КУРСОВОЙ ВЫПОНЯТЬ В СУБД SQL

Обязательные разделы: Постановка задачи, Выбор СУБД (в сравнении), Описание структуры таблиц, Описание поддержки целостности данных (связи), Скрипт на создание стуктуры БД (на SQL), запросы которые приведены в задании

Вариант 6. Разработка базы данных для работников регистратуры поликлиники

В БД должны храниться сведения о больных (ФИО, адрес, диагноз, дата заболевания, номер страхового полиса, название страховой компании), сведения о врачах (ФИО, номер кабинета, номер участка, дни и часы приема), описание болезней (название, симптомы, лекарство).

Работникам регистратуры могут потребоваться следующие сведения:

- адрес, дата заболевания, диагноз данного больного;
- ФИО лечащего врача данного больного;
- номер кабинета, дни и часы приема данного врача;
- список больных, находящихся на лечении у данного врача.

Пользователь может вносить следующие изменения: осуществлять ввод данных о новом больном, удалять из БД информацию об уволенных врачах, редактировать данные о больном.

1. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1.1 Стадии и этапы разработки баз данных

Процесс разработки БД – это итеративный процесс, в ходе которого обычно приходится многократно возвращаться к предыдущим этапам выполнения работы, вносить необходимые изменения и затем заново повторять последующие этапы до достижения необходимого результата.

Процесс разработки базы данных можно разбить на две стадии:

1. Проектирование базы данных.

2. Программная реализация базы данных.

Стадия проектирования БД включает следующие основные этапы:

1. Обследование предметной области.

2. Инфологическое проектирование (разработка инфологической модели предметной области).

3. Даталогическое проектирование.

В БД хранится информация об определенной предметной области.

Предметной областью называется часть реального мира, представляющая интерес для данного исследования (использования) и отражаемая в базе данных.

Для формирования представления о данных, их составе и использовании в конкретных условиях служат *информационные модели* (ИМ). При решении конкретных задач реальная действительность воспроизводится с существенными ограничениями, зависящими от области деятельности, поставленных целей.

Значит, для создания БД надо сначала проанализировать предметную область и создать её информационную модель.

Основой для *анализа предметной области* служат документы, которые ее отражают, и информация, которую можно получить от специалистов этой предметной области в процессе общения с ними.

Для анализа берутся те документы, которые имеют отношение к решаемой задаче. Изучение документов позволяет выявить объекты и их свойства, информацию о которых необходимо хранить в БД.

Из общения со специалистами необходимо извлечь сведения об особенностях предметной области, которые позволяют установить ограничения целостности, зависимости и связи между объектами (субъектами) предметной области.

Модель предметной области может быть описана любым удобным для разработчика способом (словесное описание, набор формул, диаграмма потоков данных и т.п.).

При проектировании баз данных используется диаграмма «сущность–связь» (ER–диаграммы (entity-relation diagram)).

На следующем этапе – этапе **даталогического** проектирования – ER-диаграмма формальным способом преобразуется в схему реляционной базы данных (БД). На основании схемы БД и описания сущностей предметной области составляются таблицы (отношения) базы данных. Потом выполняется нормализация отношений. Это необходимо сделать для того, чтобы исключить нарушения логической целостности данных и повысить, таким образом, надёжность и достоверность данных.

В результате всех этих операций создаётся схема БД – основной документ для базы данных.

Программная реализация базы данных

На этом этапе полученная схема базы данных описывается на языке DDL (Data definition language) – языке определения данных, который поддерживается выбранной СУБД.

Стадия программной реализации БД содержит работы:

1. Создание таблиц.

2. Создание межтабличных связей (для поддержки целостности данных).

3. Разработку процедур обработки данных (написание текста создания вспомогательных объектов базы данных (представления, хранимые процедуры, триггеры и т.д.)).

4. Отладку БД.

5. Тестирование БД (выполнение контрольных примеров).

6. Разработка эксплуатационной документации БД.

Если пользователей базы данных можно разделить на группы по характеру решаемых задач, то для каждой группы создаётся свой набор прав доступа к объектам БД.

