PhUSE 2010

Paper AD06

Project management (Dashboard and Metrics) with QlikView

Pirault Stéphane, Keyrus Biopharma, Levallois Perret, France
Delépine Guillaume, Keyrus Biopharma, Levallois Perret, France

ABSTRACT

To optimise the monitoring of its clinical studies, the clinical team requires a centralised application, available to all, which provides monitoring indicators and assists in the planning and performance of monitoring visits. This application must also meet the safety requirements of clinical trial data.

The purpose of this paper is to present a QlikView application which allows us to:

- Create and customise the dashboard with QlikView,
- Demonstrate, with an example, how helpful the indicators are to the CRA for scheduling monitoring visits,
- Share reports with end-users via the internet and to create end-user groups and set-up access restrictions.

INTRODUCTION

Our company Keyrus Biopharma, a clinical research organisation (CRO) provides support to the pharmaceutical and biotechnology industries for their clinical studies. To be able to control and optimise the monitoring of its clinical studies, the clinical team requires a centralised application, available to all, which provides monitoring indicators and assists in the planning and performance of monitoring visits. This application must also meet the safety requirements of clinical trial data (access to these indicators should be limited according to the user profile).

The major constraints to the development team were to be able to make an application:

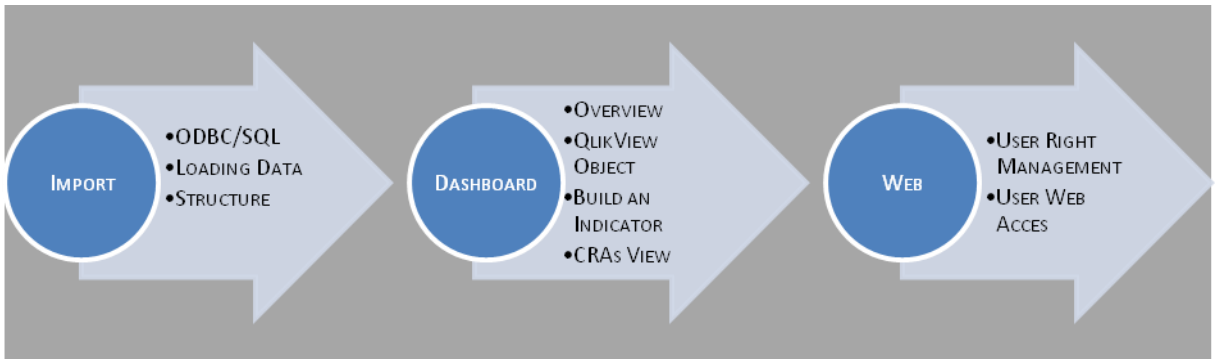
- Which can be developed and ready for use in only a few weeks,
- Which can be easily maintained with no pre-aggregation of data to calculate indicators,
- Which is easy to use and able to be used for all studies.

After having examined the various solutions available on the market, we chose QlikView, which brings together all the BI elements (data extraction, query and analytic applications, dashboards and reports) into a single integrated application.

In this paper we will:

- Describe how, with QlikView, we can set up a quick answer to the needs of the user,
- Explain how to import data with QlikView and how to create an indicator before giving an example of how the Clinical Research Associates (CRAs) use the tools,
- Describe how the application is published on the web and how to manage the end-user accounts.

These steps are described in the following flow-chart:



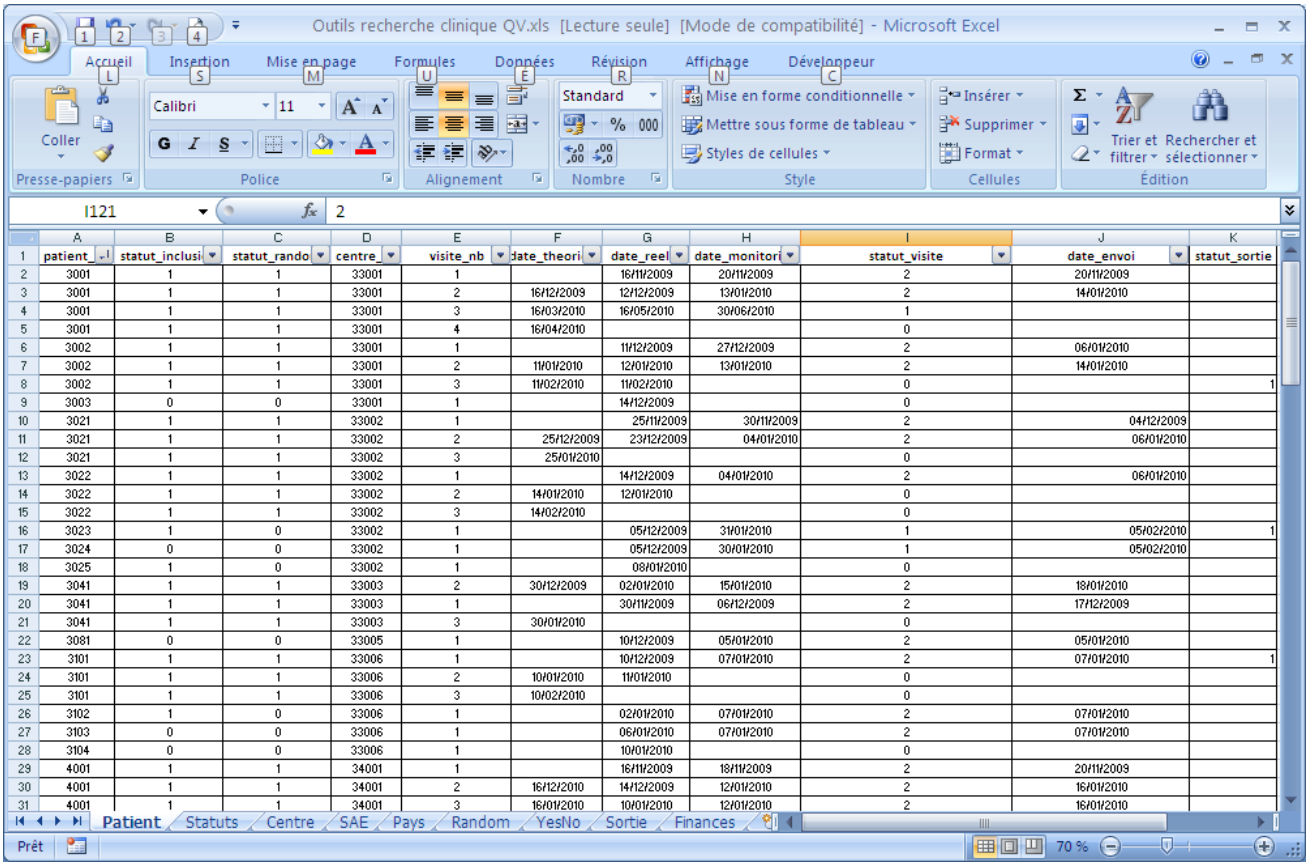
DATA IMPORT

QlikView is based on the use of flat data loaded in the live memory. The data stored in RAM are retrieved and prepared from a script defined by the user (in pseudo-SQL language). This technique, which is called in memory, can significantly reduce the volume of data (rate of about 10) whilst ensuring the uniqueness of the data fields.
When creating a QlikView project, the very first step is to load data within the application. This step is crucial because it determines the structure of the database that will be created within QlikView.

QlikView can connect to any ODBC data source including clinical tools such as Oracle Clinical and SAS. Note that in the case of ODBC connection with SAS, a SAS server with SAS Share is required.

To complete this step, the loading script must be edited. It can be done manually or with assistance from the user interface. For the present project, the application uses Excel files (see Figure 1) completed by CRAs.

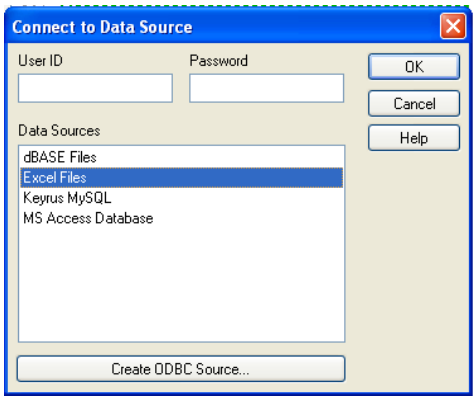
Figure 1: Data Source



	A	B	C	D	E	F	G	H	I	J	K
1	patient_	statut_inclusi	statut_rando	centre_	visite_nb	date_theori	date_reel	date_monitori	statut_visite	date_envoi	statut_sortie
2	3001	1	1	33001	1		16/11/2009	20/11/2009	2	20/11/2009	
3	3001	1	1	33001	2	16/12/2009	12/12/2009	13/01/2010	2	14/01/2010	
4	3001	1	1	33001	3	16/03/2010	16/05/2010	30/06/2010	1		
5	3001	1	1	33001	4	16/04/2010			0		
6	3002	1	1	33001	1		11/12/2009	27/12/2009	2	06/01/2010	
7	3002	1	1	33001	2	11/01/2010	12/01/2010	13/01/2010	2	14/01/2010	
8	3002	1	1	33001	3	11/02/2010	11/02/2010		0		1
9	3003	0	0	33001	1		14/12/2009		0		
10	3021	1	1	33002	1		25/11/2009	30/11/2009	2	04/12/2009	
11	3021	1	1	33002	2	25/12/2009	23/12/2009	04/01/2010	2	06/01/2010	
12	3021	1	1	33002	3	25/01/2010			0		
13	3022	1	1	33002	1		14/12/2009	04/01/2010	2	06/01/2010	
14	3022	1	1	33002	2	14/01/2010	12/01/2010		0		
15	3022	1	1	33002	3	14/02/2010			0		
16	3023	1	0	33002	1		05/12/2009	31/01/2010	1	05/02/2010	1
17	3024	0	0	33002	1		05/12/2009	30/01/2010	1	05/02/2010	
18	3025	1	0	33002	1		08/01/2010		0		
19	3041	1	1	33003	2	30/12/2009	02/01/2010	15/01/2010	2	18/01/2010	
20	3041	1	1	33003	1		30/11/2009	06/12/2009	2	17/12/2009	
21	3041	1	1	33003	3	30/01/2010			0		
22	3081	0	0	33005	1		10/12/2009	05/01/2010	2	05/01/2010	
23	3101	1	1	33006	1		10/12/2009	07/01/2010	2	07/01/2010	1
24	3101	1	1	33006	2	10/01/2010	11/01/2010		0		
25	3101	1	1	33006	3	10/02/2010			0		
26	3102	1	0	33006	1		02/01/2010	07/01/2010	2	07/01/2010	
27	3103	0	0	33006	1		06/01/2010	07/01/2010	2	07/01/2010	
28	3104	0	0	33006	1		10/01/2010		0		
29	4001	1	1	34001	1		16/11/2009	18/11/2009	2	20/11/2009	
30	4001	1	1	34001	2	16/12/2010	14/12/2009	12/01/2010	2	16/01/2010	
31	4001	1	1	34001	3	16/01/2010	10/01/2010	12/01/2010	2	16/01/2010	

