```
-- Создание базы данных FrontHub
CREATE DATABASE IF NOT EXISTS FrontHub;
-- Использование базы данных FrontHub
USE FrontHub;
-- Создание таблицы Data_storage
CREATE TABLE Data_storage (
data_id INT PRIMARY KEY AUTO_INCREMENT,
path VARCHAR(255) NOT NULL,
data_storage mediumblob NOT NULL
);
-- Заполнение таблицы Data_storage
INSERT INTO Data_storage (path, data_storage) VALUES
('/files/data_A', '/path/to/realistic_file_A.txt'),
('/files/data_B', '/path/to/realistic_file_B.docx'),
('/files/data_C', '/path/to/realistic_file_C.jpg'),
('/files/data_D', '/path/to/realistic_file_D.pdf'),
('/files/data_E', '/path/to/realistic_file_E.xlsx');
-- Создание таблицы Commit
CREATE TABLE Commit (
commit_id INT PRIMARY KEY AUTO_INCREMENT,
commit_text TEXT NOT NULL
);
-- Заполнение таблицы Commit
INSERT INTO Commit (commit_text) VALUES
('Initial commit'),
('Added new feature'),
('Fixed bug #123'),
('Refactored code'),
('Merged feature branch');
```

```
-- Создание таблицы Category
CREATE TABLE Category (
category_id INT PRIMARY KEY AUTO_INCREMENT,
category_name VARCHAR(255) NOT NULL
);
-- Заполнение таблицы Category
INSERT INTO Category (category_name) VALUES
('Web Development'),
('Mobile App Development'),
('web development studios'),
('IT departments of banks, mobile operators, marketplaces'),
('Developing and maintaining the user interface');
-- Создание таблицы Branch
CREATE TABLE Branch (
branch_id INT PRIMARY KEY AUTO_INCREMENT,
branch_name VARCHAR(255) NOT NULL
);
-- Заполнение таблицы Branch
INSERT INTO Branch (branch_name) VALUES
('main'),
('development'),
('feature_branch_123'),
('hotfix_branch'),
('release_branch_v1.0');
-- Создание таблицы Role
CREATE TABLE Role (
role_id INT PRIMARY KEY AUTO_INCREMENT,
role_name VARCHAR(255) NOT NULL,
permissions VARCHAR(255) NOT NULL
```

```
);
-- Заполнение таблицы Role
INSERT INTO Role (role_name, permissions) VALUES
('Admin', 'admin_permissions'),
('User', 'user_permissions'),
('Moderator', 'moderator_permissions');
-- Создание таблицы User_Table
CREATE TABLE User_Table (
user_id INT PRIMARY KEY AUTO_INCREMENT,
user_name VARCHAR(255) NOT NULL,
hashed_password VARCHAR(255) NOT NULL,
token VARCHAR(255),
role_id INT,
FOREIGN KEY (role_id) REFERENCES Role(role_id)
);
-- Заполнение таблицы User_Table с более реалистичными хешами и токенами
INSERT INTO User_Table (user_name, hashed_password, token, role_id) VALUES
('John_Doe', 'c4ca4238a0b923820dcc509a6f75849b', 'e5e9fa1ba31ecd1ae84f75caaa474f3a', 2),
('Alice_Smith', 'c81e728d9d4c2f636f067f89cc14862c', 'f7b3d4ebc32c1b1f1630816fa75f4b3b', 1),
('Bob_Johnson', 'eccbc87e4b5ce2fe28308fd9f2a7baf3', 'd76f7d5a96a19841f4c6791a6e4eb2e5', 1),
('Eva_Brown', 'a87ff679a2f3e71d9181a67b7542122c', 'b858cb282617fb0956d960215c8e84d9', 3),
('Charlie_White', 'e4da3b7fbbce2345d7772b0674a318d5', '2510c39011c5be704182423e3a695e91', 2);
-- Создание таблицы Repository
CREATE TABLE Repository (
repository_id INT PRIMARY KEY AUTO_INCREMENT,
data_id INT,
branch_id INT,
commit_id INT,
title VARCHAR(255) NOT NULL,
```

```
description TEXT,
FOREIGN KEY (data_id) REFERENCES Data_storage(data_id),
