

# Импорт данных из коннектора

## Пользовательские сценарии

Стартовое состояние – пользователь находится в диспетчере данных, загружен список источников

**Действие:** Пользователь нажимает на "доступный" источник

**Результат 1** (если у источника нет ошибок – "status" = "success"):

- Открывается окно выбора таблиц из источника
- Система не выдаёт ошибку
- В левой части окна загружается список таблиц источника
- Ни одна таблица не выбрана, область просмотра табличной части пустая

**Результат 2** (если у источника возникла ошибка при пройденной проверке на валидность – "status" = "aborted"):

- Открывается окно выбора таблиц из источника
- Система выдаёт ошибку, полученную при загрузке источника (поле "message")
- В левой части окна отсутствует список источников
- Область просмотра табличной части пустая

**Действие:** Пользователь нажимает на "сломанный" источник

**Результат** (аналогичен доступному источнику с ошибкой):

- Открывается окно выбора таблиц из источника
- Система выдаёт ошибку, полученную при загрузке источника (поле "message")
- В левой части окна отсутствует список источников
- Область просмотра табличной части пустая

**Действие:** Пользователь нажимает на "занятый" источник

## Результат:

- Открывается окно со статусом загрузки в процентах
- До окончания загрузки невозможно просмотреть табличные данные
- По окончании загрузки источник получает статус "доступен" или "сломан"



Стартовое состояние – открыта область выбора данных из источника

**Действие:** Пользователь нажимает на таблицу в списке

**Результат:** В области просмотра данных появляется таблица с данными

## Действия:

- Пользователь меняет тип данных поля
- Пользователь выбирает/исключает из выбора столбец/таблицу
- Пользователь меняет алгоритм генерации скрипта (Drop/Alter)

## Результат:

- Изменения фиксируются в интерфейсе
- До подтверждения загрузки никаких взаимодействий с бэкендом не происходит

**Действие:** Пользователь нажимает на кнопку "Далее"

**Результат:** Формируется скрипт загрузки из указанных таблиц с указанными столбцами в указанных типах данных по указанному алгоритму

# Необходимые изменения

Добавить Чекбокс "Исключить пустые столбцы"

