

Celoxis API Guide

Version 4.5.0

Celoxis API Guide: Version 4.5.0

Copyright © 2001-2008 Celoxis Technologies, Pvt. Ltd.

Abstract

This document serves as a guide to the Celoxis application user interface (API).

Table of Contents

1. API Guide Overview	1
1.1. Introduction	1
1.2. Examples	1
1.3. Representation Formats	1
1.4. Authentication	2
1.5. Calling a function	2
1.6. Error Handling and Debugging	2
1.7. Rate Limiting	3
2. Function Reference	4
2.1. Query	4
2.2. Create-projects	6
2.3. Update-projects	7
2.4. Create-tasks	8
2.5. Update-tasks	9
2.6. Create-task-updates	10
2.7. Create-time-entries	11
2.8. Create-users	12
2.9. Query-utilization	13
3. Table Reference	16
3.1. db_application	17
3.2. db_assignment	17
3.3. db_audit	18
3.4. db_calendar_event	18
3.5. db_category	19
3.6. db_category_app	19
3.7. db_category_sla	19
3.8. db_company	20
3.9. db_contact	21
3.10. db_custom_field_def	22
3.11. db_custom_field_rel	23
3.12. db_custom_field_val	23
3.13. db_custom_report	24
3.14. db_document	24
3.15. db_document_ci	25
3.16. db_document_co	25
3.17. db_event_invitation	26
3.18. db_expense	26
3.19. db_expense_code	27
3.20. db_expense_item	27
3.21. db_favorite	28
3.22. db_folder	28
3.23. db_folder_entry	29
3.24. db_forum	29
3.25. db_forum_subscription	29
3.26. db_forum_topics	30
3.27. db_fs_file	30
3.28. db_group	30
3.29. db_group_members	31
3.30. db_message	31
3.31. db_message_attachments	32

3.32. db_note	32
3.33. db_phone_call	33
3.34. db_process	33
3.35. db_process_document	34
3.36. db_process_form	34
3.37. db_process_state_mgr	35
3.38. db_process_type	35
3.39. db_process_type_form	35
3.40. db_process_update	36
3.41. db_process_update_form	36
3.42. db_project	36
3.43. db_project_category	39
3.44. db_project_phase	39
3.45. db_rate	40
3.46. db_rate_type	40
3.47. db_reminder	41
3.48. db_role_membership	41
3.49. db_role_permission	41
3.50. db_sla	42
3.51. db_sla_entry	42
3.52. db_state	42
3.53. db_state_transition	43
3.54. db_state_transition_form	43
3.55. db_task	44
3.56. db_task_dependency	47
3.57. db_task_expenses	48
3.58. db_task_messages	48
3.59. db_task_notes	48
3.60. db_task_phone_calls	48
3.61. db_task_update	49
3.62. db_time_code	49
3.63. db_time_entry	49
3.64. db_url	50
3.65. db_user	51
3.66. db_working_calendar	52
3.67. db_working_time	52

List of Tables

3.1. db_application	17
3.2. db_assignment	17
3.3. db_audit	18
3.4. db_calendar_event	18
3.5. db_category	19
3.6. db_category_app	19
3.7. db_category_sla	19
3.8. db_company	20
3.9. db_contact	21
3.10. db_custom_field_def	22
3.11. db_custom_field_rel	23
3.12. db_custom_field_val	23
3.13. db_custom_report	24
3.14. db_document	24
3.15. db_document_ci	25
3.16. db_document_co	26
3.17. db_event_invitation	26
3.18. db_expense	26
3.19. db_expense_code	27
3.20. db_expense_item	28
3.21. db_favorite	28
3.22. db_folder	28
3.23. db_folder_entry	29
3.24. db_forum	29
3.25. db_forum_subscription	29
3.26. db_forum_topics	30
3.27. db_fs_file	30
3.28. db_group	30
3.29. db_group_members	31
3.30. db_message	31
3.31. db_message_attachments	32
3.32. db_note	32
3.33. db_phone_call	33
3.34. db_process	33
3.35. db_process_document	34
3.36. db_process_form	34
3.37. db_process_state_mgr	35
3.38. db_process_type	35
3.39. db_process_type_form	35
3.40. db_process_update	36
3.41. db_process_update_form	36
3.42. db_project	36
3.43. db_project_category	39
3.44. db_project_phase	40
3.45. db_rate	40
3.46. db_rate_type	40
3.47. db_reminder	41
3.48. db_role_membership	41
3.49. db_role_permission	41
3.50. db_sla	42

3.51. db_sla_entry	42
3.52. db_state	42
3.53. db_state_transition	43
3.54. db_state_transition_form	43
3.55. db_task	44
3.56. db_task_dependency	47
3.57. db_task_expenses	48
3.58. db_task_messages	48
3.59. db_task_notes	48
3.60. db_task_phone_calls	48
3.61. db_task_update	49
3.62. db_time_code	49
3.63. db_time_entry	50
3.64. db_url	50
3.65. db_user	51
3.66. db_working_calendar	52
3.67. db_working_time	52

Chapter 1. API Guide Overview

Table of Contents

1.1. Introduction	1
1.2. Examples	1
1.3. Representation Formats	1
1.4. Authentication	2
1.5. Calling a function	2
1.6. Error Handling and Debugging	2
1.7. Rate Limiting	3

1.1. Introduction

The Celoxis Application Programming Interface (API) is simply another way to access Celoxis — one that makes it easy for third-party and custom tools to programatically access and interact with the service. The API is extremely simple to use, and this guide should provide everything you need to implement software that works with Celoxis.

The principal design objectives behind the API are:

- Simplicity and easy of use
- Language independence - you should be able to use any programming language
- Flexibility - you should be able to leverage the full power of SQL

1.2. Examples

All the URLs in the following examples must be prefixed by `http://<celoxis-server-name>:<port>/psa/`

Find the list of all projects that are in progress

```
api.do?function=query&table=db_project&status_id=1
```

Fetch details of task with id 100

```
api.do?function=query&table=db_task&id=100
```

Fetch tasks assigned to user with id 200

```
api.do?function=query&table=db_task&join-with=db_assignment.task_id&db_assignmentmen
```

Execute a query using SQL

```
api.do?function=query&table=generic&sql=select+id+from+db_project+where+fixed_pr
```

You can find a detailed list of functions supported in the [function reference](#) chapter.

1.3. Representation Formats

The API uses XML or JSON for requests as well as responses. This is governed by the parameter `data-format` that is to be passed in every request. It can be either XML (the default) or JSON. All XML documents sent to and returned from the service should be XML 1.0, encoded as UTF-8. In case of JSON, all strings are encoded in UTF-8 format. Date and times are represented as: YYYY-MM-DDThh:mm:ss e.g. 10 August 2008, 1:45 PM will be represented

by 2008-08-10T13:45:00. Booleans are to be represented as strings true and false. Booleans in database columns are stored as 't' for true and 'f' for false.

When the format is XML, property names and their values are returned as XML elements and its contents respectively. In case of JSON, they become the object property name and value respectively.

1.4. Authentication

The first step in using the Celoxis API is to authenticate to the server. This user must have administrative privileges. We recommend that you create a special user dedicated to using the API. On successful authentication, you will receive a token that should be passed in every API call else the call will be rejected with an authentication error. You can authenticate any user not just administrators and operations can be performed on behalf of any user; it is only the token that must be the result of authentication of an administrative user.

You can authenticate like this:

```
api.do?function=login&username=[username]&password=[password]&company-code=[company]
```

If data format is XML and your authentication is successful, you will receive XML with two elements: token and user_id. For example:

```
<result><token>ae48ef19bcfd1</token><user_id>12918</user_id></result>
```

In case the data format is JSON, you will receive an object with properties token and user_id respectively. On how to handle errors, see [Error Handling and Debugging \(Section 1.6\) \[2\]](#)

1.5. Calling a function

Before you call any function, the software must first authenticate itself. Read [Authentication \(Section 1.4\)\[2\]](#) for more details. Once you authenticate, you are ready to use the API.

To call a function, you should specify at least the following parameters:

function

The name of the function

token

The authentication token that you received on [authentication](#)

data-format

One of xml or json, default is xml. This specifies format in which data will be sent and received.

