Vitalware®
Vital Records Management

The World's Premier
Vital Records
Software

Vitalware Documentation

Release Notes: Vitalware 2.1.02

Document Version 1

Vitalware Version 2.1.02

KE Software

www.kesoftware.com

Contents

Here you will find collected together the Release Notes for Vitalware 2.1.02, alongside all documents referenced in the notes. These release notes and documents are also available on the KE Vitalware website.

This PDF document brings together a number of individually published documents: please note that page numbering below refers to this combined PDF document and not to the page numbers printed at the bottom of pages, as each individual document, e.g. Scheduled Export Facility, has its own internal numbering:

Release Notes: Vitalware 2.1.02
Release Date: 28 July 2010

Requirements

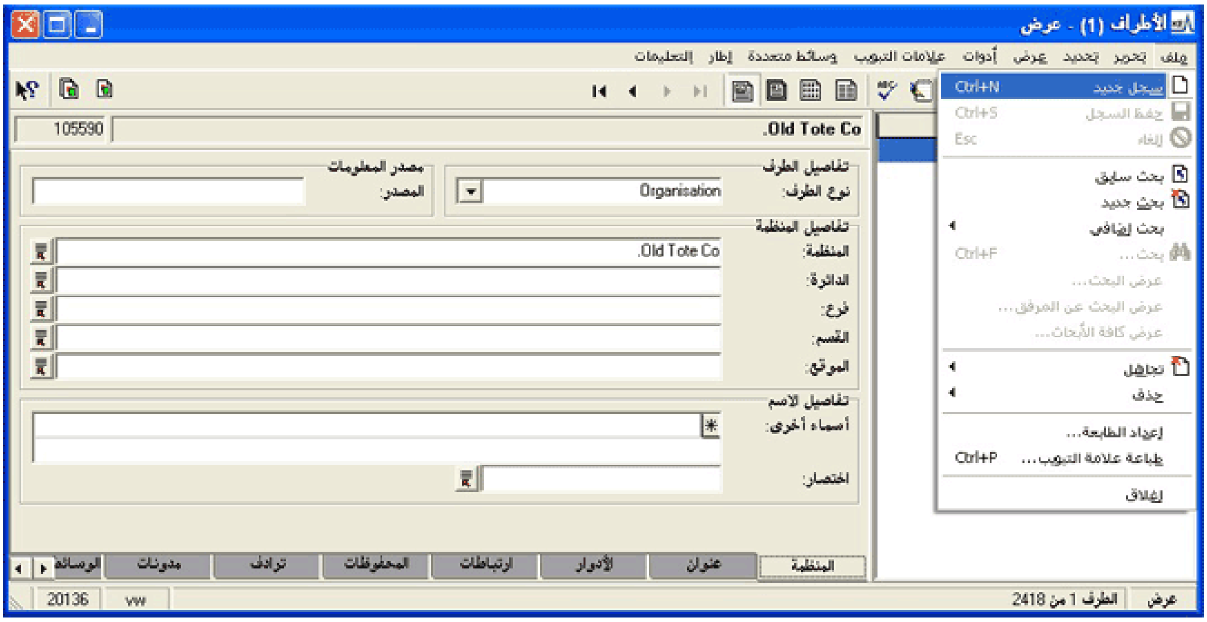
- For Windows 2000, XP, 2003, Microsoft Windows Services for UNIX (version 3.5)
- Texpress 8.2.008 or later. Note Texpress 8.2.008 or later must NOT be used with any Vitalware version pre Vitalware 2.1.02.
- TexAPI 6.0.001 or later
- Perl 5.8 or later

Download

Download a PDF document containing these release notes and associated documents.

New Features

- **Support for RTL languages:** Support for right-to-left (RTL) languages has been added to Vitalware. The user interface has been modified to reverse the display of all controls and images when using RTL languages. Dynamic switching between RTL and LTR languages is supported.



105590 .Old Tote Co

نوع الطرف: Organisation

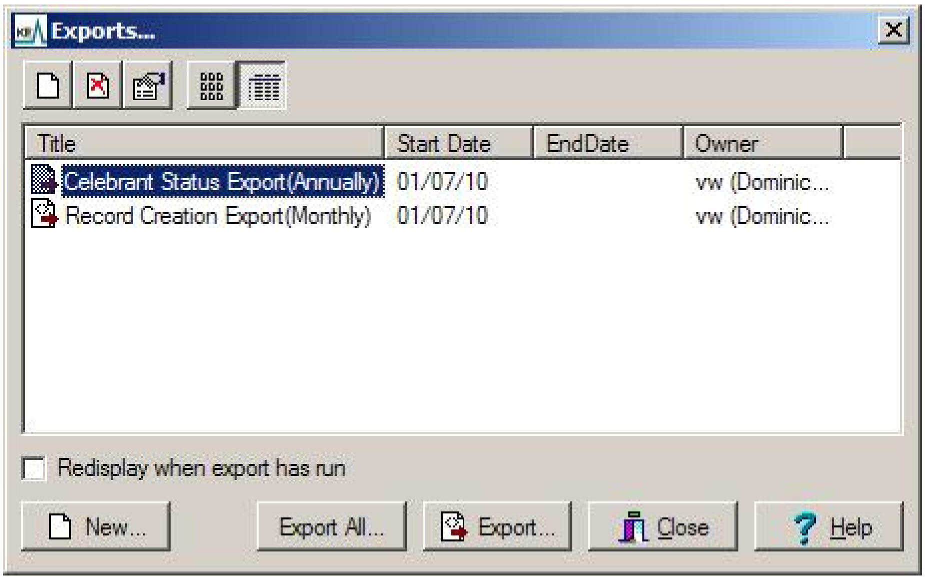
المنظمة: .Old Tote Co

الطرف 1 من 2418

20136 vw

A complete description of the support for RTL languages may be found in the Right To Left Language Support documentation.

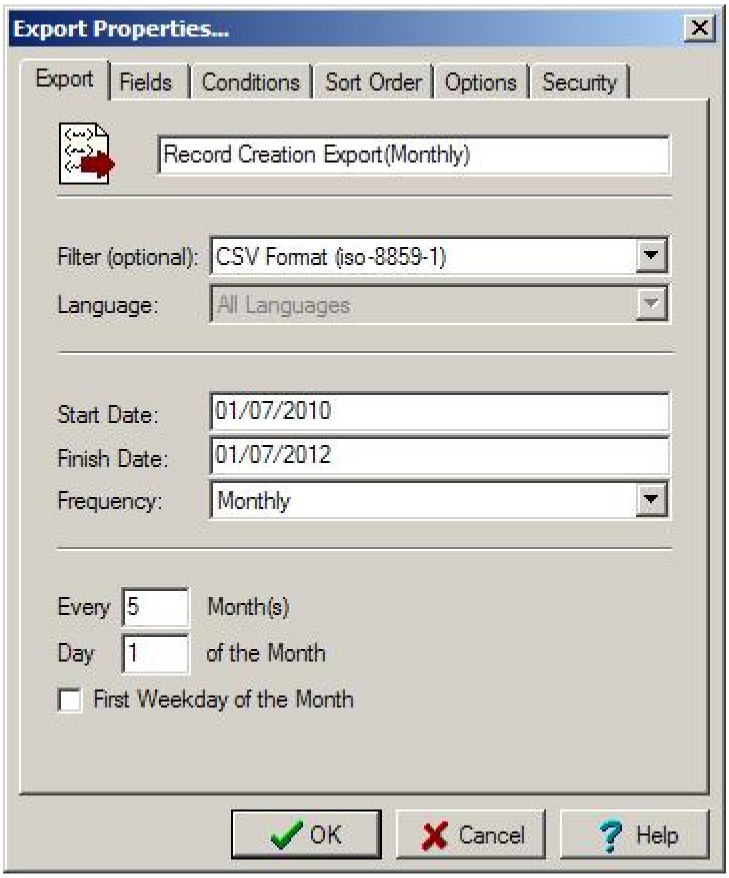
- **Export Schedules Facility:** The Export Schedules facility enables data to be exported on a regular basis. Data can be scheduled to be exported on anything from a daily to an annual basis. The export occurs outside working hours (generally early in the morning) and the results are stored on the Vitalware server. A user may then view the results and save / view the export files produced. The exported data may be processed by a filter to create the output format required (e.g. XML, CSV). The creation of a scheduled export is very similar to defining a report. A set of fields may be selected and the records may be sorted. A TexQL statement may be defined to limit the records exported.



Title	Start Date	EndDate	Owner
Celebrant Status Export(Annually)	01/07/10		vw (Dominic...
Record Creation Export(Monthly)	01/07/10		vw (Dominic...

□ Redisplay when export has run

The Export Properties dialogue box is shown below:



Record Creation Export(Monthly)

Filter (optional): CSV Format (iso-8859-1)

Language: All Languages

Start Date: 01/07/2010

Finish Date: 01/07/2012

Frequency: Monthly

Every 5 Month(s)

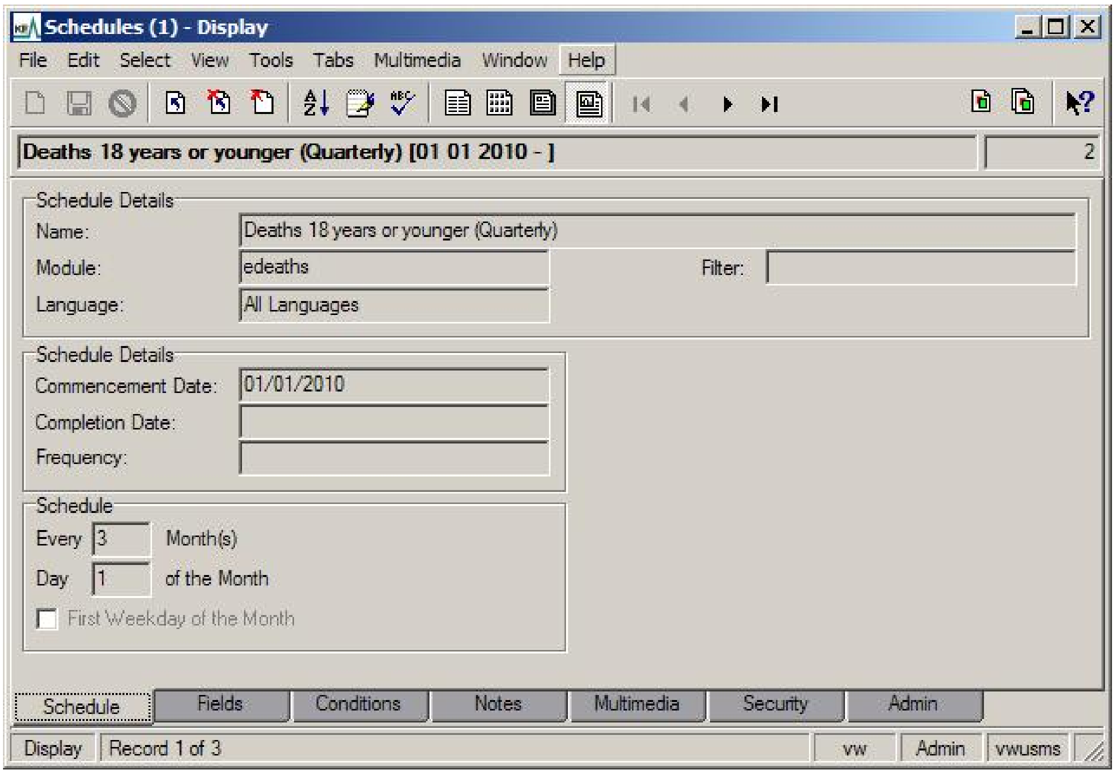
Day 1 of the Month

□ First Weekday of the Month

The Export Schedules facility provides two new modules:

- **Schedules Module**

  The Schedules module holds one record per Scheduled Export containing the definition of the export.


Schedules (1) - Display
File Edit Select View Tools Tabs Multimedia Window Help
Schedule Fields Conditions Notes Multimedia Security Admin

Deaths 18 years or younger (Quarterly) [01 01 2010 -] 2

Schedule Details

Name: Deaths 18 years or younger (Quarterly)

Module: edeaths Filter:

Language: All Languages

Schedule Details

Commencement Date: 01/01/2010

Completion Date:

Frequency:

Schedule

Every 3 Month(s)

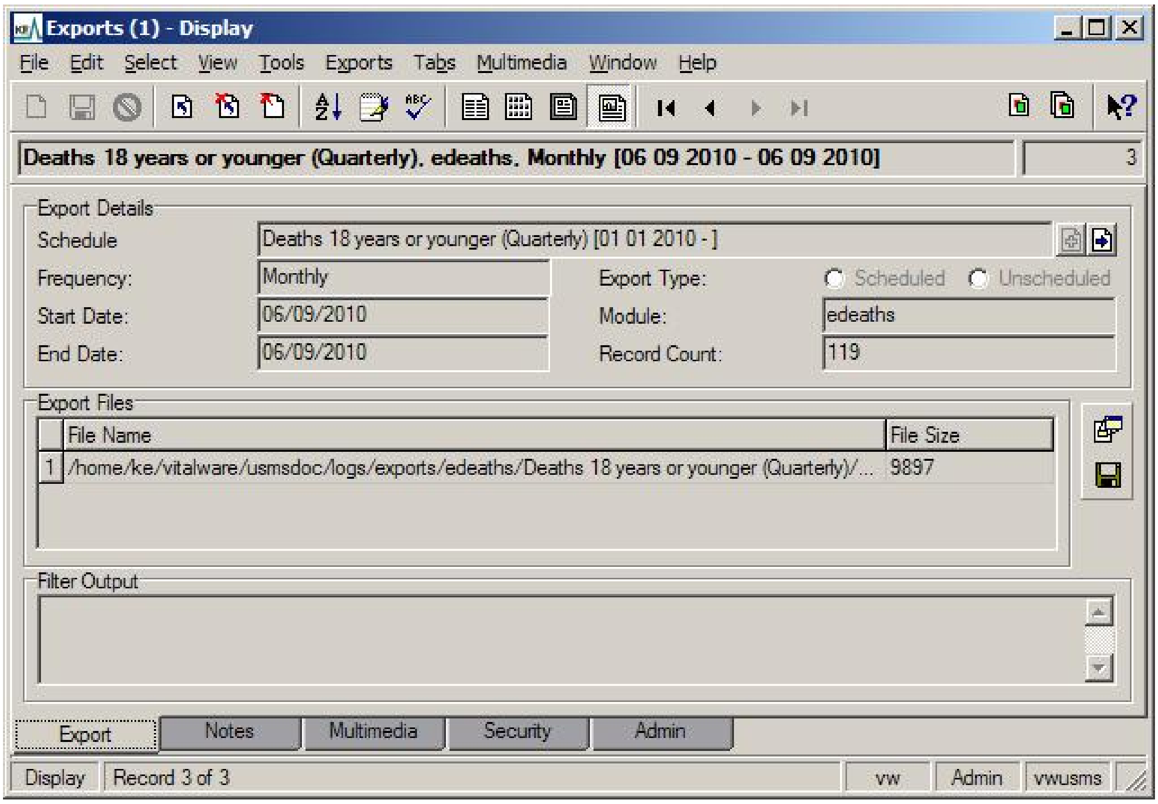
Day 1 of the Month

□ First Weekday of the Month

Display | Record 1 of 3 | vw | Admin | vwusms

- **Exports Module**

  The Exports module holds one record per set of export files produced. When a Scheduled Export is run a new exports record is created containing the output of the export process.


Exports (1) - Display
File Edit Select View Tools Exports Tabs Multimedia Window Help
Export Notes Multimedia Security Admin

Deaths 18 years or younger (Quarterly). edeaths. Monthly [06 09 2010 - 06 09 2010] 3

Export Details

Schedule: Deaths 18 years or younger (Quarterly) [01 01 2010 -]

Frequency: Monthly Export Type: ○ Scheduled ○ Unscheduled

Start Date: 06/09/2010 Module: edeaths

End Date: 06/09/2010 Record Count: 119

Export Files

	File Name	File Size
1	/home/ke/vitalware/usmsdoc/logs/exports/edeaths/Deaths 18 years or younger (Quarterly)/...	9897

Filter Output

Display | Record 3 of 3 | vw | Admin | vwusms

A complete description of the Export Schedules facility can be found in the Export Schedules documentation.

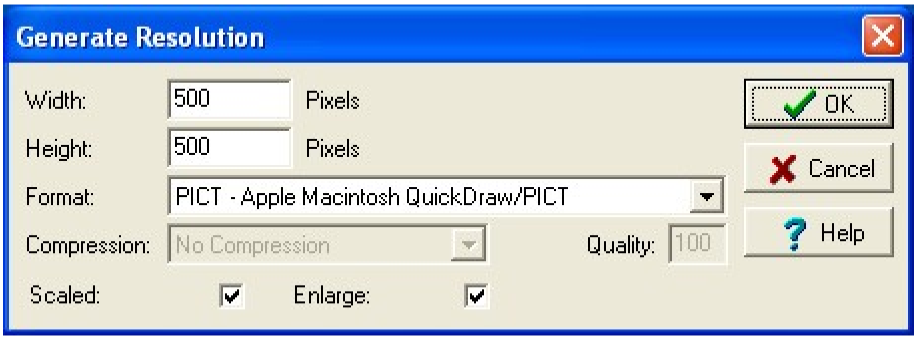
- **Append values using the Import Facility:** The existing Import facility has been extended to allow data to be prepended / replaced / appended in tables and nested tables of values. Support is provided for both XML and CSV (Comma Separated Values) data formats. A grouping mechanism allows data across multiple fields to be added to the same row, allowing grids with multiple columns to have complete rows appended.

  A complete description of the Import facility extensions can be found in the Appending data using the Vitalware Import Facility documentation.

- **Partial Invoice Payments:** The existing invoice payment facility has been extended to allow for partial payment of invoices. The partial payment is assigned sequentially against invoiced orders until the partial payment amount has been consumed.

Improvements

- **Improved time to save records:** A number of optimisations have been made to the Vitalware database server (Texpress) leading to significantly reduced record save times. In particular the speed of calculated values and lookup of data in other modules has been enhanced. The improvements affect all operations that alter data, including:
  - Global edits
  - Data imports
  - Bulk update tools (e.g. locations, condition checks, etc.)
  - Single record insertions / edits
- **Alpha channel support in images:** The generation of thumbnail and other resolution images has been enhanced to ensure the alpha channel (or transparency plane) is maintained in generated images. For image formats that do not support transparency (e.g. JPEG), the colour defined in the alpha channel is used as the background colour for derived images.
- **Improved attaching and importing of multimedia:** When adding multimedia, either by attaching an image to a Multimedia module record or by importing a set of images, a link is created to the original image. The link is used to generate the thumbnail and resolutions for the image. Using a link removes the need to copy an image into the Vitalware cache before saving it to the Vitalware server, hence saving time and disk space on the local computer.
- **Generate Resolution settings saved:** The settings of the Generate Resolution dialogue box found in the Multimedia module are now saved between invocations and across sessions. The last set of values entered is always displayed when the dialogue box is next displayed.



Generate Resolution

Width: 500 Pixels

Height: 500 Pixels

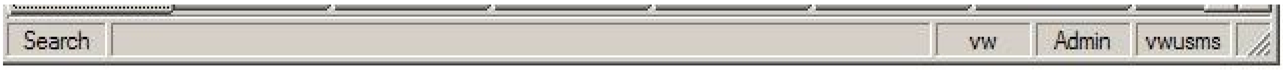
Format: PICT - Apple Macintosh QuickDraw/PICT

Compression: No Compression

Quality: 100

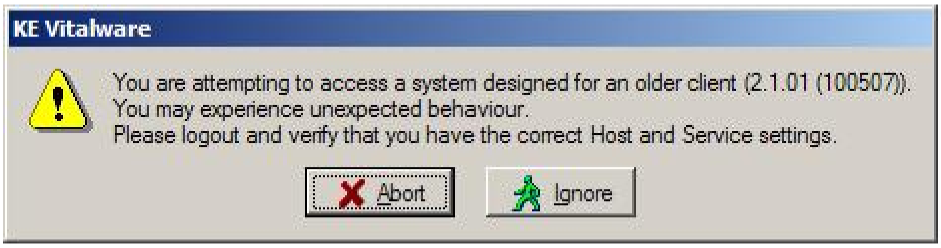
Scaled: ☑ Enlarge: ☑

- **Option to suppress module compaction:** The size of the Audit and Statistics modules will grow over time as more records are added to them. Since both modules have a very small / non-existent number of record changes it is not necessary to compact these tables as part of the weekly maintenance schedule. A new option `COMPACT="no"` may be added to a module's `vwoptions` file (stored in the module directory on the Vitalware server) to skip the compaction of the module when maintenance is performed.
- **User's group added to status bar:** The group to which the current user belongs has been added to the Vitalware status bar. The status bar now displays the user name, group and service used by the Vitalware client, along with the current mode and any messages.



Search | vw | Admin | vwusms

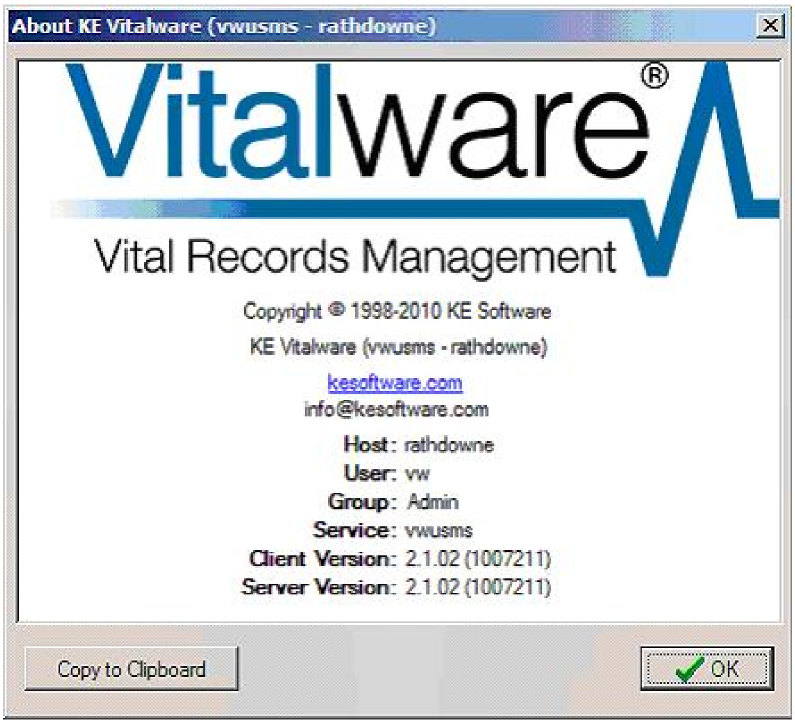
- **Abort connection to incorrect client:** The Vitalware login process has been enhanced to allow the login process to be terminated and the login dialogue redisplayed if an incorrect service is specified. The enhancement is useful for clients who run multiple clients on the one Vitalware server.