1.2. Последовательность проектирования базы данных

Процесс проектирования базы данных включает в себя следующие шаги:

1. Определение задач, стоящих перед базой данных.

2. Сбор и анализ документов, относящихся к исследуемой предметной области.

3. Описание особенностей предметной области, которые позволяют установить зависимости и связи между объектами предметной области.

4. Создание модели предметной области.

5. Определение групп пользователей и перечня задач, стоящих перед каждой группой.

6. Выбор СУБД (системы управления базой данных).

7. Создание логической схемы БД.

8. Создание схем таблиц, определение типов данных атрибутов и ограничений целостности.

9. Нормализация отношений (до третьей или четвёртой нормальной формы).

10. Определение прав доступа пользователей к объектам БД.

11. Написание текста создания основных объектов базы данных на языке SQL в синтаксисе выбранной СУБД (пользователи, таблицы и др.).

12. Написание текста создания вспомогательных объектов базы данных (представления, хранимые процедуры, триггеры, роли и т.д.).

Эти шаги можно объединить в 4 этапа:

1. Инфологическое проектирование (1-5).

2. Выбор системы управления базой данных (СУБД) и других инструментальных программных средств (6).

3. Логическое проектирование БД (7-10).

4. Программная реализация базы данных (11-12).

На сегодняшний день не существует формальных способов моделирования реальности, но инфологический подход закладывает основы методологии проектирования базы данных как модели предметной области.

1.3 Инфологическое проектирование

Цель инфологического проектирования - построение независимой от СУБД информационной структуры путем объединения информационных требований пользователей. Результатом этого этапа является представление информационных требований в виде диаграмм «сущность-связь».

Основными задачами этапа инфологического проектирования являются определение предметной области системы и формирование взгляда на неё с позиций сообщества будущих пользователей БД, т.е. информационно-логической модели предметной области.

Основными конструктивными элементами инфологических моделей являются сущности, связи между ними и их свойства (атрибуты).

Сущность – это реальный объект предметной области, о котором в системе будут накапливаться данные.

Сущности представляют собой объекты, которые пользователи считают важными в моделируемой предметной области. Примерами сущностей могут быть люди, автомобили, дома, книги, компании, штатное расписание.

На диаграммах «сущность – связь» сущности представляют в виде прямоугольника, содержащего внутри название.

Необходимо различать такие понятия, как *тип сущности* и *экземпляр сущности*. Понятие тип сущности относится к набору однородных личностей, предметов, событий или идей, выступающих как целое. Экземпляр сущности относится к конкретной вещи в наборе. Например, типом сущности может быть ГОРОД, а экземпляром – Москва, Иваново и т.д.

Каждая сущность обладает определенными свойствами. Например, у человека есть имя, дата рождения, вес, рост.

Атрибут – поименованная характеристика (свойство) сущности. Его наименование должно быть уникальным для конкретного типа сущности, но может быть одинаковым для различного типа сущностей (например, ЦВЕТ может быть определен для многих сущностей: СОБАКА, АВТОМОБИЛЬ, ДЫМ и т.д.).

Атрибуты используются для определения того, какая информация должна быть собрана о сущности.

Атрибуты на диаграмме «сущность-связь» изображаются в виде овалов, прикрепленных к соответствующей сущности (рис. 1).

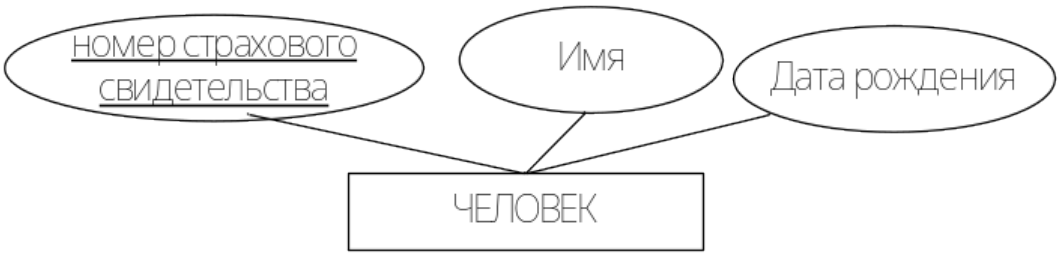

Рисунок. 1. Представление сущности и ее атрибутов

Важно понимать, что сущности нужно в концептуальном плане отделять от атрибутов, которые их описывают, т.к. значения атрибутов могут меняться, в то время как описываемый ими объект остается прежним. Например, у человека может измениться рост, вес, семейное положение, но это будет тот же самый человек.

