Министерство образования Республики Беларусь

Учреждение образования

Брестский государственный технический университет

Кафедра ИИТ

Лабораторная работа №8

За 7 семестр

По дисциплине «Компьютерное моделирование и анализ данных»

Выполнил:

студент 4 курса

Группы ПО-4(1)

Ковальчук В.В.

Проверил:

Чичурин А.В.

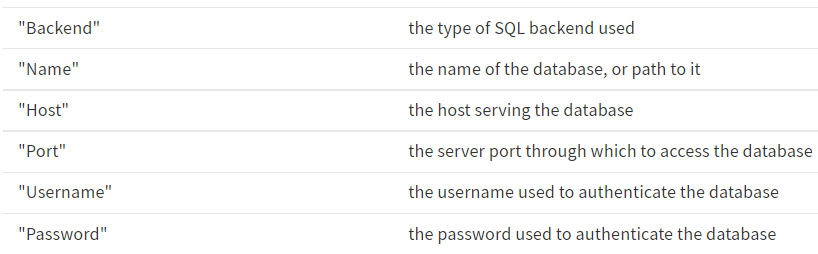
Брест 2022

**Лабораторная работа №8**

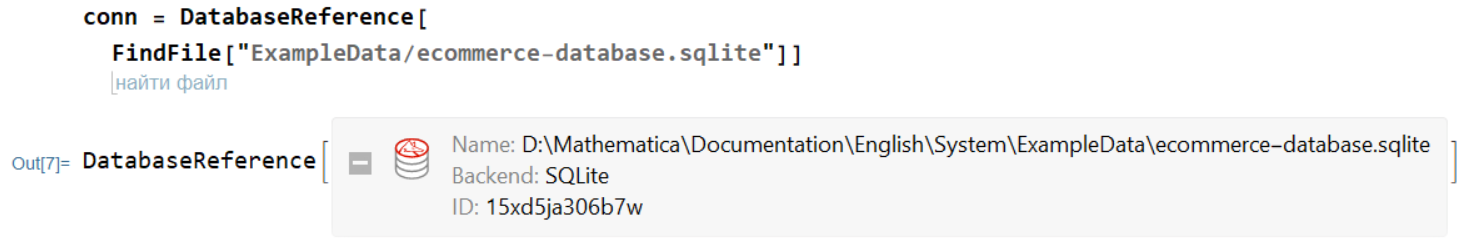
**Работа с базами данных**

В Wolfram, для огромного количества случаев, когда нужно запрашивать базу данных, можно взаимодействовать с базой данных через высокоуровневые запросы Entity Framework, построенные с использованием языка функциональных запросов Entity framework. Затем платформа отвечает за перевод запроса на соответствующий диалект SQL, выполнение его на стороне базы данных и возврат результатов обратно пользователю в соответствующем формате. Entity Framework для Wolfram в контексте реляционных баз данных во многом похож на то, что представляют собой объектно-реляционные отображения (ORM) для других языков. Несколько упрощая, сущности сопоставляются с отдельными строками таблиц базы данных, классы сущностей сопоставляются с таблицами базы данных, а свойства сущностей сопоставляются со столбцами базы данных.

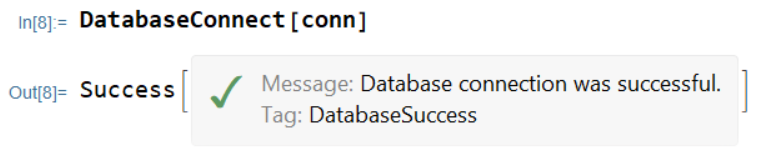
Для подключения к базе данных буду использовать DatabaseReference[assoc] Типичными элементами assoc являются



