**Вопросы к экзамену по ПМ 11 «Разработка, администрирование и защита баз данных»**

1. Приведите классификацию баз данных. Определения, основные функции, виды.

Классификация баз данных:

Иерархическая данные хранятся виде дерева

Реляционная таблицы

Сетевая в виде сети

Многомерная таблицы могут быть в виде многомерных структур

1. Основы реляционной алгебры. Определения высказываний, запись, примеры.

Основы реляционной алгебры

**Объединение(Union)**

**(**SELECT A1, A2, A3 from R1 UNION SELECT A1, A2, A3 from R2**)** - операция над двумя отношениями, в результате которой получается новое отношение, состоящее из всех кортежей (ключ значение) исходных отношений

**Пересечение(Intersect)**

**(**SELECT A1, A2, A3 from R1 INTERSECT SELECT A1, A2, A3 from R2**)** - операция над двумя отношениями, в результате которой получается новое отношение, состоящее из кортежей, принадлежащих обоим исходным отношениям.

**Вычитание(Minus)**

**(**SELECT A1, A2, A3 from R2 EXCEPT SELECT A1, A2, A3 from R1**) -** операция над двумя отношениями, в результате которой получается новое отношение, состоящее из кортежей, принадлежащих первому отношению и не принадлежащих второму.

**Декартово произведение(Times)**

**(**SELECT \* from R3, R4**)** - операция над двумя отношениями, в результате которой получается новое отношение, состоящее из всех возможных кортежей, являющихся попарными сочетаниями кортежей исходных отношений.

**Выборка(Where)**

**(**SELECT \* from R3 WHERE A3>'d0'**)** производится над кортежами одного отношения. Результат выборки - новое отношение, состоящее из кортежей исходного отношения, удовлетворяющих заданному условию.

**Проекция(Project)**

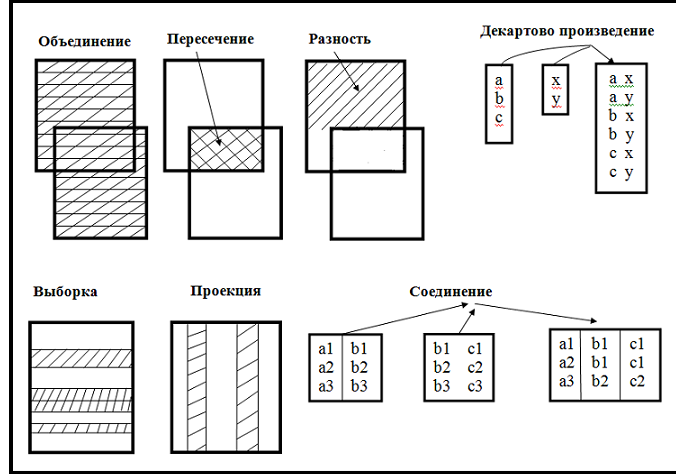
**(**SELECT DISTINCT A4, A3 from R3**)** также производится над кортежами одного отношения. Результат проекции - новое отношение содержащее только заданные атрибуты исходного отношения.

**Соединение(Join)**

(SELECT \* from R3, R4 WHERE A1>=A5)операция над двумя отношениями, имеющими общие атрибуты, в результате которой получается новое отношение, состоящее из всех атрибутов исходных отношений и объединяющее только те кортежи исходных отношений в которых значения общих атрибутов совпадают.

**Деление**

**(**SELECT DISTINCT A1, A4 from R5 WHERE  
NOT EXIST (SELECT \* from R6 WHERE NOT EXIST  
R6.A2 = R5.A2 AND  
R6.A3 = R5.A3)**) -** в целом, из таблицы A берутся значения строк, для которых присутствуют все комбинации значений из таблицы B.(рентв и стс)



1. Иерархическая модель данных. Основные понятия, графическое изображение, примеры.

**Иерархическая** **модель** **данных** — это **модель** **данных**, где используется представление базы **данных** в виде древовидной (**иерархической**) структуры

**Основные определения**

Связь Предок Потомок(1 предок – много потомков)

1. Сетевая модель данных. Основные понятия, графическое изображение, примеры.

Сетевая модель данных – как иерархическая, но возможно много предков.

1. Реляционная модель данных. Основные понятия, графическое изображение, примеры.

Таблицы(запись – строка, поле – столбец)

1. Первая, вторая, третья нормальные формы реляционной модели данных. Определение, требования, примеры.

**Первая нормальная форма**

* Исключите повторяющиеся группы в отдельных таблицах.
* Создайте отдельную таблицу для каждого набора связанных данных.
* Идентифицируйте каждый набор связанных данных с помощью первичного ключа.

Чтобы была соблюдена первая форма, все данные в полях должны быть атомарны. Т.е. одно поле - одно значение.

**Вторая нормальная форма**

* Создайте отдельные таблицы для наборов значений, которые применяются к нескольким записям.
* Свяжите эти таблицы с помощью внешнего ключа.

