УДК 004.658
DOI: 10.15587/2313-8416.2018.134769

СОВРЕМЕННЫЕ МЕССЕНДЖЕРЫ В КАЧЕСТВЕ ПОМОЩНИКА АДМИНИСТРАТОРА БАЗЫ ДАННЫХ

© В. А. Кожевников, О. Ю. Сабинин, Ю. Е. Шац

В данной статье рассматривается вопрос использования современных мессенджеров и их возможностей администраторами баз данных. Описывается процесс создания бота, который позволяет администратору получать своевременные уведомления о проблемах и ошибках, произошедших с базой данных, а также получать статистику использования. Принципиальным отличием от других подобных продуктов является реализация для мессенджеров, которые в данный момент пользуются популярностью. Предусматривается описание разработки с использованием платформ Telegram, Facebook Messenger и Slack, на основе единой библиотеки

Ключевые слова: *Бот, Java, мессенджеры, Slack, Facebook, Telegram, база данных, PostgreSQL, статистика, администрирование баз данных*

1. Введение

На сегодняшний день существует множество проблем, с которыми сталкиваются администраторы баз данных (БД) и систем управления базами данных (СУБД). Одной из основных проблем является проблема обеспечения сохранности данных. Для СУБД, которые хранят данные предприятий и управляют ими, задача обеспечения сохранности является первостепенной.

В статье рассматриваются вопросы создания утилиты администратора базы данных на основе бота для мессенджеров, основной задачей которого является оповещение администратора о состоянии базы данных, изменении ее производительности и попытках подключения к ней.

2. Анализ литературных данных

Проводя комплекс мер, направленных на защиту данных, администраторы БД стараются как можно лучше защитить их от ошибок и до предела ограничить возможности рядовых пользователей, но по различным причинам могут проявиться самые разнообразные проблемы. В монографии, направленной на сопровождение сервера баз данных PostgreSQL [1], рассматриваются аспекты защиты БД от рядовых пользователей и внутренних ошибок. Также, качественная работа базы данных критически зависит от правильной и своевременной диагностики, для достижения которой необходимо своевременное уведомление администратора. Подобная задача оповещения является частью теории push-уведомлений [2]. В работах, описывающих внутреннее устройство мессенджера Telegram [3], Slack [4] и Facebook Messenger [5], можно найти сведения о создании ботов на их основе.

3. Цель и задачи исследования

Цель исследования заключается в том, чтобы рассмотреть современные мессенджеры, как основу для создания инструментов, которые позволят упростить работу администраторов баз данных.

Для достижения цели были поставлены следующие задачи:

1) Проанализировать возможности СУБД для определения способов мониторинга.

2) Разработать утилиту на основе библиотеки для создания ботов мессенджеров, которая позволит определять:

– Как в текущий момент работает база данных - в целом и по сравнению с другими периодами;

– Какие запросы в наибольшей степени нагружают сервер (центральный процессор, память, системы хранения данных);

– Сколько запросов и какие именно были выполнены, в какое время и какими пользователями

– и т. д.

3) Провести тестирование реализованного бота.

4. Архитектура проекта

Предлагается создать бот на базе мессенджеров, который позволит администратору БД, благодаря набору команд и push-уведомлениям постоянно иметь возможность проверять состояние системы и быть в курсе ее изменений. Такое решение представляется наиболее правильным, поскольку ручная проверка занимает достаточно большое количество времени, в то время как уведомления об экстренных ситуациях (при помощи push-технологии) и возможность быстро узнать об операциях могут значительно сократить временные затраты. Также следует учитывать, что ботами пользуется огромное количество организаций, и на сегодняшний день они доступны на любых платформах, будь то стационарный компьютер, мобильный телефон или веб-браузер.

В качестве мессенджеров для реализации утилиты, были выбраны:

– Slack, как самый популярный корпоративный мессенджер [6];

– Facebook Messenger, как самый популярный мессенджер по всему миру [7];

– Telegram, как наиболее богатая платформа для создания ботов [3].

Так как реализация подобного функционала для разных СУБД является трудоемким процессом, в рамках данной статьи было решено реализовать поддержку лишь одной из них – PostgreSQL, так как эта

СУБД является бесплатной, продвинутой, открытой и быстро набирает популярность, вытесняя конкурентов (включая мощные коммерческие СУБД такие как Oracle, MS SQL Server, DB2 и др.) [8, 9].

