БАЗЫ ДАННЫХ

ПРОЕКТИРОВАНИЕ

В.Н.Лукин

2020

ПРОЕКТИРОВАНИЕ ПРОГРАММНЫХ СИСТЕМ

Проектирование складывается из проектирования процессов, данных, событий, интерфейсов и выходных документов.

Проектирование данных — процесс разработки структуры базы данных согласно требованиям.

При проектировании нужно ответить на вопросы:

- что представляют собой требования заказчиков и в какой форме они выражены;
- как они преобразуются в структуру базы данных;
- как часто и каким образом структура базы данных должна перестраиваться.

УРОВНИ АБСТРАКЦИИ

Согласно точкам зрения заказчика, разработчика и администратора БД, выделяют три уровня абстракции:

- концептуальный,
- логический,
- физический.

Они определяют три основных шага проектирования.

КОНЦЕПТУАЛЬНЫЙ УРОВЕНЬ (1)

Наиболее общее представление об информационном содержании предметной области.

Представляется в виде концептуальной модели (информационной структуры).

Модель обладает высокой степенью стабильности, она проблемно-ориентирована и не зависит от конкретной СУБД, операционной системы и аппаратного обеспечения.

Её поведение должно быть предсказуемо.

КОНЦЕПТУАЛЬНЫЙ УРОВЕНЬ (2)

Концептуальное представление оперирует элементарными данными предметной области, представленными *сущностями*.

Сущности описываются *атрибутами*.

Данные могут находиться в некотором отношении друг с другом: образовывать ассоциации, которые называются *связями*.

Концептуальная модель должна поддерживать согласованность связей в пределах уровня детализации.

КОНЦЕПТУАЛЬНЫЙ УРОВЕНЬ (3)

Для концептуального представления служит модель «*Сущность–Связь*» *(ER–модель)*, она графически выражается ER–диаграммами.

Существуют различные модификации представления (нотации) диаграмм.

Представление модели внешне напоминает структуру базы данных и служит для отображения на логическую модель.

ЛОГИЧЕСКИЙ УРОВЕНЬ

Оперирует понятиями *запись*, *компоненты* записи, *связи* между записями.

Соответствующая ему модель называется *логической*, она представляет собой отображение концептуальной модели в среду конкретной СУБД.

Иногда рассматривают не конкретную СУБД, а только ее класс (модель): иерархическую, сетевую, реляционную, постреляционную.

ФИЗИЧЕСКИЙ УРОВЕНЬ

Демонстрирует физическое хранение данных.

На этом уровне используются такие понятия, как *физические блоки*, *файлы*, *хранимые записи*, *указатели*.

Взаимосвязи между хранимыми записями, возникающие в процессе их группировки, и индексные структуры тоже рассматриваются на уровне физической модели.

Знать физическую модель полезно для получения более высокой производительности системы.

АЛЬТЕРНАТИВНАЯ КЛАССИФИКАЦИЯ (1)

Согласно стандарту ANSI / SPAC, архитектура БД представлена трехуровневой моделью с *внешним*, *концептуальным* и *внутренним* уровнями. Но это не модель проектирования, а модель оперирования данными.

Внешний уровень — на языке пользователя описывается структура данных, вид и форма их представления, а также описание операций манипулирования данными. Считается, что для описания предметной области используется несколько внешних моделей.

АЛЬТЕРНАТИВНАЯ КЛАССИФИКАЦИЯ (2)

Концептуальный уровень — наиболее общее представление об информационном содержании предметной области. Определение совпадает с приведенным ранее.

Внутренний уровень — организованная совокупность структурированных данных, отображение концептуальной модели в конкретную среду хранения. Это понятие объединяет логическую и физическую модели.

Альтернативная классификация
Обсуждение (1)

Сравнивая эти два подхода, заключаем, что первый лучше с прагматической точки зрения.

В нем предметная область рассматривается как единое целое, а не как совокупность проектных требований (внешних моделей).

Проектные требования редко можно назвать полноценной моделью, так как любая модель должна давать на каком-то уровне адекватное представление о предметной области.

А эти требования выступают как совокупность представлений разных групп пользователей.

АЛЬТЕРНАТИВНАЯ КЛАССИФИКАЦИЯ
ОБСУЖДЕНИЕ (2)

Такая ситуация возникает тогда, когда аналитик считает, что пользователи формулируют свои знания как локальные модели, совокупность которых и должна составлять требуемую модель.