Абсолютное различие между типами сущностей и атрибутами отсутствует. Атрибут является таковым только в связи с типом сущности. В другом контексте атрибут может выступать как самостоятельная сущность. Например, для автомобильного завода цвет – это только атрибут продукта производства, а для лакокрасочной фабрики цвет – тип сущности.

Ключ – это минимальный набор атрибутов, по значениям которых можно однозначно найти требуемый экземпляр сущности.

Например, для сущности ЧЕЛОВЕК ключом является атрибут Номер страхового свидетельства или набор атрибутов имя, дата рождения. На

диаграммах сущность – связь ключ подчеркивают одинарной сплошной линией (рис. 1).

Часто в качестве ключей применяют так называемые *суррогатные ключи*. Это искусственно сформированные внутренние номера без смысла вне базы данных, которые однозначно определяют элемент сущности. Использование суррогатных ключей предпочтительнее по сравнению с использованием составных ключей, так как уменьшается вероятность нарушения целостности данных из-за ошибок в одном из атрибутов.

Две и более сущности могут быть связаны между собой отношением.

Связь это ассоциирование двух или более сущностей. Если бы назначением базы данных было только хранение отдельных, не связанных между собой данных, то ее структура могла бы быть очень простой. Однако одно из основных требований к организации базы данных – это обеспечение возможности отыскания одних сущностей по значениям других, для чего необходимо установить между ними связи. А так как в реальных базах данных нередко содержатся сотни или даже тысячи сущностей, то теоретически между ними может быть установлено более миллиона связей. Наличие такого множества связей и определяет сложность инфологических моделей.

Связь устанавливается между экземплярами сущностей, а не их типами. Например, для сущностей КЛИЕНТ и ЗАКАЗ связь устанавливается между конкретным клиентом и его заказами. Для связывания экземпляров сущностей используются их уникальные идентификаторы экземпляров сущности (ключи).

Графически связи между двумя сущностями представляется в виде соединяющего их отрезка, дополненного ромбом.

Существуют две важные характеристики связей: показатель кардинальности между сущностями и класс принадлежности сущностей.

Показатель кардинальности – это число экземпляров одной сущности, связанных с одним экземпляром другой сущности.

В зависимости от показателя кардинальности различают следующие типы бинарных связей:

- один – к – одному (1:1) (one-to-one relationship);
- один – ко – многим (1:M) (one-to-many relationship);
- многие – ко – многим (M:M) (many – to – many relationship).

Класс принадлежности – признак, определяющий обязательность участие экземпляров сущности в некоторой связи.

Класс принадлежности называется обязательным, если все экземпляры данной сущности должны участвовать в некоторой связи.

В таблице 1 приведены обозначения связей с различными характеристиками.

Таблица 1

Класс принадлежности	Показатель кардинальности	
	1	*n*
Необязательный	возможно только один	возможно несколько
Обязательный	обязательно только один	по крайней мере только один

Примеры связей с различными показателями кардинальности и классами принадлежности приведены на рис. 2 .

