1. Предназначение и задачи СУБД. Языки общения

СУБД - это совокупность программ и языковых средств (специальных языков описания и манипулирования данными), предназначенных для создания, ведения и использования баз данных.

СУБД - является средством организации доступа к базам данных и при этом не решает прикладных задач. Обработка найденных СУБД, сложные вычисления формирования выходных документов по заданной форме выполняются с помощью прикладных программ, составленных с использованием языков манипулирования данными.

База данных решает следующие задачи:

хранение информации и организация защиты;

изменение хранимых данных (обновление, добавление, удаление);

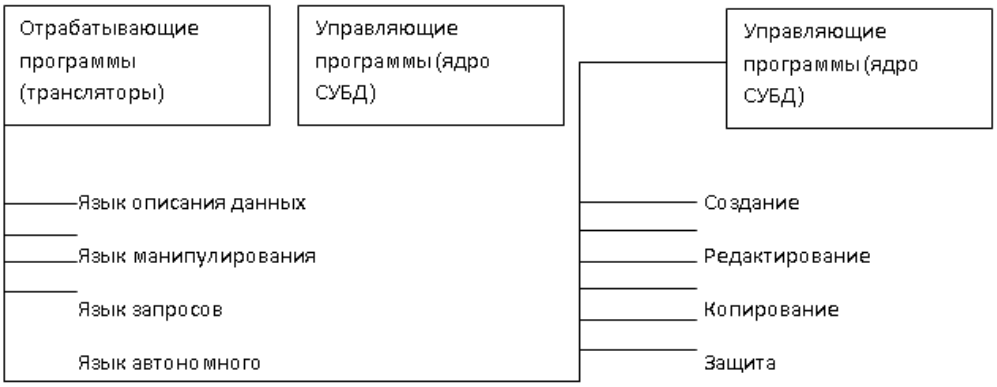
поиск и отбор данных по запросам пользователей и прикладных программ;

вывод данных в заданной форме.

В процессе функционирования каждый пользователь обращается к БД с помощью специального языка общения с ней.

Для прикладного программиста средства этого языка должны быть ориентированы на язык программирования, использующийся при написании программ. Такой язык общения получил название языка манипулирования данными.

Язык общения конечных пользователей - должностных лиц органов управления не должен быть связан с языком программирования. В качестве такого языка чаще всего выступает разновидность формализованного языка, называемого языком запросов.



. Виды обеспечений используемых в СУБД

Говоря о «банке данных» имеется в виду несколько баз данных, в том числе, технических, программных, лингвистических и информационных средств их формирования и ведения, а также о коллективе специалистов обеспечивающих его функционирование.

Техническое обеспечение БнД - это все те аппаратные средства, которые обеспечивают его функционирование и работу пользователей.

Математическое обеспечение БД - представляет собой совокупность методов, способов, математических моделей и алгоритмов управления БД и решения прикладных задач.

Программное обеспечение БнД охватывает базовое программное обеспечение - операционные системы компьютеров, используемых для работы банка данных, сетевое и телекоммуникационное программное обеспечение, так как банк данных должен работать и в локальной, и в глобальной сети, базовую СУБД, которая должна быть единой для конкретного банка данных, иначе банк данных превратится в разрозненную совокупность отдельных информационных систем, плохо стыкующихся между собой.

Информационное обеспечение БнД - представляет собой совокупность

системы классификации и кодирования информации, входных документов и вспомогательных информационных массивов.

Лингвистическое обеспечение БнД содержит множество языков, используемых в СУБД, а также набор различных словарей образующих словарный состав информационной системы.

Организационное обеспечение БнД представляет собой комплекс мероприятий и руководящих документов, направленных на организацию повседневной эксплуатации БнД и эффективное информационное обслуживание пользователей.

Примерами банков данных могли бы служить:

банк данных «таможенная информация по экспорту и импорту»;

банк данных «нарушений таможенных правил»;

банк данных «поступлений таможенных платежей»;

банк данных «нормативно-правовых актов ГТК России (ФТС)»

Данные формулировки фактически определяют цели информационных систем и достаточно точно отражают требуемое информационное наполнение БнД.

Банк данных предполагает накопление и хранение информации в течение нескольких лет. Поэтому очень важно заранее определить для БнД стабильное информационное направление, которое по возможности не должно терять актуальность и изменять структуру с течением времени. Может меняться структура входной информации, но структура самого БнД должна быть спроектирована так, чтобы оставаться стабильной.

Банк данных является автоматизированной информационной системой общего назначения, которая должна функционировать и использоваться как самостоятельно, так и в рамках информационно-расчетных систем.

