|-Internal working process if we call jdbcTemplate.execute(creator,callback)

It is looks like normal java jdbc but we are not going to manage coonections and etc.

In case of Java JDBC Connection which is the 1st and fore most thing bcz without connection we cannot do anything. Similarly in spring we have a class called JdbcTemplate which acts as centric for all the classes and to interact with DB. That it is going to get the connection from the DB using Connection class internally to interact with DB.

class StudentDao {

}

For example if we wanted to execute the SQL select query then we need to talk to the jdbcTemplate bcz it can talk to the DB so we need to create the obj of jdbcTemplate obj so that if we give SQL query to the jdbcTemplate it establishes the connection, creates the Statement, executes the Statement and iterating over the resultset. Here we are not creating the connection so JdbcTemplate need to create the connection so in order to create the connection we need to create/configure dataSource to get the connection from that dataSource. That's why we need to configure the dataSource so that jdbcTemplate will asks the dataSource to create the connection and holds with it.

Configuring the dataSource or creating the Data Source means configuring it as bean in spring bean configuration file.

In order to configure dataSource as bean we need dataSource impl class bcz javax.sql.DataSource is an interface so we cannot configure interfaces as beans bcz we/IOC cannot create obj for interfaces.

**Callback Mechanism:**

If we wanted to execute some logic of the class then we need required set of inputs to call and execute the logic which is there in the method so that it produces the output.

But if we don't have the inputs to execute it then we don't have control to execute it, that means we cannot perform the operation directly that where we need to give the logic to execute by the someone who are having the required set of inputs with them (bcz they will not give the inputs to us that's why we need to give our logic as input to them) so that if we give to our logic to them they can execute our logic and then gives back the reslut even though we don't have the reqired set of inputs which is called Callback mechanism.

In computer programming, a callback is a piece of executable code that is passed as an argument to other code, which is expected to call back (execute) the argument at some convenient time with required set of inputs which it is already having and invocation may be immediate as in a synchronous callback, or it might happen at later time as in an asynchronous callback.so that it can perform the execution and gives the results which is called call back mechanism.

Similarly Spring JDBC also will not gives the connection to us bcz if didn't give back this then it cannot perform the closing of resources. But we cannot perform the operation directly without connection that's where jdbcTemplate allows us to pass the logic as input to execute and executes with set of inputs like creating con, statement, wraps into ResultSet and gives the results back to us once it has completes the execution.

That means whenever we wanted to execute some logic and we don't have required set of objects to execute our code that's where we are giving our logic to execute by the someone (spring jdbc) so that it will call back our code/logic with required set of inputs/objects in performing the operation and gives the results back which is called as Callback mechanism.

That means we need to write our logic in class and we need to give that logic to JdbcTemplate so that it can executes it. That means we have 2-pahes.

**Phases of JdbcTemplate:**

1. Creator phase (we need to create which we need to execute and give it to the Spring JDBC)

2. Callback phase (we return this to call by the spring jdbc)

**Performing DML, DQL Operations:**

a) Working with PreparedStatement using Spring JDBC:

Example:

We wanted to execute SELECT \*FROM STUDENT; SQL query using classic approach using PreparedStatement or Statement.

**1. Creator phase:**

Creating Statement or PrepareStatement will be done in this phase to perform the operation. If we wanted to create the Statement or PrepareStatement then we need connection but we are not creating the connection but we need a connection to create the Statement or PrepareStatement that's where we need to implement form the PreparedStatementCreator and override the method which passes the con as param so that we can create the Statement or PrepareStatement obj.

class GetAllStudentsStatementCreator implements PreparedStatementCreator {

public PreparedStatement createPreparedStstemnt(Connection con) {

PreparedStatement pstmt = con.createPreparedStatement("SELECT \*FROM STUDENT");

return pstmt;

}

}

**2. Callback phase:**

Executing the Statement or PrepareStatement will be done in the Callback phase.

class GetAllStudentsStatementCallback implements

PreparedStatementCallback<List<StudentBo>> {

public List<StudentBo> doInPreparedStstemnt(PreparedStatement pstmt) {

ResultSet rs=pstmt.executeQuery();

while(rs.next()) {

StudentBo studentBo =new StudentBo();

//wrap data in to StudentBo obj

//add to student list

}

return students;

}

}

That means we are writing logic in the obj by creating the obj (either PreparedStatement or Statement) and we are passing it to the Spring JDBC which is called as Creator Pahse.

To execute or call back our logic we writing the callback phase, so we need to pass creator obj and callback obj to the spring so that it can execute our logic bcz createPrepareStatement will be called by the spring JDBC and in that method we returning the pstmt to the spring JDBC so all the resources are there with Spring JDBC so that if we pass creator obj and callback obj it executes and closes the all the resources like pstmt,con,rs as well bcz all the resources are there with spring JDBC.

The means if we create the obj's of creator and callback classes and passes to jdbcTemplate then it will takes care of getting con and creating statement by calling cretePreparedStatement() and executing the doInPreparedStateemt() by passing pstmt as param (bcz pstmt is there with jdbcTemplate) and returning result to the calee and closing all the resource takes care by the jdbcTemplate.

That means StudentDao class need to have one method to create and to pass the creator, callback obj's so that it will returns the results.

class StudentDao {

private JdbcTemplate jdbcTemplate;

public StudentDao() {

}

//inject via constructor injection

public StudentDao(JdbcTemplate jdbcTemplate) {

this.jdbcTemplate=jdbcTemplate;

}

public List<StudentBo> getAllStudents() {

return jdbcTemplate.execute(new GetAllStudentsStatementCreator(),

new GetAllStudentsStatementCallback());

}

}