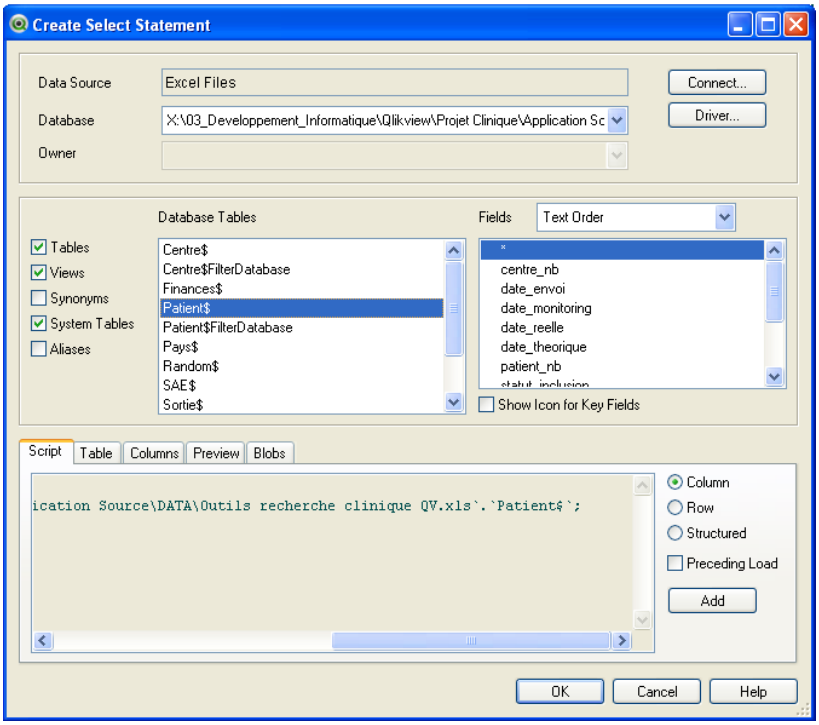
To edit the script go to the File menu then Edit Script (or CTRL+E). An ODBC data source can then be selected in the “Data” tab. After choosing “Excel file” (see Figure 2), the user is invited to select the Excel file containing the data to import.

Figure 2: Connection to Data Source Screen



Each tab of the Excel file will be considered as a table, and the interface (see Figure 3) allows the user to preview the data, to obtain information about columns (format, size, name, data type) and to select all or only some fields in the file.

Figure 3: Data Selection



QlikView detects and automatically manages the associations between the tables and when the choice is validated, SQL code is automatically generated in the script.

A sample of automatically generated code:

```
ODBC CONNECT TO [Excel Files;DBQ=X:\03_Developpement_Informatique\Qlikview\Projet
Clinique\Application Source\DATA\Outils recherche clinique QV.xls];
SQL SELECT *
FROM `X:\03_Developpement_Informatique\Qlikview\Projet Clinique\Application
Source\DATA\Outils recherche clinique QV.xls`.`Patients$`;;
//**********************************************
```

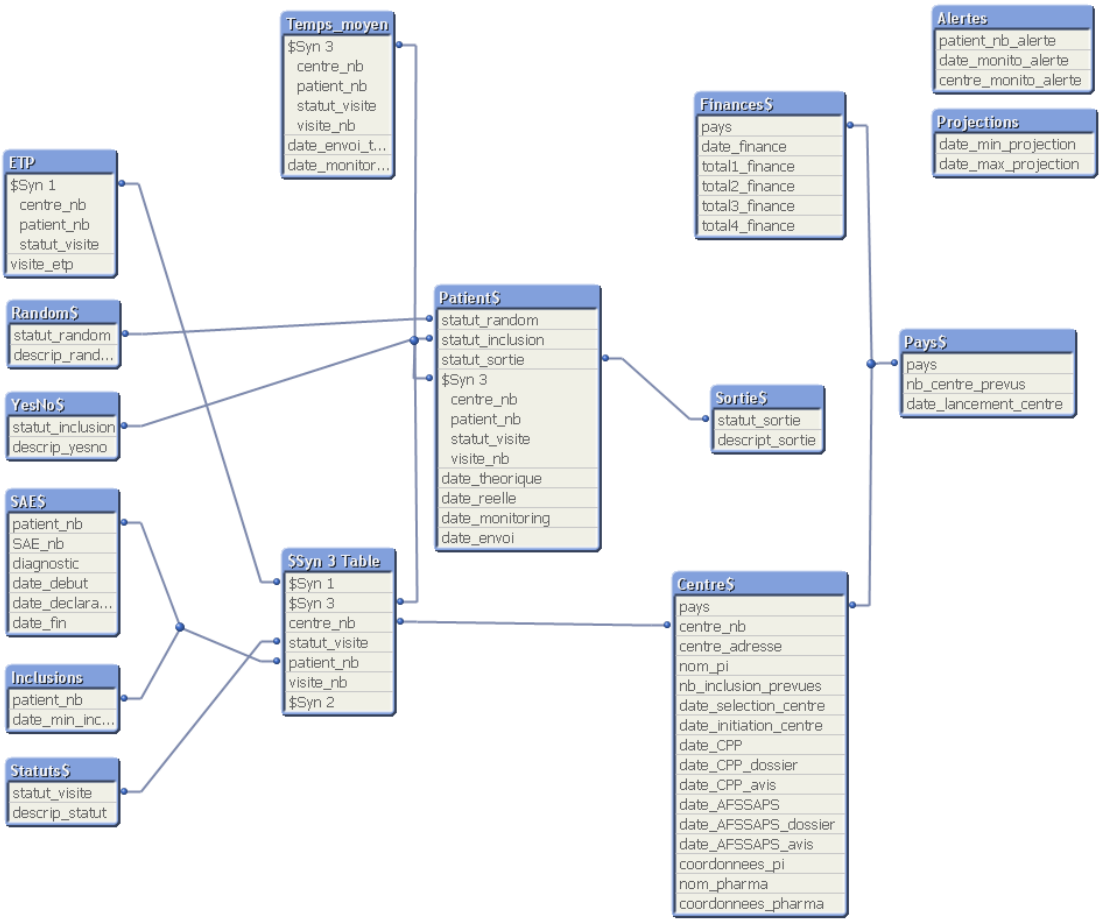
The embedded SQL engine in QlikView allows tables and fields to be renamed, restrictions to be made and functions such as ‘min()’, ‘max()’ and ‘distinct()’ to be added.
A sample of a customised query:

Projections:

```
SQL SELECT min(date_reelle) as date_min_projection, max(date_reelle) as
date_max_projection
FROM `X:\03_Developpement_Informatique\Qlikview\Projet Clinique\Application
Source\DATA\Outils recherche clinique QV.xls`.`Patient$` where statut_inclusion = 1;;
//**********************************************
```

The SQL engine also allows temporary working tables to be created or erased and junctions to be made in order to provide solutions for the creation of some indicators.
Once the script is entirely edited, data need to be loaded (File > Reload or Ctrl + R). The application will produce the structure of the data base (File > Table viewer or Ctrl + T to view it) (see Figure 4).

Figure 4: Structure of the data base



The application determines the key tables based on the field names of the loaded tables. Thus, if some tables have fields in common but have no link between them, the fields must be named differently in the loading script.
Once this is done, it is possible to create various objects to develop the application.

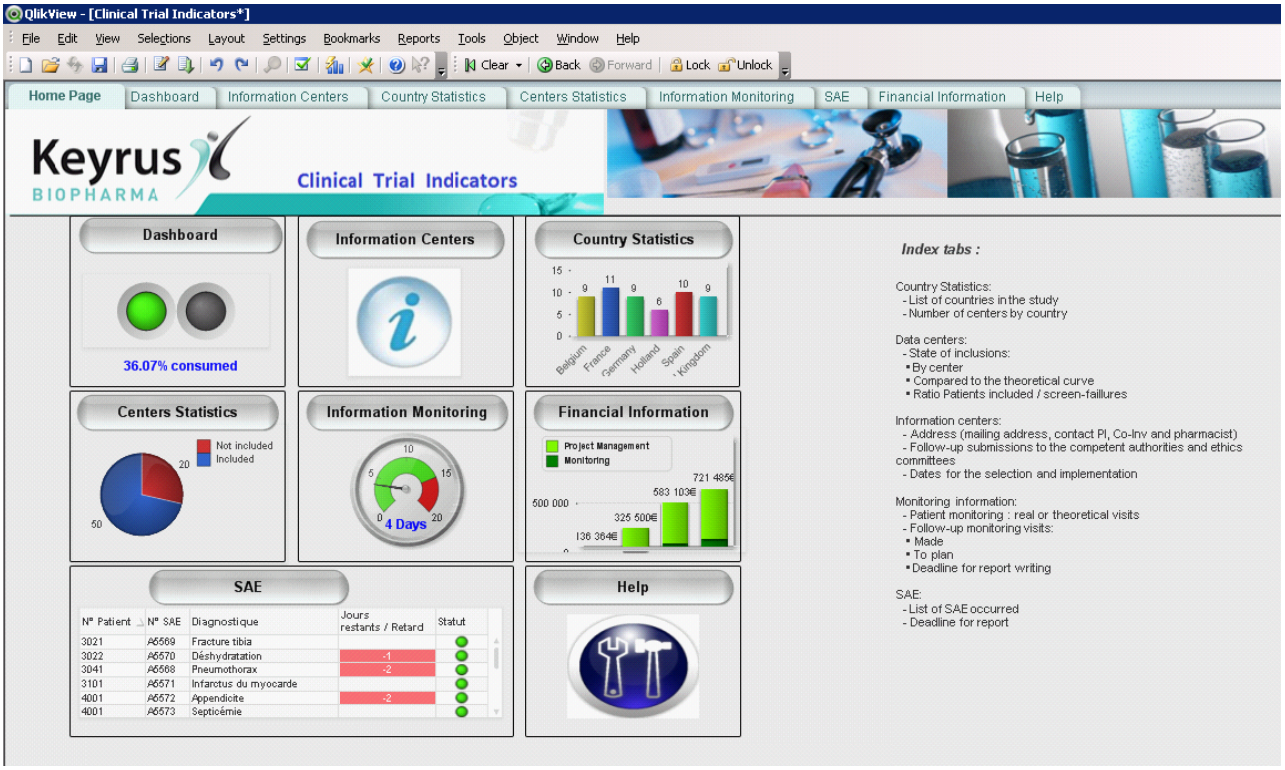
THE DASHBOARD

In this section dedicated to the indicators, the point of view of the developer and the user is discussed: the various steps for the generation of an indicator are given and the 'Monitoring Information' tab used by the clinical team is described.

First of all, the document created within QlikView is shown in different tabs as in a web browser.

Figure 5 shows an image of the application home page. Here the users find the different themes and the main indicator for the different tabs.

Figure 5: Clinical Trial Indicators (home page)



Before obtaining this result the developer must define the indicators and selection criteria. In QlikView these elements are called objects. As an example we will describe how to build a curve.