Рисунок 2. Примеры связей

Каждый экземпляр сущности ЖЕНАТЫЙ МУЖЧИНА должен быть обязательно связан с одним экземпляром сущности ЗАМУЖНЯЯ ЖЕНЩИНА, т.е. класс принадлежности с обеих сторон обязательный, а показатель кардинальности – один к одному (1:1). Предполагается, что один ИНСПЕКТОР может контролировать несколько РАБОЧИХ, а каждого РАБОЧЕГО контролирует только один ИНСПЕКТОР. Такая связь будет иметь показатель кардинальности один-ко-многим (1:M). Класс принадлежности с обеих сторон

обязательный, т.к. все ИНСПЕКТОРА занимаются контролем, а все РАБОЧИЕ контролируются. В случае сущностей АВТОМОБИЛЬ и АЗС показатель кардинальности много-ко-многим (М:М). Каждый АВТОМОБИЛЬ может заправиться на разных АЗС, а на одной АЗС могут заправляться разные АВТОМОБИЛИ. Класс принадлежности со стороны сущности АЗС необязательный, т.к. АЗС может быть закрыта на ремонт, следовательно, на этой АЗС не может заправиться ни один автомобиль, т.е. с этой АЗС не будет связан ни один экземпляр сущности АВТОМОБИЛЬ.

1.4. Примеры построения моделей «сущность – связь»

Рассмотрим пример модели данных банка. Сущностями для банка являются СБЕРЕГАТЕЛЬНЫЙ СЧЕТ, ТЕКУЩИЙ СЧЕТ, КЛИЕНТ. Клиент может иметь как текущий, так и сберегательный счет. В общем случае клиент может иметь несколько счетов, а каждым счетом могут пользоваться несколько клиентов. Таким образом, отношения ИМЕЕТ СБЕРЕГАТЕЛЬНЫЙ СЧЕТ и ИМЕЕТ ТЕКУЩИЙ СЧЕТ являются отношениями вида много-ко-многим, причем если существует счет, то он должен принадлежать хотя бы одному клиенту (класс принадлежности со стороны клиента – обязательный, со стороны счета - необязательный). Клиентом банка может быть как физическое, так и юридическое лицо. Добавив к каждой сущности атрибуты, получим модель, представленную на рис. 3.

Рисунок 3. Модель «сущность – связь» структуры данных банка

Модель данных «сущность – связь» может строиться как на основе опроса руководителей и специалистов отделов предприятия, так и на основе существующих отчетов. При разработке БД на основе документации, принятой на предприятии, сущности, атрибуты и связи между ними будут отражать существующую систему учета. Если необходимо учитывать и какие-либо другие данные, не отраженные в документах, то после разработки модели ее нужно будет корректировать и дополнять с учетом пожеланий заказчиков.

Рассмотрим пример построения модели «сущность – связь» торговой фирмы на основе накладной. Пример накладной приведен в таблице 2.

Таблица 2

Накладная № 33 Получатель Фирма АВС, г. Владимир, тел. 999999 Код получателя 999 Поставщик Фирма XXX Дата 01.01.2018				
Наименование	Номенклатурный номер	Количество	Цена	Сумма
Транзистор КТ601АМ	7125692	100	2.00	200.00
Микросхема 133ИЕ8	7504870	20	5.00	100.00
Налог				60.00
ИТОГО				360.00

Из приведенного документа можно вывести следующие сущности: НАКЛАДНАЯ, ПОЛУЧАТЕЛЬ, ТОВАР. Отношение между сущностями ПОЛУЧАТЕЛЬ и НАКЛАДНАЯ имеет показатель кардинальности один-ко-многим, так как каждая накладная оформляется одному получателю, но данному получателю могут быть оформлены несколько накладных. Класс принадлежности сущностей ПОЛУЧАТЕЛЬ и НАКЛАДНАЯ обязательный, т.к. в накладной обязательно указывается получатель товара.

Отношение ВКЛЮЧАЕТ между объектами ТОВАР и НАКЛАДНАЯ имеет показатель кардинальности много-ко-многим, так как накладная может содержать несколько товаров, а товар может встречаться в нескольких накладных, при этом одной накладной должен соответствовать хотя бы один товар (т.е. со стороны товара класс принадлежности обязательный, со стороны накладной -

необязательный). Если необходимо контролировать оплату поставленного по накладным товара, то необходимо добавить сущность ОПЛАТА с атрибутами НОМЕР ДОКУМЕНТА, ВИД ОПЛАТЫ и ДАТА. Показатель кардинальности отношения ОПЛАЧЕНА между объектами НАКЛАДНАЯ и ОПЛАТА равна один-к-одному и класс принадлежности является обязательной (подразумевается, что при выписке накладной должны также оформляться документы об оплате). Модель данных приведена на рис. 4.

