ФГБОУ ВО БРЯНСКИЙ ГОСУДАРСТВЕННЫЙ
АГРАРНЫЙ УНИВЕРСИТЕТ

**КАФЕДРА ИНФОРМАТИКИ,
ИНФОРМАЦИОННЫХ СИСТЕМ И ТЕХНОЛОГИЙ**

Лысенкова С.Н.

ОСНОВЫ ПРОЕКТИРОВАНИЯ БАЗ ДАННЫХ

*Учебно-методическое пособие
для студентов направления подготовки
09.03.03. Прикладная информатика*

Брянская область
2019

УДК 004.65 (07)
ББК 32.97
Л 88

Лысенкова, С. Н. **Основы проектирования баз данных:** учебно-методическое пособие для студентов направления подготовки 09.03.03 Прикладная информатика / С. Н. Лысенкова. – Брянск: Изд-во Брянский ГАУ, 2019. - 66 с.

Пособие должно обеспечить студентов конкретным инструментарием для практической реализации методов проектирования БД. Издание окажет помощь студентам при изучении дисциплины «Базы данных», предназначено для бакалавров направления подготовки 09.03.03 «Прикладная информатика», профиль «Программно-технические средства информатизации».

Рецензент: к.э.н, доцент заведующая кафедрой информатики, информационных систем и технологий Ульянова Н.Д.

Рекомендовано к изданию учебно-методической комиссии института энергетики и природопользования протокол № 1 от 1 октября 2019 года.

СОДЕРЖАНИЕ

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ РАБОТЫ С БАЗАМИ ДАННЫХ

Основной проблемой, обусловившей развитие теории и практики баз данных, является обеспечение надежного контролируемого хранения необходимых данных между сеансами работы, их передачи между рабочими местами и эффективного их извлечения по мере необходимости.

По мере роста требований к объему хранимой, искомой и передаваемой информации, к скорости и точности выполнения соответствующих операций, работа с данными стала узким звеном в деятельности практически всех организаций, независимо от рода деятельности. В соответствии с этим, начали создаваться и использоваться аппаратные и программные средства автоматизированной обработки информации.

Список данных – представляют собой определенным образом организованную информацию, как правило, это таблицы, в которых все строки, за исключением заголовков, имеют одинаковую структуру и типы данных

Пример.

Список представлен в таблице 1. Компания занимается арендой оборудования, такого для строительных работ, таких как краны и экскаваторы. В таблице показана лишь небольшая часть деятельности этой компании, но даже этот маленький пример показывает целый ряд важных проблем со списками. В таблице есть 10 столбцов: проект, подрядчик, телефон, тип оборудования, номер оборудования, плата за день, дата начала, дата окончания, количество дней, стоимость.

проект	подрядчик	телефон	тип оборудования	номер оборудования	плата за день	дата начала	дата окончания	количество дней	стоимость
Ремонт дороги	ООО «Маг»	23-45-67	экскаватор	234	200	1.01.14	15.01.14	30	6000
Строительство дома	ОАО «Жилье»	23-98-76	кран	367	560	10.01.14	10.02.14	31	17360
Ремонт квартиры	ОАО «Мастер»	24-44-56	перфоратор	367		15.01.14			0
Ремонт дороги	ОАО «Строй»	24-87-66	экскаватор	459	240	20.02.14	3.03.14	30	7200
Строительство гаража	ООО «Маг»	23-45-67	Грузовой автомобиль	743	180	15.01.14	20.01.14	5	900

Недостатки списков:

1. Сложность в редактировании данных. Допустим, что у компании ООО «Маг» изменится телефон на 23-55-00. Чтобы данный список остался в порядке, необходимо сделать изменения в двух строках. Если не изменить эти строки, то появятся противоречащие друг другу данные и нельзя установить, какой из этих номеров правильный. Затем предположим, что список отражает деятельность фирмы по аренде оборудования за весь квартал или год. Следовательно, в списке могут быть сотни или тысячи строк. Чтобы внести изменения в телефонный номер, необходимо произвести поиск по списку и провести изменения в каждой строке, которая содержит фирму ООО «Маг». Эти действия могут занять продолжительны период времени и привести к ошибкам.

2. Потеря всех данных при удалении части информации. Компания ОАО «Жилье» отказалась от аренды крана, теперь необходимо удалить вторую строку. В результате чего произойдет потеря данных не только о про-

кате, но также будет удален номер телефона компании и тот факт, что эта она работает над строительством дома и арендует кран за 560 долларов в день.

3. Еще одной проблемой является несовместимость данных. Если сделать опечатку в имени подрядчика и случайно ввести ОАО «Маг» вместо ООО «Маг». Пользователи этого списка не будет знать, то ли появился новый контакт, то ли произошла ошибка. Такие ошибки не возможны, если выбор подрядчика буде осуществляться из ранее созданного списка.

4. Возможно возникновение проблемы касающейся отсутствующих данных. Например, в записи о прокате перфоратора отсутствует дата окончания проката. Это может быть как ошибкой так и не полной информацией. Допустим, это значит, что аренда еще не закончена или что компания планирует пользоваться перфоратором неопределенно долго.

5. Возникают проблемы совместно используемых данных. Компания планирует предоставлять доступ достаточно широкому кругу клиентов к именам подрядчиков и к телефонным номерам из общего источника данных. Следовательно, в случае изменения номера телефона, необходимо его отредактировать в общем источнике данных компании. Иначе сотрудникам придется редактировать эти данные на каждом рабочем месте.

База данных

В процессе развития средств обработки данных были выявлены следующие характерные черты систем обработки данных.

- Обработка постоянных (перманентных) данных.
- Централизованная обработка данных на основе стандартов.
- Интеграция данных.

- Независимость (самодостаточность) данных от программ обработки.
- Целостность хранимых данных (при хранении данных необходимо обеспечить контроль их непротиворечивости и корректности связей между элементами данных).
- Эффективность обработки данных.
- Язык управления данными.

Базы данных представленные в виде групп таблиц.

Один из значимых недостатков списков, похожих на представленный в таблице 1, заключается в том, что они содержат данные на различные темы:

- строительные проекты,
- подрядчики,
- оборудование
- аренда

На рисунке 1 отражены результаты разбиения списка по отдельным тематикам. Каждая из этих тем представляет собой таблицу данных. Можно составить следующие таблицы: подрядчики, строительные подряду, оборудование и прокат.

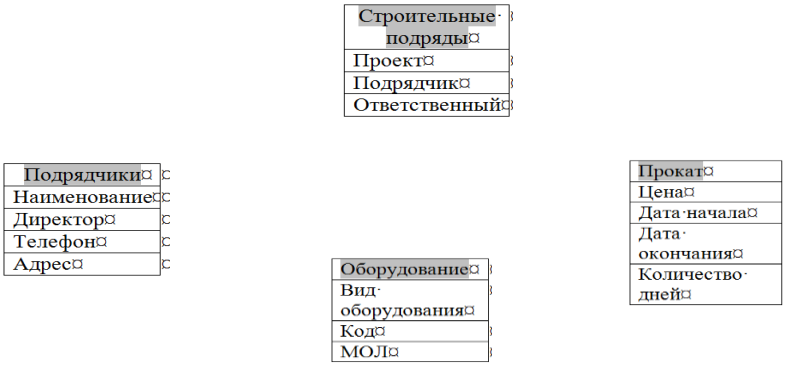


Рис. 1. Пример преобразования списка в базу данных

Деление списка на четыре таблицы, позволило решить многие проблемы, которые были рассмотрены ранее. Если компания меняет свой телефон, это изменение проводится только один раз — в таблице Подрядчики. При необходимости удалить запись о прокате, надо удалить только строку из таблицы Строительные подряды, не теряя при этом данных в других таблицах.

Возлежании несогласованности данных, можно разработать приложение баз данных с учетом того, что при необходимости добавления новых записей о прокате, пользователь может выбирать запись из таблицы Подрядчики. Следовательно, будет уменьшено количество ошибок. Кроме того это позволит контролировать появление новых клиентов. Можно создать ограничение которое позволит ограничить круг пользователей редактирующих данные. Разбиение на таблицы на отдельные темы будет способствовать внесению большей информации по каждой теме, чем отражено в списке таблицы 1.

Связи в базе данных

Данные на рисунке 1 оптимальны, но не имеют связи между таблицами. Для того чтобы получить информацию о том какой подрядчик арендовал оборудование, для какого проекта и на какой срок необходимо чтобы эти таблицы были между собой связаны.

Рассмотрим один из способов создания связи между таблицами.

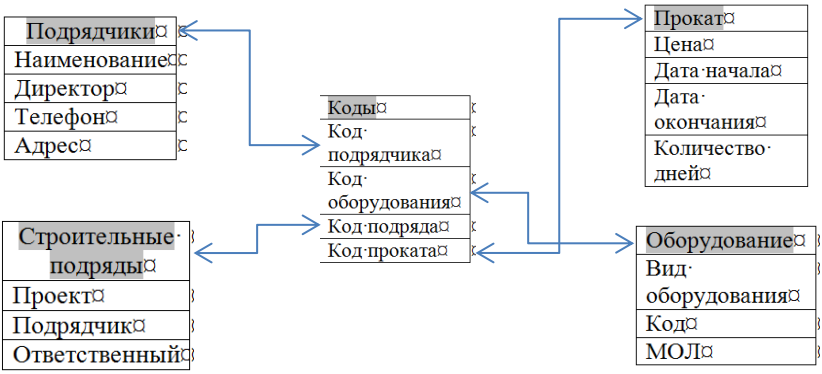


Рис. 2. Связи базы данных

Каждой строке в каждой таблице присваивается уникальный идентификатор - Индикационный Код (ИД). Этот идентификатор не несет смысловой нагрузки для пользователей; его цель —пронумеровать все строки таблицы. Значения идентификатора используется для создания связи со строками новой таблицы.

Таким образом, для решения проблем, которые возникают при работе со списками, его необходимо разбить на четыре таблицы, каждая из которых будет содержать поля относящиеся к определенной теме. Далее создается связь между таблицы с использованием уникальных идентификаторов, которые представляют собой ключи.

БАНКИ ДАННЫХ. КЛАССИФИКАЦИЯ

Для пользователей понятия "банк данных" и "база данных" являются достаточно близкими, и используются для обозначения некоторого структурированного массива информации. Однако между ними есть принципиальные различия. Так банки данных являются «контейнерами» для информации, с которой выполняется достаточно ограниченное число действий (поиск, просмотр), в то время как использование баз данных предполагает применение какой-то специальной обработки информации (с помощью специально написанных программ). Однако, для простого пользователя не всегда очевидна закономерность выбора между этими двумя близкими терминами в том или ином случае.

Банк данных (БнД) - это одна из форм информационных систем.

Банком данных называют систему специальным образом организованных баз данных, программных, технических, языковых и организационно- методических средств, предназначенных для обеспечения централизованного накопления и коллективного многоцелевого использования данных.

Классификация банков данных

1. По сферам деятельности (экономические, юридические, социальные и т.д.).

2. По условиям предоставления услуг различают бесплатные и платные банки, которые, в свою очередь, делятся на коммерческие и бесприбыльные (научные, библиотечные или социально-значимые).

3. По форме собственности БнД делятся на государственные и негосударственные.

4. По степени доступности различают общедоступные и с ограниченным кругом пользователей.