Добавить Статистику по столбцам

## Схема потока данных

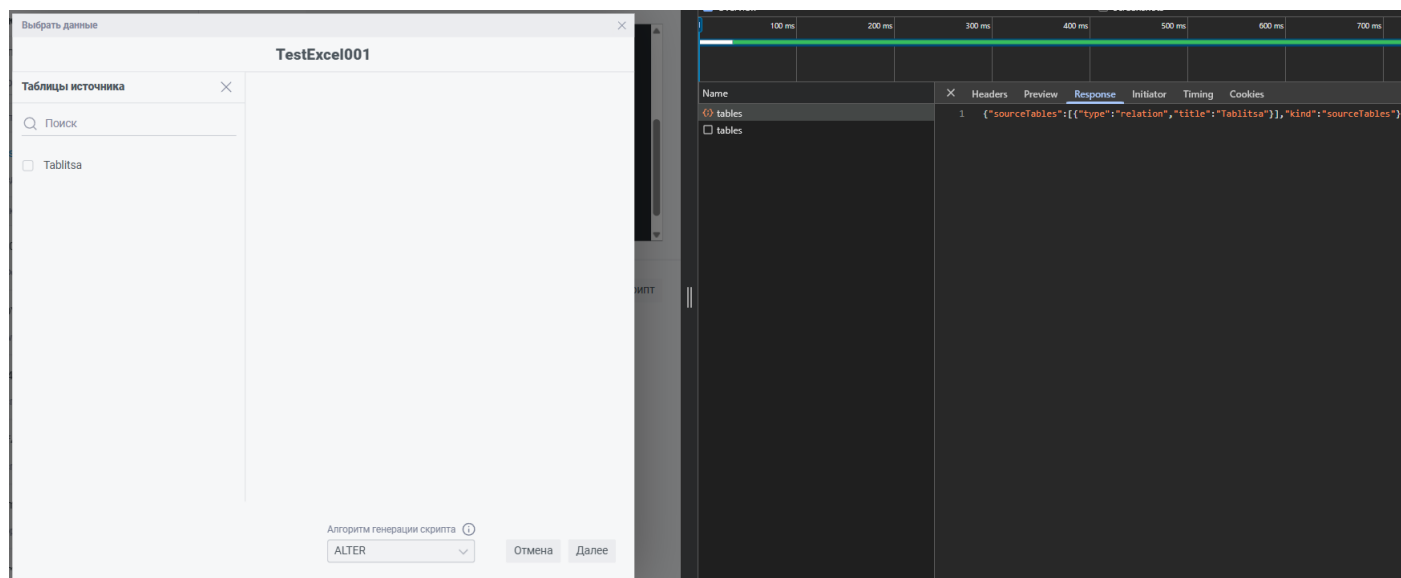
## Техническое описание

1. При нажатии на коннектор из списка в левой панели клиентская часть отправляет на сервис Nest.js GET запрос на получение соответствующих таблиц коннектора

<https://fastbord-back-dev4.fb-dev.winsolutions.ru/api/v2/source/{sourceId}/tables>, где sourceId - id выбранного коннектора

Список доступных таблиц получается путём запроса метаданных источника из бд postgres

Результат отображается в левой панели "Таблицы источника"



2. При выборе таблицы из списка "Таблицы источника" отправляются GET запросы Meta + Preview:

Meta - содержит метаданные по выбранной таблице (тип драйвера, количество записей, название таблицы, id таблицы)

```
{
  "kind": "sourceTableMeta",
  "sourceTableMeta": {
    "sourceId": "3ad0c9f4-5a92-4aa8-8776-1e5b3f0098e3",
    "tableName": "Tablitsa",
    "rowCount": 99,
    "driverType": "xlsx"
  }
}
```

<https://fastbord-back-dev4.fb-dev.winsolutions.ru/api/v2/source/{sourceId}/table/Tablitsa/meta>

Preview - содержит фактические данные для отображения превью клиенту, всю информацию по колонкам таблицы и некоторое количество записей

<https://fastbord-back-dev4.fb-dev.winsolutions.ru/api/v2/source/{sourceId}/table/Tablitsa/preview>

Данные получают путём запроса метаданных коннектора из бд postgres, поиска источника по этим метаданным на стороне сервера и дальнейшего форматирования результата запроса для отправки ответа клиентской части

Результат:

Выбрать данные

Таблицы источника

Поиск

Tablitsa

TestExcel001

Tablitsa

Исключить пустые столбцы

Кол-во строк: 99

Общее кол-во строк: 99

<input type="checkbox"/> Pokazatel Nullable(String)	<input type="checkbox"/> God Nullable(String)	<input type="checkbox"/> Kvartal Nullable(String)	<input type="checkbox"/> Vid_rynka Nullable(String)	<input type="checkbox"/> Tipy_kvartir Nullable(String)
Средняя цена 1 кв. м общей площади квар				
2023				
Страна	1 квартал			
Федеральный округ	первичный рынок жилья			
Субъект	все типы квартир	квартиры среднего качества (типовые)	квартиры улучшенного качества	элитные квартиры
Российская Федерация	127,229	115,613	127,139	292,381
Центральный федеральный округ	168,808	109,036	165,059	475,223
Белгородская область	90,760	77,660	96,009	
Брянская область	78,774	77,461	82,095	
Владимирская область	77,921	71,211	79,929	

Алгоритм генерации скрипта

ALTER

Отмена

Далее

3. Драйверы. Тип драйверов:

- clickhouse

- csv
- hive
- json
- ms
- my
- oracle
- postgres
- qvd
- rest-api
- txt
- xlsx
- xml

Каждый тип драйвера наследуется от класса `DriverDatabase` и реализует интерфейс `IDriver`

Это гарантирует, что драйвер реализует необходимые методы, такие как:

- `getCredentialsInstance` - получение данных для подключения к драйверу (если требуются)
- `getRowCount` - получение общего числа записей в источник
- `getDbMeta` - получение метаданных по бд (список таблиц в источнике, с краткой информацией по каждой)
- `getPreview2` - получение превью бд (ограниченное количество строк + список столбцов)
- `query` - выполнение любого запроса относящегося к источнику и др.

