



ADVANCED REPORTING

Copyright © 2020 Workfront, Inc.

All rights reserved.

No part of this publication may be reproduced, stored in any retrieval system, or transmitted, in any form, or by any means, whether electronic, mechanical, photocopying, sound recording, or otherwise, without the prior written consent of Workfront, Inc., except as permitted by law.

rev. December 2020

TABLE OF CONTENTS



| | | |
|---|-----|-----------|
| CHAPTER 1: REPORTING REVIEW | | |
| Reporting Terms and Definitions | 4 | |
| Creating Reports | 5 | |
| Create a Report | 7 | |
| OR Qualifier | 8 | |
| | 10 | |
| CHAPTER 2: ADVANCED FILTERS | | |
| API Explorer | 13 | |
| Data Types | 14 | |
| Variation In Verbiage | 20 | |
| Text Mode Structure and Syntax | 21 | |
| Using Text from the Course Book | 22 | |
| Using Text Mode | 23 | |
| AND Statements | 24 | |
| Using AND Statements | 26 | |
| Qualifiers | 27 | |
| Text Attribute and Field Qualifiers | 29 | |
| Using Qualifiers | 30 | |
| Date-Based Wildcards | 32 | |
| User-Based Wildcard Filter Variables | 36 | |
| Date-Based Wildcards | 37 | |
| User-Based Wildcards | 39 | |
| Referencing Related Objects | 40 | |
| Custom Prompts | 41 | |
| | 44 | |
| CHAPTER 3: ADVANCED VIEWS | | 46 |
| Anatomy of a Text View | 47 | |
| Adjusting Column Widths | 48 | |
| Essential Components of a Column View | 49 | |
| Using Custom Reporting Elements | 50 | |
| Shared Columns | 51 | |
| Cross-Object References | 54 | |
| Cross-Object References | 55 | |
| Naming Columns | 58 | |
| Using Custom Data in a View | 59 | |
| Calculated Custom Data | 60 | |
| Cross-Object References in Calculated Custom Data | 65 | |
| Calculated Custom Data on a Form | 66 | |
| Calculated Columns | 70 | |
| Using Custom Expressions in a View | 71 | |
| Calculated Custom Data vs. Calculated Columns | 74 | |
| Calculated Expression Operators | 75 | |
| Calculated Custom Data | 80 | |
| Calculated Aggregates | 84 | |
| CHAPTER 4: ADVANCED GROUPINGS | | 88 |
| Referencing Related Objects in Groupings | 89 | |
| Calculated Groupings | 93 | |
| Group By Multiple Parent Tasks | 96 | |
| APPENDIX | | 98 |
| Exercise: Calculated Custom Data Expressions | 99 | |
| Exercise: Calculated Column Expressions | 100 | |
| Custom Forms | 101 | |
| Customizing Icons | 103 | |



CHAPTER 1

REPORTING REVIEW

OBJECTIVES

After completing this chapter, you will be able to:

- Understand Workfront reporting terms and definitions
- Create reports
- Utilize 'OR' statements

Reporting Terms and Definitions



Builder Interface The Builder Interface is the series of drop-down fields presented on the View, Filter, and Grouping screens. It provides an intuitive mapping of the element relationships to assist in identifying the columns in a view, the criteria of a filter, and the common attributes of a grouping.

Camel Case Camel Case refers to a specific way to write programming elements to string multi-word attributes together. The rules are that the first letter of the first word is lower case, there is no space between the words, and the first letter of any subsequent word is uppercase. For example, Home Group would be rendered homeGroup, Resource Pool would be resourcePool, and Actual Start Date would be actualStartDate.

Fields Fields is the Workfront term for a custom data field. Fields are created and then added to a custom data form to supplement the core fields provided in the system.

Field Name The Field Name provides a list of available attributes to identify the value that is displayed in a view, used in the condition of a filter, or as the common element of a grouping. The options in the Field Name field are dependent on the Field Source selection. For example, if you create a task view and wish to display the Planned Start Date, you would select the Planned Start Date Field Name – it is an attribute that describes the task.

Field Source The Field Source provides a list of available objects that can be referenced on a view, filter, or grouping. The options in the Field Source are dependent on the object type of the UI element being created. The Field Source allows you to reference attributes from objects other than the object type of the UI element. For example, if you create a task view and wish to display the project name in a column, you would select the Project Field Source and the Name Field Name.

Filter The Filter determines the results that display in a report.

Form Form is the Workfront term for Custom Form. Fields and sections are added to forms, which are then attached to an object to extend the database beyond the core fields provided in the system.



Reporting Terms and Definitions (continued)

GroupingThe Grouping identifies how a list of results are organized. The Grouping creates blue horizontal bars throughout the report to group the results by common attributes defined by the Grouping. Groupings are used in Matrix Reports to determine the axes of charts and tables.

Object or Object TypeAn Object is a Workfront application element (i.e., Project, Task, Group, Company, Filter). The Object Type is used when creating a new Report, View, Filter, or Grouping to identify which Object is the focus of the report.

QualifierThe Qualifier field appears on the Filter screen and the Advanced Settings screen for a view. Its purpose is to determine how the Field Name of the filter or condition compares to another field or value. For example, the qualifier determines if the task's Planned Start Date is equal to, greater than, or less than today's date.

ReportA Report is the combination of a view, a filter, and (sometimes) a grouping. The purpose of a report is to display data consistently across users, to distribute information, and to eliminate the need to run the same search or query on a regular basis.

Text Mode InterfaceText Mode Interface (referred to as Text Mode in this course) provides the ability to modify/manipulate custom views, filters, and groupings created through the Builder Interface. It is suggested that report elements are initially created through the Builder Interface and then converted into the Text Mode after they have been saved to simplify advanced view, filter, and grouping creation.

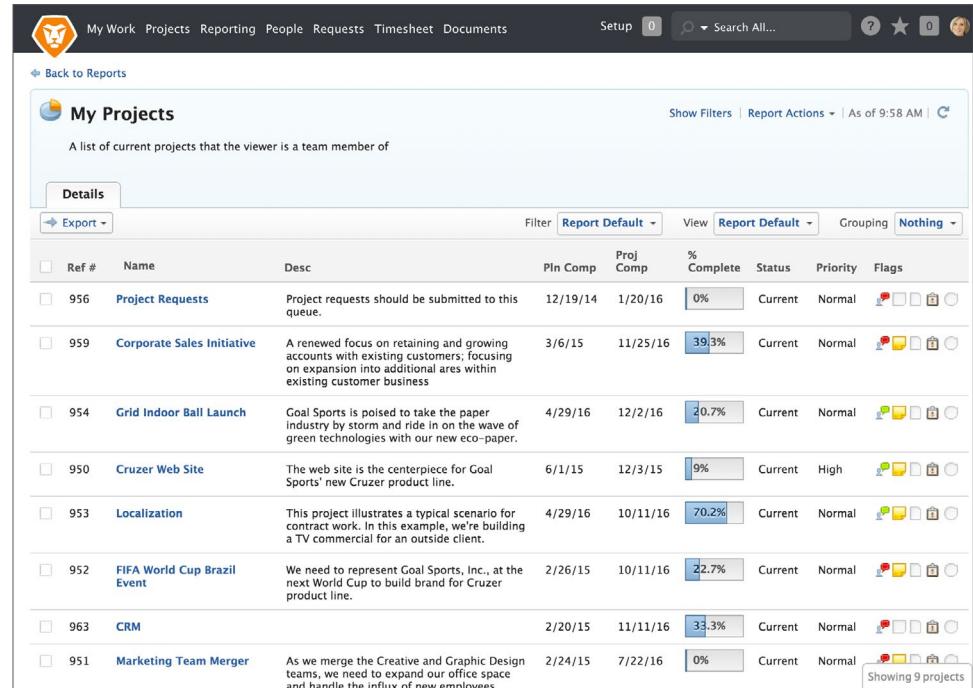
User Interface (UI)The term UI refers to the components or building blocks of a report. Namely, the View, Filter, and Grouping.

ViewThe View identifies the column headers that display across the top of a list report.

Creating Reports

The Workfront Report Creation course teaches that a filter controls the results listed in a report. The view dictates the columns shown across the top of the report. Groupings are used to organize the results based on a common attribute(s). These are the basic elements of a report.

The first step to creating a report is to identify the reporting elements needed, especially any custom view, filter, and/or grouping.



The screenshot shows the Workfront interface with the title 'My Projects'. It displays a list of nine projects with the following details:

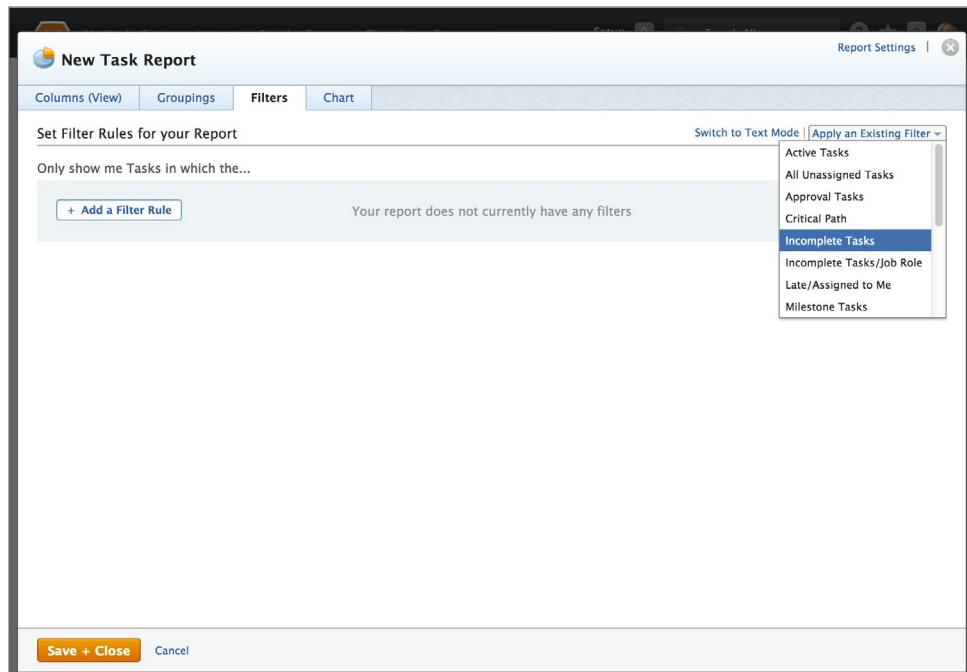
| Ref # | Name | Desc | Pln Comp | Proj Comp | % Complete | Status | Priority | Flags |
|-------|---|--|----------|-----------|------------|---------|----------|-------|
| 956 | Project Requests | Project requests should be submitted to this queue. | 12/19/14 | 1/20/16 | 0% | Current | Normal | |
| 959 | Corporate Sales Initiative | A renewed focus on retaining and growing accounts with existing customers; focusing on expansion into additional areas within existing customer business | 3/6/15 | 11/25/16 | 39.3% | Current | Normal | |
| 954 | Grid Indoor Ball Launch | Goal Sports is poised to take the paper industry by storm and ride in on the wave of green technologies with our new eco-paper. | 4/29/16 | 12/2/16 | 20.7% | Current | Normal | |
| 950 | Cruzer Web Site | The web site is the centerpiece for Goal Sports' new Cruzer product line. | 6/1/15 | 12/3/15 | 9% | Current | High | |
| 953 | Localization | This project illustrates a typical scenario for contract work. In this example, we're building a TV commercial for an outside client. | 4/29/16 | 10/11/16 | 70.2% | Current | Normal | |
| 952 | FIFA World Cup Brazil Event | We need to represent Goal Sports, Inc., at the next World Cup to build brand for Cruzer product line. | 2/26/15 | 10/11/16 | 22.7% | Current | Normal | |
| 963 | CRM | | 2/20/15 | 11/11/16 | 33.3% | Current | Normal | |
| 951 | Marketing Team Merger | As we merge the Creative and Graphic Design teams, we need to expand our office space and handle the influx of new employees. | 2/24/15 | 7/22/16 | 0% | Current | Normal | |

Showing 9 projects

Create a Report

SCENARIO — Create a report that looks at incomplete tasks. Group the tasks by priority and use the view called Status.

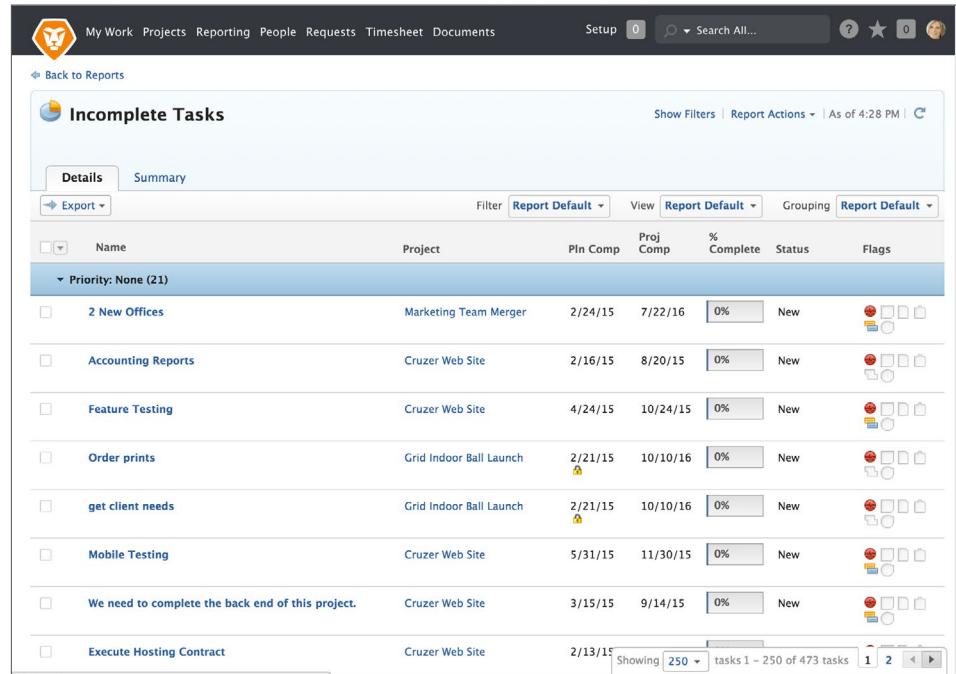
1. Navigate to Reporting in the Global Navigation bar. Select the Reports tab.
2. Using the New Report button, select the type of report to create. For this example, choose task.
3. A light box appears and begins with the column builder.
4. Create a new view or apply an existing view. For this example, apply the existing view called Status.
5. Select the Groupings tab, click Apply an Existing Grouping, and choose Priority.
6. Select the Filters tab, click Apply an Existing Filter, and choose Incomplete Tasks.
7. Navigate to Report Settings in the top right corner to define the report's name and description. Name the report 'Incomplete Tasks'.
8. Select Save & Close.



Create a Report (continued)

The report produces two tabs:

- **Details** — Produces the primary findings of the report. You can see a list of the report findings.
- **Summary** — Produced when using a grouping element. It summarizes information based on the grouping selected during report creation and gives a quick overview of the report.



The screenshot shows the Workfront software interface with the 'Incomplete Tasks' report open. The top navigation bar includes 'My Work', 'Projects', 'Reporting', 'People', 'Requests', 'Timesheet', 'Documents', 'Setup', 'Search All...', and user profile icons. Below the navigation is a toolbar with 'Back to Reports', 'Show Filters', 'Report Actions', and a timestamp 'As of 4:28 PM'. The main content area is titled 'Incomplete Tasks' with tabs for 'Details' (selected) and 'Summary'. A sub-header 'Priority: None (21)' is shown above a list of tasks. The task list includes columns for Name, Project, Pin Comp, Proj Comp, % Complete, Status, and Flags. Each task row contains a checkbox, the task name, the project name, and specific dates. The status column shows 'New' for most tasks, except for one which has a warning icon. The flags column contains various icons representing different task statuses or types. At the bottom of the list, there is a note 'Showing 250 tasks 1 - 250 of 473 tasks' with page navigation buttons.

| Name | Project | Pin Comp | Proj Comp | % Complete | Status | Flags |
|---|-------------------------|----------|-----------|------------|--------|-------|
| 2 New Offices | Marketing Team Merger | 2/24/15 | 7/22/16 | 0% | New | |
| Accounting Reports | Cruzer Web Site | 2/16/15 | 8/20/15 | 0% | New | |
| Feature Testing | Cruzer Web Site | 4/24/15 | 10/24/15 | 0% | New | |
| Order prints | Grid Indoor Ball Launch | 2/21/15 | 10/10/16 | 0% | New | |
| get client needs | Grid Indoor Ball Launch | 2/21/15 | 10/10/16 | 0% | New | |
| Mobile Testing | Cruzer Web Site | 5/31/15 | 11/30/15 | 0% | New | |
| We need to complete the back end of this project. | Cruzer Web Site | 3/15/15 | 9/14/15 | 0% | New | |
| Execute Hosting Contract | Cruzer Web Site | 2/13/15 | | | | |

OR Qualifier

Each line in a filter is treated as an AND condition. However, it is possible to include 'or' clauses to make your reports return results that meet one of several conditions.

The OR option is used to minimize the number of reports that have to be distributed to other users.

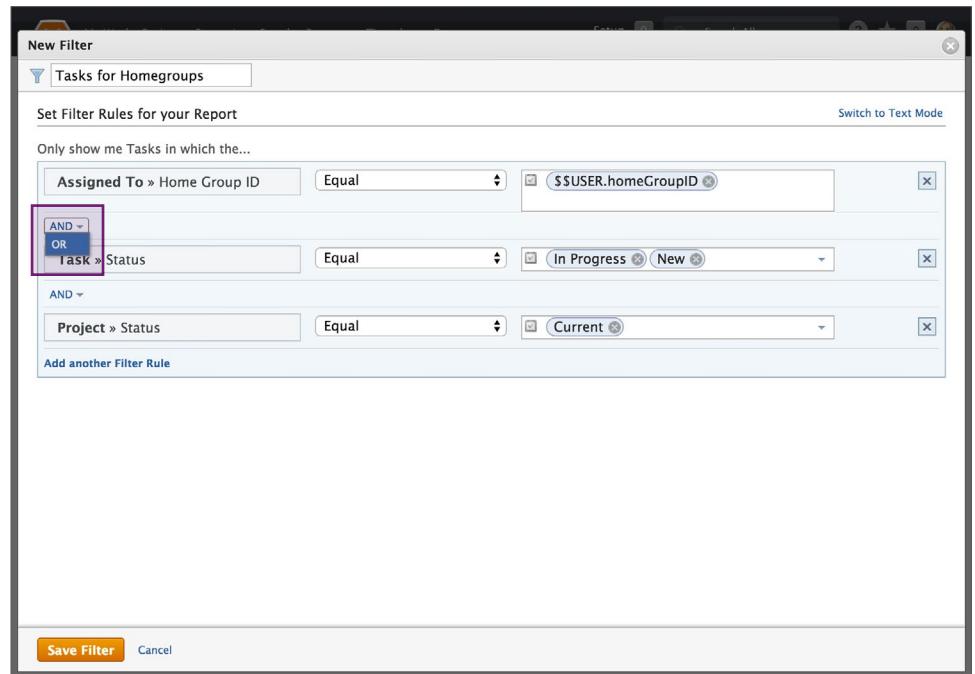
OR statements can only be used with a filter. For example, combine the filters used for the following reports:

- My Upcoming Tasks
- Unassigned Tasks in My Roles
- Late Tasks in Owned By Me Projects

An OR statement provides the ability to combine all three of the filters or clauses from these reports into a single report displaying all of the tasks of interest.

To identify a line as an OR clause, select the AND option between the filter statements and change it to an OR. Your first clause does not require an OR prefix.

Anything that represents a constant must be declared in each clause.



