МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное
учреждение высшего профессионального образования
«Юго-Западный государственный университет»
(ЮЗГУ)
Кафедра информационных систем и технологий

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О.Г. Локтионова
«15» марта 2016 г.

ТЕХНОЛОГИЯ ИСПОЛЬЗОВАНИЯ СКРИПТОВ ПРИ ОБРАБОТКЕ ИНФОРМАЦИИ В DEDUCTOR STUDIO

Методические указания к лабораторной работе № 7
для студентов направления 09.03.02 и 09.03.03

Курск 2016

УДК 004
Составитель А.В. Ткаченко

Рецензент
Кандидат технических наук, доцент С.Ю. Сазонов

Технология использования скриптов при обработке информации в Deductor Studio: методические указания к лабораторной работе № 7 по дисциплине «Технологии обработки информации» / Юго-Зап. гос. ун-т; сост. А.В. Ткаченко. Курск, 2016. 11 с. Библиогр.: с. 11.

Приводится последовательность операций при подготовке информации для ее визуализации в Deductor Studio.

Методические указания соответствуют требованиям утвержденной рабочей программы дисциплины.

Предназначены для студентов, обучающихся по направлениям: 09.03.02 «Информационные системы и технологии» и 09.03.03 «Прикладная информатика».

Текст печатается в авторской редакции.

Подписано в печать . Формат 60х84 1/16.
Усл.печ. л. . Уч.-изд. л. . Тираж 100 экз. Заказ. Бесплатно.
Юго-Западный государственный университет.
305040, г. Курск, ул. 50 лет Октября, 94.

Цель: Освоить технологию использования скриптов при обработке информации в Deductor Studio

Введение

Скрипты предназначены для автоматизации процесса добавления в сценарий однотипных ветвей обработки. Это нужно в следующих случаях:

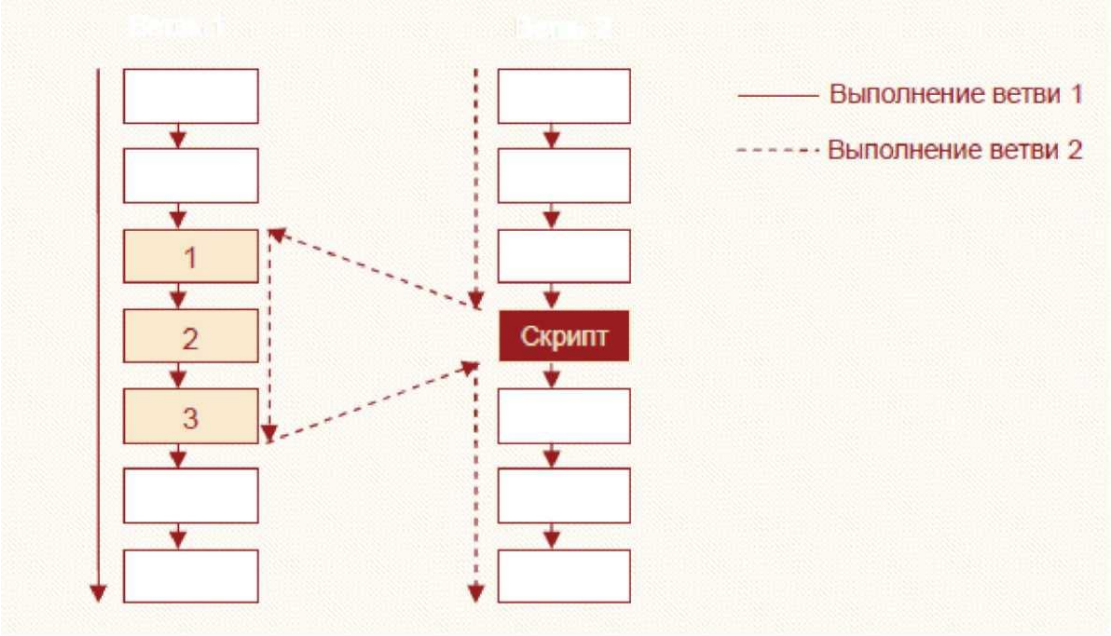
- требуется выполнить часть сценария (т.е. последовательность узлов) на другом наборе данных;
- требуется применить модель (дерево решений, нейронная сеть) на новых данных.

