МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное

образовательное учреждение высшего образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Институт информационных технологий\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

наименование института (факультета)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_МПО ЭВМ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

наименование кафедры \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Проектирование баз данных\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ наименование дисциплины в соответствии с учебным планом

ЛАБОРАТОРНАЯ РАБОТА № 2

ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

Исполнитель

студент \_\_1ПИб-02-1оп-22\_\_

группа

\_\_\_Микуцких Г. А.\_\_\_

Фамилия, имя, отчество

Руководитель \_\_\_Селяничев О.Л.\_\_\_

Ф.И.О. преподавателя

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_2025\_ год

Результат логического проектирования – ER-диаграмма (entity-relationship diagram), физического – сама база данных. ER-диаграмма – **логическая модель предметной области (**инфологическая модель).

1. Ознакомиться с понятием «валидация» и научиться применять его при проектировании.

*Валидация, значение по умолчанию, индексация. Описать на основе трёх шаблонов 4 правила и привязать к 5 объектам.* рис. 21

Валидация – это процесс проверки данных на соответствие ожидаемым форматам, типам, диапазонам значений и бизнес-правилам перед их сохранением в БД.

Значение по умолчанию – это значение, которое нужно ввести в поле записи, если никакое другое значение не задано явным образом во время ввода данных. С каждым полем или доменом можно связать значение по умолчанию (если выбранная СУБД поддерживает домены). Домен в реляционной модели данных – тип данных, то есть множество допустимых значений.

Программа ERwin поддерживает правила валидации для полей, а также значение, присваиваемое полям экземпляров по умолчанию.

В ходе выполнения лабораторной работы была создана логическая модель, описаны сущности, установлены для них правила валидации и значения по умолчанию по инфологической модели «Пансионат» (рис. 1).

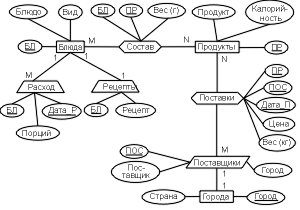


Рис. 1. Инфологическая модель базы данных «Пансионат»

Для создания модели сделано следующее (рис. 2-4).

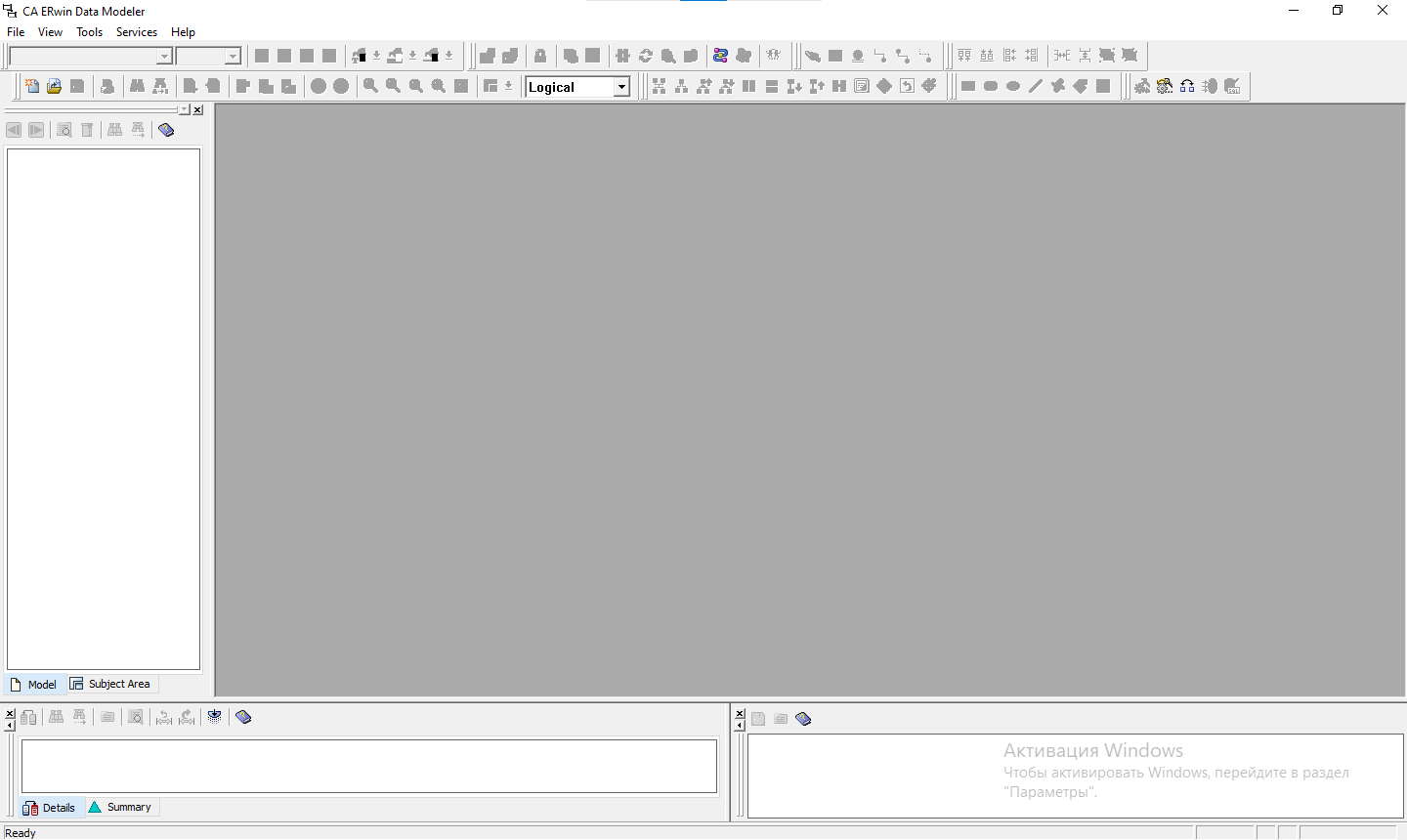


Рис. 2. Общий вид программы ERwin

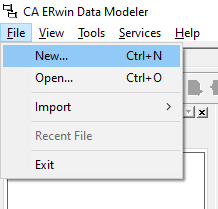


Рис. 3. Кнопка создания новой модели

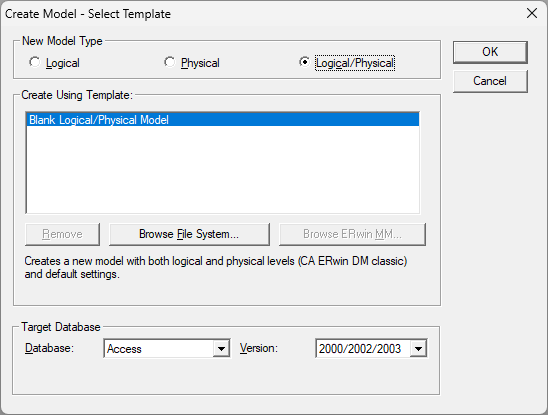


Рис. 4. Выбран Logical/Physical-тип модели и целевая БД Access

Для того, чтобы создать сущность модели (например, «Блюда»), на панели инструментов нужно выбрать инструмент «Entity», нажать ЛКМ по рабочему окну текущей модели, дать содержательное имя сущности, его первичным(-ому) ключам(-у) (маркер Primary Key), атрибутам и указать их тип данных (рис. 5-7).

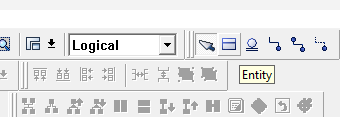


Рис. 5. Инструмент «Entity»

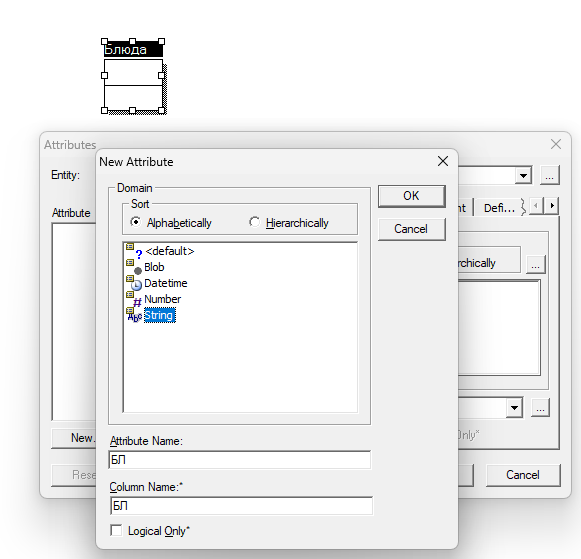


Рис. 6. Раздел «Attributes»

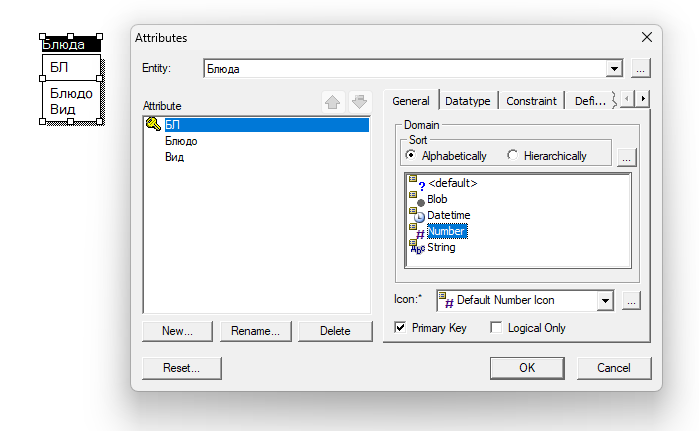


Рис. 7. Первичный ключ и атрибуты созданы

Если панели нет, то включить её отображение можно по пути на рис. 8.