Другие виды классификации связаны с отдельными компонентами БнД.

Разработка банков данных состоит из 4-х этапов:

1 этап. Формирование и анализ требований к системе: составляется спецификация системы, включающая список задач, которые должен решать БнД:

- перечень конечных пользователей и их функций;
- перечень требований к БД;
- составляется схема документооборота в организации.

2 этап. Концептуальное проектирование: создается информационная модель системы без привязки к типу ЭВМ и типу системных программных средств; строится инфологическая модель базы данных, которая наиболее полно описывает предметную область в терминах пользователя.

3 этап. Проектирование реализации логической модели: выбирается вычислительная система, системные программные средства и СУБД; проектируется структура данных и строится даталогическая модель БД (схема БД), которая представляет собой описание логической структуры БД на языке конкретной выбранной СУБД.

4 этап. Физическая реализация, которая включает в себя создание и загрузку данных в БД, разработку и отладку прикладных программ для работы с базой данных, написание документации. На этом этапе строится физическая модель БД, которая описывает используемые запоминающие устройства, способы физической организации данных. Описание физической структуры БД называют схемой хранения. В настоящее время наблюдается тенденция к сокращению этого вида работ.

Классификация баз данных

Существует огромное количество разновидностей баз данных, отличающихся по различным критериям. Основные классификации приведены ниже.

1. Классификация по модели данных: Иерархическая, Объектная и объектно-ориентированная, Объектно-реляционная, Реляционная, Сетевая, Функциональная.

2. Классификация по среде постоянного хранения

- Во вторичной памяти, или традиционная: средой постоянного хранения является периферийная энергонезависимая память (вторичная память) — как правило жёсткий диск.
- В оперативную память СУБД помещает лишь кеш и данные для текущей обработки.
- В оперативной памяти: все данные на стадии исполнения находятся в оперативной памяти.
- В третичной памяти: средой постоянного хранения является отсоединяемое от сервера устройство массового хранения (третичная память), как правило на основе магнитных лент или оптических дисков.
- Во вторичной памяти сервера хранится лишь каталог данных третичной памяти, файловый кеш и данные для текущей обработки; загрузка же самих данных требует специальной процедуры.

3. Классификация по содержимому: Географическая, Историческая, Научная, Мультимедийная, Клиентская.

4. Классификация по степени распределённости

- Централизованная, или сосредоточенная: БД, полностью поддерживаемая на одном компьютере.
- Распределённая: БД, составные части которой размещаются в различных узлах компьютерной сети в соответствии с каким-либо критерием.
  - Неоднородная: фрагменты распределённой БД в разных узлах сети поддерживаются средствами более одной СУБД
  - Однородная: фрагменты распределённой БД в разных узлах сети поддерживаются средствами одной и той же СУБД.

- Фрагментированная, или секционированная: методом распределения данных является фрагментирование (партиционирование, секционирование), вертикальное или горизонтальное.
- Тиражированная (англ. *replicated database*): методом распределения данных является тиражирование (репликация).

5. Классификация по способу доступа к данным:
- Базы данных с локальным доступом
- Базы данных с сетевым доступом.

Для всех современных баз данных можно организовать сетевой доступ с многопользовательским режимом работы. Централизованные базы данных с сетевым доступом могут иметь следующую архитектуру:
- файл-сервер;
- *клиент-сервер* базы данных;

6. Классификация по характеру организации данных:
- неструктурированные,
- частично структурированные и
- структурированные.

Этот классификационный признак относится к информации, представленной в символьном виде. К неструктурированным БД могут быть отнесены базы, организованные в виде семантических сетей. Частично структурированными можно считать базы данных в виде обычного текста или гипертекстовые системы. Структурированные БД требуют предварительного проектирования и описания структуры БД. Только после этого базы данных такого типа могут быть заполнены данными.

Другие виды БД
- Пространственная: БД, в которой поддерживаются пространственные свойства сущностей предметной области. Такие БД широко используются в геоинформационных системах.

• Временная, или темпоральная: БД, в которой поддерживается какой-либо аспект времени, не считая времени, определяемого пользователем.

• Пространственно-временная БД: БД, в которой одновременно поддерживается одно или более измерений в аспектах как пространства, так и времени.

• Циклическая: БД, объём хранимых данных которой не меняется со временем, поскольку в процессе сохранения данных одни и те же записи используются циклически.

Сверхбольшие базы данных

В конце 1990-х годов была разработана очень большая база данных, связанная с физикой элементарных частиц, молекулярной биологией и другими дисциплинами в исследовательских проектах. В рамках большого количества электронных библиотек создаются большие мультимедийные базы данных. Увеличение числа реализаций таких проектов обеспечивается созданием подходов и методов, обеспечивающих экономически эффективное сопровождение как больших коллекций информационных ресурсов, так и высокопроизводительного доступа к ним среди современных тенденций развития технологий баз данных.

Сверхбольшая база данных-это база данных, которая занимает очень много места на физическом носителе. Под термином понимаются возможные объемы баз данных, определяемые новейшими физическими технологиями хранения данных, а также программным обеспечением и методами обработки данных.

Количественное определение" очень большой объем " меняется с течением времени; в настоящее время считается, что это объем, измеренный по крайней мере в петабайтах. Для сравнения можно отметить, что в 2005 году самая большая в мире база данных объемом хранения со-

ставляла около 100 терабайт. Эксперты отмечают, что очень большие базы данных, при проектировании требовали особых подходов. Их создание часто осуществляется специальными проектами системных решений, которые смогут хоть как-то воспользоваться этим большим количеством данных. Как правило, вам потребуются специальные решения дисковой подсистемы конкретных версий операционной среды и специальные механизмы доступа к базе данных. Очень большие базы данных, сохранение и обработка исследований-это всегда теория и практика баз данных на переднем крае. С 1975 года ежегодно проводится очень большая международная конференция по базам данных.

ИЕРАРХИЧЕСКАЯ МОДЕЛЬ ДАННЫХ

1. Структура данных

Для организации данных в СУБД иерархического типа используются термины: элемент, запись (группа), групповое отношение, ключ.

Атрибут (элемент данных) – представляет собой наименьшую единицу структуры данных. Обычно каждому элементу при создании базы данных деется уникальное имя, по которому к нему обращаются при обработке.

Элемент данных является полем.

Запись - именованная совокупность атрибутов. Использование записей позволяет за одно обращение к базе получить некоторую логически связанную совокупность данных. Именно записи изменяются, добавляются и удаляются. Тип записи определяется составом ее атрибутов. Экземпляр записи - конкретная запись с конкретным значением элементов

Групповое отношение - иерархическое отношение между записями двух типов. Родительская запись (владелец группового отношения) называется исходной записью, а дочерние записи (члены группового отношения) - подчиненными. Иерархическая база данных может хранить только такие древовидные структуры.

Ключ в базе данных - это поле (совокупность полей) значение которого не повторяется у разных записей, содержимое которого однозначно определяет запись в таблице и отличает ее от других.

Родительская (корневая) запись каждого дерева обязательно должна содержать ключ с уникальным значением. Ключи дочерних (некорневых) записей должны иметь уникальное значение только в рамках группового отношения. Каждая запись идентифицируется полным сцепленным ключом, под которым понимается совокупность ключей всех записей от корневой по иерархическому пути.

2. Операции над данными, определенные в иерархической модели:

ДОБАВИТЬ в базу данных новую запись. Для родительской записи обязательно формирование значения ключа.

ИЗМЕНИТЬ значение данных предварительно извлеченной записи. Ключевые данные не должны подвергаться изменениям.

УДАЛИТЬ некоторую запись и все подчиненные ей записи.

ИЗВЛЕЧЬ:

- извлечь родительскую запись по ключевому значению, допускается также последовательный просмотр корневых записей
- извлечь следующую запись (следующая запись извлекается в порядке левостороннего обхода дерева)

В операции ИЗВЛЕЧЬ допускается задание условий выборки (например, извлечь товар стоимостью более 500 руб.)

Все операции изменения применяются только к одной "текущей" записи (которая предварительно извлечена из базы данных). Такой подход к манипулированию данных получил название "навигационного".

3. Ограничения целостности.

Поддерживается только целостность связей между владельцами и членами группового отношения (никакой потомок не может существовать без предка). Как уже отмечалось, не обеспечивается автоматическое поддержание соответствия парных записей, входящих в разные иерархии.

Пример:

В структуре предприятия имеются отделы, состоящие из сотрудников. При этом есть ряд ограничений:

- в отделе работает несколько сотрудников,
- сотрудник может работать только в одном отделе.

Для информационной системы управления персона-

лом необходимо создать групповое отношение, состоящее из родительской записи *Отдел (Наименование_Отдела, Число_Работников)* и дочерней записи *Сотрудник* (*Фамилия, Должность, Оклад*). Это отношение показано на рис. 1.а.

Для автоматизации учета контрактов с заказчиками необходимо создание еще одной иерархической структуры: заказчик - контракты с ним - сотрудники, задействованные в работе над контрактом. Это дерево будет включать записи *Заказчик (Наименование_Заказчика, Адрес), Контракт(Номер, Дата, Сумма), Исполнитель (Фамилия, Должность, Наименование_Отдела)* (рис. 1.б).

Недостатки иерархической модели БД:

Частично дублируется информация между записями *Сотрудник и Исполнитель* (такие записи называют парными), причем в иерархической модели данных не предусмотрена поддержка соответствия между парными записями.

Иерархическая модель реализует отношение между родительской и дочерней записью по схеме 1:N, то есть одной родительской записи может соответствовать любое число дочерних. Предположим, что сотрудник может участвовать в выполнении более чем в одном контракта (т.е. возникает связь типа M:N). В этом случае в базу данных необходимо ввести еще одно групповое отношение, в котором Исполнитель будет являться исходной записью, а Контракт - дочерней (рис. (с)). Следовательно происходит дублирование информации.

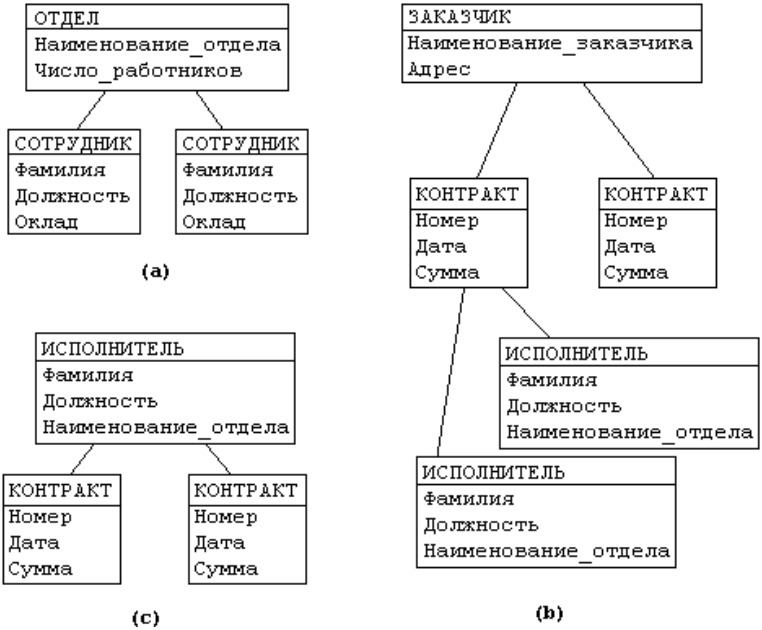


Рис. 1. Иерархическая модель БД

СЕТЕВАЯ МОДЕЛЬ ДАННЫХ

1. Структура данных

Основные элементы сетевой модели данных:

Элемент данных – минимальная информационная единица доступная пользователю.

