Saturday, June 18, 2022 9:28 AM

VARARGS = **VAR**iable number of **ARG**ument**S**

//We can pass any number of int values to the m1 method
void m1(int... nums) } nums is a parameter/argument of type varargs int

getDeclaredMethod(String mname , Class... parameters)
//after passing the method name we can pass any number of class Class objects

____

Statement stmt = con.createStatement() ; //gives us the object of subclass of Statement

Statement is a JDBC interface
Implementation of this interface is provided in the mysql connector jar component
We are using the object

stmt . executeUpdate(sql);

DDL queries = create drop alter ,......
DML queries = Insert, Update , Delete

Java Client ------------DRIVER---------------------> Mysql
We used executeUpdate() API of Statement interface

DQL queries = select

Java Client <-----------------DRIVER-------------------MySQL
We use executeQuery() API of Statement interface
Return VALUE is important = the data coming from the table

executeQuery() API returns **ResultSet** interface subclass object

ResultSet is the table data ---

rs pointer initially ------>	Id	Name	Cost
rs.next() **------->**	1	Pendrive	200
	2	Hard disk	500
	3	Mouse	350

We can move the rs pointer to each row of the table using API rs.next()
This API returns a boolean , if there is a next row then true , if end then false

Exercise ---
Write a class study.Client2

Main

Write another class study.DAO = Data Access Object = class that contains methods for DB access!!

```
  Connection myGetConnection()
{
        Return null;
}

  ResultSet   getRows(String query )
{
         return null;
}

   void    showResultSet(Resultset  rs )
```

Statement
    executeUpdate - DML
    executeQuery -DQL

If the query contains variables --- **string query creation is difficult**
  "insert into values ("+ id+",' "+name+"',"+cost+")" ;

    Insert into values(1,'pd','viacomm'200)
    Insert into values(2,null,'viacomm',300)

To overcome this complexity -----
Instead of Statement we use another interface PreparedStatement -----------

Statement	**PreparedStatement**
Statement **stmt** = con.createStatement**()**	PreparedStatement **pstmt** = con.prepareStatement(**SQL**)

stmt.executeUpdate(**SQL 1**)
 	
stmt.executeUpdate(SQL2)
 	
stmt.executeQuery(SQL3)
 	
Using same statement object we can fire many sql queries
	pstmt.executeUpdate**()** **//no parameters**
	
	For different sql queries use different PreparedStatement objects

stmt is NOT tied up to a SQL	pstmt is tied to a SQL
Variables are concatenated in the query	Variables are put as ? In the query and later the ? Value is set
Statement query is	The prepared statement query is compiled by the DRIVER long

compiled/translated by the DRIVER when the query is executed	before query execution …just when pstmt is created --- PRE COMPILATION of query----- query execution is faster

____

JDBC Interfaces in java.sql package

Statement
PreparedStatement
CallableStatement

Callable Statement = call the stored procedures !!!

Where is the stored procedure located? DB Server side
Where is it executed? DB server side
Where is it called ? Client side

EXPLORE ---- write a stored procedure with IN and **INOUT** parameter and call it from JAVA CLIENT
EXPLORE --- write a stored function that accepts IN parameter and **returns** a VALUE call it from JAVA CLIENT

```
String sql = "{? = call  func1(? ) } ";
1st ?  Is registerOUT
2nd  ? IN parameter
```

____

Transaction Management in JDBC ---------

What is a DB transaction ???

A **set of db queries** should succeed or fail together !!!

If any query in the set fails all other succeeded queries are ROLLED BACK
If all queries in the set succeed then the result is COMMITTED

Example ----

Account transfer of amount 5000 from Account A to Account B

Sql1 - Update account set balance = balance -5000 where acctId= A;
Sql2 - Update account set balance = balance +5000 where acctId= B;

If sql1 succeeds and sql2 fails ---- ROLLBACK
If both succeed ------------COMMIT

```
con.setAutoCommit(false );  //through java

        Pstmt.executeUpdate()
        Pstmt.executeUpdate()

        Con.commit();

        } EXCEPTION if any fail
```

```
    catch
   {
          con.rollBack();
   }
```

____

PROBLEM --- SQL INJECTION =

If user passes id then show record of that id only !!!

"Select * from account where acctNo ="+ var

var = "A"   then only details of A account are shown
var = 1 or 1   then details of all accounts are disclosed !!! SQL INJECTION !!!

Sometimes Prepared Statement should be used to avoid SQL INJECTION (it can occur in statement) !!!!

____

____