Как отмечалось, инфологическая модель отображает реальный мир в некоторые понятные человеку концепции, полностью независимые от параметров среды хранения данных. Существует множество подходов к построению таких моделей: графовые модели, семантические сети, модель "сущность-связь" и т.д. Наиболее популярной из них оказалась модель "сущность-связь", которая будет рассмотрена в главе 2.

Инфологическая модель должна быть отображена в компьютеро-ориентированную даталогическую модель, "понятную" СУБД. В процессе развития теории и практического использования баз данных, а также средств вычислительной техники создавались СУБД, поддерживающие различные даталогические модели.

Сначала стали использовать иерархические даталогические модели. Простота организации, наличие заранее заданных связей между сущностями, сходство с физическими моделями данных позволяли добиваться приемлемой производительности иерархических СУБД на медленных ЭВМ с весьма ограниченными объемами памяти. Но, если данные не имели древовидной структуры, то возникала масса сложностей при построении иерархической модели и желании добиться нужной производительности.

Сетевые модели также создавались для мало ресурсных ЭВМ. Это достаточно сложные структуры, состоящие из "наборов" – поименованных двухуровневых деревьев. "Наборы" соединяются с помощью "записей-связок", образуя цепочки и т.д. При разработке сетевых моделей было выдумано множество "маленьких хитростей", позволяющих увеличить производительность СУБД, но существенно усложнивших последние. Прикладной программист должен знать массу терминов, изучить несколько внутренних языков СУБД, детально представлять логическую структуру базы данных для осуществления навигации среди различных экземпляров, наборов, записей и т.п. Один из разработчиков операционной системы UNIX сказал "Сетевая база – это самый верный способ потерять данные".

Сложность практического использования иерархических и и сетевых СУБД заставляла искать иные способы представления данных. В конце 60-х годов появились СУБД на основе инвертированных файлов, отличающиеся простотой организации и наличием весьма удобных языков манипулирования данными. Однако такие СУБД обладают рядом ограничений на количество файлов для хранения данных, количество связей между ними, длину записи и количество ее полей.

Сегодня наиболее распространены реляционные модели, которые будут подробно рассмотрены в главе 3.

Физическая организация данных оказывает основное влияние на эксплуатационные характеристики БД. Разработчики СУБД пытаются создать наиболее производительные физические модели данных, предлагая пользователям тот или иной инструментарий для поднастройки модели под конкретную БД. Разнообразие способов корректировки физических моделей современных промышленных СУБД не позволяет рассмотреть их в этом разделе.

## Модели организации баз данных

**1. Иерархический подход к организации баз данных.** Иерархические базы данных имеют форму деревьев с дугами-связями и узлами-элементами данных. Иерархическая структура предполагала неравноправие между данными – одни жестко подчинены другим. Подобные структуры, безусловно, четко удовлетворяют требованиям многих, но далеко не всех реальных задач.

**2.** **Сетевая модель данных.** В сетевых БД наряду с вертикальными реализованы и горизонтальные связи. Однако унаследованы многие недостатки иерархической и главный из них, необходимость четко определять на физическом уровне связи данных и столь же четко следовать этой структуре связей при запросах к базе.

**3. Реляционная модель.** Реляционная модель появилась вследствие стремления сделать базу данных как можно более гибкой. Данная модель предоставила простой и эффективный механизм поддержания связей данных.

**Во-первых**, все данные в модели представляются в виде таблиц и только таблиц. Реляционная модель – единственная из всех обеспечивает единообразие представления данных. И сущности, и связи этих самых сущностей представляются в модели совершенно одинаково – ***таблицами***. Правда, такой подход усложняет понимание смысла хранящейся в базе данных информации, и, как следствие, манипулирование этой информацией.

