Оглавление

1 Аналитическая часть

В данном разделе описана структура рабочей программы дисциплины. Представлен анализ способов хранения данных и систем управления базами данных, оптимальных для решения поставленной задачи. Описаны проблемы кэшированных данных и представлены методы их решения.

1.1 Формализация задачи

Необходимо спроектировать и реализовать базу данных для онлайн-мониторинга состояния трасс и подъемников горнолыжного курорта. Также необходимо разработать интерфейс, позволяющий работать с данной базой для получения и изменения хранящейся в ней информации и мониторинга очередей к подъемникам в онлайн-режиме. Реализовать, как минимум, три вида ролей – пользователь, сотрудник лыжного патруля и администратор.

1.2 Формализация данных

База данных должна хранить информацию о:

- трассах;
- подъемниках;
- связях трасс и подъемников (на одном подъемнике можно добраться до нескольких трасс, и до одной трассы можно добраться на нескольких подъемниках);
- турникетах;
- проездных картах;
- считываниях карт на турникетах подъемников;
- сообщениях о происшествиях;

- пользователях;
- группах пользователей.

В таблице 1.1 приведены категории и сведения о данных.

Таблица 1.1: Категории и сведения о данных

Категория	**Сведения**
Трассы	ID трассы, название трассы, уровень сложности, открытость/закрытость.
Подъемники	ID подъемника, название подъемника, открытость/закрытость, количество мест, время подъема, время в очереди.
Связи трасс и подъемников	ID записи, ID подъемника, ID трассы.
Турникеты	ID турникета, ID подъемника, открытость/закрытость.
Проездные карты	ID карты, дата и время активации, тип.
Считывания карт на турникетах подъемников	ID записи, ID турникета, ID карты, дата и время считывания.
Сообщения о происшествиях	ID сообщения, ID отправителя, ID прочитавшего, текст сообщения.
Пользователи	ID пользователя, ID карты, email (логин), пароль, ID группы пользователей.
Группы пользователей	ID группы пользователей, права доступа.

1.3 Типы пользователей

В соответствии с поставленной задачей необходимо разработать приложение с возможностью аутентификации пользователей, что делит их, прежде всего, на авторизованных и неавторизованных. для управления приложением необходима ролевая модель: авторизованный (обычный) пользователь, сотрудник лыжного патруля и администратор.

Для каждого типа пользователя предусмотрен свой набор функций:

- неавторизованный пользователь:
  - регистрация,
  - аутентификация,

  - просмотр информации о состоянии трасс и подъемников,
  - просмотр информации о связях трасс и подъемников;
- авторизованный пользователь:
  - выход,
  - просмотр информации о состоянии трасс и подъемников,
  - просмотр информации о связях трасс и подъемников,
  - отправка сообщений о происшествиях;
- сотрудник лыжного патруля:
  - выход,
  - просмотр и изменение информации о состоянии трасс и подъемников,
  - просмотр и изменение информации о связях трасс и подъемников,
  - просмотр сообщений о происшествиях;
- администратор:
  - выход,
  - просмотр и изменение всей информации, доступной в базе данных, в том числе права доступа групп и отдельных пользователей.

1.4 Анализ баз данных и систем управления базами данных

Для реализации поставленной задачи необходимо выбрать подходящую базу данных (БД) и систему управления базой данных (СУБД).

БД – это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе [5]. СУБД – это совокупность программных и лингвистических

средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных [5].

1.4.1 Классификация баз данных по месту хранения информации

По месту хранения информации БД можно разделить на [6]:

- традиционные, которые хранят информацию на жестком диске или другом постоянном носителе;
- in-memory databases (IMDB) (резидентные базы данных), которые хранят информацию непосредственно в оперативной памяти.

IMDB появились как ответ традиционным БД в связи со снижением стоимости оперативной памяти, что позволяет хранить весь набор операционных данных непосредственно в памяти, увеличивая тем самым скорость их обработки более чем в 1000 раз [7].