In addition, you should specify the parameters required by the function you are calling. See the [function reference](#) for more details. On success, the X-Success HTTP header has the value 1 while in case of failure it has the value 0. An error message is present in the X-Error-Message header.



Important

All parameter names and values are case sensitive. Token, TOKEN and token are treated differently.

1.6. Error Handling and Debugging

On success, the X-Success HTTP header has the value 1 while in case of failure it has the value 0. On success, the HTTP status code will be the standard 200 OK. On error, an error message is present in the X-Error-Message HTTP header. In case of failure, the following HTTP status codes are returned :

401 - Unauthorized

This means that the authentication token is invalid. You will have to authenticate before calling the operation

404 - Not Found

In case something (e.g. table) is not found

400 - Bad Request

There is some other error

403 - Forbidden

Either the function call you are making is not allowed or you have exceeded the [rate limit](#).

In addition, the following headers are present that can help in further debugging :

X-Last-SQL

The last SQL statement executed

X-Error-Exception

The name of the error class e.g. SQLException, NotFoundException, etc.

1.7. Rate Limiting

Clients for hosted service are allowed 600 requests every hour. This is enough to make just over ten requests per minute, per hour, which should meet the needs of most applications. Rate limiting applies only to authenticated API requests. Rate limiting does not apply for installable version customers.

Notification that a client has exceeded the rate limit will be sent via HTTP status code of 403. The X-Error-Message header will describe the error.

In order for you not to exceed the limits, we recommend that your application that uses the Celoxis API cache items that change infrequently e.g. time codes, expense codes and users. We also recommend that you submit items in bulk while updating through the API.

If you are developing an application that requires more frequent requests to the Celoxis API, please contact us.

Chapter 2. Function Reference

Table of Contents

2.1. Query	4
2.2. Create-projects	6
2.3. Update-projects	7
2.4. Create-tasks	8
2.5. Update-tasks	9
2.6. Create-task-updates	10
2.7. Create-time-entries	11
2.8. Create-users	12
2.9. Query-utilization	13

2.1. Query

This function is used to query the database. Celoxis API has two ways to query the database. One is a simple version where you sepecify the table and the filter conditions while in the other you write the SQL yourselves. The latter is available only for installable version customers.

Function name

query

Parameters

- `table` (Required). One of the valid table names as mentioned in the [table reference](#) or `generic` for a sql query.
- *Filtering conditions* (Optional). You can specify filter conditions for rows in one of the following formats:
 1. `<column-name>=<value>`
 2. `<table-name>.<column-name>=<value>`
 3. `<table-name>.<column-name>.<operator>=<value>`

In case of 1, table name is assumed to be the table parameter. In case of 1 and 2 the operator is assumed to be '='. You can have operators like '=', '>', '<', '>=', '<=' and '!='. For example, `db_task.plan_start.>=2008-10-1` would mean tasks that start after 1st of October 2008.

You can specify multiple such conditions, separated by the '&' sign. If you specify the column name more than once, it is assumed that you want an 'OR' condition between its multiple values. For example to fetch a list of incomplete tasks in project whose id is 200 you should specify:

```
api.do?function=query&table=db_task&project_id=200&db_task.percent_complete.<=
```

- `join-with` (Optional). If you want the table to be joined with another table, you specify the 'join-with' parameter. This will be of the form `<table-name>.<column-name>`. The main table (specified by the table parameter) will be joined on its primary key with the table and column specified with the 'join-with' parameter. For example to fetch a list of tasks assigned to user with id 7 with project phase 1, you should specify :

```
api.do?function=query&table=db_task&join-with=db_assignment.task_id&db_assignm
```

- `order-by` (Optional). The column on which to order the results by. You can even append `asc` or `desc` to the column to sort that column in ascending or descending order. Separate multiple columns by comma. For example, in case of the `db_task` table, to order the tasks by `plan_start` ascending and `plan_finish` descending, specifying the `order-by` parameter as `plan_start asc, plan_finish desc`. This is not applicable when `sql` is specified.
- `where` (Optional). The SQL `where` clause. This provides more flexibility than filter parameters.
- `sql` (Optional). The entire SQL query. This is only available for installable version customers.
- `skip` (Optional). The number of rows to skip before returning the remaining rows. This is not applicable when `sql` is specified.
- `take` (Optional). The number of rows to return. This is not applicable when `sql` is specified.

Return values

In case of XML data format, you will be returned an XML string with two elements: `count` (the number of rows) and `rows` (this will contain multiple `row` elements). In addition to the requested columns, computed values (see [table reference](#) for more information) and custom field values are also returned. For example :

```
<result>
  <count>8</count>
  <rows>
    <row>
      <f>
        <n>id</n>
        <v>18181</v>
      </f>
      <f>
        <n>summary</n>
        <v>Task 1</v>
      </f>
      <f> <!-- custom field -->
        <n>My Custom Field 1</n>
        <v>12345</v>
      </f>
      <f> <!-- computed columns -->
        <n>project_name</n>
        <v>My Project</v>
      </f>
      ...
    </row>

    <row>
      <f>
        <n>id</n>
        <v>281921</v>
      </f>
      <f>
        <n>summary</n>
        <v>Task 2</v>
      </f>
      ...
  </rows>
</result>
```

```
        </row>
        ...
    </rows>
</result>
```

In case of JSON, it will return an object with two properties: `count` (the number of rows) and `rows` (an array of objects).

2.2. Create-projects

Creates projects

Function name

`create-projects`

Parameters

- `data` (Required). It is an array of project fields. For example, in case the data format is XML, you should post something like this:

```
<data>
  <item> <!-- create projects -->
    <code>PRJ-101</code>
    <project_category_id>100</project_category_id>
    <manager_id>101</manager_id>
    <client_id>200</client_id>
    <phase_id>2</phase_id>
    <summary>Project #1</summary>
    <priority>2</priority>
    <plan_start>2008-09-02T08:00:00</plan_start>
    <deadline>2008-09-04T17:00:00</deadline>
    <budget>100000</budget>
  </item>
  <item>
    <code>PRJ-102</code>
    <project_category_id>100</project_category_id>
    <manager_id>101</manager_id>
    <client_id>201</client_id>
    <phase_id>2</phase_id>
    <summary>Project #2</summary>
    <priority>1</priority>
    <plan_start>2008-09-02T08:00:00</plan_start>
    <deadline>2008-09-04T17:00:00</deadline>
    <budget>200000</budget>
  </item>
</data>
```

In case of JSON, you should submit an array of objects with the same properties.

Here are the details of the fields:

- `project_category_id` (Required). Integer. ID of the project category.
- `summary` (Required). A short description of the project.

- `plan_start` (Required). Date. The planned start date of this project.
- `deadline` (Required). Date. The planned finish date of this project.
- `manager_id` (Required). Integer. ID of the project manager user.
- `client_id` (Required). Integer. ID of the client.
- `code` (Optional). A unique code for the project.
- `priority` (Optional). Integer between 1 (Very High) and 5 (Very Low)
- `detail` (Optional). A detailed description for this project.
- `phase_id` (Optional). Number. The id of the project phase.
- `budget` (Optional). Number. The budget for this project.

Return values

Nothing

2.3. Update-projects

Updates already existing projects

Function name

`update-projects`

Parameters

- `data` (Required). It is an array of project fields. For example, in case the data format is XML, you should post something like this:

```
<data>
  <item> <!-- update projects -->
    <id>1919191</id> <!-- id of the project to update -->
    <budget>100000</budget>
  </item>
  <item>
    <id>19198192</id>
    <summary>Project #2</summary>
    <priority>1</priority>
    <plan_start>2008-09-02T08:00:00</plan_start>
    <deadline>2008-09-04T17:00:00</deadline>
    <budget>200000</budget>
  </item>
</data>
```

In case of JSON, you should submit an array of objects with the same properties.

Fields are same as those for the [create-projects](#) function but all fields are optional. In addition the following fields are required:

- `id` (Required). Integer. ID of the project.

Return values

Nothing

2.4. Create-tasks

Creates tasks in projects.