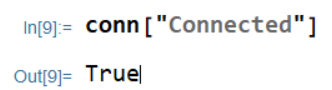
Подключимся к базе данных



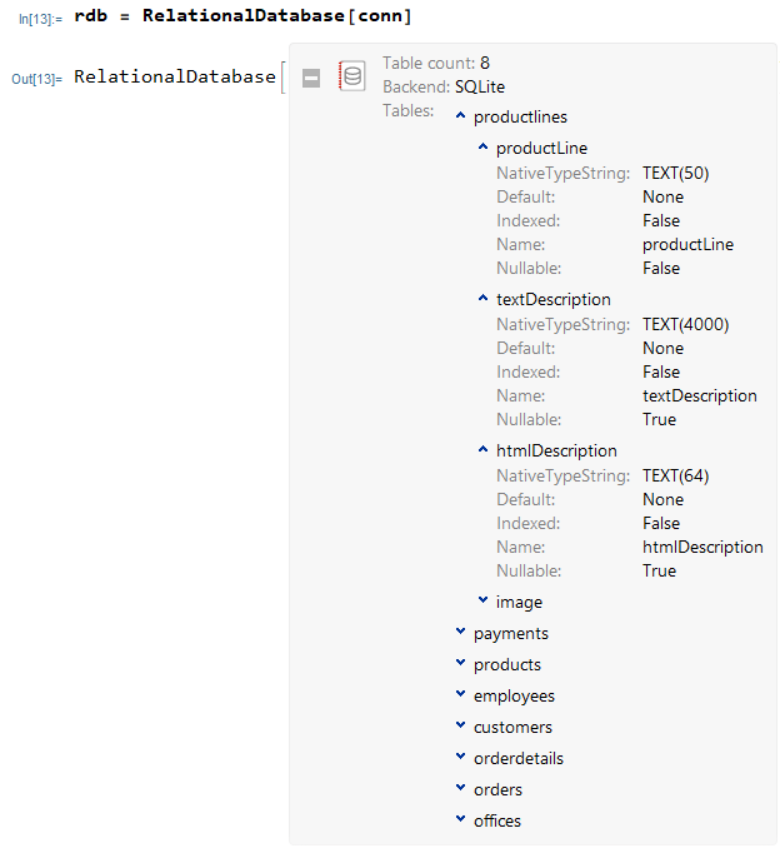
При помощи DatabaseConnect[conn] проверим наше соединение с базой данных. В нашем случае функция вернула сообщение Success, а значит подключение прошло успешно.



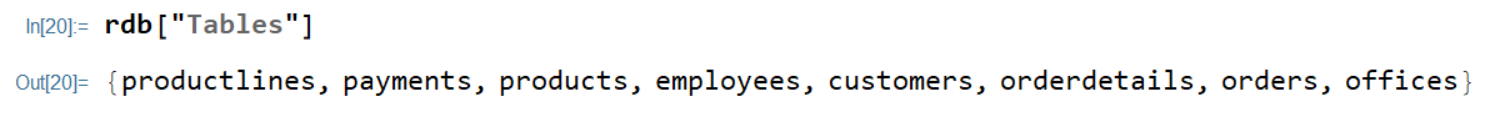
Проверим, действительно ли прошло соединение с базой данных.



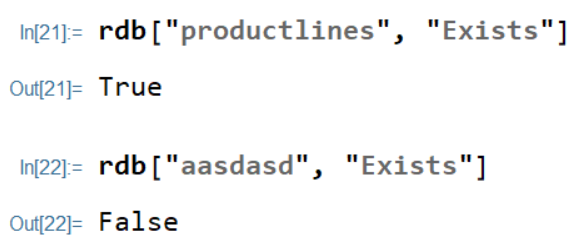
Создадим с помощью функции RelationalDatabase объект RelationalDatabase. Он содержит полную структуру базы данных, на которую мы ссылаемся (таблицы, колонки, ограничения и так далее) и он может быть использован как для визуального осмотра, так и для программного вытягивания этой информации, которая нам интересна.



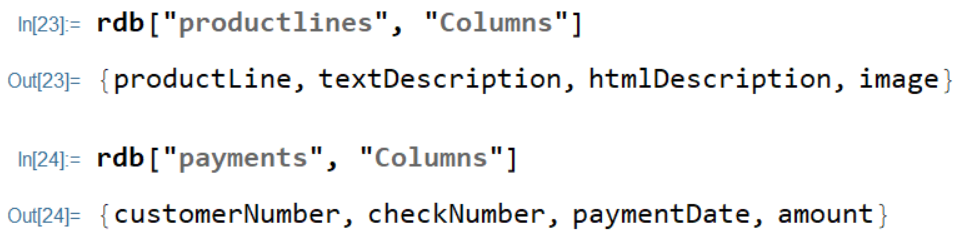
Попробуем получить информацию о таблицах нашей базы данных другим способом, а именно указав параметр “Tables”. Результат вернет нам список наших таблиц.