Ключевыми преимуществами IMDB, в сравнении с традиционными БД, считаются следующие [8]:

- быстрота выполнения операций;
- эффективное сохранение зафиксированных данных, которые используются не часто, на жестком диске;
- высокая пропускная способность систем, критичных к производительности.

Обратной стороной этих достоинств являются следующие недостатки:

- однопоточность и эффективная утилизация только одного ядра ЦП, что не позволяет в полной мере воспользоваться возможностями современных многоядерных серверов;
- энергозависимость и привязка к размеру оперативной памяти.

В практическом плане IMDB-системы особенно востребованы в тех приложениях работы с данными в реальном времени, где требуется минимальное время отклика [9].

Основным требованием к разрабатываемой БД является предоставление возможности **онлайн**-мониторинга состояния объектов горнолыжного курорта, то есть задача является типовым примером использования in-memory БД. И поскольку в современных СУБД существуют надежные и достаточно простые способы устранения указанных недостатков IMDB, было принято решение использовать именно этот подход к хранению данных.

1.5 Хранение данных о рабочих программах дисциплины

Система, разрабатываемая в рамках курсового проекта, предполагает собой приложение, которое является микросервисом [10] одной большой системы – системы управления обучения.

Предполагается, что доступ к разрабатываемому приложению будем иметь лишь только «ядро» этой системы. При этом, только у одного типа пользователя системы есть доступ к данным, хранящимся в приложении – преподавателю. Состояние гонки (англ. Race condition [11]) можно исключить - каждый преподаватель работает только с информацией из файлов, которые он самостоятельно загрузил в базу данных.

Для хранения данных о рабочей программы дисциплины необходимо использовать строго структурированную и типизированную базу данных, потому что вся информация, предоставленная в файлах программы имеет чётко выраженную структуру, которая не будет меняться от дисциплины к дисциплине.

1.5.1 Классификация баз данных по способу хранения

Базы данных, по способу хранения, делятся на две группы – строковые и колоночные. Каждый из этих типов служит для выполнения для определенного рода задач.

Строковые базы данных

Строковыми базами даных называются такие базы данных, записи которых в памяти представляются построчно. Строковые баз данных используются в транзакционных системах (англ. OLTP [12]). Для таких систем характерно большое количество коротких транзакций с операциями вставки, обновления и удаления данных - `INSERT`, `UPDATE`, `DELETE`.

Основной упор в системах OLTP делается на очень быструю обработку запросов, поддержание целостности данных в средах с множественным доступом и эффективность, которая измеряется количеством транзакций в секунду.

Схемой, используемой для хранения транзакционных баз данных, является модель сущностей, которая включает в себя запросы, обращающиеся к отдельным записям. Так же, в OLTP-системах есть подробные и текущие данных.

Колоночные базы данных

Колоночными базами данных называются базы данных, записи которых в памяти представляются по столбцам. Колоночные базы данных используется в аналитических системах (англ. OLAP [13]). OLAP характеризуется низким объемом транзакций, а запросы часто сложны и включают в себя агрегацию. Время отклика для таких систем является мерой эффективности.

OLAP-системы широко используются методами интеллектуального анализа данных. В таких базах есть агрегированные, исторические данные, хранящиеся в многомерных схемах.

1.5.2 Выбор модели хранения данных для решения задачи

Для решения задачи построчное хранение данных преобладает над колоночным хранением по нескольким причинам:

- задача предполагает постоянное добавление и изменение данных;
- задача предполагает быструю отзывчивость на запросы пользователя;
- задача не предполагает выполнения аналитических запросов;

1.5.3 Обзор СУБД с построчным хранением

В данном подразделе буду рассмотрены популярные построчные СУБД, которые могут быть использованы для реализации хранения в разрабатываемом программном продукте.