К банкам данных предъявляются следующие требования:

адекватность информации состоянию предметной области;

надежность функционирования;

быстродействие и производительность;

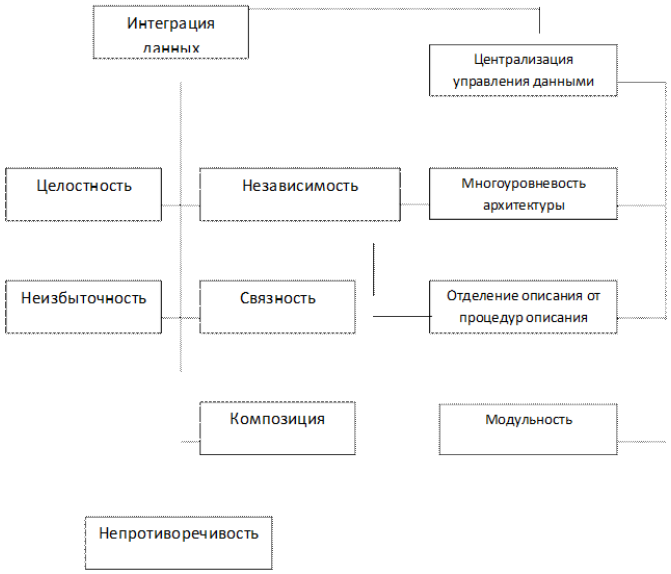
простота и удобство использования;

защита информации;

возможность расширения.

. Принципы построения банка данных

В основе построения банков данных лежат определенные научные принципы, позволяющие создавать высококачественные системы, отвечающие современным требованиям. Выбор принципов и их воплощение в конкретной системе составляют основу проектирования. Создание такой сложной автоматизированной системы, как банк данных, определяется общей закономерностью: требования порождают принципы, принципы формируют систему, система дает эффект. Перечень основных (рис.1) из множества используемых принципов можно выделить наиболее существенные, занимающие высший уровень иерархии - принцип интеграции данных и принцип централизации управления ими. Оба принципа определяют суть банка данных: интеграция является основой организации БД, централизация управления - основой организации и функционирования СУБД.



По модели данных

Примеры:

- Иерархические
- Сетевые
- Реляционные
- Объектно-ориентированные
- Объектно-реляционные

По степени распределённости

- Локальные СУБД (все части локальной СУБД размещаются на одном компьютере)
- Распределённые СУБД (части СУБД могут размещаться не только на одном, но на двух и более компьютерах).

По способу доступа к БД

- Файл-серверные

В файл-серверных СУБД файлы данных располагаются централизованно на файл-сервере. СУБД располагается на каждом клиентском компьютере (рабочей станции). Доступ СУБД к данным

осуществляется через локальную сеть. Синхронизация чтений и обновлений осуществляется посредством файловых блокировок.

Преимуществом этой архитектуры является низкая нагрузка на процессор файлового сервера.

Недостатки: потенциально высокая загрузка локальной сети; затруднённость или невозможность централизованного управления; затруднённость или невозможность обеспечения таких важных характеристик, как высокая надёжность, высокая доступность и высокая безопасность. Применяются чаще всего в локальных приложениях, которые используют функции управления БД; в системах с низкой интенсивностью обработки данных и низкими пиковыми нагрузками на БД.

На данный момент файл-серверная технология считается устаревшей, а её использование в крупных информационных системах — недостатком[3].

Примеры: Microsoft Access, Paradox, dBase, FoxPro, Visual FoxPro.

- Клиент-серверные

Клиент-серверная СУБД располагается на сервере вместе с БД и осуществляет доступ к БД непосредственно, в монопольном режиме. Все клиентские запросы на обработку данных обрабатываются клиент-серверной СУБД централизованно.

Недостаток клиент-серверных СУБД состоит в повышенных требованиях к серверу.

Достоинства: потенциально более низкая загрузка локальной сети; удобство централизованного управления; удобство обеспечения таких важных характеристик, как высокая надёжность, высокая доступность и высокая безопасность.

Примеры: Oracle Database, Firebird, Interbase, IBM DB2, Informix, MS SQL Server, Sybase Adaptive Server Enterprise, PostgreSQL, MySQL, Caché, ЛИНТЕР.

- Встраиваемые

Встраиваемая СУБД — СУБД, которая может поставляться как составная часть некоторого программного продукта, не требуя процедуры самостоятельной установки. Встраиваемая СУБД предназначена для локального хранения данных своего приложения и не рассчитана на коллективное использование в сети.

Физически встраиваемая СУБД чаще всего реализована в виде подключаемой библиотеки. Доступ к данным со стороны приложения может происходить через SQL либо через специальные программные интерфейсы.

Примеры: OpenEdge, SQLite, BerkeleyDB, Firebird Embedded, Microsoft SQL Server Compact, ЛИНТЕР.