Агрегат данных – именованная совокупность элементов данных внутри записи или другого агрегата, которую можно рассматривать как единое целое. Имя агрегата используется для его идентификации в схеме структуры данного более высокого уровня. Агрегат данных может быть простым, если состоит только из элементов данных, и составным, если включает в свой состав другие агрегаты.

Запись - совокупность агрегатов или элементов данных, отражающих некоторую сущность предметной области. Иными словами, запись - это агрегат, который не входит в состав никакого другого агрегата и может иметь сложную иерархическую структуру, поскольку допускается многократное применение агрегации. Имя записи используется для идентификации типа записи в схемах типов структур более высокого уровня.

Тип записей – эта совокупность подобных записей. Тип записей представляет некоторый класс реального мира.

Набор - именованная двухуровневая иерархическая структура, которая содержит запись владельца и запись (или записи) членов. Наборы отражают связи «один ко многим» и «один к одному» между двумя типами записей.

Наборы бывают нескольких видов:

- С одними и теми же типами записей, но разными типами наборов.

- Наборы из трех записей и более, в том числе с обратной связью.

- Сингулярный набор (только один экземпляр). У такого набора нет естественного владельца и в качестве

него выступает система. В дальнейшем такие наборы могут приобрести запись - владельца.

Основное различие иерархической и сетевой моделей данных состоит в том, что в сетевой модели запись может быть членом более чем одного группового отношения. Согласно этой модели каждое групповое отношение именуется и проводится различие между его типом и экземпляром. *Тип группового* отношения задается его именем и определяет свойства общие для всех экземпляров данного типа. *Экземпляр группового* отношения представляется записью-владельцем и множеством (возможно пустым) подчиненных записей. При этом имеется следующее ограничение: экземпляр записи не может быть членом двух экземпляров групповых отношений одного типа (т.е. сотрудник из примера 1, не может работать в двух отделах).

Каждый экземпляр группового отношения характеризуется следующими признаками:

- способ упорядочения подчиненных записей: произвольный, хронологический /очередь/, обратный хронологический /стек/, сортированный. Если запись объявлена подчиненной в нескольких групповых отношениях, то в каждом из них может быть назначен свой способ упорядочивания.
- режим включения подчиненных записей:

– автоматический - невозможно занести в БД запись без того, чтобы она была сразу же закреплена за неким владельцем;

– ручной - позволяет запомнить в БД подчиненную запись и не включать ее немедленно в экземпляр группового отношения. Эта операция позже инициируется пользователем).

- режим исключения. Выделяют три класса членства подчиненных записей в групповых отношениях:

– Фиксированное. Подчиненная запись жестко свя-

зана с записью владельцем и ее можно исключить из группового отношения только удалив. При удалении записи-владельца все подчиненные записи автоматически тоже удаляются.

– Обязательное. Допускается переключение подчиненной записи на другого владельца, но невозможно ее существование без владельца. Для удаления записи-владельца необходимо, чтобы она не имела подчиненных записей с обязательным членством.

– Необязательное. Можно исключить запись из группового отношения, но сохранить ее в базе данных не прикрепляя к другому владельцу. При удалении записи-владельца ее подчиненные записи - необязательные члены сохраняются в базе, не участвуя более в групповом отношении такого типа.

2. Операции над данными

ДОБАВИТЬ - внести запись в БД и, в зависимости от режима включения, либо включить ее в групповое отношение, где она объявлена подчиненной, либо не включать ни в какое групповое отношение.

ВКЛЮЧИТЬ В ГРУППОВОЕ ОТНОШЕНИЕ - связать существующую подчиненную запись с записью-владельцем.

ПЕРЕКЛЮЧИТЬ - связать существующую подчиненную запись с другой записью-владельцем в том же групповом отношении.

ОБНОВИТЬ - изменить значение элементов предварительно извлеченной записи.

ИЗВЛЕЧЬ - извлечь записи последовательно по значению ключа, а также используя групповые отношения - можно перейти от владельца к записям - членам, а от подчиненной записи к владельцу набора.

УДАЛИТЬ - убрать из БД запись. Если эта запись

является владельцем группового отношения, то анализируется класс членства подчиненных записей. Обязательные члены должны быть предварительно исключены из группового отношения, фиксированные удалены вместе с владельцем, необязательные останутся в БД.

ИСКЛЮЧИТЬ ИЗ ГРУППОВОГО ОТНОШЕНИЯ - разорвать связь между записью-владельцем и записью-членом.

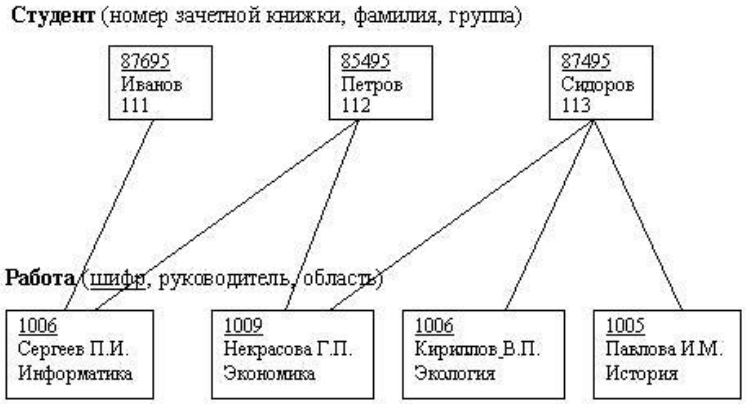
3. Ограничения целостности

Как и в иерархической модели обеспечивается только поддержание целостности по ссылкам (владелец отношения - член отношения).

Сетевая модель данных - это логическая модель данных, представляющая их сетевыми структурами типов записей и связанные отношениями мощности один-к-одному или один-ко-многим.

В отличие от реляционной модели, связи в ней моделируются наборами, которые реализуются с помощью указателей. Сетевые модели данных являются расширенной версией иерархической модели, однако основным отличием является то, что в сетевых моделях данных имеются указатели в обоих направлениях, которые соединяют родственную информацию.

Сетевую модель можно представить как граф узлами, которого является запись, а ребрами - набор. Сегменты данных в сетевых БД могут иметь множественные связи с сегментами старшего уровня. При этом направление и характер связи в сетевых БД не являются столь очевидными, как в случае иерархических БД. Поэтому имена и направление связей должны идентифицироваться при описании БД.



Пример:

Иерархическая структура (пример 1) преобразовывается в сетевую следующим образом (см. рис. 2):

- деревья (а) и (б), представленные на рисунке 1, объединены в одну сетевую структуру, в которой запись *Сотрудник* входит в два групповых отношения;
- для отображения типа M:N создана запись *Сотрудник_Контракт*, которая не имеет полей и служит только для связи записей *Контракт и Сотрудник*, см. рис. 2. (Однако, в этой записи может храниться и полезная информация, например, доля данного сотрудника в общем вознаграждении по данному контракту.)

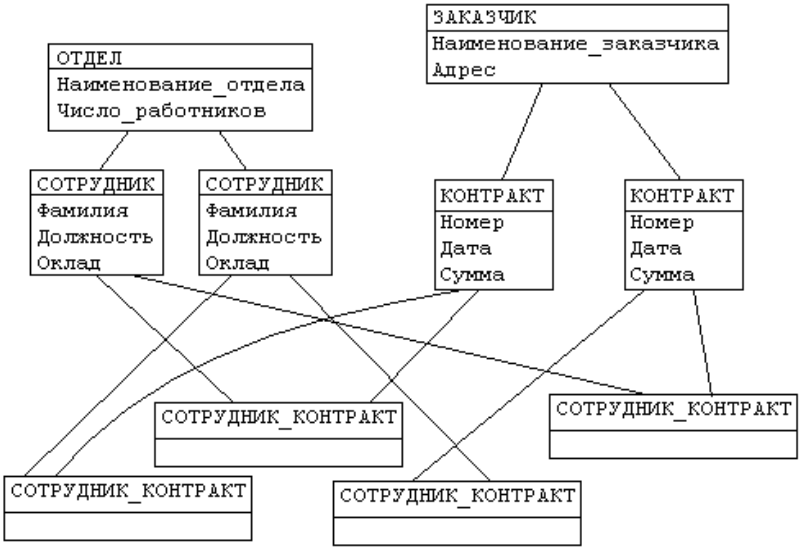


Рис. 2. Сетевая модель БД

В данном примере групповые отношения характеризуется следующими классами членства подчиненных записей:

– Фиксированное. Наличие группового отношения "ЗАКЛЮЧАЕТ" между записями "*Контракт*" и "*Заказчик*", поскольку контракт не может существовать без заказчика.

– Обязательное. Таким отношением связаны записи "*Сотрудник*" и "*Отдел*". Если отдел расформировывается, все его сотрудники должны быть либо переведены в другие отделы, либо уволены.

– Необязательное. Примером такого группового отношения может служить "ВЫПОЛНЯЕТ" между "*Сотрудники*" и "*Контракт*", поскольку в организации могут существовать работники, чья деятельность не связана с выполнением каких-либо договорных обязательств перед заказчиками.

РЕЛЯЦИОННАЯ МОДЕЛЬ БАЗ ДАННЫХ

вопросы:

1. Моделирование реляционной БД
2. Ключи отношений
3. Связывание таблиц

Пример.

Для обеспечения эффективной работы дирекции института и бухгалтерии университета необходимо разработать реляционную базу данных, которая содержала бы сведения о студентах, их успеваемости, начислениях стипендии и др.

Проектирование реляционной базы данных включает три этапа: концептуальное, логическое и физическое проектирование.

На этапе *концептуального проектирования* осуществляется анализ предметной области Институт-Бухгалтерия. Что позволило определить основные документы, которые послужат источниками данных:

• документ 1 "Карточка студента";

• документ 2 "Сведения из экзаменационных ведомостей";

• документ 3 "Справочник видов начислений";

• документ 4 "Начисления студентам";

• документ 5 "Приказ"

Вид этих документов приведен ниже.

Документ 1. "Карточка студента"

Номер зачетной книжки	Фамилия	Имя	Отчество	Дата рождения	Семейное положение	Что окончил	Обучение платное (Да/Нет)	Телефон	Образец подписи
9(6)	А(15)	А(Ю)	А(15)	ДД.ММ.ГГ	А(9)	А(15)	В	9(7)	Графический объект

Для реквизитов документов указаны форматы их значений. Так формат 9(6) указывает на то, что значения реквизитов будут принимать значения десятичные, числовые, максимум шестизначные. Формат А(15) означает, что значения реквизита принимают алфавитно-цифровые значения, содержащие максимум 15 символов. Формат В указывает на то, что значения поля имеет логическое значение.

Документ 2. "Сведения из экзаменационных ведомостей"

Номер зачетной книжки	Шифр группы	Семестр	Оценка по математике	Оценка по информатике	Оценка по экономической теории
9(6)	А (5)	9(1)	9(1)	9(1)	9(1)

В этом документе представлены сведения об успеваемости студентов за прошедший семестр.

