AGSO

AGSO ORACLE DEVELOPERS' GUIDE

by
Mirek Kucka

RECORD 1994/4

AGSO

AUSTRALIAN
GEOLOGICAL SURVEY
ORGANISATION

AGSO Oracle

Developers' Guide

Record 1994/4

Mirek Kucka

AUSTRALIAN GEOLOGICAL SURVEY ORGANISATION

DEPARTMENT OF PRIMARY INDUSTRIES AND ENERGY

Minister for Resources: Hon. David Beddall, MP
Secretary: Greg Taylor

AUSTRALIAN GEOLOGICAL SURVEY ORGANISATION

Executive Director: Harvey Jacka

ISSN: 1039-0073
ISBN: 0 642 20116 1

Table of Contents

INTRODUCTION

AGSO has two Corporate database instances called the Oracle Test and Oracle Production environments. They are also referred to as ORATEST and ORAPROD. The two environments are totally separate, contain their own data and dictionaries, and have their own copy of the Oracle software.

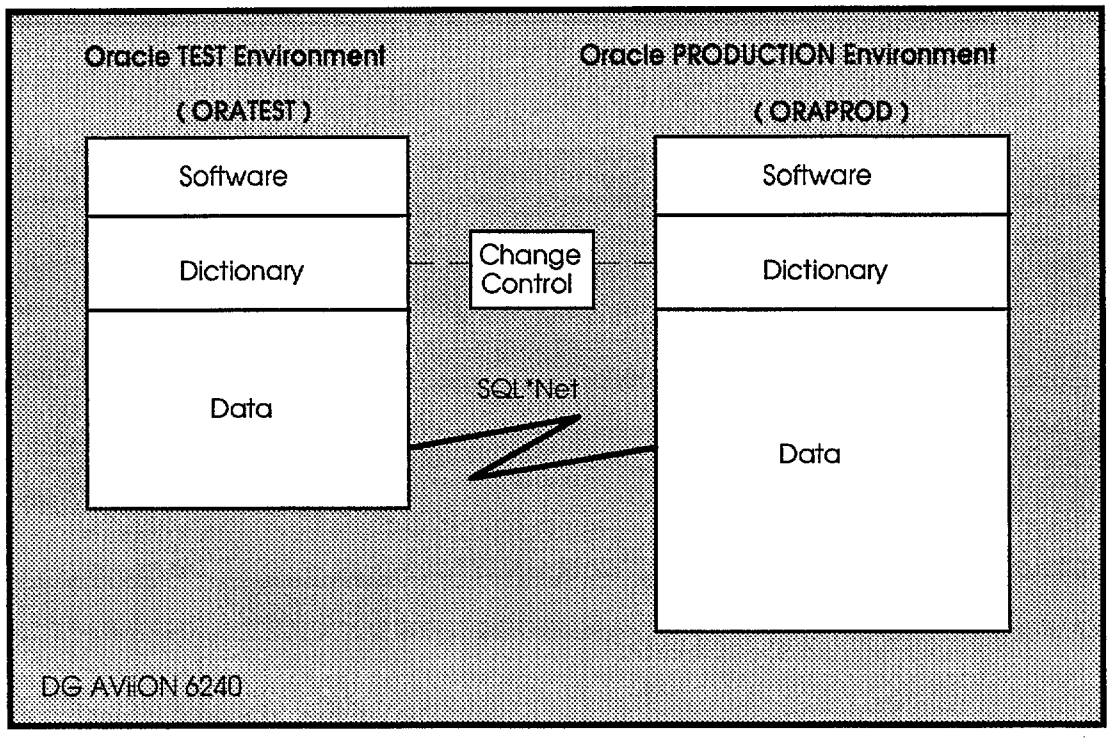

Figure 1 : Graphical representation of the Corporate Oracle environment in AGSO

This guide aims to provide AGSO developers with basic information about how the Corporate Oracle is set up and used within AGSO. The information covers a wide variety of topics, such as: how to select the required Oracle environment, change control management, terminal emulation, and SQL*Net - topics not usually covered by Oracle manuals.

In brief, this guide is a 'mixed bag' of topics and information in order to help developers find their way around the AGSO Corporate Oracle environment faster.

Many of the points touched on here are covered in more detail in AGSO Record #92/85 called 'Dual Oracle Environment and Change Control Management'.

ORACLE PRODUCTION ENVIRONMENT

Only production work is carried out in the Oracle Production environment. It is a tightly controlled environment where no development work is allowed. Change to this environment involves making the change in the Test Oracle environment first, and then migrating the change into the Production environment through change control (see Figure 1 and section titled 'Change Control Management').

Typically, the Production environment will have:

- high system priority
- high level of recoverability
- eventual performance monitoring of both the RDBMS and user programs
- inactivity timeout, starting with Oracle version 7

There are two different classes of userids in the Production environment. The **system userids** are the userids that own an Oracle application/system such as GEODX, ROCKCHEM, NGMA etc. These are the only userids that have the 'resource' attribute granted to them. However, the four data definition language (DDL) verbs 'alter', 'create', 'drop' and 'rename' have been disallowed. These userids still retain the ability to grant and revoke access rights to the system's objects.

The second class of userids are the **personal userids** - these form the majority of userids. They are normally based on the person's first initial and surname, making a maximum length of eight characters, ie Fred Bloggs would have a userid like FBLOGGS.

Personal userids only have the 'connect' attribute, which means that they cannot create tables or indexes. They can create synonyms and views, but will only be able to manipulate the data that they have been granted access to, through SQL*Forms, SQL*Plus, SQR etc.

ORACLE TEST ENVIRONMENT

The Test environment is where all development and testing takes place, such as designing new databases or modifying existing ones and developing or modifying programs (SQL*Forms, SQL*Plus reports etc). Training should take place in this environment.

