МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ

по дисциплине «Введение в нереляционные базы данных»

Тема: Разработка сервиса хранения данных биржевой торговли Forex

Студенты гр. 0381 _____ Прохоров Б.В.

_____ Соколов Д.В.

Преподаватель _____ Заславский М.М.

Санкт-Петербург

2023

ЗАДАНИЕ

Студенты

Прохоров Б.В.

Соколов Д.В.

Группа 0381

Тема работы: Разработка сервиса хранения данных биржевой торговли Forex

Исходные данные:

Необходимо реализовать веб приложение, которое будет хранить / обеспечивать поиск / анализ и визуализацию данных о рынке по указанным периодам / запросам / валютным парам. Необходимые (но не достаточные) фичи - списки валютных пар, страница архива, визуализация отдельного сигнала на графике для конкретной валютной пары, анализ для сигнала.

Содержание пояснительной записки:

«Содержание»

«Введение»

«Качественные требования к решению»

«Сценарии использования»

«Модель данных»

«Разработка приложения»

«Вывод»

«Приложение»

Предполагаемый объем пояснительной записки:

Не менее 00 страниц.

Дата выдачи задания: 22.09.2023

Дата сдачи реферата: 24.12.2023

Дата защиты реферата: 24.12.2023

Студенты гр. 0381 _____ Прохоров Б.В.

_____ Соколов Д.В.

Преподаватель _____ Заславский М.М.

АННОТАЦИЯ

В рамках данного курса предполагалось разработать какое-либо приложение в команде на одну из поставленных тем. Была выбрана тема создания веб приложение, которое позволяло бы хранить, обеспечивать поиск, анализ и визуализацию данных о рынке по указанным периодам, запросам для валютных пар. В качестве СУБД была выбрана MongoDB [1], поскольку она лучше подходит для разработки такой системы. Во внимание будут приниматься такие аспекты как производительность и удобство разработки. Найти исходный код и всю дополнительную информацию можно по ссылке: https://github.com/moevm/nosql2h23-forex

ANNOTATION

As part of this course, it was supposed to develop an application in a team on one of the given topics. The topic was chosen to create a web application that would allow storing, searching, analyzing and visualizing market data for specified periods and queries for currency pairs. MongoDB was chosen as the DBMS because it is better suited for developing such a system. Aspects such as performance and ease of development will be taken into account. Source code and all additional information at the link: https://github.com/moevm/nosql2h23-forex

Оглавление

1. Введение

Цель работы – создать высокопроизводительное и удобное решение для хранения и анализа данных о валютных парах.

Было решено разработать веб-приложение, которое позволит хранить в электронном виде данные о валютных парах, при этом позволяющее с ними удобно взаимодействовать.

2. Качественные требования к решению

Требуется разработать веб-приложение с использование СУБД MongoDB. Backend должен быть реализован на Python 3.10 с использованием фреймворка Django. Frontend должен быть реализован с помощью фреймворка Angular.

3. Сценарии использования

1. Отображение сигнала определенной валютной пары

Предусловие: пользователь находится на главной странице.

Пользователь: кликает на иконку лупы на боковой панели слева.

Система: отображает модальное окошко (по центру окна с затемнением остального интерфейса) со списком валютным пар и строкой для поиска.

Пользователь: вводит в строке поиска необходимый запрос и кликом ЛКМ на значок "+" рядом с нужной строкой выбирает нужную валютную пару

Пользователь: нажимает на кнопку "Закрыть".

Система: закрывает модальное окошко и отображает сигнал соответствующей валютной пары.

Рисунок 1 – Отображение сигнала определенной валютной пары.

