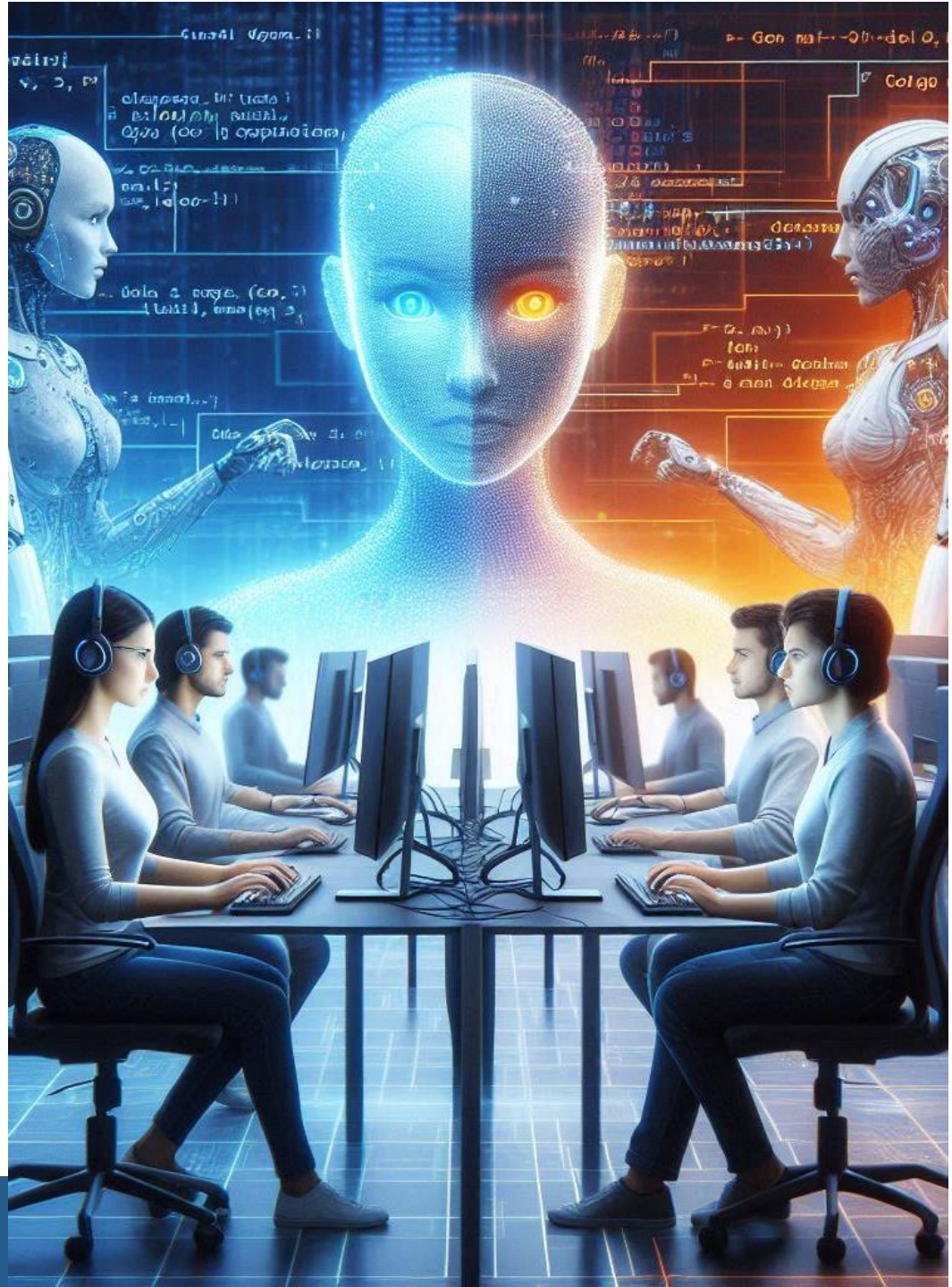
Применение ИИ для очистки метаданных и данных

