INTRODUCTION

This document describes how to adapt the Governance Dashboard when the QVPR is hosted in a Database. Customized Governance Dashboard is not Supported by Qlik, it is suggested to engage a developer to review the document and make the necessary changes.

DICLAIMER

! *The information in this article is provided as-is and to be used at own discretion. Ongoing support on the solution is not provided by Qlik Support.*

GOVERNANCE DASHBOARD CONNECTED TO SQL SERVER.

This document describes how to adapt the Governance Dashboard to read from SQL server.

The original Governance Dashboard has a subroutine called **`publisherScan`** that is the method that calls other subroutines to get the QDS information. The subroutine called `qvprScan` reads from the XML files that most of the customers use for their QVPR.

In order to adapt the dashboard, it is necessary to make 3 basic changes:

1. Create a new subroutine to read from the SQL Server tables, this includes connection to the Database and SQL queries to get the QVPR information.
2. To comment out the fraction of the code where the `qvprScan` is called.

The details are described below:

CREATE A NEW SUBROUTINE TO READ FROM THE SQL SERVER TABLES, THIS INCLUDES CONNECTION TO THE DATABASE AND SQL QUERIES TO GET THE QVPR INFORMATION.

The following code was just adapted to include the queries to the DB instead of reading XML files, there were no changes to any function or value, if the official Governance Dashboard changes in the future, this part needs to be adapted accordingly.

