Release Notes
for
Unique RUNTIME
Unique XTRA
Unique CONCEPT

Version 3.12

[Last updated: March, 27th 1995]

Table of contents

1. Contents of this release.

This release comprises version 3.12 of *Unique RUNTIME, Unique XTRA* and *Unique CONCEPT*.
These products consists of the following modules:

Unique CONCEPT RUNTIME:
- Runtime Engine
- Editor
- One Database Interface

Unique XTRA:
- All modules of *Unique CONCEPT RUNTIME*
- Interactive Xtra

Unique CONCEPT:
- All modules of *Unique XTRA*
- Start (Interactive and DBS compiler)
- Documentation
- Quick (not included in this version)

Optional modules:
SECURITY
Text Format Interface
One of the following: Word Perfect versions 5.0, 5.1 and 6.0, Notis S2, Uniplex 7.0, Rich Text Format (as used by Microsoft Word v2.0 and v6.0)
Additional Database Interface
Techra, Oracle, Sybase, Informix SQL, Informix C-Isam, Ingres, Sibas/R.

Compared with the previous main release, version 3.11, this version is extended with the inclusion of *DOC* and *START Interactive.* Improved functionality in *Unique XTRA, SECURITY* and *START DBS compiler* provides significant enhancements.

2. Installation.

Refer to the document *Installation Notes for Unique CONCEPT* and for last minute updates to the file README (README.WRI on MS Windows) included with this release.

3. New / Changed functionality

3.1. *Unique CONCEPT RUNTIME*

- See general improvements/changes

3.2. *Unique XTRA*

- Improved user interface
- Matrix reporting
- Functionality for Label printing

3.3. *Unique CONCEPT*

- The main menu now includes new entries for *START* Interactive and DOC
- Includes an upgraded version of *START Interactive*
- Includes a new version of *DOC*

3.4. *SECURITY*

- Default login procedure is **external**.
- When using **external** login, user *GUEST* will be used if private user is not defined in *SECURITY*.
- "Company list" is available per system.
- Private and Public XTRA report path may be defined per system.
- Groups are connected to system
- New entity **illegal attempts** is introduced:

  - Used to prevent users from starting applications and logging on to systems or companies where they have no access.
  - Optional **illegal attempts** logging on system level.
  - Automatic application termination and user profile set passive when number of **illegal attempts** is exceeded.

- Password duration per system.
- Supervisor per system. Access to existing system profile and user access definition.
- Super supervisor (Security supervisor). System and user profile creation and maintenance.
- Querymode available on levels such as users, systems groups and applications.

3.5. Runtime engine / General

- uqconfig has a set of new parameters. They are, with a few exceptions, defined in the manual *Unique CONCEPT Operations Guide*. Some new declarations concerning editors and file-formats are not included in the manual and are described here.

editor: **<editor name>**
file-format=<format>
file-extension=<extension>
command-line=<command>

system: **UNIQUE**
default-file-format = <format>
available-editor = <editor name1> [alias <editor name2>]

<format> = BINARY|ASCII|ND-SFORMAT|WP|MS-WORD|RTF|UNIPLEX

The editor section is new and describes editors that may be used by the Unique 4GL function ***editor()***. The old editor definition under the system section will be ignored. To make an editor available to a system, the **available-editor** declaration must be defined in the system section. If more than one editor will be used, repeat the **available-editor** declaration.

The **file-format** parameter is used primarily to define the default file-format used by the internal *Unique* editor. A secondary effect is to set the default file-extension.

The **file-extension** parameter may be used when it is necessary to create temporary files with a given extension (some editors require a pre defined extension name to work correctly). The default extension is related to the file-format as follows:

```
wp              = 'wp5'          * word perfect version 5
nd-sformat      = 'txt'           'text' on SINTRAN
ms-word='doc'
uniplex         ='uap'
rtf             ='rtf'           * Rich Text Format (e.g. Microsoft Word)
others          = no extension.
```

The **command-line** parameter is used to specify the system command needed to start the editor and open a given file. Any occurrence of %f in the command line will be replaced with the file name to be edited. To define the character % as part of the command line, %% is needed. For the internal editor the text @UNIQUE' should be used. The internal editor in read only mode is invoked by @INSPECT'.

default-file-format defines the format for all files *Unique* will write to when no explicit file-format is defined.

Examples:

```
editor: WORDPERFECT
    file-format=WP
    command-line=/usr/wp/bin/wp5 %f

editor: UNIQUE
    file-format=ND-SFORMAT
    command-line=@unique
```

- A new configuration file **uqdio.ini** is introduced to document the new document interface option (see separate documentation for each document interface). The installation includes default file header files for each document type. These files have extension **.hdr** and are located under the setup directory. The uqdio.ini file may be replaced by a non-specific file by adding the following line in the system section in uqconfig:

dio-initialization-file=/usr/unique/setup/mydio.ini

- UTD files have been upgraded.

For further information, ref. chapter 20, "Terminal and Printer definitions supplement" and *Unique CONCEPT Operations Guide.*

- It is now possible to run Unique CONCEPT against Techra 2.32 databases without the a Unique Dictionary. Since Techra does not store a name for the primary key (if the primary key is a composite, or group, column), the primary key should be named PRIMARY whenever used in a Unique application.

- The device specification of uqprint.ini may be built dynamically at printing time by utilising DIO macros. E.g.: A logical printer may be dynamically linked to a specific printer using a device specification like this:

'lp -d ^UQ,@ENVIRONMENT(LOCAL_PRINTER),-1; %f'

where LOCAL_PRINTER is the name of an environment variable assumed to hold the name of a printer.

The UQ macro also allows references to fields of the current printing application, and global fields not defined in the application. The new syntax of the device specification is:

<device> := <string> | '<composit>' | "<composit>"

<string> := sequence of any character, except '^'. To use the '^' character, type the character twice.

<composit> := <string><macro>[<composit>]

<macro> := ^DA[,<date format>]; | ^TI[,<time format>]; | ^UQ,<name>[,<length>];

<name> := <name of field in current application> |
@GLOBAL(<name of global alphanumeric field>) |
@ENVIRONMENT(<environment variable>)

<length> := -1 | 0 | <positive number>
-1 means "remove trailing spaces".
0 means "insert the number of characters a defined by the length of <name>".
0 is equivalent to not giving a <length> parameter at all.
<positive number> means "insert given number of characters".

Limitations:

- <name> may contain a maximum of 30 characters.
- The resulting device specification (after macro expansion) may contain a maximum of 256 characters.

Rules:

- If the resulting device specification contains the string '%f', the device specification is assumed to be a shell command. In such a case, the output is written to a temporary file, and '%f' is replaced by the temporary file name before the command is executed.

  Else, if the resulting device specification does not contain the string '%f', the device specification is assumed to be the name of a device or file where the output is written.

- If a name referenced by the UQ macro does not exist, the macro is ignored.
- A device specification, or any other specification in the uqprint.ini file, containing the comma character (,), must be enclosed in single or double apostrophes. If the same type of apostrophe is to appear inside such a specification, it must be typed twice.
- The leadin character of a DIO macro is the '^' character. If a device specification is to contain this character, type the character twice.
- If the device specification is to contain the '%f' string, type '%%f'.

3.6. *START Interactive*

This version of *Unique START Interactive* refines the methodology of the previous versions. Major enhancements includes:

- Module for Start dictionary database upgrade from version 3.0x format to version 3.12 format.
- Support for all available Unique 4GL datatypes.
- Implementation of proper inheritance of Start Dictionary elements thus:

FROM START TABLE	TO TABLE	ELEMENTS	CONDITIONS
Database projects	Areas	Storage[1]	
Database projects	Tables	Storage, Logging, Compression, Public access[2]	Regular (data) tables only
Areas	Tables	Storage	On users approval when specifying area
Tables	Keys	Storage	From system table if specified, else data table
Data types	Columns	Constraint	

1) Storage includes: initial extent size, next extent size, minimum number of extents, maximum number of extents, percentage increase, fillfactor, freefactor

2) Public access includes: select, insert, update, delete

EXAMPLE:

Specifying **Public access** *Select* on Area definition level, will result in the possibility to use this value as default Public access when defining tables assigned to area.

- The number of referenced editors has been reduced to one, namely the DEFAULT editor.
- The possibility to import a complete database from databases without Unique dictionary has been implemented.
- New module for running, compiling and generating applications for selected project introduced.
- Full use of the system's path references. Source is generated to the source-path, objects are generated to the object-path and run from the application-path. In addition, the user may now select among multiple source paths, according to definitions in the uqconfig file.
- The DBS file is created in the source-path.
- Zoom on project columns when defining columns has been implemented. This feature provides the opportunity to copy information from a column definition of another table. Only column name, heading and data type are copied, all other attributes are not. More advanced users must use standard elements.
- Zoom on table name and area name where appropriate.
- During database import, only regular tables are imported. There is no telling whether a table with no columns is a wide table or system table. In earlier versions all non-regular tables had been imported as system tables.

3.7. START DBS compiler

Easier redefinition of elements in the target database is introduced. The *Unique START DBS Compiler* now provides facilities for adding or removing tables and indexes. Table specific functionality includes adding or dropping single, all or selected tables. This is the first step towards full support for redefining databases with *Unique START*, and altering a user table in an existing database is no longer an unnecessarily time consuming task.

Ref. *Unique START* documentation for a complete list of supported activities per database system.

3.8. *DOC*

DOC is a module of *Unique CONCEPT* with the purpose of documenting systems made using *Unique CONCEPT*. A system is documented according to description and template files. In general description files define what should be documented, and template files define the layout and contents of the documentation.

The documentation process utilises **DOC macros**, comprising a wide variety of **commands** for text expansion and layout control.

DOC is available through the *Unique CONCEPT* main menu or the command line option **-id**.

The headings (long names) of tables and fields may now be documented through the fields TabHeading in template dtabdtl.tpl and FldHeading in the dfdtl*.tpl templates.

3.9. The Unique 4GL language

3.9.1. Matrix Support

For Xtra, a new feature, **matrix** is introduced. A matrix is defined as a table with columns and rows where the number of columns and rows are defined dynamically at run time. The intersection between a row and a column is named a cell, and it is possible to define aggregate operations on the cell. Values are inserted into the matrix using the insert(field=m) function and are grouped according to the group-by definition. Labels (or headings) may be defined and totals may be added both on columns and rows.

The matrix functionality is also made available through Interactive XTRA, where matrix reports may be produced at whim.

For further information on the matrix feature, ref. *Unique 4GL Reference Manual.*

3.9.2. CSV file format

A new stream formatting option CSV (Comma Separated Values) is implemented. The CSV format is similar to the existing ASCII- FIELD format, but the separator, the character that encloses the fields and the decimal point may all be defined in the uqconfig file.

To use the CSV format use FORMATTING=CSV on your output stream.

The default setting for the CSV format is :
- separator : , (comma)
- alpha encloser : " (double quote)
- numeric encloser : (none)
- escape character : (encloser)
- decimal point : . (full stop)

Note: The language code (EN, NO, etc.), and the use of decimal-point= in the uqconfig file, will change this. e.g. when using language code NO the decimal point is changed to comma and the separator will be set to semicolon.

"Escape character" is used to indicate that the next character is a part of the text and should not be treated as a special character (E.g. encloser or separator). If an encloser is found inside an enclosed field and the escape character is set to NONE, the encloser is used as the escape character.

To change the default settings for the CSV format include the CSV-SEPARATORS command in your uqconfig :

```
CSV-SEPARATORS='<separator>','<alpha encloser>','numeric encloser',
        '< escape character >','<decimal point>
```

(for the default settings for Norwegian language)

```
CSV-SEPARATORS=';','"",NONE,NONE,','
```

It is possible to instruct Unique CONCEPT to use the default for a value by leaving out the element. E.g. to specify that the CSV format should use comma as separator, not enclose alpha fields, use default enclosure for numeric fields, no escape character and semicolon as decimal point write:

CSV-SEPARATORS=',',NONE,,NONE,';'

The following rules applies to CSV-SEPARATORS :

- The separator can not be set to NONE
- If the escape character is set, it can not be set to the same value as the separator, alpha encloser or the numeric encloser.
- If the numeric encloser is set to NONE, the separator can not have the same value as the decimal point.

Any violation of these rules causes abortion of any reports that opens streams with the CSV format.

3.9.3. Document interface

A new, advanced document interface has been written to give enhanced support of foreign file formats. Interface for ASCII plain text files is included in the basis. Optional interfaces are available for:

- Word Perfect version 5.0 and 5.1
- Notis S-format version 2 (NOTIS-WP version N and newer).
- Microsoft Word (WINDOWS) version 2.0 and newer.
- UNIPLEX version 7.01 and 7.02

These interfaces are documented separately.

3.9.4. P.INK SQL

Full support for P.INK relational database system is introduced.

