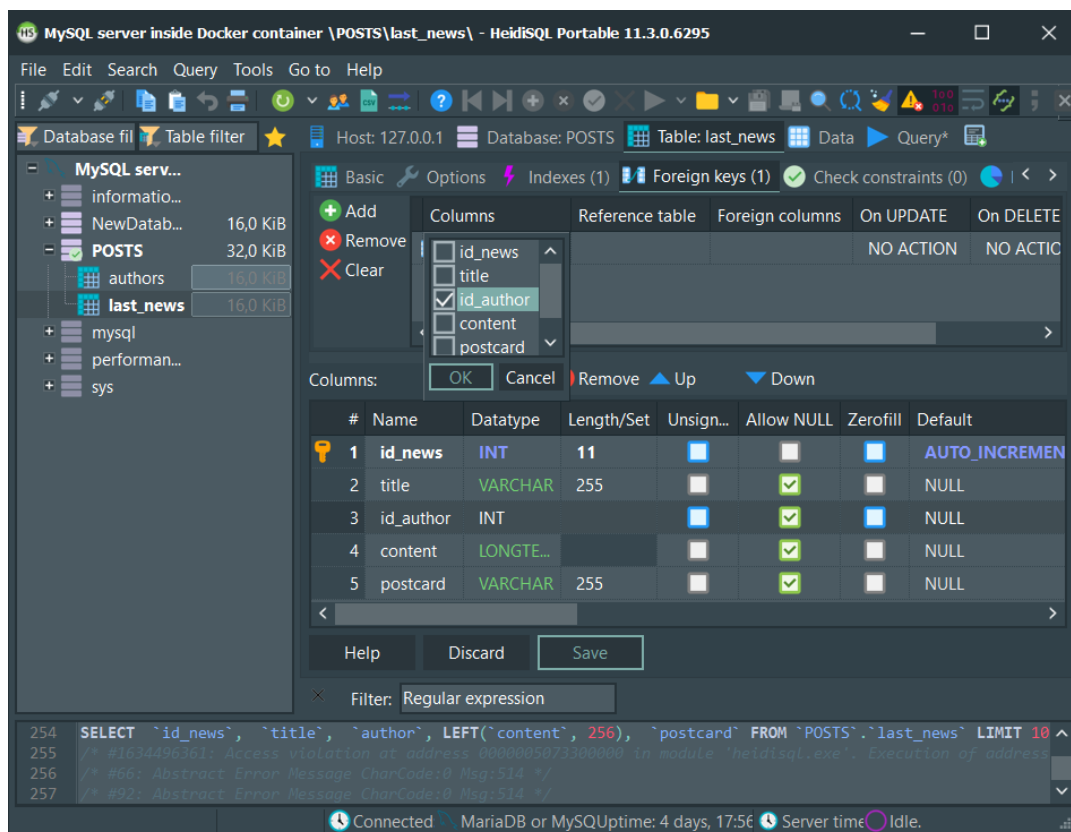


Нужно немного изменить структуру таблицы last_news. Для того, чтобы не дублировать эту строку для каждой новой статьи, написанной этим автором, и оптимизировать расходы памяти, создаётся связь между таблицей last_news и authors.