Остальные принципы в той или иной степени связаны с основными, некоторые из них являются их следствием либо одним из возможных путей реализации.

Так, например, интеграция данных предполагает взаимозависимость данных, что в свою очередь, вместе с принципом композиции позволяет свести избыточность данных к минимуму, т.е. добиться высокой степени неизбыточности данных.

Интеграция данных. Суть этого принципа состоит в объединении отдельных, невзаимосвязанных данных в единое целое. В роли единого информационного массива выступает база данных.

Целостность данных. Этот принцип отражает требования адекватности хранимой в БД информации состоянию предметной области: в любой момент времени данные должны в точности соответствовать свойствам и характеристикам объектов. Нарушение целостности возникает вследствие искажения или даже разрушения (стирания) всех или части данных, а также как результат записи в базу данных неверной информации.

Искажение данных происходит по причине некорректного выполнения программ, операторами, предусматривающие изменение содержимого записей БД. Выдача искаженной информации приводит к тяжелым последствиям. Поэтому поддержание целостности данных является одной из важнейших задач любой информационной системы. Поддержание целостности достигается контролем входной информации, периодической проверкой хранимых в БД данных, применением специальной системы восстановления данных, а также рядом других мероприятий.

Независимость данных. Одним из серьезных недостатков информационных систем ранних разработок была зависимость прикладных программ от данных. В таких системах любые изменения в логической или физической организации информационных массивов неизбежно приводили

к необходимости коррекции прикладных программ. В банках данных, в которые периодически вносятся изменения в организацию без данных, переделка множества прикладных программ привела бы к большим временным и экономическим потерям.

Достижение независимости данных позволяет сократить это потери.

Под независимостью данных будем понимать независимость прикладных программ от хранимых данных, при которой любые изменения в организации данных не требуют коррекции этих программ. Независимость обеспечивается централизацией управления данными, многоуровневостью архитектуры БД и отделением описания данных от процедур обработки данных.

Абсолютной независимости на сегодняшний день в современных системах достичь не удается.

Неизбыточность данных. Под избыточностью понимается дублирование данных. В противоположность этому неизбыточность - это состояние данных, когда каждое из них присутствует в информационном массиве в единственном экземпляре. Избыточность может иметь место как на логическом уровне, когда в структуре данных повторяются одни и те же типы данных, так и на физическом уровне, когда данные хранятся в двух или более экземплярах. Принцип интеграции позволяет свести избыточность к минимуму.

Непротиворечивость данных. Под непротиворечивостью понимается смысловое соответствие между данными. Это состояние базы данных, при котором хранимые в ней данные не противоречат друг другу.

Связность данных. Принцип связности заключается в том, что данные в БД взаимосвязаны, и в связи отражают отношения между объектами предметной области. Множество связей и множество типов данных образуют структуры данных.

Централизация управления данными.

Принцип централизации управления состоит в передаче всех функций управления данными единому комплексу управляющих программ СУБД. При этом все операции, связанные с доступом к БД, выполняются не прикладными программами, а ядром СУБД на основе информации, полученной от этих прикладных программ.

Перечисленные выше принципы построения банков данных фактически реализованы в СУБД ORACLE, и при правильном применении технологии разработки информационных систем - они не могут быть нарушены. Это относится и к интеграции данных, и к независимости данных от прикладных программ, ровно как и к неизбыточности, непротиворечивости и связности данных.

```
import wikipedia
import docx

def naive(text, summary):
    checked = []
    for i in range(2, len(summary)):
        stroka = summary[i-2] + summary[i-1] + summary[i]
        if not(stroka in checked) and stroka in text:
            checked.append(stroka)
    return checked


def remover(text):
    symbols = ['.', ',', ':', '\n', '-', ')', '1', '2',
                    '3', '4', '5', '6', '7', '8', '9', '0',
                    ';', '(', '-', '«', '»', '—', '?', '=' , '==',
                    '  ', '–']
    for i in symbols:
        if i == "  ":
            text = text.replace(i, ' ')
        elif i == "\n":
            text = text.replace(i, "\n")
        else:
            text = text.replace(i, ' ')
    return text

def text_get(file_name):
    f = docx.Document(file_name)
    text = ""
    for i in f.paragraphs:
        text += i.text
    return text

wikipedia.set_lang("ru")
text = remover(wikipedia.page("Стресс").content).split()
summary = remover(text_get("Стресс.docx"))
summary_len = len(summary)
summary = summary.split()

text_triple = [text[i-2]+text[i-1]+text[i] for i in range(2, len(text))]

pl_len = len("".join(naive(text_triple, summary)))

print("В реферате присутствует примерно %.4f" % ((pl_len/summary_len)*100), "%
плагиата", sep="")
```