**Теоретические вопросы:**

1. **Понятие администратора БД. Основные обязанности администратора БД.**

Администратор базы данных (DBA) — это специалист, который

отвечает за установку, конфигурирование, управление, поддержку,

оптимизацию и безопасность баз данных. Он занимается поддержанием

стабильной работы баз данных, резервным копированием, восстановлением,

мониторингом производительности и разрешением инцидентов.

Перечислить обязанности администратора баз данных (из

должностной инструкции).

* Установка и настройка баз данных — обеспечение правильного развёртывания системы.
* Мониторинг производительности — контроль за работой баз данных и обеспечение оптимальной производительности.
* Резервное копирование и восстановление — создание и управление резервными копиями данных, а также восстановление в случае сбоя.
* Обеспечение безопасности данных — контроль доступа пользователей, управление ролями и правами доступа.
* Оптимизация запросов — анализ и улучшение SQL-запросов для повышения скорости работы баз данных.
* Обновление и патчинг ПО — установка обновлений баз данных для устранения уязвимостей и повышения производительности.
* Диагностика и устранение неисправностей — решение проблем с базами данных, мониторинг и анализ ошибок.
* Разработка и поддержка процедур резервного копирования и аварийного восстановления.
* Планирование и управление мощностями — оценка потребностей в ресурсах и оптимизация использования серверных мощностей.
* Документирование архитектуры баз данных и изменений.

1. **Основные утилиты администратора баз данных: возможности, достоинства и недостатки.**

1 **SQL Server Management Studio (SSMS)**

* **Описание**: Инструмент для управления Microsoft SQL Server.
* **Достоинства**: Удобный интерфейс, интеграция с SQL Server.
* **Недостатки**: Поддерживает только Microsoft SQL Server.

2 **Oracle SQL Developer**

* **Описание**: Среда разработки и администрирования для баз данных Oracle.
* **Достоинства**: Богатый функционал, поддержка PL/SQL.
* **Недостатки**: Заточен только под Oracle DB.

3 **pgAdmin**

* **Описание**: Инструмент для управления PostgreSQL.
* **Достоинства**: Бесплатный, поддержка большинства функций PostgreSQL.
* **Недостатки**: Может быть медленным при работе с большими БД.

4 **MySQL Workbench**

* **Описание**: Среда для проектирования, администрирования и мониторинга баз данных MySQL.
* **Достоинства**: Бесплатный, поддержка ER-диаграмм.
* **Недостатки**: Ограниченные возможности для крупных проектов.

5 **Toad for Oracle**

* **Описание**: Платформа для администрирования баз данных Oracle.
* **Достоинства**: Многофункциональность, глубокая интеграция с Oracle.
* **Недостатки**: Высокая стоимость лицензии.

6 **Navicat for MySQL**

* **Описание**: Коммерческое ПО для управления базами данных MySQL.
* **Достоинства**: Простота использования, поддержка облачных БД.
* **Недостатки**: Платное ПО.

7 **DBVisualizer**

* **Описание**: Универсальный инструмент для работы с разными СУБД.
* **Достоинства**: Поддержка многих СУБД (PostgreSQL, Oracle, MySQL и др.).
* **Недостатки**: Ограниченная бесплатная версия.

8 **DBeaver**

* **Описание**: Многофункциональный кросс-платформенный инструмент для работы с различными базами данных.
* **Достоинства**: Бесплатный, поддержка множества СУБД.
* **Недостатки**: Некоторые продвинутые функции доступны только в платной версии.

9 **Azure Data Studio**

* **Описание**: Средство для управления SQL Server и базами данных в облачной среде Azure.
* **Достоинства**: Поддержка SQL Server и интеграция с Azure.
* **Недостатки**: Не такой функциональный, как SSMS.

10 **HeidiSQL**

* **Описание**: Бесплатная программа для управления MySQL, MariaDB и PostgreSQL.
* **Достоинства**: Легковесный и быстрый.
* **Недостатки**: Поддержка только некоторых СУБД.

1. **Понятие схемы БД. Создание схемы с помощью SQL Server Management Studio.**

**Схема базы данных** — это структурное представление данных, которое описывает, как данные организованы в базе данных. Она включает таблицы, их поля, отношения между таблицами (внешние ключи), индексы и ограничения.

* **Открыть SSMS** и подключиться к серверу базы данных.
* В **Обозревателе объектов (Object Explorer)** правой кнопкой мыши кликнуть на базе данных, в которой нужно создать схему, и выбрать **"Новая схема"**.
* В открывшемся окне ввести имя схемы и владелец.
* Нажать **"OK"** для сохранения новой схемы.
* Для создания таблиц и объектов внутри схемы можно использовать **визуальный дизайнер таблиц**:
* Правой кнопкой мыши кликнуть на схеме и выбрать **"Новая таблица"**.
* Добавить необходимые столбцы, указать типы данных, задать ключи и индексы.

1. **Понятие схемы БД. Создание схемы с помощью Transact SQL.**

**Схема базы данных** — это структурное представление данных, которое описывает, как данные организованы в базе данных. Она включает таблицы, их поля, отношения между таблицами (внешние ключи), индексы и ограничения.

**Использование T-SQL для создания схемы:**

-- Создание схемы с именем "HR" и назначением владельца

CREATE SCHEMA HR

AUTHORIZATION dbo;

1. **Понятие модели данных. Характеристики логических моделей: иерархическая, сетевая, реляционная, модель «сущность-связь», объектно-ориентированная, объектно-реляционная, их достоинства и недостатки.**

Модель данных — это формальная структура, которая описывает, как данные организованы, хранятся и обрабатываются в системе управления базами данных (СУБД). Она задаёт правила для представления данных, их типов, связей между ними и возможных операций. Модель данных определяет как логическое представление данных, так и их физическую организацию в базе данных.

3.1 Иерархическая модель

**Описание**: Структура данных организована в виде древовидной иерархии. Каждый узел (родитель) может иметь несколько потомков, но каждый узел-потомок имеет только одного родителя.

**Достоинства**:

* Простота структуры для однотипных данных.
* Хорошо подходит для представления иерархий (например, структура организаций).

**Недостатки**:

* Жёсткость структуры, сложность изменения.
* Плохая поддержка сложных связей между элементами данных.

3.2 Сетевая модель

**Описание**: Более гибкая, чем иерархическая модель. Узел может иметь несколько родителей, что позволяет создавать сложные связи между данными.

**Достоинства**:

* Более гибкая структура, чем иерархическая модель.
* Позволяет эффективно представлять сложные связи.

**Недостатки**:

* Сложная в реализации и поддержке.
* Требует детального понимания структуры для работы с данными.

3.3 Реляционная модель

**Описание**: Данные организованы в виде таблиц (отношений), где каждая строка представляет собой запись, а столбец — атрибут. Взаимосвязи между таблицами строятся через ключи.

**Достоинства**:

* Логическая простота и ясность.
* Мощные возможности для работы с данными через SQL.
* Поддержка нормализации, которая минимизирует дублирование данных.

**Недостатки**:

* Производительность может страдать при сложных запросах и больших объёмах данных.
* Ограниченная поддержка не реляционных типов данных (например, мультимедийные данные).

3.4 Модель "сущность-связь" (ER-модель)

**Описание**: Модель, которая описывает данные в виде сущностей и их связей. Сущность — это объект (например, пользователь), а связь — это отношения между сущностями.

**Достоинства**:

* Чёткое визуальное представление структуры данных.
* Легко преобразуется в реляционную модель.

**Недостатки**:

* Не подходит для описания сложных процессов и динамических данных.

3.5 Объектно-ориентированная модель

**Описание**: Данные описываются в виде объектов, которые включают в себя как данные (атрибуты), так и методы для работы с ними. Объекты могут наследовать свойства других объектов (наследование).