FOREIGN KEY (branch_id ) REFERENCES Branch(branch_id ),
FOREIGN KEY (commit_id ) REFERENCES Commit(commit_id )
);
-- Заполнение таблицы Repository
INSERT INTO Repository (data_id, branch_id, commit_id, title, description) VALUES
(1, 1, 1, 'HTML5 & CSS3. HTML5 и CSS3', 'fundamental technologies'),
(2, 2, 2, 'Flex и Grid CSS.', 'Flex и Grid CSS.'),
(3, 3, 3, 'Bootstrap 4', 'Bootstrap 4'),
(4, 4, 4, 'Bootstrap 5.3', 'Bootstrap 5.3'),
(5, 5, 5, 'CSS preprocessors', 'CSS preprocessors');
-- Создание таблицы Question
CREATE TABLE Question (
question_id INT PRIMARY KEY AUTO_INCREMENT,
user_id INT,
data_id INT,
question_text VARCHAR(255) NOT NULL,
completed BOOLEAN,
QuestionTitle Varchar(255),
FOREIGN KEY (user_id) REFERENCES User_Table(user_id),
FOREIGN KEY (data_id) REFERENCES Data_storage(data_id)
);
-- Заполнение таблицы Question
INSERT INTO Question (user_id, data_id, question_text, completed, QuestionTitle) VALUES
(1, 2, 'How to create a new branch?', TRUE, "How to create a new branch?" ),
(2, 3, 'What is the best data visualization library?', FALSE, 'What is the best data visualization library?'),
(3, 4, 'How to handle user authentication in mobile apps?', TRUE, 'How to handle user authentication in mobile apps?'),
```

```
(3,5, 'What is the purpose of cross-validation?', FALSE, 'What is the purpose of cross-validation?'),
(2, 1,'How to implement continuous integration?', TRUE, 'How to implement continuous integration?');
-- Создание таблицы QuestionCategoryConnect
CREATE Table QuestionCategoryConnect (
category_id INT,
question_id INT,
FOREIGN KEY (category_id) REFERENCES Category(category_id),
FOREIGN KEY (question_id) REFERENCES Question(question_id)
);
-- Заполнение таблицы QuestionCategoryConnect
INSERT INTO questioncategoryconnect (category_id, question_id) VALUES
(1,1), (2,1), (4,3), (3, 5), (5, 2);
-- Создание таблицы Answer
CREATE TABLE Answer (
answer_id INT PRIMARY KEY AUTO_INCREMENT,
question_id INT,
user_id INT,
data_id INT,
answer_text VARCHAR(255) NOT NULL,
correct BOOLEAN,
FOREIGN KEY (question_id) REFERENCES Question(question_id),
FOREIGN KEY (user_id) REFERENCES User_Table(user_id),
FOREIGN KEY (data_id) REFERENCES Data_storage(data_id)
);
-- Заполнение таблицы Answer
INSERT INTO Answer (question_id, user_id, data_id, answer_text, correct) VALUES
(1, 1, 1, 'To create a new branch, use the command "git branch branch_name"', TRUE),
(2, 2, 3, 'Matplotlib is a popular data visualization library in Python', FALSE),
(3, 3, 2, 'You can use Firebase for user authentication in mobile apps', TRUE),
```

```
(4, 4, 2, 'Cross-validation is used to assess the performance of a machine learning model', FALSE),
(5, 5, 3, 'Jenkins is a popular tool for implementing continuous integration', TRUE);
-- Создание таблицы UserReposotoryConnect
CREATE TABLE UserReposotoryConnect (
user_id INT,
repository_id INT,
FOREIGN KEY (user_id) REFERENCES User_Table(user_id),
FOREIGN KEY (repository_id) REFERENCES Repository(repository_id)
);
-- Заполнение таблицы UserReposotoryConnect
INSERT INTO UserReposotoryConnect (user_id, repository_id) VALUES
(1, 3),
(2, 4),
(3, 5);
```