Реально пользователь часто не в состоянии построить даже локальную информационную модель, без глобальных связей между ними.

Современные направления проектирования отрицают такой подход: он консервирует существующую технологию и не дает выделить цель производства (целостную модель).

КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ

На этом этапе определяются информационные потребности и локальные представления предметной области.

Выявляется роль, назначение, взаимосвязь данных, проводится их спецификация.

Результат этапа — описание объектов данных и их взаимосвязи без указания способа их физической организации.

Структура данных представляется концептуальной схемой.

КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ

СТАДИИ

Первая стадия — первоначальный сбор полной информации о данных предметной области на стадии раннего проектирования, выявление требований.

Вторая стадия — графическое представление полученной информации в виде схемы, которая включает результирующие данные с процессами, их формирующими, а также исходные данные со ссылкой на использующие их процессы. Здесь же выявляются связи между данными.

КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ

ИСТОЧНИКИ ПРОЕКТИРОВАНИЯ (1)

Для получения информации при проектировании данных чаще всего используются письменные источники и эксперты (специалисты).

Письменные источники можно условно разделить на документы и записи. Документы могут быть внешнего и внутреннего характера.

Внешние поступают в организацию или выходят из нее. Выходные документы в большей степени говорят о составе данных.

Внутренние не покидают пределы организации.

Записи отражают текущую деятельность.

КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ

ИСТОЧНИКИ ПРОЕКТИРОВАНИЯ (2)

Эксперты — наиболее существенный источник информации. Ими могут быть как специалисты, разбирающиеся в тонкостях производственного процесса, так и будущие пользователи системы.

Но высокая эффективность использования специалистов может быть достигнута лишь при достаточном уровне взаимодействия.

Источниками информации могут быть и выходные формы существующей программной системы (если она есть), и предлагаемые заказчиком новые формы документов.

КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ

СПОСОБЫ ВЫЯВЛЕНИЯ ТРЕБОВАНИЙ (1)

Наиболее часто применяемые способы:

- *опрос*,
- *семинар*,
- *мозговой штурм*,
- *сценарии*,
- *точки зрения*,
- *этнографический подход*.

Для проектирования данных наиболее подходящие два из них: опрос (самый простой) и этнографический подход (самый сложный).

.

КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ

СПОСОБЫ ВЫЯВЛЕНИЯ ТРЕБОВАНИЙ (2)

Опрос бывает в двух формах: письменной и устной.

Письменный опрос сводится к анкетированию. Составляется перечень вопросов, который сводится в анкету и раздается экспертам. В период проведения опроса эксперт отвечает на вопросы, которые анализируются для определения последующих действий.

Следует сказать, что это хоть и относительно дешевый, но самый худший вид опроса.

.

КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ

СПОСОБЫ ВЫЯВЛЕНИЯ ТРЕБОВАНИЙ (3)

Устный опрос (собеседование) предполагает двустороннее общение, в ходе которого аналитик задает вопросы и получает ответы. Ответы, в свою очередь, могут порождать новые, уточняющие, вопросы.

Тщательная подготовка вопросов и определение регламента приводит к полезному результату.

В отличие от письменного, этот вариант требует большего времени на подготовку и определенного мастерства при проведении.

.

КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ

СПОСОБЫ ВЫЯВЛЕНИЯ ТРЕБОВАНИЙ (4)

Этнографический подход назван по схожести процесса с исследованием этнографом какого-то малоизвестного народа.

Наблюдая производственный процесс изнутри, изучая документы и записи, можно заметить множество существенных деталей, которые трудно обнаружить другим способом.

В частности, полезно проследить движение документов и их модификацию на каждом этапе технологического процесса..

.

КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ

СПОСОБЫ ВЫЯВЛЕНИЯ ТРЕБОВАНИЙ (5)

Во время исследования выявляются особенности принятого документооборота. Регулярное наблюдение за реальной работой позволяет обнаружить «неявные» документы, такие как рукописные производственные журналы.

Однако данный подход довольно сложен: регулярное нахождение постороннего человека, может нервировать работников или им мешать.

Подход лучше сочетать с другими. Полученные знания полезно обсудить на семинаре.

ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ (1)

Роль логического проектирования — отображение концептуальной модели в выбранную модель данных (СУБД).

На этом этапе необходимо определить таблицы и атрибуты, выделить ключи. На ряд атрибутов могут быть наложены ограничения, которые выражаются в функциональных зависимостях.