PostgreSQL

PostgreSQL [14] – это свободно распространяемая объектно-реляционная система управления базами данных, наиболее развитая из открытых СУБД в мире и являющаяся реальной альтернативой коммерческим базам данных [15].

PostgreSQL предоставляет транзакции со свойствами атомарности, согласованности, изоляции, долговечности (ACID [16]), автоматически обновляемые представления, материализованные представления, триггеры, внешние ключи и хранимые процедуры. Данная СУБД предназначена для обработки ряда рабочих нагрузок, от отдельных компьютеров до хранилищ данных или веб-сервисов с множеством одновременных пользователей.

Рассматриваемая СУБД управляет параллелизмом с помощью технологии управления многоверсионным параллелизмом (англ. MVCC [17]). Эта технология дает каждой транзакции «снимок» текущего состояния базы

данных, позволяя вносить изменения, не затрагивая другие транзакции. Это в значительной степени устраняет необходимость в блокировках чтения (англ. read lock [18]) и гарантирует, что база данных поддерживает принципы ACID.

Oracle Database

Oracle Database [19] – объектно-реляционная система управления базами данных компании Oracle [20]. На данный момент, рассматриваемая СУБД является самой популярной в мире. [21]

Все транзакции Oracle Database соответствуют обладают свойствами ACID, поддерживает триггеры, внешние ключи и хранимые процедуры. Данная СУБД подходит для разнообразных рабочих нагрузок и может использоваться практически в любых задачах. Особенностью Oracle Database является быстрая работа с большими массивами данных.

Oracle Database может использовать один или более методов параллелизма. Сюда входят механизмы блокировки для гарантии монопольного использования таблицы одной транзакцией, методы временных меток, которые разрешают сериализацию транзакций и планирование транзакций на основе проверки достоверности.

MySQL

MySQL [22] – свободная реляционная система управления базами данных. Разработку и поддержку MySQL осуществляет корпорация Oracle.

Рассматриваемая СУБД имеет два основных движка хранения данных: InnoDB [23] и myISAM [24]. Движок InnoDB полностью полностью совместим с принципами ACID, в отличии от движка myISAM. СУБД MySQL подходит для использования при разработке веб-приложений, что объясняется очень тесной интеграцией с популярными языками PHP [25] и Perl [26].

Реализация параллелизма в СУБД MySQL реализовано с помощью механизма блокировок, который обеспечивает одновременный доступ к данным.

1.5.4 Выбор СУБД для решения задачи

Для решения задачи была выбрана СУБД PostgreSQL, потому что данная СУБД имеет поддержку языка plpython3u [27], который упрощает процесс интеграции базы данных в разрабатываемое приложение. Кроме того, PostgreSQL проста в развертывании.

1.6 Кэширование данных

Для ускорения быстродействия разрабатываемого приложения, можно прибегнуть к кэшированию данных. Для кэширования данных можно использовать NoSQL [28] in-memory базы данных. Такие базы данных хранят данные в оперативной памяти, что обеспечивает более быстрый доступ к данным.

1.6.1 Проблемы кэширования данных

Синхронизация данных

Приложение пишет в кэш, и в базу данных, которые между собой никак не синхронизируются. Таким образом возникает несогласованность данных. Например, в случае разрабатываемого приложения, возможна ситуация, когда данные удаляются из хранилища и их нужно удалить из кэша. Эту проблему можно решить установкой триггеров в базе данных хранения рабочих программ дисциплин, которые буду срабатывать на изменение / удаление данных и синхронизировать актуальные данные в кэше.

Проблема «холодного старта»

Когда кэш только развертывается, он пуст и в нем нет никаких данных. Все запросы идут напрямую в базу данных, и только спустя какое-то время кэш будет «разогрет» и будет работать в полную силу. Эту проблему можно решить, выбрав СУБД с журналированием всех операций: при

перезагрузке можно восстановить предыдущее состояние кэша с помощью журнала событий, который хранится на диске. При этом, при перезапуске кэша, нужно синхронизировать данные с хранилищем: возможно, какие-то данные находящиеся в кэше перестали быть актуальными за время его перезагрузки.