Утилита состоит из нескольких модулей:

– Bot Library — библиотека для взаимодействия с различными мессенджерами (Facebook Messanger, Telegram, Slack);

– Pg Log Module — модуль для работы с log-файлами PostgreSQL. Берет из файла информацию о действиях, которые происходят с БД. Анализирует ее на предмет сообщений об ошибках, чтобы выдать в кратчайшие сроки уведомление администратору о происходящем;

– Pg Stats Module — модуль для работы со статистикой, которую предоставляет PostgreSQL с помощью представлений pg_stat. Данный модуль делает запросы к представлениям pg_stat_activity и pg_stat_statements и с помощью Bot Library отправляет данные конечным пользователям.

Компонентная схема программы представлена на рис. 1.

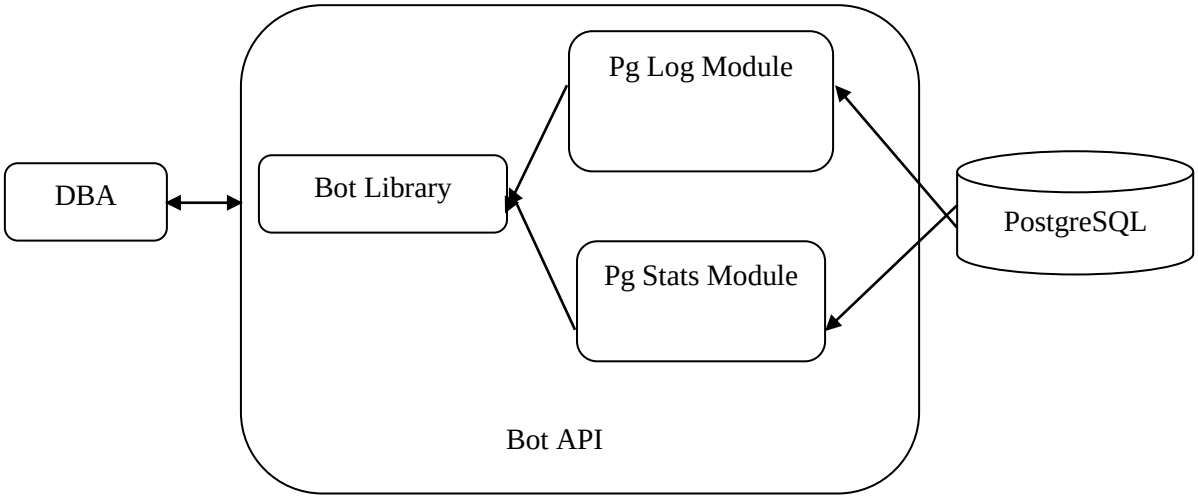

Рис. 1. Компонентная схема программы

5. Сбор статистики базы данных

В СУБД PostgreSQL существует несколько способов сбора информации о БД [10], которые мы и будем использовать для вывода администратору через бот. Основным источником такой информации является модуль pg_stat_statements, представляющий из себя набор процедур, функций и представлений для получения различной статистики о запросах, которые обрабатывает сервер. Для получения и обработки статистики этот модуль предоставляет представление pg_stat_statements и вспомогательные функции pg_stat_statements_reset и pg_stat_statements. На основе полученной статистики администратор может выполнить мониторинг запросов — посмотреть на статистику по времени. Это оказывается крайне полезно для поиска причин различных проблем и в целом для понимания того, что происходит на сервере базы данных.

При использовании набора представлений pg_stat_statements администратор получает сгруппированную статистику, то есть статистику не по каждому запросу, а по группе одинаковых, с точки зрения PostgreSQL, запросов. Помимо информации о запросах, можно получать также сведения о количестве и времени чтения и записи блоков на диске (рис. 2). Все счетчики ведут статистику с момента старта или с момента их сброса администратором БД.

