МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ
УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)» (СГАУ)

Потапов И. В.

Информационные технологии на транспорте

Электронный ресурс
Конспект лекций

САМАРА
2013

УДК 004.9(075)
ББК 32.887
П 64

Потапов И. В. Информационные технологии на транспорте [Электронный ресурс] : конспект лекций / И. В. Потапов, М-во образования и науки РФ, Самар. гос. аэрокосм. ун-т им. С. П. Королева (нац. исслед. ун-т) - Электрон. текстовые и граф. дан. - Самара, 2013. – 1 эл. опт. диск (CD-ROM).

Конспект лекций «Информационные технологии на транспорте» предназначен для студентов факультета инженеров воздушного транспорта, обучающихся по направлению подготовки бакалавров 190701.62 «Технология транспортных процессов» в 4 семестре.

Конспект лекций разработан на кафедре организации и управления перевозками на транспорте.

1 Понятие информации

В процессе своего развития человечество в любой сфере деятельности последовательно проходило стадии от ручного кустарного труда до высокотехнологичного промышленного производства. В первую очередь усилия были направлены на облегчение физического труда, а информационная сфера долгие годы была уделом умственного труда человека и с каждым годом требовала большего количества трудовых ресурсов.

Появление ЭВМ и сетей передачи данных способствовало революционным процессам в области информатизации и позволило перейти на промышленный уровень технологий и инструментальных средств.

На основе информационных технологий решается задача автоматизации информационных процессов. Информация, как продукт информационных технологий, в значительной степени структурируется и формируется в виде знаний. В любой предметной области, а также в обществе в целом, выделяется как самостоятельный компонент, информационный ресурс, приобретающий материальный характер.

Термин «информация» происходит от латинского слова «informatio» – разъяснение, изложение, осведомленность.

1.1 Проблема информации: различные точки зрения

Существуют три точки зрения на проблему информации.

1) Понятие «информация» отождествляется со *знанием*. Хотя данный подход широко критикуется в отечественной литературе, во многих научных трудах он имеет место.

2) Предметная область понятия «информация» ограничивается *социальными* и *биологическими* процессами; существование информационных процессов в неорганической природе отвергается.

3) Связана с *атрибутивным* понятием информации (широко используется в настоящее время). Впервые атрибутивное понятие информации было сформулировано Н. Винером, полагавшим, что все явления в природе охватываются тремя основными понятиями: вещество, энергия, информация. В отличие от Н.Винера, не рассматривавшего взаимосвязь этих компонентов, многие современные авторы тесно увязывают их и рассматривают как единую систему.

Понятие информации должно быть *связано* с определенным *объектом*, свойства которого она отражает.

Кроме того, имеется относительная *независимость* информации от носителя, поскольку возможны *преобразование* информации и *передача* по различным физическим средам с помощью разнообразных физических сигналов независимо от ее содержания (семантики) Это явилось центральным вопросом многих исследований, в том числе и в области философии.

Информация о любом материальном объекте может быть получена двумя путями:

1) наблюдения, натурного либо вычислительного эксперимента,

2) на основе логического вывода.

Поэтому различают:

1) доопытную (априорную) информацию;

2) послеопытную (апостериорную), полученную в результате эксперимента.

При обмене информацией имеют место:

источник – объект материального мира;

приемник – человек либо какой-то материальный объект.

Информация возникает за счет *отражения*, которое является свойством всей материи, любой материальной системы.

Свойство отражения совершенствуется по мере развития материи от элементарного отражения до высшей его формы – сознания.

Процесс отражения означает взаимодействие объектов материального мира.

Информация – результат отражения.

Информация отображает некоторый образ реального мира, который в дальнейшем может существовать независимо от материального объекта.

Для описания объектов (естественных либо искусственно созданных) используют *информационные модели*, которые в дальнейшем могут быть исходным материалом для разработки систем.

Эти модели должны быть *адекватны реальным объектам*.

Любое исследование сопровождается большим объемом информации, которая требует обработки, представления и использования зачастую в реальном масштабе времени.

Таким образом, понятие информации предполагает наличие двух объектов – источника информации и потребителя.

1.2 Прагматический, семантический и синтаксический аспекты информации

Важно, чтобы информация для потребителя имела смысл. Потребитель информации может ее оценивать в зависимости от того, где и для какой конкретной задачи информация используется.

Поэтому выделяют такие аспекты информации, как прагматический, семантический и синтаксический.

Прагматический аспект связан с возможностью достижения поставленной цели с использованием получаемой информации.

Этот аспект информации влияет на поведение потребителя. Если информация была эффективной, то поведение потребителя меняется в желаемом направлении, т.е. информация имеет прагматическое содержание. Таким образом, этот аспект характеризует поведенческую сторону проблемы.

Семантический аспект позволяет оценить смысл передаваемой информации и определяется семантическими связями между словами или другими смысловыми элементами языка.

Синтаксический аспект информации связан со способом ее представления. В зависимости от реального процесса, в котором участвует информация (осуществляется ее сбор, передача, преобразование, отражение, представление, ввод или вывод), она представляется в виде специальных знаков, символов.

1.3 Виды информации

Все виды деятельности человека по преобразованию природы и общества сопровождаются получением новой информации.

Научная информация – логическая информация, адекватно отображающая объективные закономерности природы, общества и мышления.

Делится по областям получения или использования на следующие виды: политическая, техническая, биологическая, химическая, физическая и т.д.; по назначению – на массовую и специальную.

В сфере техники при решении производственных задач используется *техническая* информация. Она сопровождает разработку новых изделий, материалов, конструкций, агрегатов, технологических процессов.

Научную и техническую информацию объединяют термином *научно-техническая* информация.

Документальная информация – часть информации, которая занесена на бумажный носитель. Любое производство при функционировании требует перемещения документов, т. е. возникает документооборот.

Верхним уровнем информации как результата отражения окружающей действительности (результата мышления) являются знания.

Знания возникают как итог теоретической и практической деятельности.

Информация в виде знаний отличается высокой структуризацией.

Это позволяет выделить полезную информацию при анализе окружающих нас физических, химических и прочих процессов и явлений.

На основе структурирования информации формируется информационная модель объекта. По мере развития общества информация как совокупность научно-технических данных и знаний превращается в базу системы информационного обслуживания научно-технической деятельности общества.

С развитием общества возникает необходимость целесообразной организации информационного ресурса, т.е. концентрации имеющихся фактов, данных и знаний по направлениям науки и техники.

Признание информации как ресурса и появление понятия информационный ресурс дало толчок развитию нового научного направления –информатики. Информатика как область науки и техники связана со сбором и переработкой больших объемов информации на основе современных программно-аппаратных средств вычислительной техники и техники связи.

Важным аспектом информации является ее главенствующая роль в процессе управления.

Круг объектов управления чрезвычайно широк и разнообразен: экономика, территория, социальная сфера, производство, научный эксперимент, образование и др.

1.4 Иерархия информации

При анализе процесса управления ввиду сложности объекта осуществляют его расчленение на части по различным признакам. Одним из главных признаков является вид иерархии.

Характерны следующие виды иерархии: временная, пространственная, функциональная, ситуационная и информационная.

Следует отметить, что деление какой-либо системы на части не может быть однозначным, так как выделение границ между частями является всегда в какой-либо мере субъективным.

Выбор того или иного принципа выделения составных частей должен удовлетворять следующим основным условиям: обеспечивать их максимальную автономность; учитывать необходимость координации их действий для

достижения общей цели функционирования, а также совместимость отдельных частей.

Временная иерархия. Признаком деления является *интервал времени* от момента поступления информации о состоянии объекта управления до момента выдачи управляющего воздействия.

Чем больше интервал, тем выше уровень (ранг) элемента.

Управление может осуществляться в реальном времени с интервалом, равным суткам, декаде, месяцу, кварталу и т.д.

При этом управляющий интервал выбирается не произвольно, а исходя из критериев, определяющих устойчивость и эффективность функционирования всей системы.

Пространственная иерархия. Признаком деления является *площадь*, занимаемая объектом управления.

Чем больше площадь объекта, тем выше его ранг.

Данный признак является субъективным, так как не всегда площадь, занимаемая объектом, соответствует ее значимости, и ее можно использовать в случае аналогичности параметров элементов одного уровня.

Функциональная иерархия. В ее основе лежит *функциональная зависимость* (подчиненность) элементов системы.

Такое разделение также является субъективным, так как в этом случае трудно выделить границы между элементами системы.

Ситуационная иерархия. Признаком деления является *эффект*, вызываемый той или иной ситуацией, например, ущерб, возникающий в результате аварии или выхода из строя оборудования.

Информационная иерархия. В настоящее время этот вид иерархии является очень существенным в связи с возросшим значением информации для управления. В основе деления на уровни лежит *оперативность* и *обновляемость* информации.

Через эти характеристики прослеживается иерархия информации по уровням управления предприятием.

На первом уровне хранится и обрабатывается повторяющаяся, часто обновляющаяся информация, необходимая для повседневной деятельности, т.е. для оперативного управления.

Следующий уровень составляет информация более обобщенная, чем оперативная, и используемая не так часто.

Информация группируется по функциональным областям и применяется для поддержки принятия решения по управлению производством.

На верхнем уровне хранится и обрабатывается стратегическая информация для долгосрочного планирования. Для нее характерны высокая степень обобщенности, неповторяемость, непредсказуемость и редкое использование.

Учет информации об объекте управления состоит в *регистрации*, *классификации* и *идентификации*.

На основе разнообразных математических моделей, описывающих реальное и требуемое состояние объекта, и критериев оптимальности анализируют информацию о состоянии объекта управления.

Окончательная модель прогнозируемого состояния объекта управления формируется в виде плана. Возникающие за счет внешних воздействий отклонения от плана корректируют путем сравнения учетной и плановой информа-

ции, нового анализа и формирования управляющих воздействий (регулирования).