OR Qualifier (continued)

SCENARIO — Create a filter that searches for incomplete tasks on current projects that are either assigned to me or unassigned in my job role.

Create an Or Statement

1. From the Reporting area menu, select an existing Task Report.
2. Create a new Filter, and name the filter (e.g. ‘Incomplete tasks/ Job role’).
3. Select the Percent Complete field name, select the Not Equal qualifier, and input ‘100’.
4. Add a new Filter rule.
5. Select Project Status, use the Equal qualifier, and input ‘Current’.
6. Add a new Filter Rule.
7. Select the Assigned To field source, select the ID field name, use the Equal To qualifier, and input ‘\$\$USER.ID’.
8. Add a new Filter Rule.
9. Select the Percent Complete field name, select the Not Equal qualifier, and input ‘100’.

The screenshot shows the 'New Filter' dialog box. The filter is titled 'Incomplete Tasks/Job Role'. It contains three filter rules:

- Task > Percent Complete: Not Equal (Case Sensitive) to 100
- Project > Status: Equal to Current
- Task > Assigned To ID: Equal to \$\$USER.ID

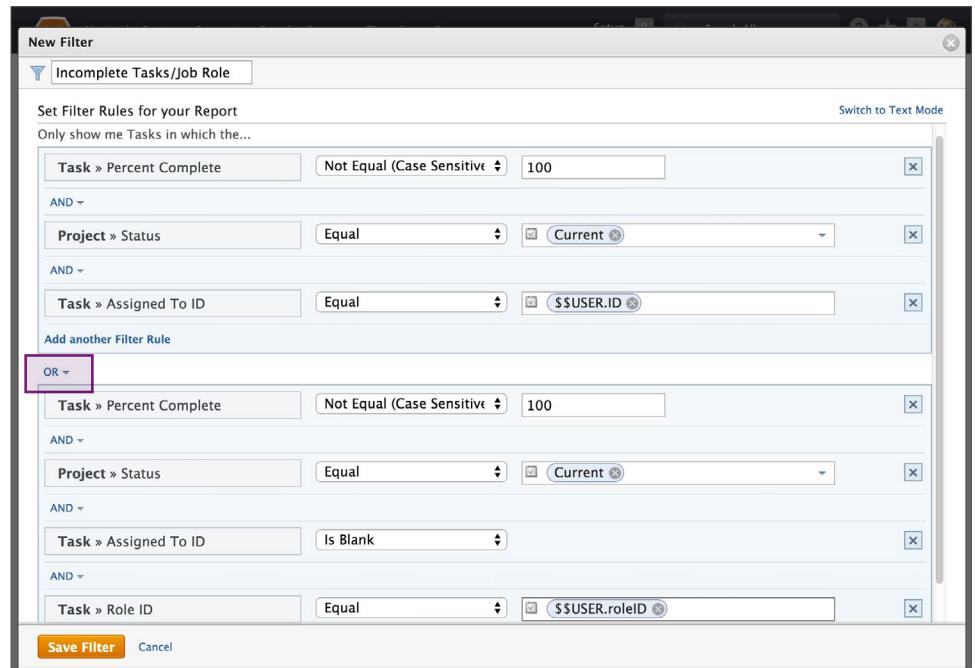
At the bottom of the dialog box are 'Save Filter' and 'Cancel' buttons.

OR Qualifier (continued)

10. Add a new Filter rule.
11. Select Project Status, use the Equal qualifier, and input 'Current'.
12. Add a New Filter rule.
13. Select the Assigned To field source, select the ID field name, use the is blank qualifier, add a new filter rule.
14. Select the Task Role ID field name, use the Equal qualifier, and input '\$\$USER.roleID'.
15. Click the AND after the third filter (Assigned to ID) and switch it to OR.
16. Save when finished.

LIMITATIONS

- There is a limitation when working with OR statements. As the filter queries the database it is limited to searching for five objects, including the object for the report. When you search this limit within the builder, only eligible field sources display in the search drop-down menu.
- Also, be aware that OR statements cannot cross objects. In other words, it does not give users the ability to combine a task list and an issue list in a single report.



CHAPTER 2

ADVANCED FILTERS

OBJECTIVES

After completing this chapter, you will be able to:

- Utilize API Explorer relationships
- Use AND statements and wildcards
- Reference related objects
- Use custom prompts

API Explorer

The API Explorer is likely the most critical reference source used in learning how to write text mode statements.

A text mode statement is the Workfront syntax used to create custom filters. The tables are a comprehensive list of fields for each object and the relationships between objects that can be used when inputting filter criteria.

Each object is displayed in a table, and each table is divided into five sections:

- Actions
- Collections
- Fields
- References
- Search

The screenshot shows the API Explorer interface with the 'Project' object selected. The left sidebar lists various objects: Share, Approval, Approval Process, Approval Path, Approval Step, ApproverStatus, and Assignment. The main content area displays the 'Project' object's details. At the top right is a 'Jump To' menu with sections like 'Limits and Guidelines', 'Basics' (Object URI, Operations, Response), 'Authentication' (Session ID, Request Header, Request Parameter, Cookie), and 'API Version 4.0'. Below the jump menu is a 'Project (Financial Data)' section with a 'FINDAT' button. The 'Project' table has columns for actions, collections, fields (which is selected), references, search, and PROJ. The 'BC Completion State' field is expanded, showing its API key (BCCCompletionState), enum type (com.attask.common.constants.BusinessCaseDetailSectionEnum), field type (string[]), and possible values (SUMMARY (Summary), INFO (Project Info), DEL (Goals), RISK (Risks), COST (Costs), ALIGN (Alignment), DE (Custom Data)). Below this are tables for ID, URL, Actual Benefit, Actual Completion Date, Actual Cost, Actual Duration Expression, Actual Duration Minutes, Actual Expense Cost, and Actual Hours Last Month, each with their respective field names and descriptions.

| | actions | collections | fields | references | search | PROJ |
|--------------------------|--|-------------|---------------|------------|--------|--------------------------|
| BC Completion State | | | | | | BCCCompletionState |
| API Key: | BCCCompletionState | | | | | |
| Enum Type: | com.attask.common.constants.BusinessCaseDetailSectionEnum | | | | | |
| Field Type: | string[] | | | | | |
| Possible Values: | SUMMARY (Summary) INFO (Project Info) DEL (Goals) RISK (Risks) COST (Costs) ALIGN (Alignment) DE (Custom Data) | | | | | |
| ID | | | | | | ID |
| URL | | | | | | URL |
| Actual Benefit | | | | | | actualBenefit |
| Actual Completion Date | | | | | | actualCompletionDate |
| Actual Cost | | | | | | actualCost |
| actualDurationExpression | | | | | | actualDurationExpression |
| Actual Duration Minutes | | | | | | actualDurationMinutes |
| Actual Expense Cost | | | | | | actualExpenseCost |
| Actual Hours Last Month | | | | | | actualHoursLastMonth |

API Explorer (continued)



API EXPLORER TABS

Reference source with a comprehensive list of fields for each object and the relationships between them. Can be used when learning to write text mode statements, which is the Workfront syntax used to create custom files.

| | |
|-------------|--|
| ACTIONS | Conducts a sequence of mini events. Not used in reporting. Used for integrating products. |
| COLLECTIONS | Represents a relationship with another object. Displays one-to-MANY relationships. |
| FIELDS | Fields or columns available for this object as defined in the database. |
| REFERENCES | Display links to other tables that the object is related to. Displays one-to-one relationships. |
| SEARCH | Lists aliases that have been created to simplify queries. With filters, you can only reference objects that are one object away. So, the Search section provides mappings for relationships that are several objects away. |



CHAPTER 2: ADVANCED FILTERS

API Explorer (continued)

Actions

Actions conduct a sequence of events, typically mini actions. This area is not used in reporting but is for users integrating products with Workfront.

Collections

Collections represent a relationship to another object and display one-to-many relationships. For example, one-to-many relationships mean you can pull in a column that shows all assignments on a task or project. This is when you need to use collections.

For example, each task may have several hour entries recorded. By referencing the Collection Name ‘hours,’ you can query for tasks where any of the hour entries match the criteria. When you input hours:entryDate=\$\$TODAY, the search returns all tasks where hours have been recorded today.

| Task | actions | collections | fields | references | search | TASK |
|------------------------|---------|-------------|--------|------------|--------|------------------------|
| acceptWork | | | | | | acceptWork |
| approveApproval | | | | | | approveApproval |
| assign | | | | | | assign |
| bulkCopy | | | | | | bulkCopy |
| bulkMove | | | | | | bulkMove |
| calculateDataExtension | | | | | | calculateDataExtension |
| markDone | | | | | | markDone |
| markNotDone | | | | | | markNotDone |
| Move | | | | | | move |
| recallApproval | | | | | | recallApproval |
| rejectApproval | | | | | | rejectApproval |
| Task | actions | collections | fields | references | search | TASK |
| replyToAssign | | | | | | |
| unacceptWork | | | | | | accessRules |
| unassign | | | | | | allPriorities |
| unassignOccu | | | | | | allStatuses |
| Template Assignment | | | | | | approverStatuses |
| Team Member | | | | | | assignments |
| | | | | | | children |
| | | | | | | documents |
| | | | | | | doneStatuses |
| | | | | | | expenses |
| | | | | | | hours |
| | | | | | | Issues |
| | | | | | | openOpTasks |
| | | | | | | predecessors |
| | | | | | | resolvables |
| | | | | | | successors |
| | | | | | | updates |

CHAPTER 2: ADVANCED FILTERS

API Explorer (continued)

Fields

The section identifies the fields or columns available for this object as defined in the database.

References

The References section displays links to the other tables the object is related to. Generally, these represent one-to-one relationships. For example, each task can only be associated with a single custom form (category), so the option to navigate to the Custom Form table is available.

The References section contains names used in cross-object searches. When creating text mode statements, reference these related objects by inputting the reference name, colon, and the field camel case from the second object (e.g., category:name=...)

| Task | actions | collections | fields | references | search | TASK |
|----------------------------|---------|-------------|-------------|------------|------------|---------------------------|
| ID | | | | | | ID |
| URL | | | | | | URL |
| Actual Completion Date | | | | | | actualCompletionDate |
| Actual Cost | | | | | | actualCost |
| Actual Duration | | | | | | actualDuration |
| Actual Duration Minutes | | | | | | actualDurationMinutes |
| Actual Expense Cost | | | | | | actualExpenseCost |
| Actual Labor Cost | | | | | | actualLaborCost |
| Actual Revenue | | | | | | actualRevenue |
| Actual Start Date | | | | | | actualStartDate |
| Actual Hour | Task | actions | collections | fields | references | search |
| Actual Hour | | | | | | |
| Approval E | | | | | | |
| Approval P | | | | | | |
| approvalPla | | | | | | |
| Approval Process | | | | | | approvalProcess |
| Assigned To | | | | | | assignedTo |
| Billing Record | | | | | | billingRecord |
| Category | | | | | | category |
| Converted Issue Originator | | | | | | convertedOpTaskOriginator |
| Current Approval Step | | | | | | currentApprovalStep |
| Customer | | | | | | customer |
| Default Baseline Task | | | | | | defaultBaselineTask |
| Entered By | | | | | | enteredBy |
| Group | | | | | | group |
| Iteration | | | | | | iteration |
| Last Condition Note | | | | | | lastConditionNote |
| Last Note | | | | | | lastNote |
| Last Updated By | | | | | | lastUpdatedBy |
| Milestone | | | | | | milestone |
| Parent | | | | | | parent |
| Primary Assignment | | | | | | primaryAssignment |
| Project | | | | | | project |



API Explorer (continued)

Search

The Search section lists aliases that have been created to simplify queries. For example, the database does not contain a column for Is Complete; just the same, the application provides a field, `isComplete`, to allow for quick query of tasks that are either complete or incomplete.

One of the limitations of filters is that you can only reference objects one degree away. The search section provides mappings, or aliases, for relationships that are several objects away.

For example, every task must belong to a project, and each project may belong to a portfolio. Browsing through the searches section, notice a `portfolioID` option that allows you to create filters based on the portfolio a task belongs to. This is another field that is not actually stored on the tasks table in the database, but the application provides it as a reference.

| Task | actions | collections | fields | references | search | TASK |
|------------------------|---------|-------------|--------|------------|--------|--|
| All Notes | | | | | | <code>allNotesOM</code> |
| Approval Required | | | | | | <code>approvalRequired</code> |
| Assignment Roles | | | | | | <code>assignmentsRolesMM</code> |
| Assignment Users | | | | | | <code>assignmentsUsersMM</code> |
| Step Approvers | | | | | | <code>awaitingApprovalStepApproversMM</code> |
| Current Role Approvers | | | | | | <code>currentRoleApproversMM</code> |
| Current User Approvers | | | | | | <code>currentUserApproversMM</code> |
| Is Approval | | | | | | <code>isApproval</code> |
| Is Complete | | | | | | <code>isComplete</code> |
| Journal Entries | | | | | | <code>journalEntries</code> |
| Notes | | | | | | <code>notes</code> |
| Pending Approval | | | | | | <code>pendingApproval</code> |
| Pending Issues | | | | | | <code>pendingIssues</code> |
| Predecessors | | | | | | <code>predecessorsMM</code> |
| Project Company | | | | | | <code>projectCompanyMM</code> |
| Project Milestone Path | | | | | | <code>projectMilestonePathMM</code> |
| Project Owner ID | | | | | | <code>projectOwnerId</code> |
| Project Owner | | | | | | <code>projectOwnerMM</code> |
| Project Portfolio ID | | | | | | <code>projectPortfolioID</code> |
| Project Program ID | | | | | | <code>projectProgramID</code> |
| Project Sponsor ID | | | | | | <code>projectSponsorID</code> |
| Project User IDs | | | | | | <code>projectUserIDs</code> |
| Project User Roles | | | | | | <code>projectUserRolesOM</code> |
| Project Users | | | | | | <code>projectUsersOM</code> |
| Request Status | | | | | | <code>requeststatus</code> |
| Pending Status | | | | | | <code>statusModifier</code> |
| Successors | | | | | | <code>successorsMM</code> |
| Team Timelineable ID | | | | | | <code>teamTimelineableID</code> |



CHAPTER 2: ADVANCED FILTERS

API Explorer (continued)

The image displays segments of the Task table taken from the API Explorer. To see more details about a field, click the name to open a display box. Clicking the name again closes it.

Fields

Each field in Workfront has an associated field label seen in the left column in the image. Notice each field has an associated field camel case (right column) that uses medial capitals.

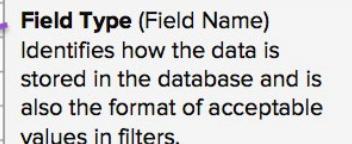
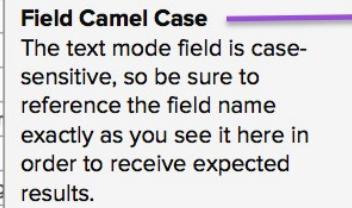
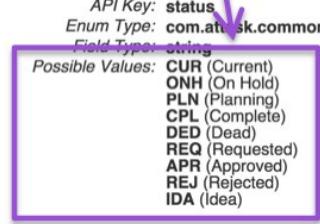
The field camel case is what you will use in text mode, and it is case-sensitive. Be sure to reference the field correctly; otherwise, you will not receive the results you anticipate.

Field Type

The Field Data Type identifies how the data is stored in the database and the format of acceptable values in filters.

Possible Values

When applicable, this column displays the acceptable values you should use in your filter. Both a label and a value are provided.

| API Explorer - fields tab | |
|---|---|
| Project | actions collections fields references search PROJ |
| BC Completion State | BCCCompletionState |
| ID | ID |
| URL | URL |
| Actual Benefit | Actual Benefit  Field Type (Field Name) Identifies how the data is stored in the database and is also the format of acceptable values in filters. |
| Actual Completion Date | actualCompletionDate |
| Actual Cost | actualCost |
| actualDurationExpression | actualDurationExpression |
| Actual Duration Minutes | actualDurationMinutes  Field Camel Case The text mode field is case-sensitive, so be sure to reference the field name exactly as you see it here in order to receive expected results. |
| Actual Expense Cost | actualExpenseCost |
| Actual Hours Last Month | actualHoursLastMonth |
| Actual Hours Last Three Months | actualHoursLastThreeMonths |
| Actual Hours This Month | actualHoursThisMonth |
| Actual Hours Two Months Ago | actualHoursTwoMonthsAgo |
| Actual Labor Cost | actualLaborCost |
| Actual Revenue | actualRevenue |
| Actual Risk Cost | actualRiskCost |
| Actual Start Date | actualStartDate |
| Actual Value | actualValue |
| Status | status |
|  API Key: status Enum Type: com.atwsk.common.constants.ProjectStatusEnum Field Type: string Possible Values: CUR (Current) ONH (On Hold) PLN (Planning) CPL (Complete) DED (Dead) REQ (Requested) APR (Approved) REJ (Rejected) IDA (Idea) | |





Data Types

Data Types are used in the API Explorer. The following table provides some data type examples. These are also used in Text Mode.

| ATTRIBUTE DATA TYPE | DESCRIPTION | EXAMPLE |
|---------------------|---|--|
| INTEGER | A whole number | numberOfOpenIssues=1 |
| BOOLEAN | A true or false value | isCritical=true |
| STRING | A word or phrase | name=Project XYZ |
| DOUBLE | A number that can use decimals | percentComplete=100.00 |
| DATE | A calendar date | plannedCompletionDate=05/24/2010 |
| BEAN | This references another searchable Workfront object with its own set of attributes. | enteredBy:lastName=Smith In this case, you may be using a task query. The enteredBy attribute references the user object. The lastName attribute of the user object is Smith. |





Variation In Verbiage

When working with the API Explorer, notice that some field camel cases don't match the Field Labels.

This is true when working with Planned and Actual Hours in Workfront. These terms are referenced as workRequired. It is important that any time you are referencing these items in text mode, do so using workRequired and not Planned Hours.

The table indicates how these fields are viewed in the interface builder versus how they are viewed by the database.

When using the interface builder to establish a reporting element and then switching to text mode, Workfront recognizes the terms and adjusts the verbiage to assist in setting up queries.

| FOUND IN INTERFACE BUILDER | FOUND IN DATABASE |
|----------------------------|----------------------|
| Planned Hours | workRequired |
| Actual Hours | actualWorkRequired |
| Original Planned Hours | originalWorkRequired |
| Number of Open Issues | numberOpenOpTasks |



Text Mode Structure and Syntax

The format of a text mode statement must be exact; otherwise, the results do not render. From a generic level, the syntax looks something like this:

```
field camel case=value
```

More specific examples are included in the table to the right. The examples here are related to the Task Field Source.

| EXAMPLE |
|---------------------|
| name=Task ABC |
| percentComplete=0 |
| taskConstraint=ASAP |
| durationType=A |



Using Text from the Course Book

This course book is full of great resources to assist in further developing your reporting needs. If formatting or expressions from the course book are copied directly into the Workfront interface, you should confirm that no additional spaces are present after statements or within the expression. This will result in breaking the format and will not return the expected results.

Watch the quotation marks

Furthermore, when writing text mode statements that include quotation marks, it is important to know that a straight double quotation mark ("") is needed for the statement to work properly. Should statements be written and transferred to the text mode builder, please check to confirm that the quotations are in the desired format ("") and not in a the standard slanted (“ ”) format.



Using Text Mode

SCENARIO — Create a report to show all working tasks (tasks with no children).

1. From the Reporting area, create a new task report.
2. Click the Filters tab.
3. Select Switch to text mode.
4. Open a new window, navigate to the API Explorer, and find the Task table.
5. Locate the field camel case for Number of Children, copy it, and paste it into your text mode filter.
6. Add an equal sign and input the number zero (0).
7. Click the Save + Close button.
8. Name the report 'All Working Tasks'.