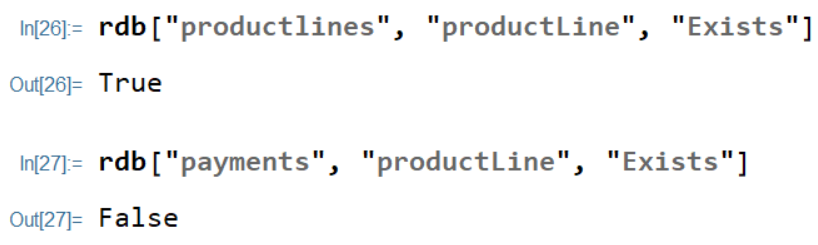
Если необходимо проверить факт существования той или иной таблицы, мы можем первым параметром указать название таблицы, которую мы и хотим проверить на существование, а вторым параметром слово Exists. Результатом будет булевое значение.



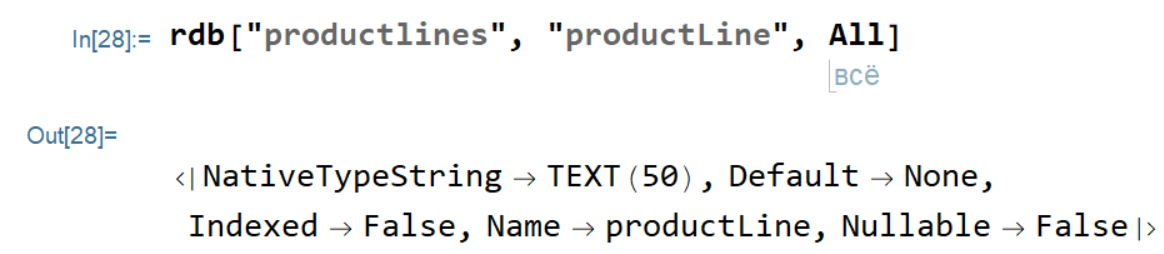
Также как названия таблиц мы можем получить и названия столбцов отдельной таблицы. Для этого первым параметром укажем название таблицы, названия колонок которой нас интресуют, а вторым параметром слово “Columns”



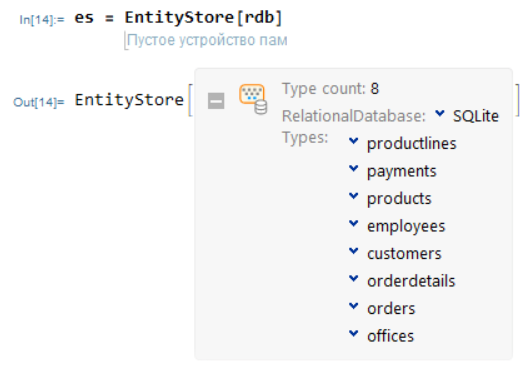
Аналогично проверке на наличие таблицы, можно сделать проверку на наличие определенного столбца.



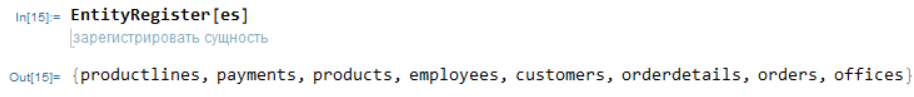
Также можно получить информацию о свойствах любого из полей таблицы. Для этого используем All после указания поля.



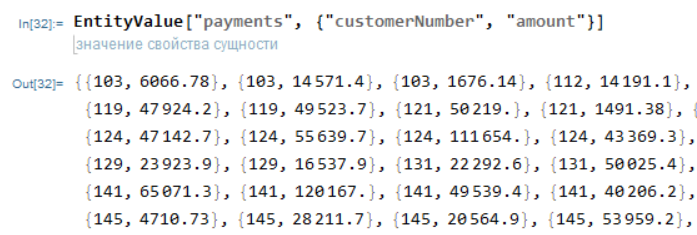
Создадим поддерживаемой базой данных объект EntityStore из объекта RelationalDatabase. В нем будут находится все сущности, которые находятся в нужной нам базе данных.