В большинстве случаев при информационном анализе процесса управления обычно рассматривают пассивную форму проявления информации, отражающую свойства внешней среды, объекта управления и самой управляющей системы.

Однако не менее важное значение имеет и активная форма информации, являющаяся причиной изменения состояния управляемого объекта.

1.5 Формы проявления информации

Принято выделять следующие качественно различимые формы проявления информации: осведомляющую, преобразующую, принятия решения и управляющую.

1) *Осведомляющая форма* – информация о состоянии внешней среды, объекта управления и управляющей системы.

2) *Преобразующая форма* – информация, содержащаяся в алгоритмах управления.

3) *Информация принятия решения* – отражение образов и целей на конечное множество принимаемых решений.

4) *Управляющая форма* – информация, вызывающая целенаправленное изменение состояния объекта управления.

В любой системе управления можно выделить два информационных канала: целевой и рабочий.

В *целевом* канале на основе информационных процессов происходит выбор цели и принятие решения по выбору управляющего воздействия.

В *рабочем* канале формируется информация, реализуемая исполнительным органом, осуществляющим целенаправленное изменение состояния объекта управления через вещественно-энергетические характеристики.

Целевой канал может находиться как на одном уровне иерархии с рабочим, так и на более высоком.

2. Информационные системы

Эффективное управление, организация и планирование работы транспорта требует сбора, передачи и обработки значительных объемов информации.

Чем выше скорость и качество сбора, передачи и обработки управленческой информации, тем быстрее руководители на разных уровнях управления реагируют на изменение ситуации в транспортной системе, и тем больше времени остается на ее анализ и принятие наилучшего решения.

Современный уровень развития персональных компьютеров, вычислительных сетей и программного обеспечения позволяет создавать информационные системы (ИС) для оперативного управления сложными транспортными объектами.

Современное состояние мировой экономики неразрывно связано с интенсивным развитием *логистики* как одной из эффективных форм интеграции снабжения, производства, транспорта и распределения рынка с широким привлечением современных *информационных технологий*.

В реализации основополагающего принципа логистики доставки грузов «*just-in-time*» (*«точно в срок»*) при минимальных затратах трудовых и матери-

альных ресурсов, первостепенное значение занимает раздел транспортной логистики, решение задач которой невозможно без широкого использования и развития систем электронного *обмена и обработки данных*.

Существует различие между понятиями «Данные» и «Информация».

«Данные» – сведения о некоторой сущности, зафиксированные в виде значений и хранящиеся на некоторых носителях – набор конкретных значений, параметров, характеризующих объект, ситуацию, другие факторы:

Петров Н.С., 20 руб. 12.04.1961

«Данные» не обладают определенной структурой, они становятся содержательной *информацией* тогда, когда

- им после осмысленной обработки задается определенная структура,
- происходит представление в нужное время в нужном месте.

Далее эти понятия будут подменять друг друга.

Основные идеи современной *информационной технологии* строятся на концепции *Баз Данных (БД)*.

Согласно ей, основой информационной технологии являются данные, которые должны быть организованы в БД с целью адекватного отображения изменяющегося реального мира и удовлетворения информационных потребностей пользователя.

Любое предприятие можно представить как информационную систему, состоящую из элементов и связей между ними, по которым циркулирует некоторая информация, определенным образом представленная, перерабатываемая, передаваемая.

Создание искусственных систем по переработке информации и автоматизации управления стало возможным с появлением ЭВМ.

ИС (*автоматизированная ИС****)*** – система, реализующая автоматизированный сбор, обработку и манипулирование данными и включающая технические средства обработки данных, программное обеспечение и технический персонал.

Термины ИС, автоматизированная система управления (АСУ), автоматизированная ИС (АИС), автоматизированная система (АС) в дальнейшем используются как синонимы.

ИС – это программно-аппаратный комплекс, предназначенный для *фиксации данных* о деятельности предприятия и обеспечения пользователей достоверной и своевременной информацией для принятия решений и автоматизации управления.

Основная задача ИС – качественно удовлетворять информационные потребности своих пользователей.

ИС обеспечивает выполнение следующих функций:

1) надежное хранение информации в памяти ЭВМ;

2) выполнение специфических преобразований информации и вычислений;

3) предоставление пользователям удобного интерфейса.

Существуют две внутренние информационные причины необходимости создания таких систем.

1. Достижение человеческим обществом состояния, когда все население Земли вместе не может переработать возросшие потоки и количество информации (академик Глушков В.М., 70-е гг).

2. Противоречие между своевременностью и достоверностью информации в управлении: пусть объект находится в состоянии S_0 в момент времени t_0

(S_0, t_0) ———— **à** (S_1, t_1)

Для выработки управляющего воздействия на объект необходимо собрать информацию $\mathbf{i}_0$. Но за время ее сбора объект перейдет в состояние (S_1, t_1), т.е. информация $\mathbf{i}_0$ уже недостоверна.

Применение ИС на базе ЭВМ сокращает это противоречие, приближая человека к созданию идеальных инструментов управления.

3 Основные понятия БД

Предметная область (ПО) – часть реального мира, подлежащая изучению с целью организации управления и в конечном счете автоматизации.

Это могут быть: транспортное предприятие, ТЭК, диспетчерская служба аэропорта и т.д.

ПО представляется множеством фрагментов: для транспортного предприятия – службой главного механика, главного диспетчера, ОТЗ, бухгалтерией, ОК и т.д.

Впервые термин «БД» появился в 1962 г.

Примеры БД – записная книжка, картотека в библиотеке, данные бухучета и т.п.

БД является центральной частью ИС.

ИС включает в себя БД и без нее не может быть построена.

БД проектируется под конкретную ИС и обеспечивает *данными* процессы ИС.

ИС и БД часто используются как синонимы.

БД – именованная *совокупность данных*, отражающая состояние объектов и их отношений в рассматриваемой ПО (определение ГКНТ).

БД – объективная форма представления и организации *совокупности данных* (например, статей, расчетов), систематизированных таким образом, чтобы эти данные могли быть найдены и обработаны с помощью ЭВМ (ФЗ №3523-1 от 23.09.92 «О правовой охране программ для ЭВМ и баз данных»)

Другие определения:

БД – унифицированная *совокупность данных*, совместно используемая персоналом предприятия, организации и т.п.

БД – *совокупность данных* для машинной оработки, которая отражает информационную модель ПО на определенном уровне абстракции.

БД представляет собой описание состояния ПО на формализованном языке.

Задача БД – хранить все представляющие интерес для предприятия данные в одном месте таким способом, который заведомо исключает их ***избыточность***.

Не всякий блок данных является БД.

БД обладает следующими ***качествами***:

1) интегрированность, направленная на решение общих задач, т.е. интеграция данных, когда все данные накапливаются и хранятся централизованно (заработная плата – кадры);

2) взаимосвязанность;

3) модельность (т.е. структурированность, отражающая некоторую часть реального мира);

4) независимость описания данных от прикладных программ, поскольку данные и их описания хранятся совместно в БД; максимально возможная независимость программ от данных.

Для создания БД используются специальные программные инструментальные системы – системы управления базой данных (СУБД).

СУБД – совокупность языковых и программных средств, предназначенных для создания, ведения и совместного использования БД многими пользователями (определение ГКНТ).

Основное ***назначение СУБД***:

1) обеспечение описания структуры и процессов информационной модели в виде БД, т.е. описание таблиц БД и связей между ними, операций над данными в таблицах;

2) автоматическое отображение информационной модели в физическую БД на носителях ЭВМ;

3) создание и манипулирование данными (выбор, вставка, обновление, удаление и т.п.);

4) контроль целостности и непротиворечивости данных в БД.

БД, создаваемые в запоминающих устройствах ЭВМ, могут содержать сотни и тысячи записей, хранящих совокупности взаимосвязанных сведений о тех или иных объектах.

Преимущество автоматизированных БД.

1) Возможность быстрого ***поиска*** необходимых сведений:

Поиск данных в автоматизированной картотеке может осуществляться не только по алфавиту (записная книжка), по адресу (картотека учета пациентов поликлиники), а по любой совокупности признаков, характеризующих искомые объекты.

Автоматизированная картотека учета сотрудников предприятия в отделе кадров может дать ответы на любой запросы.

Получение ответа на подобные запросы с помощью традиционных картотек является трудоемкой задачей.

2) Возможность ***генерирования отчетов***, т.е. выдачи выбранной информации в требуемом виде.

3) ***Компактность*** – ПЭВМ с винчестером может хранить тысячи документов.

Трехуровневая система организации БД включает в себя:

1) Внешний уровень – модель обращена к пользователю, абстрагируется от особенностей реализации данных на физическом уровне, описывается в терминах исследуемой ПО.

2) Концептуальный уровень – отражает обобщенную модель ПО (объектов реального мира), для которой создавалась БД.

3) Физический уровень (внутренняя) – собственно данные, расположенные на внешних носителях информации; приближена к технической среде, включает метод доступа к данным и т.п.

На втором уровне различают «фактографические» и «документальные» БД.

1) ***Фактографическая*** БД содержит сведения об объектах ПО

2) ***Документальная*** БД накапливает и обрабатывает произвольные текстовые документы.

Модель Данных (МД) (нет единого определения):

совокупность методов и средств, предназначенных для определения логической структуры БД и динамического моделирования в БД.

некоторая абстракция, которая будучи приложима к конкретным данным, позволяет пользователям и разработчикам трактовать их уже как *информацию*, т.е. сведения, содержащие не только данные, но и взаимосвязь между ними.

Различают сетевые, иерархические и реляционные МД.

Сетевые МД используют модель представления данных в виде произвольного графа.

В ***иерархических*** МД данные представляются в виде древовидной (иерархической) структуры.

Реляционные МД отличаются простотой базисных понятий и строгостью математических основ.

