

## Лабораторная работа 2

### ч.1 Проектирование БД

**Цель занятия.** Познакомиться с этапами проектирования баз данных, программными средствами проектирования. Научиться разрабатывать логическую и физическую модель базы данных.

#### 1. Основные понятия модели данных

Проектирование базы данных начинают с анализа предметной области и выявления требований к ней будущих пользователей. Объединяя частные представления о содержимом базы данных отдельных пользователей, вначале создается обобщенное описание создаваемой базы данных, называемое концептуальной схемой. Данный этап еще называют моделированием или логическим проектированием базы данных.

Логическая модель данных является универсальной и никак не связана с конкретной реализацией СУБД. Поэтому данное представление будет понятно даже для неспециалистов (заказчиков информационной системы).

Цель моделирования данных состоит в обеспечении разработчика ИС концептуальной схемой базы данных в форме одной модели или нескольких локальных моделей, которые относительно легко могут быть отображены в любую систему баз данных.

Наиболее распространенным средством моделирования данных являются диаграммы "сущность-связь" (ERD). С их помощью определяются важные для предметной области объекты (сущности), их свойства (атрибуты) и отношения друг с другом (связи). ERD непосредственно используются для проектирования реляционных баз данных.

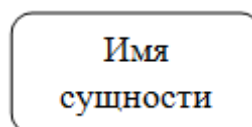
Первый шаг моделирования - извлечение информации на основе анализа предметной области и выделение сущностей.

Рассмотрим в качестве примера проектирование БД, содержащей информацию о сотрудниках предприятия. Вначале введем основные понятия.

Вначале введем понятие сущности.

**Сущность (Entity)** - реальный либо воображаемый объект, имеющий существенное значение для рассматриваемой предметной области, информацию о котором необходимо сохранить в БД.

Сущность отображается прямоугольником, внутри которого записывается имя сущности (рис.1)



*Рис.1 Отображение сущности*

Каждая сущность должна обладать следующими свойствами:

- иметь уникальное имя (задается именем существительным в именительном падеже в единственном числе), например «Сотрудник»;
- содержать один или несколько атрибутов, описывающими наиболее важные свойства объекта предметной области (например: «фамилия», «должность»);
- обладать одним или несколькими атрибутами, которые однозначно идентифицируют каждый экземпляр сущности (выделяют среди других экземпляров данной сущности) – первичным ключом, например: атрибут «табельный номер сотрудника»;
- иметь связи с другими сущностями модели.

Перейдем к рассматриваемой БД. В данной системе важны отделы и работающие в них сотрудники. Поэтому можно выделить 2 сущности: «Отдел», «Сотрудник».

Следующим важным понятием являются связи между сущностями.

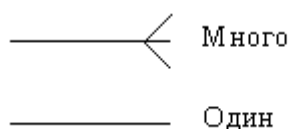
**Связь (Relationship)** - поименованная ассоциация между двумя сущностями, значимая для рассматриваемой предметной области. В этой связи-ассоциации каждый экземпляр одной сущности, называемой родительской, ассоциирован с произвольным (в том числе нулевым) количеством экземпляров второй сущности, называемой сущностью-потомком, а каждый экземпляр сущности-потомка ассоциирован в точности с одним экземпляром сущности-родителя. Таким образом, экземпляр сущности-потомка может существовать только при существовании сущности родителя.

Связи можно дать имя, выражаемое грамматическим оборотом глагола и помещаемое возле линии связи. Имя связи формируется как с позиции сущности-родителя, так и с позиции сущности-потомка.

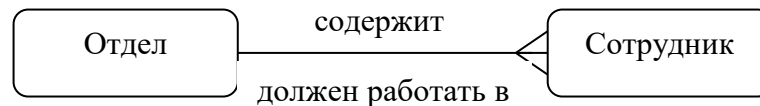
Например, связь отдела с сотрудником может быть выражена следующим образом:

- отдел (*имя родителя*) может содержать (*имя связи*) одного или более (*степень связи*) сотрудников (*имя потомка*);
- сотрудник должен работать только в одном отделе.

Степень связи (количество экземпляров сущности, которым она соответствует) графически изображаются одной или многими линиями на концах связи.



Связь сущностей Отдел и Сотрудник графически будет выражены следующим образом (рис. 2).



**Рис. 2. Связь сущностей**

Наконец еще одним важным понятием является понятие атрибута.

**Атрибут** - любая характеристика сущности, значимая для рассматриваемой предметной. Например, в сущности «Отдел» атрибутами могут быть «Код отдела», «Наименование отдела».