KE Vitalware

You are attempting to access a system designed for an older client (2.1.01 (100507)).
You may experience unexpected behaviour.
Please logout and verify that you have the correct Host and Service settings.

- **About box information:** A *Copy to Clipboard* button has been added to the Vitalware About dialogue box. The button is used to place a copy of the Vitalware connection and settings information onto the Windows clipboard. Once on the clipboard the information may be pasted into emails, etc. to help with issue diagnosis.



About KE Vitalware (vwusms - rathdowne)

Vitalware®

Vital Records Management

Copyright © 1998-2010 KE Software

KE Vitalware (vwusms - rathdowne)

kesoftware.com
info@kesoftware.com

Host: rathdowne
User: vw
Group: Admin
Service: vwusms
Client Version: 2.1.02 (1007211)
Server Version: 2.1.02 (1007211)

A sample of the information generated is displayed below:

```
[Client Information]

Client Version: 2.1.02 (1007211)

Application Name: KE Vitalware

Application Path: C:\Projects\Vitalware\Clients\USMS\vitalware.exe

Cache Path: C:\Documents and Settings\vibha\Local
Settings\Application Data\KESoftware\Cache
```

```
[Server Information]

Server Version: 2.1.02 (1007211)

Server Path: /home/ke/vitalware/usms

Server Media Path: /home/ke/vitalware/usms/multimedia


[Texpress Information]

Texpress Version: 8.2.8

TexAPI Version: 6.0.001

Uname: SunOS rathdowne 5.10 Generic_137138-09 i86pc i386 i86pc
Solaris


[Connection Information]

Host: rathdowne

User: vw

Group: Admin

Service: vwusms


[Language Information]

Prompt Language: 0

Data Language: -1

Language List: 0

Language Delimiter: ;:;
```

- **Improved auditserver processing:** The Vitalware auditserver facility handles the processing of all audit trail information. If a large number of records is modified in a small period of time, the server may not be able to process records at the rate at which they are created. In this case it falls behind and catches up when the system load is lighter. The auditserver has been optimised to allow for faster processing of audit trail records. It is about ten times faster than previous version. To take advantage of this improvement the Vitalware server must have the `XML::Parser::Expat` and `File::FcntlLock perl` packages installed.
- **Audit operation statistics gathered weekly and monthly:** The audit operations statistics (for every user in every table for every operation and the number of times the user has performed the operation) are now gathered on a weekly and monthly basis as well as the previous daily basis. The new statistics allow weekly and monthly reports to be produced detailing the operations performed by each user on all modules.

- **Improved Stock display:** The Vitalware registration modules contain the Stock tab for showing all certificates issued from the current version of the registration. An additional grid has been added to show all certificates issued from historic versions of the registration.
- **POS display:** The registration modules have a new POS tab to show the POS records that the registration has been used in.
- **Voiding previously issued stock:** The maintenance process in Vitalware creates a new copy of the current record. Each maintenance is usually associated with a legal change for a record and as such usually requires a new certificate to be issued. A new Registry entry has been added for systems that perform certificate verifications (`System|Setting|Cancel Certificate Status|Value`) so that previously issued certificates for the registration may be cancelled. When this entry is set, the text contained in the Value part of the entry is added to certificate stock status field (e.g. `Cancelled`) and the remaining stock void fields in the certificates module are completed.

Issues Resolved

Issue	Resolution
If an existing Multimedia record has a new file attached to it with the same name as an existing file but with a different file extension, then both the old and new files will be removed when the record is saved.	The old file is now removed and the new file kept in the Multimedia repository.
The copying and pasting of records from List mode may copy empty virtual fields, even if they contain values, where some of the records selected do not contain attachments.	The contents of all virtual fields are now copied correctly when records are copied from List mode.
The **fifoserver** may generate an error where the data returned contains a format specifier (e.g. `%n`). In this case an empty value is returned rather than the correct value.	The correct data is returned from the **fifoserver** even when the data contains a format specifier.
The keyboard shortcut, `CTRL+SHIFT+END` does not select all records from the current record to the last record in List and Contact Sheet modes. Only the current and last records are selected.	All records from the current record to the last record are now selected.
If a record has multiple attachments on the Multimedia tab and one of the attached records does not have any media associated with it (that is, a Multimedia record without any file, URL or reference), the thumbnail of the previous attachment will be displayed for the empty attachment rather than a blank thumbnail.	A blank thumbnail is now displayed where an attached Multimedia record does not contain any file, URL or reference.
The attachment of an image containing invalid EXIF Interop data in the Multimedia module may result in the client hanging.	The EXIF parsing has had more checks added to ensure that invalid EXIF tags do not cause the client to hang.
The Extended data is not displayed in the Groups module when thumbnails are displayed in Details mode.	The Extended data is now displayed correctly.
The attach media drop down button on the Multimedia tab in the Multimedia module may appear as a single button rather than a drop down list containing File, URL and Reference for clients who have modified the standard Multimedia module (that is, sub-classed it).	The attach media drop down button now displays correctly.
The contents of a multi-column grid may lose their row associations where the grid contains more than one virtual column for the same attachment. The row associations are affected by the Move, Sort, Delete, Insert, Clear, Paste and Insert	The row associations are maintained correctly for all row based operations.

Issue	Resolution
Paste operations available when right-clicking the grid. The issue may also occur when using drag and drop to rearrange rows within the grid.	
The **auditserver** may produce duplicate audit records if the previous instance of the server was shutdown at an "inopportune" time while processing data. The problem only arises where the **auditserver** was processing records and the offset of the records processed was not saved correctly when terminated.	Duplicate records are no longer produced.
The dialogue box displayed when generating a crystal report that prompts for a parameter may display broken images for the OK and Cancel buttons. The only means of closing the dialogue box is to click on the X in the top left of the form.	The OK and Cancel buttons now display correctly.
The error **Column operation performed before row has been accessed** may appear when deleting a Multimedia record from the Multimedia repository.	The error no longer appears when deleting a record.
A **Grid Range Index** error may appear when dragging and dropping rows in a grid where the dragged row is empty and it is dropped on the Insertion row at the bottom of the grid.	The error no longer appears when dragging empty rows in a grid to the Insertion row.
The error **Column operation performed before row has been accessed** may appear after sorting records in either List or Contact Sheet modes. The error occurs sporadically.	The error no longer appears after sorting records.
The last record displayed in List mode may be duplicated in the grid. The records retrieved are correct but not displayed correctly. The issue only arises if a false match is found in the matching records (very rare).	The correct data is displayed for all records in List mode, even if false matches occur.
The username and service displayed in the status bar at the bottom of the client form may appear centred in the bar rather than on the right hand side. The issue only occurs if Shortcuts is enabled.	The username and service are always displayed on the right hand side of the status bar.
The form window may appear truncated on the right when opening a module that was last viewed maximised and the *Save Last Position* and *Save Last Size* options are enabled.	The module form is now displayed correctly.
The Modified time on the Reports Properties dialogue box may be an hour ahead of the current time. The issue only occurs when the client computer does not have daylight savings enabled.	The correct Modified time is displayed regardless of the daylight setting on the client computer.

Issue	Resolution
The red background colour used by the Column Colour Registry entry, e.g.: `Group|Default|Table|eparties|Column Colour|NamLast|red` will appear as a white background on screens with 16 bits or less colour depth.	The red colour has been adjusted to still appear red on screens with 16 bits or less colour depth.
The Caption in a module form may have duplicate mode indicators appended to the module name, e.g. `Parties (1) - Search (1) - Search` The issue arises when switching between Prompt languages where the Caption for the module form has not been translated.	The correct module form Caption is now displayed.
The default Page View displays all prompts in English regardless of the Prompt language selected.	Language specific prompts are now displayed based on the Prompt language selected.
The **auditserver** may produce duplicate audit records if the previous instance was terminated while copying audit XML records. The copying only occurs after all existing records are processed, so the issue arises rarely.	Duplicate audit records are no longer produced if **auditserver** is terminated while copying audit XML records.
The Security, Audit and Admin tabs may not be displayed in the correct order for clients who use a modified version of the Multimedia module.	The Security, Audit and Admin tabs are now displayed in the correct order.
The font used to display data may not be set correctly where the selected font does not include a character for the current code page. In particular single value fields may exhibit the problem.	Only fonts containing characters for the current code page may be selected.
The error **Column operation performed before row has been accessed** may occur if the first matching record is discarded while in Details mode. The same error may occur if you choose to delete the record.	The error no longer occurs when discarding / deleting the first record in Details mode.
The weekly and monthly statistics generated for operation by table by user may produce incorrect values for the number of operations performed.	The correct number of operations is now computed and stored.
The **vwregdelete** utility does not delete entries containing the owner of the Registry entry correctly (e.g. `vw:System|Paths|ServerMediaPath|`). The	The correct Registry entries are deleted based on the userid supplied.

Issue	Resolution
inclusion of the owner is intended to allow the deletion to be limited to entries owned by a given userid.	
The time taken to move between matching records in Details mode may deteriorate the longer a module window is used. The issue occurs if the window is cached and is used often. Each use of the cached form results in slower times moving between records, particularly if the module has a lot of tab switching involved (e.g. Parties module).	The time required to move between records in Details mode no longer deteriorates for cached module forms.
The View Attachments button displayed next to a grid control may not be enabled where a grid contains a single attachment. The issue only arises if the attachment list is computed by the client, rather than being stored in the record (rare).	The View Attachment button is now enabled for a single attachment in a grid control.
The weekly maintenance may fail on modules with a small number of records where the size of the record contents vary greatly (e.g. Groups module). Only the affected module(s) are left off-line, all other modules are operational. The issue was caused by an incorrect configuration being generated.	The weekly maintenance now completes correctly for modules where the record size varies greatly.
The Close button on the Summary dialogue box, displayed after sorting records, may not close the form. The X button in the top right hand corner of the form may be used successfully.	The Close button now functions correctly.
The tool bar may not have the correct length when Shortcuts are enabled and the *Save Last Size* option is disabled. The right hand end of the tool bar will be truncated.	The tool bar now displays correctly.
If saving a record fails because a mandatory field has not been filled, the error message form may be hidden behind the module form. The issue only arises if the module switched tabs to focus on the field that is not filled. In this case, the module form cannot be selected as the hidden error message has focus. It appears the client has frozen.	The mandatory error message is now displayed on top of the module form, even if tabs are switched to display the offending field.
Some video and audio file formats may not be playable within the Vitalware client on Windows Vista and Windows 7. Launching an external viewer will result in the media playing correctly.	All video and audit file formats registered with Vista and Windows 7 now play correctly.
Audit Trail records may be duplicated for clients who installed a pre-release	The pre-release plugin is removed as part of the Vitalware 2.1.02

Issue	Resolution
version of Vitalware 2.1.01 and then upgraded to the official release at a later date. The duplicates occur as the pre-release version installed an audit processing plugin and the official release installed another audit processing plugin. As the pre-release plugin was not removed, each audit record was processed twice.	installation.
The Summary dialogue box may be hidden behind the sorted module where a sort was performed and a summary requested while in Details mode.	The Summary dialogue box is placed on top of the sorted module when displayed.
The Reports Properties dialogue box may not display the name of the report file, its size and modified time.	The correct Report Properties information is now displayed.
The Import Identifier and System Identifier fields are dittoed when either the ditto all or ditto tab command is selected. As these fields relate to the creation of the record via the Import facility they should not be dittoed.	The Import Identifier and System Identifier are no longer dittoed.
In very rare circumstances an active form my become hidden behind an inactive form (e.g. the Save form hidden behind the module form). The issue only occurs if Vitalware is made the active application (via `ALT+TAB`) while a form was being created.	The active form is always placed on top of the inactive form.

A user may be able to see records that should be hidden by Record Level Security (RLS). The issue only occurs where the base security system has been augmented with column specific values via the Security Registry entry, e.g.:
`Group|Staff|`
`Table|ebirths|Security|Display|SecDepartment=Staff`
The same issue may cause the Summary of sorted records to be slow.
	Users can no longer view hidden records where the base security system has been adjusted.

The thumbnail image displayed in Details mode when view Thumbnail is enabled may not appear on the far right side of the form. The Issue only arises if **Save Last Size** and **Save Last Position** are enabled and the module form is not the default size.	The thumbnail image is always displayed on the right hand side of the form.
Multimedia files larger than 2 Gb are not handled correctly. Any file over 2 Gb will result in a zero sized media file being produced.	Multimedia files over 2 Gb are now handled correctly.

Issue	**Resolution**
If a user double clicks on a group name while using the Group Retrieve dialogue box, all records in the module are retrieved, rather than the records in the group. The issue only arises for static groups.	Double clicking on a static group now retrieves the record in the group only.
The Import facility may create multiple versions of the same attachment record where the data contains non-ASCII characters. The issue only occurs for clients who have their base Vitalware language encoding set to UTF-8 (that is `langcode=utf8`).	A single version of the same attachment record is created where the data contains non-ASCII characters.
The error message **AlsoSearch column *column* does not exist** may appear when defining an AlsoSearch Registry entry where the columns to search are virtual columns that refer to the initial AlsoSearch column.	The error message is no longer displayed when the Also Search loop is detected.
If a number of records are attached to a control in Query mode (that is, is displayed in the control), the attached records cannot be cleared by using the `DEL` key or backspacing over the control contents.	The `DEL` and backspace key can now be used to clear attached records in Query mode.
The *Publish on Internet* and *Publish on Intranet* fields have associated flag fields used by some web based searches to check whether a record may be viewed on the web. In some instances the flag fields may be out of sync with the *Publish on* fields (that is, the field has a value of *Yes* and the flag is set to *N*). The field and its associated flag value should always be in sync.	Checks have been added so that when a record is saved the *Publish On* fields and their associated flag values are synchronised to ensure they always reflect the same state.
The **vwlogger** facility allows the standard configuration file (`etc/logger`) to be replaced with a localised version in `etc/local/logger`. A better mechanism would allow the local configuration to override entries in the base file rather than replace the configuration file completely. Such a set up provides a useful means of adding new services without the need to update the local configuration file.	The local configuration file now augments the entries in the base configuration file, rather than replacing them.
The Field Help module does not contain records for reference columns, that is column names ending in Ref, Ref_tab and Ref_nesttab. These columns are not accessible from within the Vitalware client, however the columns may be used for other purposes (e.g. web page design).	Reference fields have been added to the Field Help module.
The Summary and Extended data calculated for the Groups module does not generate values suitable for multilingual systems.	The Summary and Extended data generated is now multilingual.

Issue	Resolution
An error message is displayed when invalid characters are found in EXIF data fields, forcing the EXIF data to be discarded. In most cases it would be better to skip the invalid character(s) and insert the EXIF data.	Invalid characters in EXIF data are converted to a '?' character allowing the data to be extracted rather than discarded.
A very large hierarchy may cause Lookup List searches to deteriorate when retrieving values from the hierarchy. The issue only occurs where the hierarchy contains many levels with a large number of terms at each level.	The Lookup List values for hierarchies are clustered together eliminating the deterioration in search times.
The Image Display Registry entry only looks for images stored in the `etc/images` directory on the Vitalware server. The `local/etc/images` directory should be checked first to allow sites to provide localised images.	The `local/etc/images` directory is checked before `etc/images` for images.
If a number of TWAIN devices are registered on the Vitalware client machine, moving from record to record in the Multimedia module may be slow. Each twain device is queried as each record is displayed causing a long delay if multiple devices are registered.	The Multimedia module now checks for the existence of TWAIN devices when the module is invoked rather than between each record displayed.
The Open and Save buttons on the Reports Properties dialogue box may cause confusion as to their purpose. The buttons should be renamed to better indicate their functionality.	The Open button has been renamed to Download and the Save button renamed to Upload. The Download dialogue box title has been changed to *Download Report to Edit* and the Upload dialogue box title changed to *Update Report to* Vitalware.
Many modules contain date and date ranges in their Summary and Extended data calculations. The formatting of the date(s) depends on the date order (DMY, MDY, YMD) used by the Vitalware server, however in some instances partial dates may not format correctly causing "invalid" dates to be displayed.	Partial dates are now formatted correctly.
When making a payment to a transaction that was under paid and the resulting transaction was still under paid it was possible for orders to not correctly show as Pending Payment.	Orders for under payments correctly display as Pending Payment.
The formatting of the date(s) in Vitalware depends on the date order (DMY, MDY, YMD) used by the Vitalware server, however in some instances the date of event not format correctly in POS causing "invalid" dates to be displayed.	Event dates are correctly displayed in POS.
The back filling of POS event details is controlled by the Query Fields registry entry. Where a nested table was specified as a query field it was possible for	The correct event data is back filled.

Issue	Resolution
incorrect event data to be back filled.	
After changing a product name it was possible to get a not valid reversal when trying to reverse the new product name against an old order.	The new product name now reverses correctly.
When inserting records in list view it was possible for data from the current record to overwrite data inserted on a previous record causing a duplicate key error.	All records are inserted correctly.
On some occasions it was possible for the duplicate check to not be run when selecting it from the Duplicate Check menu item.	The duplicate check always runs.

Upgrade Notes

The upgrade from Vitalware Version 2.1.01 to Vitalware 2.1.02 involves a number of steps. Please follow the instructions below carefully.

Do not skip any steps under any circumstances.

Before proceeding with the update please ensure that a complete backup of the Vitalware *server exists and is restorable.*

The upgrade may take some time, particularly on systems with a large number of records.

1. Install Texpress 8.2.008 or greater.
2. Install TexAPI 6.0.001 or greater.
3. Log in as **vw**.
4. The following steps need to be repeated for each client installed on the Vitalware server.
   Please ensure that no one is using the system while the upgrade is underway.
5. Enter **`client`** ***`client`***
6. Enter **`script /tmp/output-2-1-02`**
   A new shell will start and all output recorded until the shell is terminated.
7. Enter **`upgrade-2-1-02`**
8. Enter **`vi ~/luts/default/System.txt`**
   Check each System entry has its correct translation then save the file. If the `~/local/luts/default/System.txt` file exists check whether this file has the correct translation.
9. Enter **`upgrade-2-1-02.statistics`** *(adjusts incorrect weekly / monthly statistics)*
10. Enter **`upgrade-2-1-02.webflags`** *(corrects **Publish On Internet / Intranet** flags)*
    NOTE: *It is not mandatory to run this script as Vitalware does not make use of these flags. This script can be run at any later date if required.*
11. Enter **`vwreindex`**
12. Enter **`vwlutsrebuild`**
13. Enter **`exit`**
    The script session will terminate.
14. Enter **`EDITOR=vi crontab -e`**
    Add the following entries:

    ```
    #
    # Run Scheduled Exports
    #
    0  6  *  *  *  ~vw/clientname/bin/vwrun vwexport 2>&1 |
    ~vw/clientname/bin/vwrun vwlogger -t "KE Vitalware
    Scheduled Exports Report" -z export
    ```
15. All cron commands must be entered on one line (even if they appear wrapped on the screen)! The start times of command may need to be varied to fit in with existing maintenance jobs.
16. Enter **`vi ~/data/eaudit/vwoptions`**
    Add the following line to the file then save the file:

    ```
    COMPACT="no"
    ```

17. Enter **`vi ~/data/estatistics/vwoptions`**
    Add the following line to the file then save the file:
    `COMPACT="no"`
18. Go back to step 5 to commence upgrading the next client, until all clients are complete.

Vitalware 2.1.02 does not require the new Windows client to be installed on every machine for network installations. Updating the network server is sufficient. For standalone installations a new client is required on each machine.

Optional Perl Packages

Significant speed improvements in the Vitalware auditing facility may be achieved by installing two perl packages. The installation of these packages is optional but recommended.