The screenshot shows a software interface titled "New Task Report". At the top, there are tabs: "Columns (View)", "Groupings", "Filters" (which is highlighted in blue), and "Chart". Below the tabs, the text "Set Filter Rules for your Report" is displayed. In the main area, there is a text input field containing the filter rule "numberOfChildren=0".



Using Text Mode (continued)

SCENARIO — Create a report that shows all tasks in the New status.

1. From the Reporting area, create a new task report.
2. Click the Filters tab.
3. Select Switch to text mode.
4. Open a new window, navigate to the API Explorer, and find the Task table.
5. Locate the field camel case for Status, copy it, and paste it into your text mode filter.
6. Add an equal sign.
7. Return to the task table in the API Explorer and under Possible Values find the New value. Copy and paste this after the equal sign in your text mode Filter.
8. Click the Save + Close button.
9. Name the report ‘New Tasks’.

The screenshot shows a software interface for creating a report titled 'New Tasks'. At the top, there are tabs for 'Columns (View)', 'Groupings', 'Filters' (which is currently selected), and 'Chart'. Below the tabs, the title 'Set Filter Rules for your Report' is displayed. Underneath the title, the filter rule 'status=NEW' is shown in a text input field. The background of the interface is white, and the overall layout is clean and modern.



AND Statements

The AND option is usually employed to minimize the number of reports that have to be distributed to other users. For example, combine the filters used for the following reports:

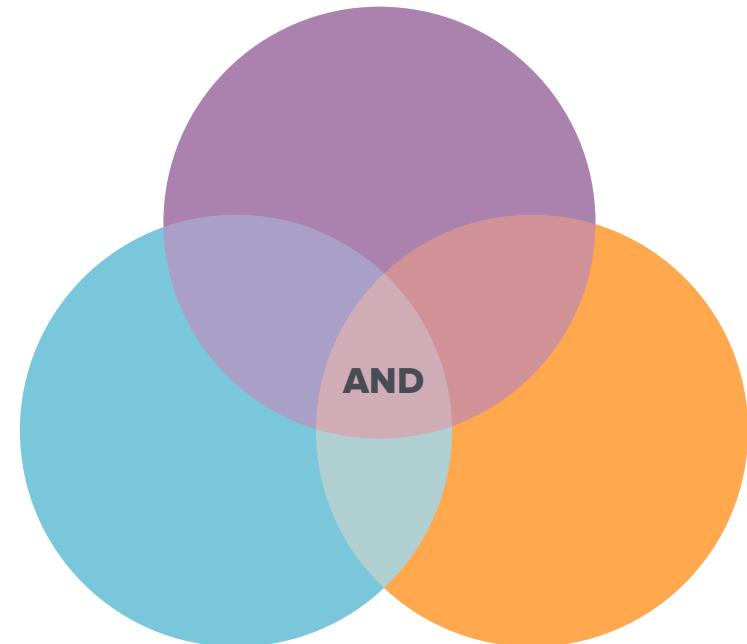
- My Upcoming Tasks
- Unassigned Tasks in My Roles
- Late Tasks in Projects Owned By Me

The default behavior in the filter builder is to remove duplicate criteria that reference the same field. The AND statement in the text mode interface gives the ability to bypass the builder restrictions.

An AND statement provides the ability to combine all three filters or clauses from these reports into a single report displaying all tasks of interest. The diagram illustrates that an AND statement provides results from all three clauses.

With AND statements, only the portion where each clause intersects are returned on the results.

To identify a line as an AND clause in text mode add 'AND:1:' or 'AND:2:' and so on, to the beginning of a line that deviates from the base clause. The first clause does not require an AND prefix. You can add up to three AND statements.





Using AND Statements

SCENARIO — For this example, assume that you are interested in viewing all projects that contain several key words in the description, including:

- Technolog -y and -ies
- Industr -y and -ies
- Green

Create the AND Filter Described in the Scenario

1. Create a new filter on a project. Name it ‘Description Technology’.
2. Using the builder, enter Project >> description contains technolog.
3. Click Switch to text mode.
4. Copy the filter text and paste it twice.
5. In front of one of the copied conditions, insert AND:1 on both lines.
6. Change the criteria to industr.
7. In front of the other copied condition, insert AND:2 on both lines.
8. Change the criteria to green.
9. Save Filter.

Using AND Statements (continued)

The project's description must contain all three words. If it only contains two it does not return the desired results. These key words can appear in any order within the text and the AND statement returns results.

If you want the results to return the project name or the description containing any of the three words, then an OR statement is more appropriate.

The benefit of an AND statement is that it is somewhat easier to compose than a Like filter because you do not have to account for each possible sequence of words. However, it has some of the same limitations that an OR filter possesses. Namely, there is a limitation of four fields. This means that a field can be referenced up to four times when using AND statements.

Customize Filter

Description Technology

Set Filter Rules for your Report

```
description=technolog
description_Mod=cicontains
AND:1:description=industr
AND:1:description_Mod=cicontains
AND:2:description=green
AND:2:description_Mod=cicontains
```

Qualifiers

Users do not always search for a discrete value; rather, they search for a range of possible values. Within the interface they have the ability to use qualifiers such as greater than, less than, between, contains, etc.

To use a qualifier hit return and input the field label followed by '_Mod=' and the qualifier. For example, to search for tasks where the percentage complete is less than 50% you could use:

```
percentComplete=50
percentComplete_Mod=lt
```

The table displays the qualifier options, the text mode syntax to reference each modifier, and an example.

| QUALIFIER | TEXT MODE SYNTAX | EXAMPLE |
|--------------------|------------------|--|
| Equal | eq | hours=5 hours_Mod=eq |
| Not Equal | ne | hours<5 hours_Mod=ne |
| In | in | priority=1,2,3 priority_Mod=in |
| Not In | notin | priority<1,2,3 priority_Mod=notin |
| Between | between | workRequired=5 workRequired_Mod=between workRequired_Range=10 |
| Not Between | notbetween | workRequired=5 workRequired_Mod=notbetween workRequired_Range=10 |
| Less Than | lt | hours=5 hours_Mod=lt |
| Less Than/Equal | lte | hours=5 hours_Mod=lte |
| Greater Than | gt | hours=5 hours_Mod=gt |
| Greater Than/Equal | gte | hours=5 hours_Mod=gte |





Text Attribute and Field Qualifiers

| QUALIFIER | TEXT MODE SYNTAX | EXAMPLE |
|------------------|------------------|--|
| Contains | contains | name=Proj name_Mod=contains |
| Does Not Contain | notcontains | name=Proj name_Mod=notcontains |
| Equal | eq | name=Project XYZ name_Mod=eq or name=Project XYZ |
| Not Equal | ne | name=Project XYZ name_Mod=ne |
| Null | isnull | actualCompletionDate=0 actualCompletionDate_Mod=isnull |
| Not Null | notnull | actualCompletionDate=0 actualCompletionDate_Mod=notnull |
| Like | like | description=%important%meeting% description_Mod=like <i>or for single character replacement use a '?'</i> description=%important?meeting% description_Mod=like |
| In | in | status=CUR,PLN,APR status_Mod=in |
| Not In | notin | status=CUR,PLN,APR status_Mod=notin |



Text Attribute and Field Qualifiers (continued)

| QUALIFIER | TEXT MODE SYNTAX | EXAMPLE |
|--|------------------|---|
| Blank | isblank | actualCompletionDate=0 actualCompletionDate_Mod=isblank |
| Not Blank | notblank | actualCompletionDate=0 actualCompletionDate_Mod=notblank |
| Sounds Like | soundex | name=their name_Mod=soundex This is the same as searching for their, there, they're, and several other similar words. |
| Contains (case insensitive) | cicontains | name=Proj name_Mod=cicontains |
| Does Not Contain (case insensitive) | cinotcontains | name=Proj name_Mod=cinotcontains |
| Equal (case insensitive) | cieq | name=project xyz name_Mod=cieq |
| Not Equal (case insensitive) | cine | name=project xyz name_Mod=cine |



Using Qualifiers

SCENARIO — Create a report to show tasks with more than 10 Actual Hours reported on them.

1. From the Reporting area, create a Task Report.
2. Click the Filters tab. Switch to text mode.
3. Go to the API Explorer, and find the task table.
4. Find the field camel case for Actual Hours on a task.
5. Copy/paste or input the field camel case into your text mode filter. Add =600.
6. Add the qualifier for ‘greater than’.
7. Go to the Columns tab and select Work Breakdown from Apply an Existing View.
8. Click the Save + Close button. Name the report ‘Tasks with Actual Hours gt 10’.

New Task Report

Set Filter Rules for your Report

| | |
|------------------------|---------------------------|
| actualWorkRequired=600 | actualWorkRequired_Mod=gt |
|------------------------|---------------------------|

NOTE

The Work Breakdown view shows actual hours. Actual Hours has a field camel case of actualWorkRequired and Planned Hours has a field camel case of workRequired. Both these values are stored in minutes.



Using Qualifiers (continued)

SCENARIO — Create a report to show all hour entries that belong to tasks.

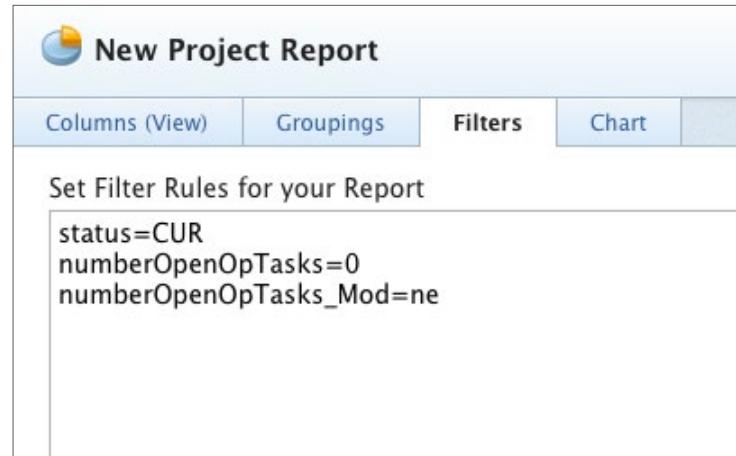
1. From the Reporting Area menu, create an Hour Report.
2. Click the Filters tab; switch to text mode.
3. Go to the API Explorer, and find the Hour table.
4. Find the field camel case for Task ID.
5. Copy/paste or input the field camel case, add+Mod=, and then add the notblank qualifier in the text mode filter.
6. Click the Save + Close button. Name the report Hours Belonging to Tasks.

The screenshot shows a software interface for creating a report. At the top, there's a title bar with a globe icon and the text 'New Hour Report'. Below the title bar is a navigation bar with four tabs: 'Columns (View)', 'Groupings', 'Filters' (which is highlighted in blue), and 'Chart'. The main area of the window is titled 'Set Filter Rules for your Report'. Inside this area, there is a single line of text: 'taskID_Mod=notblank'. The background of the window is white, and the overall interface has a clean, modern look.

Using Qualifiers (continued)

SCENARIO — Create a report to show current projects that have open issues.

1. From the Reporting Area menu, create a Project Report.
2. Click on the Filters tab; switch to text mode.
3. Go to the API Explorer and find the Project table. Find the field camel case and valid values for Current status.
4. Find the field camel case to identify projects with issues (Hint: Issues are referred to as opTasks when creating filters, views, and groupings).
5. Copy/paste or input the values you need for your search into your text mode filter.
6. Click the Save + Close button. Name the report ‘Current Projects With Open Issues’.



The screenshot shows the 'New Project Report' interface. At the top, there are tabs for 'Columns (View)', 'Groupings', 'Filters' (which is selected), and 'Chart'. Below the tabs, the title 'Set Filter Rules for your Report' is displayed. Underneath the title, three filter rules are listed:
status=CUR
numberOpenOpTasks=0
numberOpenOpTasks_Mod=ne

Using Qualifiers (continued)

SCENARIO — Create a report to show projects that have between 3 and 6 open issues.

1. From the Reporting Area menu, create a new Project Report.
 2. From the Columns tab, click on + Add Column. Set up this column to show Number of Open Issues.
 3. Click on the Filters tab. Switch to text mode.
 4. Go to the API Explorer and find the Project table.
 5. Find the field camel case for Number of Open Issues. Copy/paste or input the field camel case into your text mode filter. Add =3. This specifies one end of your range.
 6. Add the qualifier between.
 7. Now add _Range to your field camel case and add =6 after it. This specifies the other end of your range. Your text mode code should look like this:
- ```
numberOpenOpTasks=3
numberOpenOpTasks_Mod=between
numberOpenOpTasks_Range=6
```
8. Click Done to save your text mode code.
  9. Click the Save + Close button. Name the report ‘Projects with 3 to 6 Open Issues.’

The screenshot shows a report configuration interface with a purple header. The title is "Projects with 3-6 open issues". Below the title are tabs: "Columns (View)", "Groupings", "Filters" (which is highlighted in blue), and "Chart". A sub-section titled "Set Filter Rules for your Report" contains the filter rule: "numberOpenOpTasks=3", "numberOpenOpTasks\_Mod=between", and "numberOpenOpTasks\_Range=6".

# Date-Based Wildcards

Date wildcards can be combined with the attribute ‘q’, ‘h’, ‘d’, ‘w’, ‘m’, and ‘y’ for calendar quarter, hour, day, week, month, year respectively.

The modifier ‘b’ and ‘e’ stand for ‘beginning’ and ‘ending’ respectively.

The operators ‘+’ and ‘-’ are used to add or subtract values from the wildcard value.

## Example

The wildcard, \$\$TODAYb+2w is the same as saying, ‘Two weeks from the beginning of this week’. The wildcard \$\$NOW+2h is the same as saying ‘two hours from now’.

There are several built-in date wildcards like ‘This Quarter’ or ‘Next Week’ available to use in a filter as well.

There are several date filters built in Workfront that incorporate wildcards for the user’s convenience.

However, you may find the need to build upon those foundations to gather information for your reports.

| DATE-BASED WILDCARD | DESCRIPTION                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| \$\$NOW             | This wildcard looks at the date and time as of right now. This option is used in combination with any date filter attribute. For example, if you want to display all hour entries provided up to the current time, you can do this by using the following expression: Planned Start Date < \$\$NOW. This is preferred over defining a filter and using the actual current date and time. That way each time the filter runs, you won’t have to modify it. \$\$NOW is equal to the current date and time. |
| \$\$TODAY           | This wildcard looks at the date and time as of midnight today. This option can be used in combination with any date filter attribute. For example, if you want to display all tasks due before today, you could use the following expression: Planned Start Date < \$\$TODAY. This is preferable to defining a filter with today’s date so you will not have to modify the filter again tomorrow, next week, or next month. \$\$TODAY is always equal to midnight for the current day.                   |



# User-Based Wildcard Filter Variables

You can also use wildcards in text mode statements.

| USER-BASED WILDCARD  | DESCRIPTION                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| \$\$USER.ID          | \$\$USER.ID is a variable equal to the logged-in user's ID. This is the ID used to identify work assignments and who created each object. Therefore, it is the variable used on the My reports, such as My Tasks, My Projects, My Hours, etc. This wildcard option decreases the number of reports an Implementation Manager needs to create because the same report can be used for several users and the results change based on the logged in user's ID.                                                          |
| \$\$USER.name        | The \$\$USER.name variable allows you to do name matches in a filter. It provides the logged-in user's full name (first name + last name). This is not the logged in user's username.                                                                                                                                                                                                                                                                                                                                |
| \$\$USER.homeGroupID | The \$\$USER.homeGroupID variable identifies the logged-in user's home group ID. This is used primarily for group managers who want to see only objects related to their home group. For example, a manager may want to see all incomplete tasks on projects in his group: Project: Home Group ID = \$\$USER.homeGroupID Percent Complete < 100. Or a manager may want to see all incomplete tasks assigned to individuals in their group: Assigned To: Home Group ID = \$\$USER.homeGroupID Percent Complete < 100. |



# User-Based Wildcard Filter Variables (continued)

| USER-BASED VARIABLES          | DESCRIPTION                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>\$\$USER.categoryID</b>    | The \$\$USER.categoryID variable identifies the custom data category associated with the logged in user's profile and returns the ID number of the category.                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>\$\$USER.accessLevelID</b> | The \$\$USER.accessLevelID variable identifies the access level associated with the logged in user's profile and returns the ID number of the access level.                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>\$\$USER.companyID</b>     | The \$\$USER.companyID variable identifies the company associated with the logged-in user's profile and returns the ID number of the company.                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>\$\$USER.customerID</b>    | The \$\$USER.customerID variable identifies the customer account ID associated with your environment. This variable is unique because it is typically only used when building integrations through the API.                                                                                                                                                                                                                                                                                                                                                                 |
| <b>\$\$USER.otherGroupIDs</b> | The \$\$USER.otherGroupIDs variable returns an array of all of the group's ID values associated with the logged-in user's profile. The use cases for this variable are similar to the \$\$USER.homeGroupID option, except the results would only display work across all groups the manager belongs to.                                                                                                                                                                                                                                                                     |
| <b>\$\$USER.roleID</b>        | The \$\$USER.roleID variable returns the logged in user's default role assignment. This allows you to report on tasks or issues assigned to a default job role. \$\$USER.roleIDs provides an array of the logged-in user's role assignments, allowing you to create filters that return results associated with all of the logged-in user's role associations. If a user has job roles defined beyond his/her default job role, this variable returns the values of those additional roles. You can use those values to find all of the logged-in user's role associations. |
| <b>\$\$USER.roleIDs</b>       | If a user has job roles defined beyond his/her default job role, this variable returns the values of those additional roles. You can use those values to find all of the logged in user's role associations.                                                                                                                                                                                                                                                                                                                                                                |



# Date-Based Wildcards

**SCENARIO** — Create a report to show tasks with a Planned Start Date earlier than today.

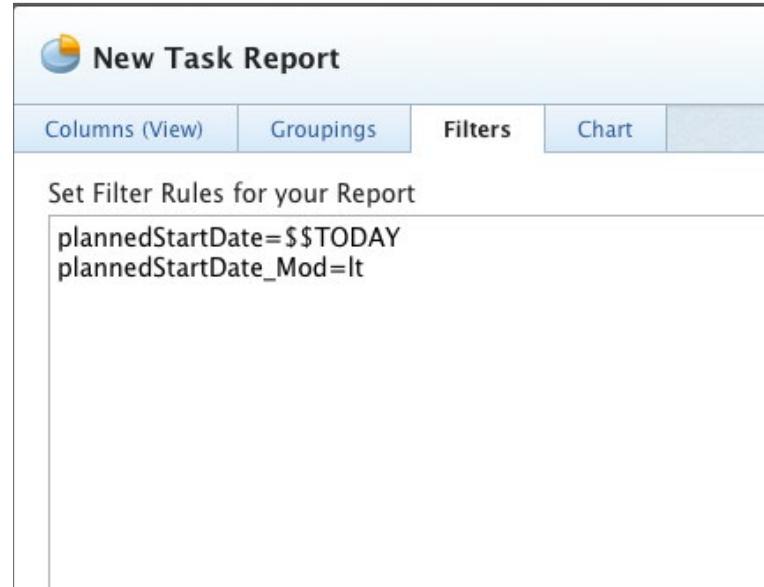
1. From the Reporting Area menu, create a Task Report.
2. Click the Filters tab, switch to text mode.
3. Go to the API Explorer and find the Task table. Find the field camel case for planned start date.
4. Copy/paste or input the field camel case into your text mode filter; use the \$\$TODAY wildcard and a Less Than qualifier.
5. Click the Save + Close button. Name the report ‘Tasks Starting Before Today’.

