Effective date:

Environment – can define the server and JDBC

def sdf = new java.text.SimpleDateFormat("yyyy-MM-dd")

return sdf.format( new Date()+1)

Expiration date:

def sdf = new java.text.SimpleDateFormat("yyyy-MM-dd")

return sdf.format( new Date()+1)

------------------------------------------------------

Generate Request XML Groovy script:

log.info "soapUI script"

def buildLimit(coverage,limit){

def StringBuilder limitXML = new StringBuilder()

if(coverage.contains('/') && !coverage.equalsIgnoreCase('N/A')){

def (perPerson, perOcc) = coverage.split('/')

limitXML.append( "<Limit><LimitAppliesToCd>PerPerson</LimitAppliesToCd><FormatCurrencyAmt><Amt>${perPerson}</Amt></FormatCurrencyAmt></Limit>")

limitXML.append("<Limit><LimitAppliesToCd>PerOcc</LimitAppliesToCd><FormatCurrencyAmt><Amt>${perOcc}</Amt></FormatCurrencyAmt></Limit>")

}else if (coverage.isNumber()){

limitXML.append( "<Limit><FormatCurrencyAmt><Amt>${coverage}</Amt></FormatCurrencyAmt></Limit>")

}else {

limitXML.append( "<Limit><FormatText>${coverage}</FormatText></Limit>")

}

testRunner.testCase.testSteps["Properties"].setPropertyValue("${limit}", limitXML.toString())

log.info testRunner.testCase.testSteps["Properties"].getPropertyValue("${limit}")

}

def buildRentalLimit(coverage,limit){

def StringBuilder limitXML = new StringBuilder()

if(coverage.contains('/') && !coverage.equalsIgnoreCase('N/A')){

def (perDay, maxAmount) = coverage.split('/')

limitXML.append( "<Limit><LimitAppliesToCd>PerDay</LimitAppliesToCd><FormatCurrencyAmt><Amt>${perDay}</Amt></FormatCurrencyAmt></Limit>")

limitXML.append("<Limit><LimitAppliesToCd>MaxAmount</LimitAppliesToCd><FormatCurrencyAmt><Amt>${maxAmount}</Amt></FormatCurrencyAmt></Limit>")

}else {

limitXML.append( "<Limit><FormatText>${coverage}</FormatText></Limit>")

}

testRunner.testCase.testSteps["Properties"].setPropertyValue("${limit}", limitXML.toString())

log.info testRunner.testCase.testSteps["Properties"].getPropertyValue("${limit}")

}

def buildWLBLimit(coverage,limit){

def StringBuilder limitXML = new StringBuilder()

if(coverage.contains('/') && !coverage.equalsIgnoreCase('N/A')){

def (monthly, maxAmount) = coverage.split('/')

limitXML.append( "<Limit><LimitAppliesToCd>Monthly</LimitAppliesToCd><FormatCurrencyAmt><Amt>${monthly}</Amt></FormatCurrencyAmt></Limit>")

limitXML.append("<Limit><LimitAppliesToCd>MaxAmount</LimitAppliesToCd><FormatCurrencyAmt><Amt>${maxAmount}</Amt></FormatCurrencyAmt></Limit>")

}else {

limitXML.append( "<Limit><FormatText>${coverage}</FormatText></Limit>")

}

testRunner.testCase.testSteps["Properties"].setPropertyValue("${limit}", limitXML.toString())

log.info testRunner.testCase.testSteps["Properties"].getPropertyValue("${limit}")

}

def buildDeductible(coverage,deductible){

def StringBuilder deductibleXML = new StringBuilder()

if (coverage.isNumber() && !coverage.equalsIgnoreCase('N/A')){

deductibleXML.append( "<Deductible><FormatCurrencyAmt><Amt>${coverage}</Amt></FormatCurrencyAmt></Deductible>")

}else {

deductibleXML.append( "<Deductible><FormatText>${coverage}</FormatText></Deductible>")

}

testRunner.testCase.testSteps["Properties"].setPropertyValue("${deductible}", deductibleXML.toString())

log.info testRunner.testCase.testSteps["Properties"].getPropertyValue("${deductible}")

}

buildLimit(context.expand('${DataSource#BI}'),"LimitBI")

buildLimit(context.expand('${DataSource#UM}'),"LimitUM")

buildLimit(context.expand('${DataSource#UNDUM}'),"LimitUNDUM")

buildLimit(context.expand('${DataSource#MEDPM}'),"LimitMEDPM")

buildLimit(context.expand('${DataSource#PD}'),"LimitPD")

buildLimit(context.expand('${DataSource#Towing}'),"LimitTowing")

buildRentalLimit(context.expand('${DataSource#Rental}'),"LimitRental")

buildDeductible(context.expand('${DataSource#COMP}'),"DeductibleCOMP")

buildDeductible(context.expand('${DataSource#COLL}'),"DeductibleCOLL")

declare @results xml

set @results = (

select AFIRequestXML

from CREEQtransactions (nolock)

where 1=1

and requesttype='quote'

and affinitygroup like'%INSURIFY\_RCX\_QUOTE%'

and GUID=:GUID)

SELECT @results.query('/CDM\_Data/Account/Vehicle[1]/CompDed/text()')

SELECT @results.query('/CDM\_Data/Account/Vehicle[1]/CollDed/text()')

SELECT @results.query('/CDM\_Data/Account/Vehicle[1]/TowingLabor/text()')

SELECT @results.query('/CDM\_Data/Account/Vehicle[1]/RentalReimbursement/text()')

SELECT @results.query('/CDM\_Data/Account/AutoPolicy/Coverage/BodilyInjury/text()')

SELECT @results.query('/CDM\_Data/Account/AutoPolicy/Coverage/PropertyDamage/text()')

SELECT @results.query('/CDM\_Data/Account/AutoPolicy/Coverage/UMBI/text()')

SELECT @results.query('/CDM\_Data/Account/AutoPolicy/Coverage/UIMBI/text()')

SELECT @results.query('/CDM\_Data/Account/AutoPolicy/Coverage/MedPmts/text()')

SELECT @results.query('/CDM\_Data/Account/Vehicle[1]/RideshareVeh/text()')







