@enterprise 11.0

Reporting

September 2023

Groiss Informatics GmbH

Groiss Informatics GmbH

Strutzmannstraße 10/4
9020 Klagenfurt
Austria

Tel: +43 463 504694 - 0
Fax: +43 463 504594 - 10
Email: support@groiss.com

Document Version 11.0.36123

Contents

1 Introduction

With the Reporting component (Report designer) you can create complex statistics and analyses on runtime data using a simple to use search mask. Reports created with this interface can be stored, executed, and edited later. The results can be shown as table, graphically, or exported to MS Excel, XML, PDF or CSV.
The Reporting is integrated into the permission system, which allows you to grant execute permissions for reports to specific user or roles.
You find the reporting component in the administration interface in the folder Search: The link "Report designer" allows you to create reports, the link "Reports" shows the list of reports.
Note that you need the right "Statistics" for creating reports.

2 Description of the Search Mask

The search mask is the central component and starting point for creating a report. The search mask is divided into following tabs (see fig. 2.1):

- Attributes (display fields): see section Types of display fields
- Conditions: see section Query conditions
- Joins: see section Selecting Joinpaths
- SQL report: This tab is available only, if the toolbar function *SQL report* (as sub entry of toolbar function *New report*) has been selected (see section SQL report). The tabs *Attributes*, *Conditions* and *Joins* are not available in this case!
- Export: see section Export of query results
- Options: see section Report options
- Preview: This tab contains a preview of the current report

2.1 Types of display fields

The display fields are the fields, which appear as columns in the query result. Add a field to the result table by selecting the field name of table *Attributes* and clicking the "Add" button. In the following we describe the display fields:

2.1.1 Display Fields of a Process instance

- **Id**: Process-Id containing a link to the process history.
- **Currently At**: Tasks and agents of the currently running steps of the process.
- **Active tasks information**: Tasks and available due dats, and agents of the currently running steps of the process.
- **Subject**: Subject of the process instance.
- **Application**: The application the process belongs to.

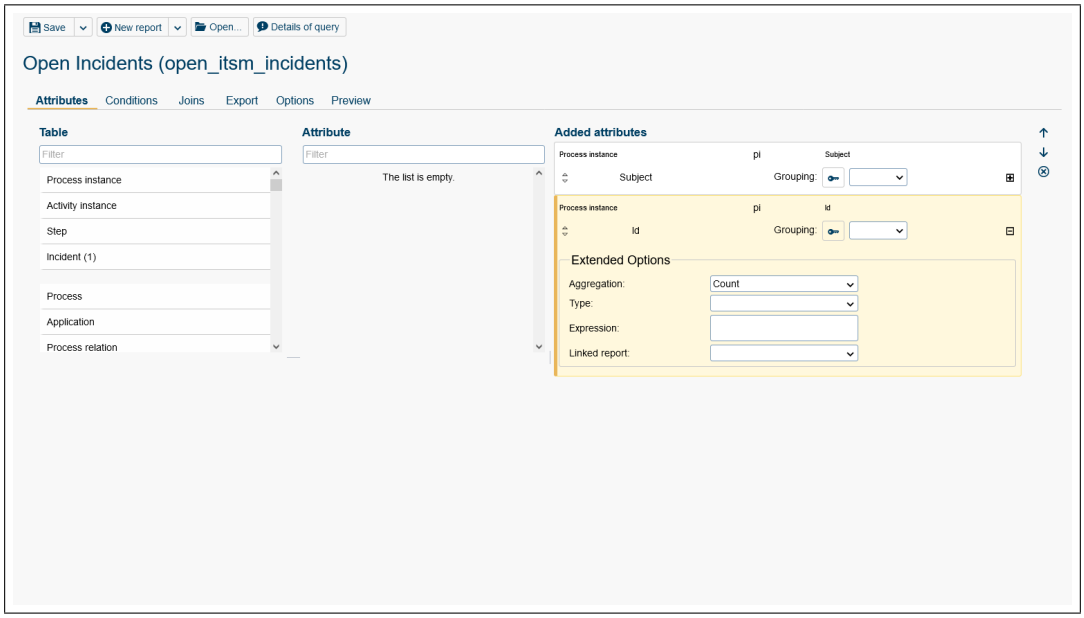

Figure 2.1: **Search Mask**

- **Process**: Name of the process.
- **Agent**: Agent who started the process.
- **Organizational Unit**: Organizational Unit where the process has been started.
- **Priority**: Value which has bee defined at process start or during run-time via appropriate task function for the process.
- **Status**: Status of the process, possible values are:
  - **Started**: The process has been started and is running.
  - **Finished**: The process has been finished.
  - **aborted**: The process has been aborted

  See Fig. 2.2 for the transitions between these states.
- **Started**: Start time of the process.
- **Finished**: End time of the process.
- **Duration**: Duration of the process (end - start).
- **Duedate**: Due Date of the process.
- **Time to duedate**: Time interval from now (execution of the query) to the due date of the process.
- **Folder**: A link to the tab *Documents* of a process is displayed in report result.

- **Object Id**: The id of the current object (in this case process instance) can be displayed in report result. This attribute can be used for e.g. the drill down functionality of Reporting component.

2.1.2 Display Fields of an Activityinstance

- **Id**: Activityinstance-Id.
- **Type**: All types of activity instances (see APIDoc of interface `com.groiss.wf.ActivityInstance` for details):
  - **BEGIN**
  - **END**
  - **IF**
  - **ELSEIF**
  - **LOOP**
  - **EXIT_WHEN**
  - **WHILE**
  - **PAR**
  - **ANDJOIN**
  - **ORJOIN**
  - **CHOICE**
  - **CHOICE_BRANCH**
  - **GOTO**
  - **TASK**
  - **SYSTEM_TASK**
  - **PROCESSES**
  - **BRANCH**
  - **END_BRANCH**
  - **SYSTEM_INTERN**
  - **SYSTEM_WAIT**
  - **PARFOR**
  - **ENDFOR**
  - **BATCH**
  - **SCOPE**
  - **END_SCOPE**
  - **END_COMPENSATION**
  - **WS_INVOKE**
  - **WS_RECEIVE**
  - **WS_REPLY**

  - **WAIT**

- **Child of**: Activity instance has a parent (the process instance of this task) and is child of it.
- **Agent**: Agent of the activity instance.
- **Stepagent**: Agent, where the task is in his worklist.
- **Organizational Unit**: Organizational Unit of this task.
- **Taken**: Time when the task has been taken from role-worklist.
- **Status**: status of task, possible values are:
  - **idle**: Two reasons exist for this state:
    1. The selection of the agent has been interrupted and the task has no agent assigned.
    2. The selection of the path in a choice control structure has been interrupted.
  - **suspended**: Task is in suspension list.
  - **finished**: The task has been finished.
  - **aborted**: Task has been aborted.
  - **active**: Task is in worklist (agent is a user).
  - **waiting**: See state *idle*.
  - **compensated**: The task has been finished and compensated.

  See Fig. 2.2 for the transitions between these states.
- **Started**: Start time of task.
- **Finished**: End time of the task.
- **Duration**: Duration of the task (end - start).
- **Duration without suspension**: Duration of the task without the time in the suspension list.
- **Idletime**: Time span from start to taken.
- **Worktime**: Time span from taken to finished.
- **Worktime without suspension**: Worktime without the time in the suspension list.
- **Suspensiontime**: Time span in the suspension list.
- **Task:Due Date**: Due date of the task.
- **Task:Time to Due Date**: Time interval from now (execution of the query) to the due date of the task.

- **Comments:** Step comment of an activity instance which has been created in *@enterprise* versions less than 10.0 (e.g. created at change agent). Step comments in *@enterprise* 10.0 and higher versions are stored in notes which can be displayed in report result by adding an attribute/condition of the entity *Note*. The JOIN selection offers the possibility to make a join between activity instance and note.

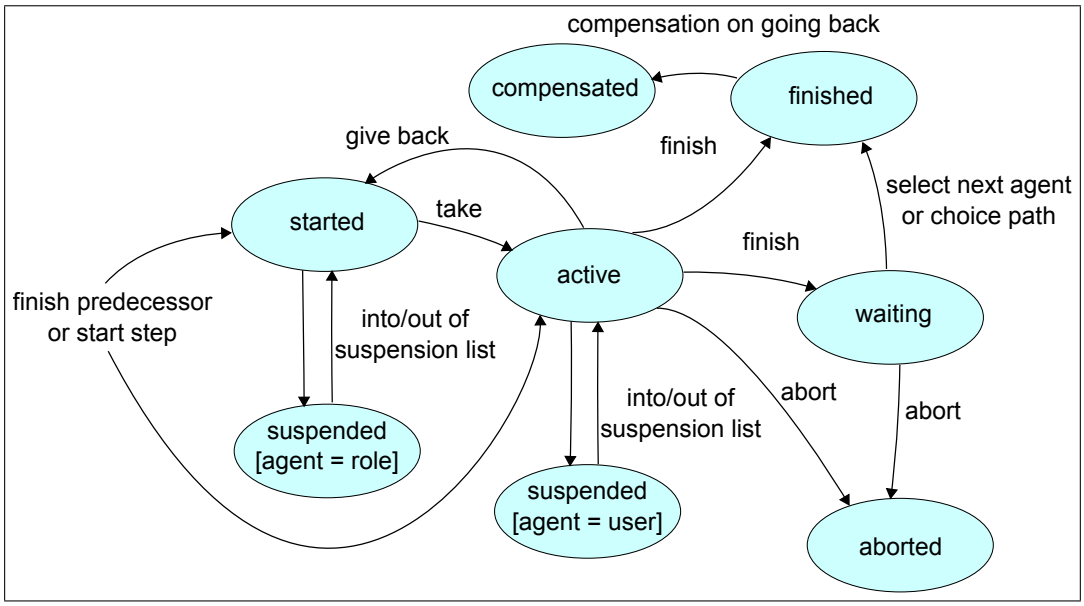

Figure 2.2: **Activityinstance states**

2.1.3 Display Fields of a Process

- **Id**: Process-Id containing a link to the process history.
- **Name**: Name of the process.
- **Version**: Version of the process.
- **Application**: Application, where the process is present (e.g. default).
- **Subject**: Subject of the process.
- **Priority**: Priority of the process.
- **Description**: Process description.
- **Active**: Activity-status of the process (active, inactive).
- **Object Id**: The id of the current object (in this case process) can be displayed in report result. This attribute can be used for e.g. the drill down functionality of Reporting component.

2.1.4 Display Fields of an Application

- **Id**: Application-Id.
- **Name**: Name of the application.
- **Description**: Description of the application.
- **Organization Hierarchy**: Hierarchy, where the application is present.
- **Application class**: Application class of the server.
- **Application Directory**: Directory, where the application is present.
- **Object Id**: The id of the current object (in this case application) can be displayed in report result. This attribute can be used for e.g. the drill down functionality of Reporting component.

2.1.5 Display Fields of Process Relation

- **Process 1** and **Process 2**: Process-Id of the process which is in relation to the searched process.
- **Process Relation Type**: Type of the process relation which can be defined in *@enterprise* administration under Configuration/Search/Process relations.

2.1.6 Display fields of Steps

As step we denote a set of executions of the same task back-to-back, i.e. without another task in between. The following display fields are available:

- **Activity Name**: Name of the activity.
- **Activity**: Activity either in tasks or in processes.
- **Started**: Start time of the step.
- **Finished**: End time of the step.
- **Duration**: Duration, related to the step in the process.
- **Worktime**: Sum of work time of the tasks.
- **Reactiontime**: Time span from start to taken in the first task of the step.
- **Idletime**: Time span from start to taken.
- **Suspensiontime**: Time span in the suspension list.
- **Duration without suspension**: Duration without the time in the suspension list.
- **Worktime without suspension**: Work Time without the time in the suspension list.

Restrictions for step fields:

- They can not be used in conditions.
- you get correct results for step-start, step-finish, etc. only if all tasks belonging to a step are in the result set.

2.1.7 Display Fields of Batch-Job

- **State**: State of batch job. Possible values are:
  - **Created**: Batch job is in State *CREATED*.
  - **Started**: Batch job is in State *STARTED*.
  - **Finished**: Batch job is in State *FINISHED*.
  - **Completed**: Batch job is in State *COMPLETED*.
  - **Aborted**: Batch job is in State *ABORTED*.
  - **Start error**: Batch job is in State *STARTERROR*.
  - **Finish error**: Batch job is in State *FINISHERROR*.
  - **Start-Error-Handling started**: Batch job is in State *STARTERRORHANDLING*.
  - **Finish-Error-Handling started**: Batch job is in State *FINISHERRORHANDLING*.
  - **Error-Handling completed**: Batch job is in State *COMPLETEDERRORHANDLING*.
  - **Start-Error-Handling pending**: Batch job is in State *STARTERRORHANDLINGPENDING*.
  - **Finish-Error-Handling pending**: Batch job is in State *FINISHERRORHANDLINGPENDING*.
  - **Start-Error-Handling failed**: Batch job is in State *STARTERRORHANDLINGFAILED*.
  - **Finish-Error-Handling failed**: Batch job is in State *FINISHERRORHANDLINGFAILED*.
- **Typ**: Java class of Batch job.
- **Result**: The result set by the *BatchAdapter*.
- **Result code**: The result code set by the *BatchAdapter*.
- **Creation time**: The timestamp in which the batch job is created.
- **Start time**: Start time of the batch job.
- **Finish time**: End time of the batch job.
- **Completed**: The time stamp in which the batch job is completed in the workflow.
- **Duration**: The duration of the batch job between creation (= Creation time) and completion (= Completed).