SQLQVPR SUBRUTINE

```
SUB sqlQVPR

ODBC CONNECT TO SQLQVPR;

LET vSQLDB='QVPR';
LET vDBO= 'dbo';
```

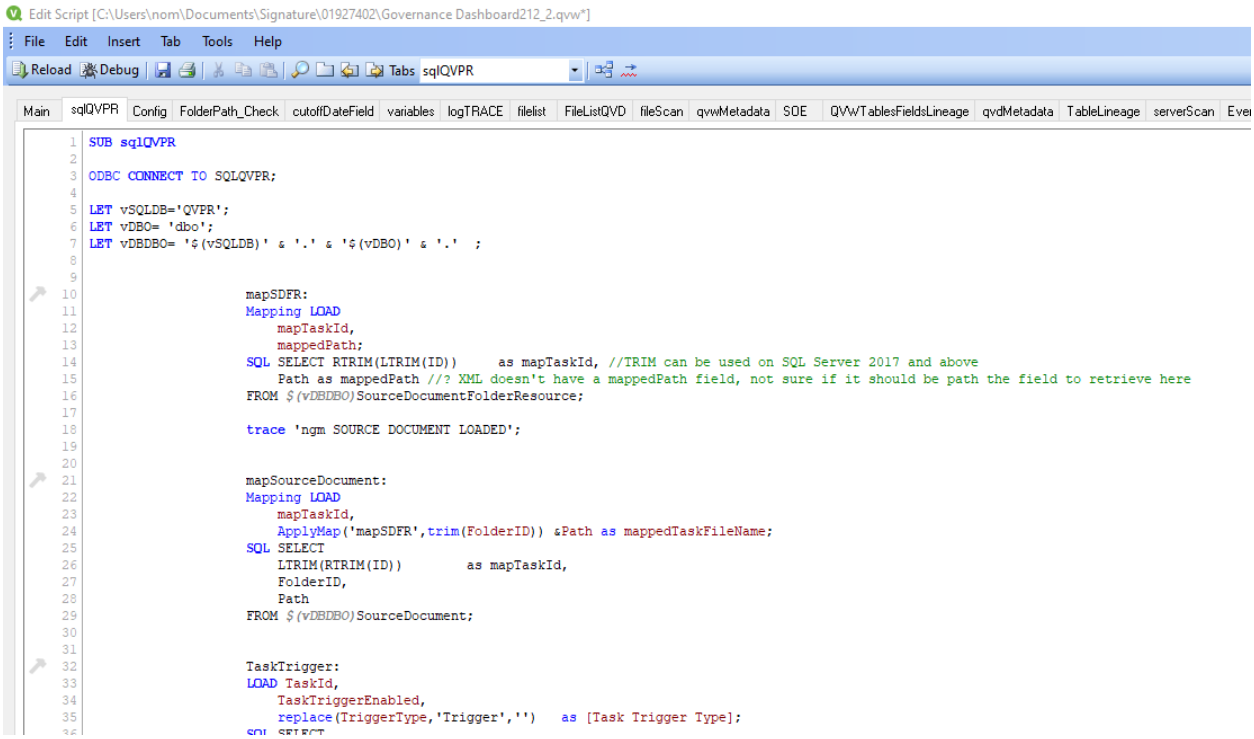
```
LET vDBDBO= '$(vSQLDB)' & '.' & '$(vDBO)' & '.'  ;


                                    mapSDFR:
                                    Mapping LOAD
                                           mapTaskId,
                                           mappedPath;
                                    SQL SELECT RTRIM(LTRIM(ID))             as mapTaskId,
//TRIM can be used on SQL Server 2017 and above
                                           Path as mappedPath //? XML doesn't have a
mappedPath field, not sure if it should be path the field to retrieve here
                                    FROM $(vDBDBO)SourceDocumentFolderResource;

                                    trace 'ngm SOURCE DOCUMENT LOADED';


                                    mapSourceDocument:
                                    Mapping LOAD
                                           mapTaskId,
                                           ApplyMap('mapSDFR',trim(FolderID)) &Path as
mappedTaskFileName;
                                    SQL SELECT
                                           LTRIM(RTRIM(ID))           as mapTaskId,
                                           FolderID,
                                           Path
                                    FROM $(vDBDBO)SourceDocument;


                                    TaskTrigger:
                                    LOAD TaskId,
                                           TaskTriggerEnabled,
                                           replace(TriggerType,'Trigger','')       as
[Task Trigger Type];
                                    SQL SELECT
                                           LTRIM(RTRIM(taskID))       as TaskId,
                                        [Enabled]              as TaskTriggerEnabled,
                                        TriggerType
                                    FROM $(vDBDBO)[Trigger] ;

                              // Keep DocumentTask as separate table due to inability to
join to existing Tasks table (root cause still unknown)
                                    DocumentTask:
                                    LOAD
                                        TaskId,   // Only join on TaskId
                                        // These fields exist already on Tasks table and
will be checked later and 'combined' with any existing field values
                                        ApplyMap('mapSourceDocument',SourceDocumentID)
       as docTask_Task_FileName,
                                        //ApplyMap('mapTaskCategory',trim(ID))
       as [Task CategoryDT],

SubField(replace(ApplyMap('mapSourceDocument',SourceDocumentID),'/','\'),'\',-1)
       as docTask_DocName,
                                           docTask_TaskName,
                                        docTask_TaskEnabled,

       alt(timestamp(ModifiedTime),timestamp(timestamp#(ModifiedTime,'$(TimestampForma
t1)')),timestamp(timestamp#(ModifiedTime,'$(TimestampFormat2)')))  as
docTask_TaskModified,
                                        // These fields are unique to DocumentTask and
are joined into the Tasks table directly
                                        AllowPluginClient,
                                        AllowMobileClient,
```

```
                                    AllowZeroFootprintClient,
                                    AllowPDFGeneration,
                                    [Task PDF ReportName],
                                    AllowDownload,
                                    [Task Distribute],
                                    SectionAccessUserName,
                                    SessionTimeout,
                                    DocumentTimeout,
                                    replace(ReloadOption,'Reload','')   as
ReloadOption;
                                SQL SELECT
                                       LTRIM(RTRIM(ID))
                                as TaskId,    // Only join on TaskId
                                       Name                                 as
docTask_TaskName,
                                    [Enabled]                               as
docTask_TaskEnabled,
                                    // These fields are unique to DocumentTask and
are joined into the Tasks table directly
                                    AllowPluginClient,
                                    AllowMobileClient,
                                    AllowZeroFootprintClient,
                                    AllowPDFGeneration,
                                    PDFReportName                  as [Task PDF
ReportName],