Function name

create-tasks

Parameters

- data (Required). It is an array of task fields. For example, in case the data format is XML, you should post something like this:

```
<data>
  <item> <!-- create tasks -->
    <project_id>100</project_id>
    <summary>Task #1</summary>
    <detail>This is the task description</detail>
    <parent_id>12345</parent_id>
    <priority>2</priority>
    <plan_start>2008-09-02T08:00:00</plan_start>
    <plan_finish>2008-09-04T17:00:00</plan_finish>
    <assignments>10001/20%,100010/40h</assignments>
    <type_id>1</type_id>
    <state_id>1290</state_id>
  </item>
  <item>
    <project_id>100</project_id>
    <summary>Task #2</summary>
    <detail>This is the task description</detail>
    <parent_id>12345</parent_id>
    <plan_start>2008-09-03T08:00:00</plan_start>
    <plan_finish>2008-09-04T17:00:00</plan_finish>
    <assignments>10001/50%</assignments>
    <type_id>1</type_id>
    <state_id>1291</state_id>
  </item>
</data>
```

In case of JSON, you should submit an array of objects with the same properties.

Here are the details of the fields:

- project_id (Required). Integer. ID of the project in which the task is to be added
- summary (Required). A short description of the task.
- plan_start (Required). Date. The planned start date of this task.
- plan_finish (Required). Date. The planned finish date of this task.
- parent_id (Optional). Integer. ID of the parent task.

- `priority` (Optional). Integer between 1 (Very High) and 5 (Very Low)
- `detail` (Optional). A detailed description for this task.
- `assignments` (Optional). A string of the form `<user-id-1>/<allocation-1>,<user-id-2>/<allocation-2>`, etc. E.g. `10001/20%,100010/40h`
- `type_id` (Optional). Number. The task type ID.
- `state_id` (Optional). Number. The state associated with the task type.
- `can_fill_time` (Optional). Boolean. Whether time can be filled against this task.
- `budget` (Optional). Number. The budget for this task.
- `est_non_labor_cost` (Optional). Number. The estimated non-labor cost (a.k.a material cost) for this task.
- `fixed_cost` (Optional). Number. The fixed cost for this task.

Return values

Nothing

2.5. Update-tasks

Updates already existing tasks

Function name

`update-tasks`

Parameters

- `data` (Required). It is an array of task fields. For example, in case the data format is XML, you should post something like this:

```
<data>
  <item> <!-- update tasks -->
    <id>101311</id> <!-- id of the task to update -->
    <plan_start>2008-09-02T08:00:00</plan_start>
  </item>
  <item>
    <id>100</id> <!-- id of the task to update -->
    <summary>Task #2</summary>
    <plan_start>2008-09-03T08:00:00</plan_start>
  </item>
</data>
```

In case of JSON, you should submit an array of objects with the same properties.

Fields are same as those for the [create-tasks](#) function but all fields are optional. In addition the following fields are required:

- `id` (Required). Integer. ID of the task to update

Return values

Nothing

2.6. Create-task-updates

Creates status updates on tasks and also submits time entries for those tasks.

Function name

`create-task-updates`

Parameters

- `data` (Required). It is an array of fields. For example, in case the data format is XML, you should post something like this:

```
<data>
  <item> <!-- update task status and fill time -->
    <creator_id>100</creator_id>
    <date>2008-10-01</date>
    <task_id>12345</task_id>
    <percent_complete>50</percent_complete>
    <actual_start>2008-09-02T08:00:00</actual_start>
    <comments>Am half way there</comments>
    <hours_worked>2.5</hours_worked>
    <time_code_id>12</time_code_id>
    <rate_type_id>1</rate_type_id>
    <send-for-approval>>false</send-for-approval>
  </item>
  <item> <!-- the following just fills an update -->
    <creator_id>100</creator_id>
    <date>2008-10-01</date>
    <task_id>18381</task_id>
    <percent_complete>100</percent_complete>
    <actual_finish>2008-09-02T17:00:00</actual_start>
    <comments>Done!</comments>
  </item>
</data>
```

In case of JSON, you should submit an array of objects with the same properties.

Here are the details of the fields:

- `creator_id` (Required). Integer. The user on behalf of which the time entry is being submitted.
- `date` (Required). Date. The date for this update. The same date is used for the time entry.
- `task_id` (Required). Integer. The task for which this time entry is being filled.
- `percent_complete` (Optional). Integer between 0 and 100. The percent complete for this task.
- `actual_start` (Optional). Date. The date this task started being worked on.
- `actual_finish` (Optional). Date. The date this task was finished. Valid only when percent complete is 100.

- `comments` (Optional). The comments for this task update. The same will be used while filling time.
- `hours_worked` (Optional). Float. The number of hours. If not present, only task status is updated and no time entry is added.
- `time_code_id` (Optional, Required if `hours_worked` parameter is submitted). Integer. The time code ID for this time entry.
- `rate_type_id` (Optional). Integer. 1 for standard, 2 for overtime (in case your organization uses overtime rates), defaults to 1.
- `send_for_approval` (Optional). Boolean. Whether this time entry is to be sent for approval. If false, this will be saved.

Return values

Nothing

2.7. Create-time-entries

Creates a list of time entries and optionally sends them for approval.

Function name

`create-time-entries`

Parameters

- `data` (Required). It is an array of time entry fields. For example, in case the data format is XML, you should post something like this:

```
<data>
  <item>
    <creator_id>100</creator_id>
    <hours_worked>2.5</hours_worked>
    <user_comments>Fixed Bugs</user_comments>
    <for_date>2008-10-01</for_date>
    <task_id>12345</task_id>
    <time_code_id>12</time_code_id>
    <rate_type_id>1</rate_type_id>
    <is_billable>>false</is_billable>
    <is_costable>>true</is_costable>
  </item>
  <item>
    <creator_id>100</creator_id>
    <hours_worked>3</hours_worked>
    <user_comments>Done</user_comments>
    <for_date>2008-10-02</for_date>
    <task_id>89019</task_id>
    <time_code_id>13</time_code_id>
    <rate_type_id>1</rate_type_id>
    <is_billable>>true</is_billable>
    <is_costable>>true</is_costable>
  </item>
</data>
```

In case of JSON, you should submit an array of objects with the same properties.

Here are the details of the fields:

- `creator_id` (Required). Integer. The user on behalf of which the time entry is being submitted.
- `for_date` (Required). Date. The date for this time entry.
- `hours_worked` (Required). Float. The number of hours.
- `task_id` (Required). Integer. The task for which this time entry is being filled.
- `time_code_id` (Required). Integer. The time code ID for this time entry.
- `rate_type_id` (Optional). Integer. 1 for standard, 2 for overtime (in case your organization uses overtime rates)
- `user_comments` (Optional). The comments entered by user for this time entry.
- `is_billable` (Optional). Boolean. Whether this time entry is billable to client.
- `is_costable` (Optional). Boolean. Whether this time entry is costable to the company.
- `send-for-approval` (Optional). Whether the entries submitted are to be sent for approval.

Return values

Nothing

2.8. Create-users

Creates users.

Function name

`create-users`

Parameters

- `data` (Required). It is an array of fields. For example, in case the data format is XML, you should post something like this:

```
<data>
  <item>
    <first_name>Joe</first_name>
    <last_name>Cole</last_name>
    <login>joe.cole</login>
    <password>kaboom1</password>
    <email>joe.cole@celoxis.com</email>
    <role_ids>1738,1912</role_ids>
  </item>
  <item>
    <first_name>Mary</first_name>
    <last_name>Jane</last_name>
    <login>mary.jane</login>
    <password>krypton2</password>
    <email>mary.jane@celoxis.com</email>
```

```
    <role_ids>1738</role_ids>
  </item>
</data>
```

In case of JSON, you should submit an array of objects with the same properties.

Here are the details of the fields:

- `first_name` (Required). String. Either the first, last name or both must be present.
- `last_name` (Required).
- `login` (Optional). String. Login name of the user. If login is present, the user will be counted towards your license count.
- `password` (Optional). String. Password of the user.
- `email` (Optional). String. Email of the user. For users with login, email is mandatory.
- `role_ids` (Optional). String. A comma separated string of role ids.

Return values

Nothing

2.9. Query-utilization

Query the planned, available and actual utilization of resources in a given time period.

Businesses have their own way of organizing resource utilization data and this function provides a flexible way to achieve that. Although XML could have been used to organize the information hierarchically thereby reducing the size of the data by avoiding duplication, we have deliberately returned the data as 'rows'. We think that the best way to use this information is to create temporary SQL tables on your client side, insert the data returned into those tables and then use the power of SQL to query to generate suitable output.