2.1.8 Display fields of Tasks

- **Id**: Task-Id.
- **Name**: Name of the task.
- **Version**: Version of the task.
- **Description**: Free Text, which is visible in the worklist via the link to the task details.
- **Method Call**: Java-Method, which is called before the task is put into the worklist of a user.
- **Postcondition**: Java-Method, which is checked at run-time when a user finishes the task (sends it to the next agent).
- **Postcondition-Message**: Free text, which will be shown when the postcondition evaluates to false.
- **Compensation**:Java-Method, which will be executed when the activity is passed when going back to an earlier step in the process.
- **Take-Hook**: Java-Method, which is called when the task is taken from the role-worklist to the personal worklist.
- **Untake-Hook**: Java-Method, which is called when the task is given back to the role worklist.
- **Active**: Activity-status of the task (active, inactive).
- **Costs**: The costs of the task. This field is not used from *@enterprise*.
- **Effort**: The effort of the task in minutes. This field is not used from *@enterprise*.
- **Object Id**: The id of the current object (in this case task) can be displayed in report result. This attribute can be used for e.g. the drill down functionality of Reporting component.

2.1.9 Display fields of Organizational Units

- **Id**: OU-Id.
- **Name**: Name of the OU.
- **Description**: Description of the OU.
- **E-Mail**: E-Mail address of the OU.
- **Address**: Address of the OU.
- **Tel.-Nr.**: Phone number of the OU.
- **Active**: Activity-status of the OU (active, inactive).
- **Type**: Type of the OU (External OU, Dependent).

- **Organization Class**: The organization class the OU belongs to.
- **Follow Org.-Unit**: Follow Org.-Unit, which replaces the current OU.
- **Object Id**: The id of the current object (in this case OU) can be displayed in report result. This attribute can be used for e.g. the drill down functionality of Reporting component.

2.1.10 Display fields of Roles

- **Id**: Role-Id.
- **Name**: Name of the role.
- **Description**: Description of the role.
- **Type**: Type of the role (local, global, hierarchic).
- **Active**: Activity-status of the role (active, inactive).
- **Reference-Role**: Reference roles are used for defining different roles with different rights but one "reference" role used in process definitions.
- **Application**: Application, where the role is known.
- **Object Id**: The id of the current object (in this case role) can be displayed in report result. This attribute can be used for e.g. the drill down functionality of Reporting component.

2.1.11 Display fields of Users

- **Id**: User-Id.
- **Surname**: Surname of the user.
- **First name**: First name of the user.
- **Description**: Description of the user.
- **E-Mail**: E-Mail address of the user.
- **Tel.-Nr.** Phone number of the user.
- **Language**: Language for the user interface.
- **Active**: Activity-status of the user (active, inactive).
- **Server**: @*enterprise*-Server, where the worklist is accessible.
- **Date of the last password change**: Date, when the password was changed.
- **Has to change password at next login**: User has to change password at next login.
- **Password never expires**: Password of the user never expires.

- **Cannot change password**: User has no right to change the password.
- **Object Id**: The id of the current object (in this case user) can be displayed in report result. This attribute can be used for e.g. the drill down functionality of Reporting component.

2.1.12 Display fields of Role Assignments

- **Organizational Unit**: OU, where the role is assigned.
- **User**: User, who has the role.
- **Role**: Id of the assigned role.

2.1.13 Display fields of Document

- **Name**: Name of the document.
- **Created**: Creation date of the document.
- **Changed**: Change date of the document.
- **Organizational Unit**: Organizational Unit, where the document is assigned.
- **Creator**: Creator of the document.
- **Form type**: Form type of the document (see chapter *Forms* in administration guide).

2.1.14 Display fields of Documentfolder

- **Folder**: Link to the folder.

2.1.15 Display fields of Link

- **Name**: Name of link.
- **Linked object**: Name of linked object.
- **Created at**: Timestamp of creation of this link.
- **Created by**: User who created this link.
- **Last changed at**: Timestamp when link was changed last time.
- **Last changed by**: User who changed this link.
- **Object Id**: The id of the current object (in this case Link) can be displayed in report result. This attribute can be used for e.g. the drill down functionality of Reporting component.

2.1.16 Display fields of Keyword

- **Keyword**: Keyword label.

2.1.17 Display fields of document versions

- **Version**: Link to version of document.
- **Created by**: User who created this version.
- **Created at**: Time stamp of creation of this version.
- **Description**: Description (free text) written by the creator.

2.1.18 Display fields of user session

- **User**: The first and last name of the user.
- **IP address**: The IP-address of the user.
- **Date of initialization**: The initialization-date of the user-session (login date of the user).
- **Date of logout**: The date when the user was logged-of from @*enterprise*.
- **Last access**: The date when the user was active in the system.

2.1.19 Display fields of Recycle Bin content

The @*enterprise* recycle bin is a temporary storage location for deleted DMS objects which is displayed per user and is active by default. DMS objects are moved to recycle bin by selecting them in DMS object table and activating toolbar function *Delete* or pressing Del on keyboard.

- **Origin**: Information of the origin storage location of the DMS object (DMS folder, process instance).
- **Deleted at**: The timestamp when DMS object has been deleted/moved to recycle bin.
- **Deleted with**: Information, if the element has been deleted directly or indirectly. Indirect means that a folder has been deleted and the element is part of this folder

2.1.20 Form fields as display fields

For adding form fields click on a form in the list *Tables*, select a formfield of the list *Attributes* and activate the button "Add". If you select a version independent form (no number after the form name), you must create a view over the form versions, see the Administration Guide, section Forms, for details. If the form is not a process form but a subform you must select the main form (process form).

2.1.21 User defined display fields

When selecting the display field "User-defined" you will see a mask with the following fields:

- **Scheme:** schema of a database

- **Table:** name of database table
- **Table alias:** a table alias for this query
- **Attribute:** name of field of table
- **Column title:** the column header, can include I18N keys (@@@key@@)
- **Aggregation:** maximum, minimum, count, average, sum, Aggregation in expression
- **Type:** Data type (e.g. String, Date, etc.)
- **Expression:** SQL-expression, used as selection to the table
- **Linked report:** If a column is linked to another report, the data value of of the clicked column will be used as parameter at execution. Therefore the linked report has to expect the parameter condition on index n+1 and n is the index of the last parameter at execution condition of the initial report.

Example 1: Surname of agent of the process where userid starts with user

Table: avw_user
Table alias: u
Attribute: surname

After activating the button *Ok* the attribute will be added. You can change the column title to *Surname* by clicking on text *avw_user.surname*. After this action following settings must be done in section *Extended Options* of the attribute:

Type: String
Expression: u.id like 'user

Example 2: Create a list of tasks with id and oid:
Attribute 1:
Table: avw_task
Table alias: t
Attribute: oid

After activating the button *Ok* the attribute will be added. You can change the column title to *oid* by clicking on text *avw_task.oid*. After this action following settings must be done in section *Extended Options* of the attribute:
Type: String

Attribute 2:
Table: avw_task
Table alias: t
Attribute: id

After activating the button *Ok* the attribute will be added. You can change the column title to *id* by clicking on text *avw_task.id*. After this action following settings must be done in

section *Extended Options* of the attribute:
Type: String

2.1.22 Aggregations

The following aggregations are possible:

- count (e.g. count(id))
- maximum (e.g. max(duration))
- minimum
- average
- sum
- Aggregation in expression: Select this option if select statement includes a sql aggregation.

Count can be used for all fields, the other aggregations can only be used for date or numeric fields (exception: *Aggregation in expression*).
Please be aware that when *Aggregation in expression* is used, an explicit type for the attribute should be declared as well. If you do not choose a type, then the type of the attribute will also be used as type for the aggregate expression.
Do also note the special semantics of NULL values in aggregations. Like in SQL, NULLs are never taken into account for aggregations; in particular, they are not COUNTED.
Aggregations of date fields

minimum: first step of a task
maximum: last step of a task
Count: counts the steps
Aggregation in expression: The aggregation can be set in the field *Expression*.

Date Format of Started Hour, Day, Week, Month, quarter, Year or a custom date format pattern which needs to fit the JAVA SimpleDateFormat patterns (see section Grouping by setting the date format): Format for Timestamps.

Example:
Proz-ID =1, execution order: order → order → a_task → order

ID	Task	Task:Started	Task:Duration
1	order	21-04-2007 10:00	70 min
1	order	21-04-2007 11:10	50 min
1	a_task	21-04-2007 12:00	60 min
1	order	21-04-2007 13:00	80 min

ID	Task	Step:Start	Step:Duration
1	order	21-04-2007 10:00	120 min
1	a_task	21-04-2007 12:00	60 min
1	order	21-04-2007 13:00	80 min

ID	Task	Count(Step:Start)	
1	order	2	
1	a_task	1	
ID	Task	MIN(Step:Start)	MAX(Step:Start)
1	a_task	21-04-2007 12:00	21-04-2007 12:00
1	order	21-04-2007 10:00	21-04-2007 13:00

2.1.23 Details to time intervals

Time intervals are computed as follows:

- **Processinstance**: *Duration* = *Started* to *Finished*
- **Activityinstance**: *Duration* = *Started* to *Finished*
- **Activityinstance**: *Idletime* = *Started* to *Taken*
- **Activityinstance**: *Worktime* = *Taken* to *Finished*
- **Activityinstance**: *Time to Due Date* = Execution time of report until *Due Date*

Fig. 2.3 shows the time intervals on a time line. If the end time of an interval is not yet known, the execution time is taken instead. If you want only intervals where the end time is known add a condition that assures that.
Fig. 2.4 shows the computation of time intervals per step.
The process structure contains three steps, each step referring to a task. The process instance contains four instance steps, the first is an instance of the first process step, the next two are instances of the second step (this can be the result of a "change agent" action), the fourth is an instance of the third step. The right column shows how the work time per step is computed.

2.1.24 Trend analysis

Time intervals can also be used to show the data progress on a time axis. To do this, you can select a date format at time intervals. As a result, the actual time interval is no longer displayed, but the corresponding result line from [Start] to [End] is duplicated in the specified date format.

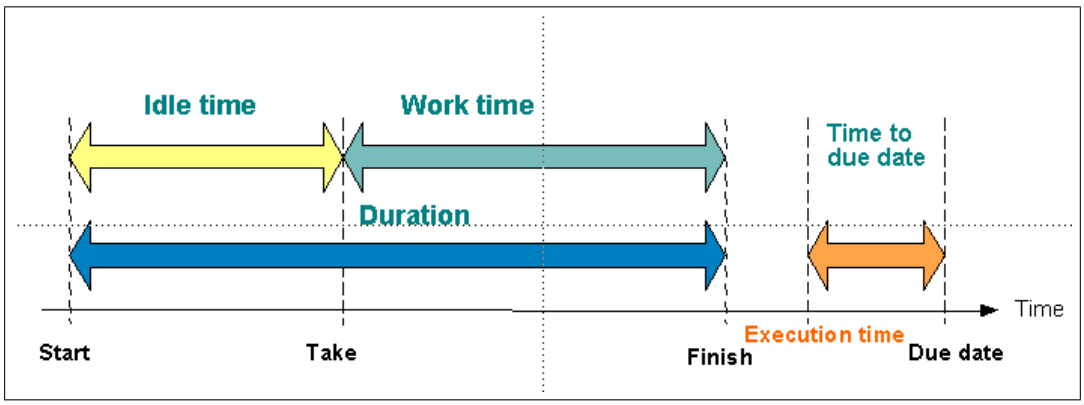

Figure 2.3: **Time intervals for tasks**

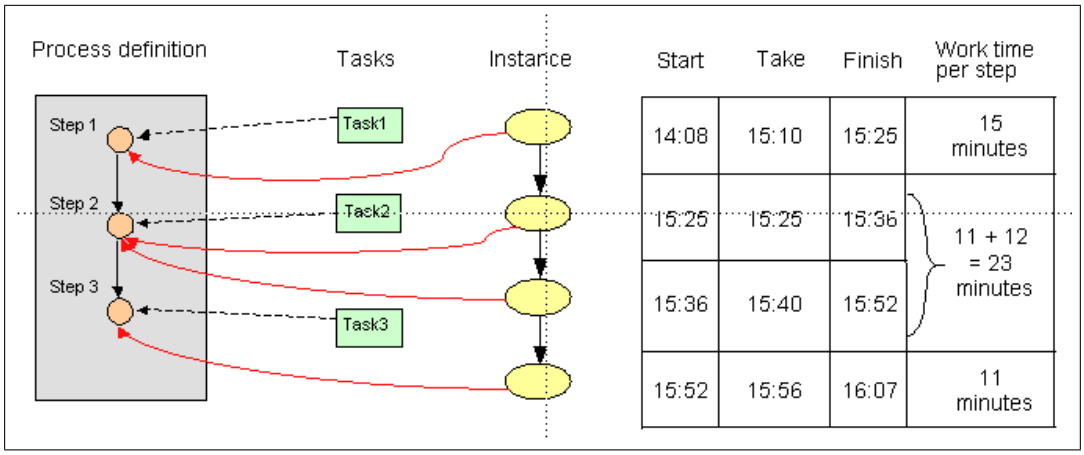

Figure 2.4: **Computation of time intervals per step**

This means:

Month (process instance (duration) [process ID]
1.1.2020 08:15–31.12.2020 08:15 4711

becomes:

January 2020 4711
February 2020 4711
March 2020 4711
...
December 2020 4711

In combination with a suitable aggregation function (e.g. Count), the number of running processes per month can be shown in a table or by using a diagram(see Fig. 2.5).
To limit the displayed intervals, a condition can be defined on one of the data fields involved