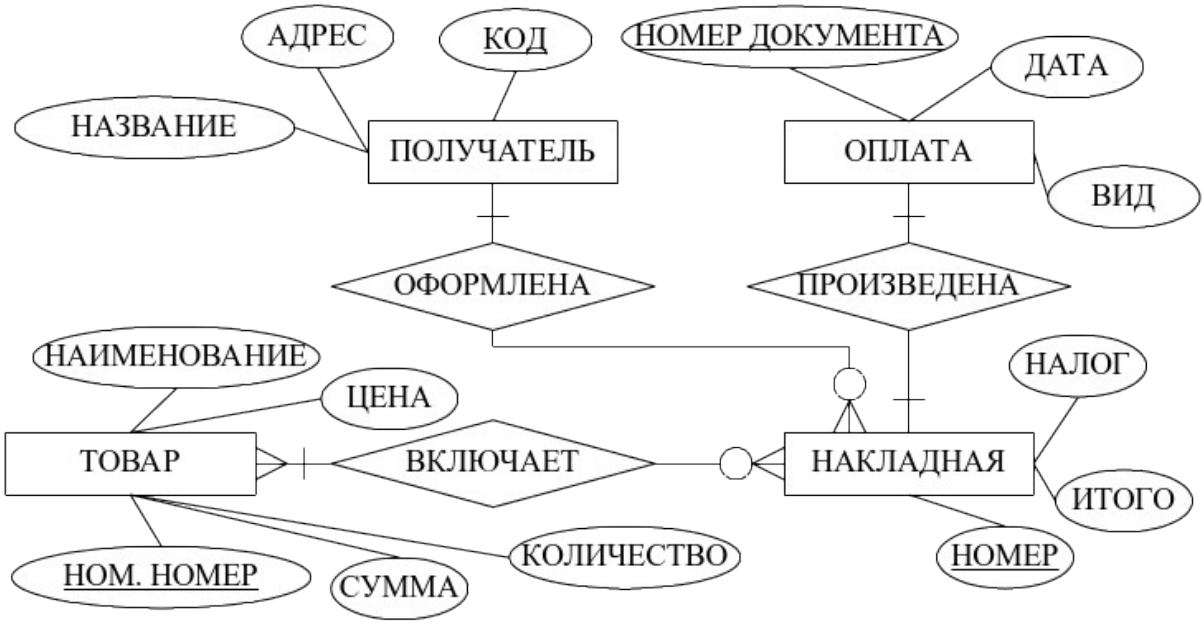

Рисунок 4. Модель «сущность – связь» торговой фирмы на основе накладной

1.5 Выбор СУБД и других программных средств

Выбор СУБД осуществляется на основании таких критериев, как тип модели данных и её адекватность потребностям рассматриваемой предметной области; характеристики производительности; набор функциональных возможностей; удобство и надежность СУБД в эксплуатации; стоимость СУБД и дополнительного программного обеспечения [1].

1.6 Логическое проектирование БД

1.6.1 Разработка логической структуры БД

На этапе логического проектирования разрабатывается логическая структура БД. Для реляционной модели существуют формальные правила, которые позволяют преобразовать инфологическую модель в виде модели «сущность-связь) в логическую схему базы данных.

Основные 6 правил генерации таблиц:

Правило 1. Если показатель кардинальности бинарной связи равен 1 : 1 и класс принадлежностей обеих сущностей является обязательным, то требуется только одна таблица. Первичным ключом этой таблицы может быть ключ любой из двух сущностей.

Правило 2. Если показатель кардинальности бинарной связи равен 1 : 1 и класс принадлежности одной сущности является обязательным, а другой - необязательным, то необходимо построение двух таблиц. Под каждую сущность нужно выделить по одной таблице, при этом ключ сущности должен служить первичным ключом для соответствующей таблицы. Кроме того, ключ сущности, для которой класс принадлежности является необязательным, добавляется в качестве атрибута в таблицу, выделенную для сущности с обязательным классом принадлежности.