3.10. What's NOT included in version 3.12

- *QUICK* module in *Unique CONCEPT*
- To define Update functionality in *Interactive XTRA*
- Support for data type LONG and LONG RAW in ORACLE
- Unload/load wide columns in applications generated by *START Interactive*.

4. Hardware and operating systems used during testing.

HP9000/730CX 64MB RAM	HP/UX 9.0
IBM RS/6000/320	AIX 3.2.3
Dolphin/Uniline88	Unix V 3.2 and 4.0
DG Aviion	DGUX 4.1
Sun SPARC	SunOS 4.1 and Solaris 2.x
Intel 486	SCO UNIX V 3.2.4
DECStation 5000/20	ULTRIX 4.2A
Siemens Nixdorf TARGON/31	UNIX
ICL DRS 6000	NX6 (Unix V 4.0)
NCR 3000	Unix V 4.0
INTEL 386 and 486	Windows 3.1 and Windows For Workgroups 3.1
INTEL 286, 386 and 486	DOS 5.0 and 6.0
INTEL 286, 386 and 486	OS/2 2.0 and 2.1
ND-5800 and 5700	SINTRAN L and M
DEC Microvax 3100	VMS 5.5

5. Database software used during testing.

Kvatro Techra	Version 2.31-01, 2.31-05 and 2.32-02
Sybase SQL Server	Version 4.2, 4.9 Servers, 4.6 Client libraries
Microsoft SQL Server	Version 1.1
Oracle with TPO	Version 6.0.36.1.1
Oracle	Version 7.
Informix C-ISAM	Version 4.00
Informix ONLINE	Version 4.10 and 5.0
Ingres	Version 6.3, 6.4/02 and 6.4/03 (Version 6.4/01 should not be used with Unique)
Sibas/R	Version B04+ and B05

6. Text processors used during testing

Word Perfect	Version 5.0 and 5.1
Notis WP	Version N, O, P
Word for Windows	Version 2.0
Uniplex	Version 7.01 D

7. UNIX supplement

7.1. Minimum system requirements.

Memory:

A minimum of 16MB RAM is recommended to run *Unique CONCEPT* on UNIX, but this figure depends largely on the number of users and other software running on the machine. Roughly the memory required by *Unique CONCEPT* at run time may be calculated as follows:

2MB + 0.3MB * <number of concurrent users>

Remember to add memory needed for the database system in use. (If this need is not known, use 4MB + 0.5 * <number of concurrent sessions>)

Disk space:

To copy from the distribution tape onto a fixed disk a minimum of 20MB free disk space on the destination drive is required.

After installation a minimum of 10MB permanent disk space is required.

8. DOS supplement

Large programs, including *Unique CONCEPT*, cannot execute under MS-DOS as traditional real mode programs. There are however two methods which can be used to overcome the inherent program size restrictions of DOS. The first involves executing the program in real mode (8086/8088) and using overlay or segmentation techniques; the second, used by *Unique*, is to execute the program in protected mode (80286 and above) by using so-called DOS extenders.

8.1. Informix C-ISAM and Unique Isam

Unique CONCEPT for these "databases" is generated using the 286/DOS-Extender from Phar Lap Software, Inc.

8.1.1. HARDWARE AND SOFTWARE REQUIREMENTS

Conventional memory : 50 KB

Extended memory : 1.5 MB
or
XMS (version 2.0) : 1.5 MB
or
EMS (version 4.0) : 1.5 MB

The OS/2 version 1.x DOS box is not supported, but the OS/2 version 2.x DOS boxes are.

8.1.2. MEMORY DRIVERS SUPPORTED

- HIMEM.SYS 2.0 or later

Hint: Set a large number of handles by including the following line in config.sys:
DEVICE=HIMEM.SYS /NUMHANDLES=127

8.1.3. EXPANDED MEMORY

- EMM386 (Microsoft Windows 3.x, MS-DOS 5.0 and 6.0)
  Warning: The EMM386 NOEMS switch turns off VCPI support,
  and can therefore not be used.
- Other expanded-memory managers supporting EMS 4.0 and the VCPI standard.

8.1.4. TO RUN UNDER WINDOWS 3.x

Standard mode:
- create a .PIF file for your EXE file
- set the XMS memory KB Limit to a minimum of 1.5 MB

Enhanced mode and no numeric coprocessor:
- insert the following in Windows SYSTEM.INI file under the 386enh section:
  DEVICE=/uqdos/bin/pharlap.386

It is not possible to have nested DOS-extended programs under Windows enhanced mode. That is, it is not possible to start another DOS-extended program from *Unique CONCEPT* if *Unique CONCEPT* is started from Windows or from a Windows DOS box.

8.1.5. TO RUN UNDER NOVELL WITH WINDOWS 3.x

To achieve a multi-user environment on the NOVELL network all the database files must have "shared access". For more information on how to change access see NOVELL documentation.

8.1.6. UTILITIES FOR TESTING THE PC's CONFIGURATION

The distribution diskette contains three utilities and documentation which may be used to detect configuration problems:

tellme.exe	: Produces a report describing your PC system.
swtest.exe	: Diagnostic program for testing your PC system.
biosdump.exe	: Dumps an image of your ROM BIOS into a file.
tellme.txt	: Documentation for tellme.exe.
swtest.txt	: Documentation for swtest.exe and biosdump.exe.

8.2. Sybase/Microsoft SQL-Server

Unique CONCEPT is generated using the DOS extender DOS/16M from Rational Systems, Inc.

8.2.1. HARDWARE AND SOFTWARE REQUIREMENTS

Conventional memory: 50-150 KB

Extended memory : 1.5 MB
or
XMS (version 2.0) : 1.5 MB
or
EMS (version 3.2) : 1.5 MB

8.2.2. MEMORY DRIVERS SUPPORTED

- HIMEM.SYS 2.0 or newer or

8.2.3. EXPANDED MEMORY

- EMM386 (Microsoft Windows 3.x, MS-DOS 5.0 and 6.0)
  Warning: The EMM386 NOEMS switch turns off VCPI support,
  and can therefore not be used.
- Other expanded-memory managers supporting EMS 4.0 and the VCPI standard.

8.2.4. TO RUN UNDER WINDOWS 3.x

Standard mode:
- create a .PIF file for your EXE file
- set the XMS memory KB Limit to a minimum of 1.5 MB

Enhanced mode:
- nothing special

It is not possible to have nested DOS-extended programs under Windows enhanced mode, i.e. it is not possible to start another DOS-extended program from *Unique CONCEPT* if *Unique CONCEPT* is started from Windows or from a Windows DOS box.

8.2.5. UTILITIES FOR TESTING THE PC's CONFIGURATION

The distribution diskette contains two utilities which may be used to detect configuration problems:

exambios.exe : Writes various information to the file bios.dat.
pminfo.exe : Writes various information to the screen.

8.3. Oracle

Unique CONCEPT is generated using Oracle Pro*C for MS-DOS.

8.3.1. HARDWARE AND SOFTWARE REQUIREMENTS

Conventional memory: 150 KB

This figure is for SQL*Net TCP/IP.

Note: Approximately 140 KB is used by Oracle TSRs, and only 10 KB by the application program
Unique CONCEPT).

extended memory: 2.5 MB

Note: *Unique CONCEPT* uses approximately 1.5 MB, the rest is used by Oracle TSRs.

To run *Unique CONCEPT* MS-DOS Oracle Client, the utility program remap.exe may need to be run on some PCs before any *Unique* program. remap.exe is used to duplicate the interrupt handler of interrupt 22 (16 HEX) to interrupt 96 (60 HEX).

Include the following statements in the AUTOEXEC.BAT file:

```
/unique/bin/remap 22 96
set UQORAINT=96
```

You may ignore the messages issued by remap.exe.

9. WINDOWS supplement

This chapter contains additional information for Microsoft Windows.

Note the following:
- Only Unique CONCEPT RUNTIME with a 4GL compiler is available in this version.
- Microsoft Windows version 3.1 is required

9.1. The contents of this release.

The following languages are available:

English

The following database interfaces are supported

Unique ISAM (included)
Techra (Unique SQL)
SQL-Server (Sybase/Microsoft)
Oracle
Informix C-ISAM
Sibas R (Formula OpenSoft)

Unique CONCEPT for Microsoft Windows supports all these databases in one executable. Dummy DLLs for all databases are supplied with the product. Replace one or more dummy DLL(s) with the real DLL(s) of the database(s) of your choice. The database "client" is a separate product, and must be purchased separately.

9.2. Minimum system requirements.

Hardware and operating systems:
286 4MB RAM VGA Windows 3.1

For Techra:
Techra Client Library 2.32 for MS-Windows

For Sybase:
Sybase Open Client DB-Library Version 4.20.00 for Windows

For Oracle:
Oracle SQL*Net
SQL13WIN.DLL
ORA6WIN.DLL

For Informix C-ISAM:
Informix C-ISAM Version 4.1 is preloaded with *Unique CONCEPT*
Note! A licence to run INFORMIX C-ISAM must be purchased separately.

9.3. Windows specific functionality.

- The file concept.ini may be created in the Windows (e.g. c:\windows) directory. The Unique home directory may be set in this file, thus making it possible to change Unique home directory without leaving Windows.

```
[concept]
unique_home=c:\uqwin
```

- The compiler constant $OS returns WINDOWS when running under Microsoft Windows.

- In screen and report forms it is possible to use the FONT directive to enhance the layout of your applications. The following font numbers are available:

Font	Description
0	Windows system font [1]
1	Draft font
2	Default Unique font [3]
3	ANSI variable pitch font
4	Small font 2
5	Font 2 with bold [3]
6	Font 2 in italic [3]
7,8,9	ANSI fixed pitch font
10-18	Tms Rmn [1] [2]
20-28	Roman [1] [2]
30-38	Helv [1] [2]
40-48	Courier [2]

Notes:
[1] Variable pitch fonts. Use with care where alignment is critical.
[2] These fonts give a larger font the larger the font number. The height is defined as follows (where x is one of 1, 2, 3 or 4):

x0 = 1/10 inch
x1 = 1/8"
x2 = 1/6"
x3 = 1/5"
x4 = 1/4"
x5 = 1/3"
x6 = 1/2"
x7 = 3/4"
x8 = 1/1"

[3] A fixed pitch font where the size depends on the screen resolution used.

・***Shell()*** function:
The parameters ***clear-screen*** and ***hold*** are ignored in this version.

The <alpha expression> must be a fully qualified filename (with extension) which refers to an executable program, .PIF file or .BAT file. If the program is in one of the directories in your PATH then the path prefix is not necessary. Default file extension is .EXE.

Therefore internal DOS commands (e.g. DIR) cannot be executed directly by the shell() function.
HINT: Create a DIR.BAT file which executes the DIR command. Alternatively send the command 'COMMAND.COM /C DIR'.

External DOS commands (e.g. XCOPY) work OK.

NOTE: the ***shell***() function returns immediately after starting the program specified; it does not wait until the program terminates.

- ***editor***() function and editor parameter in the uqconfig file:
The name of the editor in the uqconfig file must be a fully qualified filename as described for the shell() function above.

• ***dialogue-box***() function: The size parameter is ignored so that the frame is made as large as is necessary. Note that the window size will be larger under MS Windows than under DOS because MS Windows push-buttons take up more space.

• Internal *Unique* Function keys (or Programmable command words). *Unique CONCEPT* for MS Windows does not use a UTD file (ref. section "Unique Terminal Description file" in the manual *Unique CONCEPT Operations Guide*). The link between function keys on the keyboard and internal function keys in *Unique*, are hard-coded and can therefore not be changed. This also applies to the default colours used for the different screen elements. The following list shows the available *Unique* function keys and the corresponding Windows function key combination. See the manual *Unique CONCEPT Operations Guide* for a complete list of "Internal command words".

Internal command	Keyboard mapping
Application	Ctrl+O
Cancel	F12
ClearAppl	Ctrl+K
ClearRegion	Ctrl+E
CommandMode	Not available
Default	F8
DelRec	Ctrl+D or Ctrl+Y
DownField	Down Arrow
Enter	Return (Enter)
Exit	Not available (use the menu File/Close All)
Find	F7
FirstField	Ctrl+T
FirstRec	Ctrl+Home
Goto	Ctrl+G
HelpAppl	Shift+F1
HelpField	F1
HighBorder	Ctrl+H
LastRec	Ctrl+End
LeftField	Ctrl+L
NextRec	Page Down
Perform	Ctrl+Return
PgLeft	Ctrl+F
PgRight	Ctrl+B
PrevRec	Page Up
PrintRecords	Ctrl+R
PrintScreen	Ctrl+P
QueryMode	Shift+F7
Return	Ctrl+F4
RightField	Tab
Upfield	Up Arrow
Zoom	Ctrl+S
DelChar	Delete
DelLine	Not available
DelPrevChar	BackSpace
DelUpTo	Not available
EndChar	End
Expand	Not available
LeftChar	Left Arrow
LowerCase	Not available
Refresh	Not available
Restore	Alt+BackSpace
RightChar	Right Arrow
StartChar	Home

UpperCase	Not available

9.4. Printing

Unique CONCEPT uses the MS Windows print system.

