Учебный курс по SQL

Программисты делятся на две категории: первая - это те, которые запустили UPDATE без условия WHERE, и вторая – те, которым предстоит ее запустить.

Обзор курса

- Курс предназначен для быстрого изучения основ Структурного Языка Запросов (SQL). В нем будут даны формальные определения основных операторов языка, описаны как главные компоненты SQL, так и его расширения (на примере разработанного фирмой Sybase диалекта Transact-SQL).
- Отдельно будут рассмотрены вопросы приемы оптимизации программного кода.
- По окончании курса Вы должны быть способны применить полученные знания при разработке реальных деловых приложений.

Список изучаемых разделов

1.Введение

2.Выборка информации из базы данных

4 часа

3.Сортировка и методы отбора

4.Основные функции

4 часа

5.Группировка данных и агрегатные функции

6.Объединение таблиц и подзапросы

4 часа

Разделы, изучаемые в курсе «FastTrack» - 24 часа

7.Создание и управление таблицами

8.Ограничения и индексы. Представления

4 часов

9.Манипулирование данными

10.Транзакции и блокировки

8 часов

11.Переменные, хранимые процедуры, курсоры и триггеры.

12.Оптимизация и настройка производительности. Основные ошибки и как их избежать

16 часов

Разделы, изучаемые в курсе «Programming» - 16 часов

Соглашения

- Соглашения принятые в тексте.
  - Команды SQL, функции набраны в верхнем регистре

    SELECT * FROM syslogins
  - Переменные, имена пользователей, имена столбцов набраны в нижнем регистре

    … name FROM …
  - Англоязычные термины набраны в нижнем регистре жирным шрифтом

    foreign key
  - Определения отдельных понятий, название книг, ссылки на документацию и названия курсов набраны курсивом

Соглашения

- Соглашения принятые в программном коде
  - Команды SQL, функции набраны в верхнем регистре жирным шрифтом

    SELECT * FROM …
  - Переменные, названия функций и таблиц, имена пользователей, имена столбцов набраны в нижнем регистре

    … name **FROM** syslogins

Регламент

Порядок проведения занятий, лабораторных и контрольных работ.

1.Введение

- Цели
- Реляционная модель баз данных
- Определение реляционной СУБД и реляционные операции
- Язык SQL и базовые SQL предложения
- Модели данных.Объекты
- Методы проектирования
- Итог

Цели

После окончания введения вы должны:

- Настроится на рабочий лад
- Уметь умно рассуждать о теоретических и физических аспектах реляционных СУБД

Концепция реляционной модели

- В 1970 году доктор И.Ф. Кодд предложил использовать реляционный подход к управлению базами данных, основанный на математической модели, использующей методы реляционной алгебры и реляционного исчисления.
- Реляционная модель отвечает следующему неформальному определению, которое дал С. Дейт:
  1. Вся информация в базе данных представлена в виде набора таблиц(отношений)
  2. Для работы с таблицами она поддерживает набор реляционных операторов - выбора, проектирования и объединения
  3. Обеспечивает механизм целостности и непротиворечивости данных