                                    DownloadAccess                 AS AllowDownload,
                                    Distribute                            as [Task
Distribute],
                                    SectionAccessUserName,
                                    SessionTimeout,
                                    DocumentTimeout,
                                    ReloadOption,
                                    SourceDocumentID,
                                    ModifiedTime
                                FROM $(vDBDBO)DocumentTask;

                                DistributionDetail:
                                LOAD
                                       TaskId,
                                       Distribution,
                                       'Distribute to ' & RecipientType &': ' &
RecipientName as DistributionDetail,
                                    RecipientName,
                                    RecipientType,
                                    DistributionType;
                                SQL SELECT
                                       LTRIM(RTRIM(DocumentTaskID))      as TaskId,
                                       1 as Distribution,
                                       RecipientType,
                                       RecipientName,
                                    DistributionType
                                FROM $(vDBDBO)DistributionDetail;


                                Concatenate (Tasks)
                                SQL SELECT
                                    LTRIM(RTRIM(ID))                              as
TaskId,
//                                  ApplyMap('mapTaskCategory',trim(ID))       as
[Task Category],
                                       CommandLine                        as [Task
CommandLine],
                                    Name                                  as [Task
```

```
Name],
                                    Enabled                              as [Task
Enabled],
                                    Description                  as [Task
Description],
                                    IgnoreErrors                 as [Task
IgnoreErrors],
                                    1                                    as
TaskExternalProgram
                                FROM $(vDBDBO)ExternalProgramTask;


      /*    About the QVPR files

      DistributionDetail  >> Keep this separate with Recipient Type and Name because
can have multiple entries per Task
            > in DistributionDetail entity:
                  DocumentTaskID                                 as TaskId    >
links to >   TaskId in Tasks and ID in DocumentTask,
                  RecipientName="All Users" + RecipientType="Anonymous"
                  Unused Data:
DistributionType,DDDField,DDDValueType,FSPath,SubPath,QvsResourceID,ValidateEmails,ID,
ModifiedByUser,ModifiedTime,IsDynamicDistribution

      DocumentTask
            > in DocumentTask entity:
                  ID                                             as TaskId    > to
link with DistributionDetail for recipient name and type
                  Name as DT_Name                  < Already have this from Tasks!
                  Enabled as DT_Enabled            < Already have this in Tasks
                  Description as DT_Description    < Missing from tasks

      AllowPluginClient,AllowMobileClient,AllowZeroFootprintClient,AllowPDFGeneration
,
                  SourceDocumentID                 < Links with ID from
SourceDocument.xml
                  Distribute="true"

      maybe:      SectionAccessUserName,SectionAccessPassword

                  Unused Data:
NameIsAutoGenerated,PDFReportID,AlwaysOpenable,ClearLocks,ClearAll,ClearAlwaysOneSelec
ted,ReapplySelections,

      EnableAuditLogging,SetScript,OverrideXSSectionAccess,MaxOpenSessions,SessionTim
eout,DocumentTimeout,ReloadOption,  DistributionServiceID,

      TaskTrack,TimeoutMinutes,AjaxUrl,NumberOfAttempts,ScriptParameterName,ScriptPar
ameterValueField,ScriptParameterValueFilename,NameTemplate,

      CreationMode,CreatorUserNames,DownloadAccess,DownloadUsers,ExportAccess,ExportU
sers,VersionID,EnableSessionCollaboration,DocumentDescription,
                  SendNotificationEmail,ModifiedByUser,ModifiedTime

      ExternalProgramTask >>> Concatenate this onto the Tasks table. No doc will be
directly associated with it.
            >> Might not have any entries (common)
                  Name                        as [Task Name],
                  Enabled                           as [Task Enabled],
                  Description                 as [Task Description],
                  ID                                as TaskId,
                  IgnoreErrors        as [Task IgnoreErrors],
```

