ТЕСТ-КЕЙСЫ ДЛЯ ЗАГРУЖЕННЫХ ФАЙЛОВ

1

Файл: test_data_inserted.py

Тест: test_data_inserted

Цель: проверить, что данные корректно вставлены в таблицы базы данных после инициализации.

Шаги:

1. Инициализировать базу данных test_clock.db.
2. Удалить все существующие таблицы в базе данных.
3. Создать новые таблицы в базе данных.
4. Инициализировать значения в таблицах с помощью функции init_values().
5. Создать курсор для выполнения SQL-запросов.
6. Выполнить запрос для проверки наличия записи с id = 29 в таблице questions.
7. Выполнить запрос для проверки наличия записи с id = 15 в таблице questions.
8. Выполнить запрос для проверки наличия записи с id = 1 в таблице questions.
9. Выполнить запрос для проверки наличия записи с id = 2 в таблице times.
10. Закрыть курсор.
11. Закрыть соединение с базой данных.

Ожидаемый результат: Все запросы должны вернуть ненулевые значения, что подтверждает наличие записей с указанными id в соответствующих таблицах. Если какая-либо запись отсутствует, тест должен завершиться с ошибкой.

2

Файл: test_get_enabled_clock.py

Тест: test_get_enabled_clock

Цель: проверить, что функция get_enables_clocks() правильно возвращает список включенных часов из базы данных.

Шаги:

1. Инициализировать базу данных test_clock.db.
2. Удалить все существующие таблицы в базе данных.
3. Создать новые таблицы в базе данных.
4. Инициализировать значения в таблицах с помощью функции init_values().
5. Создать курсор для выполнения SQL-запросов.
6. Обновить запись с id = 1 в таблице times, установив время на '11:11:00' и is_enable на 1 (включено).

7. Обновить запись с id = 3 в таблице times, установив время на '00:10:00' и is_enable на 1 (включено).
8. Зафиксировать изменения в базе данных с помощью commit().
9. Проверить, что результат выполнения db.get_enables_clocks() равен [("11:11:00",), ("00:10:00",)].
10. Закрыть курсор.
11. Закрыть соединение с базой данных.

Ожидаемый результат: Функция get_enables_clocks() должна вернуть список включенных часов, содержащий только обновленные значения времени. Если результат не совпадает с ожидаемым, тест должен завершиться с ошибкой.

3

Файл: test_get_question.py

Тест: test_get_question

Цель: проверить, что функция get_random_question() возвращает случайный вопрос из базы данных.

Шаги:

1. Инициализировать базу данных test_clock.db.
2. Удалить все существующие таблицы в базе данных.
3. Создать новые таблицы в базе данных.
4. Инициализировать значения в таблицах с помощью функции init_values().
5. Проверить, что результат выполнения db.get_random_question() не равен None, что подтверждает наличие хотя бы одного вопроса в базе данных.
6. Закрыть соединение с базой данных.

Ожидаемый результат: Функция get_random_question() должна вернуть ненулевое значение, что подтверждает наличие случайного вопроса в базе данных. Если результат равен None, тест должен завершиться с ошибкой.

4

Файл: test_init_db.py

Тест: test_init_db

Цель: проверить, что таблицы questions и times успешно созданы в базе данных после инициализации.

Шаги:

1. Инициализировать базу данных test_clock.db.
2. Удалить все существующие таблицы в базе данных.
3. Создать новые таблицы в базе данных.

4. Создать курсор для выполнения SQL-запросов.
5. Выполнить SQL-запрос для проверки существования таблицы questions в базе данных и убедиться, что результат не равен None.
6. Выполнить SQL-запрос для проверки существования таблицы times в базе данных и убедиться, что результат не равен None.
7. Закрыть курсор.
8. Закрыть соединение с базой данных.

Ожидаемый результат: Тест должен подтвердить, что обе таблицы (questions и times) были успешно созданы в базе данных. Если хотя бы одна из таблиц отсутствует, тест должен завершиться с ошибкой.

5

Файл: test_init_db.py

Тест: test_init_db

Цель: проверить, что таблицы questions и times успешно созданы в базе данных, а также протестировать обновление записи в таблице times.

Шаги:

1. Инициализировать базу данных test_clock.db.
2. Удалить все существующие таблицы в базе данных.
3. Создать новые таблицы в базе данных.
4. Инициализировать значения в таблицах с помощью функции init_values().
5. Создать курсор для выполнения SQL-запросов.

6. Обновить запись в таблице times, установив time равным '10:10:00' и is_enable равным 1 для записи с id = 1.

7. Зафиксировать изменения в базе данных с помощью db.conn.commit().

8. Выполнить SQL-запрос для получения обновленной записи из таблицы times с id = 1.

9. Проверить, что полученная запись совпадает с ожидаемым значением (1, "10:10:00", 1).

10. Закрыть курсор.

11. Закрыть соединение с базой данных.

Ожидаемый результат: Тест должен подтвердить, что обе таблицы (questions и times) были успешно созданы, а также что обновленная запись в таблице times соответствует ожидаемым значениям. Если таблицы не существуют или обновленная запись не совпадает с ожидаемым результатом, тест должен завершиться с ошибкой.