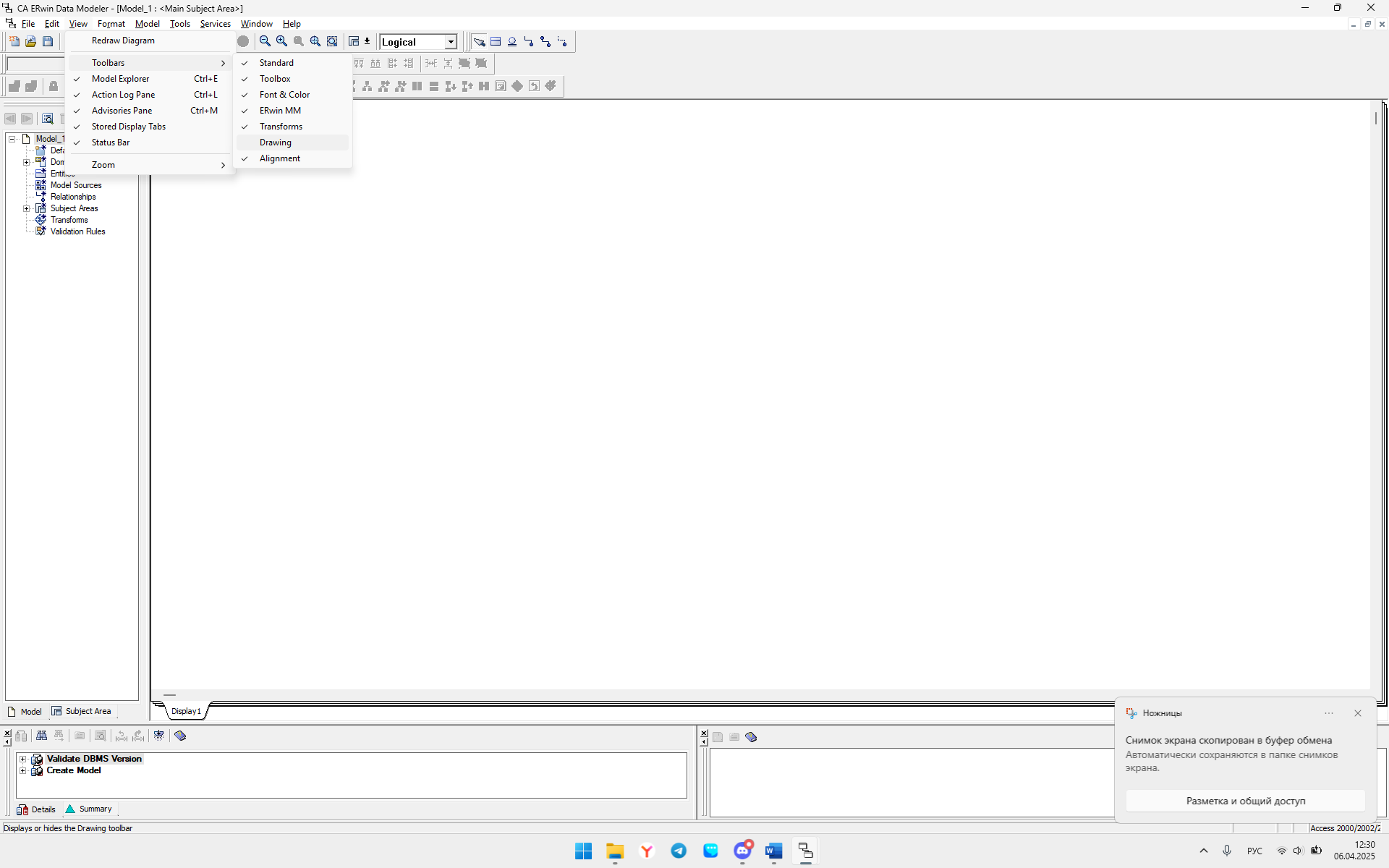


Рис. 8. Отображение «Drawing» для создания сущностей

Была создана ещё одна сущность – «Рецепты». Чтобы создать отношение (связь) между сущностями (один-к-одному, один-ко-многим, многие-ко-многим), нужно воспользоваться инструментом «Relationship». Он имеет два варианта отображения (нотации): «IDEF1X» и «IE».

Для разработки логической модели будет удобнее работать с отображением «куриная лапка», то есть «IE» (рис. 9-11).

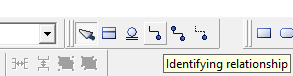


Рис. 9. Инструмент «Relationship»

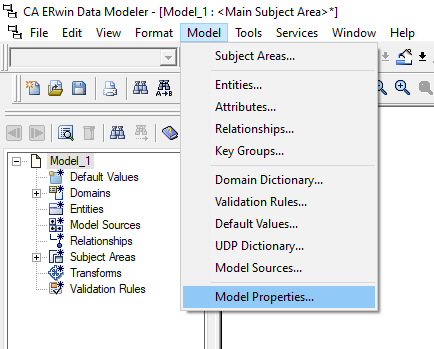


Рис. 10. «Model Properties…»

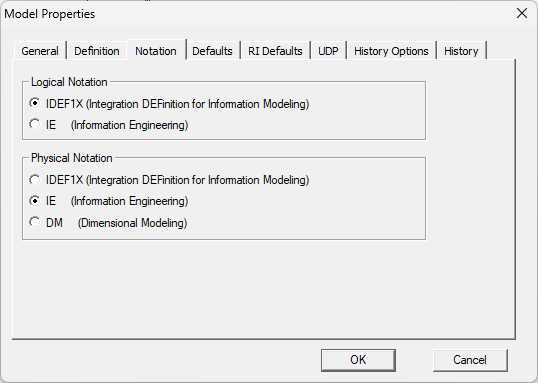


Рис. 11. Окно выбора нотации модели

Чтобы создать связь, нужно выбрать инструмент «Relationship», нажать сначала ЛКМ по сущности, из которой будет идти отношение, а затем ЛКМ по второй сущности, с которой будет связана первая. Если у первой сущности были первичные ключи, они становятся внешними ключами (Foreign Key) у второй сущности для идентификации с каким конкретно экземпляром первой сущности связан экземпляр второй сущности (рис. 12).

Скруглённые углы показывают дочернюю сущность в отношении.

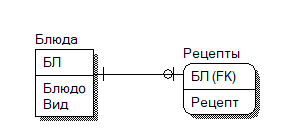


Рис. 12. Связь создана

В данном случае, создалась связь один-ко-многим, однако если сделать двойной ЛКМ по образовавшейся связи, то появится окно, на котором видно, что связь именуется как один-к-нулю-или-одному, что показывает кружок на связи возле сущности «Рецепты» (рис. 13).

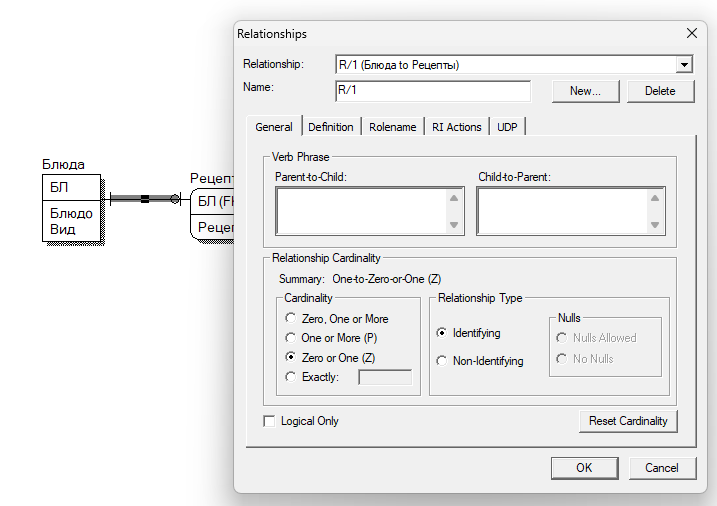


Рис. 13. Окно редактирования типа связи

Чтобы чётко указать связь один-к-одному (одно блюдо имеет один рецепт), нужно выбрать маркер «Exactly», установить значение «1», а после установить отображение мощности в настройках «Relationship Display» (рис. 14-15).