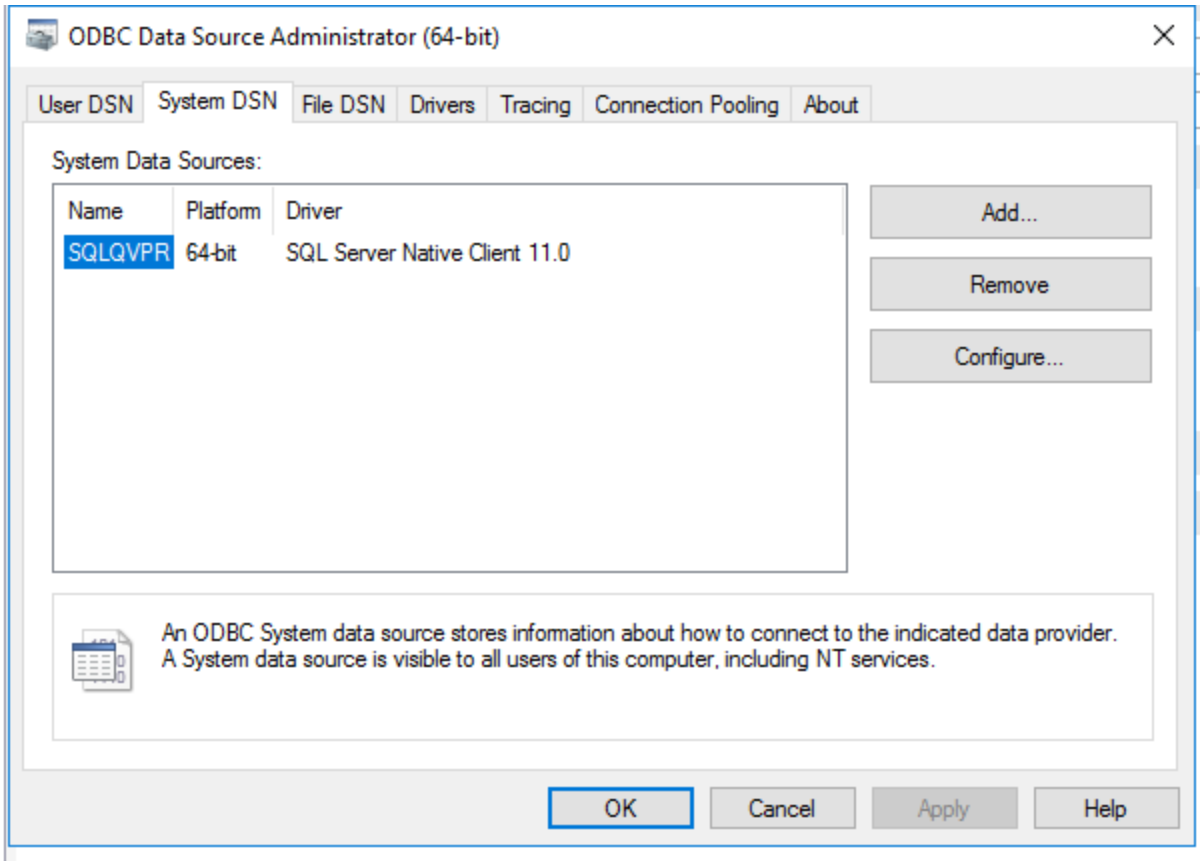
```
                    CommandLine                 as [Task CommandLine]

       SourceDocument      >> We have most of this data in Tasks &
TaskExecutionHistory xmls from QDS. However, that is just for tasks which executed.
Need this to show
                                        Information for Tasks that have not executed
(or whose last execution was before the history cutoff date (too old).
                                        Just concanetante this SourceDocument info
into SessionTaskAuditMaster to get DocName linked to TaskId

                    ID                                          as SourceDocumentID
       > link with DocumentTask
                    applymap('mapFolderResource','FolderID') & Path
       as Task_FileName

                    FolderID            > Need this linked to ____ to get full path
and name of Task_FileName (as shown in Tasks table)
                    Not used: DistributionServiceID, ModifiedByUser, ModifiedTime

       SourceDocumentFolderResource.xml        > to get the full path of the
SourceDocument, map Path to SourceDocument via ID and FolderID
                    mapFolderResource:
                    Load ID, Path
                    ....
                    Path   = full path (up to the "path" on SourceDocument)
                    ID                                          as FolderID > Link to
SourceDocument

       Trigger             >> Map this into the Tasks Table (after other QVPR tasks
are added to it) to get trigger information
              > in Trigger:
                    TaskID              links to TaskId
                    Enabled                         as [Task Trigger Enabled],
                    TriggerType                 as [Task Trigger Type]

                    Data not used (yet)
                    EnableAt,ExpireAt,RunTaskID,MaxCount,Count,StartAt,Days=""
EDXPassword,TimeContraintFrom,TimeContraintTo,DayNumbers,Months,Occourence,AndTimeCons
traintMinutes,
                    MainTriggerID,ID,ModifiedByUser,ModifiedTime,
       */

ENDSUB
```