Whenever prompted for a printer/file name you may supply the reserved filename PRINT to identify the MS Windows default printer. You may change the default printer, for the current session only, with the command File/Print Setup.

For further details on printing, please refer to the manual *Unique CONCEPT Operations Guide*.

9.5. Definitions and hints

Using external editors**:**
The editor() function in *Unique CONCEPT* has been tested as described below with the following MS Windows programs:

Program name	Test A Start external program and get program to read in the given filename.	Test B Start external program and get to read in the contents of a wide column.
Lotus 1-2-3 v1.0a	Can give input in Unique before Lotus returns.	OK, but cannot get Lotus to read empty wide columns.
Microsoft Excel v4.0	ok	ok
Microsoft Word v2.0	ok	ok
WordPerfect v5.1	ok	ok

None of the above Windows programs can be started successfully using the initial-document command in the editor function to send a document that is not in ASCII format, i.e. you cannot send an Excel spreadsheet as an initial-document to be saved in a wide column.

Some definitions:

program window: the area enclosed by the two grey message lines at the top, the status line at the bottom and the sizeable window frame (or the screen if the *Unique* program is maximised) at the sides. This is the window you see first on starting *Unique CONCEPT*. All application windows are created in this area.
application window: the area which corresponds to your application's **stream** declaration. It may or may not have a frame and a title bar containing the name of the application.

When the user navigates to a field that is not currently inside the current application window, *Unique CONCEPT* will automatically scroll the contents of the application window so that the new current field is visible. But there is no corresponding automatic scrolling of the program window. This may cause problems as shown in the following examples:

1) An application window is larger than the program window.
2) An application window is partly or wholly outside the program window.

Programmers solution to 1) Use the **size(lines, columns)** command (and not **size(?, ?)**) on your application's **stream** where **lines** and **columns** are within the limits of the program window size. This will cause the application window to fit inside the program window and will automatically scroll the application window as the user moves from field to field.

Programmers solution to 2) Use the **position(line, column)** command (and not **position(>>,>>)**) on your application's **stream** where **line** and **column** cause the application window to be completely within the limits of the program window.

Notes: Be aware that the maximum size of the program window depends upon the screen resolution used. Programmer solutions assume that the user does none of the following:
- resize the program window so that it is smaller than the application window
- resize the application window so that it is larger than the program window
- move the application window so that part of it is outside the program window

Users solutions to both problems:
- resize the program window so that it is larger than the application window
- resize the application window so that it is smaller than the program window
- move the application window so that none of it is outside the program window
- scroll the program window with the scroll bars
- maximise the application window

10. VAX/VMS supplement

10.1. Available Database Modules and Versions.

Unique is generated on **VMS v5.5**. If you are running other versions you may need to relink the programs. Databases available on this platform are **Unique Isam, Sybase** and **Ingres**.

10.2. Environment Variables.

Unique uses **symbols** and **logical names** as environment variables.

10.3. The *Unique* Terminal Type.

Unique CONCEPT uses its own terminal types. This is implemented on **VMS** by setting the environment variable **uqterm** to your terminal type. This type must correspond to a .utd file on **[unique.setup]**.

Example :

uqterm has the value : **vt220** and the corresponding file is : **[unique.setup]vt220.utd**

10.4. File Paths

Unique accepts both the file syntax with directories included in braces ([ ]), and the **Unix** file syntax with directories denoted with slashes (/). Any presentations of files will be in the **VMS** file syntax format.

10.5. File Versions

On **VMS** it is possible to have many versions of a file, denoted as : **file.ext;<vno>** where **vno=**the version. In all file presentations from *Unique* only the last version is shown. This differs from the earlier versions of *Unique* where all versions of a file were shown.

10.6. Security Considerations

For security reasons, the global **uqconfig** file must reside on **SYS$SYSDEVICE:[unique.setup].** See the **Installation Guide** and/or the Security chapter in the manual *Unique CONCEPT Operations Guide* for further information.

11. SINTRAN supplement

The $UNIQUE_HOME mechanism is not supported on Sintran

For further information on Sintran, refer the *Unique CONCEPT installation guide for Sintran*.

12. Informix C-ISAM supplement

12.1. Introduction

Informix C-ISAM is a simple, robust database system using standard operating system files and B-tree indexes.

- It requires no dedicated DBMS processes, the database software is completely embedded in the application program. This means there are no operating procedures for starting/stopping the database, it is always available.

- The physical implementation is very close to the logical structure of the database with 2 operating system files per table. The filenames correspond to the table names. The first file (extension .dat) contains the data rows in a fixed layout. The second file (extension .idx) contains all indexes (B-tree) defined on the table. On some operating systems C-ISAM will also create a third file (extension .lok) to handle record locking.

- A multi-user environment can be achieved through file sharing and record locking.

- *Unique CONCEPT* uses an "optimistic record lock" strategy on top of C-ISAM, locking records only for a very short time.

- Consistency of data files and indexes may be verified and/or regenerated by a utility called BCHECK supplied with *Unique CONCEPT*.

- See Appendix "The bcheck utility" in the Informix C-ISAM manual for details on how to use this utility.

- On DOS it is important to set the FILES = parameter in CONFIG.SYS to a sufficiently high number. The following formula may be used to calculate the minimum value:

  4 + (N + 2)*2 (where N is the maximum number of tables opened in one applications (including sub applications).)
  The number must be at least 40.

12.2. Advantages

- simple
- robust
- easy operation
- no overhead in storing data, a minimum of disk space is required.
- available on UNIX, OS/2, DOS standalone and PC networks (Novel NetWare, Microsoft LAN Manager, IBM LAN Server, Banyan Vines, etc.)
- low cost
  - inexpensive
  - low training/operating cost

12.3. Disadvantages

- not a SECURE operating environment
  - no access control in the DBMS (however the *Unique SECURITY* may be used).
  - all users have access to the operating system files; they may copy, delete and/or patch the data files.
- involves a large number of open files for large databases (2 files per table) using valuable system resources.
- the database is not performance competitive in an environment with a large number of users and transactions.

12.4. Supported version

- *Unique CONCEPT* on DOS and Windows has been tested with *Informix C-ISAM version 4.10*.

- *Unique CONCEPT* on Unix has been tested with *Informix C-ISAM version 4.00*.

Please consult your tape label for prelink information, and the Installation guide for information on how to link with other versions.

12.5. Backup, restore and recover

Backup and restore is similar to the backup/restore procedures of normal files in the system. The files may be copied using standard copy functions to and from backup media.

Informix C-ISAM supports a call log function. This log is used both for ROLL FORWARD and ROLL BACKWARD.

In *Unique* the call log function is activated by the presence of a file named <dbname>.log. The file may be created with the uqutilci program (see below).

ROLL BACKWARD is used by *Unique CONCEPT* when rolling back a transaction that failed. Transaction control is not available unless the call log is activated (see above).

ROLL FORWARD is used by a separate utility (uqutilci) supplied with *Unique START*. It should be used with a backup copy of the database files and the log file started immediately after you created the backup.

Syntax: uqutilci (-reset) (-recover) <dbname>

Used with the **-reset** option it resets (or creates if not present) the log file. It should be used after you have created the database and immediately after a backup is created.

Used with the **-recover** option it will recover the database and apply all transactions in the log file to the database files. It should be used when one or more of the Informix C-ISAM files has been corrupted or destroyed and after the most recent backup copy has been reinstalled.

If you do not wish to run with the log function, make sure the file <dbname.log> does not exist.

Using the log function will make update transactions slower than running without a log.

When using the log function, a backup copy must be saved and the log reset (uqutilci -reset <dbname>) periodically. If not, the log file will grow very large and eventually fill up the disk.

12.6. How to set up uqconfig for use with Informix C-ISAM

The following parameters may be used in the database section in the uqconfig file to define an INFORMIX C-ISAM database (name MYBASE used as an example):

```
database: MYBASE
    db-type=INFORMIX-C-ISAM
    db-long-name='Demo database C-ISAM'
    db-path=/usr/mybase/
```

db-long-name and db-path are optional.

If db-path is omitted it defaults to the current user. Db-path specifies the directory in which all the C-ISAM files are located. It may then be accessed from any directory, provided that the file accesses are OK.

See the subject 'uqconfig' in the manual *Unique CONCEPT Operation Guide* for more details.

12.7. Environment variables

Informix C-ISAM supports an environment variable called ISAMBUFS. It defaults to 16 and specifies the number of internal buffers to reduce disk I/O.

Appendix "System Administration" in the Informix C-ISAM manual gives more information on the use of this variable.

12.8. How *Unique START* generates an Informix C-ISAM database

12.8.1. AREAS

AREAS are not supported in Informix C-ISAM and are ignored by *Unique START*.

12.8.2. FOREIGN KEYS

FOREIGN KEYS are not supported in Informix C-ISAM and are ignored by *Unique START*.

12.8.3. TABLES

Two files per table are created as described earlier. All files are created in the directory defined by db-path= in uqconfig.

An additional table is created per database. It has the name of the database and contains dictionary information used in other *Unique CONCEPT* products.

12.8.4. WIDE COLUMNS

Wide columns (type TEXT and IMAGE in .DBS schema) are not supported directly in INFORMIX C-ISAM and are simulated by *Unique CONCEPT*. This is achieved by creating an additional table called "widetab" where *Unique CONCEPT* automatically maintains a chain of data blocks making up the body of the wide column.

If the database contains no wide columns, the table "widetab" is not created.

12.9. Data types

The following relations exist between *UNIQUE* data types and INFORMIX C-ISAM data types:

Unique type	INFORMIX C-ISAM type	
alphanumeric(n)	CHARTYPE	(length n)
binary(n)	CHARTYPE	(length n)
bit	INTTYPE	
date	LONGTYPE	
datetime	CHARTYPE	(length 8)
image	simulated with non standard types>	
integer1	INTTYPE	
integer2	INTTYPE	
integer4	LONGTYPE	
money	CHARTYPE	(length 9)
packed decimal(m,n)	CHARTYPE	(length (m+n)/2+1)
packed decimal(m,n) unsigned	CHARTYPE	(length (m+n)/2+1)
real4	FLOATTYPE	
real8	DOUBLETYPE	
text	<simulated with non standard types>	
time	LONGTYPE	
unpacked decimal(m,n)	CHARTYPE	(length m+n)
unpacked decimal(m,n) Separate	CHARTYPE	(length m+n+1)

12.10. Other restrictions

- The Audit Trail Facility in INFORMIX C-ISAM is not supported.
- Maximum number of open tables in *Unique CONCEPT* is 90 on DOS platforms.
- Keys of type real are not allowed.
- Database files can not be located on NFS mounted disks.
- Composite keys may consist of a maximum of 8 columns. This is a limitation in C-ISAM.

12.11. Upgrading from Informix C-ISAM version 3.x to 4.x

A new log file format was introduced in version 4.x of Informix C-Isam, so please follow these instructions before running a new version of *Unique CONCEPT*, if you used call logging with an old version of Informix C-ISAM.

1. Take a backup of your database files.
2. Reset the call log with the *Unique* utility program:
   uqutilci -reset <dbname>

Make sure that you use the uqutilci utility version 1.1, supplied with this release of *Unique CONCEPT*. It is copied into the same directory as *Unique CONCEPT*. A new version of the Informix C-ISAM Utility bcheck is also supplied with this release and is copied to the same directory as uqutilci. See the enclosed notes for more information on uqutilci and bcheck.

13. INFORMIX ONLINE and STANDARD ENGINE supplement

13.1. Introduction

Informix SQL is the term used in this document to mean either Informix ONLINE or Informix Standard Engine, both of which are advanced, relational database management systems. It is beyond the scope of this document to give a description of either.

To take advantage of this document, the reader must be familiar with basic concepts of either Informix ONLINE or Informix Standard Engine, and in particular how to operate the tbmonitor program.

13.2. Supported version

Unique CONCEPT supports versions 4.1 and 5.0 of Informix ONLINE and Informix Standard Engine.

To use *Unique CONCEPT* with Informix ONLINE or Informix Standard Engine, both the Informix SQL RDBMS kernel and Informix SQL Pro*C must be installed. *Unique CONCEPT* is linked with both Informix ONLINE and Informix Standard Engine at installation time.

If a new revision of Informix is installed, *Unique CONCEPT* must be relinked. Please consult your tape label for prelink information, and the Installation guide for information on how to link with other versions.

13.3. How to set up uqconfig

- The following parameters may be used in the database section in the uqconfig file to define an Informix SQL database (name MYBASE used as an example):

  database: MYBASE
  db-type=Informix-SQL
  db-long-name=Demo database Informix SQL
  db-name-length=<max length of object names>
  db-name-prefix=<characters to prefix all object names>

  db-long-name, db-name-length and db-name-prefix are optional.

