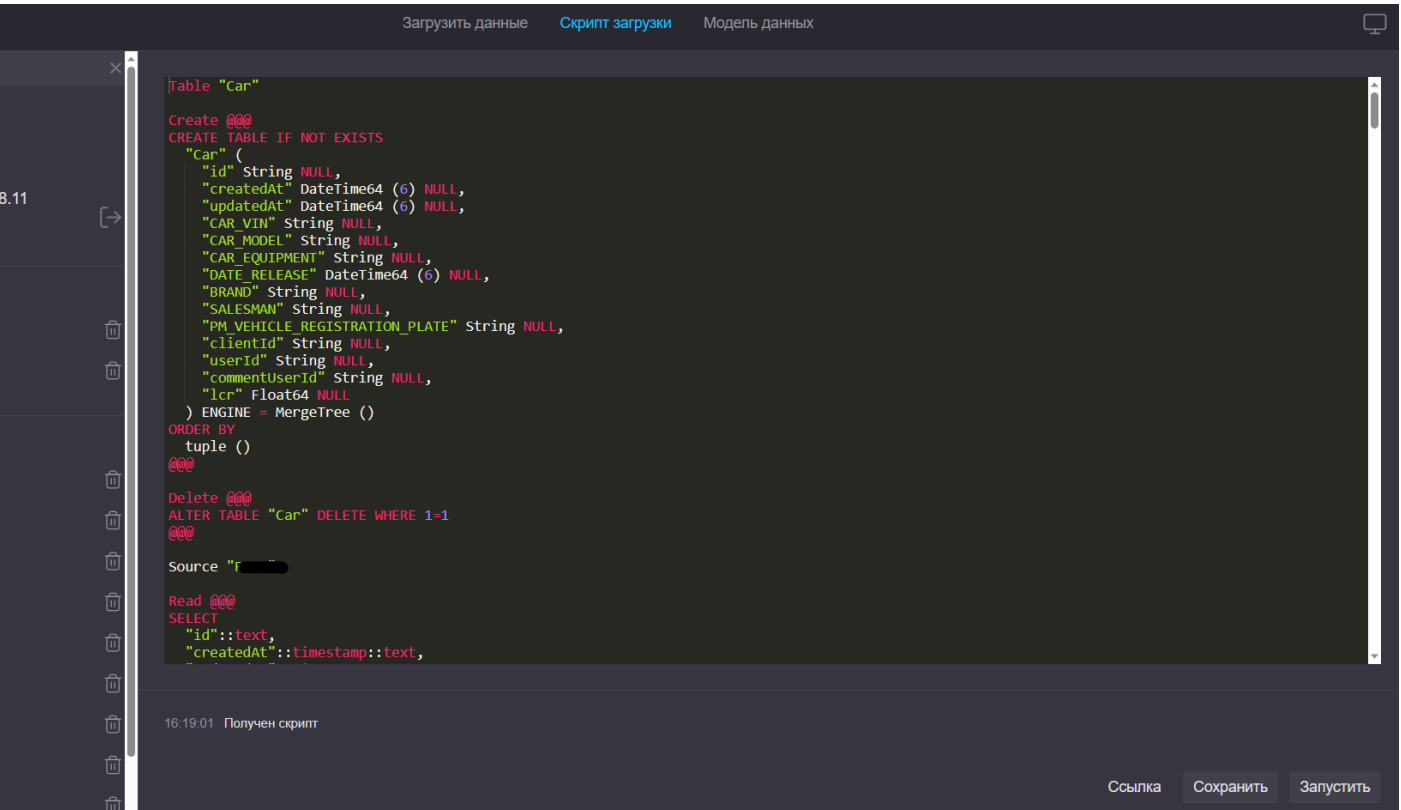


Редактор скрипта загрузки

Скрипт загрузки генерируется автоматически после [выбора источников данных](#).

Этап ручного редактирования скрипта загрузки является необязательным, однако функциональность Fastboard позволяет при необходимости внести изменения.



Правила использования редактора

Добавить источник

Шаги	Ожидаемый результат
<p>В любом месте скрипта добавить строку <i>Source "Название коннекта"</i> где:</p> <ul style="list-style-type: none">• Source - ключевое слово обозначающее вставку нового коннекта,• "Название источника" - имя коннекта из списка источников	<ul style="list-style-type: none">• При вставке кода для импорта таблиц из этого источника таблицы успешно загрузятся в БД проекта

Добавить таблицу из источника

Шаги	Ожидаемый результат
------	---------------------

- Объявить название таблицы:

Table "Car"

- Определить и указать скрипту список полей новой таблицы. Для этого:

1. Описать секцию Create (команда на языке Clickhouse для создания таблицы в БД проекта, в которую запишутся данные из таблицы-источника):

