Автореферат

Наименование Современные технологии проектирования информационных систем

Автор А.В.Михалькевич

Специальность компьютерных технологий,

Анотация

Anotation in English

Ключевые слова

Количество символов 83930

Содержание

Введение

1 Понятие информационной системы

Информационная система (ИС) — система, предназначенная для хранения, поиска и обработки информации, и соответствующие организационные ресурсы (человеческие, технические, финансовые и т. д.), которые обеспечивают и распространяют информацию.

ИС предназначена для своевременного обеспечения надлежащих людей надлежащей информацией, то есть для удовлетворения конкретных информационных потребностей в рамках определённой предметной области, при этом результатом функционирования информационных систем является *информационная продукция* — документы, информационные

массивы, базы данных и информационные услуги.

2 Классификация информационных систем

Информационные системы могут значительно различаться по типам объектов, характером и объемом, временным характеристикам и т.д. В связи с тем, что общепринятой классификации ИС до сих пор не существует, поэтому их можно классифицировать по разным признаками:

по масштабам применения - настольные и офисные

по признаку структурированности задач - структурированные (формализуемые), не структурируемые (не формализуемые), частично структурируемые. Частично-структурированные делятся на: ИС репортинга и ИС разработки альтернативных решений (модельные, экспертные). Экспертные в свою очередь делятся на:

централизованные, децентрализованные и коллективного использования;

с интеграцией по уровням управления, по уровням планирования и т.д.

по функциональному признаку – производственные, маркетинговые (анализа рынка, рекламные, снабженческие и т.п.), финансовые (бухгалтерские, статистические, и т.п.), кадровые;

по квалификации персонала и уровням управления – стратегические (топ-менеджеров), функциональные (менеджеров среднего звена) и оперативные (специалистов);

по характеру обработки информации: системы обработки данных, системы управления, система поддержки принятия решений;

по оперативности обработки данных – пакетной обработки и оперативные;

по *степени автоматизации* - ручные, автоматические, автоматизированные;

по характеру использования информации - на информационно-поисковые, информационно-справочные, информационно-решающие, управляющие, советующие и т.п.;

по степени централизации обработки информации — на централизованные, децентрализованные, информационные системы коллективного использования;

по характеру использования вычислительных ресурсов – на локальные и распределенные;

по сфере деятельности - на государственные, территориальные (региональные), отраслевые, объединений, предприятий или учреждений, технологических процессов;

по классу реализуемых технологических операций - на системы с текстовыми редакторами, системы с табличными редакторами, СУБД, СУБЗ, системы с графикой, мультимедиа, гипертекстом;

по месту в процессе управления предприятия – на АРМ специалиста, ИС руководителя, ИС внешнего контролера, интегрированные системы, объединяющие в себе часть или все из этих функций;

по концепции построения – файловые, автоматизированные банки данных, банки знаний, ХД;

по режиму работы - на пакетные, диалоговые и смешанные.

2 .1 Общая классификация систем

Системы в природе бывают самые разнообразные, тем не менее, все их можно поделить на:

Абстрактные Абстрактные системы — это продукт человеческого мышления: гипотезы, знания, теоремы.

Материальные Материальные системы получаются из материальных объектов. Всю совокупность материальных систем можно поделить на:

неорганические (технические, химические и др.),

органические (биологические) и

смешанные (где содержатся элементы как органической, так и неорганической природы). В множестве смешанных систем особо следует выделить

эрготехнические системы (систем «человек-машина») – это системы, которые состоят из человека-оператора (группы операторов) и машины (машин). В таких

системах человек с помощью машины осуществляет трудовую деятельность, связанную с производством материальных благ, услуг, а также с управлением и т.п.

2 .2 Классификация по временной характеристике

По временной характеристике системы можно классифицировать:

статические системы– это системы, в которых состояние системы с течением времени не изменяется;
динамические системы – это системы которые с течением времени изменяют свое состояние;
детерминированные - динамические системы, состояние элементов которых в данный момент времени полностью определяет их состояние в любой предыдущий или следующий момент времени;
вероятностные (стохастические) - динамические системы, в которых предусмотреть состояние в вышеописанный способ невозможно. По характеру взаимодействия системы и внешней (окружающей) среды различают: открытые системы. Открытые системы активно взаимодействуют с окружающей средой, сохраняя благодаря этому высокий уровень организованности и развиваясь в сторону осложнения;
закрытые системы. Закрытые системы изолированы от окружающей среды, все процессы, кроме энергетических, происходят лишь внутри самой системы.

2 .3 Классификация по концепции построения

Файловые системы

Информационное обеспечение построено в виде файловых систем. В современных ЭВМ операционная система берет на себя распределение внешней памяти, отображение имен файлов в соответствующие адреса во внешней памяти и обеспечение доступа к данным. Программное обеспечение ИС напрямую использует функции ОС для работы с файлами. Файловые системы обычно обеспечивают хранение слабо структурированной информации, оставляя дальнейшую структуризацию прикладным программам. В таких системах сложно решить проблемы согласования данных в разных файлах, коллективного доступа к данным, модификации структуры данных.