4. Смена названий и типов столбцов таблицы данных иницируется на клиентской части приложения, без создания дополнительных запросов к сервису `Nest.js` (см. скриншот ниже). Изменения будут учтены после продолжения импорта данных и нажатия на кнопку "далее".

Tablitsa

☐ Исключить пустые столбцы

Кол-во строк: 99    Общее кол-во строк: 99

☐ Pokazatel123  
Nullable(String)

☐ God  
Nullable(String)

☐ Kvar  
Nulla

Средняя цена 1 кв. м общей площади кваг

2023

Страна1 квартал

Федеральный округпервичный рынок жилья

Субъектвсе типы квартирквартиры

Российская Федерация127,229115,613

Центральный федеральный округ168,808109,036

Белгородская область90,76077,660

Брянская область78,77477,461

Владимирская область77,92171,211

Алгоритм генерации скрипта ⓘ

ALTER▼

Отмена

Далее

Изменённое поле будет отражено в скрипте загрузки в разделе @@Create создания базы данных clickhouse

```

Table "Tablitsa_2"

Create @@@
CREATE TABLE IF NOT EXISTS "Tablitsa_2" ("Pokazatel123" Nullable (String)) ENGINE = MergeTree ()
ORDER BY
    tuple ()
@@@

Alter @@@
ALTER TABLE "Tablitsa_2" DELETE
WHERE
    1 = 1
@@@

Source "TestExcel001"

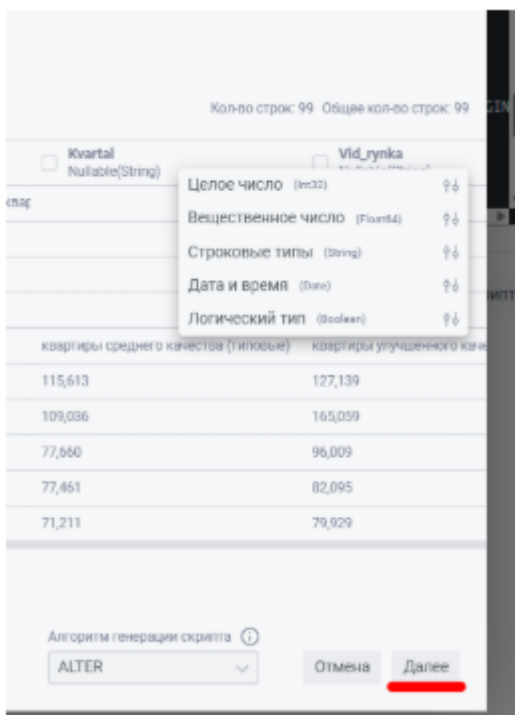
Read @@@
SELECT
    "Pokazatel"
FROM
    "Tablitsa"
@@@

Optimize @@@
OPTIMIZE TABLE "Tablitsa_2"
@@@