Альтернативный метод



info@contourcomponents.com

contourcomponents.com

Очистка HTML документов

Цели очистки

- Предварительная обработка Big Data (миллионов web страниц) для загрузки в БД и других целей
- Превращение унаследованных архивов HTML документов в тексты для загрузки в БД и единообразной визуализации

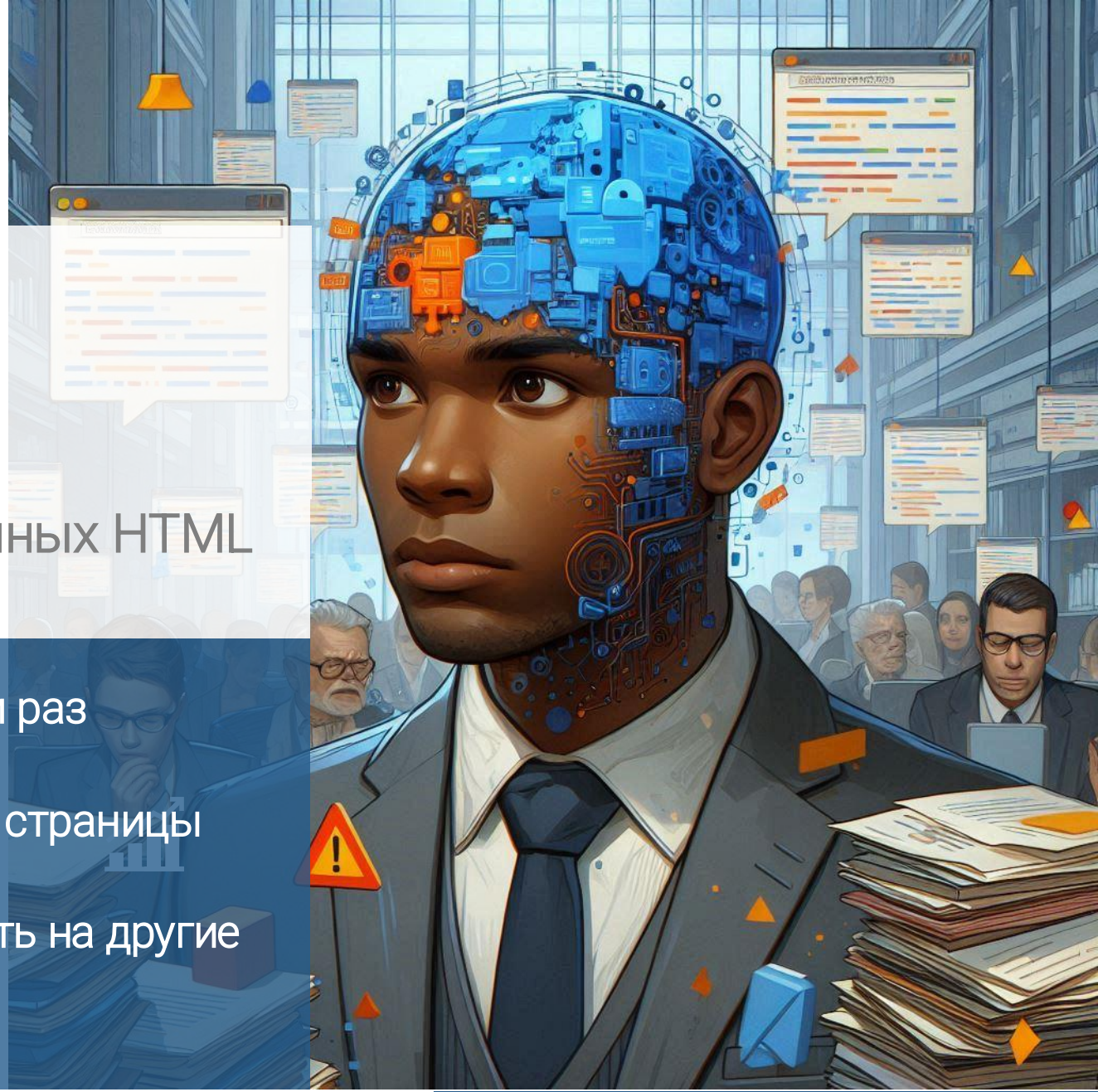
info@contourcomponents.com

contourcomponents.com

Низкое качество унаследованных HTML документов

Проблемы с архивами унаследованных HTML страниц

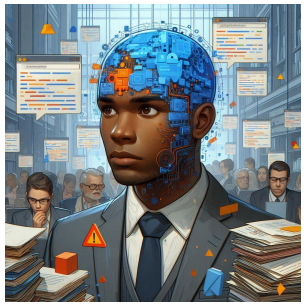
- Объем данных увеличен в десятки раз
- В приложении могут разрушаться страницы
- Сложно анализировать, переводить на другие языки



info@contourcomponents.com

Пример HTML документа низкого качества

```
<html><head><meta name="qrichtext" content="1" /><style
type="text/css">
p, li { white-space: pre-wrap; }
</style></head><body style=" font-family:''Roboto''; font-
size:8pt; font-weight:400; font-style:normal;">
<p align="center" style=" margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;"><span style=" font-
size:18pt;">Прототип</span></p></body></html>
```



Документ содержит множество ненужных тэгов и только одно значимое слово:

“Прототип”

Прямое решение

Задание:
“Пожалуйста, конвертируй этот HTML в чистый текст”

```
<html><head><meta name="qrichtext"
content="1" /><style type="text/css">
p, li { white-space: pre-wrap; }
</style></head><body style=" font-