Автоматизированные банки данных

Банк данных -это система специальным образом организованных БД, программных, технических, языковых и организационно -методических средств, предназначенных для обеспечения централизованного накопления и коллективного многоцелевого использования данных. В отличие от файловых систем, структура БД меньше зависит от прикладных программ, а все функции по работе с БД сосредоточены в специальном компоненте - системе управления базами данных (СУБД), которая играет центральную роль в функционировании банка данных, так как обеспечивает связь прикладных программ и пользователей, данными. Сведения о структуре БД сосредоточены в словаре-справочнике (репозитории). Этот вид информации называется метаинформацией. В состав метаинформации входит семантическая информация, физические характеристики данных и информация об их использовании. С

помощью словарей данных автоматизируется процесс использования метаинформации в ИС.

Интеллектуальные банки данных (банки знаний, БЗ)

Это сравнительно новый способ построения ИС, при котором информация о предметной области условно делится между двумя базами. Если БД содержит сведения о количественных и качественных характеристиках конкретных объектов, то БЗ содержит сведения о закономерностях в ПО, позволяющие выводить новые факты из имеющихся в БД; метаинформацию; сведения о структуре предметной области; сведения, обеспечивающие понимание запроса и синтез ответа. Если в традиционном банке данных знания о предметной области заложены программистом в каждую прикладную программу, а также в структуру БД, то в интеллектуальном банке данных они хранятся в базе знаний и отделены от прикладных программ. В отличие от данных, знания активны: на их основе формируются цели и выбираются способы их достижения. Например, ИБД в системе складского учета может автоматически реагировать на такое событие, как уменьшение количества деталей на складе до критической нормы, при этом ИБД без участия пользователя генерирует документы для заказа этих деталей и отправляет их по электронной почте поставщику. Другое характерное отличие знаний от данных -связность, причем знания отражают как структурные взаимосвязи между объектами предметной области, так и вызванные конкретными бизнес -процессами, например такие связи, как "происходит одновременно", "следует из...", "если -то" и др. Наконец, существенную роль в ИБД играет форма представления информации для пользователя: она должна быть как можно ближе к естественным для человека способам обмена данными (профессиональный естественный язык, речевой ввод / вывод, графическая форма).

Хранилища данных - ХД

В настоящее время в корпоративных БД накоплены гигантские объемы информации, однако она недостаточно эффективно используется в процессе управления бизнесом, поэтому бурно развивается новая форма построения ИС - склады (хранилища) данных. ХД представляет собой автономный банк данных, в котором база данных разделена на два компонента: оперативная БД хранит текущую информацию, квазипостоянная БД содержит исторические данные, например, в оперативной БД могут содержаться данные о продажах за текущий год, а в квазипостоянной БД хранятся систематизированные годовые отчеты и балансы за все время существования предприятия. Подсистема оперативного анализа данных позволяет эффективно и быстро анализировать текущую информацию. Подсистема принятия решений пользуется обобщенной и исторической информацией, применяет методы логического вывода. Для общения с пользователем служит универсальный интерфейс.

2 .4 Классификация по способу распределения ресурсов

Локальные информационные системы

Локальные ИС используют одну ЭВМ и предназначены для автоматизации отдельных функций управления на отдельных уровнях управления. Такая ИС мо жет быть однопользовательской, функционирующей в отдельных подразделениях системы управления.

Распределенные информационные системы

В распределенных ИС взаимодействуют несколько ЭВМ, связанных сетью. Отдельные узлы сети обычно территориально удалены друг от друга, решают разные задачи, но используют общую информационную базу.

2 .5 Классификация по архитекутре

По степени распределённости отличают:

настольные (*desktop*), или *локальные* ИС, в которых все компоненты (БД, СУБД, клиентские приложения) находятся на одном компьютере;
распределённые (*distributed*) ИС, в которых компоненты распределены по нескольким компьютерам (облачные системы, серверные решения).

Распределённые ИС, в свою очередь, разделяют на:

файл-серверные ИС (ИС с архитектурой «файл-сервер»);
клиент-серверные ИС (ИС с архитектурой «клиент-сервер»).

В файл-серверных ИС база данных находится на файловом сервере, а СУБД и клиентские приложения находятся на рабочих станциях.

В клиент-серверных ИС база данных и СУБД находятся на сервере, а на рабочих станциях находятся только клиентские приложения.

В свою очередь, клиент-серверные ИС разделяют на *двухзвенные* и *многозвенные*.

В двухзвенных (англ. *two-tier*) ИС всего два типа «звеньев»: сервер базы данных, на котором находятся БД и СУБД (back-end), и рабочие станции, на которых находятся клиентские приложения (front-end). Клиентские приложения обращаются к СУБД напрямую.