Реляционная модель

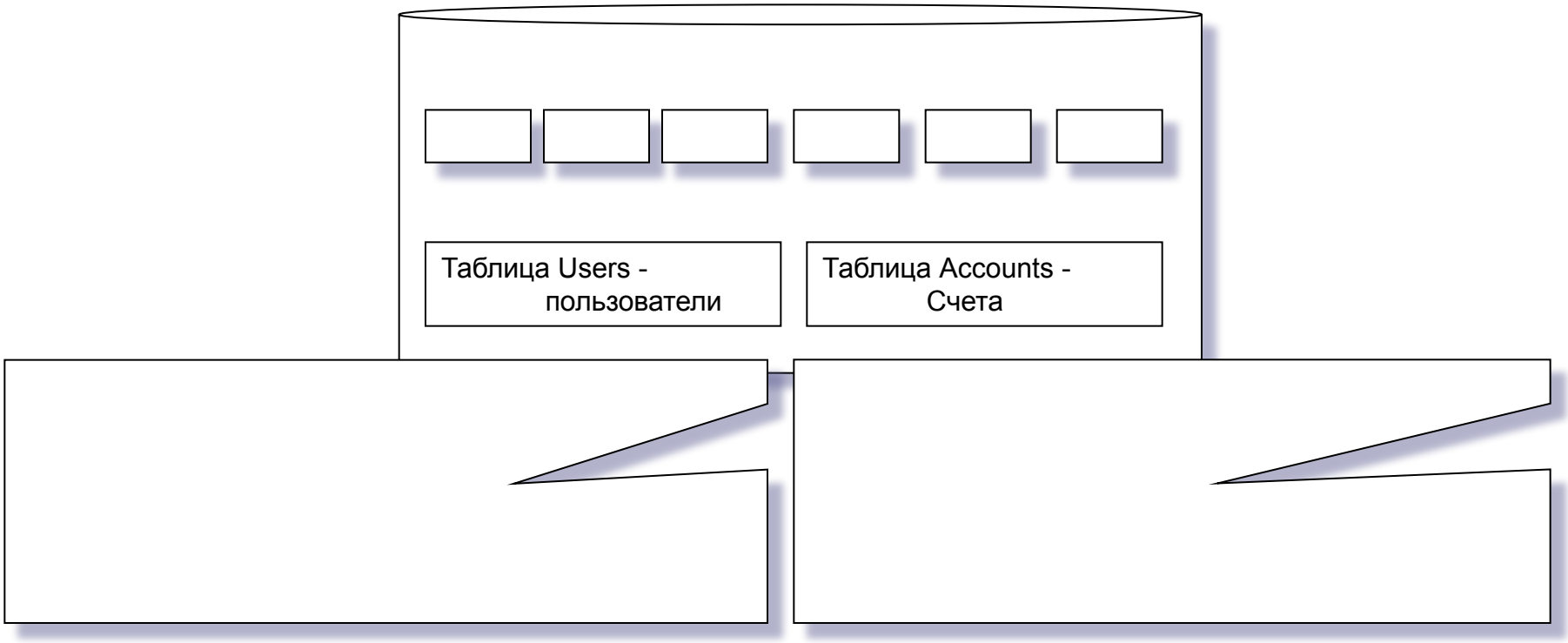
Принципы реляционной модели были впервые разработаны доктором И.Ф. Коддом в июне 1970 года в работе «A relational Model of Data for Large Shared Data Banks».В этой работе Кодд предложил использовать реляционную модель для СУБД.

Наиболее популярными в то время были сетевые модели и модели, использующие структуру плоских файлов. Кодд, разработал список критериев, которым должна удовлетворять реляционная модель. Этот список, часто ошибочно называемый «12 правилами Кодда», состоит из 13 правил и используется совместно с более общим определением Дейта.

Информационное правило (12 правил Кодда)

Вся информация в реляционной базе данных (включая таблицу и названия(имена) столбца) представляется явно как значения в таблицах.