Во-первых, ваша таблица уже должна быть в первой нормальной форме. Во-вторых, в вашей таблице обязательно должен быть первичный ключ и все записи должны от него зависеть. Те данные, которые не зависят от данного ключа, должны быть вынесены в отдельную таблицу-справочник.

**Третья нормальная форма**

* Исключите поля, которые не зависят от ключа.

Для начала ваша таблица должна быть в первой и второй нормальной форме. Чтобы была соблюдена третья нормальная форма, у вас не должно быть транзитивной зависимости. (~~город~~ - индекс)

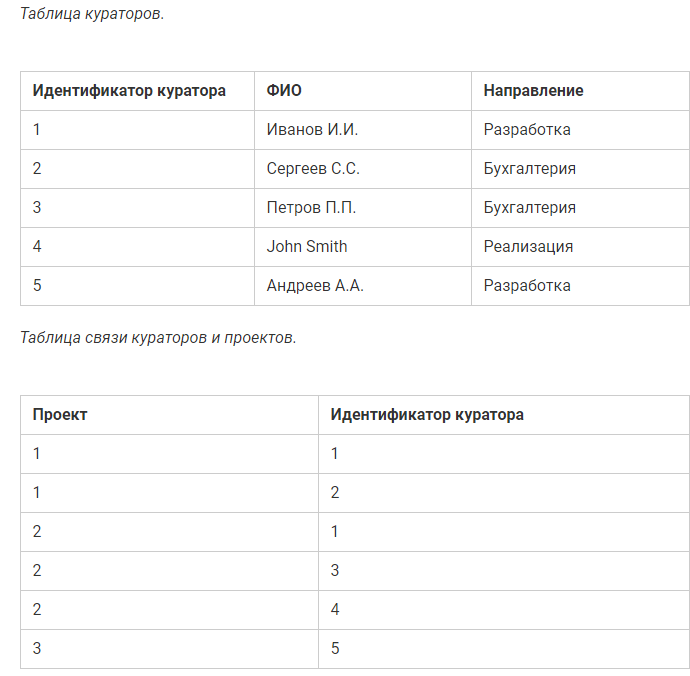
1. НФБК, четвертая нормальная форма реляционной модели данных. Определение, требования, примеры.

**НФБК** - Чтобы привести данную таблицу к нормальной форме Бойса-Кодда, необходимо, как всегда сделать декомпозицию данного отношения, т.е. разбить эту таблицу на несколько таблиц.

## Требования

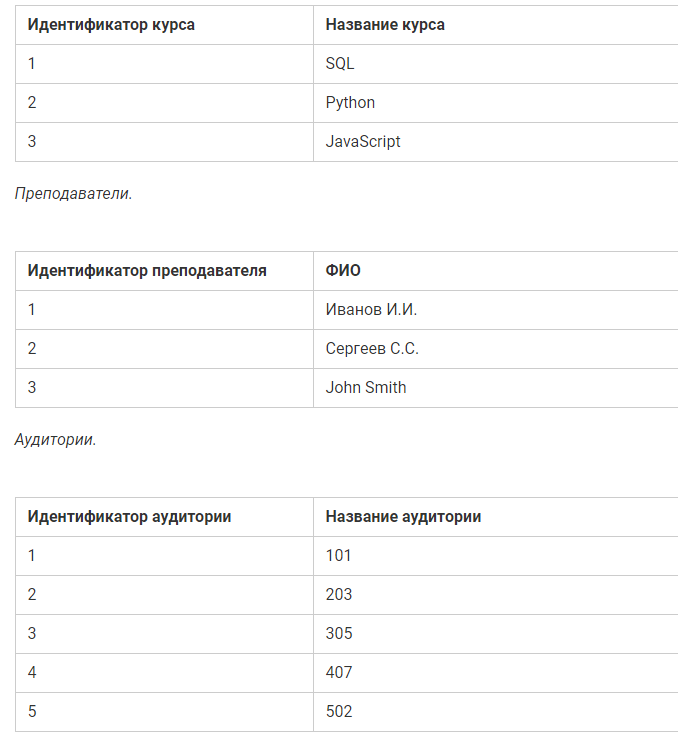
Требования нормальной формы Бойса-Кодда следующие:

* Таблица должна находиться в третьей нормальной форме. Здесь все как обычно, т.е. как и у всех остальных нормальных форм, первое требование заключается в том, чтобы таблица находилась в предыдущей нормальной форме, в данном случае в третьей нормальной форме;
* Ключевые атрибуты составного ключа не должны зависеть от неключевых атрибутов.

**Пример:** 

**Четвертая нормальная форма**

Чтобы база данных находилась в четвертой нормальной форме, необходимо, чтобы независимые элементы данных, между которыми существует связь типа многие-ко-многим, не хранились в одной таблице.

**Было: **

**Стало:** 

1. Модель сервера приложений. Архитектура, компоненты, преимущества.

**Модель** **сервера** **приложений** позволяет помещать прикладные **программы** на отдельные **серверы** **приложений**. **Программа**, выполняемая на компьютере-клиенте, решает задачу ввода и отображения данных. Прикладной **компонент** реализован как группа процессов, выполняющих прикладные функции, и называется **сервером** **приложения**.