Function name

`query-utilization`

Parameters

- `from` (Required). Date from which utilization is required.
- `to` (Required). Date till which utilization is required.
- `project_id` (Optional). Only include data for project with this ID.
- `project_phase_ids` (Optional). Only include data for projects with these phase ids.
- `zoom` (Optional). Groups utilization data by day, week or month (the default). Valid values: 5 for Day, 3 for Week and 2 for Month.
- `user_id` (Optional). Only include data for user with this ID.

Return values

In case of XML data format, you will be returned an XML string with two elements: `count` (the number of rows) and `rows` (this will contain multiple row elements). For example :

```
<result>
  <rows>
```

```
<row>
  <f>
    <v>3360</v> <!-- Planned minutes for this task for this period -->
    <n>planned</n>
  </f>
  <f>
    <v>1200</v> <!-- Actual minutes for this task for this period -->
    <n>actual</n>
  </f>
  <f>
    <v>10560</v> <!-- Available minutes for this resource in this period -->
    <n>availability</n>
  </f>
  <f>
    <v>27619</v>
    <n>resource_id</n> <!-- ID of the resource -->
  </f>
  <f>
    <v>Joe Cool</v>
    <n>resource_name</n> <!-- Name of the resource -->
  </f>
  <f>
    <v>998612</v>
    <n>task_id</n> <!-- Name of the task -->
  </f>
  <f>
    <v>43154</v>
    <n>project_id</n> <!-- ID of the project -->
  </f>
  <f>
    <v>Sample: Setting up office space</v> <!-- Name of the project -->
    <n>project_name</n>
  </f>
  <f>
    <v>In Progress</v> <!-- Project Phase -->
    <n>project_phase_name</n>
  </f>
  <f>
    <v>General</v> <!-- Category name -->
    <n>project_category_name</n>
  </f>
  <f>
    <v>Internal</v> <!-- Root client name -->
    <n>client_root_name</n>
  </f>
  <f>
    <v>2009-06-01T00:00:00</v> <!-- start date of the period -->
    <n>period_start</n>
  </f>
</row>
...
</rows>
```

`</result>`

In case of JSON, it will return an object will have the property `rows` (which will be an array of objects with `n` and `v` properties respectively)

Chapter 3. Table Reference

Table of Contents

3.1. db_application	17
3.2. db_assignment	17
3.3. db_audit	18
3.4. db_calendar_event	18
3.5. db_category	19
3.6. db_category_app	19
3.7. db_category_sla	19
3.8. db_company	20
3.9. db_contact	21
3.10. db_custom_field_def	22
3.11. db_custom_field_rel	23
3.12. db_custom_field_val	23
3.13. db_custom_report	24
3.14. db_document	24
3.15. db_document_ci	25
3.16. db_document_co	25
3.17. db_event_invitation	26
3.18. db_expense	26
3.19. db_expense_code	27
3.20. db_expense_item	27
3.21. db_favorite	28
3.22. db_folder	28
3.23. db_folder_entry	29
3.24. db_forum	29
3.25. db_forum_subscription	29
3.26. db_forum_topics	30
3.27. db_fs_file	30
3.28. db_group	30
3.29. db_group_members	31
3.30. db_message	31
3.31. db_message_attachments	32
3.32. db_note	32
3.33. db_phone_call	33
3.34. db_process	33
3.35. db_process_document	34
3.36. db_process_form	34
3.37. db_process_state_mgr	35
3.38. db_process_type	35
3.39. db_process_type_form	35
3.40. db_process_update	36
3.41. db_process_update_form	36
3.42. db_project	36
3.43. db_project_category	39
3.44. db_project_phase	39
3.45. db_rate	40
3.46. db_rate_type	40

3.47. db_reminder	41
3.48. db_role_membership	41
3.49. db_role_permission	41
3.50. db_sla	42
3.51. db_sla_entry	42
3.52. db_state	42
3.53. db_state_transition	43
3.54. db_state_transition_form	43
3.55. db_task	44
3.56. db_task_dependency	47
3.57. db_task_expenses	48
3.58. db_task_messages	48
3.59. db_task_notes	48
3.60. db_task_phone_calls	48
3.61. db_task_update	49
3.62. db_time_code	49
3.63. db_time_entry	49
3.64. db_url	50
3.65. db_user	51
3.66. db_working_calendar	52
3.67. db_working_time	52

3.1. db_application

Workflow application

Table 3.1. db_application

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
name	String		Short name
deleted_on	Date		Date of deletion

3.2. db_assignment

Describes resource assignment for this task

Table 3.2. db_assignment

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
user_id	Integer	db_user.id	User ID
unit_type_value	Integer		Assignment unit type 1 = % age, 2 = hours
units	Float		Units assigned to the resource
project_id	Integer	db_project.id	Project ID

Column	Type	References	Comments
task_id	Integer	db_task.id	Task ID
work_time	Float		Effort
est_cost	Float		Estimated cost

3.3. db_audit

Audit trail on objects

Table 3.3. db_audit

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
object_id	Integer		Object ID
object_type	Integer		Object type
user_id	Integer	db_user.id	User ID
created	Date		Date
xml_string	String		Differences

3.4. db_calendar_event

Describes an event object in the calendar

Table 3.4. db_calendar_event

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
kind	Integer		Kind of event. Meeting, Vacation, etc.
object_type	Integer		
object_id	Integer		
owner_id	Integer	db_user.id	Owner of the event
summary	String		Summary of the event
detail	String		Detail of the event
sharing	Integer		
is_all_day	Boolean (t or f)		Is this an all day event?
start_date	Date		Start date of the event
end_date	Date		Finish date of the event
repeat_type	String		Type this event repeats e.g. EVERY_WEEK
repeat_until	Date		Date until which this event repeats

3.5. db_category

Workflow category

Table 3.5. db_category

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
kind	Integer		
name	String		Short name
description	String		Description
parent_id	Integer	db_category.id	Parent category ID
root_id	Integer	db_category.id	Root category ID
path	String		Path (IDs separated by /) from root
context_id	Integer		Context object ID e.g. in case of project processes this will be id of the project
context_type	Integer		Context type

3.6. db_category_app

Mapping of applications to categories i.e which application is available in which categories

Table 3.6. db_category_app

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
app_id	Integer	db_application.id	Application ID
category_id	Integer	db_category.id	Category ID

3.7. db_category_sla

Assignment of SLAs to categories

Table 3.7. db_category_sla

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
sla_id	Integer	db_sla.id	SLA ID
category_id	Integer	db_category.id	Category ID
process_type_id	Integer	db_process_type.id	Process type ID

3.8. db_company

Stores organization related information. Every record corresponds to an organization.

Table 3.8. db_company

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
name	String		The name of the company
decimal_format	String		
currency_symbol	String		
homepage	String		
email	String		
street	String		
locality	String		
state	String		
country	String		
zip	String		
phone	String		
server_name	String		
industry	String		
created	Date		
fl	Integer		
document_fo_id	Integer	db_folder.id	ID of the document folder
forum_fo_id	Integer	db_folder.id	ID of the forum folder
admin_id	Integer	db_user.id	ID of the administrator account
admin_gid	Integer	db_group.id	ID of the administrator security role
accountant_gid	Integer	db_group.id	ID of the accountant security role
project_owner_gid	Integer	db_group.id	ID of the project manager dynamic security role
project_team_gid	Integer	db_group.id	ID of the project team member dynamic security role
staff_gid	Integer	db_group.id	ID of the staff security role (db_group)
management_gid	Integer		ID of the management security role (db_group)
company_code	String		The company code. This should be used while logging in to the API

Column	Type	References	Comments
time_zone	String		The default time zone of the organization
te_on_cp	Boolean (t or f)		Indicates whether time can be filled on archived project
te_on_any	Boolean (t or f)		Allow time on any task
te_auto_approve	Integer		Auto approve time
use_ot	Boolean (t or f)		Whether this organization uses the concept of 'Over-time' rates
all_in_alloc	Boolean (t or f)		Show all users during allocation of tasks to resources (true) or just the project team members (false)
process_client_user_visibility	Integer		Does the client see only the PM (1) or all users (0)
first_day_of_week	Integer		The first day of the week in calendar. 1 for Sunday, 2 for Monday, 3 for Tuesday, etc.
min_days_in_first_week	Integer		The minimum number of days in the first week
minutes_in_day	Integer		Minutes in day. This is used for scheduling. Default is 8 hours i.e. 480.