1. Log in as **root**.
2. Enter **`perl -MCPAN -e shell`**
3. Enter **`install XML::Parser::Expat`**
4. Enter **`install File::FcntlLock`**
5. Enter **`quit`**
6. Log in as **vw**.
7. Enter **`vwload stop audit`**
8. Enter **`vwload start audit`**

Vitalware® Vital Records Management

The World's Premier Vital Records Software

Vitalware Documentation

Right-to-Left Language Support in Vitalware

Document Version 1

Vitalware Version 2.1

KE Software

www.kesoftware.com

Contents

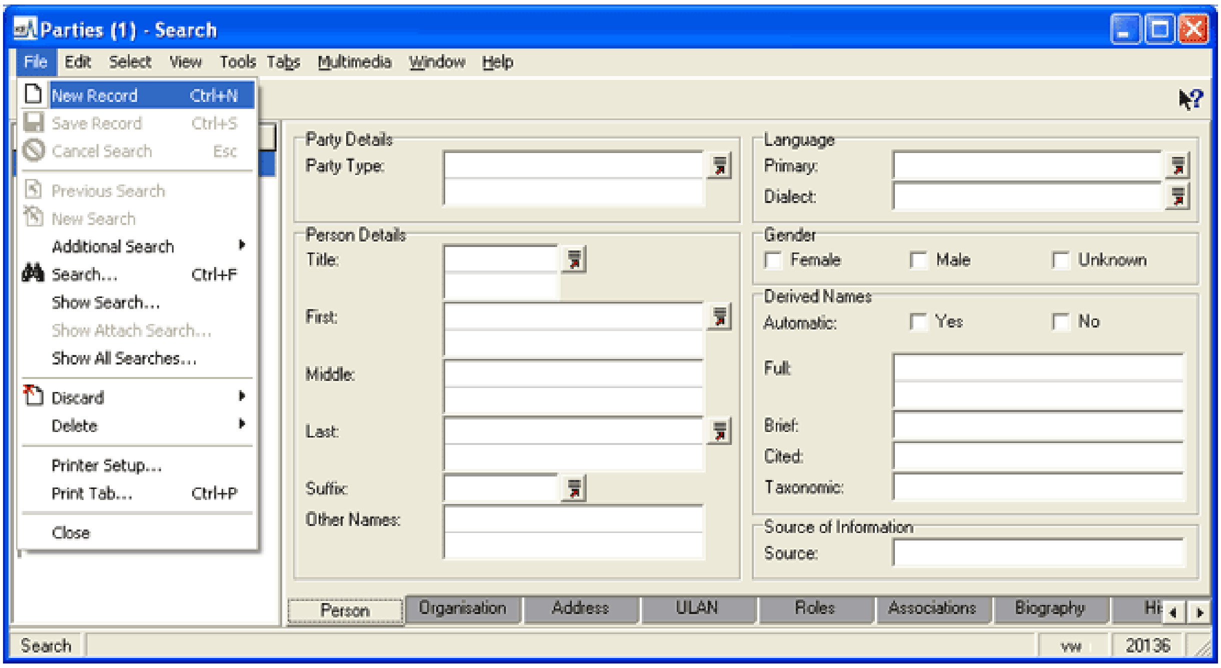
SECTION 1

Overview

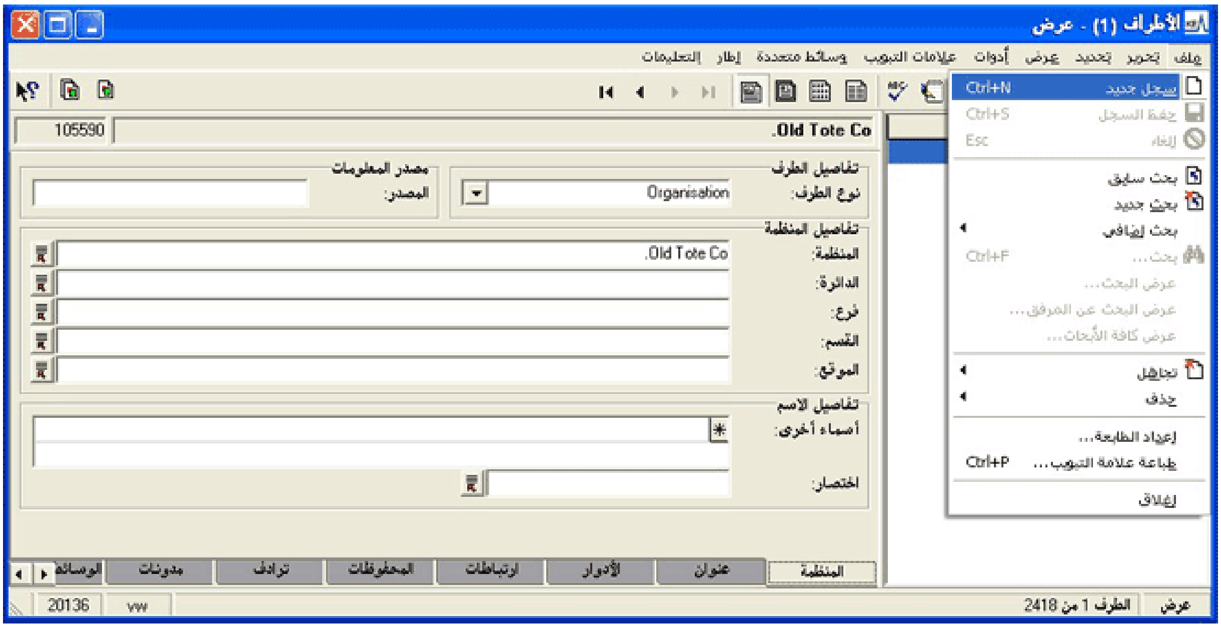
Vitalware 2.1.02 introduces enhanced support for languages, including right-to-left (RTL) languages. Arabic and Hebrew scripts are now supported, as are two additional left-to-right (LTR) languages, Greek and French (Canadian). Fourteen languages are now supported in Vitalware:

- English
- English (US)
- Arabic
- Danish
- Dutch
- French
- French (CA)
- German
- Greek
- Hebrew
- Italian
- Norwegian
- Polish
- Spanish
- Swedish

Switching between LTR and RTL languages is dynamic and does not require Vitalware to be restarted - the user interface automatically adjusts to suit the language selected. For example, selecting Arabic will result in the Menu bar moving to the right side of the module window and all controls being mirrored. The image below shows the Person tab of the Parties module with Shortcuts enabled:



This image shows a tab with Arabic selected:



As you can see, the layout of the various controls is mirrored, even down to the module Title bar. Many of the images used in Vitalware (e.g. on buttons, menus, etc.) have been altered to suit RTL languages. In some cases a new image has been constructed to ensure an authentic user experience. For example, the *What's this help?* image contains a reversed question mark in the Arabic version as this is how question marks are displayed in Arabic.

All the data controls in Vitalware 2.1.02 have been replaced with Unicode versions, which allows Vitalware to determine the character set used to store information and adjust the display to show characters correctly. Whether you use UTF-8 or ISO-8859-6 (Arabic) as your data storage encoding, the display will adjust to show the characters correctly.

> It may be necessary to install additional fonts on your computer to ensure that all characters can be displayed (e.g. Windows 2000 and Windows XP require the installation of right-to-left language files in order for Arabic and Hebrew characters to be shown).

A number of controls have been adjusted to show RTL data correctly. In particular the HTML editor provided with Vitalware (in the Field Help module) has been extended to support RTL orientation.

SECTION 2

Switching Languages

Two language settings can be altered in Vitalware:

- The first is the language used for Prompts, e.g. prompts, menus, forms, etc. Prompts in any Vitalware client can be switched to one of the fourteen supported languages.
- The second is the language used for Data.

  Data can be added in one or more languages, and it is possible to select which language or languages are used to display the data. For systems with only one language it is obviously not possible to alter the language in which Data is displayed.

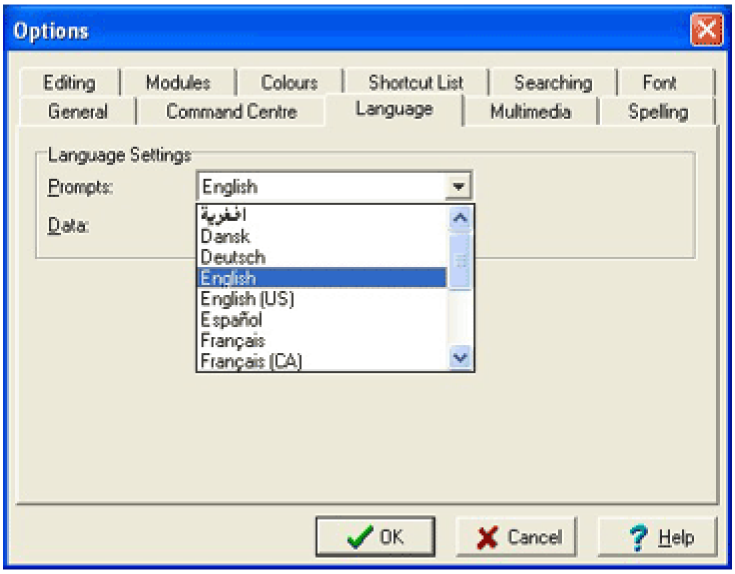
  Which languages are used is set in the Vitalware Registry via:
  `System|Setting|Language|Supported`

  For clients with multiple languages supported, it is possible to select any one language or all languages to display data.

Vitalware provides two mechanisms for switching the languages displayed in the user interface:

- The first mechanism allows the language used for Prompts and Data to be set independently.

  This is achieved using the Language tab in the Options dialogue box (select **Tools>Options** from the Menu bar). As the names suggest, the *Prompts* drop list specifies the language use for prompts, and the *Data* drop list specifies the language used to display data:



Prompts: English

Data:

When changing the language of Prompts, Vitalware adjusts the user interface to the language selected as soon as the **OK** ✓ OK button is clicked. When switching between LTR and RTL languages a small delay may occur while the user interface is adjusted.

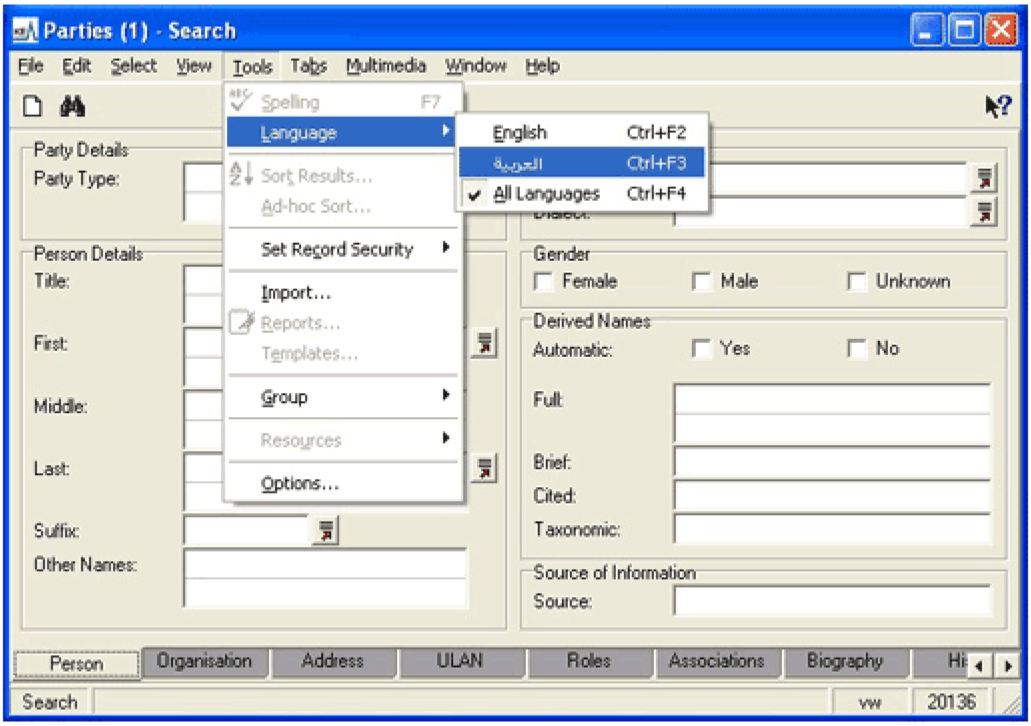
When switching the language of Data, only the data displayed in controls is affected. The layout of the user interface layout and language of Prompts is not adjusted.

- The second mechanism for switching the languages displayed in the user interface allows both the language of Prompts and Data to be changed at the same time.

  This is achieved by selecting **Tools>Language** from the Menu bar. For systems configured to allow more than one language for data, a **Tools>Language** sub-menu lists each language and an entry for **All Languages**.

  Selecting one of the menu options changes both the language of Prompts and Data at the same time.

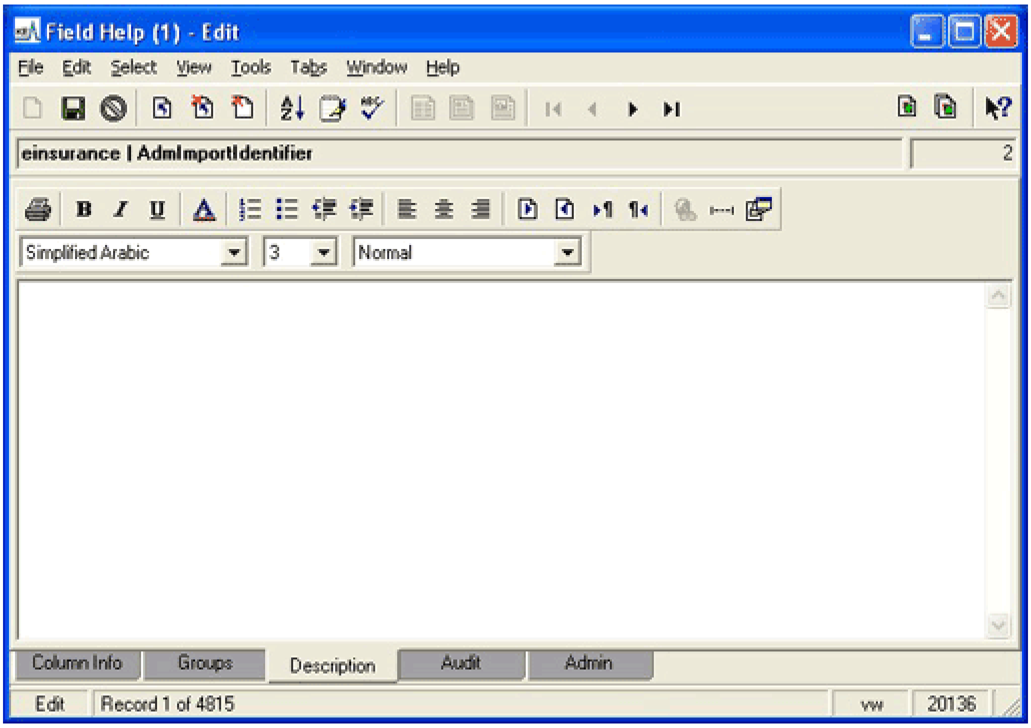
  If **All Languages** is selected, the language used for Prompts is not changed, but all languages used for Data are displayed separated by the language delimiter (normally ; : ;):



SECTION 3

HTML Editor Extensions

In order to provide full support for RTL languages, the standard Vitalware HTML editor has been extended to allow the page and paragraph orientation to be set:



einsurance | AdmImportIdentifier

2

Simplified Arabic

3

Normal

Edit

Record 1 of 4815

vw

20136

The new buttons are:

Button	Description
🗎▸	The document direction is left-to-right (same as `<html dir="ltr">`). If a direction is not specified, the default direction is `ltr`.
🗎◂	The document direction is right-to-left (same as `<html dir="rtl">`). An `rtl` document has the scroll bar located on the left side of the window with all text right aligned by default. The standard Unicode *bidi* algorithm is used to display text. That is, English text is displayed in a left-to-right order while Arabic / Hebrew is displayed right-to-left. The setting affects the default direction of all text in the document.
▸¶	The text direction for the current paragraph is set to `ltr` (same as `<p dir="ltr">`). If the document direction is `rtl`, this command will reset the direction to `ltr` for the current paragraph only. Note that Arabic text will still be displayed right-to-left but left alignment will be used. This is useful for embedding an English quote in Arabic text.
¶◂	The text direction for the current paragraph is set to `rtl` (same as `<p dir="rtl">`). This is useful for embedding an Arabic quote in English text.

The RTL language features added to the HTML editor are available for all HTML based fields in Vitalware.

SECTION 4

Unicode Data Controls

To support RTL languages in Vitalware, all data controls (e.g. RichEdit, Grid and Combobox controls) have been upgraded to support Unicode. All other user interface controls (e.g. menus, tabs, etc.) have not changed and still use single byte based character sets. The change to data controls means any Unicode character may be entered into a data field and it will be displayed and saved correctly. If the Vitalware server encoding is set to `utf8`, the data will also appear correctly in reports and on the Internet.

For clients who do not have the Vitalware server encoding set to `utf8`, a mapping takes place where all data characters are translated from the server encoding that has been used into UTF-8 before being displayed, thus causing the correct characters to be shown. A reverse mapping takes place for all Unicode data entered into controls when saved to the Vitalware server.

When the Vitalware client is invoked it queries the Vitalware server to determine the encoding used. Once the client knows the encoding, it performs all the necessary translations to ensure the data is displayed and saved correctly.

As non-data controls have not been upgraded to their Unicode equivalent they will display text (in most cases prompts) using the character set defined by the Prompts language setting. In rare instances this may result in the incorrect display of individual characters (for example, selecting French as the language for Prompts and trying to show an Arabic character). In general this should not occur as all prompt based strings are translated using the character set for the language specified for Prompts.

SECTION 5

Installation of RTL Languages

In order to display RTL based characters correctly and to install the *bidi* algorithm used for data entry and display, it may be necessary to add RTL language files to your Windows installation. Details on how to install the RTL language files are given below for each version of Windows:

Windows 2000

1. Select **Start>Settings>Control Panel>Regional Options**.
2. On the General tab, select the language to install (e.g. **Arabic**) in the *Language settings for the system* section.
3. Click **Apply**.
4. If prompted, insert the Windows 2000 CD-ROM and click **OK**.
5. Restart your computer when prompted. If you are not prompted, select **Start>Shut Down>Restart**.

Windows XP

1. Select **Start>Control Panel>Regional and Language Options**.
2. On the Languages tab, click the check box beside *Install files for complex script and right-to-left languages (including Thai)*.
3. Click **Apply** and then **OK**.
4. Restart your computer when prompted. If you are not prompted, select **Start>Shut Down>Restart**.

Windows 2003 (Server)

1. Select **Start>Control Panel>Regional and Language Options**.
2. On the Languages tab, click the check box beside *Install files for complex script and right-to-left languages (including Thai)*.
3. Click **Apply** and then **OK**.
4. Restart your computer when prompted. If you are not prompted, select **Start>Shut Down>Restart**.

Windows Vista and Windows 7

Windows Vista and Windows 7 have RTL language fonts installed by default and there is no need to add any language files.

SECTION 6

Data Entry for RTL Languages

Once the correct language files are installed, it may be necessary to configure Windows to allow data entry in multiple languages. This involves adding a keyboard layout for the language in which you wish to enter data.

For example, to type Arabic characters it is necessary to configure the keyboard to mimic an Arabic keyboard. As each version of Windows provides a different mechanism for adding keyboard layouts, details are provided for each release:

Windows 2000

1. Select **Start>Settings>Control Panel>Regional Options**.
2. Select the **Input Locales** tab.
3. Click **Add** if the language in which you wish to type is not listed under *Installed input locales*.
4. In the Add Input Locale dialogue box, select the language in which you wish to type from the Input Locale list. This automatically selects the Keyboard Layout/IME option.
5. Click **OK**.
6. In the Regional Options dialogue box, click **Apply** and then **OK**.

Windows XP

1. Select **Start>Control Panel>Regional and Language Options**.
2. Select the **Languages** tab.
3. Click **Details** in the *Text services and input languages* section. The Text services and input languages dialogue box appears.
4. If the language in which you wish to type is not listed under *Installed Services*, or if the language that you want is listed but the keyboard is not listed with it, click **Add**.
5. In the Input Language list, select the language in which you wish to type. This automatically populates the Keyboard layout/IME field.
6. Click **OK**. The selected language with a keyboard installed displays under *Installed Services*.
7. Click **OK** to close the Text Services and Input Languages dialogue box, and then click **OK** to close the Regional and Language Options dialogue box.

Windows 2003 (Server)

1. Select **Start>Control Panel>Regional and Language Options**.
2. On the Languages tab, click **Details**.
3. If the language in which you wish to type is not listed under Installed Services, or if the language that you want is listed but the keyboard is not listed with it, click **Add**.
4. In the Input Language list, select the language in which you wish to type. The Keyboard layout/IME field is automatically populated for you.
5. Click **OK**. The selected language with a keyboard installed displays under *Installed Services*.
6. Click **OK** to close the Text Services and Input Languages dialogue box, and then click **OK** to close the Regional and Language Options dialogue box.

Windows Vista and Windows 7

1. Select **Start>Control Panel>Regional and Language Options**.
2. On the **Keyboards and Languages** tab, click **Change keyboards**.
3. Under Installed services, click **Add**.
4. Double-click the language you want to add.
5. Double-click the text services you want to add
6. Select the text services options you want to add.
7. Click **OK**.

After keyboard layouts have been added, it is possible to select the input language from the Language bar, which is situated in the System Tray (it is a button with the two letter name for the current input language displayed). Click the **Language** bar for a list of all possible data entry languages. Select the input language required.

Index

Vitalware Documentation

Scheduled Exports

Document Version 1.1

Vitalware Version 2.1.02

www.kesoftware.com

Contents

SECTION 1

Overview

> To be able to use the Export Schedules facility, a user must have (or be a member of a group that has) the `daExport` operations permission.

Vitalware 2.1.02 introduces a new facility that enables data to be exported on a regular basis. Users can define:

- A set of fields for export.
- A sort order for the exported records.
- A TexQL statement to determine which records to export.
- Dates on which the export is to occur.

Data can be scheduled to be exported on anything from a daily to an annual basis. The export occurs outside working hours (generally early in the morning) and the results are stored on the Vitalware server. A user may then view the results and save / view the export files produced.

The Scheduled Exports facility provides two new modules:

- **Schedules**

  Holds one record per Scheduled Export containing the definition of the export (as defined above).
- **Exports**

  Holds one record per set of export files produced. When a Scheduled Export is run a new exports record is created containing the output of the export process.

See The Schedules and Exports Modules (page 33) for more details.

An audit of all Scheduled Exports is produced and these records are not removed from the system, so it is possible to search / view all exports performed by all users (provided that you have sufficient privileges to do so).

When a Scheduled Export is executed, the default output is XML, which is the same format used by the Vitalware reporting facility. A filter may be set to transform the XML produced into another format (CSV for example). System Administrators may define their own filters to produce custom output formats. A custom filter may also be used to:

- Email export files to users.
- Print export files.
- Copy export files to a known location (for collection).
- Transmit export files to another machine.

See Creating Filters (page 37) for more details on how to create custom filters.