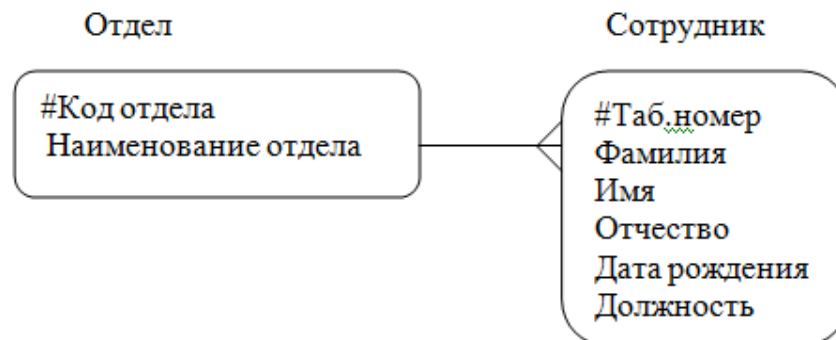
Атрибут может быть либо описательным (неключевым), либо входит в состав уникального идентификатора (первичного ключа). Атрибут может быть либо обязательным к заполнению, либо необязательным. Обязательность означает, что атрибут не может принимать неопределенных значений (null).

**Уникальный идентификатор (ключ)** - это атрибут или совокупность атрибутов, предназначенная для уникальной идентификации каждого экземпляра сущности. Например, в сущности «Отдел» таким атрибутом может быть «Код отдела».

Каждый атрибут обозначается уникальным именем, выражаемым грамматическим оборотом существительного, описывающим представляемую атрибутом характеристику (например, «Фамилия»). Атрибуты изображаются в виде списка имен внутри блока сущности, причем каждый атрибут занимает отдельную строку. Атрибуты, определяющие первичный ключ, размещаются наверху списка и выделяются знаком "#".

Каждая сущность должна обладать хотя бы одним возможным ключом. Возможный ключ сущности - это один или несколько атрибутов, чьи значения однозначно определяют каждый экземпляр сущности. При существовании нескольких возможных ключей один из них обозначается в качестве первичного ключа, а остальные - как альтернативные ключи.

Сущности с указанием атрибутов приведены на рис. 3.

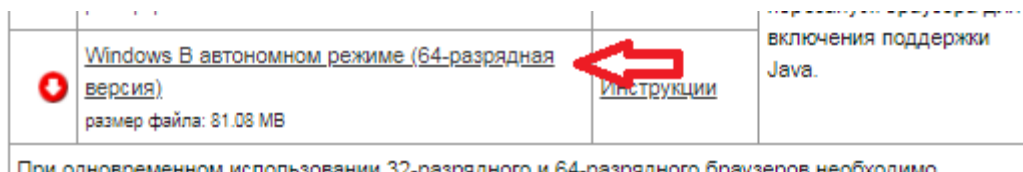


**Рис.3 Атрибуты сущностей**

## 2. Программные средства моделирования данных и проектирования БД

Для автоматизации процесса проектирования используются CASE (Computer-Aided Software/System Engineering) технологии и инструментальные средства, позволяющие максимально упростить все этапы разработки базы данных. Удобно использовать свободно распространяемые программные средства. К таким средствам относится SQL Power Architect Community Edition. Среда SQL Power Architect предназначена для визуального проектирования модели данных для распространенных СУБД и организации взаимодействия с серверами БД. Данная программная среда выбрана еще потому, что она имеет драйверы подключения к СУБД PostgreSQL. И непосредственно из среды можно по модели создать схему БД.

Программа для использования требует установки платформы Java, точнее JRE (Java Runtime Environment) версии 8 или выше. Скачать пакет можно по ссылке <https://www.java.com/ru/download/manual.jsp>, выбрав нужную операционную систему.



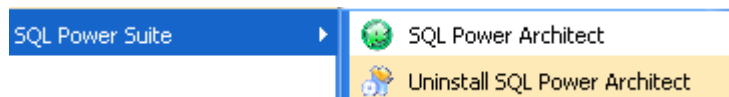
Затем необходимо запустить скачанный пакет на установку.

Для установки программы SQL Power Architect необходимо скачать установочный пакет (SQL-Power-Architect-Setup-Windows-1.0.9.exe) по адресу:  
<https://bestofbi.com/architect-download/>

Чтобы затем не скачивать драйверы для соединения с распространенными серверами БД, лучше скачать установочный пакет, содержащий эти драйверы, (SQL-Power-Architect-Setup-Windows-jdbc-1.0.8.exe) по адресу:

<http://www.bestofbi.com/page/deprecated-downloads#architect>

После скачивания необходимо запустить пакет на установку. После установки в меню *Пуск* операционной системы Windows появится новая папка.