3.9. db_contact

Contact (or addressbook entry)

Table 3.9. db_contact

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
creator_id	Integer	db_user.id	Owner
folder_id	Integer	db_folder.id	Document folder ID
visible_to_company	Boolean (t or f)		Is public?
kind	Integer		
first_name	String		First Name
middle_name	String		
last_name	String		Last Name
category	Integer		
email	String		Email
nickname	String		Nickname
company_name	String		Company Name

Column	Type	References	Comments
title	String		Job Title
home_phone	String		
mobile_phone	String		
work_phone	String		
pager_number	String		
fax	String		
other_phone	String		
primary_location	Integer		
work_street	String		
work_city	String		
work_state	String		
work_zip	String		
work_country	String		
home_street	String		
home_city	String		
home_state	String		
home_zip	String		
home_country	String		
alternate_email_1	String		
alternate_email_2	String		
personal_website	String		
company_website	String		
birthdate	Date		
anniversary	Date		
cf1	String		
cf2	String		
cf3	String		
cf4	String		
comments	String		
created	Date		
deleted	Date		

3.10. db_custom_field_def

Table 3.10. db_custom_field_def

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
label	String		

Column	Type	References	Comments
help_text	String		
display_order	Integer		
field_type	Integer		
display_width	Integer		
object_type	Integer		
options	String		
is_required	Boolean (t or f)		
default_value	String		
max_length	Integer		
max_value	Float		
min_value	Float		
formula_value	String		The formula
validations	String		
is_summable	Boolean (t or f)		
is_groupable	Boolean (t or f)		
is_sortable	Boolean (t or f)		

3.11. db_custom_field_rel

Table 3.11. db_custom_field_rel

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
custom_field_def_id	Integer	db_custom_field_def.id	
project_category_id	Integer	db_project_category.id	
application_id	Integer	db_application.id	Application ID
process_type_id	Integer	db_process_type.id	
state_transition_id	Integer	db_state_transition.id	State transition ID

3.12. db_custom_field_val

Table 3.12. db_custom_field_val

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
field_type	Integer		Indicates type of field. e.g integer, string, date etc.
raw_value	String		

Column	Type	References	Comments
document_id	Integer	db_document.id	Primary key of the document table in case the custom field is of file type
object_id	Integer		Primary key of the object that this entry represents e.g. ID of a document
object_type	Integer		Type of the object that this entry represents. E.g. 9 = folder, 6 = document, 32 = url
custom_field_def_id	Integer	db_custom_field_def.id	

3.13. db_custom_report

Custom report

Table 3.13. db_custom_report

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
name	String		Short name
description	String		Description
template_id	Integer		Report template id
version	Integer		
creator_id	Integer	db_user.id	Owner of the report
params	String		Report parameters

3.14. db_document

Table 3.14. db_document

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
created	Date		Date this document was created
last_checkin	Date		Date the last version of this document was checked in
creator_id	Integer	db_user.id	ID of the owner
name	String		Name of the document
description	String		Description
path	String		Path of this document (folder IDs separated by /)

Column	Type	References	Comments
content_type	String		Mime type of this document
company_id	Integer	db_company.id	
n_co	Integer		Number of open checkouts
head_release	Integer		Current release number of this document
head_revision	Integer		Current revision number of this document
head_file_id	Integer	db_fs_file.id	ID corresponding to the head revision
head_size	Float		Size of the latest revision
sz	Float		Sum of sizes of all its revisions
folder_id	Integer	db_folder.id	ID of this document's folder
project_id	Integer		
task_id	Integer		
process_id	Integer		
message_id	Integer		

3.15. db_document_ci

Checkin of a document

Table 3.15. db_document_ci

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
created	Date		Date of checkin
document_id	Integer	db_document.id	ID of the document for which this checkin was done
co_id	Integer	db_document_co.id	Corresponding checkout ID
release	Integer		Release number of this checkin
revision	Integer		Revision number of this checkin
ci_by	Integer	db_user.id	User who checked this in
comments	String		Comments for this checkin
file_id	Integer	db_fs_file.id	File that was checked in

3.16. db_document_co

Checkout of a document

Table 3.16. db_document_co

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
created	Date		Date of the checkout
document_id	Integer	db_document.id	The document for this checkout was done
ci_id	Integer	db_document_ci.id	Corresponding checkin ID, if null means there was no checkin
co_by	Integer	db_user.id	User who checked-out
release	Integer		Release number of this checkout
revision	Integer		Revision number of this checkout

3.17. db_event_invitation

Event invitation

Table 3.17. db_event_invitation

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
owner_id	Integer	db_user.id	Owner ID
calendar_event_id	Integer	db_calendar_event.id	Calendar event ID
status	Integer		0 = unknown, 1 = invitee accepted, 2 = invitee declined
invitation_type	Integer		0 = required, 1 = Optional, 2 = Copy
invitee_id	Integer	db_user.id	Invitee user id in case the invitee is a user of the system
invitee_email	String		Invitee Email
comments	String		Invitee entered comments (e.g. reason for refusal)

3.18. db_expense

Describes an expense

Table 3.18. db_expense

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		

Column	Type	References	Comments
created	Date		Date of this expense
client_id	Integer	db_user.id	Client ID for this expense
creator_id	Integer	db_user.id	Creator ID
summary	String		Short summary
is_billable	Boolean (t or f)		Whether this expense is billable
state	Integer		0 - saved but not submitted for approval, 1 - sent for approval, 2 - rejected, 3 - approved
company_id	Integer	db_company.id	
approver_id	Integer	db_user.id	Approved by ID
approved_on	Date		Date of approval
approval_comments	String		Approver comments
rejected_by_id	Integer	db_user.id	Rejected by
rejected_on	Date		Date of rejection
rejection_comments	String		Reason for rejection
is_temp	Boolean (t or f)		Whether this expense is 'temporary'
amount	Float		Amount

3.19. db_expense_code

Describes expense codes

Table 3.19. db_expense_code

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
deleted	Date		Date this expense code was deleted
company_id	Integer	db_company.id	
name	String		Short name
description	String		Description
ref_no	String		Accounting code for this expense code
markup	Float		
is_taxable	Boolean (t or f)		

3.20. db_expense_item

Describes line item in an expense

Table 3.20. db_expense_item

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
company_id	Integer	db_company.id	
expense_id	Integer	db_expense.id	The expense object
type_id	Integer	db_expense_code.id	The expense code
end_date	Date		The date of this expense item
description	String		A short description of the expense
amount	Float		Amount of this expense
is_temp	Boolean (t or f)		If this is a temporary line item

3.21. db_favorite

Favorites

Table 3.21. db_favorite

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
user_id	Integer	db_user.id	User ID
object_id	Integer		Favorite object ID
object_type	Integer		Favorite object type

3.22. db_folder

Details of the folder object. Folder stores other folders, documents, urls and forums

Table 3.22. db_folder

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
name	String		Name of this folder
description	String		Description of this folder
path	String		Path of this folder from its root. IDs are separated by /.
parent_id	Integer	db_folder.id	ID of this folder's parent.
root_id	Integer	db_folder.id	Root ID of this folder.
creator_id	Integer	db_user.id	ID of the creator of this folder.