SECTION 2

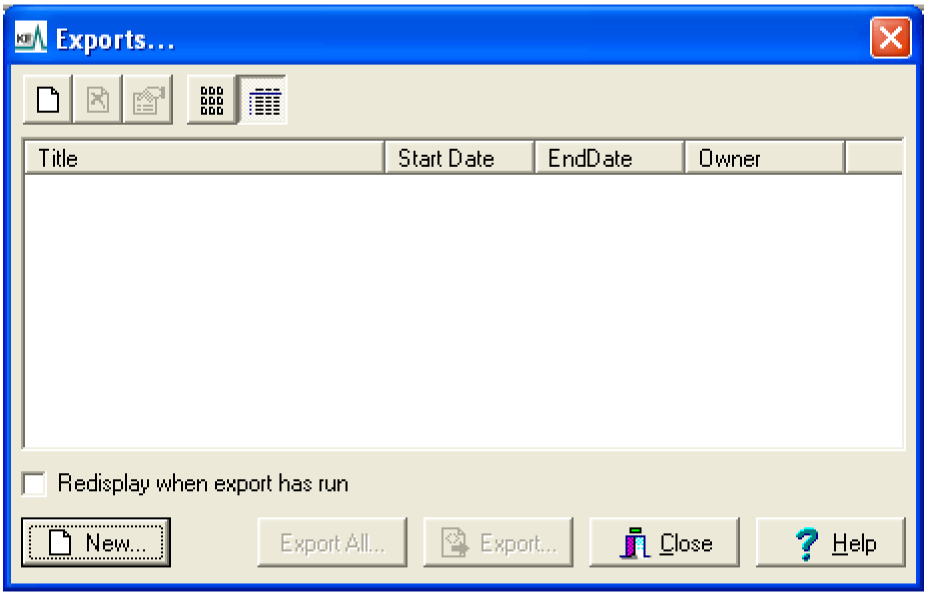
Creating a Scheduled Export

Configuring a Scheduled Export is very similar to configuring a report. In fact the Properties dialogue box used to select the fields for export is the same as used by the reporting process. In this section we step through the creation of a new Scheduled Export.

Set up the Export schedule: the Export tab

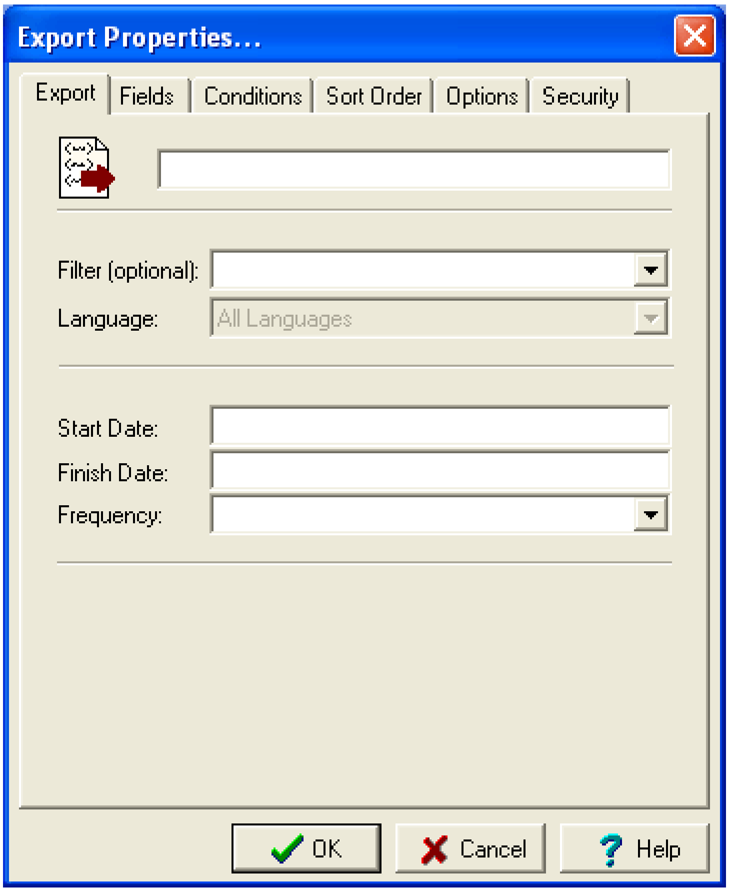
In the module from which the records are to be exported:

1. Search for or otherwise list a group of records.
2. Select **Tools>Exports** in the Menu bar.
   -OR-
   Use the keyboard shortcut, `ALT+T+X`.
   The Exports dialogue box displays with a list of all Scheduled Exports for the current module:



3. Select **New** 🗋 New...
   -OR-
   Select **New** 🗋

The Export Properties dialogue box displays:



Filter (optional):

Language: All Languages

Start Date:

Finish Date:

Frequency:

4. Enter a descriptive name for the Export Schedule in the top text field.

   The title will display in the Exports dialogue box, from where it can be selected by authorised users to run the export on an adhoc basis.

5. If you require the exported data to be transformed from XML format to another format (e.g. CSV), select a filter from the *Filter (optional)* drop list.

   Filters are defined by the System Administrator. If a filter is not selected, the exported data is saved in Vitalware XML format. See Creating Filters (page 37) for more details.

6. Select the language of the data to be exported from the *Language* drop list.

   A language may only be selected if your Vitalware system supports more than one data language. Any of the available languages in which data is recorded may be selected or `All Languages` may be used to export the complete contents of the exported fields.

7. Enter the first date on which the export will be produced in the *Start Date* field.

   This date is also used to calculate the next time an export should occur when the report frequency is specified as a number of days / months / years.

   The format of the date (e.g. `dd/mmm/yyyy`) is as specified for your Vitalware system.

   A start date must be provided.

8. Enter an end date in the *Finish Date* field.

> An end date is not required, however if specified, data will not be exported after that date has been reached. An empty end date implies no final date for exporting data.

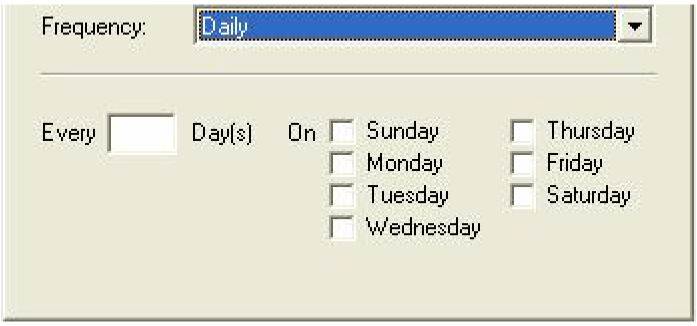
9. Select an export frequency from the *Frequency* drop list.

   There are four frequency values to choose from:

Adhoc Use **Adhoc** when defining an export that will not be run at a regular interval, for example if you are contacted irregularly to produce the export data.

As the export is not scheduled, it must be run manually to produce the export data. See Unscheduled Exports (page 27) for more details.

Daily Use **Daily** if the data is exported on anything less than a monthly basis. The following options are displayed when **Daily** is selected:



Frequency: Daily

There are three ways to configure daily exports:

1. The simplest is to select the days of the week on which the export is to occur. The export is run early in the morning of the day(s) selected.

   If no Finish Date is specified, the export will run every week on the specified day(s).

   Typically an export would be run once per week or every day.

   If there is an interval between exports (the export is scheduled for Sunday and Thursday, for instance), the records exported on Sunday would be from the previous Thursday, and the records exported on the Thursday would be from the previous Sunday.

2. Specify the number of days between exports (starting from the Start Date) in the *Every nnn Day(s)* field.

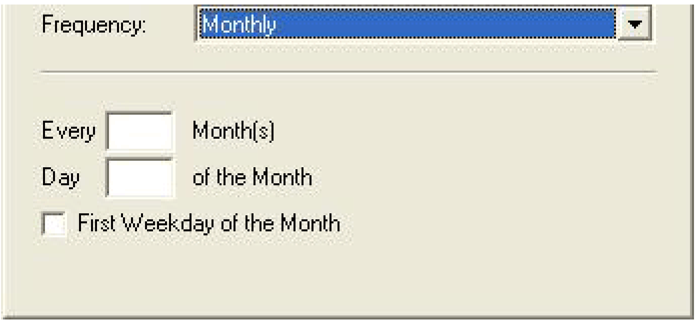
   For example, a value of `14` indicates that the export is to be run fortnightly from the Start Date.

3. Specify the day on which an export will first be run and the interval between exports.

   For example, a value of *Every* `14` *Day(s)*, with `Thursday` selected indicates that an export will commence on the first Thursday after or on the Start Date and will run every 14 days.

Monthly

Use **Monthly** when data exports occur on a monthly basis but less than annually. The following options are displayed when Monthly is selected:



Frequency: Monthly

Specify how may months must elapse between each report using the *Every nnn Month(s)* field. If a value is not given, the export will occur monthly. For example, a value of `3` indicates that the export will occur quarterly.

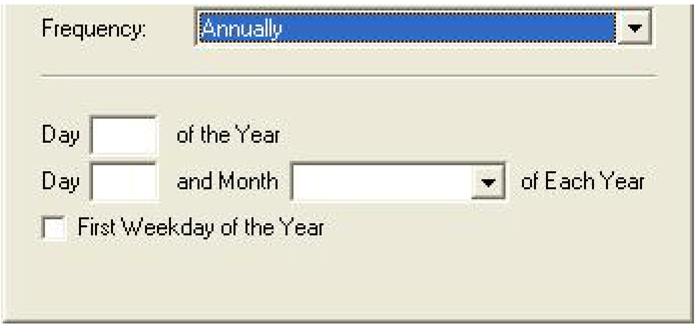
Use the *Day nnn of the Month* field to specify on what day in the month the export is to occur. A value of `1` indicates that the export will run on the first day of the month. A value of `31` implies the last day of the month, regardless of how many days are in the month.

The *First Weekday of the Month* checkbox may be selected instead of specifying a day of the month. In this case, the export is run early in the morning on the first Monday of the month.

A value must be supplied for either *Every nnn Month(s)* or the *First Weekday of the Month.*

Annually

Use **Annually** for any data export required on a yearly basis. The following options are displayed when Annually is selected:



Frequency: Annually

There are three ways to configure annual exports:

1. Specify the day of the year on which the export is to take place using the *Day nnn of the Year* field. A value between 1 and 365 can be used.
2. Specify a specific day in a given month using the *Day nnn and Month mmm of Each Year* field.

   For example, a setting of *Day* `1` *and Month* `July` *of Each Year* would allow data up to and including 30 June to be exported.
3. The *First Weekday of the Year* checkbox may be selected instead of specifying a day within a month. In this case, the export is executed early in the morning of the first Monday of the Year.

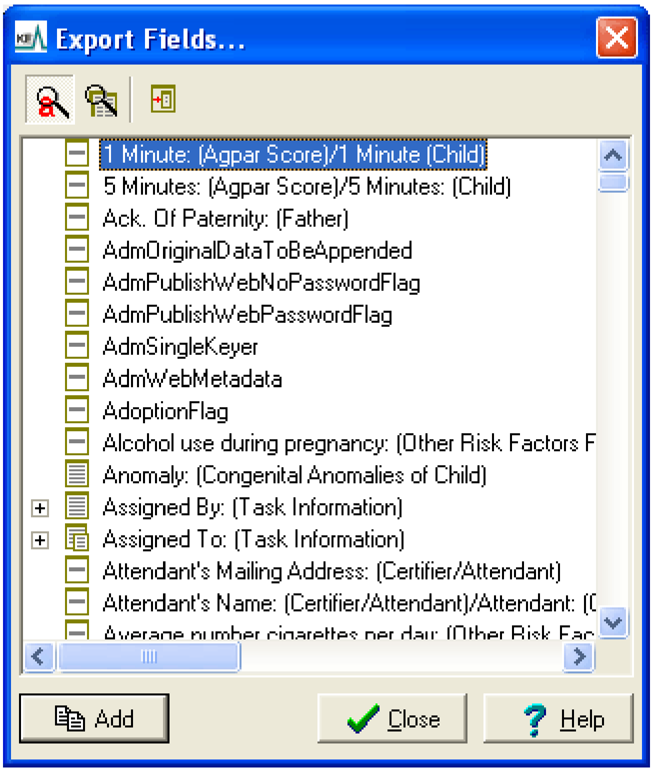
Select the fields to include in the export: the Fields tab

1. In the Export Properties dialogue box, select the **Fields** tab.

   Here we add the fields to be exported.

2. Select **Add** Add... .

   The Export Fields box displays with a list of all fields in the current module, as well as fields in attached modules (an attachment field is indicated by the plus ⊞ icon):

   1 Minute: (Agpar Score)/1 Minute (Child)
   5 Minutes: (Agpar Score)/5 Minutes: (Child)
   Ack. Of Paternity: (Father)
   AdmOriginalDataToBeAppended
   AdmPublishWebNoPasswordFlag
   AdmPublishWebPasswordFlag
   AdmSingleKeyer
   AdmWebMetadata
   AdoptionFlag
   Alcohol use during pregnancy: (Other Risk Factors F
   Anomaly: (Congenital Anomalies of Child)
   ⊞ Assigned By: (Task Information)
   ⊞ Assigned To: (Task Information)
   Attendant's Mailing Address: (Certifier/Attendant)
   Attendant's Name: (Certifier/Attendant)/Attendant: (C
   Average number cigarettes per day: (Other Risk Fac

3. In the Export Fields dialogue box, select a field and select **Add** Add

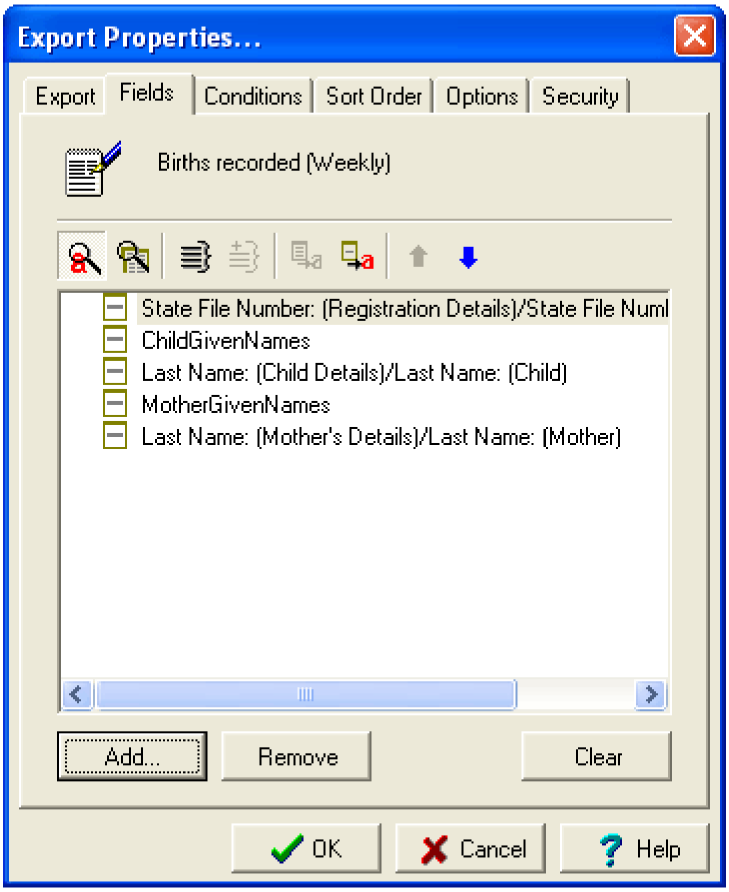
   -OR-

   Double-click a field.

   The field is added to the Fields list in the Export Properties box.

4. When all required fields are added, select Close ✓ Close in the Export Fields dialogue box.

The Export Properties dialogue box now lists all the fields that will be included in the export:



Births recorded (Weekly)

- State File Number: (Registration Details)/State File Numl
- ChildGivenNames
- Last Name: (Child Details)/Last Name: (Child)
- MotherGivenNames
- Last Name: (Mother's Details)/Last Name: (Mother)

For more details regarding the selection of fields for export see *Select the fields to include in the report: the Fields tab* in the Reports section of the Vitalware help.

Determine which records to export: the Conditions tab

1. In the Export Properties dialogue box, select the **Conditions** tab.

   On the Conditions tab we add a TexQL statement to determine which records will be exported.

2. Enter a TexQL statement into the Conditions text box.

   The following placeholders may be used in the TexQL statement:

   `#TODAY#` This placeholder is replaced with the date on which the export is run. The date order used (e.g. day/month/year, month/day/year) is as specified for your Vitalware system.

   `#STARTDATE#` This placeholder is replaced with the date of the first day after the end date of the previous export. It is used for date range queries to specify the starting date for the exported records.

   `#ENDDATE#` This placeholder is replaced with the date of the last day for the records to be exported. The end date **is always the day before the scheduled date** as exports are run early in the morning of the scheduled day.

   `#MONTH#` This placeholder is replaced with the month of the day that falls immediately prior to the day of the export.

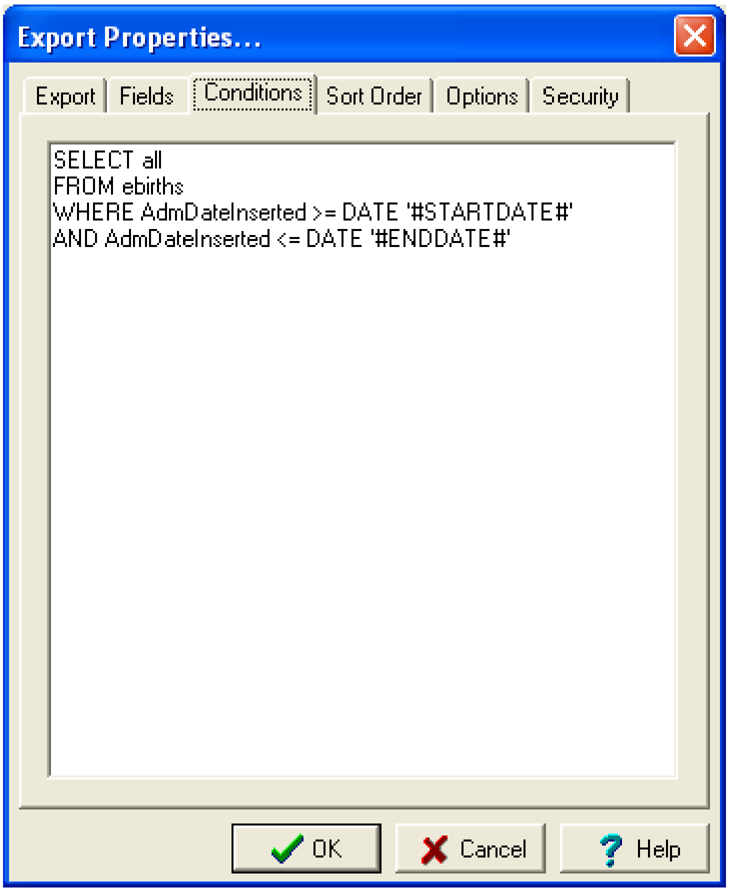
   The month is a number between 1 and 12.

   For example, if the export date is 3 May, `#MONTH#` is `5` (May); if the export date is 1 May, `#MONTH#` is `4` (April).

   `#YEAR#` This placeholder is replaced with the year of the day prior to the execution of the export.

> Placeholders use the day before the date on which the export is run as the closing date because exports are run early in the morning: the closing date for searches does not include the current date as the day is not yet complete.

When the Export Properties dialogue box is closed by selecting the **OK** button, the TexQL statement is checked to ensure that a valid query has been specified:



```
SELECT all
FROM ebirths
WHERE AdmDateInserted >= DATE '#STARTDATE#'
AND AdmDateInserted <= DATE '#ENDDATE#'
```

Hint - Determining the TexQL Conditions statement

A TexQL condition statement is a query used to retrieve the records to be exported. Any valid TexQL query statement may be used. Unfortunately the format of the statement may be hard to piece together unless you are familiar with SQL. Fortunately Vitalware provides an easy mechanism for generating a statement based on query terms entered into the current module.

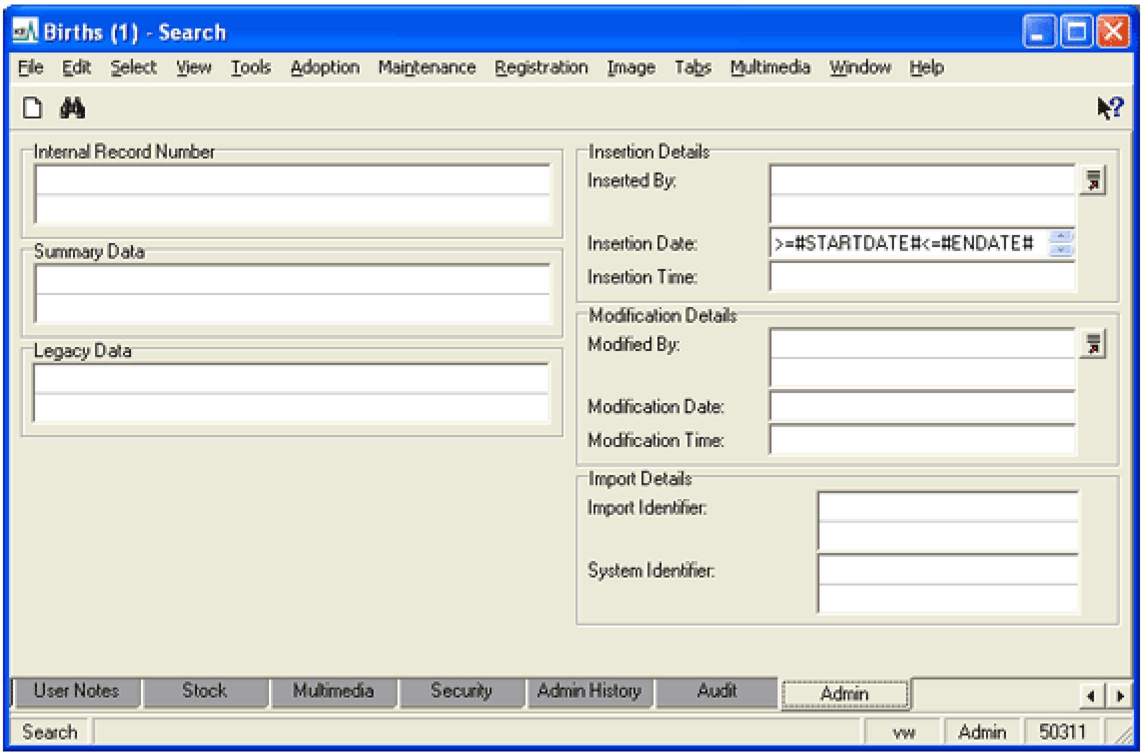
1. Open the module from which the records are to be exported.

   The module should be in Search mode.

2. Enter search terms as though you were retrieving the records you would like to export.

> Note that there may be search terms specified in a module by default. If these are not required, clear them.