4 Реляционная модель данных

Теоретическая основа реляционной модели данных (РМД) – теория отношений, основу которой заложили логики Чарльз Содерс Пирс (1839-1914, США) и Эрнст Шредер (1841-1902, Германия).

РМД предложена Эдвардом Коддом (IBM) в 1970. Им впервые сформулированы основные понятия и ограничения РМД. Он ограничил набор операций семью основными и одной дополнительной операцией.

Причины популярности РМД:

- простота и наглядность для пользователей-непрограммистов,
- серьезное теоретическое обоснование.

Основная структура данных РМД – отношение (relation – отношение, родство).

Домен – некоторое множество элементов (множество допустимых значений, которые может принимать объект по некоторому свойству – множество целых чисел; множество дат и т.п.).

Полное декартово произведение – $D = D_1 \times D_2 \times ... \times D_n$ набор всевозможных сочетаний из n элементов каждое, где каждый элемент берется из своего домена D_i.

Пусть даны n множеств $D_1, D_2, ... D_n$.

R есть ***отношение над этими множествами***, если R есть множество упорядоченных n-кортежей (т.е. состоящих из n элементов, по одному из каждого домена D_i) вида $<d_1, d_2, ... d_n>$, где d_1 элемент из D_1, d_2 – элемент из D_2, d_n – элемент из D_n.

$D_1, D_2, ... D_n$ – домены отношения R.

R принадлежит $D_1 \times D_2 \times ... \times D_n$

Пример: <Авиакомпания, ТипВС, Маршрут>.

D_1 – множество **наименований АК;**

D_2 – множество **наименований типов ВС;**

D_3 – множество **наименований маршрутов**

Отношение (Авиакомпания, ТипВС, Маршрут) состоит из кортежей, кортеж – из 3 элементов, каждый выбирается из своего домена.

Элементами отношения являются ***кортежи***.

Порядок элементов в каждом кортеже один и тот же.

Порядок кортежей в отношении не существен.

В отношении не может быть *одинаковых кортежей* – это подмножество декартова произведения, где все элементы *различны*.

Атрибут – вхождение домена в отношение; использование домена внутри отношения.

Возможно использование двух атрибутов из одного домена.

Степень (ранг) отношения – количество атрибутов в отношении.

Мощность отношения – число кортежей отношения.

Два отношения, отличающиеся *только* порядком строк или столбцов, являются одинаковыми.

Отношение – двумерная таблица при соблюдении определенных ограничивающих условий.

В этой таблице:

каждая *строка* есть *кортеж*,

каждый *столбец* соответствует одному и тому же *компоненту* декартова произведения (в нем могут появляться только элементы из соответствующего домена).

Условия и ограничения, которые позволяют таблицы считать отношениями (таблица, представляющая отношение, обладает следующими свойствами):

1) *нет одинаковых* строк;

2) значения атрибутов *атомарны* – отношения не могут иметь в качестве компонент другие отношения;

3) все строки имеют одну и ту же *структуру* – количество и названия атрибутов у всех записей одинаково (каждая строка представляет собой кортеж из n значений, принадлежащих n столбцам);

4) *имена столбцов* различны, а значения – однотипны;

5) *порядок* следования *строк* в таблице *произвольный*;

6) соблюдается ссылочная целостность для внешних ключей (см. ниже).

Набор отношений может быть использован для хранения данных об объектах реального мира и моделирования связей между ними.

Сущность – объект любой формы, о котором надо хранить информацию в БД.

Пример:

множество студентов можно назвать сущностью **Студент**, множество типов ВС – **Самолет**, множество аэропортов – **Аэропорт.**

Атрибуты – свойства, характеризующие сущность.

Это – столбцы, имеют имена.

Сущность **Студент: Номер, Фамилия, ДатаРожд, Группа;**

Сущность **Аэропорт: Название, Регион, КоличествоВПП;**

Сущность **ВС: ТипВС, Пассажировместимость, ВзлетнаяМасса, Дальность**.

Для представления набора *свойств* объектов атрибуты интерпретируются *столбцами* отношений.

Множество *допустимых значений* атрибута интерпретируется соответствующим *доменом*.

Каждый кортеж отношения играет роль описания отдельного объекта из набора.

Само отношение играет роль описания всего набора объектов.

Строка описывает экземпляр из ПО (экземпляр сущности), в каждом столбце строки размещаются значения одного свойства (атрибута).

Сущность **СТУДЕНТ**

Номер	*Фамилия*	*ДатаРожд*	*Группа*
98001	**Иванов**	**11.01.84**	**326**
98145	**Петров**	**15.02.83**	**327**
…	…	…	…

Сущность **АЭРОПОРТ**

Название	*Регион*	*КоличествоВПП*
Быково	**Московская обл.**	**2**

Курумоч	**Самарская обл.**	**1**
…	…	…

Сущность **САМОЛЕТ**

ТипВС	*Пассажировместимость*	*Взлетная масса*	*Дальность*
Ан-148	**80**	**39**	**3600**
SSJ-100	**95**	**43**	**3000**
…	…	…	…

Отношение – динамическая модель некоторого реального объекта внешнего мира.

Экземпляр отношения – отражает состояние данного набора объектов в какой-то момент времени (листинг).

Схема отношения – список имен атрибутов отношения с указанием домена, к которому они относятся.

Если отношение называется R и имеет атрибуты A_1, A_2, … A_n, относящиеся к доменам D_1, D_2, ... D_n, то схема отношения обозначается следующим образом:

$R(A_1, A_2, \ldots A_n)$, $A_i \subseteq D_i$, i = 1...n

Примеры:

СТУДЕНТ (Номер, Фамилия, ДатаРожд, Группа)

АЭРОПОРТ (Название, Регион, КоличествоВПП)

САМОЛЕТ (ТипВС, Пассажировместимость, ВзлетнаяМасса, Дальность).

Схемы двух отношений ***эквивалентны***, если они имеют одинаковую степень и возможно такое упорядочение имен атрибутов в схемах, что на одинаковых местах будут находиться сравнимые атрибуты, т.е. атрибуты, принимающие значение из одного домена.

Одной из возможных реализаций отношения в памяти ЭВМ является файл записей, формат которых соответствует схеме отношения.

Существует аналогия между схемой отношения и форматом записи, между кортежем и записью, между отношением и файлом.

Каждое отношение (таблица) в ЭВМ представляется в виде отдельного файла; все записи имеют однородную структуру. Между терминами существует соответствие:

Предметный	Математическ.	ТБД	Физический
Сущность	Отношение	Таблица	Файл
Экз. Сущности	Кортеж	Строка	Запись
Свойство	Атрибут	Столбец	Поле

Реляционная БД (РБД) – совокупность изменяемых во времени ***взаимосвязанных*** отношений различных, и возможно, меняющихся мощностей и степеней (изменение во времени – включение, удаление, коррекция), содержащих всю информацию, которая должна храниться в БД.

РБД содержит конечное множество экземпляров отношений.

Схему РБД можно представить в виде:

$R_1 (A_{1.1}, A_{1.2}, \ldots A_{1.n1})$;
$R_2 (A_{2.1}, A_{2.2}, \ldots A_{2.n2})$;
...
$R_m (A_{m.1}, A_{m.2}, \ldots A_{m.nm})$.

Часто встречается случай, когда внутри одного отношения существует атрибут, значения которого однозначно идентифицируют кортежи отношения.

Пример: название аэропорта – все кортежи отношения содержат различные значения Названия; они могут быть использованы для того, чтобы отличать кортежи один от другого.

Этот атрибут (***первичный ключ*** для этого отношения) однозначно идентифицирует отдельный объект.

Не каждое отношение имеет первичный ключ в виде единственного атрибута. Но каждое отношение имеет некоторый набор атрибутов, которые, взятые вместе, однозначно идентифицируют кортеж в отношении, т.е. ключ состоит из *нескольких атрибутов*.

Один атрибут – специальный случай.

Существование такого набора гарантируется тем, что отношение является множеством. Т.к. множества не содержат одинаковых элементов, каждый кортеж является уникальным для данного отношения.

Т.о., каждое отношение имеет (возможно, составной) ***первичный*** ключ.

Первичный ключ – атрибут или набор атрибутов, который полностью и однозначно определяет значения всех остальных атрибутов отношения.

В отношении не должно быть строк с одинаковыми ключами, т.е. не должно быть дублирования объектов.

Часто встречаются отношения, в которых существует более чем одна комбинация, обладающая свойством однозначно идентифицировать кортеж. Следовательно, эти отношения имеют более чем один ***Возможный ключ.***

Пример: паспорт студента, номер его зачетной книжки

Возможный ключ – атрибут или набор атрибутов, который может быть использован в качестве первичного ключа (первичный ключ – один из возможных ключей).

В таком случае можно выбрать произвольно один из возможных ключей в качестве первичного.

В исключительных применениях, когда содержание записи в целом может быть не уникальным, дополнительно используют в ключе порядковый номер записи.

Не ключевой атрибут – атрибут, не входящий в состав ни одного возможного ключа.

Не ключевые атрибуты *функционально зависят* от этого ключа.

Связи между отношениями поддерживаются неявным образом:

- одно отношение выступает как основное (родительское), другое в роли подчиненного (дочернего);

- оба отношения должны содержать наборы атрибутов, по которым они связаны.

В основном (родительском) отношении это – ***первичный ключ***.

В подчиненном (дочернем) отношении он – ***внешний ключ,*** поскольку является первичным ключом основного отношения.

Атрибут отношения R1 является ***внешним ключом***, если он не первичный ключ отношения R1, но его значения являются значениями первичного ключа некоторого отношения R2.

Определяет множество кортежей подчиненного отношения, которые связаны с единственным кортежем основного отношения.

