1. **Теоретические основы баз данных и анализа маршрутов**
   1. **Базы данных**

База данных – это представленная в объективной форме совокупность самостоятельных материалов, систематизированных так, чтобы эти материалы могли быть найдены и обработаны с помощью электронной вычислительной машины.

Система управления базами данных — совокупность программных и языковых средств, которые предназначенны для управления созданием и использованием баз данных.

Различают два класса СУБД, исходя из их универсальности — специализированные системы а также системы общего назначения.

СУБД общего назначения ориентированы на широкий круг задач, но при этом может быть не предназначенна для конкретной потребности пользователей. Данные системы разрабатываются, чтобы функционировать на большом числе ЭВМ и разных операционных системах. При этом данные базы данных обладают средствами настройки для работы в условиях конкретной операционной системы.

Возникают ситуации, когда СУБД общего назначения не позволяют добиться требуемых функциональных характеристик (модель данных, производительность, занимаемый объем памяти и т.д.). С другой стороны создание и использование специализированных СУБД весьма трудоемкий процесс и для его реализации, нужны очень веские основания.

**1.1.1 Функции СУБД**

Основными функциями баз данных являются:

Управление данными во внешней памяти

Данная функция предназначена для выполнения базовых операций, которые осуществляются с данными(создание, извлечение, обновление и удаление информации). Функция также включает в себя обеспечение сохранности не только данных, хранящихся в БД, но и для служебных целей (сохранение настороек БД).

Управление транзакциями

Транзакция — это последовательная группа операций над базой данных, которые рассматриваются СУБД как единое целое. Для транзакции характерно правило «либо все, либо ни чего». Транзакция состоитиз набора действий, которые выполняются для доступа или изменения содержимого базы данных. Самыми простыми транзакциями могутбыть добавление, обновление или удаление содержимого в базе данных. Сложная транзакция осуществляется когда в БД необходимо внести сразу несколько изменений. Создание транзакции происходит пользователем базы данных либо прикладной программой. Когда все действия, составляющие транзакцию успешно выполнены или возникла ошибка, транзакция должна быть завершена, для того, чтобы база данных находилась в непротиворечивом состоянии.

Восстановление базы данных

Данная функция СУБД, которая в случае логических и физических сбоев приводит базу данных в актуальное и консистентное состояние. Она является одной самых из важных требований предъявляемых к СУБД.Также часто данную функцию называют. Надежность хранения определяет, что у СУБД имеется возможность восстановить последнее согласованное состояние БД после сбоя, вызванного аппаратной или программной ошибкой. Выделяют два вида аппаратных сбоев: 

- мягкие сбои, при которых не происходит потеря информации, данные сбои возможно трактовать как внезапную остановку работы компьютера 

- жесткие сбои, в отличии от мягких сбоев происходит потеря информации во внешней памяти.

Для поддержания надежности хранения объектов в БД требуется избыточность хранения данных. При этом особое отношение получают данные, используемые для восстановления, и хранящиеся особо надежно. Для этого применяется журналирование, как наиболее распространный способ для хранения избыточной информации.

Поддержка языков БД

Для осуществеления работы с базами данных разработаны специальные языки, называемые языками запросов или языками баз данных.

В настоящее время СУБД обычно поддерживают стандартный интегрированный язык, который поддерживает основные операции для работы с базой данных, на всех этапах ее жизненого цикла, и которые обеспечивают интерфейс для работы с хранимыми данными. Наиболее распространенных языком для реляционных СУБД является язык SQL (Structured Query Language — язык структурированных запросов). SQL служит для определения схемы данных реляционной БД а также операциями управлениями данными.

Словарь данных

Одной из основополагающих идей рассмотренной выше трехуровневой архитектуры является наличие интегрированного системного каталога с данными о схемах, пользователях, приложениях и т. д. Системный каталог, который еще называют словарем данных, является, таким образом, хранилищем информации, описывающей данные в базе данных. Предполагается, что каталог доступен как пользователям, так и функциям СУБД. Обычно в словаре данных содержится следующая информация:

- имена, типы и размеры элементов данных;

- имена связей;

- накладываемые на данные ограничения поддержки целостности;

- имена пользователей, которым предоставлено право доступа к данным;

- внешняя, концептуальная и внутренняя схемы и отображения между ними;

- статистические данные, например частота транзакций и счетчики обращений к объектам базы данных.