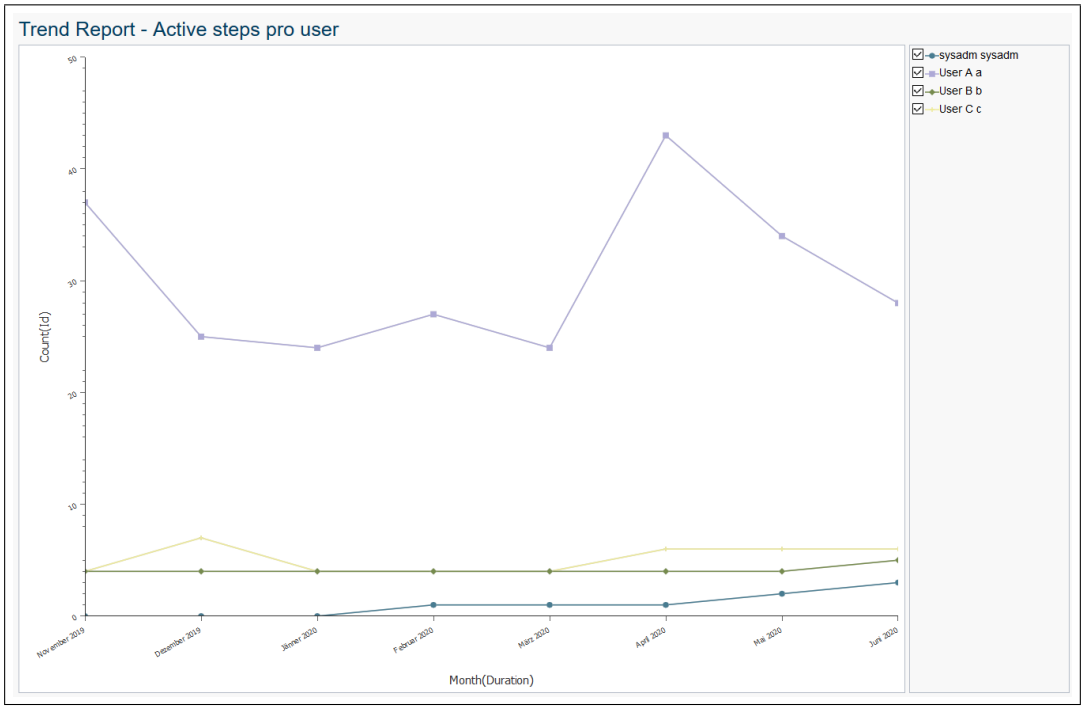

Figure 2.5: **Trend-report example**

in the interval (e.g. started or ended). The intervals are then only calculated/duplicated within the selected range (e.g. number of running processes per month in a given year).

2.1.25 Grouping by setting the date format

If you select a display field of type date (for example Process:Start), you can select the following date formats: hour, day, week, month, quarter, and year. Furthermore it is possible to enter a custom date format which needs to fit the JAVA SimpleDateFormat patterns. If such custom date formats should be available by default (selectable), the schema file *reporting.xml* must be overwritten and the mapping *dateformats* extended like in following example:

```
<!--  extend this mapping to add dateformats -->
 <mapping xmlid="dateformats">
  <mapentry key="yyyy.MM.dd" value="My date format" />
 </mapping>
```

The entry "My date format" should be available beneath the entry "Year" after reloading all reporting schema files.

With this feature you can group by time intervals. For example you can retrieve the number of processes started per quarter by selecting the following display fields (see also the example section):

- Count(Process:Name)

- Process:Started [quarter]

2.1.26 Options for fields

For each attribute (display field) a set of options can be defined:

- Infos: Shows the database table and field of selected attribute.
- Column title: The column header, can include I18N keys (@@@key@@).
- Date format: Is displayed only, if type *Date* is selected and allows the definition of hour, day, week, month, quarter, year or a custom date format pattern which needs to fit the JAVA SimpleDateFormat patterns (see section Grouping by setting the date format): grouped by time spans
- Aggregation: Select an aggregation as described in Aggregations.
- Type: By specifying this option, default type of the result data is overwritten. Ensure that the selected data fits to the selected data type. If set to *TimeInterval* two comma-separated date fields are expected in expression.
- Expression: Can be used to overwrite default select of this attribute, which is defined in schema. Ensure that Expression includes a complete SQL select statement (e.g: *pi.oid*). If aggregation is set to *Aggregation in expression*, a SQL aggregation is suspected in expression (e.g: *count(pi.status)*).
- Table alias: Is used as table alias in SQL Query. The standard alias, defined in schema, is prefilled, but can be changed. Each entity/alias pair has to be joined to the report (see Selecting Joinpaths. Once the attribute is added to the report, the alias can not be modified in edit mode to ensure that selected joins are correct.
- Linked report: If a column is linked to another report, the data value of of the clicked column will be used as parameter at execution. Therefore the linked report has to expect the parameter condition on index n+1 and n is the index of the last parameter at execution condition of the initial report.

After adding an attribute by using button *Add*, another options are available:

- Sorting: By activating the icon on the left side of attribute block the sorting direction can be changed. An upward arrow indicates ascending, downward arrow indicates descending order of the attribute in result table. If no arrow is displayed, no sort direction for this attribute is used. If you mark more than one field for sorting, the first sort criteria is the first attribute marked for sorting in the list, and so on.
- Grouping column and -key: Grouping of columns enables reporting to do a second level aggregation, which groups all display fields. Any aggregation-function of the specific column data type is usable as grouping-function. After selecting the first grouping-function, grouping keys maybe specified. So a grouping line is displayed in the result if when one of the keys is changing. Here the sorting and order of the key columns is determining.

- Order of attributes: The order of attributes can be changed by activating an attribute block and moving it to another location or by using the arrow up/down function in the toolbar.
- Remove: By using the remove-function (displayed as red X in toolbar) the attribute can be removed from report.

Fig. 2.6 shows how this options can be used to select only the ProcessInstance OID, which is not an attribute in ProcessInstance entity. So the expression is set to *pi.oid* and data type is *Long*.

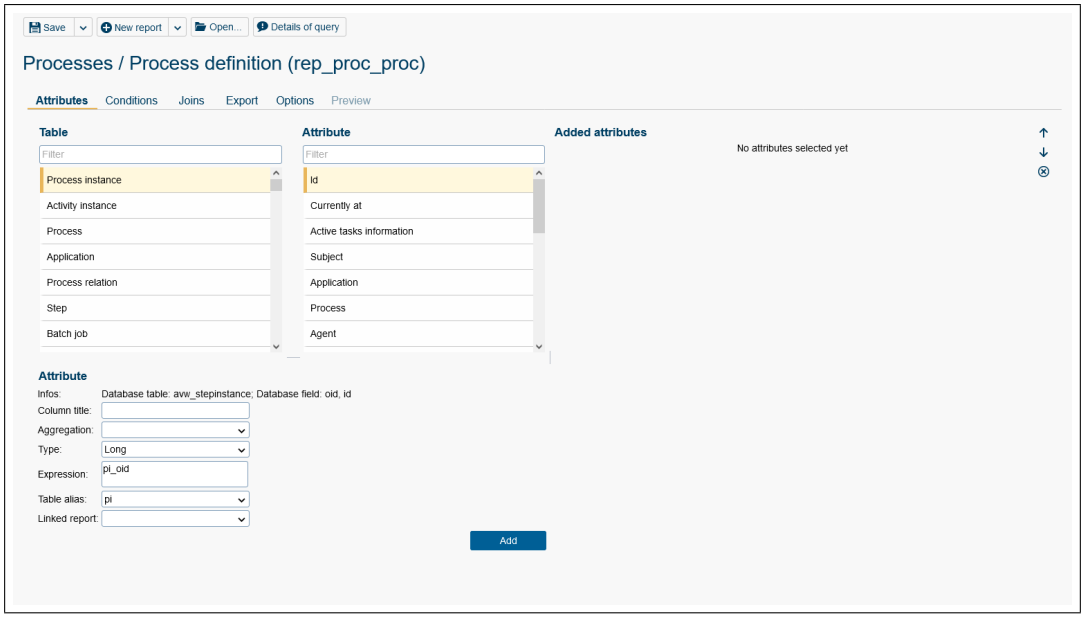

Figure 2.6: **Selecting OID of ProcessInstance via Extended Options**

2.2 Query conditions

With conditions you can reduce the amount of lines in the result set. Click the tab "Conditions" to get the condition attributes. The attributes are categorized in:

Objects (Applications, Organizational units, Processes, Tasks, Agents) You can specify sets of objects where the searched value is included or not included, see Fig. 2.7. When selecting agents you can also select the user, who executes the query. Clicking on the *Add* Icon will open a select mask. Select the object and click on *OK* to take the value. To close the object selection, click on *Close*.

Text fields (Id, Name, Subject) Specifies a string search. If operator is *like* or *not like* any string is found, which has the given text as substring (Infix-search). To do a prefix search, add a % at the end of the search pattern, for postfix search add a % at the beginning of the search pattern. Activate the checkbox *Ignore Case* to enable case insensitive search.

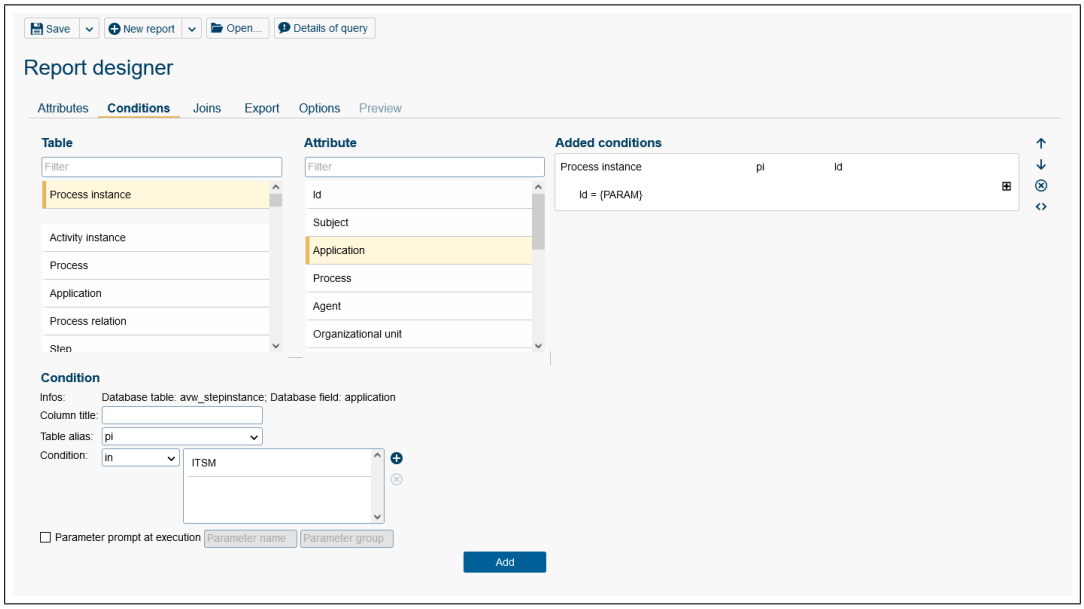

Figure 2.7: **Condition for objects**

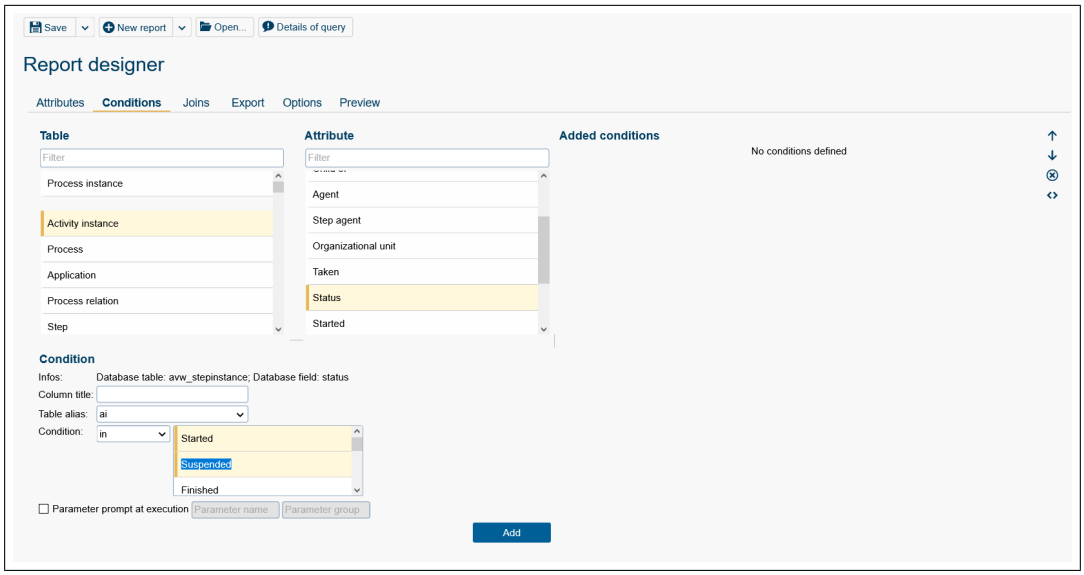

Figure 2.8: **Conditions for states**

States (Processinstance state, Activityinstance state) Select a item from the lists *Status* (see Fig. 2.8).

Dates (Started, Finished, Duedate) You have several possibilities:

- Specify a date, where the attribute value is greater (after) or equal to this date,
- The attribute value is less (before) this date.

- You specify a time span in hours, days, weeks, month, or years. The checkbox "exactly" defines the end point of the time span: When checked, the time span ends at execution time, else it ends at the begin of the execution hour, day, month, etc. The attribute value must lay in the time span ending at this defined end point.
- The attribute value is less than the start point of the time span described above.
- The attribute value is null,
- The attribute value is not null.