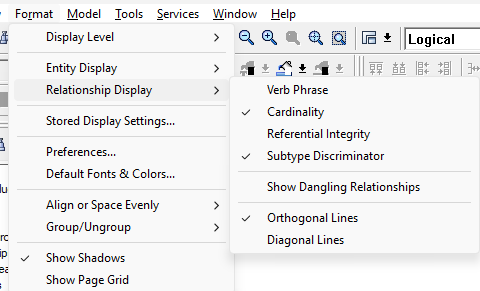


Рис. 14. Настройка отображения «Cardinality»

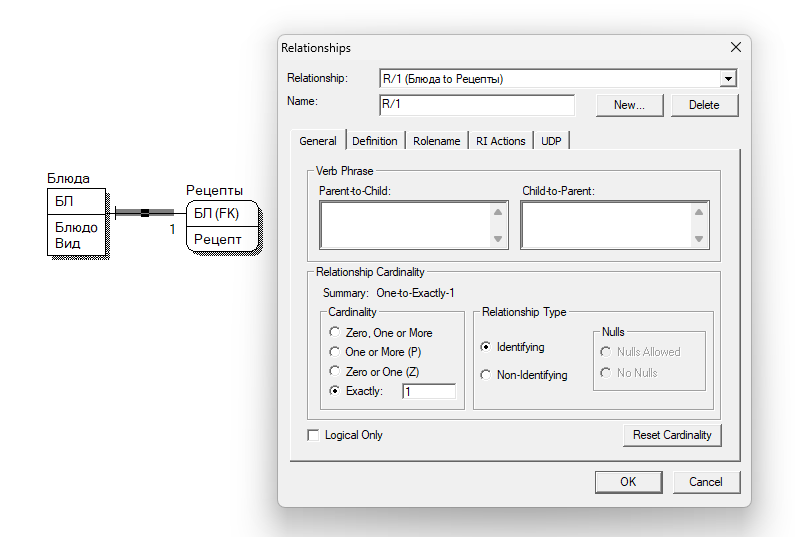


Рис. 15. Отображается связь один-ко-многим

У инструмента «Relationship» есть два режима: identifying и non-identifying.

Identifying relationship – тип связи, при котором существование дочернего объекта зависит от родительского, то есть дочерний объект не может существовать без родительского, а первичный ключ родительского объекта является частью первичного ключа дочернего.

Non-identifying relationship (слабая связь) – тип связи, при котором существование одного объекта не зависит от другого.

Были созданы остальные сущности (рис. 16).

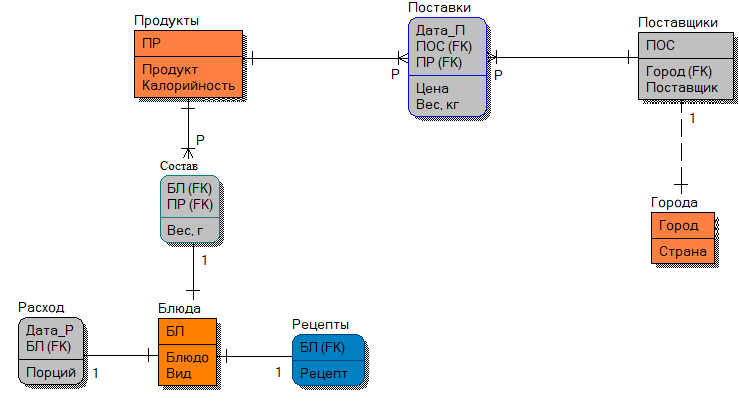


Рис. 16. Сущности модели «Пансионат»

Сущность «Поставщики» слабо связана с сущностью «Города». Город может не иметь поставщиков, но поставщик обычно работает хотя бы в одном городе.

Чтобы указать имя связи («Verb Phrase»), нужно двойным щелчком ЛКМ нажать по связи и в полях «Parent-to-Child» и «Child-to-Parent» написать по одному глаголу, отражающему взаимосвязь сущностей. Для отображения имён в рабочем окне необходимо включить соответствующую настройку (рис. 17-18).

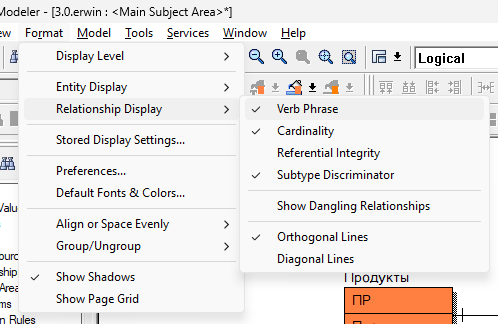


Рис. 17. Отображение имён связи включено

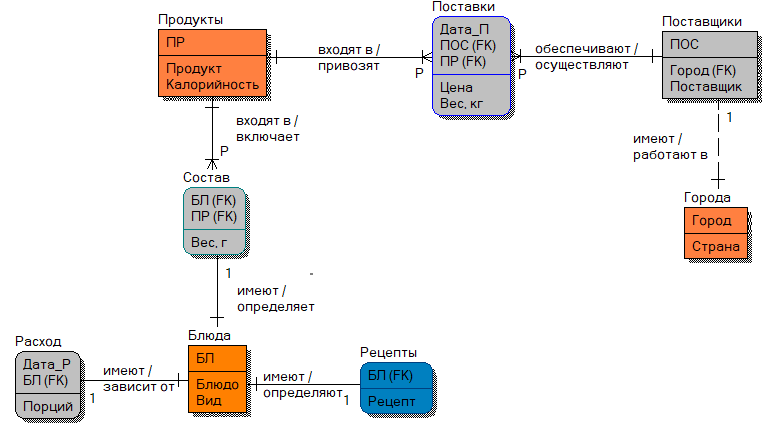


Рис. 18. Введены имена связей

В окне свойств отношения можно проверить корректно ли определено отношение между сущностями (рис. 19).

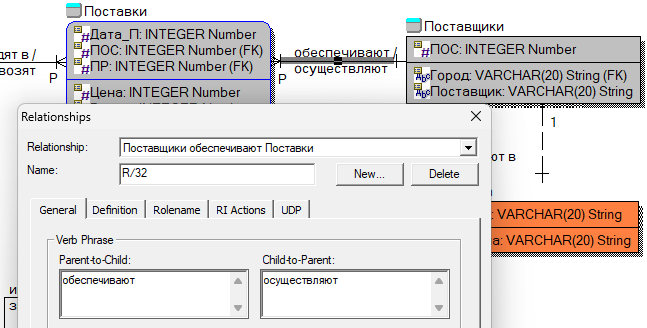


Рис. 19. Пример отношения

Чтобы включить визуализацию типов, доменов, атрибутов и ключей, можно поставить галочки в настройках отображения сущностей (рис. 20).

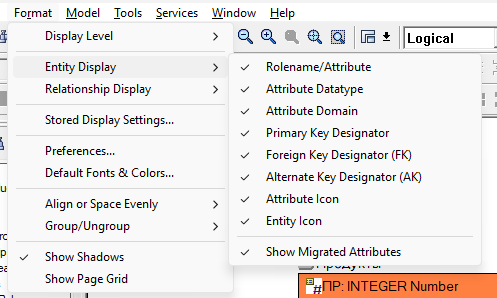


Рис. 20. Настройка «Entity Display»

Для создания правил валидации и значений по умолчанию необходимо двойным щелчком ЛКМ по сущности открыть свойства «Constraint», нажать на «…» для перехода к списку всех правил валидации, создать правило на основе трёх шаблонов и нажать «New», после указав имя. Открыть правила валидации также можно в панели меню по пути «Tools» -> «Validation Rules» (рис. 21-23).

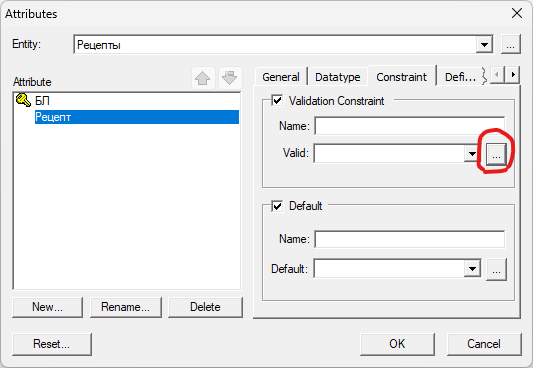


Рис. 21. Способ 1 открытия правил валидации

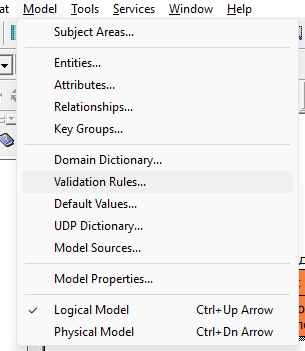


Рис. 22. Способ 2 открытия правил валидации

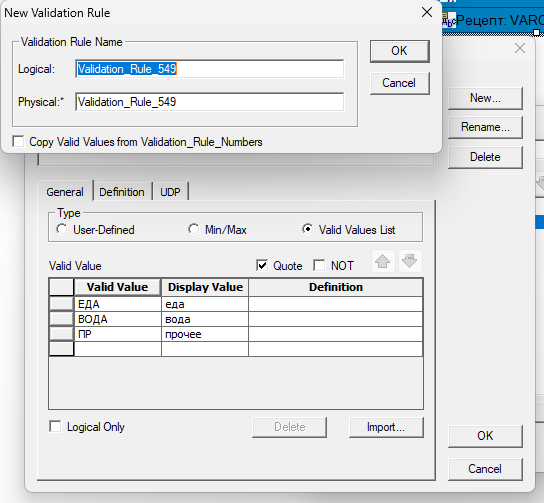


Рис. 23. Пример правила валидации

Выбрать правило валидации можно из выпадающего списка (рис. 24). Ответственность за соответствие типа атрибута и получаемого значения из выражения правила ложится на разработчика.