Преимущества:

• позволяют организовывать сети с большим количеством рабочих станций;

• обеспечивают управление учетными записями пользователей;

• предоставляют эффективный доступ к сетевым ресурсам.

недостатки:

• неисправность сервера может сделать сеть неработоспособной, что влечет как минимум потерю сетевых ресурсов;

• требует квалифицированного персонала для администрирования;

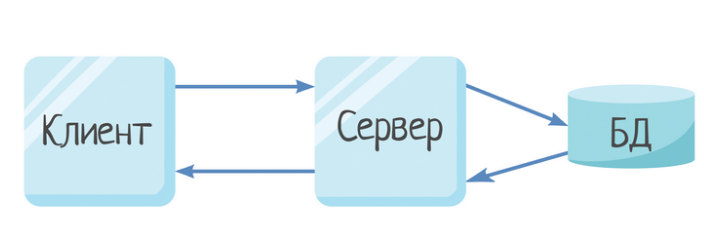
• имеет более высокую стоимость сетей и сетевого оборудования.

1. Модель клиент. Архитектура, компоненты, преимущества.

**RDA-модель**

В модели доступа к удаленным данным на компьютере-клиенте располагаются презентационная логика и бизнес-логика приложений. На сервере находится ядро СУБД. Функции сервера определяются управлением данными и обработкой запросов со стороны клиентов. Клиент обращается к серверу с запросами на языке SQL. В ответ на запрос клиент получает только данные, соответствующие запросу.

**Основное достоинство данной** модели состоит в том что, во-первых, взаимодействие пользователя с сервером осуществляется с помощью стандартного языка запросов SQL. Во-вторых, это наличие большого числа готовых СУБД и других инструментальных средств, обеспечивающих быстрое создание программ клиентской части



1. Создание структуры таблицы. Типы данных таблиц базы данных. Ключевое поле таблиц баз данных.

**Ключевое** **поле** – такое **поле**, значение которого служит для однозначного определения записи в **таблице**, обычно это цифровой код. Ключ – одно или несколько **ключевых** **полей**, позволяющих идентифицировать записи **таблицы** и организовывать связи между **таблицами**.

Структура таблицы определяется:

• названиями полей, из которых она состоит

• типами полей и размерами полей.

Типы данных

|  |  |
| --- | --- |
| **Текстовые типы данных** | |
| CHAR | Содержит строку **фиксированной** длины (может содержать буквы, цифры и специальные символы). **Фиксированный** размер указывается в скобках. Может хранить до 255 символов |
| VARCHAR | Содержит строку **переменной** длины (может содержать буквы, цифры и специальные символы). Максимальный размер указывается в скобках. Может хранить до 255 символов. **Примечание**: Если поместить большее значение, чем 255, оно будет преобразовано в текстовый тип |
| TINYTEXT | Содержит строку с максимальной длиной 255 символов |
| TEXT | Содержит строку с максимальной длиной 65 535 символов |

|  |  |
| --- | --- |
| **Типы данных чисел** | |
| INT | -2147483648 до 2147483647 нормальный. 0 до 4294967295 неподписанный \*. Максимальное количество цифр может быть указано в скобках |
| FLOAT | Небольшое число с плавающей запятой |
| DOUBLE | Большое число с плавающей запятой |

|  |  |
| --- | --- |
| **Типы данных** | |
| DATE | Хранить только дату. С 1 января 0001 по 31 декабря 9999 |
| TIME | Хранить время только для точности 100-ти секунд |
| DATETIME | С 1 января 1753 по 31 декабря 9999 с точностью 3,33 миллисекунд |

1. Заполнение таблиц данными. Установление связей между таблицами. Назначение, главная и подчиненная таблица, последовательность действий.
2. Запрос на выборку. Назначение, создание, примеры. Запрос с параметром. Назначение, создание, примеры. Создание запроса на выборку с логическими операциями в условиях отбора.

**Запрос на выборку.**

Для выборки данных в SQL используется специальная команда, которая называется SELECT.





Знак \* означает, что выбираются все колонки из таблицы direct.

Кроме того, мы можем указать название конкретных столбцов через запятую, которые мы хотели бы выбрать.



**Запрос с параметром**

Параметры в значительной степени повышают гибкость SQL-запро­сов, обеспечивая возможность при выборке данных запрашивать у пользователя численные значения критериев выборки.

Параметры задаются в тексте SQL-запроса. Для определения параметра перед его именем указывается символ двоеточия (:), например:

SELECT \*

FROM tablename

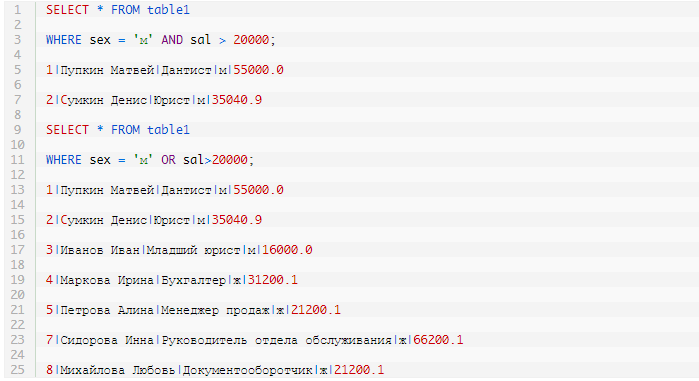
WHERE fieldl<:PARAM1

В данном запросе задан один параметр с именем PARAM1.