**Достоинства**:

* Поддержка сложных типов данных (например, мультимедийных).
* Прямое соответствие между объектами программирования и базами данных.

**Недостатки**:

* Сложность в реализации.
* Меньше инструментов и ресурсов по сравнению с реляционной моделью.

3.6 Объектно-реляционная модель

**Описание**: Гибридная модель, которая объединяет элементы объектно-ориентированной и реляционной моделей. Она поддерживает как объекты, так и отношения между ними.

**Достоинства**:

* Возможность хранения сложных данных с объектами в рамках реляционной структуры.
* Поддержка SQL с расширениями для работы с объектами.

**Недостатки**:

* Повышенная сложность реализации и производительности.
* Ограниченная поддержка в некоторых СУБД.

1. **Типы данных SQL.**

SQL поддерживает множество типов данных для хранения и обработки информации. Основные типы данных:

1. **Числовые типы**:
   * **INT**: целые числа.
   * **FLOAT, DOUBLE**: числа с плавающей точкой.
   * **DECIMAL**: точные числа с фиксированной точкой.
2. **Строковые типы**:
   * **CHAR(n)**: строка фиксированной длины.
   * **VARCHAR(n)**: строка переменной длины.

* **TEXT**: строки переменной длины, для хранения больших текстов.

1. **Дата и время**:
   * **DATE**: дата (год, месяц, день).
   * **TIME**: время (часы, минуты, секунды).
   * **DATETIME**: дата и время вместе.
2. **Логические типы**:
   * **BOOLEAN**: логический тип (TRUE или FALSE).
3. **Бинарные типы**:
   * **BINARY**: данные в двоичной форме фиксированной длины.
   * **VARBINARY**: данные в двоичной форме переменной длины.
4. **Типы для больших объектов**:
   * **BLOB**: двоичные данные большого объёма (например, изображения, видео).
   * **CLOB**: текстовые данные большого объёма.
5. **UUID**:
   * **UUID**: универсальный уникальный идентификатор для уникальной идентификации записей.
6. **JSON**:
   * **JSON**: хранение данных в формате JSON, поддержка для неструктурированных данных.
7. **Понятие журнала ошибок SQL Server. Проблемы, решаемые с помощью журналов SQL Server. Как открыть журналы с использованием SSMS.**

Журналы SQL Server известны как SQL Server Error logs. Журналы ошибок содержат информационные сообщения, предупреждения и сообщения о критичных ошибках. Вы можете просматривать некоторые из этих журналов также в просмотрщике событий Windows. Однако рекомендуется использовать журналы SQL Server для получения подробной информации

**Проблемы, решаемые с помощью журналов SQL Server.**

Журналы являются лучшим средством администратора баз данных при решении любых проблем. Эти проблемы могут быть связаны с конфигурацией сервера, запуском, восстановлением, производительностью, флагами трассировки, тупиковыми ситуациями, вводом-выводом или задержками.

**Как открыть журналы с использованием SSMS.**

Если вы подключены к экземпляру SQL Server в SSMS, перейдите к Management -> SQL Server Logs. Как показано ниже, имеется текущий журнал и шесть архивных журналов (Archive#1 - Archive #6).

1. **Понятие журнала ошибок SQL Server. Использование функции SERVERPROPERTY(). Использование менеджера конфигурации SQL Server.**

Журналы SQL Server известны как SQL Server Error logs. Журналы ошибок содержат информационные сообщения, предупреждения и сообщения о критичных ошибках. Вы можете просматривать некоторые из этих журналов также в просмотрщике событий Windows. Однако рекомендуется использовать журналы SQL Server для получения подробной информации

Мы можем использовать в запросе функцию SERVERPROPERTY, и также определить местонахождение SQL Server ERRORLOG.

SELECT SERVERPROPERTY('ErrorLogFileName') AS 'Error log location'

**Использование менеджера конфигурации SQL Server.**

Откройте SQL Server Configuration Manager и посмотрите параметры запуска. Местоположение файлов журнала указывается с помощью переключателя -e.

Вы можете развернуть каталог журналов и просмотреть текущий или архивные файлы журнала. Эти журналы ошибок можно открыть в текстовом редакторе, таком как Notepad или Visual Studio Code.

1. **Понятие журнала ошибок SQL Server. Конфигурирование числа файлов журнала SQL Server и их размеров (с использованием SSMS и скрипта T-SQL).**

Журналы SQL Server известны как SQL Server Error logs. Журналы ошибок содержат информационные сообщения, предупреждения и сообщения о критичных ошибках. Вы можете просматривать некоторые из этих журналов также в просмотрщике событий Windows. Однако рекомендуется использовать журналы SQL Server для получения подробной информации

**Конфигурирование числа файлов журнала SQL Server и их размеров (использованием SSMS и скрипта T-SQL).**

По умолчанию SQL Server поддерживает текущий и 6 архивных файлов журнала. Чтобы уточнить значение, выполните щелчок правой кнопкой на папке SQL Server Logs в SSMS и выберите Configure.

SQL Server записывает всю информацию в текущий файл журнала, независимо от размера файла журнала. В загруженной системе или в экземпляре с большим количеством ошибок вам может быть сложно просмотреть файл журнала в SSMS. SQL Server создает новый файл журнала и архивирует текущий файл в следующих случаях.

* При перезапуске службы SQL.
* При перезагрузке журнала ошибок вручную.

Однако если вы часто перезапускаете серверы по неизвестным причинам, то можете потерять все исторические данные в архивных журналах, поскольку их поддерживается только шесть. Поскольку ошибки содержат ценную информацию, которая может помочь в решении проблем, вы можете не захотеть потерять эти важные данные. Тогда, возможно, вы захотите сохранять файлы журнала производственной системы в течение недели или даже месяца.

SQL Server позволяет сконфигурировать от 6 до 99 файлов журнала ошибок. Вы не можете указать значение меньше шести, поскольку в любом случае будет поддерживаться шесть архивных журналов ошибок.

Для изменения значения по умолчанию числа файлов журнала ошибок поставьте галочку в поле с названием “Limit the number of error log files before they are recycled”. Например, следующий скриншот показывает максимальное число файлов журнала ошибок, равное 30.

Это эквивалентно выполнению скрипта T-SQL, который использует расширенную хранимую процедуру и обновляет значение регистра.

USE [master]

GO

EXEC xp\_instance\_regwrite N'HKEY\_LOCAL\_MACHINE', N'Software\Microsoft\MSSQLServer\MSSQLServer', N'NumErrorLogs', REG\_DWORD, 30

GO

Замечание. Следует перезапустить службу SQL, чтобы изменения вступили в силу.

Как утверждалось ранее, по умолчанию размер журнала ошибок не ограничен. Например, если вы не запускаете SQL Server в течение длительного периода и вручную не перегружаете файлы журнала, этот файл вырастет до громадных размеров. Поэтому в конфигурации журнала ошибок показано значение 0, соответствующее неограниченному размеру журнала.

Вы можете задать размер в Кб для ограничения размера журнала ошибок в соответствии с вашими требованиями. Например, здесь мы ограничиваем размер файла журнала в 1Гб.

Эквивалентный скрипт T-SQL обновляет ErrorLogSizeInKb в регистре SQL Server.

USE [master]

GO

EXEC xp\_instance\_regwrite N'HKEY\_LOCAL\_MACHINE', N'Software\Microsoft\MSSQLServer\MSSQLServer', N'ErrorLogSizeInKb', REG\_DWORD, 1048576

GO

1. **Понятие и назначение словаря данных. Понятия интегрированного и независимого словарей данных.**

Словарь данных — это централизованное хранилище метаданных, которое описывает структуры данных, их характеристики, отношения между ними и правила использования. Он обеспечивает систематизированное описание всех объектов данных в информационной системе, а также помогает пользователям и разработчикам правильно работать с этими данными.