Sometimes you need to search for numeric values and dates based on a range of values. This can be done by substituting the ‘\_Mod’ with ‘\_Range’ to specify a second value. For example, you might search for all tasks with a planned completion date between January 1, 2012, and February 28, 2012. This can be done using the following statement:

```
plannedCompletionDate=01/01/2012
plannedCompletionDate_Range=02/28/2012
```

Instead of using fixed dates to show tasks between January 1, 20XX, and February 28, 20XX, it may be more beneficial to display tasks through a wildcard range. The statement to display the tasks due to start in the this quarter may look like this:

```
plannedStartDate=$$TODAYbq
plannedStartDate_Range=$$TODAYeq
```



The screenshot shows the 'New Task Report' interface. At the top, there are tabs: 'Columns (View)', 'Groupings', 'Filters' (which is selected), and 'Chart'. Below the tabs, the title 'New Task Report' is displayed next to a blue gear icon. A section titled 'Set Filter Rules for your Report' contains the following filter rules:  
**plannedStartDate=\$\$TODAY  
plannedStartDate\_Mod=lt**

# User-Based Wildcards

**SCENARIO** — Create a report to show tasks assigned to one of the job roles belonging to the logged-in user.

1. From the Reporting area, create a Task report.
2. Click the Filters tab; switch to text mode.
3. Go to the API Explorer and find the Task table. Find the field camel case for Role ID.
4. Copy/paste or input the field camel case into your text mode filter; use the `$$USER.roleIDs` wildcard and the appropriate qualifier.
5. Click the Save + Close button. Name the report ‘Tasks Assigned in My Roles’.

The screenshot shows a report configuration interface with the following details:

- Title:** tasks assigned in my roles
- Filter Tab:** The "Filters" tab is selected, indicated by a blue border around it.
- Filter Rule:** The rule `roleID= $$USER.roleIDs` is listed under the "Set Filter Rules for your Report" section.
- Other Tabs:** Other tabs visible include "Columns (View)", "Groupings", and "Chart".

## NOTE

This report shows all unassigned tasks associated with one of the logged in users Primary and Other job roles.



# Referencing Related Objects

The text mode field allows you to search for fields not available in the builder. Text mode extends your filter criteria to access relationships through the References section of the API Explorer. This enables users to filter on more criteria than the builder interface allows. Keep in mind that when referencing other object types in Workfront, you can reference only one other table on a filter. Views and groupings allow you to reference more object tables, but depending on the table you start from, you are limited as to the number of references you can make.

To use a relationship, reference the object as identified in the API Explorer followed by a colon (:).

The following example produces results based on both task and project criteria in a task report.

```
percentComplete=100
percentComplete_Mod=lt
project:status=CUR,PLN
project:status_Mod=in
project:plannedCompletionDate=$$TODAY
project:plannedCompletionDate_Mod=lte
```

## NOTE

It is permissible to filter on several attributes or field labels in a single text mode filter. In this case, each line break is treated as an AND condition, meaning all the conditions must be met for an item to be included in the results.



# Referencing Related Objects (continued)

**SCENARIO** — Create a report to show projects in the IT group.

1. From the Reporting area, create a Project Report.
2. Click the Filters tab.
3. Switch to text mode.
4. Go to the API Explorer and find the Project table. Find the reference name for Group from the References tab and copy it.
5. Click the link to the Group table and find the Name field.
6. Copy/paste or input the reference name and field camel case into your text mode filter.
7. Your statement should be `group:name=IT`
8. Click the Save + Close button. Name the report ‘Projects in the IT Group’.

The screenshot shows a software interface titled "New Project Report". At the top, there are tabs: "Columns (View)", "Groupings", "Filters" (which is highlighted in blue), and "Chart". Below the tabs, the text "Set Filter Rules for your Report" is displayed. In the main area, there is a text input field containing the filter rule "group:name=IT".

# Referencing Related Objects (continued)

**SCENARIO** — Create a report to show tasks where the project progress status is Late, At Risk, or Behind.

1. From the Reporting area, create a task report.
2. Click the Filters tab. Switch to text mode.
3. Go to the API Explorer and find the Task table. Find and copy the reference name for Project from the reference tab.
4. From the Project table, find the Progress Status field label and make note of the available values for Progress Status.
5. Copy/paste or input the reference name, field camel case, and possible values into your text mode filter.
6. Your statement should be:  
project:progressStatus=LT,RK,BH  
project:progressStatus\_Mod=in
7. Click the Save + Close button. Name the report ‘Tasks on Late, At Risk, or Behind Projects’.

 Tasks on Late, At Risks, or Behind Projects

Columns (View)   Groupings   **Filters**   Chart

Set Filter Rules for your Report

```
project:progressStatus=LT,RK,BH
project:progressStatus_Mod=in
```



# Custom Prompts

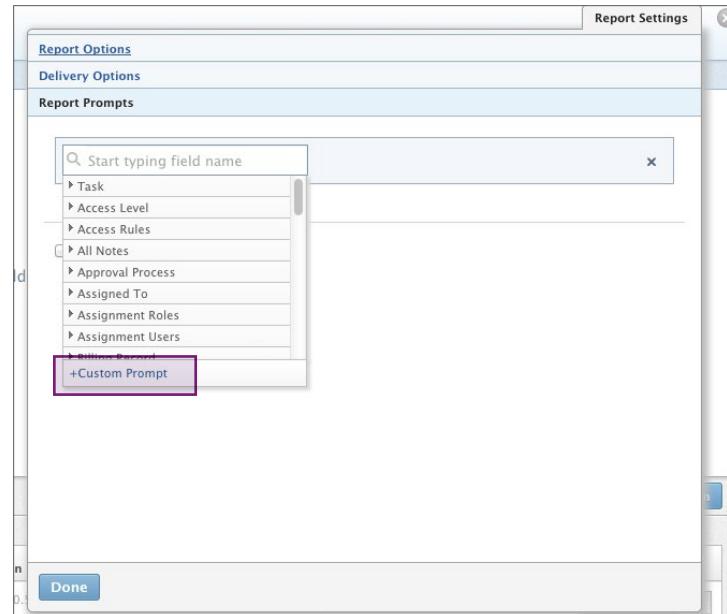
Utilize text mode syntax when building prompts on a report to build custom drop-down fields. The custom filter drop-down menu on the prompt screen allows for multiple conditions to be applied in a single field.

The last option in the Report Settings drop-down menu is Custom Prompts. The Custom Prompts option defines a custom list of drop-down menu prompts for the report.

The options that appear in the Prompts screen are defined in the drop-down menu Item Label field, and the logic behind each label is input into the Condition field.

## NOTE

The condition for each option is written in text mode syntax. Instead of hard returns in the text-box, identify line breaks using an ampersand (&).



# Custom Prompts (continued)

**SCENARIO** — Create a task report custom drop-down menu prompt that will allow the user to just see tasks on:

- All Future Projects
- All Late Projects
- Projects Due This Month
- Projects Due Next Month

For each of these options we assume the projects are in the Idea, Requested, Planning, or Current statuses.

1. From the Reporting area, create a task report.
2. Select Report Settings > Report Options in the upper-right corner.
3. Select the Add a Prompt button and Custom Prompt from the bottom of the options list.
4. In the Field Name box located on the right, enter the name Quick Filters.
5. In the Item Label field put the name of each prompt label, and include the text mode describing the condition to be executed for that prompt in the corresponding condition field.
6. Click the Save + Close button. Name the report ‘Tasks with Custom Prompt.’

| LABEL                          | CONDITION                                                                                                                                                  |
|--------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>All Future Projects</b>     | project:plannedStartDate=\$\$TODAY&<br>project:plannedStartDate_Mod=gte&<br>project:status=IDA,REQ,PLN,CUR&project:status_Mod=in                           |
| <b>All Late Projects</b>       | project:plannedCompletionDate=\$\$TODAY&<br>project:plannedCompletionDate_Mod=lt&<br>project:status=IDA,REQ,PLN,CUR&project:status_Mod=in                  |
| <b>Projects Due This Month</b> | project:plannedCompletionDate=\$\$TODAYbm&project:<br>plannedCompletionDate_Range=\$\$TODAYem&project:status=<br>IDA,REQ,PLN,CUR&project:status_Mod=in     |
| <b>Projects Due Next Month</b> | project:plannedCompletionDate=\$\$TODAYb+1m&project:<br>plannedCompletionDate_Range=\$\$TODAYe+1m&project:status=<br>IDA,REQ,PLN,CUR&project:status_Mod=in |

## CHAPTER 3

# ADVANCED VIEWS

## OBJECTIVES

After completing this chapter, you will be able to:

- **Understand the basic structure of text views**
- **Create shared columns**
- **Reference related objects**
- **Utilize calculated custom data**
- **Add calculated columns**



# Anatomy of a Text View

Views can also use text mode to create columns not otherwise available through the builder interface.

It is strongly recommended building as much of the view in the builder interface as possible, then convert the view to text to edit the view's column's definitions. Even the most experienced Workfront consultants and developers do not build a view directly in the text mode interface.

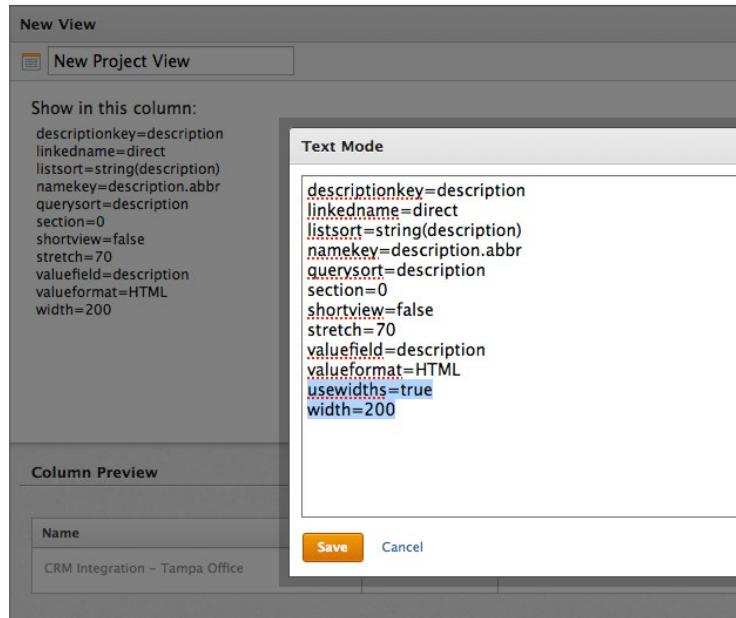
| Columns (View)                                                                                                                                                                                                                                                                           | Groupings | Filters | Chart |  |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------|---------|-------|--|
| Show in this column:                                                                                                                                                                                                                                                                     |           |         |       |  |
| <pre>descriptionkey=plannedcompletiondate isInlineEditable=false linkedname=direct listsort=atDateAsAtDate(plannedCompletionDate) namekey=due.on querysort=plannedCompletionDate section=0 shortview=false stretch=0 valuefield=plannedCompletionDate valueformat=atDate width=150</pre> |           |         |       |  |
| <a href="#">Done</a> <a href="#">Cancel</a>                                                                                                                                                                                                                                              |           |         |       |  |



# Adjusting Column Widths

The browser automatically adjusts column widths on lists based on content. Widths are manually set by going to any column in text mode and adding the text usewidths=true. You only need to enter usewidths=true in one column. This allows you to adjust the width of all columns that follow.

The defined widths on each column will then be honored. But be aware, if you go this route, you need to define the width on each column. If columns are defined outside of the available 1500 pixels they do not render. However, a scroll bar appears to assist with large columns.



# Essential Components of a Column View

| SAMPLE LINE                                              | DESCRIPTION                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>descriptionkey=<br/>- or -<br/>description=</code> | <p>This line defines the text of a tool tip as you hover over the name of the column. In this case it is using a key to translate the name value in the description text. If you want to modify the description, change this line to read:</p> <pre>description=Your Value</pre>                                                                                                                                                                                           |
| <code>namekey=<br/>- or -<br/>name=</code>               | <p>This line defines the column label. In this case it is using the abbreviated value based on the key. If you want to modify the column name, change this value to:</p> <pre>name=Your Value<br/>- or -<br/>Add the following line, which suspends the namekey:<br/>displayname=Your Value</pre> <p>'name=' allows you to enter whatever text you want for the column name, while 'namekey=' requires you enter a key that is used to translate the name of a column.</p> |
| <code>querysort=</code>                                  | <p>This line defines how the results are sorted when the column header is clicked. If it is not present then the column cannot be sorted after the report is run.</p>                                                                                                                                                                                                                                                                                                      |
| <code>valuefield=</code>                                 | <p>This line represents the text displayed in the results under the column header. The attribute input for the valuefield is the same used in a filter statement.</p>                                                                                                                                                                                                                                                                                                      |
| <code>valueformat=</code>                                | <p>This line represents the format used to display the text, number, or date.</p>                                                                                                                                                                                                                                                                                                                                                                                          |
| <code>name=</code>                                       | <p>This is the field pulled from the database and displayed in the results of a search list or report.</p>                                                                                                                                                                                                                                                                                                                                                                 |

# Using Custom Reporting Elements

**SCENARIO** — Incorporate the Parent Task x 4 view on a task report.

1. Go to the Custom Report Elements section of the Workfront Community Website (<https://community.attask.com/custom-report-elements>).
2. Select the View Element type and the Task Object type.
3. Find the Parent Task x 4 view. Open the view.
4. Select and copy the view definition listed below the Text Mode Code label.
5. Go to the Workfront view menu in the report header on the My Tasks report.
6. Select New View and call the view Parent Task x 4.
7. In the preview area, click the name of the first column.
8. Switch to text mode and delete the current text in the box.
9. Paste the text from your clipboard into the text area.
10. Click Done to save the text mode code.
11. Click Save View button when done.

| Name                                              | Object Type             |
|---------------------------------------------------|-------------------------|
| Creative Brief                                    | Design Phase            |
| Research Competitive Web Sites                    | Design Phase            |
| Market Research on indoor soccer balls.           | Design Phase            |
| Outline Web Site Features                         | Design Phase            |
| Goals Outline                                     | Design Phase            |
| Design Color Scheme                               | Design Phase            |
| Preliminary Layout                                | Design Phase            |
| Final Proposal                                    | Design Phase            |
| <b>Programming &amp; Execution</b>                |                         |
| We need to complete the back end of this project. | Programming & Execution |
| Build Front End Design                            | Programming & Execution |
| Set Up Online Payment & SSL                       | Programming & Execution |

**Text Mode Code**

```

column.0.descriptionkey=name
column.0.link.linkproperty.0.name=ID
column.0.link.linkproperty.0.valuefield=ID
column.0.link.valueformat=int
column.0.link.lookuplink.view
column.0.link.valuefield=objCode
column.0.link.valueformat=val
column.0.link.isdirect
column.0.listsort=string(name)
column.0.namekey=name_obj
column.0.querysort=name
column.0.shortview=false
column.0.valuefield=name
column.0.valueformat=HTML
column.0.width=150
column.1.descriptionkey=parent
column.1.link.linkproperty.0.name=ID
column.1.link.linkproperty.0.valuefield=parent:ID
column.1.link.valueformat=int
column.1.link.lookuplink.view
column.1.link.parentobj=objCode
column.1.link.valueformat=val
column.1.linkedname=parent
column.1.listsort=nested(parent).string(name)
column.1.namekey=parent
column.1.querysort=parent:name
column.1.shortview=false
column.1.stretch=0
column.1.valuefield=parent:name
column.1.valueformat=HTML
column.2.descriptionkey=Parent_Parent
column.2.link.linkproperty.0.name=ID
column.2.link.linkproperty.0.valuefield=parent:parent:parent:ID
column.2.link.valueformat=int
column.2.link.lookuplink.view
column.2.link.valuefield=parent:parent:objCode
column.2.link.valueformat=val

```

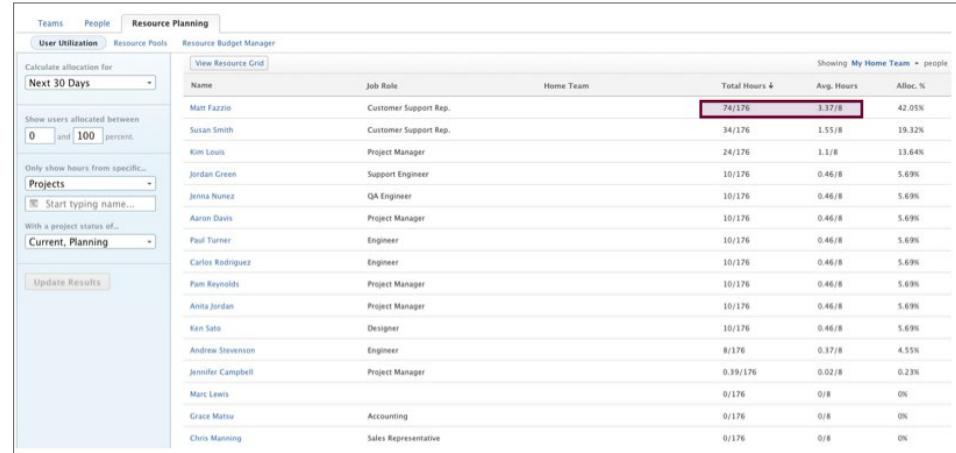
## CHAPTER 3: ADVANCED VIEWS

# Shared Columns



The shared column is a view attribute that allows for the combination of multiple data points under the same column header.

This section shows examples of how this is used in views provided in Workfront and demonstrates how to build shared column views.



The screenshot shows the 'Resource Planning' tab in the Workfront interface. On the left, there are several filter and search controls: 'Calculate allocation for' set to 'Next 30 Days', 'Show users allocated between 0 and 100 percent', 'Only show hours from specific Projects' (with 'Projects' selected), 'With a project status of...' (with 'Current, Planning' selected), and a 'View Resource Grid' button. The main area is a grid titled 'View Resource Grid' with the following columns: Name, Job Role, Home Team, Total Hours & Avg. Hours, and Alloc. %. The grid lists 18 employees with their respective roles, teams, and allocation details. The 'Alloc. %' column for Matt Fazzio is highlighted with a red border.

| Name              | Job Role              | Home Team | Total Hours & | Avg. Hours | Alloc. % |
|-------------------|-----------------------|-----------|---------------|------------|----------|
| Matt Fazzio       | Customer Support Rep. |           | 74/176        | 3.37/8     | 42.05%   |
| Susan Smith       | Customer Support Rep. |           | 34/176        | 1.55/8     | 19.32%   |
| Kim Louis         | Project Manager       |           | 24/176        | 1.1/8      | 13.64%   |
| Jordan Green      | Support Engineer      |           | 10/176        | 0.46/8     | 5.69%    |
| Jenna Nunez       | QA Engineer           |           | 10/176        | 0.46/8     | 5.69%    |
| Aaron Davis       | Project Manager       |           | 10/176        | 0.46/8     | 5.69%    |
| Paul Turner       | Engineer              |           | 10/176        | 0.46/8     | 5.69%    |
| Carlos Rodriguez  | Engineer              |           | 10/176        | 0.46/8     | 5.69%    |
| Pam Reynolds      | Project Manager       |           | 10/176        | 0.46/8     | 5.69%    |
| Anita Jordan      | Project Manager       |           | 10/176        | 0.46/8     | 5.69%    |
| Ken Sato          | Designer              |           | 10/176        | 0.46/8     | 5.69%    |
| Andrew Stevenson  | Engineer              |           | 8/176         | 0.37/8     | 4.55%    |
| Jennifer Campbell | Project Manager       |           | 0.39/176      | 0.02/8     | 0.23%    |
| Marc Lewis        |                       |           | 0/176         | 0/8        | 0%       |
| Grace Matsu       | Accounting            |           | 0/176         | 0/8        | 0%       |
| Chris Manning     | Sales Representative  |           | 0/176         | 0/8        | 0%       |

# Shared Columns (continued)

**SCENARIO** — Create a view that shows the name of the project and the owner in the same column. The information should display stacked.