Where you want to use a date range containing place markers, enter the place marker into the control:



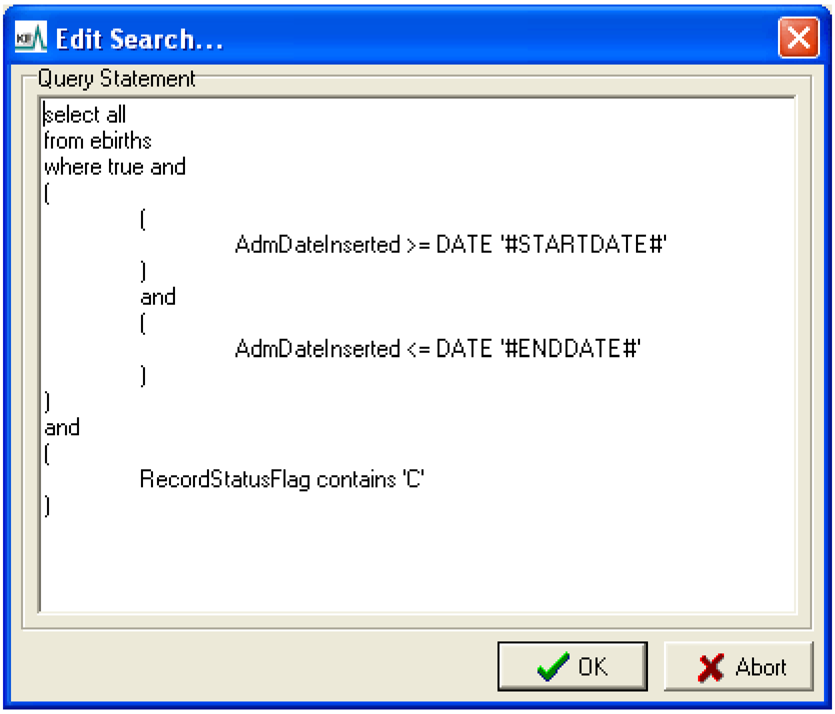
Insertion Date: >=#STARTDATE#<=#ENDATE#

3. Select **File>Show Search** in the Menu bar

   -OR-

   Use the keyboard shortcut, `ALT+F+O`.

The Edit Search dialogue box displays:



```
select all
from ebirths
where true and
(
        (
                AdmDateInserted >= DATE '#STARTDATE#'
        )
        and
        (
                AdmDateInserted <= DATE '#ENDDATE#'
        )
)
and
(
        RecordStatusFlag contains 'C'
)
```

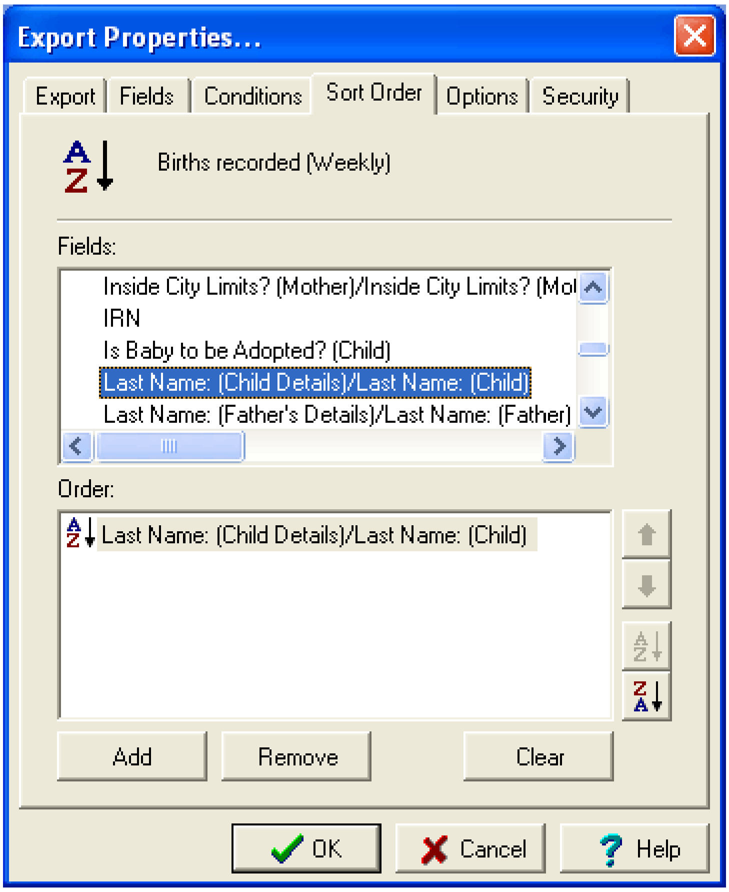
4. Copy the contents of the Edit Search box.
5. Select **Abort** ✗ Abort.

   The Edit Search box is closed.

   The TexQL condition statement to use for your export can now be pasted into the Conditions tab when defining your Export Properties.

Define sort order: the Sort Order tab

1. In the Export Properties dialogue box, select the **Sort Order** tab.
   All fields in the module are listed in the Fields list.
2. Select a field in the Fields list and select **Add** Add to add it to the Order list.
   -OR-
   Double-click the field name in the Fields list:



Births recorded (Weekly)

Fields:

Inside City Limits? (Mother)/Inside City Limits? (Mot
IRN
Is Baby to be Adopted? (Child)
Last Name: (Child Details)/Last Name: (Child)
Last Name: (Father's Details)/Last Name: (Father)

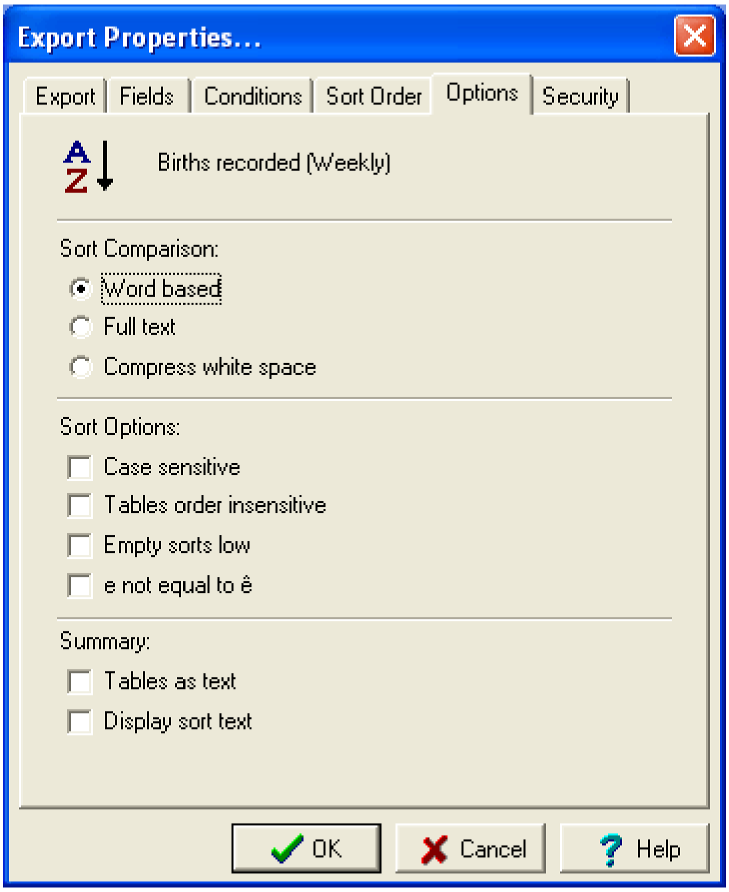
Order:

Last Name: (Child Details)/Last Name: (Child)

Add as many fields on which to sort as required.

Set sort options: the Options tab

1. In the Export Properties dialogue box, select the **Options** tab:



Births recorded (Weekly)

Sort Comparison:

● Word based
○ Full text
○ Compress white space

Sort Options:

□ Case sensitive
□ Tables order insensitive
□ Empty sorts low
□ e not equal to ê

Summary:

□ Tables as text
□ Display sort text

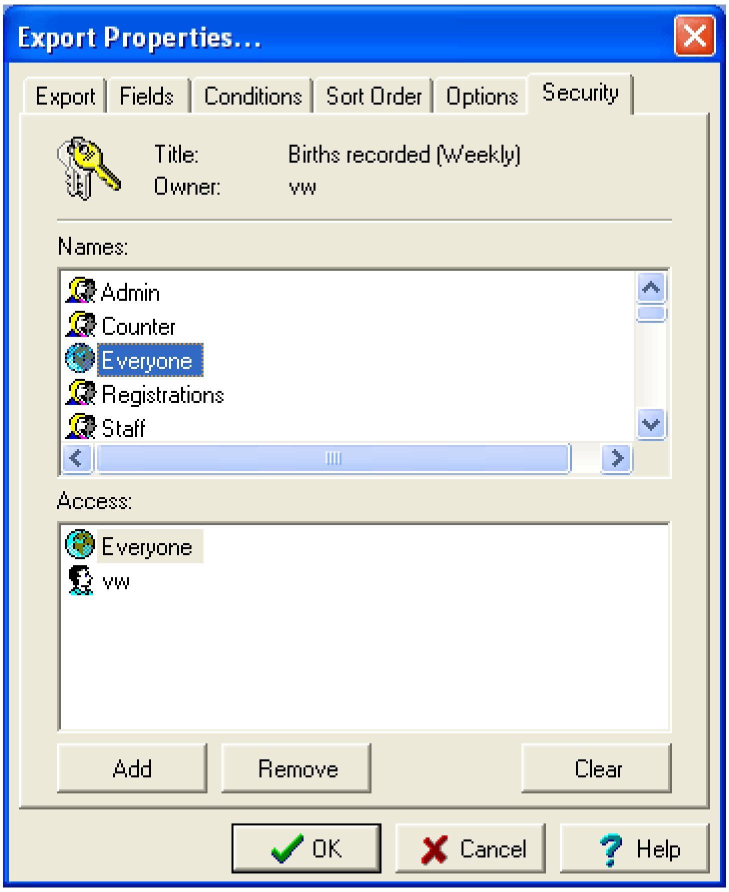
2. Select the required Sort Options.

   For more details regarding sort options see *Set sort options: the Options tab* in the Reports section of the Vitalware help.

Set security options: the Security tab

On the Security tab we determine who is authorised to run your export by adding users or groups to the Access list:

1. In the Export Properties box, select the **Security** tab:



Title: Births recorded (Weekly)
Owner: vw

Names:
- Admin
- Counter
- Everyone
- Registrations
- Staff

Access:
- Everyone
- vw

2. Select a user / group name (e.g. **Everyone**) in the Names box and select **Add** Add

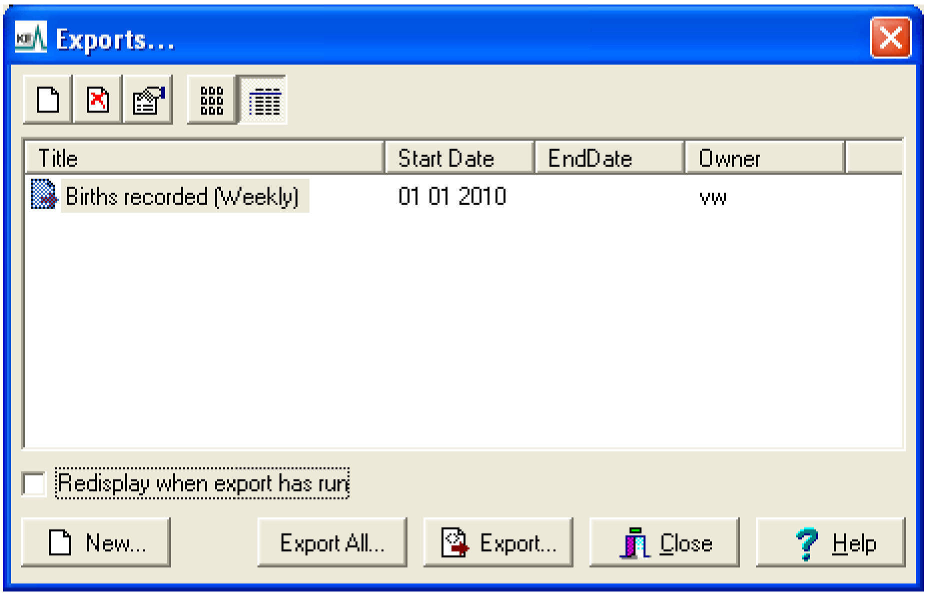
   -OR-

   Double-click the user / group name in the Names list.

   The name is added to the Access list.

3. Select **OK** ✓ OK.

Any TexQL statement added on the Conditions tab is now validated and if all is well the new scheduled export will display in the Exports dialogue box:



Title	Start Date	EndDate	Owner
Births recorded (Weekly)	01 01 2010		vw

□ Redisplay when export has run

Examples

Scenario 1

A births record creation project is under way. As part of the outputs of the project a weekly file is to be produced containing the Registration Number, Child's Given Names, Child's Surname, Date of Birth, Mother's Given Name, Mother's Surname of all records added in the week.

Solution

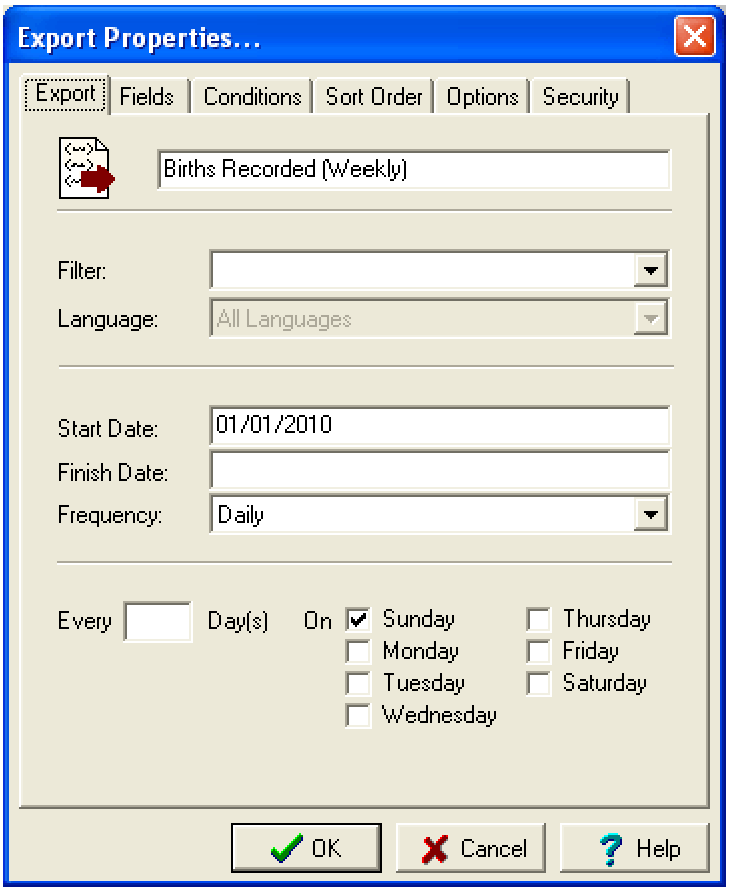
As the file is to be produced weekly, the export frequency required is `Daily` (as the frequency is less than monthly). The `Sunday` checkbox should be selected. As no Finish Date is specified, these settings will result in the export running early Sunday morning, once a week, exporting all data from the previous Sunday through to the Saturday.

The Conditions statement needs to query for all records inserted into the Births module in the previous week. As the export is run once a week the `#STARTDATE#` placeholder will be set to the date of the previous Sunday and the `#ENDDATE#` will be the date of the Saturday immediately before the export is run.

For example, if the export is run on the morning of Sunday, 14 February 2010, then `#ENDDATE#` will be 13 February 2010 (Saturday) and `#STARTDATE#` will be 7 February 2010 (the previous Sunday). The required Conditions statement is:

```
SELECT all
FROM   ebirths
WHERE  AdmDateInserted >= DATE '#STARTDATE#'
AND    AdmDateInserted <= DATE '#ENDDATE#'
```

The Export and Conditions tabs are:



Births Recorded (Weekly)

Filter:

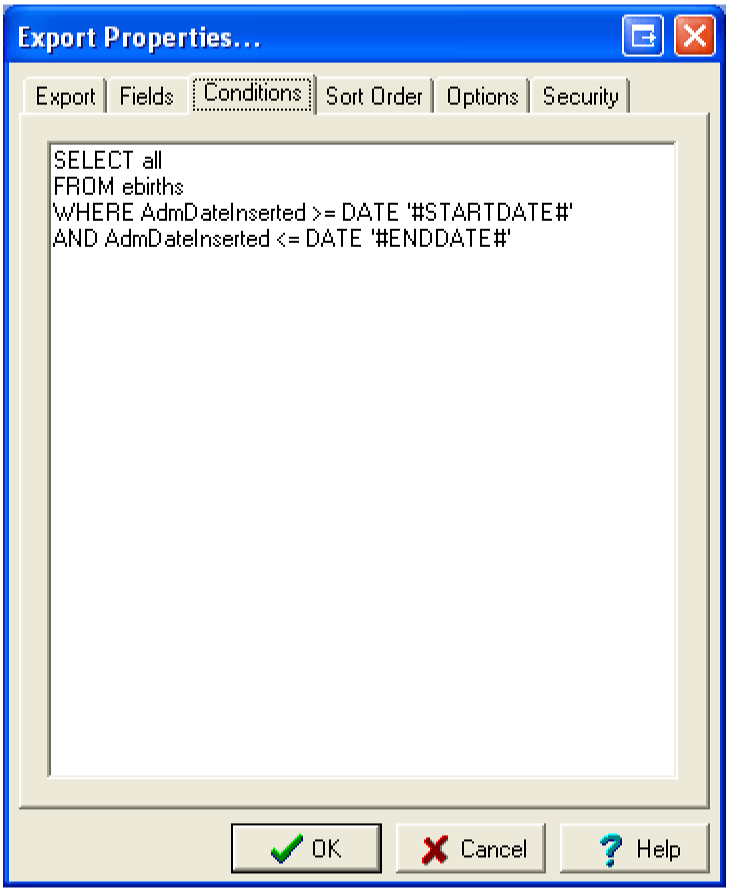
Language: All Languages

Start Date: 01/01/2010

Finish Date:

Frequency: Daily

Every Day(s) On ☑ Sunday □ Monday □ Tuesday □ Wednesday □ Thursday □ Friday □ Saturday



```
SELECT all
FROM ebirths
WHERE AdmDateInserted >= DATE '#STARTDATE#'
AND AdmDateInserted <= DATE '#ENDDATE#'
```

Scenario 2

A quarterly export of all records of deaths where the deceased's age is 18 years or younger is required by the Health Department. The export must list the Registration Number, Deceased's Given Names, Surname, Date of Death and Date of Birth.

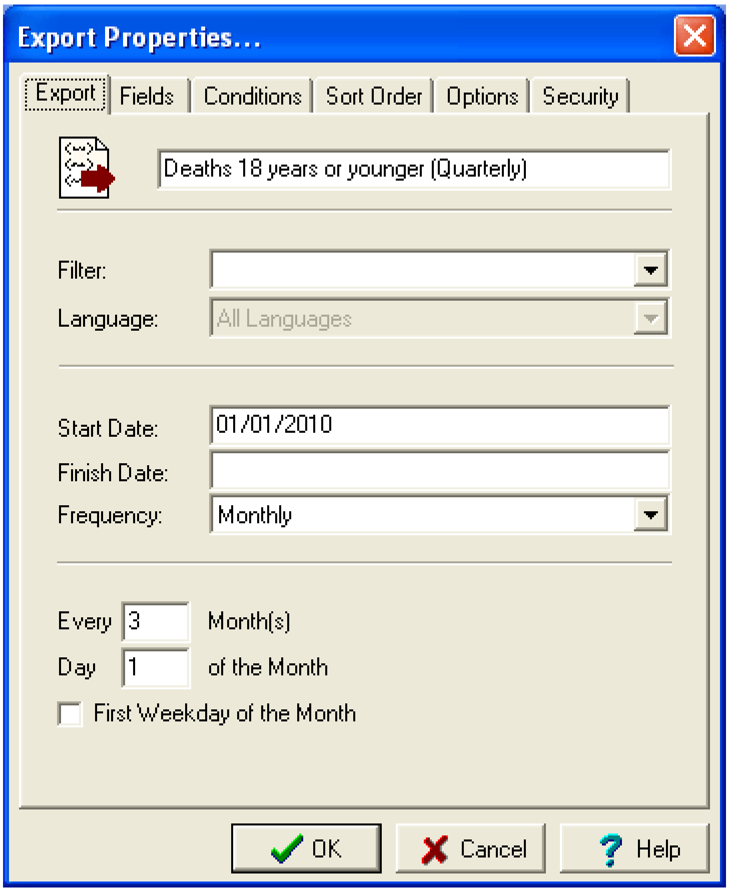
Solution

Since the export is required every three months, the `Monthly` export frequency should be chosen. The *Every nnn Month(s)* field should be set to `3` to ensure the export only occurs once every three months (that is, quarterly). The *Day nnn of the Month* field should be set to `1` as we want the export to include all records up to but not including the first day of the month. Finally the *Start Date* needs to be set to the first day of a quarterly date boundary. If our quarters start on 1 January, 1 April, 1 July and 1 October, then one of these dates should be chosen.

