

cronology

Version 4.2.7

Console User Guide

Document Version 1.0

1	Introduction.....	4
2	Supported Environments	4
3	Getting Started.....	5
3.1	User Types.....	5
3.2	Console Screen Layout.....	5
3.3	Screen Refresh and Screen Resize.....	8
3.4	Navigation and Default Actions	8
3.5	Run History	9
3.6	Job Log.....	9
3.7	Menu Structure	10
3.8	About and Support.....	11
4	Creating a Job - A Quick Example For Administrators.....	12
5	Parameter Maintenance.....	14
5.1	Creating a Parameter.....	14
5.2	Editing and Deleting Parameters.....	15
6	Job Maintenance.....	16
6.1	Creating a Job.....	16
6.2	Editing a Job.....	22
6.3	Deleting a Job.....	22
6.4	Killing a Job	22
6.5	Cancelling a Job.....	22
6.6	Taking a Job Offline	22
6.7	Exporting a Job	23
6.8	Managing Job Dependencies.....	23
7	Executing Jobs Manually	25
8	Schedule Maintenance	26
8.1	Manage	26
8.1.1	Export.....	26
8.1.2	Suspend	26
8.1.3	Advance	27
8.1.4	Audit.....	27
8.1.5	Console Access	27
8.2	Settings	28
8.2.1	Environment Script.....	28
8.2.2	SSH Environment Script.....	28
8.2.3	Job Retention History.....	28
8.2.4	Job Run Log Limit	28
8.2.5	Console Environment Indicators.....	28
8.2.6	Console Auto Text	29
8.3	Global Update	29
8.3.1	E-mail Recipients.....	29
8.3.2	DB Password	29
8.3.3	DB Connect String	30
8.3.4	SSH Connection.....	30
8.4	Messaging Options	30
8.4.1	Operator E-mail Address.....	31
8.4.2	E-mail Format	31
8.4.3	Reminders.....	31
8.4.4	Bespoke.....	32
8.4.5	Templates.....	32

cronology

Version 4.2.7

Console User Guide

8.5	Audit History	33
9	Background Processes.....	34
9.1	Job Poller	34
9.1.1	Stopping the Job Poller	34
9.1.2	Autostart.....	34
9.2	Log Writer	35
9.3	Message Server.....	35
10	Status Summary and Job Analysis	36
10.1	Status Summary	36
10.2	Job Analysis.....	36
10.2.1	Statistics	36
10.2.2	Activity	36
11	Current SQL and Database Sessions.....	37
11.1	Current SQL.....	37
11.2	Database Sessions.....	37
	Appendix A - Cronology API Package	38

1 Introduction

The Cronology Console is the GUI front end for the Cronology product. It allows operators and administrators to control, examine, set up and execute jobs.

The console provides a consolidated view of your entire job schedule and provides the ability to view job logs, job source code, view active SQL / explain plans and much more.

A key feature of the console is that all job types (e.g. SQL script, stored procedure, shell script, JAVA executable) are all administered the same way – this can be advantageous during the testing phase of a project when testers may need to run many jobs on an ad-hoc basis, view the output etc without having to worry about how to invoke the underlying source code for each job.

The console is written using Oracle APEX, Oracle's web based development platform – and is therefore accessed via a web browser.

2 Supported Environments

In order to access the console you will require either:

- Google Chrome
- Microsoft Edge

N.B. other browsers supported by Oracle APEX may well work e.g. Mozilla Firefox, Apple Safari and Microsoft Internet Explorer, but are not officially supported by Cronology.

Your display resolution should be at least 1280 x 1024 and it is recommended that the browser window should be maximised.

3 Getting Started

3.1 User Types

Console access is granted via Oracle database roles. Database users can be assigned one of three different user types in order to access the Console:

- **Administrator (CRONOLOGY_ADMINISTRATOR role)** - access to all menu options and can set up and alter the job schedule i.e. create and amend jobs, parameters, dependencies etc. Administrators may also grant Console access to other database users via the console
- **Operator (CRONOLOGY_OPERATOR role)** - access to most menu options, cannot change the job schedule but may run jobs manually
- **Read Only (CRONOLOGY_READONLY role)** - access to most menu options but can view schedule information only.

Although Operator and Read Only users may have access to certain menus, the screens may only be available in read only mode (i.e. save buttons may be disabled / not visible) – see Menu Structure.

The Cronology Console does **not** create database users – that task needs to be performed by your DBA; once users exist console access can be granted by either:

- a DBA granting the appropriate role
- an existing Administrator user can use the Console Access screen to grant access – see Console Access

3.2 Console Screen Layout

The console main screen is as follows

The screenshot displays the Cronology Console interface for a user named 'CONSOLE_ADMIN_USER'. The main area shows a table of jobs with columns for Job Name, Status, Start Date Time, End Date Time, Duration, Frequency, and RSA. One job is highlighted in red, indicating a failure.

Job Name	Status	Start Date Time	End Date Time	Duration	Frequency	RSA
Sample Remote SQL*Plus Job	FAILED	23-JUN-2021 07:16:02	23-JUN-2021 07:16:05	00:00:00:03	SQL	<input type="checkbox"/>
Sample Shell Script 2	COMPLETE	09-JUN-2021 07:25:00	09-JUN-2021 07:27:00	00:00:02:00	SQL	<input type="checkbox"/>
Sample Shell Script 2 - Copy					SQL	<input type="checkbox"/>
Sample Remote SQL*Plus Job - Copy	FAILED	03-JUL-2021 15:09:27	03-JUL-2021 15:09:29	00:00:00:02	SQL	<input checked="" type="checkbox"/>
Sample Shell Script Job	COMPLETE	22-JUN-2021 14:58:38	22-JUN-2021 14:58:48	00:00:00:10	CALENDAR	<input checked="" type="checkbox"/>
> Child 1 - Shell Script Job With Parameters	COMPLETE	22-JUN-2021 14:58:06	22-JUN-2021 14:58:17	00:00:00:11	CALENDAR	<input type="checkbox"/>
> Child 2 - SQL*Plus Job	COMPLETE	22-JUN-2021 14:58:17	22-JUN-2021 14:58:17	00:00:00:00	CALENDAR	<input type="checkbox"/>
> Child 3 - SQL*Plus Job	COMPLETE	22-JUN-2021 14:58:17	22-JUN-2021 14:58:17	00:00:00:00	CALENDAR	<input type="checkbox"/>
> Child 3 - Stored Procedure					CALENDAR	<input type="checkbox"/>
Local Stored Procedure Job	COMPLETE	09-JUN-2021 07:22:06	09-JUN-2021 07:22:07	00:00:00:01	SQL	<input type="checkbox"/>
1					ONE OFF	<input type="checkbox"/>

The bottom section shows the 'Run History' for the selected job 'Sample Shell Script Job'. It includes fields for Job Name, Source Type (Shell Script), Schedule Type (TIMED), Next Submit Date Time (THU 10-JUN-2021 00:04), Average Duration (00:00:01:16), Runtime Warning, and Status Information (No errors, Exit Code: 0). There are also buttons for 'Execute', 'Cancel', 'Kill', 'Offline', 'Export', 'Edit', 'Delete', 'Dependencies', and 'Documentation'.

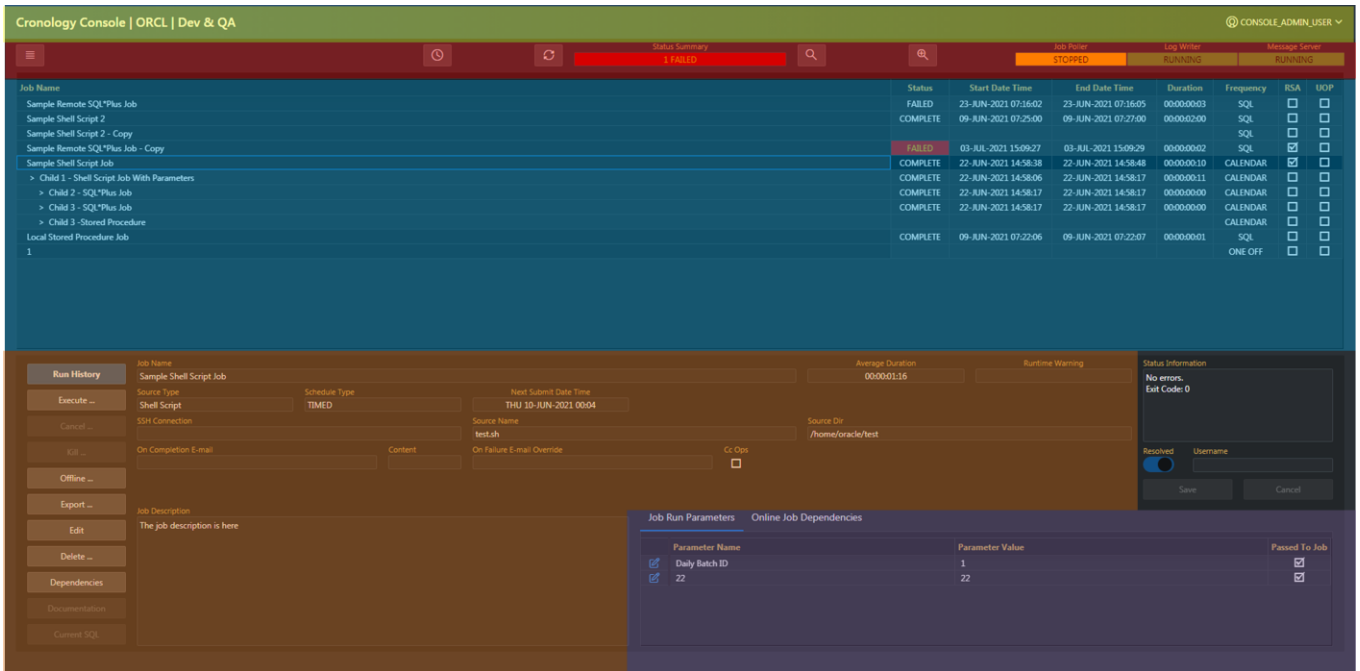
The 'Job Run Parameters' section shows a table with columns for Parameter Name, Parameter Value, and Passed To Job:

Parameter Name	Parameter Value	Passed To Job
Daily Batch ID	1	<input checked="" type="checkbox"/>
22	22	<input checked="" type="checkbox"/>






cronology

Version 4.2.7
Console User Guide

The display is split into six distinct areas:



- 1) **Title Bar** (green): showing the current database connection and any environmental description that has been configured. The user menu allows the user to log out and access the About screen.
- 2) **Header Bar** (red): showing the schedule status summary and server process statuses. This section also contains five buttons:

Button	Keyboard Shortcut	Description
	m	Main Menu The menu structure is described in detail at the end of this section
	t	Database server date and time N.B. this may be different to the date and time on your PC
	r	Refresh Display Forces a refresh of the current console display
	f	Find For finding jobs within the schedule either by job or source name
	n	Find Next Repeat the previous Find operation

Double Click Functionality:

- **Status Summary:** a double click will filter the job display to only show jobs with a status matching the status summary – for example, useful for homing in on FAILED jobs
- **Job Poller:** a double click will display the process history
- **Log Writer:** a double click will display the process history
- **Message Server:** a double click will display the process history

3) Job Schedule (blue): the job list is ordered such that jobs that are due to run next appear at the top of the display. As jobs run and complete this view changes dynamically.

Double Click Functionality:

- **Job Name:** go directly to latest job log

Ctrl + Double Click Functionality (filters):

- **Job Name:** display the job name filter dialog – options to filter on TIMED jobs only and jobs that are due for execution today only are also provided
- **Status:** immediately filter on the selected Status (double click the column header to cancel the filter)
- **Frequency:** immediately filter on the selected Frequency (double click the column header to cancel the filter)

Filters are cleared by double clicking the column header with filter applied, indicated by reverse angle brackets e.g. > Status <

4) Job Detail (brown): shows detail on the currently selected job along with action buttons (left) allowed on the job.

Double Click Functionality:

- **Next Submit Date Time:** for timed jobs, a double click will display the execution window information for the job
- **Source Name:** a double click will display the source code (if available) from the current job
- **On Completion E-mail:** if populated, a double click will display the full field – useful if the display has been truncated
- **On Failure E-mail Override:** if populated, a double click will display the full field – useful if the display has been truncated

5) Latest Execution Information (black): shows information about the latest execution. Operators use this section to update information about the job if it fails for any reason.

Double Click Functionality:

- **Status Information:** when resolving a failed job a double click will display a list of 'Autotext' options appears. This is a list of common reasons for job failures that can quickly be selected rather than re-typing each time

6) Job Parameters and Dependencies Tabs (purple): for the currently selected job, displays the parameters the job takes (displaying the latest execution values) and any dependent jobs

Double Click Functionality (Dependencies):

- **Job Name:** a double click will jump the display to the dependent job

3.3 Screen Refresh and Screen Resize

As the schedule runs the changing data necessitates the screen automatically refreshes. The default refresh rate is every 30 seconds but can be changed from the Main Menu -> Console -> Refresh. The screen can also be refreshed on demand at any time by clicking the refresh button in the header bar (or pressing 'r').

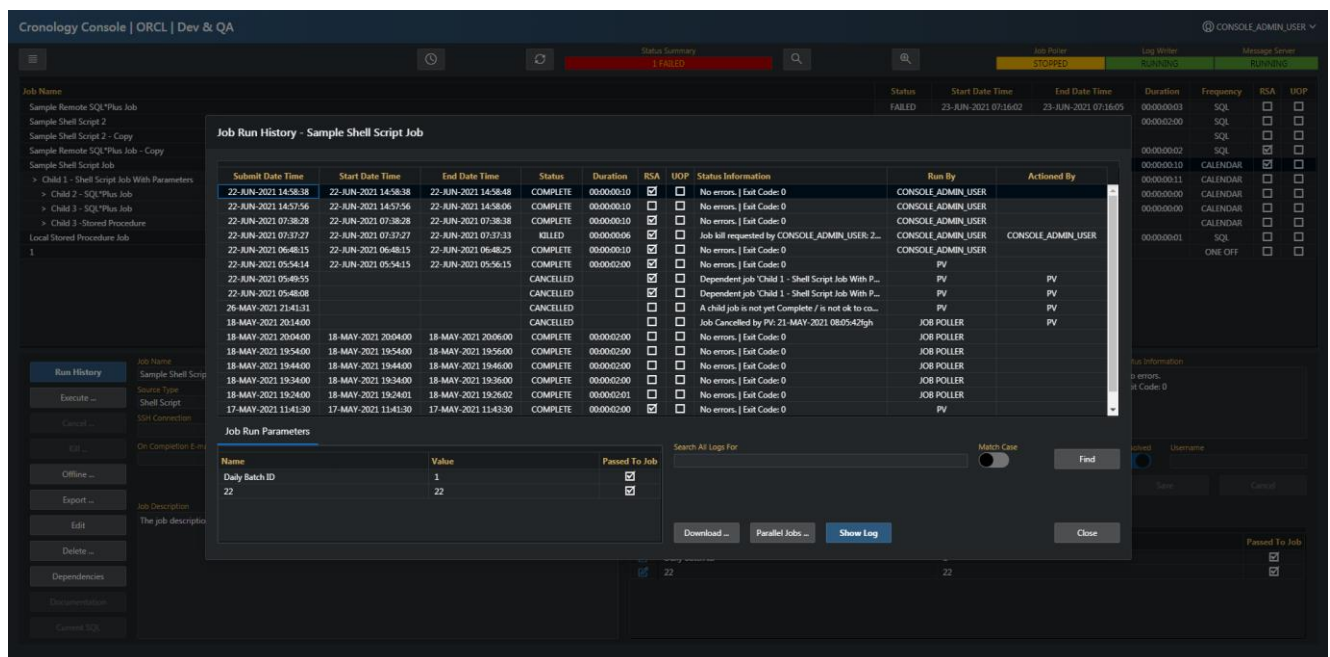
If required, you may alter the console to fit to screen mode via Main Menu -> Console -> Fit to Screen.

3.4 Navigation and Default Actions

Using the keyboard can sometimes be quicker than mouse clicks when navigating the schedule, in particular:

- the up and down arrow keys
- the page up and page down keys
- the enter key
- the ESC key

When viewing the main screen, the Run History button is the default action when the user presses the enter key. Therefore the user can navigate up and down the schedule using the arrow keys (or page up and page down), then press return to view the selected jobs run history:

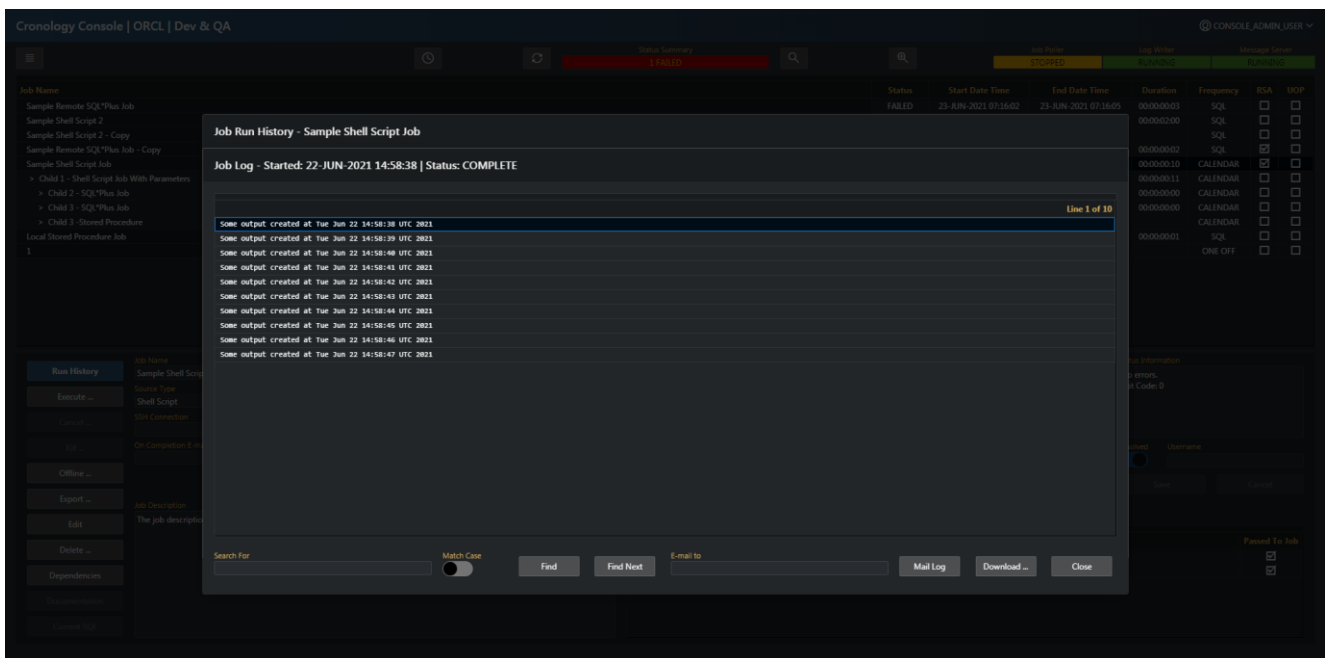


Here the Show Log button is the default action when the user presses return. The user can therefore navigate up and down the run history and simply press return to view the job log for that run:

cronology

Version 4.2.7

Console User Guide



Thus users can quickly view the latest log for a job by navigating to it in the schedule panel and hitting the return key twice. Likewise, these screens can quickly be dismissed by hitting the ESC key.