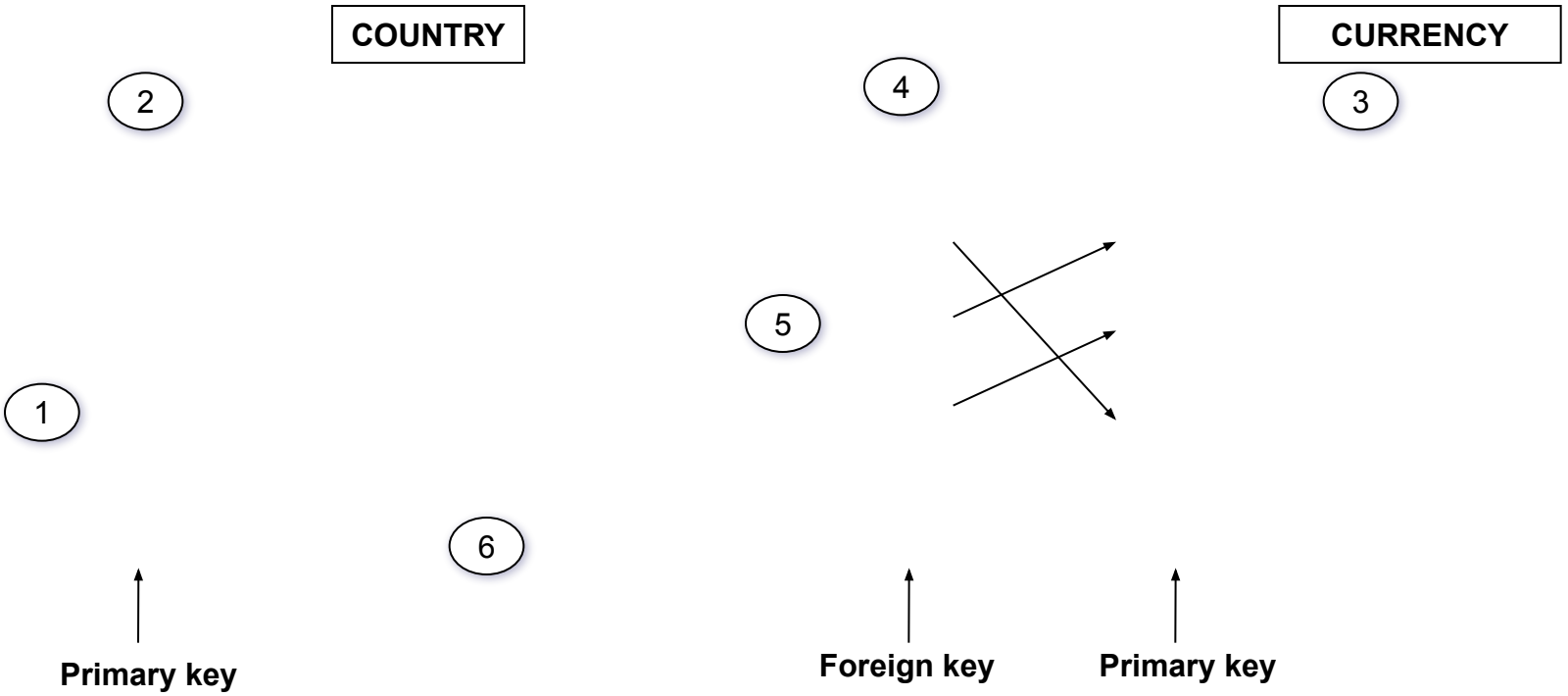
Каждая **таблица (отношение, relation**) состоит из строк и столбцов. Каждая строка имеет отдельный объект или **сущность(entity)** – человека, компанию, счет. Каждый столбец описывает одну характеристику объекта – имя человека или его адрес, тел.номер компании или название счета.



Гарантированный доступ (12 правил Кодда)

Используя комбинацию имени таблицы, первичного ключа и имени столбца гарантируется доступ к любому значению.

Пара слов о принятой терминологии.



Терминология

Реляционная база данных представляет собой множество связанных таблиц. *Таблица* - это базовая структурная единица СУБД. Все данные представляются в табличном формате – другого способа просмотреть информацию в базе нет. Каждая таблица состоит из столбцов и строк.

На предыдущем слайде представлены две таблицы – справочники стран и валют.

- Отдельная *строка* (*кортеж*) описывает отдельный *объект или сущность* (***entity***).Каждая строка в таблице может быть и должна единственным образом идентифицироваться по одному из своих уникальных значений. Порядок, в котором строки помещаются, в общем случае, значения не имеет.
- *Столбец (поле, атрибут)* описывает однотипный набор свойств множества объектов. В данном примере, это код страны. Этот поле является *первичным ключом*.
- Столбец, который не является ключевым полем.
- Столбец, который представляет собой *внешний ключ*(***foreign key***).
- Каждый элемент данных, или *поле (value)*, определяется пересечением строки и столбца таблицы. Чтобы найти требуемый элемент данных, необходимо знать имя содержащей его таблицы, столбец и значение его *первичного ключа* (***primary key***), или уникального идентификатора.
- Поле может и не иметь значения. В таком случае говорят, что поле содержит “NULL”.Работу с NULL значениями мы будем рассматривать в последующих разделах.

Поскольку данные о различных объектах содержатся в разных таблицах, часто возникает потребность в объединении информации из разных таблиц. СУБД позволяет связывать данные в разных таблицах с помощью механизма *внешних ключей*. *Внешний ключ* (***foreign key***) – это столбец или набор столбцов, которые ссылаются на первичный ключ в той же или другой таблице.