Column	Type	References	Comments
sz	Float		Size of this folder

3.23. db_folder_entry

Describes a folder entry. A folder entry represents an object that is contained in a folder

Table 3.23. db_folder_entry

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
folder_id	Integer	db_folder.id	Folder this entry is contained in
object_id	Integer		Primary key of the object that this entry represents e.g. ID of a document
object_type	Integer		Type of the object that this entry represents. E.g. 9 = folder, 6 = document, 32 = url

3.24. db_forum

Describes a forum object. A forum contains multiple 'topics'

Table 3.24. db_forum

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
name	String		Name of this forum
description	String		Description of this forum
company_id	Integer	db_company.id	
creator_id	Integer	db_user.id	Creator ID
path	String		Path of this forum in the folder tree
last_posted	Date		Date of the last post in any of the topics

3.25. db_forum_subscription

User subscription to a forum. When a user subscribes to a forum he/she will receive emails of any posts in this forum

Table 3.25. db_forum_subscription

Column	Type	References	Comments
id	Integer		Primary key

Column	Type	References	Comments
vdb_id	Integer		
created	Date		
forum_id	Integer	db_forum.id	Forum subscribed to
user_id	Integer	db_user.id	ID of the subscriber

3.26. db_forum_topics

Topics in a forum

Table 3.26. db_forum_topics

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
created	Date		Date this topic was created
forum_id	Integer	db_forum.id	Forum this topic belongs to
topic_id	Integer	db_message.id	Message ID of this topic
state	Integer		Whether this topic is OPEN or CLOSED
last_posted	Date		Date of the last post in this topic
replies	Integer		Number of replies in this topic

3.27. db_fs_file

Details of a file that is stored on the disk

Table 3.27. db_fs_file

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
content_type	String		Mime type of this file
local_path	String		Path on the disk
name	String		Name of this file
sz	Float		Size of this file

3.28. db_group

Table 3.28. db_group

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		

Column	Type	References	Comments
deleted	Date		When this group was deleted
company_id	Integer	db_company.id	
name	String		The name of the group
comments	String		
is_dynamic	Boolean (t or f)		Indicates whether membership to this group is context sensitive i.e. whether this security role is a dynamic role
kind	Integer		Allows this group to be used in different contexts like security role, department, user group, etc.
can_user_set	Boolean (t or f)		Means end users can assign members
context_id	Integer		
context_type	Integer		

3.29. db_group_members

Maps group versus user

Table 3.29. db_group_members

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
group_id	Integer	db_group.id	ID of the group
user_id	Integer	db_user.id	ID of the user

3.30. db_message

Details of emails sent and received

Table 3.30. db_message

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
summary	String		Subject of the email
detail	String		Body of the email
creator_id	Integer	db_user.id	Creator ID
original_message	Integer	db_document.id	If this mail was imported or read from an email server,

Column	Type	References	Comments
			then the ID of the document that stores this file
is_parsed_ok	Boolean (t or f)		If this mail was read from an email server, whether it was parsed without any problems
sender	String		Sender email address
to_recipients	String		To recipient email addresses
cc_recipients	String		Cc recipient email addresses
bcc_recipients	String		Bcc recipient email addresses
reply_to	String		Reply to email address
is_html	Boolean (t or f)		Whether this is HTML email
created	Date		Date this mail was created
parent_id	Integer	db_message.id	ID of this email message this was a reply to
root_id	Integer	db_message.id	Root ID of the email thread
num_attachments	Integer		Number of attachments

3.31. db_message_attachments

Email attachments

Table 3.31. db_message_attachments

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
message_id	Integer	db_message.id	ID of the email message
attachment_id	Integer	db_document.id	ID of the document corresponding to an attachment

3.32. db_note

Notes associated with tasks and contacts

Table 3.32. db_note

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
created	Date		Date this note was created

Column	Type	References	Comments
detail	String		The text of this note
creator_id	Integer	db_user.id	Creator of this note

3.33. db_phone_call

Phone call object

Table 3.33. db_phone_call

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
created	Date		Date of creation
creator_id	Integer	db_user.id	Creator ID
employee_id	Integer	db_user.id	User ID
kind	Integer		
container	Integer		
summary	String		Summary
comments	String		Detail
party_number	String		
party_name	String		
party_id	Integer	db_contact.id	
call_time	Integer		
billing_status	Integer		
answer_status	Integer		
status	Integer		
is_urgent	Boolean (t or f)		

3.34. db_process

Describes a process

Table 3.34. db_process

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
created	Date		Date created
summary	String		Short summary
detail	String		Detailed description
due_date	Date		Due date
is_delayed	Boolean (t or f)		Is this process delayed?

Column	Type	References	Comments
priority	Integer		Priority : 1 = highest, 5 = lowest
type_id	Integer	db_process_type.id	Process type ID
category_id	Integer	db_category.id	Category ID
last_state_changed	Date		Last state change on
client_id	Integer	db_user.id	Requestor ID
client_name	String		Requestor name (in case requestor is not a user of the system)
client_email	String		Requestor email (in case requestor is not a user of the system)
creator_id	Integer	db_user.id	Creator ID
assigned_to_id	Integer	db_user.id	Current owner
state_id	Integer	db_state.id	State this process is in
container_id	Integer		
container_type	Integer		
completed	Date		Completed on
is_client_visible	Boolean (t or f)		Is visible to requestor?

3.35. db_process_document

Documents associated with this process

Table 3.35. db_process_document

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
process_id	Integer	db_process.id	Process ID
process_update_id	Integer	db_process_update.id	Process update ID
document_id	Integer	db_document.id	Document ID

3.36. db_process_form

Custom fields associated with a process

Table 3.36. db_process_form

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
process_id	Integer	db_process.id	Process ID
form_name	String		Custom form name

Column	Type	References	Comments
form	String		Custom fields

3.37. db_process_state_mgr

Process state managers. A user is a manager of a particular state in a particular category.

Table 3.37. db_process_state_mgr

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
user_id	Integer	db_user.id	Manager ID
category_id	Integer	db_category.id	Category ID
state_id	Integer	db_state.id	State ID

3.38. db_process_type

Describes a process type

Table 3.38. db_process_type

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
name	String		Name
description	String		Description
deleted_on	Date		Date this was deleted
app_id	Integer	db_application.id	Application ID (every process type belongs to an application)
client_reqd	Boolean (t or f)		Whether requestor makes sense for this process type
initial_assignee_id	Integer		
initial_assignee_type	Integer		1 = None , 2 = Specific user, 3 = State manager

3.39. db_process_type_form

Forms associated with a process type

Table 3.39. db_process_type_form

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
process_type_id	Integer	db_process_type.id	Process type ID

Column	Type	References	Comments
form_id	Integer		Custom form ID

3.40. db_process_update

Updates on a process includes assignee changes, comments or state transitions

Table 3.40. db_process_update

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
created	Date		Date of update
process_id	Integer	db_process.id	Process ID
comments	String		Comments
creator_id	Integer	db_user.id	Creator ID
assignee_id	Integer		Assignee ID
state_id	Integer	db_state.id	From state ID
transition_id	Integer		State Transition ID, if none then empty
next_state_id	Integer	db_state.id	To state ID
name	String		Summary
email	String		Email

3.41. db_process_update_form

Custom fields associated with a process update

Table 3.41. db_process_update_form

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
process_update_id	Integer	db_process_update.id	Process update ID
form_name	String		Custom form name
form	String		Custom fields

3.42. db_project

Describes a project

Table 3.42. db_project

Column	Type	References	Comments
id	Integer		Primary key
code	String		A unique project code

Table Reference

Column	Type	References	Comments
vdb_id	Integer		
created	Date		Date this project was created
team_id	Integer	db_group.id	ID of the group that corresponds to this team
project_category_id	Integer	db_project_category.id	ID of its project category
client_id	Integer	db_user.id	Client ID
manager_id	Integer	db_user.id	Project Manager ID
priority	Integer		Priority. 1 is for highest and 5 for lowest.
summary	String		Short name of this project
detail	String		Detail
custom_form	String		Custom field values
charging_model	Integer		
ceiling_price	Float		
start_price	Float		
fixed_price	Float		
fixed_cost	Float		Accumulated fixed cost for this project
user_cost	Float		Accumulated user cost for this project
budget	Float		Budget
company_id	Integer	db_company.id	
folder_id	Integer	db_folder.id	ID of its document folder
forum_folder_id	Integer	db_folder.id	ID of its forum folder
archived	Date		Date when the project is in an 'end' phase
archived_by_id	Integer		ID of the user who archived this project
phase_id	Integer	db_project_phase.id	Phase of this project: 1 - In Progress, 2 - On Hold, 3 - Completed, 4 - Planning
plan_start	Date		Planned start date of this project
plan_finish	Date		Planned finish date of this project. This will be the max of plan_finish of its tasks.
deadline	Date		The deadline of this project
percent_complete	Integer		% complete
accumulated_cost	Float		Total accumulated cost
b_expense_total	Float		Total billable expense

