

Проект по курсу

Программист Си

otus.ru

Меня хорошо видно
&& слышно?



Защита проекта

Тема: *Оптимизация производительности БД PostgreSQL
путём автоматического распараллеливания задач.*



Сартаков Алексей

pismo@ohido.ru

https://t.me/Sartakov_Aleksey

Учет нагрузки сервера в БД PostgreSQL = Проект №2

План защиты



1. Цели проекта

Какие цели вы поставили и какие задачи решили своим проектом

- 1.** Увеличить скорость предстартовой обработки данных (импорт данных из БД и расчеты всех индикаторов) с 14 минут до минимальных значений.
- 2.** Экстренный старт программы сократить с 20 минут на каждый час работы QUIK с до минимума (чем меньше, тем лучше).
- 3.** Научиться работе с многопоточностью, структурами в Си, выделяемой RAM памятью, взаимодействию с БД.
- 4.** Обработка котировок ММВБ в режиме реального времени, где один цикл на принятие решения занимает максимум 5-7 секунд на полный цикл, используя сокет и многопоточность.
- 5.** Снижение нагрузки на ПК, если превышены пороговые значения по температуре или нагрузке CPU.

2. Что планировалось

Что было в начале, что знали до курса, сколько времени заняло выполнение проекта

1. Получать из БД исторические котировки (предыдущих торговых дней).
2. Получать из БД текущие котировки в режиме реального времени «за сегодня».
3. Проверять обновления котировок относительно тех, что уже есть в программе на Си.
4. Делать обновления или полные пересчёты торговых индикаторов.
5. По расчетам, один цикл полного расчета «принятия решения» в программе Си должен был занимать около 40-50 секунд, поэтому планировалось использовать сокеты, мьютексы, многопоточность.
6. Далее >>>

6.	Сохранение значений индикаторов в БД для принятия решения. Это уже для следующего проекта – работа с ИИ по его обучению.
7.	Сохранения значений индикаторов как массивы из 40 срезов в БД для последующих выводов графиков на локальном сайте.
8.	Та часть проекта, что в открытом доступе, выполнялась с начала сентября по конец ноября 2023. Имеет среднюю степень сложности (всего два-три слоя данных) и около 7 млн расчётных точек.
9.	Это второй серьёзный проект на Си. Первый = нагрузка ПК.
10.	До проекта очень хорошо знал БД PostgreSQL (SQL, настройки, оптимизация, роли, функции pl/pgsql, окна, выборки). Так же знаю Python, HTML и SVG разметку, сервер-сайта Apache2.