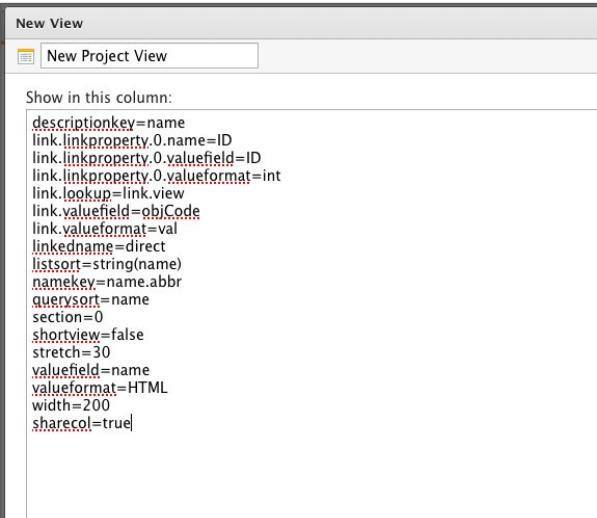
1. Navigate to the Projects area and locate the view drop-down menu.
2. Click New View.
3. In the view builder create a new column. Without selecting any value for the column, drag it and place it between the Name and the Owner columns. This placeholder column is used to stack the values.
4. Switch to text mode in this blank column and insert the following lines of code:

```
value=

valueformat=HTML
width=1
sharecol=true
```

This code shares our blank column with the Owner column to the right. It also inserts a <br> code as a line break for stacking.

5. Click Done.
6. Next, select the Name column and then Switch to text mode.
7. At the bottom of the text mode code, enter sharecol=true.



| Name                                                   | Desc                                                                                                                                  | Start On | Due On | % Complete |
|--------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------|----------|--------|------------|
| Marketing Team Merger<br>Jennifer Campbell             | As we merge the Creative and Graphic Design teams, we need to expand our office space and handle the influx of new employees.         | 6/21/    | 7/4/1  | 0%         |
| Grid Indoor Ball Launch<br>Jennifer Campbell           | Goal Sports is poised to take the paper industry by storm and ride in on the wave of green technologies with our new eco-paper.       | 5/21/    | 7/14/  | 17%        |
| Product Launch Template - Project<br>Jennifer Campbell |                                                                                                                                       | 2/19/    | 4/16/  | 0%         |
| Localization<br>Jennifer Campbell                      | This project illustrates a typical scenario for contract work. In this example, we're building a TV commercial for an outside client. | 6/4/1    | 6/9/1  | 70.2%      |
| FIFA World Cup Brazil Event                            | We need to represent Goal Sports, Inc., at the next World Cup to build brand for Cruzer product line.                                 | 6/17/    | 7/6/1  | 22.7%      |



## Shared Columns (continued)

**SCENARIO** — Create a task report with a shared column that includes the task name and assigned to name.

1. Navigate to the Reporting area and create a Task Report.
2. Change the Assignments column to Assigned To Name.
3. Insert a blank column, and drag it in between the Task Name and Assigned To Name columns.
4. Convert the blank column to text mode and insert the following lines:

```
value=

valueformat=HTML
width=1
sharecol=true
```

5. Click the Task Name column and Switch to text mode. Insert a new line with the following:

```
sharecol=true
```

6. Save and Close.



# Cross-Object References

Cross-Object References allow for the expanding of views through text mode.

One question that may be presented is, “Why can’t we find all related fields mapped out in the View Builder?” This is because the builder contains most of the primary relationships, but if everything was mapped, the builder would be cluttered. This is where the text mode interface comes in to play.

Remember, the API Explorer is a great tool for referencing and locating related fields. It contains a list of fields for each object and the relationships between them, which can be used when inputting the view fields. For example, the fields section of the API Explorer contains object tables of all listed fields and their associated camel case values. You can then use the References tab on any of these tables to refer to fields in other object types to expand what you see in your report. Keep in mind that when referencing other object types in Workfront, you can reference only one other table on a filter. Views and groupings allow you to reference more object tables, but depending on the table you start from, you are limited as to the number of references you can make.

When referencing fields from other tables, you should create the view in the standard builder interface first to get as close as possible to the intended outcome and then make adjustments for your references. To do this, insert placeholder columns where you want to include data that cannot be accessed through the builder.

# Cross-Object References

**SCENARIO** — Create a Project Report that includes a column showing the name of the Project Owner's Manager.

1. In the Reporting area create a new Project report.
2. Add the Project Name column as a placeholder and Switch to text mode.
3. Navigate to the API Explorer site in a separate browser tab and search for the Project table. Click on the References tab to refer to other tables.
4. In text mode on your Project Name column, change the following lines from this:

```
valuefield=project:name
querysort=project:name
displayname=
```

To this:

```
valuefield=owner:manager:name
querysort=owner:manager:name
displayname=Project Owner's Manager
```

5. Save and close. Name the report 'Project Owner's Manager'.

Text Mode

```
valuefield=owner:manager:name
querysort=owner:manager:name
valueformat=HTML
displayname=Project Owner's Manager
linkedname=owner
namekey=view.relatedcolumn
namekeyargkey.0=owner
namekeyargkey.1=name
```

**Save**   **Cancel**

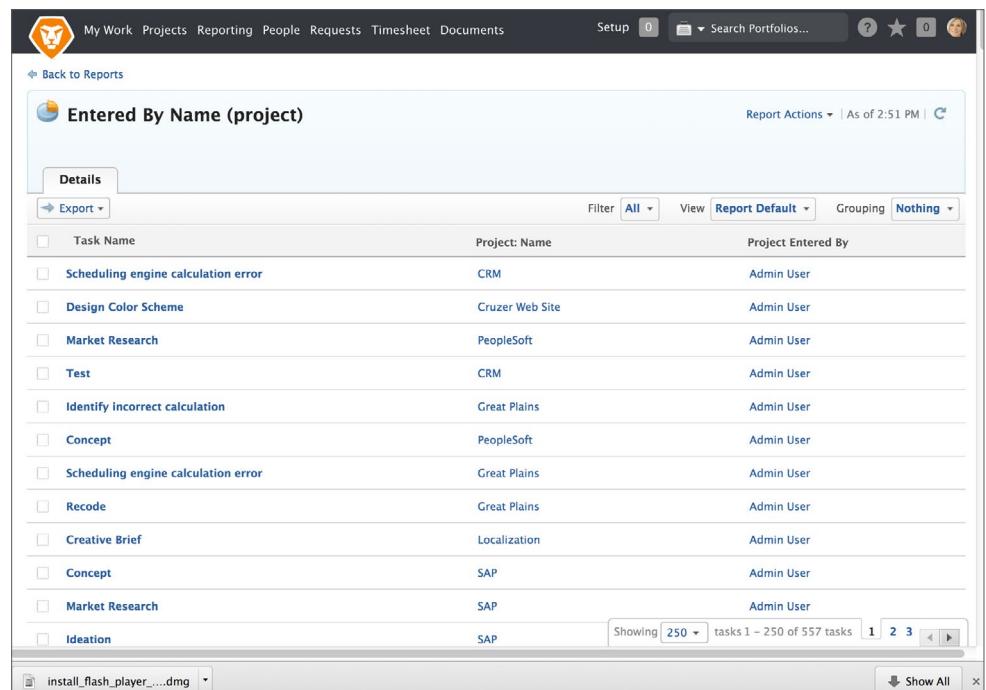
# Cross-Object References (continued)

From a task view, it is not currently possible to see who it was that entered the project the task was on. By going into text mode you can pull that information into the view through the bean relationships described in the API Explorer.

**SCENARIO** — Create a Task Report that displays the:

- Task: Name
- Project: Name
- Entered By: Name

1. Create a Task Report.
2. In the Column Preview area, click the second column tab.
3. Change the selection for column 2 to Project > Name.
4. Delete columns 4-8 and select column 3. Change the selection to Project > Name. This becomes a placeholder column.
5. Switch to text mode for column 3.



The screenshot shows a software interface with a navigation bar at the top. The main area is a report titled "Entered By Name (project)". The report has a "Details" tab selected. It lists various tasks along with their names and the names of the users who entered them. The columns are labeled "Task Name", "Project: Name", and "Project Entered By". The report includes a toolbar with "Export" options, filters, and grouping settings. At the bottom, there are pagination controls and a "Show All" link.

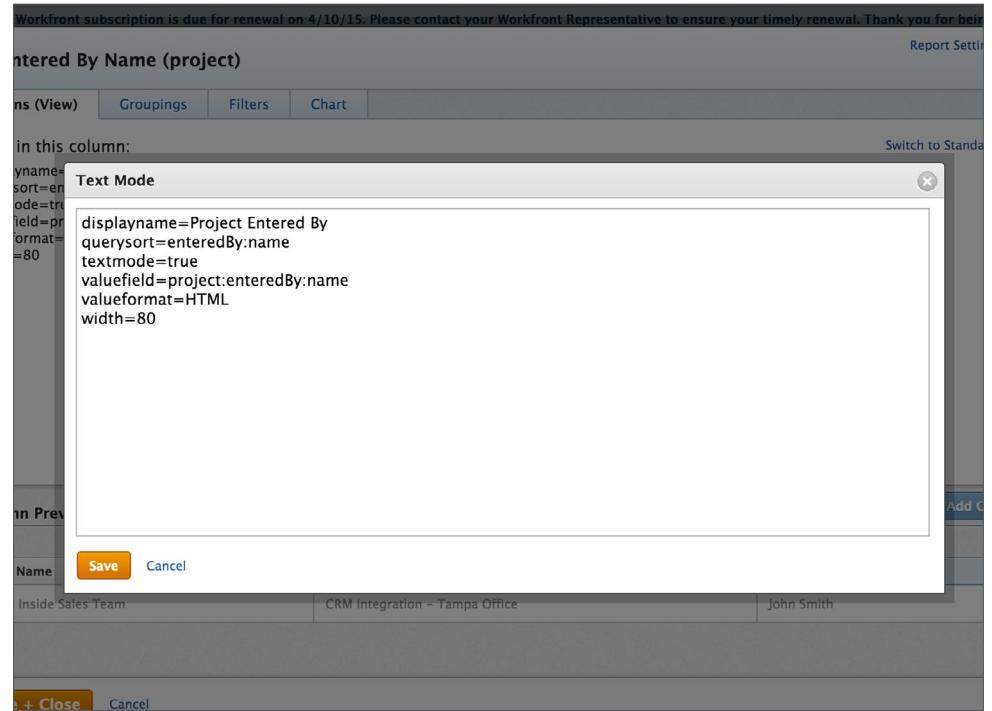
| Task Name                           | Project: Name   | Project Entered By |
|-------------------------------------|-----------------|--------------------|
| Scheduling engine calculation error | CRM             | Admin User         |
| Design Color Scheme                 | Cruzer Web Site | Admin User         |
| Market Research                     | PeopleSoft      | Admin User         |
| Test                                | CRM             | Admin User         |
| Identify incorrect calculation      | Great Plains    | Admin User         |
| Concept                             | PeopleSoft      | Admin User         |
| Scheduling engine calculation error | Great Plains    | Admin User         |
| Recode                              | Great Plains    | Admin User         |
| Creative Brief                      | Localization    | Admin User         |
| Concept                             | SAP             | Admin User         |
| Market Research                     | SAP             | Admin User         |
| Ideation                            | SAP             |                    |

# Cross-Object References (continued)

6. Change the following lines in the text mode to this:

```
displayname=Project Entered By
querysort=project:enteredBy:name
valuefield=project:enteredBy:name
```

7. Click the Done button to save the text mode code for the column. Click Save + Close. Name the report ‘Entered by Name (project).’





## Naming Columns

Instead of using the keys (name, description etc.), you can remove the word key from the line and type whatever you want following the equal sign. Alternatively, you may consider adding the displayname line, which is always used instead of the namekey.

When you change namekey= to name= the value provided is not translated. But if you include an additional line of displayname= the value translation remains in tact.

Use key or displayname, the value you provide is not translated into other Workfront-supported languages.



# Using Custom Data in a View

Custom Forms allow fields and information to be added to Workfront. A System Administrator can access and modify the custom field area by going to Setup > Custom Forms

## Create a New Custom Form

1. Go to the Setup area and click Custom Forms. Then click New Custom Form.
2. Select the object type where the custom form will be applied.
3. Select existing custom fields to add to your form, or create and add a new field from the form builder.
4. Click Save when finished

It is possible to include custom fields in reports. Custom Forms can also be referenced or included in filters and groupings.

When building a view, select a field specific to the object type to display. The Field Name displays all mapped fields in the search box for the selected Field Source. For example, use the field camel case `numberOfChildren` to identify a task as a parent task.

In addition to the built-in fields, the search box displays the available custom data fields that have been associated with the selected object.

# Calculated Custom Data

Workfront lets you take existing fields and plug them into another field. These are known as Calculated Field Expressions and are for creating numeric-based, date-based, and text-based formulas.

You may recognize many of the available options often used in spreadsheets. Some of these scenarios involve using CONCAT(). This can be used in both a value expression and a calculated custom data expression.

**SCENARIO** — You want to supplement the core fields provided in Workfront with a calculated field that displays the balance of work remaining based on the hours input on a project.

1. From the Setup area menu, click Custom Forms.
2. Select the project object type on the new Custom Form.
3. Add a new calculated field named Work Balance.
4. Double click on SUB in the Expressions box to begin your calculation.
5. Select Planned Hours from the Fields box to add it to the Calculation box. Make sure it appears before the comma.

The screenshot shows the 'New Custom Form' dialog box with the 'Field Settings' tab selected. A calculated field named 'Work Balance' is being configured. The 'Label (Required)' field contains 'Work Balance'. The 'Format' dropdown is set to 'Text'. In the 'Calculation' section, the formula 'SUB(Planned Hours,Actual Hours)/60' is entered. Below the calculation, under 'Additional settings', there is a list of functions: 'ROUND' and 'SORTASCNUM'.



## Calculated Custom Data (continued)

6. Add Actual Hours from the Fields box after the comma. Divide your formula by 60 to convert the result to hours. Remember, time in Workfront is stored in minutes. Your calculation should look like this:  
 $SUB(Planned Hours,Actual Hours)/60$
7. In Form Settings, name the form 'Project Info' and click the Save + Close button

New Custom Form

Add a Field Field Settings Form Settings

Label (Required)  
Work Balance

Instructions

Format  
Text

Additional settings

Calculation  
SUB(Planned Hours,Actual Hours)/60

SUB(number1, number2[,...])

Expressions (Double-click to add to calculation.)  
Type to filter items...  
ROUND  
SORTASCNUM

Work Balance  
12345

# Calculated Custom Data (continued)

**SCENARIO** — Modify the Work Balance field to use the CONCAT() function to display the word ‘Hours’ after the number that is produced.

1. Open the Project Info form and click on the Work Balance box in the preview area.
2. At the beginning of the calculation, input CONCAT().
3. At the end of the calculation, input ,” Hours”), which will append the word ‘Hours’ to the initial calculation and close the CONCAT() expression. Your calculation should look like this:

CONCAT((Planned Hours-Actual Hours)/60,” Hours”)



# Calculated Custom Data (continued)

## Attach a Custom Form to a Project

1. Navigate to a project and select the Project Details tab.
2. Select the second tab under Project Details to locate the custom form for the project.
3. Click Edit Custom Form, and attach the form you just created by selecting it from the drop-down menu.
4. Save your changes.

The screenshot shows the 'Edit Project' dialog box for an 'Untitled Project'. The 'Custom Form' tab is selected in the left sidebar. In the main area, there is a section titled 'Custom Form' with a dropdown menu labeled '-- Select Form --'. Below this is a 'Comment' section with a text input field and two small icons. At the bottom are 'Save Changes' and 'Cancel' buttons.

# Calculated Custom Data (continued)

**SCENARIO** — Create a new calculated field called Percent Remaining that will be displayed on your project custom form. Use the CONCAT() function to add the percent symbol to the field.

## Create the field and do the math

1. Go to the ‘Project Info’ Custom Form previously created.
2. Add a new calculated field and call it Percent Remaining.
3. Enter 100 followed by a minus sign in the Calculation box.
4. Add Percent Complete from the fields box. Your calculation should look like this:

100-Percent Complete

## Add the Percent Symbol to the Results

1. At the beginning of the calculation, input CONCAT(
2. At the end of the calculation, input "%"), which appends the percent symbol to the initial calculation and closes the CONCAT() expression. Your calculation should now look like this:

CONCAT(100-Percent Complete,"%")

3. Click the Save + Close button to submit the expression.

The screenshot shows the 'Edit Custom Form' window. In the 'Field Settings' tab, a new field is being created with the label 'Percent Remaining'. The 'Calculation' field contains the expression 'CONCAT(100-Percent Complete,%)'.

On the right side, the 'Project Info' panel shows the 'Work Balance' field with value '12345' and the newly created 'Percent Remaining' field with value '12345'.

The 'Expressions' section lists 'Date & Time', 'Mathematical', 'Text', and 'Other' categories. The 'Fields' section lists various project fields under the 'Project' category, with 'Percent Complete' selected.

# Cross-Object References in Calculated Custom Data

It is possible to create Calculated Fields using fields from other objects. Cross-Object references were established when setting filter fields. This specific relationship can be continued when setting up views using Cross-Objects in Calculated Custom Data views.

Cross-object references as part of custom data are slightly different from those used in the text mode interface for filters and views. Instead of using the bean reference (:), which is always in camel case, object aliases have been developed for the Custom Data interfaces.

Text mode attributes navigate between relationships using a colon; however, when making these connections in calculated fields, use a period. The examples show the connection to specific attributes from the task level.

Remember, it is not possible to reference data from child objects (e.g. subtasks). For example, task data cannot be pulled into a project-level expression because of the sub-level relationship. However, from the task perspective, project-level details can be included.

| OBJECT AND ATTRIBUTE | TEXT MODE REFERENCES (Filter/View) | CUSTOM DATA REFERENCES (Calculated Custom Data) |
|----------------------|------------------------------------|-------------------------------------------------|
| Project Name         | project:name                       | Project.Name                                    |
| Group Name           | group:name                         | Group.Name                                      |
| Company Name         | project:company:name               | Project.Company.Name                            |

# Calculated Custom Data on a Form

**SCENARIO** — Create a custom form field that takes the assigned hours on a task and shows what percent that is of the project as a whole.

1. Navigate to the Setup page and select Custom Forms.
2. Create a new Task Custom Form.
3. In the Add a Field area, select Calculated.
4. Give the field a name in the Label area. Call it ‘Percentage of Project Work’.
5. Change the format field to number.
6. Locate the Expressions Field and choose the Mathematical drop-down menu.
7. Select DIV. A formula appears in the Calculation area. This is the formula we will use to calculate the percentage of work this task represents in the project. Select the Task drop-down menu. Find and click Planned Hours.
8. Next, move the cursor next to the comma after Planned Hours and type in the word Project.Planned Hours (it is case sensitive).
9. Click the Form Settings tab and give the form a name. Call it ‘Additional Task Information’. Click the group and then Save and Close.