3.5 Run History

Filters are available on the Status and Actioned By columns (simply double click a value). (As can be seen from the screen shot in the previous section, the run history panel has a number of buttons available:

- **Find:** search all job run logs for a particular search string (the checkbox is for matching case exactly – the tooltip on the checkbox states its function)
- **Download:** download the run history in either CSV or HTML format
- **Parallel Jobs:** display a list of jobs that were running in parallel with this run
- **Show Log:** show the job log for the selected run

3.6 Job Log

As can be seen from the screen shot in the previous section, the job log panel has a number of buttons available:

- **Find:** search the log for a given string (the checkbox is for matching case exactly – the tooltip on the checkbox states its function)
- **Find Next:** find the next occurrence of the search string
- **Mail Log:** e-mail the job log to the specified recipients
- **Download** – download the log in either CSV or HTML format

Normal windows copy and paste keys can be used e.g. Ctrl-C, Ctrl-V to copy and paste individual lines from the log. By pressing F8 you can toggle into multi-select mode and highlight multiple lines by holding down the SHIFT key. The selection can then be cut and paste as required.

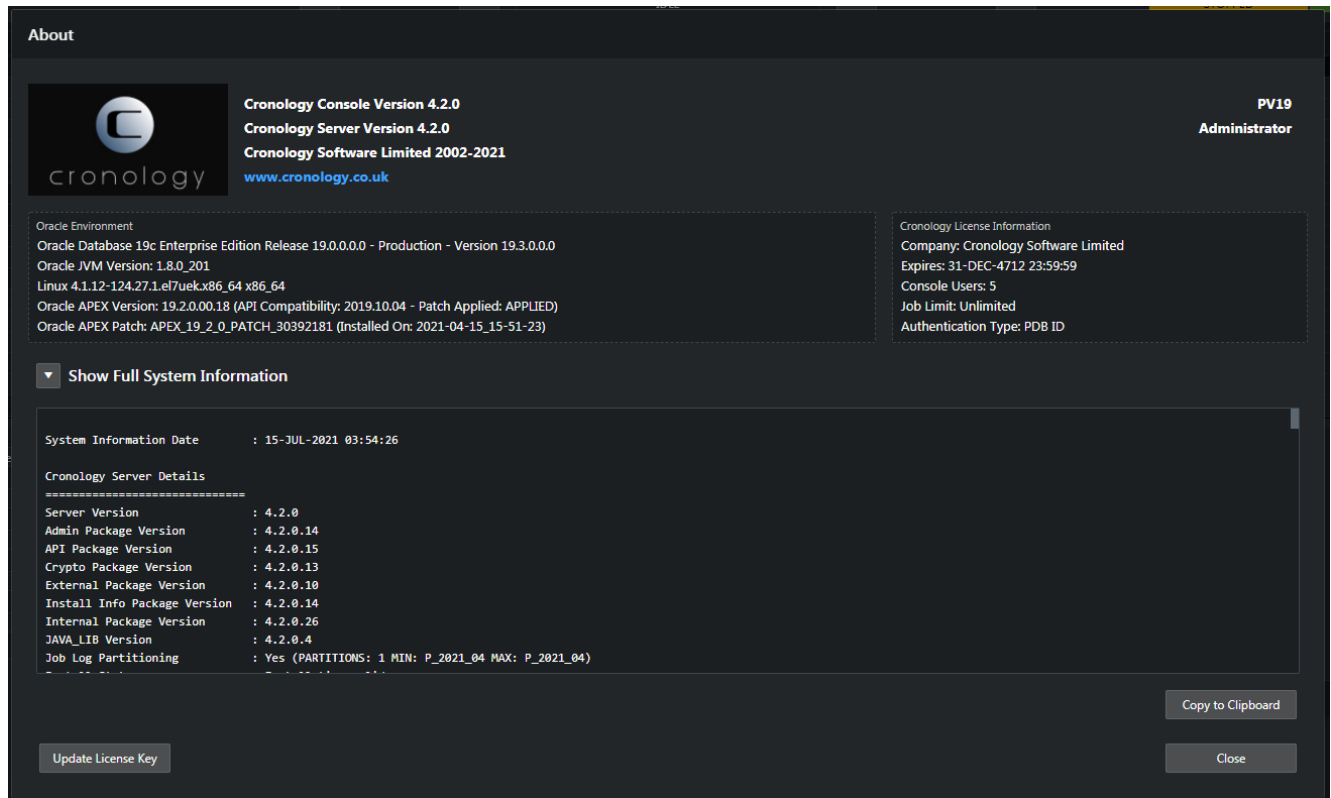
3.7 Menu Structure

Main Menu Item	Sub Menu	Screen Tab	Administrator	Operator	Read Only	
Job Parameters			✓	✗	✗	
Create Job	New		✓	✗	✗	
	Copy Selected		✓	✗	✗	
Bring Job Online			✓	✓	✗	
Schedule	Manage	Export	✓	✓	✗	
		Suspend	✓	✓	✗	
		Advance	✓	✗	✗	
		Audit	✓	✗	✗	
		Console Access	✓	✗	✗	
	Settings	Environment Script	✓	✓*	✓*	
		SSH Environment Script	✓	✓*	✓*	
		Job History Retention	✓	✓*	✓*	
		Job Run Log Limit	✓	✓*	✓*	
		Console Environment Indicators	✓	✓*	✓*	
		Console Auto Text	✓	✓*	✓*	
	Global Update	E-mail Recipients	✓	✗	✗	
		DB Password	✓	✗	✗	
		DB Connect String	✓	✗	✗	
		Server Connection	✓	✗	✗	
	Messaging Options	Operator E-mail Address	✓	✓	✗	
		E-mail Format	✓	✗	✗	
		Reminders	✓	✓	✗	
		Bespoke	✓	✓	✗	
		Templates	✓	✗	✗	
	Audit History			✓	✓	✓
	Background Processes		Job Poller	✓	✓	✓*
			Log Writer	✓	✓	✓*
		Message Server	✓	✓	✓*	
Status Summary			✓	✓	✓	
Job Analysis			✓	✓	✓	
Database Sessions			✓	✓	✓	
Console	Refresh		✓	✓	✓	
	Fit to Screen		✓	✓	✓	

- ✓ Menu or tab is visible/accessible
- ✗ Menu or tab is not visible/accessible
- ✓* Menu or tab is visible/accessible but read only

3.8 About and Support

The About screen can be reached from the user menu in the title bar.



This screen provides comprehensive details of all the software components and settings for the schedule. This information can be copied to the clipboard via the Copy to Clipboard button and may be requested by Cronology Support when dealing with support issues.

The option to update the license key is also accessible via this screen should your license key need renewing. Please contact Cronology Support for license keys.

E-mail: support@cronology.co.uk

4 Creating a Job - A Quick Example For Administrators

Let's create a SQL*Plus job that takes a single parameter and run it via the console. To run this example you will need to know the username and password for a test database account that has execute privilege on the DBMS_LOCK package.