Открытие модального окна

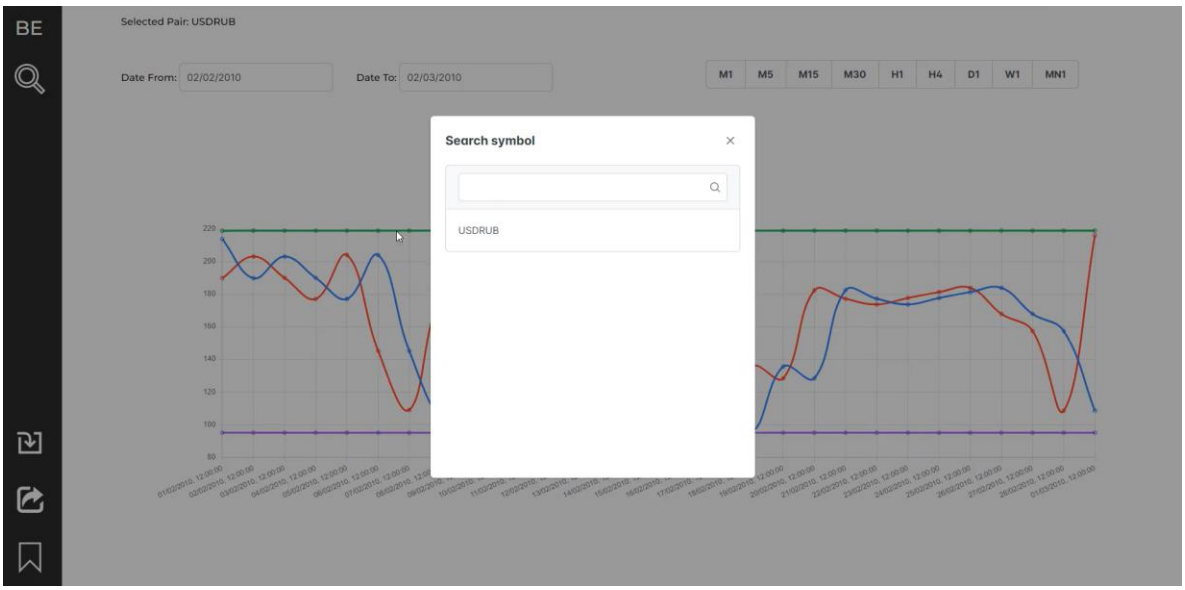

Рисунок 2 – Отображение сигнала определенной валютной пары. Выбор сигнала в модальном окне

2. Отображение информации об определенной точке сигнала

Предусловие: пользователь находится на главной странице, на которой отображается график какой-то валюты (кликом ЛКМ по нужной иконке).

Пользователь: выбирает частоту дискретизации (минута, 5 минут, 15 минут, 30 минут, 1 час, 4 часа, день, неделя, месяц) на верхней панели инструментов.

Система: отображает график соответствующим образом.

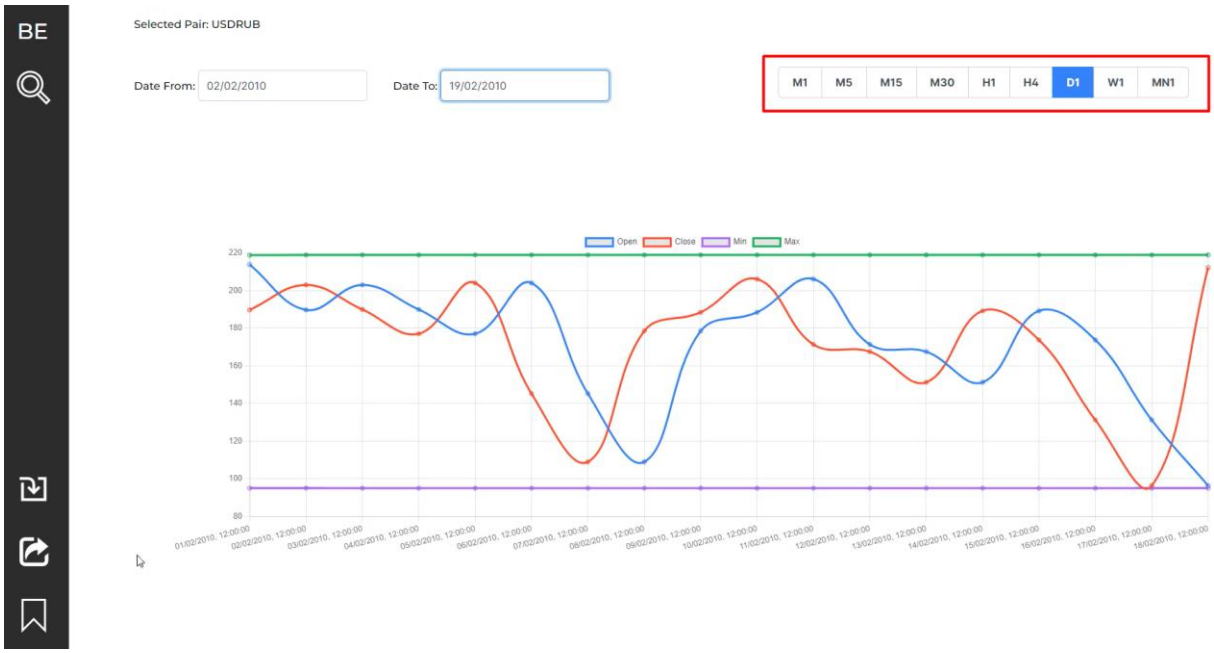

Рисунок 3 – Отображение сигнала с определённой частотой дискретизации

3. Просмотр архивных данные по валютным парам

Предусловие: пользователь находится на главной странице.

Пользователь: нажимает на кнопку архива в левой нижней части окна.

Система: перенаправляет пользователя на страницу архива, где отображает строку поиска и таблицу с информацией о валютной паре.