1.6.2 Обзор in-memory NoSQL СУБД

Tarantool

Tarantool [29] – это платформа in-memory вычислений с гибкой схемой хранения данных для эффективного создания высоконагруженных приложений. Включает себя базу данных и сервер приложений на языке программирования Lua [30].

Tarantool обладает высокой скоростью работы по сравнению с традиционными СУЬД. При этом, в рассматриваемой платформе для транзакций реализованы свойства ACID, репликация master-slave [31] и master-master [32], как и в традиционных СУБД.

Для хранения данных используется кортежи (англ. tuple) данных. Кортеж – это массив не типизированных данных. Кортежи объединяются в спейсы (англ. space), аналоги таблицы из реляционной модели хранения данных. Спейс – коллекция кортежей, кортеж – коллекция полей.

В рассматриваемой СУБД реализованы два движка хранения данных: memtx [33] и vinyl [33]. Первый хранит все данные в оперативной памяти, а второй на диске. Для каждого спейса можно задавать различный движок хранения данных.

Каждый спейс должен быть проиндексирован первичным ключом. Кроме того, поддерживается неограниченное количество вторичных ключей. Каждый из ключей может быть составным.

В Tarantool реализован механизм «снимков» текущего состояния хранилища и журналирования всех операций, что позволяет восстановить состояние базы данных после ее перезагрузки.

Redis

Redis [34] – резидентная система управлениями базами данных класса NoSQL с открытым исходным кодом. Основной структурой данных, с которой работает Redis является структура типа «ключ-значение». Данная СУБД используется как для хранения данных, так и для реализации кэшей и брокеров сообщений.

Redis хранит данные в оперативной памяти и снабжена механизмом «снимков» и журналирования, что обеспечивает постоянное хранение данных. Предоставляются операции для реализации механизма обмена сообщениями в шаблоне «издатель-подписчик»: с его помощью приложения могут создавать программные каналы, подписываться на них и помещать в эти каналы сообщения, которые будут получены всеми подписчиками. Существует поддержка репликации данных типа master-slave, транзакций и пакетной обработки комманд.

Все данные Redis хранит в виде словаря, в котором ключи связаны со своими значениями. Ключевое отличие Redis от других хранилищ данных заключается в том, что значения этих ключей не ограничиваются строками. Поддерживаются следующие абстрактные типы данных:

- строки;
- списки;
- множества;
- хеш-таблицы;
- упорядоченные множества.

Тип данных значения определяет, какие операции доступные для него; поддежриваются высокоуровневые операции: например, объединение, разность или сортировка наборов.

1.6.3 Выбор СУБД для решения задачи

Для кэширования данных была выбрана СУБД Tarantool, так как она проста в развертывании и переносимости, и имеет подходящие коннекторы для базы данных PostgreSQL.

Вывод

В данном разделе:

- рассмотрена структура рабочей программы дисциплины и выявлены её наиболее интересные части;
- проанализированы способы хранения информации для система и выбраны оптимальные способы для решения поставленной задачи;
- проведен анализ СУБД, используемых для решения задачи и также выбраны оптимальные информационные системы;
- рассмотрена проблема актуальности кэшируемых данных и предложенно ее решение;
- формализованны данные, используемые в системе.

Литература

[1] Положение о порядке разработки и утверждение рабочей программы дисциплины [Электронный ресурс]. Режим доступа: https://www.volgmed.ru/uploads/files/2010-11/1180-polozhenie_o_poryadke_razrabotki_i_utverzhdeniya_rabochej_programmy_uchebnoj_discipliny_(kursa).doc (дата обращения: 07.06.2021).

[2] Learning Management System (LMS) — HSE [Электронный ресурс]. Режим доступа: https://www.hse.ru/en/studyspravka/lms_student/ (дата обращения: 07.06.2021).