Для реляционной модели следует так определять отношения, чтобы атрибуты в них функционально полно зависели от ключей и не было транзитивной зависимости атрибутов.

ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ (2)

В результате формируется логическая схема БД в третьей нормальной форме.

Эта схема в процессе проектирования может корректироваться, что нередко приводит к нарушению нормализованности. В этом случае добавляется специальный этап нормализации.

Основное назначение этапа нормализации — получение схемы, эквивалентной данной, без некоторых отрицательных свойств, связанных с функциональными зависимостями.

ЦЕЛОСТНОСТЬ И ДОСТОВЕРНОСТЬ

Правила целостности модели определяют множество непротиворечивых состояний базы данных и множество изменений её состояний.

Ограничения целостности позволяют защитить данные от некорректных изменений.

Статические ограничения отражают множество корректных состояний БД.

Динамические определяют правильные переходы из состояния в состояние.

Контроль реализуется вызовом программ статического или динамического арбитража при модификации данных.

СТАТИЧЕСКИЙ АРБИТРАЖ

Контролирует выполнение основных правил, характерных для реляционных баз данных:

- уникальность кортежей во всех отношениях;
- уникальность и непустоту первичного ключа;
- уникальность возможных ключей;
- зависимость между атрибутами;
- формат атрибутов;
- ограничение значений атрибутов.

ДИНАМИЧЕСКИЙ АРБИТРАЖ (1)

Обеспечивает корректность БД при выполнении операций добавления, удаления и изменения.

Для этого необходимо описание условий, каждое из них может затрагивать одну таблицу или более. Кроме того, динамический арбитраж обеспечивает корректность в особых состояниях БД, которые описаны условными ограничениями.

Процедуры контроля данных по ограничениям могут быть внешними по отношению к БД и встроенными, если СУБД допускает возможность триггеров и хранимых процедур.

ДИНАМИЧЕСКИЙ АРБИТРАЖ (2)

Триггер — процедура, связанная с одной таблицей базы данных. Его роль — обеспечение контроля добавления, удаления или изменения записи.

Выполнение контроля может потребовать обращения к другим таблицам, тогда работают их триггеры, возникает каскад триггеров. Для обеспечения корректности работы нередко используют механизм транзакций.

Хранимые процедуры определяются для БД в целом. Они служат для реализации однотипной деятельности и обеспечивают уменьшение программного кода и сокращение транзакции.

БЛОКИРОВКИ (1)

Уровни изоляции при чтении:

- грязное: читается не зафиксированная ранее запись;
- невоспроизводимое: при повторном чтении выясняется, что запись модифицирована или удалена;
- фантомное: при повторном чтении обнаруживаются новые записи.

Виды чтения:

- незавершённое: читается всё;
- завершённое: читается только зафиксированное;
- воспроизводимое: блокирование ресурсов (пессимистическая блокировка);
- сериализуемое: двухфазная блокировка (фаза нарастания и фаза сжатия) или общая фиксация цепочки.

БЛОКИРОВКИ (2)

Соответствие уровня изоляции и видов чтения

	незавер-шённое	завершён-ное	воспроиз-водимое	сериализу-емое
грязное	+	–	–	–
невоспро-изводимое	+	+	–	–
фантомное	+	+	+	–

ТЕХНОЛОГИИ БЛОКИРОВОК (1)

Оптимистическая блокировка:

- читать элемент;
- сохранить копию;
- изменить элемент;
- блокировать изменяемый элемент на чтение;
- повторно читать изменяемый элемент;
- сравнить с копией;
- если совпадения нет – сообщить о невозможности изменения;
- если совпадение есть – записать изменённый элемент на место считанного;
- снять блокировку.

ТЕХНОЛОГИИ БЛОКИРОВОК (2)

Пессимистическая блокировка:

- блокировать изменяемый элемент на чтение и запись;
- читать элемент;
- изменить элемент;
- записать изменённый элемент на место считанного;
- снять блокировку.

Режим отображения изменяемых записей:

- немедленно после изменения;
- только после повторного считывания;
- в фиксированные моменты времени или через заданные временные интервалы.

ТРАНЗАКЦИИ (1)

Классическая транзакция — последовательность операций изменения БД или выборки, которая воспринимается СУБД как атомарное действие.

Дисциплина транзакций включает различные функции для поддержки приложений, основанных на коммуникациях.