Для установки русскоязычного интерфейса программы необходимо изменить параметр языка, выбрав пункт меню:

File – User Preferences

и изменить параметр Default Language на Russian.

### 3. Разработка модели данных в среде SQL Power Architect

В некоторых программных средствах четко разделены этапы логического и физического проектирования БД.

**Логическое проектирование БД** – это создание *непротиворечивого и эффективного* отображения реальных объектов предметной области в *абстрактные объекты* модели данных. Объекты модели, представляемые на логическом уровне, называются сущностями и атрибутами. Атрибуты имеют универсальные обобщенные типы данных: символьный, числовой и временной. Логическая модель данных является универсальной и не связана с конкретной реализацией СУБД.

**Физическое проектирование БД** - это представление абстрактных объектов логической модели в *структуры данных конкретной СУБД* для обеспечения эффективности выполнения запросов. Сущности называются таблицами, а атрибуты - колонками. Колонкам назначается более конкретный тип данных, из возможных типов выбранной СУБД.

В программной среде SQL Power Architect такое разделение осуществляется только на уровне имен:

Logical Names (на этом уровне будем использовать имена в кириллице);

Physical Names (будем использовать имена на английском языке).

После открытия программы появится основное окно (рис.4). Для разработки модели необходимо создать новый проект. Для сохранения проекта под выбранным именем используется пункт меню:

Файл – Сохранить Проект как

Можно настроить свойства проекта, вызвав пункт меню:

Файл – Параметры проекта

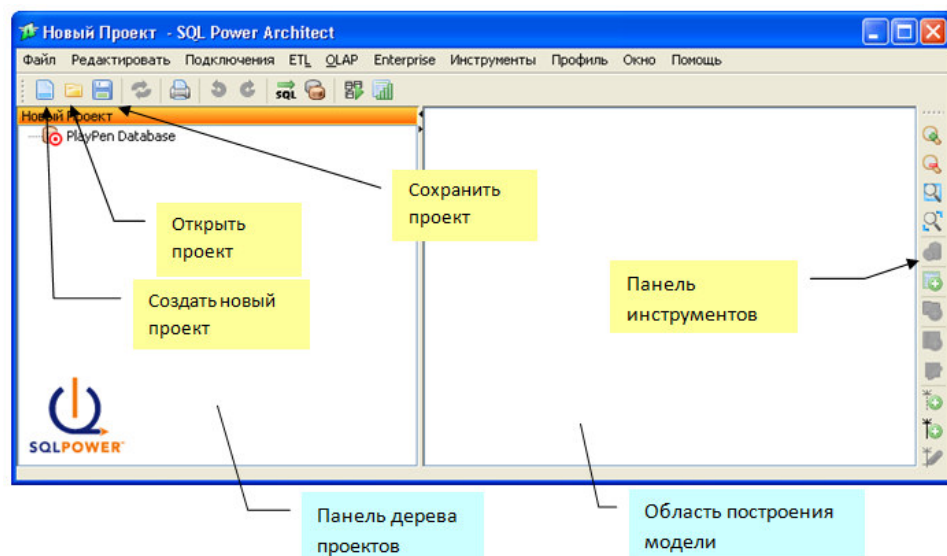
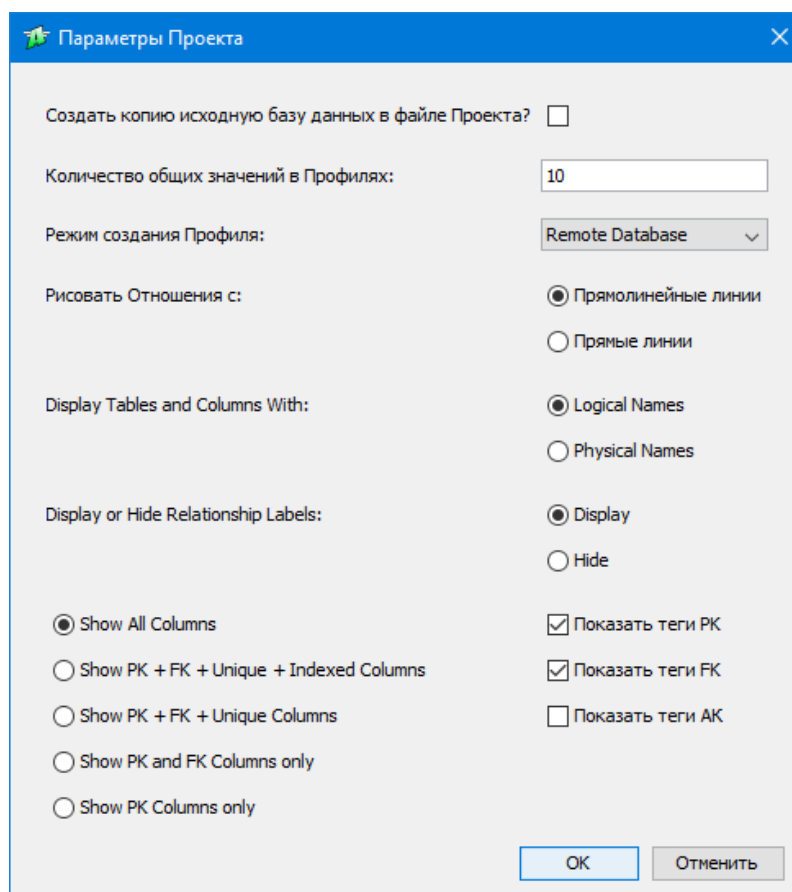