Возможность связывать данные из разных таблиц упрощает проектирование, делает его более наглядным и повышает управляемость.

Свойства первичных и внешних ключей.

- Первичный ключ не допускает повторяющихся или NULL значений
- Будучи создан, первичный ключ обычно уже не меняется.
- Внешний ключ – есть логическое понятие, его физические реализации могут быть различны.
- Внешний ключ должен соответствовать значениям существующего первичного ключа или уникального индекса, или должен быть NULL.
- Невозможно определить внешний ключ не связав его с первичным ключом или уникальным индексом.

Логическая независимость (12 правил Кодда)

Logical data independence означает, что изменение взаимосвязей между таблицами, столбцами и строками не влияет на правильное функционирование программных приложений.

Физическая независимость (12 правил Кодда)

Physical data independence - с точки зрения пользователя, представления данных не зависит от способа их физического хранения. Как следствие этого физическое перемещение данных никоим образом не может повлиять на логическую структуру СУБД и восприятие данных пользователем.

Язык данных (12 правил Кодда)

Используется по крайней мере один язык высокого уровня, обеспечивающий определение данных, манипулирование ими, правила целостности, авторизацию, и транзакции. В мире коммерческих СУБД такой язык получил название SQL. Он используется для **манипуляции с данными (*data manipulation*), - выборки и модификации, определения данных (*data definition*) и администрирования данных (*data administration*).**

Любая операция по выборке, модификации, определению или администрированию выполняется с помощью **оператора (*statement*)** или **команды (*command*)** SQL.И ANSI, и ISO приняли SQL в качестве стандарта для реляционных СУБД.

SQL операторы

Реляционность (12 правил Кодда)

Реляционная СУБД должна поддерживать основные реляционные операции.

Существует три реляционные операции по выборке данных – проектирование, выбор (иногда называемый ограничением (restriction)) и объединение, которые позволяют строго указать системе, какие данные Вы хотите увидеть. Операция проектирования выбирает столбцы, операции выбора – строки, а операция объединения собирает вместе данные из связанных таблиц.

Все эти три операции записываются с помощью одного оператора – **SELECT.**

Учитывая правила логической и физической независимости, язык SQL можно рассматривать как непроцедурный язык программирования, так как он описывает то, ЧТО вы хотите получить, а не КАК вы хотите это сделать.

Системная поддержка NULL (12 правил Кодда)

Поддержка неизвестных значений (NULL) – различаются неизвестные, нулевые значения и пропуски данных.

В реальности мы редко обладаем всей полнотой информации. Для поддержания целостности данных и определения наличие неопределенной информации в таблицах используют понятие «NULL».