**Назначение словаря данныx.**

Одно из главных назначений словаря данных состоит в документировании данных. Так как база данных обслуживает многих пользователей, крайне необходимо, чтобы они правильно понимали, что представляют собой данные.

Два важнейших назначения словаря данных состоят в централизованном ведении и управлении данными как ресурсом на всех этапах проектирования, реализации и эксплуатации системы, а также в обеспечении эффективного взаимодействия между всеми участниками проекта.

**Понятия интегрированного и независимого словарей данныx.**

Интегрированный словарь данных представляет собой часть пакета программ СУБД, а независимый словарь данных – отдельный пакет программ, служащий дополнением к СУБД.

1. **Интерфейсы словаря данных. Требования к содержанию словаря данных. Условия применения словаря данных.**

**Интерфейсы словаря данных.**

Существуют два типа интерфейсов:

1. с людьми, вовлеченными в систему АБД, системными программистами, прикладными программистами, руководителями, конечными пользователями и ревизорами;
2. с программными средствами: СУБД, компиляторами, операционной системой и генераторами отчетов.

**Требования к содержанию словаря данныx.**

Приведем перечень требований к описанию данных, которым должен удовлетворять словарь данных. Отметим, что эти требования выполняются современными пакетами программ словаря данных не обязательно в полном объеме.

Концептуальная модель.В процессе разработки концептуальной модели необходимы сведения об объектах, о представляющих их элементах данных или атрибутах и о взаимосвязях элементов. В словаре данных должна накапливаться информация о текущем или планируемом использовании данных отдельными подразделениями и отдельными пользователями, о частотных характеристиках обращения к данным. Нужны также текстуальные описания назначения и использования данных. Элементы данных, объекты и взаимосвязи следует снабжать соответствующими метками с указанием версии, статуса (планируемый, испытываемый, действующий и др.),текстуального описания, синонимов (других объектов словаря с отличной меткой, но имеющих тот же смысл), членства (составной частью каких объектов является данный элемент или какие объекты на него ссылаются), группирования элементов данных с выделением ключевых элементов.

Логическая модель.По логической модели базы данных в словаре данных должна храниться следующая информация о группировании элементов данных с указанием ключевых элементов (возможно, выделяемые группы будут подмножествами групп, специфицированных в концептуальной модели), об используемой модели данных, о взаимосвязях групп данных в рамках применяемой модели, о внешних моделях, поддерживаемых логической моделью (различные логические пути доступа к данным), о логических транзакциях, программах и модулях. Кроме того, должны храниться сведения о связях между логическими транзакциями, программами и модулями, а также таблицы соответствия между транзакциями, программами и модулями, с одной стороны, и подразделениями и функциями, которые они обслуживают, – с другой. Для программ и транзакций необходимо специфицировать язык программирования и тип (пакетная, используемая в режиме телеобработки и т.д.).

Внутренняя модель. В словаре данных должна храниться информация о физическом представлении данных. Сюда относятся длина (в символах), тип представления (битовая или символьная строка, упакованное десятичное число, число с плавающей точкой), точность (для числовых данных), выравнивание (влево, вправо), шаблон (только для вывода данных), правила проверки значения (константа, диапазон значений), алгоритм вывода (для вычисляемых данных), местоположение (последовательная позиция, на которой размещается элемент данных внутри агрегата данных), защита (код защиты для чтения и обновления), носитель (диск, лента, экран терминала), устройства, на которых размещается база данных, информация, управляющая разграничением доступа.

Идеальный словарь данных должен быть неотъемлемой составной частью всей системы обработки данных. За ввод данных в словарь ответственность несет АБД. Поскольку словарь данных является центральным звеном системы, необходимо постоянно поддерживать его копию, которая может использоваться для восстановления словаря после возникновения отказа всей системы и в случае непреднамеренного разрушения его рабочей версии. За сохранность словаря данных как жизненно важной части системы с базой данных полностью отвечает администрация базы данных.

Кроме того, в словарь данных необходимо включать информацию, позволяющую прослеживать влияние изменений в структурах данных на прикладные программы и транзакции. Желательно создать возможность предварительных (стоимостных) оценок последствий планируемых изменений.

Сбор данных для ввода в словарь. В идеальном случае словарь данных является тем инструментом, который совместно используется АБД, прикладными программистами, пользователями, администрацией и любыми программными средствами, взаимодействующими с базой данных.

Информация, управляющая разграничением доступа.В словаре данных может храниться и информация по разграничению доступа, определяющая, какие именно пользователи и каким образом могут обращаться к той или иной части базы данных. Эта информация может служить следующим целям:

* перед компиляцией программ или выдачей отчетов она позволяет проверить, нет ли противоречий между спецификациями выборки данных, заданными осуществляющим доступ лицом, и заданными АБД спецификациями;
* для предотвращения неправильного использования данных из некоторой части базы данных, совместно обрабатываемой. Многими пользователями на основе этой информации должны быть введены строгие ограничения безопасности данных.

**Условия применения словаря данныx.**

При внедрении словаря данных очень важно принять правильное решение и в нужный момент, так как неверный шаг здесь может привести к неудаче всего проекта базы данных. В то же время не следует думать, что существуют готовые решения, которые подходят для всех предприятий без исключения. Как и другие планы в системе обработки данных, план реализации словаря зависит от особенностей предметной области.

Подчас не только реализация всеобъемлющего словаря данных, но и создание интегрированной базы данных всей предметной области может оказаться преждевременным. Ведение словаря данных предполагает обучение пользователей, и до тех пор, пока в этом направлении не будут достигнуты определенные успехи, его установку производить не следует.

В качестве первого приложения словаря данных можно избрать один из следующих вариантов.

Традиционная прикладная система. Словарь данных может применяться в прикладной системе средних размеров, не слишком тесно взаимодействующей с другими системами. Кроме того, система должна решать важные для предприятия задачи. И наконец, она должна быть развивающейся. Только при этих условиях смогут быть продемонстрированы все преимущества использования словаря данных.

Несколько применений СУБД. Скорее всего эксплуатируемые с использованием СУБД прикладные программы имеют большое значение для предприятия и, кроме того, со временем они претерпевают определенные изменения. Если предполагается расширить круг прикладных программ, то внедрение системы словаря данных, по-видимому, окажется своевременным, так как это позволит показать его возможности по обеспечению взаимодействия между пользователями, в области централизованного накопления информации о прикладных программах, а также при подготовке и распространении документации.

Новые прикладные программы. Если предприятие планирует расширить круг прикладных программ или создать базу данных, то это самый подходящий случай для внедрения словаря данных. Программисты, аналитики и пользователи не всегда приветствуют необходимость документирования уже существующих прикладных программ. Однако при разработке новых прикладных систем эта работа выполняется впервые, и документирование новых приложений неизбежно.

Главным стимулом и критерием перехода к внедрению словаря данных являются те удобства и преимущества, которые оно предоставит руководству, пользователям, аналитикам, программистам, обслуживающему персоналу, операторам и т. д. СУБД обеспечивает работу оперативных систем, что выигрышно для руководства. Обычно этого нельзя сказать о словаре данных, так как он служит важным средством внутреннего управления и контроля. Поэтому при принятии решения о введении словаря и выборе первых реализуемых с его помощью приложений следует тщательно взвесить все обстоятельства.

Необходимо прежде всего определить назначение словаря данных. Чрезвычайно важно заручиться согласием администрации предприятия, иначе весь проект информационной системы может оказаться под угрозой. Затем группой разработчиков должны быть исследованы следующие характеристики имеющихся систем словарей данных: простота использования, простота внесения изменений, сопровождение поставщиком, мощность, расширяемость функций и стоимость. Эта группа должна также сопоставить специфические потребности предметной области и возможности, предоставляемые рассматриваемыми системами. Необходимо разработать стратегию и план сбора данных и начать обучение сотрудников. Сбор и анализ информации и будет первым этапом в создании словаря данных.