Рис.4 Основное окно программы








В окне настройки свойств проекта (рис.5) можно задать:

- 1) Вид отображаемой модели (переключатель Display Tables and Columns):  
Logical Names - логической  
Physical Names - физической
- 2) Переключатель Show All Columns – отображение всех колонок
- 3) Отображение спецификаций ключей (флажки). Например флаг *Показать теги РК* позволяет отобразить/скрыть пометку первичного ключа.



**Рис. 5** Окно настройки свойств проекта

Для создания модели данных используются кнопки панели инструментов:

- |   |                                   |
|---|-----------------------------------|
|  | добавить сущность (таблицу)       |
|  | изменить свойства таблицы         |
|  | добавить новый индекс             |
|  | добавить атрибут (колодку)        |
|  | добавить неидентифицирующую связь |
|  | добавить идентифицирующую связь   |
|  | изменить свойства связи           |

**В качестве примера** рассмотрим предметную область, связанную с поставками сырья. Рассмотрим процесс проектирования базы данных (БД) на предметной области поставок сырья поставщиками.

***Предприятие закупает сырье для производства продукции. Сырье закупается у поставщиков. Сырье поставляется по накладным.***

В рассматриваемой предметной области можно выделить два типа объектов: поставщики, выполненные поставки. Для упрощения модели не будем выделять еще один объект – сырье. Каждый экземпляр объекта содержит одинаковый набор атрибутов (характеристик), из которых следует выбрать только те, которые необходимы для решения поставленной задачи (в нашем случае – учет поставок сырья). В таблице1 приведено описание объектов предметной области с указанием выбранных свойств этих объектов.

**Таблица 1 – Описание сущности предметной области Учет поставок сырья**

<b>Сущность</b>	<b>Имя атрибута</b>
<b>Поставщик</b>	Код поставщика
	Наименование поставщика
	Адрес поставщика
	Тип поставщика (варианты: физическое или юридическое лицо)
<b>Книга поставок</b>	Дата поставки
	Номер документа
	Наименование сырья
	Поставщик
	Количество сырья
	Цена за 1 ед

### **3.1 Разработка логической модели данных**

Разработка модели данных включает несколько шагов.

#### **Шаг 1. Создание проекта**

Создадим новый проект и сохраним его под именем Поставки.

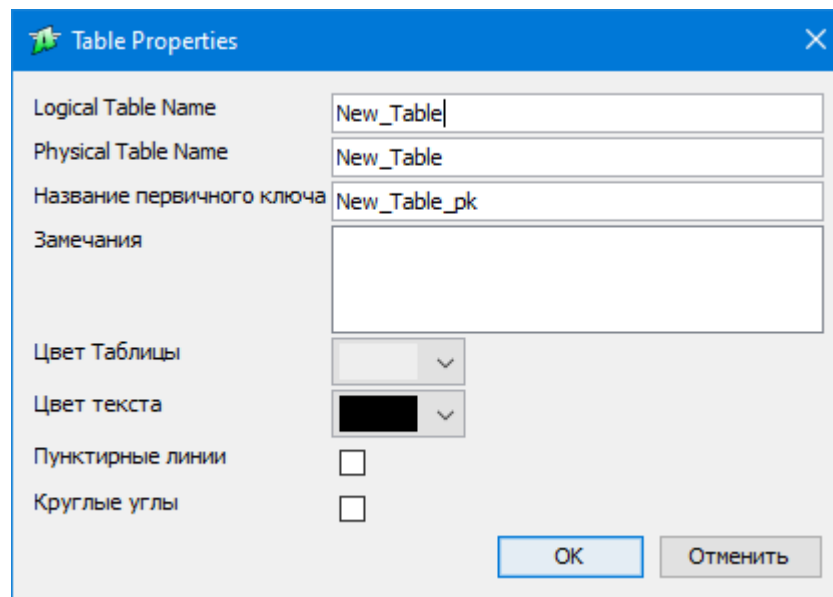
#### **Шаг 2. Создание сущностей (отношений)**

В соответствии с описанием предметной области можно выделить следующие сущности: Поставщик, Книга поставок (описывает партию поставленного сырья).