It will be necessary to create a new tab an paste the above code, I just created a new tab next to the Main tab,

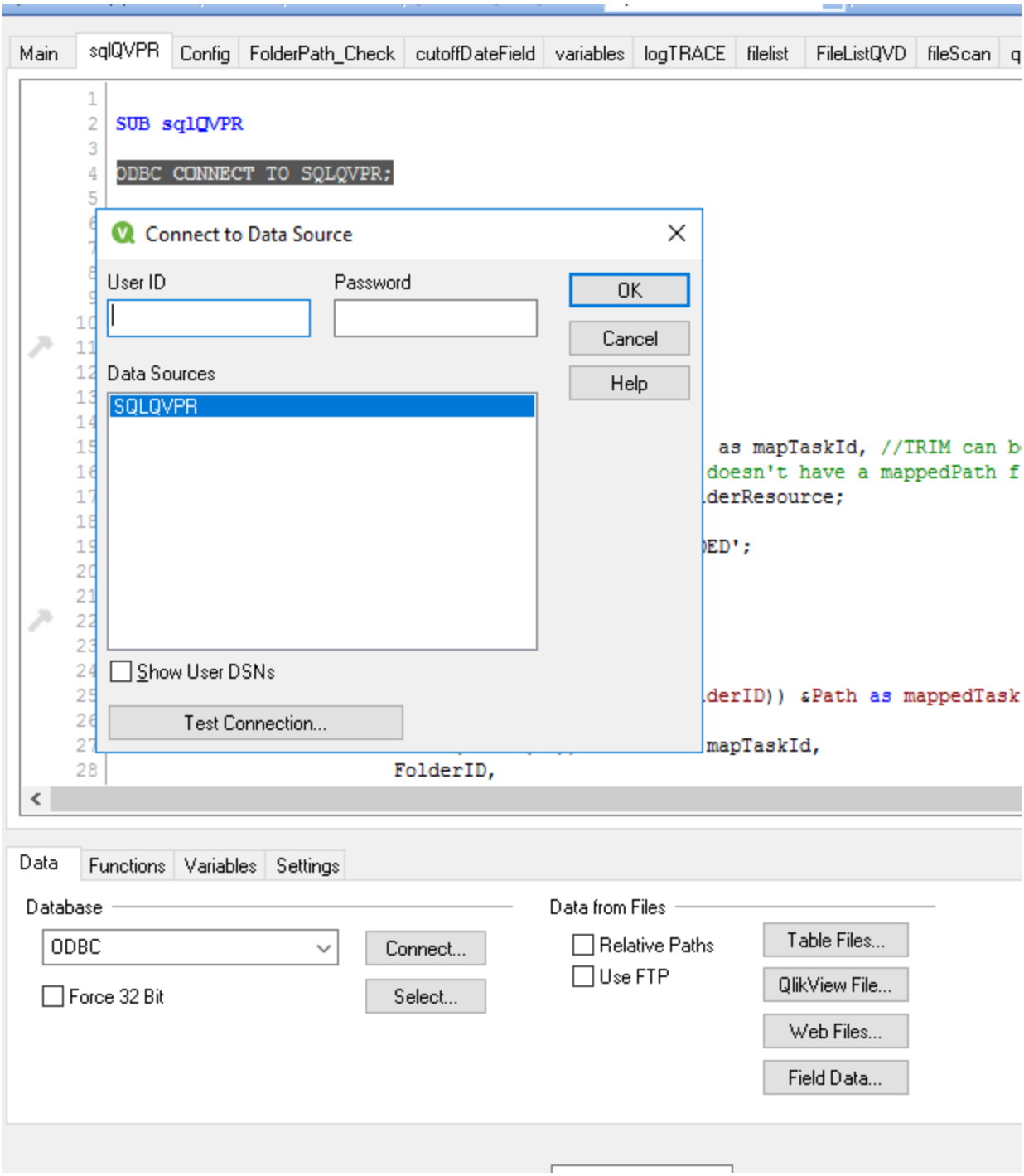


Once the code was pasted there are a couple of lines that need to be replaced.

1. The Connection, for this test I used a ODBC DSN



The Connection string in the script needs to be replaced inserting