QlikView Objects

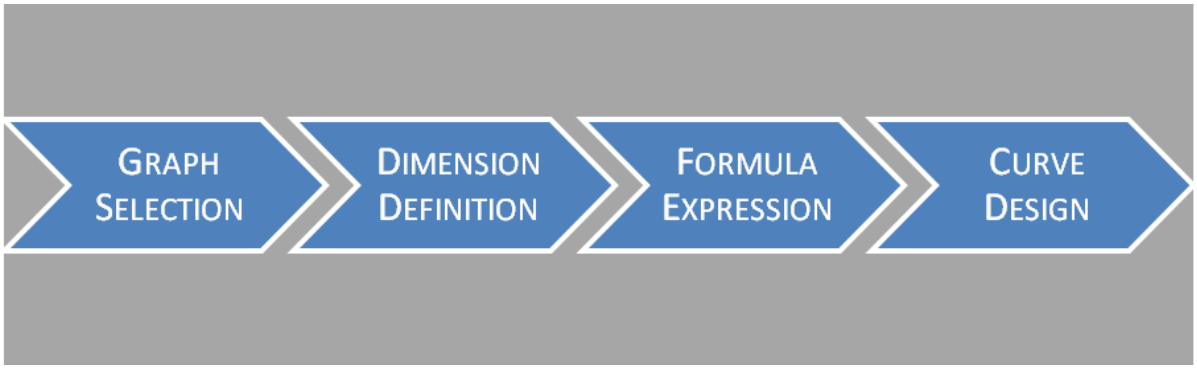
To add indicators to a page, a variety of objects is available. All these objects can be grouped by functions:

- Application of filters on the data,
- Display of data as table, chart or gauge,
- Design of application.

Among them, the chart object offers a variety of chart types (histograms, curves, grid, pie chart etc.) and also the opportunity to make a pivot table or a straight table which could host calculated fields or coloured indicators. Moreover, a function named 'Fast type chart' offers to the user several different representations for the same graph and allows the user to switch between them using a button in the object header.

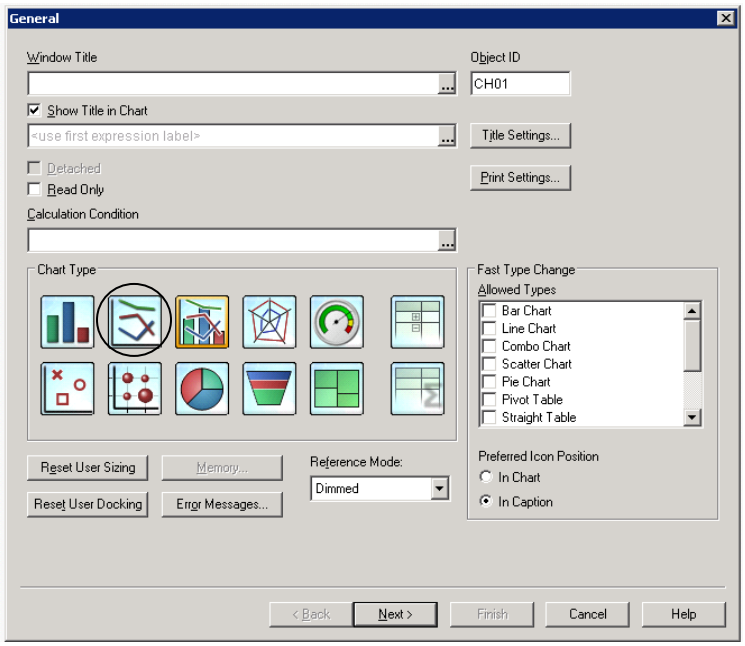
Build a curve

During the collection of clinical team requirements, we were informed of the need for a theoretical inclusion curve, which allows evaluation of the inclusion rate within the various investigator centres. This highlights the way in which the developers have enhanced the user-friendliness of the tool by providing a desired indicator.
The following flowchart describes the different steps of curve building.



Graph Selection: First of all, the developer must choose the chart type by clicking on the object tab, then 'New sheet object' and finally 'Choose chart'. The window shown in Figure 6 appears and displays the kinds of graphics available in QlikView.

Figure 6: Graph Selection



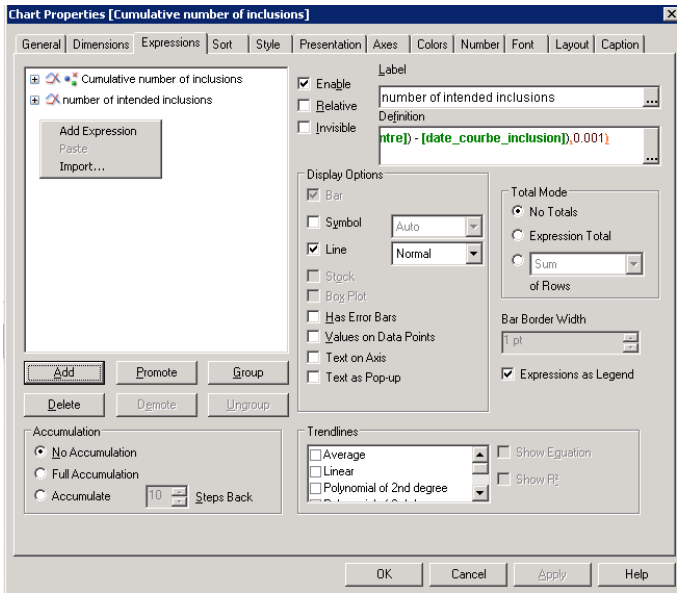
Dimension Definition: selection of the field "date_min_inclusion" which is the date of inclusion for all the patients who have been included within the study.

Formula Expression: completion of the formula to draw the desired curve (within expression): definition of the formula of the curve (see Figure 7) to display the number of intended inclusions. The code used is specific to QlikView, but has some similarities with formulas found in Microsoft Excel. In this field it must be defined whether the current filters should apply to the formula or not.

Curve Design: sets design and display of the curve such as the label (Number of intended inclusions), the colour (red), the symbol (none), the font and the size to be used for the text (arial / 12 pts), etc..