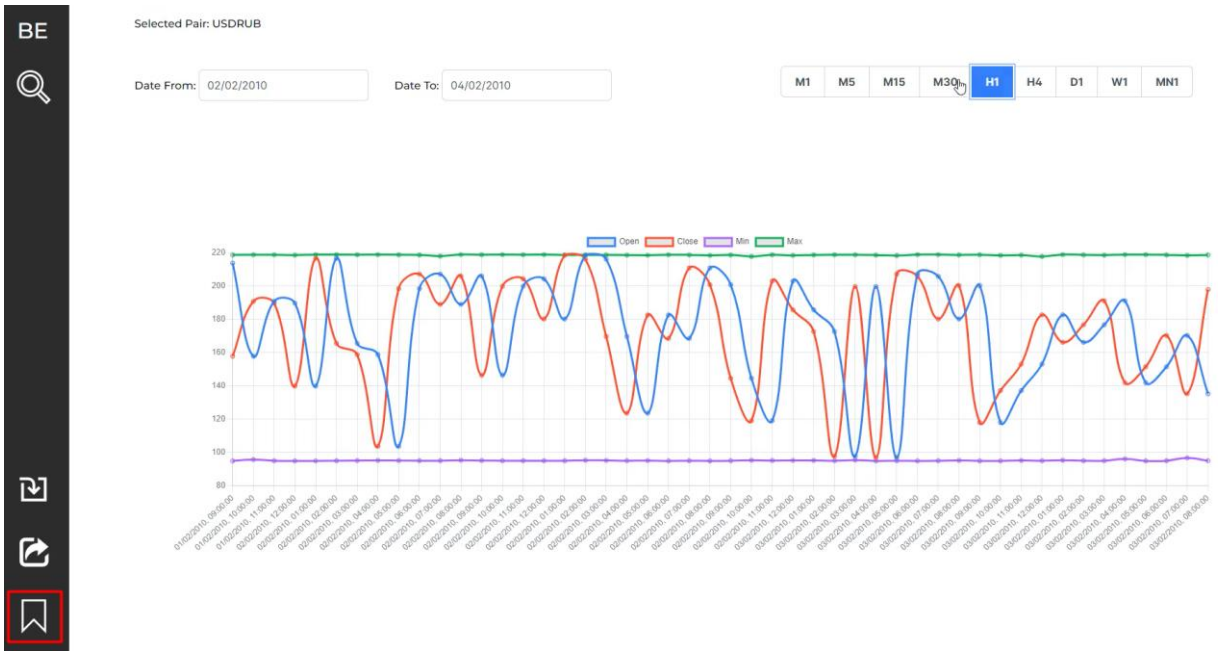

Рисунок 4 – Просмотр архивных данные по валютным парам. Открытие страницы архив

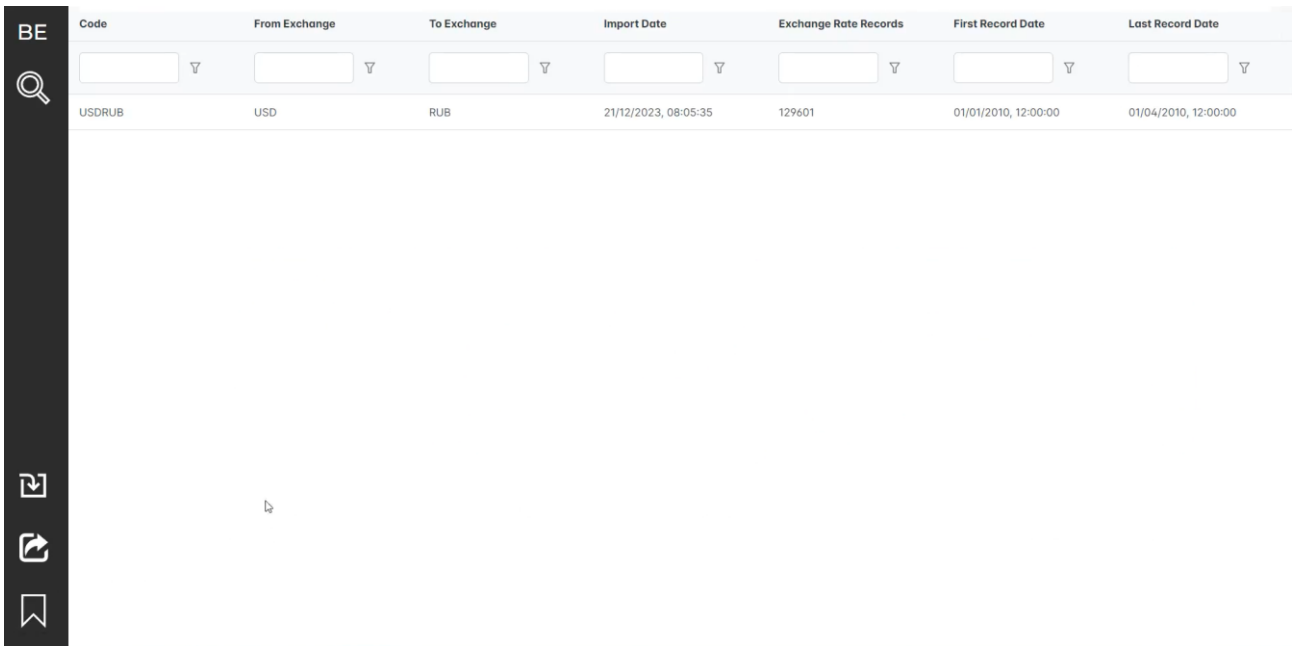

Рисунок 5 – Просмотр архивных данные по валютным парам. Страница архив

4. Отображение информации об определенной точке сигнала

Предусловие: пользователь находится на главной странице, на которой отображается график какой-то валютой.

Пользователь: наводит на желаемую точку сигнала.

Система: отображает информацию о желаемом моменте (timestamp, open, close, min, max) с помощью всплывающего окна.

Рисунок 6 – Отображение информации об определенной точке сигнала

5. Выбрать период для отображения

Предусловие: пользователь находится на главной странице, на которой отображается график какой-то валютой.

Пользователь: вводит данные диапазона в соответствующем поле над графиком.

Система: отображает график на соответствующем периоде.