После ввода текста запроса в свойство SQL автоматически производится заполне­ние массива в свойстве Params.

**запроса на выборку с логическими операциями**

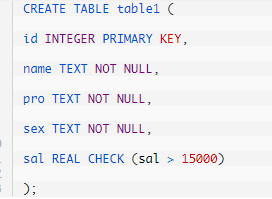
AND (и), OR (или) являются бинарными операторами, что для них следует использовать два операнда (два значения: слева и справа).



1. Запрос на создание таблицы. Запрос на удаление. Назначение, создание, примеры. Запрос на обновление. Запрос на добавление. Назначение, создание, примеры.

**Запрос на создание таблицы**

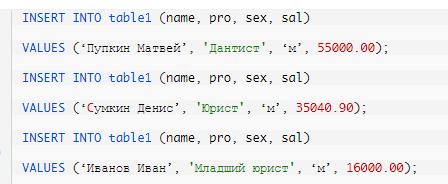
Для начала [создадим таблицу в базе данных](https://zametkinapolyah.ru/zametki-o-mysql/chast-10-1-sozdanie-tablic-v-bazax-dannyx-sqlite.html), воспользовавшись [командой CREATE](https://zametkinapolyah.ru/zametki-o-mysql/chast-6-1-komanda-create-v-sqlite3-ddl-operator-create.html):



[PRIMARY KEY](https://zametkinapolyah.ru/zametki-o-mysql/chast-11-3-pervichnye-klyuchi-v-bazax-dannyx-sqlite-primary-key-ogranichenie-pervichnogo-klyucha.html) нужен для двух вещей: ускорить выборку данных из базы данных и [обеспечить целостность данных в базе данных](https://zametkinapolyah.ru/zametki-o-mysql/tema-11-obespechenie-celostnosti-dannyx-v-bazax-dannyx-sqlite3.html)

**Запрос на добавление**

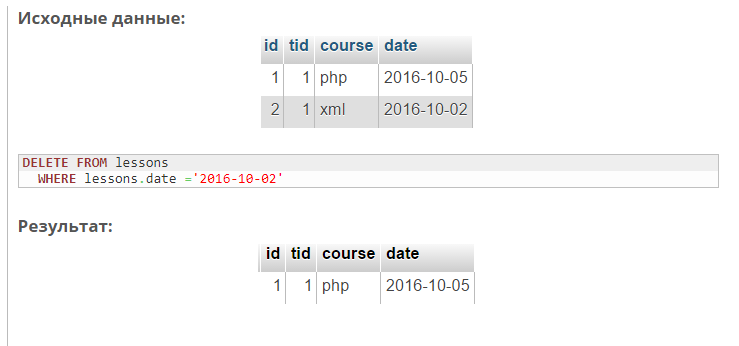
[добавим данные в таблицу базы данных](https://zametkinapolyah.ru/zametki-o-mysql/chast-10-4-dobavlenie-dannyx-i-strok-v-tablicy-baz-dannyx-sqlite.html), для этого воспользуемся [командой INSERT INTO](https://zametkinapolyah.ru/zametki-o-mysql/chast-7-2-komanda-insert-v-sqlite3-operator-insert-v-sqlite3.html):



**Запрос на удаление**

Оператор Delete или DROP удаляет записи из таблицы БД.

**DELETE** **FROM** <таблица> [**WHERE** <условие>];



**Запрос на обновление**

В SQL, изменить запись в таблице БД можно с помощью команды UPDATE. В самом минимальном виде команда обновления данных выглядит следующим образом:

UPDATE таблица SET поле = значение

Здесь, **UPDATE** – команда указывающая, что запрос на обновление данных;

**таблица** – название таблицы, в которой будет проводиться изменения;

**SET** – команда, после которой через запятую указываются поля с назначаемыми им значениями;

**поле** – поле таблицы, в которое будет внесено изменение;

**значение** – новое значение, которое будет внесено в поле.

1. Опишите основные принципы построения концептуальной, логической и физической модели данных.

### **Концептуальный уровень моделирования**

На концептуальном уровне моделирования мы определяем основные понятия предметной области и их взаимосвязь.

**Базовый элемент концептуальной модели:** бизнес-сущность или бизнес-объект.

**Примеры бизнес-сущностей:**

* клиент (как вариант, клиент-ФЛ, клиент-ЮЛ),

### **Логический уровень моделирования**

Логическая модель является уточнением и детализацией концептуальной модели. Но это лишь с одной стороны.  На построение логической модели также влияет:

* тип планируемой СУБД, которая будет воплощать модель
* класс проектируемой системы: операционная (транзакционная) или аналитическая (BI)

Логический уровень моделирования – это уровень логики организации данных, то есть какие данные и как сгруппированы и связаны друг с другом

### **Физический уровень моделирования**

Физический уровень – это уровень таблиц для реляционных моделей данных.