Fig. 2.10 show some examples of date conditions.

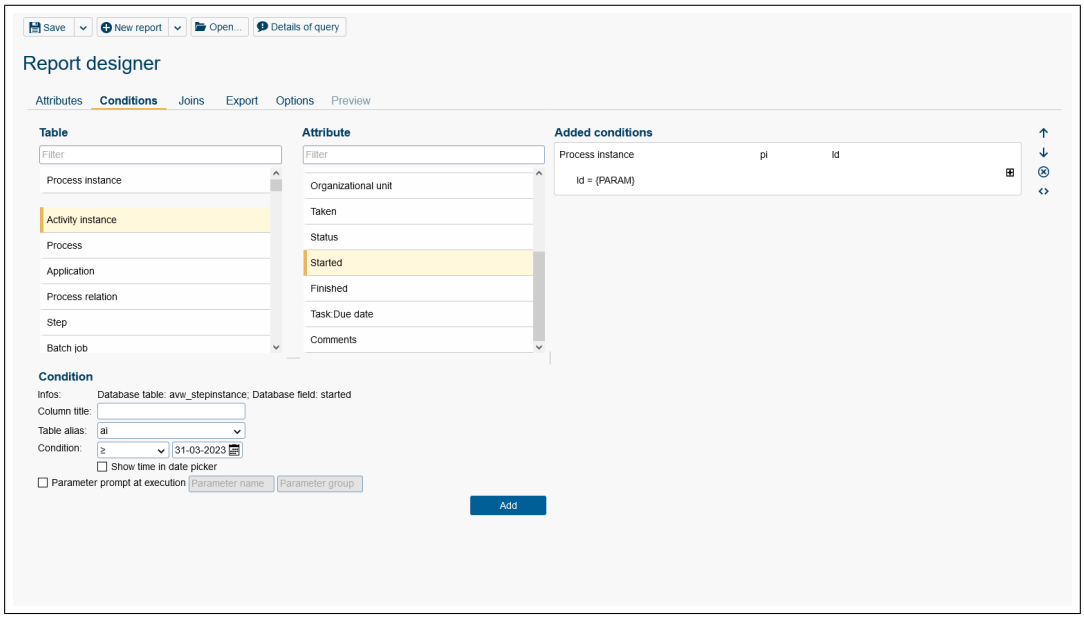

Figure 2.9: **Conditions for date fields**

Form fields Select a form of the list *Tables* and then a formfield of the list *Attributes* to set conditions in the condition mask.
If you activate the icon *Form* of an attribute, you will get a preview of the form, where you can enter values in the fields. After applying, the entered values will be set as condition in the tab *Conditions*.
Otherwise you can specify a formfield condition like any other condition. see Fig. 2.12.

2.2.1 Connection of conditions

All conditions you add can be connected with "AND" or "OR" (AND is "stronger" than OR). Furthermore, you can set parentheses by selecting one or more conditions and click on the corresponding toolbar function *(..) Set Parentheses*.

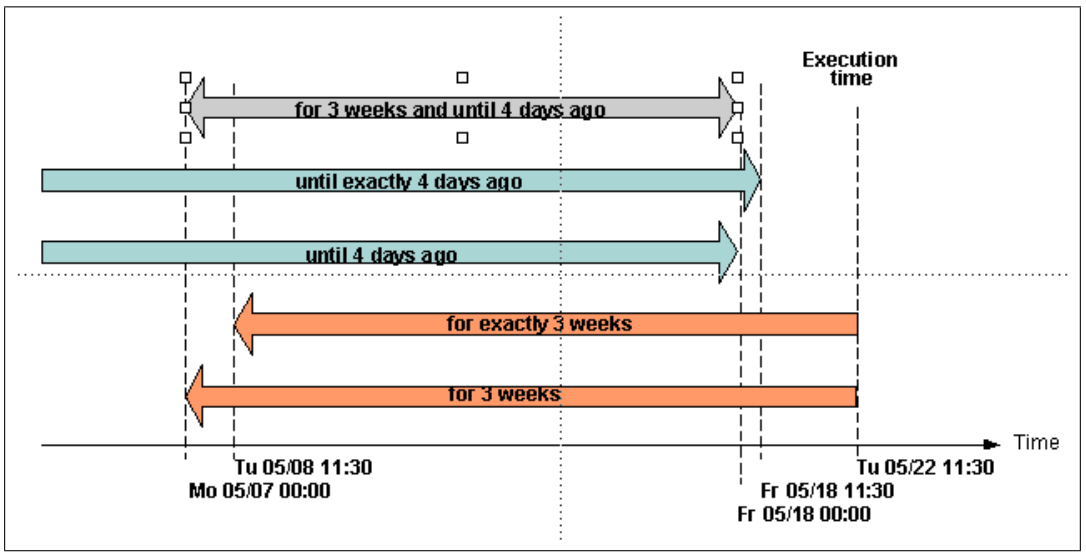

Figure 2.10: **Date condition examples**

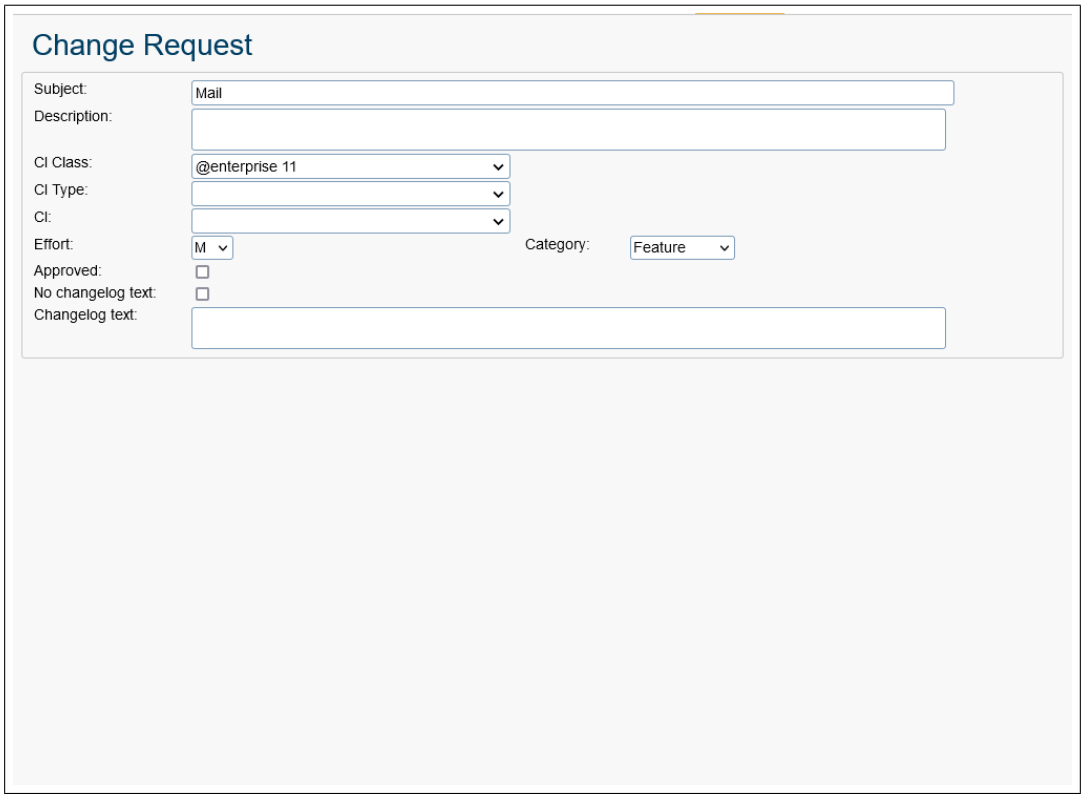

Figure 2.11: **Set condition by entering values in the formfields**

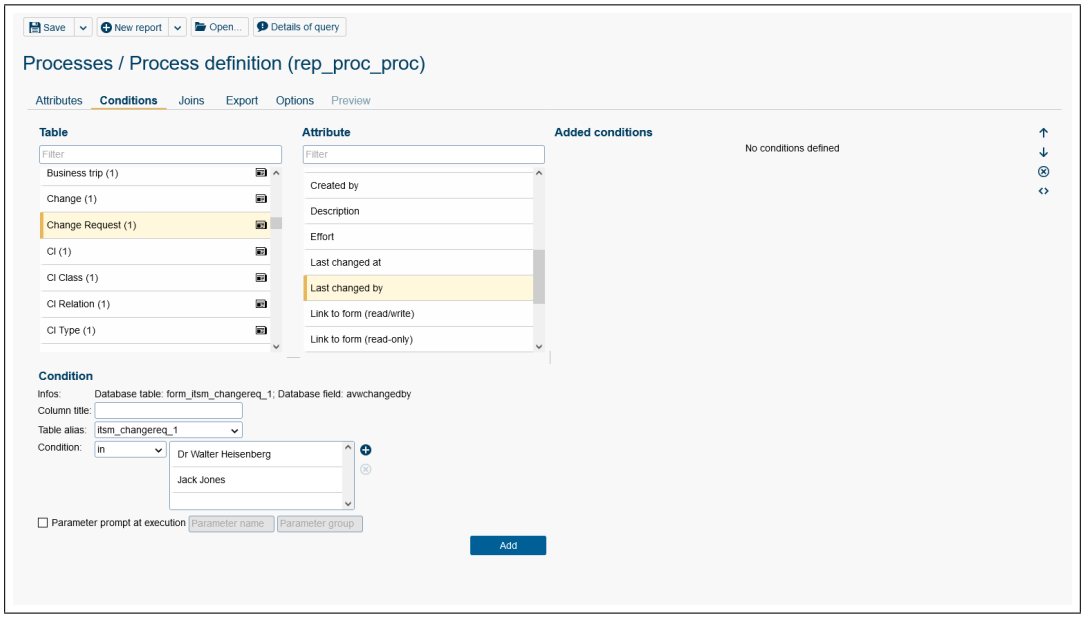

Figure 2.12: **Set formfield condition by entering values in condition form**

2.2.2 Queries with parameters

You can define queries where some parameters are set at execution time (explicit parameter). For example, a query returning all instances of a given process. Which process you want, is specified at execution time. When you execute the query, the system shows you a mask for filling in the concrete values for the parameters (see Fig. 2.13).
To enable explicit parameter condition, activate the *Parameter prompt at execution* checkbox. Its even possible to assign default values to operator and value fields, which will be prefilled in the param mask. The label (= condition text) is created automatically by report designer with placeholder {PARAM}. This label is displayed on dialog *Result details* (see section Administrate reports) and the placeholder is replaced.
Attention: If the label has been changed manually, the placeholder is substituted by the value only. It is assumed that the operator is part of the label!

Another feature is the possibility to group parameters by entering an arbitrary text into field *Parameter group* (also possible to internationalize text by entering @@@). Parameters with the same parameter group text are displayed on param mask in the same group.
It is also possible to define queries with implicit parameters. The difference to explicit parameters is that the implicit parameters are determined by the context and not over a mask. For example: Agent at execution returns only results where agent is the user who executes the report.
The following implicit parameter conditions are possible:

- relative time conditions (e.g. since 3 days)
- Agent at execution conditions

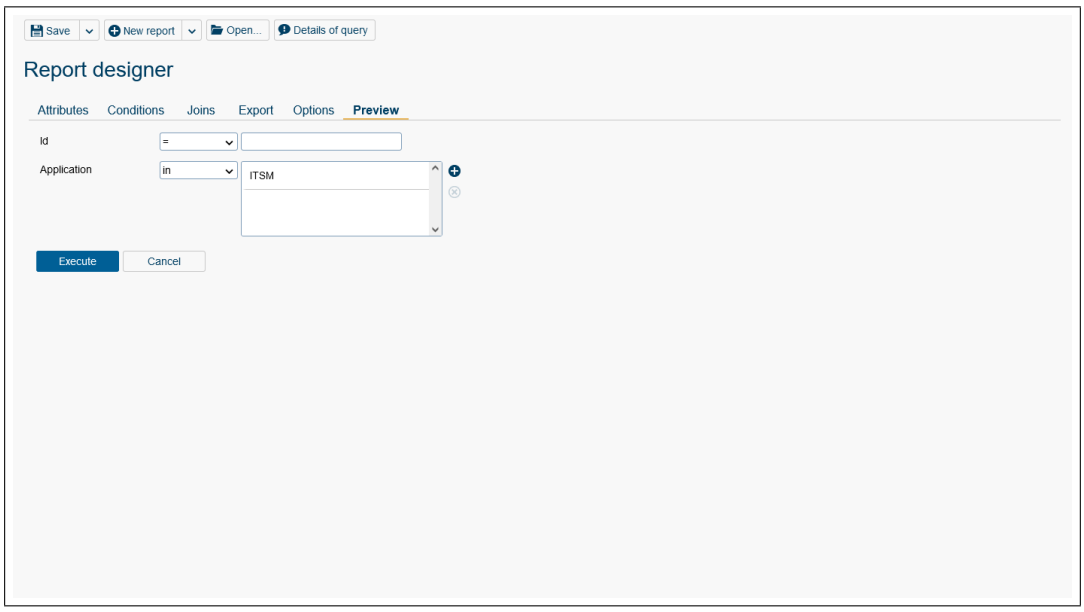

Figure 2.13: **Setting the parameters**

2.2.3 User-defined conditions

As last case in the section about conditions we present the possibility to define arbitrary conditions. You should know some SQL to define such conditions (see example Example 4 (User-defined condition)). Depending on the selected display or condition fields you can refer the table aliases used in the report.