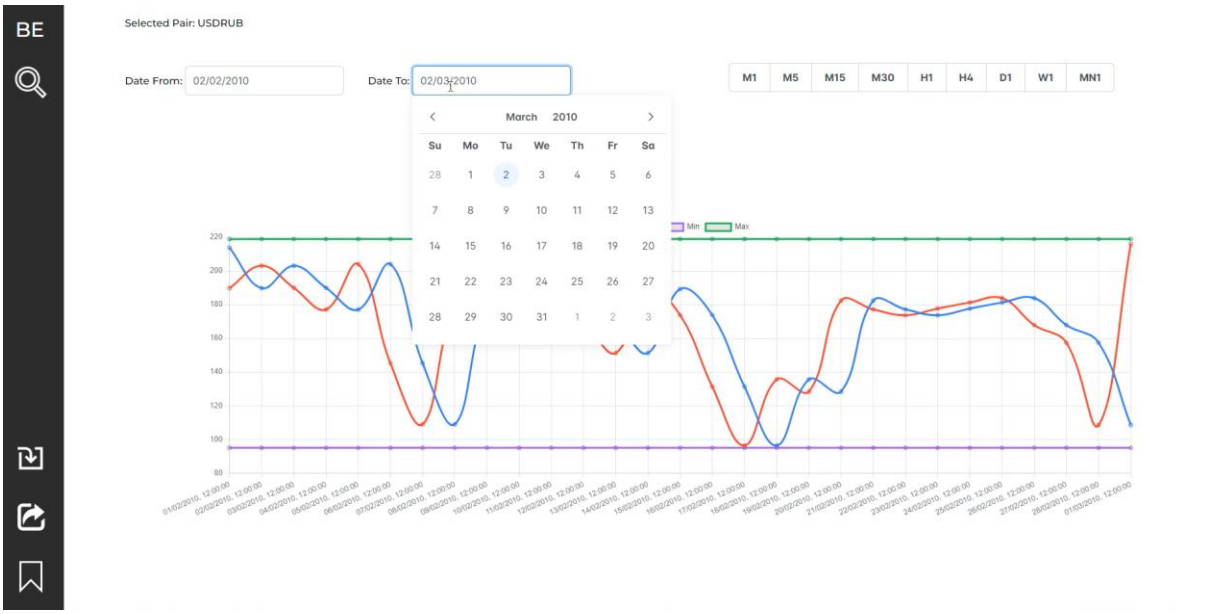

Рисунок 7 – Выбор периода для отображения

4. Модель данных

Нереляционная модель

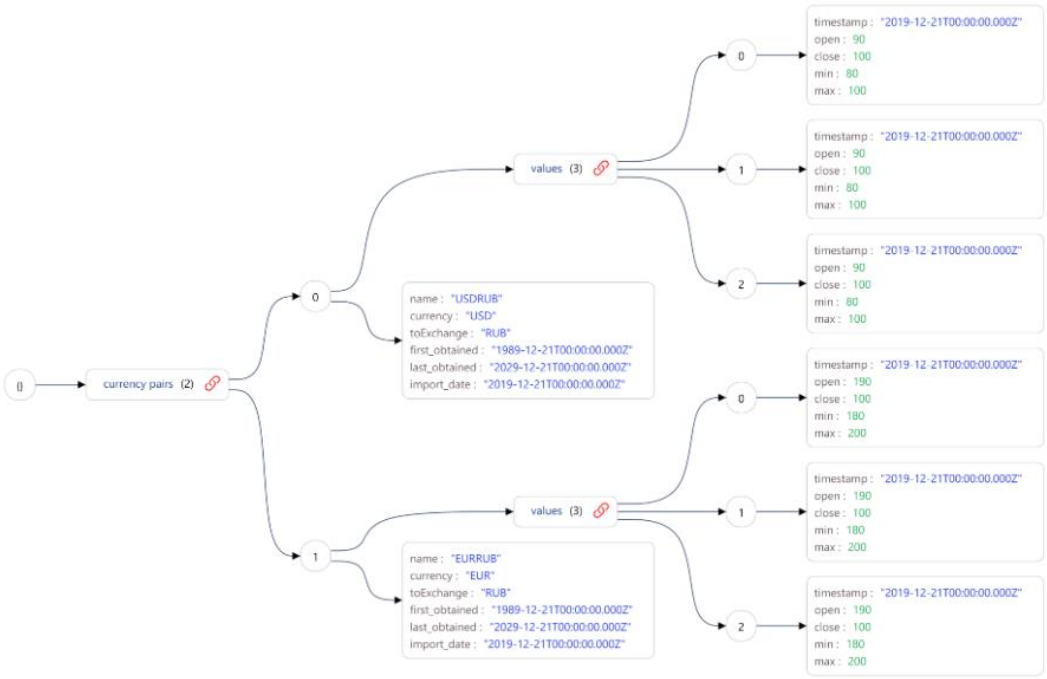

Рисунок 8 – Графическое представление нереляционной модели

Описание и оценка объёма

Решаемая задача: анализ данных биржевой торговли Forex.

Сущность: валютная пара – коллекция, представляющая собой массив, содержащий объекты, каждый из которых содержит информацию об одной конкретной валютной паре.

Валютные пары

Таблица 1. Основная информация о валютных парах

Поле	Описание	Тип	Объём, байт
name	наименование сигнала (код из 6 символов, например USDRUB	string	6
currency	валюта, стоимость которой смотрим (код из 3 символов)	string	3
toExchange	валюта, в которую преобразовываем (код из 3 символов, например USDRUB)	string	3
first_obtained	дата самых ранних данных о котировке валютной пары	Date	8
last_obtained	дата самых последних данных о котировке валютной пары	Date	8

import_date	дата внесения данных о котировках в БД	Date	8

MongoDB не имеет четкого ограничения размера строк. Однако, мы знаем, точное количество символов, которые хранятся в полях name, currency и toExchange. Это 6, 3 и 3 символа, соответственно. Исходя из того, что на хранение одного символа требуется 1 байт были рассчитаны объемы, представленные выше. Следовательно, для конкретной валютной пары основная информация займет V_main(1) = 36 байт.

Данные котировок валютной пары:

В коллекции для каждой валютной пары также есть массив “values”, содержащий документы с информацией о котировках конкретной валютной пары. Т.к. временной промежуток, за который мы храним данные произвольный, то количество элементов массива не ограничено.

В таблице ниже приведен объем одного элемента данного массива.

Таблица 2. Пример элемента массива “values”

Поле	Описание	Тип	Объём, байт
timestamp	отметка времени для конкретного сигнала	Date	8
open	цена на момент открытия (начало периода)	Int32	4
close	цена на момент закрытия (конец периода)	Int32	4
min	минимальная цена за период	Int32	4

max	максимальная цена за период	Int32	4

Размер одной котировки одной валютной пары равен V_ER(1) = 24 байта.

Соответственно для одной валютной пары с одним сигналом котировки объем будет равен: V_C(1) = 60 байт.

Избыточность модели

В штатном режиме работы с БД данные о котировках валютной пары и данные о самой валютной паре дублироваться не должны. Однако, одна и та же валютная пара может быть записана как два разных соотношения.

Например, USD к RUB:

- USD / RUB
- RUB / USD

В таком случае, формально одни и те же данные будут записаны дважды.

Отношение между фактическим и чистым объемом данных = 2/1.

Направление роста модели

В общем случае для N валютных пар с таким же количеством (N) сигналов котировок объем вычисляется по формуле:

V(N) = V_main(N) + V_ER(N) = N * V_main(1) + N * V_ER(1) = 36 * N + 24 * N = 60 * N.

Соответственно, рост модели линеен.

Запросы

Ниже приведены запросы по usecase'ам, связанным с взаимодействием с БД.