Лучше всего приурочить внедрение словаря к моменту определения требований к системе. При этом пользователи смогут помочь разработчикам глубже понять систему. Как уже отмечалось выше, здесь может сыграть свою положительную роль тот психологический фактор, что никто из разработчиков и пользователей еще не затратил усилий на сбор и анализ исходных данных. Использование словаря данных при определении требований к системе создаст прочную основу для управления и контроля при реализации проекта, а также при осуществлении взаимодействия между разработчиками и пользователями.

Наибольший эффект словарь данных может дать при определении требований к новой прикладной системе, при значительном перепроектировании существующей системы или при интеграции двух и более систем.

1. **Понятие распределенной базы данных, распределенной СУБД. Примеры распределенных СУБД. Характеристики распределенной СУБД.**

Распределенная база данных –набор логически связанных между собой разделяемых данных (и их описаний), которые физически распределены в некоторой компьютерной сети.

Распределенная СУБД –программный комплекс, предназначенный для управления распределенными базами данных и позволяющий сделать распределенность информации прозрачной для конечного пользователя.

Пример.

Пользователь желает определить, какое количество выпускников общеобразовательных учреждений написало ЕГЭ по математике не менее чем на 90 баллов. Для этого он должен сделать запрос к распределенной базе данных, островами информации в которой являются образовательные сайты субъектов РФ. Но пользователь производит запрос к базе данных таким образом, будто она централизованная, и результаты запроса ему выдаются таким образом, как будто работа идет с централизованной СУБД.

**Характеристики распределенной СУБД.**

* Базы данных в коллекции логически взаимосвязаны друг с другом. Часто они представляют собой единую логическую базу данных.
* Данные физически хранятся на нескольких сайтах. Данными на каждом сайте может управлять СУБД независимо от других сайтов.
* Процессоры на сайтах подключены через сеть. Они не имеют никакой многопроцессорной конфигурации.
* Распределенная база данных - это не слабо связанная файловая система.
* Распределенная база данных включает в себя обработку транзакций, но это не синоним системы обработки транзакций.

1. **Понятие распределенной базы данных, распределенной СУБД. Достоинства и недостатки распределенных СУБД. Виды и функции, распределенных СУБД.**

Распределенная база данных –набор логически связанных между собой разделяемых данных (и их описаний), которые физически распределены в некоторой компьютерной сети.

Распределенная СУБД –программный комплекс, предназначенный для управления распределенными базами данных и позволяющий сделать распределенность информации прозрачной для конечного пользователя.

**Преимущества распределенных СУБД.**

Отражение структуры организации

Крупные организации, как правило, имеют множество отделений, которые могут находиться в разных концах страны и даже за ее пределами. Вполне логично будет предположить, что используемые этими организациями базы данных должны быть распределены между отдельными офисами. В каждом отделении может поддерживаться своя база данных. В подобной базе данных персонал отделения сможет выполнять необходимые ему локальные запросы. Руководству компании может потребоваться выполнять глобальные запросы, предусматривающие получение доступа к данным, сохраняемым во всех существующих отделениях компании.

Разделяемостъ и локальная автономность

Географическая распределенность организации может быть отражена в распределении ее данных, причем пользователи одного сайта смогут получать доступ к данным, сохраняемым на других сайтах. Данные могут быть помещены на тот сайт, на котором зарегистрированы пользователи, которые их чаще всего используют. В результате заинтересованные пользователи получают локальный контроль над требуемыми им данными и могут устанавливать или регулировать локальные ограничения на их использование. Администратор глобальной базы данных (АБД) отвечает за систему в целом. Как правило, часть этой ответственности делегируется на локальный уровень, благодаря чему АБД локального уровня получает возможность управлять локальной СУБД.

Повышение доступности данных

В централизованных СУБД отказ центрального компьютера вызывает прекращение функционирования всей СУБД. Однако отказ одного из сайтов СУРБД или линии связи между сайтами сделает недоступным лишь некоторые сайты, тогда как вся система в целом сохранит свою работоспособность. Распределенные СУБД проектируются таким образом, чтобы обеспечивать продолжение функционирования системы, несмотря на подобные отказы. Если выходит из строя один из узлов, система сможет перенаправить запросы к отказавшему узлу в адрес другого сайта.

Повышение надежности

Если организована репликация данных, в результате чего данные и их копии будут размещены на более чем одном сайте, отказ отдельного узла или соединительной связи между узлами не приведет к недоступности данных в системе.

Повышение производительности

Если данные размещены на самом нагруженном сайте, который унаследовал от систем-предшественников высокий уровень параллельности обработки, то развертывание распределенной СУБД может способствовать повышению скорости доступа к базе данных (по сравнению с доступом к удаленной централизованной СУБД). Более того, поскольку каждый сайт работает только с частью базы данных, уровень использования центрального процессора и служб ввода/ вывода может оказаться ниже, чем в случае централизованной СУБД.

Экономические выгоды

В шестидесятые годы мощность вычислительной установки возрастала пропорционально квадрату стоимости ее оборудования, поэтому система, стоимость которой была втрое выше стоимости данной, превосходила ее по мощности в девять раз. Эта зависимость получила название закона Гроша [7]. Однако в настоящее время считается общепринятым положение, согласно которому намного дешевле собрать из небольших компьютеров систему, мощность которой будет эквивалентна мощности одного большого компьютера. Оказывается, что намного выгоднее устанавливать в подразделениях организации собственные маломощные компьютеры, кроме того, гораздо дешевле добавить в сеть новые рабочие станции, чем модернизировать систему с мейнфреймом.

Второй потенциальный источник экономии имеет место в том случае, когда базы данных географически удалены друг от друга и приложения требуют осуществления доступа к распределенным данным. В этом случае из-за относительно высокой стоимости передаваемых по сети данных (по сравнению со стоимостью их локальной обработки) может оказаться экономически выгодным разделить приложение на соответствующие части и выполнять необходимую обработку на каждом из сайтов локально.

Модульность системы.

В распределенной среде расширение существующей системы осуществляется намного проще. Добавление в сеть нового сайта не оказывает влияния на функционирование уже существующих. Подобная гибкость позволяет организации легко расширяться. Перегрузки из-за увеличения размера базы данных обычно устраняются путем добавления в сеть новых вычислительных мощностей и устройств дисковой памяти. В централизованных СУБД рост размера базы данных может потребовать замены и оборудования (более мощной системой), и используемого программного обеспечения (более мощной или более гибкой СУБД).

**4. Недостатки распределенных СУБД.**

Повышение сложности

Распределенные СУБД, способные скрыть от конечных пользователей распределенную природу используемых ими данных и обеспечить необходимый уровень производительности, надежности и доступности, безусловно, являются более сложными программными комплексами, чем централизованные СУБД. Тот факт, что данные могут подвергаться репликации, также добавляет дополнительный уровень сложности в программное обеспечение СУРБД. Если репликация данных не будет поддерживаться на требуемом уровне, система будет иметь более низкий уровень доступности данных, надежности и производительности, чем централизованные системы, а все изложенные выше преимущества превратятся в недостатки.

Увеличение стоимости

Увеличение сложности означает и увеличение затрат на приобретение и сопровождение СУРБД (по сравнению с обычными централизованными СУБД). Разворачивание распределенной СУБД потребует дополнительного оборудования, необходимого для установки сетевых соединений между сайтами. Следует ожидать и роста расходов на оплату каналов связи, вызванных возрастанием сетевого трафика. Кроме того, возрастают затраты на оплату труда персонала, который потребуется для обслуживания локальных СУБД и сетевых соединений.

