НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

КУРСОВАЯ РАБОТА НА ТЕМУ: ФОРМАТЫ ДАННЫХ TEMPLATE И REGEXP. УСКОРЕНИЕ BATCH INSERT С ВЫЧИСЛИМЫМИ ВЫРАЖЕНИЯМИ ПУТЁМ ВЫВОДА ШАБЛОНОВ.

Токмаков Александр Викторович, студент группы БПМИ165

Научный руководитель:
Миловидов Алексей Николаевич,
руководитель группы разработки СУБД ClickHouse, Яндекс

Москва, 2019

ПРЕДМЕТНАЯ ОБЛАСТЬ И АКТУАЛЬНОСТЬ

Форматы данных в ClickHouse

Запросы вида **INSERT INTO** … **FORMAT** CSV ... **и** **SELECT** ... **FORMAT** CSV ...

- Поддерживается много различных текстовых форматов: CSV, TSV, JsonEachRow и другие
- Все эти форматы очень похожи: отличия в разделителях и способах экранирования значений
- Иногда нужен необычный формат, который не реализован в ClickHouse
- Приходится использовать awk, sed или что-то ещё, чтобы преобразовать данные к одному из поддерживаемых форматов
- Или добавлять в ClickHouse ещё один формат или ещё одну настройку

ЗАДАЧА 1:

Реализовать более гибкие форматы: Template и Regexp

Template для ввода и вывода:

- Позволяет указать произвольную форматную строку:

```
delim_1${col_1:serializeAs_1}delim_2${col_2:serializeAs_2}...delim_N
```

- Вместо `${column:serializeAs}` выводится или считывается значение в столбце column, экранированное как `serailizeAs` (`CSV`, `JSON`, `Quoted`, `Raw`, ...)
- Позволяет указать префикс, суффикс, разделитель между строками, вывести дополнительную информацию

Regexp для ввода: Заполняет значения столбцов из match'ей регулярного выражения

ПРИМЕР

Использование Template для вывода HTML-страницы

```
SELECT SearchPhrase, count() AS c FROM test.hits GROUP BY SearchPhrase ORDER BY c DESC LIMIT 5
FORMAT Template SETTINGS
format_schema = '<!DOCTYPE HTML>
<html> <head> <title>Search phrases</title> </head>
  <body>
    <table border="1"> <caption>Search phrases</caption>
      <tr> <th>Search phrase</th> <th>Count</th> </tr>
        ${data}
   </table>
   <table border="1"> <caption>Max</caption>
      ${max}
   </table>
   <b>Processed ${rows_read:XML} rows in ${time:XML} sec</b>
  </body>
</html>',
format_schema_rows = '<tr> <td>${SearchPhrase:XML}</td> <td>${c:XML}</td> </tr>',
format_schema_rows_between_delimiter = '\n    '
```

ПРИМЕР

Использование Template для вывода HTML-страницы

```
<!DOCTYPE HTML>
<html> <head> <title>Search phrases</title> </head>
 <body>
  <table border="1"> <caption>Search phrases</caption>
    <tr> <th>Search phrase</th> <th>Count</th> </tr>
    <tr> <td></td> <td>8267016</td> </tr>
    <tr> <td>bathroom interior design</td> <td>2166</td> </tr>
    <tr> <td>yandex</td> <td>1655</td> </tr>
    <tr> <td>spring 2014 fashion</td> <td>1549</td> </tr>
    <tr> <td>freeform photos</td> <td>1480</td></tr>
  </table>
  <table border="1"> <caption>Max</caption>
    <tr> <td></td> <td>8873898</td> </tr>
  </table>
  <b>Processed 3095973 rows in 0.1569913 sec</b>
 </body>
</html>
```

Search phrases

Search phrase	Count
	8267016
bathroom interior design	2166
yandex	1655
spring 2014 fashion	1549
freeform photos	1480