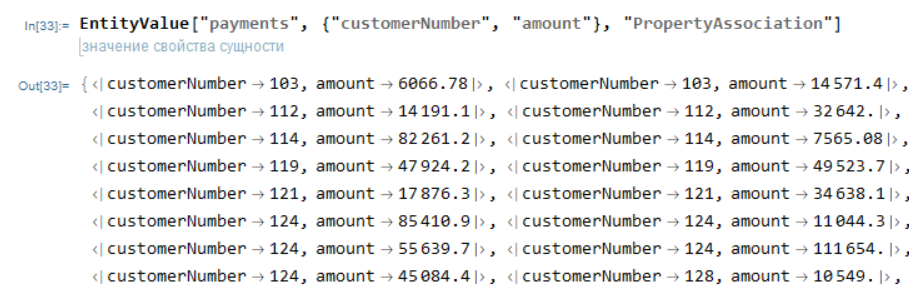
Затем зарегистрируем сущности в объекте EntityStore, чтобы можно было получить к ним доступ напрямую через Entity.



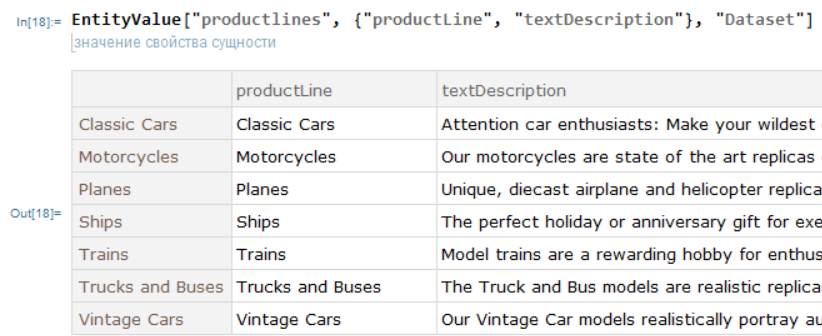
Теперь можно работать с нашей базой данных. Для начала попробуем извлечь информацию о customerNumber и amount из таблицы payments. Для этого используется функция EntityValue. Он используется для выполнения запроса и возвращает результат в различных формах. Первым параметром укажем сущность (таблицу) payments, которая недавно была зарегистрирована, а вторым параметром – список извлекаемых свойств (полей) сущности.



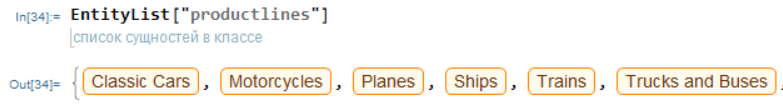
Добавим параметр “PropertyAssociation”, чтобы к возвращаемым значениям еще добавлялись названия свойств.



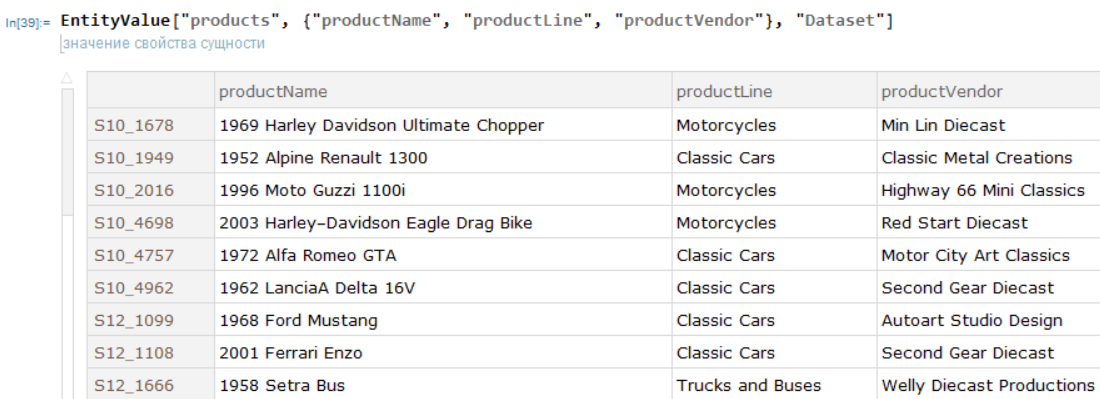
Получим результат в виде таблицы значений прописав параметр “Dataset”.