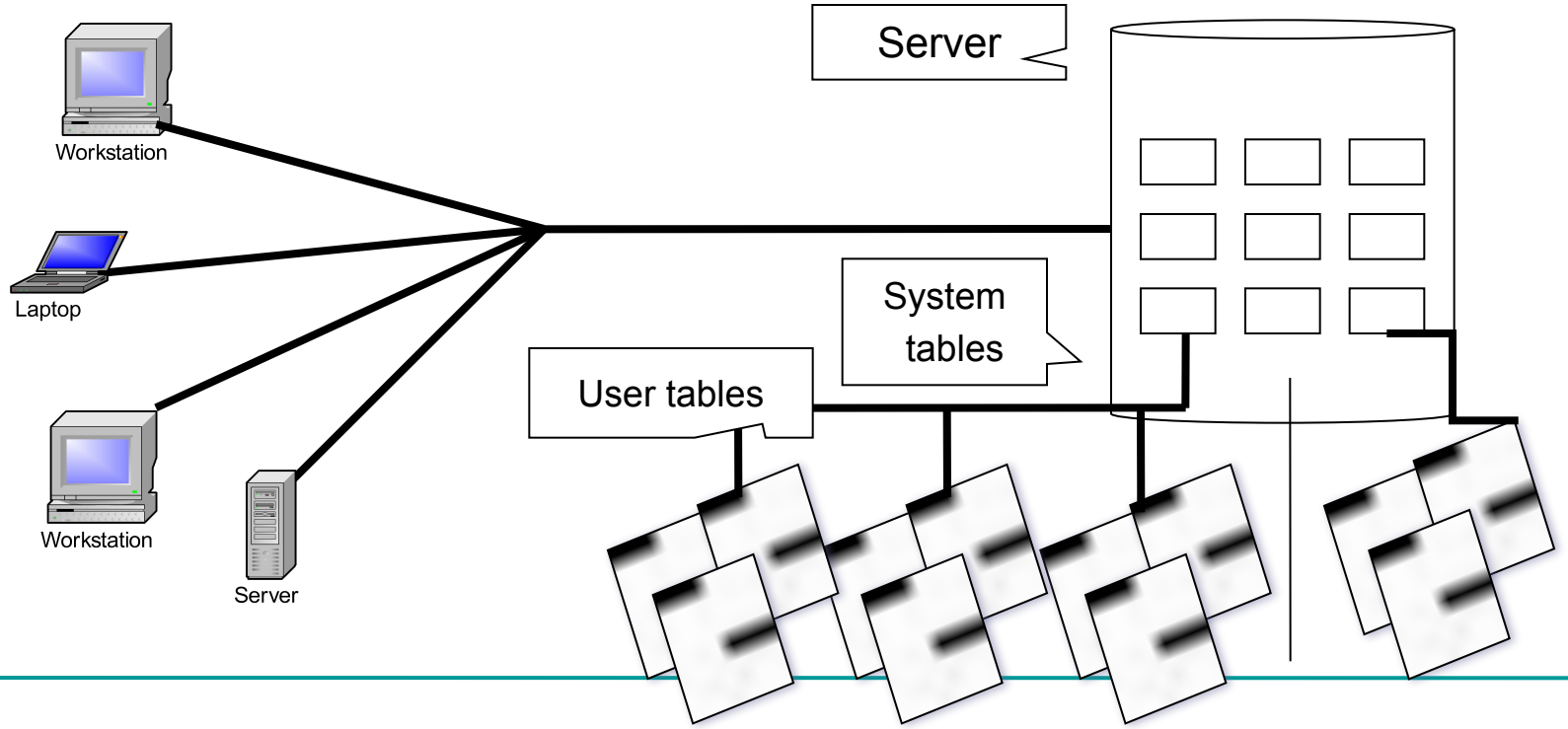
NULL – не означает пустое поле или обычный арифметический нуль. Он отражает тот факт, что значение неизвестно. Существенно, что использование NULL инициирует переход с двухзначной логики (да/нет, что-то/ничего) на трехзначную (да/нет/может быть или что-то/ничего/не уверен).

Системный каталог (12 правил Кодда)

Реляционный каталог - описание базы данных представляется на логическом уровне как таблицы и доступно для интерактивных запросов к словарю данных

В реляционных СУБД существует два типа таблиц – пользовательские и системные таблицы.

Пользовательские таблицы содержат информацию, для которой собственно и создавалась база данных – данные по сделкам, заказам, персоналу и т.д. Системные таблицы – содержат описание самой базы данных.



Виртуальные таблицы (12 правил Кодда)

Поддержка механизма альтернативного способа просмотра и модификации данных.

View – можно рассматривать, как перемещаемую по таблицам рамку, через которую можно увидеть только необходимую вам часть информации. *View* можно получить из одной или нескольких таблиц, включая другие *View,* и используя любые операции выбора, проектирования и объединения.

View – в отличие от базовых таблиц, физически не хранятся в СУБД.

View – это не копия некоторых данных помещенных в другую таблицу. Когда изменяются данные в View , то тем самым изменяются данные в базовых таблицах.

Множественные операции (12 правил Кодда)

Отвечая этому правилу, СУБД должна поддерживать не только множественные выборки, но также множественные операции вставки, модификации и удаления.

Целостность данных(12 правил Кодда)

Несогласованность или противоречивость данных может возникать по многим причинам:

вследствие сбоя системы, проблемы с аппаратным обеспечением, ошибки в системном программном обеспечении или логических ошибок в пользовательских приложениях.

