

Article

[Danny Wijnschenk](#) · Jul 19, 2022 4m read

Caution with Mixing OO and SQL

Mixing Object syntax with SQL is one of the nice features in Object Script. But in one case, it gave strange results, so I decided to isolate the case and describe it here.

Let's say you need to write a classmethod that updates a single property on disk. Usually, i would write that using SQL like this :

```
ClassMethod ActivateSQL(customerId) as %Status
{
    &sql(Update Test.Customer Set Active=1 Where ID=:customerId)
    If SQLCODE'=0 {
        Set exception = ##class(%Exception.SQL).CreateFromSQLCODE(SQLCODE, $Get(%msg))
        Quit exception.AsStatus()
    } Else {
        Quit $$$OK
    }
}
```

and call this classmethod wherever i need to in my application.

But if the application code has the instance opened when this classmethod is called, and is doing a %Save afterwards, it will overwrite the updates that happened in the classmethod :

```
Set objCust=##class(Test.Customer).%OpenId(id)
Do objCust.ActivateSQL(id)
Set objCust.Name = "something"
Set sc = objCust.%Save()
```

By changing the order of the lines, the problem would be solved, but you should be very carefull with this kind of mix :

```
Do ##class(Test.Customer).ActivateSQL(id)
Set objCust=##class(Test.Customer).%OpenId(id)
Set objCust.Name = "something"
Set sc = objCust.%Save()
```

When the classmethod would be written using OO syntax like this :

```
ClassMethod ActivateOO(customerId) as %Status
{
    Set objCust = ##class(Test.Customer).%OpenId(customerId)
    Set objCust.Active = 1
    Quit objCust.%Save()
}
```

there would not be a problem since the open instance in the calling code and the opened instance in the classmethod would point to the same instance in memory.

(Besides a performance penalty since opening an instance with lots of properties to just update one property is slower than a SQL update)

So as a conclusion : beware of opening instances 'too long' along your code if you are using also SQL.

I have attached the full test class in case you want to see it for yourself, call `Do ##class(Test.Customer).Test(0)` to see the code using only OO, and `.Test(1)` with using the SQL (and see that the SQL update is overwritten)
Any comments are appreciated !

```
Class Test.Customer Extends %Persistent
{

Property Name As %String;
Property Active As %Boolean;
ClassMethod ActivateSQL(customerId) As %Status
{
    #Dim exception

    &sql(Update Test.Customer Set Active=1 Where ID=:customerId)
    If SQLCODE'=0 {
        Set exception = ##class(%Exception.SQL).CreateFromSQLCODE(SQLCODE, $Get(%msg))
        Quit exception.AsStatus()
    }

    &sql(Select Name, Active Into :name, :active From Test.Customer Where ID
    = :customerId)
    Write !,"Result After SQL Update : ",!
    Write "Name    : ",name,!
    Write "Active  : ",active,!
    Quit
}

ClassMethod ActivateOO(customerId) As %Status
{
    #Dim objCust as Test.Customer
    #Dim sc as %Status
    Set objCust = ##class(Test.Customer).%OpenId(customerId)
    Set objCust.Active = 1
    Set sc = objCust.%Save()
    If sc'=$$$OK Quit sc
    &sql(Select Name, Active Into :name, :active From Test.Customer Where ID
    = :customerId)
    Write !,"Result After %Save : ",!
    Write "Name    : ",objCust.Name,!
    Write "Active  : ",objCust.Active,!
    Quit
}

ClassMethod Test(mode = 0)
{
    #Dim objCust as Test.Customer
    #Dim sc as %Status
    #Dim id as %Integer
    ;Create an instance and keep the id in memory
    Set objCust = ##class(Test.Customer).%New()
    Set objCust.Name = "Danny"
```

```
Set sc = objCust.%Save() If sc'=1 Write "Could not save",!
Set id = objCust.%Id()
Kill objCust

;Open and display the created instance
Set objCust=##class(Test.Customer).%OpenId(id)
Write "Name    : ",objCust.Name,!
Write "Active  : ",objCust.Active,!

;Call a classmethod that updates the id with SQL or OO
If mode=0 {
  Do objCust.ActivateOO(id)
} else {
  Do objCust.ActivateSQL(id)
}
;Change the instance (that is still in memory)
Set objCust = ##class(Test.Customer).%OpenId(id)
Set objCust.Name = objCust.Name_ - edited"
Set sc = objCust.%Save() If sc'=1 Write "Could not save",!
Write "Name    : ",objCust.Name,!
Write "Active  : ",objCust.Active,!
;the sql update in the classmethod is overwritten with the instance that was still i
n memory
;Open and display the created instance
Kill objCust
Set objCust = ##class(Test.Customer).%OpenId(id)
Write "Name    : ",objCust.Name,!
Write "Active  : ",objCust.Active,!
}
}
```

[#Coding Guidelines](#) [#ObjectScript](#) [#Caché](#)

Source URL: <https://community.intersystems.com/post/caution-mixing-oo-and-sql>