Если повторное выполнение части сценария можно обойти, используя копирование веток, то в случае применения аналитической модели к новым данным без обработчика **Скрипт** обойтись невозможно.

По сути скрипт представляет собой динамическую копию выбранного участка сценария. Скрипт является готовой частью сценария, и поэтому входящие в него узлы не могут быть изменены отдельно от исходной ветки сценария. Тем не менее, в скрипте отражаются все изменения, вносимые в ветку, на которую он ссылается, т.е. при переобучении или перенастройке узлов этой ветки все сделанные изменения будут внесены в работу скрипта.

Предположим, что после импорта данных из двух разных текстовых файлов требуется провести определенную предобработку (поменять названия столбцов, заменить данные, добавить несколько расчетный столбцов), а затем экспортировать полученные данные обратно. Для первой ветви (первого текстового файла) эти действия проводятся как обычно -последовательными шагами строится цепочка обработчиков.

Для второго же источника (второго файла) достаточно создать узел импорта, к которому присоединить узел Скрипт, основанный на уже построенной первой ветви. В этом скрипте будут выполнены точно такие же действия, как в оригинальной ветви. На выходе скрипта ставится узел экспорта, и вторая ветвь обработки готова к использованию. Эту идею иллюстрирует рисунок ниже.
Ветвь 1 Ветвь 2



На рисунке показана схема выполнения ветви со скриптом, включающего три узла из другой ветки сценария. Сначала (до узла со скриптом) последовательно выполняются узлы второй ветки. Затем осуществляется переход на начальный узел скрипта, находящийся в Ветви 1 .

Далее последовательно выполняются уже узлы первой ветки, пока не будет достигнут конечный узел скрипта. После этого осуществляется возврат к Ветви 2 на следующий после скрипта этап обработки, и выполнение продолжается. На ход выполнения первой ветви скрипт при этом не оказывает никакого влияния.

Особенность использования скрипта вместо копирования ветви заключается в том, что внесенные в главную ветвь изменения автоматически наследуются всеми скриптами, которые ссылаются на узлы главной ветви. В большинстве случаев это преимущество, однако, иногда при создании сценариев необходимо именно копирование узлов.

Аналогом скрипта является функция или процедура в языках программирования. Ветвь обработки строится один раз, а затем скриптами она тиражируется в другие места сценария.

