**27. Explain Spring JDBC API and its classes.**

* **Spring JDBC:** Spring provides a simple way in the form of a JDBC abstraction layer to establish a bridge between database and application. It reduces boilerplate code and configurations.
* **Key classes:**
  + JdbcTemplate: Provides simple methods for executing SQL statements and working with data exchange for applications.
  + DataSource: Establish the connection(bridge) of data exchange from database.
  + SimpleJdbcCall: method present in Spring JDBC API, used for interacting with database-stored procedures.

**28. What are the advantages of JdbcTemplate in Spring?**

* **Reduces boilerplate code:** no need to write raw JDBC codes, also bundles common operations.
* **Exception handling:** auto handling and conversion of SQLExceptions into Spring’s DataAccessException.
* **Prepared statements:** Uses prepared statements to prevent SQL injection attacks.
* **Data binding:** instead of SQL statements it uses prepared statements, which have better-
  + Security – prevents SQL injection attacks
  + Performance – improved query performance

**29. Fetching records using Spring JdbcTemplate?**

Use the query method of JdbcTemplate with the appropriate SQL query and result extractor.

List<User> users = jdbcTemplate.query("SELECT \* FROM users", new BeanPropertyRowMapper<>(User.class));

This code snippet fetches all users from the user’s table and maps them to User objects using the BeanPropertyRowMapper.

### What are the advantages of JdbcTemplate in spring?

**Less code**: By using the JdbcTemplate class, you don't need to create connection,statement,start transaction,commit transaction and close connection to execute different queries. You can execute the query directly.

Let's see the methods of spring JdbcTemplate class.

|  |  |  |
| --- | --- | --- |
| **No.** | **Method** | **Description** |
| 1) | public int update(String query) | is used to insert, update and delete records. |
| 2) | public int update(String query,Object... args) | is used to insert, update and delete records using PreparedStatement using given arguments. |
| 3) | public void execute(String query) | is used to execute DDL query. |
| 4) | public T execute(String sql, PreparedStatementCallback action) | executes the query by using PreparedStatement callback. |
| 5) | public T query(String sql, ResultSetExtractor rse) | is used to fetch records using ResultSetExtractor. |
| 6) | public List query(String sql, RowMapper rse) | is used to fetch records using RowMapper. |

How can you fetch records by spring JdbcTemplate?

You can fetch records from the database by the **query method of JdbcTemplate**. There are two interfaces to do this:

1. [ResultSetExtractor](https://www.javatpoint.com/ResultSetExtractor-example)
2. [RowMapper](https://www.javatpoint.com/RowMapper-example)

### What is the advantage of NamedParameterJdbcTemplate?

NamedParameterJdbcTemplate class is used to pass value to the named parameter. A named parameter is better than ? (question mark of PreparedStatement).

### What is the advantage of SimpleJdbcTemplate?

The **SimpleJdbcTemplate** supports the feature of var-args and autoboxing.

What are the transaction management supports provided by spring?

Spring framework provides two type of transaction management supports:

1. **Programmatic Transaction Management**: should be used for few transaction operations.
2. **Declarative Transaction Management**: should be used for many transaction operations.

Q: explain @enabletransactionmanagement below @springboot application?

Q: @transactional on service class with attributes… readonly, isolationlevel and propagation

Q: what is transaction propagation? How do you handle it in spring?

Q: explain @modifying?

Q: how would you do a “group by” using Spirng Data JPA?