[3] Microsoft Word - Word Processing Software | Microsoft 365 [Электронный ресурс]. Режим доступа: https://www.microsoft.com/en-us/microsoft-365/word (дата обращения: 07.06.2021).

[4] What is a REST API? - Red Hat [Электронный ресурс]. Режим доступа: https://www.redhat.com/en/topics/api/what-is-a-rest-api (дата обращения: 07.06.2021).

[5] Что такое база данных | Oracle Россия и СНГ [Электронный ресурс]. Режим доступа: https://www.oracle.com/ru/database/what-is-database/ (дата обращения: 29.04.2022).

[6] Системы и технологии баз данных в памяти | Microsoft [Электронный ресурс]. Режим доступа: https://docs.microsoft.com/ru-ru/sql/relational-databases/in-memory-database?view=sql-server-ver15 (дата обращения: 29.04.2022).

[7] In-Memory. База данных в оперативной памяти |ECM-Journal [Электронный ресурс]. Режим доступа: https://ecm-journal.ru/material/In-Memory-Baza-dannykh-v-operativnojj-pamjati (дата обращения: 29.04.2022).

[8] In-Memory Database [Электронный ресурс]. Режим доступа: https://ru.bmstu.wiki/IMDB_(In-memory_Database) (дата обращения: 29.04.2022).

[9] 4 крупных примера внедрения Tarantool | Big Data School [Электронный ресурс]. Режим доступа: https://www.bigdataschool.ru/blog/tarantool-use-cases-and-advantages.html (дата обращения: 29.04.2022).

[10] Что такое микросервисная архитектура: простое объяснение | MCS Mail.ru [Электронный ресурс]. Режим доступа: https://mcs.mail.ru/blog/prostym-jazykom-o-mikroservisnoj-arhitekture (дата обращения: 07.06.2021).

[11] Race conditions and deadlocks - Microsoft Docs [Электронный ресурс]. Режим доступа: https://docs.microsoft.com/en-us/troubleshoot/dotnet/visual-basic/race-conditions-deadlocks (дата обращения: 07.06.2021).

[12] What is OLTP? | IBM [Электронный ресурс]. Режим доступа: https://www.ibm.com/cloud/learn/oltp (дата обращения: 07.06.2021).

[13] What is OLAP? | IBM [Электронный ресурс]. Режим доступа: https://www.ibm.com/cloud/learn/olap (дата обращения: 07.06.2021).

[14] PostgreSQL: Документация. [Электронный ресурс]. Режим доступа: https://postgrespro.ru/docs/postgresql/ (дата обращения: 07.06.2021).

[15] PostgreSQL: вчера, сегодня, завтра [Электронный ресурс]. Режим доступа: https://postgrespro.ru/blog/media/17768 (дата обращения: 07.06.2021).

[16] Транзакции, ACID, CAP | GeekBrains [Электронный ресурс]. Режим доступа: https://gb.ru/posts/acid_cap_transactions (дата обращения: 07.06.2021).

[17] Documentation: 12: 13.1. Introduction - PostgreSQL [Электронный ресурс]. Режим доступа: https://www.postgresql.org/docs/12/mvcc-intro.html (дата обращения: 07.06.2021).

[18] Применение блокировок чтения/записи | IBM [Электронный ресурс]. Режим доступа: https://www.ibm.com/docs/ru/aix/7.2?topic=programming-using-readwrite-locks (дата обращения: 07.06.2021).

[19] SQL Language | Oracle[Электронный ресурс]. Режим доступа: https://www.oracle.com/database/technologies/appdev/sql.html (дата обращения: 07.06.2021).

[20] Oracle | Integrated Cloud Applications and Platform Services [Электронный ресурс]. Режим доступа: https://www.oracle.com/index.html (дата обращения: 07.06.2021).

[21] DB-Engines Ranking [Электронный ресурс]. Режим доступа: https://db-engines.com/en/ranking (дата обращения: 07.06.2021).