If the condition needs to consider the current user who executes the report, the placeholder &USER can be defined in the condition, e.g. u.oid = &USER

2.2.4 Date operators

Report Designer offers numerous possibilities to time-limit the search results. You can find some examples in the table 2.14.
* All results are the same as with the "=" operator, only the comparison is different.

Note: For "=" and "!=" operators with empty accuracy field *(minute(s), hour(s), etc.)*, the value "day(s)" is set.

2.3 *Selecting Joinpaths*

Reporting enables the user to define the way how the entities of the reports are joined. This increases the amount of possible reports while complexity of reporting designer increases too. Any new entity (except the first) has to be joined to the report. An entity is defined by its table alias. When adding a new column or condition of an entity, which wasn't added to the report already, engine will ask the user for the join to use. In tab *Joins* the existing joins

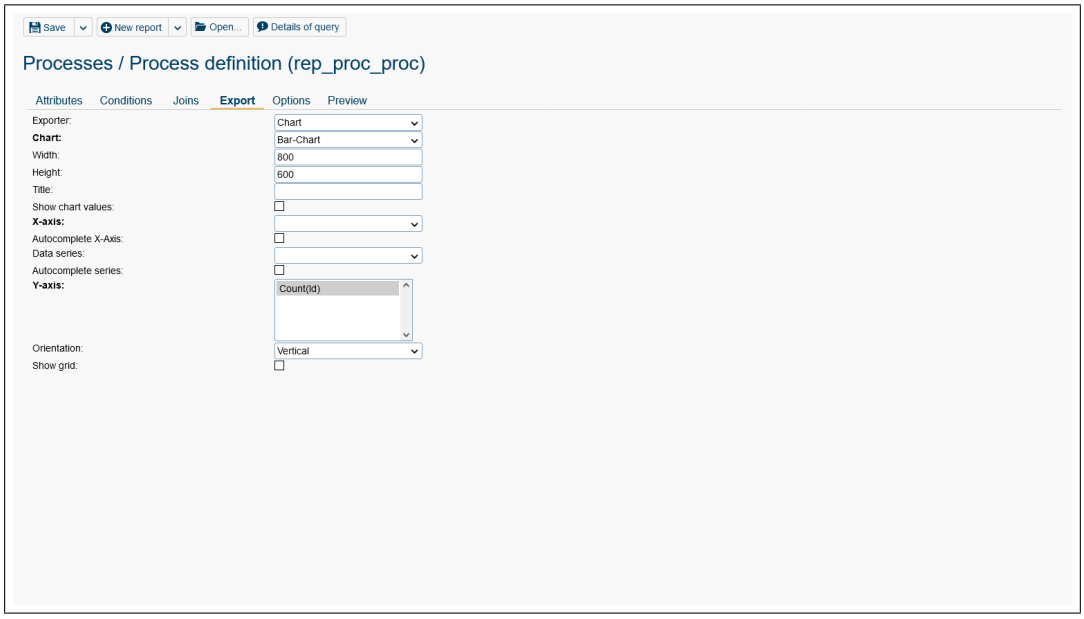

Figure 2.14: **Examples with different date operators**

can be adapted.

These possible join ways are predefined in the reporting schema (reporting.xml). Figure 2.15 shows the join path selection for entity activity instance to a report, which has already columns of the process instance. By default the system creates in the most cases INNER JOINS, but it is possible to change them to OUTER JOINS by activating the black triangle icon beside the radio button and selecting the desired join path. To apply the join path finally, select the radio button and click on *Ok*.
Fig. 2.16 shows how join selection works if there are more than one entity already added to the report and more than one predefined joinpaths exists to these entities. Each join target can be chosen with different paths, so in front of the target (e.g: *ProcessInstance*) the triangle icon is located. Click on this icon to see the possible paths, choose the path and click on *OK*.
If the predefined joins do not include the needed join, user-defined joins can be declared. Fig. 2.17 shows how 2 entities (processinstance pi, processinstance pi2) can be joined via id to get all subprocesses.

2.4 *Export of query results*

In this tab it is also possible to define the exporter for query results. You can select between following exporters:

- **HTML Table**: The result is shown in the browser (default). The fields *Clickable value*, *Type of selection* and *Hide clickable value in result table* are relevant, if toolbar functions are defined for the report result (see section Toolbar functions). It is also

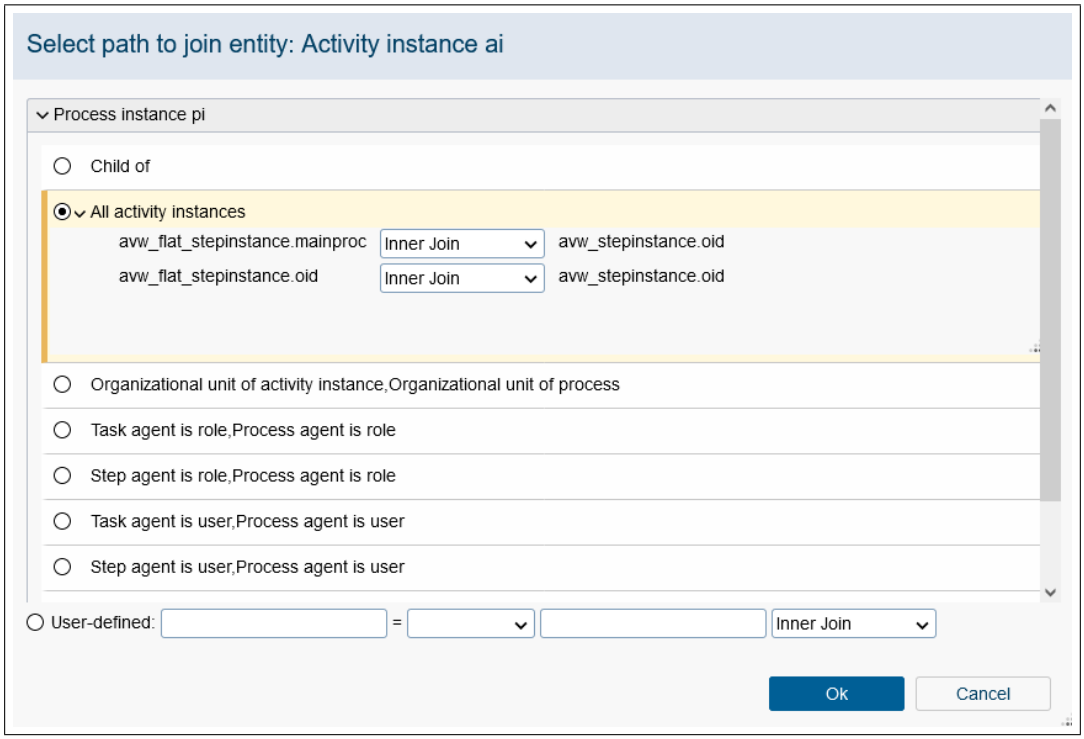

Figure 2.15: **Selecting Joinpath for Processinstance and Activityinstance**

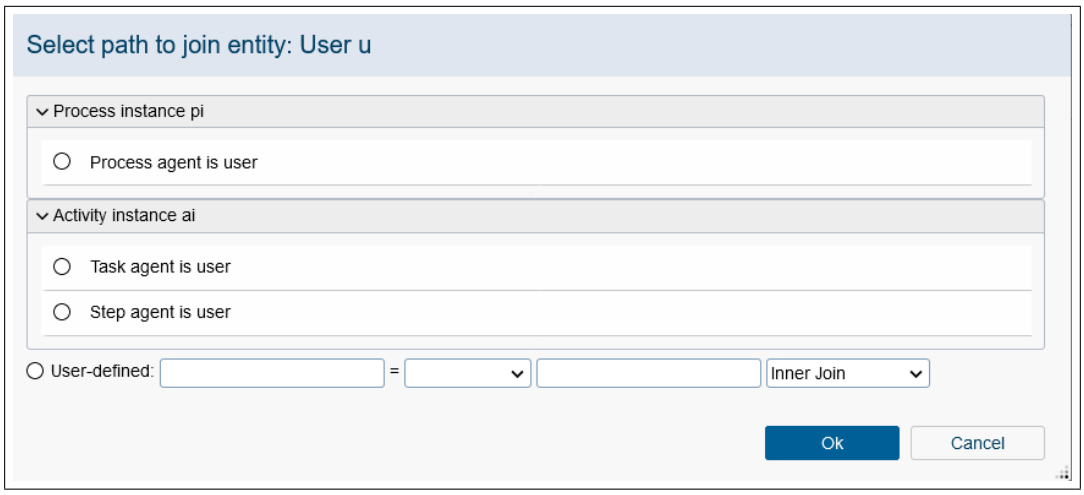

Figure 2.16: **Joining User either to ProcessInstance or ActivityInstance**

possible to define a fix width of the result table (in pixel). In the *Column settings* the user can specify the width (in percent depending on result table size) and CSS Class for each column. The checkboxes *Invisible* and *May not hide* allow to set the column visible or collapsible. Additionally there is possibility to specify the width, visibility, CSS Class, RowSpan and ColSpan for the *@enterprise* Mobile Client in the *Mobile*

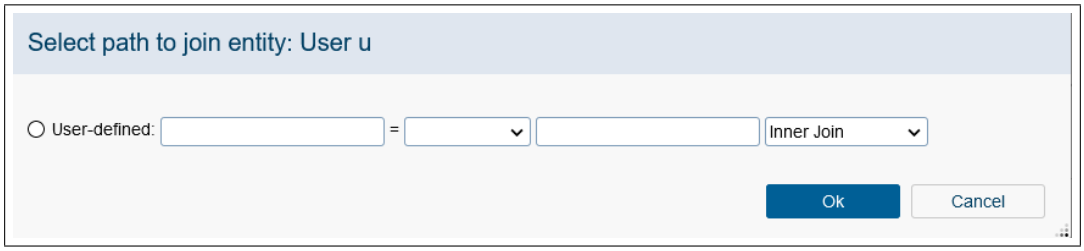

Figure 2.17: **User defined Joins**

Column Settings.

Hint: Please note that the *ColSpan* is only applicable in the second or greater rows and only if its value is greater than 1.

- **Chart**: See chapter Showing results graphically
- **Export to Excel**: The query result will be stored in a XLS-file, which can be downloaded. If checkbox *Hide info* is activated, the query result details (e.g. condition details, execution time, etc.) are not written into file.
- **Delimiter Separated Values - Exporter**: Similar to *Export to Excel*. Additionally you have to enter a *Delimiter* to separate the results. The target file is a CSV-file, which can be downloaded. If data fields contain the delimiter, its replaced by *@delimiter_removed@*. The checkbox *Use UTF-16 Little Endian Encoding* can be used, if result contains special characters (e.g. umlauts) to get correct representation in file. If checkbox *Hide info* is activated, the query result details (e.g. condition details, execution time, etc.) are not written into file.
- **XML export**: Similar to *Export to Excel*, whereas you can determine a style sheet. The target file is a XML-file. If checkbox *Hide info* is activated, the query result details (e.g. condition details, execution time, etc.) are not written into file.
- **PDF-Exporter**: Report result will be written into a pdf file using the *itext* library. Page format an the widths of the columns (in percent of page size) are configurable (see fig. 2.18). The PDF-Exporter uses the font *Helvetica*. This can be configured by a hidden configuration parameter. If the result shows Unicode signs, the configured font has to be a truetype font. If checkbox *Hide info* is activated, the query result details (e.g. condition details, execution time, etc.) are not written into file.
- **Open-Office Export:** This exporter allows to export in diverse output formats by defining an Open-Office template (ODT template). The output formates are PDF, ODT, DOC and DOCX. Please note that DOCX (Office Open XML) is usable with LibreOffice only! The template must be placed in the classpath of *@enterprise* and consists of placeholders in XPath syntax (see *Application Development Guide* - section *Office Templates*). In the following an example of an ODT template is shown:

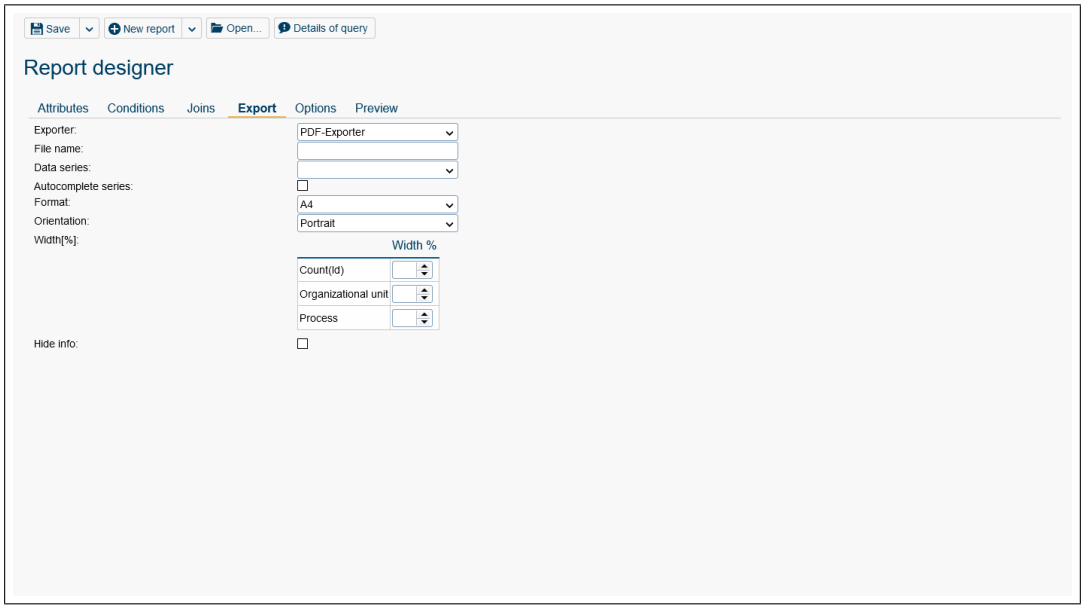

Figure 2.18: **Options of PDF-Exporter**

```
${REPEAT $row in $data}
 ${com.groiss.reporting.export.OpenOfficeExporter.
   getRowData($query,$row,0)}
 ${com.groiss.reporting.export.OpenOfficeExporter.
   getRowData($query,$row,1)}
${END}
```

The method *com.groiss.reporting.export.OpenOfficeExporter.getRowData* allows to read one row of the whole result table. The first parameter $query defines that the report result should be read. The second parameter $row indicates that a row should be read and the third parameter indicates which cell of the row should be read. If checkbox *Hide info* is activated, the query result details (e.g. condition details, execution time, etc.) are not written into file.