- If you are using the Standard Engine interface you should use the db-path entry in the uqconfig to specify where the database is located (the Informix <dbname>.dbs file), i.e.:

  db-path=/mydir/

  Not necessary if the Informix environment variable DBPATH is set.

- If you are running Informix SQL in a client/server environment, please read section "Special Notes on Client Server" later in this chapter.

- See the subject 'uqconfig' in the manual *Unique CONCEPT Operations Guide* for more details.

13.4. Environment variables

Environment variables are normally platform dependant, (see Informix installation guide).

- INFORMIXDIR - Home directory of Informix SQL software

- SQLEXEC - Informix-SE only. Points to the SQLEXEC program, normally "$INFORMIXDIR/bin/sqlexec"
- UQINFOVER - Version number of the Informix SQL software, used by *Unique START* to determine the strategy for NULL values and DEFAULTS. The default value for this variable is 4.1 . Example: UQINFOVER=5.0
- DBPATH - Physical location of the database files (Standard Engine only).

13.5. How *Unique START* generates an Informix SQL database

13.5.1. DATABASE

A *Unique* database is implemented as a DATABASE in Informix SQL.

The database section in the uqconfig file must comply with certain rules when used from *Unique START:*

The current UNIX user and password will be used to log into the Informix server.

If you are running Standard Engine, *Unique START* will place the database in the directory specified by db-path or by DBPATH if this variable is set. If none of the above is used, the database will be placed in the current directory.

13.5.2. AREAS

Creation of AREAS in *Unique* DBS language are not implemented in this version of *Unique START*. If AREAS are specified in the .DBS file, they must already exist before running *Unique START*.

If you are using the Informix-SE you will have to specify a LOG-AREA in your DBS-file.
Example: create area MYLOGAREA on '/mydir/mylog.log' default-log-area

13.5.3. NULL VALUES

Informix SQL software version 5.0 or higher supports use of the keyword DEFAULT to specify a default value for uninitialised columns different from NULL, and this is strongly recommended for databases used by *Unique CONCEPT*.

If you are running Informix SQL software version 5.0 or higher set the environment variable UQINFOVER to your version number before running *Unique START* to create your database. This will direct *Unique START* to add the attributes NOT NULL and DEFAULT <value> when creating columns, unless otherwise is stated in your DBS file.

If you are running Informix SQL software version 4.1 and intend to insert records from *Unique* applications not containing all columns of the table, add the attribute DEFAULT NULL to the create schema statement in your DBS file, and the db-nullvalues=ON to the database section in your uqconfig file.

13.5.4. FOREIGN KEYS

FOREIGN KEYS are implemented in Informix SQL but are not supported by this version of *Unique START*. *Unique START* will ignore all foreign key declarations in the .DBS schema.

13.6. Data types

The following relations exist between *Unique CONCEPT* data types and Informix SQL data types:

Unique type	INFORMIX SQL Type
alphanumeric(n)	char(n)
binary(n)	char(n)
bit	smallint
date	date
datetime	datetime, year-to-second
image	byte 1)
integer1	smallint
integer2	smallint
integer4	integer
money	money(16,2)
packed decimal(m,n)	decimal(m+n),n)
packed decimal(m,n) unsigned	decimal(m+n),n)
real4	smallfloat
real8	float
text	text 1)
time	datetime, hour-to-second
unpacked decimal(m,n)	decimal(m+n, n)
unpacked decimal(m,n) Separate	decimal(m+n, n)

Byte and Text types are not implemented in Informix Standard Engine and are not (in this version) simulated by *Unique CONCEPT*.

13.7. Special Notes on Client/Server.

If you operate in a Client/Server environment, use the uqconfig entry db-server to specify the name of the server machine, e.g. :

```
db-server=myhost
```

More information on this subject is to be found in the Informix-Star and Informix-Net manuals.

NOTE: Informix advises you to use the same revision of the Database Engine and the Informix-Star/Informix-Net products. It is also advisable to use the same revisions on both the client and the server.

13.8. Other restrictions

- *Unique CONCEPT* does not differentiate between a column with value 0 (or spaces) and the special NULL value in Informix SQL. All 0 values will be stored with NULL. When fetching rows, both values will be considered equal. This restriction will not apply if your database is created using the column attributes NOT NULL and DEFAULT (see 12.5.3).

- The name of a key must never contain the character sequence _q, since this sequence has special meaning to *Unique CONCEPT*.

- If you are using version 4.1 of Informix-SQL it is necessary to configure your system with db-null=on. This must be done in the uqconfig file. Similarly if you are using version 5.0 of Informix-SQL and have generated the database using *Unique START DBS Compiler* version 3.03-99 or earlier it is also necessary to set db-null=on (see 12.5.3).

- For Informix SE, composite keys may consist of a maximum of 8 columns, and the total length of the key members must not exceed 120 bytes. This is a limitation in Informix SE.

- For Informix Online, composite keys may consist of a maximum of 16 columns, and the total length of the key members must not exceed 255 bytes. This is a limitation in Informix Online.

13.9. Further notes

It is strongly recommended that after multiple changes to one or more tables in the databases you use the command UPDATE STATISTICS in Interactive SQL.

14. SYBASE supplement

14.1. Introduction

The scope of this chapter is to describe the SYBASE interface of *Unique CONCEPT*, and especially any differences to other database management systems supported by *Unique CONCEPT*. The reader must be familiar with basic concepts in SYBASE and in particular how to operate ISQL and the administrative procedures.

Note: In this document (and in all *Unique* products) SYBASE SQL Server and Microsoft SQL Server are treated as the same product.

14.2. Supported version

Unique CONCEPT gives support for SYBASE version 4.0 on UNIX and SQL Server version 1.1 on OS/2. Newer versions may be used as long as they are backward compatible with the supported versions.

To use *Unique CONCEPT* with SYBASE, both the SQL Server kernel and the Open Client DB-Library/C must be installed. On DOS, also the Open Client Net-Library must be installed. On Unix, *Unique CONCEPT* is linked with DB-Library/C at installation time. The current version of Unique CONCEPT's SYBASE interface is compiled with SYBASE db-library version 4.6 on UNIX and version 4.2 on Windows, MS-DOS and OS/2, and must be linked with the same version of the SYBASE library. Linking with a newer or older version of the SYBASE library might cause malfunction. On MS-DOS, *Unique* is provided prelinked with DB-Library/C.

14.3. How to set up uqconfig for use with SYBASE

The following parameters of the database section of the uqconfig file have a special meaning in the definition of a SYBASE database:

db-type = SYBASE

Specifies that the database is a SYBASE database.

db-server = <SYBASE server>

Specifies the SYBASE server handling the database. The syntax of <SYBASE server> depends on operating system and communication protocols used. On Unix and DOS/Named Pipes, it normally is the name of a SYBASE server. On DOS/TCP/IP, it may be the name of an environment variable that further defines the connection to the proper SYBASE server using internet and port numbers. Please consult the appropriate SYBASE documentation .

db-user = <SYBASE user name> | ?

Specifies the user name to be used when connecting (login) to the SYBASE server. A question mark indicates that the *Unique* user will be prompted for a SYBASE user name.

db-password = <SYBASE user password> | ?

Specifies the password to be used when connecting (logging in) to the SYBASE server. A question mark indicates that the *Unique* user will be prompted for a SYBASE password.

db-connection = <min>:<max>

Controls the usage of Sybase User Connections (DBPROCESSes). Generally, the larger <min> and <max> values, the better performance, but since the total available number of User Connections on a SYBASE Server is limited, the value of <max> should never exceed

(N - U) / U

where N is the total number of User Connections on the server available for *Unique* users, and U is the maximum number of *Unique* users concurrently using the server.

The <min> value must be in the range 0 to <max>.

Unique CONCEPT's default use of user connections is :
1 pr. server which is logged on.
1 pr. server if columns of type image/text are written to the database.
1 pr. server for the SA login (see separate chapter).
+ the number specified as maximum for db-connections pr. server

db-nullvalues = <ON> | <OFF>

This parameter must be set to ON if your database allows NULL values in any columns. This will affect performance and it is recommended that you generate your database with NULLS not allowed and with default bindings for each column (for further information see the *Unique START* manuals and/or the
SYBASE documentation of SP_BINDEFAULT/CREATE TABLE).

db-timeout = <query timeout> : <update timeout>

Specifies the amount of time an operation in SYBASE might last before returning control to *Unique*. This is used by *Unique CONCEPT* to warn the user about a long lasting operation, and to resolve any "deadlock with yourself " situations. The default timeout is 20 seconds for query operations and 15 seconds for modify operations (See also the documentation of UQUNLOCK and UQOWNTO). The value 0 means no timeout at all.

db-name-case = <UPPER> | <LOWER>

Default letter case on objectnames in SYBASE is lowercase. Notice that *Unique CONCEPT* only allows lower or upper case on object names.

Please also note the **db-name-prefix** and **db-name-length** parameters.

14.4. Environment variables

Some optional attributes of the *Unique CONCEPT* SYBASE interface are controlled through the usage of environment variables. (The SYBASE installation might require some environment variables to be set, e.g. SYBASE and DSQUERY.)

Note: Using Bourne shell on Unix, remember to export new values of environment variables.

14.4.1. UQROWCNT

UQROWCNT=<number of records>

Defines the maximum number of records to be returned by the Sybase server as a result of a SQL SELECT generated by a *Unique CONCEPT* application using a unique main key. Note: This does not apply to *XTRA* applications at all, nor to ONLINE applications using an ambiguous main key (see UQAROWCNT).

A UQROWCNT value greater than zero on a table without repeat lines means "get UQROWCNT number of records". If the table has repeat lines, it means "(get UQROWCNT plus number of repeat lines) number of records". UQROWCNT=0 means "get all records". The default value is 10.

A UQROWCNT greater than 0 does not imply that the user will never get all records, only that *Unique CONCEPT* may get the records by issuing several SELECT commands. The reason for this mechanism is that Sybase may place shared locks on the table being processed by a SELECT command. These locks may inhibit updates on the same table and possibly generate deadlock situations. Using UQROWCNT with a sufficiently small value, ensures that no shared locks will be placed (for a longer period of time). The disadvantage is that overhead may be increased by executing several SELECT commands instead of one. However, in ONLINE applications this is not normally a problem, since the user rarely scrolls sequentially through large amounts of data. In *XTRA* applications, however, this is the normal operation, which is the reason why UQROWCNT affects ONLINE applications only.

To verify that the chosen value of UQROWCNT ensures that no locks are placed, requires some investigation to be done on site. It is dependant on the database, operating system and network versions used. You may use the following procedure to verify that the chosen value does not place shared locks:

1) If possible, make sure you are alone using the chosen database.

2) Choose a large database table (large with respect to both record length and number of records).

3) Turn the row count mechanism off, i.e. by setting UQROWCNT=0.

4) Make a simple ONLINE application accessing the table using a unique main key, compile it and run it. Note: Use the default ascending sort order.

5) In the ONLINE application, give the FirstRec command.

6) Using e.g. the Sybase utility ISQL, call the stored procedure sp_lock. This will report any locks (shared locks and others) currently placed in the databases. The previously issued FirstRec command from the ONLINE application should have placed shared locks. If not, the table chosen is too small. Choose another and go to step 4.

7) Exit *Unique CONCEPT*, decrease the value of UQROWCNT (initially, start with say 100), restart *Unique CONCEPT* with the application made and issue the FirstRec command.

8) Repeat step 7 until sp_lock no longer reports any locks placed.

9) An applicable value for UQROWCNT is now found.

The rule of thumb for choosing a value for UQROWCNT is that it should be as small as possible, but not so small that it decreases performance for the average ONLINE application user.

14.4.2. UQAROWCNT

UQAROWCNT=<number of records>

Defines the maximum number of records to be returned by the Sybase server as a result of a SQL SELECT generated by an ONLINE application using an ambiguous main key. Note: This does not apply to XTRA applications at all, nor to ONLINE applications using a unique main key (see UQROWCNT).

As UQAROWCNT has the same functionality with ambiguous main keys as UQROWCNT has with unique main keys, the general description of UQROWCNT also applies for UQAROWCNT.

The reason for distinguishing between unique and ambiguous main keys is that the row count mechanism does not always work on ambiguous keys, i.e. the user is not guaranteed to get all records when UQAROWCNT is greater than zero. Consider the following example:

> A table contains 100 records with the same ambiguous key value 'A', plus more records with other key values. If the UQAROWCNT is less than 100, then the user of an ONLINE application accessing the table using the ambiguous key will never get all 100 records with key value 'A', he will at most get the "first" UQAROWCNT records with that key value.