В многозвенных (англ. *multi-tier*) ИС добавляются промежуточные «звенья»: серверы приложений (*application servers*). Пользовательские клиентские приложения не обращаются к СУБД напрямую, они взаимодействуют с промежуточными звеньями. Типичный пример применения трёхзвенной архитектуры — современные веб-приложения, использующие базы данных. В таких приложениях помимо звена СУБД и клиентского звена, выполняющегося в веб-браузере, имеется как минимум одно промежуточное звено — веб-сервер с соответствующим серверным программным обеспечением.

2 .6 Классификация по разным признакам

Имеется пять основных классификационных признаков информации:

Место возникновения

Входная информация – это информация, поступающая в фирму или ее подразделения.

Выходная информация – это информация, поступающая из фирмы в другую фирму, организацию (подразделение).

Внутренняя информация возникает внутри объекта, внешняя информация – за пределами объекта.

Стабильность

Переменная информация отражает фактические количественные и качественные

характеристики производственно-хозяйственной деятельности фирмы. Она может меняться для каждого случая как по назначению, так и по количеству.

Постоянная (условно-постоянная) информация – это неизменная и многократно используемая в течение длительного периода времени информация.

Стадия обработки

Первичная информация – это информация, которая возникает непосредственно в процессе деятельности объекта и регистрируется на начальной стадии.

Вторичная информация – это информация, которая получается в результате обработки первичной информации и может быть промежуточной и результатной.

Промежуточная информация используется в качестве исходных данных для последующих расчетов.

Результатная информация получается в процессе обработки первичной и промежуточной информации и используется для выработки управленческих решений.

Способ отображения

Текстовая информация – это совокупность алфавитных, цифровых и специальных символов, с помощью которых представляется информация на физическом носителе (бумага, изображение на экране дисплея).

Графическая информация – это различного рода графики, диаграммы, схемы, рисунки и т.д.

Функция управления

Плановая информация – это информация о параметрах объекта управления на будущий период. На эту информацию идет ориентация всей деятельности фирмы.

Нормативно-справочная информация содержит различные нормативные и справочные данные. Ее обновление происходит достаточно редко.

Учетная информация – это информация, которая характеризует деятельность фирмы за определенный прошлый период времени. На основании этой информации могут быть проведены следующие действия: скорректирована плановая информация, сделан анализ хозяйственной деятельности фирмы, приняты решения по более эффективному управлению работами и пр. (информация бух. учета, статистическая информация и т.д.).

Оперативная (текущая) информация – это информация, использующаяся в оперативном управлении и характеризующая производственные процессы в текущий (данный) период времени. К оперативной информации предъявляются серьезные требования по скорости поступления и обработки, а также по степени ее достоверности. От того, насколько быстро и качественно проводится ее обработка, во многом зависит успех фирмы на рынке.

Степень автоматизации

По степени автоматизации ИС делятся на:

> *автоматизированные*: информационные системы, в которых автоматизация может быть неполной (то есть требуется постоянное вмешательство персонала);
> *автоматические*: информационные системы, в которых автоматизация является полной, то есть вмешательство персонала не требуется или требуется только эпизодически.
> ручные информационные системы («без компьютера») существовать не могут, поскольку существующие определения предписывают *обязательное* наличие в составе ИС аппаратно-программных средств. Вследствие этого понятия «автоматизированная

информационная система», «компьютерная информационная система» и просто «информационная система» являются синонимами.

3 Информация и данные

Термин *информация* происходит от латинского informatio, что означает разъяснение, осведомление, изложение. С позиций материалистической философии информация есть отражение реального мира с помощью сведений (сообщений). Сообщение – это форма представления информации в виде речи, текста, изображения, цифровых данных, графиков, таблиц и т.п. В широком смысле информация – это общенаучное понятие, включающее в себя обмен сигналами между живой и неживой природой, людьми и устройствами.

Информация – сведения об объектах и явлениях окружающей среды, их параметрах, свойствах и состоянии, которые уменьшают имеющуюся о них степень неопределенности, неполноты знаний.

Наряду с информацией употребляется понятие **данные**. Данные могут рассматриваться как признаки или записанные наблюдения, которые по каким-то причинам не используются, а только хранятся. Если появляется возможность использовать эти данные для уменьшения неопределенности в чем-либо, эти данные превращаются в информацию. Поэтому информацией являются используемые данные.

Одной из важнейших разновидностей информации является информация экономическая. Ее отличительная черта – связь с процессами управления коллективами людей, организацией. Экономическая информация сопровождает процессы производства, распределения, обмена и потребления материальных благ и услуг.

Экономическая информация – совокупность сведений, отражающих социально-экономические процессы и служащих для управления этими процессами и коллективами людей в производственной и непроизводственной сфере.

Особенности экономической информации:

1.Отражает деятельность предприятия через систему, в основном, натуральных и стоимостных показателей.