Избежать трудностей манипулирования позволяет **второй элемент** модели – реляционно-полный язык (отметим, что язык является неотъемлемой частью любой модели данных, без него модель не существует). Полнота языка в приложении к реляционной модели означает, что он должен выполнять любую операцию реляционной алгебры или реляционного исчисления (полнота последних доказана математически Э.Ф. Коддом). Более того, язык должен описывать любой запрос в виде операций с таблицами, а не с их строками. Одним из таких языков является SQL.

**Третий элемент** реляционной модели требует от реляционной модели поддержания некоторых ограничений целостности. Одно из таких ограничений утверждает, что каждая строка в таблице должна иметь некий уникальный идентификатор, называемый ***первичным ключом***. Второе ограничение накладывается на целостность ссылок между таблицами. Оно утверждает, что атрибуты таблицы, ссылающиеся на первичные ключи других таблиц, должны иметь одно из значений этих первичных ключей.

**4. Объектно-ориентированная модель.** Новые области использования вычислительной техники, такие как научные исследования, автоматизированное проектирование и автоматизация учреждений, потребовали от баз данных способности хранить и обрабатывать новые объекты – текст, аудио- и видеоинформацию, а также документы. Основные трудности объектно-ориентированного моделирования данных проистекают из того, что такого развитого математического аппарата, на который могла бы опираться общая объектно-ориентированная модель данных, не существует. В большой степени поэтому до сих пор нет базовой объектно-ориентированной модели. С другой стороны, некоторые авторы утверждают, что общая объектно-ориентированная модель данных в классическом смысле и не может быть определена по причине непригодности классического понятия модели данных к парадигме объектной ориентированности. Несмотря на преимущества объектно-ориентированных систем – реализация сложных типов данных, связь с языками программирования и т.п. – на ближайшее время превосходство реляционных СУБД гарантировано.

### 5.3.3 Модели данных и концептуальное моделирование

Выше уже упоминалось, что схема создается с помощью некоторого языка определения данных. На самом деле она создается на основе языка определения данных конкретной целевой СУБД, являющегося языком относительно низкого уровня; с его помощью трудно описать требования к данным так, чтобы созданная схема была доступна пониманию пользователей самых разных категорий. Чтобы достичь такого понимания, требуется составить описание схемы на некотором, более высоком уровне, которое будем называть моделью данных. При этом под моделью данных мы будем понимать интегрированный набор понятий для описания данных, связей между ними и ограничений, накладываемых на данные в пределах некоторой предметной области.

Модель является представлением объектов и событий предметной области, а также существующих между ними связей. Модель данных можно рассматривать как сочетание трех указанных ниже компонентов.

* Структурная часть, т.е. набор правил, по которым может быть построена база данных.
* Управляющая часть, определяющая типы допустимых операций с данными (сюда относятся операции обновления и извлечения данных, а также операции изменения структуры базы данных).
* Набор ограничений поддержки целостности данных , гарантирующих корректность используемых данных.

Цель построения модели данных заключается в представлении данных в понятном виде. Если такое представление возможно, то модель данных можно будет легко применить при проектировании базы данных. Для отображения архитектуры ANSI-SPARC можно определить следующие три связанные модели данных:

* внешнюю модель данных, отображающую представления каждого существующего в организации типа пользователей;
* концептуальную модель данных, отображающую логическое (или обобщенное) представление о данных, независимое от типа выбранной СУБД;
* внутреннюю модель данных, отображающую концептуальную схему определенным образом, понятным выбранной целевой СУБД.

В литературе предложено и опубликовано достаточно много моделей данных. Они подразделяются на три категории: объектные (object-based) модели данных, модели данных на основе записей (record-based) и физические модели данных. Первые две используются для описания данных на концептуальном и внешнем уровнях, а последняя - на внутреннем уровне.

**Объектные модели данных.** При построении объектных моделей данных используются такие понятия как сущности, атрибуты и связи. Сущность - это отдельный элемент (сотрудник, изделие, понятие или событие) предметной области, который должен быть представлен в базе данных. Атрибут - это свойство, которое описывает некоторый аспект объекта и значение которого следует зафиксировать, а связь является ассоциативным отношением между сущностями. Ниже перечислены некоторые наиболее общие типы объектных моделей данных.