This is not a foolproof solution, and a new implementation is being considered. But knowing about this deficiency should make it possible to take the required precautions, e.g. by using a large enough UQAROWCNT value or limiting the number of ambiguous keys used in ONLINE applications.

14.4.3. UQNOINDX

By default, *Unique CONCEPT* uses the (yet undocumented) "force index" feature of Sybase. This is supported from version 4.01 of SQL Server, and in most cases gives significantly increased performance for *Unique CONCEPT* applications. If, however, the version of Sybase being used does not support this feature, or one simply does not want *Unique CONCEPT* to use it, just define the UQNOINDX variable.

14.4.4. UQTPLAN

For development and debugging purposes: Define this variable to get the Sybase execution plan on the *Unique CONCEPT* trace file. (Issues the Sybase SQL statement "set showplan on".)

14.4.5. UQTIO

For development and debugging purposes: Define this variable to get the Sybase IO statistics on the *Unique CONCEPT* trace file. (Issues the Sybase SQL statement "set statistics IO on".)

14.4.6. UQTTIME

For development and debugging purposes: Define this variable to get the Sybase time statistics on the *Unique CONCEPT* trace file. (Issues the Sybase SQL statement "set statistics time on".)

14.4.7. UQUNLOCK

UQUNLOCK=<number of modify timeouts before any "deadlock with yourself " situations are resolved>

If a modify operation times out the number of times specified by UQUNLOCK, *Unique CONCEPT* will try to release all selections that might lock the operation.

14.4.8. UQOWNTO

Some versions of the SYBASE db-lib do not support the timeout mechanism as *Unique CONCEPT* expects. If UQOWNTO is set, *Unique CONCEPT* will handle timeout operation itself. Set UQOWNTO only if you have indications that SYBASE's own timeout doesn't work (Set db-timeout to 1:1 and run an operation which should last for more than one second (E.g. FIRST on a large table). If SYBASE's timeout works, the warning "long lasting database operation" should be given).

14.4.9. UQSYBTS

UQSYBTS=<maximum number of bytes to be read from text/image columns>

The default textsize set by a SYBASE server is 65536. If your text/image columns contains more data than this, you will have to set UQSYBTS to an appropriate value. Be aware that setting this value might affect the memory usage on your machine.

15.4.10. ENVIRONMENT VARIABLES ON VMS

You can specify the variables on VMS either as symbols or as logical names. It is advisable though to use the system logical name table to ensure a homogeneous environment. Ex.: $ define <SYSVAR> "<value>".

14.5. Allowing applications to use a SA login

Unique CONCEPT needs to access the SYBASE server with SA privileges in two situations. When a user aborts an ongoing operation the operation has to be aborted on the SYBASE server using the SYBASE command KILL, and this command can only be done with SA privileges.
If your applications are generating stored procedures (see separate chapter), these will be made through the SA login so that they can be used by any user of the database.

Since the SA password should only be known by the database manager/system supervisor *Unique CONCEPT* includes a small program called *UQCRYPT* which cryptifies a file in such a manner that it is not readable for other users. So to let *Unique CONCEPT* use a SA login do the following:

1) Make a file with the following layout : (one line for each server)
<server name>,sa,<sa password>
E.g
SYBASE,sa,julian

2) Run the *UQCRYPT* program which should reside on your setup directory, as follows :
uqcrypt -i<file from (1)> -osybase.par
Now the file sybase.par should reside on your setup directory (must have public read access).

3) Now remove the file from (1) so that the password is not available to other users. Remember to repeat this when the SA password is changed.

14.6. Stored Procedures

In order to increase performance this version of *Unique CONCEPT* takes advantage of Stored Procedures at run-time and uses them if they already exists for a given set of applications (for detailed information on Stored Procedures consult your SYBASE documentation).

Normally *Unique CONCEPT* issues a SQL statement when a database operation is done. This statement is then compiled by the SYBASE server, which includes converting data to SYBASE native format, setting up an access plan and checking the statement for logical errors. This task is done for each database operation.

With Stored Procedures this task only needs to be done once. Instead of passing a SQL statement, *Unique CONCEPT* runs a Stored Procedure. The first time an application (defined to run Stored Procedures) runs, it creates Stored Procedures which are compiled and stored on the server. There will be one Stored Procedure produced per database/table operation (i.e. FIRST, LAST, FIND, INSERT and DELETE). The next time the actual SQL statement is called for, the Stored Procedure is run directly with all data passed in native format. Since all Stored Procedures are made using the SA login, the Stored Procedures are available to any user of the database.

To turn this option on, it is necessary to make a file, for each system, on the setup directory which includes the set of applications that should generate Stored Procedures. The name of the file must be generated as :

< systemname>.ssp E.g. for system *UNIQUE* make the file : **unique.ssp**
(If the operating system is case sensitive, the filename must be in lowercase).

The contents of the file should be : (one line for each application)

<application name> [< Logical name>]

The logical name is optional, and defaults to system name (first 8 characters) plus application name (first 14 characters) if not given, but it should be used in case the default combination is ambiguous amongst any systems accessing the database, or in case an application is used by more than one system (then use the same logical name on both system).
The logical name must not include any character which is illegal for object names in SYBASE (consult your SYBASE documentation), maximum length is 22 characters.

The name of the stored procedures generated is :

uq<Version number><Logical name><Table indx number><Operation code>

- The version number is currently : **a2**
- Table indx number is application specific.
- Operation code is :

  F - First
  L - Last
  J - Find
  I - Insert
  D - Delete

If this option is in use *Unique CONCEPT* will generate Stored Procedures for the following database operations : FIRST (includes SCROLL), LAST, FIND (includes JOIN), INSERT and DELETE.

Note : Stored Procedures have a static nature, and will not change when the application changes. Thus, it is necessary to drop all Stored Procedures made by an application if the application changes, this can be done by the database manager/system supervisor by the use of any interactive SYBASE SQL tool. It might be FATAL to run a changed application using Stored Procedures generated by the previous version of the application. As a rule of thumb, don't generate Stored Procedures for applications which are under development.

Note : The privileges granted to the Stored Procedure overrule any privileges granted to the user. Therefore, Stored Procedures should not be used in systems where the SYBASE users logged on have different access rights defined in the database.

Note : When a Stored Procedure is compiled SYBASE determines an access plan for that procedure. If the conditions that determined that access plan change, the Stored Procedure should be recompiled or regenerated for optimal performance. It is the database manager/system supervisor's task to determine when such actions are required.

14.7. Trigger handling

Unique CONCEPT handles messages from triggers programmed with RAISERROR. The action taken from *Unique CONCEPT* is defined by the error number returned by the RAISERROR (default) or the return value from a stored procedure named "uq_raiserror" which takes the error number from RAISERROR as a parameter.

The default is :

ErrorNumber	Action
20001-29999	The message is presented as a notification message to the user. The statement executed was not aborted by the trigger (No rollback transaction was done). (equals uq_raiserror returning 0)
30000-39999	The message is treated as a normal SYBASE error, and *Unique CONCEPT* assumes that the current operation was denied (rollback transaction done in the trigger). (equals uq_raiserror returning 1)
>40000	Fatal error, *Unique CONCEPT* will abort. (equals uq_raiserror returning 2)

Example of "uq_raiserror" for the default :

```
create procedure uq_raiserror @errno int as
begin
        if( @errno >= 20000 and @errno <= 29999 )
                return 0                /* 0 = notification message */
        else if( @errno >= 30000 and @errno <= 39999 )
                return 1                /* 1 = Treat as a SYBASE error */
        else
                return 2                /* 2 = Fatal error         */
end
```

If using "uq_raiserror" remember to make it accessible for all users.

Unique CONCEPT will ignore any messages passed from the trigger by using PRINT .

14.8. Special notes

14.8.1. LOGGING INTO A SERVER

Unique CONCEPT has adopted the SYBASE strategy when logging into a server. This means that if several databases are accessed on one server from *Unique CONCEPT*, unique will be logged in with the same user on each database even if the UQCONFIG file states otherwise. The user selected is the user/password selection from the first database accessed.

The UQCONFIG parameter db-connections is treated the same way.

14.8.2. SYBASE server version 4.9.2 and the 1610 flag

The SYBASE server 4.9.2 introduces a new flag called 1610, which will increase performance for queries which selects large records. For further information on the 1610 flag consult your SYBASE documentation or contact your local SYBASE dealer.

14.9. How *Unique START* generates a SYBASE database

14.9.1. DATABASE

A *Unique* database has a one to one relationship with a database in SYBASE. In which server the database exists is defined by the parameter db-server in uqconfig.

When *Unique START* creates a database, the size clause in CREATE SCHEMA is used to define the size of the database. The default size in SYBASE is used if the size clause in Start is omitted.

The ISQL command sp_configure may be used to change the default database size in SYBASE.

The size may be incremented at a later time using the ALTER DATABASE command in ISQL. This command may also be used to let the database span several devices (see next section).

14.9.2. AREAS

Versions 1.2 and later of Microsoft SQL Server and 4.2 and later of Sybase SQL Server support segments. *Unique START* provides a comprehensive system for the definition of areas and will create the devices corresponding to area definitions within the DBS file. These devices will be referenced in the creation of the database and if the SQL Server supports segments, segments will be created on the devices and referred to in the creation of tables and indexes.

See SYBASE System Administration Guide for further details.

14.9.3. WIDE COLUMNS

Wide columns are supported in SYBASE. *Unique CONCEPT* uses the data types image and text in SYBASE to implement this functionality.

It is not possible to place the contents of these columns on specific segments. *Unique START* will ignore references to wide tables and areas used for wide columns in the .DBS schema.

14.10. Data types

The following relation exists between *Unique CONCEPT* data types and SYBASE data types:

Unique type	SYBASE-Type	
alphanumeric(n)	char(n), varchar(n)	1)
binary(n)	binary(n)	
bit	bit	
date	int	
datetime	datetime	
image	image	
image generic	image	
integer1	tinyint	
integer2	smallint	
integer4	int	
money	money	
packed decimal(m,n)	binary(m+n),/2+1)	2)
packed decimal(m,n)unsigned	binary(m+n),/2+1)	

real4	real	3)
real8	float	
serial	int	4)
text	text	
text generic	text	
time	int	
unpacked decimal(m,n)	binary(m+n)	2)
unpacked decimal(m,n) Separate	binary(m+n+1)	2)

1) Varchar(n) can be generated by using the *Unique START* type varchar(n) or string(n). *Unique CONCEPT* treats varchar(n) and char(n) in exactly the same way.

2) Packed and Unpacked decimal types can be implemented as money in SYBASE if within the database definition in the uqconfig file the clause db-options=decimal-as-money is specified.

3) If the SYBASE data type real is not implemented, *Unique START* will create it of type float.

4) Serial is only available as a type in *Unique START*, not in *Unique CONCEPT RUNTIME*, where serial columns may be used as I4.

15. ORACLE supplement

15.1. Introduction to ORACLE RDBMS

ORACLE is an advanced, relational database management system. It is beyond the scope of this document to give a description of ORACLE.

To take advantage of this document, the reader must be familiar with basic concepts in ORACLE and in particular how to operate sqldba (and/or sqlplus).

15.2. Supported version

Unique CONCEPT version 3.12-07 supports Oracle version 7. (Oracle version 6 is supported on some platforms using an older database interface.)

To use *Unique CONCEPT* with ORACLE, both the ORACLE RDBMS kernel and ORACLE Pro*C must be purchased and installed. *Unique CONCEPT* is linked with ORACLE at installation time. If a new revision of ORACLE Pro*C is installed, *Unique CONCEPT* must be relinked. Please consult the Installation guide for more information on how to link with other versions of Oracle.

15.3. How to set up uqconfig for use with ORACLE

The uqconfig database options have the following usage for an Oracle database:

Option	Value	Description
DB-CHARSET	<standard usage>	Note: First available in Unique CONCEPT version 4.0
DB-CONNECTIONS	<not used>	
DB-DIRECTORY	<not used>	
DB-LONGNAME	<standard usage>	
DB-NAME-CASE	<standard usage>	
DB-NAME-LENGTH	<standard usage>	
DB-NAME-PREFIX	<standard usage>	
DB-NULLVALUES	<not used>	
DB-OPTIONS	CONVERT_NAMES IGNORE_UQDIC	Standard usage of the listed options - other options are not used.
DB-OWNER	<not used>	
DB-PASSWORD		Oracle password for user defined by DB-USER
DB-PATH	<not used>	
DB-PHYSICAL-NAME	<standard usage>	
DB-PROCESS	<not used>	
DB-PURPOSE-PREFIX	<standard usage>	
DB-SERVER		Oracle host machine name used at login.
DB-TIMEOUT	<not used>	
DB-TYPE	ORACLE ORACLE6	The ORACLE keyword invokes the default Oracle7 interface. The ORACLE6 keyword invokes the old Oracle version 6 interface.
DB-USER		Oracle user name used at login.