Обработчик **Скрипт** находится в группе узлов **Прочее.**

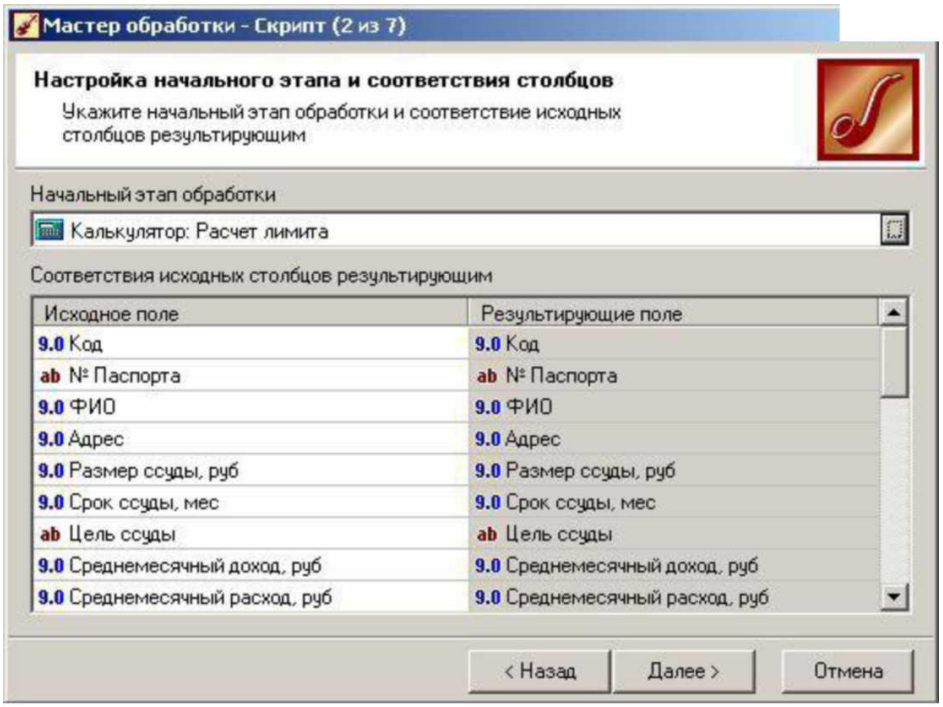
Создание и настройка скрипта

Настройка скрипта состоит из следующих шагов.

Шаг 1. Задание начального узла обработки и соответствия полей. Это осуществляется в окне **Настройка начального этапа и соответствия столбцов** мастера обработки узла **Скрипт.**

Для выбора начального узла нужно нажать кнопку^ , после чего на экране появится окно **Выбор узла.** В этом окне показано все дерево сценария. Кнопка **Ok** подтверждает выбор текущего узла в качестве начального узла скрипта, кнопка **Отмена** закрывает окно, не внося изменений.

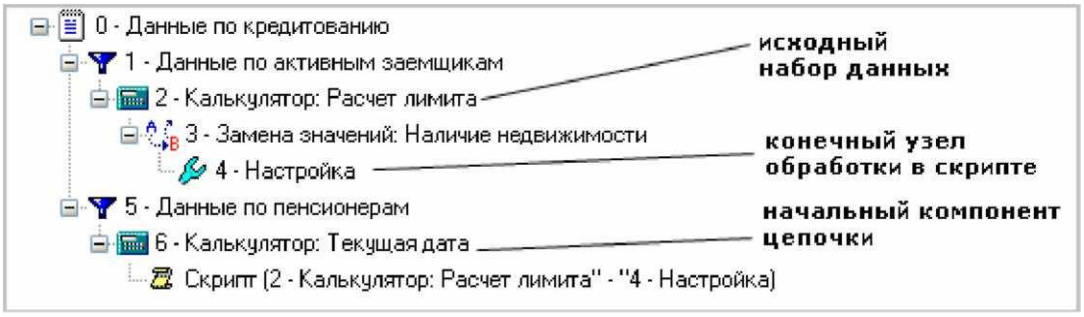
При выборе начального узла существуют следующее ограничение: начальным узлом может быть только узел обработчика (узел импорта или экспорта данных не может быть выбран).



В случае, когда исходный набор данный имеет меньшее число столбцов, чем начальный компонент цепочки, на экран будет выдано предупреждение: Количество столбцов начального компонента цепочки не должно быть больше чем количество столбцов исходного набора данных.

При этом в момент обработки скрипта будет принята попытка выполнить с имеющимся набором полей. Если какое-то из отсутствующих полей является критичным для любого узла, содержащегося в скрипте, то обработка будет остановлена с выдачей сообщения об ошибке.

Под исходным набором данных подразумевается тот набор данных, к которому применяется обработчик **Скрипт,** под начальным компонентом цепочки - набор данных, на который настраивается **Скрипт.**



После выбора начального узла следует задать соответствия столбцов исходного набора данных полям выбранного узла. В нижней части экрана находится таблица со списком полей исходного набора в левом столбце и полей выбранного узла - в правом. Для каждого поля начального узла надо задать поле-источник исходного набора. Для этого следует, щелкнув два

раза в левом столбце напротив имени нужного поля, выбрать из выпадающего списка имя столбца входного набора.

Мастер обработки узла Скрипт устроен так, что пытается автоматически сопоставить поля в источниках, совпадающие по названию и/или типу.