1. Отображение данных о котировке определенной валютной пары
   db.currency_pairs.find({ “name”: "USDRUB" })
2. Выбор данных определённой валюты на определённом диапазоне и с определённой частотой дискретизации
   db.currency_pairs.aggregate({$match: {"name": "EURRUB"}}, {$unwind: {"$values"}}, {$match: {$expr: {$and: {$gt: [$values.timestamp,

ISODate(2019, 12, 20, 0, 0, 0)]}, {$lt: [$values.timestamp, ISODate(2020, 11, 11, 0, 0, 0)]} }}}, {$group: {"_id": {$toDate: {$substract: [{"toLong": {$values.timestamp}}, {$mod : [{"toLong": {$values.timestamp}, 1000 * 60 * 15] } }] }} "open": {$first: $values.open}, "close": {$last: $values.close}, "min": {$min: $values.min}, "max": {$max: $values.max} }, {$sort: {"_id": 1}} })

3. Просмотр подробной статистики по сигналу из архива
   db.currency_pairs.find({ “name”: "USDRUB" })
4. Отображение информации об определенной точке сигнала
   db.currency_pairs.find({ “name”: "USDRUB", “values.timestamp”: ISODate(“2020-04-12”) })

Реляционная модель

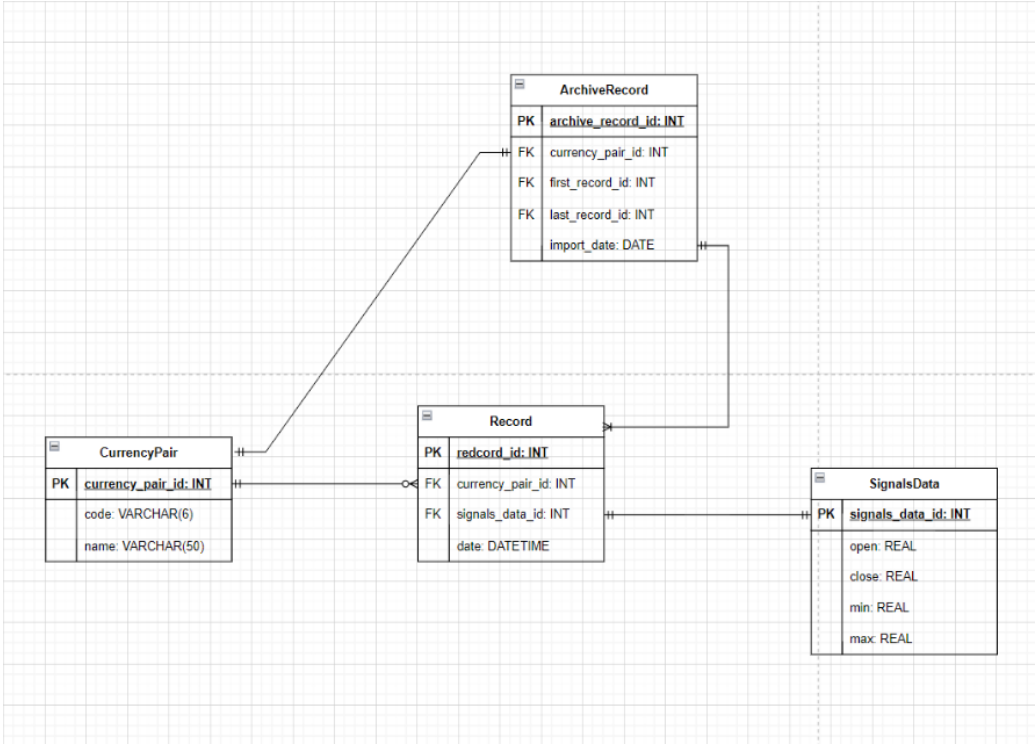

Рисунок 9 – Графическое представление реляционной модели

Описание и оценка объема

Исходя из предметной области было выделено 4 сущности:

- валютная пара - таблица CurrencyPair
- архивная запись - таблица ArchiveRecord
- запись - таблица Record
- данные по сигналам - таблица SignalsData

Таблица 3. Сущность CurrencyPair

Поле	Описание	Тип	Объём, байт
currency_pair_id	идентификатор валютной пары	INT	4
code	Краткое обозначение валютной пары (код из 6 символов)	VARCHAR(6)	6
name	Полное название валютной пары	VARCHAR(50)	50

Для одной записи в таблице объем равен: V_C(1) = 60 байт.

Таблица 4. Сущность ArchiveRecord

Поле	Описание	Тип	Объём, байт
archive_record_id	идентификатор архивной записи	INT	4
currency_pair_id	идентификатор валютной пары	INT	4
first_record_id	идентификатор первой записи о котировках	INT	4
last_record_id	идентификатор последней записи о котировках	INT	4

Для одной записи в таблице объем равен: V_A(1) = 24 байт.