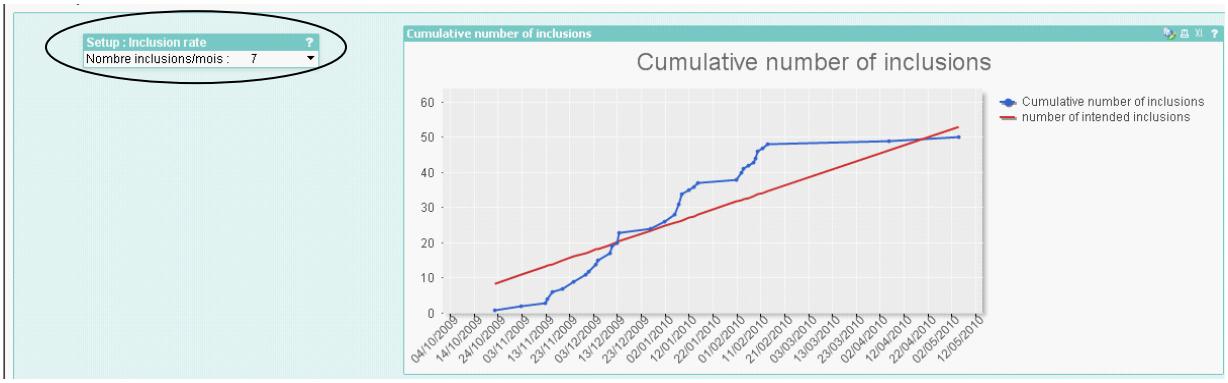
Other curves can be added by clicking on 'Add Expression' and restarting the process.
For this application we simply needed to generate the theoretical curve versus real curve. It is of course quite possible to create any type of curve using this method.

Figure 7: Chart Properties



At the end of these steps, the result shown in Figure 8 is achieved.

Figure 8: Cumulative number of inclusion



The formula of the curve is the following:

```
round(([rythme_inclusion]/30.4375) * fabs([date_courbe_inclusion] -
[date_min_inclusion])+([rythme_inclusion]/30.4735) * fabs(min(all {$}
[date_initiation_centre]) - [date_courbe_inclusion]),0.001)
```

Where 30.4375 is the average number of days per month.

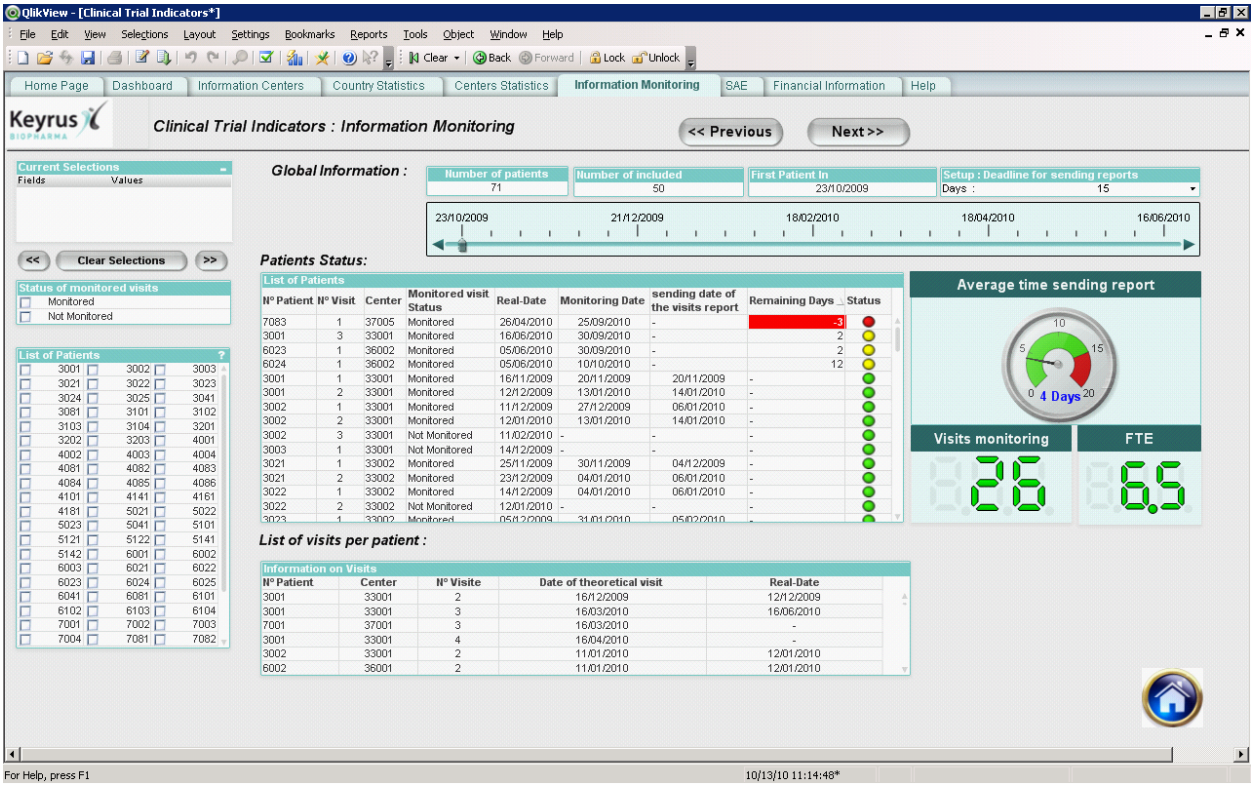
All the variables in this example are to be found in the source data except for 'rythme_inclusion'. This parameter, from which the theoretical inclusion curve is calculated, is able to be configured by the user using the circled object in Figure 8. The curve is updated in real-time which allows the inclusion period to be adjusted if the inclusion rate is not achieved.

All the objects available in QlikView are created according to the same process. The monitoring information tab presents the links between objects.

MONITORING INFORMATION TAB

Figure 9 shows how the needs of the user have been met in the Monitoring Information Tab.