Max

	8873898

Processed 3095973 rows in 0.1569913 sec

ПРЕДМЕТНАЯ ОБЛАСТЬ И АКТУАЛЬНОСТЬ

Форматы данных в ClickHouse: Values

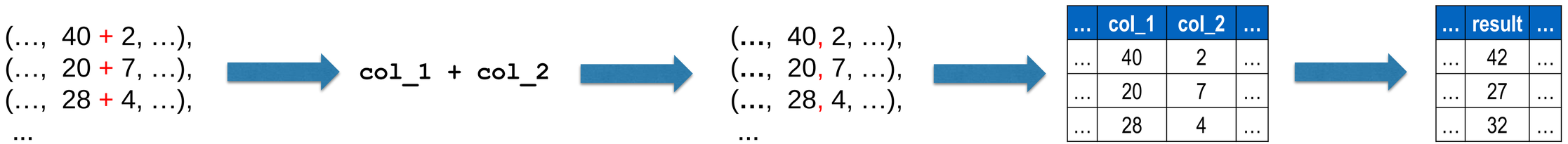
Запросы вида **INSERT INTO** … **VALUES** ... **(или** **INSERT INTO** ... **FORMAT** Values ... **)**

- Быстрый потоковый парсер: просто читает значения, разделённые запятыми:
  ('2019-06-07', 42, 'hello'), ('2019-06-07', 27, 'world'), ...
- Но могут встречаться вычислимые выражения:
  (**today**(), 40 + 2, **lower**('Hello')), (**today**(), 20 + 7, **lower**('World')), ...
- Интерпретация для одной строки – огромный оверхэд
- Скорее всего, выражения одинаковы во всех строках

ЗАДАЧА 2

Вывод шаблонов SQL-выражений в Values

- Если встретилось вычислимое выражение – вывести шаблон
- Пытаться применить шаблон для чтения следующих строк
- При чтении по шаблону литералы в выражении считываются во временные столбцы в памяти
- Вычислить выражение сразу для всех строк, к которым подошёл шаблон



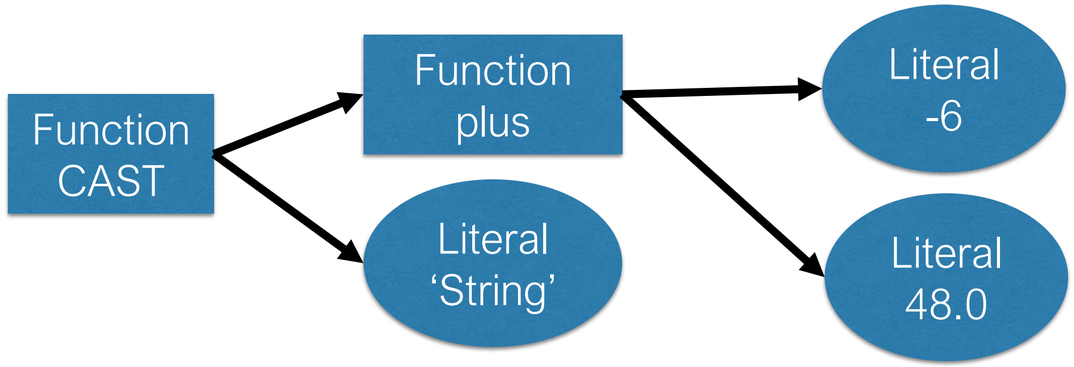
ПАЙПЛАЙН ВЫЧИСЛЕНИЯ ВЫРАЖЕНИЯ

CAST(-6 + 48.0, 'String')

Lexer

BareWord	OpenRb	Minus	Number	Plus	Number	Comma	String	CloseRb
CAST	(	-	6	+	48.0	,	'String'	)