- **Escalation Exporter:** With this exporter escalations can be fired for processes of the reporting result. For this purpose you have to select a *Process Id Attribute* which identifies a process (should be process instance id). Furthermore an *Action* can be defined which is executed, if escalation is fired:

  - Send an email: Sends an email to current agents or a self-defined list of email recipients
  - Call method: The entered Java method is executed

  The report must be stored first before this exporter can be used! Escalations are fired one time for each process of reporting result. If escalation has been fired correctly for a process, this one will be skipped in future.

- **SVG chart:** Analog to option *Chart* (see also section Showing results graphically). In addition to ordinary charts it is possible to define a drill-down report (linked report): If a display attribute is linked to another report, the data value of of the clicked display value will be used as parameter at execution. Therefore the linked report has to expect the parameter condition on index n+1 and n is the index of the last parameter at execution condition of the initial report.

Note that stored reports save the chosen export type. If you save the report e.g. with export excel, the report will be exported to excel by default.

2.4.1 Matrix reports

Grid-based exporters (HTML, Excel, Pdf ...) are able to format the data in a matrix-like structure. The reports require at least one aggregated attribute (for example *count*). By using the export options, you have to specify an attribute being used as "data series". All the attribute values will be added dynamically as columns to the output. The aggregated values will be used as cell values.

You can easily illustrate the number of started processes per organizational unit and per agent. (see fig. 2.19).

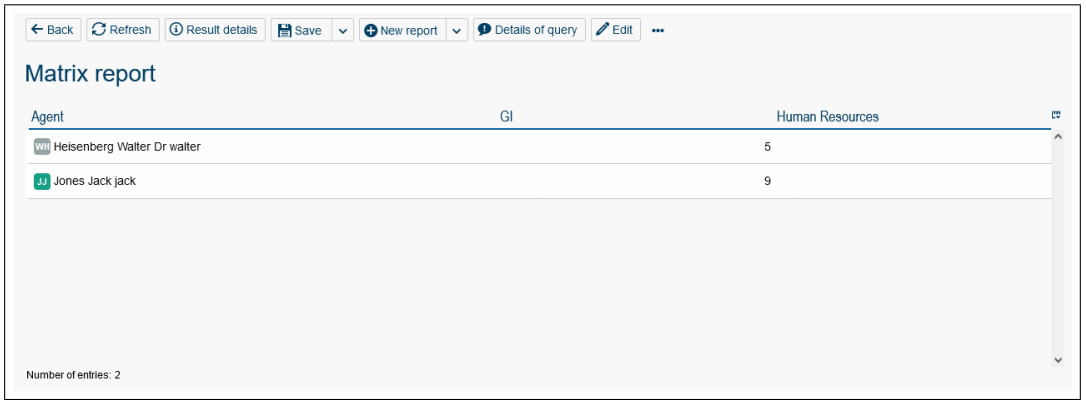

Figure 2.19: **Matrix report**

If using the HTML exporter, it is also possible to specify a "drill-down-report" on the aggregated values.

The values of the series-attribute can also be auto-completed. For this purpose you have to activate the corresponding checkbox. By using the auto-completion, gaps in the values will be filled. Auto-completion works on mapped attributes (value lists), persistent data and date fields. Mapped attributes and persistent data will be completed with known values in the DB or the schema respectively. If auto-completion for date-fields is used, gaps between the minimum and maximum value will be filled.

2.4.2 Showing results graphically

Additionally to the tabular output you can show the query results graphically. You can determine the chart settings before or after the execution.

A chart consists of categories and data series (see Fig. 2.21). Each row of the tabular view is a series. The categories are the values of the numerical display fields. Chosen categories are displayed as colored areas in a pie-chart.

With this exporter you can determine the chart type (see Fig. 2.20). You can enter a chart-title and specify the *Height* and *Width* of the chart (is equivalent to the area inside the x- and y-axis). You can choose between following chart types:

- Bar-Chart
- Pie-Chart
- Line-Chart
- Pie-Chart 3D
- Bar-Chart 3D
- Stacked-Bar-Chart
- Multiple-Pie-Chart

Figure 2.20: **Diagram settings**

For all bar-charts you can select the *Orientation* (vertical or horizontal). You can also determine the data series or categories for the chart.

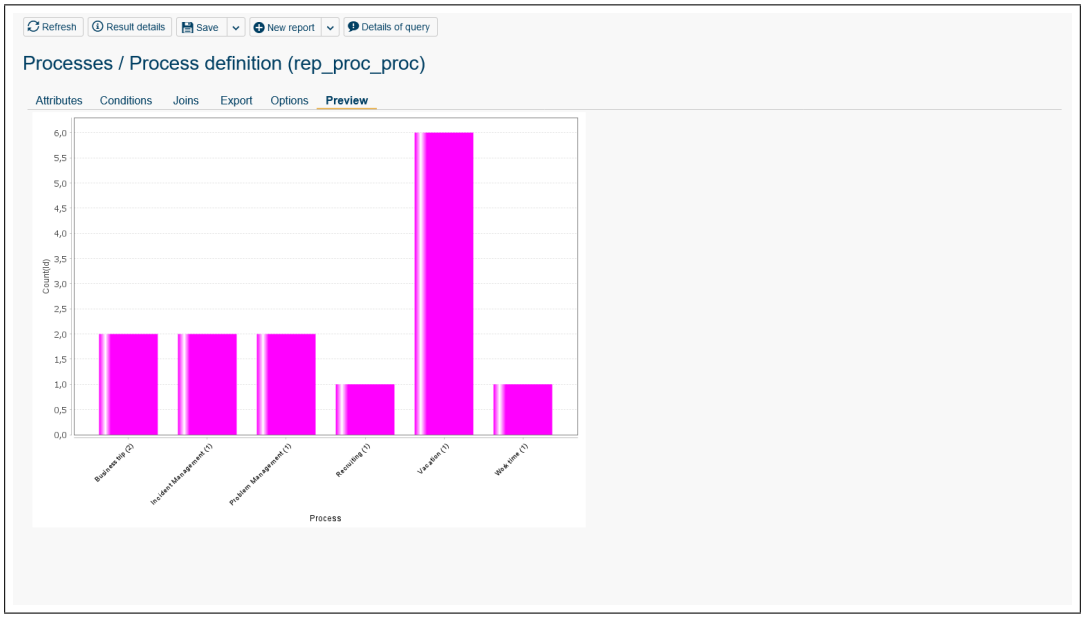

Figure 2.21: **Example diagram**

2.5 Report options

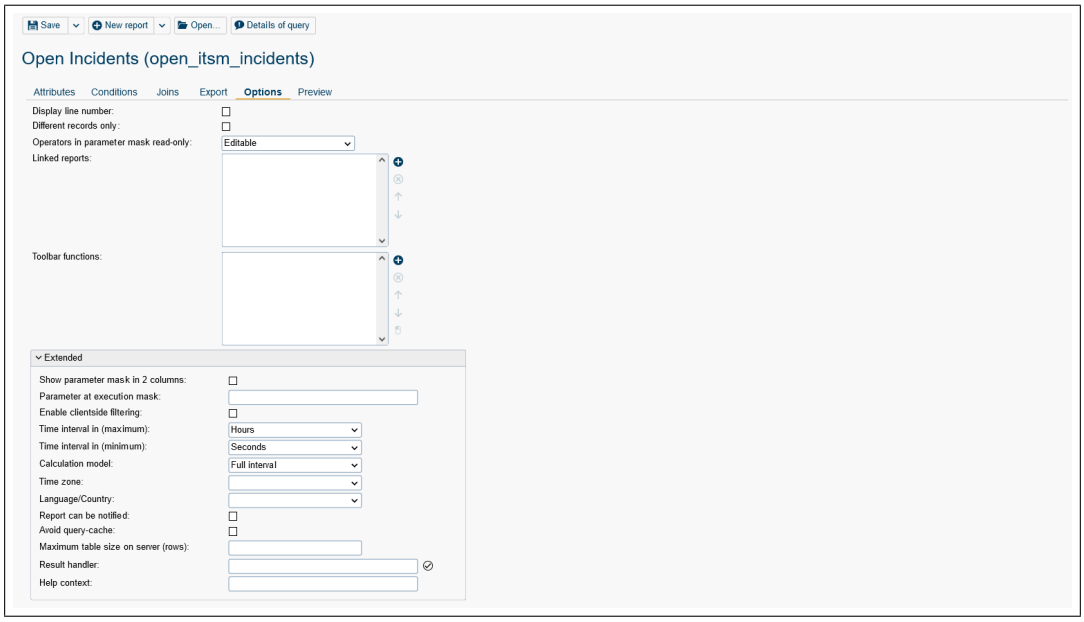

Figure 2.22: **Report options**

2.5.1 Checkbox *Display line number*

If this checkbox is activated, row numbers will be displayed in result.

2.5.2 Checkbox *Different records only*

If this checkbox is activated, same records will not be shown a second time in the result.

Example: Search all process names, which are available.
We assume that the process name *parfor* exists a second time. If this checkbox is activated and you start the search, you will see the name *parfor* one time only.

2.5.3 Show operators at parameter mask

Default values for operator and value of parameter at execution conditions can be chosen since @*enterprise* 8.0. Depending on the selection of following value the representation of operators can be manipulated:

- Editable: Operators are editable.
- Disable: Operators are not editable, but are displayed.
- Hidden: Operators are hidden and therefor not editable.

More information about param mask is available in section Queries with parameters.

2.5.4 Related reports

Any stored report can be linked to a report. HTML links to execute the reports are displayed in the upper-right corner of the result page.

2.5.5 Toolbar functions

Analog to tables in GUI configuration (see *Administration Guide*), toolbar functions can be configured. These functions can be declared as task functions in administration or as XML node in a config-file. The toolbar function can use the result data, if in tab *Export*

1. the exporter *HTML Table* is selected,
2. a column in field *Clickable value* is defined which is used as referenced column for the function and
3. the *Type of selection* is selected depending on the expectation of the function (e.g. *Set read/unread* expects only one selected entry).

One toolbar function may be marked as double-click action only which fires when the user double-clicks on the result line.

Function	*Referenced column*
Set read/unread	Activity instance/Id
Set due date	Process instance/Id (only process due date can be set), Activity instance/Id (process and task due)
Into clipboard	Process instance/Id, Activity instance/Id
Copy to ...	Activity instance/Id
Add parfor steps	Activity instance/Id
Set priority	Process instance/Id, Activity instance/Id
Add process relation	Process instance/Id, Activity instance/Id
Follow the process	Process instance/Id, Activity instance/Id

Table 2.1: ***@enterprise* Functions as toolbar functions**

Usage of standard *@enterprise* functions

@enterprise offers by default functions which could be used in Reporting, but it is not guaranteed that the reporting result will be refreshed after executing the function! Following table contains some standard functions and which referenced column is needed:
Following action id's can be used for entries in worklist:

Action id	*Name*	*Referenced column*
finish	Complete	Activity instance/Id
finishAndSelect	Complete and assign	Activity instance/Id
goBack	Go back	Activity instance/Id
seeLater	Suspend	Activity instance/Id
makeVersion	Make version	Activity instance/Id
setAgent	Reassign	Activity instance/Id
cut	Cut	Activity instance/Id

Table 2.2: ***@enterprise* Action id's as toolbar functions**

2.5.6 Parameter in section *Extended*

Checkbox *Show parameter mask in 2 columns*

If this checkox is activated, the defined conditions are displayed in 2 columns on param mask.

More information about param mask is available in section Queries with parameters.

Parameter at execution mask

This field allows to define an own parameter mask. This mask can be either a HTML file (must be valid XHTML), which is interpreted as DOJO widget (see example below) or a DOJO widget itself (*.js file). This field must be stored in classpath under

alllangs/scripts, e.g. if file is stored under
classes/alllangs/scripts/ep/widget/MyParamMask.html, then the value of this field must be ep/widget/MyParamMask.html.

All input fields on parameter mask require the attribute data-dojo-attach-point. If a parameter name has been entered for the condition (can be done via GUI and is recommended), the value of data-dojo-attach-point must be defined under this directive:
paramname_value
If no parameter name has been entered, the automatic assigned number (always starting with 0 for first condition) must be used under this directive: value<NR>

Example for parameter mask as HTML file:

```
<div>
 <h1>Customized param mask</h1>
 <table>
  <tr>
   <td>Process-ID: </td>
   <td><input data-dojo-type="dijit/form/ValidationTextBox"
        data-dojo-attach-point="value0"></input>
   </td>
  </tr>
  <tr>
   <td>Priority: </td>
   <td><input data-dojo-type="dijit/form/ValidationTextBox"
        data-dojo-attach-point="c1_value"></input>
   </td>
  </tr>
 </table>
</div>
```

The parameter mask is interpreted as DOJO widget, therefor no further HTML tags (e.g. html, body, etc.) are needed! In the example above the text value + the number of the condition (in this case the number of first condition in XML) is used as *data-dojo-attach-point* for *Process-ID*; the condition *Priority* uses the entered parameter name + _value.