The Conditions statement needs to retrieve all records where age at death is 18 years or younger. As there are no date restrictions, we do not require a date range as part of the statement. A suitable Conditions statement would be:

```
SELECT all
FROM   edeaths
WHERE  DeceasedAge <=18
```

The Export and Conditions tabs are:



Deaths 18 years or younger (Quarterly)

Filter:

Language: All Languages

Start Date: 01/01/2010

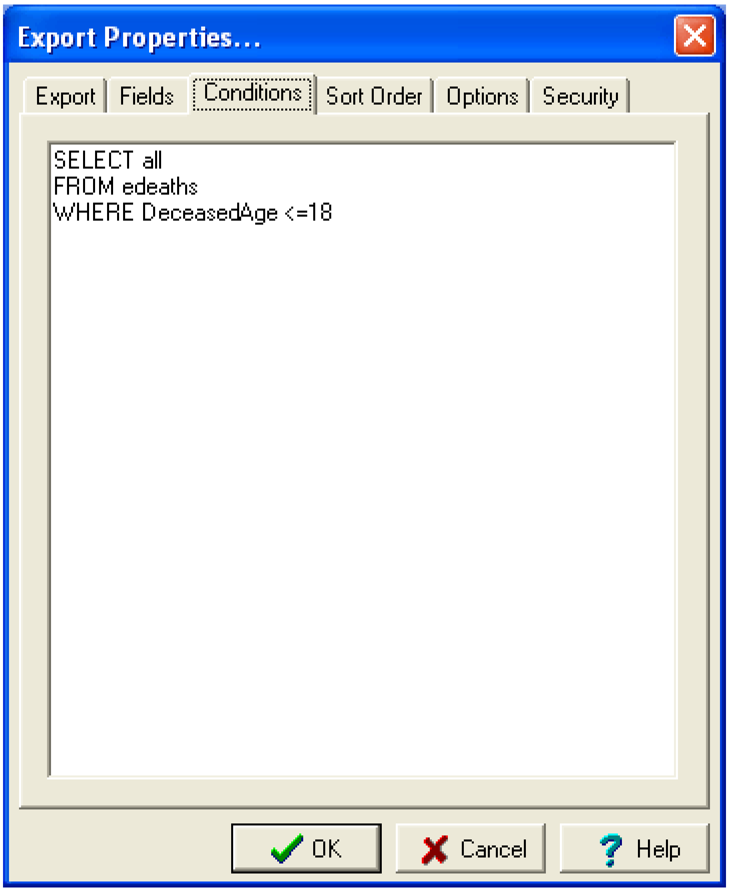
Finish Date:

Frequency: Monthly

Every 3 Month(s)

Day 1 of the Month

□ First Weekday of the Month



```
SELECT all
FROM edeaths
WHERE DeceasedAge <=18
```

Scenario 3

In order to send a copy of the change in the Marriage Act to all Marriage Celebrants, an export file containing mailing addresses is required. The report is only sent to Marriage Celebrants registered on 31 December in the given year. The export should produce a CSV file.

Solution

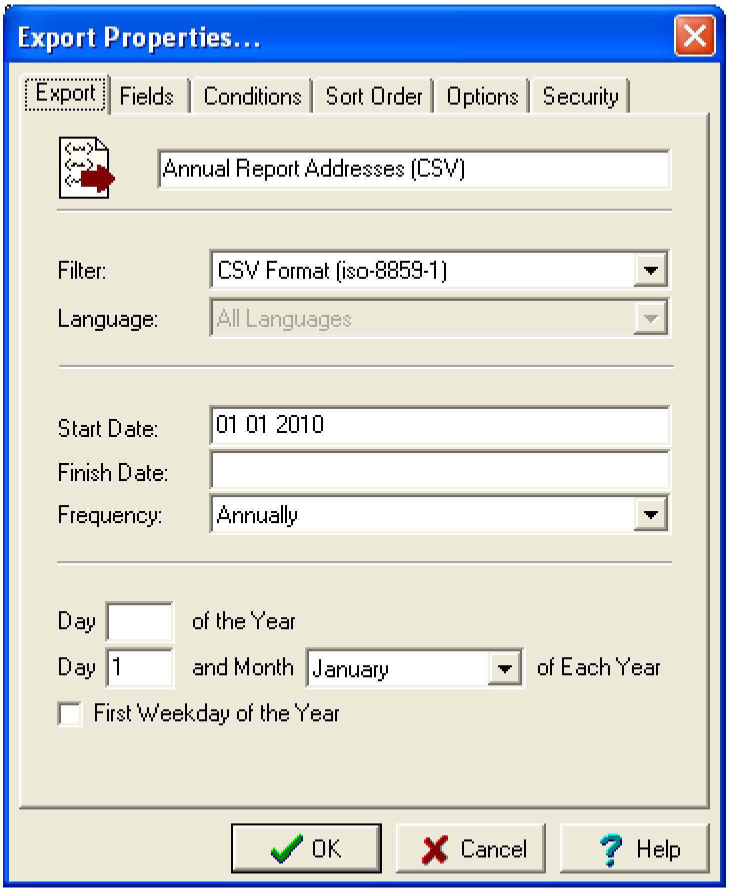
Since we are dealing with people, the Scheduled Export needs to be created in the Parties module. The data is generated annually, so the `Annually` export frequency should be used. The *Day nnn and Month mmm of Each Year* should be set to *Day* `1` *and Month* `January` *of Each Year*. As the export is run early in the morning on 1 January, it will include all records up to and including 31 December. As CSV (Comma Separated Values) output is required, the *Filter* should be set to `CSV Format (iso-8899-1)`.

The Conditions statement needs to find all Marriage Celebrants registered on 31 December. We only need to find Parties records containing a *Type* of `Marriage Celebrant`. A suitable Conditions statement is:

```
SELECT all
FROM eparties
WHERE true and
(
 not
 (
     NamOrganisation contains 'Cancelled'
 )
)
and
(
 NamPartyType contains 'MarriageCelebrant'
)
```

The required Export and Conditions tabs are:

In this example the Conditions statement was copied from the Show Search dialogue box. See Hint - Determining the TexQL Conditions statement (page 12) for more details.



Annual Report Addresses (CSV)

Filter: CSV Format (iso-8859-1)

Language: All Languages

Start Date: 01 01 2010

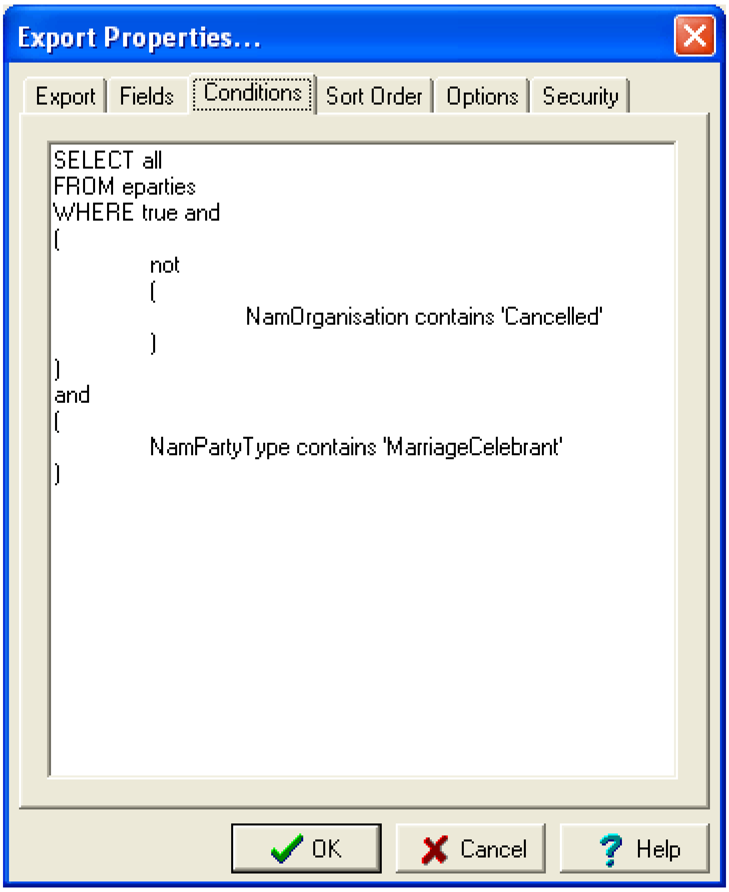
Finish Date:

Frequency: Annually

Day of the Year

Day 1 and Month January of Each Year

□ First Weekday of the Year



```
SELECT all
FROM eparties
WHERE true and
(
            not
            (
                        NamOrganisation contains 'Cancelled'
            )
)
and
(
            NamPartyType contains 'MarriageCelebrant'
)
```

Scenario 4

From time to time the Cancer Council requests a data file containing a list of all deceased persons who died of cancer.

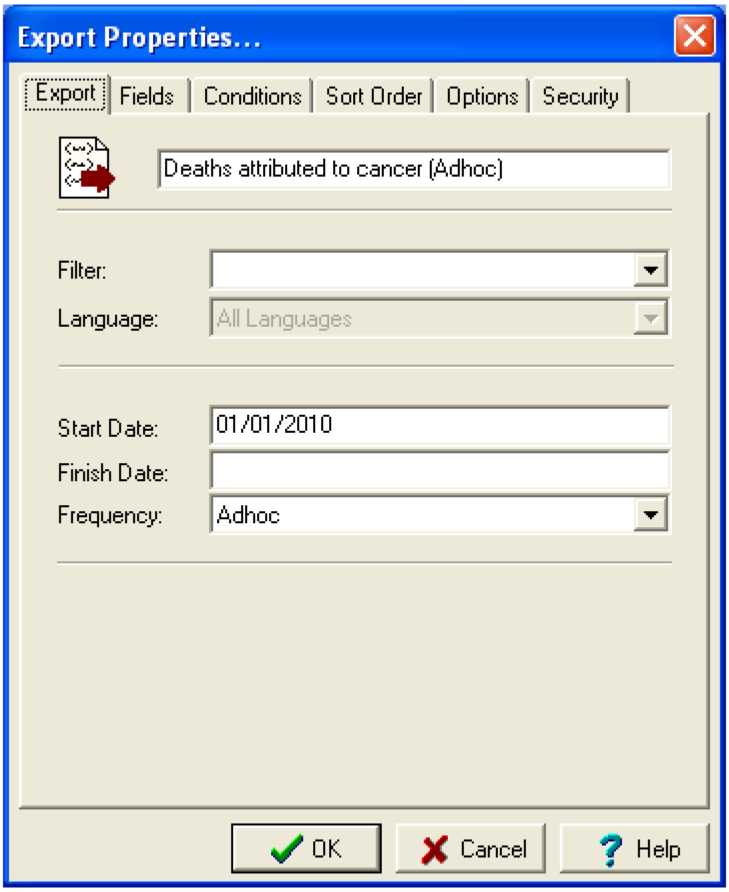
Solution

The data exports are not required on a regular basis, so the `Adhoc` export frequency should be used. As a schedule is not defined for Adhoc exports you will need to run an Unscheduled Export to produce the required data file. See Unscheduled Exports (page 27) for more details.

The Conditions statement needs to retrieve all cancer related deaths. The Cause of Death field allows us to find the required Deaths records. A suitable Conditions statement would be:

```
SELECT all
FROM   edeaths
WHERE ImmediateCauseOfDeath CONTAINS 'Cancer'
```

The required Export and Conditions tabs are:



Deaths attributed to cancer (Adhoc)

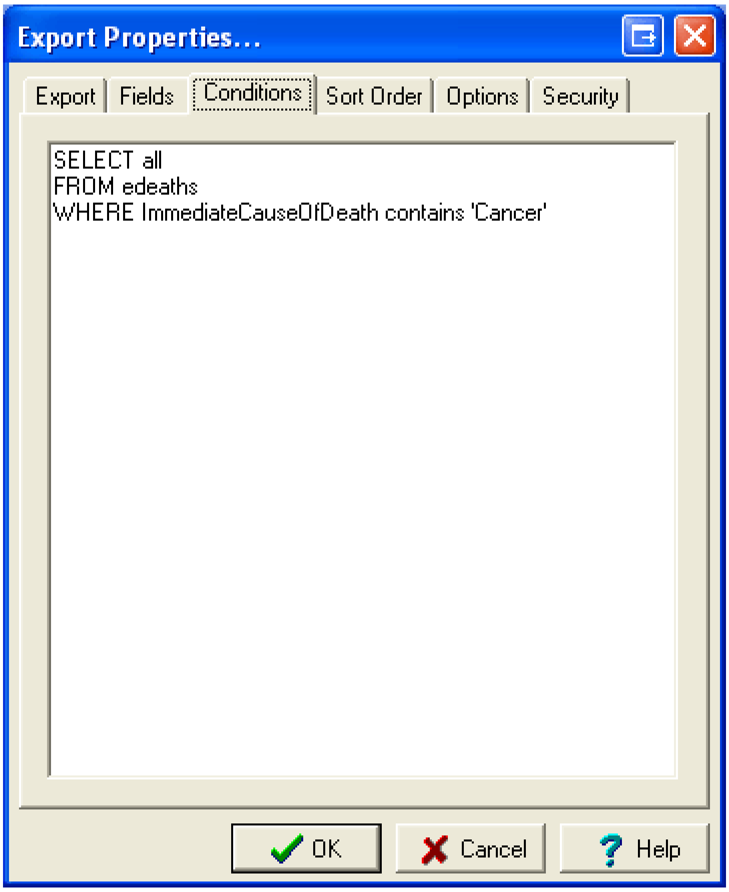
Filter:

Language: All Languages

Start Date: 01/01/2010

Finish Date:

Frequency: Adhoc



```
SELECT all
FROM edeaths
WHERE ImmediateCauseOfDeath contains 'Cancer'
```

SECTION 3

Unscheduled Exports

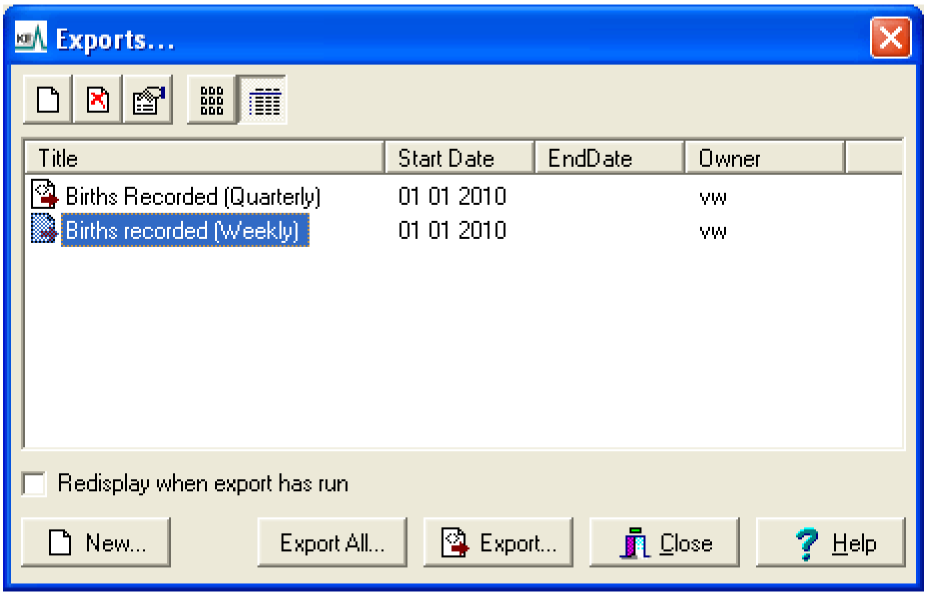
The Scheduled Exports facility allows you to have data exports occur at defined date intervals. It may also be useful to perform a data export immediately rather than at the scheduled time. In this case, we generate the export data by running the export manually. These manual runs are known as Unscheduled Exports.

Another difference between an Unscheduled and Scheduled Export is that Unscheduled Exports use the current matching records as the data source, whereas a Scheduled Export uses the Conditions statement to retrieve the records to be exported. Unscheduled Exports ignore the Conditions statement.

Running Unscheduled Exports

To run an Unscheduled Export:

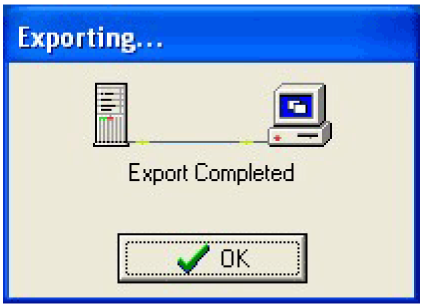
1. Search for or otherwise list a group of records.
2. Select **Tools>Exports** in the Menu bar.
   -OR-
   Use the keyboard shortcut, `ALT+T+X`.
   The Exports dialogue box displays:



Title	Start Date	EndDate	Owner
Births Recorded (Quarterly)	01 01 2010		vw
Births recorded (Weekly)	01 01 2010		vw

□ Redisplay when export has run

3. Select the export to run.
4. Select **Export** to export the current record or selected records
   -OR-
   Select **Export All** to export all listed records.
   The export is run with a status dialogue box displayed. When the export has finished, a completed dialogue box displays:



Export Completed

5. Select **OK** .
   The results of the export are stored on the Vitalware server. To view the results it is necessary to look in the Exports module. See Viewing Export Results (page 29) for more details.

SECTION 4

Viewing Export Results

Scheduled Exports are run automatically by Vitalware. For each export executed a record is created in the Exports module. The record includes:

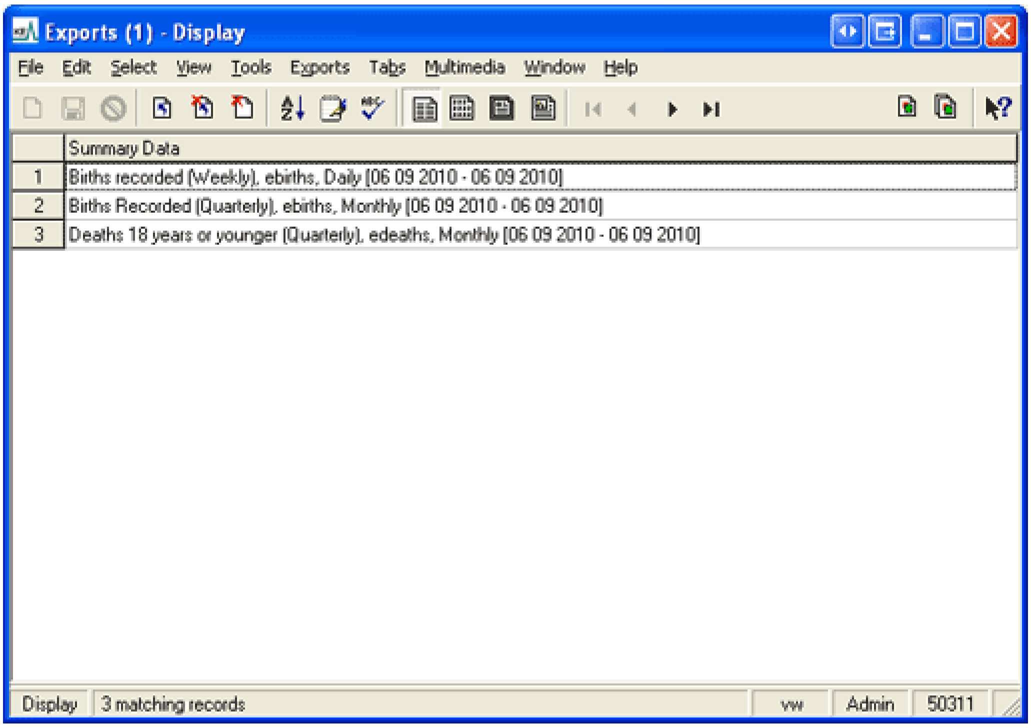
- Name of the Scheduled Export
- Number of records exported
- The module from which the records were exported
- The filter used (if any)
- The `#STARTDATE#` and `#ENDDATE#` values used
- Any errors that occurred
- The export data files

View Export Results

1. Select **Exports** from the Command Centre.
   The Exports module displays in Search mode.
2. Enter the name of the Export to view in *Schedule: (Export Details)*.

> If you're looking for a recently run export, you could run a search without entering search terms in *Schedule: (Export Details)* as exports are listed with most recent exports first.

3. Select **File>Search** in the Menu bar
   -OR-
   Select **Search** in the Toolbar
   -OR-
   Use the keyboard shortcut, `CTRL+F`.
   A list of matching exports displays:

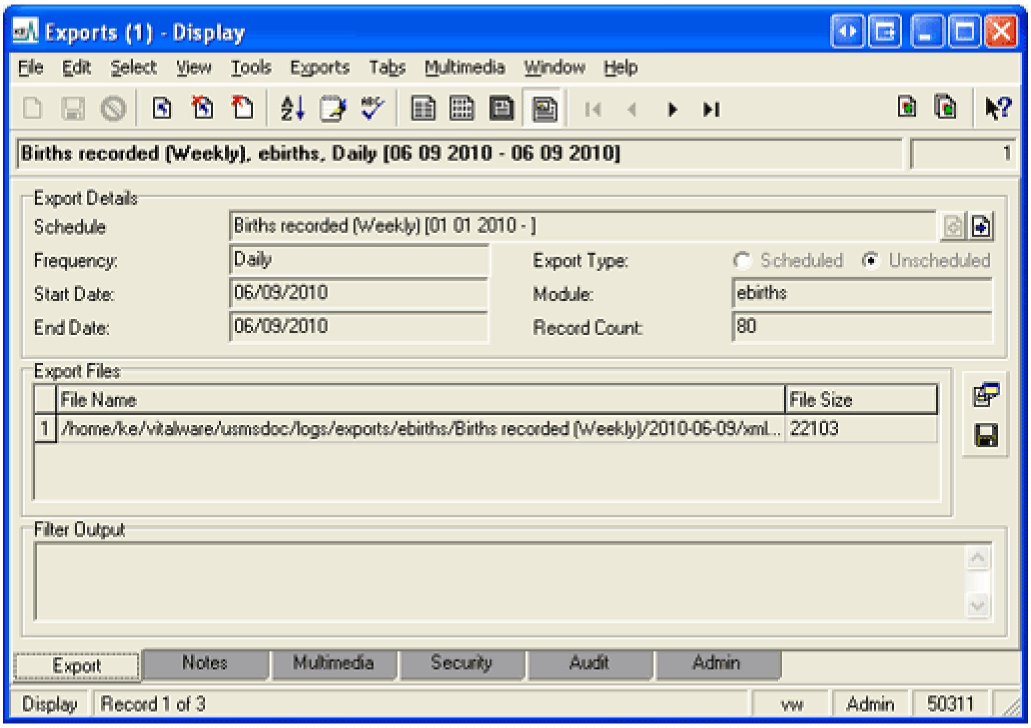

Exports (1) - Display
File Edit Select View Tools Exports Tabs Multimedia Window Help

	Summary Data
1	Births recorded (Weekly), ebirths, Daily [06 09 2010 - 06 09 2010]
2	Births Recorded (Quarterly), ebirths, Monthly [06 09 2010 - 06 09 2010]
3	Deaths 18 years or younger (Quarterly), edeaths, Monthly [06 09 2010 - 06 09 2010]

Display
3 matching records
vw
Admin
50311

4. Select the export you want to view.
5. Select **View>Details** in the Menu bar
   -OR-
   Select **View Details** in the Toolbar
   -OR-
   Use the keyboard shortcut, `ALT+V+D`.