Проблемы защиты

В централизованных системах доступ к данным легко контролируется. Однако в распределенных системах потребуется организовать контроль доступа не только к данным, реплицируемым на несколько различных сайтов, но и защиту сетевых соединений самих по себе. Раньше сети рассматривались как совершенно незащищенные каналы связи. Хотя это отчасти справедливо и для настоящего времени, тем не менее, в отношении защиты сетевых соединений достигнут весьма существенный прогресс.

Усложнение контроля за целостностью данных

Целостность базы данных означает корректность и согласованность сохраняемых в ней данных. Требования обеспечения целостности обычно формулируются в виде некоторых ограничений, выполнение которых будет гарантировать защиту информации в базе данных от разрушения. Реализация ограничений поддержки целостности обычно требует доступа к большому количеству данных, используемых при выполнении проверок, но не требует выполнения операций обновления. В распределенных СУБД повышенная стоимость передачи и обработки данных может препятствовать организации эффективной защиты от нарушений целостности данных.

Отсутствие стандартов

Хотя функционирование распределенных СУБД зависит от эффективности используемых каналов связи, только в последнее время стали вырисовываться контуры стандарта на каналы связи и протоколы доступа к данным. Отсутствие стандартов существенно ограничивает потенциальные возможности распределенных СУБД. Кроме того, не существует инструментальных средств и методологий, способных помочь пользователям в преобразовании централизованных систем в распределенные.

Недостаток опыта

В настоящее время в эксплуатации находится уже несколько систем-прототипов и распределенных СУБД специального назначения, что позволило уточнить требования к используемым протоколам и установить круг основных проблем. Однако на текущий момент распределенные системы общего назначения еще не получили широкого распространения. Соответственно, еще не накоплен необходимый опыт промышленной эксплуатации распределенных систем, сравнимый с опытом эксплуатации централизованных систем. Такое положение дел является серьезным сдерживающим фактором для многих потенциальных сторонников данной технологии.

Усложнение процедуры разработки базы данных

Разработка распределенных баз данных, помимо обычных трудностей, связанных с процессом проектирования централизованных баз данных, требует принятия решения о фрагментации данных, распределении фрагментов по отдельным сайтам и организации процедур репликации данных.

**Виды распределенных СУБД.**

Однородный: В этом типе DDBMS все участвующие сайты должны иметь одинаковое программное обеспечение и архитектуру СУБД, что обеспечивает согласованность всех базовых систем на всех сайтах. Обеспечивает упрощенный обмен данными и интеграцию.

Гетерогенный: В этом типе DDBMS участвующие сайты могут принадлежать нескольким сайтам и использовать различное программное обеспечение СУБД, модели данных или архитектуры. Эта модель сталкивается с небольшой проблемой интеграции, поскольку представление данных всего сайта и язык запросов могут отличаться друг от друга.

Федеративные: Здесь локальные базы данных поддерживаются отдельными сайтами или федерациями. Эти локальные базы данных подключены через систему промежуточного программного обеспечения, которая позволяет пользователям получать доступ к данным из нескольких распределенных баз данных и запрашивать их. Федерация объединяет различные локальные базы данных, но сохраняет автономность на местном уровне.

Реплицируемая: В этом типе DDBMS поддерживает несколько копий одного и того же фрагмента данных на разных сайтах. Используется для обеспечения доступности данных, отказоустойчивости и бесперебойной работы. Пользователи могут получить доступ к любым данным из ближайшей реплики, если root по какой-либо причине отключен. Однако требуется выполнить высококачественную синхронизацию изменений данных при репликации.

Секционированный: В секционированной DDBMS вся база данных разделена на отдельные разделы, и каждый раздел назначен определенному сайту. Разделение может выполняться в зависимости от конкретных условий, таких как диапазон дат, географическое местоположение и функциональные модули. Каждый сайт управляет своим собственным разделом, и доступ к данным из других разделов должен осуществляться посредством связи и координации между сайтами.

Гибрид: Это просто комбинация множества других пяти типов DDBMS , которые обсуждались выше. Комбинация разработана для удовлетворения конкретных требований и задач сложных распределенных сред. Гибридные СУБД обеспечивают более оптимизированную производительность и высокую масштабируемость.

**6. Функции распределенных СУБД.**

СУРБД должна предоставлять следующий набор функциональных возможностей**.**

* Расширенные службы установки соединений должны обеспечивать доступ к удаленным сайтам и позволять передавать запросы и данные между сайтами, входящими в сеть.
* Расширенные средства ведения каталога, позволяющие сохранять сведения о распределении данных в сети.
* Средства обработки распределенных запросов, включая механизмы оптимизации запросов и организации удаленного доступа.
* Расширенные функции управления параллельностью, позволяющие поддерживать целостность реплицируемых данных.
* Расширенные функции восстановления, учитывающие возможность отказов в работе отдельных сайтов и отказов линий связи.

1. **Правила Дейта, которым должна соответствовать распределенная СУБД.**
2. Локальная автономия или независимость от локального сайта – Каждый сайт работает самостоятельно, а также действует как независимая автономная централизованная СУБД. За безопасность, контроль параллелизма, резервное копирование и восстановление отвечает каждый сайт.
3. Независимость от центрального сайта – Все сайты равны, и ни один сайт не зависит от центрального сайта в выполнении каких-либо услуг. Мы можем сказать, что нет такого сайта, который не мог бы, какая система не могла бы работать. Есть некоторые службы, для которых центральный сервер не требуется, это управление транзакциями, оптимизация запросов, обнаружение взаимоблокировок и управление глобальным системным каталогом.
4. Непрерывная работа – Сбой сайта никак не влияет на систему. Система продолжает свою работу даже в случае сбоя сайта или любого расширения сети.
5. Локальная независимость – Чтобы получить какие-либо данные в системе, необходимо знать о хранилище данных, то есть о том, где данные хранятся в системе.
6. Независимость от фрагментации – Пользователь может видеть только одну логическую базу данных. Фрагментация данных прозрачна для пользователя. Для извлечения любых фрагментов базы данных нет необходимости знать названия фрагментов базы данных.
7. Независимость от репликации – Данные могут реплицироваться и храниться на разных сайтах. DDBMS управляет всеми фрагментами прозрачно для пользователя.
8. Независимость от распределенных запросов – Выполнение одного запроса в другом месте не может удовлетворить прозрачный запрос. Таким образом, оптимизация запросов имеет решающее значение и выполняется DDBMS прозрачно.
9. Независимость от распределенных транзакций – Транзакция способна прозрачно обновлять данные на разных сайтах, но контроль восстановления и параллелизм достигаются с помощью агентов.
10. Аппаратная независимость – У DDBM должна быть возможность работать на разных аппаратных платформах.
11. Независимость от операционной системы – У DDBM должна быть возможность работать на разных платформах операционных систем.
12. Независимость от сети – Система DDBMS способна работать на любой сетевой платформе.
13. Независимость от базы данных – Система должна поддерживать продукт любого поставщика баз данных.
14. **Понятие сервера, сервера БД. Классификация серверов.**

Сервер — это компьютер или программное обеспечение, которое предназначено для обработки и хранения данных, а также для предоставления доступа к этим данным через сеть.

Серверы используются в различных областях, включая бизнес, науку, образование, медиа, развлечения и технологии. Они могут быть физическими устройствами, которые установлены в офисе или в дата-центре, или виртуальными машинами, которые запускаются на удаленном сервере.

Термин "сервер баз данных" обычно используют для обозначения всей СУБД, основанной на архитектуре "клиент-сервер", включая и серверную, и клиентскую части. Такие системы предназначены для хранения и обеспечения доступа к базам данных.

Хотя обычно одна база данных целиком хранится в одном узле сети и поддерживается одним сервером, серверы баз данных представляют собой простое и дешевое приближение к распределенным базам данных, поскольку общая база данных доступна для всех пользователей локальной сети.