Используемые технологии

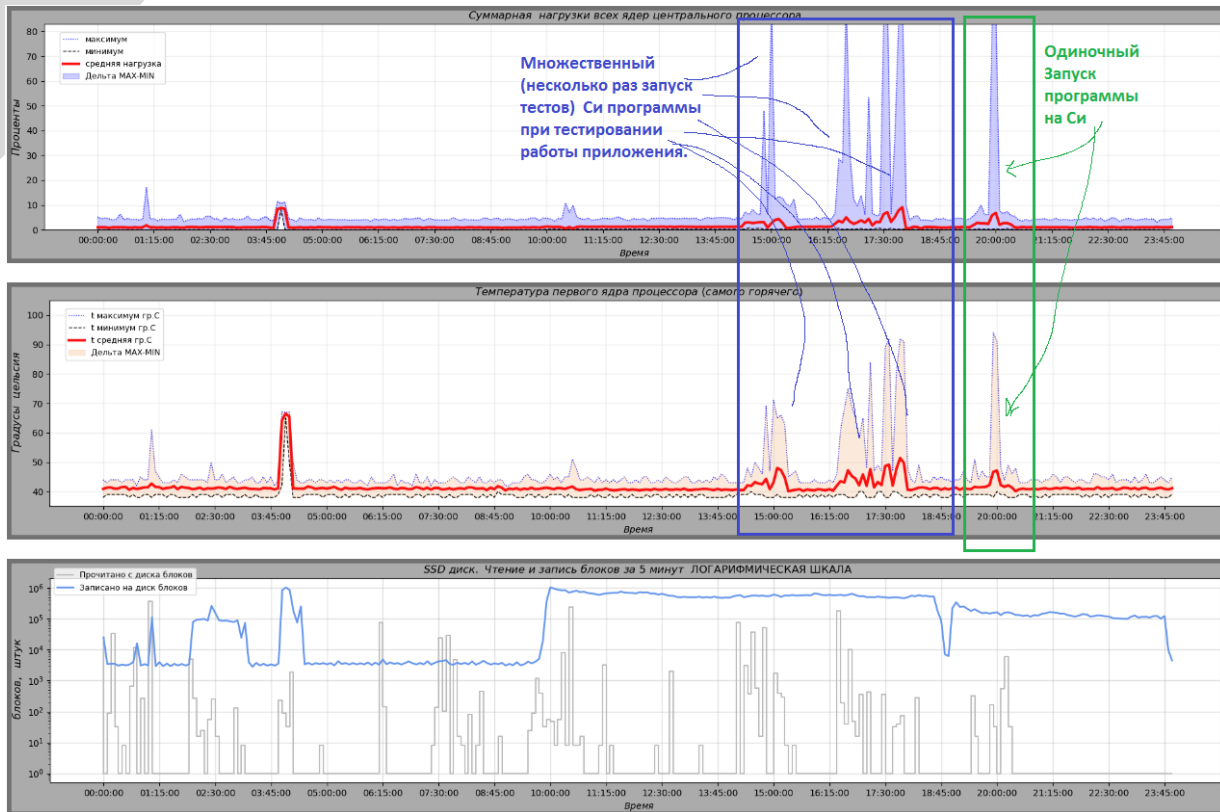
1. Язык Си. БД PostgreSQL.
2. pl/pgSQL язык для написания встроенных функций в БД PostgreSQL.
3. Python. Для обработки скриптов на стороне backend site и для генерации графиков на сайте.
4. SQL – работа с запросами из Си к БД.
5. HTML – формы на сайте для отображения графиков.
SVG язык/разметка для вывода графиков.
6. Bash-скрипты для работы с демонами systemd/автозагрузок в Debian 11.

Какие технологии использовались и
какое у вас мнение о новых технологиях

Что получилось

Контроль нагрузки ПК и
уменьшения его нагрузок.

Более подробно
представлено в отчёте
[>>> перейти.](#)