Системы обработки транзакций охватывают базы данных, сети, операционные системы. Понятие обработки транзакций применимо к любой компьютерной среде, особенно в средах с масштабными центрами обработки информации.

ТРАНЗАКЦИИ (2)

Транзакция объединяет несколько действий, которые рассматриваются как единое целое, поэтому должно быть обозначено начало транзакции и ее завершение, которое может быть как успешным, так и неуспешным.

В первом случае транзакция фиксируется, во втором механизм транзакций обязан вернуть базу данных в исходное состояние — откатить транзакцию.

ТРАНЗАКЦИИ (3)
ПРИНЦИПЫ (ACID)

атомарность (*Atomicity*) — выполнение действий по принципу «все или ничего»: транзакция либо фиксируется, либо откатывается;

целостность (*Consistency*) — корректные преобразования состояний системы: база данных переводится из одного корректного состояния в другое;

изолированность (*Isolation*) — обеспечение невидимости данных до окончания фиксации;

долговременное сохранение (*Durability*) — сохранение результатов зафиксированной транзакции, в том числе при аппаратных или программных сбоях.

ПЛОСКИЕ ТРАНЗАКЦИИ

Плоские транзакции обладают единственным уровнем управления для произвольного числа элементарных действий. Согласно правилам обработки, транзакция откатывается, если хотя бы один компонент не завершается.

Для распределенной транзакции, которая связывает данные в удаленных узлах сети, эта модель неприемлема: вероятность отказа велика, транзакцию приходится повторять.

Модификация, основанная на контрольных точках, размечает поток вычислений. При откате работа повторяется с этой точки.

МНОГОЗВЕННЫЕ ТРАНЗАКЦИИ

Ключевое свойство сложной модели транзакции – возможность разбивать полную транзакцию на компоненты – субтранзакции, которые могут перезапускаться при сбоях, обрабатываться синхронно или асинхронно с другими субтранзакциями, подчиняться «главной транзакции», которая может аварийно завершить каждую субтранзакцию, даже если та закончила свою часть работы благополучно.

В модели многозвенных транзакций каждый этап вычислений фиксируется как субтранзакция, она же и откатывается при неудаче.

Если все субтранзакции зафиксированы, фиксируется транзакция в целом.

Только в этом случае она считается завешенной.

ВЛОЖЕННЫЕ ТРАНЗАКЦИИ

Вложенная транзакция – иерархия транзакций, управляемая транзакцией верхнего уровня.

Существуют три правила управления:

- *фиксация*: для каждой субтранзакции только фиксация делает результаты видимыми для родительского уровня;
- *откат*: откат транзакции ведет к откату всей иерархии;
- *видимость*: родительской транзакции доступны результаты дочерней после их фиксации, а дочерней результаты родительской — всегда, соседним одноуровневым субтранзакциям результаты друг друга недоступны.

Последнее правило позволяет выполнять субтранзакции параллельно.

ПРОТОКОЛ ДВУХФАЗНОЙ ФИКСАЦИИ

1. Если необходима фиксация всех транзакций, главная направляет им запрос на подготовку к фиксации. Каждая транзакция фиксирует во внешней памяти содержание буфера журнала изменений без разблокировки.
2. Если всё благополучно – отправляется команда на фиксацию. При неготовности хотя бы одной субтранзакции выполняется откат.

ПРОТОКОЛ ТРЁХФАЗНОЙ ФИКСАЦИИ

1. Если необходима фиксация всех транзакций, главная направляет им запрос на подготовку к фиксации. Каждая транзакция фиксирует во внешней памяти содержание буфера журнала изменений без разблокировки.
2. Если все субтранзакции подтвердили готовность – отправляется команда на предварительную фиксацию (precommit).
3. Если все субтранзакции выполнили предварительную фиксацию – отправляется команда на окончательную фиксацию (commit).

ТРАНЗАКЦИИ (4)

Существуют способы сочетаний моделей транзакций для поддержания функционирования сложных сред. Важно правильно выбрать одну из моделей с учетом специфики предметной области.

Так, вложенные транзакции представляют весьма гибкие средства управления субтранзакциями, но они сложны в реализации. В ряде случаев достаточно многозвенной модели, если для данной организации не характерна асинхронная обработка субтранзакций.