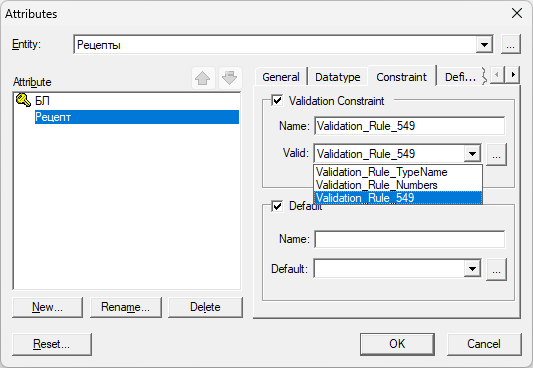


Рис. 24. Выпадающий список всех правил валидации

Шаблонов правил три:

* определяемое пользователем (используя синтаксис целевой БД);
* диапазон допустимых числовых значений;
* список (лист) допустимых значений.

Чтобы автоматически заключить каждое значение в одинарные кавычки, нужно включить опцию Quote. Опция NOT инвертирует условие.

В ходе выполнения лабораторной работы были заданы четыре правила валидации, представленные на рис. 25.

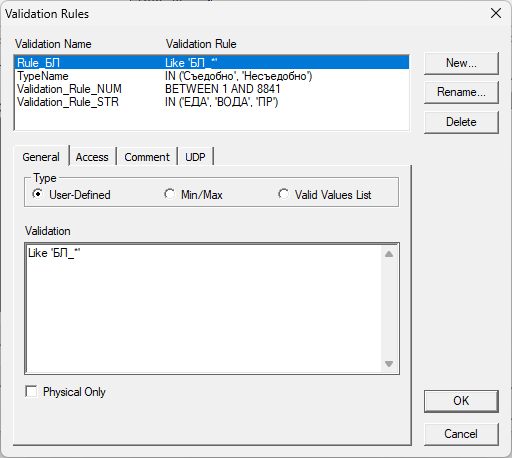


Рис. 25. Правила валидации для модели «Пансионат»

Создание значения по умолчанию проводится схожим образом. Были сделаны следующие значения по умолчанию (рис. 26).

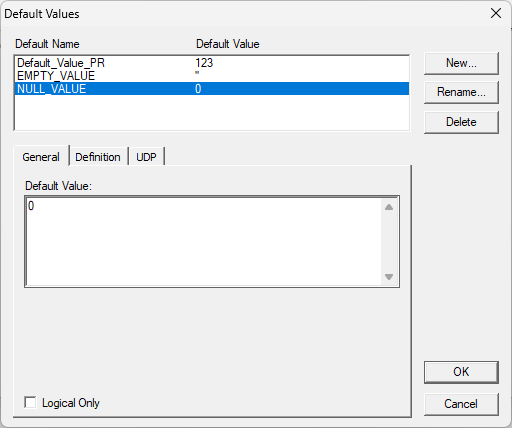


Рис. 26. Значения по умолчанию

Правила были применены к следующим атрибутам (рис. 27-31).

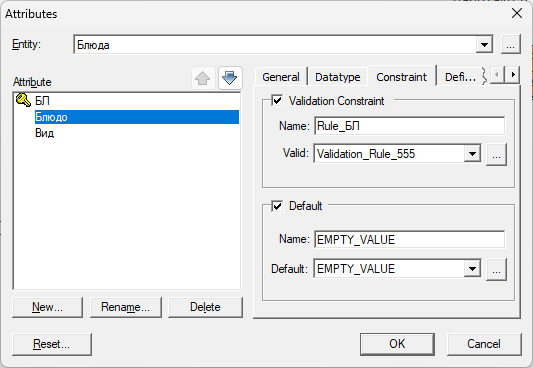


Рис. 27. Установлены правила для атрибута «Блюдо»

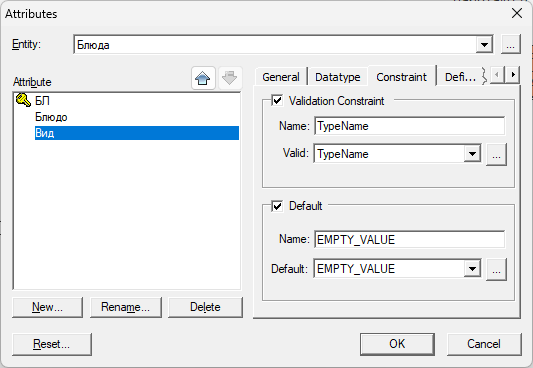


Рис. 28. Установлены правила для атрибута «Блюдо»

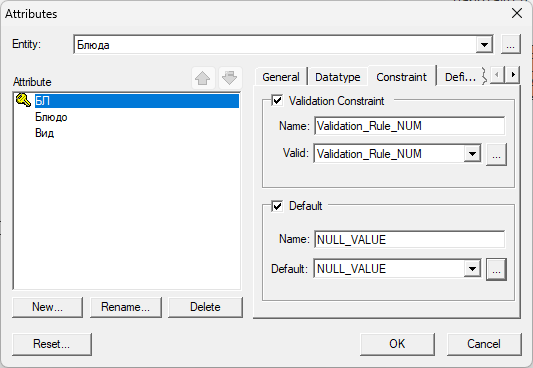


Рис. 29. Установлены правила для атрибута «БЛ»

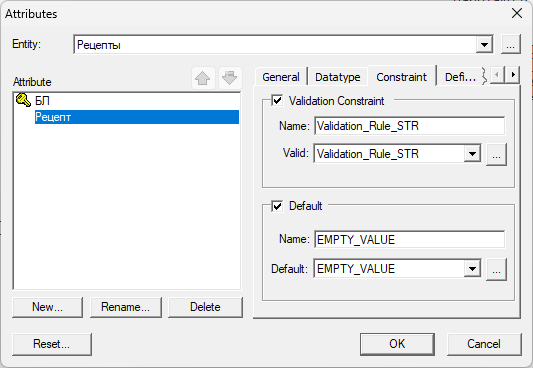


Рис. 30. Установлены правила для атрибута «Рецепт»

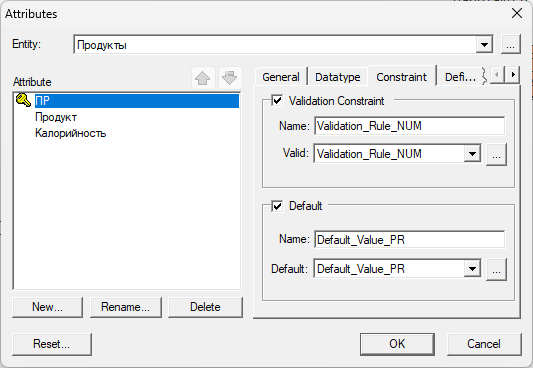


Рис. 31. Установлены правила для атрибута «ПР»

1. Изучить правила обращения ERwin’а с индексами.

В таблице БД данные обычно хранятся в том же порядке, в котором их ввели в таблицу. Многие реляционные СУБД имеют страничную организацию, при которой физически таблица может храниться фрагментарно в разных областях диска, причём записи таблицы располагаются на страницах неупорядоченно. Хотя такой способ хранения и позволяет быстро вводить новые данные, но для того, чтобы найти нужную запись, придётся просмотреть всю таблицу. В промышленных системах каждая таблица может содержать миллионы записей, поэтому простой перебор ведёт к катастрофическому падению производительности ИС.

Чтобы решить проблему поиска данных, СУБД использует особый объект, называемый индексом. Он подобен содержанию книги, которое указывает на все номера страниц, посвящённых конкретной теме. Индекс содержит отсортированную по полю или нескольким полям информацию и указывает на записи, в которых хранится конкретное значение поля.

При генерации схемы физической БД ERwin автоматически создаёт отдельный индекс на основе первичного ключа каждой таблицы, а также на основе всех альтернативных ключей, внешних ключей и инверсионных входов, поскольку эти поля наиболее часто используются для поиска данных. Можно отказаться от генерации индексов по умолчанию и для повышения производительности создать собственные индексы. Администратор СУБД должен анализировать наиболее часто выполняемые запросы и создавать индексы с различными полями и порядком сортировки для увеличения эффективности поиска при работе конкретных приложений.

При создании индекса на основе ключа ERwin вводит в его состав все поля ключа. Следовательно, на уровне логической модели можно неявно создать индекс, включая поля в состав альтернативных ключей и инверсионных входов.