Для предотвращения подобного типа ошибок реляционные СУБД предлагают ряд механизмов и технологий.

Один из таких механизмов, обычно называемый управлением транзакциями (**transaction management**), гарантирует, что определенный пользователем набор команд SQL будет или выполнен до конца, или будет полностью отменен.

Другой тип обеспечения целостности данных, называемой объектной целостностью (**entity integrity**), связан с корректным проектированием базы данных.В частности, объектная целостность требует, чтобы ни один первичный ключ не имел значения NULL.

Третий тип целостности, называемый ссылочной целостностью, означает непротиворечивость между частями информации, повторяющимися в разных таблицах.

Независимость распределения (12 правил Кодда)

Работа пользовательских приложений логически не зависит ни первоначального ни от последующего распределения данных.

Устойчивость (непротиворечивость) (12 правил Кодда)

Язык баз данных должен предоставлять возможность определения правил целостности. Они должны быть сохранены в репозитарии, и не существует способа обойти правила целостности, определенные через язык баз данных, используя языки низшего уровня.

Модели данных

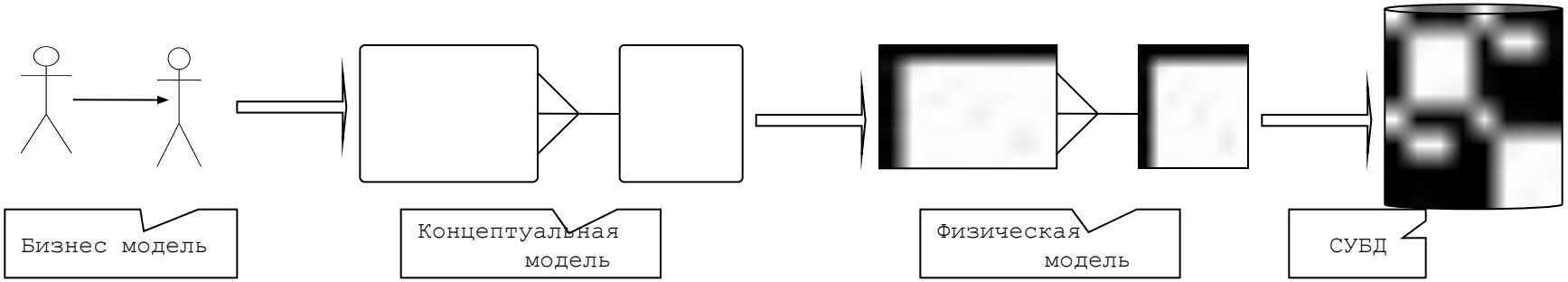
«Создать нужную структуру базы данных (по крайней мере в простых случаях) зачастую проще, чем строго сформулировать, какой она должна быть»

К.Дж. Дейт.

Модель – краеугольный камень дизайна приложений, использующих СУБД. Модель помогает связать, описать, конкретизировать определенные концепции и идеи, а потом исследовать и проанализировать их, имитируя реальные процессы.

Хорошая модель, это в первую очередь, «прозрачная» структура. Прозрачная структура должна максимально упрощать взаимодействие с базой данных, обеспечивать представление информации в понятном и общепринятом формате, обеспечивать непротиворечивость данных и механизмы анализа и предотвращения ошибок.

Этапы большого пути



Создание модели при проектирование системы обычно состоит из следующих фаз:

- Описание бизнес-процессов и создание бизнес-спецификаций, являющихся прямым отражением преставления в человеческом мозгу исследуемой концепции (бизнес модель).
- Разделение данных на отдельные сущности (объекты) и построение диаграммы зависимостей между ними (E-R diagram). На этом этапе бизнес-данные отделяются от бизнес-действий. Тогда как, типы взаимодействий между объектами могут изменяться, сами объекты имеет тенденцию оставаться постоянными (концептуальная модель).
- На этапе преобразования концептуальной модели в физическую, построенные логические взаимосвязи между сущностями преобразуются в конкретные механизмы их реализации для данной СУДБ. На этом этапе можно определить низкоуровневые настройки и ограничения модели данных.