КОНТРОЛЬ ПОЛНОМОЧИЙ
(РАЗГРАНИЧЕНИЕ ПРАВ ДОСТУПА)

Служит защитой от несанкционированного доступа. Чаще всего используется система паролей, отражающая статусы пользователей. Обычно рассматриваются право на вход в систему, право доступа к данным для чтения и право доступа для модификации.

Рассматривается возможность манипуляции с данными: изменения, удаления и добавления.

При обмене через каналы связи производится идентификация и аутентификация источника (приемника) данных.

СРЕДСТВА СОЗДАНИЯ МОДЕЛИ

Создание ER-модели обычно сопровождается ее графическим представлением. Нотации ER-диаграмм поддерживаются средствами проектирования (CASE-средствами).

Наиболее удачное средство – ERwin. Аналогичных свободно распространяемых CASE-средств не много, например, Ferret или MySQL Workbench.

В некоторых случаях подобные средства включаются в инструментальную среду создания баз данных.

CASE-СРЕДСТВО ERWIN

Согласно стандарту IDEF1X, в ERwin поддерживаются диаграммы двух уровней: логического (концептуального) и физического (логического).

Основные элементы модели — сущности, их атрибуты, домены, связи, индексы.

Среди атрибутов выделяются первичные ключи. Кроме них, используются альтернативные ключи и ключи поиска. Внешние ключи формируются автоматически при определении связей.

МОДЕЛЬ ДАННЫХ В ERWIN

Диаграмма сущность-связь включает сущности и связи между ними, отражающие основные понятия предметной области. Диаграмма может включать связи «многие ко многим» и не содержать описание ключей.

Модель данных, основанная на ключах — более подробное представление данных. Включает описание всех сущностей и первичных ключей.

Полная атрибутная модель — наиболее детальное представление структуры данных: она включает все сущности, атрибуты и связи.

СУЩНОСТИ

Построение модели данных предполагает определение сущностей и атрибутов: определяется, какая информация соответствует конкретной сущности и как она представляется атрибутами.

Сущность именуется существительным в единственном числе с однозначным смыслом, «технические» наименования не рекомендуются.

Атрибут должен иметь однозначный смысл. Его полезно снабдить дополнительным описанием.

КЛЮЧИ

Ключи в модели обеспечивают доступ к данным, но роли их различны.

Первичный ключ (*primary key*) однозначно идентифицирует экземпляр сущности.

Претенденты на роль первичного ключа называются *возможными* или *потенциальными* ключами (*candidate key*). Если возможных ключей больше одного, один из них объявляется первичным, а остальные — *альтернативными*.

Внешние ключи (*Foreign Key*) служат для связи родительской сущности и дочерней.

Связи (1)

В стандарте IDEF1X определяются следующие виды связи:

- идентифицирующая;
- неидентифицирующая;
- неспецифическая;
- категориальная.

Связи двух первых типов устанавливаются между независимой и зависимой сущностями. Сильная зависимость определяется *идентифицирующей* связью, а слабая — *неидентифицирующей*.

Связи (2)

Мощность связи (*Cardinality*) служит для обозначения отношения числа экземпляров родительской сущности к числу экземпляров дочерней.

Имя связи (*Verb Phrase*) — глагольная фраза, характеризующая отношение между родительской сущностью и дочерней.