The Unique term "database" is implemented as a Oracle user owning the database tables. The name of this user is the Unique database name, alternativly the name given by DB-PHYSICAL-NAME, prefixed by the optional DB-NAME-PREFIX.

Unique START requires that the DB-USER name must be equal to the concatenation of DB-NAME-PREFIX and the database name, alternativly DB-PHYSICAL-NAME.

15.4. Environment variables

The following environment variables may be used to change default settings:

UQO7ARRAY=<number>

<number> defines the number of records to be fetched in each Oracle "fetch call". The optimum value is site dependant, but as a rule of thumb "10 is much better than 1", and "100 is only slightly better than 10". As a golden rule, it should not be higher than necessary. The default is 10.

UQO7COMMIT=<number>

<number> defines the number of database updates/inserts to do before they automatically are commited. The default is 100.

UQO7HINT=<mode>

<mode> is a number defining how Unique CONCEPT uses the Oracle hint mechanism:

<mode>=0: Do not use hints. This is the default - the same as not setting UQO7HINT.

<mode>=1: On FIRST, NEXT, LAST and PREVIOUS: Use FIRST_ROWS from Online applications, ALL_ROWS from Xtra applications, INDEX_ASC() for ascending searches, and INDEX_DESC() for descending searches.

<mode>=2: Same as <mode>=1, except for not using the INDEX_ASC() and INDEX_DESC hints.

<mode>=3: Same as <mode>=1, except for not using the FIRST_ROWS and ALL_ROWS hints.

<mode>=4: Same as <mode>=1, but acts on FIND/JOIN as well, where the INDEX() hint is used generally, and FIRST_ROWS is used from Online applications.

<mode>=5: Same as <mode>=1, except for not using the INDEX_ASC() hint.

Oracle itselfe requires some environment variables set, notabley ORACLE_HOME and ORACLE_SID (platform dependant). If these are not set correctly, *Unique CONCEPT* will not be able to communicate with the database.

15.5. How *Unique START* generates an ORACLE database

15.5.1. DATABASE

A *Unique* database is implemented as a USER in ORACLE. *Unique CONCEPT* can only access one physical ORACLE database (defined by ORACLE_SID), all *Unique* logical databases are mapped to the same physical database.

All objects owned by a given user (database owner) represent a data-base with the user's name used as the database name by *Unique CONCEPT* products.

When creating a database, *Unique START* will create a user with the same name as the database and grant CONNECT access to the user. RESOURCE access will be granted to the user on specified areas (Oracle tablespaces).

The database section in the uqconfig file must comply with certain rules when used from *Unique START*:

- <db-physical> name and <db-user> must be identical!
NOTE: db-name-prefix is used, <db-user> must include the prefix!

- <db-password> can be specified at runtime or in the uqconfig file. If the user account already exists, the password will be changed to <db-password>. Note that reserved SQL words like SELECT, TABLE, UNIQUE can NOT be used as password!

The password for an ORACLE dba (database administrator) account (user) e.g. SYS, may be requested by *Unique START* at runtime.

15.5.2. AREAS

AREA in *Unique* DBS language is mapped to TABLESPACE in ORACLE. *Unique START* will automatically create one table space for each area defined. All tables will be created on the specified tablespaces when creating databases.

Before running Start, verify/update area definition in <dbname>.dbs. Edit the area names and filenames in <dbname>.dbs. Use full path names in the ON clause. If SIZE is omitted, a value of 10 MegaBytes is used.

Ex: create area MYAREA on '/usr/db/myarea.dat'

Unique START may later issue a warning message saying that the area already exists, but the database will be created.

ALTERNATIVE (MANUAL) CREATION OF TABLESPACES (AREAS).

1. Start SQLPLUS (or SQLDBA) as an ORACLE dba.
e.g. sqlplus SYS/<sys password>

2. Create the tablespaces manually in SQLPLUS.

See the SQL Reference Manual on CREATE TABLESPACE for details.

Before creating a second table space in an ORACLE database, you must CREATE and ACTIVATE a second ROLLBACK segment in table space SYSTEM. If the database already has 2 or more rollback segments this is not necessary.

(Still logged in as a dba in SQLPLUS or SQLDBA).

To check the current number of rollback segments:

SELECT SEGMENT_NAME FROM DBA_ROLLBACK_SEGS;

To create a new rollback segment:

CREATE PUBLIC ROLLBACK SEGMENT RBS2;

Shut down the database (SQLDBA command="shutdown").
Restart the database (SQLDBA command="startup shared").

NOTE: User oracle must have write access to the directory in which the file will be created. Create the directory and assign oracle as the owner.

Ex:

```
cd /usr
mkdir db
chown oracle db
```

start SQLDBA and log in as dba.

```
CREATE TABLESPACE MYAREA DATAFILE '/usr/db/myarea.dat' SIZE 20M DEFAULT
STORAGE ( INITIAL 100K NEXT 200K );
```

Note that the file must NOT exist prior to being created using the above syntax (REUSE must be used if the file already exists).

Also note that the SQL command DROP TABLESPACE will NOT delete the file from the file system!

You may add more rollback segments to the system on the new table space (recommended for increased performance).

```
CREATE PUBLIC ROLLBACK SEGMENT RBS3 TABLESPACE MYAREA;
```

3. Update UQCONFIG with the new database entry:

```
database: <dbname>
    db-type=ORACLE
    db-user=<dbname>
    db-password=<dbname password>
```

4. Run *Unique START* to create the database.

Remember to the set the area usage option to PRIVATE if you want tablespaces to be created exclusively for the tables in your database.The area usage option DEFAULT will use default table space as decided by ORACLE. The area usage option COMMON indicates to START that it should only create the tablespaces if they do not already exist. When using COMMON differences between the attributes of areas in the DBS file and those of an existing tablespace with the same name will be ignored.

Warning messages regarding CREATE TABLESPACE may be ignored if you have created them manually.

15.5.3. FOREIGN KEYS

FOREIGN KEYS are implemented in ORACLE, but not enforced. *Unique START* will ignore all foreign key declarations in the DBS schema.

15.6. Data types

The following relations exist between *Unique CONCEPT* data types and ORACLE data types:

Unique type	ORACLE-Type	
alphanumeric(n)	char(n), varchar(n), varchar2(n)	1)
binary(n)	raw(n)	
bit	number(1)	
date	number(8)	
datetime	date	
image	<simulated>	2)

image generic	long raw	3)
integer1	number(3)	4)
integer2	number(5)	5)
integer4	number(12)	6)
money	number(14,2)	7)
packed decimal(m,n)	number(m+n,n)	
packed decimal(m,n) unsigned	number(m+n,n)	
real4	number	
real8	number	
serial	number(12)	8)
text	<simulated>	2)
text generic	long	3)
time	number(6,0)	
unpacked decimal(m,n)	number(m+n, n)	
unpacked decimal(m,n) Separate	number(m+n, n)	

1) *START* uses the varchar2(n) datatype. *Unique CONCEPT* handles all 3 character types, but you should stick to only one type within one database.

2) Simulated wide columns are used instead of the generic Oracle types of long and long raw because of the limitation that only one long or long raw can exists in each table.

3) Generic types are not supported in this version of *Unique CONCEPT*.

4) Oracle type number(2) will also be treated as *Unique* type integer1

5) Oracle type number(4) will also be treated as *Unique* type integer2

6) Oracle types number(6), number(7), number(8), number(9), number(10) and number(11) will also be treated as *Unique CONCEPT* type integer4

7) Money is implemented as number(14,2). It will be extended to number(16,2) in the future.

8) Serial is only available as a type in *Unique START*, not in *Unique CONCEPT RUNTIME* where serial columns may be used as I4.

15.7. Other restrictions

- SEQUENCE GENERATORS are not supported in the Unique 4GL language (but used internally in Unique when simulating wide columns).

- *Unique CONCEPT* does not differentiate between a column with "zero value" and the special NULL valueE. Thus, NULL values should be avoided, and Unique CONCEPT uses zero (0) in numeric columns, one space (' ') in character columns and other appropriate "zero" values in columns of other datatypes..

16. INGRES supplement

16.1. Introduction to INGRES.

INGRES is an advanced relational database management system. It is beyond the scope of this document to give a description of INGRES.

To take advantage of this document, the reader must be familiar with basic concepts in INGRES and in particular how to operate ISQL/SQL and the administrative procedures.

16.2. Supported versions

Unique CONCEPT gives support for version 6.3 and 6.4 of INGRES on UNIX. Newer versions may be used as long they are backward compatible with the supported versions.

Note: Version 6.4/01 has proven to give poor performance with *Unique CONCEPT*. Use subversion /02 or later!

To use *Unique CONCEPT* with INGRES, the INGRES/SQL SERVER kernel must be licensed and installed. Unique is linked with INGRES at installation time. If a new revision of INGRES is installed, *Unique CONCEPT* should be relinked. Please consult the Installation guide for information on how to link with other versions.

16.3. How *Unique CONCEPT* interacts with INGRES.

16.3.1. SESSIONS AND CURSORS IN INGRES.

Every *Unique CONCEPT* user uses two sessions for each database in INGRES, one for query processing and the second for updates. This means that the number of sessions asked for during installation of INGRES must be configured to be at least two times the number of concurrent users.

Unique CONCEPT uses cursors when querying INGRES databases, so the number of concurrently open cursors must be configured to be at least as high as the highest number of open regions in the applications.

16.3.2. OPTIMIZING SQL STATEMENTS

Unique CONCEPT has its own logic to optimise the SQL statements that is built at runtime.
In ONLINE applications it is more important to get a quick answer than the overall time for the whole search. In most cases a sorted search on other column(s) than the primary key will give shorter response time by letting *Unique* search on a secondary index directly, and getting the rest of the column for each of the found records in the search. This strategy will be chosen under the following circumstances :

- ONLINE application with ascending sorting.
- A secondary index on the sorted columns exists.
- The 'db-options=no_override_optimiser' is not set in the **uqconfig** file (see next section).

16.3.3. INNER and OUTER joins.

Unique CONCEPT supports join tables defined as INNER or OUTER. With INGRES *Unique CONCEPT* takes full advantage of inner joins, but INGRES does not have support for outer joins (even if it is possible to construct rather complex select expression to obtain the same result). *Unique CONCEPT* will execute outer joins as dynamic joins with INGRES, the joins will be done manually by *Unique CONCEPT*.

16.4. How to set up uqconfig for use with INGRES.

The following parameters may be used in the database section in the uqconfig file to define an INGRES database (name MYBASE used as an example):

database: MYBASE
db-type = INGRES
db-server = <INGRES server name>
db-user = INGRES user account>
db-long-name = Demo database INGRES
db-name-prefix = <characters to prefix all object names>
db-owner = <INGRES user account>
db-options = [no_override_optimiser, ignore_uqdic]
db-timeout = <time interval in sec> : <time interval in sec> | 20 : 15
db-name-length = <max length of object names>
db-null-values = [ON | OFF]

- all parameters except db-type are optional.

- db-server : defines the name of the server to access (default is local machine), to be used **only** if this is a 'client' installation with INGRES/NET.

- db-user : only the DBA can use this option to test an application as another user, everyone else gets their login name as user name. This parameter must be used for testing only - it should **never** be used in production.

- db-owner : defines the name of the owner of the tables. To be used if a database has two tables with the same name but different owners.

- db-options : *no_override_optimiser* disables *Unique*'s optimise strategy, and all is taken care of by the INGRES optimise.
ignore_uqdic defines that *Unique*'s dictionary should not be used even if it exists.

- db-timeout : defines the time-out interval. The first interval specifies the time-out for the query session, and the second for update transactions. Timeout is the interval Unique is to wait for if there is a lock on the wanted data, before a message is given to the user and the application returns to its prior state.

- db-nullvalues : defines whether null values are allowed in the database or not. It is strongly recommended by both INGRES and *Unique* that null values are disallowed.

See the subject 'uqconfig' in the manual *Unique CONCEPT Operations Guide* for more details.

16.5. Environment variables.

One environment variable must be set:

II_SYSTEM : home directory for INGRES software (defined at installation time)

All users accessing INGRES should have this variable defined in their .**profile** or .**login** file.

Unique tolerates all settings of the INGRES environment variables, but overrides a few of them. These are :

```
set autocommit [ON | OFF]
set lockmode session where   level = page,
                             readlock = nolock | shared
                             timeout = <value from uqconfig>
```

The *autocommit* and *lockmode* statements are used by Unique where necessary and override prior settings of these INGRES environment variables at runtime.

16.6. How Unique Start generates an INGRES database

16.6.1. DATABASE

A Unique database has a one to one relationship with a database in INGRES. In which server the database exists is defined by the parameter db-server in uqconfig (omitted in one machine configurations). START DBS may **not** be run on a client when creating or dropping a database.

