Oracle Tools and Utilities, System and Oracle Environment Variables, Oracle Registry Parameters and Their Relations

Радослав Русинов
Radoslav.Rusinov@dir.bg

Версия 2.3

Въведение

Предназначението на този документ е да се опита да обясни каква е връзката между системните обкръжаващи променливи, Registry параметрите и техните стойности (свързани по някакъв начин с Oracle) – какво е тяхното предназначение, как те оказват влияние на Oracle Tools и Utilities, на Oracle като база данни, и как всеки потребител на базата данни може да ги използва в ежедневната си работа. Тъй като тук се разглеждат връзките на Oracle с Windows Registry, може да се приеме че той важи само за Windows операционна система.

Понятия

Въвеждането на определени понятия е продиктувано от факта, че всеки специалист, използващ ежедневно компютър в работата си употребява много термини, които са или непреводими на български или в точен превод предизвикват объркване. Поради тази причина всички такива понятия тук са на английски език (на езика, на които те се употребяват най-често в ежедневната работа). Използването на термини без да са преведени на български, както и въвеждането на нови, важат само в рамките на този документ и се въвеждат единствено с цел по-лесното четене и използване на материала, както и за пълнота.

- Windows Registry (или Registry) – Registry е главното хранилище (repository) за Windows като операционна система. В него освен, че се съхраняват важни параметри за нормалното функциониране и поведение на операционната система, се съхраняват и параметри за апликациите, работещи върху нея.
- Registry Ключ (или Registry Подключ) – Структурата на Registry е дървовидна, като всеки ключ може да има свой подключ, както и множество registry параметри, подчинени на него
- Registry Параметър – всеки такъв параметър има име, тип и стойност, и е подчинен на определен registry ключ
- Windows Service – една от главните функции на един такъв Service е възможността му да стартира една апликация във фонов режим (background), без да се налага потребителя на операционната система да го прави. Всеки един такъв Service си има собствен процес
- Oracle Service – това е Windows Service, но който стартира апликация, имаща пряко отношение към работата на Oracle
- Oracle Tool – SQL*Plus, Oracle Enterprise Manager, Oracle Universal Installer, Oracle Home Selector
- Oracle Utility – SQL*Loader, Export, Import, ORADIM
- Oracle Home – директорията, в която се намира софтуера, нужен за работата на Oracle Tools и Utilities, както и на Oracle като база данни.

Когато се налага използването на допълнителни понятия, в скоби ще бъде изписвано тяхното наименование на английски.

Multiple Oracle Homes

Един Oracle Home е свързан с работещият под него Oracle софтуер и с операционната система. Компонентите, които зависят от даден Oracle Home са:

- Местоположението на инсталирания Oracle софтуер
- Обкръжаващата променлива PATH, указваща пътя до изпълнимите файлове за инсталирания Oracle софтуер
- Registry ключове
- Имена на Services

- Групи програми (Start -> Programs -> Oracle – OraHome92)

Докато под Unix винаги е съществувала възможността на един компютър да се инсталират няколко Oracle Home, то за Windows това става едва с излизането на Oracle версия 8.0.4.
С помощта на тази възможност на един компютър могат да бъдат инсталирани и да работят независимо една от друга различни версии на Oracle база данни, както и на Oracle Tool и Utilities. Например, на един компютър може да има инсталиран софтуер за Oracle база данни версии 8.1.7, 9.0.1, 9.2.0.5, и за Management Server, всеки в своя собствена директория или така наречения Oracle Home. Всеки Oracle Home има собствено име, асоциирано с пътя до него. За повече информация [изт.17 и 3]
Инсталирането и поддържането на няколко Oracle Homes е много често срещано в практиката и в същото време се оказва в основата на много често срещани проблеми.

Въвеждане на Нови Понятия

Системни и Oracle Обкръжаващи Променливи

В документацията на Oracle навсякъде се използва понятието "обкръжаващи променливи" (environment variables), но в практиката това предизвиква объркване поради разминаванията между очакваните области на видимост, начини за настройка и за достъп до тези променливи. Винаги, когато в документацията се говори за обкръжаваща променлива, потребителя я възприема в термините на операционната система, а в същото време начина по който всички Oracle Tools и Utilities ги използват в определени случаи се различава от стандартните методи, с които той е свикнал да работи на ниво операционна система. Затова в този документ всички обкръжаващи променливи, които от Oracle наричат "обкръжаващи променливи" ще бъдат наричани "Oracle обкръжаващи променливи", а тези на ниво операционна система ще се наричат "системни обкръжаващи променливи" (това понятие често се среща в практиката като синоним на "обкръжаващи променливи" на ниво ОС).
За да се установи дали има разлика между тях и каква е тя, трябва да се разгледа поведението на една системна обкръжаваща променлива. Тя може да се настройва на ниво операционна система, което позволява да бъде използвана от всички апликации, работещи под Windows. Техните стойностти могат да бъдат настройвани от "Control Panel -> System -> Advanced", като биха могли да бъдат достъпни единствено за конкретния ОС потребител или за всички потребители. Системните обкръжаващи променливи могат да бъдат настройвани с помощта на DOS командата SET за конкретен команден прозорец (command window), като нейната област на действие е локална и важи само за апликациите, стартирани от този прозорец. Когато променлива с едно и също име, например PATH е настроена на няколко места, то се взима стойността на тази с най-голям приоритет. Приоритетът им се определя по следния начин - с най-малък е променливата, настроена за всички ОС потребители, след това тази за конкретния потребител и с най-голям приоритет е ръчно настроената в конретния команден прозорец.
Oracle обкръжаващата променлива има същите характеристики като системната, с тази разлика че нейната стойност може да бъде настроена и в Registry (и то на точно определено място), както и в това че съответно се появява и нов приоритет за определяне на стойността им в случай на дублиращи се настройки. Всеки Oracle Tool и Utility, както и Oracle като база данни имат достъп до тези променливи (както имат и до системните), и ги използва по различни начини – за да изчисли точното местоположение до определен файл, да определи поведението и режима си на работа и т.н. Приоритетът се определя по следния начин: с най-малък приоритет е стойността, настроена на ниво операционна система, след това на ниво потребител, след това на ниво Registry и с най-голям приоритет отново е настроената в командния прозорец. Объркването на потребителя се получава, когато в документацията започне да среща изискванията на Oracle за използването на тези променливи, като например това - в никакъв случай да не се настройват на ниво операционна система (или в краен случай), въпреки че е възможно. Т.е. може да се прави нещо, но не трябва да се прави. Може да се направи, защото по същество всяка такава променлива е системна променлива, а не бива, защото Oracle очаква да ги намери на друго място, откъдето лесно могат да бъдат контролирани и пренастройвани, а именно в Windows Registry.
Потребителят приема изискването и започва всеки път, когато в документацията му споменават за обкръжаващата променлива ORACLE_HOME например, да си напомня, че всъщност това не е обкръжаваща променлива, а малко по-различна и трябва да я настройва само в Registry, но и да не забравя че когато се наложи може да я настрои в команден прозорец, както и в никакъв случай да не я настройва на ниво ОС.

Oracle Registry Параметри

Объркването става още по-голямо, когато след малко тестове се установява, че два различни параметъра, настроени в Registry, имат различно поведение – единият може да бъде настроен и на ниво операционна система, а другият се

взима предвид само ако е настроен в Registry, независимо дали съществува настроен като системна обкръжаваща променлива. Въпросите вече са няколко: какъв се явява втория и какъв първия параметър, коя променлива да се нарича Oracle обкръжаваща променлива и коя да се нарича "такава, която се настройва в Registry", защо и кога да се настройват в команден прозорец, как да се разбере коя може да се настрои по този начин, и коя има съвсем друго предназначение и различен начин на достъп от Oracle, за какво са изобщо системните обкръжаващи променливи при работата с Oracle.
Навсякъде понятията Registry, обкръжаваща променлива, настройване в команден прозорец, настройване на ниво операционна система така се дублират и самоотричат, че в един момент настъпва объркване.
За може да се открие някаква логика и закономерност, както и заради по-лесното четене, тук може да се въведе понятието Oracle Registry Параметър. Един такъв параметър се намира на точно определено място в Registry (HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\HOMEID) и може да бъде настройван единствено оттук.
Тестовете показват, че главно Oracle като база данни и ORADIM Utility използват параметри с такова поведение.

Външни Параметри

Обединението от двете множества: на всички Oracle обкръжаващи променливи и Oracle Registry параметри определя списъка с всички параметри, които могат да влияят върху работата на базата данни и на Oracle Tools и Utilities, и се нарича множеството на всички Външни параметри. Тези параметри нямат нищо общо с инициализационните параметри на базата или настройките, които се създават и поддържат в конфигурационни файлове - те са изкуствено създаден термин в рамките на този документ.
За да не станат тези нови понятия причина за нови объркания, тяхното поведение и предназначение може да бъде обобщено така:

- Параметрите, които по принцип се настройват в Registry, но когато се наложи могат да бъдат настройвани локално в команден прозорец, се наричат Oracle обкръжаващи променливи
- Параметрите, които могат да бъдат настройвани единствено в Registry, се наричат Oracle Registry параметри

За по-голяма яснота по-надолу при изброяването на всички такива параметри, ще бъде уточнявано какво е тяхното поведение – дали на Oracle обкръжаващи променливи, или на Oracle Registry параметри.
Някои от тези параметри (ORACLE_HOME например) трябва задължително да имат стойност, като за това обикновено се грижи Oracle Universal Installer и няма нужда от никаква допълнителна намеса от страна на потребителя. За всички останали има стойност по подразбиране и независимо, че конкретния параметър може да не е дефиниран никъде, то Oracle използва стойността му по подразбиране.

Кратки Дефиниции

- Системна Обкръжаваща Променлива (System Environment Variable) – променлива, чиято стойност може да бъдат четена от апликациите, работещи под Windows, както и от самата операционна система
- Oracle Обкръжаваща Променлива (Oracle Environment Variable) – променлива, която е достъпна за всички Oracle Tools, Utilities, както и за Oracle като база данни
- Oracle Registry Параметър – Registry параметър, имащ пряко отношение към Oracle Tools и Utilities, както и към Oracle като база данни
- Външен Параметър – всеки параметър, който настроен в Registry или като обкръжаваща променлива, може да окаже влияние на работата на Oracle Tool и Utilities, както и на самата база

Практически Проблеми

Настройка на ORACLE_BASE

Тази Oracle обкръжаваща променлива указва най-горната директория като ниво, в която би трябвало да се намират всички файлове имащи нещо общо с Oracle. Тя най-общо изглежда така: X:\oracle, където X може да бъде всеки дял на твърдия диск, а стойността по подразбиране е C:\oracle. Според стандартите на OFA (Optimal Flexible Architecture) тази директория съдържа в себе си следните поддиректории [изт. 3]:

- Всички ORACLE_HOME директории
- Всички oradata директории - за файловете с данните (data files)
- Всички admin директории - за файловете за администрация на базата

Тази променлива може да улесни потребителя, когато се използва в автоматични скриптове, извършващи промени в базата. При инсталирането на Oracle Home, който ще се използва за Oracle база данни, тази променлива се настройва автоматично от Oracle Universal Installer в Registry, но само за този Home. При следващите инсталации на други Oracle Homes, тя се настройва автоматично само при инсталирането на нови Oracle бази данни. За да може да бъде достъпвана от изпълнимите файлове от всички Oracle Homes, независимо дали съдържат Oracle Database софтуер, може тя да се настрои на ниво операционна система, или да се добави ръчно за всеки от останалите Homes в Registry. Добавянето на ниво операционна система не е подходящо в случаи, че ще има такава директория на повече от един дял на компютъра или пък ще има 2 или повече, напр. D:\oracle и D:\oracle_base. След като веднъж е настроена и има инсталирана и работеща база върху дадения Oracle Home, то тя не би трябвало да се променя. Промяната и в такъв момент ще е причина някои от Oracle Tools и Utilities (които се опитват да я използват при изпълнение на автоматични скриптове) и самата Oracle като база данни да не успеят да си намерят необходимите им файлове, ако тяхното местоположение се изчислява от ORACLE_BASE. Това са и всички пътища които се намират под нивото на ORACLE_BASE, но не в ORACLE_HOME, като например admin и oradata директориите. По принцип този параметър би трябвало да се използва само от базата, и то в определени случаи. При автоматичното създаване на нова база с помощта на шаблон (template), този параметър също се използва.

НАСТРОЙКА НА ORACLE_HOME

ORACLE_HOME е Oracle обкръжаваща променлива и указва местоположението на даден Oracle Home.
Често в практиката при опит да се стартира SQL*Plus, Export или Import от команден прозорец, например SQL*Plus:

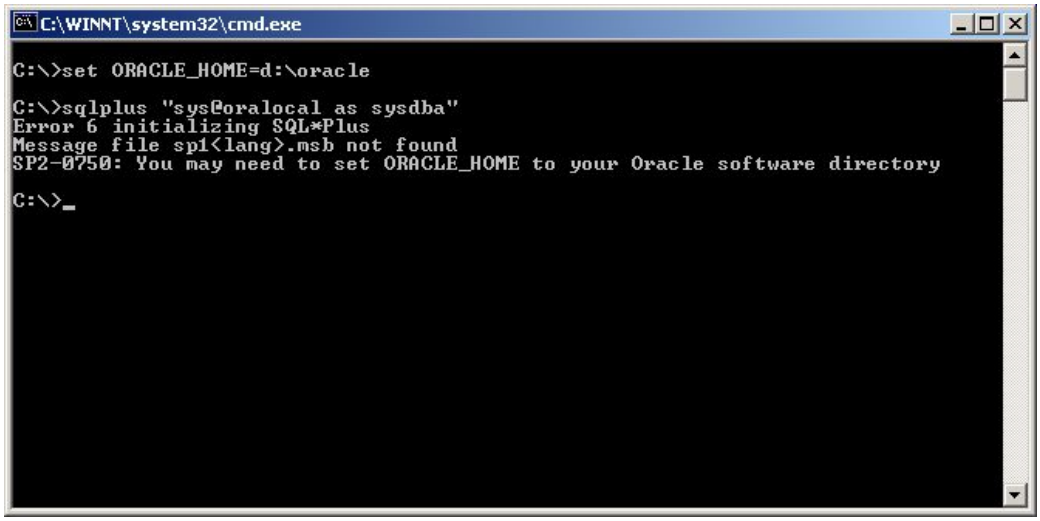
```
C:\>sqlplus system@ORALOCAL
```

той връща грешката "`ORA-12154: TNS:could not resolve service name`", което е изненадващо за потребителя, сблъскващ се за първи път с подобна ситуация. След дълго ровене и търсене се оказва, че проблема е възникнал след инсталацията на Oracle софтуер в нов Oracle Home. В случая, SQL*Plus се опитва да се свърже към базата, търсейки конфигурационния файл tnsnames.ora в грешна директория, защото е стартиран от грешен Oracle Home, но това не може да бъде забелязано. Този конкретен проблем има косвена връзка с ORACLE_HOME. За да може да се избягват този и всички проблеми, свързани по някакъв начин с ORACLE_HOME, трябва да се разгледа в детайли принципа на достъп на Oracle до тази променлива, както и нейното предназначение.
За конкретния пример се приема, че има инсталирани два Oracle Home:

- Софтуер за Oracle база данни, версия 9.0.1, който е току-що инсталиран. Намира се в директорията D:\oracle\ora91
- Софтуер за Oracle база данни, версия 9.2.0.5, който е използван от потребителя за някакъв период от време преди другата инсталация. Намира се в директорията D:\oracle\ora92

Всеки Oracle Tool, Utility, както и самата база се стартират от точно определена директория, наречена bin и чието име се запазва, независимо какъв Oracle софтуер е инсталиран в конкретния Oracle Home. Веднага след стартирането на даден изпълним файл (напр. D:\oracle\ora92\bin\Sqlplus.exe), се извиква библиотеката ORACORE9.DLL [изт. 19], която от своя страна използвайки функцията GetModuleFileName(), открива директорията в която се намира и съответно е бил стартиран този файл (напр. d:\oracle\ora92\bin). След като тази информация е открита, се отваря друг файл, намиращ се в току-що откритата директория, наречен oracle.key (в конкретния случай съдържанието на този файл е: "Software\ORACLE\HOME0"), и оттук се взима точното местоположение на всички външни параметри вътре в Registry. С помощта на този вътрешен механизъм, изпълнимият файл успява да открие от кой Registry ключ да чете всички стойности, някои от които са му нужни още при стартирането, а други по време на работата му.
Ако има нужда да се разбере кой Registry ключ се използва от всички продукти в даден Oracle Home, то трябва да се потърси файла oracle.key в bin директорията и да се провери информацията, която той съдържа. Този файл се създава по време на инсталацията от Oracle Universal Installer, като обикновено връзката между конретната bin директория и указаните външни параметри в Registry съвпадат логически, т.е. в директорията d:\oracle\ora92ю\bin се намира файла oracle.key и той съдържа указателя: "Software\ORACLE\HOME0", и съответно на това място в Registry в същия път се откриват външните параметри: "ORACLE_HOME=d:\oracle\ora92" или SQLPATH=d:\oracle\ora92\dbs". Ако се наложи, даден Oracle Home може да бъде асоцииран с различен Registry ключ от този който е по подразбиране, което става именно с редактиране на oracle.key.
Винаги, когато в Windows се стартира изпълним файл без да се указва път, неговото местоположение се открива с помощта на системната обкръжаваща променлива PATH, а когато в PATH има повече от един деклариран път, например "d:\oracle\ora91\bin;d:\oracle\ora92\bin" приоритетът на търсене е от първия към последния в списъка. При всяка нова инсталация или преинсталация на даден Oracle Home, Oracle Universal Installer редактира тази променлива и добавя пътят към новия OracleHome най-отпред в списъка. При следващия опит да се стартира SQL*Plus без да се указва пълния път до него, се взима първият открит от списъка с възможните пътища, указан в

PATH – d:\oracle\ora91\bin. Тогава се стартира SQL*Plus от директорията D:\oracle\ora91\bin и той, спазвайки гореспоменатия механизъм, открива своя Registry ключ "Software\ORACLE\HOME1", взима всички нужни параметри, а за всички, които не са указани там, използва стойността им по подразбиране. В случая, за да се свърже към базата му е нужен файла tnsnames.ora, чието местоположение се съхранява от външния параметър TNS_ADMIN и ако той не е указан излишно, стойността му се определя от съдържанието на ORACLE_HOME, а именно TNS_ADMIN=%ORACLE_HOME%\network\admin, оттук TNS_ADMIN=d:\oracle\ora91\network\admin. В резултат SQL*Plus не успява да открие такъв файл и връща грешка. В същото време потребителят не търси проблема в тази посока, защото знае, че такъв файл съществува и той го е използвал многократно. Този файл обаче се намира в d:\oracle\ora92\network\admin. За да се разрешат подобни проблеми, предизвикани от поддържането на няколко Oracle Homes, съществува Oracle Tool, наречен Oracle Home Selector [изт. 16]. С него лесно може да бъде променян този Oracle Home, който е по подразбиране или да бъде определян т.нар Primary Oracle Home. Той, като всеки Oracle Tool също чете нужните му параметри в Registry (списъкът с тези параметри и тяхното предназначение може да бъде открит по-долу). Ако потребителят разглежда съдържанието на Registry остава с усещането, че при настройването на Primary Oracle Home, този софтуер променя и някакви настройки в Registry, главно поради наличието на параметърът DEFAULT_HOME, намиращ се под ключа HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\ALL_HOMES\. Всъщност той няма отношение към действията, извършвани от Oracle Home Selector. Единственото, което се променя е редът на пътищата в системната обкръжаваща променлива PATH.
Затова ръчното настройване на PATH може напълно да замести действията, които се извършват от Oracle Home Selector, без това да предизвика допълнителни проблеми. Винаги, когато се направи такава промяна (ръчно или не) на ниво операционна система, промените важат само за новоотворените командни прозорци, но не и за отворените по време на промяната.
Тъй като ORACLE_HOME е Oracle обкръжаваща променлива, тя може да бъда настройвана и на ниво операционна система или за даден команден прозорец. Ето какво се случва обаче, когато бъде настроена грешна стойност на ORACLE_HOME в команден прозорец преди стартирането на SQL*Plus.



Тъй като настроените променливи в командния прозорец са с най-голям приоритет, SQL*Plus взима тази стойност, независимо от това дали има такава настроена в съответстващия на неговия Home Registry ключ. В случая, той се опитва да си намери message файловете (в които се съхраняват текстовете за показване и грешките, нужни за нормалната му работа) в погрешна директория и изобщо не успява да се стартира. Въпреки, че успява да си вземе всички нужни му променливи от Registry, той използва стойността на ORACLE_HOME, която е настроена в командния прозорец и съответно е с най-голям приоритет. След това всяка променлива, която зависи от стойността на ORACLE_HOME се изчислява грешно. Тази грешка може да възникне и когато ORACLE_HOME не е настроена нито в Registry, на ниво ОС, или в командния прозорец. Затова винаги трябва да се проверява дали тя е дефинирана за всеки Oracle Home и дали има коректна стойност. Oracle препоръчва настройването на тази променлива извън Registry да не се прави, защото може да предизвика много специфични проблеми. Тестовете показват, че при настроена грешна стойност на ORACLE_HOME, нито един Oracle Tool или Utility не може да се стартира нормално и поведението на апликацията трудно може да бъде свързано с грешно настроена стойност на тази променлива.

НАСТРОЙКА НА ORACLE_SID

Друга важна Oracle обкръжаваща променлива е ORACLE_SID. Винаги, когато в даден Oracle Home има конфигурирана база данни, използваща софтуера, намиращ се в този Home, в Registry автоматично се създава Oracle обкръжаващата променлива ORACLE_SID, указваща името на базата, работеща под този Home. При създаването на нова база, ORACLE_SID се променя автоматично да отговаря на нейното име. Поради тази причина, винаги когато има инсталирани няколко бази, работещи под определен Oracle Home и трябва да се осъществи ОС автентикация е задължителна ръчната настройка на ORACLE_SID за конкретния команден прозорец. В противен случай, ако не се укаже излишно, потребителят се свързва към базата, чието име е указано като стойност на ORACEL_SID в Registry, а ако изобщо няма дефинирана такава променлива, то SQL*Plus връща грешката: "`ORA-12560: TNS:protocol adapter error`".
Често при писането на автоматични скриптове (batch scripts), които извършват конкретни промени в базата се налага използването на Oracle обкръжаващите променливи ORACLE_BASE и ORACLE_SID. Това намалява възможността от допускането на грешки при въвеждането на пътища и имена, и прави скриптовете универсални за всяка една база. Ето примерен израз, който може да се изпълни в SQL*Plus:

```
CREATE TABLESPACE TEST
 DATAFILE '%ORACLE_BASE%\oradata\%ORACLE_SID%\TEST01.DBF'
 SIZE 25M REUSE AUTOEXTEND ON NEXT 10240K MAXSIZE 100M
 EXTENT MANAGEMENT LOCAL
 SEGMENT SPACE MANAGEMENT AUTO;
```

Тук възниква въпроса – какво ще се случи, ако за конкретния Oracle Home, от който е стартиран SQL*Plus е настроен ORACLE_SID за друга база или пък няма такъв.
От разгледаните в по-горните точки променливи, очакваното поведение е да се вземе стойността на ORACLE_SID от Registry (или настроената в командния прозорец) и така да бъде генериран пътя, независимо че тази стойност може да причини създаването на това таблично пространство (tablespace) при файловете с данни (data files), принадлежащи на друга база данни. Тестовете обаче сочат, че това не става, независимо дали има настроена променлива или тя съдържа името на друга база. SQL*Plus изобщо не проверява каква е стойността в Registry, нито взема предвид зададената в командния прозорец (ако се използва такава). Винаги, когато горната команда бъде изпълнена, табличното пространство се създава в директорията с името на базата, към която SQL*Plus е свързан в момента. При включено трасиране по време на изпълнение на командата се открива, че името на базата не се чете и от някоя от таблиците от речника на базата (data dictionary). Това показва, че при използването на този параметър в скриптове, няма значение дали и къде е настроена тази променлива, стойността и винаги е името на базата, към която е свързан SQL*Plus в момента. Ако се налага създаването на таблично пространство на друго място, то използването на която и да е друга Oracle обкръжаваща променлива успява да заобиколи задължителното създаване на това таблично пространство в ORACLE_SID директорията.

НАСТРОЙКА НА NLS_LANG

Oracle обкръжаваща променлива, която указва езика и територията, които се използват от даден Oracle Tool или Utility при работата му. Стойността на тази променлива указва също и набора от символи (character set), които да бъдат използвани от клиента, и които съответно ще се използват при набиране и показване на данни на клиентската страна. NLS_LANG има три съставни части: език, територия и набор от символи. Дефиницията му е:
NLS_LANG=LANGUAGE_TERRITORY.CHARSET
При всяка инсталация Oracle Universal Installer се грижи за настройката на този параметър, като избира стойността му в зависимост от локалните настройки на ОС потребителя. Причината за погрешно настроена стойност на NLS_LANG може да е причинена от неправилно избраната езикова настройка на ниво операционна система, която впоследствие е използвана и от Oracle Universal Installer. Ако поради някаква причина в Registry не съществува такъв параметър, то се взима стойността му по подразбиране: AMERICAN_AMERICA.US7ASCII
Всеки компонент на тази променлива контролира някакво подмножество от поведението на клиентската страна:

- LANGUAGE – определя вида на използвания език за показване на Oracle грешките, начина на сортиране, имената на дните, имената на месеците.
- TERRITORY – определя формата на датата, паричните единици и формата на числовите данни
- CHARSET – определя набора от символи, които ще се използват за показване на екрана на клиента

Всеки език има асоциран набор от символи по подразбиране (default character set).
Всички компоненти на променливата не са задължителни и NLS_LANG може да бъде дефинирана без да бъдат указани всичките три. Например, тези дефиниции са коректни [изт. 5, стр. 3-5]:
NLS_LANG=AMERICAN_.CL8MSWIN1251
NLS_LANG=AMERICAN_AMERICA.

NLS_LANG=AMERICAN_.
, дори и тази:
NLS_LANG=_.
Тук всяка една липсваща стойност се заменя с нейната по подразбиране, като трябва само да се спазва правилото че за набора от символи, текстовата поредица трябва да завършва с точка “.”, а ако липсва територията след езика да стои знака “_”. Въпреки, че тази гъвкавост позволява каквато и да е комбинация от стойности на компонентите, при неправилно настроена такава комбинация, те просто няма да работят коректно. Затова, понякога липсата на дефинирана променлива може да връща по-нормални резултати, отколкото след ръчното и настройване.
Стойността на този параметър определя използваните език и територия не само от клиентската програма, а и от сесията, поддържана от Oracle Server като комуникация между базата и клиентската апликация (например изпълнението на SQL изрази). По принцип настройката на тази променлива трябва да съвпада с тази на базата данни, което няма да позволи извършването на автоматична конверсия на данните при транспортирането им от и към клиента, и които биха могли да доведат до показването на повредени данни на клиентската страна. В идеалния случай те трябва да съвпадат. Тази настройка обаче зависи и от това, какъв набор от символи използва ОС потребителят, който в момента работи на самия клиентски компютър. NLS_LANG трябва да е така настроена, че да е максимално близка като стойност и до използваната кодова таблица (code page) от самата операционна система. Стойността трябва да е такава, че да позволи правилна конверсия на данните от използвания набор от символи от клиента към този на базата. В противен случай дори и настройката на NLS_LANG от клиентската страна да съвпада с тази на базата данни, то ако операционната система не извърши правилна конверсия, данните биха могли да се повредят. В общи линии може да се използва правилото, че може да бъде извършена коректна конверсия от от даден набор символи към втори набор символи, само ако първия се явява подмножество на втория. Обратното е възможно, но в никакъв случай не е задължително. Например, коректна конверсия може да се извърши от WE8ISO8859P1 към UTF8.
Това е характерен проблем при използването на Export и Import Utilities, които много често при използването им извеждат подобно на това съобщение (в случая съобщението е показано от Export Utility):

```
Export done in US7ASCII character set and AL16UTF16 NCHAR character set
server uses WE8ISO8859P1 character set (possible charset conversion)
```

Потребителят трябва да знае, че това не гарантира че специалните символи ще бъдат експортнати коректно и няма да има загуба на данни, и в практиката много често се срещат такива проблеми [изт. 12, стр. 359]. Затова винаги, за да бъде сигурно, че е създадено коректно копие на съществуващите данни, генерирания файл трябва да бъде импортнат и да бъде тествано дали не са повредени данните със специални символи. В случаите, когато се борави с огромен обем данни, биха могли да се тестват само специално подбрани таблици, които да докажат, че е извършена коректна конверсия и няма повредени специални символи.

Каква Стойност Използва в Момента SQL*PLUS

За да се установи каква стойност на конкретен външен параметър използва SQL*Plus в сесията си, неговата стойност може да се вземе с помощта на стандартния начин за получаването и в команден прозорец - “%ORACLE_SID%”. Тъй като SQL*Plus очаква единствено команди, които може да изпълнява, то най-лесният начин да се получи тази информация е като се изпълни следната команда [изт. 14]:

```
SQL> @.[%ORACLE_SID%]
```

Резултата би трябвало да изглежда горе-долу така:

```
SP2-0310: unable to open file ".[ORALOCAL]"
```

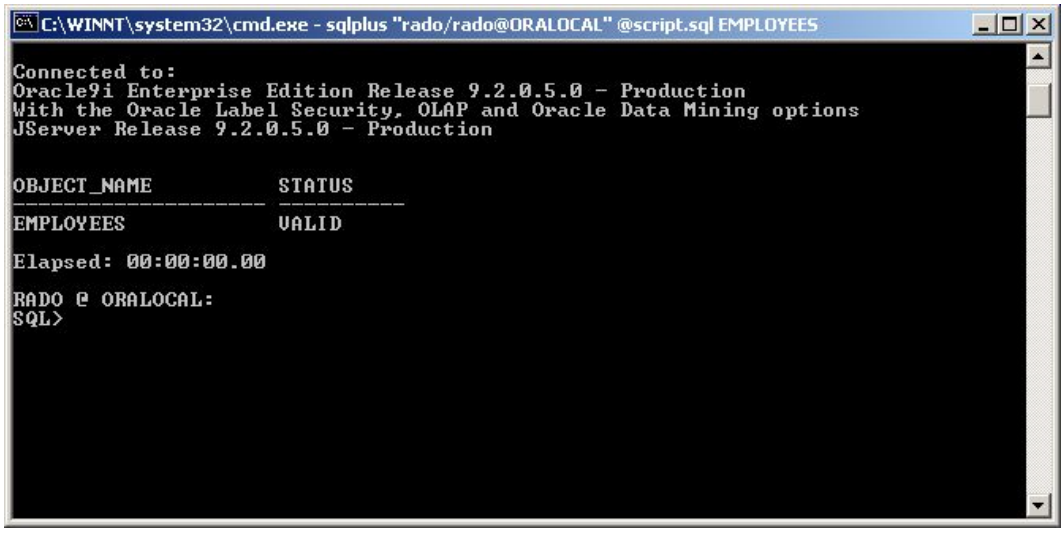
По този начин може да бъде изведена всяка една стойност на външен праметър, независимо дали е дефиниран в Registry или извън него.

Използване на Стойностите на Външни Параметри в SQL Израз

В практиката може да се наложи даден външен параметър да бъде достъпен в SQL израз, или да бъде подаден като параметър при изпълнението на PL/SQL процедура. Тук ще се разгледат начините на взимане на стойностите и на двата вида параметри.
Стойности на системни обкръжаващи променливи:
В SQL*Plus може да се вземе стойността на системна или Oracle обкръжаваща променлива (ако тя е дефинира на ниво ОС или в командния прозорец) по следния начин [изт. 15]:

```
set ORA_TESTVAR=EMPLOYEES
sqlplus "rado/rado@ORALOCAL" @script.sql %ORA_TESTVAR%
където съдържанието на scripts.sql е:
```

```
set echo off
set verify off
COLUMN object_name FORMAT A20
COLUMN status FORMAT A10
define var=&1;
SELECT object_name, status FROM user_objects WHERE object_name='&var';
```
Резулата от изпълнението на първите два реда в команден прозорец е:



Логиката е същата както, когато се изпълнява скрипт с изброени статични параметри на същия ред.

Стойности на параметри от Registry:
Докато SQL*Plus има механизъм, с който може да прочете стойността на една външна променлива, която е дефинирана в Registry и да я използва, това не може да бъде направено явно в SQL израз, например не би могло да се изпълни следния израз и той да използва истинската стойност от Registry:
```
SELECT object_name, status FROM user_objects WHERE object_name='%ORACLE_SID%';
```
Единият възможен вариант това да се постигне е да се използва възможността на изпълнимия файл regedit.exe да бъде стартиран в команден прозорец (а не в стандартно използвания от всеки графичен режим). За повече информация [изт. 20]. По този начин могат да се експортнат стойностите на всички параметри, намиращи се под желания ключ:
```
regedit /e C:\temp\ORA_HOME0.txt  "HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\HOME0"
```
Генерирания файл изглежда горе-долу по следния начин:
```
.....
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\HOME0\]
"ID"="0"
"ORACLE_GROUP_NAME"="Oracle - OraHome92"
"ORACLE_HOME_NAME"="OraHome92"
.....
```
и т.н.

След като този файл съществува, той би могъл да се обработи по подходящ начин със програма и тези стойности да бъдат подадени като параметри. В случай, че скрипта, на който му се налага да използва такива параметри, се изпълнява на компютъра, на който работи базата данни, то би могло този файл да се прочете и обработи от PL/SQL. Това може да стане с помощта на инициализационния параметър UTL_FILE_DIR и възможността на PL/SQL чете файлове, намиращи се в директорията указана от този параметър. Друг вариант би могъл да е използването на възможността на Oracle да чете файлове като външни таблици (External Tables) и по този начин да се вземат стойностите на нужните Registry параметри.

Друг по-универсален и лесен за приложение вариант, също използващ експортнати данни от даден Registry ключ (приема се, че преди това в Registry е създаден параметър с име ORA_TESTVAR и стойност “EMPLOYEES”, но примера може да се приложи за всеки външен параметър, дефиниран в Registry) е:

```
START /W REGEDIT /E D:\Temp\orareg.txt "HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\HOME0"

FOR /F "tokens=1* delims==" %A IN ('TYPE "D:\Temp\orareg.txt" ^¦ FIND
"ORA_TESTVAR"') DO IF %A=="ORA_TESTVAR" SET ORA_TESTVAR=%B

DEL D:\Temp\orareg.txt

sqlplus "rado/rado@ORALOCAL" @script.sql %ORA_TESTVAR%
```

По този начин биха могли да се прочетат данните от Registry, независимо къде се изпълнява скрипта – на клиентската страна или на сървърската.
Ако примера се използва като код в batch файл, то трябва да се добави още един един допълнителен символ “%” пред параметъра A и B. В случай, че скрипта се тества директно в команден прозорец, то горния код може да се използва директно.
Резултатът от изпълнението на горния скрипт би трябвало да е същия, както в показания в примера за използване на системни обкръжаващи променливи, с тази разлика, че параметъра ORA_TESTVAR съществува само в Registry.

Използване на Собствени Външни Параметри

В Registry биха могли да се дефинират и използват собствени параметри (например ORA_NEW_TBS_LOCATION_NAME), които да играят ролята на Oracle обкръжаваща променлива. Те биха могли да се използват в автоматични скриптове. Гореспоменатият израз за създаване на таблично пространство би могъл да изглежда по следния начин:

```
CREATE TABLESPACE TEST
 DATAFILE '%ORACLE_BASE%\oradata\% ORA_NEW_TBS_LOCATION_NAME%\TEST01.DBF'
 SIZE 25M REUSE AUTOEXTEND ON NEXT 10240K MAXSIZE 100M
 EXTENT MANAGEMENT LOCAL
 SEGMENT SPACE MANAGEMENT AUTO;
```

Този израз ще създаде таблично пространство TEST в директорията с име, указано в параметъра ORA_NEW_TBS_LOCATION_NAME.

Препоръки

При ръчната настройка на системна или Oracle обкръжаваща променлива [изт. 11]не трябва да се оставят интервали между знака за равно (=), например:

```
set ORACLE_SID= ORALOCAL
```

Появата на интервал между знака за равно може да доведе до асоцииране на грешна стойност към тази променлива, което от своя страна да доведе до възникването на проблеми при опит параметъра да бъде използван. В конкретния случай няма да може да бъде осъществена връзка с базата.
Външните параметри не са чувствителни към големи и малки букви (case sensitive), нито на ниво ОС, в Registry или в командния прозорец. Няма значение и как ще бъдат изписани стойностите на променливите. Например, тези команди изпълняват една и съща операция и дефинират една и съща променлива с еднаква стойност.

```
set ORACLE_SID=ORALOCAL
set oracle_sid=oralocal
```

Опитите да се дефинира външен параметър в Registry на по-високо ниво от ключа за съответния Oracle Home е неправилно и не води до очаквания резултат, а именно, този параметър да важи за всички ключове на Oracle Homes от по-долните нива. Т.е., настройката на ниво HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE не се отчита от програмите, използващи параметрите в HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\HOME0, независимо дали те ще намерят параметъра, който им трябва в своя ключ или не. Настройването на параметрите в Registry винаги трябва да става за желания Home. Възможно е някои параметри да бъдат настройвани под ключа HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\ALL_HOMES и да важат за всички инсталирани Oracle Homes – те ще бъдат разгледани по-долу.

Настройка на Автоматичното Спиране на Базата при Рестартиране на Компютъра

Тук ще се обърне внимание на един малък спор – спира ли базата по нормалния начин, когато се наложи да се рестартира компютъра, на който тя работи. Много администратори, твърдейки че "не вярват на Windows", винаги спират ръчно базата, преди да рестартират компютъра, а другата страна на спора твърди, че това не е необходимо и няма никакъв проблем. Очакваното поведение е базата да спре, защото Oracle Service, който отговаря за нейното спиране и пускане се спира автоматично при нормалното рестартиране на компютъра.
При автоматичното или ръчно създаване на една база, в Registry автоматично се създават четири Oracle Registry параметъра: ORA_SID_AUTOSTART, ORA_SID_SHUTDOWN, ORA_SID_SHUTDOWN_TIMEOUT, ORA_SID_SHUTDOWNTYPE (където SID е името на конкретната база), които имат пряко отношение към това поведение. Тези параметри се създават от ORADIM Utility, независимо дали базата се инсталира ръчно или автоматично с Database Configuration Assistant. Те определят как новосъздадения Windows Service с име "OracleServiceSID" ще си комуникира със самата база и как ще я управлява. Този Service ще спира и стартира базата посредством ORADIM, а съответно неговото повдение се определя от тези параметри.
ORA_SID_AUTOSTART – указва дали базата да се стартира автоматично заедно със стартирането на асоциирания Service към базата. Възможните стойности са TRUE и FALSE, а стойността по подразбиране е TRUE
ORA_SID_SHUTDOWN – указва дали базата да спре, когато спре и асоциирания към базата Service. Възможните стойности са TRUE и FALSE, а стойността по подразбиране е TRUE
ORA_SID_SHUTDOWNTYPE – указва с какъв метод да се спре базата, когато асоциирания към базата OracleServiceSID бъде спрян. Валидните стойностти са "a" – abort, "i" – immediate и "n" – normal. Стойността по подразбиране е "i". Този параметър, заедно с предишните два доказва, че поведението на базата е точно такова, каквото се очаква да бъде. Практиката показва, че това не винаги се случва.
Последния, четвърти параметър имащ отношение към разглеждания проблем е:
ORA_SID_SHUTDOWN_TIMEOUT – указва времето в секунди, за което асоциирания към базата с име SID, Service, да изчака базата, за да завърши тя процеса си на спиране. Стойността по подразбиране е 30 секунди. Ако това време бъде превишено, то OracleServiceSID ще спре принудително базата. За повече информация [изт. 21].
Този параметър трябва да има достатъчно голяма стойност, за да може базата да завърши спирането си по нормалния начин. В противен случай, при последващото стартиране на базата, тя ще трябва да извърши т.нар "Instance Recovery". За да може да се избегне всякакво ненормално спиране на базата, то освен този параметър, трябва да се настрои още един параметър в Registry, който не е свързан конкретно с Oracle, но има отношение към работата на всички Windows Services, в частност и на Oracle Services:
WaitToKillServiceTimeout – указва колко време трябва да се изчака един Service, когато той се опитва да спре, преди да бъде спрян принудително. Този ключ се намира в Registry ключа HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control. Стойността му подразбиране е 20 000 микросекунди (или измерен в секунди – 20). Оттук се вижда, че дори и параметъра ORA_SID_SHUTDOWN_TIMEOUT да е настроен на 30 секунди, то базата ще бъде спряна след 20 секунди изчакване. Стойността на "WaitToKillServiceTimeout" трябва бъде достатъчно голяма, за да може базата да спре нормално. За да има ефект тази настройка, то след всяка промяна компютъра трябва да бъде рестартиран.
Тези параметри помагат базата да бъде така настроена, че да спира нормално.
Какво се случва, ако базата въпреки настройките (независимо дали е имало някаква намеса от страна на потребителя или стойностите на горните параметри са тези по подразбиране) не спира нормално.
Ако OracleServiceSID не спре базата, то нейното спиране няма да е подобно на изпълнението на командата "shutdown abort", а дори по-лошо защото Windows просто ще изчисти частта от паметта, заета от съответния Oracle процес и ще продължи с рестартирането си. Това спиране е много близо до принудителното спиране на базата, когато електрозахранването на компютъра прекъсне изненадващо.
Един от начините да се провери дали конкретна база се спира нормално е да се погледне нейния alert.log файл – всяко наличие на ред, подобен на следващия, ще покаже че базата не спира по нормалния начин:

```
Beginning crash recovery of 1 threads
```

Най-сигурният начин потребителят да се увери, че базата му изпитва такъв проблем, е като настрои съответния OracleServiceSID да не се стартира автоматично, след това направи копие на сегашното състояние на alert.log файла на дадената база, да отвори оригиналния файл и да изчисти съдържанието му, като го остави празен. След това да рестартира компютъра. Ако след рестартирането файла alert.log остане празен, това е най-сигурния показател, че базата не спира по нормалния начин.
Друг начин това да се разбере има отношение към следващия Oracle Registry параметър:

ORA_CWD (CWD идва от Current Working Directory) - указва директорията, в която се намира лог файла на ORADIM Utility – oradim.log. Стойността му по подразбиране е %ORACLE_HOME%\database. Този файл записва неуспешните действия, извършени от ORADIM – като например всеки негов неуспешен опит да спре или стартира базата. Той може да бъде изключително полезен при диагностициране на проблеми, които не се записват никъде другаде.
Конкретен пример е един специфичен проблем – след успешно създаване на база, която по принцип работи нормално, и след последващо рестартиране се оказва, че въпреки че OracleServiceSID има статус "Started", то базата не е отворена и всеки път се налага тя да се отваря ръчно от потребителя. Този проблем е труден за диагностициране, защото базата всъщност не е стартирана и съответно не е записала нищо в своя alert.log файл. Преглеждането на oradim.log би могло да помогне – много често проблема идва от грешно зададен път до инициализационния файл например. Друг специфичен проблем възниква, когато базата данни работи на компютър, който е член на домейн и съответно за стартирането на Oracle Services се използва потребител на операционната система, който е член на домейна. Тогава при спирането на базата се извършват специфични действия и в определени случаи, тяхното неуспешно изпълнение е причина базата също да не спре нормално.
С помощта на този файл всеки би могъл да разбере дали дадена база успява да се спре по нормалния начин. Липсата на такъв файл в директорията по подразбиране, е сигурен индикатор, че няма такъв проблем.
При диагностициране на конкретен проблем, проверката и на записите в oradim.log може да окаже голяма помощ в откриването и разрешаването му.

Възможни Параметри за Настройка в Registry

Тук се прави опит да се направи максимално пълен списък на всички т.нар. външни параметри, тяхното предназначение, стойности, и как могат да се настройват. С всяка следваща версия на документа, този списък ще бъде допълван и разширяван, включително и с нови източници, от които може да се разбере в детайли как те могат да бъдат използвани.

Препоръка: *Преди всяка промяна в Windows Registry трябва да бъде направено копие на текущото му състояние, защото има опасност да възникне сериозен проблем в нормалното функциониране на операционната система, както и на апликацият,е работещи под нея.*

Ключ *HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE*
Грешно е тук да се слагат параметри, чието място по принцип е в HOMEx ключа, с идеята че този параметър ще важи за всички инсталирани такива. Ако има настроен параметър на това място, то той ще важи само за първия инсталиран Home. Всички останали ще използват Registry ключа за своя Home.
INST_LOC – указва местонахождението на файловете на Oracle Universal Installer, стойността по подразбиране е: System Drive:\Program Files\oracle\inventory. Той се инсталира в тази директория. Винаги, когато се налага да се прави backup на даден Oracle Home не трябва да се забравя и тази директория, защото тук се пази цялата информация за инсталираните компоненти в момента, както и всички инсталационни логове. Без нея, при опита да се възстанови този Oracle Home, версията съхранявана в Inventory няма да съвпадне с тази във възстановения Oracle Home.
OI_NLS32 – използва се от Oracle Installer в Oracle 7.3, не се поддържа, но продължава да се извиква от Oracle библиотеките

Подключ *HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\ALL_HOMES*
DEFAULT_HOME - указва името на първия инсталиран Oracle Home. Според документацията, стойността на този параметър си остава винаги първия инсталиран Oracle Home и по принцип след това не се променя, това обаче може да бъде направено ръчно, ако има такава нужда. При такава промяна трябва да се направи и съответната промяна в ключа ID0, който да отговаря на новата въведена стойност.
Този параметър понякога се обърква с действията, които извършва Oracle Home Selector (поради смесването на понятията Default и Primary Oracle Home), но всъщност този Tool не прави никакви промени в Registry и няма нищо общо с този параметър
HOME_COUNTER – броят на всички инсталирани Oracle Homes
LAST_HOME – номера на последния инсталиран Home
OSAUTH_PREFIX_DOMAIN – може да стои и под този ключ и да важи за всички Oracle Homes. Ще бъде разгледан малко по-надолу.

IDx – този подключ е свързан със съответния си подключ HOMEx в *HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE*.
Параметрите, намиращи се под него, се използват от Oracle Universal Installer.

- NLS_LANG – набора от символи, използван от Oracle Universal Installer
- PATH – директорията, в която е избран да бъде инсталиран Oracle софтуера
- NAME - името на Oracle Home, което е дадено по време на инсталацията

Дори и да не бъде направена инсталация, а само да е избран нов Oracle Home, и след няколко стъпки от инсталацията потребителя да реши да я прекъсне, този ключ се създава и не се изтрива. При следващото си стартиране, Oracle Universal Installer приема, че такъв Home съществува. В такъв случай ръчното премахване на този ключ може да помогне той да не се появява повече в списъка за избиране.
Параметрите в този ключ, се използват и от Oracle Home Selector за да покаже имената на инсталираните Oracle Homes и тяхното местоположение.

Подключ *HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\HOMEID*

Параметри, Използвани от Oracle Tools и Utilities

Тук са включени параметрите, използвани от Oracle Tools и Utilities, които в ролята им на клиенти се свързват към дадена база. Всички те са Oracle обкръжаващи променливи и ако има изключения, те ще бъдат указани. Няма информация за поведението на недокументираните параметри.

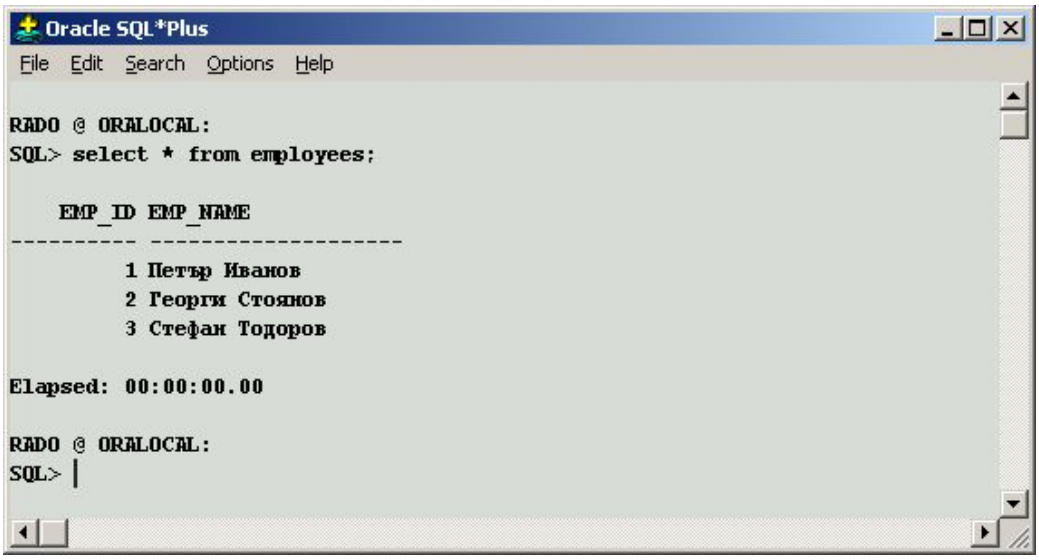
ORACLE_HOME_KEY – указва местоположението на външните параметри в Registry под ключа HKEY_LOCAL_MACHINE\
ORACLE_HOME_NAME – указва името на дадения Oracle Home
ORACLE_GROUP_NAME – указва името на групата за инсталираните Oracle продукти (името на групата в Start -> Programs). Създава се още при инсталирането, стойността му по подразбиране е Oracle – HOME_NAME
MSHELP_TOOLS – указва местоположението на помощните (help) файлове на Windows. По подразбиране е: %ORACLE_HOME%\MSHELP
ORACLE_BUNDLE_NAME – недокументиран, указва версията на инсталирания софтуер. Откритите възможни стойности са “Enterprise” и “Standard”
NLS_LANG – беше разгледан по-горе
NLS_CALENDAR – указва типа на използвания календар. Календарите се различават по: първия ден на седмицата, първата календарна седмица на годината, броя на дните и на месеците в годината
NLS_CREDIT – указва символа, който се използва за показване на кредитите във финансовите репорти
NLS_DEBIT - указва символа, който се използва за показване на дебитите във финансовите репорти
NLS_CURRENCY – указва знака на използваната валута
NLS_ISO_CURRENCY – указва уникалния знак на използваната валута, който за разлика от дефинирания с NLS_CURRENCY не позволява да има дублираните на знаците за някои държави
NLS_DUAL_CURRENCY – указва знака на втората валидна валута в дадената територия
NLS_DATE_LANGUAGE – указва език, на който ще се показват дните и месеца на датата
NLS_DATE_FORMAT – указва какъв да бъде покзвания формат на датата
NLS_TERRITORY – указва територията
NLS_TIMESTAMP_FORMAT – указва TIMESTAMP формата по подразбиране при работа с функциите TO_CHAR и TO_TIMESTAMP
NLS_TIMESTAMP_TZ_FORMAT - указва TIMESTAMP (с включен часови пояс) формата по подразбиране при работа с функциите TO_CHAR и TO_TIMESTAMP_TZ
NLS_TIME_FORMAT – указва формата на показвания час
NLS_TIME_TZ_FORMAT – указва формата на показвания час (с включен часови пояс)
NLS_MONETARY_CHARACTERS – указва вида на символа, който служи за разделител на хилядните от десетиците в парични изрази
NLS_NUMERIC_CHARACTERS – указва начина, по който да се форматират числовите данни
NLS_SORT – указва типа на сортиране на текстовите данни – по азбучен ред или двоично
NLS_COMP – указва да не се използва стойността на NLS_SORT в SQL изразите, може да бъде много полезна при създаването на т.нар. лингвистични индекси
NLS_LIST_SEPARATOR – указва какъв знак да бъде използван за разделяне на стойностите в един списък
За подробен списък и описание и начин на използване на гореописаните NLS параметри [изт. 5].

Приоритета, според който се избира NLS стойността, която ще бъде използвана в клентската сесия в SQL*Plus е:

- 1 (най-висок приоритет) – настроените в SQL израз. Ето пример как може да се покаже месеца на български, без да се променя параметъра NLS_DATE_LANGUAGE никъде, дори и в сесията:
  ```
  SELECT to_nchar(SYSDATE, 'DD-MON-YYYYY
  HH24:MI','NLS_DATE_LANGUAGE=BULGARIAN') FROM dual
  ```
- 2 – избраните с помощта на ALTER SESSION
- 3 - избраните като обкръжаващи променливи
- 4 – избраните в инициализационния файл
- 5 (най-малък приоритет) – стойността по подразбиране

ORA_NLSxx (където xx е 32 за Oracle 7.3, 33 за версии 8, 8i и 9i, а 10 за 10g) – указва пътя до NLS библиотеките на Oracle при създаване на база. Задължителен е под Unix в определени случаи, за Windows не. За повече информация [изт. 26]
ORA_NLS_PROFILExx – недокументиран, има отношение към ORA_NLSxx
NLS_UNION_CURRENCY – използва се, за да се предифинира символа за валута, който в момента се използва по подразбиране
NLS_STORAGE_ORDER – недокументиран
NLS_DISPLAY – използва се за да поддържа арабски шрифтове за Oracle Developer, и то под Unix. Въпреки всичко, този параметър се извиква и от библиотеките в Windows
ORA_NLS_CHARACTERSET_CONVERSION – в Oracle 7 с негова помощ е било възможно да се забрани character set конверсията между клиента и сървъра. От Oracle 8.0.4 вече не се използва, но продължава да съществува в кода на доста бинарни файлове
ORA_SDTZ – указва часовата зона на клиента по подразбиране
NCHAR_CONV_EXCP - недокументиран
NCHAR_IMP_CONV - недокументиран
ORA_ENCRYPT_LOGIN – указва паролата, която се изпраща от клиента по мрежата при свързване към базата да бъде криптирана, стойността по подразбиране е FALSE. Този параметър трябва да бъде настроен на TRUE. От Oracle версия 7.1 паролата винаги се праща криптирана по мрежата, но все още има поне два начина паролата да бъде изпратена в чист текст и затова този параметър винаги трябва да бъде TRUE, когато даден потребител се свързва към базата. Единия случай е, ако за свързване към базата се използва Oracle Tool, който съответно използва Oracle 7.1 библиотеки - тогава той може да изпраща некриптирани пароли към базата, защото има възможност да работи в стария “clear text mode”. Втория случай е, ако конекцията към базата се прекъсне и в същото време одита на базата е включен - тогава Oracle се опитва да провери конекцията за втори път и точно в този момент, той изпраща некриптирана парола, при положение, че стойността на ORA_ENCRYPT_LOGIN е FALSE при клиента. За повече информация [изт. 2].
TNS_ADMIN – указва директорията, в която се съхраняват мрежовите конфигурационни файлове като: sqlnet.ora, tnsnames.ora, listener.ora, ldap.ora, names.ora, cman.ora. Ако стойността по подразбиране %ORACLE_HOME%\network\admin е сменена и Oracle не открие нужния му файл в тази директория, той търси файла и директорията по подразбиране. Ако се налага да се сменя тази стойност, то в пътя по подразбиране не трябва да остава старата версия или копие на файла, за да не се позволи Oracle да използва друг файл без това изрично да му е указвано
SQLPLUS – недокументиран, указва пътя на SQL*Plus, откъдето да си вземе message файловете, ако те не се намират в мястото, което е по подразбиране
SQLPATH – указва коя е директорията по подразбиране за стартиране на sql скриптове. Дадения Oracle Tool първо търси в директорията, от която е стартиран, след това в тази, която е указана от този параметър. Стойността по подразбиране е %ORACLE_HOME%\dbs. Много по-удобно е този параметър да има стойност %ORACLE_HOME%\rdbms\admin, поради често налагащото се в практиката стартиране на скриптове от тази директория
Следващите три параметъра се използват за ръчно настройване на изгледа на прозореца на SQL*Plus в графичен режим [изт. 34] (стартиращ се от изпълнимия файл sqlplusw.exe).
SQLPLUS_FONT – с негова помощ може да се укаже какъв фонт да се показва в прозореца. Ако потребителя иска да използва удебелен шрифт, то трябва да се добави думата bold в края на стойността. Пример: Courier New Bold
SQLPLUS_FONT_SIZE – указва големина на използвания фонт
SQLPLUS_FONT_CHARSET – ако потребителя иска SQL*Plus да показва специални символи, например български, то като стойност на този параметър могат да се настроят точно определен списък от езици, списъка може да се вземе от [изт. 1, стр. C-5].

Пример: ако има настроена и работеща база с кирилица, то потребителя може да сложи като език “Russian” (за съжаление няма български, но руския показва българските символи), то ето как може да изглежда SQL*Plus след настройка и на трите параметъра:



ORA_TZFILE – указва пътя до използвания файл за времевите зони. Ако трябва да се използва по-пълен такъв файл, то местоположението му се указва с този параметър
ORA_OCI_CACHE – има отношение към Oracle Database Cache опцията във Oracle9i AS, с помощта на този параметър апликацията в middle tier може да се възползва от тази възможност на Oracle. Когато с помощта на този параметър кешът е включен, се прави една допълнителна конекция (shadow) за всяка конекция към базата, която се прави, за повече информация [изт. 33]
ORA_OCI_UCBKPG - недокументиран
ORA_OCI_SHARED_MODE - недокументиран
ORA_OCI_NO_OPTIMIZED_FETCH - има отношение OCI/Precompiler Prefetch функционалността, за повече детайли за тази функционалност [изт. 9]. До версия 8.1.7 се поддържа, не е тестван за Oracle9i. За повече информация за самия параметър [изт. 31]
NETWORK – недокументиран, указва къде се намира network директорията
ORA_MIG_CLIENT - недокументиран
UTILITY_MSG – указва пътя до файловете с грешките, които се използват от някои Oracle апликации в DOS прозорец
EVENT_10235 – недокументиран – най-вероятно с помощта на подобните на този параметри могат да се настройват поведението на апликацията при възникването на определени събития. Няколко други, които също се използват: EVENT_10049, EVENT_32761, EVENT_10842
ORA_DEBUG_JDWP – указва адреса и порта на сървъра, който се използва за отдалечено дебъгване. Накратко, в Oracle9i има възможност за отдалечено дебъгване, която се осъществява чрез протокола JDWP (Java Debug Wire Protocol). Например, с помощта на Oracle9i JDeveloper може да бъде осъществено такова дебъгване с помощта на JDWP Listener, който осъществява връзката в такъв режим. За повече информация [изт. 13]

Параметри, Използвани от Oracle База Данни

Тук са включени параметрите, използвани от Oracle като база данни, както и от ORADIM Utility, или това са т.нар параметри на сървърската страна. Всички параметри, описани по-долу са Oracle Registry параметри, ако има някакви изключения, те ще бъдат указани за съответния параметър. Няма информация за поведението на недокументираните параметри.

ORACLE_BASE – беше разгледан по-горе
ORACLE_HOME – беше разгледан по-горе
TNS_ADMIN - беше разгледан по-горе

В случая горните два параметъра се използват от базата, но само ако са настроени в Registry. Или с други думи, по отношение на базата и ORADIM тези параметри имат поведение на Oracle Registry параметри.
ORACLE_BUNDLE_NAME – беше разгледан по-горе
RDBMS_ARCHIVE – указава местоположението на backup файловете с данни (data files), които могат да се използват в случай на възникнал проблем с базата
RDBMS_CONTROL - указава местоположението на backup контролните файлове (control files), които могат да се използват в случай на възникнал проблем с базата
RDBMS – недокументиран
ORA_CWD - беше разгледан по-горе
ORA_SID_CWD – същото значение като ORA_CWD, само че важи за базата с име SID
ORA_SID_PFILE – указва пълния път до инициализационния файл, използван от базата с име SID. Ако трябва да се използва SPFILE и потребителя иска да не редактира този параметър, то съществуващия файл може да се остави празен само с един ред, например: "SPFILE=D:\oracle\ora92\database\SPFILEsid.ORA"
Със следващите три параметъра се прави действителната връзка между service-а, който всички сме свикнали да гледаме OracleServiceSID и действителната работа на базата.
ORA_PWFILE – указва местоположението на password файла
ORA_SID_PWFILE – указва местоположението му на password файла за базата с име SID. Ако няма дефиниран такъв параметър, тогава Oracle търси параметър ORA_PWFILE, и ако не го намери чак след това отива в директорията по подразбиране: ORACLE_HOME\database\PWDsid.ORA
ORA_SID_AUTOSTART - беше разгледан по-горе
ORA_SID_SHUTDOWN – беше разгледан по-горе
ORA_SID_SHUTDOWN_TIMEOUT – беше разгледан по-горе
ORA_SID_SHUTDOWNTYPE – беше разгледан по-горе
Опитите да бъде повлияно на работата на ORADIM в команден прозорец чрез пренастройката на горните шест параметъра на ниво команден прозорец показват, че ORADIM взима предвид единствено стойностите, зададени в Registry.
ORA_DEBUG_NLS_SERVER_CONV – недокументиран, използва се от ORADIM
ORA_TZFILE – беше описан по-горе, в случая се използва от ORADIM
OSAUTH_PREFIX_DOMAIN – указва на базата дали да прави разлика между имената на един потребител (който се свързва с помощта на ОС автентикация) – дали името е локално, дали принадлежи на неговия домейн, или идва от друг домейн. Стойността по подразбиране е FALSE. От съображения за сигурност се препоръчва стойността му да бъде настроена на TRUE. За повече информация [изт. 22].
OSAUTH_X509_NAME – позволява на клиентите, съвместими с X.509 (международен стандарт за електронни сертификати) да се свързват към Oracle9i база данни. Използва се при настройка на Microsoft Active Directory и Oracle. За повече информация [изт. 8,стр. E-17]
OSAUTH_ENFORCE_STRICT – указва на базата, че може да допуска ОС автентикация от потребители, които са членове на локално дефинираната група ORA_USER или ORA_SID_USER. За повече информация [изт. 22]
Следващите три параметъра се използват от пакета UTL_HTTP, за да може да изпълнява HTTP заявки от SQL и PL/SQL. Тези параметри се използват за да може се използва конкретен и настроен Proxy Server в мрежата. Друг начин те да бъдат настроени е с помощта на процедурата UTL_HTTP.SET_PROXY. За повече информация [изт. 7, стр. 96-21].
Те може да се използват и при работа с XML, когато например се наложи да се използва DTD, което се намира някъде в мрежата или интернет.
NO_PROXY – ако този параметър е името на домейна, то не се използва HTTP Proxy Server за линковете (URL) в този домейн
HTTP_PROXY – адреса на използвания Proxy Server, понякога проблем възниква когато в адреса е пропуснато да се добави "http://" преди адреса. Ако се използва друг порт, различен от стандартния, то той също трябва да се добави
HTTP_PROXY_USER – потребителско име за дадения Proxy Server
HTTP_PROXY_PASS – парола за дадения Proxy Server
USE_SHARED_SOCKET – с помощта на този параметър, може да се реши проблема - как да става свързването към една база, при положение че тя стои зад firewall, позволяващ връзка само през няколко порта към нея. За повече информация [изт. 23]
Следващите седем параметъра се използват от базата за по-добрата и интеграция с Microsoft Transaction Server (MTS) функционалността на Microsoft Windows. MTS се използва за управление на отправените заявки за транзакции към апликации и към базата за сметка на клиента (или от негово име), който ги изпраща. За да се възползва от тази възможност на Windows, Oracle използва т.нар. "Oracle Services for MTS" – те изпълняват ролята proxy между базата

данни и “MTS Distributed Transaction Coordinator”. Тези services позволяват “connection pooling” на клиентската страна с помощта на MTS, като дори могат да работят с бази данни, които работят под която и да е операционна система. За повече информация [изт. 10]
ORAMTS_CONN_POOL_TIMEOUT – указва колко дълго една заявка може да няма активност и в същото време да стои отворена за повторно използване от клиента
ORAMTS_NET_CACHE_TIMEOUT – частта от конекцията, свързана с горния параметър отговаря за сесийни данни като потребителско име и парола. Частта от конекцията, свързана с този параметър отговаря за изпращането и получаването на данните, което е например една Oracle Net конекция. Отварянето на нова сесия изисква повече ресурси, отколкото използването на кешираната. Препоръчва се стойността на този параметър да е по-голяма от тази на горния параметър
ORAMTS_NET_CACHE_MAXFREE – указва максималния брой на свободни сесии на сървъра за поддържането на клиентски connection pooling
ORAMTS_OSCREDS_MATCH_LEVEL – указва какво да бъде нивото на сигурност върху конекциите, когато инициализационния параметър на базата OS_ROLES е избран да бъде TRUE
ORAMTS_SESS_TXNTIMETOLIVE – недокументиран преди Oracle 10G, но се използва и в Oracle9i – указва колко дълго да бъде кеширана една Oracle клиент/сървър конекция, инициирана с функцията OraMTSSvcGet()
ORAMTS_CP_TRACE_DIR – указва местоположението за файла, който ще трасира активността на oramts.dll
ORAMTS_CP_TRACE_LEVEL – указва нивото на трасиране
Следващите четири параметъра се използват при такива машини, на които Oracle работи с други апликации, или където на една и съща машина работят тестови и продукционни бази данни. Използването на тези параметри трябва да става заедно с настройването и на този инициализационен параметър: PRE_PAGE_SGA – този параметър указва на Oracle да зареди цялата SGA в паметта още при стартирането на базата (което не е поведението му по подразбиране). Настройването на този параметър може да забави стартирането на базата, но позволява тя да достигне пълния си капацитет веднага след стартирането си. За повече информация [изт. 25]
ORA_WORKINGSETMAX – максималното заемано място от Oracle.exe процеса като памет. Единицата, която се използва за настройка е MB
ORA_SID_WORKINGSETMAX – важи само за база с име SID
ORA_WORKINGSETMIN - минималното заемано място от Oracle.exe процеса като памет. Единицата, която се използва за настройка е MB. По-подходящият за използване в практиката.
ORA_SID_WORKINGSETMIN – важи само за база с име SID
Преди да се използват тези параметри, трябва да се провери дали page файла на Windows е достатъчно голям. Проведените тестове с горните параметри показват, че те се вземат предвид само ако се стартира базата с нейния Service или съответно с ORADIM. Върху тестова база и при извършвана една и съща активност (в случая – изчисляване на статистики за цялата база) – със и без включен параметър ORA_WORKINGSETMAX, резултатите показват че желания размер на заеманата памет не се спазва с абсолютна точност, но има разлика в достиганите максимални стойности (като заемана системна памет) на дадения oracle.exe процес.
ORA_XCPT – указва на Oracle да логва във файл своите т.нар. грешки идващи от ядрото на Oracle (core errors), стойността му по подразбиране е 1 – т.е. разрешено е. Понякога обаче е възможно при възникването на специфични проблеми, да започнат да се генерират огромен брой файлове и то с големи размери. За да се спре генерирането на тези файлове, този параметър трябва да се настрои 0 [изт. 11, стр. 104], както и [изт. 27]
ORA_SID_XCPT – важи само за база с име SID
ORACLE_SID_DISABLE_CHECK – недокументиран
ORA_LPENABLE – за да се използва възможността на Windows 2003 Server за т.нар. Large Page Support, то за да се разрешат т.нар. Large Pages за SGA на една Oracle база, то трябва този параметър да се настрои на 1.
ORA_SID_LPENABLE – за конкретна база
AWE_WINDOW_MEMORY и VLM_BUFFER_MEMORY – всеки потребител, който има памет по-голяма от 4GB и иска Oracle да използва тази възможност, трябва да използва и тези параметри. Те са свързани с опцията, наречена Address Windowing Extensions (AWE). За повече информация [изт. 28]
SKGFR_DELETE_ENABLED - недокументиран
ORA_SID_PGAPROTECT - недокументиран
ORA_MAX_ALLOC – използва се при проблеми, възникващи при опит да се използва по-голяма памет за Oracle на компютър, имащ 2 и повече GB памет. За повече информация [изт. 25]
ORACLE_PRIORITY – указва приоритета на нишките, работещи в Oracle процеса. По принцип такъв не се създава, а стойността му по подразбиране е тази, която операционната система дава на нишките във всеки един стартиран процес. С негова помощ могат да се укаже не само приоритета на дадена нишка, но и приоритета и на целия Oracle процес. Например, възможен начин за използване е при системи, генериращи огромен брой redo log файлове, в

такъв случай може да се увеличи приоритета на нишката LGWR [изт. 32], която по този начин да успее да се справи с писането в тези файлове.
ORACLE_AFFINITY – свързан е с работата на процесора и Oracle процеса. Позволява при многопроцесорни машини да се асоциира целия Oracle процес (или дадена нишка в него) към точно определен процесор. Например, асоциирането на DBW0 нишката [изт. 32] към един процесор може да спре прехвърлянето на управлението и от един процесор към друг и по този начин да окаже влияние върху бързодействието на базата.
Следващите осем параметъра са недокументирани, но явно имат отношение към работата на нишките, отговарящи за работата на Oracle процесите - DBW0, LGWR, PMON, SMON, CKPT и т.н. Ето някои от тях :
DBW - недокументиран
LGW - недокументиран
PM0 - недокументиран
SM0 - недокументиран
CKP - недокументиран
REC - недокументиран
CJQ - недокументиран
QMN - недокументиран
EVENT_32762 – недокументиран – най-вероятно с помощта на подобните на този параметри могат да се настройва поведението на базата данни при възникването на определени събития. Други, които биха могли да бъдат настройвани: EVENT_24910
Като цяло при работата си базата и ORADIM прочитат много параметри от Registry (включително всички NLS параметри), но в горния списък са включени само тези, за които след тестове, според конкретен източник е установено или се предполага (за недокументираните), че се използват от тях по някакъв начин.
Трябва да се има предвид, че има параметри (както за клиентската, така и за сървърската страна), които е възможно да са използвани в предишни версии и вече да не се поддържат. Те обаче продължават да се извикват от библиотеките, конкретния пример е ORA_NLS_CHARACTERSET_CONVERSION, който не се използва но пордължава да се чете от Registry от някои Oracle Tools и Utilities.

Подключ *HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\OracleMTSRecoveryService*
Подключ ProtidX – има три параметъра – Host, Name, Port, които указват съответно името на компютъра, протокола и порта използвани от Oracle Services за MTS
Подключ Setup – не е документиран

Ключ *HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services*
Тук могат да се открият всички Oracle Services и ако се налага да бъдат премахнати от “Services” то могат да бъдат изтрити съответните им ключове в registry.
Много по-добро решение е, поне за service-ите отговарящи за базата OracleServiceSID да бъдат премахвани с помощта на ORADIM utility.

Ключ *HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters*
Тези параметри не съществуват в registry и могат да бъдат използвани за да променят поведението по подразбиране на TCP/IP драйвера. Използването на тези параметри трябва да става внимателно, защото могат да предизвикат сериозни проблеми, ако не бъдат използвани по правилния начин. В същото време те могат наистина да се окажат полезни за решаването на определени проблеми, свързани с мрежата.
За да имат ефект тези настройки, то след всяка промяна компютъра трябва да бъде рестартиран.
TcpMaxDataRetransmissions
TcpMaxConnectRetransmissions
KeepAliveTime
KeepAliveInterval
DefaultTTL
За повече информация [изт. 29]

Интерактивност между SQL*PLUS и Batch Scripts

В тази точка се разглежда един често срещан проблем при писане на автоматични скриптове (batch scripts) - как да една PL/SQL процедура да върне резултата от изпълнението си към скрипта, след което той да направи или не определено действие. Например при грешка от изпълнението на съхранената процедура, batch скрипта да прекъсне

изпълнението си и да не продължи със следващите стъпки. Същото изискване може да има и ако някакъв SQL израз не се изпълни успешно.
За да стане това, трябва да се извика SQL*Plus, който да изпълни процедурата или SQL израза, като обаче тук трябва да се осъществи някаква комуникация между двете програми.
В този случай може да бъде много удобно използването на една системна обкръжаваща променлива, която се нарича ERRORLEVEL. Тази обкръжаваща променлива съдържа числовата стойност на EXIT кода на последната изпълнена програма. В този случай, за конкретния команден прозорец, в който се изпълнява batch скрипта, това именно се явява SQL*Plus.
Не всички програми под Windows връщат такъв код, но SQL*Plus има такава възможност. Командата EXIT в SQL*Plus се състои от параметри, които позволяват да се осъществи точно такава връзка между SQL*Plus и командния прозорец. Синтаксиса на EXIT командата е:

```
{EXIT|QUIT} [SUCCESS|FAILURE|WARNING|n|variable|:BindVariable] [COMMIT|ROLLBACK]
```

Вижда се, че дори параметъра "n" може да е напълно достатъчен за да се получи комуникацията, но възможността да се върне променлива може също да бъде много полезна, ако има изискване да се разбере стойността на даден запис в таблица.
С помощта на командата EXIT SQL.SQLCODE може да бъде върнат номера на последната възникнала грешка.
С помощта на още две команди: WHENEVER OSERROR и WHENEVER SQLERROR [изт. 6] изпълнението на един скрипт може да стане много гъвкаво.
Синтаксиса на двете команди е:

```
WHENEVER OSERROR
{EXIT [SUCCESS|FAILURE|n|variable|:BindVariable] [COMMIT|ROLLBACK]
|CONTINUE [COMMIT|ROLLBACK|NONE]}

WHENEVER SQLERROR
{EXIT [SUCCESS|FAILURE|WARNING|n|variable|:BindVariable]
[COMMIT|ROLLBACK]|CONTINUE [COMMIT|ROLLBACK|NONE]}
```

Ето един малък пример как може да бъде използвана тази функционалност:

```
whenever sqlerror exit 1;
declare
v_id number;
begin
select emp_id into v_id from employees
where emp_name = 'Scott';
end;
/
exit 0
```

Този SQL скрипт ще мине успешно само ако има служител с име Scott.
В batch скрипта може да се използва следния код:

```
@echo off
sqlplus rado/rado@ORALOCAL @check_employee.sql
goto ExitCode%ERRORLEVEL%
:ExitCodet0
echo on
 start some action…
exit
:ExitCode1
echo There is no such employee
exit
```

Друг вариант на batch скрипта може да бъде:

```
@echo off
sqlplus rado/rado@ORALOCAL @check_employee.sql
IF %ERRORLEVEL%==0 goto :NextOne
```

```
goto :NextTwo
:NextOne
start some action...
exit
:NextTwo
echo There is no such employee
exit
```

Тестове

За тестовете, които са проведени е използвана инсталирана Oracle9i база данни, версия 9.2.0.5. Използваната операционна система е Windows 2000 Professional с инсталиран Service Pack 4.

Обобщение

В практиката много често потребителите на Oracle под Windows се сблъскват с проблеми, които имат пряко или косвено отношение към разбирането на начина, по който Oracle комуникира със средата около себе си, и как тя може да му повлияе. Връзките между системните обкръжаващи променливи, настройките в Registry, тяхното поведение и начина, по който могат да бъдат използвани, продължава да предизвиква объркване дори и при задълбочено познаване на принципите на работа на базата данни. Въпреки, че официалната документация се опитва да покрие и да изясни всички неясноти, все още използването на всички тези параметри, начините им на настройване и как те могат да влияят на Oracle, предизвиква объркване. Липсата на цялостна официална документация, която да покрие изцяло тази тематика е също една от причините за създаването този документ. С него се прави опит да се събере и синтезира цялата налична информация по темата, както и да се изяснят всички неясноти, доколкото това е възможно. В този вариант на документа не са включени новите параметри, въведени в Oracle 10G, имащи отношение към темата, както и параметрите имащи отношение към RAC функционалността.

Използвани Източници

1. Janelle Simmons (2002) Oracle9i Client Installation Guide (Release 2) for Windows, Oracle Corporation
2. Craig B. Foch (2002) Oracle9i Database Administrator's Guide (Release 2) for Windows, Oracle Corporation
3. Craig B. Foch (2002) Oracle9i Database Getting Started (Release 2) for Windows, Oracle Corporation
4. Ruth Baylis (2002) Oracle9i Database Administrator's Guide (Release 2), Oracle Corporation
5. Cathy Baird (2002) Oracle9i Database Globalization Support Guide (Release 2), Oracle Corporation
6. Simon Watt (2002) SQL*Plus User's Guide and Reference (Release 2), Oracle Corporation
7. D.K. Bradshaw (2002) Oracle9i Supplied PL/SQL Packages and Types Reference (Release 2), Oracle Corporation
8. Laurel Hale (2002) Oracle Advanced Security Administrator's Guide (Release 2), Oracle Corporation
9. Jack Melnick (2002) Oracle Call Interface Programmer's Guide (Release 2), Oracle Corporation
10. Herbert Kelly, Mark Kennedy and Tamar Rothenberg (2002) Oracle Services for Microsoft Transaction Server Developer's Guide (Release 2) for Windows, Oracle Corporation
11. Scott Jesse, Matthew Hart, Mike Sale (2001) Oracle9i for Windows 2000 Tips & Techniques, Oracle Press
12. Thomas Kyte (2001) Expert One-on-One Oracle, Wrox Press
13. Mark Townsend (2002) Less Pain, More Gain - Use the new PL/SQL Features in Oracle9i Database, Oracle World Copenhagen
14. Oracle Support Team (2003) Obtaining Values of Oracle Registry Variables in Server Manager or SQL*Plus, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/showdoc?db=NOT&id=74001.1
15. Oracle Support Team (2001) Passing Parameters To SQL*PLUS in an MSDOS or NT CMD Prompt Setting, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/showdoc?db=NOT&id=2193.1
16. Oracle Support Team (2002) Oracle Home Selector, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=NOT&p_id=66464.1
17. Oracle Support Team (2002) Using Multiple ORACLE HOMES on Windows platform, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=NOT&p_id=73963.1
18. Oracle Support Team (2002) ORACLE_HOME in Oracle8i, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=NOT&p_id=70215.1

19. Oracle Support Team (2004) How To Identify Which Registry Hive Is Used By A Specific Oracle Executable, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_id=77412.1&p_database_id=NOT
20. Oracle Support Team (2002) How to Run regedit from Command Line to Dump Oracle Registry Entries, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/showdoc?db=NOT&id=197190.1
21. Oracle Support Team (2004) Oracle8i: Startup, Shutdown Related Registry Entries on WindowsNT & Windows2000, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=NOT&p_id=136214.1
22. Oracle Support Team (2003) Setup O/S Authentication, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=NOT&p_id=60634.1
23. Oracle Support Team (2004) How to configure USE_SHARED_SOCKET on Windows NT/2000, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=NOT&p_id=124140.1
24. Oracle Support Team (2004) Firewalls, Windows NT and Redirections, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_id=66382.1&p_database_id=NOT
25. Oracle Support Team (2004) Oracle Database and the Windows NT memory architecture - Technical Bulletin, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=NOT&p_id=46001.1
26. Oracle Support Team (2004) ORA_NLS (ORA_NLS32, ORA_NLS33, ORA_NLS10) Environment Variables Explained, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=NOT&p_id=77442.1
27. Oracle Support Team (2002) How To Disable Core Dumps On Windows NT/2000/XP, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=NOT&p_id=183540.1
28. Oracle Support Team (2004) Implementing Address Windowing Extensions (AWE) or VLM on Windows Platforms, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=NOT&p_id=225349.1
29. Oracle Support Team (2003) Windows NT Settings for TCP/IP Timeouts, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=NOT&p_id=208497.1
30. Oracle Support Team (2004) How to Perform Client-Side Tracing of Programmatic Interfaces on Windows Platforms, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=NOT&p_id=216912.1
31. Helen Schoone (2002) What is ORA_OCI_NO_OPTIMIZED_FETCH?, Metalink Support Forum, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=FOR&p_id=394011.999
32. David Colello (2002) Oracle9i Release 2 Database Architecture on Windows, Oracle Corporation
33. Oracle Support Team (2001) Oracle9iAS Database Cache Architecture, Metalink Support Site, URL: http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=NOT&p_id=117232.1
34. Simon Watt (2002) SQL*Plus Getting Started (Release 2), Oracle Corporation