family:''Roboto''; font-size:8pt; font-
weight:400; font-style:normal;">
<p align="center" style=" margin-top:0px;
margin-bottom:0px; margin-left:0px; margin-
right:0px; -qt-block-indent:0; text-
indent:0px;"><span style=" font-
size:18pt;">Prototype</span></p></body></ht
ml>
```

Чистый текст:
“Прототип”

info@contourcomponents.com

Альтернативное решение

Задание:
“Пожалуйста создай функцию, конвертирующую подобные HTML в текст”

```
<html><head><meta name="qrichtext"
content="1" /><style type="text/css">
p, li { white-space: pre-wrap; }
</style></head><body style=" font-
family:''Roboto''; font-size:8pt; font-
weight:400; font-style:normal;">
<p align="center" style=" margin-top:0px;
margin-bottom:0px; margin-left:0px; margin-
right:0px; -qt-block-indent:0; text-
indent:0px;"><span style=" font-
size:18pt;">Prototype</span></p></body></ht
ml>
```

Исходный текст функции

info@contourcomponents.com

Выполнение функции, разработанной ИИ

Параметр: HTML документ

```
<html><head><meta name="qrichtext"
content="1" /><style type="text/css">
p, li { white-space: pre-wrap; }
</style></head><body style=" font-
family:''Roboto''; font-size:8pt; font-
weight:400; font-style:normal;">
<p align="center" style=" margin-top:0px;
margin-bottom:0px; margin-left:0px; margin-
right:0px; -qt-block-indent:0; text-
indent:0px;"><span style=" font-
size:18pt;">Prototype</span></p></body></ht
ml>
```