Управление параллельным доступом

Одна из основных целей создания и использования СУБД заключается в том, чтобы множество пользователей могло осуществлять параллельный доступ к совместно обрабатываемым данным. Параллельный доступ сравнительно просто организовать, если все пользователи выполняют только чтение данных, поскольку в этом случае они не могут помешать друг другу. Однако когда два или больше пользователей одновременно получают доступ к базе данных, конфликт с нежелательными последствиями легко может возникнуть, например, если хотя бы один из них попытается обновить данные.

СУБД должна гарантировать, что при одновременном доступе к базе данных многих пользователей подобных конфликтов не произойдет.

Управление буферами оперативной памяти

СУБД обычно работают с БД значительного размера. Понятно, что если при обращении к любому элементу данных будет производиться обмен с внешней памятью, то вся система будет работать со скоростью устройства внешней памяти. Практически единственным способом реального увеличения этой скорости является буферизация данных в оперативной памяти. В развитых СУБД поддерживается собственный набор буферов оперативной памяти с собственной дисциплиной замены буферов.

Контроль доступа к данным

СУБД должна иметь механизм, гарантирующий возможность доступа к базе данных только санкционированных пользователей и защищающий ее от любого несанкционированного доступа. В современных СУБД поддерживается один из двух широко распространенных подходов к вопросу обеспечения безопасности данных: избирательный подход или обязательный подход. В большинстве современных систем предусматривается избирательный подход, при котором некий пользователь обладает различными правами при работе с разными объектами. Значительно реже применяется альтернативный, обязательный подход, где каждому объекту данных присваивается некоторый классификационный уровень, а каждый пользователь обладает некоторым уровнем допуска.

Поддержка целостности данных

Термин целостность используется для описания корректности и непротиворечивости хранимых в БД данных. Реализация поддержки целостности данных предполагает, что СУБД должна содержать сведения о тех правилах, которые нельзя нарушать при работе с данными, и обладать инструментами контроля за тем, чтобы данные и их изменения соответствовали заданным правилам.

1.1.2 Языки баз данных

В СУБД поддерживается несколько специализированных по своим функциям подъязыков. Их можно разбить на две категории: 

язык определения данных БД — ЯОД {DDL — Data Definition Language);

язык манипулирования данными— ЯМД (DML — Data Manipulation, Language).

Язык определения данных

Язык определения данных — описательный язык, с помощью которого описывается предметная область: именуются объекты, определяются их свойства и связи между объектами. Он используется главным образом для определения логической структуры БД.

Схема базы данных, выраженная в терминах специального языка определения данных, состоит из набора определений.

Язык ЯОД используется как для определения новой схемы, так и для модификации уже существующей. Результатом компиляции ЯОД — операторов является набор таблиц, хранимый в системном каталоге, в котором содержатся метаданные — т. е. данные, которые включают определения записей, элементов данных, а также другие объекты, представляющие интерес для пользователей или необходимые для работы СУБД. Перед доступом к реальным данным СУБД обычно обращается к системному каталогу.

Языки манипулирования данными

Язык манипулирования данными содержит набор операторов манипулирования данными, т. е. операторов, позволяющих заносить данные в БД, удалять, модифицировать или выбирать существующие данные.

Множество операций над данными можно классифицировать следующим образом:

1. операции селекции;

2. действия над данными:

включение — ввод экземпляра записи в БД с установкой его связей; 

удаление — исключение экземпляра записи из БД с установкой новых связей;

модификация — изменение содержимого экземпляра записи и коррекция связей при необходимости.

Языки манипулирования данными делятся на два типа. Это разделение обусловлено коренным различием в подходах к работе с данными, а следовательно, различием в базовых конструкциях в работе с данными.

Первый тип — это процедурный ЯМД.

Второй тип — это декларативный (непроцедурный) ЯМД.

К процедурным языкам манипулирования данными относятся и языки, поддерживающие операции реляционной алгебры, которую основоположник теории реляционных баз данных Э. Ф. Кодд ввел для управления реляционной базой данных. Реляционная алгебра — это процедурный язык обработки реляционных таблиц, где в качестве операндов выступают таблицы в целом.

Декларативные языки предоставляют пользователю средства, позволяющие указать лишь то, какие данные требуются. Решение вопроса о том, как их следует извлекать, берет на себя процессор данного языка, работающий с целыми наборами записей.

Реляционные СУБД обычно включают поддержку непроцедурных языков манипулирования данными — чаще всего это бывает язык структурированных запросов SQL или язык запросов по образцу QBE.

В настоящее время нормой является поддержка декларативного языка SQL, в основе которого лежит реляционное исчисление, также введенное Э Коддом. Этот язык стал стандартом для языков реляционных баз данных, что позволяет использовать один и тот же синтаксис и структуру команд при переходе от одной СУБД к другой.