When a database creation is successfully completed, START will run the utilities optimisedb and sysmod on the database. Finally a checkpoint will be taken using the utility "ckpdb +j" which also activates journaling.

optimisedb should be run again after loading data into the database and regularly in production according to the guidelines given in the INGRES Database Administrator Guide.

Sizes in DBS schema are ignored with INGRES.

16.6.2. AREAS

Unique START will map AREAS to INGRES locations. Note that all needed locations must be defined using accessdb prior to running Unique START. This is due to the fact that there is no line oriented utility to create locations in INGRES.

Unique START will define one data location and the journaling location (LOG AREA) when creating the database using the utility createdb. If more than one data location is needed, Unique START will escape to a UNIX shell and prompt the user to run accessdb to extend the database with the defined locations. After exiting the shell, Unique START will complete the creation of the database.

An alternative method is to create the database and extend it prior to running Unique START. Unique START will recognise an empty database and add all definitions to it bypassing the creation step.

See INGRES System Administration Guide for further details.

16.6.3. INDEXES

Unique START will create secondary indexes using btree, hash or Isam structures as defined in the DBS schema. The default structure is btree.

Primary or clustered indexes will be implemented issuing a "modify <table> to .." statement.

Note that the Unique 4GL function *key* will run more efficiently when based in a btree index because the INGRES optimiser will recognise such indexes to be sorted. However, for *join-keys* other structures will prove as efficient as btree.

If you plan to load large volumes of data from files into the database it will be beneficial to first create the database without indexes , load the data and finally create all indexes using the Generate Indexes command in Unique START.

In Unique START, index names and table names do not need to be Unique throughout the database. This is not the case in INGRES and may cause a problem. Unique START solves this by post fixing the index name with the string "_q<i>" where <i> is a sequential number starting from 1, when duplicate, secondary index names are detected. Note that this will cause the name to be 3 characters longer, and consequently the db-name-length parameter in the uqconfig should be increased (db-name-length defaults to 8).

If care is taken to design the database with unique index names throughout the database, this situation will never occur.

16.6.4. WIDE COLUMNS

Wide columns are not supported in INGRES. Unique uses a separate table (widetab) to simulate this functionality. START will create this table automatically when needed.

16.6.5. CONSTRAINTS IN INGRES

Unique START does not create INGRES constraints, but these may be created with ISQL/SQL after the database is created, and Unique will display error messages if these constraints are breached at runtime.

16.7. Data types

The following relations exist between Unique data types and INGRES data types:

Unique type	INGRES-Type	
alphanumeric(n)	char(n), varchar(n)	1)
binary(n)	char(n)	
bit	integer1	
date	date	
datetime	date	
image	<simulated>	
integer1	integer1	
integer2	integer2, smallint	
integer4	integer4, integer	
money	float	
packed decimal(m,n)	float	
real4	float4, real	
real8	float8, float, money	
text	<simulated>	
time	integer4	
unpacked decimal(m,n)	float	
unpacked decimal(m,n) Separate	float	

The Ingres types object_key and table_key are not supported.

1) Varchar(n) can be generated by using the START type varchar(n) or string(n). Unique 4GL treats varchar(n) and char(n) in exactly the same way.

17. SIBAS supplement

17.1. Introduction to SIBAS

SIBAS is an advanced database management system. It is beyond the scope of this document to give a description of SIBAS.

17.2. How *Unique START* generates a SIBAS database

When generating a SIBAS database with Unique Start, The DBS compiler will first create DDC and DRL files and then optionally run SIB-DRL. In order for SIB-DRL to generate the database successfully, ensure that the database administrator specified in the SW-Config file has access to the database files.

Names used for identifying columns, indexes, tables, areas and the database may be up to 8 characters long.

17.2.1. CREATE DEFAULT

DEFAULTS are not implemented in SIBAS and are ignored if they are encountered in the DBS file when generating a SIBAS database.

17.2.2. CREATE TYPE

The **create type** statement in the DBS file is equivalent to the **define type** statement in the DDC file.

If purpose text is defined with the type in the DBS file then this will not appear in the DDC file but will be used as the default purpose for columns defined using this type.

The key word **generic** is meaningless when generating SIBAS databases. All wide columns are generic.

The key words **null** and **not null** are meaningless when generating SIBAS databases. SIBAS columns cannot have a null value. If numeric SIBAS columns are given a zero value or alphanumeric columns are given a blank value no entry will be made in the index trees.

The display code in the DBS file is compatible with that used by SIBAS products.

17.2.3. CREATE CATALOGUE-AREA

The **create catalogue-area** statement in the DBS file has no meaning when generating SIBAS databases and is ignored.

17.2.4. CREATE AREA

The **create area** statement in the DBS file is equivalent to the **new os-file** statement in the DRL file.

Os-files can be dedicated to the following :

- System-tables
- Wide-storage-tables
- Tables and Hash-tables

The following rules apply:

- Many System-tables may be stored on the same os-file.
- Many Wide-storage-tables may be stored on the same os-file.

- Many Tables and Hash-tables may be stored on the same os-file.
- Os-files containing System-tables may not contain any other type of table.
- Os-files containing Wide-storage-tables may not contain any other type of table.

If the page size (calculated by Unique Start) is different to the standard 512 words it will be specified in the DRL file. Note that a word in the context of SIBAS databases is 2 bytes.

EXAMPLE

```
new os-file    MY-AREA
  pagesize   1024.
```

The size component in the DBS file will not appear in the DRL file. It will however be used in creating the files as contiguous.

Other components in the DBS file dealing with extents have no meaning when creating SIBAS databases and are ignored.

If the name of the OS-file contains an underscore character (_) it will be converted to a hyphen (-) when generating databases on operating systems which do not support the use of underscore in filenames.

17.2.5. CREATE SYSTEM-TABLE

The **create system-table** statement in the DBS file is equivalent to the **new system-table** statement in the DRL file.

EXAMPLE

```
create area  MYAREA
    on     'MY-AREA'
create system-table SYSTAB1
    area MYAREA
```

The above section of a DBS file will cause the following to be included in the DRL file when generating SIBAS databases.

```
new os-file      MY-AREA.
new system-table SYSTAB1
   os-file      MY-AREA
   table-size   100.
```

The tablesize will be calculated by Unique Start based upon the largest index to be stored in the system-table, the number of rows to be stored in the regular (data) tables and the internal space requirements SIBAS needs for the system table. The default maximum number of rows which are allocated to regular tables is 5000.

Heading and purpose texts may also be specified in the DBS file and they will be reproduced in the DRL file.

Other components in the DBS file dealing with extents have no meaning when creating SIBAS databases and are ignored.

OS-FILES which are used for storing system-tables should not be used for storing other types of table.

ADDITIONAL OS-FILES cannot be specified in the DBS file. If you wish to include these in the DRL file you must introduce them manually using an editor.

17.2.6. CREATE WIDE-TABLE

The **create wide-table** statement in the DBS file is equivalent to the **new wide-storage-table** statement in the DRL file.

EXAMPLE

```
create area          WIDEAREA
create wide-table WIDETAB1
      area          WIDEAREA
      size          10M
```

The above section of a DBS file will cause the following to be included in the DRL file when generating SIBAS databases.

```
new os-file          WIDEAREA.
new wide-storage-table WIDETAB1
   os-file          WIDEAREA
   table-size       10240.
```

The **size** specified in the DBS file is in bytes. This is converted to a number of pages in the DRL file. If the default pagesize of 512 words is used then 10M will be converted to 10240. This is explained as follows:

10M is 10485760 bytes.
1 page is 512 word which is 1024 bytes.

10M is therefore equivalent to 10485760 / 1024 pages which is 10240 pages.

Heading and purpose texts may also be specified in the DBS file and they will be reproduced in the DRL file.

Other components in the DBS file dealing with extents have no meaning when creating SIBAS databases and are ignored.

OS-FILES which are used for storing wide-storage-tables should not be used for storing other types of tables.

ADDITIONAL OS-FILES cannot be specified in the DBS file. If you wish to include these in the DRL file you must introduce them manually using an editor.

17.2.7. CREATE TABLE

The **create table** statement in the DBS file is equivalent to the **new table** statement in the DRL file.

The **size** specified in the DBS file in the **create table** statement refers to the maximum number of rows that can be stored in this table. The *Unique START DBS compiler* calculates the number of pages to be allocated to table from the following:

- The length of rows to be stored in the table. This includes space allocated to set-teferrals and is calculated by Unique Start.
- The number of rows allocated to the table specified in the **size** clause in the **create table** statement in the DBS file.
- The page size calculated by Unique Start based upon the maximum length of table rows stored upon that os-file.
- Internal requirements of the SIBAS DBSM.

Heading and purpose texts may also be specified in the DBS file and they will be reproduced in the DRL file.

If there is a key defined on the table which has the **hash** attribute then a hash-table will be created instead of a regular table.

17.2.8. COLUMN

The **column** statement in the DBS file is equivalent to the **new column** statement in the DRL file.

Most attributes which are specified in the **create type** statement may also be specified in the **column** statement. Those that are specified at the column level will be included in the **new column** statement in the DRL file. In addition if the type of the column is wide (text or image) at the primitive level you may specify the wide-table to be used by the wide column and this will appear in the **new column** statement in the DRL file.

17.2.9. CREATE PRIMARY /UNIQUE/DUPLICATE/KEYWORD KEY

The **create ... key** statements in the DBS file are equivalent to the **new index** statement in the DRL file.

A Unique key is implemented as a simple unique index. When creating the DRL file the group keys will be generated after the simple keys.

Heading and purpose texts may also be specified in the DBS file and they will be reproduced in the DRL file when generating group-columns but not on simple columns.

Keyword keys may also be used.

The keyword hash will cause this key to be used as a hash key and will alter the definition of the table.

If the keyword **virtual** is used no index will be generated in the DRL file for this key definition.

The keywords **clustered**, **unclustered**, and all of those associated with extents have no meaning when generating SIBAS databases and will be ignored.

17.3. Data types

The following relations exist between Unique data types and SIBAS storage code and data types:

Unique type	SIBAS storage code	SIBAS type
alphanumeric(n)	alphanumeric(n)	character
binary(n)	nonstandard(n)	integer
bit	nonstandard(1)	integer
date	date	integer
datetime	nonstandard(8)	integer
image	nonstandard	wide
integer1	nonstandard(1)	integer
integer2	integer2	integer
integer4	integer4	integer
money	nonstandard(9)	integer
packed decimal(m,n)	packed decimal(m,n)	integer
real4	real4	floating
real8	real8	floating
text	text	wide
time	time	integer
unpacked decimal(m,n) Sep	unpacked decimal(m,n) Sep	integer
unpacked decimal(m,n)	unpacked decimal(m,n)	integer
varchar(n)	alphanumeric(n)	character

1) Varchar(n) can be generated by using the START type varchar(n) or string(n). *Unique CONCEPT* treats varchar(n) and char(n) in exactly the same way.

The following SIBAS storage codes are not supported: numeric text, real6, week, text s, text t, alphanumeric and alphanumeric parity.

Columns repeated using the keyword REPEAT are not supported in the START DBS syntax.

18. P.INK SQL supplement

18.1. Introduction to P.INK SQL.

P.INK SQL is an advanced relational database management system. It is beyond the scope of this document to give a description of P.INK SQL.

To take advantage of this document, the reader must be familiar with basic concepts in P.INK SQL and in particular how to operate and administrate a P.INK SQL database.

Unique START is not updated to create P.INK SQL databases, databases must therefore be created using the tools provided with P.INK SQL.

18.2. Using SQL-EXEC on P.INK SQL databases

Unique CONCEPT does the following when SQL-EXEC is executed :

1) The SQL command is sent to the server for execution.
2) If a BIND-LIST is defined *Unique CONCEPT* sends the command *"fetch next"* and a *"print"* for each column in the BIND-LIST. Therefore it is not legal to use *"fetch"* or *"print"* as the SQL command in SQL-EXEC.

18.3. Data types

The following relations exist between Unique data types and P.INK SQL data types:

Unique type	P.INK SQL Type	
alphanumeric(n)	char(n), varchar(n)	1)
binary(n)	binary(n), varbinary(n)	2)
bit	boolean	
date	date	
datetime	timestamp	
image	<not supported>	
integer1	<not supported>	
integer2	smallint	
integer4	integer	
money	money	
packed decimal(m,n)	decimal	
real4	smallfloat	
real8	float	
text	<not supported>	
time	integer	
unpacked decimal(m,n) Separate	<not supported>	
unpacked decimal(m,n)	<not supported>	

1) *Unique CONCEPT* treats varchar(n) and char(n) in exactly the same way.
2) *Unique CONCEPT* treats varbinary(n) and binary(n) in exactly the same way.

19. TECHRA supplement

19.1. Introduction to TECHRA dbms

TECHRA is an advanced relational database management system. It is beyond the scope of this document to give a description of TECHRA.

To take advantage of this document, the reader must be familiar with basic concepts in TECHRA and in particular how to operate and administrate a TECHRA database.