ERwin автоматически генерирует имя индекса, созданного на основе ключа по принципу «X» + имя ключа + имя таблицы (физическое имя таблицы, а не логическое имя сущности!), где имя ключа «РК» для первичного ключа, «IFn» - для внешнего, «AKn» - для альтернативного, «IEn» -для инверсионного входа.

Изменить характеристики существующего индекса или создать новый можно в редакторе «Key Groups». Для его вызова следует щёлкнуть ПКМ по таблице и выбрать во всплывающем меню «Key Groups» (рис. 32-34).

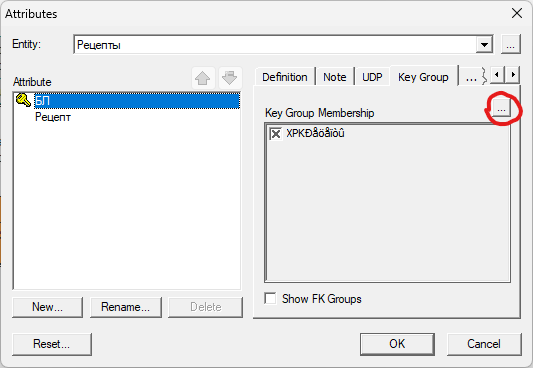


Рис. 32. Способ 1 открытия индексов

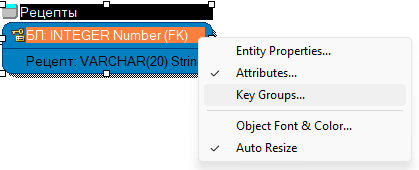


Рис. 33. Способ 2 открытия индексов

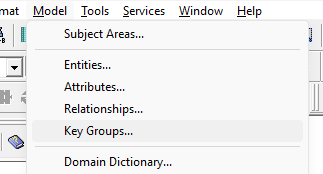


Рис. 34. Способ 3 открытия индексов

В редакторе можно изменить имя индекса, изменить его определение так, чтобы он принимал уникальные (Alternate Key) или дублирующиеся значения (Inversion Entry), или изменить порядок сортировки данных (рис. 35-36).

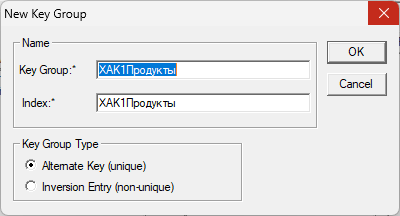


Рис. 35. Окно кнопки «New»

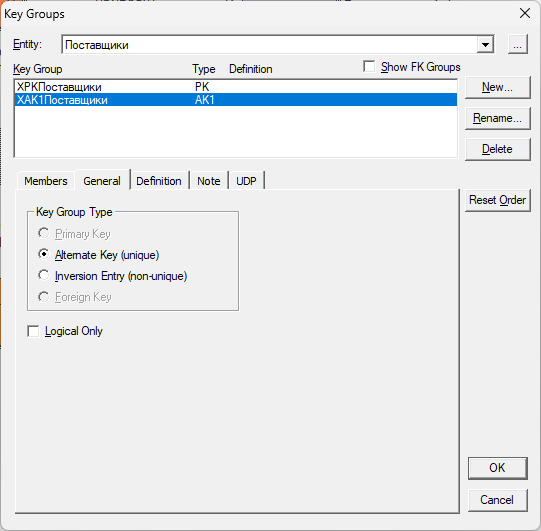


Рис. 36. Раздел «General»

Были созданы следующие дополнительные индексы (рис. 37-40).

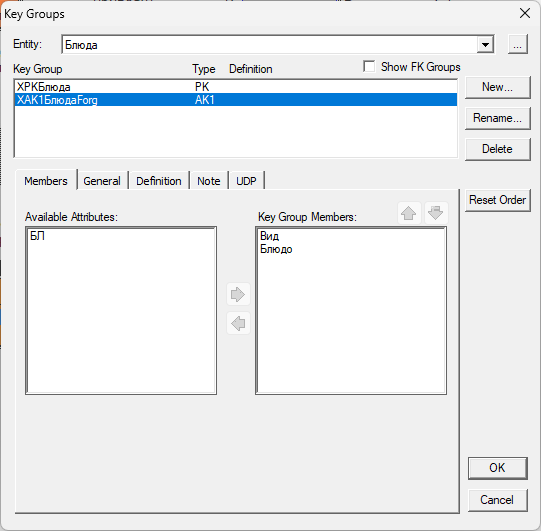


Рис. 37. Индексы «Блюд»

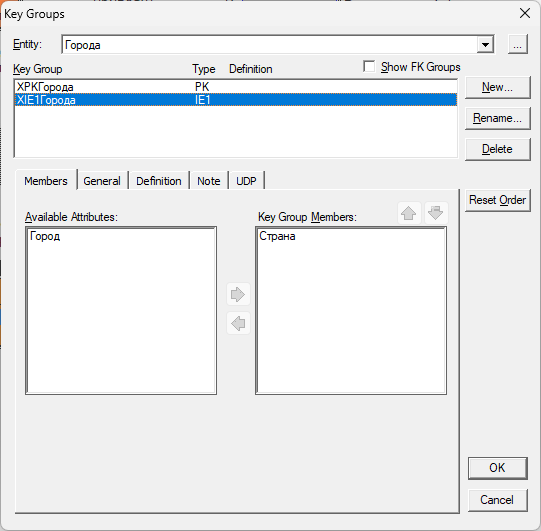


Рис. 38. Индексы «Городов»

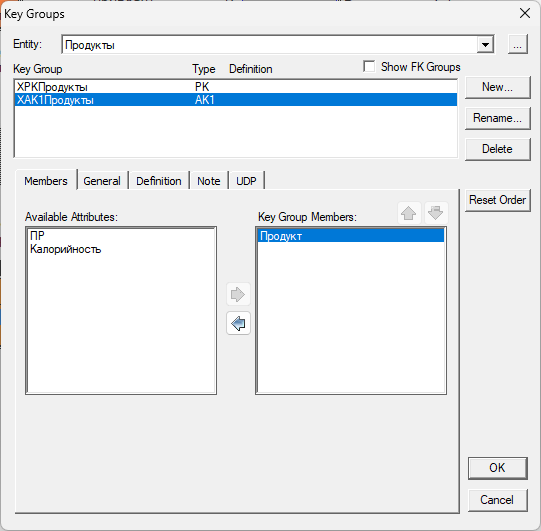


Рис. 39. Индексы «Продуктов»

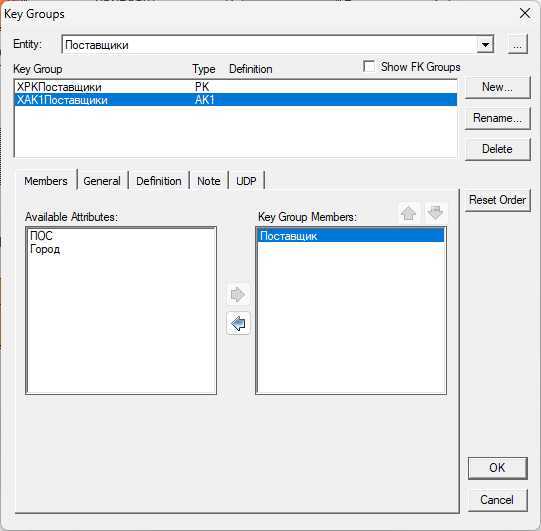


Рис. 40. Индексы «Поставщиков»

Результат логического проектирования показан на Рис. 41.