Parser



SyntaxAnalizer

EpressionAnalizer

Interpreter

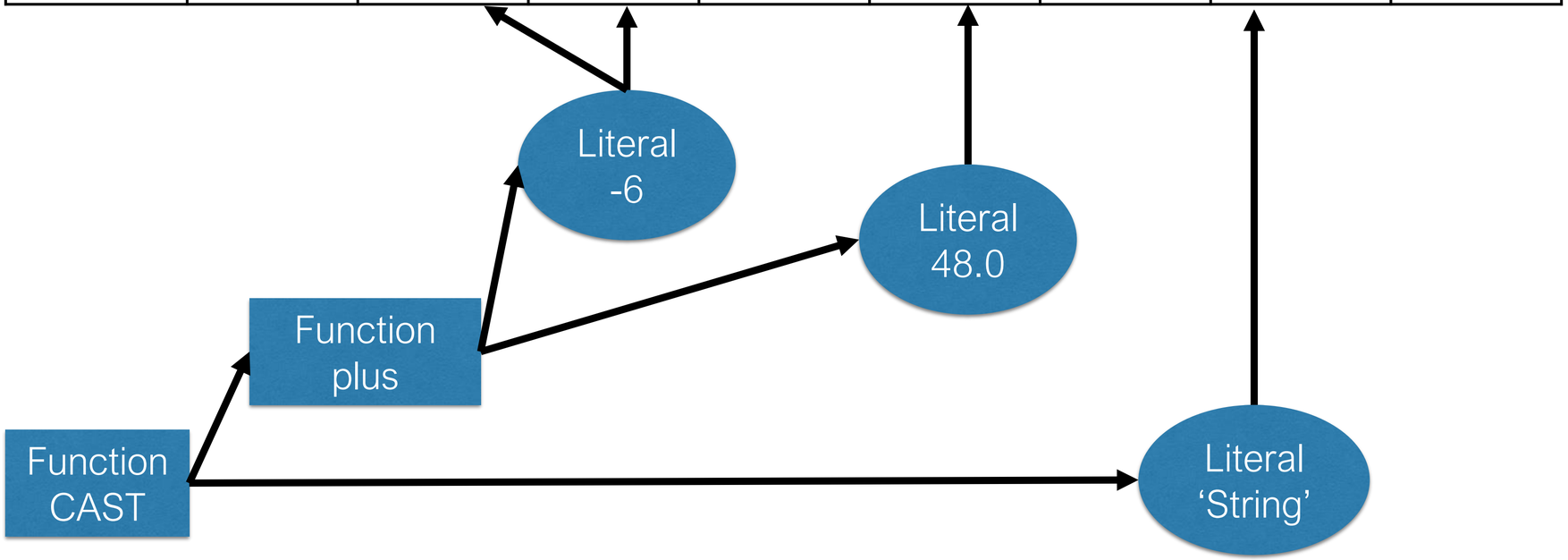
```
                -6:    Int8
              48.0: Float64
plus(Int8, Float64): Float64
cast_to_string(Float64):  String
```

ВЫВОД ШАБЛОНА

- Литералы ссылаются на свои токены

CAST(-6 + 48.0, 'String')

BareWord	OpenRb	Minus	Number	Plus	Number	Comma	String	CloseRb
CAST	(	-	6	+	48.0	,	'String'	)

