**Документація по роботі серверу CS\_MiniSQLiteServer v1.0**

1. Запуск сервера:
   1. Перевірка налаштувань на помилки, якщо хоча б один з параметрів не вірний, сервер генерує файл параметрів за замовчуванням та припиняє роботу.
   2. Перевірка прав доступу до файлів бекапу, головної БД, тимчасової бази
   3. “integrity check” головної БД. Якщо БД має помилки, то сервер запише до логу помилки БД і продовжить роботу
   4. Перевірка чи є несинхронізовані запити в тимчасовій БД. Якщо є, то сервер автоматично виконує синхронізацію.
   5. Перехід до режиму очікування користувачів
2. Команди серверу
   1. Командою вважається лексем(а|и), яка входить до *таблиці 1*. Всі інші лексеми сервер сприймає як запит до головної БД
   2. Якщо запит починається лексемою select або SELECT, сервер не блокує БД і виконує запит. Якщо запит починається на будь-яку іншу лексему, сервер блокує головну БД, виконує запит і знімає блокування БД. (в SQLite це має назву “**BEGIN EXCLUSIVE TRANSACTION;” та “COMMIT;”)**

|  |  |  |
| --- | --- | --- |
| **Command** | **Коротко** | **Опис** |
| **UPDATE Config SET PlaceFree ….** | оновити КЕШ PlaceFree | Виконується запит, його результат копіюється до КЕШу PlaceFree. |
| **get\_place\_free** | повернути КЕШ PlaceFree | Повертається PlaceFree з КЕШу. Якщо КЕШ не містить PlaceFree, виконується запит до БД, повертається та копіюється до КЕШу. |
| **backup\_db** | бекап головної бази | починається процес копіювання головної БД у фоновому процесі, при цьому створюється тимчасова БД, в яку записуються всі запити від користувачів крім тих, що починаються на SELECT. Після завершення процесу копіювання виконується перевірка бекапу на помилки за допомоги команди SQLite “PRAGMA integrity\_check;”. Якщо немає помилок, починається процес синхронізації тимчасової БД з головною у фоновому режимі. Всі збережені запити в тимчасову БД по черзі виконуються в основній та видаляються з тимчасової.  Якщо процес створення бекапу запущено і серверу було надіслано команду **"backup\_db"** ще раз, відповідь буде “**backup in progress [35%]”**.  Після успішного виконання бекапу встановлюється таймаут (з налаштувань) та доки він не закінчиться, користувачеві на команду **"backup\_db"** буде надіслана відповідь **"backup db complete [100%]"**. |
| **get\_db\_backup\_progress** | повернути відсоток виконання бекапу | Приклад:   1. “**backup in progress [35%]”** 2. **"backup not started"** |
| **"login <username>"** | встановлення поточного імені клієнту | Дозволяє авторизуватись клієнту. Ця команда виключно для тестування. В штатній роботі не використовується. |
| **"get\_db\_backup"** | повернути файл бекапу | Перевіряє, чи існує в системі файл бекапу БД та повертає цей файл, якщо він існує. Якщо ні, повертає повідомлення про помилку. |
| **"who"** | список усіх клієнтів, які наразі підключені до серверу | Ця команда виключно для тестування. В штатній роботі не використовується |
| **"restore\_db"** | заміна робочої БД на ту, яка вказана в налаштуваннях | Сервер перевіряє новий файл БД на помилки, якщо немає, то блокує будь-яку взаємодію з користувачами та починає процес заміни теперішньої БД на ту, яка вказана у файлі налаштувань. Після заміни сервер продовжує роботу в штатному режимі. |
| **"fibo <number>"** | число Фібоначчі | Для тестування швидкодії серверу в одно поточному та багато поточному режимі було запропоновано додати можливість підрахування числа Фібоначчі. |
| **"exit"** | відключення клієнта від серверу | Сервер розриває з’єднання з поточним клієнтом та звільняє ресурси, які були виділені для роботи з клієнтом |

*Таблиця 1*

3. Пояснення до кожного з налаштувань присутнє при генерацію налаштувань за замовчуванням.