Для добавления новой сущности необходимо выбрать мышью иконку «Добавить новую таблицу» и следом щелкнуть в любом месте рабочей области построения модели. Откроется окно «Свойства таблицы» (рис. 6).

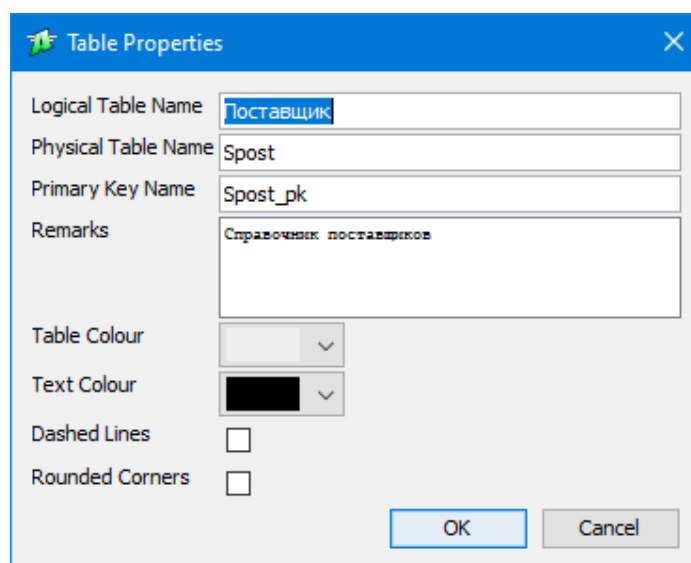
В окне необходимо задать:

- имя сущности логической модели (Logical Table Name)
- имя таблицы физической модели (Physical Table Name)
- имя ограничения первичного ключа
- комментарий (замечание) к названию таблицы (будет сохранен в виде комментария в БД)

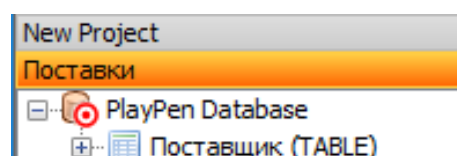


**Рис.6 Окно создания новой таблицы**

На рис. 7 показано заполненное окно свойств для сущности *Поставщик*. После нажатия кнопки «ОК» новая таблица появится в дереве проекта (рис.8).



**Рис.7 Окно создания новой сущности Поставщик**



**Рис. 8 Новая сущность в дереве проекта**

Добавим в сущность *Поставщик* атрибуты: код, наименование, адрес, тип. Для добавления атрибутов сущности необходимо выделить требуемую сущность и на панели инструментов щелкнуть по кнопке «Вставить столбец».



Другой способ - в контекстном меню можно выбрать пункт «Новый столбец». Откроется окно редактирования свойств столбца (рис. 9).

Свойства Столбца New Column

Source for ETL Mapping  
Ничего не определено

Logical Name  
New Column

Physical Name

☐ В Первичном Ключе

Тип  
VARCHAR

Точность ☐ 0 Масштаб ☐ 0

Позволить Nulls  
☐ No

Автоувеличение  
☐ No

Значение по умолчанию  
☐

Название последовательности  
Book\_New Column\_seq

Заметки

OK Отменить

*Рис.9 Окно редактирования свойств столбца*

В окне необходимо задать (не все элементы обязательны):

- имя атрибута логической модели (Logical Name);
- имя колонки физической модели (Physical Name);
- для атрибута первичного ключа включить флажок «В первичном ключе»;
- тип данных;
- для некоторых типов задать длину поля (Точность) и количество дробных разрядов (Масштаб);
- обязательность заполнения поля (Позволить Null);
- признак автоматического увеличения для целочисленных полей первичного ключа; при таком выборе необходимо указать имя объекта автоувеличения в БД (последовательность – sequence);
- значение по умолчанию (присваивается колонке, если не задано явного значения);
- заметки (необходимы для формирования комментария к колонке в БД).

На рис. 10 показано окно для создания атрибута первичного ключа сущности *Поставщик*.