Функция

Результат

Чистый текст:
“Прототип”

info@contourcomponents.com

Исходный текст функции от ИИ

```
DROP FUNCTION IF EXISTS tools.strip_html;
CREATE OR REPLACE FUNCTION tools.strip_html
(
    html_text TEXT,
    save_structure BOOLEAN DEFAULT true, -- save returns in Ansi text, ignored for HTML
    destination_format TEXT DEFAULT 'text' -- 'text' or 'html'
)
RETURNS TEXT
LANGUAGE plpgsql
AS $$
DECLARE
    clean_text TEXT;
    destination_formats TEXT[] := '{text, html}';
BEGIN

    destination_format := TRIM(LOWER(destinati

    IF NOT destination_format = ANY (destinati
        RAISE EXCEPTION 'Unknown destination f
    END IF;

    IF destination_format = 'html' THEN
        save_structure := true;
    END IF;

    -- Remove conditional comments and their content
    clean_text := regexp_replace(html_text, '<!--\\[if[^>]*>.*?<!\\[endif\\]-->', '', 'gi');

    -- Remove XML-like tags and their content
    clean_text := regexp_replace(clean_text, '<xml[^>]*?>.*?</xml>', '', 'gi');

    -- Remove specific tags like <o:p> and their content
    clean_text := regexp_replace(clean_text, '<o:p[^>]*?>.*?</o:p>', '', 'gi');

    -- Remove <style> and <script> tags and their content
    clean_text := regexp_replace(clean_text, '<style[^>]*?>.*?</style>', '', 'gi');
    clean_text := regexp_replace(clean_text, '<script[^>]*?>.*?</script>', '', 'gi');
```

Нечеловеческий стиль программирования

```
    -- Remove <head> and <meta> tags and their content
    clean_text := regexp_replace(clean_text, '<head[^>]*?>.*?</head>', '', 'gi');
    clean_text := regexp_replace(clean_text, '<meta[^>]*?>', '', 'gi');

    -- If any { ... } block appears, it's most likely CSS, remove cautiously
    clean_text := regexp_replace(clean_text, '{[^}]*}',

    -- Replace HTML entities (like  )
    clean_text := regexp_replace(clean_text, ' ',

    -- Handle the rest based on the desired output forma
    IF destination_format = 'text' THEN
        IF save_structure THEN
            -- Add a newline after </o:p></font></p> and
            clean_text := regexp_replace(clean_text, '</

            -- Replace <p> and <br> tags with newline for structure preservation
            clean_text := regexp_replace(clean_text, '</?p[^>]*>', E'\n', 'gi');
            clean_text := regexp_replace(clean_text, '<br[^>]*>', E'\n', 'gi');
            -- Replace <div> and </div> tags with newline for structure preservation
            clean_text := regexp_replace(clean_text, '</?div[^>]*>', E'\n', 'gi');

            -- Remove all other HTML tags while preserving the inner text content
            clean_text := regexp_replace(clean_text, '<[^>]+>', '', 'g');

            -- Adjust the regex to collapse multiple newlines into one newline, not two
            clean_text := regexp_replace(clean_text, '(\n\s*){2,}', E'\n', 'g');

            -- Trim leading and trailing whitespace and newlines
            clean_text := trim(both E'\n' FROM clean_text);
        ELSE
            -- For ANSI format without structure, replace </p> and </b> with a space
            clean_text := regexp_replace(clean_text, '</(p|b)>', ' ', 'gi'); -- Replace </p> and </b> with a space
            clean_text := regexp_replace(clean_text, '<[^>]+>', '', 'g'); -- Remove all other HTML tags
            clean_text := regexp_replace(clean_text, '\s+', ' ', 'g'); -- Collapse multiple spaces
            clean_text := trim(both ' ' FROM clean_text); -- Trim leading and trailing spaces
        END IF;
    ELSE
```

Описание каждого шага

info@contourcomponents.com

contourcomponents.com

Выполнение функции, разработанной ИИ

Вызов функции

```
99
100  SELECT * FROM tools.strip_htm
101  (
102  '<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN" "http://www.w3.org/TR/REC-html40/strict.dtd">
103  <html><head><meta name="qrichtext" content="1" /><style type="text/css">
104  p, li { white-space: pre-wrap; }
105  </style></head><body style=" font-family:''Roboto''; font-size:8pt; font-weight:400; font-style:normal;">
106  <p align="center" style=" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"><
107  true,
108  'text'
109  );
110
111  SELECT * FROM
```