Следует отметить, что язык SQL имеет сразу два компонента: язык DDL (ЯОД) для описания структуры базы данных, и язык DML (ЯМД) для выборки и обновления данных.

Другим широко используемым языком обработки данных является язык QBE, который заслужил репутацию одного из самых простых способов извлечения информации из базы данных. Особенно это ценно для пользователей, не являющихся профессионалами в этой области Язык предоставляет графические средства создания запросов на выборку данных с использованием шаблонов Ответ на запрос также представляет собой графическую информацию.

Часть непроцедурного языка ЯМД, которая отвечает за извлечение данных, называется языком запросов Язык запросов можно определить как высокоуровневый узкоспециализированный язык, предназначенный для удовлетворения различных требований по выборке информации из базы данных.

**1.2 Теоретические основы теории графов**

Графом G называется совокупность из некоторого (обычно конечного) множества V , элементы которого называются вершинами, и некоторого выделенного подмножества E множества V 2 пар элементов множества V (называемых ребрами). Обычно подразумевается, что пары вершин неупорядочены (граф неориентированный) и элементы в каждой паре различны (нет петель). Если рассматриваются упорядоченные пары, граф называется ориентированным.

1.2.1 Основные алгоритмы теори графов.

Для графов было разработано большое количество алгоритмов. В основном они рассчитаны на нахождение взаимосвязи между вершинами, нахождение кратчайших путей и другие. Далее представлены основные алготимы на графах.

Поиск в глубину

Поиск в глубину (англ. Depth-first search) — алгоритм обхода графа. Основой поиска в глубину состоит в том, чтобы идти насколько возможно «вглубь» графа. Данный алгоритм является рекурсивным: перебираются все исходящие из данной вершины рёбра. Если ребро ведёт в нерассмотренную ранее вершину,то запускается алгоритм от этой нерассмотренной вершины, а после возвращаемся и продолжаем перебирать рёбра. Возврат происходит в том случае, если в рассматриваемой вершине не осталось рёбер, которые ведут в нерассмотренную вершину. Если после завершения алгоритма не все вершины были рассмотрены, то необходимо запустить алгоритм от одной из нерассмотренных вершин.

Поиск в ширину

Поиск в ширину (англ. breadth-first search, BFS) –алгоритм обхода графа. BFS подразумевает поуровневое исследование графа: выбирается произвольно выбранный узел, который будет корнем он будет выбран в начале, затем происходит обход всех потомков данного узла, далее посещаются потомки потомков и т.д. Вершины анализируются в порядке удаления от корня их расстояния.

Пусть задан граф G=(V, E) и корень s, с которого начинается обход. После посещения узла s, следующими за ним будут посещены смежные с s узлы (множество смежных с s узлов обозначим как q; очевидно, что q⊆V, то есть q – некоторое подмножество V). Далее, эта процедура повториться для вершин смежных с вершинами из множества q, за исключением вершины s, т. к. она уже была посещена. Так, продолжая обходить уровень за уровнем, алгоритм обойдет все доступные из s вершины множества V. Алгоритм прекращает свою работу после обхода всех вершин графа, либо в случае выполнения наличествующего условия.

Алгоритм Дейкстры

Алгоритм изобретённ нидерландским учёным Эдсгером Дейкстрой в 1959 году. Алгоритм находит кратчайшие пути от одной из вершин графа до всех остальных. Чтобы его использовать необходимо, чтобы ребра были не отрицательного веса. Данный алгоритм является широко распространееным в программировании и технологиях, к примеру, алгоритм используют протоколы маршрутизации OSPF и IS-IS.

Каждой вершине из V сопоставим метку — минимальное известное расстояние от этой вершины до a. Алгоритм работает пошагово — на каждом шаге он «посещает» одну вершину и пытается уменьшать метки. Работа алгоритма завершается, когда все вершины посещены.

Инициализация. Метка самой вершины a полагается равной 0, метки остальных вершин — бесконечности. Это отражает то, что расстояния от a до других вершин пока неизвестны. Все вершины графа помечаются как непосещённые.

Шаг алгоритма. Если все вершины посещены, алгоритм завершается. В противном случае, из ещё не посещённых вершин выбирается вершина u, имеющая минимальную метку. Мы рассматриваем всевозможные маршруты, в которых u является предпоследним пунктом. Вершины, в которые ведут рёбра из u, назовём соседями этой вершины. Для каждого соседа вершины u, кроме отмеченных как посещённые, рассмотрим новую длину пути, равную сумме значений текущей метки u и длины ребра, соединяющего u с этим соседом. Если полученное значение длины меньше значения метки соседа, заменим значение метки полученным значением длины. Рассмотрев всех соседей, пометим вершину u как посещённую и повторим шаг алгоритма.