Create @@@ - начало секции

CREATE TABLE IF NOT EXISTS - команда KX на создание таблицы

"id" String NULL, - поле таблицы, в которое будет записано значение из

источника: "название поля" / тип данных / может ли быть пустым (если нет, not null)

"createdAt" DateTime64 (6) NULL,

"updatedAt" DateTime64 (6) NULL,

) ENGINE = MergeTree ()

ORDER BY

tuple ()

@@@ - конец секции Create

2. Описать секцию Delete (для удаления временной таблицы):

Delete @@@

ALTER TABLE "Car" DELETE WHERE 1=1

@@@

3. Описать секцию Read (Для выбора полей из таблицы-источника):

Read @@@

SELECT - команда (на языке источника) на выбор полей

"id"::text, - поле таблицы, которое будет взято из источника. "название поля" / тип данных. Для избежания ошибок при импорте всем полям данной секции прописывается тип ::text

"createdAt"::text,

"updatedAt"::text,

FROM

"Car" - таблица источника

@@@

4. Описать секцию Write

Write @@@

INSERT INTO - команда на языке KX на вставку данных созданную в KX

"Car" (

"id",

"createdAt",

"updatedAt"

)

@@@

5. Необязательная секция

Optimize @@@

OPTIMIZE TABLE "Car" DEDUPLICATE

@@@

- Добавленная таблица появилась в списке доступных на странице модели данных проекта

- Новая таблица содержит все поля указанные в скрипте загрузки

- Новая таблица может быть использована в модели данных

Удалить таблицу

Шаги	Ожидаемый результат
<ol style="list-style-type: none">1. Удалить строку объявляющую таблицу <i>Table "Car"</i>2. Удалить секции Create, Delete, Read, Write для этой таблицы3. Нажать кнопку «Сохранить». Для записи изменений в тексте скрипта4. Нажать кнопку «Запустить». Для старта импорта данных согласно сохраненному скрипту загрузки	Удаленная таблица исчезла из списка таблиц на странице модели данных

Добавить поле из таблицы источника в таблицу импорта

Шаги	Ожидаемый результат
<ol style="list-style-type: none">1. В список полей таблицы куда вставляем секции Create добавить строку: <i>"id" String NULL</i> - "название поля" / тип данных / может быть пустым Null (если нет, not null)2. В список полей таблицы откуда импортируем для секции Read добавить строку с названием поля: <i>"id":text</i> - "название поля" / тип данных (всегда текст)3. В список полей данной таблицы в секции Write добавить строку с названием поля: <i>"id"</i>4. Нажать кнопку «Сохранить». Для записи изменений в тексте скрипта5. Нажать кнопку «Запустить». Для старта импорта данных согласно сохраненному скрипту загрузки	На странице модели данных в данной таблице появилось поле "id" с типом данных, указанным в секции Create

Удалить поле из таблицы

Шаги	Ожидаемый результат
<div><div>1. В списке полей таблицы в секциях Create, Delete, Read, Write удалить необходимое поле</div><div>2. Нажать кнопку «Сохранить». Для записи изменений в тексте скрипта</div><div>3. Нажать кнопку «Запустить». Для старта импорта данных согласно сохраненному скрипту загрузки</div></div>	Удаленное поле исчезло таблицы на странице модели данных

Изменить тип данных поля

Шаги	Ожидаемый результат
<div><div>1. В строке поля таблицы в секции Create изменить тип на один из поддерживаемых КХ: String, Int64, Float32 и т.д. Полный список типов</div><div>2. Нажать кнопку «Сохранить». Для записи изменений в тексте скрипта</div><div>3. Нажать кнопку «Запустить». Для старта импорта данных согласно сохраненному скрипту загрузки</div></div>	Тип поля изменился в таблице

В настоящий момент для применения любых изменений в таблицах (создание поля, переименование поля, изменение типа поля и т.д.) необходимо пересоздать таблицу в БД проекта. Для этого после внесения всех изменений к имени таблицы можно добавить, например _1, после этого сохранить, затем запустить скрипт. При необходимости вернуть таблице старое название тем же способом. *(Это связано с текущими ограничениями парсера. Мы над этим работаем)*

Создать вычисляемое поле в таблице

Шаги	Ожидаемый результат
------	---------------------

При импорте таблицы из источника перед вставкой таблицы в КХ можно создать поле которого нет в исходной таблице, но которое будет вычислено и создано на основе заданного выражения. Для этого:

1. В список полей таблицы в секции Create добавить новую строку с названием, типом, null, например:
`"field_name" Int32 NULL`
2. В секции Read добавить строку вида:
`func::text as "field_name"` , где
func - вычисляемое выражение,
например `"field_name1" + "field_name2"`,
`::text` - тип данных (всегда текст),
`as "field_name"` - алиас поля
3. В список полей данной таблицы в секции Write добавить строку с названием поля:
`"field_name"`
4. Нажать кнопку «Сохранить». Для записи изменений в тексте скрипта
5. Нажать кнопку «Запустить». Для старта импорта данных согласно сохраненному скрипту загрузки

В таблице появится поле с назначенным типом данных и рассчитанными по заданному выражению значениями

Создать новую таблицу "Календарь"

Вставить в скрипт загрузки следующий текст (обратите внимание на комментарии), после выполнения скрипта загрузки выполнить JOIN таблицы Calendar к вашей таблице фактов.