2.Она возникает рассредоточено, в низовых точках объекта небольшими порциями.

3.Большая часть данных фиксируется в первичных документах, непригодных для автоматизированного ввода, поэтому возникает необходимость набора данных вручную (из документа на клавиатуре).

4.Она характеризуется большими объемами данных для обработки и простыми алгоритмами расчетов (преобладают 4 арифметических действия: +, -, *, /).

5.Она характеризуется повторяемостью циклов возникновения и обработки в установленных временных пределах.

При работе с информацией всегда имеется ее источник и потребитель (получатель). Пути и процессы, обеспечивающие передачу сообщений от источника информации к ее потребителю, называются информационными коммуникациями. Для потребителя информации очень важной характеристикой является ее адекватность.

Адекватность информации – это уровень соответствия полученной информации образу реального объекта, процесса, явления и т.п. В реальной жизни вряд ли возможна ситуация, когда вы сможете рассчитывать на полную адекватность информации реальному состоянию объекта или процесса. Всегда присутствует некоторая степень неопределенности. От степени адекватности информации зависит правильность принятия решений человеком.

4 Пользователи информационных систем

Пользователей ИС (информационных систем) можно разделить на несколько групп:

случайный пользователь, взаимодействие которого с ИС не обусловлено служебными обязанностями;

конечный пользователь (потребитель информации) - лицо или коллектив, в интересах которых работает ИС. Он работает с ИС повседневно, связан с жестко ограниченной областью деятельности и, как правило, не является программистом, например, это может быть бухгалтер, экономист, руководитель подразделения;

коллектив специалистов (персонал ИС), включающий администратора банка данных, системного аналитика, системных и прикладных программистов.

Состав и функции персонала ИС - информационных систем:

Администратор - это специалист (или группа специалистов), который понимает потребности конечных пользователей, работает с ними в тесном контакте и отвечает за определение, загрузку, защиту и эффективность работы банка данных. Он должен координировать процесс сбора информации, проектирования и эксплуатации БД, учитывать текущие и перспективные потребности пользователей.

Системные программисты - это специалисты, которые занимаются разработкой и сопровождением базового математического обеспечения ЭВМ (ОС, СУБД, трансляторов, сервисных программ общего назначения).

Прикладные программисты - это специалисты, которые разрабатывают программы для реализации запросов к БД.

Аналитики - это специалисты, которые строит математическую модель предметной области, исходя из информационных потребностей конечных пользователей; ставит задачи для прикладных программистов. На практике персонал небольших ИС часто состоит из одного - двух специалистов, которые выполняют все перечисленные функции.

5 База данных, как информационная система

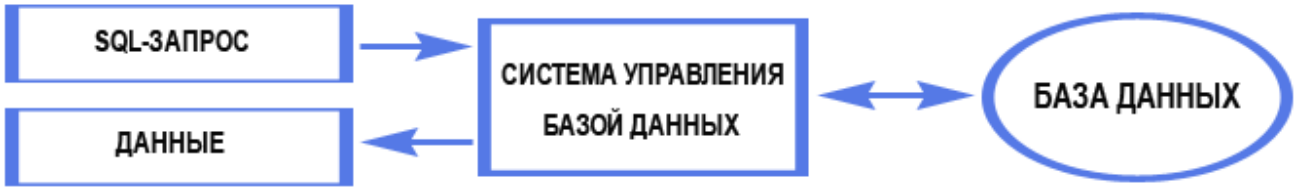
База данных - набор сведений, хранящихся некоторым упорядоченным способом. Можно сравнить базу данных со шкафом, в котором хранятся документы. Иными словами, база данных - это хранилище данных. Сами по себе базы данных не представляли бы интереса, если бы не было систем управления базами данных (СУБД).

Система управления базами данных - это совокупность языковых и программных средств, которая осуществляет доступ к данным, позволяет их создавать, менять и удалять, обеспечивает безопасность данных и т.д. В общем СУБД - это система, позволяющая создавать

базы данных и манипулировать сведениями из них. А осуществляет этот доступ к данным СУБД посредством специального языка - SQL.

SQL - язык структурированных запросов, основной задачей которого является предоставление простого способа считывания и записи информации в базу данных.

Итак, простейшая схема работы с базой данных выглядит примерно так:



По характеру использования СУБД делят на однопользовательские (предназначенные для создания и использования БД на персональном компьютере) и многопользовательские (предназначенные для работы с единой БД нескольких компьютеров, объединенных в локальные сети). Вообще деление по характеру использования можно представить следующей схемой:

Наиболее известные однопользовательские СУБД - Microsoft Visual FoxPro и Access, многопользовательские - MS SQL Server, Oracle и MySQL.