Отдельно стоит отметить, что в представление pg_stat_statements попадают только завершенные запросы. То есть, если запрос долгое время что-то выполняет и до сих пор не закончил работу, то его будет видно только в представлении pg_stat_activity.

```
postgres=# \d pg_stat_statements;
View "public.pg_stat_statements"
        Column        |       Type
----------------------+------------------
 userid               | oid
 dbid                 | oid
 queryid              | bigint
 query                | text
 calls                | bigint
 total_time           | double precision
 rows                 | bigint
 shared_blks_hit      | bigint
 shared_blks_read     | bigint
 shared_blks_dirtied  | bigint
 shared_blks_written  | bigint
 local_blks_hit       | bigint
 local_blks_read      | bigint
 local_blks_dirtied   | bigint
 local_blks_written   | bigint
 temp_blks_read       | bigint
 temp_blks_written    | bigint
 blk_read_time        | double precision
 blk_write_time       | double precision
```

Рис. 2. Переменные, в которых содержится информация о БД

Также, информацию о пользователях БД и выполненных ими транзакциях можно получать из log-файлов PostgreSQL (рис. 3). Для преобразования информации и вывода пользователю, по log-файлам нужно производить контекстный поиск по интересу-

ющим администратора (пользователя утилиты) запросам. Например, запись connection authorized говорит о подключении пользователя, запись statement – о запросах, выполняемых пользователем, а запись duration обозначает время выполнения запроса.

```
LOG:  database system is ready to accept connections
LOG:  connection received: host=127.0.0.1 port=33291
LOG:  connection authorized: user=Julia database=postgres
LOG:  statement: drop table Test;
LOG:  duration: 48.856 ms
LOG:  statement: create table Test (id Integer, description Text);
LOG:  duration: 75.258 ms
```

Рис. 3. Пример log - файла PostgreSQL

6. Разработка модуля Bot Library

В основе программирования библиотеки для работы с ботами лежит понимание того, что веб-сервер устанавливает так называемые вебхуки (Webhooks) – адреса, на которые сервер мессенджера будет присылать информацию о событиях в чате с пользователем. По сути, сервер получает сообщение, и возвращает пользователю ответ, который был прописан разработчиком. Поэтому можно считать, что большая часть программного интерфейса ботов сводится к получению или отправке информации в формате JSON на специальные адреса на сервере мессенджера.

Для разработки модуля было решено использовать язык Java, т. к. это один из самых популярных языков программирования [11], в стандартной библиотеке которого имеется все необходимое для того, чтобы реализовать взаимодействие с мессенджерами [12]. Использование только стандартных классов и методов связано с необходимостью писать множество повторяющегося низкоуровневого кода, поэтому удобнее использовать библиотеки, упрощающие взаимодействие с мессенджерами. Для этого нами был реализован набор библиотек, который является составной частью Bot Library, описанной ниже.

Для того чтобы созданные библиотеки можно было использовать в других проектах, необходимо воспользоваться фреймворком для автоматизации сборки (например, таких как Maven или Gradle) для подключения внешних зависимостей из репозиториев, таких как Maven Repository [13]. Поэтому для удобного подключения реализованной библиотеки был использован данный подход.

После того, как библиотеки были созданы и собраны, основная задача состояла в том, чтобы объединить их в библиотеку Bot Library, дабы разработчик мог сразу же транслировать своего бота для нескольких платформ.

Рассмотрим пример тривиального бота, который будет отправлять приветствие на каждое полученное сообщение:

Для начала был создан объект обработчика сообщений (BotHandler) и реализовали в нем метод OnMessage, определяющий отклик на сообщения конечных пользователей (рис. 4).

Далее, идет подключение к аккаунтам на каждой из платформ (рис. 5).

Остается соединить аккаунты ботов с обработчиком. Для этого передаем в обработчик, объявленный ранее, аккаунты ботов, передаем дополнительные настройки из конфигурационного файла и запускаем обработку (рис. 6). В данный момент запуск обработчика полностью блокирует поток в ожидании получаемого сообщения. В будущем планируется реализовать асинхронную обработку.

```
public static void main(String[] args) {
    BotHandler handler = new BotHandler() {
        @Override
        public void onMessage(Message message) {
            Message reply = new TextMessage(message.getChat(), text: "Hi!");
            try {
                this.send(reply);
            } catch (SendMessageException e) {
                System.out.println(e.getMessage());
            }
        }
    };
```