Data Output | Messages | Notifications

SQL

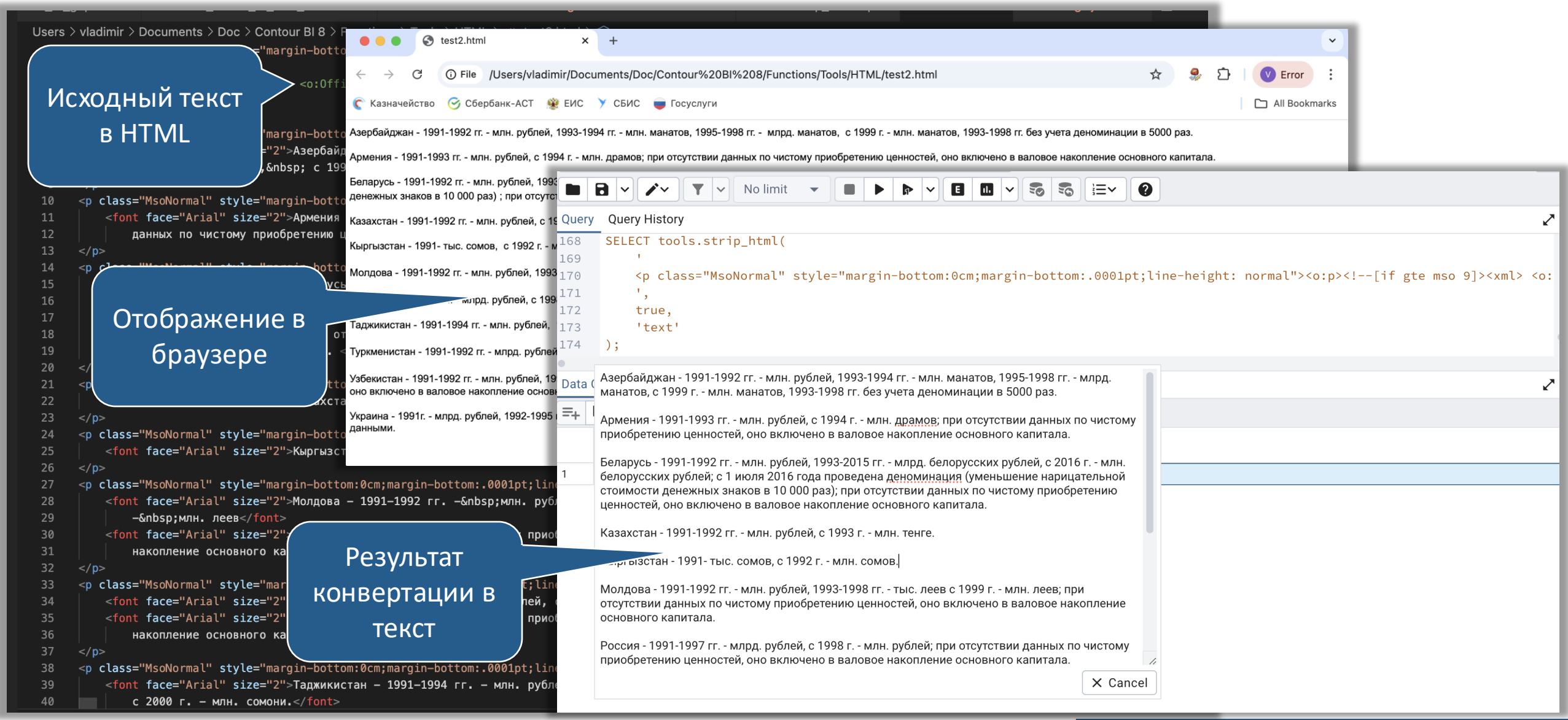
	strip_html text
1	Prototype

Результат: чистый текст «Прототип»

info@contourcomponents.com

contourcomponents.com

Другой пример



info@contourcomponents.com

Итерационная разработка

Задание

1. Создай код для очистки HTML
2. Имитируй структуру HTML – переводы строк, перечисления, и т.д.
3. Удали лишние пробелы
4. Добавь пробелы после знаков пунктуации
5. Переведи первые буквы предложений в верхний регистр
6. …