5 .1 Реляционные базы данных

Реляционная база данных – это набор данных с предопределенными связями между ними. Эти данные организованны в виде набора таблиц, состоящих из столбцов и строк. В таблицах хранится информация об объектах, представленных в базе данных. В каждом столбце таблицы хранится определенный тип данных, в каждой ячейке – значение атрибута. Каждая стока таблицы представляет собой набор связанных значений, относящихся к одному объекту или сущности. Каждая строка в таблице может быть помечена уникальным идентификатором, называемым первичным ключом, а строки из нескольких таблиц могут быть связаны с помощью внешних ключей. К этим данным можно получить доступ многими способами, и при

этом реорганизовывать таблицы БД не требуется.

Важные аспекты реляционных БД

SQL

SQL (Structured Query Language) – основной интерфейс работы с реляционными базами данных. SQL стал стандартом Национального института стандартов США (ANSI) в 1986 году. Стандарт ANSI SQL поддерживается всеми популярными ядрами реляционных БД. Некоторые из ядер также включают расширения стандарта ANSI SQL, поддерживающие специфичный для этих ядер функционал. SQL используется для добавления, обновления и удаления строк данных, извлечения наборов данных для обработки транзакций и аналитических приложений, а также для управления всеми аспектами работы базы данных.

Целостность данных

Целостность данных – это полнота, точность и единообразие данных. Для поддержания целостности данных в реляционных БД используется ряд инструментов. В их число входят первичные ключи, внешние ключи, ограничения «Not NULL», «Unique», «Default» и «Check». Эти ограничения целостности позволяют применять практические правила к данным в таблицах и гарантировать точность и надежность данных. Большинство ядер БД также поддерживает интеграцию пользовательского кода, который выполняется в ответ на определенные операции в БД.

Транзакции

Транзакция в базе данных – это один или несколько операторов SQL, выполненных в виде последовательности операций, представляющих собой единую логическую задачу. Транзакция представляет собой неделимое действие, то есть она должна быть выполнена как единое целое и либо должна быть записана в базу данных целиком, либо не должен быть записан ни один из ее компонентов. В терминологии реляционных баз данных транзакция завершается либо действием COMMIT, либо ROLLBACK. Каждая транзакция рассматривается как внутренне связный, надежный и независимый от других транзакций элемент.

Соответствие требованиям ACID

Для соблюдения целостности данных все транзакции в БД должны соответствовать требованиям ACID, то есть быть атомарными, единообразными, изолированными и надежными.

Атомарность – это условие, при котором либо транзакция успешно выполняется целиком, либо, если какая-либо из ее частей не выполняется, вся транзакция отменяется. **Единообразие** – это условие, при котором данные, записываемые в базу данных в рамках транзакции, должны соответствовать всем правилам и ограничениям, включая ограничения целостности, каскады и триггеры. **Изолированность** необходима для контроля над согласованностью и гарантирует базовую независимость каждой транзакции. **Надежность** подразумевает, что все внесенные в базу данных изменения на момент успешного завершения транзакции считаются постоянными.

5 .1 .1 MySQL

MySQL представляет систему управления реляционными базами данных (СУБД). На сегодняшний день это одна из самых популярных систем управления базами данных.

Изначальным разработчиком данной СУБД была шведская компания MySQL AB. В 1995 году она выпустила первый релиз MySQL. В 2008 году компания MySQL AB была куплена компание Sun Microsystems, а в 2010 году уже компания Oracle поглотила Sun и тем самым приобрела права на торговую марку MySQL. Поэтому MySQL на сеголняшней день развивается под эгидой Oracle.

Текущей актуальной версией СУДБ является версия 8.0, которая вышла в январе 2018 года.

MySQL обладает кроссплатформенностью, имеются дистрибутивы под самые различные ОС, в том числе наиболее популярные версии Linux, Windows, MacOS.

Официальный сайт проекта: https://www.mysql.com/.

Установка MySQL

Для установки MySQL необходимо загрузить дистрибутив по адресу http://dev.mysql.com/downloads/mysql/ и выбрать нужную версию.

Создание базы данных

Для создания базы данных используется команда `CREATE DATABASE`:

```
CREATE DATABASE [IF NOT EXISTS] имя_базы_даных;
```

В конце команды указывается имя базы данных.

Первая форма `CREATE DATABASE имя_базы_даных` пытается создать базу данных, но если такая база данных уже существует, то операция возвратит ошибку.

Вторая форма `CREATE DATABASE IF NOT EXISTS имя_базы_даных` пытается создать базу данных, если на сервере отсутствует бд с таким именем.

Установка базы данных

После создания БД с ней производятся различные операции: создание таблиц, добавление и получение данных и т.д. Но чтобы установить производить эти операции, надо установить определенную базу данных в качестве используемой. Для этого применяется оператор USE:

```
USE productsdb;
```

Удаление базы данных

Для удаления базы данных применяется команда DROP DATABASE, которая имеет следующий синтаксис:

```
DROP DATABASE [IF EXISTS] имя_базы_даных;
```

Первая форма `DROP DATABASE имя_базы_даных` пытается удалить базу данных, но если такая база данных отсутствует на сервере, то операция возвратит ошибку.