# Calculated Custom Data on a Form (continued)

## Calculating the Percentage on a Custom Form

1. Select the Calculated Custom Data field you just created.
2. To add the percentage to our calculation, add \*100. The formula appears similar to the following:

`DIV(Planned Hours,Project.Planned Hours)*100`

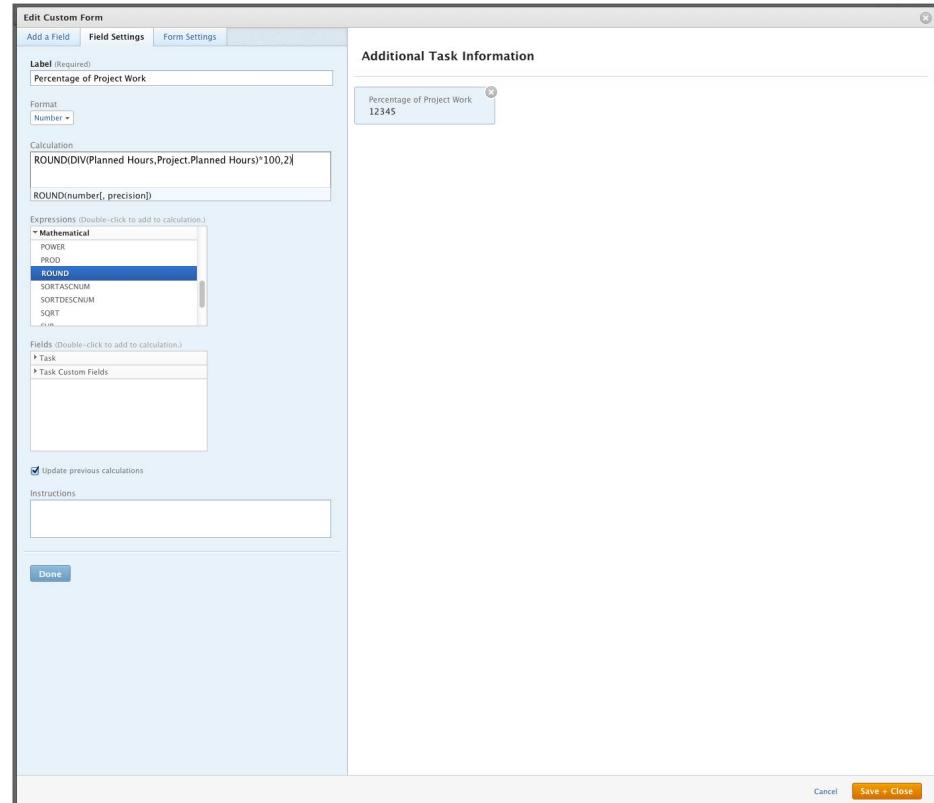
To make this calculation round to the nearest percent, we are going to wrap an additional formula around the one we just created.

3. Navigate back to the mathematical expressions drop-down menu and double click the ROUND expression. This inserts the formula at the end of the equation. Move it to the beginning.
4. Notice the format for the ROUND expression appears:

`ROUND(number[, precision])`

# Calculated Custom Data on a Form (continued)

5. The number section is calculated from the division and multiplication we are doing.
6. The precision determines how many decimal places the calculation uses.
7. Add a comma at the end of the formula and the number to represent the decimal places. Similar to the following:  
  
ROUND(DIV(Planned Hours,Project.Planned Hours)\*100,2)
8. Save and Close.



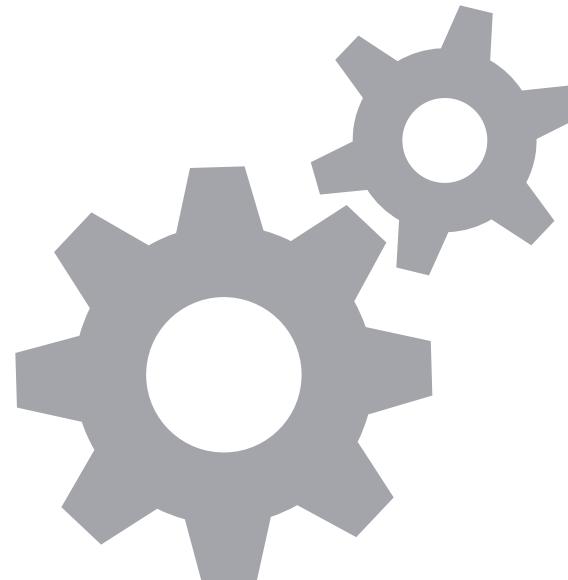


## PRACTICE EXERCISES

1. Write the Calculated Custom Data expressions for the following prompts:

- Hours Remaining for a Task
- Percent Remaining for a Task
- Variance of Days Between Planned Start Date and the Projected Completion Date for a Task
- Percentage of Task Planned Hours to Project Planned Hours
- Users' Employment Duration \*

\*Assume that you have created a custom data field on the user's profile called Hire Date and use the \$\$TODAY wildcard.





# Calculated Columns

The custom value is set by creating a placeholder, changing the valuefield to valueexpression, and then inserting the expression.

Formatting for the column is also set in the valueexpression statement. Using the expressions provided in earlier tables, statements can be made to set the column's format to the desired display. To ensure your format displays correctly, modify valueformat to = HTML. This ensures the greatest chance your format displays as intended.

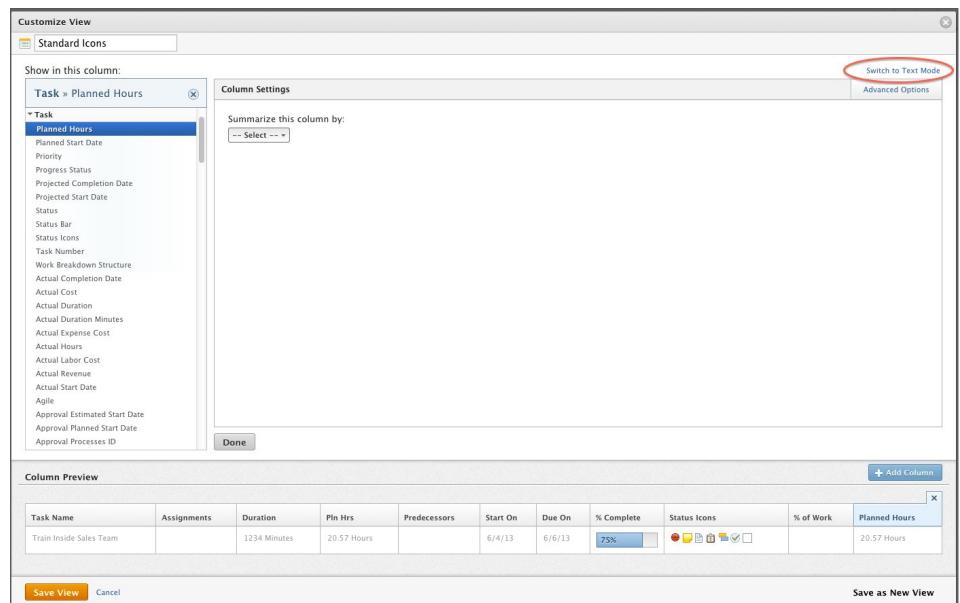
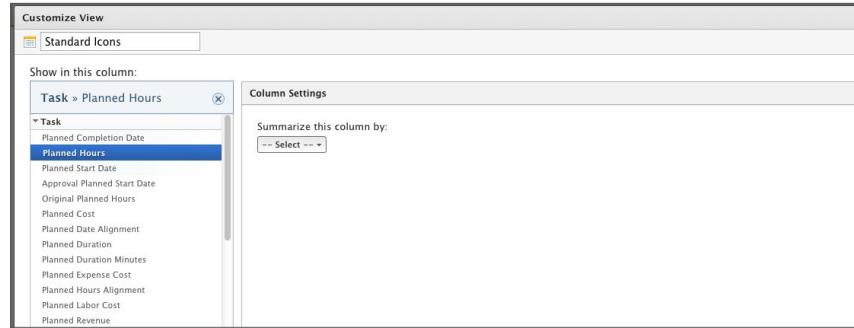
| CUSTOM DATA         | TEXT MODE FILTER     | CALCULATED VIEW        |
|---------------------|----------------------|------------------------|
| Planned Hours       | workRequired         | workRequired           |
| Actual Hours        | actualWorkRequired   | {actualWorkRequired}   |
| Project.Duration    | project:duration     | {project}.{duration}   |
| A Custom Data Field | DE:Custom Data Field | {DE:Custom Data Field} |

# Using Custom Expressions in a View

**SCENARIO** — We want to create a custom expression that automatically calculates the percentage of work on a project every time we apply the view.

## Create View

1. Navigate to a task list. Select the view drop-down menu and customize the view.
  2. Add a new column in the type ahead, enter Planned Hours.
  3. Select the Planned Hours column, and switch to text mode.
  4. Change valuefield to valueexpression. Insert the following formula in the value expression:
- valueexpression=DIV([workRequired],[project].[workRequired])
5. At the end of the formula multiply the equation by 100 so it becomes a percentage.



# Using Custom Expressions in a View (continued)

- To make this expression round to the nearest percent, we have to include the percentage formula. Insert ROUND in front of the DIV. Include the number of decimal places you want separated by a comma. In this case use:

```
ROUND(DIV({workRequired},{project}.{workRequired})*100,2)
```

- Next, change the valueformat to HTML, and the displayname to % of Work.
- To create a visual reminder that this view is a percentage value, add CONCAT to the value and a % sign.

```
valueexpression=CONCAT(ROUND(DIV({workRequired}, {project}.{workRequired})*100,2), "%")
```

- Click Save View.

## NOTE

Generally, you want to remove the querysort column. This line determines which field the column sorts on when you click the column header. It is suggested you remove this ability to reduce confusion for users accessing data.

Because you can only sort on stored values, you do not want to provide the option to sort on only one of the components making up the calculation. The results will be reorganized, but not necessarily according to what is being displayed.

**Customize View**

Standard Icons

Show in this column:

```
valueformat=HTML
querysort=workRequired
textmode=true
linkedname=direct
namekey=workrequired
valueexpression=CONCAT(ROUND(DIV({workRequired},{project}.{workRequired})*100,2),"")
viewalias=workrequired
displayname=% of Work
valuefield=workRequired
```

**ActTask** My Work Projects Reporting People Requests Timesheet Setup Help Search Projects... ○ ★ 🔍

**Grid Indoor Ball Launch**

Project Owner: Jennifer Campbell, Project Manager, Team Gear

Status: Current Condition: On Target Planned Completion: Jun 1, 2013 Percent Complete: 17%

| # | Task Name            | Assignments              | Duration | Pln Hrs  | Predecessors | Start On | Due On  | % Complete | Status Icons                                                                                                                                                                                                                  |       |
|---|----------------------|--------------------------|----------|----------|--------------|----------|---------|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------|
| 1 | Concept              | Andrew Stevenson         | 10 Days  | 45 Hours |              | 4/9/13   | 4/20/13 | 100%       | <span style="color: red;">●</span> <span style="color: yellow;">■</span> <span style="color: green;">□</span> <span style="color: blue;">□</span> <span style="color: orange;">□</span> <span style="color: purple;">□</span> | 4.42% |
| 2 | Market Research      | Grace Matsu, Susan Smith | 2 Weeks  | 20 Hours |              | 4/9/13   | 4/20/13 | 100%       | <span style="color: red;">●</span> <span style="color: yellow;">■</span> <span style="color: green;">□</span> <span style="color: blue;">□</span> <span style="color: orange;">□</span> <span style="color: purple;">□</span> | 1.96% |
| 3 | Idation              | Susan Smith              | 1 Week   | 15 Hours |              | 4/9/13   | 4/13/13 | 100%       | <span style="color: red;">●</span> <span style="color: yellow;">■</span> <span style="color: green;">□</span> <span style="color: blue;">□</span> <span style="color: orange;">□</span> <span style="color: purple;">□</span> | 1.47% |
| 4 | Proof of Concept     | Jenna Nunez              | 1 Week   | 10 Hours |              | 4/9/13   | 4/13/13 | 100%       | <span style="color: red;">●</span> <span style="color: yellow;">■</span> <span style="color: green;">□</span> <span style="color: blue;">□</span> <span style="color: orange;">□</span> <span style="color: purple;">□</span> | 0.98% |
| 5 | Design & Engineering | Project Manager          | 10 Days  | 33 Hours |              | 4/9/13   | 4/20/13 | 66.7%      | <span style="color: red;">●</span> <span style="color: yellow;">■</span> <span style="color: green;">□</span> <span style="color: blue;">□</span> <span style="color: orange;">□</span> <span style="color: purple;">□</span> | 3.24% |
| 6 | Conceptual Mockups   | Designer                 | 2 Weeks  | 10 Hours |              | 4/9/13   | 4/20/13 | 100%       | <span style="color: red;">●</span> <span style="color: yellow;">■</span> <span style="color: green;">□</span> <span style="color: blue;">□</span> <span style="color: orange;">□</span> <span style="color: purple;">□</span> | 0.98% |
| 7 | Watermark Design     | Designer                 | 1 Week   | 8 Hours  |              | 4/9/13   | 4/13/13 | 100%       | <span style="color: red;">●</span> <span style="color: yellow;">■</span> <span style="color: green;">□</span> <span style="color: blue;">□</span> <span style="color: orange;">□</span> <span style="color: purple;">□</span> | 0.79% |
| 8 | Engineering          | Engineer                 | 2 Weeks  | 10 Hours |              | 4/9/13   | 4/20/13 | 20%        | <span style="color: red;">●</span> <span style="color: yellow;">■</span> <span style="color: green;">□</span> <span style="color: blue;">□</span> <span style="color: orange;">□</span> <span style="color: purple;">□</span> | 0.98% |

Showing All tasks in the Standard Icons view, grouped by Nothing

Showing All tasks in the Standard Icons view, grouped by Nothing

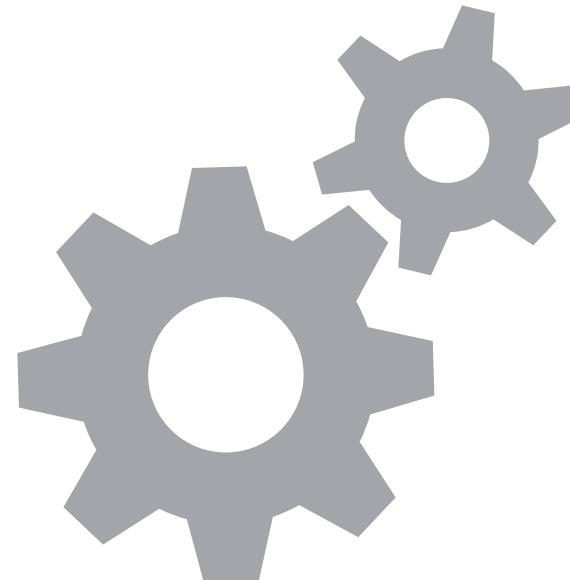


## PRACTICE EXERCISES

1. Practice writing the Calculated Column Value expressions for the following prompts:

- Hours Remaining for a Task
- Percent Remaining for a Task
- Task Planned Hours Percentage of Project Planned Hours
- Users' Employment Duration
- Users' Employment Duration multiplied by the Users' Hourly Rate
- Users' Employment Duration on a Task View

\*Assume that you have created a custom data field on the users' profiles called Hire Date and use the \$\$TODAY wildcard.





# Calculated Custom Data vs. Calculated Columns

When determining whether to make calculations based on calculated custom data or calculated columns, keep in mind that custom data can be used on filters, groupings, conditional formatting on a view, and the axes on a chart. Calculated columns cannot. Calculated columns, however, can solve a key deficiency in calculated custom data. The following material is a great reference for understanding the differences between the two.

## When to Use Calculated Columns

- When you need to see real time data needs on reports.
- When you don't plan to group by aggregated results.
- If you do not plan to aggregate the data beyond the initial view setup (data can only be aggregated once).

## When to use Calculated Custom Data

- To group the aggregated results.
- To further aggregate the data beyond the initial data setup.
- Okay with once-a-day updates.

## When do Calculated Custom Data updates occur?

- When a user edits the object
- On bulk edit with activated Recalculate Custom expressions
- Modifications to the form with selected 'Update previous calculations' option

Due to the limitations described earlier, it is strongly recommended that Calculated Custom Data does not transcend objects. Calculations should only reference other fields on the custom form.

## Calculated Custom Data is necessary in the following cases

- Calculated custom data is needed on filters. The system only allows you to filter on stored values.
- Calculated custom data is always needed for conditional formatting on a view.
- Calculated data is needed for groupings and axes on aggregate charts.

Keep in mind that calculated custom data has the potential to become stale if certain conditions are present. For example, native fields like Work Required or Actual Work Required can be modified without editing a task directly. If a custom data calculation using these fields has been made on the task itself, the calculation will not update as changes are made to those fields. Additionally, native fields that are tied to calculated custom data may not update instantaneously as custom fields are not updated nightly. Calculated views are always fresh, because the calculation is made when the report is run or when the view is applied.

# Calculated Expression Operators

| DATE & TIME EXPRESSION | EXAMPLE                       | DESCRIPTION                                                                                                         |
|------------------------|-------------------------------|---------------------------------------------------------------------------------------------------------------------|
| <b>ADDDAYS</b>         | ADDDAYS(date, number)         | Adds the number of days to the date.                                                                                |
| <b>ADDMONTHS</b>       | ADDMONTHS(date, number)       | Adds the number of months to the date.                                                                              |
| <b>ADDYEARS</b>        | ADDYEARS(date, number)        | Add the number of years to the date.                                                                                |
| <b>CLEARTIME</b>       | CLEARTIME(date)               | Clears the time portion of a date.                                                                                  |
| <b>DATE</b>            | DATE(string)                  | Converts a string to a date.                                                                                        |
| <b>DMAX</b>            | DMAX(date1, date2[,...])      | Returns the latest date in the list.                                                                                |
| <b>DMIN</b>            | DMIN(date1, date2[,...])      | Returns the earliest date in the list.                                                                              |
| <b>DATEDIFF</b>        | DATEDIFF(date1, date2)        | Returns the number of days between two dates.                                                                       |
| <b>WEEKDAYDIFF</b>     | WEEKDAYDIFF(date1, date2)     | Returns the number of weekdays between two dates.                                                                   |
| <b>WORKMINUTESDIFF</b> | WORKMINUTESDIFF(date1, date2) | Returns the number of scheduled minutes between the dates according to the default schedule.                        |
| <b>DAYSINYEAR</b>      | DAYSINYEAR(date)              | Returns the total days in the year of a given date as a number.                                                     |
| <b>DAYSINMONTH</b>     | DAYSINMONTH(date)             | Returns the total days in months of the given date as a number.                                                     |
| <b>DAYSINSPLITWEEK</b> | DAYSINSPLITWEEK               | Returns the total weekdays between the date and the end of the week or the end of the month, whichever comes first. |
| <b>YEAR</b>            | YEAR(date)                    | Returns the year of the given date as a number.                                                                     |



# Calculated Expression Operators (continued)

| DATE & TIME EXPRESSION | EXAMPLE         | DESCRIPTION                                                                                                  |
|------------------------|-----------------|--------------------------------------------------------------------------------------------------------------|
| <b>MONTH</b>           | MONTH(date)     | Returns the month of the given date as a number.                                                             |
| <b>DAYOFMONTH</b>      | DAYOFMONTH      | Returns the day of the month for the given date as a number.<br>The first day of the month has a value of 1. |
| <b>DAYOFWEEK</b>       | DAYOFWEEK(date) | Returns the day of the week for the given date as a number between 1 (Sunday) and 7 (Saturday).              |
| <b>HOUR</b>            | HOUR(date)      | Returns the hour of the given date as a number between 0 and 23.                                             |
| <b>MINUTE</b>          | MINUTE(date)    | Returns the minute of the given date as a number.                                                            |
| <b>SECOND</b>          | SECOND(date)    | Returns the second of the given date as a number.                                                            |