Hint: The options *Show operators at parameter mask* and *Show parameter mask in 2 columns* have no effect for the own defined parameter mask!

Checkbox *Enable client side filtering*

If this checkox is activated, it enables client side refining of query result (Search function).

Time interval

Can be shown in seconds, minutes, hours, days or weeks.

Computing of time intervals (Calculation model)

For computing time intervals you can use two computing models: "full interval" and "no weekends", in the latter the weekends are removed from the time interval. In the reporting schema you can add additional computing models. See the API documentation for how to implement a time model.

Timezone and Language/Country

To execute a report in a specific timezone and/or Language/Country, you can specify it here. If nothing is chosen, the timezone and locale of the executing user is taken.

Checkbox *Report can be notified*

This checkbox defines, if the report should be refreshed by a notification. The notification is done with method `QueryEngine.sendReportRefreshToClient(String reportId)` - for more details see *@enterprise* APIDoc.

Checkbox *Avoid query-cache*

If this checkbox is activated, the report result will not been cached anymore and will be read from database every time.

Maximum table size on server (rows)

Maximum table size the server will handle. If the table size exceeds this value, the operation is canceled and an error message is produced.

Result handler

Enter an own implemented result handler class which is responsible for displaying the report result. This class has to implement the interface *com.groiss.reporting.ResultModifier*.

Help context

Add a context help to a report. For more informations please take a look *Application Development Guide, Section5.4.1 Using context sensitive help in applications*.

2.6 SQL report

The function *SQL report* is a special form of the Report Designer and allows the creation of reports with the aid of SQL statements. The display attributes must be added separately, the FROM-block allows the definition of an arbitrary SQL statement. The SQL statement can also have some parameterizable conditions.

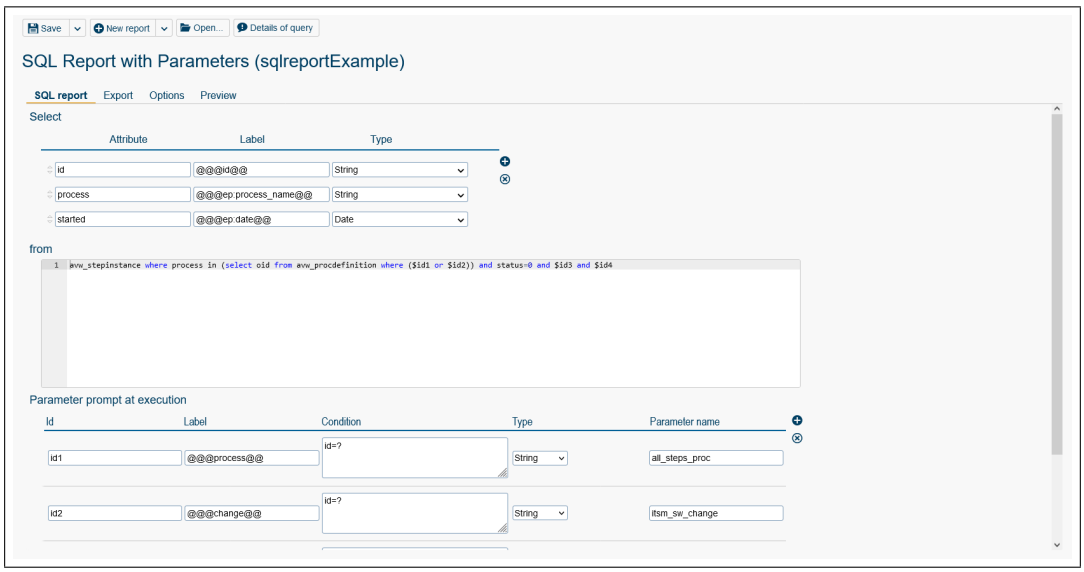

Figure 2.23: **SQL report**

2.6.1 Display attributes

Display attributes can be added/delete with the appropriate toolbar functions and moved in their positions via drag & drop.
The display attribute are consists of following elements:

- SQL attribute: The column name of a table.
- Label: The column label (header) which is displayed in report result.
- Data type: Selection of diverse data types:
  - *String*
    Description: any string
    SQL attribute: any varchar field of a database table
  - *Double*
    Description: rational number
    SQL attribute: any decimal field of a database table
  - *Long*
    Description: natural number
    SQL attribute: decimal field (without decimal places)
  - *Date*
    Description: date object
    SQL attribute: date field in database table
  - *TimeInterval*
    Description: time interval
    SQL attribute: 2 comma separated timestamps

- *Persistent*
  Description: reference to another database table
  SQL attribute: referenced oid field, the corresponding _class field is added automatically
- *Implementation of Persistent*
  Description: reference to another database table
  SQL attribute: referenced oid field
- *com.groiss.reporting.data.impl.ProcessLink*
  Description: Link to process history, the id is used as link label
  SQL attribute: OID and ID (comma separated) of a process instance
- *Implementation of com.groiss.reporting.data.ReportingData*
  Description: formatted due to the implementation
  SQL attribute: expected attributes due to implementation

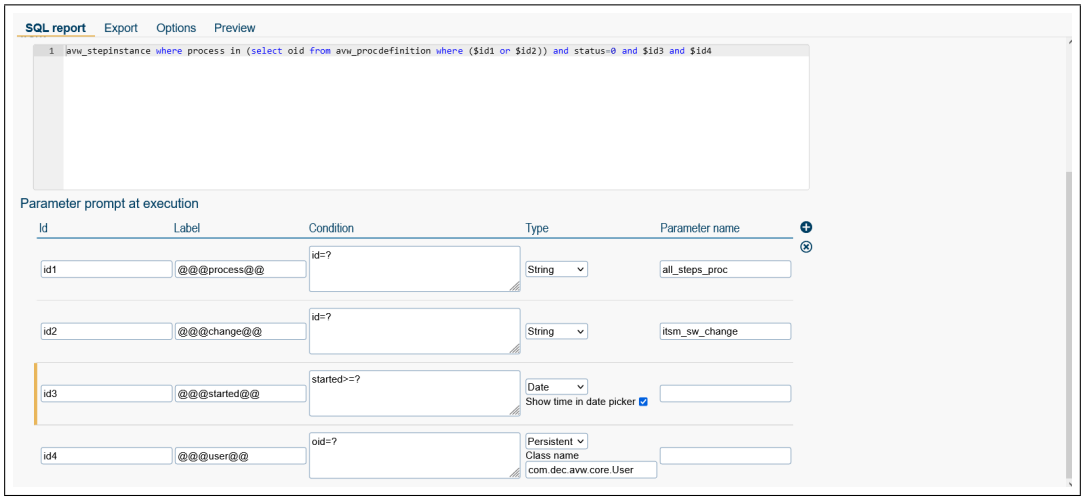

Figure 2.24: **SQL report with parameters**

2.6.2 Parameter prompt at execution

Similar to user-defined conditions in the classic Report Designer, parameterizable conditions are also available when entering a SQL report. All parameters entered in the FROM block must have an ID and it is necessary to define one or more placeholders in the SQL statement (e.g. *$mycondition1*). All placeholders are replaced by the parameter values when executing a report. A placeholder can appear more than once in the SQL statement, i.e. it will be replaced by the same condition at execution. The parameters are defined in section **Parameter prompt when executing** (see figure 2.24) which consists of the following elements:

- **Id**: A mandatory field in which the parameter ID from the SQL statement is entered (without placeholder character *$*).

- *Label*: An optional parameter label that is displayed in the parameter input.
- **Condition**: This is a mandatory field. Here you can construct conditions.
- *Type*: Here you can choose between different data types:
  - *String:* any string
  - *Double:* rational number
  - *Long:* natural number
  - *Date:* date object
  - *Persistent:* reference to another database table
- *Parameter name*: Similar to the standard reports, it is possible to define for several parameters the same name with the effect that several conditions are affected with just one input field. An example can be seen in the figure 2.24 where two fields are named *user*. When executing, the parameter mask is displayed as in the figure 2.25. There is only one *user* field although two conditions are defined. If you change a parameter name (e.g. into *user2*), two *user* fields will be displayed in the parameter mask and each field will take its own value.

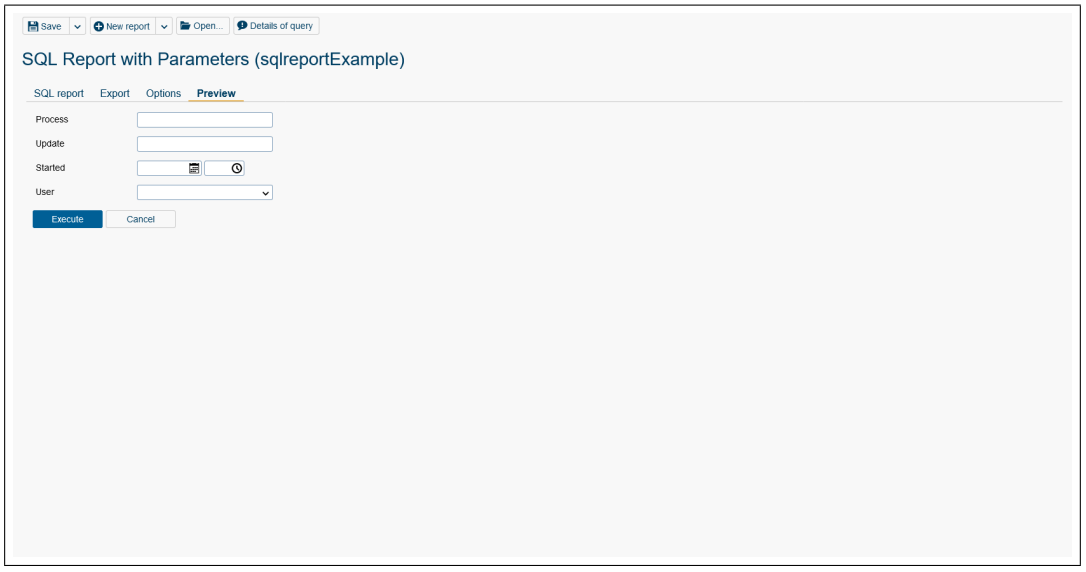

Figure 2.25: **Parameter mask**

2.7 Report subscriptions

Users who have the *Execute* right for a specific report and the permission to execute the *Subscribe report* function are able to subscribe to reports. Before a subscription can be added any report must be executed first. After executing the *Subscribe report* toolbar function a parameter mask for configuring a subscription is displayed with following fields (see fig. 2.26):

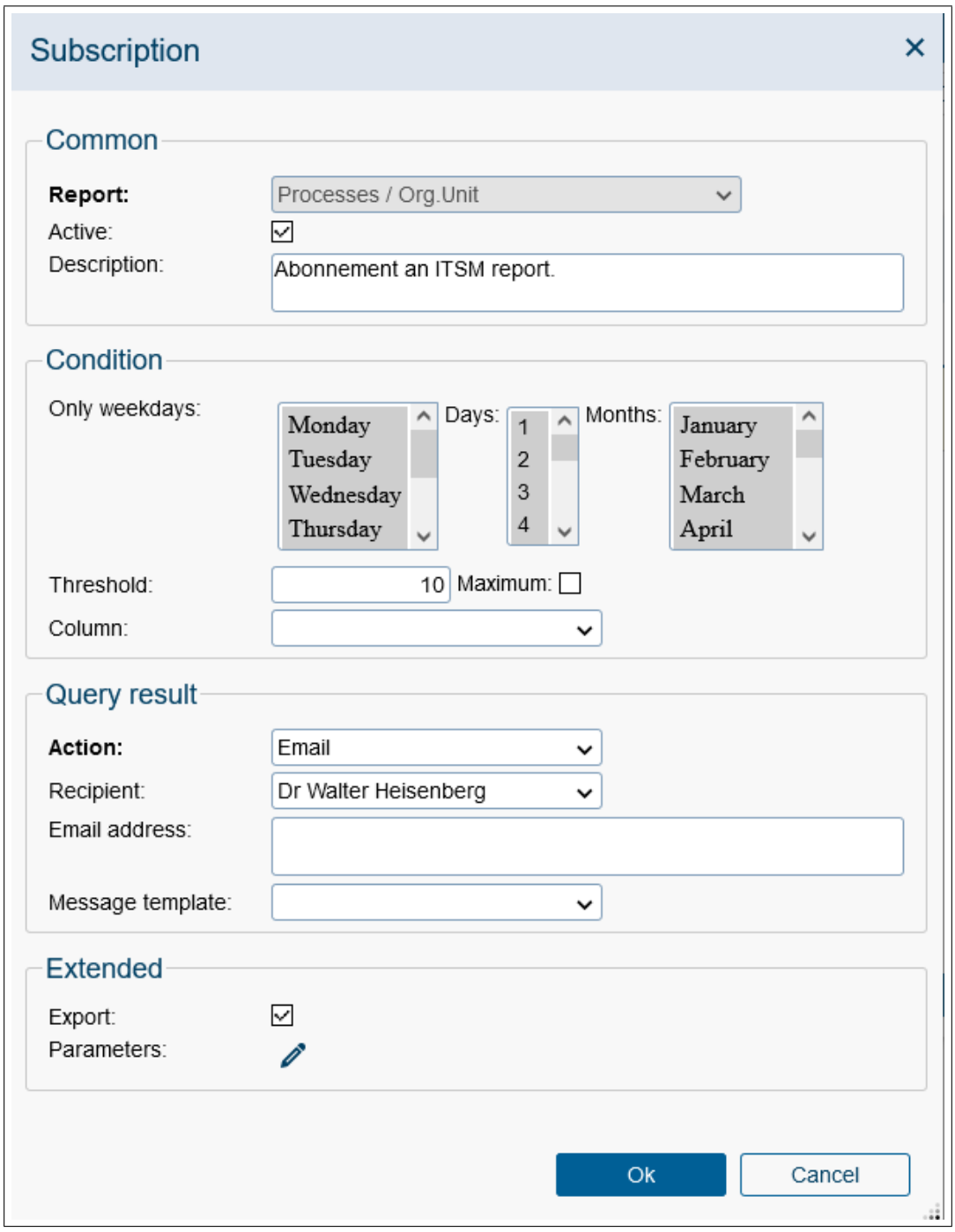

Figure 2.26: **Configuration of a report subscription**

- **Report**: Name of the report. This field cannot be edited, the executed report is preselected automatically.
- *Active*: If checkbox is activated, the subscription will be executed (e.g. by *ReportSubscriptionTimer*).