**Классификация серверов.**

Серверы могут быть классифицированы по назначению на различные типы. Рассмотрим основные типы серверов:

1. Сервер рабочей группы. Это сервер, который предназначен для работы в малом офисе или домашней сети. Обычно на таком сервере установлены приложения, которые используются всеми пользователями сети, а также различные ресурсы, такие как принтеры и файловые хранилища.
2. Сервер контроллер домена. Используется для управления сетевой безопасностью и аутентификацией пользователей в доменной среде. Он содержит информацию о пользователях и группах, а также определяет права доступа пользователей к ресурсам сети.
3. Прокси-сервер. Используется для контроля доступа пользователей к Интернету, блокировки нежелательных сайтов и защиты сети от вирусов и других вредоносных программ.
4. Сервер электронной почты. Этот сервер предназначен для обработки и доставки электронной почты. Он отвечает за хранение электронных писем, а также за передачу и получение писем между клиентами.
5. Веб-сервер. Используется для хранения и обработки веб-сайтов. Он отвечает за обработку запросов пользователей и передачу веб-страниц на клиентские компьютеры.
6. Терминальный сервер. Используется для предоставления доступа к приложениям на удаленных компьютерах. Он позволяет пользователям подключаться к серверу и запускать приложения на удаленном компьютере.
7. Сервер базы данных. Используется для хранения и управления данными в базах данных. Он обеспечивает быстрый доступ к данным и защищает их от несанкционированного доступа.
8. Файловый сервер. Используется для хранения и обмена файлами в сети. Он предоставляет доступ к общим папкам и файлам, которые могут использоваться несколькими пользователями.
9. Серверы приложений. Серверы используются для запуска и управления приложениями на удаленных компьютерах. Они позволяют пользователям подключаться к серверу и запускать приложения на удаленном компьютере.
10. Брандмауэры и файрволы – это серверы, которые используются для обеспечения безопасности в сети. Они контролируют доступ к сети, фильтруют трафик, блокируют нежелательные соединения и препятствуют атакам извне.
11. Серверы DHCP используются для автоматической настройки сетевых устройств на подключение к сети. Они выдают IP-адреса, маски подсетей и другие параметры, необходимые для подключения к сети.
12. Серверы FTP предназначены для обмена файлами между устройствами в сети. Они обеспечивают доступ к файловой системе сервера и управляют передачей файлов между клиентами.
13. Принт-серверы – это серверы, которые обеспечивают доступ к сетевому принтеру и управляют печатью. Они позволяют настроить права доступа к принтеру, управлять очередью печати и мониторить состояние устройства.
14. Домашний сервер – это сервер, который используется в домашней сети для хранения и обмена файлами между устройствами в сети. Он может также использоваться для резервного копирования данных, централизованного управления мультимедийными файлами и других задач.

Серверы могут быть классифицированы по классу на несколько типов, в зависимости от их производительности, масштабируемости и надежности. Рассмотрим основные типы серверов по классу:

1. Серверы начального уровня (Entry-level server) - обеспечивают базовую функциональность для небольших организаций или отделов. Они имеют ограниченную производительность и масштабируемость, могут работать с несколькими пользователями и приложениями.
2. Серверы для рабочих групп (Workgroup-level server) - используются для обслуживания небольших рабочих групп или подразделений. Они обеспечивают более высокую производительность и масштабируемость, чем серверы начального уровня, и могут поддерживать большее количество пользователей и приложений.
3. Серверы уровня департамента (Department-level server) - предназначены для обслуживания отдельных департаментов или подразделений организации. Они обеспечивают более высокую производительность и масштабируемость, чем серверы для рабочих групп, и могут обслуживать большее количество пользователей и приложений.
4. Серверы уровня предприятия (Enterprise-level server) - предназначены для обеспечения высокой производительности и масштабируемости в больших организациях. Они могут обслуживать большое количество пользователей и приложений, имеют большую емкость хранения данных, поддерживают более высокий уровень доступности и надежности.

Серверы могут быть классифицированы по типу серверного шасси на несколько типов, в зависимости от их конструкции и предназначения. Рассмотрим основные типы серверов по типу серверного шасси:

1. Стоечные серверы (Rack-серверы) — это серверы, которые устанавливаются в стойки, обычно размером 19 дюймов. Этот тип серверов обеспечивает эффективное использование пространства, позволяя устанавливать несколько серверов в одной стойке. Стоечные серверы обладают высокой масштабируемостью, надежностью и производительностью.
2. Блейд-серверы — это серверы, которые устанавливаются в специальных модулях, называемых блейдами, которые затем устанавливаются в шасси. Блейд-серверы обеспечивают высокую плотность установки и эффективное использование ресурсов, так как они занимают меньше места, чем стоечные серверы. Блейд-серверы обладают высокой масштабируемостью, производительностью и надежностью, и часто используются в больших центрах обработки данных.
3. Серверы башенного типа "Tower" — это серверы, которые имеют конструкцию башни и устанавливаются на полу. Этот тип серверов наиболее распространен в малых организациях, где не требуется большая плотность установки. Серверы башенного типа обеспечивают высокую масштабируемость, надежность и производительность, но занимают больше места, чем стоечные серверы.
4. **Понятие сервера, сервера БД. Xарактеристики и функции сервера БД. Функции клиентского приложения БД.**

Сервер — это компьютер или программное обеспечение, которое предназначено для обработки и хранения данных, а также для предоставления доступа к этим данным через сеть.

Серверы используются в различных областях, включая бизнес, науку, образование, медиа, развлечения и технологии. Они могут быть физическими устройствами, которые установлены в офисе или в дата-центре, или виртуальными машинами, которые запускаются на удаленном сервере.

Термин "сервер баз данных" обычно используют для обозначения всей СУБД, основанной на архитектуре "клиент-сервер", включая и серверную, и клиентскую части. Такие системы предназначены для хранения и обеспечения доступа к базам данных.

Хотя обычно одна база данных целиком хранится в одном узле сети и поддерживается одним сервером, серверы баз данных представляют собой простое и дешевое приближение к распределенным базам данных, поскольку общая база данных доступна для всех пользователей локальной сети.

**Характеристики сервера БД.**

Современные серверные СУБД:

* существуют в нескольких версиях для различных платформ, как правило, для различных коммерческих версий UNIX – Solaris, HP/UX. Многие производители также выпускают версии своих серверов баз данных для Windows NT Workstation Windows 95/98, а также версии для Linux;
* в большинстве случаев поставляются с удобными административными утилитами;
* осуществляют резервное копирование и архивацию данных и журналов транзакций;
* поддерживают несколько сценариев репликаций;
* позволяют осуществлять параллельную обработку данных в многопроцессорных системах. Серверы, допускающие параллельную обработку, разрешают нескольким процессорам обращаться к одной БД, что обеспечивает высокую скорость обработки транзакций;
* поддерживают создание хранилищ данных и OLAP. Хранилище данных – это совокупность данных, полученных прямо или косвенно их информационных систем, которые содержат текущую и деловую информацию, а также из некоторых внешних источников.
* выполняют распределенные запросы и транзакции;
* дают возможность использовать различные средства проектирования схем данных – универсальные или ориентированные на конкретную СУБД;
* имеют средства разработки клиентских приложений и генераторы отчетов;
* поддерживают публикацию баз данных в Интернет;
* обладают широкими возможностями управления пользовательскими привилегиями и правами доступа к различным объектам БД.

**Функции сервера БД.**

* выполнение пользовательских запросов на выбор и модификацию данных и метаданных, получаемых от клиентских приложений, функционирующих на персональных компьютерах локальной сети;
* хранение и резервное копирование данных;
* поддержка ссылочной целостности данных согласно определенным в базе данных правилам;
* обеспечение авторизованного доступа к данным на основе проверки прав и привилегий пользователей;
* протоколирование операций и ведение журнала транзакций.