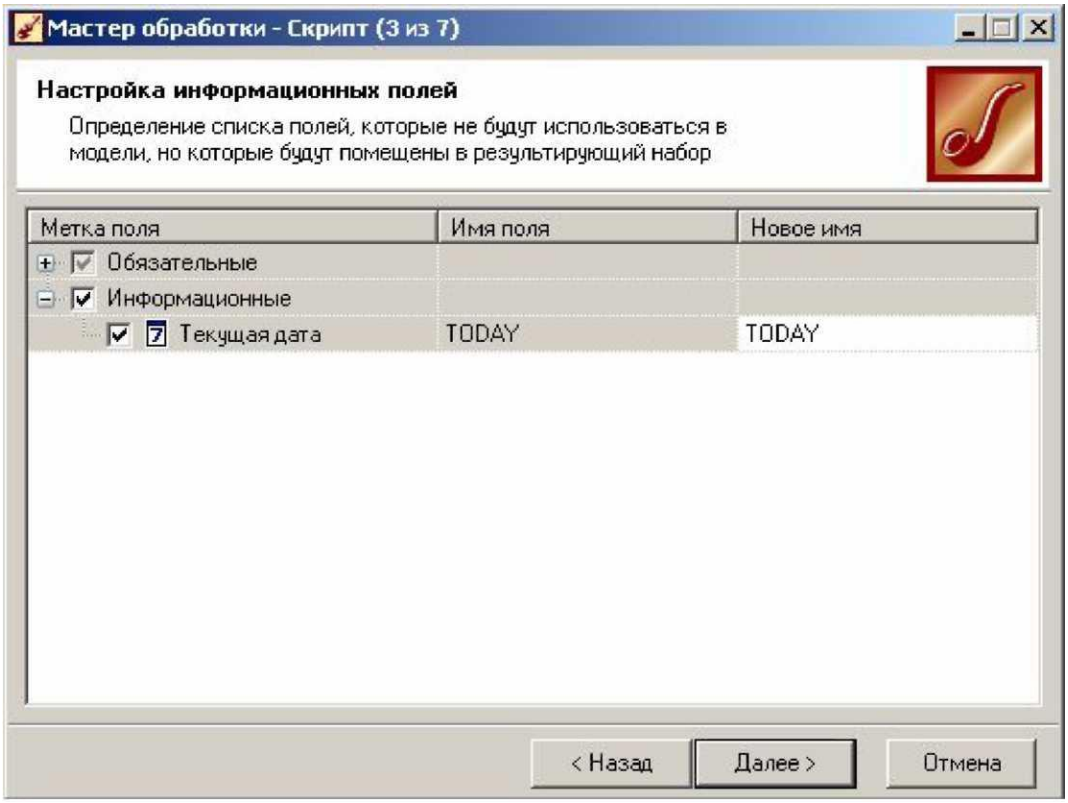
Настроить соответствия столбцов, которые имеют различный тип, невозможно.

Исключение имеется только для типов целый и вещественный, однако *рекомендуется всегда настраивать соответствие столбцов, имеющих одинаковый тип* (т.е. целый-целый, вещественный-вещественный).

Возможна ситуация, когда столбцам начального компонента цепочки нет сопоставимых столбцов в исходном наборе данных. В такой ситуации система выдаст следующее сообщение:

«Столбцам начального компонента цепочки нельзя сопоставить столбцы исходного набора данных». При этом в момент обработки скрипта будет принята попытка выполнить с имеющимся набором полей. Если какое-то из отсутствующих полей является критичным для любого узла содержащегося в скрипте, то обработка будет остановлена с выдачей сообщения об ошибке.

Шаг 2. Этап настройки информационных полей. Это необязательный шаг мастера, который появляется в том случае, когда исходный набор данных содержит большее количество полей, чем набор данных, являющийся начальным компонентом цепочки. Под информационными полями понимаются те поля, которые не будут использоваться в скрипте, но которые будут помещены в результирующий набор данных.



Шаг 3. Задание конечного узла обработки. Здесь существуют следующие правила.

Начальный и конечный узлы должны находится на одной ветви сценария, т.е. конечный узел должен являться потомком начального узла в дереве сценария.