Вторая форма `DROP DATABASE IF EXISTS имя_базы_даных` пытается удалить базу данных, если на сервере имеется бд с таким именем.

Например, удалим выше созданную базу данных productsdb:

```
DROP DATABASE productsdb;
```

Создание таблицы

Таблица может существовать только в контексте базы данных. Вначале создается база данных. И затем, чтобы указать, что все дальнейшие операции, в том числе создание таблицы, будут производиться с этой базой данных, применяется команда USE.

Для создания таблиц используется команда CREATE TABLE. Эта команды применяет ряд операторов, которые определяют столбцы таблицы и их атрибуты. Общий формальный синтаксис команды CREATE TABLE:

```
CREATE TABLE название_таблицы
(название_столбца1 тип_данных атрибуты_столбца1,
 название_столбца2 тип_данных атрибуты_столбца2,
 ..............................................
 название_столбцаN тип_данных атрибуты_столбцаN,
 атрибуты_уровня_таблицы
)
```

После команды CREATE TABLE идет название таблицы. Имя таблицы выполняет роль ее идентификатора в базе данных, поэтому оно должно быть уникальным. Затем в скобках перечисляются названия столбцов, их типы данных и атрибуты. В самом конце можно определить атрибуты для всей таблицы. Атрибуты столбцов, а также атрибуты таблицы указывать необязательно.

Создадим простейшую таблицу. Для этого выполним следующий скрипт:

```
CREATE DATABASE productsdb;

USE productsdb;

CREATE TABLE Customers
(
    Id INT,
    Age INT,
    FirstName VARCHAR(20),
    LastName VARCHAR(20)
);
```

Далее собственно идет создание таблицы, которая называется Customers. Она определяет четыре столбца: Id, Age, FirstName, LastName. Первые два столбца представляют идентификатор клиента и его возраст и имеют тип `INT`, то есть будут хранить числовые значения. Следующие столбцы представляют имя и фамилию клиента и имеют тип `VARCHAR(20)`, то есть представляют строку длиной не более 20 символов. В данном случае для каждого столбца определены имя и тип данных, при этом атрибуты столбцов и таблицы в целом отсутствуют.

И в результате выполнения этой команды будет создана база данных productsdb, в которой будет создана таблица Customers.

Переименование таблиц

Если после создания таблицы мы захотим ее переименовать, то для этого нужно использовать команду RENAME TABLE, которая имеет следующий синтаксис:

```
RENAME TABLE старое_название TO новое_название;
```

Полное удаление данных

Для полного удаления данных, очистки таблицы применяется команда TRUNCATE TABLE. Например, очистим таблицу Clients:

```
TRUNCATE TABLE Clients;
```

Удаление таблиц

Для удаления таблицы из БД применяется команда DROP TABLE, после которой указывается название удаляемой таблицы. Например, удалим таблицу Clients:

```
DROP TABLE Clients;
```

Добавление данных. Команда INSERT

Для добавления данных в БД в MySQL используется команда INSERT, которая имеет следующий формальный синтаксис:

```
INSERT [INTO] имя_таблицы [(список_столбцов)] VALUES (значение1, значение2,
... значениеN)
```

После выражения INSERT INTO в скобках можно указать список столбцов через запятую, в которые надо добавлять данные, и в конце после слова VALUES скобках перечисляют добавляемые для столбцов значения.

Важно, чтобы между значениями и типами данных столбцов было соответствие.

Выборка данных. Команда SELECT

Для выборки данных из БД в MySQL применяется команда SELECT. В упрощенном виде она имеет следующий синтаксис:

```
SELECT список_столбцов FROM имя_таблицы
```

Получим все данные из таблицы products:

```
SELECT * FROM products;
```

Символ звездочка * указывает, что нам надо получить все столбцы. Вместо символа * можно перечислить название требуемых полей через запятую.

```
SELECT name, price FROM products;
```

Для выбора по условию, можно воспользоваться фильтром WHERE:

```
SELECT * FROM Products
WHERE Manufacturer = 'Samsung';
```

Если условие истинно, то строка попадает в результирующую выборку. В качестве можно использовать операции сравнения, которые сравнивают два выражения:

- =: сравнение на равенство
- !=: сравнение на равенство
- <>: сравнение на неравенство
- <: меньше чем
- >: больше чем
- <=: меньше чем или равно
- >=: больше чем или равно

К примеру, выберем всех товары, производителем которых является компания Samsung:

Обновление данных. Команда UPDATE

Команда UPDATE применяется для обновления уже имеющихся строк. Она имеет следующий формальный синтаксис:

```
UPDATE имя_таблицы
SET столбец1 = значение1, столбец2 = значение2, ... столбецN = значениеN
[WHERE условие_обновления]
```