**Функции клиентского приложения БД.**

* Ввод-вывод данных (презентационная логика). Это часть кода клиентского приложения, которая определяет, что пользователь видит на экране, когда работает с приложением.
* Бизнес-логика. Это часть кода клиентского приложения, которая определяет алгоритм решения конкретных задач приложения.
* Обработка данных внутри приложения (логика базы данных). Это часть кода клиентского приложения, которая связывает данные сервера с приложением. Для этой связи используется процедурный язык запросов SQL, с помощью которого осуществляется выборка и модификация данных в серверных СУБД.

1. **Понятие API, функции API. Достоинства и недостатки API.**

**Понятие API, функции API.**

API (Application programming interface) — это программный интерфейс приложений, набор инструкций, который позволяет разным приложениям общаться между собой.

Функции API включают получение, отправку, изменение или удаление информации. Для этого одно приложение отправляет запрос в другое, а то, в свою очередь, формирует ответ.

Некоторые конкретные функции API:

* отправка курса валют;
* авторизация в CRM-системе с возможностью добавлять, удалять и изменять информацию;
* проведение платежа через банковскую систему;
* распознавание голоса и перевод его в текст;
* перевод текста;
* получение информации об актуальных остатках товара.

API позволяет не разрабатывать свои функции, а пользоваться уже разработанными.

**Достоинства и недостатки API.**

Преимущества протоколов API

1. Стандартизация

Протоколы API определяют общие структуры и форматы данных, что способствует стандартизации и обеспечивает согласованность взаимодействия между разными приложениями и сервисами.

1. Модульность

Протоколы API позволяют разбить сложные системы на более мелкие, независимые компоненты, что упрощает разработку и поддержку приложений.

1. Улучшенное взаимодействие

Протоколы API способствуют сотрудничеству между разными командами разработчиков и организациями, позволяя им создавать приложения и сервисы, которые взаимодействуют друг с другом.

1. Безопасность

Многие протоколы API предоставляют механизмы аутентификации и авторизации, что позволяет контролировать доступ к данным и функциональности.

1. Расширяемость

API-протоколы могут быть расширены и изменены без необходимости изменения всей системы. Это упрощает добавление новых функций и возможностей.

Недостатки протоколов API

1. Сложность

Некоторые API-протоколы могут быть сложными и требовать глубокого понимания для их использования и реализации.

1. Совместимость

Изменения в протоколах API могут привести к проблемам совместимости, особенно если старые версии клиентов или серверов не поддерживают новые изменения.

1. Производительность

Некоторые протоколы могут иметь накладные расходы в виде лишнего объема данных или дополнительных запросов, что может сказаться на производительности.

1. **Банк данных (БД): понятие, состав, требования к БД, архитектура БД.**

Банк данных— этоавтоматизированная информационная система централизованного хранения и коллективного использования данных**.**

В состав банка данных входят одна или несколько баз данных, справочник баз данных, система управления базами данных (СУБД), а также библиотеки запросов и прикладных программ.

Понятие банка данных также может обозначать организации, специализирующиеся на сборе и обработке информации и информационном обслуживании с использованием баз данных.

**Состав банка данных (схема).**

Состав банка данных включает в себя следующие компоненты:

1. Информационная компонента. Ядро банка данных, включает в себя базу данных (БД) — совокупность данных, организованных по определённым правилам. В состав БД включаются не только хранимые данные о предметной области, но и описание БД (метаинформация). Центральное хранилище метаинформации называется словарем данных.
2. Языковые средства. Предназначены для общения пользователей различных классов с банком данных, служат для описания различных компонентов БнД, а иногда и внешних по отношению к банку элементов.
3. Программные средства. Представляют собой сложный комплекс, обеспечивающий взаимодействие всех частей информационной системы в процессе её существования. В составе программных средств БнД можно выделить программы управления данными, трансляторы с языков БнД, различные вспомогательные программы (утилиты), программные средства, обеспечивающие взаимодействие пользователей и технических средств в различных режимах, операционные системы.
4. Технические средства. К ним относятся все технические компоненты вычислительной техники, устройства ввода-вывода, внешние запоминающие устройства, каналы связи.
5. Организационно-методические инструменты. Включают в себя методические материалы для пользователей различных категорий.

**Требования к банку данных.**

Некоторые требования к банку данных:

Адекватность отображения предметной области. Полнота, целостность и непротиворечивость данных, актуальность информации.

Возможность взаимодействия пользователей разных категорий и в разных режимах. Обеспечение высокой эффективности доступа для разных приложений.

Дружелюбность интерфейсов и малое время на освоение системы, особенно для конечных пользователей.

Обеспечение секретности и конфиденциальности для некоторой части данных. Определение групп пользователей и их полномочий.

Обеспечение взаимной независимости программ и данных.

Обеспечение надёжности функционирования банка данных. Защита данных от случайного и преднамеренного разрушения. Возможность быстрого и полного восстановления данных в случае их разрушения.

Приемлемые характеристики функционирования банка данных (стоимость обработки, время реакции системы на запросы, требуемые машинные ресурсы и др.).

Простота и удобство обращения пользователей за информацией и выдача информации в удобном для пользователя виде.

Возможность одновременного обслуживания большого числа пользователей.

**Архитектура банка данных.**

Архитектура банка данных

На концептуальном уровне поддерживается одна модель для всех приложений.

Третьим уровнем архитектуры является внутренний уровень. Внутренняя модель (ВнМД) есть представление самого низкого уровня всей базы данных; она состоит из различных экземпляров типов внутренних записей. Термин внутренняя запись применяется для конструкции, которая называется хранимой записью. Таким образом, внутренняя модель является ещё одним шагом в сторону от физического уровня, так как она не строится в терминах физических записей или блоков. ВнМД описывается посредством внутренней схемы, которая не только определяет различные типы хранимых записей, но и то, какие индексы существуют, как представлены хранимые поля, какова физическая последовательность хранимых записей и так далее.

В этой архитектуре имеется два уровня отображения: между внешним и концептуальным уровнями системы и между концептуальным и внутренним уровнями. Отображение “концептуальный – внутренний” определяет соответствие между моделью данных и хранимой базой данных; оно указывает, как концептуальные записи и поля отображаются в их хранимые копии. Если структура хранимой базы данных изменяется, то есть если изменяется определение структуры хранения, отображение “концептуальны – внутренний” должно быть соответственно изменено так, чтобы концептуальная схема оставалась неизменной.

Отображение “внешний – концептуальный” определяет соответствие между конкретной внешней моделью и моделью данных. В общем случае между этими двумя уровнями могут существовать те же виды различий, что и между моделью данных и базой данных. Например, поля могут иметь различные типы данных, записи могут быть по-разному упорядочены и так далее. Несколько внешних моделей может существовать одновременно; несколько пользователей могут совместно использовать данную внешнюю модель; различные внешние модели могут пересекаться.

Система управления базой данных (СУБД) является программой (инструментом), которая управляет всем доступом к базе данных.

1. **Конфигурирование сети.**

**Конфигурирование сети —** это процесс настройки и управления компонентами компьютерной сети, чтобы обеспечить их правильную работу, безопасность и эффективность. Включает в себя настройки сетевых устройств, таких как маршрутизаторы, коммутаторы, серверы, а также настройку параметров сетевых протоколов и безопасности. Это важный аспект в работе как локальных сетей (LAN), так и широких (WAN) или глобальных сетей (Internet).

**1. Регистрация удаленных серверов**

Перед использованием локального или удаленного сервера в среде MS SQL Server Management Studio его необходимо зарегистрировать.

При запуске MS SQL Server Management Studio первый раз регистрация локального сервера происходит автоматически. И только при работе с удаленными серверами их необходимо регистрировать, используя среду MS SQL Server Management Studio для выполнения команд меню, запуска Register Server Wizard или интерпретации команд Transact-SQL и системной хранимой процедуры sp-addserver.

При регистрации сервера необходимо указать следующую информацию:

1. Имя сервера.