**1.3 Пространственный анализ**

Пространственный анализ – это произведение вычислительных операций над геоданными с целью извлечения из них дополнительной информации. Для пространственного анализа используются разные методы манипулирования пространственными и атрибутивными данными, которые выполняются при обработке пользовательских запросов. К таким методам относятся средства анализа сетевых структур либо выявления объектов по определенным заданным признакам.

Различные ГИС-пакеты используют свой набор средств для пространственного анализа, которые обеспечивают задач пользователя. При этом возможно выделить ряд основных функций, включанные в практически любой ГИС-пакет. К таким функцис относятся выбор и объединение объектов в соответствии с заранее заданными условиями, построение буферных зон, анализ наложений, операции вычислительной геометрии, сетевой анализ.

1.3.1 Функции пространственного анализа данных

Геометрические функции: здесь включены расчеты геометрических параметров объектов а также их взаимного расположения в пространстве, для этого используются формулы из теории графов и аналитической геометрии. Так для линейных объектов вычисляются их длина, для площадных объектов - занимаемые ими площади и расстояния между ними.

Выборка объектов по заданному запросу: двусторонняя операция, может использоваться как для выбора получения характеристик выбранного объекта, так и поиск объекта по заданным характеристикам. В более сложных примерах могут использоваться запросы на выбор объектов по нескольким признакам, таких как выбор объектов исходя расстояния между ними, расположенные на разных слоях, но совпадающие объекты и т. д.

Для выбора данных в соответствии с заданными условиями могут использована стандартные реляционные базы данных. Для выполнения более сложных запросов реализованы возможности использования при составлении запросов математических и статистических функций, а также географических операторов, позволяющих выбирать объекты на основании их взаимного расположения в пространстве (например, находится ли анализируемый объект внутри другого объекта или пересекается с ним).

Объединение данных может производиться по равенству значений определенного атрибута, в частности для зонирования территории.Другой способ группировки - объединение объектов одного тематического слоя в соответствии с их размещением внутри полигональных объектов других тематических слоев.

Построение буферных зон. Одним из средств анализа близости объектов является построение буферных зон. Буферные зоны - это районы (полигоны), граница которых отстоит на заданном расстоянии от границы исходного объекта. Границы таких зон вычисляются на основе анализа соответствующих атрибутивных характеристик. При этом ширина буферной зоны может быть как постоянной, так и переменной. Например, буферная зона вокруг источника электромагнитного излучения, будет иметь форму круга, а зона загрязнения от дымовой трубы завода с учетом розы ветров будет иметь форму близкую к эллипсу.

Для анализа пространственных сетей связных объектов применяется сетевой анализ. Он применяется в частности для анализа дорог, линий электропередач и т.д. Обычно сетевой анализ применяется если необходимо выявление ближайшего либо наиболее выгодного пути, составление маршрута по заданным точкам, выявление уровня нагрузки на сеть и т.д.

1.3.2 Анализ пространственного распределения объектов

Часто требуется знать не только площадь пространства, которые занимаемают объектам, но и пространственное расположение объектов, характеризующееся количеством объектов находящихся в заданной области, например, расположение дорог внутри города. В тех случаях когда требудется анализ распределения используется точечный анализ, для которого мерой является плотность. Чтобы найти плотность необходимо разделить анализируемые точки на занимаюмаю ими площадь. Другим метод анализа является оценка формы распредения анализируемых точек. Выделяют четыре возможные варианта точечных распределений: кластерном(если точки тесно сгруппированы), регулярном (если точки, которые разделены равными интервалами по всей области, расположены в узлах сетки), равномерном(если количество точек в каждой подобласти равно количеству в любой другой подобласти), случайном.

Для описания точечного распределения может применяться не только количествоо точек в пределах площади подобластей. Также применяется анализ отношений внутри пар точек. Для этого среди возможных пар близжайших точек производится определение среднего расстояния до ближайшей точки. Это позволяет произвести оценку разреженности точек в распределении.

Для оценки распределения линий используется плотность. Он применяется для выявления схожести разных географических областей, например сравнение гидрографической сети по густоте. Для оценки линий используются также возможные пересечения, расстояния между ними, связанность и направленность линий а также их ориентация.

При анализе распределения полигонов используется подход применяемый для анализа точек, с той лишь разницей, что выявляется отностильная доля площади полигона, а не само количество полигонов.