a new connection from QlikView desktop. Data -> ODBC -> Connect



The other information that needs to be replaced is the following:

```
LET vSQLDB='QVPR';
LET vDBO= 'dbo';
```

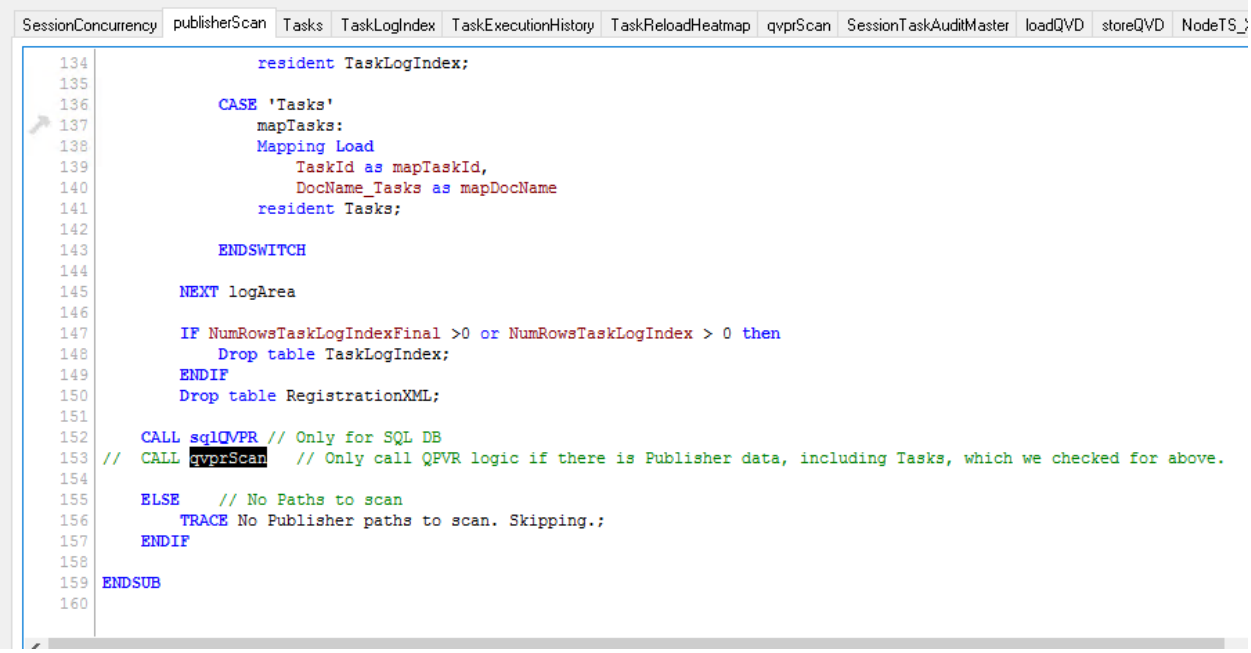
Where:

vSQLDB is the database name

vDBO is the database owner

TO COMMENT OUT THE FRACTION OF THE CODE WHERE THE QVPRSCAN IS CALLED.

