Os comandos de query do SQL, retornam informações que estão nas tabelas do banco de dados, o comando utilizado é o SELECT.

Assim como os comandos de update, antes de enviar um comando de query é preciso envolve-lo em um objeto do tipo **Statement**, que posteriormente será encaminhado diretamente ao banco de dados, ou ao driver de conexão do fornecedor conforme a arquitetura de conexão.

A sintaxe para criar um statement que envia comandos SQL de query ao servidor do banco de dados:

Sintaxe:

```
Statement <objeto> = <conexão>.createStatement();
```

Exemplo:

```
Statement stmt = con.createStatement();
```

Executar a query

Sintaxe:

```
ResultSet <resultado da query> = <obj de Statement>.executeQuery(<Query>);
```

O resultado de um statement do tipo query é um objeto do tipo **ResultSet** contendo os registros que foram retornados pelo comando de query.

Exemplo:

```
ResultSet rs = stmt.executeQuery( "SELECT * FROM animal" );
```

A subclasse **PreparedStatement**, pode envolver comandos de SQL do tipo query também., e como vantagem permite a substituição de parâmetros da query antes de executá-la, conforme exemplo a seguir.

Exemplo:

```
String comandoSQL = "SELECT * FROM animal WHERE especie=?";
PreparedStatement stmt = con.prepareStatement( comandoSQL );
stmt.setString(1, "cachorro");
```

Nota: Os parâmetros são representados através de um sinal de interrogação

Executar a query

Sintaxe:

```
ResultSet <resultado da query> = <obj PreparedStatement>.executeQuery();
```

O resultado do método **executeQuery** é um objeto do tipo **ResultSet** contendo os registros que foram retornados pelo comando de query.

Exemplo:

```
ResultSet res = stmt.executeQuery();
```

O objeto **ResultSet** possui uma estrutura com os dados da consulta, e através de um padrão chamado **Iterator** é possível buscar as informações nesta estrutura, sem necessariamente conhecer como estes dados estão organizados.

Como exemplo será feita uma consulta na tabela **animal** contendo os campos abaixo:

Field *	Type * ▲	Null *	Key *
id	bigint(20)	NO	PRI
nascimento	datetime	YES	
peso	float	YES	
sexo	varchar(10)	YES	
nome	varchar(100)	YES	
nome_dono	varchar(100)	YES	
especie	varchar(50)	YES	
raca	varchar(50)	YES	

A consulta baseada na query a seguir, retornará os dados conforme o quadro abaixo:

```
String comandoSQL = "SELECT * FROM animal WHERE especie=?";
PreparedStatement stmt = con.prepareStatement( comandoSQL );
stmt.setString(1, "cachorro");
ResultSet rs = stmt.executeQuery();
```

🧠 id *	especie	nascimento	nome	nome_dono	peso	raca	sexo
1	Cachorro	04/11/2015 22:09:33	Rex	Antonio	12	Collie 🔻	masculino
2	Cachorro	04/11/2015 22:09:48	Rex	Antonio	12	Collie	masculino
3	cachorro	03/03/2012 21:00:00	Principe	Cintia	4,5	Beagle	macho
4	cachorro	04/11/2015 22:00:00	Toto	Antonio	7,4	Vira Lata	macho

Os dados do **ResultSet** podem ser resgatados através de um cursor, que se move conforme os métodos **next()**, **previous()**, **first()** e **last()** disponíveis no objeto do tipo **ResultSet**

first() – Encaminha o cursor para o primeiro registro existente no ResultSet e retorna true, caso não seja possível apontar para o primeiro registro o método retornará false.

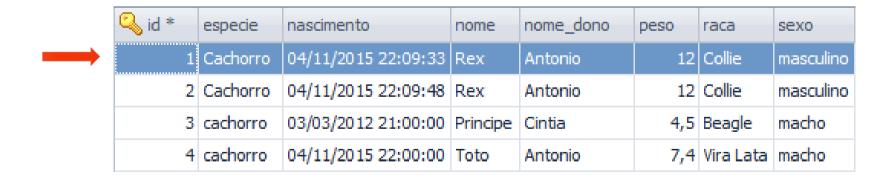
last() – Encaminha o cursor para o último registro existente no ResultSet e retorna true, caso não seja possível apontar para o último registro o método retornará false.

next() – Encaminha o cursor para o próximo registro existente no ResultSet e retorna true, caso não seja possível apontar para o próximo registro o método retornará false.

previous() – Encaminha o cursor para o registro anterior existente no ResultSet e retorna true, caso não seja possível apontar para o registro anterior o método retornará false.

Inicialmente o cursor do ResultSet não aponta para nenhum registro. Após a execução do método first() ou next() o cursor apontará para o primeiro registro.

```
rs.first();
```



Uma vez com o cursor em algum registro, é possível buscar as informações de um campo através dos métodos getString(), getInt(), getFloat(), getLong(), getDate() e outros, conforme o tipo do campo na tabela.

A informação de qual campo será resgatado pode ser identificado por um parâmetro como o nome do campo ou um número que indique a ordem.



🥝 id *	especie	nascimento	nome	nome_dono	peso	raca	sexo
1	Cachorro	04/11/2015 22:09:33	Rex	Antonio	12	Collie	masculino
2	Cachorro	04/11/2015 22:09:48	Rex	Antonio	12	Collie	masculino
3	cachorro	03/03/2012 21:00:00	Principe	Cintia	4,5	Beagle	macho
4	cachorro	04/11/2015 22:00:00	Toto	Antonio	7,4	Vira Lata	macho

```
Para navegar por todos os registros faça um loop do tipo while conforme abaixo:
while( rs.next() == true ) {
    String n = rs.getString("nome");
    System.out.println( n );
}
```