1) Create the following SQL source file called cronology_test.sql in a directory of your choosing (remember the UNIX oracle user must have access to this directory):

```
prompt Anonymous PL/SQL block ...
begin
  cronology.api.log_line('Realtime output from PL/SQL block ... sleeping for 60 seconds ...');
  dbms_lock.sleep(60);
  cronology.api.log_line('The supplied parameter was: &1');
end;
/

prompt All done ... bye.
```

2) Logon to the Cronology Console and connect to your testing database

3) Create a parameter:

- Go Main Menu -> Job Parameter. Click the 'Create' button
- Use the following attributes:
 - Name: Test Parameter
 - Description: Test
 - Data Type: Character
 - Parameter Type: Fixed
 - Value: Hello World
- Click 'Create' – the parameter will be created and you will enter edit mode
- Click 'Cancel' to go back the Parameters list
- Click 'Close'

4) Create a job:

- Go Main Menu -> Create Job -> New
- Use the following attributes:
 - Name: Test job
 - Priority: 3
 - Success Code: 0
 - Runtime Warning: <leave blank>
 - If DB Bounced: Do Nothing
 - Overdue Behaviour: Catch Up
 - Description: Test

Source Tab

- Source Type: SQL*Plus Script
- SSH Connection: <leave blank>
- Source Directory: <your directory>
- Source Name: cronology_test.sql
- DB Username: <test username>
- DB Password: <test username password>

cronology

Version 4.2.7

Console User Guide

- DB Connect String: <leave blank>
- Connect As: <leave blank>

Schedule Tab

- Schedule Type: TIMED
- Next Submit Date Time: <double click for current date then add 1 to the year>
- Execution Window Start: <leave blank>
- Execution Window End: <leave blank>
- Frequency: ONE OFF

Parameters Tab

- Click the 'Add' button and select the parameter you created named 'Test Parameter'

Click 'Create' – the job will be created and you will enter edit mode

Click 'Cancel'

5) Run the job:

- Ensure the job is selected by clicking on the job name on the main screen
- Click the 'Execute ...' button
- Click 'Execute'

6) Look at the job log:

- Click 'Run History ...' (or press return)
- Click 'Show Log ...' (or press return)

5 Parameter Maintenance

As parameters are part of a jobs definition you typically create parameters before creating a job. The same parameters can of course be used by many jobs.

5.1 Creating a Parameter

To create a new parameter, from the main screen click on the Main Menu button (or press m). Select Job Parameters. A list of available parameters will be displayed. To create a new parameter click the 'Create' button.

The Create Parameter panel will appear in the lower portion of the screen:

Name: A unique name for the parameter

Copy Existing Button: click this button to select an existing parameter and copy its attributes into this new parameter definition, the name field will not be copied.

Description: A meaningful description of the parameter and its use

Data Type: Either 'Character', 'Date', 'Number', 'Boolean' or 'Timestamp'

Parameter Type: Either:

- **Environment Variable:** for UNIX environment variables. The variable should be defined in the application environment script. The location of this script is set via Main Menu -> Schedule -> Settings -> Environment Script Tab
- **Fixed:** for literal values
- **SQL Statement:** for dynamic / derived parameters. SQL statements provide a powerful means to obtain parameter values from anywhere in your database e.g. they could be values held in your own application tables, Oracle / application functions, Oracle sequences, sysdate / systimestamp etc.

Value: The value of the parameter. For Fixed parameters this will be a literal value. For SQL based parameters this field should contain the SQL statement that will be used to fetch the value. No trailing semi-colon should be specified. Remember to prefix object names with the schema owner (it is the CRONOLOGY database user that will be evaluating the parameters). It is highly recommended you use the Test Result button to ensure your SQL statement behaves as expected. If your SQL statement contains an Oracle sequence you will be asked if you wish to continue testing the parameter (will advance the Oracle sequence) or syntax check the statement only. For Environment variables specify the variable name e.g. \$MY_VAR.

Conceal Values: Checking this toggle will mean the parameter value will appear as asterisks (*) on the main screen Job Run Parameters panel. This is useful if the parameter contains sensitive information that operators or read only users should not be allowed to see.

Test Result Button: pressing this button will evaluate the parameter and display its value. N.B. if your parameter is an Oracle sequence testing the parameter will cause the sequence to increase.

5.2 Editing and Deleting Parameters

Parameters can be edited or deleted by clicking on the Main Menu button (or press m) and selecting Job Parameters. Click the blue edit icon next to the parameter you wish to edit.

You may quickly search for parameters by clicking on any column header and adding a filter.

Use the 'Save' button to save changes, or 'Delete' to delete the parameter

The edit screen also contains an 'Audit History' button that shows all changes made to the parameters (see Schedule Management - Audit).

N.B. You may only delete a parameter if it is not used by any job. Use the 'Show Uses' button to see if the parameter is currently in use.

6 Job Maintenance

Important note: any local jobs (no Server Connection specified) that have a source type of Shell, Perl, Python, Java or Binary Executable are executed on the DB server as whichever UNIX user started the Oracle database. This is typically a user called oracle. As such it is vital that the oracle user has all privileges necessary to access and run the jobs source script.

If the job you are about to create requires parameters, make sure you have created them / they exist before creating the job.

6.1 Creating a Job

To create a new job, from the main screen click on the Main Menu button (or press m). Select **Create Job -> New**. The Create Job panel will appear in the lower portion of the screen:

The screenshot shows the 'Create Job' form with the following fields and sections:

- Name:** A text input field.
- Priority:** A dropdown menu with '0' selected.
- Success Code:** A text input field.
- Runtime Warning:** A text input field.
- If DB Bounced:** A dropdown menu with 'Do Nothing' selected.
- Override Behaviour:** A dropdown menu with 'Catch Up' selected.
- Description:** A large text area.
- Documentation Link:** A text input field.
- Template Messaging Text:** A text input field.
- Source Section (Active):**
 - Source Type: A dropdown menu.
 - SSH Connection: A text input field.
 - Source Directory: A text input field.
 - Source Name: A text input field.
 - DB Username: A text input field.
 - DB Password: A text input field.
 - DB Connect String: A text input field.
 - Connect As: A text input field.
- Buttons:** 'Cancel' and 'Create' buttons at the bottom right.

To quickly copy an existing job, select the source job in the main schedule then click on the Main Menu button (or press m). Select Create Job -> Copy Selected.

Job Name: A unique name for the job

Priority: A number to represent the priority of the job. This number is only used in failure message notifications.

Success Code: The maximum UNIX exit code that represents successful completion of the job. The default is zero. Only change this value if your source script exits successfully with a code other than zero.

Runtime Warning: Specify a runtime threshold time in days, hours, minutes and seconds after which a warning message will be sent. This is useful to notify operators if a job exceeds its expected runtime.

If DB Bounced: If a running job is interrupted by the database being shutdown this setting can be used to specify whether or not to automatically restart the job:

- Do Nothing: take no action, i.e. leave job in failed state
- Resubmit: automatically resolve the job and resubmit it for execution

If Overdue: Specify the action to take if the job becomes overdue e.g. a failed job is resolved after its next scheduled run time:

- **Catch Up:** stick to the schedule and run every overdue occurrence. This may result in a job running a number of times consecutively
- **Advance:** advance the jobs schedule to the next execution due after the current date and time

Job Description: A meaningful description of the job and its purpose

Documentation Link: specify a link to a document (e.g. web page on your intranet, document on your LAN) that relates to the job. This link can then be used by Console users to access any relevant information for the job

Template Messaging Text: job specific text to use for the #JOB_MESSAGE_TEXT# substitution variable. This is only applicable if using messaging templates (see later).

Source Tab

Source Type: Select a source type from one of the following:

- **Binary Executable** – this could be an executable that is part of your application (e.g. compiled C code) or a UNIX command (e.g. rm)
- **File Watcher** – a job of this type will wait for the specified file to arrive on the OS before completing successfully. A file watcher job essentially takes as its parameters: a fully qualified file name, a timeout (in minutes) and an optional polling interval (in minutes, if omitted the default is 1 minute). For added flexibility the file name may be specified as 2 separate parameters (a directory and a file name). This means a file watcher job has 4 allowable parameter combinations:

Parameter Combination 1 (uses default polling interval)

1. Fully qualified file name (directory + file name)
2. Timeout in minutes

Parameter Combination 2

1. Fully qualified file name (directory + file name)
2. Timeout in minutes
3. Polling interval in minutes

Parameter Combination 3 (uses default polling interval)

1. File directory
2. File name
3. Timeout in minutes

Parameter Combination 4

1. File directory
2. File name
3. Timeout in minutes
4. Polling interval in minutes

cronology

Version 4.2.7

Console User Guide

- **Java (Class / Jar)**
- **Perl Script**
- **Placeholder** – as the name suggests this is purely a placeholder in the schedule. There is no job source associated with a placeholder, its acts as a dummy job. Placeholders are useful when you wish to group jobs together e.g. you might put all your month end processing jobs as child jobs of a “Month End” placeholder. Any dependencies that month end processing has can then be set up on the placeholder job rather than repeating the dependencies for each child job.
- **Python Script**
- **SQL*Plus Script**
- **Shell Script**
- **Stored Procedure**

Source Name: the name of the source file / database procedure (this may need to be fully qualified depending on the user that connect to the database e.g. <OWNER>.<PACKAGE>.<PROCEDURE>)

Source Directory: the directory location of the source file

SSH Connection: Specify valid SSH connection details in the format <user>@<server>. This allows Cronology to run jobs on remote servers. To use this feature you must:

1. have password-less SSH configured from the Cronology DB server to the user and server specified
2. ensure the Cronology SSH_EXE_DIRECTORY system parameter is set – contact your DBA to configure this via the Cronology ADMIN package

DB Username and Password: Specify a database username and password

DB Connect String: Leave this field blank to connect to the current database, or specify a valid TNS alias to connect to a remote database. **If you are running a job on a remote database, Cronology must be installed on the remote database in order for the job to run correctly.**

Connect As: a SYS role (SYSASM, SYSDBA or SYSOPER) for the DB user if applicable

Schedule Tab

Schedule Type: Select from either ‘TIMED’ or ‘CHILD’. A timed job runs at a specific time (launched by the Job Poller process). A child job is launched only when it parent job completes successfully.

For CHILD jobs:

Parent Job: Select the job to act as a parent for this job i.e. this job will be launched only upon successful completion of the parent.