Table "calendar"

Create @@@

CREATE TABLE IF NOT EXISTS

"calendar" (

"id" Int32 NULL,

"key_date" String NULL,

"date" Date32 NULL,

"index_day" Int32 NULL,

"day" String NULL,

"week" Int32 NULL,

"quarter" Int32 NULL,

"year" Int32 NULL,

"index_month" Int32 NULL,

```
"month" String NULL
```

```
) ENGINE = MergeTree ()
```

```
ORDER BY
```

```
tuple ()
```

```
@@@
```

```
Delete @@@
```

```
ALTER TABLE "calendar" DELETE WHERE 1=1
```

```
@@@
```

Source "promo_fb (RomanS)" -- Укажите любой существующий источник, чтобы сохранить в него вашу таблицу

```
Read @@@
```

```
SELECT
```

```
  a."id"::text,
```

```
  a."key_date"::text,
```

```
  a."date"::text,
```

```
  a."index_day"::text,
```

```
  a."day"::text,
```

```
  a."week"::text,
```

```
  a."quarter"::text,
```

```
  a."year"::text,
```

```
  a."index_month"::text,
```

```
  a."month"::text
```

```
FROM
```

```
(
```

```
select distinct
```

```
  row_number() over() as id,
```

```
  date(date)::text as key_date,
```

```
  date::date,
```

```
  extract('isodow' from date) as index_day,
```

```
  CASE
```

```
    WHEN extract('isodow' from date) = 1 then 'ПН' -- Если нужно, укажите названия дней (и месяцев  
ниже)
```

```
    WHEN extract('isodow' from date) = 2 then 'BT'
```

```
    WHEN extract('isodow' from date) = 3 then 'CP'
```

```
    WHEN extract('isodow' from date) = 4 then 'ЧТ'
```

```
    WHEN extract('isodow' from date) = 5 then 'ПТ'
```

```

    WHEN extract('isodow' from date) = 6 then 'СБ'
    WHEN extract('isodow' from date) = 7 then 'ВС' end as day,
extract('week' from date)    as week,
extract('quarter' from date ) as quarter,
extract('year' from date)    as year,
extract('month' from date)   as index_month,
CASE
    WHEN extract('month' from date) = 1 then 'Январь'
    WHEN extract('month' from date) = 2 then 'Февраль'
    WHEN extract('month' from date) = 3 then 'Март'
    WHEN extract('month' from date) = 4 then 'Апрель'
    WHEN extract('month' from date) = 5 then 'Май'
    WHEN extract('month' from date) = 6 then 'Июнь'
    WHEN extract('month' from date) = 7 then 'Июль'
    WHEN extract('month' from date) = 8 then 'Август'
    WHEN extract('month' from date) = 9 then 'Сентябрь'
    WHEN extract('month' from date) = 10 then 'Октябрь'
    WHEN extract('month' from date) = 11 then 'Ноябрь'
    WHEN extract('month' from date) = 12 then 'Декабрь' end as month
from generate_series(date'2015-01-01',date(now()),interval '1 day')as t(date) -- Выберите дату начала и
интервал
order by
    date desc) a
@@@

```

Таблица в результате:

calendar

calendarINNERdata2

=

Таблица

+ Связать

12 id	T key_date	📅 date	12 index_day	T day	12 week	12 quarter	12 year	12 index_month	T month
3183	2023-09-18	2023-09-18	1	ПН	38	3	2023	9	Сентябрь
3182	2023-09-17	2023-09-17	7	ВС	37	3	2023	9	Сентябрь
3181	2023-09-16	2023-09-16	6	СБ	37	3	2023	9	Сентябрь

Предварительный просмотр

Некоторые особенности работы разными типами источников

MS SQL

1. При импорте данных с типом «Дата» нужно будет поля с датами в секции READ для этой таблицы сконвертировать в строку. КХ сам сконвертирует тип дата при создании таблицы у себя и они снова станут датами

```
Read @@@
SELECT
  "CreatedDate" ::text,
  "ActivatedDate" ::text,
  "PlannedEndDate" ::text,
```

2. Поля с кодировкой UTF16 (тип varchar) нужно будет обернуть в cast. В той же секции @READ для этой таблицы. Иначе русскоязычные значения не распознаются в этих полях и появятся вопросительные знаки

```
Read @@@
SELECT
  "id",
  cast("name" as nvarchar) as "name"
```

Revision #16

Created 5 April 2023 07:56:21

Updated 22 February 2024 07:29:10 by Станислав