Пример:

ФАКУЛЬТЕТЫ(Номерфак, Название, Декан) и **КАФЕДРЫ(Каф, Ауд, Телефон, Номерфак)**

В основном (родительском) отношении **ФАКУЛЬТЕТЫ** ***первичный ключ*** – атрибут **Номерфак**. В подчиненном (дочернем) отношении **КАФЕДРЫ** для моделирования связи присутствует атрибут **Номерфак**, соответствующий первичному ключу **Номерфак** отношения **ФАКУЛЬТЕТЫ**.

Пример:

КАФЕДРЫ(Каф, Ауд, Телефон, Номерфак) и **ПРЕПОДЫ(ФИО,Долж,Стаж,Каф)**

В основном (родительском) отношении **КАФЕДРЫ** ***первичный ключ*** – атрибут **Каф**.

В подчиненном (дочернем) отношении **ПРЕПОДЫ** для моделирования связи присутствует атрибут **Каф**, соответствующий первичному ключу **Каф** отношения **КАФЕДРЫ**.

Для обеспечения целостности данных (***ссылочной целостности***) каждому значению внешнего ключа в подчиненном (дочернем) отношении обязательно должен соответствовать кортеж основного (родительского) отношения.

Иначе может быть ссылка на объект, о котором ничего не известно.

В отношении **ФАКУЛЬТЕТЫ** обязательно должны присутствовать строки, соответствующие значениям атрибута **Номерфак** в отношении **КАФЕДРЫ** – если для кафедры указан номер факультета, то этот факультет должен существовать!!!

В отношении **КАФЕДРЫ** обязательно должны присутствовать строки, соответствующие значениям атрибута **Каф** в отношении **ПРЕПОДЫ**. – если для преподавателя указан кафедра, то эта кафедра должна существовать!!!

При представлении данных отношение используется двояко:

- для представления набора объектов (т.е. группы подобных объектов);
- для представления связей между наборами объектов.

Система БД должна иметь возможность представлять два типа объектов – собственно «объекты» и «связи».

Между ними не существует принципиального различия; связь представляет собой просто специальный вид объекта.

В РБД связи между объектами представляются, так же как и сами объекты, т.е. кортежами в отношении.

Отношения в зависимости от содержания подразделяют на два класса.

Объектное отношение хранит данные об объектах (экземплярах сущности).

Связное отношение хранит ключи двух или более отношений (внешние ключи), т.е. по внешним ключам устанавливаются связи между объектами отношений. Оно может иметь кроме связываемых ключей и другие атрибуты, которые функционально зависят от этой связи.

Пример:

БД **Студент-Предмет** содержит три типа информации:

1. О студентах: отношение **СТУДЕНТ(НомерЗачетки, ФИО, ДатаРожд, Группа)**;
2. Об изучаемых предметах: отношение **ПРЕДМЕТ (Название, Семестр, Кафедра)**;
3. О студентах и предметах: отношение **ИЗУЧАЕТ (НомерЗачетки, Название, Оценка)** .

В отношении **ИЗУЧАЕТ** атрибуты **НомерЗачетки** и **Название** являются внешними ключами, связывающими объекты (студентов и предметы), атрибут **Оценка** функционально зависит от них (связи).

В отношениях **СТУДЕНТ** и **ПРЕДМЕТ** обязательно должны присутствовать строки, соответствующие значениям атрибутов **Номер** и **Название** в отношении **ИЗУЧАЕТ**.

Пример:

БД **Аэропорт-ВС** содержит три типа информации:

1. Об Аэропортах: отношение **АЭРОПОРТ** (**Название, Регион, КоличествоВПП**);
1. О Воздушных судах: отношение **САМОЛЕТ(ТипВС, Пассажировместимоть, ВзлетныйВес, Дальность)**;
3. О Воздушных судах и Аэропортах: отношение **ПРИНИМАЕТ(ТипВС, Название, Билеты)**.

В отношении **ПРИНИМАЕТ** атрибуты **ТипВС** и **Название** являются внешними ключами, связывающими объекты (воздушные суда и аэропорты), атрибут **Билеты** функционально зависит от них (связи).

В отношениях **САМОЛЕТ** и **АЭРОПОРТ** обязательно должны присутствовать строки, соответствующие значениям атрибутов **ТипВС** и **Название** в отношении **ЛЕТАЕТ**.

5 Операции над отношениями

Показано, что множество отношений замкнуто относительно некоторых специальных операций – образуют вместе с этими операциями абстрактную алгебру.

Алгебра – множество объектов с заданной на нем совокупностью операций, замкнутых относительно этого множества, называемого основным множеством.

5.1 Теоретико-множественные операции реляционной алгебры

1) ***Объединение*** двух отношений – отношение, содержащее множество кортежей, принадлежащих либо первому, либо второму исходным отношениям, либо обоим отношениям одновременно.

$R_3 = R_1$ **U** R_2

Пример

R_1 (Ту-154, Ту-134, Як-40) – ВС первой авиакомпании;

R_2 (Ту-154, Ту-134, Як-42, Л-410) – ВС второй авиакомпании

R_3 (Ту-154, Ту-134, Як-40, Як-42, Л-410) – ВС двух авиакомпаний.

2) ***Пересечение*** двух отношений – отношение, содержащее множество кортежей, принадлежащих одновременно и первому и второму отношениям.

$R_3 = R_1$ **I** R_2

R_3 - (Ту-154, Ту-134).

3) ***Разность*** двух отношений – отношение, содержащее множество кортежей, принадлежащих первому и не принадлежащих второму.

$R_3 = R_1 \setminus R_2$ **R_3 (Як-40).**

$R_3 = R_2 \setminus R_1$ **R_3 (Як-42, Л-410).**

1 и 2 операции – коммутативные операции (результат не зависит от порядка аргументов в операции); 3 – несимметричная операция (результат различен для разного порядка). Все они применимы только к отношениям с эквивалентными схемами.

Конкатенация (сцепление) кортежей – кортеж, полученный добавлением значений второго в конец первого:

$c = \langle c_1, c_2, \ldots, c_n \rangle$ и $q = \langle q_1, q_2, \ldots, q_m \rangle$, $(c, q) = \langle c_1, c_2, \ldots, c_n, q_1, q_2, \ldots, q_m \rangle$

4) ***Расширенное декартово произведение*** отношения $R_1(A_1, A_2, \ldots, A_n)$ степени n и отношения $R_2(B_1, B_2, \ldots, B_m)$ степени m есть отношение $R_3(A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m)$ степени $n+m$, содержащее кортежи, полученные сцеплением каждого кортежа отношения R_1 с каждым кортежем отношения R_2.

$R_3 = R_1 \varnothing R_2$

R_1 (ТипВС,ПассВм)

Ту-154	164
Ту-134	76
Як-42	120

R_2(АК)

Пулково
Самара

R_3(ТипВС,ПассВм,АК)

Ту-154	164	Пулково
Ту-154	164	Самара
Ту-134	76	Пулково
Ту-134	76	Самара
Як-42	120	Пулково

Як-42 120 Самара

5.2 Специальные операции реляционной алгебры

1) ***Горизонтальный выбор (операция фильтрации)*** заданный на отношении R.

α - булевское выражение, составленное из термов сравнения с помощью связей И, ИЛИ, НЕ и, возможно, скобок. Может быть сколь угодно сложным.

Терм сравнения: A ***ос*** a,

где ***ос*** – одна из операций сравнения = <> > >= < <=

A – имя атрибута из домена,

a – константа из этого же домена

АК="Пулково"

ПассВм > 100

АК="Самара" И ПассВм <= 100

Терм сравнения: A ***ос*** B

A и B – имена атрибутов из одного и того же домена.

Откуда <> Куда

Результат – отношение R[α(r)], включающее кортежи из исходного отношения, для которых истинно условие выбора или фильтрации.

Является одной из основных при работе с РБД.

2) ***Проектирование***

Проекция отношения R на набор (вектор) атрибутов B есть отношение со схемой, соответствующей набору атрибутов B, содержащее кортежи, получаемые из кортежей исходного отношения R путем удаления из них значений, не принадлежащих атрибутам из набора B.

R[B] = {r[B]}

Все дублирующие кортежи удаляются из результирующего отношения (по определению отношения).

R_1(ТипВС, ПассВм, АК)

Ту-154 164 Пулково
Ту-154 164 Самара
Ту-134 76 Пулково
Ту-134 76 Самара
Як-42 120 Пулково
Як-42 120 Самара

R_2 = R_1[Тип, ПассВм]

Ту-154 164
Ту-134 76
Як-42 120

3) ***Условное соединение***

Операция выполняется над двумя отношениями R_1 и R_2.

В каждом отношении R_1 и R_2 выделяются атрибуты X и Y (*оба атрибута принадлежат одному и тому же домену*), по которым проводится соединение. Результирующее отношение R_3 включает в себя кортежи, которые являются конкатенаций кортежа, принадлежащего R_1, и кортежа, принадлежащего R_2, для которых выполняется условие ***x*** ос ***y*** (ос = <> >= <= > <); (x – значение X из R_1, y – значение Y из R_2)

R_1(ТипВС, АК)

Ту-154 Пулково
Ту-134 Кр.Авиалинии

R_2(ТипВС, ПассВм)

Ту-154 164
Ту-134 76

Як-42	**Самара**	**Л-410**	**12**
		Ан-24	**44**
		Як-42	**120**

Условие соединения R_1**.ТипВС =** R_2**.ТипВС**

R_3**(ТипВС, АК, ПассВм)**

Ту-154	**Пулково**	**164**
Ту-134	**Кр.Авиалинии**	**76**
Як-42	**Самара**	**120**

1) и 2) – унарные операции, 3) – бинарная

4) ***Деление***

Операция между бинарным отношением (делимое) и унарным (делитель), в результате которой получается унарное (частное)

Делимое A имеет атрибуты X,Y: A (X,Y)

Делитель B имеет атрибут Z: B(Z)

Y и Z принадлежат одному домену.

Деление дает частное C, определенное на том же домене, что и X: C(X)

Значение *x* появится в частном тогда и только тогда, когда пара <*x, y*> появится в A для всех значений *y*, содержащихся в B.

Частное состоит из тех *x*-компонентов делимого, где соответствующие *y*-компоненты делимого включают ***каждый*** компонент делителя.

A (X,Y)		**B(Z)**	**B(Z)**	**B(Z)**
s1	**p1**	**p1**	**p2**	**p1**
s1	**p2**		**p4**	**p2**
s1	**p3**	**C(X)**		**p3**
s1	**p4**	**s1**	**C(X)**	**p4**
s1	**p5**	**s2**	**s1**	**p5**
s1	**p6**		**s4**	**p6**
s2	**p1**			
s2	**p2**			**C(X)**
s3	**p2**			**s1**
s4	**p2**			
s4	**p4**			
s4	**p5**			

6 Проектирование БД с использованием нормализации

Проектирование БД начинается с предварительной структуризации ПО:

1) объекты реального мира подвергаются классификации, фиксируется совокупность подлежащих отображению в БД типов объектов;

2) для каждого типа объектов фиксируется совокупность свойств, посредством которых будут описываться в БД конкретные объекты этого типа;

3) фиксируются виды взаимосвязей между объектами.

Затем решаются вопросы о том, какая информация об этих объектах должна быть представлена в БД, и как ее представить с помощью данных.

6.1 Требования к БД

В БД должно обеспечиваться:

1) поддержание целостности (непротиворечивость и корректность) данных в ходе обновления;

2) простое обновление данных;

3) приемлемые затраты памяти и быстродействие;

4) защищенность от несанкционированного доступа к данным;

5) удобные средства восстановления при сбоях.

6.2 Цели проектирования БД

Наиболее важные цели проектирования.

1. Обеспечение возможности *хранения всех необходимых* данных в БД – определение всех атрибутов для БД.

2. Исключение *избыточности* данных.

Сущность этой цели не всегда очевидна для начинающих проектировщиков. Следует различать Неизбыточное дублирование данных (в примере – одинаковые номера телефонов) и Избыточное Дублирование данных (в примере – одинаковые номера комнат с одинаковыми телефонами).

Пример:

Сотр	Тел	Комн
1	3101	222
2	3101	222
3	3202	444
4	3202	444
5	3303	333

Сотр	Тел	Комн
1	3101	222
2	-	222
3	3202	444
4	-	444
5	3303	333

Можно оставить Телефоны в Комнате только один раз и для других искать по имеющимся.

1) Надо это уметь

2) В случае удаления записи Сотрудника с Телефоном информация исчезнет.

3. *Нормализация* отношений (для упрощения решения проблем, связанных с обновлением и удалением) – *разбиение* отношения на два или более

Проведем разбиение отношений – стандартную процедуру проектирования.

Сотр	Комн
1	222
2	222
3	444

Тел	Комн
3101	222
3202	444
3303	333

4 444
5 333

4. Сведение *числа* отношений БД *к минимуму* – разбиение желательно для исключения проблем, но неудобно для пользователя.

Цели 3 и 4 – противоречивы, нужен компромисс.

В результате проектирования необходимо построить ***концептуальную модель*** (определить количество отношений и конкретные атрибуты в них):

1) список всех необходимых атрибутов:

Атрибут	Тип	Длина	Точность
…			
…			

2) Схемы отношений с указанием первичных ключей

6.3 Использование универсального отношения

Пример: спроектировать БД для хранения информации об отправках партий грузов клиентам в различных городах.

Клиент	*Город*	*Тел.*	*Расст.*	*Тариф*	*Груз*	*Упаковка*	*Объем*
Восток	Уфа	224466	400	10	Продукты	Коробки	15
					Мебель	Контейнер	20
					Запчасти	Ящики	25
Заря	Казань	446677	500	12	Продукты	Коробки	10
					Запчасти	Ящики	12
					Мебель	Контейнер	16
Маяк	Уфа	553311	400	10	Запчасти	Ящики	17
					Продукты	Коробки	10
					Мебель	Контейнер	11
Салют	Казань	334455	500	12	Продукты	Коробки	22

Часть полей атомарны, часть – нет. Необходима реконструкция: требуется добавлять большой объем избыточных данных.

Отношение R1

Клиент	*Город*	*Тел.*	*Расст.*	*Тариф*	*Груз*	*Упаковка*	*Объем*
Восток	Уфа	224466	400	10	Продукты	Коробки	15
Восток	Уфа	224466	400	10	Мебель	Контейнер	20
Восток	Уфа	224466	400	10	Запчасти	Ящики	25
Заря	Казань	446677	500	12	Продукты	Коробки	10
Заря	Казань	446677	500	12	Запчасти	Ящики	12
Заря	Казань	446677	500	12	Мебель	Контейнер	16
Маяк	Уфа	553311	400	10	Запчасти	Ящики	17
Маяк	Уфа	553311	400	10	Продукты	Коробки	10
Маяк	Уфа	553311	400	10	Мебель	Контейнер	11
Салют	Казань	334455	500	12	Продукты	Коробки	22

Это – универсальное отношение (корректное отношения), включающие все атрибуты. Для малых БД (до 15...20 атрибутов) является отправной точкой при проектировании.

Начинающий разработчик будет использовать универсальное отношение (УО) в качестве завершенной БД, поскольку оно включает всю необходимую

информацию. Однако при этом возникают три проблемы-аномалии, т.е. отклонения от нормы.

1) Проблема вставки.

При появлении нового клиента (но договор не заключен), надо включать кортеж с нулевыми и пустыми значениями.

Тексты будут пустыми, а число – 0. Выборка будет неверной.

2) Проблема обновления.

Имеется большое количество избыточных данных. Она свидетельствует о возможности модификации только ***части*** требуемых данных с помощью операции обновления.

Присутствует явная и неявная избыточность данных.

Явная:

клиент и его реквизиты появляются несколько раз.

Изменять значения реквизитов надо в нескольких местах.

Неявная:

один и тот же тариф для одних и тех же городов; при изменении тарифа только в одном месте возникает противоречивость – где правильно?

При печати – разные тарифы для одного города.

3) Проблема удаления.

Есть ***Клиент***, у которого только один договор. При удалении договора исчезает и ***Клиент***.

6.4 Функциональные зависимости. Нормализация отношений

Корректная *схема БД* – схема БД, в которой отсутствуют нежелательные зависимости между атрибутами отношений.

Логическое проектирование БД – процесс разработки корректной схемы РБД.

Существуют два пути проектирования схемы БД:

1) декомпозиция (разбиение) – исходное множество отношений, входящих в схему БД заменяется другим множеством отношений (число их возрастает), являющихся *проекциями* исходных отношений;

2) синтез – компоновка схемы БД из заданных исходных зависимостей между объектами ПО.

Первый путь.

Проектирование РБД связано с теорией ***нормализации***,

Процесс проектирования с использованием декомпозиции (замены одного отношения несколькими) – процесс последовательной нормализации схем отношений.

Для проектирования необходимо получить ответы на вопросы:

1) Из какого источника взять отношение перед началом?

2) Как распознать отношение, нуждающееся в декомпозиции?

3) Каким образом декомпозиция осуществляется?

4) Что является признаком завершения декомпозиции?

Теория нормализации основана на том, что определенный набор отношений обладает лучшими свойствами при вставке, обновлении и удалении данных, чем все остальные наборы отношений, с помощью которых могут быть представлены те же данные.

Введены несколько уровней нормализации схем отношений и соответственно нормальные формы (НФ) отношений: первая НФ (1НФ), вторая НФ (2НФ), третья НФ (3НФ), четвертая НФ (4НФ), пятая НФ (5НФ).

Эти формы подчиняются правилу вложенности по возрастанию номеров: если отношение находится в 5НФ, то оно будет соответствовать 3НФ, 2НФ, 1НФ.

Обратно: если отношение находится в 1НФ, но не в 2НФ, то оно не будет соответствовать ни 3НФ, ни 4НФ, ни в 5НФ.

Каждой НФ соответствует некоторый набор ограничений. Отношение находится в некоторой НФ, если удовлетворяет свойственному ей набору ограничений.

Основные свойства НФ:

1) каждая следующая НФ улучшает свойства предыдущей;

2) при переходе к следующей НФ свойства предыдущих НФ сохраняются.

Каждая следующая итерация соответствует НФ более высокого уровня и обладает лучшими свойствами по сравнению с предыдущими:

Нормализация основана на анализе функциональных зависимостей (ФЗ) между атрибутами отношений.

Понятие ФЗ – фундаментальное в теории нормализации РБД.

ФЗ определяют устойчивые отношения между объектами и их свойствами в рассматриваемой ПО.

Атрибут В ***функционально зависит*** от атрибута А (**A®B)**, если в любой момент времени каждому значению атрибута А соответствует ровно одно значение В. (А и В могут быть составными).

Во всех кортежах с одинаковым значением атрибута А атрибут В также имеет одно и то же значение.

От ключа зависят неключевые атрибуты – любое значение ключа может встретиться лишь в одной записи, для него нет различных значений атрибутов.

ФЗ определяют не текущее состояние БД, а все возможные ее состояния, т.е. отражают связи между атрибутами, которые присущи реальному объекту, моделируемому с помощью БД.

Определить ФЗ по текущему состоянию БД можно только тогда, когда экземпляр БД содержит абсолютно полную информацию.

В реальности это невыполнимо, поэтому набор ФЗ уточняет постановщик задачи – специалист в ПО.

Схемы БД ***эквивалентны,*** если содержание исходной БД может быть получено путем естественного соединения отношений, входящих в результирующую схему, и при этом не появляется новых кортежей в исходной БД.

Декомпозиция должна сохранять *эквивалентность* схем БД при замене одной схемы на другую.

При ***составном*** ключе атрибут может зависеть и от его ***части***.

Полная ФЗ - зависимость неключевого атрибута от всего ключа (составного).

Частичная (неполная) ФЗ - зависимость неключевого атрибута от части составного ключа.

Взаимно-независимые атрибуты – атрибуты, которые не зависят функционально один от другого.

Атрибут С зависит от атрибута А ***транзитивно***, если для атрибутов А,Б,С выполняются условия А→Б, Б→С, но обратная зависимость отсутствует.

Детерминант – атрибут или набор атрибутов, от которого зависит другой атрибут (левая часть ФЗ).

Нормализация отношений – процесс пошаговой декомпозиции (разложения) исходных отношений на более простые с целью устранения нежелательных зависимостей атрибутов, а вместе с тем – избыточности данных и уменьшения вероятности аномалий при работе с БД.

Начальная точка проектирования – универсальное отношение, поскольку в нем присутствуют все нужные атрибуты; каждый кортеж состоит из атомарных атрибутов.

Отношение находится в 1НФ *тогда и только тогда, когда на пересечении каждого столбца и каждой строки находятся только элементарные значения атрибутов (атрибуты атомарны).*

Отношение должно быть в 1НФ даже до постановки вопроса о его разбиения.

Если отношение находится в 1НФ, то все неключевые атрибуты функционально зависят от ключа с различной степенью зависимости.

Отношение находится во 2НФ *тогда и только тогда, когда оно находится в 1НФ, и каждый неключевой атрибут функционально полно зависит от первичного ключа.*

Преобразование, исключающее частичную зависимость атрибутов, – приведение к 2НФ.

Если таких зависимостей не было, то отношение изначально находится в 2НФ.

R1**(Клиент,Груз,Город,Расстояние,Телефон,Тариф,Упаковка,Объем)**

ФЗ: Клиент+Груз®Объем поставки Груз®Упаковка

Клиент®Город, Телефон, Расстояние, Тариф

Город®Тариф, Расстояние

Атрибуты *Город, Телефон, Тариф* зависят от части *Клиент* составного ключа, что ведет к дублированию данных, если клиент обслуживается не один раз.

Атрибут *Упаковка* зависит от части *Груз* составного ключа, что ведет к дублированию данных, если есть несколько одинаковых грузов.

Избыточность:

изменение любого из этих атрибутов требует менять его во всех кортежах для данного ***клиента***;

невозможно хранить клиента без поставок;

невозможно хранить груз без клиента.

Вместо *R1* проекции *R2, R3, R4*:

1) на составной первичный ключ и атрибуты, находящиеся в полной ФЗ от него:

R2 (Клиент, Груз, Объем)

Клиент	*Груз*	*Объем*
Восток	Продукты	15
Восток	Мебель	20
Восток	Запчасти	25
Заря	Продукты	10
Заря	Запчасти	12
Заря	Мебель	16
Маяк	Запчасти	17
Маяк	Продукты	10
Маяк	Мебель	11
Салют	Продукты	22

2) на часть *Клиент* составного ключа и атрибуты, зависящие от нее:

R3 (Клиент, Город, Телефон, Расстояние, Тариф)

Клиент	*Город*	*Тел.*	*Расст.*	*Тариф*
Восток	Уфа	224466	400	10
Заря	Казань	446677	500	12
Маяк	Уфа	553311	400	10
Салют	Казань	334455	500	12

3) на часть *Груз* составного ключа и атрибуты, зависящие от нее:

R4 (Груз, Упаковка)

Груз	*Упаковка*
Продукты	Коробки
Мебель	Контейнер
Запчасти	Ящики

Сокращение избыточности, *R2, R3, R4* – вместо *R1*.

***Отношение находится в 3НФ**, тогда и только тогда, когда оно находится в 2НФ, и не содержит транзитивных зависимостей.*

Преобразование отношения, находящегося в 2НФ, исключающее ВСЕ транзитивные зависимости неключевых атрибутов от ключа – приведение к 3НФ.

Если таких зависимостей нет, то отношение находится в 3НФ.

В отношении *R3* атрибуты *Тариф* и *Расстояние* зависят от ключа не прямо, а через зависимость от *Город* (*транзитивно*). Если стоимость транспортировки любого груза одинакова – избыточность (в одном городе несколько поставщиков, стоимость одна и та же).

Вместо отношения *R3* – две проекции *R5* и *R6*:

1) на ключ и все атрибуты, кроме зависящих от него транзитивно:

R6 (Клиент, Город, Телефон)

Клиент	*Город*	*Тел.*
Восток	Уфа	224466
Заря	Казань	446677
Маяк	Уфа	553311
Салют	Казань	334455

2) на связный атрибут транзитивной ФЗ и атрибуты, зависящие от него:

R5 (Город, Расстояние, Тариф)

Город	*Расст.*	*Тариф*
Уфа	400	10
Казань	500	12

Отношения *R5* и *R6* заменяют *R3*.

Система отношений, приведенная к 3НФ:

R2 (Клиент, Груз, Объем поставки)

R4 (Груз, Упаковка)

R5 (Город, Расстояние, Тариф)
R6 (Клиент, Город, Телефон)

Связи между отношениями задаются следующим образом.

R2-R4: по полю **Груз** (в таблице **R2** это внешний ключ, в таблице **R4** это первичный ключ);

R2-R6: по полю **Клиент** (в таблице **R2** это внешний ключ, в таблице **R6** это первичный ключ);

R6-R5: по полю **Город** (в таблице **R6** это внешний ключ, в таблице **R5** это первичный ключ).

Проведенные преобразования позволяют иметь в БД данные о потенциальных клиентах и городах (которые пока не обслуживаются), потенциальных грузах и упаковках (которые пока не заказаны).

Если это делать в исходном отношении, то при прекращении поставок информация о клиенте исчезает.

В тех случаях, когда отношение имеет только один ключ, ЗНФ освобождает от избыточности и аномалий выполнения операций включения, удаления и модификации.

В тех случаях, когда в отношении имеется два (и более) ***возможных*** ключа, ЗНФ может иметь аномалии операций. В этом случае рассматривают усиленную ЗНФ или нормальную форму Бойса-Кодда (НФБК).

Большинство потенциальных аномалий в БД будет устранено в случае должной декомпозиции в НФБК, при этом используется понятие *возможного* ключа.

Отношение находится в НФБК, если и только если оно находится в ЗНФ и каждый детерминант отношения является возможным ключом.

В рассмотренном примере:

Возможные ключи: **Клиент+Груз**

Детерминанты: Клиент+Груз, Город, Клиент, Телефон

Пример: отношение моделирует сдачу студентом текущих экзаменов.

Студент может сдавать экзамен несколько раз.

Кроме номера зачетной книжки, для электронного учета текущей успеваемости у каждого студента имеется уникальный **Код.**

C1(Номер, Код, Дисциплина, Дата, Оценка)

Возможные ключи:

Номер+Дисциплина+Дата

Код+Дисциплина+Дата

ФЗ:

Номер+Дисциплина+Дата → Оценка

Код+Дисциплина+Дата → Оценка

Номер→Код

Код→Номер

Отношение находится в ЗНФ:

нет неполных ФЗ непервичных атрибутов от атрибутов возможного ключа;

нет транзитивных ФЗ

В последних двух зависимостях зависимым не является непервичный атрибут (не входящий ни в один возможный ключ)

Есть два детерминанта, которые не являются возможными ключами.

Первая возможная декомпозиция:

С2(Код, Дисциплина, Дата, Оценка)

С3(Номер, Код)

Вторая возможная декомпозиция:

С2(Номер, Дисциплина, Дата, Оценка)

С3(Номер, Код)

Схемы равнозначны с точки зрения нормализации.

Выбирать необходимо из конкретных условий:

если зачетная книжка теряется и восстанавливается с тем же номером, то разницы нет; если с новым, то предпочтительнее первая схема.

В обоих случаях получаемые отношения **С2** и **С3** находятся в НФБК и им не свойственны отмеченные аномалии.

Выводы

Алгоритм метода проектирования реляционных БД с помощью декомпозиции включает в себя следующие этапы.

1) Разработка универсального отношения для БД.

2) Определение всех ФЗ между атрибутами отношения.

3) Проверка на нахождение в ЗНФ: если «Да» – проектирование завершается; если «Нет» – разложить на два.

Необходимо удалить межатрибутные зависимости:

- частичные зависимости неключевых от ключа (2НФ)
- транзитивные зависимости неключевых от ключа (ЗНФ)
- зависимости атрибутов составных ключей от неключевых (НФБК)

4) Шаги 2 и 3 повторяются для каждого полученного отношения.

7 Метод «Сущность-связь»

Метод нормализации удобен при малом количестве атрибутов, для более 20 атрибутов он оказывается громоздким.

РМД в силу простоты и лаконичности не позволяет отобразить семантику (смысл) ПО.

Существует другой подход, когда ФЗ привлекается не на начальном, а конечном этапе проектирования. В реальном проектировании структуры базы данных применяются так называемое ***семантическое моделирование***. Оно представляет собой моделирование структуры данных, опирающееся на смысл этих данных. Применяется после словесного описания ПО.

В качестве инструмента семантического моделирования используются ***ER-метод (Entity-Relationship)*** (схемы «Сущность-Связь»).

На использовании разновидностей ER-модели основано большинство современных подходов к проектированию БД.

Модель была предложена Питером Пин-Шэн Ченом (Chen) в 1976 г.

Моделирование ПО базируется на использовании графических диаграмм, включающих небольшое число разнородных компонентов.

Все варианты диаграмм «Сущность-Связь» исходят из одной идеи - рисунок всегда нагляднее текстового описания. Все такие диаграммы используют графическое изображение сущностей предметной области, их свойств (атрибутов), и взаимосвязей между сущностями.

Здесь описывается работа с ER-диаграммами, близкими к нотации Баркера.

7.1 Основные понятия ER-модели

Сущность - класс однотипных объектов, информация о которых должна быть учтена в модели.