* + Модель типа "сущность-связь", или ER-модель (Entity-Relationship model).
	+ Семантическая модель.
	+ Функциональная модель.
	+ Объектно-ориентированная модель.

В настоящее время ER-модель стала одним из основных методов концептуального проектирования баз данных. Объектно-ориентированная модель расширяет определение сущности с целью включения в него не только атрибутов, которые описывают состояние объекта, но и действий, которые с ним связаны, т.е. его поведение. В таком случае говорят, что объект инкапсулирует состояние и поведение.

Модели данных на основе записей. В модели на основе записей база данных состоит из нескольких записей фиксированного формата, которые могут иметь разные типы. Каждый тип записи определяет фиксированное количество полей, каждое из которых имеет фиксированную длину. Существует три основных типа логических моделей данных на основе записей: реляционная модель данных (relational data model), сетевая модель данных (network data model) и иерархическая модель данных (hierarchical data model).

**Реляционная модель данных.** Реляционная модель данных основана на понятии математических отношений. В реляционной модели данные и связи представлены в виде таблиц, каждая из которых имеет несколько столбцов с уникальными именами.

При этом в реляционной модели данных единственное требование состоит в том, чтобы база данных с точки зрения пользователя выглядела как набор таблиц. Однако такое восприятие относится только к логической структуре базы данных, т.е. к внешнему и концептуальному уровням архитектуры ANSI/SPARC. Оно не относится к физической структуре базы данных, которая может быть реализована с помощью разнообразных структур хранения.

**Сетевая модель данных.** В сетевой модели данные представлены в виде коллекций записей, а связи - в виде наборов. В отличие от реляционной модели, связи здесь явным образом моделируются наборами, которые реализуются с помощью указателей. Сетевую модель можно представить как граф с записями в виде узлов графа и наборами в виде его ребер.

**Иерархическая модель данных.** Иерархическая модель является ограниченным подтипом сетевой модели. В ней данные также представлены как коллекции записей, а связи - как наборы. Однако в иерархической модели узел может иметь только одного родителя. Иерархическая модель может быть представлена как древовидный граф с записями в виде узлов (которые также называются сегментами) и множествами в виде ребер. Для моделирования информации с помощью древовидной структуры используется обобщенное дерево, состоящее из узлов, соединенных связями, называемых дугами или ребрами. Самый верхний узел называется корневым узлом. В структуре дерева могут быть выделены поддеревья, каждое из которых исходит из одного родительского узла (дочернего для узла более высокого уровня). Все узлы дерева, за исключением корневого, должны иметь родительский узел. Узлы представляют интересующие нас объекты, а связи между ними определяются самим расположением узлов и ребер, образующих данную древовидную структуру.

Основанные на записях (логические) модели данных используются для определения общей структуры базы данных и высокоуровневого описания ее реализации. Их основной недостаток заключается в том, что они не дают адекватных средств для явного указания ограничений, накладываемых на данные. В то же время в объектных моделях данных отсутствуют средства указания их логической структуры, но за счет предоставления пользователю возможности указать ограничения для данных, они позволяют в большей мере представить семантическую суть хранимой информации.

Большинство современных систем БД основано на реляционной парадигме, тогда как самые первые системы баз данных строились на основе сетевой или иерархической модели. При использовании последних двух моделей от пользователя требуется знание физической организации базы данных, к которой он должен осуществлять доступ, в то время как при работе с реляционной моделью независимость от данных обеспечивается в значительно большей степени. Следовательно, если в реляционных системах для обработки информации в базе данных принят декларативный подход (т.е. они указывают, какие данные следует извлечь), то в сетевых и иерархических системах - навигационный подход (т.е. они указывают, как их следует извлечь).

**Физические модели данных.** Физические модели данных описывают то, как данные хранятся в компьютере, представляя информацию о структуре записей, их упорядоченности и существующих путях доступа.