Рис. 4. Определения обработчика сообщений

```
Bot slackBot = new SlackBot( identifier: "someIdentifier", password: "somePassword");
Bot telegramBot = new TelegramBot( token: "someToken");
Bot facebookBot = new FacebookBot( token: "someToken");
```

Рис. 5. Пример подключения

```
handler
    .handle(slackBot)
    .handle(telegramBot)
    .handle(facebookBot)
    .configFile("config.properties")
    .start();
```

Рис. 6. Настройка и запуск обработчика сообщений

7. Анализ и тестирование

Библиотека, использовавшаяся для создания утилиты администратора, может конкурировать с уже существующими решениями. Сравнение характеристик приведено в табл. 1.

Следует отметить, что для сравнения не были взяты Chatfuel, Recast.AI, Pandorabots, Microsoft Bot Framework (bot builder SDK), потому как они имеют иной функционал.

Таблица 1

Сравнение реализованной библиотеки с существующими

Библиотеки			
Реализованная библиотека	*BotKit*	*Claudia Bot Builder*	*Universal Bot Framework*
Используемый язык программирования			
Java	Java-Script	Java-Script	JavaScript
Поддерживаемые системы			
Slack, Facebook Messenger, Telegram	Slack, Facebook Messenger, Twilio, Microsoft Bot Framework	Facebook Messenger, Slack, Skype, Viber, Telegram, Twilio, Amazon Alexa, Line, Kik, GroupMe	Facebook Messenger,Telegram, Kik, Skype
На кого нацелены			
На разработчиков	Разработчиков и рядовых пользователей	На разработчиков	На разработчиков
Объединение интерфейсов			
Полная поддержка	Полная поддержка	Нет поддержки дополнительных функций	Нет поддержки дополнительных функций
Поддержка полного функционала			
Поддерживает	Поддерживает	Поддерживает	Не поддерживает

Из данных, приведенных в табл. 1 видно, что реализованная библиотека нуждается в добавлении новых платформ.

Для тестирования работоспособности утилиты, была создана база данных, в которой хранятся данные для выполнения тестовых запросов на получение уведомлений. В качестве сервера был использован Open Server и поставляемое с ним программное обеспечение, предоставляющее графический интерфейс для работы с базой данных – PgAdmin, позволяющий ускорить процесс настройки окружения для разработки. Было создано несколько пользователей, посредством которых проведено несколько транзакций, а также эмулированы ошибки для проверки push-уведомлений. В качестве иллюстрации, на рис. 7, для использованных платформ можно увидеть текст запроса администратора для поиска выполненных «select»-операций и результат проведенного поиска.

Таким образом, администратор путем получения уведомлений или запросов к боту, может иметь актуальную информацию о состоянии БД, пользователях, выполняемых скриптах, ошибках и т. д.

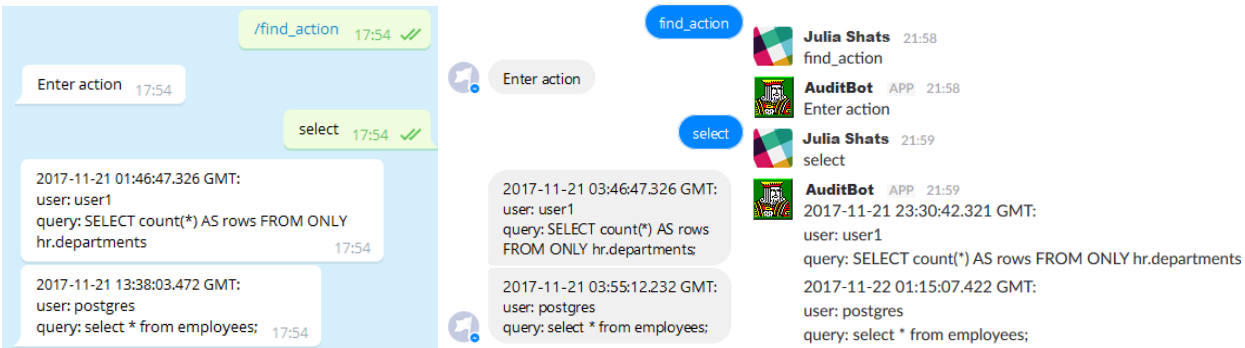

Рис. 7. Проверка выполненных запросов к БД на платформах Telegram, Facebook и Slack

8. Результаты исследования

Исходя из предложенных в статье принципов построения, утилита позволяет администратору баз данных в любое время удобно и быстро искать не только информацию о состоянии базы данных, статистике использования, но и, при помощи push-уведомлений, мгновенно получать уведомления о критических ситуациях. Достоинством предложенной архитектуры является ее кроссплатформенность, что позволяет использовать реализованную систему на широком круге устройств (клиенты на стационарном компьютере, веб-браузер, мобильный телефон).