Каждая сущность имеет наименование, *уникальное в пределах моделируемой системы, как правило,* существительное

Примеры сущностей: «Аэропорт», «ТипВС», «Авиакомпания».

Соответствует некоторому ***классу*** однотипных объектов, следовательно, в системе должно существовать множество ***экземпляров*** данной сущности.

Экземпляр сущности – конкретный представитель данной сущности.

Пример: представителем сущности «ТипВС» может быть «Ту-154».

Экземпляры сущностей должны быть *различимы*, т.е. сущности должны иметь некоторые свойства, уникальные для каждого экземпляра этой сущности.

Имя сущности – это имя типа, а не некоторого конкретного экземпляра этого типа.

Атрибут сущности – именованная характеристика, являющаяся некоторым свойством сущности.

Наименование атрибута выражается существительным в единственном числе (возможно, с характеризующими прилагательными).

Пример атрибутов сущности «ТипВС»: «Дальность полета», «Пассажировместимость», «Запас топлива».

Каждый экземпляр сущности должен быть отличим от любого другого экземпляра той же сущности (это требование аналогично требованию отсутствия кортежей-дубликатов в таблицах РБД).

Ключ сущности – *неизбыточный* набор атрибутов, значения которых в совокупности являются *уникальными* для каждого экземпляра сущности; атрибут или набор атрибутов, однозначно идентифицирующий конкретный экземпляр сущности.

Неизбыточность заключается в том, что удаление любого атрибута из ключа нарушается его уникальность.

Сущность может иметь несколько различных ключей.

Связь – некоторая ассоциация (***соединение***) между *двумя* сущностями.

Каждая связь имеет два конца и одно или два наименования. Наименование (как правило) выражается глаголом. Каждое из наименований относится к своему концу связи. Иногда наименования не пишутся ввиду их очевидности.

Связи позволяют по одной сущности находить другие сущности, связанные с нею.

Если связь устанавливается между двумя сущностями, то она устанавливает взаимосвязь между экземплярами первой и второй сущностей.

Экземпляр связи однозначно определяется набором ключей сущности, определяемых этой связью.

Связь является *обязательной*, если в данной связи должен участвовать ***каждый*** экземпляр сущности – экземпляр одной сущности *обязан быть связан не менее чем с одним* экземпляром другой сущности.

Связь является *необязательной*, если в данной связи должен участвовать ***не каждый*** экземпляр сущности – экземпляр одной сущности *может быть связан* с одним или несколькими экземплярами другой сущности, *а может быть и не связан* ни с одним экземпляром.

Связь может быть обязательной с одной стороны и необязательной с другой.

Как и сущность, ***связь*** – типовое понятие, все экземпляры обеих пар связываемых сущностей подчиняются правилам связывания.

Пример. **Студент-Препод**

Руководство дипломным проектированием – 1:M

Обязательна со стороны СТУДЕНТ, необязательна со стороны ПРЕПОД: у каждого ДИПЛОМНИКА один и только один РУКОВОДИТЕЛЬ; у каждого ПРЕПОД может быть несколько ДИПЛОМНИКОВ, но не обязательно.

Чтение лекций – N:M

Обязательна со стороны СТУДЕНТ, обязательна со стороны ПРЕПОД: у каждого СТУДЕНТА несколько ЛЕКТОРОВ; у каждого ПРЕПОДА несколько СТУДЕНТОВ.

7.2 Диаграммы ER-модели

Сущность – прямоугольник, содержащий имя сущности.

Имена атрибутов заносятся в прямоугольник, изображающий сущность, под именем сущности, возможно, с примерами.

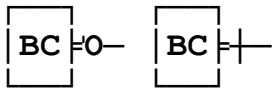
Связь – графически изображаемая ассоциация, устанавливаемая между двумя сущностями; линия, соединяющая две сущности или ведущая от сущности к ней же самой.

На каждом из концов связи указывается:

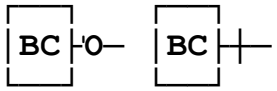
1) имя конца связи;

2) степень связи (сколько экземпляров данной сущности связывается):

трехточечный вход в прямоугольник сущности, если для этой сущности в связи могут использоваться много экземпляров сущности:

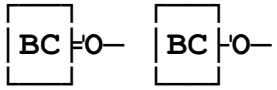


одноточечный вход, если в связи может участвовать только один экземпляр сущности:

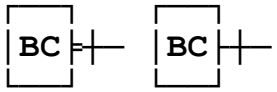


3) обязательность связи (т.е. любой ли экземпляр данной сущности должен участвовать в данной связи):

обязательный конец – кружочек,



необязательный – перпендикуляр.



7.3 Получение отношений из диаграмм ER-типа

Процесс включает в себя следующие этапы.

1) Построение диаграммы, включающей в себя все сущности и связи, важные для рассматриваемой задачи.

2) Построение набора предварительных отношений и указание предполагаемого первичного ключа для каждого отношения.

3) Подготовка списка всех представляющих интерес атрибутов (которые не были ключами) и назначение каждого из этих атрибутов одному из предварительных отношений с тем условием, чтобы эти отношения были в НФБК – должны быть определены межатрибутные ФЗ для проверки НФБК.

Если полученные отношения не находятся в НФБК, необходимо пересмотреть диаграмму.

7.3.1 Связь 1:1

Каждый экземпляр сущности в левой и в правой частях диаграммы связывается максимально с одним экземпляром в противоположной части.

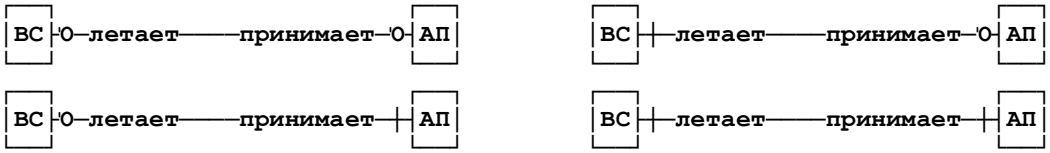
Возможны 4 варианта набора правил функционирования системы.

1) Каждый тип ВС летает ровно (только) в один АП; АП принимает ровно (только) один тип ВС – связь обязательна с ***обеих*** сторон.

2) Каждый тип ВС летает ровно (только) в один АП, АП принимает не более, чем один тип ВС – связь обязательна со стороны ***первой*** сущности.

3) Каждый тип ВС летает не более, чем в один АП, АП принимает ровно (только) один тип ВС – связь обязательна со стороны ***второй*** сущности.

4) Каждый тип ВС летает не более, чем в один АП, АП принимает не более чем один тип ВС – связь ***не обязательна*** ни с одной стороны.



Ни одно из этих правил не позволяет каждому типу ВС летать более чем в один АП и каждому АП принимать более, чем один тип ВС.

Правило 1. Если степень бинарной связи 1:1 и связь с обеих сторон является обязательной, то требуется только одно отношение (ключ – ключ любой из двух сущностей).

Здесь не будет ни пустых, ни повторяющихся строк.

Правило 2. Если степень бинарной связи 1:1 и связь с одной стороны является обязательной, а другой нет, то требуется два отношения: по одному на каждую сущность (ключ сущности – ключ отношения). Ключ отношения, со стороны которого связь является необязательной, добавляется в качестве атрибута в отношение, где связь является обязательной.

При одном отношении появляются пробелы; непонятно, куда вставить ключ из другой сущности. Связь содержит все объекты по одному разу, но при этом только те, кто действует.

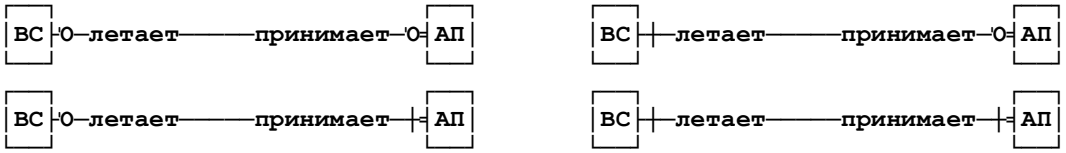
Правило 3. Если степень бинарной связи степени 1:1 и связь с ни одной стороны не является обязательной, то требуется три отношения: по одному на каждую сущность (ключ сущности – ключ отношения) и одно под связь (в связи составной ключ из ключей сущности).

В связи могут быть свои атрибуты.

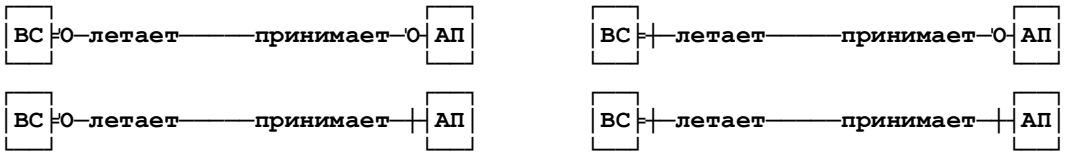
7.3.2 Связь 1:N

Каждый экземпляр сущности в правой части диаграммы связывается максимально с одним экземпляром в противоположной части; каждый экземпляр сущности в левой части диаграммы связывается с несколькими экземплярами в противоположной части.

1:n – каждый тип ВС может летать во много АП, каждый АП принимает не более чем один тип ВС.



n:1 – каждый тип ВС летает не более, чем в один АП, каждый АП принимает несколько типов ВС.



Существую две проблемы:

- пробелы в АП, куда никто не летает;
- повторение данных о ВС.

Если бы связь со стороны АП была бы обязательной, то пробелы бы исчезли; но повтор остается.

В отличие от связи 1:1, правил только два – различаются по обязательности связи N-связной сущности; обязательность 1-связной сущности не влияет.

Правило 4. Если степень бинарной связи 1:N и связь со стороны N-связной сущности является ***обязательной***, то требуется два отношения: по одному на каждую сущность (ключ сущности - ключ отношения). Ключ 1-связной сущности должен быть добавлен как атрибут в отношение N-связной сущности.