Проектирование

Основными методами проектирования является моделирование зависимостей (entity-relationship modeling) и нормализация (normalization).И если построение E-R диаграмм облегчает и упрощает сам процесс моделирования, то нормализация наиболее полезна для проверки созданной структуры.

Возможны четыре вида связей между объектами (степеней отношения):

- **Первый тип** – связь ОДИН-К-ОДНОМУ (1:1): в каждый момент времени каждому экземпляру сущности A соответствует 1 или 0 экземпляров сущности B
- **Второй тип** – связь ОДИН-КО-МНОГИМ (1:M): одному экземпляру сущности A соответствуют 0, 1 или несколько экземпляров сущности B.Так как между двумя сущностями возможны связи в обоих направлениях, то существует еще два типа связи МНОГИЕ-К-ОДНОМУ (M:1) и МНОГИЕ-КО-МНОГИМ (M:N).

Каждый такой тип связи может быть объявлен обязательным (mandatory):

каждому экземпляру сущности A должен (must be) соответствовать 1 или более экземпляров сущности B

Или опциональным:

каждому экземпляру сущности A может (may be) соответствовать 0 или более экземпляров сущности B

Нормализация

Существует набор стандартов проектирования данных, называемых нормальными формами. Общепринятыми считаются пять нормальных форм, хотя их было предложено значительно больше. Создание таблиц в соответствии с этими стандартами называется нормализацией.

На практике, нормализация сводится к разбиению таблицы на две или более, обладающих лучшими свойствами при включении, изменении и удалении данных. Окончательная цель нормализации сводится к получению такого проекта базы данных, в котором *каждое значение появляется лишь в одном месте*, т.е. исключена избыточность информации. Это делается не столько с целью экономии памяти, сколько для исключения возможной противоречивости хранимых данных.

Нормальные формы

- Таблица находится в *первой нормальной форме* (*1НФ*) тогда и только тогда, когда ни одна из ее строк не содержит в любом своем поле более одного значения и ни одно из ее ключевых полей не пусто. Т.е, первая нормальная форма требует, чтобы на любом пересечении строки и столбца находилось одно значение, которое должно быть атомарным.

- Таблица находится во *второй нормальной форме* (*2НФ*), если она удовлетворяет определению 1НФ и все ее поля, не входящие в первичный ключ, связаны полной функциональной зависимостью с первичным ключом. (*Функциональная зависимость*. Поле В таблицы функционально зависит от поля А той же таблицы в том и только в том случае, когда в любой заданный момент времени для каждого из различных значений поля А обязательно существует только одно из различных значений поля В). Т.е., любой не ключевой столбец зависит от всего первичного ключа.

- Таблица находится в *третьей нормальной форме* (*3НФ*), если она удовлетворяет определению 2НФ и ни одно из ее не ключевых полей не зависит функционально от любого другого не ключевого поля.

- Таблица находится в *пятой нормальной форме* (*5НФ*) тогда и только тогда, когда в каждой ее полной декомпозиции все проекции содержат возможный ключ.