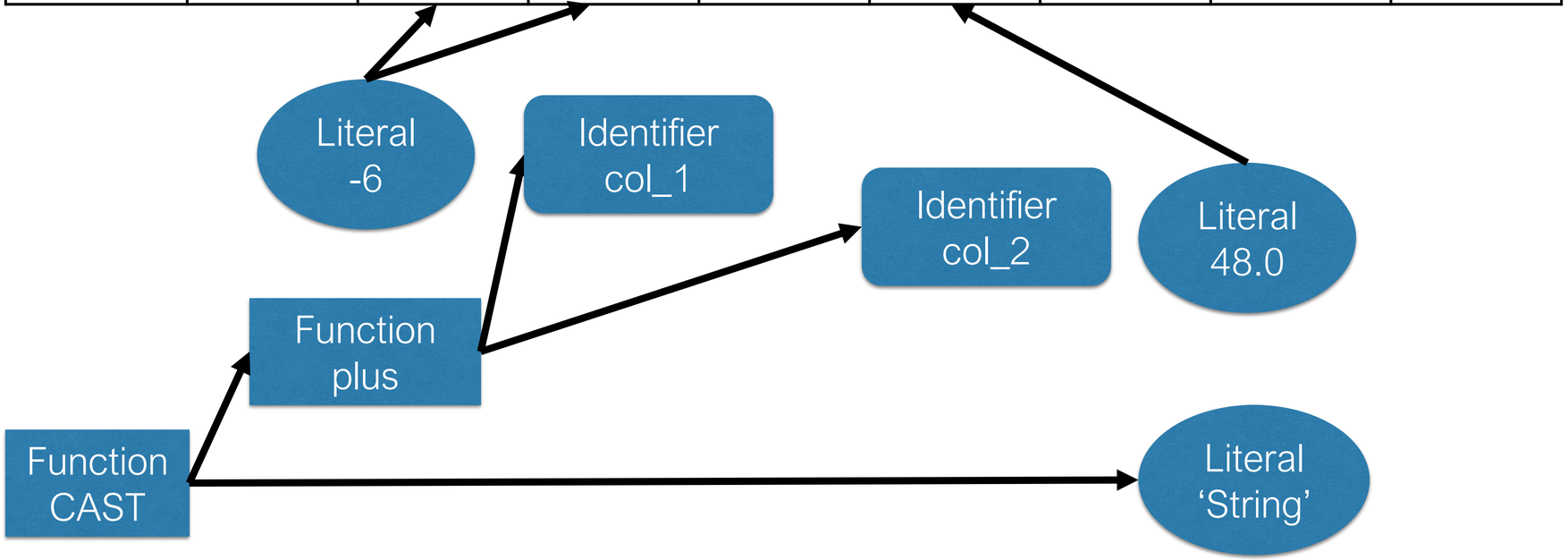


ВЫВОД ШАБЛОНА

- Литералы ссылаются на свои токены
- Литералы в дереве заменяются идентификаторами фиктивных столбцов
- Если литерал – константный аргумент функции, он не заменяется

CAST(-6 + 48.0, 'String')

BareWord	OpenRb	Minus	Number	Plus	Number	Comma	String	CloseRb
CAST	(	-	6	+	48.0	,	'String'	)