Искусственный интеллект

Исходный текст функции

Выполнение функции

Результат

Итерации

- Интеллектуальные возможности человека – постановщика задачи не безграничны
- Пошаговое улучшение и развитие алгоритмы проще
- Скорость генерации новой версии имеет решающее значение

ИИ против программиста

Программист

- **2 дня/версия**
- **Нарастание сложности кода замедляет разработку**

3 итерации в неделю

ИИ

- **2 сек/версия**
- **Нарастание сложности кода не влияет на скорость разработки**

30 итераций в час

info@contourcomponents.com

contourcomponents.com

Документирование функции

Задание:
“Пожалуйста, опиши алгоритмы конвертации HTML в текст”

Детальная документация

Функция tools.strip_html направлена на преобразование вводимых HTML данных в чистый текст или HTML, при этом при необходимости сохраняются определенные структурные элементы, такие как переводы строк. Ниже приведена разбивка необходимых шагов и используемых алгоритмов:

1. Валидация и подготовка входных данных:
Функция принимает параметры html_text (ввод HTML), save_structure (следует ли сохранять форматирование, например, переносы строк) и destination_format (текст или html). destination_format проверяется на валидность, и если это не 'text' или 'html', возникает исключение. Если destination_format имеет значение 'html', save_structure автоматически устанавливается значение true.

2. Удаление условных комментариев:
regexp_replace используется для поиска и удаления условных комментариев HTML, таких как <!--[if ...]-->... <![endif]-->. Это делается с помощью регулярных выражений, которые соответствуют началу и концу этих блоков, обеспечивая их удаление из содержимого.

3. Удаление определенных XML-подобных тегов:
Подобные регулярные выражения удаляют блоки <xml>...</xml> и другие пользовательские теги HTML/XML, такие как <o:p>...</o:p>, которые часто являются остатками редакторов форматированного текста, таких как Microsoft Word.

4. Удаление тегов script, style, meta и head:
<style>...</style>, <script>...</script>, <head>...</head>, и <meta> теги (и их содержимое, если применимо) удаляются, так как они не нужны для обычного текста. Эти элементы удаляются, потому что они обычно содержат невидимое содержимое, такое как скрипты или метаданные, не внося свой вклад в основное текстовое содержимое документа.

5. Работа с блоками CSS:
Все блоки CSS (обозначенные фигурными скобками {...}) удаляются. Это осторожный подход, чтобы не оставить позади встроенные стили CSS, которые не имеют отношения к текстовому выводу.

6. Замена HTML-сущностей:
Некоторые распространенные HTML-сущности, такие как , <, >, &, ", и ' заменяются соответствующими символами (пробел, <, >, &, ", и '). Это гарантирует, что выходные данные будут читаемыми без артефактов кодировки HTML.

7. Структурная консервация (опционально):
Если save_structure истинно, некоторые ключевые структурные элементы HTML, такие как ,
, и заменяются переводами строк для сохранения форматирования содержимого. Это гарантирует, что текст, разбитый на несколько строк в HTML, останется визуально отформатированным в обычной текстовой версии. Несколько переводов строк сворачиваются в одну новую строку, чтобы избежать ненужных пробелов. Если save_structure имеет значение false, структурные теги типа and
 заменяются одним пробелом, что приводит к более компактному представлению без видимой структуры...