It will be necessary to find the publisherScan subroutine and comment out the call to the subroutine qvprscan by sqlQVPR

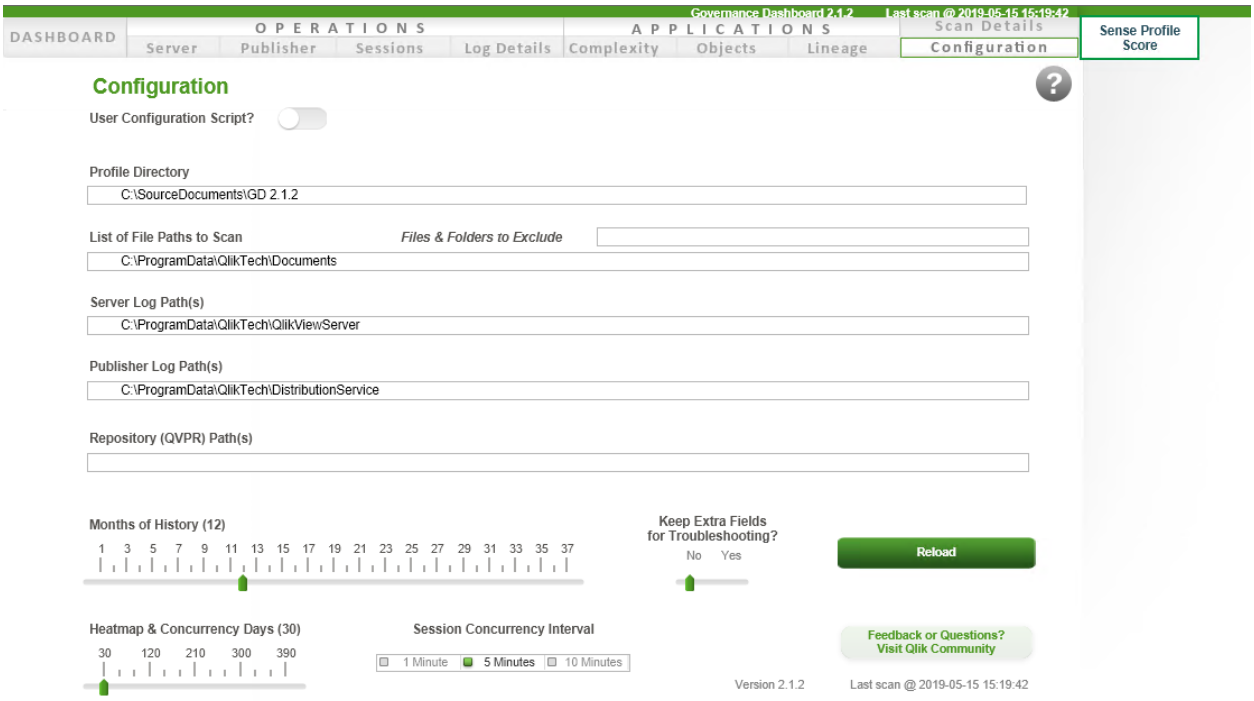


Save and create a duplicate of the QVW.

It is suggested to keep a copy of the GD that hasn't been reloaded for future use.

CONFIGURATION OF THE GOVERNANCE DASHBOARD

The configuration for the governance dashboard should be the same documented in the Supported Governance Dashboard, the only Path will be no necessary is the QVPR path as the dashboard will be reading data from the SQL Server database.



Now the configuration was completed the dashboard can be reloaded.