**Концептуальное моделирование.** Как показывает изучение трехуровневой архитектуры СУБД, концептуальная схема является "сердцем" базы данных. Она поддерживает все внешние представления, а сама поддерживается средствами внутренней схемы. Однако внутренняя схема является всего лишь физическим воплощением концептуальной схемы. Именно концептуальная схема призвана быть полным и точным представлением требований к данным в рамках некоторой предметной области. В противном случае определенная часть информации о предприятии будет упущена или искажена, в результате чего могут возникнуть трудности при попытках полной реализации одного или нескольких внешних представлений.

Концептуальное моделирование базы данных - это процесс конструирования модели использования информации. Этот процесс не зависит от таких подробностей, как используемая СУБД, прикладные программы, языки программирования или любые другие вопросы физической организации информации. Подобная модель называется концептуальной моделью данных.

Первой и самой важной функцией базы данных, является функция хранения информации. Информация должна хранится упорядоченно для более быстрого и понятного пользователю доступа к ней. Упорядоченность информации в базе данных, помимо удобств доступа, может привести к значительному сокращению аппаратных ресурсов, необходимых для ее обслуживания. Упорядоченность достигается путем нормализации.

Здесь мы вплотную подошли ко второй функции базы данных – ввод информации. Какую информацию будет вводить пользователь? Хорошая база данных построена из главного документа, справочников, из которых пользователь вводит информацию и нескольких полей для ручного ввода, например, текстов назначения платежа в платежных поручениях и суммы. База данных должна заполняться средствами, наиболее полно автоматизирующими этот процесс. При этом плохим тоном являются:

ввод информации об одном объекте разными способами или в разных местах;

ввод одной и той же информации в нескольких местах;

ввод информации разрозненно, без поддержания общей структуры объекта. Одной из основных функций базы данных является автоматизация. Под автоматизацией, как правило, понимают автоматическое создание выходных документов и пересчет данных, например печать накладной, счета фактуры и протокола согласования цен в складской программе для исходящей накладной.

Далее, нужно вспомнить о системах принятия решений. Информационная система должна позволять создавать статистические отчеты в реальном режиме времени о состоянии описываемого в базе данных процесса. Эта функция удобна для руководителей подразделений, которые могут прогнозировать поведение описываемой системы на основе статистических данных, полученных из базы данных.

Собственно, описанные выше функции информационной системы являются “джентльменским набором”, которого достаточно в большинстве случаев. Из дополнительных функций необходимо упомянуть возможность поиска по нескольким взаимосвязанным характеристикам.

В единой информационной системе необходима возможность идентификации пользователя с целью ограничения доступа пользователя к определенным частям базы данных и введения информации о создателе документа и лиц, редактировавших его. Это придаст пользователям ощущение ответственности за выполняемые действия.

Хорошая информационная система должна легко расширяться при необходимости добавления в нее новых возможностей. Расширяемость подразумевает элементы объектной ориентированности, встроенные в базу данных. Настраивая эти объекты, возможно вносить незначительные изменения в структуру базы данных, что продляет срок морального устаревания всей информационной системы. Одним из факторов расширяемости является возможность сочленять разнородные базы данных в единый комплекс. Такая возможность сейчас реализуется через дополнительные модули, которые по своей сути являются серверами приложений, или правильное построение базы данных по классическим реляционным законам. Последний случай затрудняется тем, что некоторые серверы базы данных не могут выполнить один SQL запрос к разным базам данных, тем более находящимся в географической удаленности друг от друга. Еще одна удобная функция в базе данных – это сквозное прохождение по документам.

Описанные выше функции в разных реализациях информационных систем имеют специфические черты, ориентированные на конкретное прикладное применения.

* 1. ***Уровни моделей базы данных***

Разработка любой информационной системы включает в себя несколько взаимно перекрывающихся во времени процессов:

1. определение пользователей системы и формулировка их требований к ней;
2. анализ стоящей задачи;
3. проектирование (базы данных, приложений и т.д.);
4. реализация (в том числе, программирование);
5. документирование;
6. тестирование и возврат к одному из предыдущих процессов.

Следует заметить, что анализ и проектирование базы данных являются самыми ответственными частями работы, поскольку от одной БД обычно зависит создание сразу многих приложений, а требования к БД меняются медленнее, тем требования к приложениям. Чем крупнее информационная система, тем больший упор должен быть сделан на проектирование базы данных по сравнению с остальными процессами разработки.

Результатом анализа и проектирования информационной системы являются модели. Они используются для следующих целей:

1. связывание понятий различных участников разработки информационной системы;
2. формализация и систематизация этих понятий (в т.ч. разбиение по категориям);
3. детальное описание (спецификация) компонентов системы и связей между ними;
4. анализ этих компонентов для лучшего понимания структуры системы и её дальнейшего развития (что возможно благодаря наглядному представлению модели).

Прежде всего, разработчики информационной системы создают обобщенное и не слишком формальное описание базы данных, объединяя частные представления об её содержимом, полученные из опроса пользователей (сотрудников организации, для которой предназначена система). Это описание, выполненное с использованием естественного языка, таблиц, формул, графиков и тому подобных средств, называют инфологической (или информационной, или концептуальной, или семантической) моделью данных (см. Рис. 2.1). Такая ориентированная на человека модель полностью независима от физических параметров среды хранения данных, и этой средой может быть, например, память человека, а не компьютера. Остальные модели ориентированы не на смысл (семантику) данных, а на их компьютерное представление. На базе второй модели – даталогической (или просто логической) – СУБД предоставляет доступ к хранимым данным лишь по их именам, не заботясь о физическом размещении этих данных. Даталогические модели должны быть описаны на языке описания данных этой СУБД (к счастью, разные СУБД имеют близкие языки, см. 1.3.2). Нужные данные отыскиваются СУБД на внешних запоминающих устройствах в соответствии с третьей – физической – моделью данных. Структура данных этой модели (данных конфигурации) уже слишком зависит от СУБД, поэтому в данном курсе она практически не рассматривается.

Три уровня моделей БД (семантический, логический и физический) обеспечивают независимость данных от использующих их приложений. Разработчик (или администратор) информационной системы может при необходимости переписывать данные на другие носители информации, оптимизировать их физическую структуру и даже переносить систему на другую СУБД, изменяя тем самым лишь физическую модель данных. Он также может подключить к системе любое число новых пользователей и новых приложений, дополняя, если надо, логическую модель. Указанные изменения физической и логической моделей не будут замечены существующими пользователями системы, которые работают с БД на семантическом уровне. Приложения же зависят от данных логического уровня, поэтому их код может меняться, однако при правильном проектировании БД изменения незначительны.

Данный курс напрямую касается лишь двух из перечисленных в начале этого раздела процессов разработки – проектирования БД (часть пункта 3) и реализации доступа к ней (часть пункта 4), которые в совокупности представляют собой формирование логической (даталогической) модели данных. Однако формулировку требований и анализ (пункты 1 и 2), результаты которых выражаются в семантической (инфологической) модели, также необходимо кратко рассмотреть, поскольку на них основывается дальнейшая работа по проектированию БД.

## Этапы проектирования баз данных

Создание и внедрение в практику современных информационных систем

автоматизированных баз данных выдвигает новые задачи проектирования, которые

невозможно решать традиционными приемами и методами. Большое внимание

необходимо уделять вопросам проектирования баз данных. От того, насколько

успешно будет спроектирована база данных, зависит эффективность

функционирования системы в целом, ее жизнеспособность и возможность

расширения и дальнейшего развития. Поэтому вопрос проектирования баз данных

выделяют как отдельное, самостоятельное направление работ при разработке

информационных систем.