Документ 3 "Справочник видов начислений"

Код начисления	Вид начисления
9(2)	А(25)

Документ 4 "Начисления студентам"

Номер зачетной книжки	Код начисления	Сумма начисленная, руб.	За какой месяц начислено
9(6)	9(2)	9(5)	А(10)

Документ 5 "Приказ"

Назначить плату за обучение в 2004/2005 учебном году в размере:

- дневное отделение - 450 у. е.;
- заочное отделение - 300 у. е.:
- дистанционное обучение - 150 у. е.

Ректор университета
профессор, д.э.н. Б. Н. Иванов

После проведения анализа документов выделяются объекты, являющиеся источниками информации. Для каждого объекта определяется ключевой реквизит, который однозначно идентифицирует экземпляры объекта. Например, реквизит "Номер зачетной книжки" однозначно идентифицирует студента.

Таблица 1.1 - Информационные объекты предметной области

Информационный объект	Наименование реквизита	Имя реквизита
СВЕДЕНИЯ	Номер зачетной книжки	НОМ_ЗАЧ
	Фамилия	ФАМ
	Имя	ИМЯ
	Отчество	ОТЧ
	Дата рождения	ДАТ_РОЖ
	Семейное положение	СЕМ_ПОЛ
	Что окончил	ЧТО_ОКОН
	Обучение платное (Да/Нет)	ОБУЧ
	Плата за обучение	ПЛАТА
	Телефон	ТЕЛ
	Образец подписи	ПОДП
УСПЕВАЕМОСТЬ	Номер зачетной книжки	НОМ_ЗАЧ
	Шифр группы	ГРУП
	Семестр	СЕМЕСТР
	Оценка по математике	ОЦ МАТЕМ
	Оценка по информатике	ОЦ_ИНФ
	Оценка по экономической теории	ОЦ_ЭКОН
СПРАВОЧНИК	Код начисления	КОД_НАЧ
	Вид начисления	ВИД_НАЧ
НАЧИСЛЕНИЯ	Номер зачетной книжки	НОМ_ЗАЧ
	Код начисления	КОД_НАЧ
	Сумма начисленная, руб.	СУММА
	За какой месяц начислено	ЗА_МЕСЯЦ

В таблице ключевые реквизиты выделены жирным шрифтом.

Далее необходимо определить связи между информационными объектами. Связь устанавливается между двумя информационными объектами. Наличие связи и ее тип обуславливаются природой реальных объектов, процессов, явлений, отображаемых информационными объектами. В примере представлены связи следующих типов:

- один к одному (1:1);
- один ко многим (1:M):

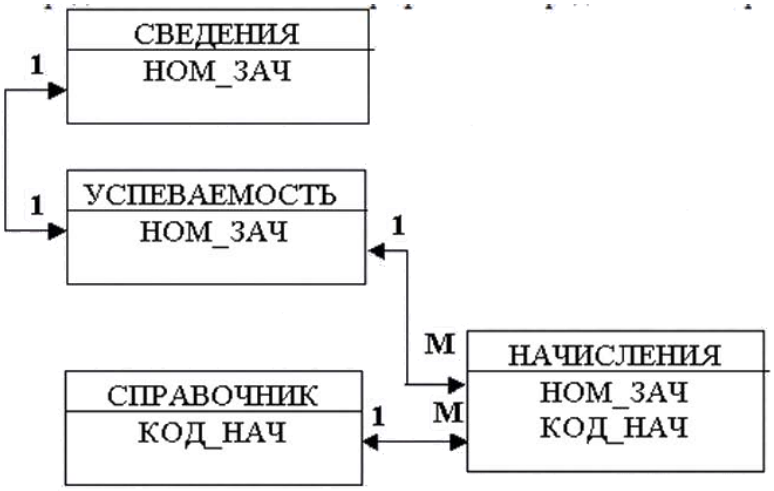


Рис. 1. Концептуальная модель предметной области

ИНСТИТУТ-БУХГАЛТЕРИЯ

На этапе *логического проектирования* выбирается СУБД для создания базы данных. Информационно-логическая модель предметной области преображается в логическую модель, основанную на структурных единицах той базы, которая реализуется в выбранной СУБД. По-

скольку дальнейшая реализация базы данных предполагается в СУБД Access, то каждый информационный объект следует представить определенной таблицей и установить связи между таблицами. Графическое изображение логической модели базы данных приведено на рис. 2.

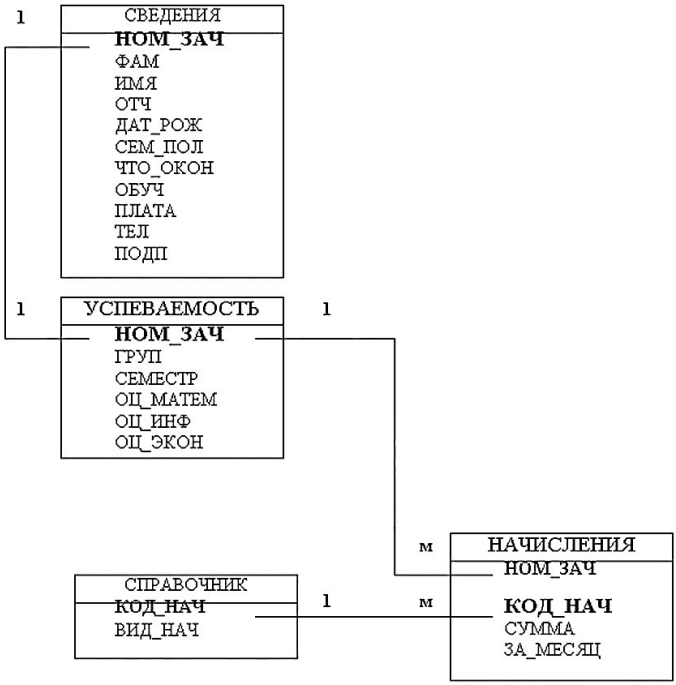


Рис. 2. Логическая модель базы данных

Далее спроектированная база данных оптимизируется, то есть осуществляется минимизация избыточности данных. С этой целью ее таблицы-отношения анализируются на соответствие требованиям нормализации.

Отношение находится *в первой нормальной форме* (1НФ), если все его поля являются простыми (то есть в клетках таблицы не должно содержаться несколько значений). Таблицы спроектированной базы данных отвечают требованиям 1НФ.

Отношение находится *во второй нормальной форме* (2НФ). если оно удовлетворяет требованиям 1НФ и неключевые поля функционально полно зависят от ключа. Полная функциональная зависимость означает, что значение каждого неключевого поля однозначно определяется значением ключа. Таблицы спроектированной базы данных отвечают требованиям 2НФ.

Отношение находится *в третьей нормальной форме* (3НФ). если оно удовлетворяет требованиям 2НФ и при этом неключевые поля зависят от ключа нетранзитивно. *Транзитивной* называется такая зависимость, при которой какое-либо неключевое поле зависит от другого неключевого поля, а то. в свою очередь, зависит от ключа. Таблицы спроектированной базы данных отвечают требованиям 3НФ.

Затем следует этап *физического проектирования.* На этом этапе база данных создается на внешних носителях информации.

Связывание таблиц (отношений) и ключи отношений

Пример.

А) *Создать связь между двумя таблицами «Покупатель» и «Заказы».*

Таблица 1 - «Покупатель»

Код покупателя	Город	Страна

Таблица 2 - «Заказы»

Код покупателя	Номер заказа	Дата заказа	Адрес

В таблице «Покупатель» ключевым полем является поле Код покупателя. Данное поле является счетчиком и содержит уникальные значения для каждой записи этой таблицы. Поле данной таблицы назовем первичным ключом.

Если каждый покупатель имеет право сделать только один заказ, то в таблице «Заказы» ключевым полем может являться аналогичное первой таблице поле-счетчик Код покупателя. В таблице «Заказы» ключевое поле Код покупателя будем называть внешним ключом.

В этом случае тип связи, установленной между первичным и внешним ключом, называют связью один к одному. Этот тип связи представлен на рисунке 3.

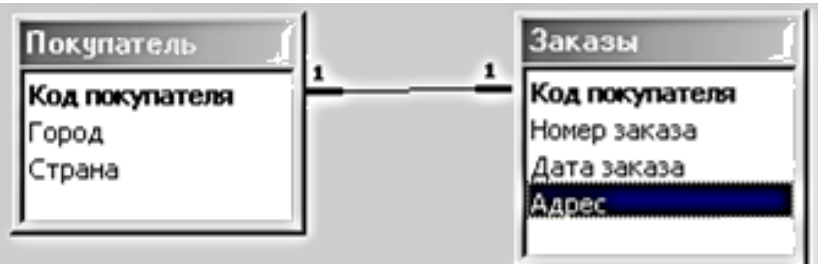


Рис. 3. Связь 1:1

На практике связи вида 1:1 используются сравнительно редко, так как имеющуюся в двух таблицах информацию легко объединить в одну таблицу, которая занимает гораздо меньше места в памяти ЭВМ.

Б) *Связь вида 1:М*

Если в таблице «Заказы» один покупатель имеет право сделать несколько заказов, то поле Код покупателя уже не будет уникальным, так как может повторяться многократно. В этом случае тип данных в этом поле может принимать числовые значения, а ключевым полем с уникальными значени-

ями может быть определено поле Номер заказа. В этом случае тип связи, установленной между одноименными полями Код покупателя в обеих таблицах, называют связью один ко многим, как показано на рисунке 4.

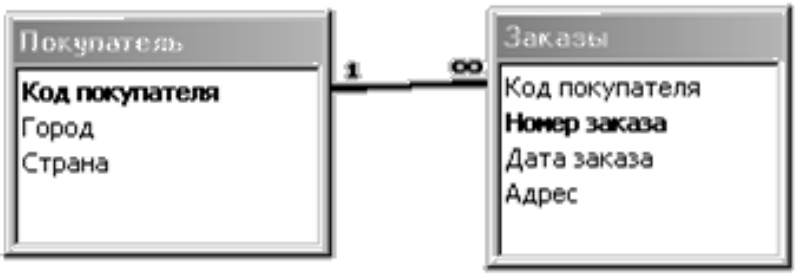


Рис. 4. Связь 1:М

В) *Связь вида М:1*.

Связь М:1 имеет место в случае, когда одной или нескольким записям основной таблицы ставится в соответствие одна запись дополнительной таблицы.

Пример связи вида М:1. Рассмотрим связь таблиц 1 и 2. В основной таблице 1 содержится информация о названиях деталей, видах материалов, из которого детали можно изготовить, и марках материала. В дополнительной таблице 2 содержатся сведения о производстве: наименование, планируемых сроках изготовления и стоимости заказов.

Таблица 1 – Продукция

Наименование продукции	Материал	Марка материала
Стол	дерево	230
Стол	пластик	340
Стул	пластик	340

Таблица 2 - Производство

Наименование продукции	Дата производства	Цена
Стол		
Стул		

Полученная связь может быть полезна при планировании или принятии управленческих решений, когда необходимо иметь все возможные варианты исполнения заказов по каждому изделию. Отметим, что таблица 1 не имеет ключей и в ней возможно повторение записей. Если таблицу 2 сделать основной, а таблицу 1 — дополнительной, получим связь вида 1:М. Поступив аналогично с таблицами 1 и 2, можно получить связь вида М:1.Отсюда следует, что вид связи (1:М или М:1) зависит от того, какая таблица является главной, а какая дополнительной.

