RUBY DBI METHODS FOR READ OPERATION

 Copyright © tutorialspoint.com

DBI provides several different methods to fetch records from the database. Assuming **dbh** is a database handle and **sth** is a statement handle:

S.N.	Methods with Description
1	**db.select_one***stmt, * bindvars* => aRow | nil** Executes the *stmt* statement with the *bindvars* binding beforehand to parameter markers. Returns the first row or *nil* if the result-set is empty.
2	**db.select_all** ***stmt, * bindvars* => [aRow, ...] | nil** **db.select_all***stmt, * bindvars*{ |aRow| aBlock }** Executes the *stmt* statement with the *bindvars* binding beforehand to parameter markers. Calling this method without block returns an array containing all rows. If a block is given, this will be called for each row.
3	**sth.fetch => aRow | nil** Returns the *next* row. Returns *nil* if no further rows are in the result-set.
4	**sth.fetch { |aRow| aBlock }** Invokes the given block for the remaining rows of the result-set.
5	**sth.fetch_all => [aRow, ...]** Returns all remaining rows of the result-set collected in an array.
6	**sth.fetch_many***count* **=> [aRow, ...]** Returns the next *count* rows collected in an [aRow, ...] array.
7	**sth.fetch_scroll***direction, offset* = 1 **=> aRow | nil** Returns the row specified by the *direction* parameter and *offset*. Parameter *offset* is discarded for all but SQL_FETCH_ABSOLUTE and SQL_FETCH_RELATIVE. See a table below for possible values of *direction* parameter.
8	**sth.column_names => anArray** Returns the names of the columns.
9	**column_info => [aColumnInfo, ...]** Returns an array of DBI::ColumnInfo objects. Each object stores information about one column and contains its name, type, precision and more.
10	**sth.rows => rpc** Returns the Row Processed *Count* of the executed statement or *nil* if no such exist.

11 **sth.fetchable? => true | false**

Returns *true* if it's possible to fetch rows, otherwise *false*.

12 **sth.cancel**

Frees the resources held by the result-set. After calling this method, it is no longer possible to fetch rows until you again call *execute*.

13 **sth.finish**

Frees the resources held by the prepared statement. After calling this method no further methods can be called onto this object.

The direction Parameter:

Following values could be used for the direction Parameter of the *fetch_scroll* Method:

Constant	Description
DBI::SQL_FETCH_FIRST	Fetch first row.
DBI::SQL_FETCH_LAST	Fetch last row.
DBI::SQL_FETCH_NEXT	Fetch next row.
DBI::SQL_FETCH_PRIOR	Fetch previous row.
DBI::SQL_FETCH_ABSOLUTE	Fetch row at position offset.
DBI::SQL_FETCH_RELATIVE	Fetch the row that is offset rows away from the current.

Example:

The following example shows how to get metadata for a statement. Consider EMPLOYEE table, which we created in last chapter.

```
#!/usr/bin/ruby -w

require "dbi"

begin
     # connect to the MySQL server
     dbh = DBI.connect("DBI:Mysql:TESTDB:localhost",
                       "testuser", "test123")
     sth = dbh.prepare("SELECT * FROM EMPLOYEE
                       WHERE INCOME > ?")
     sth.execute(1000)
     if sth.column_names.size == 0 then
        puts "Statement has no result set"
        printf "Number of rows affected: %d\n", sth.rows
     else
        puts "Statement has a result set"
        rows = sth.fetch_all
        printf "Number of rows: %d\n", rows.size
        printf "Number of columns: %d\n", sth.column_names.size
        sth.column_info.each_with_index do |info, i|
          printf "--- Column %d (%s) ---\n", i, info["name"]
          printf "sql_type:         %s\n", info["sql_type"]
          printf "type_name:        %s\n", info["type_name"]
```

```
          printf "precision:        %s\n", info["precision"]
          printf "scale:            %s\n", info["scale"]
          printf "nullable:         %s\n", info["nullable"]
          printf "indexed:          %s\n", info["indexed"]
          printf "primary:          %s\n", info["primary"]
          printf "unique:           %s\n", info["unique"]
          printf "mysql_type:       %s\n", info["mysql_type"]
          printf "mysql_type_name:  %s\n", info["mysql_type_name"]
          printf "mysql_length:     %s\n", info["mysql_length"]
          printf "mysql_max_length: %s\n", info["mysql_max_length"]
          printf "mysql_flags:      %s\n", info["mysql_flags"]
      end
   end
   sth.finish
rescue DBI::DatabaseError => e
     puts "An error occurred"
     puts "Error code:    #{e.err}"
     puts "Error message: #{e.errstr}"
ensure
     # disconnect from server
     dbh.disconnect if dbh
end
```

This will produce the following result:

```
Statement has a result set
Number of rows: 5
Number of columns: 5
--- Column 0 (FIRST_NAME) ---
sql_type:         12
type_name:        VARCHAR
precision:        20
scale:            0
nullable:         true
indexed:          false
primary:          false
unique:           false
mysql_type:       254
mysql_type_name:  VARCHAR
mysql_length:     20
mysql_max_length: 4
mysql_flags:      0
--- Column 1 (LAST_NAME) ---
sql_type:         12
type_name:        VARCHAR
precision:        20
scale:            0
nullable:         true
indexed:          false
primary:          false
unique:           false
mysql_type:       254
mysql_type_name:  VARCHAR
mysql_length:     20
mysql_max_length: 5
mysql_flags:      0
--- Column 2 (AGE) ---
sql_type:         4
type_name:        INTEGER
precision:        11
scale:            0
nullable:         true
indexed:          false
primary:          false
unique:           false
mysql_type:       3
mysql_type_name:  INT
mysql_length:     11
mysql_max_length: 2
```

```
mysql_flags:      32768
--- Column 3 (SEX) ---
sql_type:         12
type_name:        VARCHAR
precision:        1
scale:            0
nullable:         true
indexed:          false
primary:          false
unique:           false
mysql_type:       254
mysql_type_name:  VARCHAR
mysql_length:     1
mysql_max_length: 1
mysql_flags:      0
--- Column 4 (INCOME) ---
sql_type:         6
type_name:        FLOAT
precision:        12
scale:            31
nullable:         true
indexed:          false
primary:          false
unique:           false
mysql_type:       4
mysql_type_name:  FLOAT
mysql_length:     12
mysql_max_length: 4
mysql_flags:      32768
```

Loading [MathJax]/jax/output/HTML-CSS/fonts/TeX/fontdata.js