For TIMED jobs:

Next Submit Date Time: Specify the data and time you wish the job to next execute.

Execution Window Start / End Date Time: If you only want the job to run for a specified period of time (e.g. you want it to run every 30 minutes but only for next week) then specify the window using the start and end date time fields

Frequency and Interval: Select a frequency for the job to run at. A jobs execution schedule is always based on the Next Submit Date Time and its Frequency, e.g. if you want to run a job every Monday at 3:00pm starting on the 2nd March 2020, set the Next Submit Date Time to be the date and time of the first execution, 02-MAR-2020 15:00 and set the frequency to 'WEEK(S)' and the interval to 1.

Interval based frequencies (values 1-999 allowed):

- MIN(S)
- HOUR(S)
- DAY(S)
- WEEK(S)
- MONTH(S)
- YEAR(S)

Remember for frequencies other than MINS or HOURS the time element is preserved when Cronology calculates the next execution time e.g. if you want the job to run **on** the hour make sure the Next Submit Date Time is on the hour e.g. 04-OCT-2010 15:00. To run the job hourly at a quarter past the hour the Next Submit Date Time would be 04-OCT-2010 15:15

Other frequency types:

- SUN-THU – run the job once a day from Sunday to Thursday only
- MON-FRI – run the job once a day from Monday to Friday only
- SAT-SUN – run the job once a day on Saturday and Sunday only
- ONE OFF – a special case, run the job at the Next Submit Date Time **and never run the job again**
- CRON – specify a frequency in UNIX cron schedule format e.g. to run a job at 03:30 every day specify 30 3 * * *
- SQL – if you wish to schedule jobs based on data held in the database, specify a SQL query that returns a valid Oracle **date or timestamp** type. Cronology will use this query to calculate the next run date time once the job has run
- CALENDAR – select specific dates and times to run the job if there is no regular frequency to a job. When selected, an additional tab appears where specific run dates and times can be entered.

Calendar Tab (only visible for Calendar based frequencies)

Maintain all the specific run times required for the job. Cronology does not (indeed cannot) automatically calculate the next run date time for Calendar based jobs, therefore all required run times must be specified manually.

Parameters Tab

Add any job parameters the job may require. Parameters can be added but do not necessarily have to be passed to the job; they can be present purely to maintain a job dependency (see later). If you do not wish to actually pass a parameter to the job but require it for dependency purposes set **Pass To Job** to No. For Stored Procedure jobs a button will be visible which will show the procedure specification i.e. you can see what parameters the procedure takes.

Status Overrides Tab

There may be times when you wish to mark a job as completed or failed even though it has not actually done so. Once a job finishes (be it successfully or unsuccessfully) Cronology allows you to scan the job log generated by the job for keywords or phrases. You may force the job status to either FAILED or COMPLETE based on the search results. E.g. say the job produces a “ORA-01403 No data found” error which would normally mark the job as failed, if you know that for this job it is OK to ignore this error you could set an override as follows:

Source	Schedule	Parameters	Status Overrides	E-mail Options
Log Search String			Condition	Mark As
%ORA-01403%			Found	COMPLETE

The use of standard Oracle ‘like’ wildcards (% and _) can be used in the search strings.

If you add multiple overrides they are processed in the order they are listed. E.g. say the same job also writes a sales revenue amount to the log, if the amount is zero you could force the job to fail by specifying:

Source	Schedule	Parameters	Status Overrides	E-mail Options
Log Search String			Condition	Mark As
%ORA-01403%			Found	COMPLETE
%Sales Revenue: £0.00%			Found	FAILED

Even if the ORA-1403 is raised and detected, the zero value sales revenue is the last override to be processed and hence will supersede any previous overrides.

You can also override the job if it does NOT find a specified string; do this by changing the Condition from “Found” to “Not found”.

N.B. The Mark As drop down also contains a COMPLETE* option. This will mark the job complete but will NOT launch any child jobs. For example, say a File Watcher job times out – this would normally result in a job failure. If this is not actually an issue for your processing, the following override would mark the job complete and not process any subsequent child jobs:

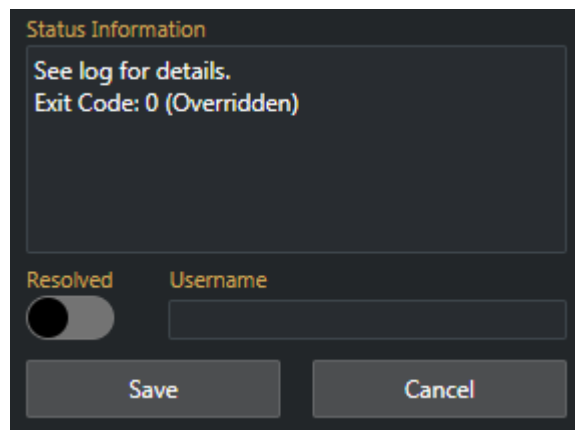
cronology

Version 4.2.7

Console User Guide

Source	Schedule	Parameters	Status Overrides	E-mail Options
Log Search String			Condition	Mark As
%File Watcher timeout reached%			Found	COMPLETE*

If an override has taken effect you will see an indicator in the status information on the main console screen:



E-mail Options Tab

This tab allows various job specific email options to be set.

On Completion Tab: By default no messages are sent when jobs complete successfully. Entries here allow a custom message to be sent when the job completes.

E-mail: Specify a valid e-mail address, separate multiple addresses with a single space.

E-Mail Content: If a completion e-mail is to be sent, specify the content by selecting from:

- **<NONE>** means the body of the e-mail will be empty
- **Job Log** will always e-mail the contents of the job log (even if the job produced no output)
- **Job Log *** will only send an e-mail if the job created some log output. This is useful for creating alert/monitoring type jobs. If the job completes and creates some output in the job log (e.g. testing or looking for a certain condition in the application or database) then an e-mail of the log content will be sent. If the job completes but produces no output (i.e. the condition is not met) then no e-mail is sent
- **Message** allows you to specify a custom message

On Failure Tab: By default the operators will be notified of job failures. Entries here will override this. This is useful if you wish direct failure notifications for specific jobs to specific parties.

E-mail Override: Specify a valid e-mail address, separate multiple addresses with a single space.

Cc Operators: If you have specified an override address for failures but still want the default Operator E-mail address to be copied in then check this box.

6.2 Editing a Job

To edit the currently selected job, on the main screen click the 'Edit' button in the job detail section (stack of buttons on the left hand side).

The Edit Job screen will appear. Fields are as described in Creating a Job.

The edit screen also contains an 'Audit History' button that shows all changes made to the job (see enabling Auditing).

6.3 Deleting a Job

To delete the currently selected job, on the main screen click on the 'Delete ...' button in the Job Detail section.

You will be asked to confirm the delete, **please note that all historical job information (job run history and job logs) will also be deleted.**

6.4 Killing a Job

When a job is RUNNING the job may be terminated immediately by clicking the 'Kill ...' button.

For Stored Procedure and SQL*Plus jobs, Cronology issues a kill session (immediate) command on the database session. For other types of jobs (e.g. shell scripts) the parent process and all spawned child process are killed using kill -9

6.5 Cancelling a Job

If a job is in a WAITING or SUSPENDED state (see later) the job may be cancelled by clicking the 'Cancel ...' button. This will prevent it from launching once its dependency is released.

6.6 Taking a Job Offline

If you wish to remove a job from the schedule but do not want to delete it, you can take the job offline by clicking on the 'Offline ...' button. All the jobs details will be retained but it will no longer appear in the Job Schedule section and will not be executed. Jobs can be brought back online via the Main Menu button -> Bring Job Online.

6.7 Exporting a Job

By clicking the 'Export ...' button Cronology allows you to export job definitions by one of two methods:

- **Release script.** A SQL script will be created that inserts the required metadata into a CRONOLOGY schema in another database. The script can form part of your project release / code base. When using this option you will be asked if you wish to use any existing database and server connection details, or have the script prompt for input when the script is running (the script can always be edited to suit your needs later).
- **DB Direct.** This option allows you to connect to another database running Cronology and create the job directly / immediately. You will need to provide valid console logon credentials for the target database. The database connect string must be resolvable from the current database server.

Options are also available to specify if subsequent child jobs and job dependencies should also be exported.

6.8 Managing Job Dependencies

Cronology provides three mechanisms to control when dependent jobs are launched:

Parent – child dependencies. These need no introduction and are straight forward to configure when creating the job. Child jobs launch when the parent completes successfully, in a sense creating a 'chain' of jobs. That is the extent of the relationship. It is simple, but effective. Chains of jobs can easily be setup in this way, some child jobs may run in parallel with each other, and in turn of course, child jobs may themselves have subsequent child jobs under them.

Parallel dependencies. Sometimes you simply want to ensure that two jobs do not run in parallel e.g. due to potential resource conflicts. Cronology allows you to specify if a job should not launch if the another is in a RUNNING state.