- *Description*: Free text which allows to add a detailed description.
- *Restricted period selection*:
  - *only Weekdays*: Subscription is executed only on selected days of the week.
  - *Days*: Subscription is executed only on selected days of the month.
  - *Months*: Subscription is executed only on selected months.
- *Threshold*: Here you define the limit value that must be reached in order to process the report result. If no column is specified, the number of rows is used.
- *Maximum*: If activated, the limit is set to the maximum number of rows. If not checked, the limit is set to the minimum number of rows.
- *Column*: The *Threshold* and *Maximum* parameters are applied on the selected column.
- **Action**: This selections allows to define an action which should be applied on this subscription:
  - *Email* - An email with the report result is sent to a defined recipient(s):
    - *Recipient***: @*enterprise* user to whom the email should be sent. The current agent is preselected.
    - *Email address**: Custom list of email addresses. If email addresses are entered here, no email will be sent to the selected user in the *recipient* field.
    - *Message template**: Here you can select a pre-defined message template. If no specific message template is selected, the default message template with the id *reportTimer* is used.
  - *File**:
    - *Destination directory*: Here you can specify the target directory (absolute or relative to server root path) in which the report results are going to be saved.
  - *DMS*:
    - *Folder*: Here you can select a folder of the @*enterprise* DMS in which the report results are going to be saved.
  - *Start process*:
    - *Recipient***: The process will be created and displayed in the worklist of the selected user.
    - *Process*: A new instance of the selected process will be started.
  - *None**: No additional action will be taken. This could be necessary, if the report itself has defined an appropriate Exporter.
- **Export***: If activated, this subscription is exported too with the report when using the admin function *Export in XML*. More information about how to export a report can be found in *System Administration Guide*, section *Import/Export*.

- **Parameters***: The parameters specified when executing a parameterizable report are automatically recognized and entered here. Please **do not** edit this field without proper knowledge.

* Only available, if the user has the *Statistics* right
** Without the *Statistics* right, other users cannot be selected

Activating the *Manage subscriptions* function allows the user to see or edit all report subscriptions (if right *Statistics* is assigned). Users without *Statistics* right are allowed to see and edit their own subscriptions only.

3 Examples

This sections shows a few example reports. You need the right "statistics" to create and execute the following examples.

3.1 Example 1 (Aggregation; Date fields)

Compute the minimum, average, and maximum work times per step for all finished tasks (in days), which has been started in the last month. Sort the results by process-id.
Select the following display fields:

- ProcessInstance:Id .... to have the process in the result table (ascending sorted).
- ActivityInstance:Worktime with aggregation *minimum*.
- ActivityInstance:Worktime with aggregation *average*.
- ActivityInstance:Worktime with aggregation *maximum*.

Then add the following conditions:

- ActivityInstance:Started >= exactly 1 month
- ProzessInstance:Status in (Finished)

Then set the option "Time Interval in" to "days".

3.2 Example 2 (Grouping over time intervals; implicit and explicit parameters)

How often per week a task of a specified process is taken by the user executing the query? The process is given as parameter (specified at run-time). Sort the result by the week.
Select the following display fields:

- Task:Name

- ActivityInstance:Id with aggregation "count"
- ActivityInstance:Taken with date format "week" and sort order "ascending"

Then add the following conditions:

- ActivityInstance:Taken is not empty .... select only tasks which has been taken
- ProcessInstance:Id = PARAM ... process is specified when query is executed
- ActivityInstance:Agent = Agent at execution

3.3 Example 3 (Null-Values: grouping, aggregation; form fields)

Which user has started how many processes, where in the form "Jobform" the subject field is empty and the recipient is "James".
Additionally show the field "description" of the "Jobform" form.

Select the following display fields:

- ProcessInstance:Agent ... start agent of a process,
- ProcessInstance:Id with aggregation "'count"' ... instead of count(Process:agent) to count processes without agents also,
- Jobform:description ... the form field description of the form Jobform.

Then add the following conditions:

- Jobform_1:recipient = Berger
- Jobform_1:subj is null

3.4 Example 4 (User-defined condition)

Which agents of aborted tasks can be reached via email?

We select the display field "ActivityInstance:Agent" and the following conditions:

- ActivityInstance:Status in (aborted)
- ai.agent in (select oid from avw_user where email is not null) ... a user-defined condition using a sub-query with the table avw_user. First select the button *User-defined* in tab *Conditions* and select the entry *SQL command*. After activating the button *Ok* the *Extended Options* of this condition can be opened where field *SQL command* can be filled.

4 Administrate reports

If you want to reuse queries, it is possible to store it. Then you can open the query for executing, editing or deleting. Following functions are available:

- **Save**: This function allows you to save your report. Enter an *Id* and a *Name* and activate the button *Save* or *Save and close* (see Fig. 4.1). If you have already opened a stored report and activate this function, you can store this report with the same name (=*Update*) or delete it by activating the button *Delete*.
  If the appropriate rights has been assigned to current user, beside the field *Name* the I18n-link is displayed with the translations of this key. The translations can be edited directly (e.g. in case of admin session) and stored with tooltip dialog button *Save* or viewed only. The changes are stored in the resource file of the given application.

- **New report**: With this function you can create a new search. All fields will be cleared.

- **Open...**: By activating this function a dialog will be opened where a stored report can be selected and loaded into Report Designer.

- **Details of query**: In this mask the build query is shown in XML-format and as SQL-statement. If you want to see full details of a query, you have to activate this function in the result-mask.
  There is a possibility to display the query plan. This works only when using *Oracle* and if the user has the *STAT* right. More information about how to configure this functionality can be found in section *Installation/ Database Preparation/Oracle* of the *Installation and Configuration* manual.

- **Refresh**: This function is available in the result-mask only and refreshes the result. If queries with user defined parameters are refreshed with this function, the parameter input mask will not be displayed anymore.

- **Reload after new parameter input:** This function is available in the result-mask only when a query with user defined parameter inputs has been started and refreshes the result with the possibility to use other parameters.

- **Result details**: This function is available in the result-mask only and shows details of the query result.

- **Edit**: This function is available in search result only. With this function you can edit your build query.
- **Print...:** This function is available for HTML tables only and allows to print the displayed result.
- **Export Options**: See chapters Showing results graphically and Export of query results.
- **Export Options**: The function will be displayed only if the user has appropriate rights. More information can be found in the chapter Report subscriptions .

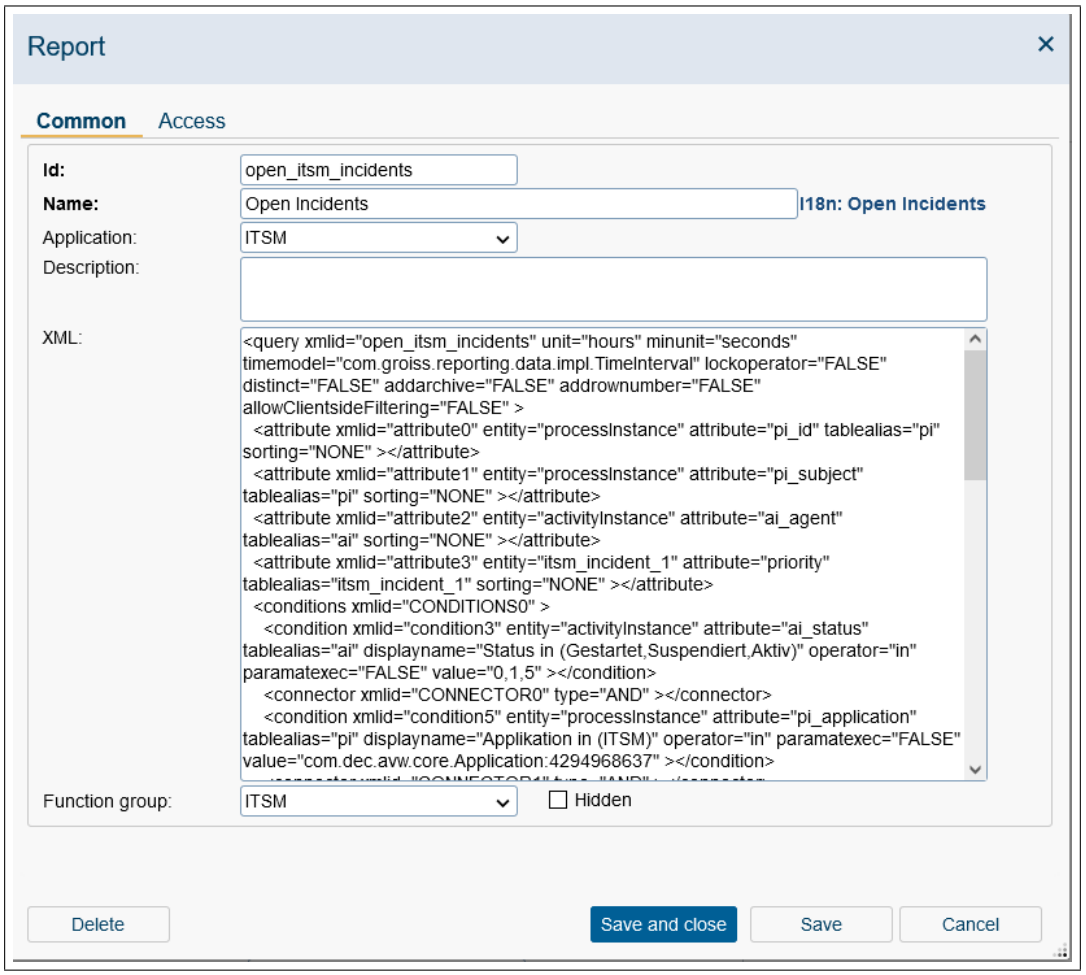

Figure 4.1: **Save query**

5 Configuration of the reporting component

5.1 Permissions

You need the right *Statistic* for creating reports and select all fields.
You can define which user may execute which query. One possibility to define this, is when you save the query (see above). It you want to give a user or a role the right to execute a stored report, go the the list of users or roles, click on permissions and add a new permission: Select the right "execute", the object class "Reports" and select the query you want. To enable user to set permissions to stored reports, the user needs the right "edit permissions".
Note that in the list of stored reports every user sees only the queries he may execute. The creator of a report has the permission to edit, delete, execute and assign permissions to his report.

5.2 Public execution (without login)

Reports are executable without being logged in if the user *guest* is allowed to execute the report. The URL below will show you a list of executable reports. Log out before calling the url to check public queries.

`http://<host>:<port>/<contextRoot>/servlet.method/`
`com.groiss.reporting.gui2.PublicReportingGui.listQueries`

Add the parameter
`groupId=<idOfFunctionGroup>` or `group=<nameOfFunctionGroup>`
to get a list of only one functiongroup.

5.3 Version independent views for forms

If you want to create queries on forms version-independent, you need to create views over the form versions. To do so, go to the form administration, select a form and click the button "Create View". You will see the SQL-statement you can now execute. For further information how to create a view, please take a look in the *System Administration Guide*.

5.4 Server Configuration

The following configuration parameter affect the reporting component.

- Maximum Table Size on Server (rows): Due to the possible very large amount of report results, reporting may cause performance problems. To avoid this, you can specify a maximum amount of lines. If the report returns more results, the report is canceled and a exception is thrown.
- Cache Interval (minutes): To avoid high server load its able to cache query results. This parameter defines how long a query should be cached.
- Maximum Number of Cached Queries: This parameter defines the size of the cache.
- Maximum Number of Simultaneous Queries: To ensure that reporting does not cost too much performance, this parameter limits the number of queries which can be executed concurrently.
- Maximum Number of Startable Queries: Defines the amount of queries which are queued if the maximum number of simultaneous queries is exceeded.
- Order Process-Ids by OID: When checked, ProcessIds are sorted by the Processoid.
- Show all rows, even when no View Right: DMSObjects in result set will not be displayed if the user who executes the report has no view right on the DMSObject. This feature can be disabled by activating this checkbox.
- Open Forms in Edit Mode: If checked forms links (attribute: oid) open the forms editable.

Hidden configuration parameters are documented in the application development guide, section *Hidden Configuration* of chapter *Using the Reporting API*.

5.5 Reporting-Cache

Report results are cached in order to take load off the DB when the same report is executed repeatedly. This guarantees a faster result, especially if an existing result is to be exported in a different format.
Explicitly "refreshing" a report result via the toolbar action always reloads the data. In the "Result details" you can see when the query was actually executed.
How long and how many results are saved can be set in the configuration (Configuration/Search).
The behavior can be deactivated for special reports, which should always use the "live" data. To do this, activate the "Avoid query-cache" checkbox in the "Options", "Extended" tab in the report designer.

6 Support

For further questions, contact us under the email `support@groiss.com`.