Таблица 5. Сущность Record

Поле	Описание	Тип	Объём, байт
record_id	идентификатор записи о котировках	INT	4
currency_pair_id	идентификатор валютной пары	INT	4
signals_data_id	идентификатор сигнала (отметка времени и значения open, close, min, max)	INT	4
date	дата создания записи	DATETIME	8

Для одной записи в таблице объем равен: V_R(1) = 20 байт.

Таблица 6. Сущность SignalsData

Поле	Описание	Тип	Объём, байт
signals_data_id	идентификатор сигнала (отметка времени и значения open, close, min, max)	INT	4
open	цена на момент открытия (начало периода)	REAL	4
close	цена на момент закрытия (конец периода)	REAL	4

min	минимальная цена за период	REAL	4
max	максимальная цена за период	REAL	4

Для одной записи в таблице объем равен: V_S(1) = 20 байт.

Таким образом, для создания одной записи о курсе конкретной валютной пары требуется V(1) = 124 байт.

Избыточность модели

Дублирование данных в модели обусловлено в основном разбиением на независимые сущности:

- В таблицах CurrencyPair и ArchiveRecord повторяется поле currency_pair_id
- В таблицах Record и SignalsData повторяется поле signals_data_id
- record_id из таблицы Record может повторятся от 1 до 2 раз в полях first_record и last_record таблицы ArchiveRecord. (2 раза в случае, если у нас всего одна запись в таблице Record для конкретной валютной пары)

Направление роста модели

Для создания N записей о курсе конкретной валютной пары:

$V_{er}(N) = 84 + V_R(N) + V_S(N)$.

Для создания N записей о курсах N валютных пар, требуется:

$V(N) = V_C(N) + V_A(N) + V_R(N) + V_S(N) = N * (V_C(1) + V_A(1) + V_R(1) + V_S(1)) = N * (60 + 24 + 20 + 20) = N * 124$.

В обоих случаях рост линеен.

Запросы

1. Отобразить сигнал (open) по определённой валютной паре:
   SELECT date, open FROM CurrencyPair INNER JOIN Record USING(currency_pair_id) INNER JOIN SignalsData USING (signals_data_id) WHERE code = ‘EURUSD’;
2. Отобразить сигнал с определённой частотой дискретизации:
   SELECT DISTINCT DATE_FORMAT(date, '%Y-%m-%d %H:00:00') AS hour_period, open FROM CurrencyPair INNER JOIN Record USING(currency_pair_id) INNER JOIN SignalsData USING(signals_data_id) WHERE code = ‘EURUSD’ ORDER BY hour_period;
3. Посмотреть архивные данные по определённой валютной паре:
   SELECT cp.code AS currency_pair_code, r1.date AS first_record_date, r2.date AS last_record_date FROM CurrencyPair cp

JOIN ArchiveRecord ar ON cp.currency_pair_id = ar.currency_pair_id JOIN Record r1 ON ar.first_record_id = r1.record_id JOIN Record r2 ON ar.last_record_id = r2.record_id WHERE cp.code = 'EURUSD';

4. Отобразить информацию об определённой точке сигнала:
   SELECT r.date, sd.open, sd.close, sd.min, sd.max FROM CurrencyPair cp JOIN Record r USING(currency_pair_id) JOIN SignalsData sd USING(record_id) WHERE r.date = '2018-04-20 00:00:00' AND cp.code=’EURUSD’;

Сравнение моделей

- NoSQL модель требует в 2 раза меньше памяти, чем SQL-модель.
- При работе с данными в NoSQL также не требуется объединения таблиц для запросов к архивным или основным данным валютных пар.

Вывод

Модель на MongoDB подходит для данной задачи лучше чем БД MySQL, мы значительно уменьшаем используемую память и получаем более удобный доступ к архивным данным валютных пар.