Figure 9: Monitoring Information Tab



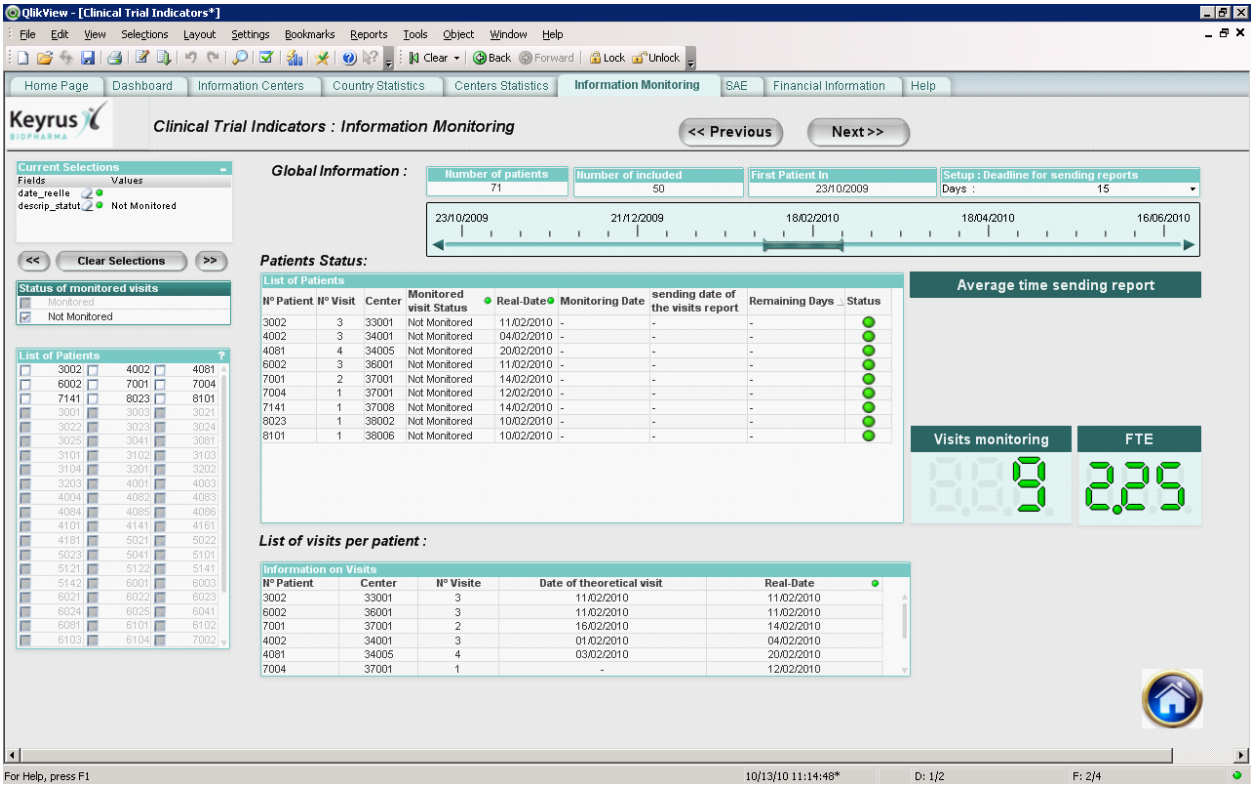
By accessing this page, the user will find information related to the:

- Project: number of selected patients, number of included patients and 'First Patient In',
- Visits: theoretical date, real date, and monitoring date,
- Patient status,
- Monitoring reports: date report is sent, time remaining.

Indicators inform the CRAs of the number of visits to be monitored and deliver an estimate of the amount of work this represents.
Another indicator informs the project manager of the average time for writing monitoring reports.
The left side of the screen contains objects which allow the user to apply filters to centres and to visit status. A field of active selections is present, which informs the user of the active filters.
A slider representing a time axis allows theoretical visit dates to be shown for a certain period of time. This slider can be moved along the axis thus enabling CRAs to plan monitoring visits effectively.

In Figure 10 we see that indicators have automatically been calculated and lists of patients have been updated depending on the period selected by the user.

Figure 10: Monitoring Information Tab with active selection

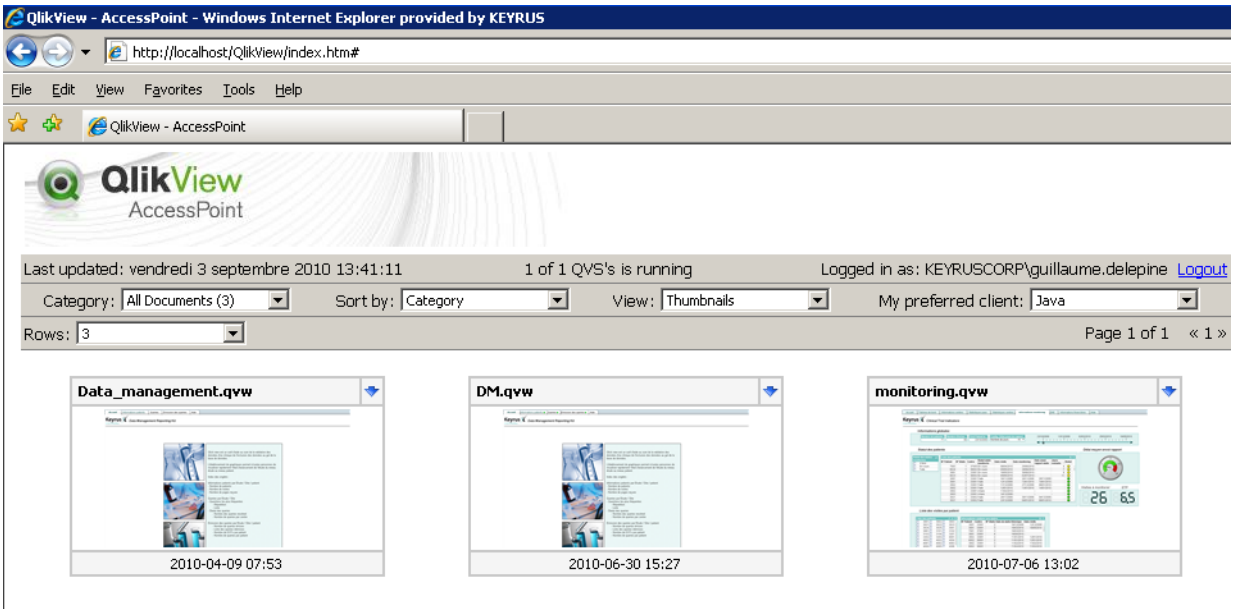


PUBLISHING AND WEB DEPLOYMENT

One of our clinical department's need was to have a centralised application which is available to everyone, whilst limiting the access of information and data depending on the user profile.