**Базовый элемент физической модели:** таблица.

**Примеры таблиц:**

* клиент   -► table\_1
* продукт  -► table\_2
* сделка -► table\_3

Не исключается, что одна таблица физического уровня может участвовать в моделировании сразу нескольких логических сущностей.

**Фокус моделирования**:

* Выделение отдельных таблиц, в том числе как результат нормализации данных.
* Выделение таблиц-справочников.
* Определение ключей.
* Разделение атрибутов на простые атрибуты и перечисления (будущие справочники)

1. Расскажите про структуры данных СУБД, общий подход к организации представлений, таблиц, индексов и кластеров

Индексов

#### Повышение производительности запросов: Кластеры

Самой медленной операцией, выполняемой СУБД, является операция чтения данных с диска или запись данных на диск. Если существует возможность уменьшить в несколько раз число таких операций, то общая производительность базы данных может заметно увеличиться.

*Кластеризация* (Clustering) - это способ физического размещения рядом, на одной *физической странице* данных, строк, доступ к которым осуществляется при помощи одинакового значения колонки (ключа) с целью увеличения производительности.

1. Расскажите про основные принципы структуризации и нормализации базы данных.

**Нормализации базы данных**.

**Нормализация** – это метод проектирования базы данных, который позволяет привести базу данных к минимальной избыточности т.е. (удаления избыточных данных).

**+ нормальные форма**

**Структуризации**

Создание БД должно начинаться с ее проектирования. Процесс

проектирования БД включает следующие основные этапы:

* 1. Определение назначения БД.
  2. Определение структуры таблиц
  3. Определение полей
  4. Определение ключевых полей
  5. Определение связей между таблицами

1. Создание новой базы данных
2. Создание таблиц
3. Создание ключевых полей
4. Создание связей между таблицами
5. Опишите методы описания схем баз данных в современных СУБД. Расскажите про структуры данных СУБД.

### **Трехуровневая архитектура баз данных. Три уровня абстракции описания данных.**

Самым важным аспектом трехуровневой **архитектуры базы данных** является то, что логическая структура, с которой взаимодействует пользователь, отделена от физической структуры баз данных, с которой взаимодействует машина.

Давайте посмотрим, что включает в себя трехуровневая архитектура баз данных:

* В базах данных с трех уровневой архитектурой всегда есть внешний уровень, на котором пользователю предоставляется интерфейс для манипуляции данных, на внешнем уровне задаются права доступа пользователей на взаимодействие с базой данных, приоритеты и т.д.
* Внутренний уровень, на котором СУБД и операционная система воспринимают и обрабатывают данные
* Поскольку архитектура у нас трех уровневая, то третий уровень отвечает за связь между внешним уровнем (пользовательским интерфейсом) и внутренним уровнем (физическим представлением данных в базе данных). Третий уровень получил название концептуальный уровень представления данных. Концептуальный уровень представления данных предназначен для отображения внешнего уровня на внутренний, он обеспечивает независимость между этими уровнями.

**Расскажите про структуры данных СУБД.**

Описание структуры данных на любом из трех уровней называется **схемой данных**

**Это 1 уровень**

Внешний уровень – это пользовательский уровень, по-другому – пользовательский интерфейс. **Внешняя**

**схема данных**описывает то, как видят различные группы пользователей архитектуру базы данных.

**Это 2 уровень**

**Концептуальный уровень представления данных** – это промежуточное звено между внутренним и внешним уровнями, это интерфейс взаимодействия между этими двумя уровнями.

**Это 3 уровень**

**Внутренний уровень представления данных** – это третий и последний по счету уровень архитектуры базы данных. Внутреннее представление данных не связано с их физическим представлением, так как каждая СУБД и каждый сервер баз данных имеет собственное представление данных на физическом уровне.

1. Расскажите про методы организации целостности данных в БД.

Целостность—актуальность и непротиворечивость информации, ее защищенность от разрушения и несанкционированного изменения.

Поле. Для него чаще всего используются следующие виды ограничений.

* Тип и формат поля. Тип поля определяет допустимые для данного поля символы, а иногда и более жесткие ограничения на допустимые значения (как, например, для полей типа дата или логическое).
* Задание диапазона значений. Обычно используется для числовых полей.
* Признак непустого поля. Характеризует недопустимость пустого значения поля в БД.
* Задание домена. Поле может принимать значение из заданного множества.
* Специфическим ограничением на значение поля является признак его уникальности. Это ограничение проверяет допустимость значения данного поля, но при этом просматривается вся таблица (файл).
* Очень важным видом ограничений целостности являются функциональные зависимости.

1. Опишите современные инструментальные средства проектирования схемы базы данных.