Исчезают пробелы и повторы.

Пусть связь с обеих сторон необязательна.

Здесь появляется три проблемы:

- пробелы в АП, куда никто не летает;
- пробелы в ВС, которые не летают;
- повторение данных о ВС, когда много АП.

Если бы связь со стороны АП была бы обязательной, то пробелы в АП исчезли бы; но повтор и пробелы у ВС остаются.

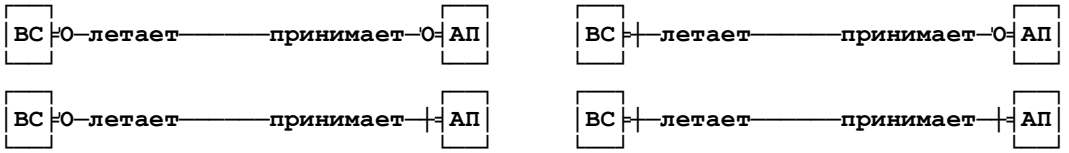
При использовании Правила 4 проблемы исчезают, кроме пробелов в номере ВС у АП.

Правило 5. Если степень бинарной связи 1:N и связь со стороны N-связной сущности является ***необязательной***, то требуется три отношения: по одному на каждую сущность (ключ сущности – ключ отношения) и одного отношения связи (ключ – два ключа сущностей).

7.3.3 Связь M:N

Каждый экземпляр сущности в левой и в правой частях диаграммы связывается с несколькими экземплярами в противоположной части.

n:m – каждый тип ВС летает в несколько АП, каждый АП принимает несколько типов ВС.



В этом случае всегда необходимы 3 отношения (независимо от обязательности связи). Всегда будут повторы или пробелы.

Правило 6. Если степень бинарной связи M:N, то требуется три отношения: по одному на каждую сущность (ключ сущности – ключ отношения) и одного отношения связи (ключ – два ключа сущностей).

7.3.4 Связи более высокого порядка.

Бинарные связи могут описать много ситуаций реального мира, но иногда этого бывает недостаточно.

Правило 7. В случае N-сторонней связи требуется N+1отношения: по одному для каждой сущности (ключ сущности – ключ отношения) и одно для связи (ключ – из N ключей сущностей).

Выводы

В ER-схемах (как и в РМД) используется понятие НФ; их смысл близко соответствует смыслу реляционных НФ.

1НФ: устраняются повторяющиеся атрибуты или группы атрибутов (выявляются неявные сущности, «замаскированные» под атрибуты).

2НФ: устраняются атрибуты, зависящие только от части уникального идентификатора (эта часть определяет ***отдельную сущность***).

3НФ: устраняются атрибуты, зависящие от атрибутов, не входящих в уникальный идентификатор (являются основой ***отдельной сущности***)

8 Ведение БД на ПК

Файл БД состоит из отдельных записей, каждая из которых соответствует одной строке таблицы.

Запись состоит из отдельных полей, количество которых соответствует числу атрибутов (столбцов).

Поля записи содержат данные одного из нескольких возможных типов (символьный, числовой, логический, даты, меморандум).

В файлах БД содержатся определение схемы (структуры) отношения и собственно данные в строках (записях). Они имеют специальный формат.

Программная реализация спроектированной БД начинается с описания структуры хранящихся в рассматриваемом файле записей. Указывается количество полей, каждому полю присваивается идентификатор, устанавливается тип и ширина поля (для числовых и символьных полей)

Следующий этап – ***ввод данных***.

В процессе ввода данных может выясниться, что структура описана не совсем точно и ее необходимо модифицировать (изменить).

В процессе модификации можно переопределить длины полей, изменить их тип, имена, добавить новые поля.

Редактирование – процесс внесения изменений и исправления ошибок в записях БД.

Ведение БД – характеризует операции, выполняемые над БД для поддержания ее в актуальном состоянии.

Включает в себя

- дополнение файла БД новыми записями;
- удаление существующих записей;
- изменение данных в отдельных записей.

Использование БД предполагает выполнение действий по поиску и получению в заданном виде хранимой в ней информации.

Возможно осуществление выборки записей, которые удовлетворяют некоторым условиям (горизонтальный выбор, операция фильтрации)

Записи могут быть отсортированы (в алфавитном порядке, по возрастанию или убыванию числовых значений, по возрастанию или убыванию дат).

При вводе могут заполняться не все поля – их значения заполняются в результате выполнения арифметических или других операций, при этом значения этих полей могут вычисляться с использованием значений других полей.

Возможно получение средних значений, сумм, а также количества записей, удовлетворяющих определенным условиям.

8.1 Понятие транзакции

Транзакция - воздействие на БД, переводящее ее из одного целостного состояния, в другое. Воздействие выражается в изменении данных в таблицах БД.

Если одно из изменений в БД неуспешно, то должен быть произведен откат к состоянию БД, имевшему место до начала транзакции.

Все изменения в рамках Транзакции либо одновременно подтверждаются, либо нет.

Пример: в БД «Полеты» добавлены таблицы «Статистика по АП» и «Статистика по ВС», где копятся сумма проданных билетов по каждому АП и каждому типу ВС.

Транзакция по добавлению записи в основную таблицу БД («Рейсы») состоит в следующем:

1) добавление нового рейса в таблицу «Рейсы»;

2) отыскание в таблице «Статистика по АП» соответствующей записи (аэропорта); если такой еще не было, то ее создание;

3) увеличение суммы проданных билетов по данному аэропорту;

4) отыскание в таблице «Статистика по ВС» соответствующей записи (типа ВС); если такой еще не было, то ее создание,

5) увеличение суммы проданных билетов по данному типу ВС.

Если происходит сбой в одной из этих операций, то отменять необходимо все, иначе «Статистика» будет не верна.

Здесь (в отличие от ссылочной целостности) другой вид ***целостности – смысловая***; данные в БД должны изменяться так, чтобы не нарушались смысловые связи.

В случае сбоя произойдет нарушение достоверности данных, хотя ссылочная целостность не нарушена.

8.2 Типы таблиц БД по виду их изменения

Различаются по способу формирования в них информации и по типу их изменения в процессе работы с программами, обеспечивающими доступ к ним.

Существуют три основных типа ТБД по способу формирования в них значений и их дальнейшему использованию

Справочные ТБД – содержат информацию справочного характера, обладают невысокой степенью изменчивости по сравнению с ТБД других типов. Как правило, находятся в отношении 1:N с другими, являясь при этом родительскими.

Приложение должно быть спроектировано так, чтобы при необходимости внесения значения в поле, выбор производился из текущего содержимого справочных ТБД; это необходимо, чтобы коды были идентичны.

Пример: Типы ВС к Рейсам, Аэропорты к Рейсам

Используются для хранения справочных сведений о характеристиках объекта (аэропорта, типа ВС и т.п.) – цена топлива, расстояние и т.д.; в силу принципа нормализации хранение такой информации в других ТБД приводит к избыточности; всякий раз, когда для каких-то целей необходимо получить цену топлива, она отыскивается в справочнике.

Справочники имеют различную степень изменчивости.

Некоторые не меняются практически никогда (характеристики ВС, телефонные коды городов, адреса покупателей, характеристики ДУ).

В других информация меняется часто (значения цен на товары, услуги). Но если требуется хранить историю изменения параметра, необходимо специальное поле (начало действия новой цены на топливо).

Операционные ТБД – ТБД, в которых происходит устойчивое во времени непрерывное или периодическое обновление или добавление информации.

Как правило, операционные ТБД находятся в подчиненном отношении к справочным. Данные в них являются источником для формирования транзак-

ционных ТБД. На основе данных в операционных ТБД формируются итоговые отчеты.

Информация обновляется ежедневно, изменение данных в них вызывает одновременные или периодические изменения в таблицах «Статистика …».

Пример: «Рейсы».

Транзакционные ТБД – служат для накапливания данных, основанных на значениях в других ТБД. Механизмы обновления зависят от конкретной реализации СУБД.

Пример: «Статистика по АП» и «Статистика по ВС».

8.3 Типы ИС по виду накапливаемой информации

В ходе эксплуатации ИС, работающих с ТБД, имеется информация исходная и результирующая.

Исходная поступает на вход ИС, трансформируется и хранится в БД. Позднее объединяется по различным показателям (с применением сложных алгоритмов) и на выходе появляется итоговая или результирующая.

В таблицу «Рейсы» ежедневно поступает информация. На выходе – суммарные данные.

Возможны два подхода к формированию итогов.

1) Постепенное накапливание итоговых или промежуточных данных по мере поступления в систему исходной информации.

При этом требуется введение транзакционных алгоритмов, реализующих немедленное изменение итоговых или промежуточных данных при подаче на вход ИС исходных данных.

Преимущество – возможность практически немедленной выдачи итоговых данных по любому периоду, т.к. большинство расчетов уже произведено при добавлении исходной информации.

Недостатки:

- трудоемкая реализация алгоритмов;
- необходимость расходования ресурсов на накапливание промежуточных данных в момент добавления информации;
- необходимость проведения повторных расчетов при сбоях.

Этот подход эффективен для случаев, когда итоги требуются непрерывно и в кратчайшие сроки; а также, когда итоги предыдущего периода входят в состав итогов последующего.

2) Формирование итоговых данных в тот момент, когда они необходимы.

Преимущества:

- при добавлении в ИС исходной информации не происходит дополнительных расчетов, что улучшает быстродействие при добавлении исходных данных;
- отсутствие алгоритмов немедленных расчетов обуславливает отсутствие необходимости их реализации в системе, что делает проектирование и физическую реализацию простой, логику работы более понимаемой.

Недостаток: часто требуются значительные временные ресурсы.

Этот подход используется, когда итоги требуются редко, не зависят от предыдущего периода, выдаются за значительный промежуток и есть время для проведения расчетов.