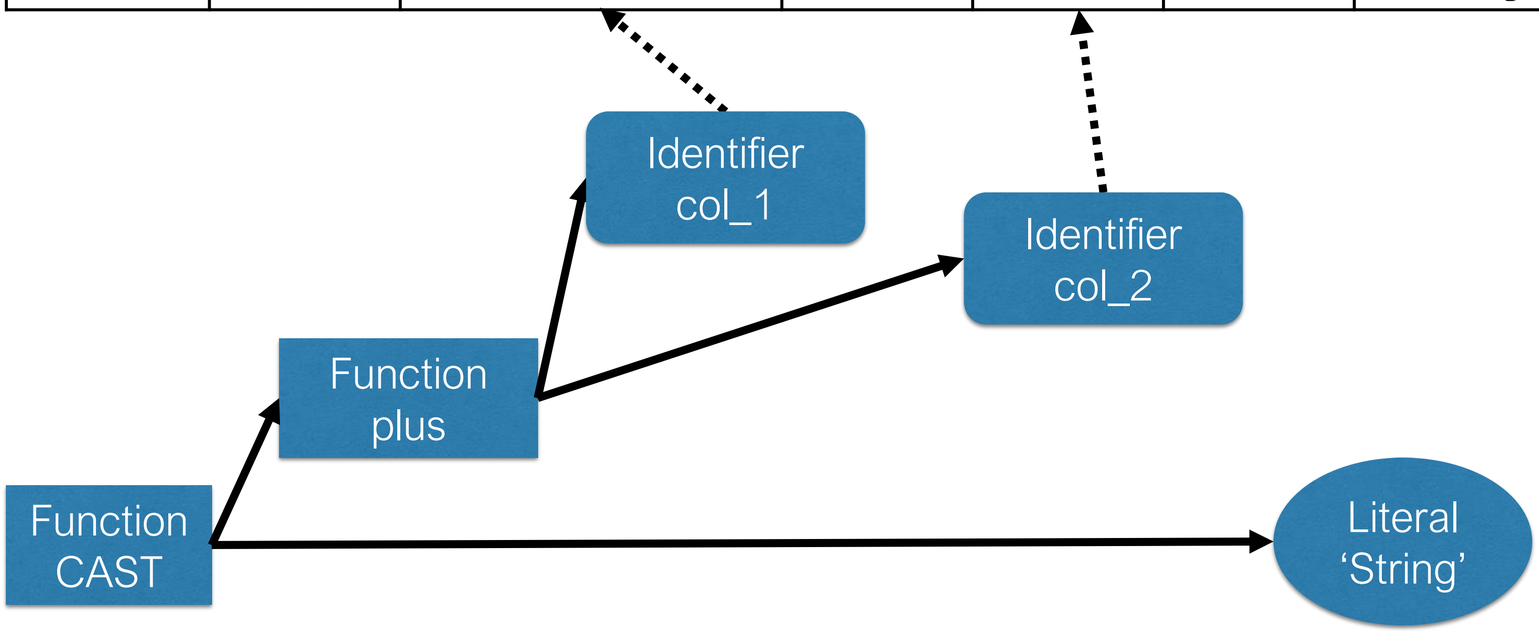


ВЫВОД ШАБЛОНА

- Литералы ссылаются на свои токены
- Литералы в дереве заменяются идентификаторами фиктивных столбцов
- Если литерал – константный аргумент функции, он не заменяется
- Определяются типы столбцов для литералов

CAST(-6 + 48.0, 'String')

BareWord	OpenRb	Int64	Plus	Float64	Comma	String	CloseRb
CAST	(	???	+	???	,	'String'	)



ВЫВОД ШАБЛОНА

Шаблон выражения:

CAST	(	Int64	+	Float64	,	'String'	)

Временная «таблица»:

col_1 Int64	col_2 Float64
...	...
...	...
...	...

Внутренние представление для интерпретатора

```
                  col_1:   Int64
                  col_2: Float64
plus(Int64, Float64): Float64
cast_to_string(Float64):  String
```

СРАВНЕНИЕ ПРОИЗВОДИТЕЛЬНОСТИ

Лучший случай: шаблон подходит для всех строк

100000 строк из датасета wikistat, 7 стобцов, к четырём искусственно добавлены выражения:

```
(
        '2016-01-01',
        timeSlot(CAST('2016-01-01 09:00:00', 'DateTime')),
        'en',
        '',
        lower(TRIM (BOTH ' \t\n' FROM ' MonoBook/bullet.gif  ')),
        12 + 1*42 - 42,
        CAST(81266 + 0.0, 'UInt64')
) ,
(
        '2016-01-02',
        timeSlot(CAST('2016-01-02 07:00:00', 'DateTime')),
        'am',
        '',
        lower(TRIM (BOTH ' \t\n' FROM ' w/index.php  ')),
        7 + 1*42 - 42,
        CAST(5540 + 0.0, 'UInt64')
) , …
```

Среднее время, 10 запусков:

Старая версия с построчной интерпретацией выражений	Новая версия с выводом шаблонов
real 0m40.9153s user 0m40.4882s sys 0m0.2959s	real 0m0.3204s user 0m0.2444s sys 0m0.0299s

СРАВНЕНИЕ ПРОИЗВОДИТЕЛЬНОСТИ

Худший случай: для каждой строки выводится новый шаблон

- Те же данные
- Симулируется поведение, будто шаблон всегда не подходит
- Количество попыток вывести шаблон – 100000 (по кол-ву строк)
- Медленнее в ~3 раза
- Но можно использовать эвристики

Среднее время, 10 запусков:

```
real    1m25.74109999999999s
user    1m25.13500000000002s
sys     0m0.438s
```

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