 ***Проектирование баз данных*** — это итерационный, многоэтапный процесс

принятия обоснованных решений в процессе анализа информационной модели

предметной области, требований к данным со стороны прикладных программистов и

пользователей, синтеза логических и физических структур данных, анализа и

обоснования выбора программных и аппаратных средств. Этапы проектирования баз

данных связаны с многоуровневой организацией данных. Рассматривая вопрос

проектирования баз данных, будем придерживаться такого многоуровневого

представления данных: внешнего, инфологического, логического (даталогического)

и внутреннего.

Такое представление уровней данных не единственное. Существуют и другие

варианты многоуровневого представления данных. Так, в соответствии с

предложениями исследовательской группы по системам управления данными

Американского национального института стандартов ANSI/X3/SPARC, а также

CODASYL (Conference on Data Systems Languages), как правило, выделяется три

уровня представления данных:

 **внешний** уровень (с точки зрения конечного пользователя и прикладного

программиста),

 **концептуальный** уровень (с точки зрения СУБД),

 **внутренний** уровень (с точки зрения системного программиста).

В соответствии с этой концепцией внешний уровень это часть (подмножество)

концептуальной модели, необходимая для реализации какого-либо запроса или

прикладной программы. То есть, если концептуальная модель выступает как

схема, поддерживаемая конкретной СУБД, то внешний уровень — это некоторая

совокупность подсхем, необходимых для реализации конкретной прикладной

программы или запроса пользователя.

Существует также другая точка зрения, в соответствии с которой под внешним

уровнем понимают более общие понятия, связанные с изучением и анализом

информационных потоков предметной области и их структуризацией. Некоторые

авторы вводят вспомогательный уровень (промежуточный между внешним и

даталогическим уровнями), который называется инфологическим. Он может

выступать как самостоятельный или быть составной частью внешнего уровня.

Такая концепция более целесообразна с точки зрения понимания процесса

проектирования БД.

Поэтому будем рассматривать инфологический уровень как самостоятельный

уровень представления данных. Внешний уровень в этом случае выступает как

отдельный этап проектирования, на котором изучается все внемашинное

информационное обеспечение, то есть формы документирования и представления

данных, а также внешняя среда, в которой будет функционировать банк данных с

точки зрения методов фиксации, сбора и передачи информации в базу данных.

При проектировании БД **на внешнем уровне** необходимо изучить

функционирование объекта управления, для которого проектируется БД, всю

первичную и выходную документацию с точки зрения определения того, какие именно

данные необходимо сохранять в базе данных. Внешний уровень это, как правило,

словесное описание входных и выходных сообщений, а также данных, которые

целесообразно сохранять в БД. Описание внешнего уровня не исключает наличия

элементов дублирования, избыточности и несогласованности данных. Поэтому для

устранения этих аномалий и противоречий внешнего описания данных выполняется

инфологическое проектирование. Инфологическая модель является средством

структуризации предметной области и понимания концепции семантики данных.

Инфологическую модель можно рассматривать в основном как средство

документирования и структурирования формы представления информационных

потребностей, которая обеспечивает непротиворечивое общение пользователей и

разработчиков системы.

Все внешние представления интегрируются на инфологическом уровне, где

формируется инфологическая (каноническая) модель данных, которая не является

простой суммой внешних представлений данных.

 **Инфологический уровень** представляет собой информационно-логическую модель

(ИЛМ) предметной области, из которой исключена избыточность данных и отображены

информационные особенности объекта управление без учета особенностей и

специфики конкретной СУБД. То есть инфологическое представление данных

ориентированно преимущественно на человека, который проектирует или использует

базу данных.

 **Логический (концептуальный)** уровень построен с учетом специфики и

особенностей конкретной СУБД. Этот уровень представления данных ориентирован

больше на компьютерную обработку и на программистов, которые занимаются ее

разработкой. На этом уровне формируется концептуальная модель данных, то есть

специальным способом структурированная модель предметной области, которая

отвечает особенностям и ограничениям выбранной СУБД. Модель логического уровня,

поддерживаемую средствами конкретной СУБД, называют еще даталогической.