ФИЗИЧЕСКАЯ МОДЕЛЬ

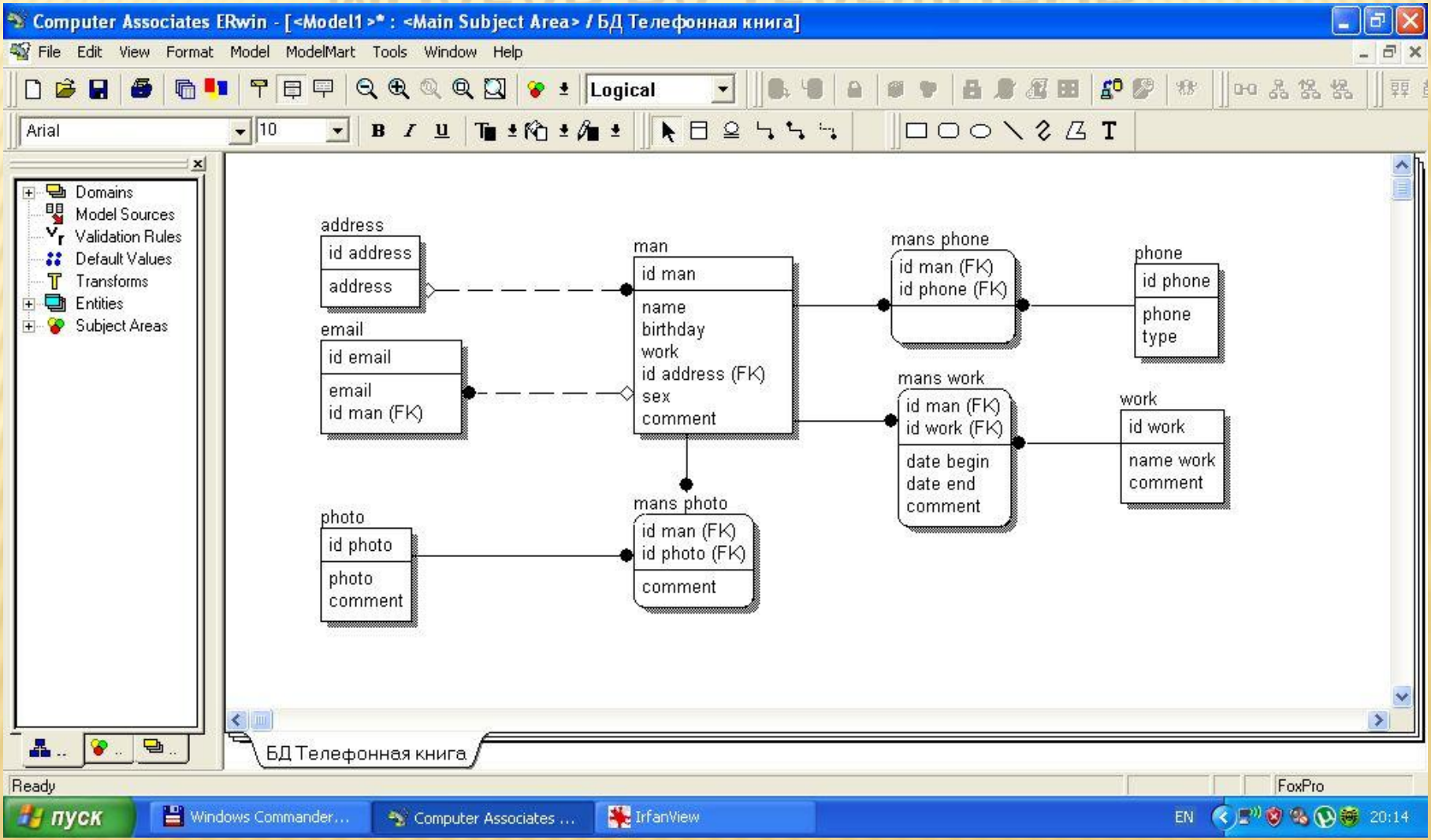
Здесь выбирается конкретная СУБД с её типами данных. Кроме того, задаются индексы и хранимые процедуры.

В этой модели сущности называются таблицами, атрибуты — столбцами.