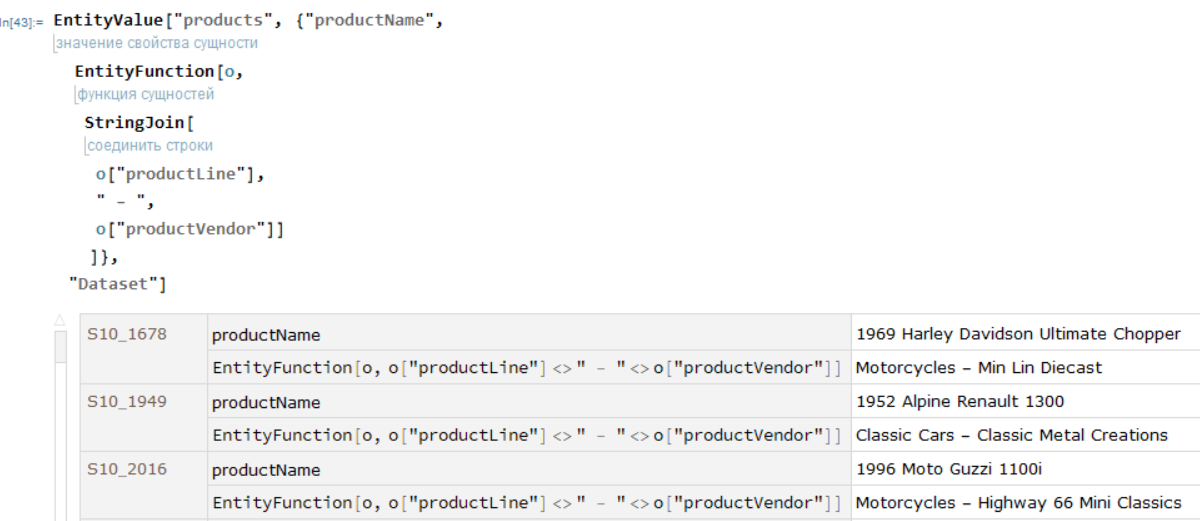


Если есть необходимость в получении списка сущностей, то для этого можно использовать EntityList.

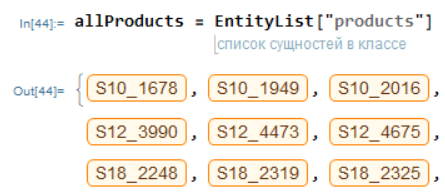


Кроме извлечения существующих свойств, можно также извлекать вычисляемые свойства – те свойства, которые вычисляются на лету. Такие свойства должны быть выражены через функцию EntityFunction. Пример: если вывести данные таблицы Companies, то получатся Название, Страна и Адрес. Логичнее было бы объединить Страну и Адрес в одно поле.



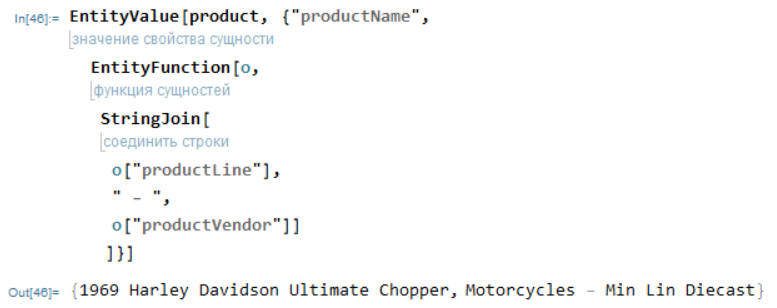


Entity Framework позволяет работать с одиночными сущностями. Так, например, получим из списка сущностей таблицы Companies первую сущность. Для этого сначала получим все наши сущности через EntityList, а затем получим первый элемент через функцию First, куда передадим наш список сущностей.





Затем можно вывести детальную информацию о сущности.



Если по какой-либо причине ранее зарегистрированные сущности нам будут не нужно, то их можно просто-напросто удалить. Для этого используется функция EntityUnregister.



После завершения работы с базой данных, желательно разорвать соединение с ней. Функция DatabaseDisconnect используется для это, а в качестве параметра передается ссылка соединения с ней, созданная в самом начале.