2. Тип безопасности, используемый для входа на сервер.

3. Имя учетной записи и пароль, используемые для входа на сервер.

4. Имя группы в иерархии групп, в которой необходимо зарегистрировать сервер.

Утилита MS SQL Server Management Studio представляет собой всего лишь графический интерфейс для выполнения системных хранимых процедур SQL Server. Таким образом, она является клиентским средством, устанавливающим соединение с SQL Server и выполняющим те или иные процедуры. Поэтому, прежде чем эта программа сможет выполнить какие-либо операции с сервером, она должна получить соответствующие права доступа, т.е. пройти аутентификацию.

**2. Запуск, остановка и приостановка служб сервера**

До выполнения каких-либо работ по администрированию сервера MS SQL Server или баз данных, а также манипулированию данными необходимо запустить сервер. Точнее говоря, запустить его основную службу MSSQLServer. Только после запуска этой службы и проверки прав доступа пользователя, пользователь сможет выполнять функции, определенные его правами и разрешениями. Остальные службы являются вспомогательными, и их работа строится на фундаменте, обеспечиваемом службой MSSQLServer.

Например, служба SQLServerAgent запускается лишь тогда, когда требуется автоматическое администрирование и управление системой на базе SQL Server.

Служба MSSearch используется для работы с электронными документами, обеспечивает полнотекстовый поиск информации и, как правило, используется автономно.

Служба MSDTC позволяет организовать доступ к распределенным источникам информации и управлять распределенными транзакциями.

Дополнительные службы запускаются отдельно и устанавливают соединение с сервером, подобно обычным клиентам. Каждая такая служба самостоятельно подключается к основной службе MSSQLServer, используя определенные учетные записи с соответствующими правами доступа.

Для сетевого варианта установки управлять службами можно как локально, так и удаленно даже средствами операционной системы. Запускать, останавливать и приостанавливать сервер можно также при отсутствии сети.

1. **Команды SQL для создания БД, создания, удаления и изменения таблиц.**

**1. Создание базы данных (CREATE DATABASE)**

Для создания новой базы данных используется команда CREATE DATABASE. Эта команда позволяет задать имя базы данных и ее параметры.

**2. Удаление базы данных (DROP DATABASE)**

Чтобы удалить базу данных, используется команда DROP DATABASE. Она удаляет саму базу данных и все данные в ней.

**3. Создание таблицы (CREATE TABLE)**

Для создания новой таблицы в базе данных используется команда CREATE TABLE. С помощью этой команды можно указать название таблицы, а также определить структуру таблицы — имена столбцов и их типы данных.

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

FullName VARCHAR(100),

Position VARCHAR(50),

Department VARCHAR(50),

DateOfHire DATE

**4. Удаление таблицы (DROP TABLE)**

Команда DROP TABLE используется для удаления существующей таблицы из базы данных. Вся информация, содержащаяся в таблице, будет утрачена.

**5. Изменение структуры таблицы (ALTER TABLE)**

Команда ALTER TABLE используется для изменения структуры существующей таблицы. С ее помощью можно добавлять, изменять или удалять столбцы, а также изменять другие свойства таблицы.

ALTER TABLE Employees

ADD Email VARCHAR(100);

6**. Изменение имени таблицы (RENAME TABLE)**

Для изменения имени таблицы используется команда RENAME TABLE. Однако в различных СУБД синтаксис может отличаться, и в некоторых случаях требуется использовать команду sp\_rename в SQL Server.

RENAME TABLE Employees TO Staff;

1. **Работа с журналом аудита базы данных.**

При работе сервера необходимо отслеживать события, связанные с его работой. Главная цель при этом – как можно быстрее обнаруживать проблемы, возникающие на сервере, и оперативно реагировать на них.

Самое простое средство мониторинга работы MS SQL Server – журналы сервера. Считается, что рабочий день администратора на предприятии должен начинаться с просмотра журналов событий на всех серверах. Для MS SQL Server начиная с версии 2005 выделяются 4 журнала:

* журнал событий самого SQL Server;
* журнал событий SQL Server Agent;
* журнал событий операционной системы Windows;
* журнал событий приложений Windows.

Журналы можно просматривать разными способами. Самый простой и рекомендованный – использовать просмотрщик, который встроен в SQL Server Management Studio. Запустить его можно из контейнера Management | SQL Server Logs (Управление) | Журналы SQL Server. В списке журналов нужно щелкнуть правой кнопкой мыши по требуемому журналу и в контекстном меню выбрать View SQL Server Log (Просмотреть журнал SQL Server). Откроется окно просмотрщика журналов

При помощи просмотрщика журналов можно просматривать не только журналы SQL Server, но и журналы SQL Server Agent, Windows и Database Mail, можно экспортировать данные из журналов при помощи кнопки Export (Экспортировать), в том числе и в очень удобный для загрузки в базу данных формат CSV, можно настраивать фильтрацию и производить поиск нужной информации.

Журналы событий SQL Server можно просматривать и «вручную», при помощи любого текстового редактора. По умолчанию они находятся в каталоге C:\Program Files\Microsoft SQL Server\MSSQL\LOG. Там же находятся и журналы SQL Server Agent.

Если вам нужен более подробный протокол событий, происходящий на сервере, можно воспользоваться параметром C2 Audit Tracing. Его можно установить на вкладке Security (Безопасность) свойств сервера в Management Studio. В этом режиме в каталоге Data (Данные) для данного экземпляра сервера будут автоматически создаваться текстовые файлы с очень подробной информацией, которая может потребоваться для аудита в соответствии со стандартом безопасности C2.

1. **Создание SQL запросов с использованием специальных операторов: принадлежность множеству; принадлежность диапазону; соответствие шаблону; соответствие регулярному выражению; неопределенное значение. Привести примеры.**

**1. Принадлежность множеству — оператор IN**

Оператор IN используется для проверки того, принадлежит ли значение одному из элементов заданного множества. Это особенно удобно, если нужно проверить несколько значений в одном запросе.

**Синтаксис:**

SELECT \*

FROM Employees

WHERE Position IN ('Manager', 'Developer', 'Analyst');

**2. Принадлежность диапазону — оператор BETWEEN**

Оператор BETWEEN используется для проверки того, лежит ли значение в пределах определенного диапазона. Диапазон может быть задан числовыми значениями, датами и т.д.

**Синтаксис:**

SELECT \*

FROM Employees

WHERE Salary BETWEEN 50000 AND 100000;

**3. Соответствие шаблону — оператор LIKE**

Оператор LIKE используется для поиска значений, которые соответствуют определенному шаблону. Шаблон может содержать специальные символы подстановки, такие как % (любой набор символов) и \_ (один символ).

**Синтаксис:**

SELECT \*

FROM Employees

WHERE FullName LIKE 'J%';

**4. Соответствие регулярному выражению — оператор PATINDEX или REGEXP**

SQL Server не поддерживает прямую работу с регулярными выражениями, однако можно использовать функцию PATINDEX для поиска строк, которые соответствуют определенному шаблону. Для работы с регулярными выражениями в MySQL и других СУБД можно использовать REGEXP.

**Синтаксис:**

SELECT \*

FROM Employees

WHERE FullName REGEXP '^[A-Za-z]+$';

(Этот запрос выберет сотрудников, чьи имена содержат только буквы.)

**5. Неопределенное значение — оператор IS NULL и IS NOT NULL**

Оператор IS NULL используется для проверки, является ли значение в столбце неопределенным (NULL). Оператор IS NOT NULL используется для проверки, что значение не является NULL.

**Синтаксис:**

SELECT \*

FROM Employees

WHERE Department IS NULL;

(Этот запрос выберет всех сотрудников, у которых не указан отдел (NULL).

SELECT \*

FROM Employees

WHERE Department IS NULL;

(Этот запрос выберет всех сотрудников, у которых указан отдел (не NULL).)