Что получилось

Ускорение скорости вычислений **от 278 раз:**
с 13m:15s до 2.85 секунд.

```
postgres@honor-srv: ~/Документы/дом_зад_с
Файл Правка Вкладки Справка
m15[222]GAZP 22:00(ср): откр 163.22 макс
m15[221]GAZP 21:45(ср): откр 163.16 макс
m15[220]GAZP 21:30(ср): откр 163.21 макс
m15[219]GAZP 21:15(ср): откр 163.25 макс
m15[218]GAZP 21:00(ср): откр 163.35 макс
m15[217]GAZP 20:45(ср): откр 163.24 макс
m15[216]GAZP 20:30(ср): откр 163.06 макс
m15[215]GAZP 20:15(ср): откр 163.10 макс
m15[214]GAZP 20:00(ср): откр 163.14 макс
m15[213]GAZP 19:45(ср): откр 163.37 макс
m15[212]GAZP 19:30(ср): откр 163.27 макс
m15[211]GAZP 19:15(ср): откр 163.34 макс
m15[210]GAZP 19:00(ср): откр 162.82 макс
Расчёт предварительных настроек всего: 2.85 сек.
С момента запуска программы прошло: 2.85 сек.
=====
Расчёт ВСЕХ индикаторов занял: 0.00 сек.
С момента запуска программы прошло: 2.85 сек.
Расчёт всех индикаторов завершили в: 2024.01.17 09:24:52, среда
Программа запущена с количеством потоков = 12 шт.
2024.01.17 09:24:52, среда Нагрузка ПК: 20.43% Температура CPU: 50.8 C (текущая: 65.8)
Пауза в работе программы на 2078 секунд до 09-59-30
```

```
Результат 1 Вывод X
Enter a part of a message to search for here
-->>>> secid=УККА. Этот эмитент не совпадает по последнему торговому дню (<NULL> < 2023-11-29).
-->>>> "URKZ" не прошёл порог в 5 млн рублей средненежного оборота = 1503.4 тыс руб.
-->>>> "USBN" не прошёл порог в 5 млн рублей средненежного оборота = 3843.8 тыс руб.
-->>>> "UTAR" не прошёл порог в 5 млн рублей средненежного оборота = 3606.4 тыс руб.
-->>>> "VEON-RX" не прошёл порог в 5 млн рублей средненежного оборота = 3057.4 тыс руб.
-->>>> "VGSB" не прошёл порог в 5 млн рублей средненежного оборота = 3710.2 тыс руб.
-->>>> "VGSBP" не прошёл порог в 5 млн рублей средненежного оборота = 2393.7 тыс руб.
-->>>> "VJGZ" не прошёл порог в 5 млн рублей средненежного оборота = 985.2 тыс руб.
-->>>> "VJGZP" не прошёл порог в 5 млн рублей средненежного оборота = 1225.5 тыс руб.
-->>>> "VRSBP" не прошёл порог в 5 млн рублей средненежного оборота = 4820.6 тыс руб.
-->>>> "VSYD" не прошёл порог в 5 млн рублей средненежного оборота = 1924.4 тыс руб.
-->>>> "VSYDP" не прошёл порог в 5 млн рублей средненежного оборота = 1573.2 тыс руб.
-->>>> "WTCM" не прошёл порог в 5 млн рублей средненежного оборота = 1175.2 тыс руб.
-->>>> "WTCMP" не прошёл порог в 5 млн рублей средненежного оборота = 986.8 тыс руб.
-->>>> "YKEN" не прошёл порог в 5 млн рублей средненежного оборота = 2943.2 тыс руб.
-->>>> "YKENP" не прошёл порог в 5 млн рублей средненежного оборота = 1513.8 тыс руб.
-->>>> "YRSB" не прошёл порог в 5 млн рублей средненежного оборота = 2560.1 тыс руб.
-->>>> "YRSBP" не прошёл порог в 5 млн рублей средненежного оборота = 2328.7 тыс руб.
-->>>> "ZILL" не прошёл порог в 5 млн рублей средненежного оборота = 3558.0 тыс руб.
Шаг №3. Всего получился список из 164 акций, который упрощает по дате и по обороту
копируем свечи price_4_цены (шаг 3) ----- для ABIO
Для HTML запросов. Полностью обработали secid= ABIO. Работа= 00:00:13.159296.
копируем свечи price_4_цены (шаг 3) ----- для ABRD
MSK ru_RU Запись
```

До **155** раз =
“полные расчеты”.

Подробнее:
[>>> перейти.](#)

```
postgres@honor-srv: ~/Документы/_ммвб_си_работа/исходники_ммвб_си/build-Debug/bin
Файл  Правка  Вкладки  Справка
дошли до ГАЗПРОМа6 таймфрэйм=m12
дошли до ГАЗПРОМа6 таймфрэйм=m10
дошли до ГАЗПРОМа6 таймфрэйм=m15
дошли до ГАЗПРОМа6 таймфрэйм=m20
дошли до ГАЗПРОМа6 таймфрэйм=m30
Ожидаем завершения локального thrd_join №=1 из глобального = 1 (m2)
Ожидаем завершения локального thrd_join №=2 из глобального = 2 (m3)
Ожидаем завершения локального thrd_join №=3 из глобального = 3 (m4)
Ожидаем завершения локального thrd_join №=4 из глобального = 4 (m5)
Ожидаем завершения локального thrd_join №=5 из глобального = 5 (m6)
Ожидаем завершения локального thrd_join №=6 из глобального = 6 (m10)
Ожидаем завершения локального thrd_join №=7 из глобального = 7 (m12)
Ожидаем завершения локального thrd_join №=8 из глобального = 8 (m15)
Ожидаем завершения локального thrd_join №=9 из глобального = 9 (m20)
Ожидаем завершения локального thrd_join №=10 из глобального = 10 (m30)
2023.12.12 16:42:22, вторник      Нагрузка ПК: 50.24%      Температура CPU: 75.1 C (текущая: 76.1)

Время завершения программы: 12/12/23 13:42:22.372897191 UTC      Затрачено времени: 5.413478

postgres@honor-srv:~/Документы/_ммвб_си_работа/исходники_ммвб_си/build-Debug/bin$
```

Для 12-ти потоков эта разница уже составляет более 155 раз (5.4 секунды и более 14 минут). $14 \cdot 60 / 5.4 = 155$

Более того, и что самое важное – код на Си можно еще ускорить. Так распараллеливание работы программы делалось только в одном месте, а расчеты всех индикаторов делались в однопоточном режиме. То есть можно смело еще на 0,5 секунды увеличить скорость работы (были тесты этой задачи в другом месте).

Возникшие проблемы

1.	Скорость тестовой БД по выборке данных увеличилась с 14 минут до почти 28 минут.
2.	Охлаждение ноутбука.
3.	Работа с массивом текстовых указателей. Они постоянно 'сыпались'.
4.	Даты. В БД время (дата+время) и в проекте на Си. Разные типы. При переводе из одной в другую время постоянно меняется. Если ничего не делать, время все равно меняется, хотя и было указано, что все делается как TZ+0.
5.	При работе с БД – на каждый запрос нужно отдельное соединение в многопоточном режиме. Это сильно снижает скорость многопоточного режима. Очень сильно. Создание нового каждого подсоединения к БД занимает 0,024 сек. Если же оно уже существует то 0,0004. Отличия в 60 раз.
6.	Самое сложное – создать структуру заголовочных файлов (*.h)
7.	Собрать проект как CMakeList.txt
8.	Разработать многослойную, универсальную структуру с произвольным ко-вом переменных, понятную и прозрачную и при этом не закольцованную саму на себя.
9.	Работа с русскими названиями в БД при передаче данных в БД PostgreSQL.

Возникшие проблемы при реализации проекта.

Пути их решения

3. Пришлось везде заметить массив указателей на текстовые строки на массив из строк. Все проблемы сразу исчезли.
4. Что бы не делал, как бы и где бы не приводил дату+время к TZ+0, все равно данные, что выходят из БД и потом возвращаются имеют разницу в 3 часа, хотя в Си никаких операций с датой не делалось. Все работы делались с целым силом (время в минутах с начала эпохи). В итоге сделал костыль «-3 часа» при возврате времени в БД. Этот показатель не критичен, т.к. это просто подпись к графикам.
9. Здесь целая эпопея. Но вкратце – просто под массив с русскими буквами выделял в 2 раза больше ячеек +1, чем русских букв. И все заработало без проблем.



Схема взаимодействия
ПК и блоков проекта.

СИСТЕМА №1

- ✓ OS -> Windows
- ✓ тор. терминал -> QUIK
- ✓ БД -> PostgreSQL 15

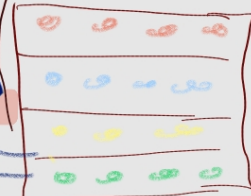
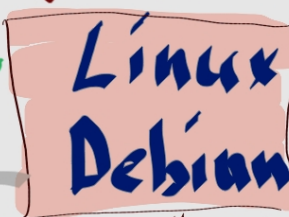
2



3

СИСТЕМА №2

- ✓ OS -> Debian (2023)
- ✓ БД -> PostgreSQL 15
- ✓ Код на Си



1

4

4



Выводы и планы по развитию



Разбор срезов торгового дня (обучения) ИИ.



Вывод графиков через Python на сайте.



Кнопка экстренного ручного выхода из всех позиций по всем счетам (если торговый робот «взбесится»).



Побольше отдыхать.

Запланируйте пару минут на рефлекссию в конце защиты проекта и расскажите о планах по развитию



Спасибо за внимание!