The Test environment has less restrictions than the Production environment, except for one detail: the Test environment is smaller than Production in terms of space. Typically the Test environment will have

- restrictions on the amount of space most userids can allocate. The Test environment should only contain test data or subsets or parts of the Production data.
  There will be instances where certain userids will not have space limitation due to their particular needs (such as massaging data from other sources, AGSO or external)
- lower priority to that of Production. However, in most cases the response time from Test will be faster than Production due to the smaller amount of data that has to be accessed/manipulated
- lower level of recoverability
- inactivity timeout, starting with Oracle version 7

The Test environment has the same classes of userids as Production.

The **system userids** have the 'resource' attribute granted to them with no restrictions on DDL verbs.

The **personal userids**, based on the person's initial and surname can also have the 'resource' attribute, if it is requested.

SETTING THE ORACLE ENVIRONMENT

As there are two Oracle environments, you first have to select the Oracle environment that you wish to work in. Two commands have been provided for selecting the desired environment:

The **Bourne Shell** is the default shell on the AViiON.

setoratest - sets up the Test Oracle environment
setoraprod - sets up the Production Oracle environment

Due to the peculiarities of the Bourne shell, these commands do not work properly under certain conditions. If this is the case, you will need to enter:

. *setoratest* (there is a space between the dot and the command.)
. *setoraprod*

The commands for the **C Shell** are:

source SETORATEST - sets the Test Oracle environment
source SETORAPROD - sets the Production Oracle environment

Make sure that your PATH includes the directory '/usr/local/bin'.

If you attempt to use Oracle without setting the appropriate environment, you will receive an error message like:

..... not found
Sh: File does not exist

Once the Oracle environment is set, you will be able to access it. If you are only using one environment, insert this command into your '.profile' file if using Bourne Shell (or '.login' file if using C Shell), so that the desired Oracle environment is always set on login.

When you are in a SQL*Plus session, the SQL prompt will indicate which environment you are in, ie

prompt in Oracle Production is SQL-Prod>
in Oracle Test is SQL-Test>

Note that this informative prompt is not available in a client-server type access.

Environment Table

A publicly accessible table has been provided called ENVIRONMENT which contains information on the Oracle environment (ie Test or Production) and also the organisation details. This table can be used by programs to display the environment in the forms (so users can see which environment they are running in), or reports etc.

The structure of the table is:

Field Name	Value
ORG_KEY	1 (This is the primary key)
ORA_ENVIRONMENT	TEST/PRODUCTION
ORG_SHORT_NAME	AGSO
ORG_LONG_NAME	AUSTRALIAN GEOLOGICAL SURVEY ORGANISATION
ORG_ADDRESS1	GPO BOX 378
ORG_ADDRESS2	CANBERRA ACT 2601

This table will be kept up-to-date, and should be selected by:

```
select ....., ....., .....
        from environment where org_key=1;
```

For BRS clients, ORG_KEY value of 2 reflects the BRS organisation details.

CHANGE CONTROL MANAGEMENT

The change control management, in short, is about controlling DDL (Data Definition Language) changes to the Oracle Production environment. The change control process itself is described in 'Dual Oracle Database Environment and Change Control Management' (AGSO Record 1992/85).

This guide is an addition to the above record, describing the actual process of requesting a change to a system which is in the Oracle Production environment.

The Change Control Directory

The most important part of the physical implementation of change control management is the use of the directory -

/home/oraprod/change_control

This directory holds pending changes to systems as well as changes that have been applied to systems (ie it contains a complete audit of historic changes).

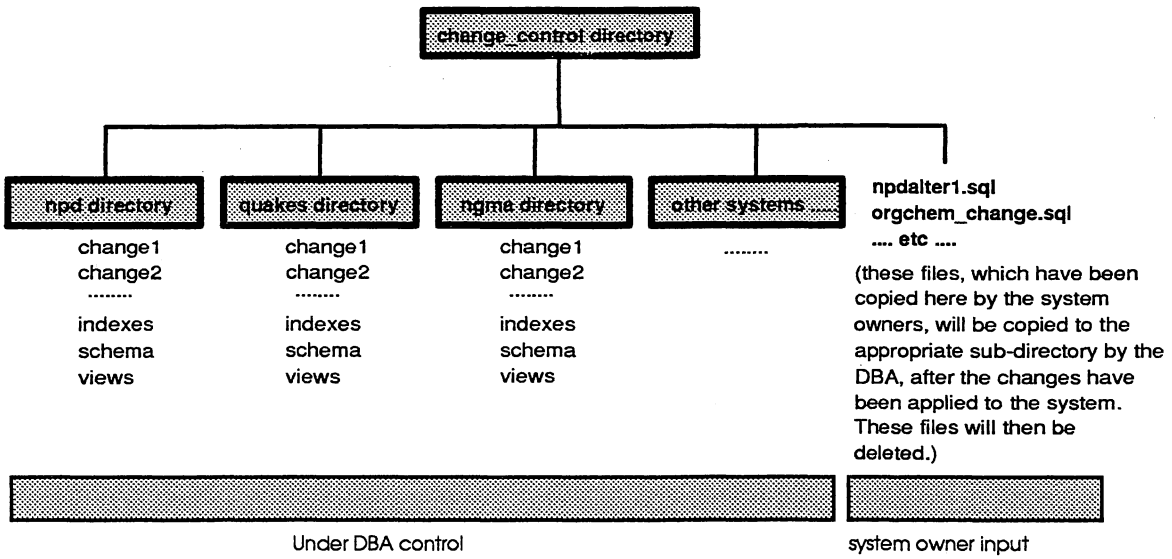

Figure 2: Graphical representation of the change control directory structure

Historic Changes

The historic changes to a particular system are contained in sub-directories with the same name as the system. These sub-directories may contain four types of files. Note that all the files are read-only.

i files with names 'change1, change2, change3....' and so on. These files are the actual SQL scripts that were applied to the system (ie these are really the audit-trail of all changes to the system).

ii file called 'schema'. This file contains the SQL statements of the original system schema as it was when the system was first locked from change.

iii file called 'views'. This file is kept up-to-date by the DBA, and contains all the views that belong to the system.

iv file called 'indexes'. This file is kept up-to-date by the DBA, and contains all indexes belonging to the system.

Pending Changes

The change_control directory also holds SQL script files which have been copied there, typically by the system owner.

These script files contain SQL changes that are to be applied to systems. They are stored in this directory temporarily, until the changes have been applied. After the SQL script has been run by the DBA, it is copied to the system's sub-directory as 'change*x*' and then deleted from the change_control directory.

Steps in requesting changes to Production systems

1 The system owner/administrator sends a minute to the DBA requesting the change to the Production system. All other necessary documentation as specified in 'Dual Oracle Database Environment and Change Control Management' must also be attached.

The required SQL script (machine readable) should also be forwarded to the DBA. This can be by:

copying the SQL script directly into the directory:

/home/oraprod/change_control

or

sending the SQL script via Email to the userid 'oraprod' (the DBA will then copy the script to the above directory).

ie

mail oraprod < filename (where filename is the file you want to send)

Please include the name of the system in the file name if copying into the change control directory.

2 At a time agreed with the system owner, the DBA will run the SQL script to make the requested changes to the system. Once the changes are completed, the script is moved from the change_control directory to the appropriate system sub-directory as a read-only file, called 'change*x*'.

In cases where the change is a new/modified view, the DBA will update the file 'views' to ensure that it contains all up-to-date views that belong to that system.

Similarly, in cases where the change is to indexes, the DBA will update the file 'indexes' to ensure it contains all up-to-date indexes belonging to the system.

TERMINAL EMULATION

There are a number of different terminal types within AGSO that can be used to access the Oracle environments. In general, access to the AViiON is through VT220 emulation, although native DG mode terminals through the SYTEK network and VT100 emulation are also supported under some circumstances.

Oracle requires many different key sequences to map functions in SQL*Forms and SQL*Menu. The key mappings as supplied by Oracle are either not available for some terminals, or simply ergonomically unusable.

AGSO has designed and customised ergonomic key mapping layouts for the common terminal types, which resemble the PC Oracle key mapping as closely as possible. The AGSO key mappings have similar 'look and feel' on a number of different terminals, such as PCs in VT220 emulation, PCs under Racal Interlan, real VT220 devices, SUN workstations, and D412 terminals in VT220 mode.

The Unix environment variable 'TERM' tells both Unix and Oracle what terminal type you are using. Oracle also uses the phrase 'device type' to indicate the terminal type.

The valid 'device type' and 'TERM' values that both Unix and Oracle understand are:

TERM and Device type	Description
d412-dg	native DG mode, ie 410, 411 and 412 terminals in DG mode. This mapping is based on the key mapping of the old Data General MV machine
d412-vt	D412 terminal in VT220 emulation. The mapping is based on the 'pc-vt' mapping below
pc-vt	PC VT220 emulation, resembling key mapping of PC Oracle. The Racal Interlan also uses this key mapping with some minor differences. See below for more information on Racal
sun-k4	Sun terminal type 4, mapping based on 'pc-vt' key mapping
vt220	standard, real VT220 device, mapping based on 'pc-vt' key mapping
vt100	standard, real VT100 device, currently as supplied by Oracle

When invoking SQL*Forms/Menu, Oracle automatically looks at the TERM environment parameter to determine which device type you are using. It is possible to instruct Oracle to use a different device type than that specified by TERM, ie

1 *runform30 <form name>*
2 *runform30 -c <terminal device> <form name>*

In the first syntax, SQL*Forms uses the TERM environment parameter to determine which device type you are using. This will be the most common method of accessing Oracle Forms and Menu, and TERM should be set in your Unix '.profile' file.

The second syntax is an alternate method of invoking Forms and Menu which overrides the TERM parameter, however the device type has to be one of the device types listed above.

The same parameters can be used with the commands 'sqlforms30, sqlmenu50 and runmenu50'.

Using pc-vt Keyboard Mapping (for PCs)

Both Oracle Production and Test can use the device type 'pc-vt' to correctly map PC keys to Oracle functions, for PCs that are in VT220 emulation mode. This applies to VT220 emulators such as LAN Workplace for DOS, TurboTerm and the Racal Interlan.

The preferred method to make Oracle use this key mapping is to set the TERM variable to 'pc-vt', as Oracle by default uses the value of TERM to determine which device type to use:

TERM=pc-vt (using Bourne shell)
export TERM

or

setenv TERM pc-vt (using C shell)

This will typically be included in your '.profile' if you are using Bourne shell or '.login' if you are using C shell, so that TERM will be automatically set on login.

An alternative method is to tell Oracle explicitly what device type you are using when invoking Forms/Menu. ie:

sqlforms30 -c pc-vt (the parameter '-c pc-vt' tells Oracle to use the 'pc-vt'
runform30 -c pc-vt ... keyboard mapping)
sqlmenu50 -c pc-vt
runmenu50 -c pc-vt .....

LAN Workplace for DOS

LAN Workplace for DOS requires a tailored file 'bmr220.bin', which in conjunction with 'pc-vt' correctly maps the PC keyboard with Oracle. This file can be obtained from ISB, and it is suggested that it reside in a network directory, for instance *xln\bin40* .

In this case, set up a DOS batch file called *av.bat* in your DOS root directory as follows:

tnvt220 -f c:\xln\bin40\bmr220.bin av

Note: running Oracle from Host Presenter is not recommended, as the Windows keyboard mapping is different, some function keys are not available, and the appearance of the form/menu screen is different. Running Oracle from Windows as a DOS Window will work, however Windows (3.0 and 3.1) appears to hang frequently.

TurboTerm

TurboTerm requires a tailored file 'avtt220.scf' to correctly map PC keyboard with Oracle. This file, which has to reside in the TurboTerm directory, is available from ISB.

The command to invoke TurboTerm is -

tterm2 -avtt220 av

Racal Interlan

It is possible to access Oracle using PCs through the Racal Interlan VT220 emulator. There are several known differences with the Racal key mapping. The keys 'Insert', 'Home', 'PageUp', 'Delete', 'End' and 'PageDown' map to different Oracle functions, as described below, and also have to be invoked with a 'Ctrl' key.

PC Key	Normal Oracle function	Racal function	Racal PC Keys
Insert	Insert/Replace	Next Block	Ctrl Insert
Home	Next Block	Insert/Replace	Ctrl Home
PageUp	Previous Record	Previous Block	Ctrl PageUp
PageDown	Next Record	Next Record	Ctrl PageDown
Delete	Delete character (backwards)	Previous Record	Ctrl Delete
End	Previous Block	Delete character (backwards)	Ctrl End

Additionally, under Racal, the 'Esc Z' key does not work (equates to Oracle function 'Delete Line').

X-terminal emulation

AGSO is currently examining and evaluating the X-terminal emulation software package called 'eXceed'. PCs running this kind of software may possibly prove to be the best method of accessing the Corporate Oracle environment.

The potential benefits offered by X-terminal emulation technology include the use of mouse driven and full GUI (Graphical User Interface) screens - particularly relevant with the image handling capabilities of Oracle SQL*Forms version 4, which AGSO will be installing shortly on the Corporate database server.

Additionally, this Open standard technology may also provide a better alternative to the costly client-end software in client-server computing, and offer centralised control over programs used to access the Corporate databases. One possible downside to X-terminal technology may be an unacceptable stress placed on the network.

VersaTerm-PRO (for Apple Macintosh)

Macintosh users can access Oracle through VersaTerm-PRO's VT220 emulator. The software takes you through several setup screens. The AViiON's (av) TCP/IP address is 192.104.43.110. In the keyboard settings screen set <RETURN> to New Line, the Delete key to <BACKSPACE>, highlight the Numeric option for the DEC VT100 keypad, and DEC VT220 keyboard as extended keyboard option. In the Extras screen under the Settings menu the following options should be set on:

Text Emulation Options:

vt220/7-bit, Auto DEC vt220 Entry, Auto Horizontal Scroll, Auto Wraparound, Add NL after CR, Return is New Line, Standard Tab Stops, Ignore vt100 Answerback.

General Options:

Multifinder "Auto Zoom", Enable Sounds, Enable Remote File Access, Enable DA's while Printing.

Graphics Emulation Options:

High Resolution PICT's.

There are also other options you can set according to your needs, such as the shape of the cursor, the format of text files created, word wrapping at a particular column, prompt character, and printer type. The available options are somewhat different in newer versions of the software. With a bit of trial and error, and consultation with other Mac users you will find the most appropriate settings.

ORACLE TABLESPACES

Both Oracle environments use the same tablespace naming convention. These are: TBSPA, TBSPB, TBSPC, INDXA, INDXB and INDXC. All six tablespaces are accessible to you (in Oracle Test environment).

Only the table data should reside in tablespace TBSP(A/B/C). The table's indexes should reside in INDX(A/B/C) respectively. That is if table FRED resides in TBSPB, then all of FRED's indexes should reside in INDXB, etc.

All userids have a default tablespace assigned to them, which may be TBSP(A/B/C).

Note that when creating tables, the storage for the initial and next extent is in bytes, not blocks as in version 5. For example:

```
create table FRED
    (   fredkey         number not null primary key,
        etc1            char(5),
        ....            .....
        lastetc         char(10)   )
tablespace tbspb
storage (initial 10240 next 2048 pctincrease 0);
```

will create the table in tablespace TBSPB, with the initial extent of 10,240 bytes (10KB) and next extent of 2048 bytes (2KB). Note you can also use the suffix 'M' or 'm' to indicate Megabytes, and 'K' or 'k' to indicate Kilobytes. If no suffix is provided, bytes is assumed.

The same storage parameters also apply to indexes.

COMMON AND USEFUL DICTIONARY VIEWS

Oracle have provided many views that can be interrogated to find out information about database objects from the Oracle dictionary.

Starting in Oracle Version 6, the dictionary views have been changed to have better naming consistency. The view names themselves have also changed.

Generally, the dictionary views usually have a prefix of either 'USER_' or 'ALL_' . The views with the prefix of 'USER_' will obtain information from the dictionary about objects that are owned (ie created) by you. The views with the prefix of 'ALL_' will obtain information about all objects to which you have access.

Below are some of the more useful views that are available to you, along with a brief description of each.

ALL_INDEXES	description of indexes on all tables accessible to user
ALL_IND_COLUMNS	columns of the indexes on all tables accessible to user
ALL_SYNONYMS	all synonyms accessible to the user
ALL_TAB_COMMENTS	comments on all tables and views accessible to the user
ALL_TABLES	description of all tables accessible to the user
ALL_USERS	information about all users of the database
ALL_VIEWS	description of all views accessible to the user. Note that the field TEXT is of datatype long, which defaults to a length of 80 characters. To see more of the view's text, issue: *set long 3000* in this case, up to 3000 characters of the view text will be

USER_INDEXES	description of the user's own indexes
USER_IND_COLUMNS	columns of the user's indexes on user's tables
USER_OBJECTS	objects owned by user
USER_SEQUENCES	description of the user's own sequences
USER_SYNONYMS	user's private synonyms
USER_TAB_COMMENTS	comments on the tables and views owned by the user
USER_TAB_GRANTS	grants on objects for which the user is the owner, grantor, or grantee
USER_TAB_GRANTS_MADE	all grants on objects owned by the user
USER_TAB_GRANTS_RECD	grants on objects for which the user is the grantee
USER_TABLES	description of user's own tables
USER_VIEWS	text of views owned by the user. Note that the field TEXT is of datatype long, which defaults to a length of 80 characters. To see more of the view's text, issue: *set long 3000* in this case, up to 3000 characters of the view text will be retrieved

Full description of these views can be found in Appendix A.

Another useful view is:

DICTIONARY	description of data dictionary tables and views. This view lists all the Oracle dictionary views and their description.

ORACLE and UNIX PATHS

Generally, Oracle tools ignore the PATH Unix parameter. Each Oracle product uses its own special environment parameter as described below. An exception to the case is SQL*Menu, which uses a different system to locate and execute Menus.

When you type *setoratest* or *setoraprod*, Unix parameters special to Oracle are automatically created for you. They are:

Oracle Path	Used by
ORACLE_PATH	user SQL*Forms
SQLPATH	user SQL*Plus scripts

For more information regarding a specific product, see below.

SQL*Forms and ORACLE_PATH

ORACLE_PATH, once set in the environment, instructs Oracle to execute SQL*Forms residing in other directories to the current working directory. To execute Forms located in say

/home/fred/prod/forms

from any directory on the AViiON, you need to append this directory to ORACLE_PATH and then export it (using the Bourne Shell):

ORACLE_PATH=$ORACLE_PATH:/home/fred/prod/forms
export ORACLE_PATH

Oracle will then execute forms found in your current working directory and in the

/home/fred/prod/forms

directory.

ORACLE_PATH commands (there may be several of them, as the forms comprising the whole application may reside in several different directories) are usually contained in the Bourne shell script to run the main menus of the various databases/systems (for an example, see Figure 3).

SQL*Plus and SQLPATH

SQLPATH, once set in the environment, instructs Oracle to execute SQL*Plus scripts residing in other directories to the current working directory. This allows you to start/run SQL scripts within an SQL*Plus session from any directory.

For example, using the Bourne Shell:

SQLPATH=$SQLPATH:/home/fred/prod/sql
export SQLPATH

SQL*Menu

SQL*Menu does not use path variables like SQL*Forms and SQL*Plus. SQL*Menu uses information stored in the Oracle database itself.

The information is entered in SQLMENU50 by

Menu
Application
Directory: This is where you type in the directory where the *XXX*.dmm file resides

This means that the menu can be called from any directory

ie *runmenu50 menuname* (assuming the device type is correctly specified in TERM)

SQR

SQR (a third party product distributed by Sequel Information Management) does not use any path variables. You have to specify the full pathname of the program you wish SQR to execute.

ie *sqr /home/minloc/prod/sqr/sqrprog*

where *sqrprog* is the program residing in /home/minloc/prod/sqr directory.

Alternatively you can define your own variables to make the pathname more flexible.

ie *SQRMINLOC=/home/minloc/prod/sqr*
export SQRMINLOC

sqr $SQRMINLOC/sqrprog

in the case of compiled SQR programs,

sqr $SQRMINLOC/sqrprog -RT

Note that the 'RT' parameter has to be upper case if invoking SQR from under SQL*Menu, as command type 2.

SQL*Plus

Before you can invoke SQL*Plus, you need to run *setoratest* or *setoraprod*. Once the Oracle environment is set, type:

sqlplus

The SQL*Plus prompt will indicate which Oracle environment you are running against (unless running as a client) -

	ORAPROD	ORATEST
prompt =	SQL-Prod>	SQL-Test>

The command to terminate an SQL*Plus session, is *exit* or *quit* (not bye).

SQL*Forms

The Oracle environment (Test or Production) has to be set before you can execute SQL*Forms.

The current Corporate Oracle RDBMS runs SQL*Forms Version 3. All Forms should now be either native Version 3, or Version 3 using 2.3 triggers (converted from 2.3 to 3 with the 'convert30' utility).

In SQL*Forms, there are two different 'sets' of Forms, namely SQL*Forms Designer and SQL*Forms Runtime.

The commands to invoke SQL*Forms Version 3 are :

runform30 (preferred command)	Runtime Form
iap30	
sqlforms30 (preferred command)	
iad30	Form design
design30	

There are two different ways of invoking Forms, depending on whether you wish to use the device type specified in TERM, or specifying a different one. ie -

1 *runform30 <form name>*
2 *runform30 -c <terminal device> <form name>*

(the syntax is the same with the *sqlforms30* command)

In the first syntax, SQL*Forms uses the TERM environment parameter to decide which device type you are using. This will be the most common method of accessing Oracle Forms.

The second syntax is an alternate method of invoking Forms, however the terminal device has to be one of the device types supported (see Terminal Emulation).

Forms should not be *saved* to the database, and should be *generated* only. This will create two files, '*xxx*.inp' file containing the source of the Form, and an '*xxx*.frm' file which is the executable.

SQL*Menu

The Oracle environment (Test or Production) has to be set before you can execute SQL*Menu.

In SQL*Menu, as in SQL*Forms, there are two different 'sets' of Menu, SQL*Menu Designer, and SQL*Menu Runtime.

The commands to invoke SQL*Menu are:

runmenu50	Runtime Menu
sqlmenu50	Design Menu

As with Forms, there are two ways of invoking SQL*Menu, depending on whether you wish to use the TERM environment parameter or not.

1 *runmenu50 <menu name>*

2 *runmenu50 -c <terminal device> <menu name>*

(the syntax is the same with the *sqlmenu50* command)

Moving SQL*Menu applications

Moving SQL*Menu applications from one Oracle environment to another is achieved through the UNLOAD option in SQL*Menu. This option creates a SQL script which then has to be run in the environment you want to shift the Menu application to. This is done from SQL*Plus, using the creation/owner userid.

Once the SQL script has been run, run SQLMENU50, and LOAD the application. You may also have to change the directory path to reflect the new residence of the '*XXX*.dmm' file.

ie *Menu*
Application
Directory: This is where you type in the directory where the '*XXX*.dmm' file resides

GENERATE and SAVE in the normal manner.

SQL*Menu security

Two Menu security roles have been provided:

INTERNAL - all Oracle AGSO userids belong to this role
EXTERNAL - all future clients external to AGSO will belong to this role

Additionally, each system has a security role based on the system name followed by '_role' suffix. The system userid is automatically included as a member of this role - ie system MURBO has a role called MURBO_ROLE, of which the MURBO userid is a member.

It is up to the system developer to decide on the level of access to his/her Menu application. Generally speaking, there are two major access levels to Menu applications

- restricted to system and users of system only — only grant the system role to the main menu. Only users belonging to this role will be able to use the Menu application.
- restricted to system and all AGSO users — grant the system role as well as the role INTERNAL to the main menu .

There are many other ways of restricting access, all performed on the Grant role in the Item screen. One can request additional roles from the DBA to make the security more granular.

SETTING UP GLOBAL APPLICATIONS

It may be very desirable to make an application/system globally known, so that it can be invoked by anyone from anywhere on the AViiON.

This is achieved through the use of **symbolic links** in the directory -
/usr/local/bin
a directory which is included in every userid's PATH statement on the AViiON.

The link points to and executes a shell script in the application's directory, which usually invokes the application, after first setting the Unix and Oracle environments.

The symbolic link is owned by root, and only updatable by the root userid. This is a flexible arrangement in the sense that the shell script can be changed or modified by the system owner as the system develops, but the name of the link remains the same. Therefore the ongoing modifications are transparent to the invoking user.

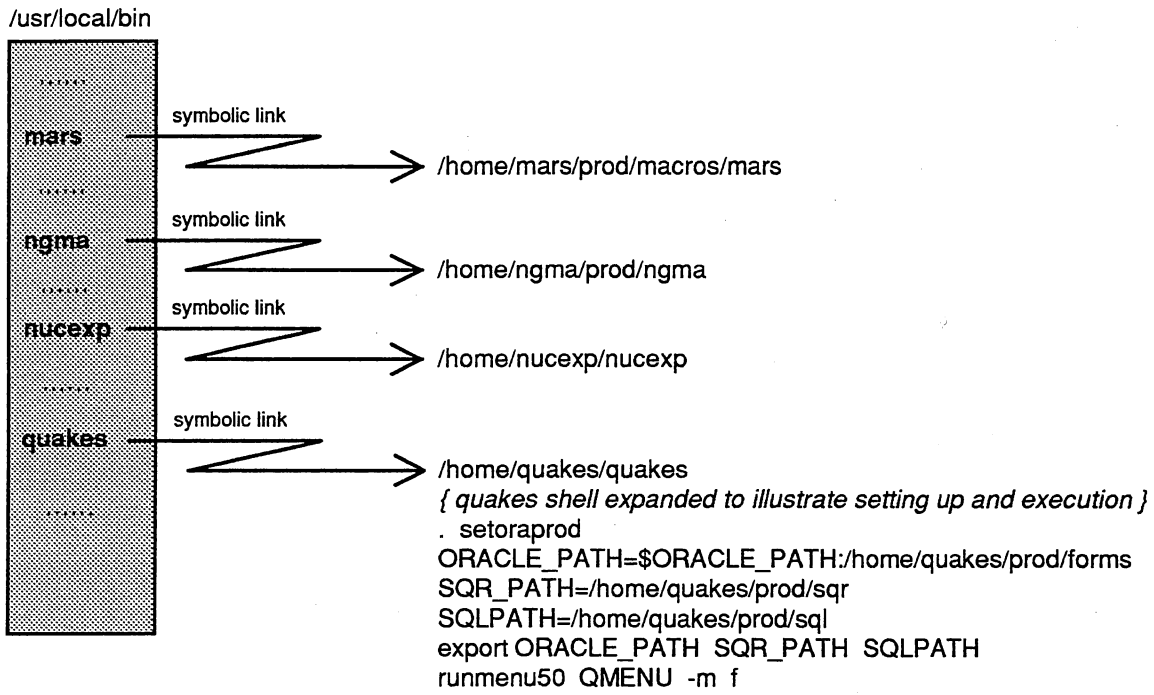

Figure 3: Graphical representation of symbolic links

The usual security considerations still apply - the invoking user must have been granted all the appropriate Oracle security permissions, in order to successfully run the application.

SQL*Net

SQL*Net (TCP/IP protocol) on the AViiON is a connectivity product that allows applications to access data from other Oracle instances, or more appropriately in AGSO - to access data in one Oracle environment from the other (for example, access the Oracle Production data from the Test environment). This data access is achieved through **database links**.

SQL*Net can also be used to access Oracle by other third party products such as Arc/Info.

Public Database Links

A publicly accessible database link has been provided in both Oracle Production and Test, to enable access to the other database.

These links are called 'oratest' in Oracle Production, and 'oraprod' in Oracle Test.

For example, to access data from Oracle Production when logged on to Test, enter

select .... from fred.table1@oraprod
where .....

This will retrieve data from table fred.table1 in Production.

You can create your own synonyms to make this even more transparent, ie

create synonym fredtab1 for fred.table1@oraprod;

so then the above query could also be written as:

select ..... from fredtab1
where .....

For convenience, your Oracle userid and password are automatically passed to the remote database. Therefore if you are planning to use SQL*Net in this fashion, your userids and passwords should be the same on both Oracle Production and Test.

Private Database Links

Sometimes, there is a requirement to access data from the remote database using a different userid and/or password to the ones you are currently using. Since the public database links 'oraprod' or 'oratest' automatically pass your current userid and password to the other database, you will need to create your own (private) database link specifying a different userid and password combination for the other (remote) database environment.

This approach is not as 'neat and tidy' as using the publicly accessible link provided (which should be used whenever possible). At this stage there appears to be no way of creating a dynamic database link, which prompts you for the username and password on the remote database as soon as the link is invoked.

When creating a private database link please remember that only you can see and use the database link you created. You cannot give other users access to your database link.

The SQL syntax for creating a private database link is:

create database link <linkname> connect to <username> identified by <password>
using '<connect string>';

where <linkname> is the name of the link you wish to create
<username> and <password> are the userid and password that exist on the remote database you wish to access
<connect string> is the connect id of the remote database. The string is:
'T:av:oraprod' for the Oracle Production environment

and 'T:av:oratest' for the Oracle Test environment

ie *create database link scottlink connect to scott identified by tiger using 'T:av:oraprod';*

will create a private link called scottlink using the username and password of scott/tiger on the remote database ORAPROD. This assumes that you are currently in the Test environment. Your username and password could be totally different on the Test environment, because you are supplying scott/tiger as the username/password combination for the Production environment. Needless to say it has to be a valid username/password combination.

For example,

select .... from fred.table1@scottlink
where .....

This will retrieve data from table fred.table1 in Production, logged on as scott/tiger in Production.

BASIC SURVIVAL GUIDE TO 'vi'

'vi' is a full screen, command driven editor that is standard across most Unix machines. Even though 'vi' is a powerful editor, it is initially hard to use. This one page summary briefly describes some of the basic commands for 'vi' that are useful when editing files.

To invoke 'vi' type:

vi *<filename>*	where *<filename>* is the name of the file you wish to edit

'vi' automaticallly starts in 'command' mode, which means you can issue editing commands. Two very important things to remember in 'vi' are:

:q!	quits 'vi' without writing to the file
escape key	puts 'vi' back in 'command' mode

The basic command set for 'vi' is (*note 'vi' commands are case sensitive*):

Screen navigation

^ (up arrow) or k	move the cursor up one line
v (down arrow) or j	move the cursor down one line
< (left arrow) or h	move the cursor left one character
> (right arrow) or l	move the cursor right one character
w	advances cursor by one word
^f (control f)	moves screen down one page
^b (control b)	moves screen up one page
$	moves cursor to end of line

Editing

a (see note 1)	append text after the cursor
i (see note 1)	insert text before the cursor
cw (see note 1)	replace the current word with new text
dw	deletes the current word
o	opens a line for input below the current line
O	opens a line for input above the current line
J	join next line to line with cursor
r	replace single character under the cursor
x	delete single character under the cursor
dd	delete the current line (note that 4dd deletes 4 lines etc.)

Searching

/*text*	search for the *text* string
n	repeat the last search

Substitutions

:%s/a/b/g	substitute 'b' for 'a' in the whole file
:s/a/b/g	substitute 'b' for 'a' in current line

Miscellaneous

u	undo the last change
G	go to the last line of file
*n*G	go to line '*n*' of the file (ie 4G will go to the fourth line of the file)
ZZ	write and save file and exit
:q!	quit without writing the file
:r filename	inserts text from filename starting below cursor

1 These commands put the editor in 'insert' mode, and must be terminated by pressing the ESCAPE key to put 'vi' back into 'command' mode.

ACKNOWLEDGEMENTS

The compilation of this record is the result of forward planning and practical experience. Many people contributed to this record: Rob DeNardi, Sonja Lenz, Keith Porritt, Rod Ryburn, Andrew Tucker, David Walton and others.

This Record has benefited greatly from reviews by Sonja Lenz, Rod Ryburn and Andrew Tucker.

REFERENCES

Kucka, M., 1992, Dual Oracle Database Environment and Change Control Management. *Australian Geological Survey Organisation, Record* 1992/85.

Lenz, S.L., Ryburn, R.J., & Kucka, M., 1993 - Users' Guide to AGSO's Oracle Database System. *Australian Geological Survey Organisation, Australia, Record* 1993/81.

BIBLIOGRAPHY

Oracle User Guides and Reference Manuals:

ORACLE RDBMS V.6.0 Database Administrator's Guide, *Oracle Corporation.*

ORACLE V.6.0 Utilities User's Guide, *Oracle Corporation.*

PL/SQL User's Guide and Reference V.1.0, *Oracle Corporation.*

SQL*Forms V.3.0 Designer's Reference, *Oracle Corporation.*

SQL*Forms V.3.0 Designer's Tutorial, *Oracle Corporation.*

SQL*Plus V.3.0 User's Guide and Reference, *Oracle Corporation.*

SQL Language Reference Manual V.6.0. *Oracle Corporation 1990.*

SQL*Menu User's Guide and Reference V.5.0, *Oracle Corporation.*

SQL*Net TCP/IP User's Guide V.1.2, *Oracle Corporation.*

SQL*Report User's Guide V.1.1, *Oracle Corporation.*

SQR Structured Query Report Writer User Guide, 1990, *distributed by Sequel Information Management Pty Ltd.*

SunOS Documentation Tools , *SUN Microsystems.* (For more information on using the 'vi' editor)

Appendix A: Description of common Data Dictionary Views

(From ORACLE RDBMS V.6.0 Database Administrator's Guide, Oracle Corporation.)

ALL_INDEXES	**Description of indexes on tables accessible to the user**
OWNER	Username of the owner of the index
INDEX_NAME	Name of the index
TABLE_OWNER	Owner of the indexed object
TABLE_NAME	Name of the indexed object
TABLE_TYPE	Type of the indexed object
UNIQUENESS	Uniqueness status of the index: "UNIQUE" or "NONUNIQUE"
TABLESPACE_NAME	Name of the tablespace containing the index
INI_TRANS	Initial number of transactions
MAX_TRANS	Maximum number of transactions
INITIAL_EXTENT	Size of the initial extent
NEXT_EXTENT	Size of secondary extents
MIN_EXTENTS	Minimum number of extents allowed in the segment
MAX_EXTENTS	Maximum number of extents allowed in the segment
PCT_INCREASE	Percentage increase in extent size
PCT_FREE	Minimum percentage of free space in a block

ALL_IND_COLUMNS	**Columns of the indexes on accessible tables**
INDEX_OWNER	Index owner
INDEX_NAME	Index name
TABLE_OWNER	Table or cluster owner
TABLE_NAME	Table or cluster name
COLUMN_NAME	Column name
COLUMN_POSITION	Position of column within index
COLUMN_LENGTH	Indexed length of the column

ALL_SYNONYMS	**All synonyms accessible to the user**
OWNER	Owner of the synonym
SYNONYM_NAME	Name of the synonym
TABLE_OWNER	Owner of the object referenced by the synonym
TABLE_NAME	Name of the object referenced by the synonym
DB_LINK	Name of the database link referenced, if any

ALL_TAB_COMMENTS	**Comments on tables and views accessible to the user**
OWNER	Owner of the object
TABLE_NAME	Name of the object
TABLE_TYPE	Type of the object
COMMENTS	Comment on the object

ALL_TABLES	**Description of tables accessible to the user**
OWNER	Owner of the table
TABLE_NAME	Name of the table
TABLESPACE_NAME	Name of the tablespace containing the table
CLUSTER_NAME	Name of the cluster, if any, to which the table belongs
PCT_FREE	Minimum percentage of free space in a block
PCT_USED	Minimum percentage of used space in a block
INI_TRANS	Initial number of transactions
MAX_TRANS	Maximum number of transactions
INITIAL_EXTENT	Size of the initial extent in bytes
NEXT_EXTENT	Size of secondary extents in bytes
MIN_EXTENTS	Minimum number of extents allowed in the segment
MAX_EXTENTS	Maximum number of extents allowed in the segment
PCT_INCREASE	Percentage increase in extent sizc
BACKED_UP	Has table been backed up since last modification

ALL_USERS	**Information about all users of the database**
USERNAME	Name of the user
USER_ID	ID number of the user
CREATED	User creation date
ALL_VIEWS	**Text of views accessible to the user**
OWNER	Owner of the view
VIEW_NAME	Name of the view
TEXT_LENGTH	Length of the view text
TEXT	View text
USER_INDEXES	**Description of the user's own indexes**
INDEX_NAME	Name of the index
TABLE_OWNER	Owner of the indexed object
TABLE_NAME	Name of the indexed object
TABLE_TYPE	Type of the indexed object
UNIQUENESS	Uniqueness status of the indcx: "UNIQUE" or "NONUNIQUE"
TABLESPACE_NAME	Name of the tablespace containing the index
INI_TRANS	Initial number of transactions
MAX _TRANS	Maximum number of transactions
INITIAL_EXTENT	Size of the initial extent in bytes
NEXT_EXTENT	Size of secondary extents in bytes
MIN_EXTENTS	Minimum number of extents allowed in the segment
MAX_EXTENTS	Maximum number of extents allowed in the segment
PCT_INCREASE	Percentage increase in extent size
PCT_FREE	Minimum percentage of free space in a block
USER_IND_COLUMNS	**Columns of the user's indexes or on user's tables**
INDEX_NAME	Index name
TABLE_NAME	Table or cluster name
COLUMN_NAME	Column name
COLUMN_POSITION	Position of column within index
COLUMN_LENGTH	Indexed length of the column
USER_OBJECTS	**Objects owned by the user**
OBJECT_NAME	Name of the object
OBJECT_ID	Object number of the object
OBJECT_TYPE	Type of the object
CREATED	Timestamp for the creation of the object
MODIFIED	Timestamp for the last DDL change to the object
USER_SEQUENCES	**Description of the user's own sequences**
SEQUENCE_NAME	SEQUENCE name
MIN_VALUE	Minimum value of the sequence
MAX_VALUE	Maximum value of the sequence
INCREMENT_BY	Value by which sequence is incremented
CYCLE_FLAG	Does sequence wraparound on reaching limit
ORDER_FLAG	Are sequence numbers generated in order
CACHE_SIZE	Number of sequence numbers to cache
LAST_NUMBER	Last sequence number written to disk
USER_SYNONYMS	**The user's private synonyms**
SYNONYM_NAME	Name of the synonyrn
TABLE_OWNER	Owner of the object referenced by the synonyrn
TABLE_NAME	Name of the object referenced by the synonyrn
DB_LINK	Database link referenced in a remote synonyrn

USER_TAB_COMMENTS	**Comments on the tables and views owned by the user**
TABLE_NAME	Name of the object
TABLE_TYPE	Type of the object: "TABLE" or "'VIEW"
COMMENTS	Comment on the object
USER_TAB_GRANTS_MADE	**All grants on objects owned by the user**
GRANTEE	Name of the user to whom access was granted
TABLE_NAME	Name of the object
GRANTOR	Name of the user who performed the grant
SELECT_PRIV	Permission to SELECT from the object
INSERT_PRIV	Permission to INSERT into the object
DELETE_PRIV	Permission to DELETE from the object
UPDATE_PRIV	Permission to UPDATE the object
REFERENCES_PRIV	Permission to make REFERENCES to the object
ALTER_PRIV	Permission to ALTER the object
INDEX_PRIV	Permission to CREATE/DROP INDEX on the object
CREATED	Timestamp for the grant
USER_TAB_GRANTS_RECD	**Grants on objects for which the user is the grantee**
OWNER	Columns of user's tables, views, and Owner of the object
TABLE_NAME	Name of the object
GRANTOR	Name of the user who ~erformed the grant
SELECT_PRIV	Permission to SELECT from the object
INSERT_PRIV	Permission to INSERT into the object
DELETE_PRIV	Permission to DELETE from the object
UPDATE_PRIV	Permission to UPDATE the object
REFERENCES_PRIV	Permission to make REFERENCES to the object
ALTER_PRIV	Permission to ALTER the object
INDEX_PRIV	Permission to create/drop an INDEX on the object
CREATED	Timestamp for the grant
USER_TABLES	**Description of the user's own tables**
TABLE_NAME	Name of the table
TABLESPACE_NAME	Name of the tablespace containing the table
CLUSTER_NAME	Name of the cluster, if any, to which the table belongs
PCT_FREE	Minimum percentage of free space in a block
PCT_USED	Minimum percentage of used space in a block
INI_TRANS	Initial number of transactions
MAX_TRANS	Maximum number of transactions
INITIAL_EXTENT	Size of the initial extent in bytes
NEXT EXTENT	Size of secondary extents in bytes
MIN_EXTENTS	Minimum number of extents allowed in the segment
MAX_EXTENTS	Maximum number of extents allowed in the segment
PCT_INCREASE	Percentage increase in extent size
BACKED_UP	Has table been backed up since last modification
USER_VIEWS	**Text of views owned by the user**
VIEW_NAME	Name of the view
TEXT_LENGTH	Lengrth of the view text
TEXT	View text