Конечным узлом не может являться узел экспорта.

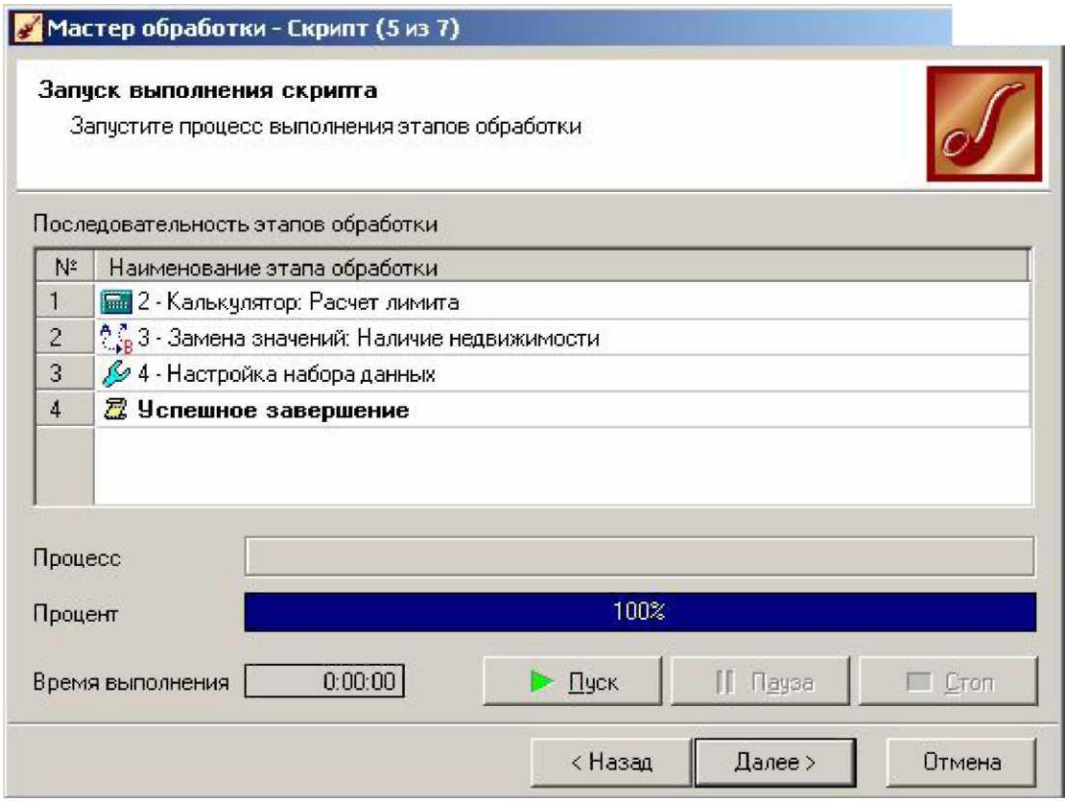
На число и типы промежуточных узлов не накладывается никаких ограничений, т.е. промежуточными узлами могут быть и скрипты.

Шаг 4. Запуск процесса обработки. На данном шаге запускается собственно процесс выполнения скрипта.

В секции **Последовательность этапов обработки** показан список всех узлов, входящих в скрипт. Узлы, которые еще не выполнялись, отображаются с серыми иконками, выполненные - с