Пример использования:

```
UPDATE Products
SET Manufacturer = 'Samsung Inc.'
WHERE Manufacturer = 'Samsung';
```

Также можно обновлять сразу несколько столбцов:

```
UPDATE Products
SET Manufacturer = 'Samsung',
    price = '150'
WHERE Manufacturer = 'Samsung Inc.';
```

Удаление данных. Команда DELETE

Команда DELETE удаляет данные из БД. Она имеет следующий формальный синтаксис:

```
DELETE FROM имя_таблицы
[WHERE условие_удаления]
```

Например, удалим строки, у которых производитель - Huawei:

```
DELETE FROM Products
WHERE Manufacturer='Huawei';
```

5 .2 NoSQL базы данных

NoSQL – группа типов БД, предлагающих подходы, отличные от стандартного реляционного шаблона. Говоря NoSQL, подразумевают либо «не-SQL», либо «не только SQL», чтобы уточнить, что иногда допускается SQL-подобный запрос.

Базы данных «ключ-значение»

В базах данных «ключ-значение» для хранения информации вы предоставляте ключ и объект данных, который нужно сохранить. Например, JSON-объект, изображение или текст. Чтобы запросить данные, отправляете ключ и получаете **blob-объект**.

Особенности:

- хранилища обеспечивают быстрый и малозатратный доступ;
- часто хранят данные конфигураций и информацию о состоянии данных, представленных словарями или хэшем;
- нет жёсткой схемы отношения между данными, поэтому в таких БД часто хранят одновременно различные типы данных;
- разработчик отвечает за определение схемы именования ключей и за то, чтобы значение имело соответствующий тип/формат.

Примеры:

- Redis
- memcached
- etcd

5 .3 Документные базы данных

Документные базы данных (также документоориентированные БД или хранилища документов), совместно используют базовую семантику доступа и поиска хранилищ ключей и значений. Такие БД также используют ключ для уникальной идентификации данных. Разница между хранилищами «ключ-значение» и документными БД заключается в том, что вместо хранения blob-объектов, документоориентированные базы хранят данные в структурированных форматах – JSON, BSON или XML.

Особенности:

- база данных не предписывает опредёленный формат или схему;
- каждый документ может иметь свою внутреннюю структуру;
- документные БД являются хорошим выбором для быстрой разработки;
- в любой момент можно менять свойства данных, не изменяя структуру или сами данные.

Примеры:

- MongoDB

RethinkDB

6 Программирование информационных систем

Можно выделить несколько направлений программирования информационных систем, в каждом из которых используют свой набор языков. Перечислим основные направления в порядке возрастания сложности:

Веб-разработка. Популярные языки: JavaScript, PHP, Python, Ruby.
Мобильная разработка. Популярные языки: Java, Kotlin, Swift.
Разработка игр и программ для настольных компьютеров. Популярные языки: C++, C#, C.
Big Data, машинное обучение. Популярные языки: Python, R, Scala.

6 .1 Фреймворки в веб-разработке

Фреймворки — это программные продукты, которые упрощают создание и поддержку технически сложных или нагруженных проектов. Фреймворк, как правило, содержит только базовые программные модули, а все специфичные для проекта компоненты реализуются разработчиком на их основе. Тем самым достигается не только высокая скорость разработки, но и большая производительность и надёжность решений.

Веб-фреймворк — это платформа для создания сайтов и веб-приложений, облегчающее разработку и объединение разных компонентов большого программного проекта. За счёт широких возможностей в реализациии бизнес-логики и высокой производительности эта платформа особенно хорошо подходит для создания сложных сайтов, бизнес-приложений и веб-сервисов.

Основные преимущества фреймворков

Экономическая эффективность и целесообразность использования фреймворков

С точки зрения бизнеса разработка на фреймворке почти всегда экономически эффективнее и качественнее по результату, нежели написание проекта на чистом языке программирования без использования каких-либо платформ. Разработка без использования платформы может быть правильным решением только в двух случаях — либо проект совсем простой и не требующий дальнейшего развития, либо очень нагруженный и требует очень низкоуровневой оптимизации (например, веб-сервисы с десятками тысяч обращений в секунду). Во всех других случаях разработка на программной платформе быстрее и качественнее.

Если сравнивать фреймворки с другими классами платформ — SaaS, CMS или CMF — то фреймворки значительно эффективнее использовать в проектах со сложной бизнес-логикой и с высокими требованиями к скорости работы, надёжности и безопасности. Но в простых и типовых проектах без значимых требований скорость и стоимость разработки на фреймворке будет выше, нежели на SaaS или CMS.