Г) Связь вида М:М.

Пусть в основной таблице 1 содержится информация о том, на каких станках могут работать рабочие некоторой бригады. Таблица 2 содержит сведения о том, кто из бригады ремонтников какие станки обслуживает.

Таблица 1 - Цех 1

Работник цеха	Станок
Иванов	Станок1
Иванов	Станок2
Петров	Станок1
Петров	Станок3
Сидоров	Станок2

Таблица 2 - Ремонтная мастерская

Работник РМ	Станок
Голубев	Станок1
Голубев	Станок3
Зыков	Станок2
Зыков	Станок3

Исходя из определения полей связи этих таблиц можно составить новую таблицу 3 таблицы«1+2». Записям полученной таблицы можно придать смысл возможных смен, составляемых при планировании работы. Для удобства, поля новой таблицы переименованы (кстати, такую операцию предлагают многие из современных СУБД).

Таблица 3

Работник цеха	Станок	Работник РМ
Иванов А.В.	станок1	Голубев Б.С.
Иванов А.В.	станок2	Зыков А.Ф.
Петров Н.Г.	станок1	Голубев Б.С.
Петров Н.Г.	станок3	Голубев Б.С.
Петров Н.Г.	Станок3	Зыков А.Ф.
Сидоров В.К.	станок2	Зыков А.Ф.

Приведенную таблицу можно использовать, например, для получения ответа на вопрос: «Кто обслуживает станки, на которых трудится Петров Н.Г.?».

Очевидно, аналогично связи 1:1, связь M:M не устанавливает подчиненности таблиц. Для проверки этого можно основную и дополнительную таблицу поменять местами и выполнить объединение информации путем связывания. Результирующие таблицы «1+2» и «2+1» будут отличаться порядком следования первого и третьего полей, а также порядком расположения записей.

Замечание. На практике в связь обычно вовлекается сразу несколько таблиц. При этом одна из таблиц может иметь различного рода связи с несколькими таблицами. В случаях, когда связанные таблицы, в свою очередь, имеют связи с другими таблицами, образуется иерархия или дерево связей.

НОРМАЛИЗАЦИЯ БАЗЫ ДАННЫХ

В теории реляционных баз данных обычно выделяется следующая последовательность нормальных форм:

Первая нормальная форма (1NF). Переменная отношения находится в первой нормальной форме тогда и только тогда, когда в любом допустимом значении отношения каждый его кортеж содержит только одно значение для каждого из атрибутов.

Пример.

Исходная ненормализованная (то есть не являющаяся правильным представлением некоторого отношения) таблица:

Сотрудник	Номер телефона
Иванов И. И.	283-56-82 390-57-34
Петров П. П.	708-62-34

Таблица, приведённая к 1NF (являющаяся правильным представлением некоторого отношения):

Сотрудник	Номер телефона
Иванов И. И.	283-56-82
Иванов И. И.	390-57-34
Петров П. П.	708-62-34

Вторая нормальная форма (2NF). Вторая нормальная форма требует, чтобы неключевые столбцы таблиц зависели от первичного ключа в целом, но не от его части.

Пример.

Пусть в следующем отношении первичный ключ образует пара атрибутов {Сотрудник, Должность}:

Сотрудник	Должность	Зарплата	Наличие компьютера
Гришин	Кладовщик	20000	Нет
Васильев	Программист	40000	Есть
Иванов	Кладовщик	25000	Нет

Зарплату сотруднику каждый начальник устанавливает сам (хотя её границы зависят от должности). Наличие же компьютера у сотрудника зависит только от должности, то есть зависимость от первичного ключа неполная.

В результате приведения к 2NF исходное отношение следует декомпозировать на два отношения:

Сотрудник	Должность	Зарплата
Гришин	Кладовщик	20000
Васильев	Программист	40000
Иванов	Кладовщик	25000

Должность	Наличие компьютера
Кладовщик	Нет
Программист	Есть

Третья нормальная форма (3NF). Третья нормальная форма требует, чтобы в таблице не имелось транзитивных зависимостей между не ключевыми полями, то есть, чтобы значение любого поля, не входящего в первичный ключ, не зависело от другого поля, также не входящего в первичный ключ.

Пример.

Отношение находится во 2NF, но не соответствует 3NF:

Таблица 1 – Информация о сотрудниках

Сотрудник	Отдел	Телефон
Гришин	Бухгалтерия	11-22-33
Васильев	Бухгалтерия	11-22-33
Петров	Снабжение	44-55-66

В отношении атрибут «Сотрудник» является первичным ключом. Личных телефонов у сотрудников нет, и телефон сотрудника зависит исключительно от отдела.

Таким образом, в отношении существуют следующие функциональные зависимости: Сотрудник → Отдел, Отдел → Телефон, Сотрудник → Телефон.

Зависимость Сотрудник → Телефон является транзитивной, следовательно, отношение не находится в 3NF.

В результате разделения отношения Таб. 1 получаются два отношения, находящиеся в 3NF:

Таблица 2 – Информация об отделе

Отдел	Телефон
Бухгалтерия	11-22-33
Снабжение	44-55-66

Таблица 3 – Информация о сотрудниках

Сотрудник	Отдел
Гришин	Бухгалтерия
Васильев	Бухгалтерия
Петров	Снабжение

Первоначальное отношение таблицы 1 при необходимости легко получается в результате операции соединения отношений таблицы 2 и 3.

Нормальная форма Бойса-Кодда (BCNF). Таблица находится в BCNF, если она находится в 3NF, и при этом отсутствуют функциональные зависимости атрибутов первичного ключа от неключевых атрибутов.

Пример.

Необходимо разработать таблицу бронирования для теннисных кортов на день: {Номер корта, Время начала, Время окончания, Тариф, Член клуба}. Тариф зависит от выбранного корта и членства в клубе, для каждого из кортов имеется тариф для членов теннисного клуба и для сторонних клиентов. Тарифы для кортов не повторяются.
Возможны следующие составные первичные ключи: {Номер корта, Время начала}, {Номер корта, Время окончания}, {Тариф, Время начала}, {Тариф, Время окончания}.
Таблица соответствует второй и третьей нормальной форме. Требования второй нормальной формы (2NF) выполняются, так как все атрибуты входят в какой-то из потенциальных ключей, а неключевых атрибутов в отношении нет. Также нет и транзитивных зависимостей, что соответствует требованиям третьей нормальной формы (3NF).

Тем не менее, существует функциональная зависимость тарифа от номера корта. То есть, по ошибке можно нарушить логическую целостность и, например, приписать тариф Premium для первого корта, хотя тариф Premium может относиться только ко второму корту.

Можно улучшить структуру, разбив таблицу на две: {Тариф, Время начала, Время окончания} и {Тариф, Номер корта, Член клуба}. Данное отношение будет соответствовать BCNF.

Четвертая нормальная форма (4NF); Таблица находится в 4NF, если она находится в BCNF и не содер-

жит нетривиальных многозначных зависимостей.

Пример.

Предположим, что рестораны производят разные виды пиццы, а службы доставки ресторанов работают только в определенных районах города. Составной первичный ключ соответствующей переменной отношения включает три атрибута: {Ресторан, Вид пиццы, Район доставки}.

Такая переменная отношения не соответствует 4НФ, так как существует следующая многозначная зависимость: {Ресторан} $\twoheadrightarrow$ {Вид пиццы}

{Ресторан} $\twoheadrightarrow$ {Район доставки}

То есть, например, при добавлении нового вида пиццы придется внести по одному новому кортежу для каждого района доставки. Возможна логическая аномалия, при которой определенному виду пиццы будут соответствовать лишь некоторые районы доставки из обслуживаемых рестораном районов.

Для предотвращения аномалии нужно декомпозировать отношение, разместив независимые факты в разных отношениях. В данном примере следует выполнить декомпозицию на {Ресторан, Вид пиццы} и {Ресторан, Район доставки}. Однако если к исходной переменной отношения добавить атрибут, функционально зависящий от потенциального ключа, например цену с учётом стоимости доставки ({Ресторан, Вид пиццы, Район доставки} → Цена), то полученное отношение будет находиться в 4НФ и его уже нельзя подвергнуть *декомпозиции без потерь*. Указанные выше многозначные зависимости в данном случае называются внедрёнными зависимостями.

Пятая нормальная форма, или нормальная форма проекции-соединения (5NF или PJ/NF). Таблица находится в 5NF, если она находится в 4NF и любая многозначная зависимость соединения в ней является тривиальной.

Пример.

Предположим, что нужно хранить данные об ассортименте нескольких продавцов, торгующих продукцией нескольких фирм (номенклатура товаров фирм может пересекаться):

Таблица - Ассортимент (продавцы, фирмы, товары)

Продавец	Фирма	Товар
Иванов	Рога и Копыта	Пылесос
Иванов	Рога и Копыта	Хлебница
Петров	Безенчук&Ко	Сучкорез
Петров	Безенчук&Ко	Пылесос
Петров	Безенчук&Ко	Хлебница
Петров	Безенчук&Ко	Зонт
Сидоров	Безенчук&Ко	Пылесос
Сидоров	Безенчук&Ко	Телескоп
Сидоров	Рога и Копыта	Пылесос
Сидоров	Рога и Копыта	Лампа
Сидоров	Геркулес	Вешалка

Если дополнительных условий нет, то данное отношение, которое находится в 4-ой нормальной форме, является корректным и отражает все необходимые ограничения.

Теперь предположим, что нужно учесть следующее ограничение: каждый продавец имеет в своём ассортименте ограниченный список фирм и ограниченный список типов товаров и предлагает товары из списка товаров, производимые фирмами из списка фирм. То есть продавец не имеет право торговать какими угодно товарами каких угодно фирм. Если продавец П имеет право торговать товарами фирмы Ф, и если продавец П имеет право торго-

вать товарами типа Т, то в этом случае в ассортимент продавца П входят товары типа Т фирмы Ф при условии, что фирма Ф производит товары типа Т.

Такое ограничение может быть вызвано, например, тем, что список типов товаров продавца ограничен имеющимися у него лицензиями, либо знаниями и квалификацией, необходимыми для их продажи, а список фирм каждого продавца определён партнёрскими соглашениями.

В рассматриваемом примере, в частности, предполагается, что продавец Иванов имеет право торговать товарами только фирмы «Рога и Копыта», продавец Петров — товарами только фирмы «Безенчук&Ко», зато продавец Сидоров не имеет право торговать хлебницами и сучкорезами и т.д.

Предложенное выше отношение не может исключить ситуации, при которых данное ограничение будет нарушено. Ничто не препятствует занести данные о торговле товаром, который данная фирма вообще не выпускает, либо данные о торговле товарами той фирмы, которую данный продавец не обслуживает, либо данные о торговле таким типом товара, который данный продавец не имеет право продавать.

Отношение не находится в 5NF, поскольку в нём есть нетривиальная зависимость соединения *{{Продавец, Фирма}, {Фирма, Товар}, {Продавец, Товар}}, однако подмножества {Продавец, Фирма}, {Фирма, Товар}, {Продавец, Товар} не являются суперключами исходного отношения.