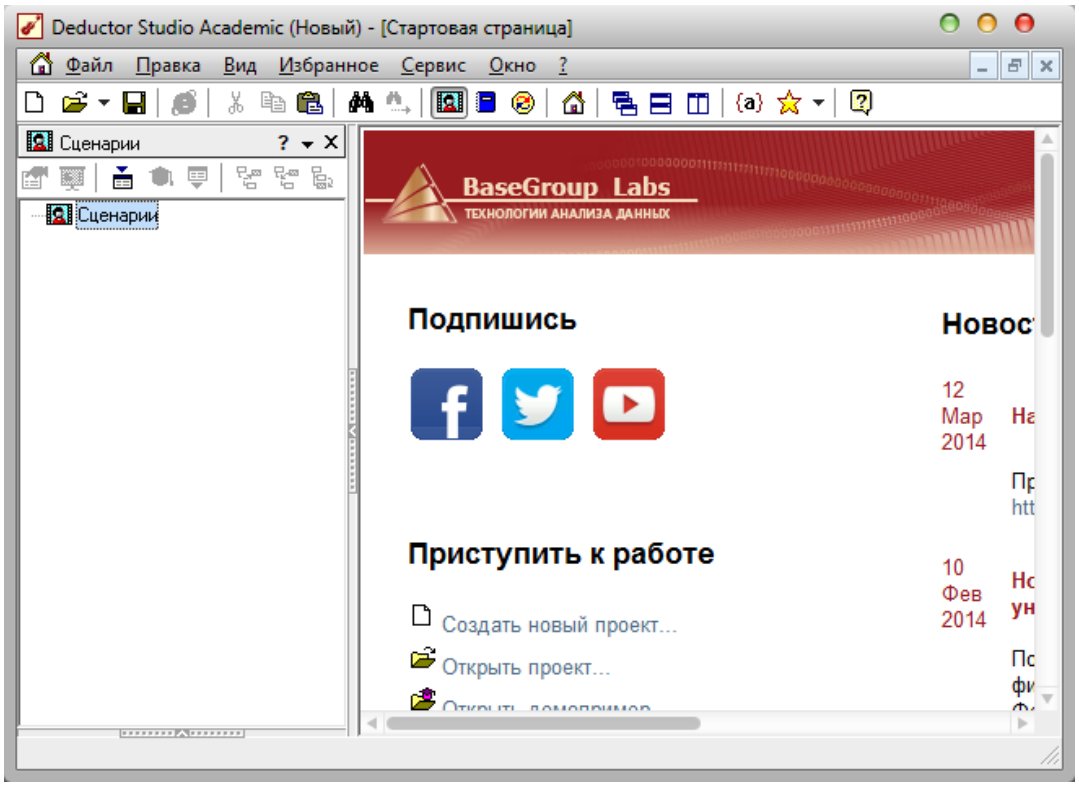
цветными. Имя текущего обрабатываемого узла отображается жирным шрифтом.

Если процесс обработки остановился, это сигнализирует о возможных проблемах. Остановка может произойти в случае несоответствия типов данных алгоритму обработки, наличия в обрабатываемых полях недопустимых значений и т.д. В этом случае возможно появление окна с сообщением об ошибке. Если обработка данных была завершена успешно, то в секции **Название процесса** появится сообщение «Успешное завершение».