9. Выводы

1. В статье рассмотрены возможности СУБД PostgreSQL, которые позволяют собирать информацию, необходимую для работы администраторов баз данных.

2. Создана архитектура, на основе библиотеки, позволяющей на своей основе, создавать боты любого содержания сразу на нескольких платформах, что дает разработчикам прирост скорости разработки.

3. Проанализирована работоспособность утилиты и выявлены такие возможности для улучшения реализованного продукта, как:

– добавление новых мессенджеров;

– добавление графиков влияния на нагрузку сервера;

– добавление возможности частичного управления сервером (например, ограничение количества подключений, блокирование пользователей).

Литература

1. Drake J., Worsley J. Practical PostgreSQL. O'Reilly Media, Inc., 2002. 640 p.

2. Introduction to Push Notifications // Google Developers. 2018. URL: https://developers.google.com/web/ilt/pwa/introduction-to-push-notifications

3. Telegram Bot API // Telegram. URL: https://core.telegram.org/bots/api

4. Документация Slack // Slack. 2018. URL: https://slack.com/developers

5. Документация Facebook // Facebook for Developers. 2018. URL: https://developers.facebook.com/docs/messenger-platform

6. Duffy J., Moore B. The Best Business Messaging Apps of 2018 // PCMAG. 2018. URL: https://www.pcmag.com/roundup/355674/the-best-team-messaging-apps

7. Most popular messaging apps 2018 // Statista. 2018. URL: https://www.statista.com/statistics/258749/most-popular-global-mobile-messenger-apps/

8. Smith L. What PostgreSQL has over other open source SQL databases: Part I // Compose Articles. 2015. URL: https://www.compose. com/articles/what-postgresql-has-over-other-open-source-sql-databases/

9. Smith L. What PostgreSQL has over other open source SQL databases: Part II // Compose Articles. 2015. URL: https://www.compose.com/articles/what-postgresql-has-over-other-open-source-sql-databases-part-ii/

10. PostgreSQL 9 Administration Cookbook / Riggs S. et. al. Packt Publishing, 2015. 504 p.

11. Putano B. Most Popular and Influential Programming Languages of 2018 // Stackify. 2017. URL: https://stackify.com/popular-programming-languages-2018/

12. Kozhevnikov V. A., Sabinin O. Y., Shats J. E. Library Development for Creating Bots on Slack, Telegram and Facebook Messengers // Theoretical & Applied Science. 2017. Vol. 50, Issue 6. P. 59–62. doi: http://doi.org/10.15863/tas.2017.06.50.4

13. O'Brien T., Van Zyl J. Maven: The Definitive Guide. O'Reilly Media, Inc., 2009. 250 p.

Рекомендовано до публікації д-р техн. наук, професор Колосова О. В.
Дата надходження рукопису 08.05.2018

Кожевников Вадим Андреевич, старший преподаватель, кафедра компьютерных интеллектуальных технологий, Санкт-Петербургский политехнический университет Петра Великого, ул. Политехническая, 29, г. Санкт-Петербург, Российская Федерация, 195251
E-mail: vadim.kozhevnikov@gmail.com

Сабинин Олег Юрьевич, доцент, кафедра компьютерных интеллектуальных технологий, Санкт-Петербургский политехнический университет Петра Великого, ул. Политехническая, 29, г. Санкт-Петербург, Российская Федерация, 195251
E-mail: olegsabinin@mail.ru

Шац Юлия Ефимовна, кафедра компьютерных интеллектуальных технологий, Санкт-Петербургский политехнический университет Петра Великого, ул. Политехническая, 29, г. Санкт-Петербург, Российская Федерация, 195251
E-mail: julia7476@gmail.com