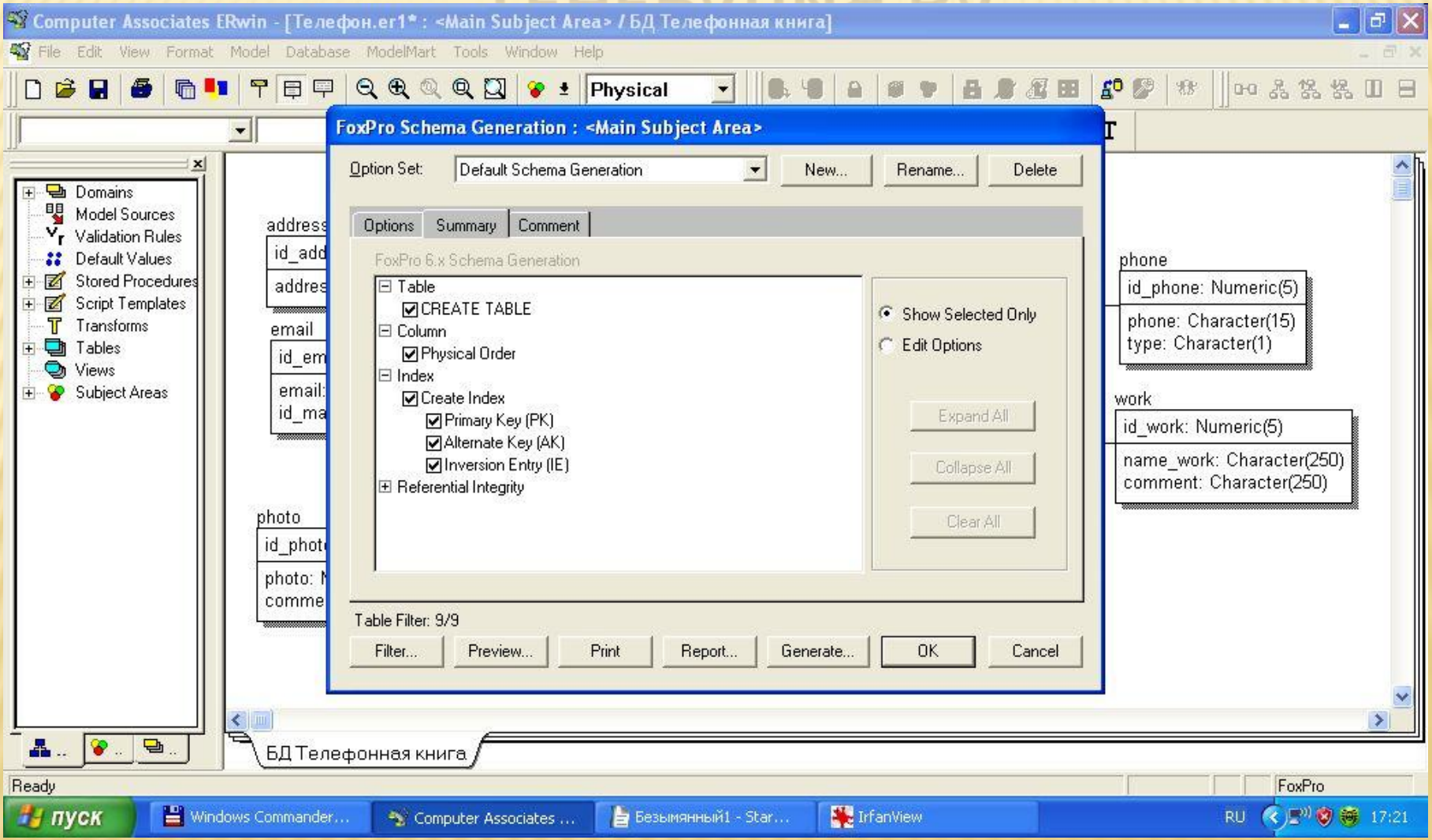
Завершающая работа — автоматическая генерация пустой базы данных по разработанной модели.

База данных проверяется, заполняется тестовыми данными и тестируется. Обычно для создания работоспособной базы данных нужно несколько итераций, в которых модель изменяется.

МОДЕЛЬ БД ТЕЛЕФОНОВ



ГЕНЕРАЦИЯ БД



ГЕНЕРАЦИЯ ТАБЛИЦЫ АБОНЕНТА

```
CREATE TABLE man (
        name                    Character(20) NULL,
        id_man                  Numeric(5) NULL,
        birthday                Date NULL,
        id_address              Numeric(5) NULL,
        sex                     Character(1) NULL,
        comment                 Character(250) NULL
)
```

ГЕНЕРАЦИЯ ТАБЛИЦЫ ТЕЛЕФОНА

```
CREATE TABLE phone (
        id_phone                Numeric(5) NULL,
        phone                   Character(15) NULL,
        type                    Character(1) NULL
)

CREATE TABLE mans_phone (
        id_man                  Numeric(5) NOT NULL,
        id_phone                Numeric(5) NOT NULL
)
```

ЛИТЕРАТУРА

1. *Дейт К.* Введение в системы баз данных.
2. *Диго С. М.* Базы данных: проектирование и использование
3. *Конноли Т. и др*. Базы данных. Проектирование, реализация и сопровождение. Теория и практика..
4. *Кренке Д.* Теория и практика построения баз данных.
5. *Маклаков С. В.* Erwin и BPwin. Case-средства разработки информационных систем..
6. *Марков А. С.*, *Лисовский К. Ю*. Базы данных. Введение в теорию и методологию.
7. *Мейер Д.* Теория реляционных баз данных.
8. *Тиори Т.*, *Фрай Дж.* Проектирование структур баз данных.