В данном случае для приведения к 5NF отношение должно быть разбито на три: {Продавец, Фирма}, {Фирма, Товар}, {Продавец, Товар}.

Таблица -Товары продавцов

Продавец	Товар
Иванов	Пылесос
Иванов	Хлебница
Петров	Сучкорез
Петров	Пылесос
Петров	Хлебница
Петров	Зонт
Сидоров	Телескоп
Сидоров	Пылесос
Сидоров	Лампа
Сидоров	Вешалка

Таблица - Товары фирм

Фирма	Товар
Рога и Копыта	Пылесос
Рога и Копыта	Хлебница
Рога и Копыта	Лампа
Безенчук&Ко	Сучкорез
Безенчук&Ко	Пылесос
Безенчук&Ко	Хлебница
Безенчук&Ко	Зонт
Безенчук&Ко	Телескоп
Геркулес	Вешалка

Таблица - Фирмы продавцов

Продавец	Фирма
Иванов	Рога и Копыта
Петров	Безенчук&Ко
Сидоров	Безенчук&Ко
Сидоров	Рога и Копыта
Сидоров	Геркулес

Доменно-ключевая нормальная форма (DKNF). Отношение в DKNF не имеет аномалий модификации. Другими словами, что бы ни менялось — ничего не потеряется, если соблюдены все ограничения относительно ключей и доменов. Формулировка слишком общая, но суть ее заключается в том, что если выполнять некоторые правила, то при любых действиях с таблицей ее целостность не пострадает и вся необходимая информация сохранится.

Пример.

Правила построения DKNF действуют примерно так: нельзя просто удалить категорию из таблицы категорий, если с этой категорией связаны, например, продукты из таблицы продуктов. Прежде чем удалять категорию, необходимо выполнить предварительные действия в таблице продуктов (например, поле отвечающее за id категории этого товара нужно сделать NULL).

Шестая нормальная форма (6NF). Таблица находится в 6NF, если она находится в 5NF и удовлетворяет требованию отсутствия нетривиальных зависимостей. Зачастую 6NF отождествляют с DKNF.

Пример.

Идея «декомпозиции до конца» выдвигалась до начала исследований в области хронологических данных, но не нашла поддержки. Однако для хронологических баз

данных максимально возможная декомпозиция позволяет бороться с избыточностью и упрощает поддержание целостности базы данных.

Для хронологических баз данных определены U_операторы, которые распаковывают отношения по указанным атрибутам, выполняют соответствующую операцию и упаковывают полученный результат. В данном примере соединение проекций отношения должно производится при помощи оператора U_JOIN.

Работники

Таб. №	Время	Должность	Домашний адрес
6575	[01-01-2000:10-02-2003]	слесарь	ул. Ленина, 10
6575	[11-02-2003:15-06-2006]	слесарь	ул. Советская, 22
6575	[16-06-2006:05-03-2009]	бригадир	ул. Советская, 22

Переменная отношения «Работники» не находится в 6НФ и может быть подвергнута декомпозиции на переменные отношения «Должности работников» и «Домашние адреса работников».

Должности работников

Таб. №	Время	Должность
6575	[01-01-2000:15-06-2006]	слесарь
6575	[16-06-2006:05-03-2009]	бригадир

Домашние адреса работников

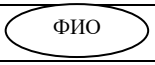
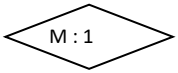
Таб. №	Время	Домашний адрес
6575	[01-01-2000:10-02-2003]	ул. Ленина, 10
6575	[11-02-2003:05-03-2009]	ул. Советская, 22

МЕТОД СУЩНОСТЬ-СВЯЗЬ (ER-МОДЕЛЬ)

Работа с базой данных начинается с построения модели предметной области. Наиболее распространенной является **ER-модель** (entity-relationship model) – метод (модель) «**Сущность-связь»**.

ER-модели можно просто в ручную, а также можно использовать специализированные программные средства (ERModeler**)**.

Базовые обозначения:

Сущность (объект)	
Атрибут сущности (свойство, характеризующее объект)	
Ключевой атрибут (атрибут, входящий в первичный ключ)	
Связь	 Работает

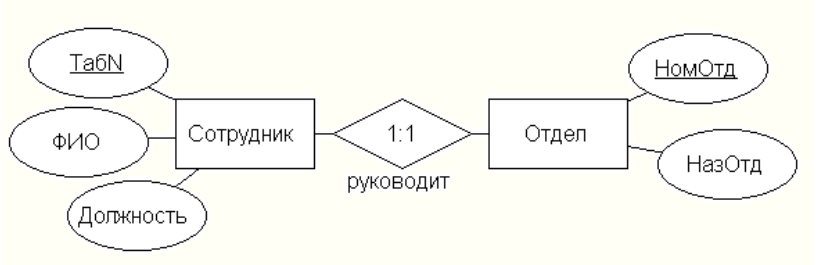
Первичный ключ может состоять из нескольких атрибутов, тогда подчеркивается каждый из них.

Объект и его атрибуты соединяются ненаправленными дугами.

Связи между объектами могут быть 4-х типов:

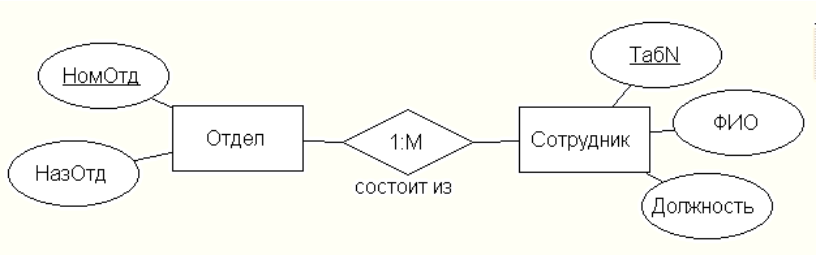
Один – к одному.

Например: сотрудник может руководить только одним отделом, и у каждого отдела есть только один руководитель.



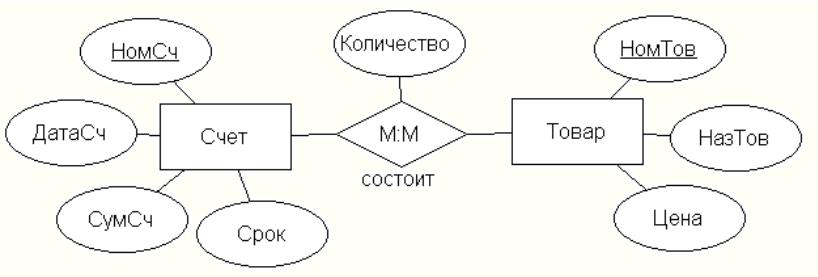
Один – ко многим *(или в обратную сторону* ***Многие – к*** *одному).*

Например: в каждом отделе может быть множество сотрудников, но каждый сотрудник работает только в одном отделе.

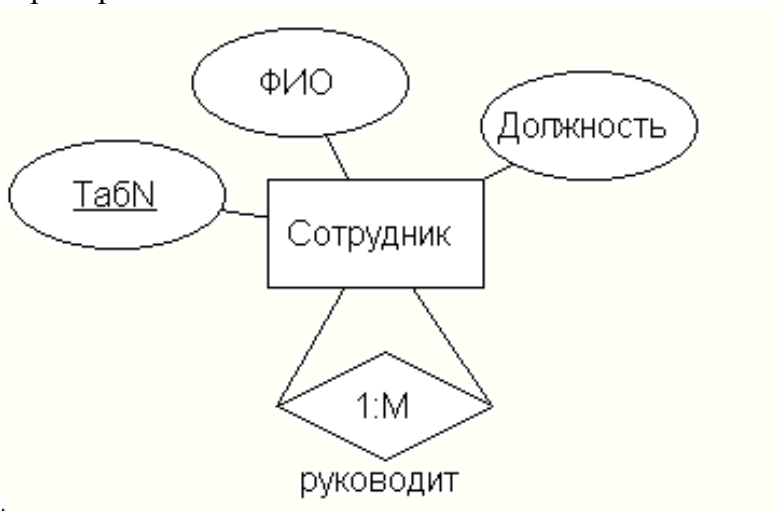


Многие – ко многим.

Например: каждый счет может включать множество товаров, и каждый товар может входить в разные счета.



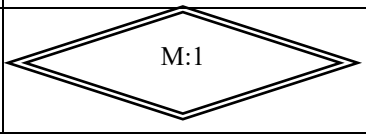
Связь может соединять сущность саму с собой, например:

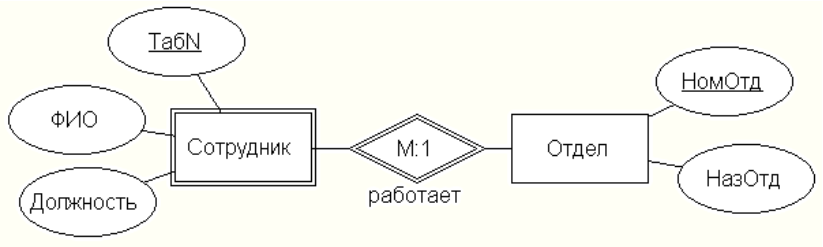


Иногда используют такое понятие, как ***слабая сущность***. Это сущность, которая не может быть однозначно идентифицирована с помощью собственных атрибутов, а только через связь с другой сущностью.

Пусть, например, номер сотрудника является уникальным только в пределах отдела, т.е. в разных отделах могут быть сотрудники с одинаковыми номерами. Уникальной в данном случае будет комбинация атрибутов «*НомерСотрудника, НомерОтдела*». Сущность «Сотрудник» является слабой.

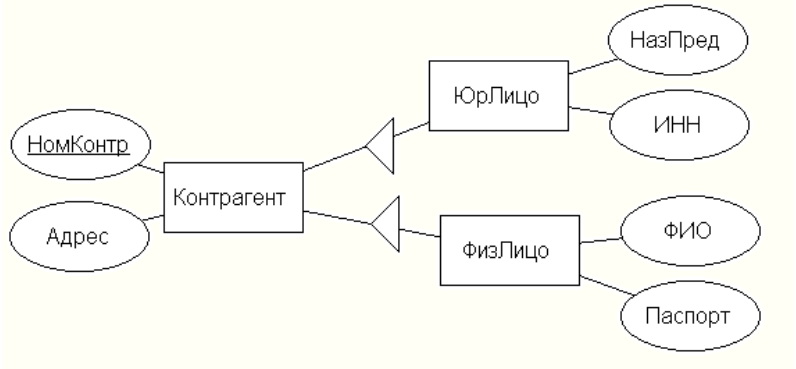
На схеме слабые сущности и их идентифицирующие связи собозначаются двойными линиями.

Слабая сущность	
Связь слабой сущности	



Этот-же пример связи ***многое – к одному***.

Иногда для более удобной классификации используются так называемые ***подтипы сущностей*.** Их обозначают с помощью треугольника, направленного от подтипа к надтипу. Этот треугольник может сопровождаться надписью «есть» или «is a» (т.е., «является»).

