


Exercise 6: read data from database

 ex06/_Table_read.txt in the metacode project map

execute the code generation after each step and watch the result in the destination project

In this example we created first some order.

the primary key classes go in the map tablepk, and we loose the prefix "ex06"

same goes for the table classes, and we put them in the map tables

in _Table_read, we look at reading data from a SQL query result and put that in our table class.

we introduce these metacode

- getcolumndbfunction
- metacoder_date
- //Custom code, do not change this line

first we define again our class

for documentation purposes, we can add the data this file is last generated

```
//generated at :metacoder_date:

CLASS :Table:read.txt
```

we can put custom variables and functions that will not be overwritten next time the metacoder updates the file. To test this out, run the generator, put some text between these lines, and run the generator again. Your changes will still be there, while the other code will be updated according to your template or database changes

```
//Custom code, do not change this line
//Custom code, do not change this line
```

for this example we just need 1 function, readrecord, with a parameter SQLRESULT representing data coming from a database query.

```
FUNCTION :Table: readrecord(SQLRESULT sqlresultset)
{
}
}
```

first, we create the primary key and assign it to the table result

getcolumndbfunction is for each field type defined in the language definition txt.xml

```
:Table:PK :table:PK = new :Table:PK(:repeatpkfields::columncast:
sqlresultset::getcolumndbfunction::,::repeatpkfields:)
:Table: :table: = new :Table:(:table:PK)
```

we add the foreign keys

```

:repeatforeignkeys:
:notpk:
    :table:.set:Uniquename:PK(new :Pktable:PK(:repeatforeignkeyfields:
sqlresultset.:getcolumnndbfunction::,:repeatforeignkeyfields:))
:notpk:
:repeatforeignkeys:

```

and last we add the remaining fields

```

:repeatfields:
:iffieldtype:
:java.lang.Object:
    :table:.set:Column:(sqlresultset.getBytes(":column_o:"))
:java.lang.Object:
:other:
    :table:.set:Column:(sqlresultset.:getcolumnndbfunction:)
:other:
:iffieldtype:
:repeatfields:

```

The complete file looks like this

```

//generated at :codegenerator_date:

CLASS :Table:read.txt

//Custom code, do not change this line
//Custom code, do not change this line

    FUNCTION :Table: readrecord(SQLRESULT sqlresultset)
    {
        :Table:PK :table:PK = new :Table:PK(:repeatpkfields:sqlresultset.:
getcolumnndbfunction::,:repeatpkfields:)
        :Table: :table: = new :Table:(:table:PK)
:repeatforeignkeys:
:notpk:
        :table:.set:Uniquename:PK(new :Pktable:PK(:repeatforeignkeyfields:
sqlresultset.:getcolumnndbfunction::,:repeatforeignkeyfields:))
:notpk:
:repeatforeignkeys:
:repeatfields:
:iffielddtype:
:java.lang.Object:
        :table:.set:Column:(sqlresultset.getBytes(":column_o:"))
:java.lang.Object:
:other:
        :table:.set:Column:(sqlresultset.:getcolumnndbfunction:)
:other:
:iffielddtype:
:repeatfields:
        RETURN :table:
    }
}

```