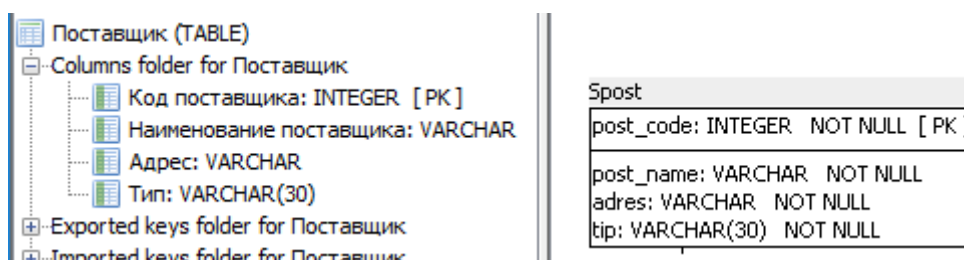
*Рис.10 Создание атрибута первичного ключа сущности Поставщик*

На рис 11 показана созданная сущность *Поставщик* в рабочей области модели и в панели с деревом проекта. Для атрибутов-столбцов выбраны следующие типы данных:

*Integer* - целочисленный тип;

*Varchar* - символьный тип, число в скобках указывает максимально возможное количество символов;

*Numeric* - числовой вещественный тип. Числа в скобках означают: 10 - общее количество позиций на все число, 2 - количество разрядов в дробной части числа.



*Рис.11 Сущность Поставщик*

Создадим сущность *Книга поставок* с атрибутами: Дата поставки, Номер накладной, Сырье, Цена, Количество (рис. 12). Атрибут первичного ключа добавим позже. Информацию о поставщике добавим при установке связи.

Книга поставок	
Дата поставки:	DATE NOT NULL
Номер накладной:	INTEGER NOT NULL
Сырье:	VARCHAR NOT NULL
Цена:	NUMERIC(10, 2) NOT NULL
Количество:	NUMERIC(7, 1) NOT NULL

**Рис.12 Начальный вид сущности Книга поставок**

Выберем в сущности *Книга поставок* первичный ключ. В качестве первичного ключа нельзя выбрать атрибут *Номер накладной* (номера могут повторяться из года в год) или атрибут *Дата поставки* (в один день может быть несколько поставок). Из существующих атрибутов в первичный ключ можно включить атрибуты *Номер накладной* и *Дата поставки*. Но это будет составной ключ. Желательно использовать простой первичный ключ, состоящий из одного атрибута. В этом случае добавляется суррогатный атрибут первичного ключа (не существующий в реальном объекте предметной области). Такой атрибут обычно называют идентификатором, т.к. его назначение идентифицировать экземпляры сущности. Имя таких атрибутов начинается со слова Идентификатор, к которому добавляется имя сущности. В сущности *Книга поставок* имя атрибута первичного ключа: *Идентификатор поставки*. Значения таким атрибутам присваиваются автоматически по принципу автоувеличения на 1 от значения ключа предыдущей записи таблицы.

На рис. 13 показана настройка первичного суррогатного ключа в сущности *Книга поставки*. Обратите внимание на установку флага *Автоувеличение (Auto Increment)* в значения *Yes*.

### **Шаг 3. Установка связи (отношения) между сущностями**

В модели данных сущности связаны так же, как объекты в реальной действительности. Например, сущность *Книга поставок* связана с сущностью *Поставщик*, т.е. каждая поставка выполнена некоторым поставщиком. В связи *Поставщик-Книга поставок* сущность *Поставщик* является родительской, а *Книга поставок* - подчиненной. Связь с *Поставщиком* в сущности *Книга поставок* устанавливается через атрибут, который будет указывать на экземпляр записи в сущности *Поставщик*. В качестве такого атрибута всегда выбирается первичный ключ родительской таблицы, т.к. только он однозначно идентифицирует конкретного *Поставщика*. При установке связи ключевой атрибут будет мигрировать (копироваться) в дочернюю сущность.

**Column Properties of Идентификатор поставки**

Source for ETL Mapping: None Specified

Logical Name: Идентификатор поставки

Physical Name: book\_id

☒ In Primary Key

Type: INTEGER

Precision: 0 Scale: 0

☐ Allows Nulls: No

☒ Auto Increment: Yes

☐ Default Value:

Sequence Name (Only applies to target platforms that use sequences): book\_seq

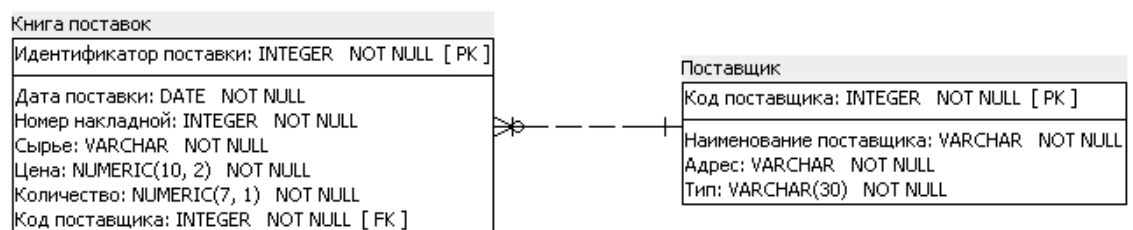
Remarks:

OK Cancel

**Рис.13 Создание атрибута суррогатного первичного ключа сущности Книга поставки**

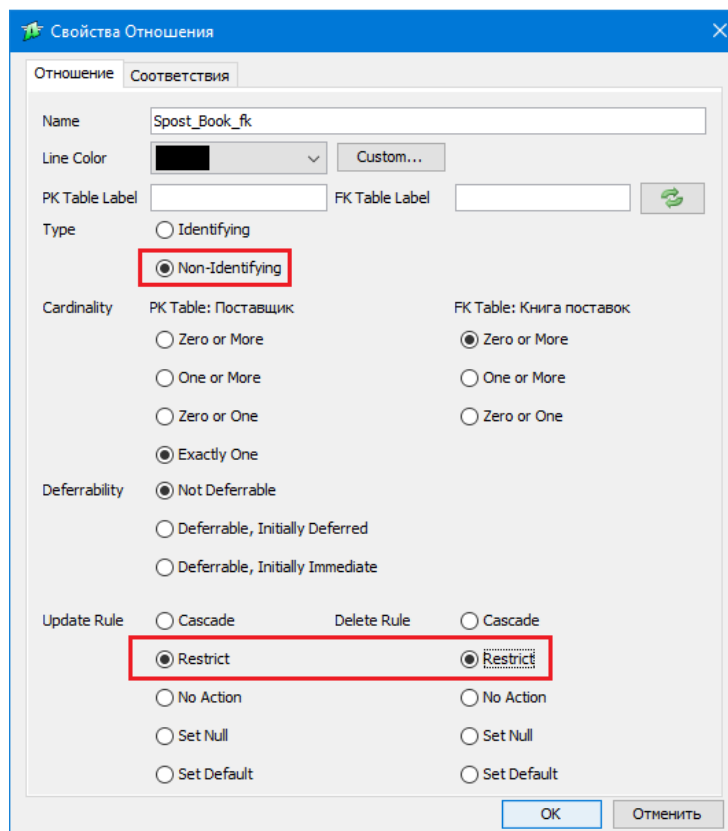
Выберем неидентифицирующую связь. При выборе такого типа связи атрибут первичного ключа родительской сущности (в примере *Код поставщика* в сущности *Поставщик*) при миграции его в дочернюю сущность *Книга поставок* не будет являться частью первичного ключа этой сущности.

Для установления связи необходимо в панели инструментов выбрать кнопку инструмента *Новая неидентифицирующая связь*, а затем щелкнуть вначале по родительской сущности, а потом по подчиненной. На диаграмме между сущностями будет установлена связь, отображаемая пунктирной линией (рис.14). Атрибут первичного ключа родительской таблицы мигрирует в подчиненную таблицу и будет отмечен аббревиатурой FK (Foreign Key).



**Рис. 14 Связь сущностей Книга поставок – Поставщик**

Свойства связи можно отредактировать дважды щелкнув по линии связи. Откроется окно (рис. 15). Группа переключателей *Type* задает тип связи (для выделенной связи тип неидентифицирующая). Группа переключателей *Cardinality* задает мощность связи. Мощность в данном случае «один-многим» - одному покупателю может соответствовать много договоров (договора обычно каждый год заключаются заново). Группа переключателей *Update Rule* определяет действия при изменениях первичного ключа в родительской сущности (значение атрибута Код покупателя), когда в подчиненной сущности имеются связанные строки: *Restrict* (*Запретить*), *Cascade* (*Каскадировать*). Выберем *Restrict*. Аналогично для группы переключателей *Delete Rule*, которые определяют действия при удалении строки в родительской сущности при наличии в подчиненной сущности связанных строк. следует выбрать *Restrict*.



**Рис. 15 Установка свойств связи**

### **3.2 Физическое проектирование БД**

Необходимо переключиться на физический уровень представления модели. Для этого необходимо в окне настройки свойств проекта (рис. 5) задать вид отображаемой модели (переключатель *Display Tables and Columns*) на *Physical Names*.

Далее необходимо убедиться, что все имена на физическом уровне (таблиц, колонок, индексов) заданы в латинском алфавите.

При выборе типов колонок целесообразно следовать следующим рекомендациям.

Для колонок таблицы следует выбирать типы данных:

для дат – тип DATE;

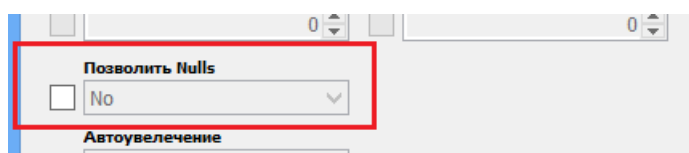
для целых числовых значений – INTEGER (для чисел длиной 4 байта, т.е. в диапазоне от -2147483648 до +2147483647) или BIGINT (8 байт);

для вещественных (дробная часть отделяется «точкой») – NUMERIC;

для символьных – VARCHAR(XXX), где XXX – максимальное количество символов в колонке. Допускается не задавать количество колонок.

Для количественных колонок вещественного типа (объем, количество и т.п.) зададим тип *Numeric(m,n)*. Здесь *m* – задает общее количество позиций, занимаемое числовым значением (включая дробную часть), а *n* – количество позиций в дробной части. Для колонок для стоимостных значений (цена, стоимость и т.п.) *n=2*.

Для колонок обязательных к заполнению необходимо задать опцию NOT NULL. Для этого необходимо выключить флаг *Позволить Nulls* (рис.16).



**Рис. 16 Настройка обязательности заполнения колонки**

Для колонок первичного ключа, являющихся суррогатными ключами, выбирается тип *Integer* с заданием параметра автоувеличения (рис.17). Дополнительно задается имя объекта последовательности, которая реализует механизм автоувеличения. Имя последовательности будем задавать по шаблону «имя\_таблицы\_seq». Для таблицы *Книга продаж* имя последовательности *book\_seq*.

**Свойства Столбца Идентификатор поставки**

Source for ETL Mapping  
Ничего не определено

Logical Name  
Идентификатор поставки

Physical Name  
book\_id

☒ В Первичном Ключе

Тип  
INTEGER

Точность 0 Масштаб 0

☒ Позволить Nulls  
No

☒ Автоувелечение  
Yes

☒ Значение по умолчанию

Название последовательности  
book\_seq

Заметки

OK Отменить

**Рис. 17 Настройки столбца суррогатного первичного ключа**

Для количественных колонок целесообразно задать значения по умолчанию, которое будет присвоено колонке, если не задано явного значения. Если значение не задано, целесообразно присвоить «нулевое» значение 0 (рис.18).

**Свойства Столбца Количество**

Source for ETL Mapping  
Ничего не определено

Logical Name  
Количество

Physical Name  
Kol

☐ В Первичном Ключе

Тип  
NUMERIC

Точность 10 Масштаб ☒

☒ Позволить Nulls  
No

☐ Автоувелечение  
No

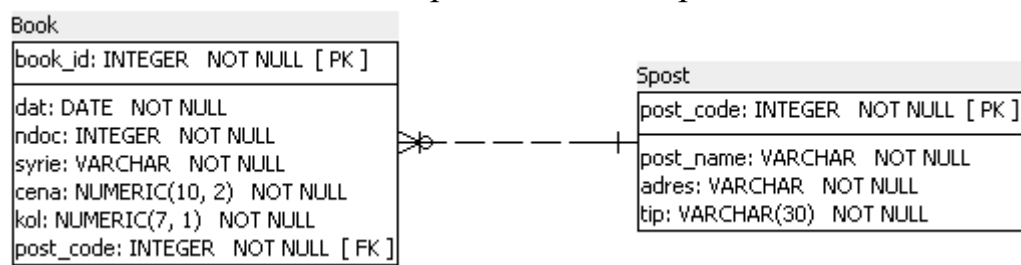
☒ Значение по умолчанию  
0

Заметки

OK Отменить

**Рис. 18 Настройка значения по умолчанию**

Физическая модель данных представлена на рис.19.



**Рис. 19 Физическая модель данных Поставки**

### **Задание.**

Для своего варианта индивидуального задания разработать логическую и физическую модели данных

В таблицах БД необходимо задать

- первичный ключ. В таблицах, не являющихся справочниками в качестве первичного ключа использовать идентификатор, формируемый автоматически и не совпадающий с заданными полями таблицы в индивидуальном задании;
- связи между таблицами (ссылочную целостность);
- значения по умолчанию для колонок таблиц

Индивидуальные задания находятся в отдельном файле

### **Содержание отчета**

- 1) Текст индивидуального задания
- 2) Схема логической и физической модели БД
- 3) Скриншот свойств колонки суррогатного первичного ключа
- 4) Скриншоты свойств колонок со значением по умолчанию
- 5) Скриншот свойств связи