The QlikView server can meet these requirements. A management console allows the developer to run the server and set the repository folder from which the applications will be displayed.
Registered users connect via a portal dedicated to QlikView (see Figure 11) and can have access to only certain information depending on their profile.

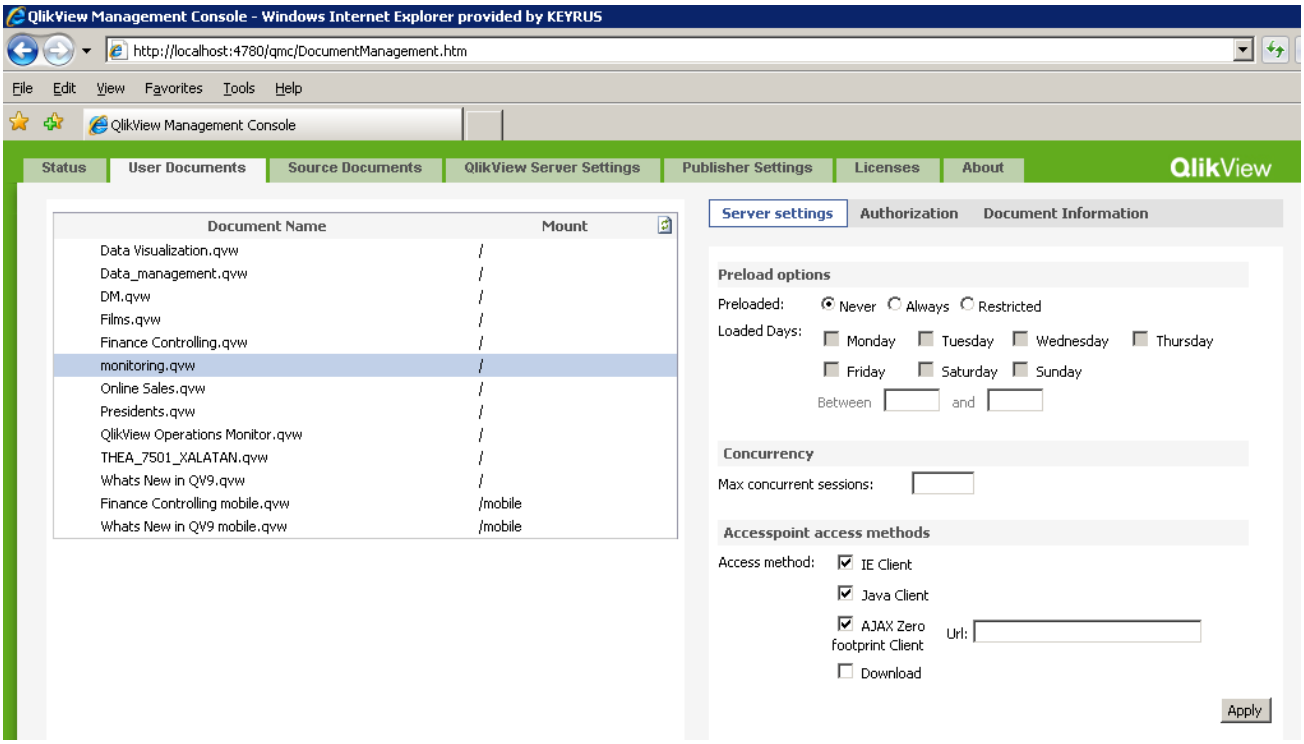
Figure 11: QlikView access Portal



The management of user groups is available within the application, allowing access to certain tabs or certain data to be defined according to the user profile (studies, financial information, centres etc.). The developed application is filed in the Qlikview server repository folder and user rights are defined.

The QlikView Management Console (see Figure 12) also offers the possibility to define in what form(s) the application is available (AJAX, Java, IE plug-in or download). The various choices can cover a wide range of configurations at both the operating system level and web browser level.

Figure 12: QlikView Management Console



CONCLUSION

In choosing QlikView to create this application we ensured that development was rapid and that maintenance costs are low. The clinical department now has an intuitive application which allows them to have an overview of each of their studies: either globally or in detail. Moreover, with the management of access rights, the use of this application is consistent with the pre-requisites of quality assurance.
QlikView has been proved to be an easy tool to use, both from the developer's and user's point of view, and is efficient and user-friendly. The solution provides interactive dashboards which are easily adaptable depending on requirements. However, improvements can still be made to the selection and movement of objects during the design of the application. Also, it is unfortunate that the command 'where' is absent in the programming of expressions because it forces the developer to use pre-recorded filters.
In addition to the solutions given to the users, QlikView also has the advantage of being available on the Web and can therefore provide clients with information on their projects. The release of this application in the clinical department has already aroused interest in the department of biometry at Keyrus Biopharma and other development projects are being studied.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Pirault Stéphane
Keyrus BioPharma
53 rue Baudin
92300 Levallois-Perret
France
Stephane.pirault@keyrus.comEmail: