Проект по курсу

Программист Си

otus.ru

Меня хорошо видно
&& слышно?

Защита проекта

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

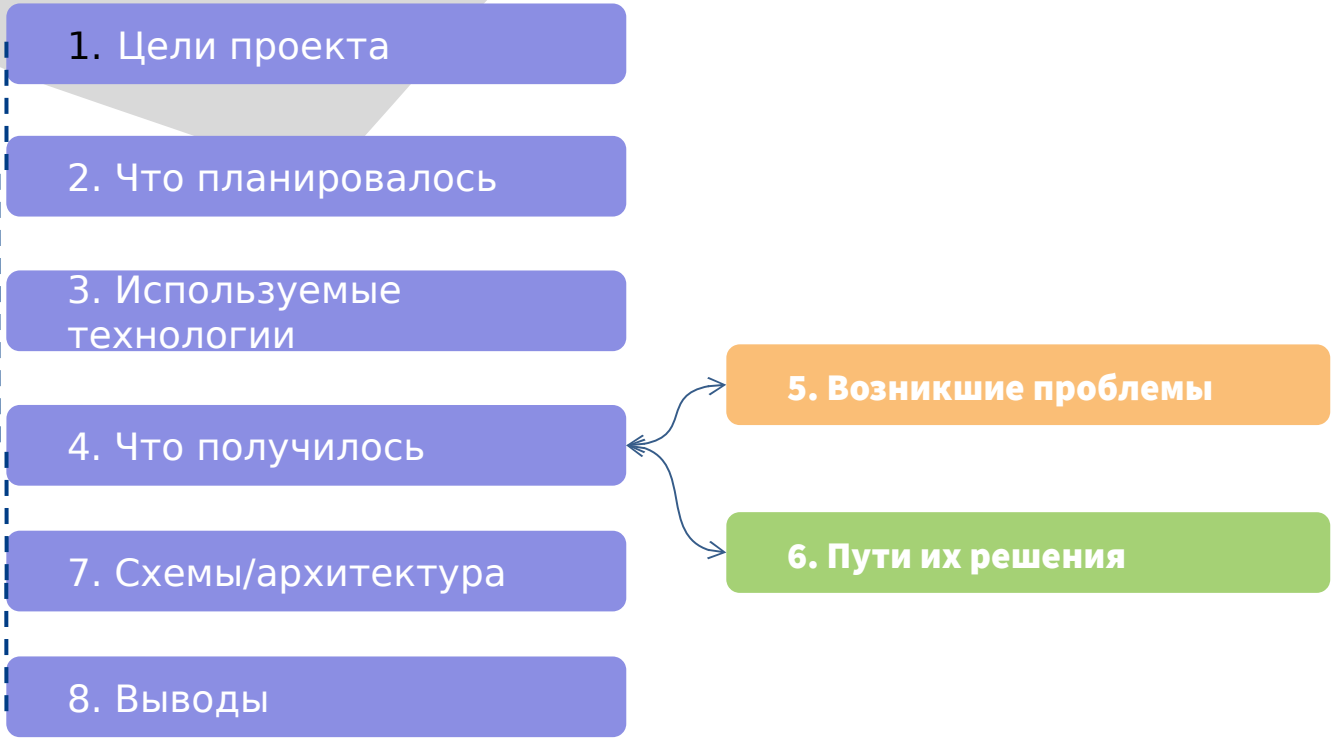
Сартаков Алексей

pismo@ohido.ru

https://t.me/Sartakov_Aleksey

Учет нагрузки сервера в БД PostgreSQL = Проект №2

План защиты


1. Цели проекта
2. Что планировалось
3. Используемые технологии
4. Что получилось
5. Возникшие проблемы
6. Пути их решения
7. Схемы/архитектура
8. Выводы

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 сокеты 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

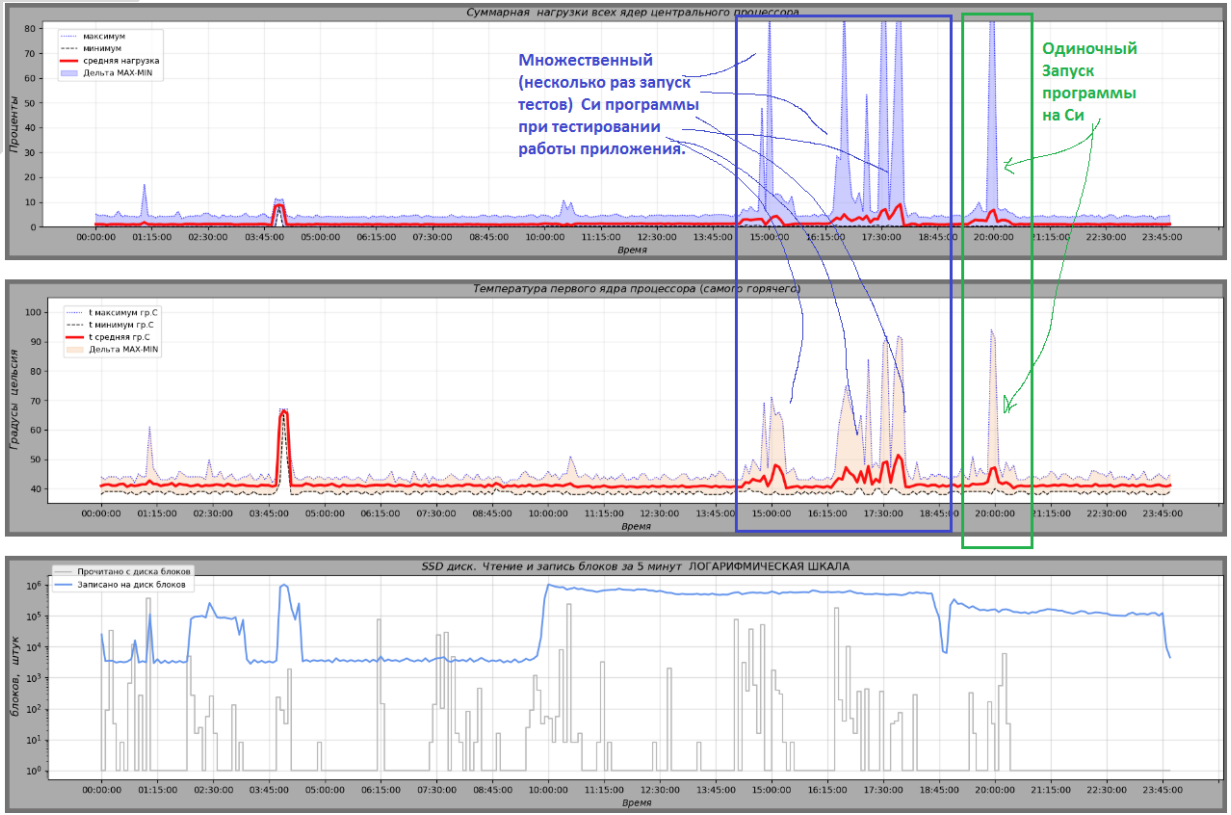
1.	Язык Си. БД PostgreSQL.
2.	pl/pgSQL язык для написания встроенных функций в БД PostgreSQL.
3.	Python. Для обработки скриптов на стороне backend site и для генерации графиков на сайте.
4.	SQL – работа с запросами из Си к БД.
5.	HTML – формы на сайте для отображения графиков. SVG язык/разметка для вывода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

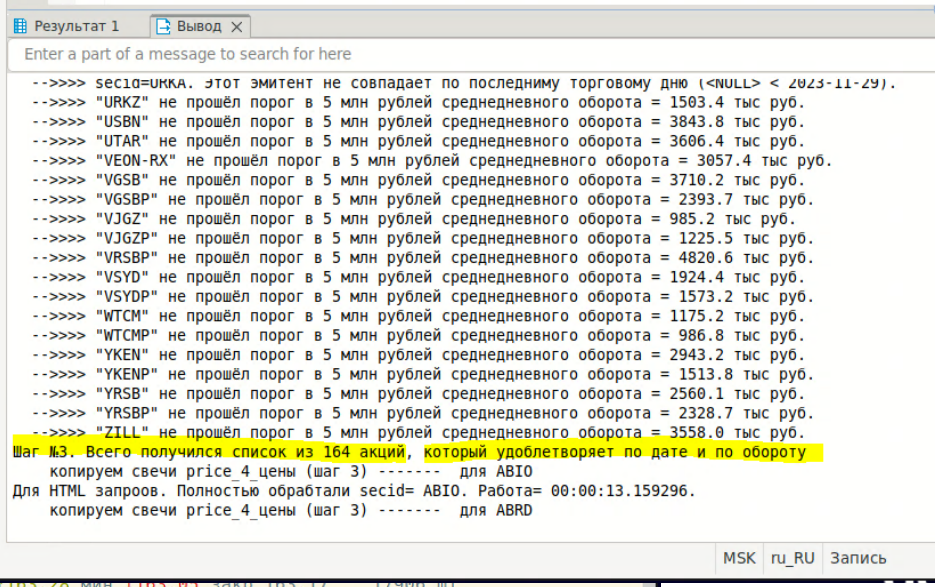
Контроль нагрузки ПК и уменьшения его нагрузок.

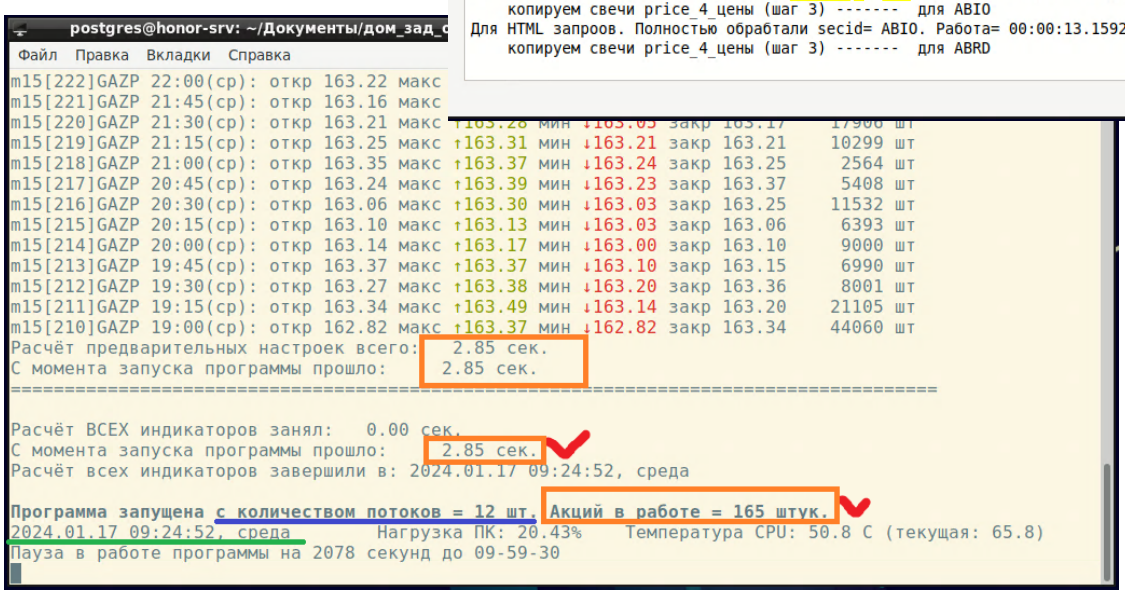
Более подробно представлено в отчёте
[>>> перейти.]



Что получилось

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

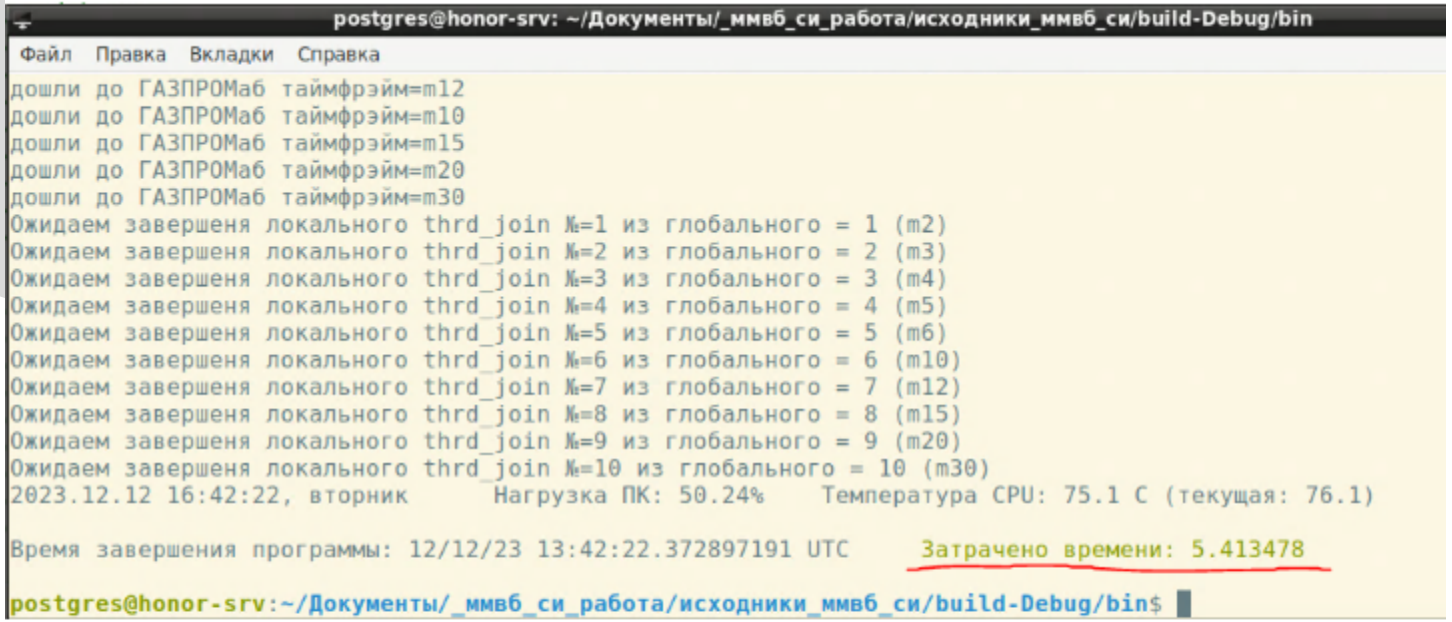




До **155** раз = “полные расчеты”.

Подробнее:
[>>> перейти.]

Выборка из БД + вычисления всех индикаторов всех акций. 6 потоков.



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

Более того, и что самое важное – код на Си можно еще ускорить. Так распараллелива-ние 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.

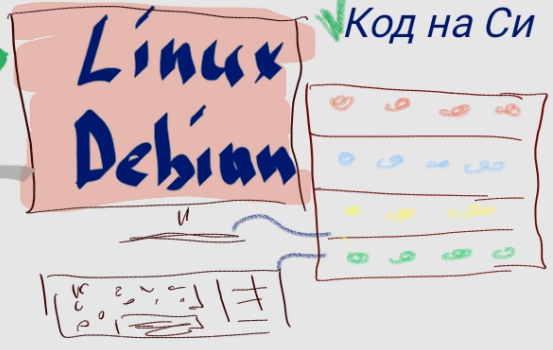
СИСТЕМА №1

- OS -> Windoews
- тор.терминал -> QUIK
- БД -> PostgreSQL 15

Си

СИСТЕМА №2

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



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 рефлексию в конце защиты проекта и расскажите о планах по развитию

Спасибо за внимание!