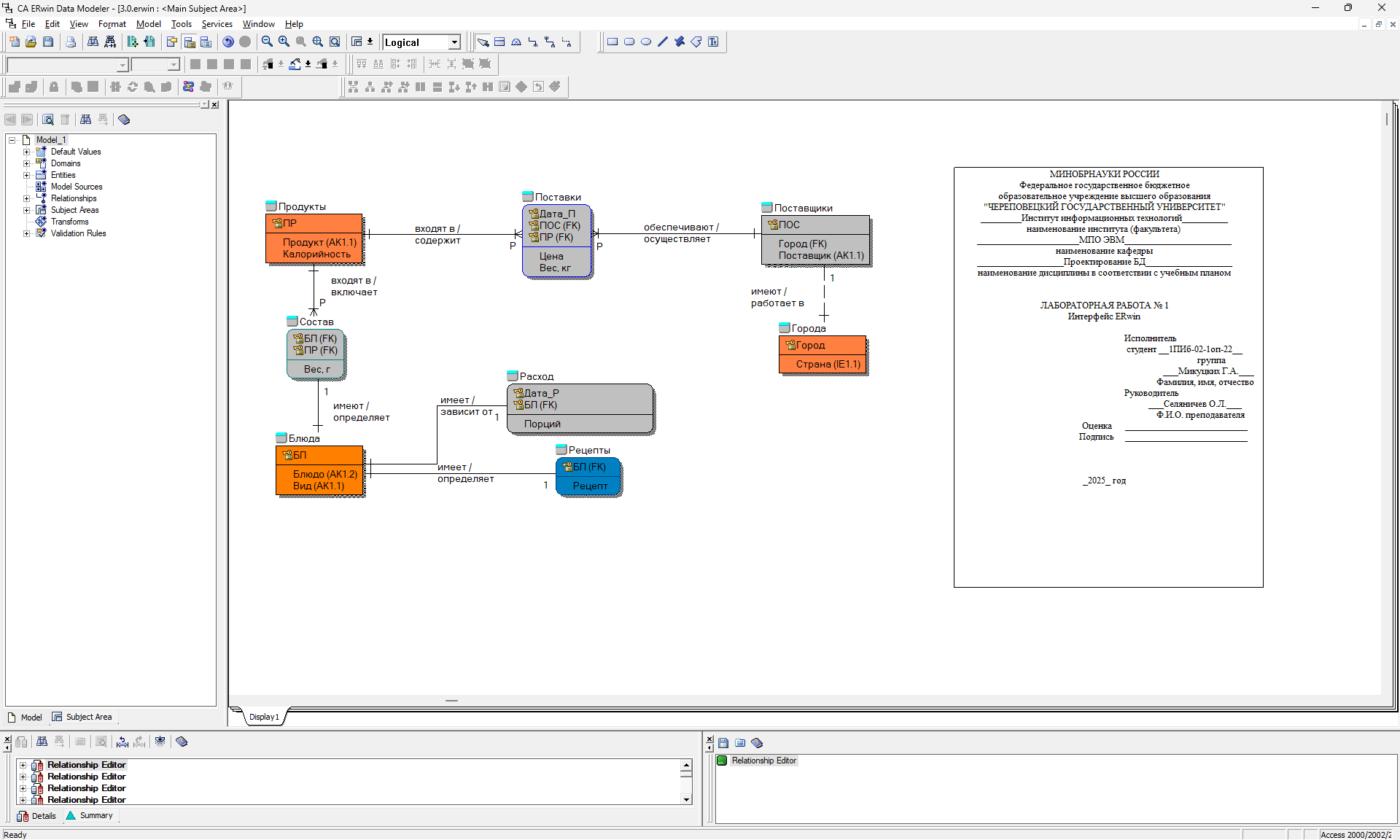


Рис. 41. Результат логического проектирования «Пансионата»

1. Рассмотреть работу на этапах логического и физического проектирования, уделив особое внимание последнему.

ERwin – это средство концептуального моделирования БД, использующее методологию IDEF1X. Методология IDEF1X и была разработана с учётом таких требований, как простота изучения и возможность автоматизации. Базируется на методе IDEF1, разработанном Т. Рэмей (T. Ramey), основанном на подходе Питера Чена. Метод IDEF1 позволяет построить модель данных, эквивалентную реляционной модели в третьей нормальной форме; используется для формирования графических представлений информационных моделей, которые отражают структуру и семантику информации внутри среды или системы.

Erwin является средством логического моделирования и позволяет работать в двух режимах:

* Logical – логическое представление логической модели предметной области;
* Physical – физическое представление логической модели предметной области.

Логический уровень – это абстрактный взгляд на данные, на нём данные представляются так, как выглядят в реальном мире, и могут называться так, как они называются в реальном мире, например «Постоянный клиент», «Отдел» или «Фамилия сотрудника». Объекты модели, представляемые на логическом уровне, называются сущностями и атрибутами. Логическая модель данных может быть построена на основе другой логической модели, например на основе модели процессов. Логическая модель данных является универсальной и никак не связана с конкретной реализацией СУБД.

Связь может дополнительно определяться с помощью указания степени или мощности (количества экземпляров сущности-потомка, которое может существовать для каждого экземпляра сущности-родителя). В IDEF1X могут быть выражены следующие мощности связей:

* каждый экземпляр сущности-родителя может иметь ноль, один или более связанных с ним экземпляров сущности-потомка;
* каждый экземпляр сущности-родителя должен иметь не менее одного связанного с ним экземпляра сущности-потомка;
* каждый экземпляр сущности-родителя должен иметь не более одного связанного с ним экземпляра сущности-потомка;
* каждый экземпляр сущности-родителя связан с некоторым фиксированным числом экземпляров сущности-потомка.

Если экземпляр сущности-потомка однозначно определяется своей связью с сущностью-родителем, то связь называется идентифицирующей, в противном случае - неидентифицирующей.

Физическая модель данных, напротив, зависит от конкретной СУБД, фактически являясь отображением системного каталога. В физической модели содержится информация о всех объектах БД. Так как стандартов на объекты БД не существует (нет стандарта на типы данных), физическая модель зависит от конкретной реализации СУБД. Следовательно, одной и той же логической модели могут соответствовать несколько разных физических моделей. Если в логической модели не имеет значения, какой конкретно тип данных имеет атрибут, то в физической модели важно описать всю информацию о конкретных физических объектах (таблицах, полях, индексах, процедурах и т.д.).

Для переключения между логической и физической моделью данных служит выпадающий список выбора в панели инструментов (рис. 42).



Рис. 42. Поле переключения между моделями

Основные компоненты диаграммы ERWin – это сущности, атрибуты и связи. Каждая сущность является множеством подобных индивидуальных объектов, называемых экземплярами. Каждый экземпляр индивидуален и должен отличаться от всех остальных экземпляров. Атрибут выражает определённое свойство объекта. С точки зрения БД (физическая модель) сущности соответствует таблица, экземпляру сущности - запись в таблице, а атрибуту - поле таблицы.

Ключи. Каждый экземпляр сущности должен быть уникален и отличаться от других атрибутов.

Первичный ключ (primary key) — это атрибут или группа атрибутов, однозначно идентифицирующая экземпляр сущности. Атрибуты первичного ключа на диаграмме не требуют специального обозначения — это те атрибуты, которые находятся в списке атрибутов выше горизонтальной линии.

Выбор первичного ключа может оказаться непростой задачей, решение которой может повлиять на эффективность будущей ИС. В одной сущности могут оказаться несколько атрибутов или наборов атрибутов, претендующих на роль первичного ключа. Такие претенденты называются потенциальными ключами (candidate key). Ключи могут быть сложными, т.е. содержащими несколько атрибутов. Сложные первичные ключи не требуют специального обозначения — это список атрибутов выше горизонтальной линии:"

Внешние ключи (Foreign Key) создаются автоматически, когда связь соединяет сущности: связь образует ссылку на атрибуты первичного ключа в дочерней сущности и эти атрибуты образуют внешний ключ в дочерней сущности (миграция ключа). Атрибуты внешнего ключа обозначаются символом (FK) после своего имени.

Зависимая сущность может иметь один и тот же внешний ключ из нескольких родительских сущностей. Сущность может также получить один и тот же внешний ключ несколько раз от одного и того же родителя через несколько разных связей. Когда ERWin обнаруживает одно из этих событий, он распознает, что два атрибута одинаковы, и помещает атрибут внешнего ключа в зависимой сущности только один раз. Хотя в закладке Key Group диалога Attribute Editor этот атрибут будет входить в два внешних ключа, на диаграмме он показывается только один раз. Это комбинирование или объединение идентичных атрибутов называется унификацией.

ERWin не содержит полного алгоритма нормализации и не может проводить нормализацию автоматически, однако его возможности облегчают создание нормализованной модели данных. Запрет на присвоение неуникальных имён атрибутов в рамках модели (при соответствующей установке опции Unique Name) облегчает соблюдение правила «один факт - в одном месте». Имена ролей атрибутов внешних ключей и унификация атрибутов также облегчают построение нормализованной модели.

Денормализация, как правило, проводится на уровне физической модели. ERWin позволяет сохранить на уровне логической модели нормализованную структуру, при этом построить на уровне, физической модели структуру (возможно, денормализованную), которая обеспечивает лучшую производительность, используя особенности конкретной СУБД и бизнес-правил предметной области.

Физическое проектирование – это создание схемы базы данных для конкретной системы управления базами данных (СУБД).

Физическая модель содержит всю информацию, необходимую для реализации конкретной БД. Трансформационная модель содержит информацию для реализации отдельного проекта, который может быть частью общей ИС и описывать подмножество предметной области. ERwin поддерживает ведение отдельных проектов, позволяя проектировщику выделять подмножество модели в виде предметных областей (Subject Area). Трансформационная модель позволяет проектировщикам и администраторам БД лучше представлять, какие объекты БД хранятся в словаре данных, и проверить, насколько физическая модель данных удовлетворяет требованиям к ИС.

Модель СУБД автоматически генерируется из трансформационной модели и является точным отображением системного каталога СУБД. ERwin реализует проектирование схемы БД, генерацию её описания на языке целевой СУБД (ORACLE, Informix, Ingres, Sybase, DB/2, Microsoft SQL Server, Progress и др.) и реинжиниринг существующей БД.

По умолчанию ERwin генерирует имена таблиц и индексов по шаблону на основе имён соответствующих сущностей и ключей логической модели. Окна Table Name Macro и Index Name Macro позволяют изменить шаблон генерации имён, заданный по умолчанию. В дальнейшем имена таблиц и индексов можно изменить вручную. Если в имени сущности или атрибута встречается пробел, он будет заменён на символ "\_".

При смене СУБД ERwin предлагает автоматически преобразовать тип данных, связанный с каждым атрибутом, на ближайший, доступный для новой СУБД.