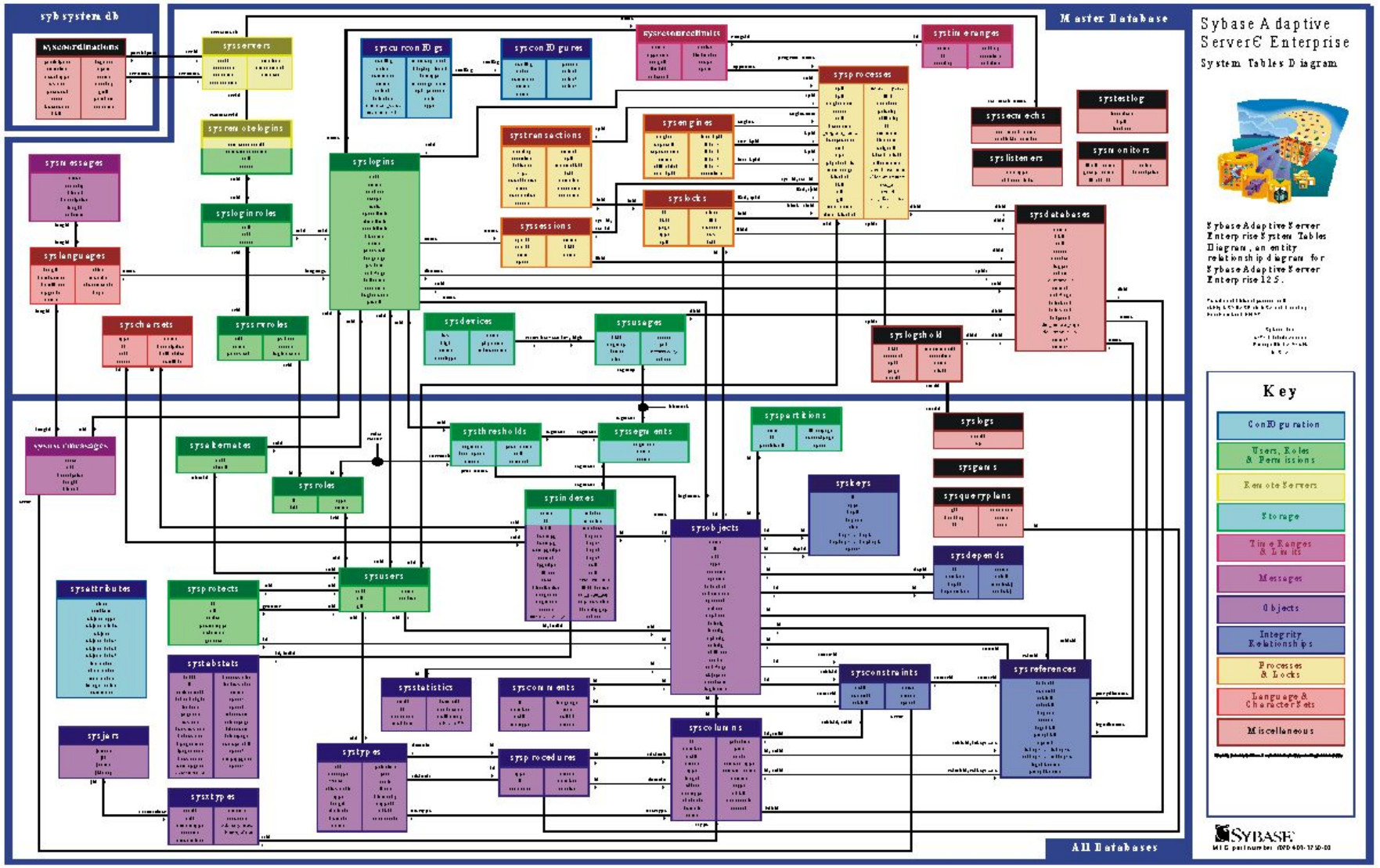
- *Четвертая нормальная форма* (*4НФ*) является частным случаем 5НФ, когда полная декомпозиция должна быть соединением ровно двух проекций.

Итог

- Реляционные СУБД состоят из таблиц (отношений), над которыми определены реляционные операции и ряд дополнительных специфических правил обеспечения логической целостности, определяемая пользователем.

- Sybase ASE - является реляционной СУБД, которая позволяет хранить данные и манипулировать ими с помощью расширения SQL – Transact-SQL.

Таблицы, используемые в этом курсе



Источники дополнительных сведений

- *Transact-SQL® User's Guide. Adaptive Server Enterprise Version 12.5 Document ID: 32300-01-1250-02 Last Revised: May 2001, Sybase, Inc*
- *Reference Manual. Adaptive Server® Enterprise Version 12.5 Document ID: 36271-01-1250-01 Last Revised: June 2001, Sybase, Inc*
- *Performance and Tuning Guide. Adaptive Server Enterprise Version 12.5 Document ID: 33621-01-1250-02 Last Revised: May 2001, Sybase, Inc*
- *Кузнецов С.Д. Введение в стандарты языка баз данных SQL (http://www.citforum.ru/database/sqlbook/index.shtml)*
- *Ryan K. Stephens, Ronald Plew, Bryan Morgan, Jeff Perkins.Teach Yourself SQL in 21 Days, Second Edition Macmillan Computer Publishing.*
- *Мартин Грабер. Понимание SQL.*