Правило 3.Если показатель кардинальности бинарной связи равен 1 : 1 и класс принадлежности ни одной сущности не является обязательным, то необходимо использовать три таблицы: по одной для каждой сущности, ключи которых служат в качестве первичных в соответствующих таблицах, и одна таблица для связи. Среди своих атрибутов таблица связи будет иметь по одному ключу каждой сущности.

Правило 4. Если показатель кардинальности бинарной связи равен 1 : M и класс принадлежности n-связной сущности является обязательным, то достаточным является использование двух таблиц, по одной на каждую сущность, при условии, что ключ каждой сущности служит в качестве первичного ключа для соответствующей таблицы. Дополнительно ключ 1-связной сущности должен быть добавлен как атрибут в таблицу, отводимую для n-связной сущности.

Правило 5. Если показатель кардинальности бинарной связи равен 1 : M и класс принадлежности n-связной сущности является необязательным, то необходимо формирование трех таблиц: по одной для каждой сущности, причем ключ каждой сущности служит первичным ключом соответствующей таблицы, и одной таблицы для связи. Таблица связи должна иметь среди своих атрибутов ключ каждой сущности.

Правило 6. Если показатель кардинальности бинарной связи равен M : M, то для хранения данных необходимо три таблицы: по одной для каждой сущности, причем ключ каждой сущности используется в качестве первичного ключа соответствующей таблицы, и одной таблицы для связи. Таблица связи должна иметь в числе своих атрибутов ключ каждой сущности.

Рассмотрим преобразование ранее созданной модели «сущность-связь» торговой фирмы (рис. 4).

Связь ОФОРМЛЕНА между сущностями ПОЛУЧАТЕЛЬ и НАКЛАДНАЯ имеет показатель кардинальности один-ко-многим и класс принадлежности обеих сущностей обязательный. Поэтому для преобразования этой связи воспользуемся правилом 4. В соответствии с этим правилом надо построить только две таблицы: по одной на каждую сущность. Причём ключ односвязной сущности добавляется как неключевой атрибут в таблицу, отводимую n-связной сущности (Код). В результате получим следующие таблицы:

ПОЛУЧАТЕЛЬ (Код, Адрес, Название);

НАКЛАДНАЯ (Номер, Налог, Итого, Код).

Сущности ТОВАР и НАКЛАДНАЯ соединены связью типа М:М. Для преобразования модели «сущность-связь» потребуется правило 6, при этом класс принадлежности обеих сущностей не играет никакой роли. В этом случае требуется 3 таблицы: по 1 для каждой сущности и 1 таблица связи, которая будет иметь по одному ключу от каждой сущности, а первичным ключом таблицы связи будет составной ключ:

ТОВАР (Ном.номер, Наименование, Цена, Количество, Сумма);

НАКЛАДНАЯ (Номер, Налог, Итого, Код);

ВКЛЮЧАЕТ (Ном.номер, Номер).

Между сущностями НАКЛАДНАЯ и ОПЛАТА связь типа один-к одному и класс принадлежности обеих сущностей обязательный. Следовательно, для преобразования надо воспользоваться правилом 1, т.е. потребуется только одна таблица

НАКЛАДНАЯ (Номер, Налог, Итого, Код, Номер_документа, Дата, Вид).

Таким образом, в итоге была получена следующая реляционная модель:

ПОЛУЧАТЕЛЬ (Код, Адрес, Название);

НАКЛАДНАЯ (Номер, Налог, Итого, Код, Номер_документа, Дата, Вид);

ТОВАР (Ном.номер, Наименование, Цена, Количество, Сумма);

ВКЛЮЧАЕТ (Ном.номер, Номер).

1.6.2 Нормализация базы данных

Проектирование схемы БД должно решать задачи минимизации дублирования данных и упрощения процедур их обработки и обновления. При неправильно спроектированной схеме БД могут возникнуть аномалии модификации данных.

Для решения подобных проблем проводится **нормализация таблиц**.