Перед физическим моделированием необходимо переключиться на физическое представление модели (см. рис. 42). Также необходимо заранее создать БД формата «.mdb» (если используется СУБД Access), так как форматы новее ERwin не поддерживает, и подключиться к файлу БД (рис. 43-46).

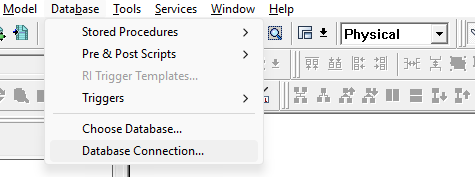


Рис. 43. Переход в «Database Connection»

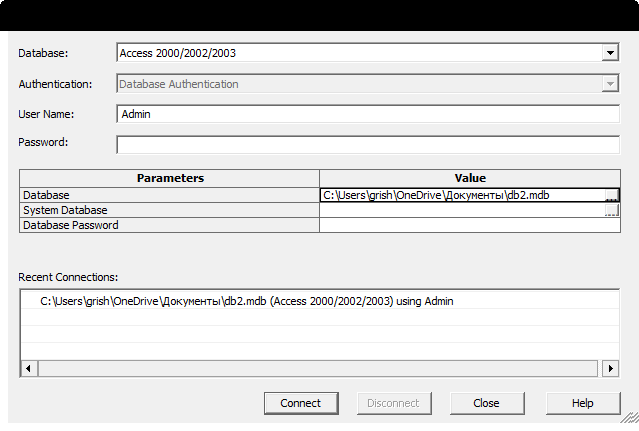


Рис. 44. Пример данных для ввода



Рис. 45. ERwin работает только с форматом «.mdb» при использовании Access



Рис. 46. ERwin использует БД Access

Для генерации базы данных Access был проведён «прямой инжиниринг» из физического представления логической модели (рис. 47-49).

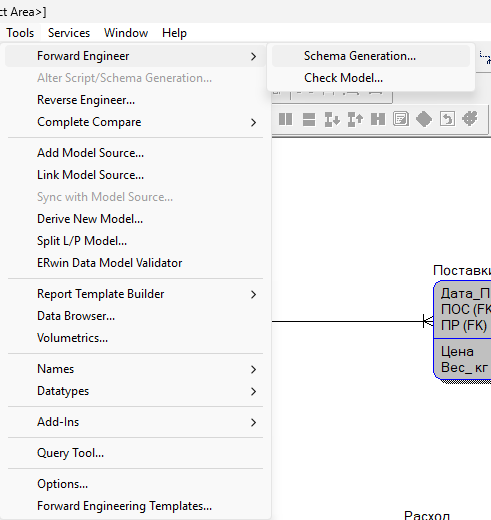


Рис. 47. «Schema Generation»

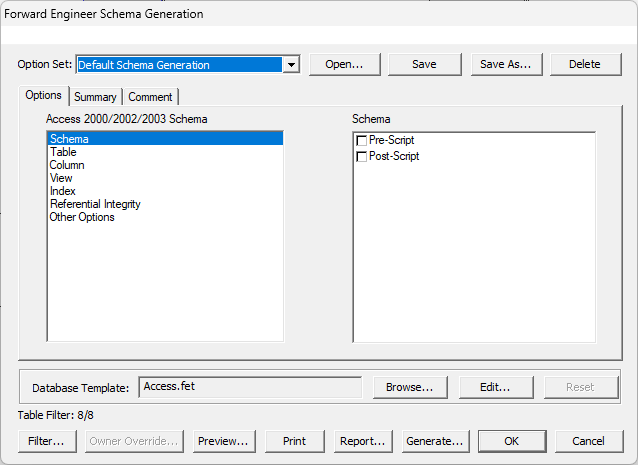


Рис. 48. «Generate» для «Schema»

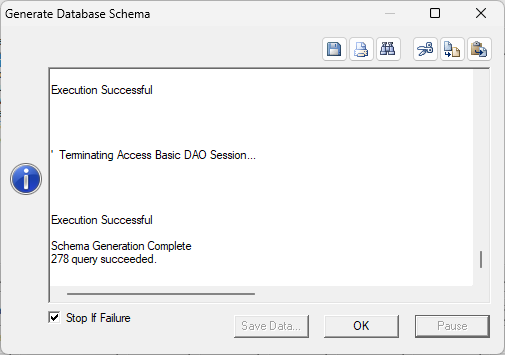


Рис. 49. Окно после генерации (прошла успешно)

Также можно задать дополнительные параметры для генерации или предварительно посмотреть на реализуемые запросы (рис. 50-51).

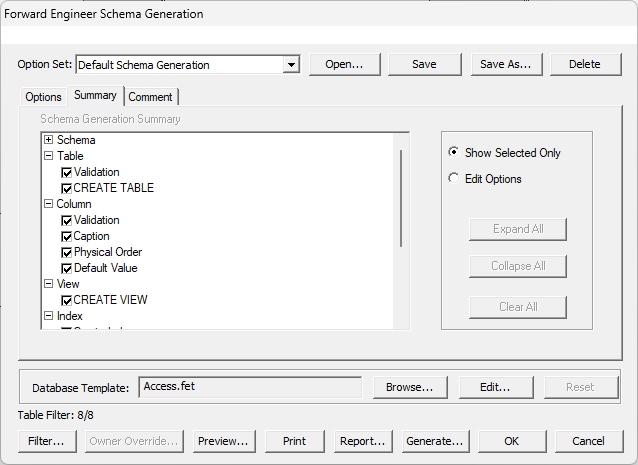


Рис. 50. «Summary»

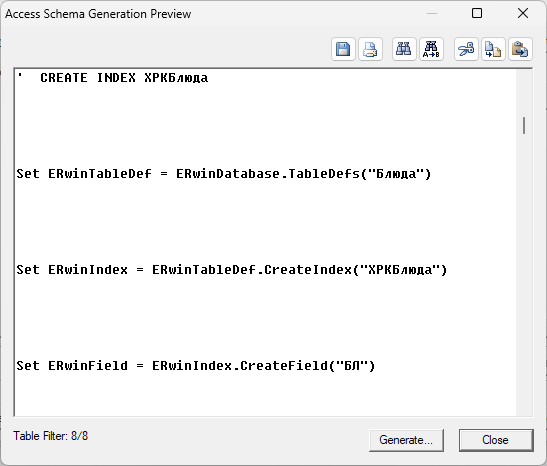


Рис. 51. «Preview»

Если открыть БД после генерации, то видно, что на каждую сущность создана своя таблица, а в «Работа с базами данных» → «Схема данных» можно увидеть отношения между сущностями (рис. 52).