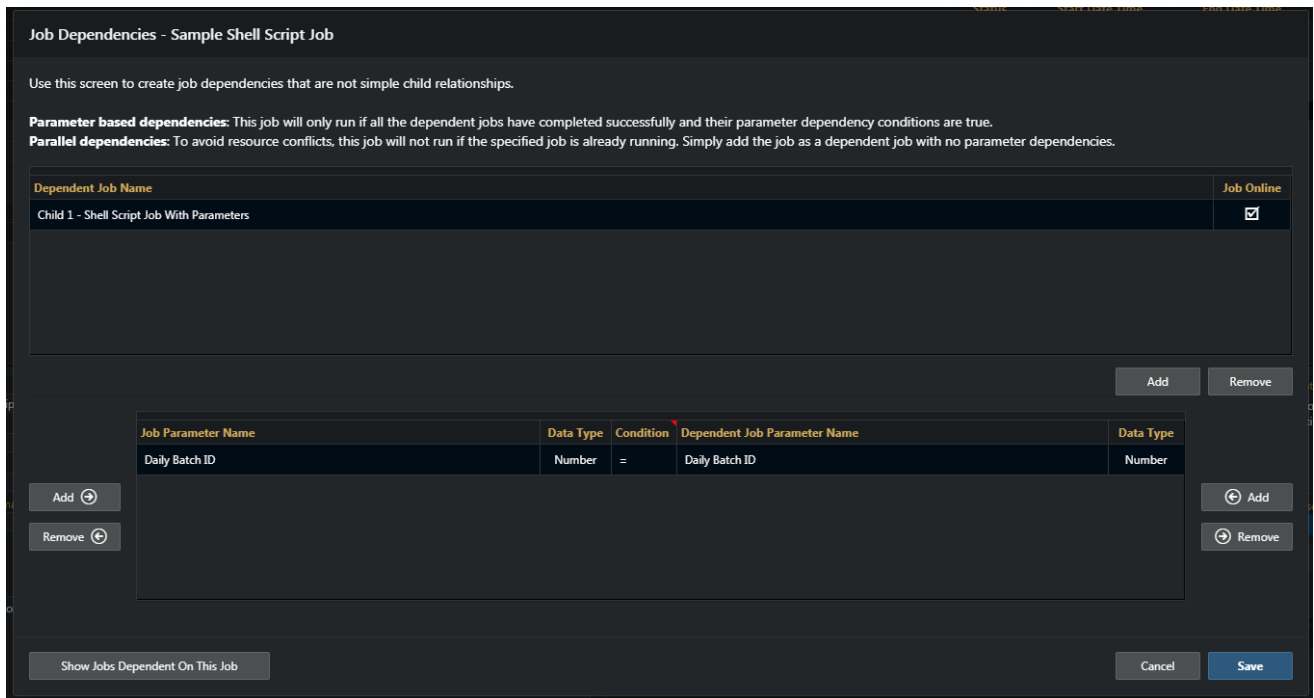
Parameter based dependencies. What if a job in one parent-child chain depends on another in a separate chain however? Cronology allows you make jobs dependent on one another via their **job parameters**. A simple example of this could be a system which uses a common close of business (COB) date as a parameter to many of its daily running jobs. Jobs using this parameter can be made dependent on one another allowing us to set up dependencies like: "Job B will only launch for COB date X if Job A has successfully completed using COB date X"

The linkage does not necessarily have to be made using the same parameter. You are free to link any parameter of Job A to any parameter of Job B (although they should be the same data type). Likewise, you are not restricted to say that parameters must be equal – the following operators are supported:

- =
- >
- <
- >=
- <=

To manage parallel and parameter based jobs dependencies click on the 'Dependencies ...' button in the job detail section.

The following panel will appear:



The top section of the screen allows you to add or remove jobs upon which the current job will be dependent on.

Once a dependent job is added you have two choices as to the type of dependency you wish to create:

- 1) **Parameter based dependency.** Link parameters from the current job (Add and Remove buttons on the left hand side of the screen) to parameters from the selected dependent job (Add and Remove buttons on the right hand side of the screen). Select an appropriate condition for the dependency. Only when all the parameter dependency conditions evaluate to true will the current job be allowed to begin execution. If conditions are not met the job will enter a WAITING status until such time as conditions are met; only then will the job run and the status move to RUNNING.
- 2) **Parallel dependency.** To ensure the current job does not run if the dependent job is in a RUNNING state, simply add the dependent job without adding any parameters. If the dependent job is RUNNING when the current job launches it will enter a WAITING state and only move to RUNNING once the dependent job either succeeds or fails. If the dependency is in place for resource conflict purposes it is advisable to add a corresponding parallel dependency on the dependent job back to the current job. Cronology will ask if you would like to automatically add this reciprocal dependency if it is not already in place.

7 Executing Jobs Manually

If you wish to run the currently selected job manually, click on the 'Execute ...' button in the job details section. The following panel will appear:

The screenshot shows a dialog box titled "Execute Job - Sample Shell Script Job". It contains a table of parameters and a section for execution options.

Parameter Name	Type	Data Type	Value	Override Value
Daily Batch ID	FIXED	NUMBER	1	
22	FIXED	CHARACTER	22	

Execution Options:

- Use Override Parameters:
- Run Stand Alone:
- Later: [Calendar icon]

Buttons: Execute, Cancel

Execution options available:

- Use Override Parameters (UOP) – if you wish to run the job with parameters values other than the standard values then check this box and specify any Override Values in the 'Job Parameters' section.
- Run Stand Alone (RSA) – if this check box is selected then no child jobs will be spawned upon completion of the job.
- Later – if the Job Poller is running you can specify a time in the future to run the job (for TIMED jobs however this time must not be later than the Next Submit Date Time).

N.B. Executing a job manually does NOT affect the scheduled run times of the job i.e. the Next Submit Date Time will NOT be affected by running a job manually.

8 Schedule Maintenance

8.1 Manage

A number of functions exist to manage the schedule. Select Main Menu -> Schedule -> Manage.

8.1.1 Export

Select the Export tab to export the schedule. Two options exist for exporting the entire job schedule:

- **Documentation** – choose this option to create a CSV file of the schedule. This file contains the following columns:
 - Job Name
 - Priority
 - Source Type
 - Source Directory
 - Source Name
 - Frequency
 - Next Run Date Time
 - Description
 - DB Username
 - DB Connect String
 - DB Connect As
 - SSH Connection
 - Overdue Behaviour
 - Average Duration
- **Release Script** – use this option if you wish to create a SQL*Plus script that will recreate the entire schedule. This is useful if you are cloning environments or want to include the creation of the schedule as part of a project release. When using this option you will be asked if you wish to use any existing passwords for jobs or be prompted when the script is running (the script can always be edited to suit your needs later).

8.1.2 Suspend

If for any reason you need to suspend all job processing you can suspend the entire schedule. Suspending the schedule differs from stopping the Job Poller as stopping the poller only stops new TIMED jobs from being launched; any child jobs of parents that succeed / any dependent jobs will still launch. Suspending the schedule prevents any dependent jobs from launching – instead they will enter a WAITING state until the schedule is unsususpended.

N.B. suspending the schedule does not kill any jobs that are **currently** running – it merely stops any subsequent jobs (timed or dependent) from running.

8.1.3 Advance

The entire schedule can be advanced to start at a given date and time in the future. For example, say you want to skip all this week's processing and have the schedule start again next Monday, enter the date and time you want the next job in the schedule to run. Cronology will then advance all jobs "Next Submit Date Time" attributes according to their frequencies such that they conform to the new schedule start date and time.

8.1.4 Audit

Cronology has a built in auditing which can be enabled or disabled by an administrator. When enabled, if a user makes any changes to the schedule (e.g. create or edit jobs) the changes are logged in the database. Audit history can be viewed in two ways:

- **Individual jobs and parameters:** an Audit History button is available on the respective Edit screens
- **Entire schedule:** select Main Menu -> Schedule -> Audit History.

N.B. This settings only affects the auditing of user created Jobs and Parameters. Schedule management and changes to schedule settings e.g. changes to messaging options, are **always** audited.

8.1.5 Console Access

Administrators can control who has access to the Cronology Console and with which privileges. This tab by default shows ALL the users on the database. An administrator can then edit the required user to either add, change or remove console access.

The Cronology Console does **not** create database users – that task needs to be performed by your DBA; once users exist their access can then be maintained via this screen.

8.2 Settings

To access the settings for the schedule select Main Menu -> Schedule -> Settings.

8.2.1 Environment Script

Optionally specify an environment script to be called each time a job is launched locally (i.e. no SSH Server Connection is specified for the job). This script can be used to define application specific environment variables. Variables are visible to all local jobs and may be used in job source directory definitions as well as job parameters. The script should be coded using Bourne shell (/sh) conventions i.e.

```
VARIABLE=value  
export VARIABLE
```

8.2.2 SSH Environment Script

Optionally specify an environment script to be called each time a job is launched remotely (i.e. SSH Server Connection is specified for the job). The script can be used to define application specific environment variables on remote servers. Please note:

The script may differ in content from server to server, but it must have the same name and reside in the same location on each server

Environment variables may be used in job source directory definitions as well as job parameters. The script should be coded using Bourne shell (/sh) conventions i.e.

```
VARIABLE=value  
export VARIABLE
```

8.2.3 Job Retention History

Specify a number of months after which the CRONOLOGY.API.DELETE_HISTORY stored procedure will purge historical job data. To automatically purge job run history data, schedule a Stored Procedure job within the schedule to call this procedure at a time and frequency suitable to your system. The database user running this job must have either a DBA, Cronology Administrator or Cronology Operator role assigned.