Table Reference

Column	Type	References	Comments
nb_expense_total	Float		Total non-billable expense
b_timesheet_total	Float		Total billable time sheet billing
nb_timesheet_total	Float		Total non-billable time sheet billing
estimated_work	Float		Total estimated work in minutes
estimated_labor_cost	Float		Total estimated labour cost
estimated_non_labor_cost	Float		Total estimated non-labour cost
estimated_overheads	Float		Total estimated fixed cost
expected_on	Date		Projected finish date
b_timesheet_hours	Float		Billable time sheet hours
nb_timesheet_hours	Float		Non-billable time sheet hours
c_timesheet_hours	Float		Costable time sheet hours
inherit_pc	Boolean (t or f)		Whether % complete is to be automatically calculated
note_count	Integer		Total number of notes associated with its tasks
message_count	Integer		Total number of email messages associated with its tasks
phone_call_count	Integer		Total number of phone calls associated with its tasks
status_update_count	Integer		Total number of task status updates associated with its tasks
expense_count	Integer		Total number of expenses associated with its tasks
time_entry_count	Integer		Total number of time entries associated with its tasks
COMPUTED COLUMNS:			
spi	Float		The schedule performance index using earned value method
cpi	Float		The cost performance index using earned value method
bcws	Float		The budgeted cost of work scheduled using earned value method

Column	Type	References	Comments
bcwp	Float		The budgeted cost of work performed using earned value method
acwp	Float		The actual cost of work performed using earned value method
schedule_status	String		The schedule status. Will be one of: ON_TIME, AT_RISK, OVERDUE, BLOCKED, FUTURE and COMPLETED
budget_status	String		The budget status. Will be one of: ON_BUDGET, AT_RISK, OVER_BUDGET
client_root_name	String		The name of the client root in this project's client hierarchy
client_name	String		The name of this project's client
manager_name	String		The name of this project's manager

3.43. db_project_category

Describes the project category

Table 3.43. db_project_category

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
deleted	Date		Date this category was deleted
company_id	Integer	db_company.id	
name	String		Name of the project category
description	String		Description of the project category
form_id	Integer		

3.44. db_project_phase

Describes project phases

Table 3.44. db_project_phase

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
name	String		A short name
description	String		Description
company_id	Integer		
end_state	Boolean (t or f)		Whether this phase can be considered an 'end' state in the project lifecycle
position	Integer		Its position in drop-down lists

3.45. db_rate**Table 3.45. db_rate**

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
object_id	Integer		Object ID associated with the rate. ID of time code OR User
object_type	Integer		22 = time code, 24 = user
context_id	Integer		The context in which this rate is defined. For example, user cost rates can be defined either at a company level OR at a project level. ID of project or company
context_type	Integer		17 = project, 3 = company
amount	Float		Rate amount
period	Integer		
effective	Date		Effective from
kind	Integer		Which rate: 1 = cost, 2 = client billing
sub_kind	Integer		The rate type 1 = regular, 2 = overtime

3.46. db_rate_type**Table 3.46. db_rate_type**

Column	Type	References	Comments
id	Integer		Primary key

Column	Type	References	Comments
vdb_id	Integer		
company_id	Integer	db_company.id	
name	String		
description	String		

3.47. db_reminder

Describes a reminder of a calendar event

Table 3.47. db_reminder

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
owner_id	Integer	db_user.id	Owner ID
calendar_event_id	Integer	db_calendar_event.id	ID of the calendar event object this reminder corresponds to
last_reminded_at	Date		Date last reminded at (in case of recurring events)
due_date	Date		Date of the reminder

3.48. db_role_membership

Workflow application security role membership. A user is a member of a role in a particular category.

Table 3.48. db_role_membership

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
user_id	Integer	db_user.id	User ID
role_id	Integer	db_group.id	Role ID
category_id	Integer	db_category.id	Category ID

3.49. db_role_permission

Permission associated with the role

Table 3.49. db_role_permission

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
permission_id	Integer		Permission ID
role_id	Integer	db_group.id	Role ID

3.50. db_sla

Time out policy

Table 3.50. db_sla

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
name	String		Short name
process_type_id	Integer	db_process_type.id	Process type ID
context_id	Integer		Context of this SLA e.g. category ID
context_type	Integer		Context type of this SLA e.g. category

3.51. db_sla_entry

Timeout rules

Table 3.51. db_sla_entry

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
sla_id	Integer	db_sla.id	SLA ID
state_id	Integer	db_state.id	State ID for timeout
ttl_minutes	Integer		Timeout in minutes
priority	Integer		Priority: 1 = highest, 5 = lowest
transition_id	Integer	db_state_transition.id	State transition to follow on timeout
user_specific_ttl	Boolean (t or f)		If timeout is to be entered by user doing the state transition
mark_delayed	Boolean (t or f)		Whether to mark the process as delayed when timeout happens

3.52. db_state

Workflow process state

Table 3.52. db_state

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		

Column	Type	References	Comments
name	String		Short name
description	String		Description
is_start_state	Boolean (t or f)		Is this a 'start' state?
is_end_state	Boolean (t or f)		Is this an 'end' state? An end state means that the process is assumed to be completed.
context_id	Integer		
context_type	Integer		
deleted_on	Date		Date this state was deleted
position	Integer		Order in its siblings

3.53. db_state_transition

Workflow state transition

Table 3.53. db_state_transition

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
name	String		Short name
state_id	Integer	db_state.id	State ID
next_state_type	Integer		Next state type: 1 = same, 2 = specific, 3 = previous
next_state_id	Integer	db_state.id	Next state ID (in case next state type is specific)
next_assignee_type	Integer		Next assignee type: 0 = unassigned, 1 = same, 2 = state manager, 3 = prompt user in role, 4 = last assigned for this state, 5 = reporter
next_assignee_id	Integer		Next assignee ID (in case next assignee type is 3 i.e. prompt user in role)

3.54. db_state_transition_form

Forms associated with state transitions

Table 3.54. db_state_transition_form

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
state_transition_id	Integer		State transition ID

Column	Type	References	Comments
form_id	Integer		Custom form ID

3.55. db_task

Describes a task

Table 3.55. db_task

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
project_id	Integer	db_project.id	ID this task's project
parent_id	Integer	db_task.id	Parent ID
company_id	Integer	db_company.id	
folder_id	Integer		Document folder ID
priority	Integer		Priority. 1 for highest 5 for lowest
summary	String		Short name
detail	String		Detailed description
duration_value	Float		Duration value (e.g. if duration is 2.5days this field will be 2.5)
actual_start	Date		Actual start
plan_start	Date		Planned start
plan_finish	Date		Planned finish
is_critical	Boolean (t or f)		True if task is on critical path
is_milestone	Boolean (t or f)		True if milestone
early_start	Date		Early start
early_finish	Date		Early finish
late_start	Date		Late start
late_finish	Date		Late finish
completed_on	Date		Actual finish
percent_complete	Integer		% complete
b_expense_total	Float		Total billable expense
nb_expense_total	Float		Total non billable expense
b_timesheet_total	Float		Total billable time entries billing
nb_timesheet_total	Float		Total non-billable time entries billing
ru_b_expense_total	Float		Total rolled-up billable expense

Table Reference

Column	Type	References	Comments
ru_nb_expense_total	Float		Total rolled-up non billable expense
ru_b_timesheet_total	Float		Total rolled-up billable time entries billing
ru_nb_timesheet_total	Float		Total rolled-up non-billable time entries billing
budget	Float		Budget
est_cost	Float		Estimated cost to complete this task
ru_est_cost	Float		Rolled-up estimated cost to complete this task and its descendants
est_non_labor_cost	Float		Estimated non labour cost
est_labor_cost	Float		Estimated labour cost
ru_est_labor_cost	Float		Rolled-up estimated labour cost
fixed_cost	Float		Fixed cost
user_cost	Float		Actual labour cost till now
ru_user_cost	Float		Rolled-up actual labour cost till now
can_fill_time	Boolean (t or f)		Can fill time on this task?
is_dep_solved	Boolean (t or f)		Have all predecessors finished?
last_status_update	Date		Date of the last status update
is_assigned	Boolean (t or f)		Is this task assigned to anyone
nb_timesheet_hours	Float		Total non-billable time (hours)
b_timesheet_hours	Float		Total billable time (hours)
ru_nb_timesheet_hours	Float		Total rolled-up non-billable time (hours)
ru_b_timesheet_hours	Float		Total rolled-up billable time (hours)
c_timesheet_hours	Float		Total costable time (hours)
ru_c_timesheet_hours	Float		Total rolled-up costable time (hours)
constraint_date	Date		Constraint date
constraint_type_value	Integer		Constraint type: 1 = ASAP, 2 = ALAP, 3 = MSO, 4 = MFO, 5 = SNET, 6 = SNLT, 7 = FNET, 8 = FNLT.