```

## 5. Создание скрипта загрузки

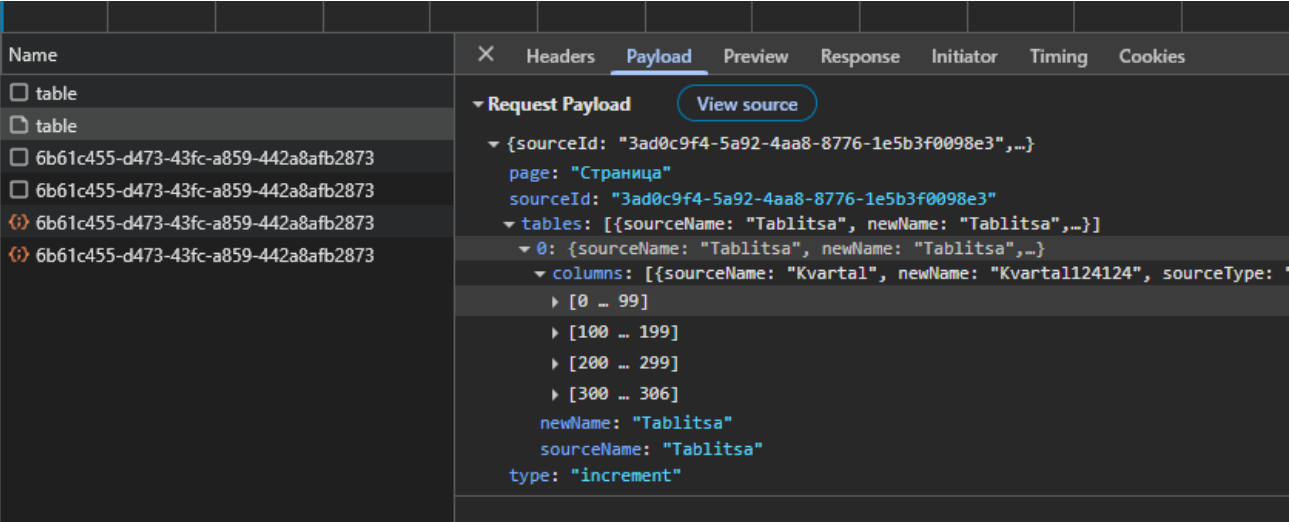
Нажатие на кнопку далее инициирует процесс формирования скрипта загрузки на основе обновленной информации



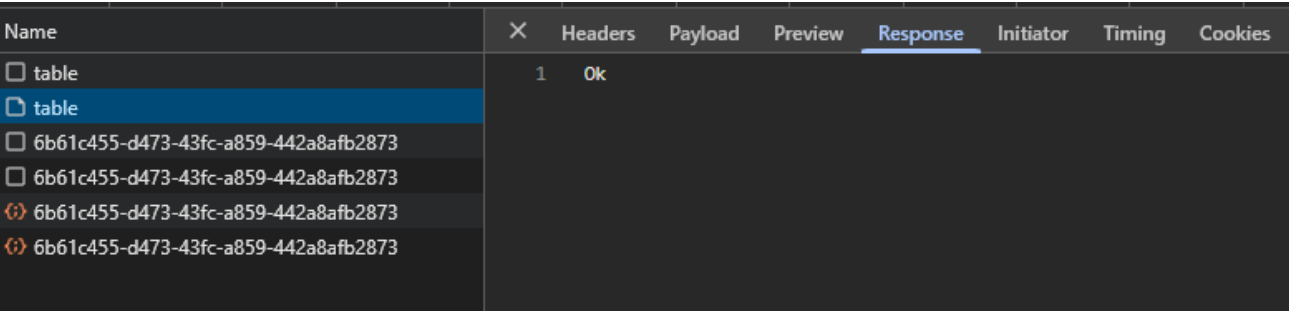
Процесс состоит из двух последовательных запросов:

а) Обновлённая структура таблицы загружается на сервис Nest.js с помощью POST запроса:

<https://fastbord-back-dev4.fb-dev.winsolutions.ru/api/v1/script/{projectId}/table>, где projectId - id проекта



Ответ (подтверждение успешного выполнения операции - "Ok"):



б) Для отображения скрипта загрузки на клиентской части выполняется GET запрос: <https://fastbord-back-dev4.fb-dev.winsolutions.ru/api/v1/script/{projectId}>, где projectId - id проекта

Ответ:  
список страниц с кодом скрипта загрузки для обновления ui клиентской части