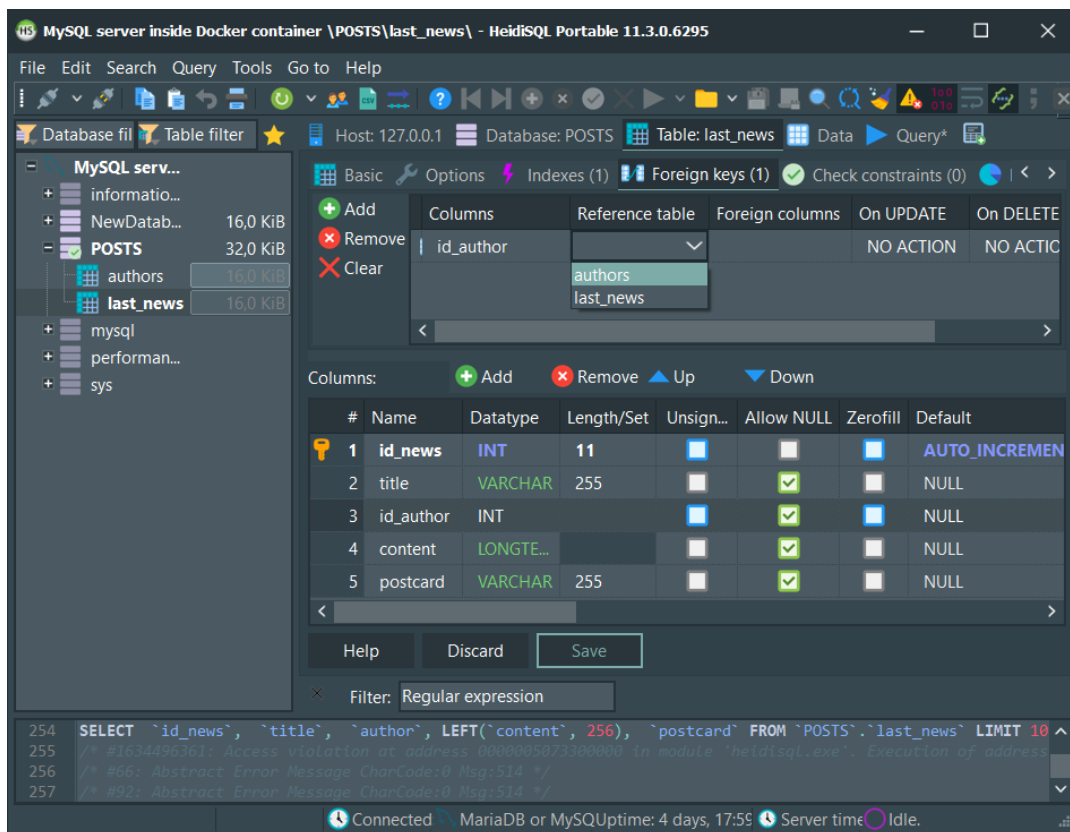
Для начала в таблице last_news все поля authors, содержащие «Мурчалло Постоянни», заменяется на 1 (это значение id в таблице authors, соответствующие данному автору).



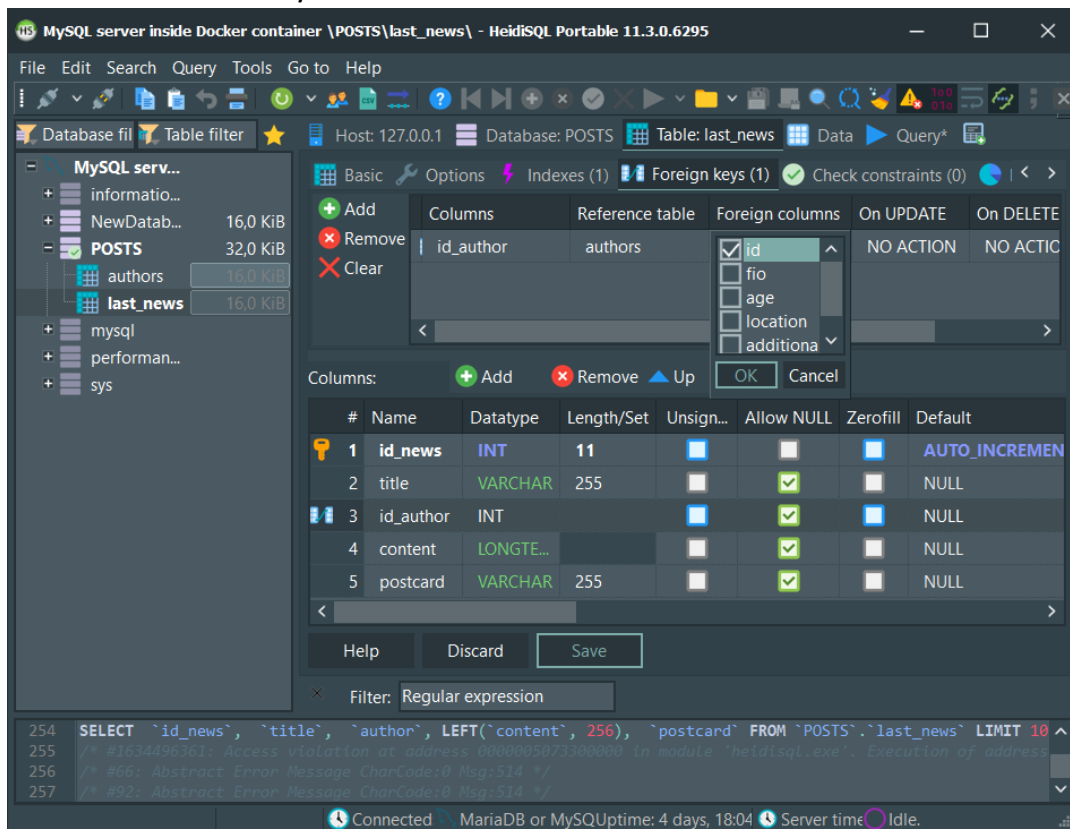
Открывается вкладка «Foreign keys» («Внешние ключи»). Название столбца author заменяется на id_author, а его тип изменяется на INT (числовой). В верхней части вкладки нажимается кнопка «Add» («Добавить»), представляющая из себя белый знак «+» на фоне зелёного круга, двойным щелчком ЛКМ на столбец «Columns» («Столбцы») вызывается список со столбцами, в нём устанавливается галочка напротив id_author. Выбор подтверждается нажатием на кнопку «OK»:



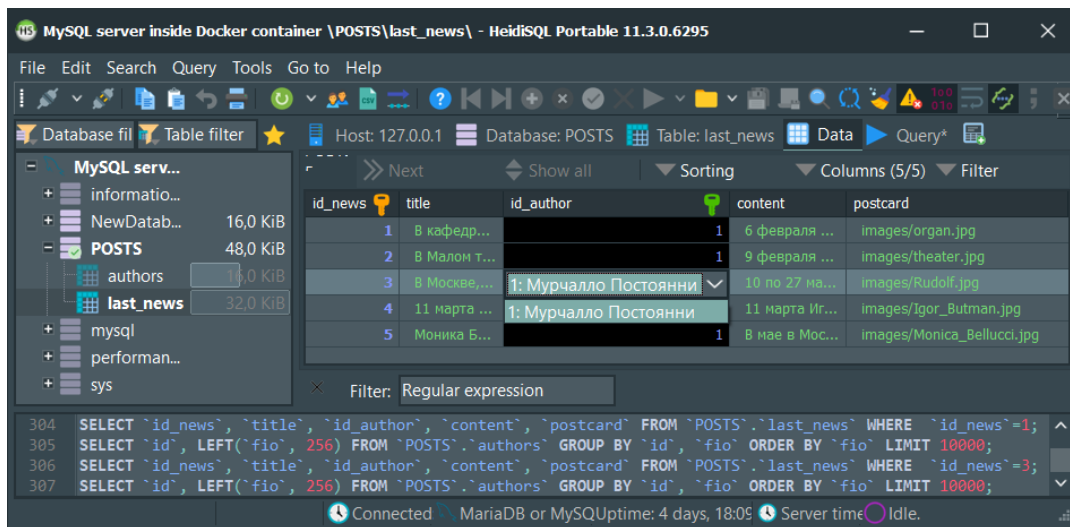
Аналогичным образом в столбце «Reference table» («Справочная таблица») выбирается таблица authors:



В столбце «Foreign columns» («Внешние столбцы») вызывается список со столбцами, в нём устанавливается галочка напротив id. Выбор подтверждается нажатием на кнопку «OK»:



Для сохранения внесенных в таблицу изменений нажимается кнопка «Save» («Сохранить»). Открывается вкладка «Data» («Данные») и двойным щелчком ЛКМ на одной из записей в столбце `id_author` вызывается список, в котором будем всего одно значение — «Мурчалло Постоянни»:



Поскольку `id` — это автоинкрементный столбец, ключи которого повторяться не могут, то реализована связь один ко многим.

Для реализации страницы, выводящей информацию обо всех авторах, а также страницы, выводящей информацию об одном авторе, в проект добавляются новые файлы.

Файлы в директории `journal/models`:

- **model_about.php** — описание класса модели с информацией об авторах статей.

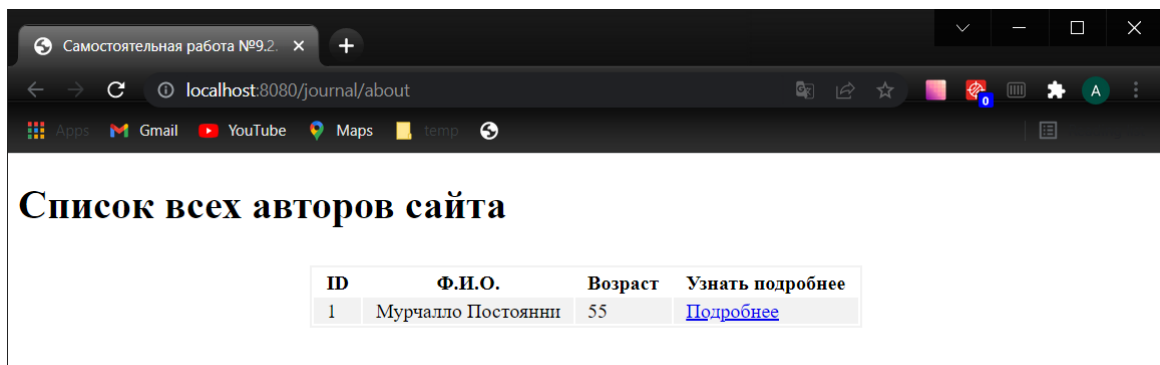
Файлы в директории `journal/views`:

- **authors_view.php** — шаблон представлений для вывода информации обо всех авторах.
- **author_view.php** — шаблон представлений для вывода информации об одном авторе.

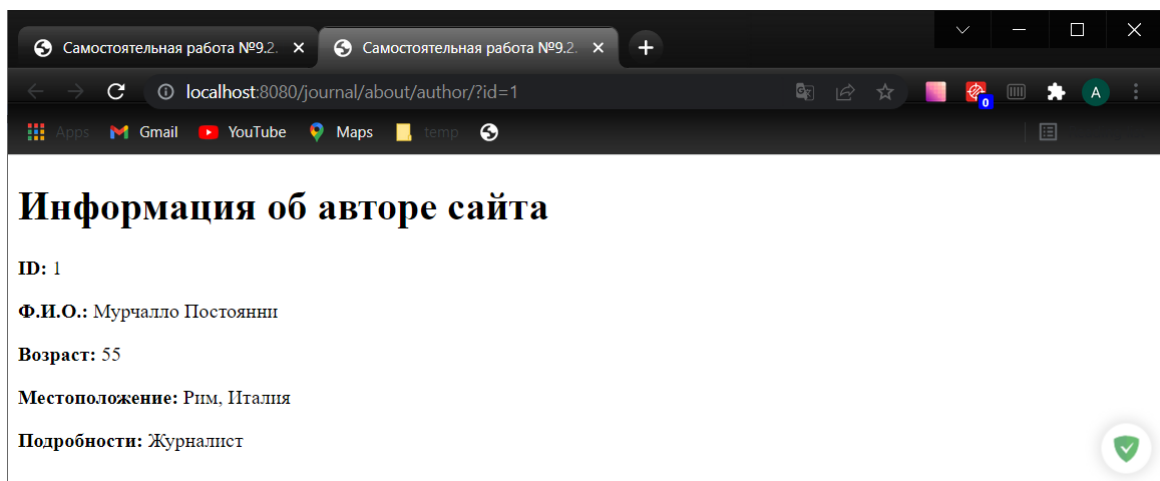
Файлы в директории `journal/controllers`:

- **controller_about.php** — описание класса контроллера для реализации страницы со списком авторов статей.

Страница со списком всех авторов доступна из браузера по адресу <http://localhost:8080/journal/about>



Страница со информацией об авторе, которому соответствует значение id, равное 1, доступна из браузера по адресу <http://localhost:8080/journal/about/author/?id=1>



Добавление страницы с обратной связью

После нажатия ЛКМ (левой кнопки мыши) по названию БД POSTS, в правой части экрана откроется новая вкладка «Database: POSTS» («База данных: POSTS»). Осуществляется переход в эту вкладку, на пустом месте в этой вкладке нажимается ПКМ (правая кнопка мыши), в появившемся контекстном меню выбирается «Create new» -> «Table» («Создать» -> «Таблица»). В появившейся вкладке «Table» («Таблица») заполняются следующие поля ввода:

Name (Имя): feedback.

Comment (Комментарий): Таблица с данными, полученными через форму обратной связи.

Для того, чтобы создать автоинкрементный столбец (column) в этой таблице, нажимается кнопка «Add» («Добавить»), затем указываются следующие параметры столбца:

- **Name (Имя):** id_feedback.
- **Datatype (Тип данных):** INT.
- **Default (По умолчанию):** AUTO_INCREMENT.

Для того, чтобы сделать столбец id_feedback первичным ключом, на его названии нажимается ПКМ и выбирается «Create new index» («Создать новый индекс») -> «PRIMARY». После успешного создания уникального индекса рядом с названием столбца отобразится иконка ключа, а во вкладке «Indexes» («Индексы») появится его название.