Нормализация – это разбиение таблицы на две или более таблицы, обладающих лучшими свойствами при вставки, изменении и удалении данных. Окончательная цель нормализации сводится к получению такого проекта базы данных, в котором *каждый факт 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.

Первая нормальная форма (1НФ)

Таблица находится в *первой нормальной форме* тогда и только тогда, когда ни одна из ее строк не содержит в любом своем поле более одного значения.

Вторая нормальная форма (2НФ).

Вторая нормальная форма основана на понятии *функциональной зависимости*. Пусть X и Y – атрибуты некоторой таблицы. Если в любой момент времени каждому значению X соответствует единственное значение Y, то говорят, что Y функционально зависит от X (X→Y). Атрибут X в функциональной зависимости X→Y называется *детерминантом* отношения.

В нормализованной таблице все неключевые атрибуты функционально зависят от ключа таблицы. Неключевой атрибут функционально полно зависит от составного ключа, если он функционально зависит от ключа, но не находится в функциональной зависимости ни от какой части составного ключа.

Таблица находится во 2НФ, если она приведена к 1НФ и каждый неключевой атрибут функционально полно зависит от составного первичного ключа.

Таким образом, если таблица в 1НФ имеет простой первичный ключ, она сразу находится во второй нормальной форме.

Для того чтобы привести таблицу ко 2НФ, нужно:

1. В исходной таблице выявить атрибуты, зависящие от части ключа и создать новую таблицу без атрибутов исходной таблицы, находящихся в частичной функциональной зависимости от первичного ключа.
2. Создать таблицу, атрибутами которой являются части составного ключа и атрибуты, зависящие от этих частей.

Третья нормальная форма (3НФ).

Третья нормальная форма основана на понятии *транзитивной зависимости*. Пусть X, Y, Z – атрибуты некоторой таблицы. При этом X→Y и Y→Z, но обратное соответствие отсутствует, т.е. Z не зависит от Y или Y не зависит от X. Тогда говорят, что Z транзитивно зависит от X (X→→Z).

Таблица находится в 3НФ, если она находится во 2НФ и каждый неключевой атрибут нетранзитивно зависит от первичного ключа.

Для того чтобы привести отношение к 3НФ, нужно:

1. В исходной таблице выявить транзитивно зависящие атрибуты (X→Y и Y→Z), и создать новую таблицу, исключив из исходной таблицы атрибуты, транзитивно зависящие от ключа (атрибут Z).
2. Построить новую таблицу, атрибутами которой являются части транзитивной зависимости (атрибуты Y и Z).

Четвертая нормальная форма (4НФ)

Четвертая нормальная форма основана на понятии *многозначной зависимости*.

Многозначная зависимость существует, если заданным значениям атрибута X соответствует множество значений атрибута Y (X–»Y).

Таблица находится в 4НФ в том и только в том случае, если она находится в 3НФ и в случае существования многозначной зависимости A -» B все остальные атрибуты таблицы функционально зависят от A.

Для того чтобы привести отношение к 4НФ, нужно построить две или более проекции исходного отношения, каждая из которых содержит ключ и одну из многозначных зависимостей.

1.7 Программная реализация базы данных

Программная реализация базы данных заключается в описании ее схемы на языке определения данных (DDL, Data Definition Language) выбранной СУБД. Принятые на этом этапе решения оказывают огромное влияние на производительность системы.

Важнейшими составляющими проекта базы данных являются разработка декларативных и процедурных средств поддержки целостности данных, а так же реализация операций над данными (поиск, вставка, удаление, обновление).

Не менее важным вопросом разработки базы данных является защита БД от несанкционированного доступа.

Для защиты от несанкционированного доступа каждому пользователю доступ к данным предоставляется только в соответствии с его правами доступа, набор которых также является составной частью проекта БД.