The details of the export displays:



Exports (1) - Display
File Edit Select View Tools Exports Tabs Multimedia Window Help

Births recorded (Weekly), ebirths, Daily [06 09 2010 - 06 09 2010] 1

Export Details

Schedule: Births recorded (Weekly) [01 01 2010 -]

Frequency: Daily

Start Date: 06/09/2010

End Date: 06/09/2010

Export Type: ○ Scheduled ● Unscheduled

Module: ebirths

Record Count: 80

Export Files

	File Name	File Size
1	/home/ke/vitalware/usmsdoc/logs/exports/ebirths/Births recorded (Weekly)/2010-06-09/xml...	22103

Filter Output

Export Notes Multimedia Security Audit Admin
Display Record 1 of 3 vw Admin 50311

The export files generated in the export process are listed in the *Export Files* table. The files are stored on the Vitalware server. Any messages or errors are shown in the *Filter Output* box.

View Export Data

1. Click the row in the *Export Files* table with the file to be viewed.
2. Select **Launch Viewer** next to the Export Files table.
   -OR-
   Select **Exports>Launch** in the Menu bar and select the file to be viewed from the sub-menu that displays.

The application / viewer associated with the file extension is invoked to display the file.

Save all Export Data files

1. Select **Save All** 💾 beside the *Export Files* table
   -OR-
   Select **Exports>Save>All** in the Menu bar.
   The Browse for Folder dialogue box displays.
2. Choose the directory into which all export files will be saved.
3. Select **OK** OK.

Save an Export Data file

1. Select **Exports>Save** in the Menu bar
   -OR-
   Use the keyboard shortcut, ALT+X+S.
   The Save sub-menu lists all export data files.
2. Select the file to save from the sub-menu.
   The Save As dialogue box displays.
3. Choose the location to save the export file.
4. Select **Save** Save.

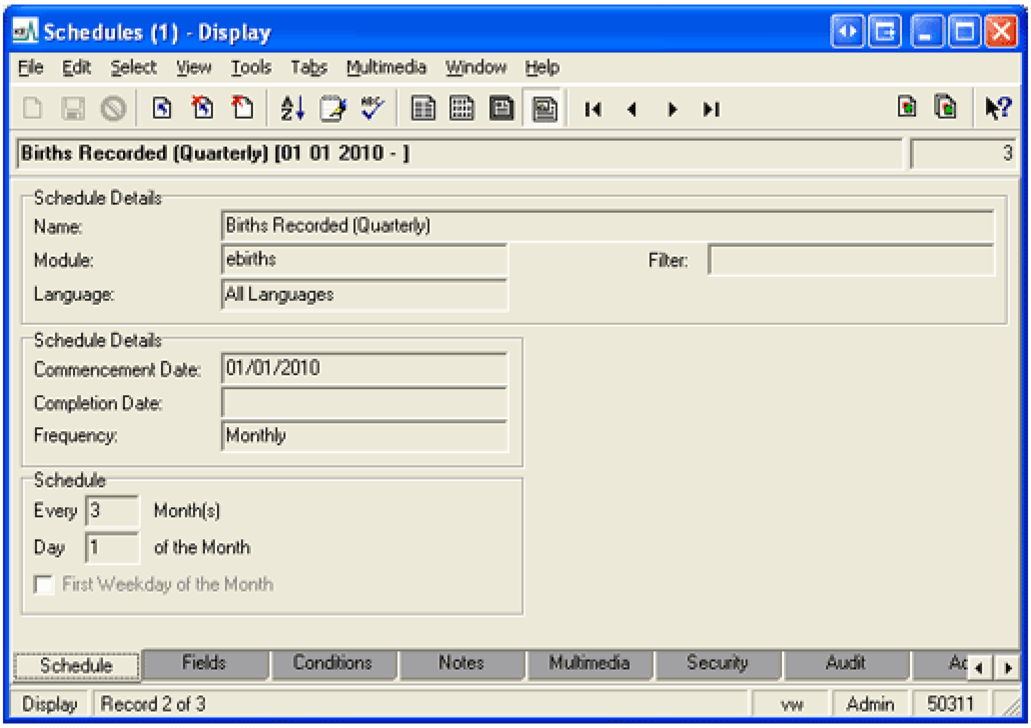
SECTION 5

The Schedules and Exports Modules

The Scheduled Exports facility uses two modules to store export details:

- Schedules Module Schedules

  The Schedules module holds one record for each export defined in the Export Properties dialogue box:



Births Recorded (Quarterly) [01 01 2010 -] 3

Schedule Details

Name: Births Recorded (Quarterly)

Module: ebirths

Filter:

Language: All Languages

Schedule Details

Commencement Date: 01/01/2010

Completion Date:

Frequency: Monthly

Schedule

Every 3 Month(s)

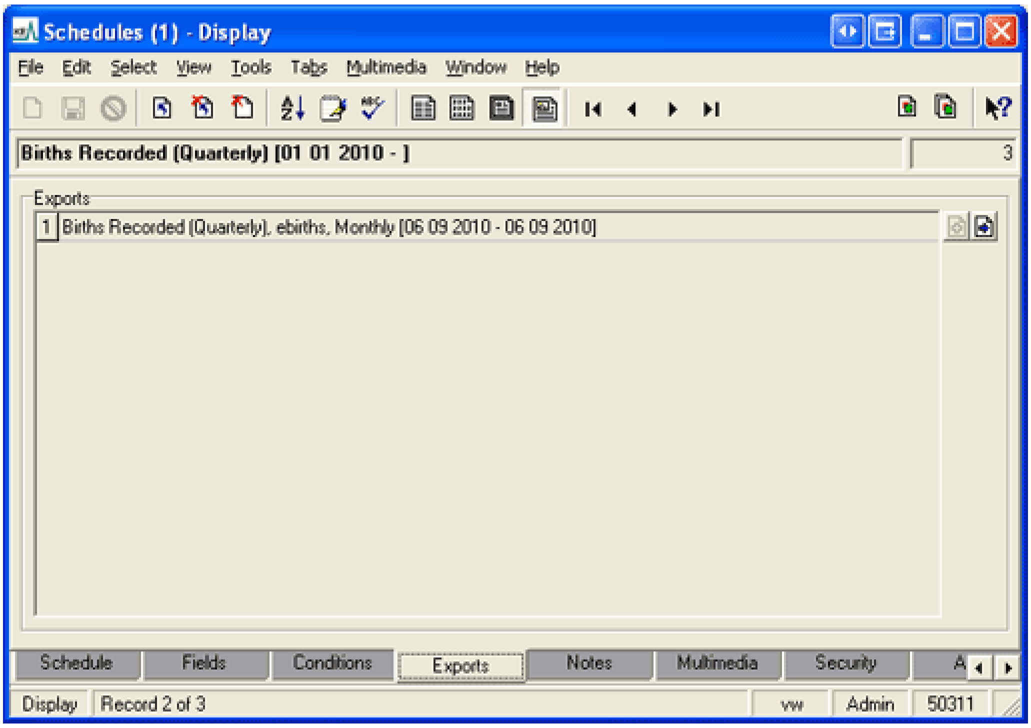
Day 1 of the Month

☐ First Weekday of the Month

Display | Record 2 of 3 | vw | Admin | 50311

  The Scheduled Export settings on the Schedule, Fields, Conditions and Exports tabs are read-only as the values are maintained via the Export Properties dialogue box; all other Tabs are editable.

The Exports tab lists all data exports performed via this Schedules record. The list is sorted from most recent export to the oldest. Both Scheduled and Unscheduled Exports are shown:



Schedules (1) - Display

Births Recorded (Quarterly) [01 01 2010 -] 3

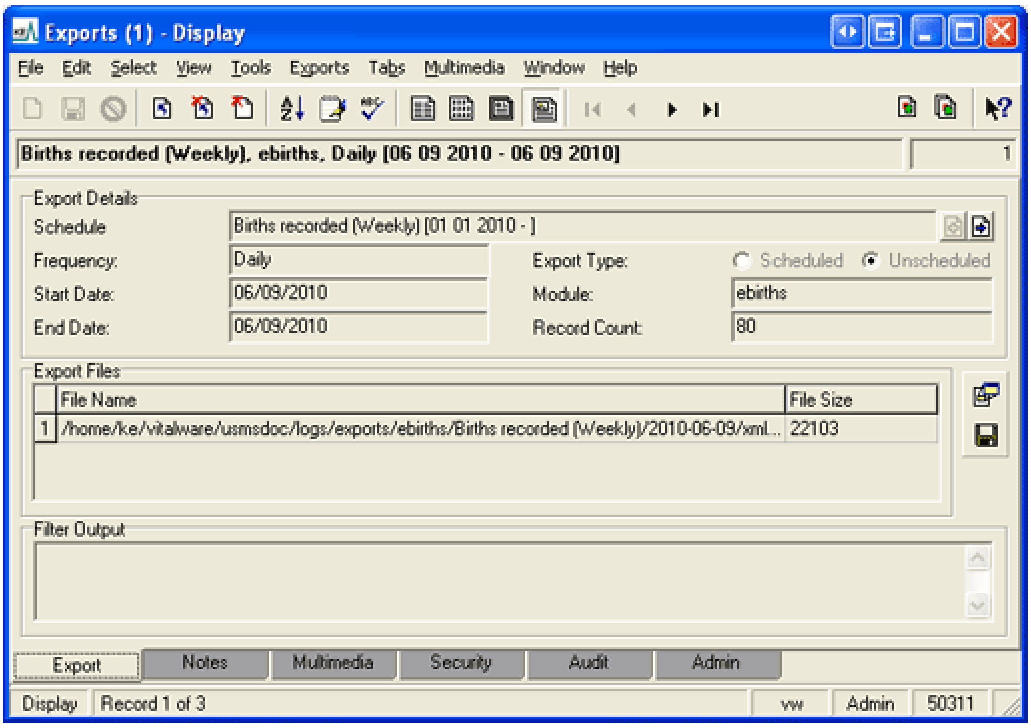
Exports

1 Births Recorded (Quarterly), ebirths, Monthly [06 09 2010 - 06 09 2010]

Display | Record 2 of 3 | vw | Admin | 50311

- Exports Module Exports

The Exports module holds one record per export executed. An Exports record is linked to its associated Schedules record:


File Edit Select View Tools Exports Tabs Multimedia Window Help
Export
Notes
Multimedia
Security
Audit
Admin

Exports (1) - Display

Births recorded (Weekly), ebirths, Daily [06 09 2010 - 06 09 2010] 1

Export Details

Schedule: Births recorded (Weekly) [01 01 2010 -]

Frequency: Daily

Start Date: 06/09/2010

End Date: 06/09/2010

Export Type: ○ Scheduled ● Unscheduled

Module: ebirths

Record Count: 80

Export Files

	File Name	File Size
1	/home/ke/vitalware/usmsdoc/logs/exports/ebirths/Births recorded (Weekly)/2010-06-09/xml...	22103

Filter Output

Display | Record 1 of 3 | vw | Admin | 50311

The *Export Files* table lists all files created when the export was run, along with their size. Any messages or errors generated display in the *Filter Output* box. Data values displaying on the Export tab are read-only; all other Tabs are editable.

As with other Vitalware modules, the Schedules and Exports modules provide reporting, sorting, etc.

SECTION 6

Creating Filters

When defining a Scheduled Export in the Export Properties dialogue box, it is possible to select a filter that transforms the standard XML export data format into another format suitable for delivery (e.g. CSV). If a filter is not specified, the export files contain Vitalware XML formatted data (as used by the Vitalware reporting facility). As final delivery formats may vary widely, System Administrators may need to create localised filters. In this section we look at how to add a filter into the Scheduled Exports facility.

Where are filters stored?

Filters, which are programs or scripts, reside under the `etc/exports` or `local/etc/exports` directory on the Vitalware server.

They may be module specific or may be applicable to all modules. Module specific filters are stored under a sub-directory named after the module to which they apply. For example, Parties module specific filters would be located under `etc/exports/eparties` or `local/etc/exports/eparties`.

> When adding your own filters to the Vitalware server, they should be placed under `local/etc/exports` as `etc/exports` is reserved for use by Vitalware upgrades.

The name of the file containing the program or script is the label displayed in the Filters drop list when specifying a new Scheduled Export.

How is a filter invoked?

When an export is performed the initial data export is in Vitalware XML format. The data is placed in a temporary file on the Vitalware server and passed as an argument to the filter specified for the Scheduled Export. The filter receives only one argument, the name of the file containing the exported data.

Several environment variables are set and available to the filter when it executes:

EXPORTTITLE The title given to the Scheduled Export when it was created.

EXPORTIRN The IRN (Internal Record Number) of the Schedules record used to generate the exported data file. Using `EXPORTIRN` it is possible to retrieve details from the Schedules record.

EXPORTTABLE The module from which the data was exported.

EXPORTSTARTDATE The value of `#STARTDATE#` used by the export process.

EXPORTENDDATE The value of `#ENDDATE#` used by the export process.

EXPORTRECORDCOUNT The number of records exported.

Filter Output and Errors

If the filter is unable to process the exported data, any error messages should be written to `STDERR` and the export terminated with a non-zero exit status. This notifies Vitalware that the filter did not complete successfully and an export failure is recorded. Any text written to `STDERR` is displayed in the *Filter Output* box for an Exports record in the Exports module.

Once the filter has processed the export data it should print out to `STDOUT` a list of all the files generated. These files are moved under the `logs/exports` directory on the Vitalware server. The file names are displayed in the *Export Files* table in the Exports record created. If the filter completed successfully, it should return a zero exit status.

Filtering Data

The filter may manipulate the export data in any way required. It may email the data (via `sendmail`), print it (via `lpr` or `lp`) or even send it to remote machines (via `rsh` or `ssh`). A copy of the data exported should always be attached to the Export record so it can be reviewed at a later date, if required.

An example filter

In this example filter we convert the export data from XML to CSV (Comma Separated Values) and email the file(s) produced to a mail alias address (`csvexport@client.org`) on our local mail server. The mail server will then forward the email onto a host of recipients.

We can use `texxmlodbc` to convert the export XML to CSV files. The filter is written using the perl scripting language as it provides a nice package for sending multi-part email messages.

The filter is:

```
#!/usr/bin/perl

use strict;
use warnings 'all';
use MIME::Lite;

#
#  First we convert the Vitalware XML to CSV and save
#  the list of files generated.
#
if (! open(FILES, "texxmlodbc --output-encoding=iso-8859-1
'$ARGV[0]' |"))
{
        print STDERR "Cannot execute texxmlodbc";
        exit(1);
}
my @files = <FILES>;
close(FILES);

#
#  Build up the message to send.
#
my $body = <<END;
The attached files were generated by the VW Scheduled Export
facility.
The details of the export are:

Title: $ENV{EXPORTTITLE}
Module: $ENV{EXPORTTABLE}
Start Date: $ENV{EXPORTSTARTDATE}
End Date: $ENV{EXPORTENDDATE}
END

#
#  Create an email message to csvexport@client.org
#
my $msg = MIME::Lite->new(
        To      => 'csvexport@client.org',
        Subject => 'VW Scheduled Export CSV Files',
        Type    => 'multipart/mixed'
        );
```

```
#
#  Add the email text first.
#
$msg->attach(
        Type    => 'TEXT',
        Data    => $body
        );

#
#  Add each file.
#
foreach my $file (@files)
{
        #
        #  Figure out the file mime type and attach it
        #
        chomp($file);
        my $type = ($file =~ /\.ini$/i) ? 'TEXT' : 'text/csv';
        $msg->attach(
                Type            => $type,
                Path            => $file,
                Disposition     => 'attachment'
                );
}

#
#  Send the email.
#
if (! $msg->send())
{
        print STDERR 'Cannot send email to csvexport@client.org';
        exit(1);
}
print STDERR 'Email sent to csvexport@client.org successfully';

#
#  Output the file list for the export record.
#
print join("\n", @files);
exit(0);
```

Notice how the filter outputs the names of the CSV files at the end so they can be attached to the Export record created. Once attached these files may be reviewed at any time.

Section 7

Cron setup

In order for Scheduled Exports to be produced the `vwexport` command must be run, typically early each morning. The server based cron facility provides this ability. The following crontab entry needs to be installed to ensure the correct functioning of the Scheduled Export facility:

```
0 1 * * * ~vw/bin/vwrun vwexport 2>&1 | ~vw/bin/vwrun vwlogger -t
"KE Vitalware Scheduled Exports Report" exports
```

where `~vw` is replaced with the full path to where Vitalware is installed on the server machine.

The time at which the command is run (1:00 am in the example above) should be adjusted to fit in with any Vitalware maintenance scheduled.

SECTION 8

Using vwexport

The `vwexport` command is the program used to execute Scheduled Exports. It interacts with the Schedules and Exports modules to ensure that data exports occur when scheduled. The program may be used in four different ways:

1. **Run all Scheduled Exports**

   Usage: `vwexport`

   Any Scheduled Exports required to be run will be executed. The current date is used to determine what exports are required. This form of the command is used by cron on a daily basis to ensure all Scheduled Exports for the given day are performed.

2. **Run an Unscheduled Export**

   Usage: `vwexport -f filename -g recordcount -i irn`

   The `filename` argument is the path to a file containing the export data in Vitalware XML format. A suitable file can be generated using the Vitalware reporting facility and selecting a Report *Type* of **Export XML Document**. The `recordcount` is the number of records in the XML file and `irn` is the IRN of the Schedules record to execute. An Unscheduled Export run within Vitalware uses this form of `vwexport`.

3. **Run Scheduled Exports for a specific date**

   Usage: `vwexport -R date [-i irn]`

   The `date` argument indicates the date to use to determine what Scheduled Exports need to be run. The date format expected is `yyyy/mm/dd`. An optional IRN of a Schedules record may be supplied to force a particular Scheduled Export to execute using the supplied date. This form of `vwexport` is useful for re-running Scheduled Exports for a given date, generally because of problems with the nightly automatic run. It can also be used for internal testing of `vwexport`.

4. **Run a specific Scheduled Export**

   Usage: `vwexport -i irn [-s startdate -e enddate]`

   The `irn` argument is the IRN of a Schedules record to be executed. Optional `startdate` and `enddate` values (in `yyyy/mm/dd` format) may be used to override the computed `#STARTDATE#` and `#ENDDATE#` values respectively. This form of `vwexport` is very useful for testing filters as it allows a specific schedule to be run without waiting for the Scheduled Export date to arrive.

In all usages of `vwexport` the appropriate Exports record is created for each export executed.

Index

Vitalware Documentation

Appending data using the Vitalware Import Facility

Document Version 1

Vitalware Version 2.1.02

www.kesoftware.com

Contents

SECTION 1

Overview

KE Vitalware 2.0.01 added the ability to import records specified in either XML (eXtensible Markup Language) or CSV (Comma Separated Values). The new Import Facility enabled records to be created, including the creation of any records referenced. Hence it is possible to import a new POS record and have a Party record created for the *Assigned By* field if the Party record does not already exist. The Import Facility also provided a mechanism for updating records. Using the update feature it is possible to replace the contents of an existing field with new data. Unfortunately the complete contents of the field are replaced with the data imported.

In some instances it may be useful to update a list of values by:

- appending values after a given position or after the last value.
- prepend values before a given position or before the first value.
- replace values at a given position.

Vitalware 2.1.02 extends the Import Facility, allowing tables and nested tables of values to be updated rather than replaced. It is now possible to append, prepend and replace values at given positions with imported data.

A new *group* mechanism allows separate columns to be tied together when appending data, ensuring all values are placed on the same row in each column. For example, when updating a POS record you may want the *Description: (Task Information)* and *Assigned By: (Task Information)* values to appear on the same row, as each piece of information is related to the other.

In the following section we will look at the extensions added to the Import Facility to support the appending of data for both XML and CSV based data sources.

SECTION 2

Appending data

The Vitalware Import Facility allows XML or CSV to be used for defining records for importing. The XML format used is the same as that produced by the Vitalware Reporting Facility, while CSV is a de facto standard for data interchange. Consider the XML below:

```
<table table='eparties'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='NamOtherNames_tab'>
            <tuple>
                <atom>Smith</atom>
            </tuple>
        </table>
    </tuple>
</table>
```

or the equivalent CSV:

irn	NamOtherNames_tab(1)
151	Smith

When imported, all values in the *NamOtherNames_tab* column will be replaced with the value `Smith`. To provide support for updating rows in a table a new row attribute has been added to the `<tuple>` tag for XML, allowing a row position to be specified. The row attribute provides a mechanism for indicating what type of update should be applied and which row is affected. The format of the attribute is:

<tuple row='*value*'>

For a CSV data source the row number appears between brackets. CSV allows two formats to be used to specify the row attribute:

(row='*value*')

or the shorter form:

(*value*)

While double quotes may be used to enclose the row *value*, single quotes are recommended for CSV files as the CSV format uses double quotes to enclose data values. When saving CSV files in Excel, incorrect output will be produced if double quotes are used when specifying the row *value*.

The *value* of the row attribute may be:

nnn where **nnn** is a row number. The number indicates the row position in the list of values to be modified. The first row is numbered 1. In essence this setting replaces the current value at that position.

+ indicates the row position is after the last value in the table. Any data will be appended to the list of values.

- indicates the row position is before the first value in the table. Any data added will be put at the start of the list with existing values moved down.

= indicates the row position is row one. Any data added will replace the existing value at this position.

nnn+ appends any data after row number *nnn*.

nnn- prepends any data before row number *nnn*.

nnn= replaces any data at row number *nnn*.

In all cases, if the new row number does not exist, it is created. For example, if the row setting is:

```
<tuple row='12+'>
```

or for CSV:

`(row='12+')` or `(12+)`

and there are not twelve values in the table, the table would be padded out to twelve values (with empty rows) and the new value appended, creating a thirteenth row. If a row attribute is not defined, the default behaviour is to append to the end of the table.