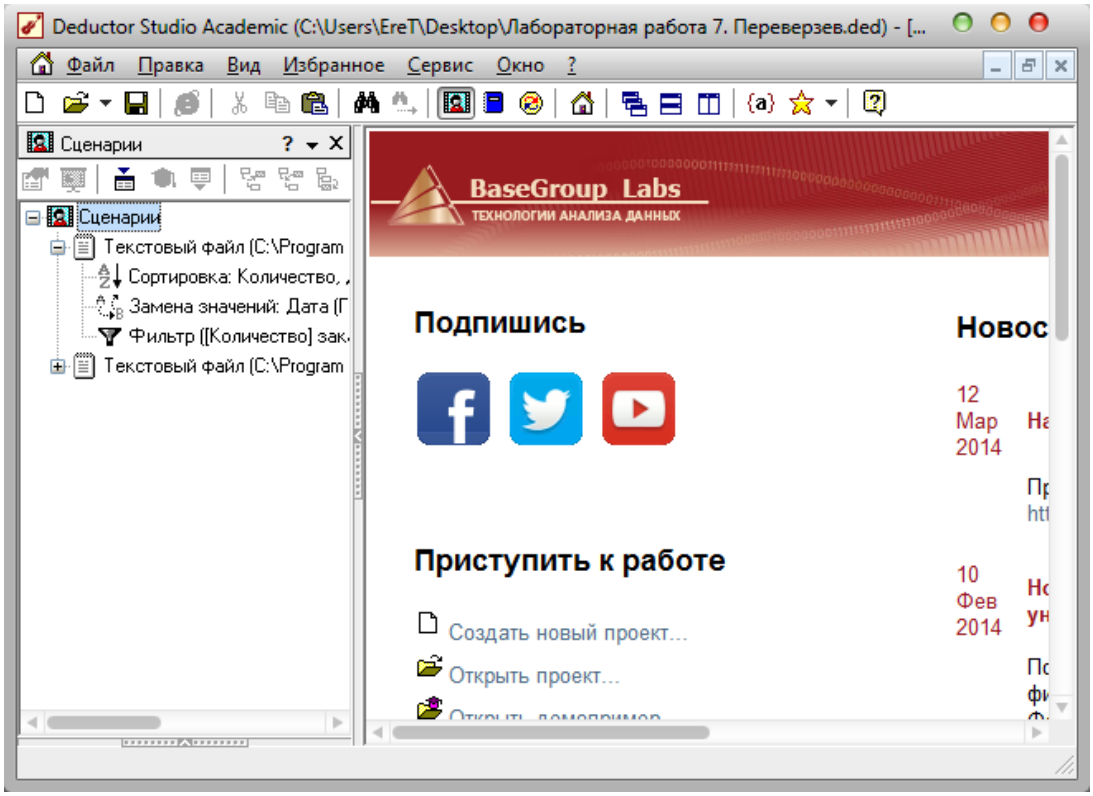


Практическая работа:

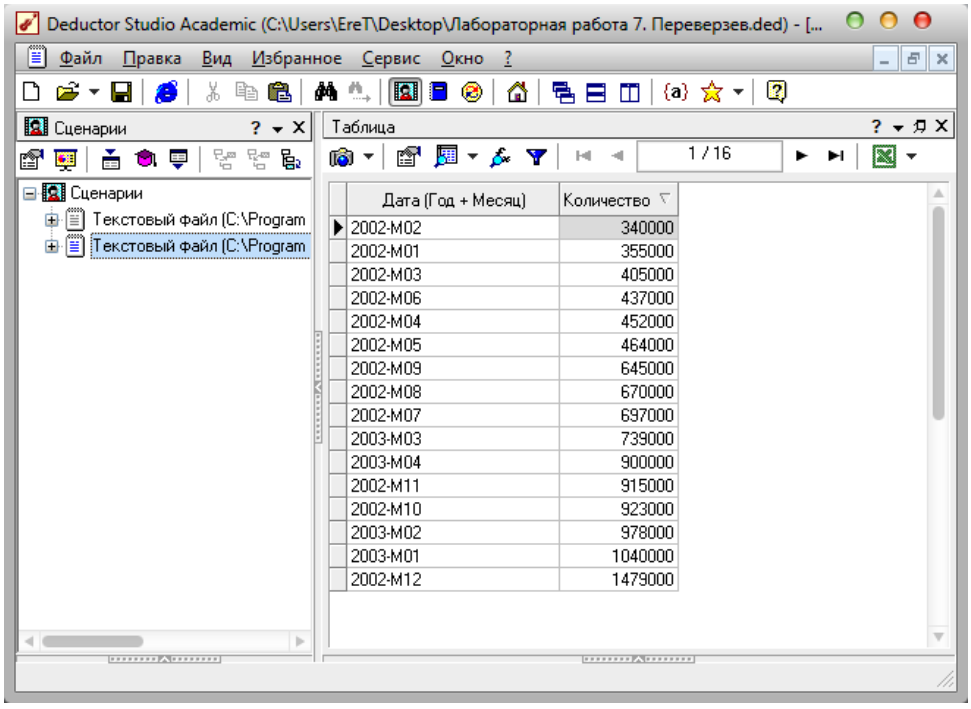
1. Создайте новый проект. Импортируйте в него текстовый файл **Trade.txt,** идущий в поставке Deductor (по умолчанию расположен в каталоге /Samples директории установки Deductor).