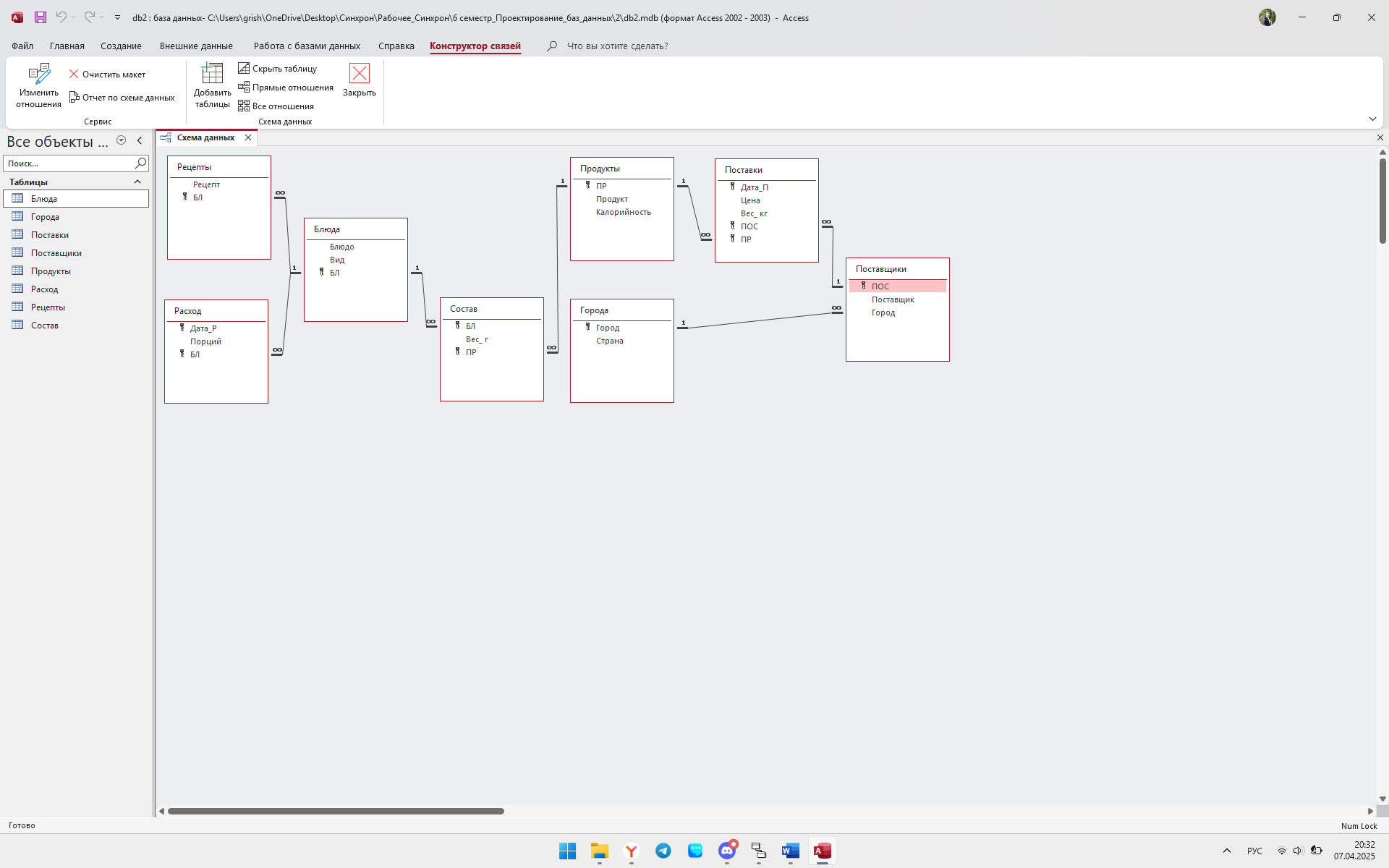


Рис. 52. Сущности в БД Access созданы

В «Конструкторе таблиц» можно увидеть, что правила из ERwin перенеслись в поля «Правило проверки», «Значение по умолчанию», а ключевой атрибут стал «Индексированным полем» без совпадений (рис. 53). Инверсионный ключ допускает совпадения, является индексируемым, но не обязательным.

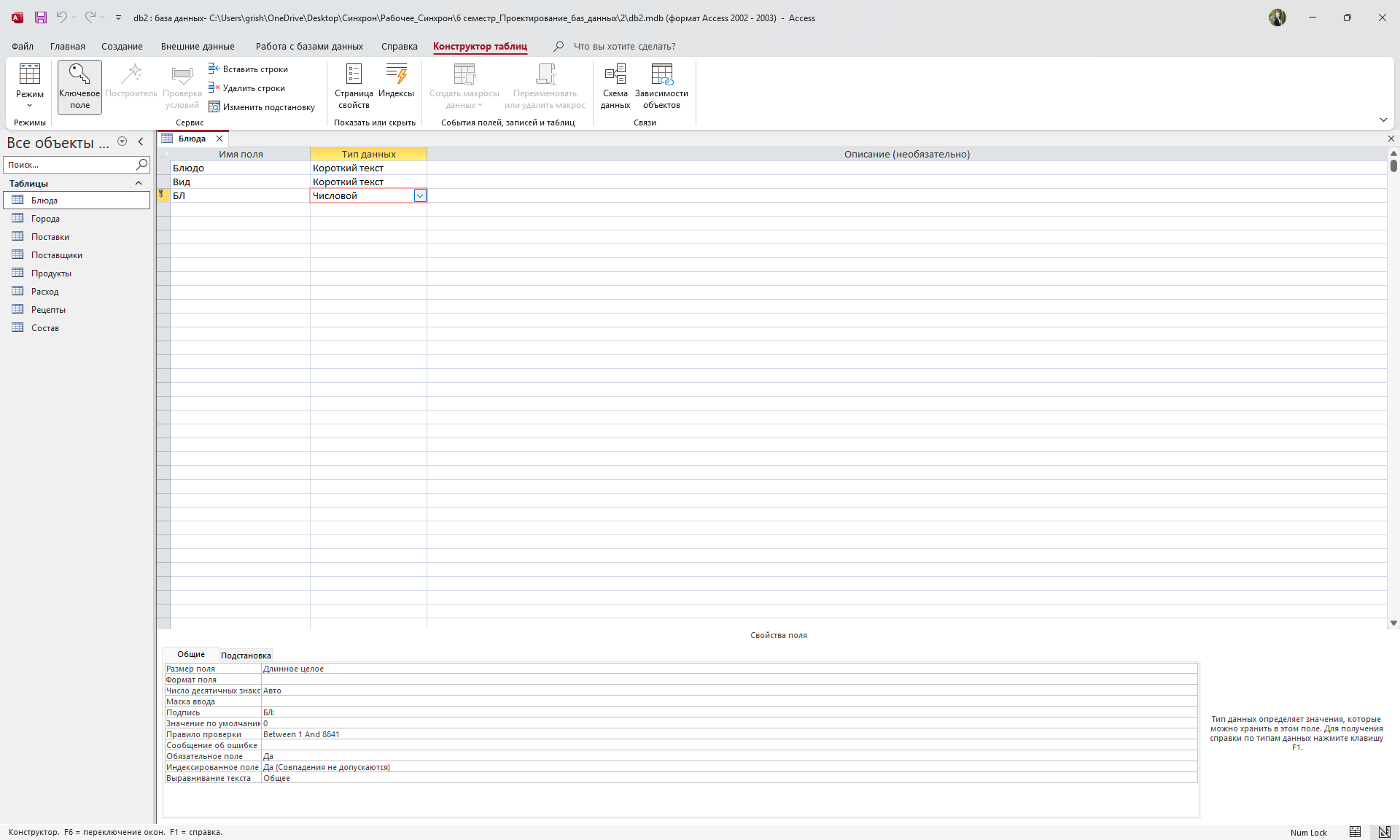


Рис. 53. Свойства из ERwin перенеслись в БД Access

При попытке ввести значение, нарушающее правило валидации, Access выдаст ошибку (рис. 54-56).

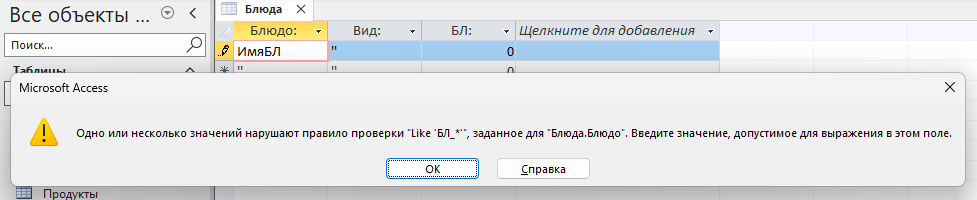


Рис. 54. Пример нарушения правила валидации

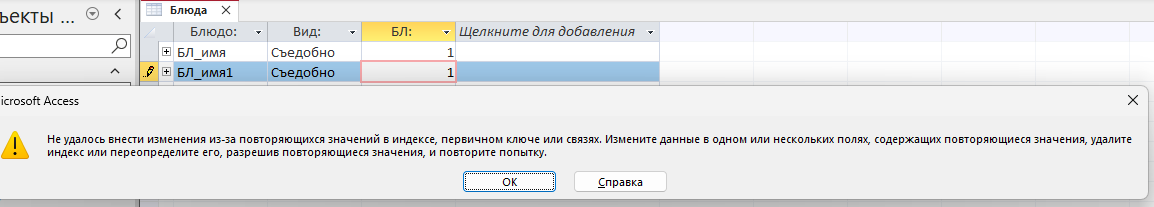


Рис. 55. Окно ошибки при повторении значения первичного ключа

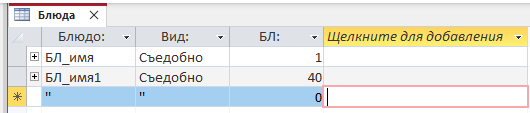


Рис. 56. Пример корректных данных

При отсутствии записи родительской сущности «Блюда» невозможно создать дочернюю «Расход», так как она ссылается по первичному ключу (которого, очевидно, нет), о чём предупреждает Access (рис. 57). Ввод первичного ключа, с которым существует экземпляр «Блюда», не вызывает ошибки.

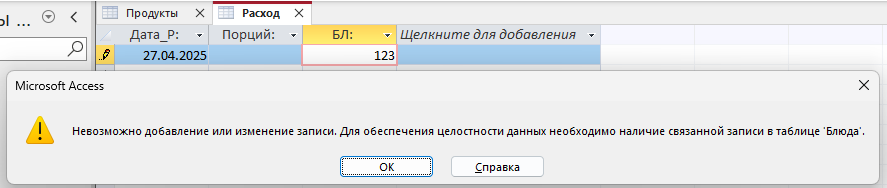


Рис. 57. Ошибка при отсутствии связной записи

При необходимости можно поменять базу данных (см. рис. 43, рис. 58).

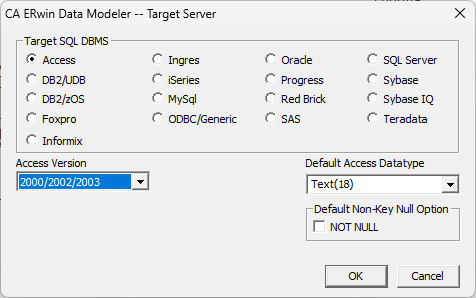


Рис. 58. Раздел «Choose Database»