Примеры хранения данных

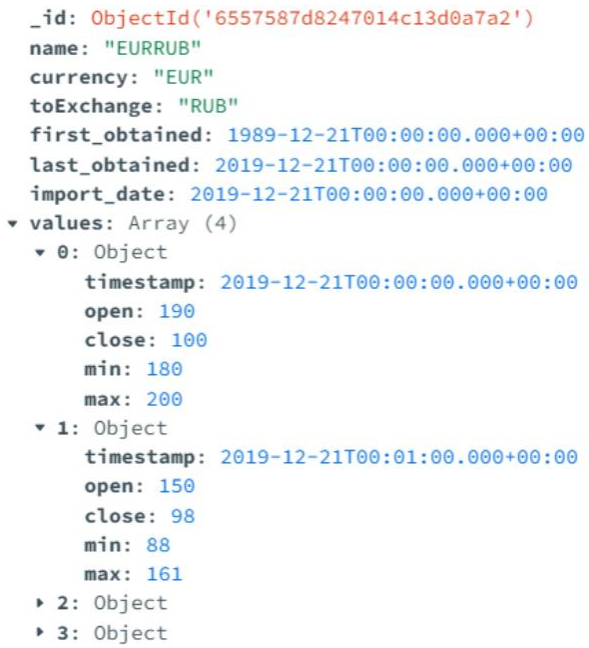

Рисунок 10 – Валютная пара, отображаемая в GUI MongoDBCompass

Рисунок 11 – Валютная пара, экспортированная в json-файл

5. Разработанное приложение

Краткое описание

Backend представляет из себя приложение, разработанное на Python 3.10 с использованием фреймворка Django 4, представляет собой инструментарий для обработки серверной логики. Система взаимодействует с базой данных MongoDB через библиотеку pymongo, обеспечивая надежное хранение и манипуляцию данными.

Функциональность Backend приложения включает в себя обработку запросов, работу с БД и другие серверные задачи. Благодаря Django, код организован по принципу модульности, что обеспечивает легкость поддержки и масштабирования приложения.

Frontend – это web-приложение, построенное с использованием Angular 16, что гарантирует высокую производительность и удобство в разработке интерфейса. Angular предоставляет эффективные средства для создания динамичных и отзывчивых пользовательских интерфейсов, а также обеспечивает прозрачное взаимодействие с Backend-частью посредством REST API.

Схема экранов приложения

Экраны приложения и переходы между ними отображены на рис. 12.

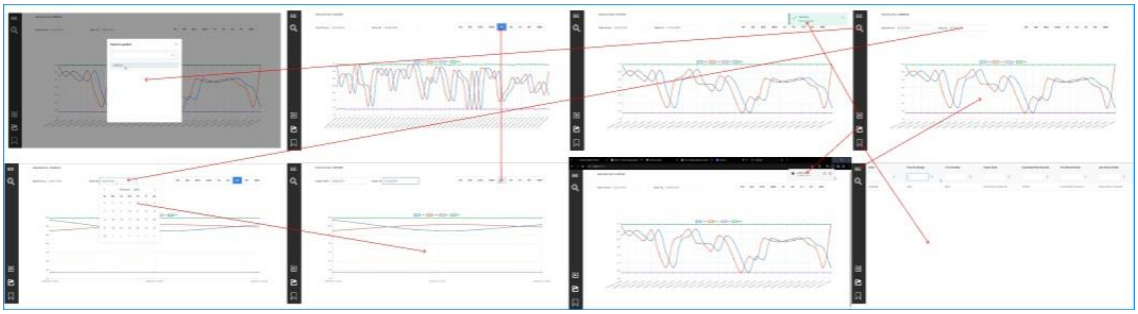

Рисунок 12 – Схема экранов приложения

Использованные технологии

БД: MongoDB.

Backend: Python 3.10, Django 4, pymongo.

Frontend: HTML, CSS, JavaScript, TypeScript, Angualar.

Ссылка на Приложение

Ссылка на GitHub: https://github.com/moevm/nosql2h23-forex

6. Вывод

Результаты

В ходе работы разработано приложение, которое хранит, обеспечивает поиск, анализ и визуализацию данных о рынке по указанным периодам, запросам, валютным парам. Приложение предоставляет возможности просмотра списков валютных пар, страницу архива, визуализации отдельного сигнала на графике для конкретной валютной пары и анализа для сигнала.

Недостатки и пути для улучшения полученного решения

На данный момент реализован недостаточный инструментарий для технического анализа сигнала котировки валютной пары. Не хватает рисования трендовой и горизонтальных линий, индикаторов скользящих средних (простая скользящая средняя, экспоненциальная средняя) и осцилляторы (индекс относительной силы, схождение и расхождение среднего)

Решением для данной проблемы является модернизация как Backend, так и Frontend приложений. Для Backend приложения — это будет добавление новых REST API, а для Frontend приложения – отрисовка данных, полученных по добавленным REST API.

Будущее развитие решения

Планируется разработка нативных приложений для OS Windows, Linux и MacOS.

7. Приложения

Документация по сборке и развёртыванию приложений

Для установки зависимостей необходимо установить на компьютер менеджеры пакетов pep и npm.

Backend

1. Скачать проект из репозитория (указан в ссылке на приложение)
2. Установить зависимости с помощью команды cd ./backend && pip install -r requirements.txt
3. Запустить Backend приложение с помощью команды python manage.py runserver
4. Открыть приложение в браузере по адресу http://localhost:8000 при желании проверки работоспособности

Frontend

1. Установить зависимости с помощью команды cd ../prototype/frontend && npm i
2. Запустить Frontend приложение с помощью команды ng serve
3. Открыть приложение в браузере по адресу http://localhost:4200

8. Используемая литература

1. Документация к MongoDB: https://docs.mongodb.com/manual