Pg admin

Sql server management studio

My sql workbench

20.  Введение в SQL и его инструментарий. Подготовка систем для установки SQL-сервера. Расскажите про установку и настройку SQL-сервера, автоматизация управления SQL

**Рекомендации**

Установку не удастся выполнить, если запустить программу установки через удаленный рабочий стол, но носитель при этом будет расположен на клиенте RDC. Чтобы выполнить установку удаленно, установочный носитель должен быть расположен на общем сетевом ресурсе или в локальной папке физической или виртуальной машины. Установочный носительSQL Server может быть расположен на общем сетевом ресурсе, сопоставленном диске, локальном диске, или он может быть представлен в виде ISO-образа на виртуальной машине.

Программа установкиSQL Server устанавливает следующие компоненты, необходимые для продукта:

Собственный клиент SQL Server

SQL Server Файлы поддержки программы установки

|  |  |
| --- | --- |
| **Инструмент** | |
| [Azure Data Studio](https://docs.microsoft.com/ru-ru/sql/azure-data-studio/download?view=sql-server-ver15) | Простой редактор, с помощью которого можно выполнять SQL-запросы по требованию, а затем анализировать и сохранять результаты в виде текста, а также в форматах JSON или Excel. Редактируйте данные, упорядочивайте избранные подключения к базам данных и просматривайте объекты базы данных в знакомом интерфейсе. |
| [SQL Server Management Studio (SSMS)](https://docs.microsoft.com/ru-ru/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15) | Управление экземпляром SQL Server или базой данных с полной поддержкой GUI. Возможности доступа, настройки, администрирования и разработки всех компонентов SQL Server, Базы данных SQL Azure и Azure Synapse Analytics, а также управления ими. Среда SSMS предоставляет единую полнофункциональную служебную программу, которая сочетает в себе обширную группу графических инструментов с рядом отличных редакторов сценариев для доступа к SQL для разработчиков и администраторов баз данных всех профессиональных уровней. |
| [SQL Server Data Tools (SSDT)](https://docs.microsoft.com/ru-ru/sql/ssdt/download-sql-server-data-tools-ssdt?view=sql-server-ver15) | Это современное средство разработки, позволяющее создавать реляционные базы данных SQL Server, базы данных SQL Azure, модели данных Analysis Services (AS), пакеты Integration Services (IS) и отчеты Reporting Services (RS). С помощью SSDT вы можете проектировать и развертывать любые типы содержимого SQL Server так же просто, как разрабатывать приложения в [Visual Stud](https://visualstudio.microsoft.com/downloads/) |
| [Visual Studio Code](https://code.visualstudio.com/) | [Расширение mssql](https://marketplace.visualstudio.com/items?itemName=ms-mssql.mssql) для Visual Studio Code — это официальное расширение SQL Server, которое поддерживает подключения к SQL Server и расширенные возможности редактирования для T-SQL в Visual Studio Code. Написание скриптов T-SQL в упрощенном редакторе. |

**SQL** Server Agent — это служба **SQL** Server, основное назначение которой — **автоматизация** выполнения административных операций.

21.  Расскажите про выполнение мониторинга SQLServer с использование оповещений и предупреждений.

Панель мониторинга производительности - Панель мониторинга производительности в SQL Server Management Studio помогает быстро выявлять наличие узких мест производительности в SQL Server.

Панель мониторинга производительности состоит из следующих разделов и вложенных отчетов:

1. Загрузка ЦП системы
2. Текущие ожидающие запросы
3. Текущее действие

* Запросы пользователей
* Сеансы пользователей
* Коэффициент попадания в кэш

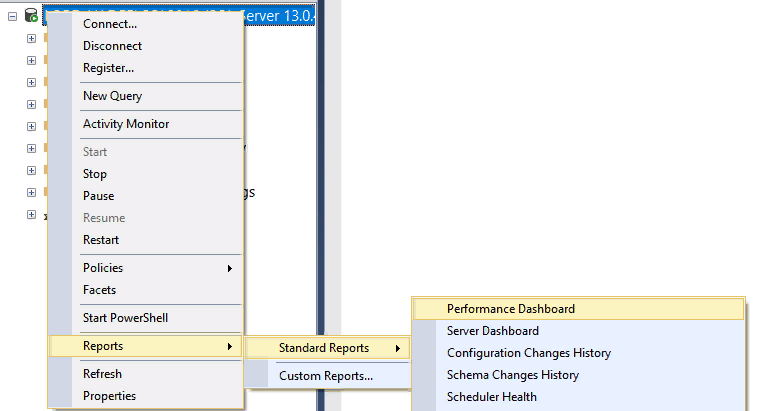
1. Исторические сведения

* Ожидания
* Кратковременные блокировки
* Статистика ввода-вывода
* Ресурсоемкие запросы

1. Прочие сведения

* Активные трассировки
* Активные сеансы xEvent
* Базы данных
* Отсутствующие индексы

Открытие панели мониторинга производительности

Чтобы открыть панель мониторинга производительности, щелкните правой кнопкой мыши имя экземпляра SQL Server в обозревателе объектов и последовательно выберите пункты Отчеты, Стандартные отчеты и Панель мониторинга производительности.