2. Добавьте после узла импорта 2-3 обработчика из изученных ранее.



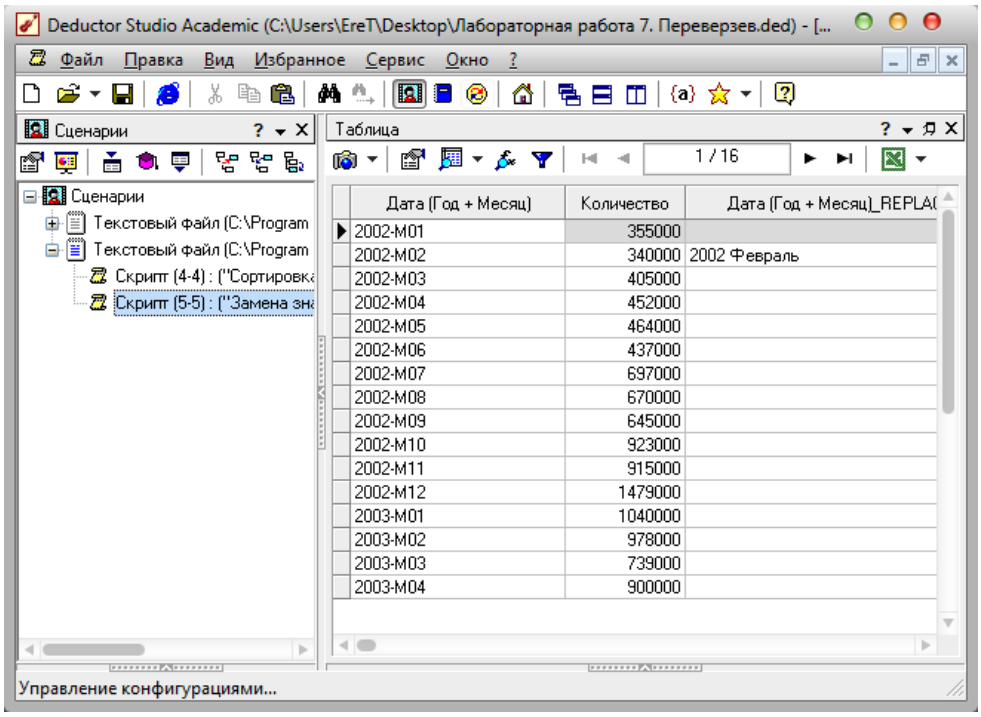
3. Импортируйте в него текстовый файл **TradeSales.txt,** (он расположен там же). Добавьте к нему поле Номер строки (используйте функцию калькулятора **RowNum()).**



Дата (Год + Месяц)	Количество
2002-M02	340000
2002-M01	355000
2002-M03	405000
2002-M06	437000
2002-M04	452000
2002-M05	464000
2002-M09	645000
2002-M08	670000
2002-M07	697000
2003-M03	739000
2003-M04	900000
2002-M11	915000
2002-M10	923000
2003-M02	978000
2003-M01	1040000
2002-M12	1479000

4. Добавьте к набору данных скрипт, выполняющий те же действия с набором данных, что и в п. 2.

5.



Вопросы для самопроверки:

1. Для чего предназначен обработчик **Скрипт?**
2. В каких случаях возникает необходимость добавить в сценарий скрипт?
3. Что такое исходный набор данных, начальный и конечный узел при настройке обработчика **Скрипт?**
4. Чем отличается копирование ветви от применения скрипта?
5. Можно ли настроить соответствия столбцов, которые имеют различный тип?
6. Какие ограничения накладываются на выбор конечного узла обработки в скрипте?

Библиографический список

1. Deductor Studio [Электронный ресурс]: www.basegroup.ru/download/deductor/.
2. Решения по построению хранилищ данных [Электронный ресурс]: http://ibarus.ru/solutions/dwh/?
3. Основные обработчики в Deductor Studio [Электронный ресурс]: http://deductor.org/Deductor_help_manual/deductor-help.html.