8.2.4 Job Run Log Limit

The maximum number of log lines that will be captured for each individual job run. This can be used to prevent 'run away' processes from producing excessive amounts of logging and filling up the logging tables.

8.2.5 Console Environment Indicators

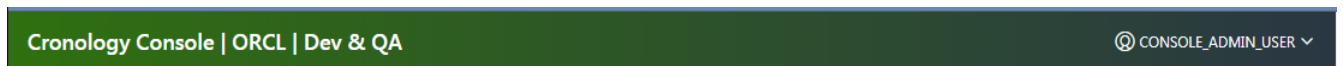
It is often useful to label or colour code UIs so that users are reminded of the environment they are working on, development, QA, production etc.

Optionally specify an environment label and colour indicator for the Cronology Console. The label and colour will be applied to the title bar of the console e.g.

No environment indicators:



A label of "Dev & QA" with colour green selected:



Please note, for consistency, indicators are not user specific and apply to all console users of the schedule. Once saved users will need to log off / on or refresh their browser page (F5) for changes to take effect.

8.2.6 Console Auto Text

When resolving failed jobs, users are required to update the Status Information in the Latest Execution Information section on the main screen. By double clicking the Status Information field a list of common / useful phrases (auto text) are available that can be quickly appended.

Use this tab to maintain or customise the "auto text" that is available to console users.

8.3 Global Update

To access the Global Update facility for the schedule select Main Menu -> Schedule -> Global Update.

This screen, as the name suggests, allows you to update job attributes in bulk across the entire schedule. Four tabs are available:

8.3.1 E-mail Recipients

Over time jobs will be set up with custom e-mail recipients (On Completion / On Failure). When individuals move on or change roles it would be very time consuming to examine each job and maintain the e-mail recipient lists individually. This screen allows you to remove or replace specific e-mail addresses from all jobs in the schedule. There is also the facility to remove **ALL** e-mail addresses from all jobs. This might be useful if the schedule has been imported from a live environment into a test environment and the live users should not receive e-mails from the test system.

8.3.2 DB Password

If a DB password for a specific DB user and connect string combination needs updating, this screen can be used to globally update all jobs using that DB user and connect string.

8.3.3 DB Connect String

If a DB connect string needs updating, this screen can be used to globally update all jobs using the old connect string.

8.3.4 SSH Connection

If an SSH Connection needs updating, this screen can be used to globally update all jobs using the old SSH Connection.

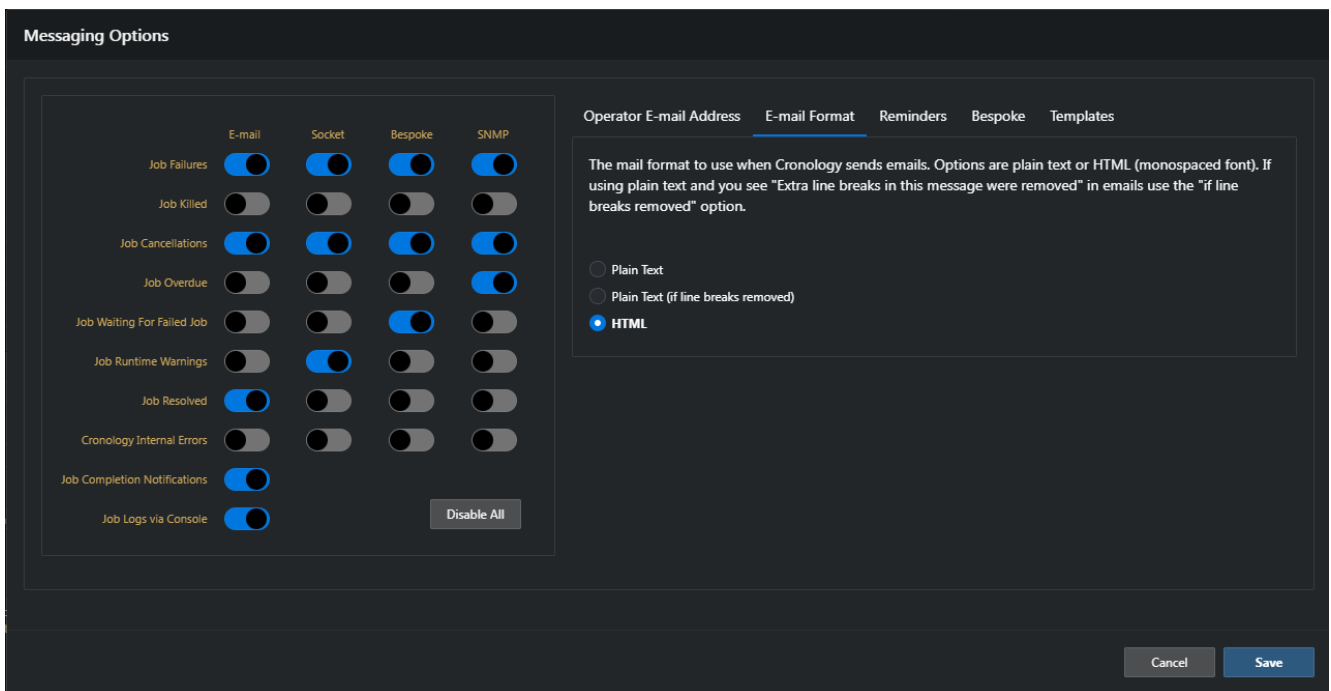
8.4 Messaging Options

To access the Messaging Options for the schedule select Main Menu -> Schedule -> Messaging Options.

Cronology offers four types of messaging:

Messaging Type	Description	Prerequisite to Use	Prerequisite – Who?
E-mail	In plain text or HTML format	1. Configure settings via the supplied PL/SQL procedure: CRONOLOGY.ADMIN.SET_SMTP_SETTINGS 2. Specify an Operator E-mail Address	DBA Console Administrator
Socket	Simple TCP/IP messaging to a predefined server and port	1. Configure settings via the supplied PL/SQL procedure: CRONOLOGY.ADMIN.SET_SOCKET_SETTINGS	DBA
Bespoke	A bespoke Oracle PL/SQL procedure can be specified to consume a message produced by Cronology and process it in any way your organisation requires	1. Create a PL/SQL procedure in the database that accepts a single VARCHAR2 input parameter. The supplied parameter will be the message generated by Cronology, this can be processed however your organisation requires	DB developer
SNMP	SNMP Trap messaging	1. Configure settings via the supplied PL/SQL procedure: CRONOLOGY.ADMIN.SET_SNMP_EXE_DIR 2. Configure message template(s) for SNMP Trap messaging	DBA DBA / Console Administrator

Cronology messaging options are highly configurable. Messages for each type of event (job failure, runtime warnings etc.) can be enabled or disabled for each messaging type:



8.4.1 Operator E-mail Address

Specify an e-mail address for the application operators. If e-mail messaging is enabled for events this is the default address that will be used. Only job specific e-mail overrides (if configured) will override this.

8.4.2 E-mail Format

Choose a format for all e-mails. Options are plain text or HTML (monospaced font). If using plain text and you see "Extra line breaks in this message were removed" in emails received, then use the "if line breaks removed" option.

8.4.3 Reminders

Cronology can be configured to send reminder messages at a repeating interval for the following events:

- Unresolved job failures
- Runtime warnings

For example, in the above screen shot, if a job remains in a FAILED state **and is not resolved** after 2 hours, a repeat message will be sent out. This will happen every 2 hours within the specified window – 8am to 6pm, Monday-Friday.

Some prerequisites from reminders to be sent:

- Messaging must be enabled for either Job Failures or Job Runtime Warnings
- The Job Poller must be running

8.4.4 Bespoke

Use this tab to specify an Oracle PL/SQL procedure in the database that accepts a single VARCHAR2 input parameter. The supplied parameter will be the message generated by Cronology, this can be processed however your organisation requires. A test button is available to invoke the procedure for testing purposes.

8.4.5 Templates

Templates are only applicable to Socket, Bespoke and SNMP Trap messaging. They are optional for Socket and Bespoke messaging, but mandatory for SNMP Trap messaging. Templates allow you to define custom messages, by combining your own text with Cronology substitution variables.

Cronology uses the following precedence to determine message content:

- 1) message event specific template (e.g. Failed or Runtime Warning)
- 2) default template
- 3) standard message (no templates defined i.e. all templates are set to <NONE>)

When defining templates the following substitution variables may be used – the variables must be specified in upper case and the # characters must be included in the template for the substitution to take place:

#JOB_NAME#	The job name that triggered the message
#STATUS#	The status of the triggering event, status will be either: 'KILLED', 'OVERDUE', 'CANCELLED', 'FAILED', 'WAITING', 'RESOLVED', 'RUNTIME' (job has exceeded runtime warning threshold), 'INTERNAL' (internal Cronology error)
#SUBJECT#	The default / standard Cronology message
#INSTANCE_NAME#	The Oracle instance name Cronology is running on
#HOST_NAME#	The host name Cronology is running on
#INFO#	Additional information Cronology may provide with the message
#SYSDATE_<DATE FORMAT>#	The current date and time, please provide a valid Oracle date format e.g. #SYSDATE_DD-MON-YYYY HH24:MI:SS#
#USER_NAME#	The database user who initiated the message (only really useful for job kill, cancellation and resolved messages)
#JOB_PRIORITY#	The priority of the job that triggered the message
#JOB_MESSAGE_TEXT#	The supplemental messaging information defined for the job

SNMP Trap messaging requires the template to contain all the **command line options** that would normally be passed to the snmptrap (snmp_trapsnd for Tru64) UNIX executable, e.g.