# Calculated Expression Operators (continued)

| MATHEMATICAL EXPRESSION | EXAMPLE                           | DESCRIPTION                                                      |
|-------------------------|-----------------------------------|------------------------------------------------------------------|
| <b>ABS</b>              | ABS(number)                       | Returns the absolute value of the number.                        |
| <b>AVERAGE</b>          | AVERAGE(number 1, number 2[,...]) | Returns the average of the numbers.                              |
| <b>CEIL</b>             | CEIL(number)                      | Rounds a number up to the nearest integer.                       |
| <b>DIV</b>              | DIV(number 1, number 2[, ...])    | Divides all the numbers in the order provided.                   |
| <b>FLOOR</b>            | FLOOR(number)                     | Rounds a number down to the nearest integer.                     |
| <b>LN</b>               | LN(number)                        | Returns the natural logarithm value of the number.               |
| <b>LOG</b>              | LOG(number 1, number 2[, ...])    | Returns the logarithm value of number 2 to the base of number 1. |
| <b>MAX</b>              | MAX(number 1, number 2[, ...])    | Returns the largest number provided in this list.                |
| <b>MIN</b>              | MIN(number 1, number 2[, ...])    | Returns the smallest number provided in this list.               |
| <b>NUMBER</b>           | NUMBER(string)                    | Converts a string to a number.                                   |
| <b>POWER</b>            | POWER( number, power)             | Returns a number raised to a power.                              |
| <b>PROD</b>             | PROD(number 1, number 2[, ...])   | Multiplies all the numbers provided in the list.                 |
| <b>ROUND</b>            | ROUND(number[, precision])        | Rounds the number up to the specified decimals of the precision. |
| <b>SQRT</b>             | SQRT(number)                      | Returns the square root of the number.                           |
| <b>SUB</b>              | SUB(number 1, number 2[, ...])    | Subtracts all the numbers in the order provided.                 |
| <b>SUM</b>              | SUM(number 1, number 2[, ...])    | Add all the numbers provided in the list.                        |



# Calculated Expression Operators (continued)

| TEXT EXPRESSION  | EXAMPLE                             | DESCRIPTION                                                                                                                                        |
|------------------|-------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>CONCAT</b>    | CONCAT(string 1, string 2[, ...])   | Concatonates the strings.                                                                                                                          |
| <b>LEFT</b>      | LEFT(string, length)                | Returns a specified number of characters from the left side of the string.                                                                         |
| <b>LEN</b>       | LEN(string)                         | Returns the length of the string.                                                                                                                  |
| <b>LOWER</b>     | LOWER(string)                       | Returns the string in lower case.                                                                                                                  |
| <b>REPLACE</b>   | REPLACE(string 1, string 2[, ...])  | Replaces all occurrences of string2 with string3 in string1.                                                                                       |
| <b>RIGHT</b>     | RIGHT(string, lenght)               | Returns a specified number of characters from the right side of the string.                                                                        |
| <b>STRING</b>    | STRING(number[, precision])         | Converts a number to a string with the specified decimals of precision.                                                                            |
| <b>SUBSTR</b>    | SUBSTR(string,start[, end])         | Returns characters of a string based on the start and end index specified.                                                                         |
| <b>CONTAINS</b>  | CONTAINS(findText, withinText)      | Returns true if the findText string is found within the withinText string.                                                                         |
| <b>SEARCH</b>    | SEARCH(findText, withinText[start]) | Returns the index of the first occurrence of the findText in the string withinText, starting at the given position or -1 if the text is not found. |
| <b>UPPER</b>     | UPPER(string)                       | Returns the string in upper case.                                                                                                                  |
| <b>ENCODEURL</b> | ENCODEURL(string)                   | Extracts any special characters in the string so they can be included in a URL argument.                                                           |
| <b>TRIM</b>      | TRIM(string)                        | Removes whitespace from the beginning and end of a string.                                                                                         |

# Calculated Expression Operators (continued)



| OTHER EXPRESSIONS | EXAMPLE                                                           | DESCRIPTION                                                                                                                                           |
|-------------------|-------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>IF</b>         | IF(condition, trueExpression, falseExpression)                    | Evaluates the condition parameter and returns the trueExpression if it is true, or the falseExpression if it is false.                                |
| <b>CASE</b>       | CASE(indexNumber, value1, [value2, ...])                          | Chooses a value from a list, based on an index number.                                                                                                |
| <b>ISBLANK</b>    | ISBLANK(value)                                                    | Returns true if the value is null or empty, false if the value is otherwise.                                                                          |
| <b>IN</b>         | IN(value, value1[, value2 ...])                                   | Returns true if the value equals one of the provided value1, value2 .... otherwise it returns false.                                                  |
| <b>IFIN</b>       | IFIN(value, value1[, value2...], trueExpression, falseExpression) | If the value equals one of the value1, value2... then returns the trueExpression, otherwise returns falseExpression. Must have at least 4 parameters. |

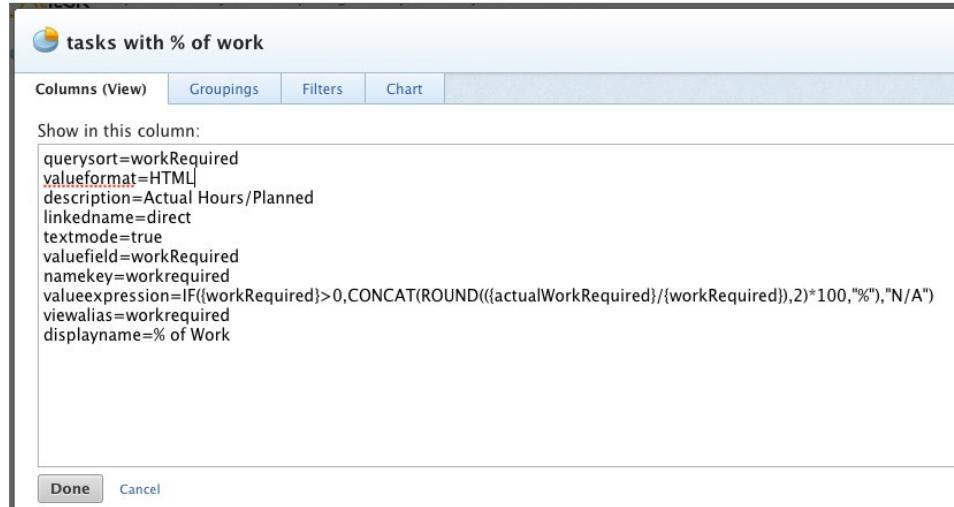


# Calculated Custom Data

**SCENARIO** — Create a Task Report that displays the following:

- Task: Name
- Task: Actual Hours
- Task: Planned Hours
- % of Work (this will be a calculated column that divides Actual Hours by Planned Hours)

1. Create a new Task report.
2. In the Column Preview area, click the second column tab.
3. Leave the first column as Task > Name.
4. Change column 2 to Task > Actual Hours and column 3 to Task > Planned Hours.
5. Change the selection for column 4 to Task > Planned Hours.  
This becomes your placeholder column.
6. Delete columns 5 thru 8.





## Calculated Custom Data (continued)

7. Select column 4 and then Switch to text mode.
8. Replace all the text mode code in column 4 with the following:

```
description=Actual Hours / Planned
valueexpression=IF({workRequired}>0,CONCAT(ROUND(
({actualWorkRequired}/{workRequired}),2)*100, "%"),"N/A")
displayname=% of Work
```

9. Click the Done button to save the text mode code for the column.

Click the Save + Close button. Name the report 'Tasks with % of Work'.

**tasks with % of work**

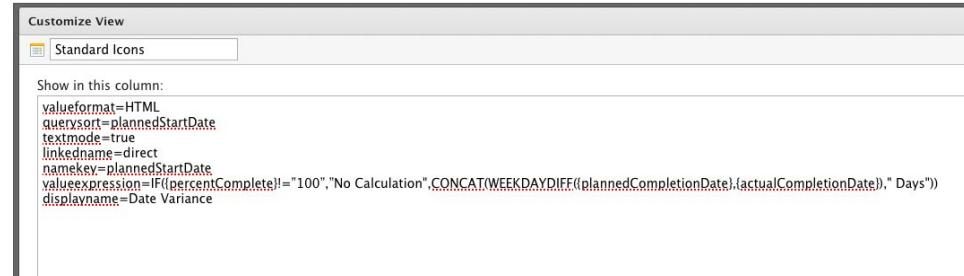
Report Actions ▾ | As of 10:39 AM |

| Task Name                           | Actual Hours | Planned Hours | % of Work |
|-------------------------------------|--------------|---------------|-----------|
| Scheduling engine calculation error | 3 Hours      | 7 Hours       | 43%       |
| Test                                | 0 Hours      | 1 Hour        | 0%        |
| Identify incorrect calculation      | 0 Hours      | 2 Hours       | 0%        |
| Market Research                     | 41 Hours     | 20 Hours      | 205%      |
| Design Color Scheme                 | 0 Hours      | 4 Hours       | 0%        |
| Concept                             | 62 Hours     | 45 Hours      | 138%      |
| Scheduling engine calculation error | 3 Hours      | 7 Hours       | 43%       |
| Recode                              | 0 Hours      | 3 Hours       | 0%        |
| Creative Brief                      | 0 Hours      | 6 Hours       | 0%        |
| Concept                             | 62 Hours     | 45 Hours      | 138%      |

# Calculated Custom Data (continued)

**SCENARIO** — Create a View with a Calculated Column for tasks that display the following columns:

- Name
- Planned Completion Date
- Actual Completion Date
- Date Variance



Use the WEEKDAYDIFF operator. Include an IF condition to display 'No Calculation' if the task is not complete.

1. Create a Task Report.
2. Leave the first column as Task > Name.
3. Click the second column tab in the Column Preview area.
4. Change the selection for column 2 to Task > Planned Completion Date.
5. Change the selection for column 3 to Task > Actual Completion Date.

# Calculated Custom Data (continued)

6. Change the selection for column 4 to Task > Planned Completion Date. This becomes your placeholder column.
7. Delete columns 5 through 8.
8. Click on column 4 and then Switch to text mode.
9. Replace all the text mode code in column 4 with the following:

```
description=Planned – Actual Completion Dates
valueexpression=IF({percentComplete}!=100,"No
Calculation",CONCAT(WEEKDAYDIFF
({plannedCompletionDate},{actualCompletionDate})," Days"))
displayname=Date Variance
valueformat=HTML
```

10. Click the Done button to save the text mode code for the column.
11. Select the Save + Close button. Name the report ‘Tasks with Date Variance.’

**Grid Indoor Ball Launch**

The screenshot displays a project grid titled "Grid Indoor Ball Launch". At the top, there's a status bar showing "Project Owner: Jennifer Campbell, Project Manager, Team Gear", "Status: Current, Condition: On Target, Planned Completion: Jun 1, 2013, Percent Complete: 17%", and buttons for "Edit Project" and "Project Actions". Below the status bar is a navigation bar with tabs: "Tasks" (selected), "Project Details", "Updates", "Documents (1)", "Issues", "Risks", and "More". Under "Tasks", there are two sub-tabs: "Task List" (selected) and "Gantt Chart". Below the navigation is a toolbar with "New Task", "Export", and other options. The main area is a table with 19 rows, each representing a task. The columns include: Task Name, Assignments, Duration, Pln Hrs, Predecessors, Start On, Due On, % Complete, Status Icons, % of Work, and Date Variance. The tasks are categorized under sections like "Concept", "Market Research", "Ideation", etc. Each row also includes a small profile picture of the assigned team member.

| Task Name                          | Assignments              | Duration | Pln Hrs   | Predecessors | Start On | Due On  | % Complete | Status Icons | % of Work | Date Variance  |
|------------------------------------|--------------------------|----------|-----------|--------------|----------|---------|------------|--------------|-----------|----------------|
| 1 Concept                          | Andrew Stevenson         | 10 Days  | 45 Hours  |              | 4/9/13   | 4/20/13 | 100%       | Green        | 4.42%     | 0 Days         |
| 2 Market Research                  | Grace Matsu, Susan Smith | 2 Weeks  | 20 Hours  |              | 4/9/13   | 4/20/13 | 100%       | Green        | 1.96%     | 0 Days         |
| 3 Ideation                         | Susan Smith              | 1 Week   | 15 Hours  |              | 4/9/13   | 4/13/13 | 100%       | Green        | 1.47%     | 0 Days         |
| 4 Proof of Concept                 | Jenna Nunez              | 1 Week   | 10 Hours  |              | 4/9/13   | 4/13/13 | 100%       | Green        | 0.98%     | 0 Days         |
| 5 Design & Engineering             | Project Manager          | 10 Days  | 33 Hours  |              | 4/9/13   | 4/20/13 | 66.7%      | Yellow       | 3.24%     | No Calculation |
| 6 Conceptual Mockups               | Designer                 | 2 Weeks  | 10 Hours  |              | 4/9/13   | 4/20/13 | 100%       | Green        | 0.98%     | 0 Days         |
| 7 Watermark Design                 | Designer                 | 1 Week   | 8 Hours   |              | 4/9/13   | 4/13/13 | 100%       | Green        | 0.79%     | 19 Days        |
| 8 Engineering                      | Engineer                 | 2 Weeks  | 10 Hours  |              | 4/9/13   | 4/20/13 | 26%        | Yellow       | 0.98%     | No Calculation |
| 9 Production Planning              | Jennifer Campbell        | 1 Week   | 5 Hours   |              | 4/9/13   | 4/13/13 | 60%        | Yellow       | 0.49%     | No Calculation |
| 10 Legal                           | Grace Matsu              | 20 Days  | 20 Hours  |              | 4/9/13   | 5/4/13  | 0%         | Red          | 1.96%     | No Calculation |
| 11 Patents, Trademarks, CopyRights | Joe Stevens              | 1 Month  | 20 Hours  |              | 4/9/13   | 5/1/13  | 0%         | Red          | 1.96%     | No Calculation |
| 12 Manufacturing                   | Project Manager          | 40 Days  | 820 Hours |              | 4/9/13   | 6/1/13  | 0%         | Red          | 80.55%    | No Calculation |
| 13 Sourcing                        | Jennifer Campbell        | 2 Months | 5 Hours   |              | 4/9/13   | 6/1/13  | 0%         | Red          | 0.49%     | No Calculation |
| 14 Procurement                     | Chris Manning            | 3 Weeks  | 15 Hours  |              | 4/9/13   | 4/27/13 | 0%         | Red          | 1.47%     | No Calculation |
| 15 Production Models               | Jenna Nunez              | 1 Month  | 160 Hours |              | 4/9/13   | 5/4/13  | 0%         | Red          | 15.72%    | No Calculation |
| 16 Production                      | Kim Louis, Susan Smith   | 2 Months | 640 Hours |              | 4/9/13   | 6/1/13  | 0%         | Red          | 62.87%    | No Calculation |
| 17 Marketing                       | Chris Manning            | 20 Days  | 80 Hours  |              | 4/9/13   | 5/4/13  | 0%         | Red          | 7.86%     | No Calculation |
| 18 Advertising & Promotions        | Matt Fazio               | 1 Month  | 80 Hours  |              | 4/9/13   | 5/4/13  | 0%         | Red          | 7.86%     | No Calculation |
| 19 Distribution                    | Jack Oliver              | 10 Days  | 20 Hours  |              | 4/9/13   | 4/20/13 | 0%         | Red          | 1.96%     | No Calculation |



## Calculated Aggregates

Calculated Aggregates work very much like Calculated Views and are likely to be part of any text mode calculated view that results in a numeric output.

Apply aggregate custom expressions the same way you do the sum aggregate for the Planned Hours field, so they accurately appear in the grouping bar.

For the results shown in the image, expect the Work Balance column to show the sum of all the values in the column based on the groupings (3157 hours), not the original Planned Hours values (3371 hours).

# Calculated Aggregates (continued)

The following shows the text mode code for the view. Notice the aggregator section in bold.

```

valueformat=compound
aggregator.displayformat=minutesAsHoursString
aggregator.function=SUM
aggregator.valueformat=val
aggregator.valuefield=workRequired
aggregator.namekey=workrequired
linkedname=direct
textmode=true
valuefield=workRequired
namekey=workrequired
valueexpression=CONCAT(ROUND({{workRequired}-
{actualWorkRequired}}/60,2)," Hours")
viewalias=workrequired
displayname=Work Balance

```

In order to get the aggregated value in the grouping to display the aggregated difference between the Planned Hours and Actual Hours fields, input the same equation into the aggregator.valuefield line.

| Name                                     | Planned Hours | Actual Hours | Work Balance |
|------------------------------------------|---------------|--------------|--------------|
| Company: Goal Sports, Inc. Corporate (5) | 3371 Hours    | 12840        | 3157         |
| Grid Indoor Ball Launch                  | 938 Hours     | 95 Hours     | 843 Hours    |
| Corporate Sales Initiative               | 1700 Hours    | 27 Hours     | 1673 Hours   |
| Cruzer Web Site                          | 589 Hours     | 91 Hours     | 498 Hours    |
| FIFA World Cup Brazil Event              | 109 Hours     | 0 Hours      | 109 Hours    |
| Localization                             | 35 Hours      | 1 Hour       | 34 Hours     |
| Company: No Value (5)                    | 2645.75 Hours | 840          | 2631.75      |
| Upgrade Servers                          | 728 Hours     | 0 Hours      | 728 Hours    |
| Integrate eCommerce into Website         | 718 Hours     | 3 Hours      | 715 Hours    |
| Marketing Team Merger                    | 240 Hours     | 8 Hours      | 232 Hours    |
| CRM                                      | 14.75 Hours   | 3 Hours      | 11.75 Hours  |
| Product Launch Project                   | 945 Hours     | 0 Hours      | 945 Hours    |

# Calculated Aggregates (continued)

The aggregator.displayformat used for the Planned Hours column converts minutes to hours. Because the Planned Hours field was used as a placeholder, this line doesn't need to be adjusted.

The minutesAsHoursString definition means there is no need to divide each field by 60 as done on the valueexpression for the results.

In this aggregator.valuefield=workRequired becomes: aggregator.valueexpression=ROUND(({workRequired}-{actualWorkRequired}),2)

Below is what you should have displayed in text mode:

```
valueformat=compound
aggregator.displayformat=minutesAsHoursString
aggregator.valueexpression=ROUND(({workRequired}-
{actualWorkRequired}),2)
aggregator.function=SUM
aggregator.valueformat=val
aggregator.namekey=workrequired
linkedname=direct
textmode=true
valuefield=workRequired
namekey=workrequired
valueexpression=CONCAT(ROUND(({workRequired}-
{actualWorkRequired})/60,2)," Hours")
viewalias=workrequired
displayname=Work Balance
```

| Name                                            | Planned Hours | Actual Hours | Work Balance |
|-------------------------------------------------|---------------|--------------|--------------|
| <b>Company: Goal Sports, Inc. Corporate (5)</b> |               |              |              |
| Grid Indoor Ball Launch                         | 938 Hours     | 95 Hours     | 843 Hours    |
| Corporate Sales Initiative                      | 1700 Hours    | 27 Hours     | 1673 Hours   |
| Cruzer Web Site                                 | 589 Hours     | 91 Hours     | 498 Hours    |
| FIFA World Cup Brazil Event                     | 109 Hours     | 0 Hours      | 109 Hours    |
| Localization                                    | 35 Hours      | 1 Hour       | 34 Hours     |
| <b>Company: No Value (5)</b>                    |               |              |              |
| Upgrade Servers                                 | 728 Hours     | 0 Hours      | 728 Hours    |
| Integrate eCommerce into Website                | 718 Hours     | 3 Hours      | 715 Hours    |
| Marketing Team Merger                           | 240 Hours     | 8 Hours      | 232 Hours    |
| CRM                                             | 14.75 Hours   | 3 Hours      | 11.75 Hours  |
| Product Launch Project                          | 945 Hours     | 0 Hours      | 945 Hours    |

## NOTE

Because there are several lines involved in creating an aggregator, it is recommended to define the view in the builder interface first on another column and then go to text mode to produce this code.

The querysort line has been removed in the sample code above. This line is not needed and may cause confusion if left in the view, because it allows the column header to sort by the field label to the right of the equal sign, which in this case has been changed to an expression.