Панель мониторинга производительности в меню. Панель мониторинга производительности откроется на новой вкладке.

22.  Расскажите про поиск и решение типичных ошибок, связанных с администрированием.

23.  Расскажите про способы контроля доступа к данным и управления привилегиями.

После проектирования логической структуры базы данных, связей между таблицами, ограничений целостности и других объектов необходимо определить категории пользователей, которые будут иметь доступ к базе данных.

Как правило, в СУБД организована двухуровневая настройка ограничения доступа к данным. На первом уровне необходимо создать так называемую учетную запись для подключения к самому серверу, что еще не дает возможности доступа к базам данных. На втором уровне для каждой базы данных сервера на основании учетной записи необходимо создать запись пользователя для подключения к базе данных. Учетную запись пользователя можно добавить, изменить или удалить.

Таким образом, сервер обеспечивает двухуровневую защиту данных:

* 1) аутентификацию на уровне сервера;
* 2) идентификацию на уровне базы данных

Права доступа устанавливаются командой GRANT. Пользователь, выдающий права командой GRANT, может присвоить другому пользователю только те права, которыми он владеет сам, или более слабый набор прав. Для работы с таблицей или с представлением пользователь должен обладать правами на выполнение команд SELECT, INSERT, UPDATE, DELETE или REFERENCES. Для того чтобы задать привилегии на все эти команды сразу, можно использовать привилегию ALL. Для вызова хранимой процедуры в приложении пользователь или объект должны обладать правом на выполнение команды EXECUTE.

Перед тем как присвоить пользователю новые права, необходимо забрать у него старые. Это позволяет сделать команда REVOKE. Привилегии, которые были предоставлены указанному пользователю другими пользователями, не могут быть затронуты оператором REVOKE.

Как правило, в работе с учетными записями используют более крупные объекты — роли. Роль представляет собой объект базы данных, обладающий некоторыми правами и содержащий в себе учетные записи пользователей. Работать с ролью гораздо удобнее, чем с отдельными пользователями. Намного проще переопределить права одной роли, чем каждой учетной записи в отдельности. Роль позволяет объединить в одну группу пользователей, выполняющих одинаковые функции. Для создания роли используется оператор:

CREATE ROLE <имя\_роли>

Одна учетная запись может состоять сразу в нескольких ролях. Но во время одной сессии клиент может работать только под одной ролью. Это может быть удобно только в тех случаях, когда один пользователь должен иметь разные права доступа в зависимости от ситуации. Пользователь, работающий под правами какой-либо роли, наследует собственные права.

Каждая СУБД должна поддерживать механизм, гарантирующий, что доступ к базе данных смогут получить только те пользователи, которые имеют соответствующее разрешение. Язык SQL включает операторы GRANT и REVOKE, предназначенные для организации защиты таблиц в базе данных. Механизм защиты построен на использовании идентификаторов пользователей, предоставляемых им прав владения и привилегий.

24.  Опишите алгоритм проведения процедуры резервного копирования.

 Резервная копия в MSSMS

После подключения к соответствующему экземпляру Microsoft Компонент SQL Server Database Engine в обозревателе объектов разверните дерево сервера.

Разверните узел Базы данных и выберите пользовательскую базу данных или разверните узел Системные базы данных и выберите системную базу данных.

Щелкните правой кнопкой мыши базу данных, резервную копию которой нужно создать, наведите указатель на пункт Задачи и выберите команду Создать резервную копию... .

В диалоговом окне Резервное копирование базы данных выбранная база данных приводится в раскрывающемся списке (ее можно изменить на любую другую базу данных на сервере).

В раскрывающемся списке Тип резервной копии выберите нужный тип. По умолчанию выбран тип Полный.

В разделе Компонент резервного копирования выберите База данных.

В разделе Назначение проверьте расположение по умолчанию для файла резервной копии (в папке ../mssql/data).

(Необязательно) Просмотрите другие доступные параметры на страницах Параметры носителя и Параметры резервного копирования.

Нажмите кнопку ОК, чтобы запустить резервное копирование.

После успешного завершения резервного копирования нажмите кнопку ОК, чтобы закрыть диалоговое окно SQL Server Management Studio.

Восстановление (recovery) - это процесс использования устройства резервной копии для замены неподтвержденных, несогласованных или потерянных данных.

25.  Опишите модели восстановления SQL-сервера.

SQL Server имеет три различных модели восстановления: простая (Simple), полная (Full) и с неполным протоколированием (Bulk-Logged).

Simple recovery model является самой простой из моделей восстановления. Когда используется эта модель, каждая транзакция по-прежнему записывается в журнал транзакций. Записи журналов транзакций будут автоматически удаляться , если используется простая модель восстановления. Этот процесс удаления происходит для всех завершенных транзакций, когда устанавливается контрольная точка. Поскольку записи журнала удаляются при возникновении контрольной точки, резервирование журналов транзакций не поддерживается при использовании простой модели восстановления.