For example, in its most basic form, a trap could be invoked with the following options:

```
-v 1 -ci <SNMP_Server_Host> <Sender_OID> <Sender_Host> <Trap_Type> <Specific_Type>
'' <OID> s "<Content1>" <OID> s "<Content2>" <OID> s "<Content3>"
```

So the template may look like (with greatly simplified OIDs):

```
-v 1 -ci 192.168.1.21 1.2.3.4 #HOST_NAME# 6 1 '' 1.1 s "Job: #JOB_NAME# Status:
#STATUS# Timestamp: #SYSDATE_DD-MON-YYYY HH24:MI:SS#" 1.2 s "#JOB_PRIORITY#"
```

8.5 Audit History

To access the Audit History of the schedule select Main Menu -> Schedule -> Audit History. This screen shows the audit history of the entire schedule. As well as Job and Parameter changes you will also see type “Schedule Setting” showing the audit history of the Cronology schedule settings.

Filtering and ordering options are available by clicking on the column headers. Audit entries are stored in JSON format, for update operations both a “before” and “after” image is stored.

Click the download button to download the history in CSV or HTML format.

9 Background Processes

Cronology has three background process that may or may not be running depending on the environment. Each can be controlled from Main Menu -> Background Processes.

9.1 Job Poller

The Job Poller is responsible for launching all timed jobs in the schedule. It polls the schedule once a minute to look for jobs that have become due for execution and then launches them. The Job Poller would obviously be running in a production environment, but may not be running on for example, a development database.

To start the Job Poller click the Start Job Poller button. Please note that if jobs have become overdue a warning will display advising that overdue jobs will run immediately and may run multiple to times to catch up on missed executions. If you do not want overdue jobs to execute either:

- Advance the schedule before starting the Job Poller (Main Menu -> Schedule -> Manage -> Advance) or
- Set the individual jobs "If Overdue" behaviour to "Advance"

9.1.1 Stopping the Job Poller

Two options are available:

- **Immediate:** click the Stop Job Poller button.
- **Delayed:** Switch the Delayed Stop option to the ON position and select a date and time you wish the poller to stop, then click the Stop Job Poller button. The Job Poller RUNNING status changes to light green to indicate a delayed stop has been requested

9.1.2 Autostart

By setting Autostart to ON the Job Poller will automatically start each time the database is restarted.

If a database has been shut down (with Autostart left ON) and you not wish the Job Poller to automatically start (and hence possibly launch jobs) when the database starts up, one of the following two methods may be employed **by your DBA**

Method 1:

- startup mount;
- execute `dbms_application_info.set_module('CRONOLOGY','DISABLE_AUTOSTART');`
- alter database open;

The Job Poller will detect this setting when starting up and immediately shut itself down - it will also set the Autostart setting to OFF so that subsequent database startups will not start the Job Poller. Autostart can be turned on again if required via the console.

Method 2:

- startup mount;
- alter system set "_system_trig_enabled" = FALSE;
- alter database open;

This method will disable the database system trigger that starts the Job Poller, **it does not set the Autostart setting to OFF, so Autostart should be turned off via the console if required.**

WARNING: ALL database startup triggers will be disabled if method 2 is used.

9.2 Log Writer

The Log Writer performs a number of functions, the main one being capturing all job output and writing it to the database. It also acts a conduit when viewing source code, so you will notice the Log Writer may start just by viewing the source code of a job. The Log Writer always starts automatically when needed, so there is no option to manually start the Log Writer.

Under normal operation the Log Writer will start up the first time a job is run, it will then remain running until either the database is shut down or an administrator stops it via the console.

If problems are suspected, a diagnostic test can be run to 'ping' the Log Writer to check its performance. Performing this action sends a message to Log Writer and waits for a response. A healthy Log Writer should return a response almost instantly.

If the Log Writer for any reason receives input for which it cannot identify a running job (i.e. it does not know which job log to write to) it will classify these lines of input as 'lost'. Lost log lines can be viewed via this screen. **On a healthy system there should be NO lost log lines.**

9.3 Message Server

The Message Server is responsible for servicing all messaging. When the schedule requires any messages to be sent (be they e-mails, socket, bespoke or SNMP trap messages) they are put on a message queue so they can be dealt with asynchronously without holding up schedule processing. The Message Server reads messages off the queue and actions the messages.

Like the Log Writer, the Message Server starts on demand, there is therefore no manual option to start the Message Server.

If you are experiencing messaging issues (non delivery etc.) use the Message History button to inspect the full message history. You can filter the results using the search bar at the top of the display. Enter a search string to filter across all the displayed columns e.g. enter an email address to filter for all messages sent to a specific address, or filter for 'error' to filter on all errored messages. Clicking on a row in the history displays a pivoted view of the row which can be easily copied to the clipboard.

10 Status Summary and Job Analysis

10.1 Status Summary

The Status Summary dialog (Main Menu -> Status Summary) provides an overview of the all the jobs in the schedule counted by status.

Click on a row to filter the schedule by the selected status. Clear the filter by double clicking on the Status column heading in the main display.

10.2 Job Analysis

The Job Analysis screen (Main Menu -> Job Analysis) allows you to query the run history of the schedule. From and to dates can be specified (or leave blank to indicate no boundary).

10.2.1 Statistics

Select the Statistics radio button so display a list of all the jobs in the schedule together with statistics of the total number of executions, the number of complete, failed, killed and cancelled executions. Actuals as well as percentages are provided.

Data can be sorted by clicking on the column headings. Data can be downloaded in CSV or HTML format.

10.2.2 Activity

Select the Activity radio button to display an activity list i.e. all the jobs that executed within the specified time range.

Data can be sorted by clicking on the column headings. Data can be downloaded in CSV or HTML format.

11 Current SQL and Database Sessions

11.1 Current SQL

On the main screen, when a job is running the Current SQL button is enabled. This allows the user to inspect any current SQL the job may be running. Explain plan and SQL monitoring options are also available.

11.2 Database Sessions

The Database Sessions screen (Main Menu -> Database Sessions) provides details of both the current user connections to the database and the users with active Cronology Console sessions.

The database sessions screen highlights in blue any Cronology related sessions. It also provides information on any current SQL the session may be running. Explain plan and SQL monitoring options are also available.

Appendix A - Cronology API Package

The API package owned by Cronology (CRONOLOGY.API) has execute privilege granted to PUBLIC and provides a number of procedures that application developers may use in their code to interface with the Cronology Server processes:

- **PROCEDURE LOG_LINE**

This procedure may be called from within a PL/SQL program unit to allow real-time output to the job log when the job is run via Cronology. Traditionally DBMS_OUTPUT would be used to produce output from within PL/SQL, this however waits until the program unit has completed before producing the output. Programmers may use CRONOLOGY.API.LOG_LINE where they would normally use DBMS_OUTPUT.PUT_LINE. If the program unit is run outside of Cronology then LOG_LINE acts exactly the same as DBMS_OUTPUT.PUT_LINE. An option parameter (P_MODE) exists for the procedure. If LOG_LINE is called with P_MODE = 'A' then the line will be appended to previous line.

- **PROCEDURE EXECUTE_JOB**

This procedure will execute a job in the Cronology schedule. This procedure is overloaded to take either a job name (CRONOLOGY.JOB table JOB_NAME column) or the job id (CRONOLOGY.JOB table JOB_ID column – using job ids is not recommended other than for development or testing purposes as job ids are unlikely to be consistent across your environments). Other parameters: P_RSA = run stand alone (Y or N), P_UOP = use override parameters (Y or N), P_FAIL_ON_ERROR = if the job fails raise an exception (Y or N). This procedure allows jobs to be executed programmatically via Cronology from within PL/SQL program units if needs be.

- **PROCEDURE DELETE_HISTORY**

A procedure named DELETE_HISTORY has been provided in the CRONOLOGY.API package to delete historical job information. This is a recommended Cronology housekeeping routine, as such a job should be created in the schedule to call the CRONOLOGY.API.DELETE_HISTORY procedure at an interval and time that suits resource availability on your system. The procedure deletes data based on the JOB_HISTORY_RETENTION parameter, set via the Console under Main Menu -> Schedule -> Settings -> Job History Retention). **The CRONOLOGY.API package has execute privilege granted to PUBLIC, but internal validation inside the DELETE_HISTORY procedure ensures that only DB users granted the DBA, CRONOLOGY_ADMINISTRATOR or CRONOLOGY_OPERATOR role may execute it. Two optional parameters are available for this procedure:**

P_SHRINK_SPACE	BOOLEAN	defaults to FALSE
P_COMMIT_BATCH_SIZE	NUMBER	defaults to NULL

Shrink space when set to TRUE will compact the space used by the JOB_LOGS table.

Commit batch size will delete job run history rows in batches of this size and commit after each batch. This may be useful if the DELETE_HISTORY procedure is being invoked for the first time and a lot of historical data is to be deleted.

Other procedures exist in the API package are not intended for general use, they are primarily used for supporting the Cronology processes or for use when instructed by Cronology support.