# Calculated Aggregates (continued)



| AGGREGATOR ATTRIBUTES         | DESCRIPTION                                                                                                                                                                                                                                                                                                                                                                                            |
|-------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| column.#aggregator.           | The aggregator tag precedes all aggregate definition lines to separate these attributes from other view attributes for the column.                                                                                                                                                                                                                                                                     |
| displayformat                 | Controls how aggregate value will be rendered in the grouping. Through this line you define any text that will be displayed with the value.<br><br>Minutes ..... minutesAsString<br>Hours ..... MinutesAsString<br>Days ..... minutesAsDaysString<br>Currency ..... \$1234.56 ..... currencyStringCurrency<br>Percent ..... (12.34%) ..... doubleAsPercent<br>..... (12%) ..... doubleAsPercentRounded |
| function                      | Identifies the mathematical function used on the values returned in the results. (i.e. SUM, AVG, MIN, MAX, COUNT)                                                                                                                                                                                                                                                                                      |
| name or namekey               | The aggregate name is not necessary for list view aggregates; it is used as a label on aggregate charts.                                                                                                                                                                                                                                                                                               |
| valuefield or valueexpression | Defines the values that will be used by the aggregator for each result in the report. The aggregator only interprets numeric values, so when using the valueexpression there is no point in using CONCAT to append text to the number (i.e. '1234 Hours'). The word Hours is attached to the aggregated value through displayformat.                                                                   |
| valueformat                   | Describes the format of the value produced by the valuefield or valueexpression. The valueformat is usually always Int or IntAsInt.                                                                                                                                                                                                                                                                    |



## CHAPTER 4

# ADVANCED GROUPINGS

## OBJECTIVES

After completing this chapter, you will be able to:

- **Expand objects through groupings**
- **Utilize calculated groupings**

# Referencing Related Objects in Groupings

**SCENARIO** — Group the task list by the portfolio and program of the project where the task resides. These two options do not appear in the Builder; however, we know a relationship exists, because each task must reside on a project. And projects can belong to both portfolios and programs.

1. From the Reporting Area, create a task report.
2. Select the Groupings tab and click on the Add Grouping button.
3. Select group by project name. This becomes the placeholder grouping for Portfolio.
4. Select Add another Grouping.
5. Select group by project name again. This becomes the placeholder grouping for Program.
6. Switch to text mode.
7. Change the following lines in group 0 to extend it to the Portfolio:

group.0.valuefield=project:name

to

group.0.valuefield=project:portfolio:name

The screenshot shows a software interface for report configuration. At the top, there's a navigation bar with tabs: 'Columns (View)', 'Groupings' (which is highlighted in blue), 'Filters', and 'Chart'. Below this, a section titled 'Group your Report:' contains the following text:

```
textmode=true
group.0.valuefield=project:portfolio:name
group.0.displayname=Portfolio Name
group.0.namekey=view.relatedcolumn
group.0.linkedname=project
group.0.namekeyargkey.0=name
group.0.namekeyargkey.1=name
group.0.valueformat=string
group.1.linkedname=project
group.1.valueformat=string
group.1.displayname=Program Name
group.1.namekeyargkey.0=name
group.1.namekeyargkey.1=name
group.1.namekey=view.relatedcolumn
group.1.valuefield=project:program:name
```

# Referencing Related Objects in Groupings (continued)

8. Add the following line in group 0 to name the grouping:

```
group.0.displayname=Portfolio Name
```

9. Change the following lines in group 1 to extend it to the Program:

```
group.1.valuefield=project:name
```

to

```
group.1.valuefield=project:program:name
```

10. Add the following line in group 1 to name the grouping:

```
group.1.displayname=Program Name
```

11. Click the Save + Close button. Name the report ‘Tasks by Portfolio and Program’.

## NOTE

Text mode groupings also allow you to build four-level groupings; whereas, the builder interface only provides drop-down menus for three tiers.

```

textmode=true
group.0.valuefield=project:portfolio:name
group.0.displayname=Portfolio Name
group.0.namekey=view.relatedcolumn
group.0.linkedname=project
group.0.namekeyargkey.0=project
group.0.namekeyargkey.1=name
group.0.valueformat=string
group.1.linkedname=project
group.1.valueformat=string
group.1.displayname=Program Name
group.1.namekeyargkey.0=project
group.1.namekeyargkey.1=name
group.1.namekey=view.relatedcolumn
group.1.valuefield=project:program:name

```

# Referencing Related Objects in Groupings (continued)

**SCENARIO** —Create a Note report with a grouping that displays these tiers:

- Owner: Company: Name
- Project: Name

1. From the Reporting area menu, create a note report.
2. Select the Groupings tab; click the Add Grouping button.
3. Select to group by owner name. This becomes the placeholder grouping for Owner:Company.
4. Click Add another Grouping. Select Group By Project Name.
5. Click Switch to text mode.

| Type           | P                        | User       | Source | Entry Date      |
|----------------|--------------------------|------------|--------|-----------------|
| Owner » Name   |                          |            |        |                 |
| Project » Name | <input type="checkbox"/> | John Smith |        | 4/15/11 2:04 PM |

# Referencing Related Objects in Groupings (continued)

6. Change the following lines in group 0 to extend it to the Owner:Company.

```
group.0.namekey=view.relatedcolumn
```

```
to
```

```
group.0.name=Owner Company
```

```
group.0.valuefield=owner:name
```

```
to
```

```
group.0.valuefield=owner:company:name
```

7. Click the Done button to save text mode code.

8. Click the View tab and add at least one column to the view.

9. Save + Close. Name the report ‘Notes by Company and Project’.

| Type           | P                        | User       | Source | Entry Date      |
|----------------|--------------------------|------------|--------|-----------------|
| Owner » Name   |                          |            |        |                 |
| Project » Name |                          |            |        |                 |
|                | <input type="checkbox"/> | John Smith |        | 4/15/11 2:04 PM |
|                |                          |            |        |                 |





## Calculated Groupings

When working with groupings it is important to know that aggregated custom data does not show on summary reports.

When viewing reports you may find it useful to group the results by percent complete for each task. This is a useful way to see which tasks are complete, nearly complete, nowhere near complete, and not started. This can be useful, but when you have tasks that are 0%, 12%, 24%, 25%, 75%, 82%, 83%, 99%, etc., each task with a unique percentage complete will appear in its own grouping.

# Calculated Groupings (continued)

**SCENARIO** — Create ranges of percentages to simplify the organization of the task list. Use the following grouping breakdown:

- 0%
- 0% to 25%
- 25% to 50%
- 50% to 75%
- 75% to 100%
- 100%

1. From the Reporting area menu, create a new Task report.
2. Select the Groupings tab. Click the '+ Add Grouping' button.
3. Select Group by Percent Complete.
4. Switch to text mode.

Tasks by Percent Complete

Group your Report:

```
textmode=true
group.0.valueexpression=If((percentComplete)=0,"0%",If((percentComplete)<25,"0% to 25%",If((percentComplete)<50,"25% to 50%",If((percentComplete)<75,"50% to 75%",If((percentComplete)<100,"75% to 100%","100%"))))
group.0.linkedname=direct
group.0.valueformat=HTML
group.0.namekey=percentComplete
```

Done

Grouping Preview

| Task Name               | Assignments | Duration     | Pln Hrs     | Predecessors | Start On | Due On  | % Complete |
|-------------------------|-------------|--------------|-------------|--------------|----------|---------|------------|
| Train Inside Sales Team |             | 1234 Minutes | 20.57 Hours |              | 4/15/13  | 4/17/13 | 75%        |

Save + Close Cancel

# Calculated Groupings (continued)

5. Replace the value field line with the following value expression:

```
group.0.valueexpression=
IF({percentComplete}=0,"0%",
IF({percentComplete}<25,"0% to 25%",
IF({percentComplete}<50,"25% to 50%",
IF({percentComplete}<75,"50% to 75%",
IF({percentComplete}<100,"75% to 100%"
IF({percentComplete=100,"100%"))}))
```

6. Click Save + Close. Name the report Tasks by Percent Complete.
7. To adjust the order of the results, edit the report and add the Percent Complete column to the view.
8. With the Percent Complete column selected, check the option to sort by this column. Sort by ascending order.

## NOTE

The valueexpression text must be on a single line in the text mode interface with no line breaks. It is shown above with returns to make it easier to read.

| Task Name               | Assignments | Duration     | Pln Hrs     | Predecessors | Start On | Due On  | % Complete |
|-------------------------|-------------|--------------|-------------|--------------|----------|---------|------------|
| Train Inside Sales Team |             | 1234 Minutes | 20.57 Hours |              | 4/15/13  | 4/17/13 | 75%        |

# Group By Multiple Parent Tasks

**SCENARIO** — Suppose that in your organization you use many levels of subtasks. In some cases subtasks have the same names, and can only be accurately identified by their associated parent task(s). Your boss comes to you with a request that you create a task grouping that groups based on the parents of a task up to four generations back.

Your boss wants all the parent tasks to display in the grouping bar, separated by commas. If there is a grouping with fewer than four generations of parents, your boss only wants the actual parents shown, and doesn't want to see a lone comma in the place of a blank parent.

1. Create a task grouping that groups by the name of the parent task.
2. Switch to text mode.
3. Use a CONCAT statement in a valueexpression such as:  

```
group.0.valueexpression=CONCAT({parent}.{parent}.{parent}.name," ",{parent}.{parent}.{parent}.name," ",{parent}.{parent}.name," ",{parent}.name)
```

## CHAPTER 4: ADVANCED GROUPINGS

# Group By Multiple Parent Tasks (continued)

This works well with the exception of tasks that have fewer than four parents.

How to handle these?

Put an IF statement in the CONCAT to check to see if a parent name is blank. Only place a comma between parents that are not blank, as shown in the image.

```
group.0.valueexpression=CONCAT({parent}.{parent}.
{parent}.{parent}.{name},IF(ISBLANK({parent}.{parent}.
{parent}.{parent}.{name})," ", },{parent}.{parent}.{parent}.
{name}),IF(ISBLANK({parent}.{parent}.{parent}.{name})," ",
, },{parent}.{parent}.{name},IF(ISBLANK({parent}.{parent}.
{name})," ",),{parent}.{name})
```

The screenshot shows two overlapping windows. The top window is a list of tasks grouped by parent. The bottom window is a 'Customize Grouping' dialog box.

**Task List (Top Window):**

| #  | Name                                    | Parent              | Parent Parent  | Parent Parent Parent | Parent Parent Parent Parent |
|----|-----------------------------------------|---------------------|----------------|----------------------|-----------------------------|
| 1  | Design Phase                            |                     |                |                      |                             |
| 10 | Programming & Execution                 |                     |                |                      |                             |
| 18 | Testing & Review                        |                     |                |                      |                             |
| 22 | Implementation                          |                     |                |                      |                             |
| 29 | Go Live!                                |                     |                |                      |                             |
| 2  | Creative Brief                          | Design Phase        |                |                      |                             |
| 3  | Research Competitive Web Sites          | Creative Brief      | Design Phase   |                      |                             |
| 4  | Market Research on indoor soccer balls. | Creative Brief      | Design Phase   |                      |                             |
| 5  | Outline Web Site Features               | Creative Brief      | Design Phase   |                      |                             |
| 6  | Goals Outline                           | Creative Brief      | Design Phase   |                      |                             |
| 7  | Design Color Scheme                     | Goals Outline       | Creative Brief | Design Phase         |                             |
| 8  | Preliminary Layout                      | Design Color Scheme | Goals Outline  | Creative Brief       | Design Phase                |

**Customize Grouping Dialog (Bottom Window):**

Group your Report:  
textmode=true  
group.0.valueexpression=CONCAT({parent}.{parent}.{parent}.{name},IF(ISBLANK({parent}.{parent}.{parent}.{name})," ", },{parent}.{parent}.  
{parent}.{name}),IF(ISBLANK({parent}.{parent}.{parent}.{name})," ", },{parent}.{parent}.{name}),IF(ISBLANK({parent}.{parent}.{name})," ", ),{parent}.{name})  
group.0.name=Parents  
group.0.linkedname=parent  
group.0.namekeyargkey=parent  
group.0.namekeyargkey.1=name  
group.0.valueformat=string

Grouping Preview

| Task Name               | Assignments | Duration     | Pin Hrs     | Predecessors | Start On | Due On  | % Complete |
|-------------------------|-------------|--------------|-------------|--------------|----------|---------|------------|
| Train Inside Sales Team |             | 1234 Minutes | 20.57 Hours |              | 9/15/13  | 9/17/13 | 75%        |

Buttons: Done, Save Grouping, Cancel, Save as New Grouping

# APPENDIX

## OBJECTIVES

After completing this chapter, you will be able to:

- Create a custom form
- Create custom icons



# Exercise: Calculated Custom Data Expressions

Write the Calculation Custom Data expressions for the following fields:

- Hours Remaining for a Task  
SUB(Planned Hours,Actual Hours)/60 or  
SUB(Planned Hours/60,Actual Hours/60) or  
(Planned Hours/60)-(Actual Hours/60) or  
(Planned Hours-Actual Hours)/60
- Percent Remaining for a Task  
SUB(100,Percent Complete) or  
100-Percent Complete
- Variance of Days Between Planned Start Date and the Projected Completion Date for a Task  
ABS(DATEDIFF(Planned Start Date, Projected Completion Date)) or  
ABS(WEEKDAYDIFF(Planned Start Date, Projected Completion Date))
- Percentage of Task Planned Hours to Project Planned Hours  
ROUND(DIV(Planned Hours,Project.Planned Hours)\*100,2)
- User's Employment Duration  
DATEDIFF(\$\$TODAY,Hire Date)  
or  
WEEKDAYDIFF(\$\$TODAY,Hire Date)  
or  
CONCAT(DATEDIFF(\$\$TODAY,Hire Date), "Days")  
or  
CONCAT(WEEKDAYDIFF(\$\$TODAY,Hire Date)," Week Days")





# Exercise: Calculated Column Expressions

Write the Calculation Custom Data expressions for the following fields:

- Hours Remaining for a Task

`SUB({workRequired},{actualWorkRequired})/60`

- Percent Remaining for a Task

`SUB(100,{percentComplete})`

- Percentage of Task Planned Hours to Project Planned Hours

`ROUND(DIV({workRequired},{project}.{workRequired})*100,2)`

- User's Employment Duration

`DATEDIFF($$TODAY,{DE:Hire Date})`

or

`CONCAT(DATEDIFF($$TODAY,{DE:Hire Date})," Days")`

- User's Employment Duration multiplied by the User's Hourly Rate

`DATEDIFF($$TODAY,{DE:Hire Date})*{billionPerHour}`

or

`{DE:User's Employment Duration}*[billingPerHour]`

- User's Employment Duration on a Task View

`DATEDIFF($$TODAY,{assignedTo}{DE:Hire Date})`



# Custom Forms

Custom Forms allow fields and information otherwise not included to be added to Workfront. Organizations have the freedom to customize Workfront.

A System Administrator can access and modify the custom field area by going to Setup ► Custom Forms.

## Create a New Custom Form

1. Click the New Custom Form button.
2. Select the object type where the custom form will be applied.
  - Custom Forms can be added to Projects, Tasks, Issues, Documents, Portfolios, Expenses, Programs, People, Companies, and Iterations. When a form is applied to a record, the sub-tab is relabeled from ‘custom form’ to the name of the form you selected.
3. Select existing custom fields to add to your form, or create and add a new field from the form builder.
4. Click Save when finished.

## Custom Forms (continued)

Existing custom fields can be added to newly created forms. Just search for them in the Field Library and drag them onto the form preview on the right.

A warning will be displayed if a field is used on other forms, indicating that a change to the field settings will change the field on all forms.

Creating a new field right on the form is simple as well. Just choose the type of field and drag it over to the form.

Field settings and options can be modified when selecting a field in the form. Users can add choices, make a field required, set the data type, and add instructions that appear as tool tips.

Organize the fields on the form with section breaks or by dragging the fields into position. By dragging a field to the right of another, you can add fields to share the same row.

In order to create forms users must have edit rights to all custom form fields (parameters).

| CUSTOM FORM TERMS |                                                                                                     |
|-------------------|-----------------------------------------------------------------------------------------------------|
| CUSTOM FORM       | A way to create and include data you want to track on projects, tasks, and other Workfront objects. |
| FIELD             | A customized field where a user can enter and view information on the form.                         |
| SECTION BREAK     | A section heading or divider on the form. Helps organize and group the various fields.              |

The screenshot shows the 'Edit Issue' dialog box for a project titled 'Comprehensive Advertising Blitz'. On the left, there is a sidebar with tabs: Overview (selected), Settings, Custom Form (highlighted with a red arrow), Assignments, and Comment. In the main area, there is a section labeled 'Custom Form' with a red arrow pointing to its tab. Below this, there are fields for 'Risk' (dropdown menu), 'Risk Description' (text input), 'Risk Severity' (dropdown menu), 'Risk Likelihood' (dropdown menu), and 'Project Risk Impact' (checkboxes for Benefit, Budget, and Deliverables). There is also a 'Risk' dropdown menu at the top right of the main area.

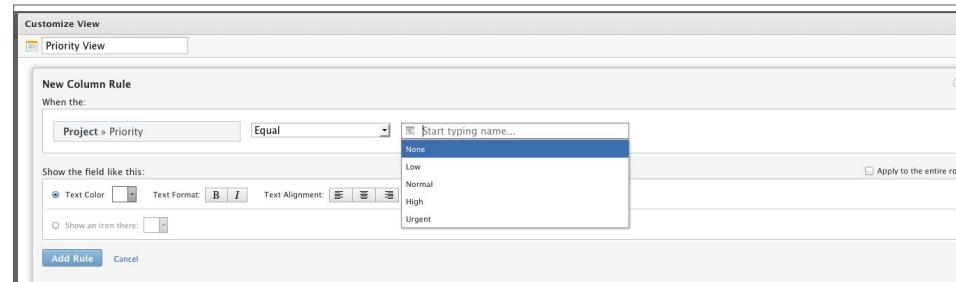
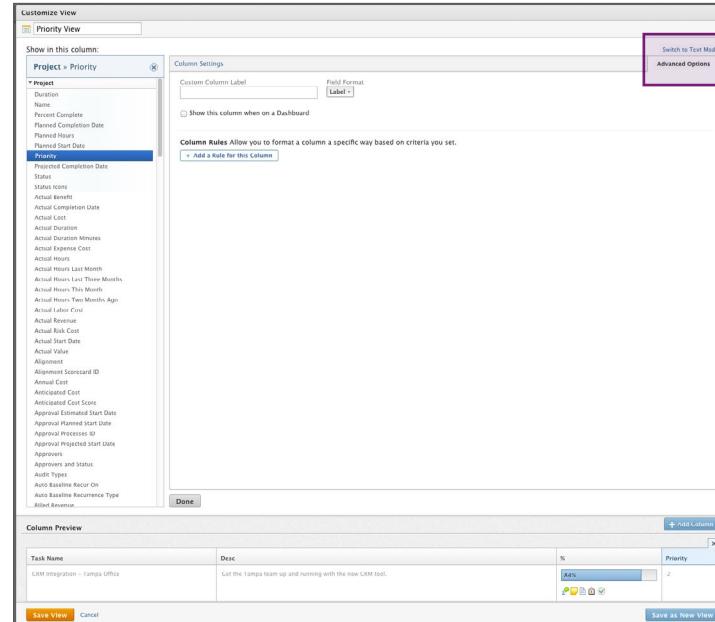
# Customizing Icons

Placing an image in a view presents a number of powerful ways to communicate and display data in a more concise and easy-to-read report.

For example, images can be used to create stop light reports, which provide report viewers a quick and intuitive way to read and interpret relevant data. When images are used as icons the view becomes easier to read because it is not cluttered with unnecessary text.

## Customize Icons in a View

1. Upload the image as a document on a project.
2. Preview the document and copy the complete URL located in the browser.
3. Create a new column on a view. Create a project priority column.
4. Select Advanced Options. Add a rule.
5. Select Not Null as the qualifier. This ensures the image always appears. Select ‘show an icon here’. Choose any icon. This becomes your placeholder.



# Customizing Icons (continued)

6. Click Add Rule.
7. Click Switch to text mode.
8. Locate the line that contains `truetext=`
9. Highlight and delete all text after `truetext=`
10. Replace the deleted text with the complete URL you copied earlier.

11. Add the following line:

```
image.width=20
```

12. Click Done.