Как следует из названия, полная модель восстановления поддерживает все варианты для создания резервных копий и восстановления базы данных. Подобно простой модели восстановления, все транзакции записываются в журнал транзакций, но они остаются в журнале транзакций и после завершения транзакции. Записи остаются в журнале транзакций до тех пор, пока не будет сделана резервная копия журнала. При выполнении резервирования журнала транзакций для базы, которая находится в режиме полного восстановления, записи журнала записываются в бэкап журнала транзакций, и записи завершенных транзакций удаляются из журнала транзакций.

Модель восстановления с неполным протоколированием минимизирует использование пространства журнала транзакций при операциях с неполным протоколированием, подобных BULK INSERT, SELECT INTO или CREATE INDEX. Функцилнальность этой модели подобна полной модели восстановления за исключением того, что записи в журнал транзакции минимально протоколируются при выполнении указанных операций. Минимальное протоколирование помогает поддерживать журнал меньших размеров, но при этом журнализируется не так много информации

26.  Расскажите про резервное копирование баз данных, восстановление баз данных

Резервное копирование — процесс создания копии данных на носителе, предназначенном для восстановления данных в оригинальном или новом месте их расположения в случае их повреждения или разрушения

Резервная копия в MSSMS

После подключения к соответствующему экземпляру Microsoft Компонент SQL Server Database Engine в обозревателе объектов разверните дерево сервера.

Разверните узел Базы данных и выберите пользовательскую базу данных или разверните узел Системные базы данных и выберите системную базу данных.

Щелкните правой кнопкой мыши базу данных, резервную копию которой нужно создать, наведите указатель на пункт Задачи и выберите команду Создать резервную копию... .

В диалоговом окне Резервное копирование базы данных выбранная база данных приводится в раскрывающемся списке (ее можно изменить на любую другую базу данных на сервере).

В раскрывающемся списке Тип резервной копии выберите нужный тип. По умолчанию выбран тип Полный.

В разделе Компонент резервного копирования выберите База данных.

В разделе Назначение проверьте расположение по умолчанию для файла резервной копии (в папке ../mssql/data).

(Необязательно) Просмотрите другие доступные параметры на страницах Параметры носителя и Параметры резервного копирования.

Нажмите кнопку ОК, чтобы запустить резервное копирование.

После успешного завершения резервного копирования нажмите кнопку ОК, чтобы закрыть диалоговое окно SQL Server Management Studio.

Восстановление (recovery) - это процесс использования устройства резервной копии для замены неподтвержденных, несогласованных или потерянных данных.

Восстановление БД

В обозревателе объектов подключитесь к экземпляру компонента Компонент SQL Server Database Engine и разверните его.

Щелкните правой кнопкой мыши узел Базы данных и выберите команду Восстановить базу данных...

Чтобы указать источник и расположение восстанавливаемых резервных наборов данных, используйте страницу Общие , раздел Источник . Выберите один из следующих вариантов.

В разделе Назначение , в поле База данных автоматически появится имя базы данных для восстановления. Для изменения имени базы данных введите новое имя в окно База данных .

В поле Восстановить до оставьте значение по умолчанию До последней выбранной резервной копии или щелкните Временную шкалу , чтобы перейти в диалоговое окно Временная шкала резервной копии и выбрать вручную конкретное пороговое время восстановления.

В сетке Резервные наборы данных для восстановления выберите нужные резервные наборы. В этой сетке отображаются резервные копии, доступные в указанном месте.

При необходимости нажмите кнопку Файлы на панели Выбор страницы и перейдите в диалоговое окно Файлы . Отсюда можно восстановить базу данных в новое расположение, определив новое место восстановления для каждого файла в сетке Восстановить файлы базы данных как .

Чтобы просмотреть или выбрать дополнительные параметры, на странице Параметры на панели Параметры восстановления вберите любой из следующих параметров, если он подходит к данной ситуации.

Нажмите кнопку ОК.

27.  Расскажите про аутентификацию и авторизацию пользователей, назначение серверных ролей и ролей баз данных, авторизацию пользователей при получении доступа к ресурсам. (Надо дополнить но я хз что от меня хотят)

Аутентификация — процедура проверки подлинности, например проверка подлинности пользователя путем сравнения введенного им пароля с паролем, сохраненным в базе данных.

Авторизация — предоставление определенному лицу или группе лиц прав на выполнение определенных действий.

Роль сервера - это набор программ, которые при правильной установке и настройке позволяют компьютеру выполнять определенную функцию для нескольких пользователей или других компьютеров в сети.

Роли базы данных — это специальные объекты, которые используются для упрощения предоставления разрешений в базах данных. В отличие от серверных ролей, которые могут быть только встроенными, роли баз данных могут быть как встроенными, так и пользовательскими. Встроенные роли баз данных обладают предопределенным набором разрешений, а пользовательские роли можно использовать для группировки пользователей при предоставлении разрешений. Обычно пользовательские роли используются только для логинов SQL Server, поскольку для группировки логинов Windows обычно удобнее и проще использовать группы Windows.