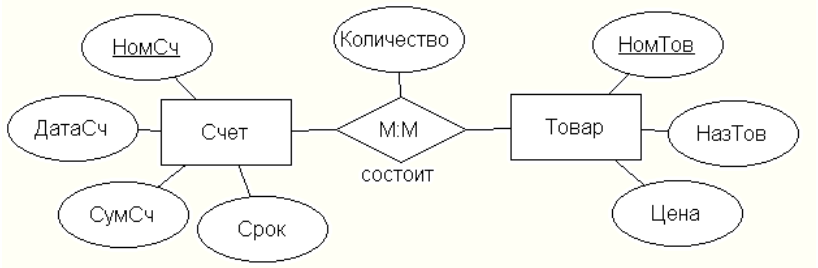


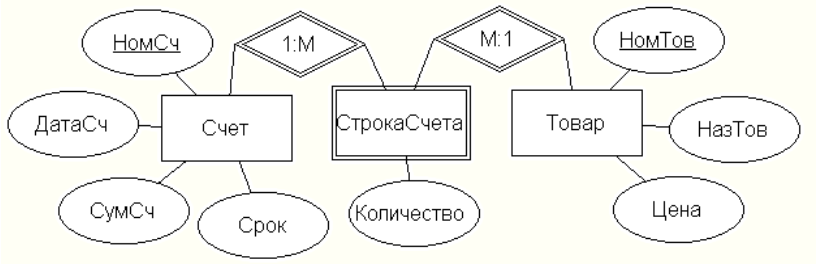
Например, среди контрагентов могут быть как физические, так и юридические лица. Поскольку они имеют разные атрибуты, то удобно создать для них подтипы.

Сущность «Контрагент» является ***надтипом*** для своих подтипов. Обратите внимание, что у подтипов обычно не бывает собственных первичных ключей.

Примечание по поводу связи M:M

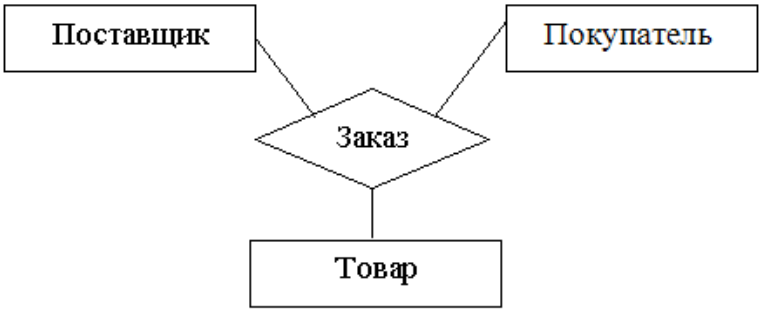
На самом деле этот тип связи представляет собой «замаскированную» слабую сущность, которая связана с другими двумя сущностями идентифицирующими связями многие – к одному:





Если связь соединяет две сущности, она называется ***бинарной***.

Связь может соединять более двух сущностей, например, связь, соединяющая три сущности, называется ***тернарной:***



Связь более 2 обычно имеет тип **многие – ко многим** по отношению ко всем связанным сущностям.

КОНЦЕПТУАЛЬНОЕ И ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

Выполнить проецирование и разработку базы данных **для предметной** области «Склад». Для этого необходимо:

1. Определение типов сущности;
2. Определение типов связи;
3. Определение атрибутов и связывание с типами сущностей и **свази;**
4. Определение доменов атрибутов;
5. Определение атрибутов являющихся потенциальным первичными ключами;
6. Преобразование локально-концептуальной модели в локально - логическую модель данных;
7. Проверка модели с помощью правил нормализации;
8. Проверка модели в отношение транзакций **пользователя** и выполнение запроса;
9. Построение диаграммы сущность-связь;
10. Физическая реализация спроектированной БД в среде реализация СУБД MS Access.

Решение

Определение типов сущностей

Из набора атрибутов выберем наиболее крупные объекты представляющие интерес для пользователя: поставщик, заказчик, товар, МОЛ (Таблица 1).

Таблица 1 - Сведения о типах сущностей

Тип сущности	Описание	Псевдонимы	Особенности использования
Поставщик	Информация о поставщике	Продавец, производитель	Поставщик может поставлять несколько видов товара
Заказчик	Информация о заказчике	Потребитель, покупатель	Каждый заказчик может заказать несколько товаров
Товар	Информация о товаре	Продукт	Каждый товар может быть заказан несколькими заказчиками. Товар может поставляться несколькими поставщиками. Товар может храниться только у 1 МОЛ
МОЛ	Информация о материально ответственном лице	Заведующий складом, сторож	Каждое МОЛ принимает на хранение несколько видов товара

Определение типов связей

Далее определяют типы связей и присваиваются им осмысленные имена, которые должны быть понятны многим. Полученные связи представлены в таблице «Сведения о типах связей» (табл. 2).

Тип сущности	Значение связи	Тип сущности	Тип связи
Поставщик	Поставляет	Товар	M:M
Заказчик	Покупает	Товар	M:M
МОЛ	Принимает	Товар	1 :M

Определение атрибутов и связывание их с типами сущностей и связей

Атрибуты бывают:

- простые - имеют одно уникальное имя
- составные - состоят из простых атрибутов
- производные - атрибуты, значения которых могут быть установлены с помощью значений других атрибутов

Для определения атрибутов выделенных сущностей

необходимо из описания предметной области выбрать все существительные содержащие их фразы. Каждому выделенному атрибуту следует присвоить свое уникальное имя (Таблица 3).

Таблица 3 - Сведения об атрибутах Поставщик

Атрибут	Тип данных	Вид атрибута	Синонимы	Описание
Код поставщика	Счетчик	Простой	Номер	Порядковый номер поставщика
Фирма поставщик	Текстовый	Простой	Организация	Название фирмы поставщика
Реквизиты	Числовой	Простой	Счет в банке	Банковские реквизиты поставщика
Адрес	Текстовый	Составной (город, улица, № дома)	Месторасположение	Адрес поставщика
Телефон	Текстовый	Простой	Контактный номер	Телефон поставщика
E-mail	Текстовый	Простой	Электронный адрес	Адрес электронной почты
Представитель	Текстовый	Составной (фамилия, имя, отчество)	Контактное лицо	ФИО представителя

Сведения об атрибутах Заказчик

Атрибут	Тип данных	Вид атрибута	Синонимы	Описание
Код заказчика	Счетчик	Простой	Номер	Порядковый номер заказчика
Фирма заказчик	Текстовый	Простой	Организация	Название фирмы заказчика
Реквизиты	Числовой	Простой	Счет в банке	Банковские реквизиты поставщика
Адрес	Текстовый	Составной (город, улица, № дома)	Месторасположение	Адрес поставщика
Телефон	Текстовый	Простой	Контактный помер	Телефон поставщика- у^/л:-<.
E-mail	Текстовый	Простой	Электронный адрес	Адрес электронной почты
Представитель	Текстовый	Составной (фамилия, имя, отчество)	Контактное лицо	ФИО представителя
Дата заказа	Дата, время	Простой	Число, месяц, год	Дата совершения сделки

Сведения об атрибутах Товар

Атрибут	Тип данных	Вид атрибута	Синонимы	Описание
Код товара	Счетчик	Простой	Номер	Числовой идентификатор товара
Товар	Текстовый	Простой	Продукт	Название товара
Единицы измерения	Текстовый	Простой	Штук	В чем измеряется количество товара
Количество	Числовой	Простой	Объем	Товар, выраженный в ед. измерения
Цена	Денежный	Простой	Стоимость	Стоимость товара
Способ доставки	Текстовый	Простой	Способ отгрузки	Название способа доставки
Цена доставки	Денежный	Простой	Стоимость	Стоимость доставки товара 1

Сведения об атрибутах МОЛ

Атрибут	Тип данных	Вид атрибута	Синонимы	Описание
Код МОЛ	Счетчик	Простой	Номер	Порядковый номер МОЛ
МОЛ	Текстовый	Составной (фамилия, имя, отчество)	Сторож, охранник	ФИО материально ответственного лица
№ паспорта	Числовой	Простой	Паспортные данные	Номер паспорта МОЛ
Адрес	Текстовый	Составной (город, улица, № дома)	Место жительства	Место жительства МОЛ
Телефон	Текстовый	Простой	Контактный номер	Телефон МОЛ
Дата рождения	Дата, время	Простой	Число, месяц, год	Дата рождения МОЛ

Определение доменов атрибутов

Задача этого этапа построения локальной концептуальной модели данных - определение доменов атрибутов для всех атрибутов, присутствующих в спецификации на проекте. Домен - это набор допустимых значений одного или нескольких атрибутов.

Выделим в отдельную таблицу (табл. 4) домены и сведения о них, характерные для разрабатываемой базы данных.

Таблица 4 - Сведения о доменах атрибутов

Имя домена	Характеристика домена	Примеры допустимых значений
Фирма (поставщик)	Строка произвольной длины до 30 символов	ООО "Мираж"
Реквизиты	Строка фиксированной длины в 20 символов	69420751096004812460
Адрес	Строка произвольной длины до 100 символов	Брянская обл., Выгонический р-н, с. Кокино, ул. Советская 4,
Телефон	Строка фиксированной длины в 10 символов	(952) 963-30-31
E-mail	Строка произвольной длины до 30 символов	mirag@mail.ru
Представитель	Строка произвольной длины до 40 символов	Иванов Иван Иванович
Товар	Строка произвольной длины до 30 символов	Книги
Единицы измерения	Строка произвольной длины до 10 символов	шт
Количество	Строка произвольной длины до 10 символов	100
Цена	Строка произвольной длины до 10 символов	300 рублей
Дата	Строка фиксированной длины в 8 символов	31.10.2011
Способ доставки	Строка произвольной длины до 30 символов	самовывоз
МОЛ	Строка произвольной длины до 50 символов	Петров Петр Петрович
№ паспорта МОЛ	Строка фиксированной длины в 10 символов	1849 - 664521

Определение атрибутов, являющихся потенциальными и первичными ключами

Производится определение атрибутов, являющихся потенциальными и первичными ключами. Кроме того, выделяются атрибуты, являющиеся альтернативными и внешними ключами.

Для уже созданных сущностей в качестве первичных ключей создадим искусственные первичные ключи

(табл. 5), так как нет атрибутов, которые можно было бы выбрать в качестве естественных первичных ключей.

Таблица 5 - Сведения о первичных и альтернативных ключах

Тип сущности	Первичный ключ	Альтернативный ключ	Внешний ключ
Поставщик	Код поставщика	Реквизиты	Код товара
Заказчик	Код заказчика	Реквизиты	Код товара
Товар	Код товара	-	Код МОЛ
МОЛ	Код МОЛ	№ паспорта	-

Создание диаграммы «сущность-связь»

Цель: Разработка диаграмм сущность - связь содержащих концептуальное отражение представление пользователя о предметной области приложения.

Где описывается тип сущности, тип связи и сущность - связь.

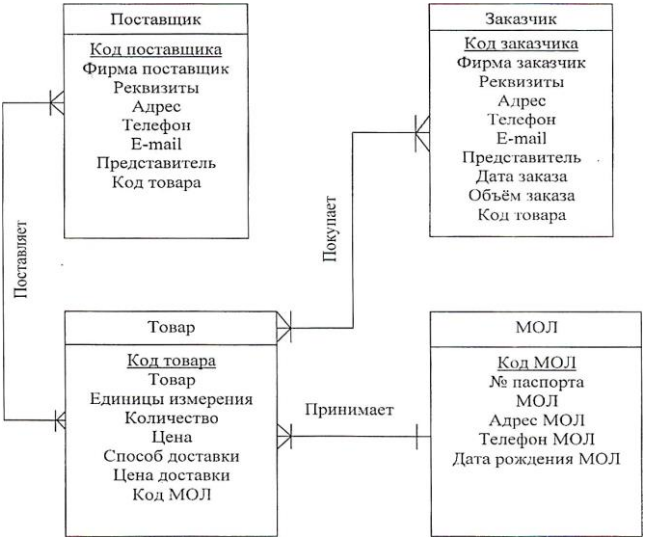


Рис. 1. ER – диаграмма «сущность-связь»

Логическое проектирование структуры базы данных

Цель - построение логической модели данных на основе концептуальной модели данных, отражающей представление отдельного пользователя о предметной области приложения, и проверка полученной модели с помощью методов нормализации и контроля выполнения транзакций.

Основная задача состоит в доработке этих моделей с целью удаления из них всех элементов, затрудняющих реализацию данных моделей в среде реляционных СУБД. В результате выполнения этих действий, структуры концептуальных моделей данных будут изменены таким образом, чтобы полностью отвечать требованиям, выдвигаемым реляционной моделью организации баз данных.

Преобразование локальной концептуальной модели данных логическую модель

Необходима доработка локальной концептуальной модели с целью удаления из не нежелательных элементов и преобразование полученных моделей в локальные логические модели данных.

Первым этапом является удаление связи «многие ко многим». Такая связь наблюдается дважды в данной предметной области между сущностями ПОСТАВЩИК и ТОВАР, и между ТОВАРОМ и ЗАКАЗЧИКОМ. Для удаления такого типа связи введем в первый вариант еще одну сущность ПОСТАВКА. Введенная сущность будет содержать следующие атрибуты: код поставщика, код товара два этих атрибута будут являться внешними ключами.

Рис. 2. Связь «Поставщик-Поставка-Товар»

Для разбития второй связи введем другую сущность ЗАКАЗ. Введенная сущность будет содержать следующие атрибуты: код заказчика, код товара эти два атрибута будут являться внешними ключами.

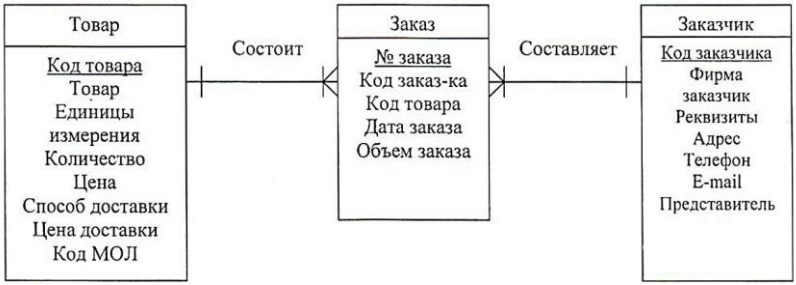


Рис. 3. Связь «Товар-Заказ-Заказчик»

При необходимо удалить сложные и рекурсивные связи. Сложной называется связь, существующая между тремя и больше типами сущностей. Такую связь следует устранять с помощью промежуточной сущности. Сложная связь заменяется необходимым количеством бинарных связей типа 1:M. Рекурсивные связи - это связь между сущностью и ее же самой.

Далее происходит удаление связей с атрибутами -

это связь, имеющая дополнительные характеристики.

На следующем этапе происходит удаление множественных атрибутов - множественный атрибут следует удалить путем введения новой сущности.

Затем необходима перепроверка связей типа 1:1.

На последнем этапе происходит удаление избыточных связей - связь является избыточной, если одна и та же информация может быть получена не только через нее, но и при помощи другой связи.

В результате получаем упрощенную концептуальную модель данных, из которой удалены все структуры, реализация которых в среде реляционных СУБД затруднительна.

Проверка моделей с помощью правил нормализации

Основной задачей данного этапа является проверка корректности состава каждого из созданных отношений посредством применения к ним процедуры нормализации. Процесс нормализации включает следующих три основных этапа:

- приведение к 1НФ, позволяющее удалить из отношений повторяющиеся группы данных;
- приведение к 2НФ, позволяющее устранить частичную зависимость атрибутов от первичного ключа;
- приведение к 3НФ, позволяющее устранить транзитивную зависимость атрибутов от первичного ключа.

Первая нормальная форма (1НФ) гласит, что на пересечении каждого столбца и каждой строки должно находиться только одно значение. Чтобы привести ненормализованную таблицу к 1НФ, нужно:

1) «выровнять» таблицу, т.е. заполнить пустые ячейки копиями повторяющихся групп данных;

2) выделить повторяющиеся группы данных в отдельные отношения.

2НФ гласит, что отношение должно находиться в 1НФ и не иметь частичной функциональной зависимости атрибутов от составного первичного ключа.

3НФ гласит, что отношение должно находиться во 2НФ и не содержать транзитивной зависимости.

Удалим транзитивную зависимость в отношение Товар. Для этого перенесём атрибуты Способ доставки и Цена доставки из отношения Товар в отношение Заказ.

1. Поставщик: Код поставщика, фирма поставщик, реквизиты поставщика, адрес поставщика, телефон поставщика, e-mail поставщика, представитель поставщика;

2. Заказчик: Код заказчика, фирма заказчик, реквизиты заказчика, адрес заказчика, телефон заказчика, e-mail заказчика, представитель заказчика;

3. Заказ: № заказа. Код заказчика, Код товара, дата заказа, объем заказа, способ доставки, цена доставки;

4. Поставка: № поставки. Код поставщика, Код товара;

5. Товар: Код товара, единица измерения, количество, цена, код МОЛ;

6. Материально ответственное лицо (МОЛ): Код МОЛ, материально ответственное лицо (МОЛ), № паспорта МОЛ, адрес МОЛ, телефон МОЛ, дата рождения МОЛ.

Построение окончательной диаграммы "сущность-связь"

Целью этапа является создание окончательного варианта диаграммы сущность-связь, являющихся локальным логическим представлением данных, используемых отдельными пользователями приложения. Данные на этих диаграммах были проверены с применением методов нормализации, а также проконтролированы на предмет возможных ошибок.

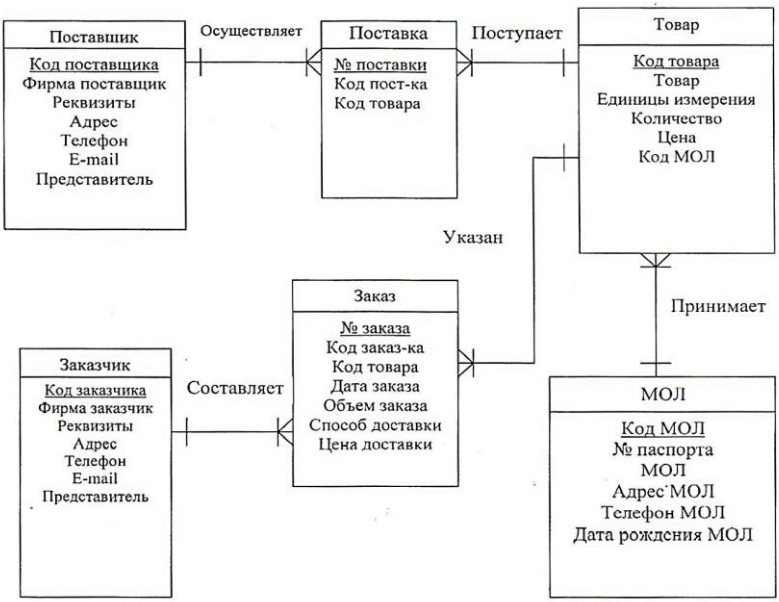


Рис. 4. Логическая модель базы данных

Список литературы

	Авторы, составители	Заглавие	Издательство, год	Количество
6.1.1. Основная литература				
Л1.1	Борзунова Т.Л.	Базы данных освоение работы в MS Access 2007 [Электронный ресурс] : электронное пособие / Т.Л. Борзунова, Т.Н. Горбунова, Н.Г. Дементьева. — Электрон. текстовые данные. — Саратов: Вузовское образование, 2014. — 148 с. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/20700.html	Саратов: Вузовское образование, 2014.	ЭБС «IPRbooks»
Л1.2	Швецов В.И.	Базы данных [Электронный ресурс] / В.И. Швецов. — Электрон. текстовые данные. — М. : Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 218 с. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/52139.html	М. : Интернет-Университет Информационных Технологий (ИНТУИТ), 2016	ЭБС «IPRbooks»
Л1.3	Медведкова И.Е.	Базы данных [Электронный ресурс] : учебное пособие / И.Е. Медведкова, Ю.В. Бугаев, С.В. Чикунов. — Электрон. текстовые данные. — Воронеж: Воронежский государственный университет инженерных технологий, 2014. — 104 с. — 978-5-00032-060-0. — Режим доступа: http://www.iprbookshop. ru/47418.html	Воронеж: Воронежский государственный университет инженерных технологий, 2014.	ЭБС «IPRbooks»

Л1.4	Селина Е.Г.	Селина Е.Г. Создание реляционных баз данных средствами СУБД Microsoft Access [Электронный ресурс] : учебно-методическое пособие / Е.Г. Селина. — Электрон. текстовые данные. — СПб. : Университет ИТМО, 2016. — 46 с. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/68137.html	СПб. : Университет ИТМО, 2016.	ЭБС «IPRbooks»

6.1.2. Дополнительная литература

	Авторы, составители	Заглавие	Издательство, год	Количество
Л2.1	Советов Б.Я.	Базы данных. Теория и практика : учеб. для бакалавров. 463 с.	М. :Юрайт, 2012.	10
Л2.2		Официальный учебный курс Microsoft: Microsoft Office Acces 2003 - 528 с.	М. : Эком, 2006.	2
Л2.3	Советов Б.Я.	Базы данных. Теория и практика : учеб. для бакалавров / 2-е изд. 463 с.	М.: Юрайт, 2012	10
Л2.4	Карчевский Е.М., Филиппов И.Е.	Access 2010 в примерах. [Электронный ресурс]: пособие/ Режим доступа: http://window.edu.ru/ resource/066/76066	Казанский федеральный университет, 2011	ЭБС Единое окно
Л2.5	Ожерельева М. В.	Базы данных: электронное учебно-метод. пособие	Брянск: БГСХА, 2013	CD-Диск

6.1.3. Методические разработки				
	Авторы, составители	Заглавие	Издательство, год	Количество
Л3.1	Лысенкова С.Н.	Методические указания для выполнения самостоятельных работ по курсу «Базы данных» для подготовки бакалавров очной формы обучения по направлению 09.03.03 Прикладная информатика в экономике	Брянск: Издательство Брянский ГАУ, 2017	50

Учебное издание

Лысенкова Светлана Николаевна

ОСНОВЫ ПРОЕКТИРОВАНИЯ БАЗ ДАННЫХ

Учебно-методическое пособие для студентов направления подготовки 09.03.03. Прикладная информатика

Редактор Лебедева Е.М.

Подписано к печати 15.11.2019 г. Формат 60х84 $^{1}/_{16.}$
Бумага офсетная. Усл. п. л. 3,83. Тираж 25 экз. Изд. № 6563.

Издательство Брянского государственного аграрного университета
243365 Брянская обл., Выгоничский район, с. Кокино, Брянский ГАУ