When using 4GL the db-user and db-password elements will be used when logging into the TECHRA server identified by db-server. When using *Unique START*, in order to create a new logical database (directory) in a physical Techra database it is necessary to know the name and password of a TECHRA user defined with either system (S) or Total (T) privilege. You must specify these when called for by *Unique START*.

Names used for identifying columns, indexes, tables, areas and the database may be up to 16 characters long. By default the names are truncated to 8 characters. If you wish to increase this you must specify the maximum length by specifying **db-name-length** in the **uqconfig** file. The **db-name-length** cannot be extended to beyond 16.

Names used for identifying columns, indexes, tables, areas and the database may not be the same as any TECHRA reserved word. To avoid this conflict the identifiers for columns, indexes, tables and directories can be prefixed by specifying **db-name-prefix** in the uqconfig file. The length of the identifiers including the prefix must not exceed 16 characters.

Note that the default actual name for databases is the logical database name prefixed with the **db-name-prefix**. Except for the prefix, the name can be overruled by specifying the **db-physical-name** in the uqconfig file.

Since areas in a physical TECHRA database may be shared amongst several logical databases each with a different **db-name-prefix**, no prefix is applied to the name for areas by default. This can be changed using the **Mode** menu of *Unique START*.

EXAMPLE

```
database:MYBASE
  db-type=TECHRA
  db-physical=MYBASE
  db-name-length=16
  db-name-prefix=u_
  db-server=prodbase
  db-user=uniquser
  db-password=?
```

Before trying to create a TECHRA database ensure that the relevant TECHRA processes are active.

19.2. How Unique START generates a TECHRA database

19.2.1. CREATE SCHEMA

The **create schema** statement is equivalent to the Techra **create directory** statement.

EXAMPLE

```
create schema
    database-name ACCOUNTS
    version     '3-12-0'
    heading      "Accounting database"
    purpose     ('Customer accounts'
                   'from 1975 onwards')
```

The above section of a DBS file will produce the following SQL statement:-

```
create directory ACCOUNTS
```

The version, heading and purpose texts will be stored in the Unique data dictionary table after the database, all its tables and indexes have been created.

19.2.2. CREATE DEFAULT

DEFAULTS are not implemented in TECHRA and are ignored if they are encountered in the DBS file when generating a TECHRA database.

19.2.3. CREATE TYPE

The **create type** statement in the DBS file has no direct equivalent in TECHRA databases but the primitive storage attribute will be used in column definitions in create table statements. The other attributes of the type statement are inherited by the columns. All of the attributes of the columns are stored in the Unique data dictionary and are of use to Unique 4GL when developing and compiling applications.

19.2.4. CREATE AREA

If Start is configured to use default areas for the server all areas definitions will be ignored. Under such circumstances the database tables and the data dictionary will be stored on the standard system area called sysare.

It is not possible to separate index trees and data when creating TECHRA databases - consequently system-table statements will be ignored.

It is not possible to separate wide column data from normal table data when creating TECHRA databases. Wide columns are implemented as sequences in the TECHRA table - consequently wide-table statements will be ignored.

Areas which are not referred to by tables will be ignored.

If Start is configured to use private areas the **create area** statement in the DBS file will first check to see if the area exists already and give an error message if it does. If the area does not exist it will be created and will be available for referencing in the creation of tables.

If Start is configured to allow common areas, the areas may exist already and contain tables from other databases. If the area does not exist it will be created and will be available for referencing in the creation of tables.

EXAMPLE

```
create area  MYAREA
     on   '/unique/database/test/john'
```

The above section of a DBS file will produce the following TECHRA statement:-

```
create area  MYAREA
     file="/unique/database/test/john"
     access=(OWNER=T PUBLIC=W)
```

The access information OWNER=T is specified to indicate that the owner has total access to the area. PUBLIC=W is specified so that everybody can store information in tables defined on this area. This access information cannot be changed using the DBS syntax. To change the access of an area you must do so in
either *sql* or *sqline*.

If no path is specified in the area definition in the DBS file the area is created in the directory specified by the **db-path** option in the **uqconfig** file. If this is not specified then the current directory is used.

It is imperative that if a new area is to be created the file must not be used by another area. Great care should be taken to ensure that one physical file is not used by two areas. If several physical TECHRA databases are used, ensure that the files for the areas in all the physical databases are kept separate.

Never delete a file that is used by a TECHRA area. The consequences of doing so are catastrophic and may result in the entire physical database with all of its constituent databases (directories) being lost.

19.2.5. CREATE SYSTEM-TABLE

The **create system-table** statement in the DBS file has no meaning when generating TECHRA databases (directories) and will be ignored.

19.2.6. CREATE WIDE-TABLE

The **create wide-table** statement in the DBS file has no meaning when generating TECHRA databases (directories) and will be ignored.

19.2.7. CREATE TABLE, COLUMN, CREATE PRIMARY KEY

The **create table** statement in the DBS file with all of its subsequent column definitions and primary key is equivalent to the **create table** statement in TECHRA.

EXAMPLE

```
create table      EMPLOYEE
     area         DATAFILE1

     column       EMPNAME  NAME
     column       EMPNUM   NUMBER
     column       DATEBORN DATE
     column       TAXNUM   NUMBER

     create primary   key     EMPNUM
```

The above section of a DBS file will cause the following TECHRA statement to be generated.

```
create table EMPLOYEE (EMPNUM   longint  key,
                       EMPNAME  char(10),
```

```
DATEBORN longint,
TAXNUM   longint)      AREA=DATAFILE1
```

Note that the order of the columns in TECHRA database table is altered such that the primary key column(s) are placed at the beginning of the table. This is a feature of TECHRA which cannot be ignored. To ensure that the sequence of columns in the resulting table is the same as that which you intend, define the columns of the primary key before all others in the **create table** statement in the DBS file. If the primary key consists of several columns the order of the columns in the primary key definition should be the same as that of the first columns in the table definition.

Every table definition must have a primary key. If no primary key is located in the DBS file for a particular table the first unique key will be promoted to being a primary key for that table. If the table contains neither primary nor unique keys an error message will be issued.

The heading and purpose texts will be stored in the Unique data dictionary table after the database and all its tables and indexes have been created.

19.2.8. CREATE UNIQUE KEY, CREATE DUPLICATE KEY

The **create unique key** and **create duplicate key** statement are used to generate secondary keys (indexes) on the relevant tables. If an unique key is promoted to being a primary key due to the lack of a primary key definition for a particular statement then that key will not also be used to generate a secondary key.

The heading and purpose texts will be stored in the Unique data dictionary table after the database and all its tables and indexes have been created only for those keys which are group keys.

19.2.9. CREATE KEYWORD KEY

TECHRA does not support keyword indexing. Specifying the **create keyword key** statement in the DBS file when generating a TECHRA database will result in an error message.

19.3. Techra 2.31 and 2.32

Unique CONCEPT can be linked with both the 2.32 and 2.31 version of TECHRA, please consult your tape label for prelink information and the installation guide for information on how to link with different DBMS versions.

19.4. Data types

The following relations exist between Unique data types and TECHRA data types:

Unique type	TECHRA Type
alphanumeric(n)	char(n), string(n)
binary(n)	byte(n)
bit	int
date	longint
datetime	byte(8)
image	seq byte
image generic	seq byte
integer1	int
integer2	int
integer4	longint
money	byte(9)
packed decimal(m,n)	byte((m+n)/2+1)
packed decimal(m,n) unsigned	byte((m+n)/2+1)
real4	real
real8	longreal
serial	longint
text	seq char
text generic	seq char
time	longint
unpacked decimal(m,n)	byte(m+n)
unpacked decimal(m,n) separate	byte(m+n+1)
week	long int

- The Techra 2.32 type decimal is not supported by *Unique START*. It is equivalent to byte.
- Group items are not supported in the DBS syntax.

20. UQISAM supplement

20.1. Introduction to UQISAM dbms

UQISAM is a simple, but robust database system using standard operating system files.

It requires no dedicated DBMS processes, the database software is completely embedded in the application program. This means there are no operating procedures for starting/stopping the database, it is always available.

UQISAM is available for single user environments only. Only one UQISAM database can be open simultaneously.

20.2. *UqKeyGen* program for UQISAM dbms

The *UqKeyGen* program is a separate utility to rebuild all the indexes in a corrupt UQISAM database.

Usage:

uqkeygen <options> dbname

Options:

-?	Help information
-p<path>	Path containing database files
-o<filename>	Output file

Note:

The key files must exist before running uqkeygen. If the -p option is used, the path must end with a '/'.

21. Terminal and Printer definitions supplement

21.1. Terminal supported

Unique Terminal Description (.UTD) files included in this version:

File name	Description	Language
Standard files. These files are included by Unique CONCEPT at runtime, and must therefore not be deleted.		
STDED	Standard function keys in Unique editor	
STDUQ	Standard functions keys in Unique	
VTCHARS	Standard definition of CHARSET4 in VT100/VT220 UTD's	
VTCOLS	Standard definition of COLOURS in VT100/VT220 UTD's	
Supported terminal files:		
IBM3151	IBM 3151 monochrome	Norwegian
M303	ICL M303 monochrome	English
M305	ICL M305 colour	English
ND320	ND-320 (Tandberg TDV 2200/9S)	
	ND-246 (Tandberg TDV 2200/9) / ND-246/V2	Norwegian
TDV2200	Tandberg TDV 2200/9S	Norwegian
PC	PC (DOS and OS/2)	Norwegian
SUN	SunView	English
TDV1200	ND-110140 (Tandberg 1200-1 el. TINY)	Norwegian
VT100	vt100 - 7-bit/8-bit characters	English
VT220	vt220 - 7-bit/8-bit characters	English
VT220PC	vt220 - 7-bit/8-bit characters	
	with PC keyboard (terminal emulator)	English
XTERM	XTERM	English

We distribute some terminal description files that we do not support or that may be used as templates for your own terminal description files. These files are:

ANSI
ANSI-COL
HP-2392A (ANSI mode)
IBM6091
PC-ANSI
PC-NOTIS

If special definitions are required, e.g. vt220 emulator definition - copy the original .utd file to a different name, edit the file and set the new environment variable **UQTERM** to the value of the new **.utd** file. This allows you to have more than one vt220 .utd file.

21.2. Printers supported

Unique Printer Description (.UPD) files included with this version:

The files may be used as templates, and customised to cover your needs.

File name	Description
EPSON	Epson LX80,FX80,RX80,LX800 and FX800
HERMES	Hermes Line Printer
IBM4224	IBM 4224 Printer
IBM5204	IBM 5204 Printer
IBMLP	IBM 4019/4029 - A4 Portrait (ASCII mode)
IBMLPL	BM 4019/4029 - A4 Landscape (ASCII mode)
IBMLPW	IBM 4019/4029 - A4 Portrait wide (ASCII mode)
IBMPRO	IBM Proprinter X24E / XL24E
JETIID	Laserjet IID Printer
JETIIDL	Laserjet IID Printer (Landscape)
JETIIDW	Laserjet IID Printer (Wide)
LBP8A12	Canon LBP-8 A1/A2 Laser Printer
LBP8IIR	Canon LBP-8IIR Laser Printer
LBP8IIT	Canon LBP-8IIT Laser Printer
LBP8IITL	Canon LBP-8IIT Laser Printer (Landscape)
LINPRINT	Line Printer
NECPIN	NEC PINWRITER P60/P70
PCL	HP PCL 4
PHILGP	Philips GP 300, GP 150, GP 100

See the Unique Concept Operation Guide - Printing from Unique CONCEPT.

NOTE: With MS Windows any printer supported by Windows may be used!

22. Unique Debugger supplement

Unique CONCEPT complies a versatile debugging environment. Available documentation includes a brief description of the structure and the accessible elements in the Unique environment. This documentation can be obtained by contacting customer support.

23. Application Stack supplement

The Application Stack is the memory area where Unique stores loaded applications. Unique will keep as many applications as possible in the Application Stack. If the size of an application exceeds available space in the Application Stack, all currently loaded applications will be swapped to a file, and reread into memory when needed.

The Application Stack is introduced to reduce the need for dynamically allocated memory, and in most cases should eliminate the danger of running out of memory (MEMORY_EXHAUSTED). It was first introduced to avoid limitations in the available memory on MS-DOS, but is also used on other platforms.

If the requested size of an application exceeds the total space of the Application Stack, a warning will be given and loading will be aborted. To be able to run that application you have to restart Unique with a larger Application Stack.

The Unique-4GL compiler will also use the Application Stack when compiling. The Application Stack is then split in two parts; one part for fields and one for functions. A "pass 0" is performed by the compiler to count the number of fields in the application, to be able to calculate how to split the Application Stack. The command line option -k may also be used.

If the size of the Application Stack is exceeded during compilation, Unique will abort with an appropriate message.