Инфологическая и даталогическая модели, которые отображают модель одной

предметной области, зависимы между собой. Инфологическая модель может легко

трансформироваться в даталогическую модель.

 **Внутренний уровень** связан с физическим размещением данных в памяти ЭВМ.

На этом уровне формируется физическая модель БД, которая включает структуры

сохранения данных в памяти ЭВМ, в т.ч. описание форматов записей, порядок их

логического или физического приведения в порядок, размещение по типам

устройств, а также характеристики и пути доступа к данным.

От параметров физической модели зависят такие характеристики функционирования

БД: объем памяти и время реакции системы. Физические параметры БД можно

изменять в процессе ее эксплуатации с целью повышения эффективности

функционирование системы. Изменение физических параметров не предопределяет

необходимости изменения инфологической и даталогической моделей.

Схема взаимосвязи уровней представления данных в БД изображена на рис. 4.1. В

соответствии с этими уровнями проектируется БД. Проектирование БД— это

сложный и трудоемкий процесс, который требует привлечения многих

высококвалифицированных специалистов. От того, насколько квалифицированно

спроектирована БД, зависят производительность информационной системы и

полнота обеспечения функциональных потребностей пользователей и прикладных

программ. Неудачно спроектированная БД может усложнить процесс разработки

прикладного программного обеспечения, обусловить необходимость использования

более сложной логики, которая, в свою очередь, увеличит время реакции

системы, а в дальнейшем может привести к необходимости перепроектирования

логической модели БД. Реструктуризация или внесение изменений в логическую

модель БД это очень нежелательный процесс, поскольку он является причиной

необходимости модификации или даже перепрограммирование отдельных задач.

Все работы, которые выполняются на каждом этапе проектирования, должны

интегрироваться со словарем данных. Каждый этап проектирования

рассматривается как определенная последовательность итеративных процедур, в

результате которых формируется определенная модель БД.

 Рис. 4.1. *Схема взаимосвязи уровней представление данных в БД*

 **Внешний уровень — подготовительный этап инфологического проектирования**

Целью проектирования на внешнем уровне является разработка внемашинного

информационного обеспечения, которое включает систему входной (первичной)

документации, характеризующую определенную предметную область, систему

классификации и кодирования технико-экономической информации, а также

перечень соответствующих выходных сообщений, которые нужно формировать с

помощью БнД.

Существуют два подхода к проектированию баз данных на внешнем уровне: «от

предметной области» и «от запроса». Подход «от предметной области» состоит в

том, что формируется внешнее информационное обеспечение всей предметной

области без учета потребностей пользователей и прикладных программ. Иногда

этот подход называют еще объектным или непроцессным.

При подходе «от запроса» основным источником информации о предметной области

есть изучение запросов пользователей и потребностей прикладных программ. Этот

подход также называется процессным или функциональным. При таком подходе БД

проектируется для выполнения текущих задач управления без учета возможности

расширение системы и возникновение новых задач управление.

Преимущество подхода «от предметной области» это его объективность,

системность при отображении ПО и стойкость информационной модели, возможность

реализации большого количества прикладных программ и запросов, в том числе

незапланированных при создании БД. Недостатком этого подхода является

значительный объем работ, которые необходимо выполнить при определении

информации. подлежащей хранению в БД, что, соответственно, усложняет и

увеличивает срок разработки проекта.

Функциональный подход ориентирован на реализацию текущих требований

пользователей и прикладных программ без учета перспектив развития системы.

При его использовании могут возникнуть сложности в агрегации требований

разных пользователей и прикладных программ. Тем не менее, при таком подходе

значительно уменьшается трудоемкость проектирования, и поэтому возможно

создать систему с высокими эксплуатационными характеристиками.

