1. Назначение и возможности JDBC?

С помощью **JDBC** API можно создавать соединения с БД, выполнять SQL запросы, хранимые процедуры и обрабатывать результаты. **JDBC** API упрощает работу с базами данных из Java программ

2. Что такое JDBC драйвер? Какие бывают типы?

**JDBC** – Java DataBase Connectivity – соединение с базами данных на Java. Это стандарт взаимодействия с БД. Его интерфейс поставляется в стандартной JDK в виде пакета java.sql. Какой бы библиотекой для работы с базой данных вы ни пользовались, в ее основе почти всегда лежит низкоуровневый JDBC.  
  
**JDBC-драйвер** – реализация JDBC для определенной базы данных. В приложении может быть зарегистрировано несколько разных драйверов. При соединении к базе, нужный выбирается исходя из URL соединения. Например, для jdbc:mysql://localhost:3306/db\_name будет использован MySQL JDBC драйвер (при наличии). Драйверы делятся на 4 типа:  
  
**Тип 1 – JDBC-ODBC мост.** Делегирует работу с базой установленному в системе ODBC-драйверу. Платформозависимый. Не поддерживается с Java 8.  
  
**Тип 2 – Native API.** Делегирует работу с базой библиотеке, установленной в системе. Платформозависимый. Библиотеки в отличие от ODBC специфичны для конкретной базы, поэтому такой драйвер обычно работает быстрее.  
  
**Тип 3 – драйвер сетевого протокола.** Работает с промежуточным слоем (JavaEE сервером приложений), который транслирует запросы в сетевой протокол, с которым работает конкретная БД. Такой драйвер полностью реализован на Java.  
  
**Тип 4 – драйвер протокола БД/тонкий драйвер.** Полная реализация протокола взаимодействия с базой данных. Отличается от типа 3 тем, что здесь логика протокола находится не на внешнем промежуточном слое, а прямо в самом драйвере.

3. Какие шаги (перечислите) нужно выполнить для извлечения данных?

Следенить, запрос куда извлечь.

4. Как и где задать параметры соединения?

Connection connection = DriverManager.*getConnection*(connectionUrl,userName,password)

5. Перечислите основные JDBC API компоненты.

1. DriverManager:
   * Это класс, использующийся для управления списком Driver (database drivers).
2. *Driver*:
   * Это интерфейс, использующийся для соединения коммуникации с базой данных, управления коммуникации с базой данных. Когда загружается Driver, программисту не нужно конкретно вызывать его.
3. *Connection*:
   * Интерфейс со всеми методами связи с базой данных. Он описывает коммуникационный контекст. Вся связь с базой данных осуществляется только через объект соединения (connection).
4. *Statement*:
   * Это интерфейс, включающий команду SQL отправленный в базу данных для анализа, обобщения, планирования и выполнения.

*ResultSet :*

* + *ResultSet* представляет набор записей, извлеченных из-за выполнения запроса.

6. Какая разница между еxecute, executeQuery, executeUpdate?

Метод executeQuery  
Оператор, который производит один набор результатов, такой как оператор SELECT. Наиболее часто используемый метод для выполнения операторов SQL - executeQuery. Этот метод используется для выполнения оператора SELECT, который является почти наиболее используемым оператором SQL.

Метод executeUpdate  
Используется для выполнения операторов INSERT, UPDATE или DELETE и операторов SQL DDL (язык определения данных), таких как CREATE TABLE и DROP TABLE.

Метод execute следует использовать только в том случае, если оператор может вернуть несколько объектов ResultSet, несколько счетчиков обновлений или комбинацию объектов ResultSet и счетчиков обновлений. При выполнении хранимой процедуры или динамическом выполнении неизвестной строки SQL (т. Е. Программист приложения неизвестен во время компиляции), могут появиться несколько результатов, хотя это редко.

7. Расскажите об интерфейсе ResultSet.

Для получения результата

8. Как и для чего используют PreparedStatement?

**PreparedStatement используется** для выполнения SQL-запросов с или без входных параметров; добавляет методы управления входными параметрами.

9. Как и для чего используют транзакции?

**Транзакцию** дают нам возможность контролировать когда и где сохранять изменения в БД. Благодаря этому мы, например, можем объединить группу **SQL** – запросов в одну логическую группу и, если один из запросов не пройдёт – мы отменяем всю **транзакцию**.

10.Перечислите и поясните типы чтения данных в транзакциях. Что такое уровни изоляции?

Read uncommitted

Уровень, имеющий самую плохую согласованность данных, но самую высокую скорость выполнения транзакций. Название уровня говорит само за себя — каждая транзакция видит незафиксированные изменения другой транзакции (феномен **грязного чтения**).

Read committed

Для этого уровня параллельно исполняющиеся транзакции видят только зафиксированные изменения из других транзакций. Таким образом, данный уровень обеспечивает защиту от **грязного чтения**.

Repeatable read

Уровень, позволяющий предотвратить феномен **неповторяющегося чтения**. Т.е. мы не видим в исполняющейся транзакции измененные и удаленные записи другой транзакцией. Но все еще видим вставленные записи из другой транзакции. **Чтение фантомов** никуда не уходит.

Serializable

Уровень, при котором транзакции ведут себя как будто ничего более не существует, никакого влияния друг на друга нет. В классическом представлении этот уровень избавляет от эффекта **чтения фантомов**.

11.Что такое DatabaseMetaData интерфейс

**Интерфейс DatabaseMetaData**

\* и содержит методы, описывающие таблицы базы данных, поддержку SQL, хранимые процедуры и другие сведения, относящиеся к базе данных и текущему Connection, которые не относятся непосредственно к выполнению команд и извлечению результирующих наборов.

12.Зачем используют шаблон DAO?

**DAO** абстрагирует сущности системы и делает их отображение на БД, определяет общие методы использования соединения, его получение, закрытие и (или) возвращение в Connection Pool