Table Reference

Column	Type	References	Comments
duration_type	Integer		Duration units: 1 = Day, 2 = Hour, 3 = Week, 4 = Month, 5 = Year, 6 = Minute
is_summary	Boolean (t or f)		Is this a summary task i.e. does this task have any sub tasks
assigned_work	Float		Assigned effort in minutes
ru_assigned_work	Float		Rolled-up assigned effort in minutes
is_duration_estimated	Boolean (t or f)		Is the duration an approximation
projected_start_date	Date		Projected start date
projected_deadline	Date		Projected deadline
work_time	Float		Effort in minutes
cost	Float		Accumulated cost for this task
ru_cost	Float		Accumulated cost for this task's descendants
project_phase_id	Integer		Phase of this task's project
archived	Date		Date when project was archived
child_number	Integer		Number in siblings
local_key	Integer		
billing_type	Integer		
billing_amount	Float		
tree_level	Integer		
note_count	Integer		Total number of notes associated with this task
message_count	Integer		Total number of email messages associated with this task
phone_call_count	Integer		Total number of phone calls associated with this task
status_update_count	Integer		Total number of task status updates associated with this task
expense_count	Integer		Total number of expenses associated with this task
time_entry_count	Integer		Total number of time entries associated with this task
is_expanded	Boolean (t or f)		Is this task node expanded while displaying project tasks

Column	Type	References	Comments
projected_start	Date		Projected start date
projected_finish	Date		Projected finish date
type_id	Integer		Task type ID
state_id	Integer		Task type state ID
COMPUTED COLUMNS:			
spi	Float		The schedule performance index using earned value method
cpi	Float		The cost performance index using earned value method
bcws	Float		The budgeted cost of work scheduled using earned value method
bcwp	Float		The budgeted cost of work performed using earned value method
acwp	Float		The actual cost of work performed using earned value method
schedule_status	String		The schedule status. Will be one of: ON_TIME, AT_RISK, OVERDUE, BLOCKED, FUTURE and COMPLETED
budget_status	String		The budget status. Will be one of: ON_BUDGET, AT_RISK, OVER_BUDGET
project_summary	String		The name of the project

3.56. db_task_dependency

Describes task dependencies

Table 3.56. db_task_dependency

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
project_id	Integer	db_project.id	The project
task1_id	Integer	db_task.id	The predecessor task
task2_id	Integer	db_task.id	The successor task
lag_value	Float		Lag duration value
lag_type	Integer		Lag duration type (see task duration for valid values)

3.57. db_task_expenses

Task expenses

Table 3.57. db_task_expenses

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
task_id	Integer	db_task.id	Task ID
expense_id	Integer	db_expense.id	Expense ID

3.58. db_task_messages

Task messages (aka emails)

Table 3.58. db_task_messages

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
task_id	Integer	db_task.id	Task ID
message_id	Integer	db_message.id	Message ID

3.59. db_task_notes

Task notes

Table 3.59. db_task_notes

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
task_id	Integer	db_task.id	Task ID
note_id	Integer	db_note.id	Note ID

3.60. db_task_phone_calls

Task phone calls

Table 3.60. db_task_phone_calls

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
task_id	Integer	db_task.id	Task ID

Column	Type	References	Comments
phone_call_id	Integer	db_phone_call.id	Phone Call ID

3.61. db_task_update

Status updates on tasks

Table 3.61. db_task_update

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
created	Date		Date of update
task_id	Integer	db_task.id	Task associated with this update
state_id	Integer		Task type state ID associated with this update
creator_id	Integer	db_user.id	Creator of this update
percent_complete	Integer		% complete
comments	String		Comments
actual_start	Date		Actual start
actual_finish	Date		Actual finish

3.62. db_time_code

Table 3.62. db_time_code

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
created	Date		
deleted	Date		Date this time code was deleted
name	String		Short name
description	String		Description
company_id	Integer	db_company.id	
ref_no	String		Accounting code
is_taxable	Boolean (t or f)		
is_costable	Boolean (t or f)		if true, time entries filed against this time code will be marked as costable by default

3.63. db_time_entry

Describes an individual time entry in the time sheet

Table 3.63. db_time_entry

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
for_date	Date		Date for this time entry was filled
hours_worked	Float		Hours
creator_id	Integer	db_user.id	Submitted by ID
task_id	Integer	db_task.id	Task ID
project_id	Integer	db_project.id	Project ID
time_code_id	Integer	db_time_code.id	Time code ID
rate_type_id	Integer	db_rate_type.id	Rate type ID, 1 = regular, 2 = overtime
user_comments	String		Comments
approver_comments	String		Approver comments
approved_on	Date		Approved on
approver_id	Integer	db_user.id	Approved by ID
rejected_by_id	Integer	db_user.id	Rejected by ID
rejected_on	Date		Rejected on
rejection_comments	String		Reason for rejection
state	Integer		0 - saved but not submitted for approval, 1 - sent for approval, 2 - rejected, 3 - approved
is_billable	Boolean (t or f)		Is billable?
is_costable	Boolean (t or f)		Is costable?
user_cost	Float		Labour cost corresponding to this time entry
cached_billing_rate	Float		the billing rate to be used for this time entry; billing rate is calculated from the db_rate which keeps a historic date for rates

3.64. db_url

Details of web links

Table 3.64. db_url

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		

Column	Type	References	Comments
creator_id	Integer	db_user.id	Owner of this URL
name	String		A friendly name for this URL
url	String		The actual URL
path	String		The path where this URL belongs in the folder tree
keywords	String		Keywords associated with this URL
company_id	Integer	db_company.id	
folder_id	Integer	db_folder.id	The folder this URL belongs to.

3.65. db_user

Stores resources, clients and users

Table 3.65. db_user

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
first_name	String		First name of the user
middle_name	String		
last_name	String		Last name
login_name	String		Login name
pwd	String		Encrypted password
phone	String		
pager_email	String		
email	String		Email address
company_id	Integer	db_company.id	
created	Date		
archived	Date		Date on which this user was archived
is_client	Boolean (t or f)		If true, then this user is a client
folder_id	Integer	db_folder.id	ID of the personal document folder of this user
contact_id	Integer	db_contact.id	ID of the contact information for this user
keywords	String		Keywords related to this user. Skills, affiliations, etc. can be entered in this column

Column	Type	References	Comments
cf1	String		Custom field 1
cf2	String		Custom field 2
cf3	String		Custom field 3
in_out_status	String		
in_out_updated_on	Date		
last_access	Date		The last time the user accessed the system
is_in	Boolean (t or f)		If this user is in or out
parent_id	Integer		The parent of this record. In case of client this is the ID of the parent.
path	String		Path separated by '/' from root user object.
locale_id	String		
time_zone_id	String		Time zone ID of this user
calendar_id	Integer		The working calendar of this user

3.66. db_working_calendar

Working calendar. There is always a default calendar for every company.

Table 3.66. db_working_calendar

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
name	String		Name of this calendar.
is_default	Boolean (t or f)		Whether this is the default calendar.

3.67. db_working_time

Describes the working time for a day. It supports 3 intervals for now.

Table 3.67. db_working_time

Column	Type	References	Comments
id	Integer		Primary key
vdb_id	Integer		
calendar_id	Integer	db_working_calendar.id	ID of the working calendar this record belongs to
s1_from	Integer		Start time in minutes of the 1st interval.

Table Reference

Column	Type	References	Comments
s1_to	Integer		Finish time in minutes of the 1st interval.
s2_from	Integer		Start time in minutes of the 2nd interval.
s2_to	Integer		Finish time in minutes of the 2nd interval.
s3_from	Integer		Start time in minutes of the 3rd interval.
s3_to	Integer		Finish time in minutes of the 3rd interval.
dow	Integer		Day of week this working time refers to. 1 for Sunday, 2 for Monday, etc.