Однако взятый в отдельности любой из этих методов не может дать достаточно

информации для проектирования рациональной структуры БД. Поэтому при

проектировании БД целесообразно совместно использовать эти два подхода. Если

схематично представить процесс проектирования БД на внешнем уровне, то он

состоит из таких работ.

1. Определение функциональных задач предметной области, которые

подлежат автоматизированному решению. Поскольку основной целью создания БД

есть обеспечение информацией функций обработки данных, то, прежде всего,

необходимо изучить все функции предметной области (объекта управления), для

которой разрабатывается база данных, и проанализировать их особенности.

Функции и функциональные особенности объекта управление необходимо изучать в

неразрывной связи с изучением функциональных требований к данным со стороны

будущих пользователей информационной системы. Изучение и анализ

предусматривают выявление информационных потребностей и определения

информационных потоков. Эти работы можно выполнять обследованием предметной

области и анкетированием ее сотрудников. Результатом такого изучения может

быть перечень функциональных задач, которые должны решаться

автоматизированным способом с использованием БД.

2. Изучение и анализ оперативных первичных документов. Изучив функции и

определив перечень функциональных задач, которые подлежат автоматизированному

решению, переходят к изучению оперативных документов, которые используются на

входе каждой задачи или их комплекса. Изучив и проанализировав все

оперативные документы (как внешние, так и внутренние), которые используются

на входе каждой задачи, определяют, какие реквизиты этих документов нужно

сохранять в БД.

3. Изучение нормативно-справочных документов. На третьем шаге изучают и

анализируют всю нормативно-справочную документацию. К такой документации

принадлежат различные классификаторы, сметы, договоры, нормативы,

законодательные акты по налоговой политике, плановая документация и т.п.

Распределение и отдельный анализ оперативной и нормативно-справочной

информации обусловлены технологически. В базы данных различаются технологии

создания и ведения файлов условно-постоянной информации, размещенной в

нормативно-справочной документации, и файлов оперативной информации.

4. Изучение процессов преобразования входных сообщений в выходные.

Прежде всего, изучаются все выходные сообщения, которые выдаются на печать

или на экран и сохраняются в виде выходных массивов на МД. Это необходимо для

того, чтобы определить, которые из атрибутов входных сообщений нужно

сохранять в БД для получения выходных сообщений. Кроме того, на этом этапе

определяются те показатели, которые получают во время решения задачи в

результате выполнения определенных вычислений. По каждому расчетному

показателю следует определить алгоритм его формирования и убедиться в том,

что этот показатель можно получить на основе атрибутов оперативной и

нормативно-справочной информации, которые были определены на втором и третьем

шагах. Если определенных данных не хватает для полного выполнения расчетов,

необходимо возвратиться назад, провести дополнительное исследование и

определить, где и каким способом можно получить атрибуты, которых не хватает.

Кроме того, нужно определиться, какие из расчетных показателей целесообразно

сохранять в БД. Показатели, полученные расчетным путем, как правило, в БД не

сохраняются. Исключением являются случаи, когда расчетный показатель нужно

использовать для решения других задач или для данной задачи, но в следующие

календарные периоды.

При проведении проектных работ на внешнем уровне надо учитывать то, что для

выполнения определенных функций в БД необходимо сохранять дополнительные

данные, которые не отображены в документах (данные календаря, статистические

данные и т.п.). Обобщенная схема процесса изучения документов и данных при

проектировании на внешнем уровне изображена на рис. 4.2.

 Рис.4.2. *Обобщенная схема процесса проектирование на внешнем уровне*

Такое изучение необходимо провести по каждой функциональной задаче или их

комплексу, которые будут решаться с помощью БД.

Результатом проектирования на внешнем уровне будет перечень атрибутов

(реквизитов) оперативной и условно-постоянной информации, которые необходимо

хранить в БД, с указанием источников их получения и формы представления.

Однако этот перечень не исключает возможности существования в нем

избыточности, дублирования, несогласованности и других недостатков. Поэтому

на этом процесс не заканчивается, а осуществляется переход к этапу

инфологического проектирования.