Технические преимущества фреймворков

Одним из главных преимуществ в использовании фреймворков является то, что фреймворк определяет унифицированную структуру для построенных на его базе приложений. Поэтому приложения на фреймворках значительно проще сопровождать и дорабатывать, так как стандартизированная структура организации компонентов понятна всем разработчикам на этой платформе и не требуется долго разбираться в архитектуре, чтобы понять принцип работы приложения или найти место реализации того или иного функционала. Большинство фреймворков для разработки веб-приложений использует парадигму MVC (модель-представление-контроллер) — то есть очень во многих фреймворках идентичный подход к организации компонентов приложения и это ещё больше упрощает понимание архитектуры приложения даже на незнакомом разработчику фреймворке.

Проектирование архитектуры ПО при разработке на фреймворке тоже очень упрощается — в методологиях фреймворков обычно заложены лучшие практики программной инженерии и просто следуя этим правилам можно избежать многих проблем и ошибок в проектировании. По сути, фреймворк — это множество конкретных и абстрактных классов, связанных между собой и упорядоченных согласно методологии фреймворка. Конкретные классы обычно реализуют взаимные отношения между классами, а абстрактные классы представляют собой точки расширения, в которых заложенный во фреймворк базовый функционал может быть использован «как есть» или адаптирован под задачи конкретного приложения. Для обеспечения расширения возможностей в большинстве фреймворков используются техники объектно-ориентированного программирования: например, части приложения могут наследоваться от базовых классов фреймворка или отдельные модули могут быть подключены как примеси.

Экосистемы веб-фреймворков также богаты на готовые реализации многих функциональных возможностей. Разработчикам при работе над типовыми задачами не надо «изобретать велосипеды», так как они могут воспользоваться уже созданной сообществом реализацией. А это не только сокращает затраты времени и денег, но и позволяет добиться более высокой стабильности решения — компонент, который используется и дорабатывается тысячами других разработчиков обычно более качественно реализован и лучше протестирован на всевозможных сценариях, нежели решение, которое может в адекватные сроки разработать один разработчик или даже небольшая команда.

Фреймворки — это не библиотеки

Библиотека — это более простой компонент архитектуры программного обеспечения. Программная библиотека может быть использована просто как набор подсистем близкой функциональности, не влияя на архитектуру основного программного продукта и не накладывая на неё никаких ограничений.

Фреймворк же не просто даёт разработчику нужный функционал, но ещё и диктует правила построения архитектуры приложения, задавая на начальном этапе разработки поведение по умолчанию, формируя каркас, который нужно будет расширять и изменять согласно указанным требованиям. Фреймворк также может включать вспомогательные программы,

библиотеки кода, язык сценариев и другое ПО, облегчающее разработку и объединение разных компонентов большого программного проекта.

Сравнение с альтернативами

Перед веб-разработчиками часто стоит выбор между коробочными CMS и фреймворками для реализации проекта. У каждого из подходов есть свои плюсы и минусы, ниже мы рассмотрим преимущества и недостатки разработки на фреймворках.

Плюсы фреймворков

Разработка на фреймворке (в отличие от самописных решений) позволяет добиться простоты сопровождаемости проекта.
Возможна (и относительно проста) реализация любых бизнес-процессов, а не только тех, которые изначально заложены в систему. Также проекты на базе фреймворков легко масштабируемы и модернизируемы.
Решения на фреймворках, как правило, работают значительно быстрее и выдерживают большую нагрузку, чем CMS и самописные системы. Именно поэтому много популярных интернет-магазинов работают не на коробочных CMS, а на фреймворках. По уровню безопасности решения на фреймворках значительно превосходят самописные системы и сравнимы с CMS (как правило, сайты на фреймворках даже безопаснее).

Минусы фреймворков

Сроки разработки типового функционала на фреймворках больше, чем при использовании CMS. Фреймворки содержат только базовые компоненты бизнес-логики уровня приложения, поэтому многие функции реализовываются индивидуально.
Для разработки на фреймворке требуется понимание бизнес-процессов, которые требуется реализовать. Например, если в CMS уже есть некий предустановленный процесс обработки заказов, то фреймворки такого не предоставляют.

Популярные фреймворки для веб-разработки

Популярные php-фреймворки:

Yii
Symphony
Zend
Laravel
Kohana
CodeIgniter

Node.js фрэймворк

Express

Наиболее популярные Ruby-фреймворки:

Ruby on Rails (явный лидер)
Sinatra

Популярные Python-фреймворки:

Django
Plone
Twisted
Flask
Tornado

Популярный Elixir-фреймворк:

Phoenix

Популярные Go-фреймворки:

Gin
Martini

Популярные Java-фреймворки:

Spring MVC
JSF
GWT

Популярные Kotlin-фреймворки:

Spring
ktor

Фреймворки от Microsoft (мультиязычные):

ASP.NET (ASP.NET MVC) и .Net Framework

Заключение

Список использованных источников

Приложения

1. [электронный документ] 5ef22783c581f_Заявка диплом.docx
2. [электронный документ] 5ef227b42e2a6_zayavka.docx
3. [электронный документ] 5ef25a754ac4c_diploma_task.docx