[22] MySQL Database Service is a fully managed database service to deploy cloud-native applications. [Электронный ресурс]. Режим доступа: https://www.mysql.com/ (дата обращения: 07.06.2021).

[23] MySQL Reference Manual 8.0: The InnoDB Storage Enginee [Электронный ресурс]. Режим доступа: https://dev.mysql.com/doc/refman/8.0/en/innodb-storage-engine.html (дата обращения: 07.06.2021).

[24] MySQL Reference Manual 16.2: The MyISAM Storage Engine [Электронный ресурс]. Режим доступа: https://dev.mysql.com/doc/refman/8.0/en/myisam-storage-engine.html (дата обращения: 07.06.2021).

[25] PHP: Hypertext Preprocessor [Электронный ресурс]. Режим доступа: https://www.php.net/ (дата обращения: 07.06.2021).

[26] The Perl Programming Language [Электронный ресурс]. Режим доступа: https://www.perl.org/ (дата обращения: 07.06.2021).

[27] PostgreSQL: Документация: 9.6: 44.1. Python 2 и Python 3. [Электронный ресурс]. Режим доступа: https://postgrespro.ru/docs/postgresql/9.6/plpython-python23 (дата обращения: 07.06.2021).

[28] Что такое NoSQL? | Amazon AWS [Электронный ресурс]. Режим доступа: https://aws.amazon.com/ru/nosql/ (дата обращения: 07.06.2021).

[29] Tarantool – Платформа In-memory вычислений [Электронный ресурс]. Режим доступа: https://www.tarantool.io/ru/ (дата обращения: 07.06.2021).

[30] The Programming Language Lua [Электронный ресурс]. Режим доступа: http://www.lua.org/ (дата обращения: 07.06.2021).

[31] Tech Confronts Its Use of the Labels «Master» and «Slave» [Электронный ресурс]. Режим доступа: https://www.wired.com/story/tech-confronts-use-labels-master-slave/ (дата обращения: 07.06.2021).

[32] How To Set Up MySQL Master-Master Replication [Электронный ресурс]. Режим доступа: https://www.digitalocean.com/community/tutorials/how-to-set-up-mysql-master-master-replication (дата обращения: 07.06.2021).

[33] Движки базы данных | Tarantool [Электронный ресурс]. Режим доступа: https://www.tarantool.io/ru/doc/latest/book/box/engines/ (дата обращения: 07.06.2021).

[34] Redis is an open source (BSD licensed), in-memory data structure store, used as a database, cache, and message broker [Электронный ресурс]. Режим доступа: https://redis.io/ (дата обращения: 07.06.2021).

[35] LRU, метод вытеснения из кэша | Habr [Электронный ресурс]. Режим доступа: https://habr.com/ru/post/136758/ (дата обращения: 07.06.2021).

[36] The official home of the Python Programming Language. [Электронный ресурс]. Режим доступа: https://www.python.org/ (дата обращения: 07.06.2021).

[37] Welcome to Flask — Flask Documentation (2.0.x) [Электронный ресурс]. Режим доступа: https://flask.palletsprojects.com/en/2.0.x/ (дата обращения: 07.06.2021).

[38] Python driver for Tarantool - GitHub [Электронный ресурс]. Режим доступа: https://github.com/tarantool/tarantool-python (дата обращения: 07.06.2021).

[39] Psycopg – PostgreSQL database adapter for Python [Электронный ресурс]. Режим доступа: https://www.psycopg.org/docs/ (дата обращения: 07.06.2021).

[40] Docker: Empowering App Development for Developers [Электронный ресурс]. Режим доступа: https://www.docker.com/ (дата обращения: 07.06.2021).

[41] pytest: helps you write better programs — pytest documentation [Электронный ресурс]. Режим доступа: https://docs.pytest.org/en/6.2.x/ (дата обращения: 07.06.2021).