In order to provide backwards compatibility with the existing Import Facility and to allow all values in a table to be replaced, the first <tuple> in a table for XML, or the first row specifier for CSV, define whether the values in the table are being updated or whether all values are being replaced. If the row attribute is not specified in XML, or just a row number is specified, then the contents of the column will be cleared and the imported values added. If a row attribute is provided and it contains an update operator (that is **+**, **-** or **=** with or without a leading row number), the contents of the table are updated, with the existing values retained.

The following section contains examples on how to replace or update the contents of a table using the new row attribute. The examples provide both XML and CSV solutions.

Updating tables

The examples below show how the row attribute can be used to overwrite or update the contents of table based columns.

Example one - Replacing the contents of a table

XML

```
<table table='eparties'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='NamOtherNames_tab'>
            <tuple>
                <atom>new othername 1</atom>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

irn	NamOtherNames_tab(1)
151	new othername 1

As the row attribute value in the *NamOtherNames_tab* column does not contain an update modifier (that is, **+**, **-** or **=**), the contents of the column are cleared before adding in the new data. The tables below show before and after cases for the above import:

Before

1	othername 1
2	othername 2
3	othername 3
4	othername 4

After

1	new othername 1

Example two - Replacing the contents of a table and skipping rows

XML

```
<table table='eparties'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='NamOtherNames_tab'>
            <tuple row="1">
                <atom>new othername 1</atom>
            </tuple>
            <tuple row="3">
                <atom>new othername 3</atom>
            </tuple>
            <tuple row="5">
                <atom>new othername 5</atom>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

irn	NamOtherNames_tab(1)	NamOtherNames_tab(3)	NamOtherNames_tab(5)
151	new othername 1	new othername 3	new othername 5

Once again as the first row attribute does not contain an update modifier, the contents of *NamOtherNames_tab* will be cleared before the imported values are added. The use of the row attribute allows specific row positions to be set. The example below shows before and after cases of the above import:

Before

1	othername 1
2	othername 2
3	othername 3
4	othername 4

After

1	new othername 1
2	
3	new othername 3
4	
5	new othername 5

Example three - Appending a value to the end of a table

XML

```
<table table='eparties'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='NamOtherNames_tab'>
            <tuple row="+">
                <atom>append othername 1</atom>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

irn	NamOtherNames_tab(+)
151	append othername 1

As the row attribute value contains an update modifier, +, the existing contents of the *NamOtherNames_tab* field are retained and added to:

Before

1	othername 1
2	othername 2
3	othername 3
4	othername 4

After

1	othername 1
2	othername 2
3	othername 3
4	othername 4
5	append othername 1

Example four - Appending multiple values to the end of a table

XML

```
<table table='eparties'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='NamOtherNames_tab'>
            <tuple row="+">
                <atom>append othername 1</atom>
            </tuple>
            <tuple row="+">
                <atom>append othername 2</atom>
            </tuple>
            <tuple row="+">
                <atom>append othername 3</atom>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

irn	NamOtherNames_tab(+)	NamOthernames_tab(+)	NamOtherNames_tab(+)
151	append othername 1	append othername 2	append othername 3

Since the first row attribute contains an update modifier (the + modifier), the imported data updates the existing table values. Notice the use of the append modifier for each row to be appended:

Before

1	othername 1
2	othername 2
3	othername 3
4	othername 4

After

1	othername 1
2	othername 2
3	othername 3
4	othername 4
5	append othername 1
6	append othername 2
7	append othername 3

Example five - Prepending a value to the front of a table

XML

```
<table table='eparties'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='NamOtherNames_tab'>
            <tuple row="-">
                <atom>prepend othername 1</atom>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

irn	NamOtherNames_tab(-)
151	prepend othername 1

Inserting a value into the first position in a table column is similar to appending, except that the prepend row operator (-) is used. When the value is inserted, existing values are moved down the table:

Before

1	othername 1
2	source 2
3	othername 3
4	othername 4

After

1	prepend othername 1
2	othername 1
3	othername 2
4	othername 3
5	othername 4

Example six - Replacing values in a table

XML

```
<table table='eparties'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='NamOtherNames_tab'>
            <tuple row="2=">
                <atom>replace othername 1</atom>
            </tuple>
            <tuple row="3=">
                <atom>replace othername 2</atom>
            </tuple>
            <tuple row="6=">
                <atom>replace othername 3</atom>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

irn	NamOtherNames_tab(2=)	NamOtherNames_tab(3=)	NamOtherNames_tab(6=)
151	replace othername 1	replace othername 2	replace othername 3

The = row operator is used to replace the contents of a row in a table. If the row does not exist, it is created and the appropriate value set:

Before

1	othername 1
2	othername 2
3	othername 3
4	othername 4

After

1	othername 1
2	replace othername 1
3	replace othername 2
4	othername 4
5	
6	replace othername 3

Example seven - Appending a reference to the end of a table

XML

```
<table table='eparties'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='AssAssociationRef_tab'>
            <tuple row="+">
                <atom name="NamFirst">firstname</atom>
                <atom name="NamLast">lastname</atom>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

irn	AssAssociationRef_tab(+).NamFirst	AssAssociationRef_tab(+).NamLast
151	firstname	lastname

All the update modifiers available for tables of values may be used for tables of references. For CSV data sources, columns with the same name and row attribute are handled as one update. For example, the CSV row attributes above both use the append operator (+). This does not result in two new rows, rather the same row is used to contain the reference:

Before

1	reference 1
2	reference 2
3	reference 3
4	reference 4

After

1	reference 1
2	reference 2
3	reference 3
4	reference 4
5	append reference 1

Updating nested tables

The mechanism for updating nested tables, that is tables within tables, is very similar to updating a single table. The only difference is that nested tables have an outer row number and an inner row number. Each outer row contains a table of inner row values:

Outer Row	Inner Row	Data
1	1	opinion 1, comment 1
	2	opinion 1, comment 2
	3	opinion 1, comment 3
2	1	opinion 2, comment 1
3	1	opinion 3, comment 1
	2	opinion 3, comment 2

As you can see from the table above, outer row one contains a table with three values, while outer row two has one value and outer row three, two values. The XML representation of the above table would be:

```
<table table='ebirths'>
    <tuple>
        <table name='Comment_nesttab'>
            <tuple>
                <table>
                    <tuple>
                        <atom>opinion 1, comment 1</atom>
                    </tuple>
                    <tuple>
                        <atom>opinion 1, comment 2</atom>
                    </tuple>
                    <tuple>
                        <atom>opinion 1, comment 3</atom>
                    </tuple>
                </table>
            </tuple>
            <tuple>
                <table>
                    <tuple>
                        <atom>opinion 2, comment 1</atom>
                    </tuple>
                </table>
            </tuple>
            <tuple>
                <table>
                    <tuple>
                        <atom>opinion 3, comment 1</atom>
                    </tuple>
                    <tuple>
                        <atom>opinion 3, comment 2</atom>
                    </tuple>
        </table>
      </tuple>
    </table>
  </tuple>
</table>
```

The nested table *Comment_nesttab* starts with the `<table name='Comment_nesttab'>` tag. Each outer row is enclosed in a `<tuple>` tag. The outer rows are coloured green. Within each outer row is a table of inner rows. The inner rows are enclosed in red `<tuple>` tags. When updating nested tables both the outer and inner row `<tuple>` tags may have row attributes. Hence it is possible to append/prepend/replace outer and/or inner rows.

The equivalent CSV representation is shown below. The table has been turned on its side for ease of viewing. The column names should appear in the first row rather than the first column:

irn	151
Comment_nesttab(1:1)	opinion 1, comment 1
Comment_nesttab(1:2)	opinion 1, comment 2
Comment_nesttab(1:3)	opinion 1, comment 3
Comment_nesttab(2:1)	opinion 2, comment 1
Comment_nesttab(3:1)	opinion 3, comment 1
Comment_nesttab(3:2)	opinion 3, comment 2

The outer and inner rows for nested tables in CSV are recorded after the column name enclosed in brackets separated by a colon. The outer row is shown in green, while the inner row is red. The example above uses the short form for the row attributes. The long form would look like:

```
Comment_nesttab(row='1':row='1')
```

As with XML the outer and/or inner rows may contain update modifiers. The following examples detail some common uses.

Example one - Replacing the contents of a nested table

XML

```
<table table='ebirths'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='Comment_nesttab'>
            <tuple>
                <table>
                    <tuple>
                        <atom>new opinion 1, new comment 1</atom>
                    </tuple>
                </table>
            </tuple>
            <tuple>
                <table>
                    <tuple>
                        <atom>new opinion 2, new comment 1</atom>
                    </tuple>
                    <tuple>
                        <atom>new opinion 2, new comment 2</atom>
                    </tuple>
                </table>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

irn	Comment_nesttab(1:1)	Comment_nesttab(2:1)	Comment_nesttab(2:2)
151	new opinion 1, new comment 1	new opinion 2, new comment 1	new opinion 2, new comment 2

As the first outer row and first inner row do not contain row attributes the current contents of the *Comments_nesttab* column will be cleared:

Before

1	1	opinion 1, comment 1
	2	opinion 1, comment 2
	3	opinion 1, comment 3
2	1	opinion 2, comment 1
3	1	opinion 3, comment 1
	2	opinion 3, comment 2

After

1	1	new opinion 1, new comment 1
2	1	new opinion 2, new comment 1
	2	new opinion 2, new comment 2

Example two - Appending an outer row to a nested table

XML

```
<table table='ebirths'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='Comment_nesttab'>
            <tuple row="+">
                <table>
                    <tuple>
                        <atom>append opinion 1, comment 1</atom>
                    </tuple>
                    <tuple>
                        <atom>append opinion 1, comment 2</atom>
                    </tuple>
                </table>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

irn	Comment_nesttab(+:1)	Comment_nesttab(+:2)
151	append opinion 1, comment 1	append opinion 1, comment 2

The first outer row has a row attribute with a value of +, indicating an outer row is to be appended to the existing data. The inner table contains the values to be added:

Before

1	1	opinion 1, comment 1
	2	opinion 1, comment 2
	3	opinion 1, comment 3
2	1	opinion 2, comment 1
3	1	opinion 3, comment 1
	2	opinion 3, comment 2

After

1	1	opinion 1, comment 1
	2	opinion 1, comment 2
	3	opinion 1, comment 3
2	1	opinion 2, comment 1
3	1	opinion 3, comment 1
	2	opinion 3, comment 2
4	1	append opinion 1, comment 1
	2	append opinion 1, comment 2

Example three - Appending values to an inner table in a nested table

XML

```
<table table='ebirths'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='Comment_nesttab'>
            <tuple row="2">
                <table>
                    <tuple row="+">
                        <atom>opinion 2, append comment 1</atom>
                    </tuple>
                    <tuple>
                        <atom>opinion 2, append comment 2</atom>
                    </tuple>
                </table>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

irn	Comment_nesttab(2:+)	Comment_nesttab(2:+)
151	opinion 2, append comment 1	opinion 2, append comment 2

The outer row contains a fixed row number while the first inner row contains an update modifier, the append (+) modifier. The combination indicates that values are to be appended to the end of the second outer row:

Before

1	1	opinion 1, comment 1
	2	opinion 1, comment 2
	3	opinion 1, comment 3
2	1	opinion 2, comment 1
3	1	opinion 3, comment 1
	2	opinion 3, comment 2

After

1	1	opinion 1, comment 1
	2	opinion 1, comment 2
	3	opinion 1, comment 3
2	1	opinion 2, comment 1
	2	opinion 2, append comment 1
	3	opinion 2, append comment 2
3	1	opinion 3, comment 1
	2	opinion 3, comment 2

Example four - Prepending an outer row to a nested table

XML

```
<table table='ebirths'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='Comment_nesttab'>
            <tuple row="2-">
                <table>
                    <tuple>
                        <atom>prepend opinion 2, comment 1</atom>
                    </tuple>
                    <tuple>
                        <atom>prepend opinion 2, comment 2</atom>
                    </tuple>
                </table>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

irn	Comment_nesttab(2-:1)	Comment_nesttab(2-:2)
151	prepend opinion 2, comment 1	prepend opinion 2, comment 2

The outer row attribute value of `2-` will prepend an outer row before row two and move the existing second row and subsequent rows down. The inner table contains the values to be set:

Before

1	1	opinion 1, comment 1
	2	opinion 1, comment 2
	3	opinion 1, comment 3
2	1	opinion 2, comment 1
3	1	opinion 3, comment 1
	2	opinion 3, comment 2

After

1	1	opinion 1, comment 1
	2	opinion 1, comment 2
	3	opinion 1, comment 3
2	1	prepend opinion 2, comment 1
	2	prepend opinion 2, comment 2
3	1	opinion 2, comment 1
4	1	opinion 3, comment 1
	2	opinion 3, comment 2

Example five - Replacing an outer row in a nested table

XML

```
<table table='ebirths'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='Comment_nesttab'>
            <tuple row="2=">
                <table>
                    <tuple>
                        <atom>replace opinion 2, comment 1</atom>
                    </tuple>
                    <tuple row="3">
                        <atom>replace opinion 2, comment 3</atom>
                    </tuple>
                </table>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

irn	Comment_nesttab(2=:1)	Comment_nesttab(2=:3)
151	replace opinion 2, comment 1	replace opinion 2, comment 3

The 2= row attribute on the outer row indicates row two is to be replaced. Note the use of the row attribute in the inner table to skip row two (causing a blank value to be created):

Before

1	1	opinion 1, comment 1
	2	opinion 1, comment 2
	3	opinion 1, comment 3
2	1	opinion 2, comment 1
3	1	opinion 3, comment 1
	2	opinion 3, comment 2

After

1	1	opinion 1, comment 1
	2	opinion 1, comment 2
	3	opinion 1, comment 3
2	1	replace opinion 2, comment 1
	2	
	3	replace opinion 2, comment 3
3	1	opinion 3, comment 1
	2	opinion 3, comment 2

Example six - Replacing an inner row in a nested table

XML

```
<table table='ebirths'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='Comment_nesttab'>
            <tuple row="3">
                <table>
                    <tuple row="1=">
                        <atom>opinion 3, replace comment 1</atom>
                    </tuple>
                    <tuple row="+">
                        <atom>opinion 3, append comment 1</atom>
                    </tuple>
                </table>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

irn	Comment_nesttab(3:1=)	Comment_nesttab(3:+)
151	opinion 3, replace comment 1	opinion 3, append comment 1

The outer row attribute value indicates row 3 is to be changed, while the first inner row indicates row one is to be replaced. The second inner row attribute will append a value to the end of the inner table:

Before

1	1	opinion 1, comment 1
	2	opinion 1, comment 2
	3	opinion 1, comment 3
2	1	opinion 2, comment 1
3	1	opinion 3, comment 1
	2	opinion 3, comment 2

After

1	1	opinion 1, comment 1
	2	opinion 1, comment 2
	3	opinion 1, comment 3
2	1	opinion 2, comment 1
3	1	opinion 3, replace comment 1
	2	opinion 3, comment 2
	3	opinion 3, append comment 1

Groups

The ability to append data to an existing list allows new values to be added to columns over time. One issue that may arise is the need to append values to more than one column, ensuring the values are all added at the same row position. Consider the following update:

XML

```
<table table='epos'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='TasDescription_tab'>
            <tuple row="+">
                <atom>append description 1</atom>
            </tuple>
        </table>
        <table name='TasResults_tab'>
            <tuple row="+">
                <atom>append results 1</atom>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

irn	TasDescription_tab(+)	TasResults_tab(+)
151	append description 1	append results 1

with the following result:

Before

1	description 1	results 1
2	description 2	results 2
3	description 3	

After

1	description 1	results 1
2	description 2	results 2
3	description 3	append results 1
4	append description 1	

The resulting data is almost certainly not what was envisioned. The two columns, *TasDescription_tab* and *TasResults_tab* are related and appear in the one grid. As such there is an implied relationship between values in a row. For example, all values in the first row apply to the first opinion, while values in the second row apply to the second opinion and so on. When appending a new opinion, all the added data values need to appear in the same row regardless of any empty values already present.

As the Import Facility does not know the relationship between columns, a new attribute, the group attribute, has been added allowing columns to be tied together. The attribute is also used to set the row position to be the first row to which data can be appended across all the columns in the group. For XML data, the group attribute appears in `<tuple>` tags along with the row attribute. For CSV, the group attribute is placed after the row attribute. It only makes sense to set a group where the row attribute is `row="+"`.

Consider the following update where a group is specified:

XML

```
<table table='epos'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='TasDescription_tab'>
            <tuple row="+" group="opinion">
                <atom>append description 1</atom>
            </tuple>
        </table>
        <table name='TasResults_tab'>
            <tuple row="+" group="opinion">
                <atom>append results 1</atom>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

irn	TasDescription_tab(+ group='opinion')	TasResults_tab(+ group='opinion')
151	append description 1	append results 1

Notice how the group attribute is defined on both the *TasDescription_tab* and *TasResults_tab* columns. The group value can be any string. All columns with the same group value are examined to determine the row position to use when appending data. More than one group may be used where disjoint (unrelated) groups of related columns exists. Each disjoint set of related columns should have its own group name. The above update will produce:

Before

1	description 1	results 1
2	description 2	results 2
3	description 3	

After

1	description 1	results 1
2	description 2	results 2
3	description 3	
4	append description 1	append results 1

which is probably what was required in the first place. Groups may be used with nested tables in the same way they can be with tables. A group may be applied to the outer and/or inner tuples in the nested table. The following update ensures the Description (*TasDescription_tab*), Results (*TasResults_tab*), and Assigned To (*TasPersonAssignedToRef_nesttab*) values all append to the same row:

XML

```
<table table='epos'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='TasDescription_tab'>
            <tuple row="+" group="opinion">
                <atom>append description 1</atom>
            </tuple>
        </table>
        <table name='TasResults_tab'>
            <tuple row="+" group="opinion">
                <atom>append results 1</atom>
            </tuple>
        </table>
        <table name='TasPersonAssignedToRef_nesttab'>
            <tuple row="+" group="opinion">
                <table>
                    <tuple>
                        <atom name="NamFirst">new opinion 1,
firstname 1</atom>
                        <atom name="NamLast">new opinion 1,
lastname 1</atom>
                    </tuple>
                    <tuple>
                        <atom name="NamFirst">new opinion 1,
firstname 2</atom>
                        <atom name="NamLast">new opinion 1,
lastname 2</atom>
                    </tuple>
                </table>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

The table has been turned on its side for ease of viewing.

irn	151
TasDescription_tab(+ group='opinion')	append description 1
TasResults_tab(+ group='opinion')	append results 1
TasPersonAssignedToRef_nesttab(+ group='opinion':1).NamFirst	new opinion 1, firstname 1
TasPersonAssignedToRef_nesttab(+ group='opinion':1).NamLast	new opinion 1, lastname 1
TasPersonAssignedToRef_nesttab(+ group='opinion':2).NamFirst	new opinion 1, firstname 2
TasPersonAssignedToRef_nesttab(+ group='opinion':2).NamLast	new opinion 1, lastname 2

Notice how the group attribute is applied to the outer row to ensure the new first name and last name values are added to the same row position as the description and results values:

Before

1	1	description 1	results 1	opinion 1, reference 1
	2			opinion 1, reference 2
	3			opinion 1, reference 3
2	1	description 2		opinion 2, reference 1

After

1	1	description 1	results 1	opinion 1, reference 1
	2			opinion 1, reference 2
	3			opinion 1, reference 3
2	1	description 2		opinion 2, reference 1
3	1	append description 1	append results 1	new opinion 1, reference 1
	2			new opinion 1, reference 2

The final point to cover concerning groups is that the row position at which new data is appended is calculated the first time the group is encountered in the data file. Once the row position is determined it is used for all columns in the group for the current import record. This means the group name used to bind columns together should only appear on the first row to be appended, rather than every row. If the group name appears on every row, each value updated will overwrite the previous value. Consider the update:

XML

```
<table table='epos'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='TasDescription_tab'>
            <tuple row="+" group="opinion">
                <atom>append description 1</atom>
            </tuple>
            <tuple row="+" group="opinion">
                <atom>append description 2</atom>
            </tuple>
        </table>
        <table name='TasResults_tab'>
            <tuple row="+" group="opinion">
                <atom>append results 1</atom>
            </tuple>
            <tuple row="+" group="opinion">
                <atom>append results 2</atom>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

The table has been turned on its side for ease of viewing.

irn	151
TasDescription_tab(+ group='opinion')	append description 1
TasDescription_tab(+ group='opinion')	append description 2
TasResults_tab(+ group='opinion')	append results 1
TasResults_tab(+ group='opinion')	append results 2

This results in:

Before

1	description 1	results 1
2	description 2	

After

1	description 1	results 1
2	description 2	
3	append description 2	append results 2

You may have noticed the value `append description 2` has overwritten the value `append description 1`. The reason is that when group `opinion` was encountered there were two rows in *TasDescription_tab*. Hence group `opinion` refers to row three. As the second `<tuple>` uses the same group name as the first, they both refer to the row used by group `opinion` (row three in this case). Hence the second value overwrites the first value. If you want to append multiple values where the columns are related, a separate group name should be given for each row to be appended. That is:

XML

```
<table table='epos'>
    <tuple>
        <atom name='irn'>151</atom>
        <table name='TasDescription_tab'>
            <tuple row="+" group="opinion 1">
                <atom>append description 1</atom>
            </tuple>
            <tuple row="+" group="opinion 2">
                <atom>append description 2</atom>
            </tuple>
        </table>
        <table name='TasResults_tab'>
            <tuple row="+" group="opinion 1">
                <atom>append results 1</atom>
            </tuple>
            <tuple row="+" group="opinion 2">
                <atom>append results 2</atom>
            </tuple>
        </table>
    </tuple>
</table>
```

CSV

The table has been turned on its side for ease of viewing.

irn	151
TasDescription_tab(+ group='opinion 1')	append description 1
TasDescription_tab(+ group='opinion 2')	append description 2
TasResults_tab(+ group='opinion 1')	append results 1
TasResults_tab(+ group='opinion 2')	append results 2

Which results in:

Before

1	description 1	results 1
2	description 2	

After

1	description 1	results 1
2	description 2	
3	append description 1	append results 1
4	append description 2	append results 2

Index