2. ВЫПОЛНЕНИЕ КУРСОВОЙ РАБОТЫ

2.1. Задачи, решаемые в курсовой работе

Курсовая работа включает в себя решение следующих задач:

1. Разработка модели «сущность – связь» (инфологическое проектирование).
2. Обоснование выбора СУБД и создание БД в выбранной СУБД.
3. Даталогическое проектирование реляционной БД на основе модели «сущность – связь», полученной на предыдущем этапе.
4. Нормализация полученной базы данных. При этом необходимо обратить внимание на то, что при переходе от инфологической модели к реляционной модели таблицы имели бы наивысшую нормальную форму.
5. Определение характеристик атрибутов и правил декларативной поддержки ограничений целостности данных (обязательные данные, целостность сущностей, ссылочная целостность, требования конкретного предприятия (бизнес-правила)).
6. Разработка хранимых процедур и триггеров, обеспечивающих процедурную поддержку целостности данных (курсовая работа должна содержать не менее двух хранимых процедур и двух триггеров).
7. Реализация операций над данными (поиск, вставка, удаление, обновление) в соответствии с вариантом задания с помощью языка SQL.

Задание выполняется индивидуально каждым студентом в соответствии с вариантом, утвержденным преподавателем.

2.2. Организация процесса выполнения курсовой работы

Преподавателем индивидуально выдаются задания на курсовую работу. Вариант задания берется из списка вариантов заданий, приведенном в разделе 3. При этом номер варианта соответствует номеру студента в журнале группы. Студент имеет право предложить самостоятельное задание на курсовую работу. В

этом случае задание на курсовую работу согласовывается с руководителем курсовой работы.

Задание подписывается студентом, преподавателем и заведующим кафедрой.

Примерный график выполнения курсовой работы:

- 1-я неделя. Получение задания.
- 2-я – 3-я недели. Инфологическое проектирование.
- 4-я неделя. Обоснование выбора СУБД и создание БД в выбранной СУБД.
- 5-я неделя. Описание реляционной модели данных.
- 6-я неделя. Нормализация полученной базы данных.
- 7-я неделя. Определение характеристик атрибутов и правил декларативной поддержки ограничений целостности данных.
- 8-я и 9-я недели. Определение характеристик атрибутов и правил декларативной поддержки ограничений целостности данных.
- 10-я и 11-я недели. Разработка хранимых процедур и триггеров, обеспечивающих процедурную поддержку целостности данных.
- 12-я и 15-я недели. Реализация операций над данными (поиск, вставка, удаление, обновление) в соответствии с вариантом задания с помощью языка SQL.
- 16-я неделя. Оформление пояснительной записки.
- 17-я и 18-я недели. Защита курсовой работы.

После предоставления пояснительной записки преподавателю и ее проверки, при отсутствии существенных замечаний к содержанию, назначается дата защиты. Защита курсовой работы проводится перед комиссией, состав которой утверждается кафедрой, и в присутствии студентов данной учебной группы. При защите курсовой работы студент должен привести сведения о поставленной задаче и ее особенностях, описать предлагаемую структуру данных, обосновать выбор СУБД и провести анализ проделанной работы.

2.3. Оформление пояснительной записки

Пояснительная записка выполняется на писчей бумаге формата А4 (297×210 мм). Отступы от границ листа: левое поле – 25 мм, правое, верхнее, нижнее – 20 мм. Разделы, подразделы, рисунки, таблицы и страницы нумеруются.

Содержание пояснительной записки:

1. Титульный лист (приложение 1).
2. Лист задания (приложение 2).
3. Аннотация, включающая в себя сведения об объеме курсовой работы, количестве рисунков, таблиц, краткое описание задачи, оценку результатов и т.д.
4. Содержание.
5. Основная часть.
   - 5.1. Концептуальное проектирование.
   - 5.2. Обоснование выбора СУБД
   - 5.3. Даталогическое проектирование.
     - 5.3.1. Преобразование концептуальной модели в реляционную модель.
     - 5.3.2. Нормализация базы данных.
     - 5.3.3. Определение характеристик атрибутов.
   - 5.4. Создание БД в выбранной СУБД.
   - 5.5. Поддержка целостности данных.
     - 5.5.1. Декларативная поддержка ограничений целостности.
     - 5.5.2. Процедурная поддержка ограничений целостности.
   - 5.6. Реализация операций над данными.
6. Список литературы.