Очистка статистических метаданных

Ошибки в справочниках

- Пустые коды
- Пустые значения
- Неверные значения
- Дубли
- Неверные типы данных
- ...

info@contourcomponents.com

contourcomponents.com

Прямое решение

Задание:

“Пожалуйста найди ошибки в справочнике и кратко опиши их в третьей колонке”

Код	Значение
1	
2	Fish
	Meat
4	Male
5	Vegetables
01/01/2022	
7	$123

Here's a table with the errors identified and described:

Code	Value	Error Description
1	(empty)	Missing value
2	Fish	No error
(empty)	Meat	Missing code
4	Male	No error
5	Juice	No error
01/01/2022	Bread	Invalid code format (date instead of a number)
7	$123	Invalid format (contains special character "$")

info@contourcomponents.com

contourcomponents.com

Альтернативное решение

Задание:

“Пожалуйста создай функцию на языке PL/pgSQL, которая находит ошибки в справочнике и возвращает таблицу с описанием ошибок”

Код	Значение
1	
2	Fish
	Meat
4	Male
5	Vegetables
01/01/2022	
7	$123

To create a PL/pgSQL function that receives a code list, identifies errors, and returns a table with error descriptions, we can design the function to accept an input table or a recordset and perform validation checks based on conditions such as missing values, invalid formats, or special characters.

Here’s an example of such a function:

Function Definition:

sql Copy code

```
CREATE OR REPLACE FUNCTION find_code_list_errors(code_list TEXT[])
RETURNS TABLE (code TEXT, value TEXT, error_description TEXT) AS
$$
DECLARE
    code_value RECORD;
BEGIN
    -- Loop through the provided code list (array of records)
    FOREACH code_value IN ARRAY code_list
    LOOP
        RETURN QUERY
        SELECT
            code_value.code,
            code_value.value,
            CASE
                -- Check if the code is NULL or empty
                WHEN code_value.code IS NULL OR code_value.code = '' TH

                -- Check if the value is NULL or empty
                WHEN code_value.value IS NULL OR code_value.value = ''

                -- Check if the code contains non-numeric characters
                WHEN code_value.code !~ '^[0-9]+$' THEN 'Invalid code

                -- Check if the code contains special characters or inv
                WHEN code_value.code ~ '[^0-9]' THEN 'Invalid characte

                -- No errors
                ELSE 'No error'
            END AS error_description;
    END LOOP;
END;
$$ LANGUAGE plpgsql;
```

info@contourcomponents.com

contourcomponents.com

Выполнение функции, созданной ИИ

Code	Value
1	
2	Fish
	Meat
4	Male
5	Vegetable
01/01/2022	
7	$123

Code	Value	Error Description
1	(empty)	Missing value
2	Fish	No error
(empty)	Meat	Missing code
4	Male	No error
5	Juice	No error
01/01/2022	Bread	Invalid code format (date instead of a number)
7	$123	Invalid format (contains special character "$")

Преимущества

- Бесплатно: Мы не используем ChatGPT для обработки данных
- Скорость работы: Используются собственные серверы и базы данных
- Удобство: Созданная функция встраивается в приложение

info@contourcomponents.com

contourcomponents.com

Преимущества альтернативного метода применения ИИ для очистки данных

Генерация программного кода вместо обработки данных

- ❑ Бесплатно
- ❑ Независимо
- ❑ Встраивается в приложение
- ❑ Безопасно

info@contourcomponents.com

contourcomponents.com

Советы ИИ

Вопрос: Как можно использовать ИИ для верификации и очистки статистических данных?

Ответ:

Искусственный интеллект (ИИ) может быть использован для проверки статистических данных несколькими способами, в первую очередь за счет автоматизации процессов проверки данных, выявления аномалий, обеспечения согласованности и повышения общей точности наборов данных.

- Автоматизированная очистка данных
- Предиктивный анализ
- Сверка данных
- Контроль качества ввода данных
- Обнаружение предвзятости и анализ честности
- Статистическое тестирование на основе ИИ
- Верификация больших данных в режиме реального времени

Спасибо!

info@contourcomponents.com

contourcomponents.com