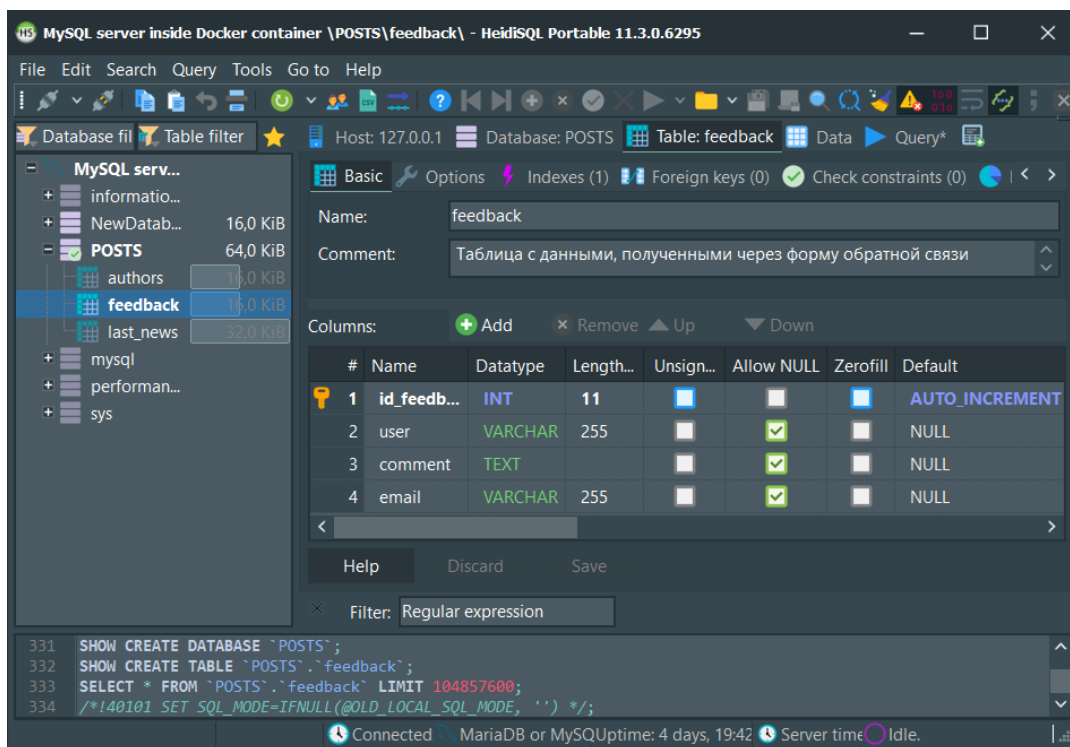
Добавляются ещё два столбца с типом данных **VARCHAR** (строковый):

- **user** — имя пользователя;
- **email** — адрес электронной почты.

Предпоследний столбец имеет тип данных **TEXT**:

- **comment** — текст отзыва.

Для сохранения внесенных в таблицу изменений нажимается кнопка «Save» («Сохранить»).



В результате формируются следующие команды SQL, создающие таблицу feedback:

```
USE `POSTS` ;
```

```
CREATE TABLE IF NOT EXISTS `feedback` (
```

```
`id_feedback` int(11) NOT NULL AUTO_INCREMENT,
```



```

`user` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT
NULL,

`comment` text COLLATE utf8mb4_unicode_ci,

`email` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT
NULL,

PRIMARY KEY (`id_feedback`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci COMMENT='Таблица с данными,
полученными через форму обратной связи';

```

Для реализации страницы, позволяющей заполнить форму обратной связи, в проект добавляются новые файлы.

Файлы в директории `journal/models`:

- `model_feedback.php`.

Файлы в директории `journal/views`:

- `api_template_view.php`.
- `feedback_result_view.php`.
- `feedback_view.php`.

Файлы в директории `journal/controllers`:

- `controller_feedback.php`.

Страница с формой обратной связи доступна из браузера по адресу `http://localhost:8080/journal/feedback`

Самостоятельная работа №9.2. x

localhost:8080/journal/feedback

Apps Gmail YouTube Maps temp

Оставьте обратную связь о нашем сайте!

Сальвадор Вблиз

Salvador@google.es

Восхитительно!
Великолепно!

Отправить Очистить

После заполнения формы и нажатия на кнопку «Отправить» выводится сообщение о том, что отзыв добавлен, а в таблице feedback появляется новая запись:

