

## Capitolul 2. SERVERUL DE BAZE DE DATE

Viața (destul de lungă, furtunoasă și, nu în ultimul rând, misterioasă) a „serverelor de baze de date” este marcată am putea spune de două evenimente „cheie” în istoria tehnologiilor legate de aplicabilele tranzacționale de prelucrare a datelor: pe de o parte afirmarea, confirmarea și contestarea *arhitecturilor client/server* care au luat în considerare separarea managementului datelor de aspectele de prelucrare concrete, și pe de altă parte proliferarea sistemelor de organizare a bazelor de date *relaționale*.

Ca urmare prin „definiție”, dar și istorie, SGBD-ul produs de compania Oracle se integrează în sistemele client/server și are în primul rând „valențe” relaționale (dar nu numai).

### 2.1.Ce este un server de BD ? Arhitectură generală

Un *server* în general reprezintă o entitate care poate oferi servicii în cadrul unei arhitecturi distribuite. Prin urmare serverul de baze de date reprezintă o entitate care oferă servicii legate de stocarea și gestiunea datelor. Aceste servicii rezultă de fapt din clasarea acestor servere în categoria SGBD-urilor.

Influența paradigmei client/server în domeniul aplicațiilor cu baze de date a avut ca efect migrarea de la tehnologiile legate de partajarea fișierelor la aplicațiile de rețea. Astfel, în primul caz, accesul la o bază de date însemna în primul rând transferarea pe client atât a executabilului SGBD-ului cât și a copiilor indecșilor fișierelor de date. Rezultatul: trafic considerabil în rețea, mecanisme slabe și „fragile” pentru tranzații și suport pentru concurență precară (mecanismele de blocaj asupra datelor realizându-se exclusiv la nivelul fișierelor fizice). *Serverele de baze de date* au însemnat înlocuirea transferului de fișiere cu transfer de mesaje conținând fraze SQL (în cazul cererilor clienților) și date (în cazul răspunsurilor la cererile clienților). Rezultatul: reducerea traficului pe rețea, gestiune centralizată și creșterea siguranței datelor, mecanisme îmbunătățite pentru asigurarea seriabilității tranzațiilor în condiții de concurență. Trebuie să recunoaștem că, respectând ceea ce afirmă teoria, chiar și sistemele de partajare a fișierelor sunt sisteme client/server, însă în acest caz paradigma este aplicabilă doar la nivelul serviciilor sistem (este vorba de servere de fișiere), pe când serverele de baze de date „aplică” modelul client/server pe un nivel superior, cel al aplicațiilor.

Pentru a corespunde cerințelor legate strict de gestiunea datelor în contextul sistemelor de aplicații client/server, arhitectura actuală SGBD-urilor arată ca în figura următoare.

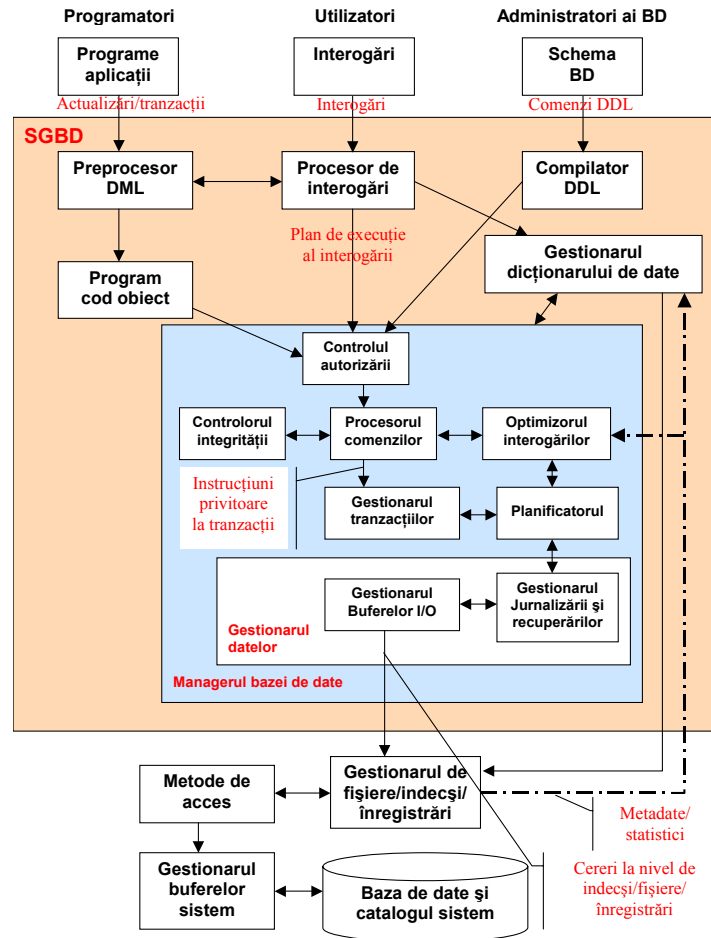


Figura 2.1. Arhitectura actuală a SGBD-urilor

Serverul bazei de date se circumscrie, dincolo de structura software-ului funcțional al SGBD-ului, și structurilor de organizare a datelor cu care lucrează acesta. În legătură cu aceste structuri trebuie făcută din capul locului distincția între: structurile logice de organizare a datelor (coloane, tabele, indecși, cluster-e etc.) și structurile interne de organizare a spațiului destinat stocării datelor (spații logice de stocare – tablespaces-uri – și fișierele sistemului de operare care materializează fizic aceste spații).

Din complicata și probabil și incompleta descriere de mai sus a arhitecturii SGBD-urilor, putem (și trebuie pentru a rămâne ceva cât de cât „persistent” și memoria RAM a cititorului) sintetiza *serviciile* furnizate de către SGBD-uri astfel:

- Stocarea, regăsirea și actualizarea datelor
- Descrierea datelor accesibile utilizatorilor
- Suport pentru tranzacții

- Servicii de control pentru concurență
- Servicii de recuperare a bazelor de date în cazul căderilor (avariilor)
- Servicii de autorizare a accesului de date (securitatea)
- Servicii de comunicare a datelor
- Servicii privind integritatea datelor
- Servicii care să asigure integritatea datelor
- Servicii utilitare pentru importul datelor, facilități de monitorizare și analiză statistică a activității bazei de date, de audit al bazei de date ș.a.

## 2.2. Probleme generale ale administrării unei baze de date

În legătură cu aspectele manageriale și tehnice privind datele deținute de o organizație se face de regulă o distincție funcțională (și formală) în *administrarea datelor*, *administrarea bazei de date* și *administrarea sistemului*. Aspectele considerate mai „manageriale” (decizii pe termen mediu și lung) cad în sarcina *administratorului de date* care ar trebui să se ocupe în primul rând cu stabilirea politicilor și standardelor generale privind datele pentru întreaga organizație. Detaliile tehnice privind proiectarea structurilor de date și implementarea lor în contextul unui anumit SGBD sunt considerate un atribut definitoriu al administratorului bazei de date. Există organizații (în general, de mari dimensiuni) care luând în considerare impactul implementării diferitelor tehnologii de infrastructură, printre care se găsesc și SGBD-urile, au și *un administrator de sistem* ce instalează și configurează SGBD-urile. Deși aceste trei funcții sunt frumos structurate și definite în teorie, în practică însă administratorul bazei de date este considerat „omul bun la toate” în privința datelor, personalizat sub forma unui *ins adesea morocănos, grăbit, imperativ și stresat, aflat și în postura „managerială” de a lua decizii cu privire la achiziționarea software-ului privind datele și în postura „tehnică” (sau tehnocrată) de a rezolva problemele oricărui utilizator de date sau dezvoltator de aplicații cu aceste date 24 de ore din 24, plătit totuși civilizat pentru această muncă (bineînțeles, în țările civilizate).*

Ca urmare a acestei unificări practice, atribuțiile funcției unui administrator de baze de date sunt separate în două categorii diferite: prima desemnează rolul managerial, iar cealaltă rolul tehnic:

### 1) Rolul managerial al administratorului unei baze de date

Suportul pentru utilizatorii finali – culegerea și satisfacerea cerințelor utilizatorilor finali (rezolvarea necesităților informaționale, asigurarea calității și integrității datelor și aplicațiilor pe care le accesează, câștigarea încrederii – sau neîncrederii – utilizatorilor finali, asigurarea suportului tehnic și instruirii acestora cu privire la funcțiile și utilizarea programelor sau utilităților puse la dispoziție de către SGBD-ul instalat);

Construirea și aplicarea politicilor, procedurilor și standardelor. Un DBA trebuie "să definească, comunice și să aplice" proceduri care vor acoperi arii de acțiune cum sunt: modelarea și proiectarea bazei de date, documentarea bazei de date, testarea, securitatea și integritatea acestora plus crearea copiilor de siguranță și restaurarea ei.

Asigurarea securității și integrității bazei de date – adică managementul accesului utilizatorilor (definirea utilizatorilor, atribuirea parolelor, definirea privilegiilor de acces), definirea view-urilor pentru protejarea și controlul domeniului de acces la date al utilizatorilor autorizați.

Crearea copiilor de siguranță și restaurarea bazei de date, plus administrarea căderilor și pierderilor de date. În acest cadru se impun: măsuri privind backup-ul bazei de date, cu alte cuvinte, crearea periodică a copiilor de siguranță a datelor, identificarea corectă a acestora, stocarea corectă și sigură a lor, protecția fizică atât în ceea ce privește software-ul cât și hardware-ul, și măsurile privind managementul tranzacțiilor și concurenței în BD.

Utilizarea datelor distribuite – pentru asigurarea integrității tranzacțiilor cu aceste date și pentru asigurarea unor performanțe rezonabile privind timpii de interogare.

## 2) Rolul tehnic al administratorului unei baze de date

Selectarea, evaluarea și instalarea SGBD-ului ținând cont de factori precum: modelul de date (relațional, obiectual), scalabilitatea în ceea ce privește capacitatea de stocare, suportul pentru dezvoltarea aplicațiilor, caracteristicile de securitate și integritate a datelor, utilitățile de efectuare a copiilor de siguranță (backup) și recuperare a datelor (recovery), controlul accesului concurent, performanța, instrumentele de administrare, suportul pentru distribuirea datelor, portabilitatea etc.

Proiectarea și implementarea bazelor de date și aplicațiilor.

Testarea și implementarea bazelor de date și aplicațiilor.

Operarea cu SGBD-ul, utilitățile și aplicațiile acestuia. Aceste operații, efectuate apelând la utilitățile de bază ale SGBD-ului, pot fi împărțite în patru arii importante: suportul software al SGBD-ului, monitorizarea performanțelor și optimizarea bazei de date, efectuarea activităților de *backup* și *recovery*, monitorizarea și auditul securității.

Menținerea SGBD-ului, a utilităților și a aplicațiilor în starea funcțională, ceea ce presupune inclusiv actualizarea lor funcție noile versiuni. Uneori se pot lua măsuri mai radicale de migrarea spre un alt SGBD ceea ce implică din partea administratorului eforturi specifice cu privire la migrarea și conversia datelor.

## 2.3. Serverul de baze de date Oracle

Până acum am privit lucrurile oarecum detașat de o tehnologie proprietară anume și am irosit astfel vreo patru pagini în încercarea de a forma o imagine

generală, sau să-i zicem principială, asupra a ceea ce reprezintă un server de baze de date și personalul (sau impersonalul) din preajma lui. Se cuvine însă să dăm cezarului ce-i al cezarului, și lui Oracle ce-i al lui Oracle. Prin urmare în continuare vom „pune lupa” (nu microscopul electronic) asupra serverului Oracle 8/8i/9i ...

### 2.3.1. Structurile de organizare a elementelor unei baze de date oferite de serverul Oracle

În cazul Oracle structurile de memorie internă și procesele care asigură funcționalitatea clasică a SGBD-ului iau forma „concentrată” a unei *istanțe Oracle*. La nivelul sistemului de operare această structură se materializează sub forma unui serviciu distinct cu numele *OracleService{SID}*, unde SID reprezintă numele personalizat al instanței.

Din punct de vedere logic serverul Oracle pune la dispoziția utilizatorilor următoarele structuri de organizare logică a datelor:

Tabele, coloane, restricții

Tabelele constituie mecanismul de stocare a datelor specific unei baze de date Oracle și conțin un set fix de coloane ce descriu attribute ale relațiilor (în cazul modelului relațional) sau obiectelor (în cazul modelului orientat obiect) din schema logică.

Caracteristicile unei coloane se referă la: tipul de date și lungimea (plus precizia și scala în cazul în care tipul de date este NUMBER). Tipurile de date de bază suportate în sistemul Oracle sunt:

pentru șiruri de caractere: CHAR, de lungime fixă și VARCHAR2, de lungime variabilă;

pentru numere: NUMBER;

pentru dată calendaristică DATE;

pentru date binare RAW;

pentru obiecte de mari dimensiuni: LONG, LONG RAW, BLOB, CLOB, BFILE.

Odată cu crearea unei tabele se pot specifica, printr-o serie de clauze, diferite restricții, cum ar fi:

chei primare – PRIMARY KEY;

restricții pentru volorile nenule – NOT NULL;

valori implicite – DEFAULT;

restricții de unicitate, diferite de cheia primară – UNIQUE;

restricții de cheie străină – FOREIGN KEY.

Restricțiile implementate prin clauzele comenzii SQL CREATE TABLE mai sunt numite și *restricții neprocedurale* pentru că se referă la restricții formalizate prin expresii foarte exacte (vezi capitolul 4). Implementarea restricțiilor de integritate se poate face și sub formă de reguli active cunoscute de obicei sub numele de

declanșatoare (*trigger-e* – vezi capitolul 10). Acest gen de restricții sunt considerate *procedurale*.

Declanșatoarele sunt proceduri ce se execută atunci când are loc un eveniment specific bazei de date ce afectează înregistrările unei anumite tabelă. Ele pot fi folosite pentru a implementa reguli sau restricții referențiale mai complexe care nu pot fi formalizate prin clauzele comenzii `CREATE TABLE`. De asemenea, mai pot fi folosite și pentru a aplica anumite caracteristici de securitate sau anumite opțiuni de *auditare*. În versiunile Oracle există de regulă două categorii principale de declanșatoare la nivel de tabelă:

la nivel de frază SQL – se declanșează o singură dată la nivel de frază SQL;

la nivel de linie – se declanșează pentru fiecare linie din tabelă afectată de fraza SQL executată.

Pentru implementarea logicii aplicației, într-un sistem Oracle sunt disponibile următoarele structuri procedurale: *proceduri stocate*, *funcții* și *pachete* (*package-uri*).

*Procedurile stocate* reprezintă blocuri ce cuprind cod sursă (PL/SQL în cazul Oracle) păstrate în dicționarul bazei de date, apelabile din declanșatoare sau de către eventualele aplicații implementate.

*Funcțiile* se aseamănă cu procedurile, însă spre deosebire de acestea din urmă sunt obligate să returneze anumite valori programelor ce le apelează.

*Pachetele* sunt structuri mai complexe fiindcă sunt folosite pentru a organiza sau reuni procedurile și funcțiile în grupuri logice. Anumite elemente dintr-un pachet (variabile, proceduri, funcții) pot fi declarate publice sau private. Elementele publice sunt accesibile din afara package-ului, în timp ce cele private sunt accesibile numai procedurilor sau funcțiilor din interiorul package-ului.

#### Tipuri abstracte de date

Tipurile abstracte de date sunt *tipuri de date definite de utilizatori*. Ele se bazează pe tipurile scalare definite mai sus și pot fi folosite la definirea coloanelor din tabele de date relaționale sau la definirea atributelor obiectelor stocate în *tabelele de obiecte*. O tabelă de obiecte reprezintă o tabelă în care toate liniile reprezintă obiecte ce posedă un identificator unic (OID), generat independent de valorile atributelor. În acest context se pot crea referințe între liniile din tabelele relaționale și obiectele din tabelele de obiecte, sau între obiectele stocate în tabelele de obiecte.

Alte structuri interne specifice Oracle sunt *tabelele încapsulate* (*nested table*) și *tablourile de mărime variabilă* (*varying arrays*). O tabelă încapsulată poate reprezenta o coloană dintr-o tabelă relațională care conține mai multe valori pentru o singură linie. Un tablou variabil este asemănător cu o tabelă încapsulată, însă este limitat sub aspectul numărului de linii pe care le poate conține (pentru detalii, vezi capitolul 8). Aceste tipuri de structuri reprezintă de fapt *constructori colectori* putând implementa entități aflate în relație una-la-mai-multe ce se regăsesc de obicei, prin normalizare, în două tabele relaționale distincte.

În ce privește implementarea schemei logice a bazei de date în locații distribuite, soluția în sistemele Oracle se bazează pe *partiții*. Astfel, schema de fragmentare a unei baze de date distribuite poate fi implementată în Oracle folosind partițiile. Acestea porționează tabelele de date după anumite criterii precizate și le poziționează în localitățile stabilite prin schema de alocare.

### Clustere

Tabelele care sunt accesate frecvent împreună, în general cele aflate într-o relație una-la-mai-multe, pot fi stocate fizic împreună. Pentru a le stoca în acest mod este creată o structură specială numită *cluster* (sau dacă vreți „ciorchine”), care va păstra în aceleași locații liniile legate între ele ale ambelor tabele. Scopul constă în minimizarea fluxului de I/E generat de interogările și actualizările datelor. Coloanele prin care este făcută legătura dintre cele două tabele formează cheia cluster-ului. Această cheie este folosită la generarea index-ului clusterului, valorile acesteia fiind stocate o singură dată pentru multiplele tabele ale căror linii sunt stocate împreună în cluster.

### Tabele virtuale - view-uri

Ca structură o tabelă virtuală este în mare parte similară cu o tabelă obișnuită: este formată din linii și coloane. În schimb, deși se pot interoga sau actualiza date, un view nu stochează fizic date. Acestea se găsesc în tabelele pe baza cărora s-a creat respectivul view. Conceptual, un view poate fi înțeles ca o “mască” suprapusă peste tabelele de bază prin mecanismul unei fraze `SELECT`. Ceea ce se stochează în bază este definiția view-ului, adică interogarea pe care se bazează, formatul coloanelor, privilegiile acordate. Spre deosebire de tabele, view-urile nu pot fi indexate.

Din punct de vedere al proiectării tabelele virtuale pot fi invocate ca mecanisme pentru maparea schemei externe pe schema conceptuală implementată prin structura internă a bazei de date. În acest context, view-urile pot fi folosite și pentru aplicarea anumitor politici de securitate la nivelul liniilor sau coloanelor din tabele.

### Indecși

Un index reprezintă o structură utilizată de server-ul bazei de date pentru a regăsi rapid înregistrările din tabelele bazei de date. Există trei tipuri de indecși: indecși pentru tabele, indecși pentru clustere și indecși bitmap.

Indecșii conțin o listă de valori de intrare, fiecare cuprinzând o valoare a cheii după care au fost definiți și un RowID. Un RowID reprezintă un tip de date special ale cărui valori sunt astfel construite încât să se poată deduce locația fizică a fiecărei înregistrări: fiecare linie dintr-o tabelă are un unic RowID. Intrările unui index de tabelă sau cluster sunt definite într-o bază de date Oracle utilizând mecanismul B\*-tree care garantează cel mai rapid acces (cea mai scurtă cale) la o valoare cheie. Fluxul de intrare-ieșire (I/E) cerut pentru a găsi o valoare cheie este

minim, iar odată găsită, este citit RowID-ul pereche cu care se va avea acces direct la înregistrarea corespunzătoare din tabela de bază.

Indecșii sunt creați automat, implicit, la crearea unei tabele prin clauze de tip `UNIQUE`, `PRIMARY KEY` care implementează anumite restricții de integritate. Crearea explicită a indecșilor, în special din rațiuni de optimizare a anumitor interogări, se realizează printr-o comandă `CREATE INDEX`.

#### Alte structuri Oracle

În dicționarul unei baze de date Oracle se mai pot găsi și definiții de *secvențe*. Secvențele au fost create pentru a ușura efortul de programare, furnizând liste secvențiale de valori unice. Când este interogată pentru prima dată, o secvență va returna o valoare predefinită. Fiecare interogare succesivă asupra secvenței va produce o valoare care crește (sau descrește) cu un anumit increment. Secvențele pot fi ciclice sau pot crește succesiv până când este atinsă o valoare maximă (sau minimă) specificată.

Pentru a identifica complet un obiect dintr-o bază de date trebuie specificat cel puțin numele schemei (deținătorul obiectului) și numele acestuia (la care se adaugă în bazele de date distribuite numele mașinii – al calculatorului gazdă - și numele serverului de baze de date – al instanței). Pentru a simplifica acest proces, pot fi create *sinonime* care să puncteze direct către respectivele obiecte din baza de date, astfel că utilizatorii nu trebuie să cunoască decât numele sinonimului. Sinonimele publice pot fi utilizate de către toți utilizatorii bazei de date, în timp ce cele private sunt vizibile doar de către cei care le-au creat și prin urmare le dețin.

Bazele de date Oracle au posibilitatea să facă referințe la date care sunt stocate în afara bazei de date locale. În mod normal, pentru a realiza astfel de referințe trebuie specificat numele complet al obiectului stocat la distanță. Printr-un sinonim se poate preciza doar numele obiectului și al deținătorului, neputându-se indica și locația. Pentru a specifica întreaga cale de acces pentru un obiect stocat într-o bază de date la distanță este necesară crearea unui *legături de baze de date*. Când este creat un astfel de obiect, care poate fi la fel ca un sinonim – public și privat, se precizează contul, parola și numele de serviciu pentru baza de date aflată la distanță, conexiunea devenind astfel transparentă.

### 2.3.2. Structurile logice de stocare a datelor

O bază de date este împărțită în general în mai multe unități logice de stocare numite *tablespace*-uri. Rațiunea acestei divizări poate fi legată de exemplu de gruparea informațiilor stocate în baza de date funcție de compartimentele organizației sau de ariile de acțiune ale sistemelor de aplicații. Spațiul de stocare administrat de aceste tablespace-uri provine din fișierele de date care fac parte din baza de date. Regula este ca un fișier de date să funizeze spațiu de stocare unui singur tablespace.



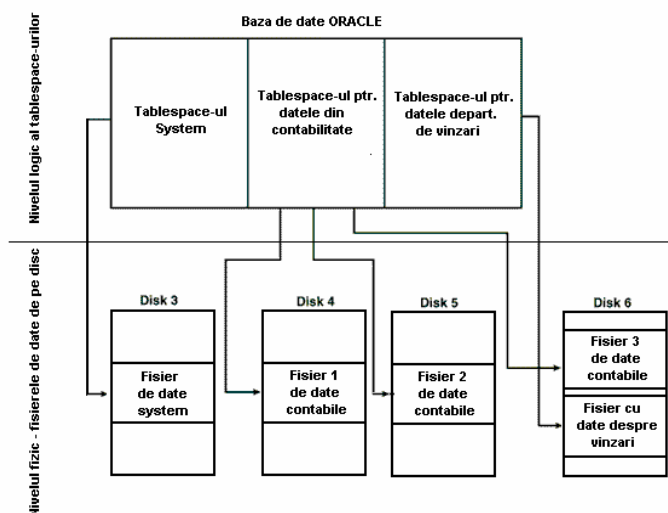


Figura 2.2. Tablespace-uri într-o bază de date Oracle

Spațiul utilizat de o bază de date Oracle este controlat prin următoarele structuri logice:

*blocul de date* – unitatea de stocare Oracle cea mai mică (unitatea de bază). Dimensiunea acestuia reprezintă un multiplu al dimensiunii blocului sistemului de operare (parametrul DB\_BLOCK\_SIZE din fișierul de parametri).

*extent-ul* – constă din mai multe blocuri de date continue pe disc. Poate fi considerat unitatea de alocare a spațiului pentru segmentele de date.

*segmentul* – constă dintr-un set de extenturi utilizate pentru a stoca un anumit tip specific de date.

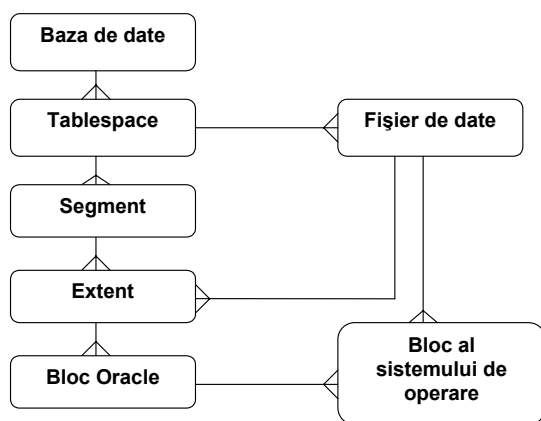


Figura 2.3. Diagrama E-R pentru relațiile dintre structurile logice și fizice de stocare

Segmentele pot fi socotite corespondentul fizic al obiectelor din baza de date. Segmentele stochează date (tabele, indecși etc.), spațiul de stocare necesar fiind preluat din tablespace-urile bazei de date. Alocarea spațiului de stocare pentru un segment de date se face prin intermediul extent-urilor care sunt formate din seturi de blocuri de date adiacente – continue pe disc. Atunci când este creat un segment, acesta constă din cel puțin un extent. Odată cu creșterea dimensiunii segmentelor, noi extent-uri le sunt alocate din tablespace-urile respective.

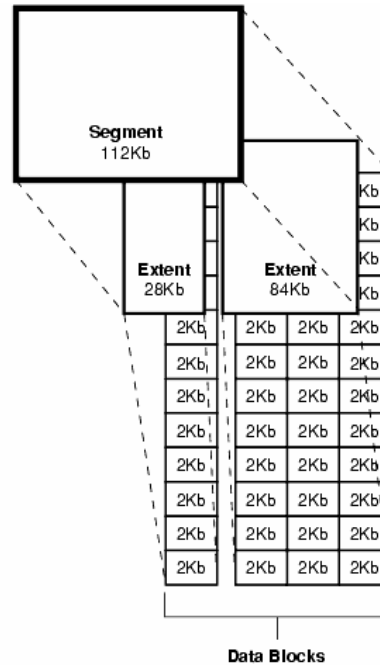


Figura 2.4. Formarea segmentelor din extent-uri și a extenturilor din blocuri de date

#### *Tipuri de segmente de date*

Am menționat că segmentele reprezintă de fapt obiectele ce ocupă spațiul din baza de date. Din acest punct de vedere se întâlnesc următoarele tipuri de segmente:

*Tabelele* neclusterizate și nepartitionate sunt cel mai cunoscute structuri de stocare a datelor într-o bază de date. Datele dintr-o tabelă sunt stocate fără o ordine precis determinată, astfel că administratorul bazei de date nu prea are control asupra locației fizice a liniilor din blocurile de date.

*Partițiile din tabele.* Din motive de scalabilitate și disponibilitate, o tabelă relațională poate fi stocată în mai multe partiții, fiecare din ele putându-se afla într-un alt tablespace și astfel chiar într-o altă locație. Dacă o tabelă este partitionată, fiecare partiție este un segment separat cu parametri de stocare specifici.

*Cluster-e.* Un cluster (un „ciorchine”) conține una sau mai multe tabele (grupate după o coloană comună – key-column). Tabelele dintr-un cluster aparțin aceleiași segment și deci au parametri de stocare comuni.

*Indecși.* Fiecare index are un segment de date separat, astfel că dacă pentru o tabelă sunt definiți trei indecși, atunci sunt necesare trei segmente diferite. Pentru un index partiționat fiecare partiție corespunde unui segment separat.

*Segmentele rollback.* Segmentele rollback sunt utilizate de tranzacțiile care modifică datele din baza de date. Înainte de a face o modificare asupra datelor, procesul server salvează vechea valoare în segmentul de rollback. Această imagine este utilizată pentru:

- anularea (Undo) modificărilor dacă tranzacția este abandonată (sau rollback-uită cum se mai spune adeseori);
- împiedicarea celorlalte tranzacții să „vadă” modificările necomise determinate de o frază DML (asigurarea consistenței la citire);
- refacerea bazei de date într-o stare consistentă în cazul căderilor.

*Segmente temporare.* Pentru comenzi SQL cum sunt CREATE INDEX, SELECT DISTINCT și SELECT GROUP BY, serverul Oracle va trebui să efectueze sortări suplimentare în memorie, sortări care sunt, de regulă, mari consumatoare de spațiu. Când apare o astfel de sortare (cum ar fi în cazul în care este construit un index pe o tabelă de dimensiuni mari) sunt stocate pe disc rezultatele intermediare. În asemenea cazuri sunt create segmente temporare.

Alte tipuri de segmente se referă la: *segmente LOB* pentru coloanele din tabele în care se stochează obiecte mari (LOB – Large Binary Objects adică documente, imagini, secvențe video), *segmente index LOB* create pentru segmentele LOB propriu-zise, *tabele încapsulate* (nested table) pentru coloanele definite ca tabele în tabelele relaționale obișnuite și *segmentul bootstrap* necesar la crearea bazei de date (rularea scriptului sql.bsq) și la inițializarea zonei *data dictionary cache* din *Shared pool* în momentul deschiderii bazei de date de către o instanță (scuzați avalanșa de anglicisme și tehnicisme, dar sperăm ca următoarele două paragrafe să mai aducă ceva lumină, dacă nu mai mult întuneric).

### 2.3.3. Structurile fizice de stocare a datelor

O bază de date Oracle reprezintă, de fapt, structuri fizice fiind compusă din fișiere ale sistemului de operare de următoarele tipuri:

*Fișiere de date:* stochează dicționarul bazei de date, obiectele utilizatorilor, imaginile inițiale ale datelor schimbate de tranzacțiile curente (segmentele rollback). Există cel puțin un astfel de fișier rezervat, bineînțeles, catalogului bazei de date.

*Fișierele de jurnalizare, sau redo log:* conțin înregistrări ale schimbărilor făcute în baza de date pentru a o putea reconstrui în cazul unor căderi. Sunt necesare cel puțin două astfel de fișiere.

*Fișierele de control:* conțin informațiile necesare menținerii și verificării integrității bazei de date. Acestea de fapt dau informațiile concrete asupra

localizării celorlalte fișiere din baza de date și asupra stării lor. Este necesar cel puțin un astfel de fișier.

În momentul creării bazei de date, automat se alocă și formează spațiul pe disc, care însă nu conține încă datele utilizatorilor. Cu alte cuvinte, serverul Oracle rezervă pe disc spațiu pentru viitoarele segmente alocate în respectivele tablespace-uri. Un tablespace poate fi format din unul sau mai multe fișiere de date. Cum segmentele de date sunt alocate într-un anumit tablespace, atunci se poate presupune că schema unui utilizator se poate întinde în mai multe fișiere de date. Dimensiunea fișierelor de date poate fi modificată ulterior, funcție de necesitățile bazei de date.

Fișierele de jurnalizare (redo log) sunt utilizate pentru minimizarea pierderilor de date într-o bază de date. Ele sunt valorificate în momentul în care o instanță cade („colapsează”), iar modificările tranzacțiilor declarate comise nu sunt scrise încă pe disc. La repornirea instanței aceste fișiere sunt citite pentru refacerea bazei de date în stare consistentă, recuperând toate tranzacțiile comise și anulând toate tranzacțiile necomise.

Un fișier de jurnalizare face parte dintr-un *grup redo log*. Membrii unui grup *redo log* sunt identici din punct de vedere al dimensiunii, numărului secvenței log<sup>1</sup> și informației stocate, procesul LGWR (vezi paragraful următor) scriind simultan în toate fișierele unui astfel de grup. Aceste grupuri sunt utilizate într-o manieră circulară: după ce ultimul grup este plin se trece la suprascrierea primului.

Alte fișiere esențiale în funcționarea serverului Oracle, dar în afara bazei de date, sunt:

- *fișierul de parametri*: definește caracteristicile instanței Oracle.
- *fișierul de parole*: autentifică utilizatorii privilegiați ai bazei de date.
- *fișierele redo log arhivate*: copii offline ale fișierelor *redo log* care ar putea fi necesare în procesul de restaurare în cazul unor căderi ale bazei de date.

#### 2.3.4. Structura internă a instanței

Când un utilizator cere o conexiune la baza de date printr-un *proces utilizator* (sau proces client, de exemplu SQL\*PLUS), cererea acestuia va fi preluată în prima fază de un serviciu numit *listener*<sup>2</sup>. Sarcina acestuia este să interpreteze *string*-ul de conectare (numele bazei de date furnizat de utilizator) și să verifice dacă există un serviciu-instanță corespunzător. Dacă instanța indicată există și este activă, se creează câte un *proces server* pentru fiecare conexiune (într-o arhitectură dedicată) care va prelua sarcina administrării acesteia și va satisface cererile exprimate prin comenzi SQL ale utilizatorului. În cazul unei arhitecturi MTS (Multi-Threaded-Server) un singur proces server (*shared server process*) poate deservi mai multe conexiuni simultane.

---

<sup>1</sup> log sequence number – numărul alocat de serverul Oracle la fiecare nouă scriere în grupul redo log

<sup>2</sup> la nivelul sistemului de operare, numele acestuia este, de regulă, *Oracle{HOME}TNSListener*

*Instanța Oracle* reprezintă elementul esențial care asigură funcționalitatea specifică SGBD-ului. La nivelul sistemului de operare se materializează sub forma unui *serviciu* specific care asigură practic funcționalitatea atribuită *serverului* de baze de date. Concret, aceasta este formată dintr-o zonă de memorie internă - numită *system global area* - și o serie de *procese background* care funcționează într-o manieră colaborativă.

**System Global Area** conține date și informații de control pentru serverul Oracle. SGA este formată din următoarele structuri:

*Shared SQL Pool* - stochează informații cum ar fi, pe de o parte, cele mai recente fraze SQL executate (*Library Cache*), adică textul SQL, arborele de analiză și planul de execuție, și, pe de altă parte, datele din dicționarul de date cele mai recent utilizate (*Data dictionary cache*), adică definiții de tabele și coloane, nume de utilizatori, parole și privilegii. Dimensiunea componentei Shared Pool din instanță este determinată de parametrul SHARED\_POOL\_SIZE din fișierul de parametri.

*Database Buffer Cache (DB Cache)*, bufferele pentru date - este zona în care sunt stocate blocurile de date provenind din segmentele de date cel mai recent utilizate. Mărimea fiecărui buffer din această zonă este egală cu mărimea unui bloc de date specificată prin parametrul DB\_BLOCK\_SIZE, iar numărul de buffer-e este precizat prin parametrul DB\_BLOCK\_BUFFERS. Pentru a găzdui noi blocuri de date în *DB Cache*, Oracle folosește un algoritm prin care sunt trimise pe disc blocurile de date care nu au fost accesate recent (LRU - Last Recent Use) pentru a face loc celor noi.

*Redo Log Buffer*, bufferele pentru înregistrările *redo log* - memorează modificările făcute asupra datelor din BD de către instanță. Un *buffer redo log* stochează înregistrări *redo* (redo records) care în mare conțin identificatorul (adresa) blocului de date modificat, locația modificării (în blocul de date) și noua valoare. Un astfel de buffer este circular, adică este refolosit pe măsură ce se completează, nu însă înainte de a trimite înregistrările *redo* în fișierele *redo log*. Mărimea (în bytes) a buffer-elor *redo log* este stabilită prin parametrul LOG\_BUFFER din fișierul de parametri.

**Procesele de background** execută funcțiile obișnuite necesare satisfacerii cererilor utilizatorilor concurenți, asigurând integritatea datelor și menținând performanțele sistemului. Cele mai importante sunt:

*Database Writer (DBWR)*: responsabil de scrierea datelor modificate din *buffer cache* pe disc, în fișierele bazei de date. Prin urmare, blocurile de date care conțin înregistrările afectate de comenzile DML (trimise spre execuție de către utilizatori) sunt încărcate mai întâi în bufferele interne (buffer cache), dacă nu se găsesc deja acolo, după care sunt modificate. Procesul DBWR va scrie (la momentul oportun) aceste buffere înapoi pe disc în fișierele de date de unde au fost citite, asigurând astfel existența unui număr suficient de buffere libere (care nu conțin blocuri de date modificate sau care pot fi suprascrise atunci când sunt cerute noi blocuri de date de pe disc) în *DB cache*.

*Log Writer (LGWR)*: are sarcina de a scrie în fișierele redo log înregistrările redo din buffer-ul redo log.

*System Monitor (SMON)*: are în primul rând funcția să verifice consistența bazei de date și să inițieze restaurarea acesteia în momentul când este pornită instanță și este deschisă BD. Deasemenea mai are rolul și de a "curăța" baza de date prin eliminarea obiectelor tranzacționale care nu mai sunt necesare.

*Process Monitor (PMON)*: eliberează resursele blocate când "cade" unul din procesele utilizator (pentru ca procesele de pe server care le deserveșc să nu blocheze inutil resurse). La fel ca și SMON, PMON verifică periodic dacă este necesar să intervină pentru menținerea unei funcționări corespunzătoare a instanței.

*Procesul Checkpoint (CKPT)*: este responsabil de actualizarea informațiilor despre starea bazei de date de fiecare dată când schimbările din DB cache sunt înregistrate definitiv în baza de date.

*Procesul ARCH*. Procesul LGWR scrie în fișierele redo log active (on line) într-o manieră circulară, astfel că în momentul în care un astfel de fișier este plin scrierea se face în cel de al doilea ș.a.m.d. După ce ultimul fișier redo log on line este plin se reîncepe scrierea de la primul. Dacă serverul Oracle este în modul arhivare (ARCHIVELOG MODE) atunci fiecare fișier redo log, înainte de a fi suprascris, este copiat într-o locație specială. Această sarcină este efectuată de procesul ARCH-iver.

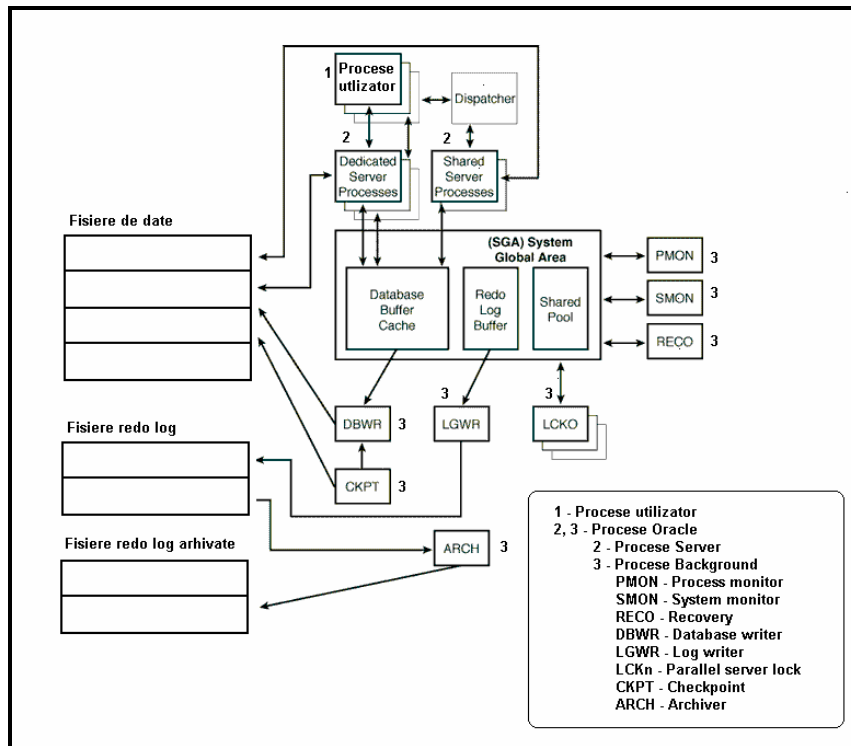


Figura 2-2 Serverul Oracle: Instanța (SGA și procesele background) și fișierele care compun baza de date

*Procesele server* – sunt dedicate administrării conexiunilor la baza de date și inițiază fluxuri de I/E între instanța bazei de date și fișierele de date în scopul satisfacerii cererilor utilizatorilor.

### 2.3.5. Instalarea software-ului Oracle pentru serverul bazei de date

În paragraful anterior am trecut în revistă structura stufoasă, nerecomandată cardiacilor, a serverului Oracle (instanța plus fișierele ce formează baza de date) fără însă a furniza vreun indiciu asupra modului cum vor putea fi obținute respectivele elemente. În aceste condiții este normal ca pe umerii unui nou venit în lumea serverelor de baze de date să apese un sentiment de împovărare și frustrare, dată fiind avalanșa de detalii tehnice (prea) deseori criptice legate de intimitatea SGBD-ului. Vestea bună este că nivelul asistenței în domeniul instalării software-ului specific SGBD-ului și obținerii unei baze de date „vioaie” și funcționabile este destul de bun. Vestea proastă este că, și pentru un dezvoltator de aplicații cu baze de date, o parte din detaliile tehnice amintite sunt absolut necesare, chiar dacă scopul real nu este administrarea BD, ci obținerea unei aplicații viabile cu performanțe care să-i asigure fezabilitatea.

Prin urmare să purcedem la trecerea în revistă a pașilor implicați de instalarea software-ului Oracle. Câteva cerințe premergătoare însă sunt neapărat necesare:

Faceți rost de CD-urile cu Oracle (9i) – cel mai bine cu tot cu licență (și fără banii necesari achiziționării ei) sau procurați-vă (doar în scopuri „explorative” și pentru un utilizator solitar) kit-urile disponibile la <http://otn.oracle.com/> corespunzătoare sistemului de operare „îndrăgit” de calculatorul d-voastră;

Mai este nevoie bineînțeles și de un calculator (proprietate personală sau în leasing) cu ceva RAM consistent (un 128 măcar, un 256+ ar fi și mai bine) măcar cu Windows2000 (sau un sistem de operare incompatibil cu Windows-ul, dar compatibil cu kit-ul pe care îl dețineți);

Timp de încercare și ceva răbdare (dacă nu și tutun).

Lăsând tonul șugubăț la o parte, este bine să consultați mai înainte detaliile privind cerințele minime din documentația de instalare: primul disc al kit-ului de instalare conține un document *Welcome.html* de unde mergând pe link-ul *Documentation* veți putea avea acces la *Oracle 9i Database Installation Guide*.

Utilitarul de instalare al Oracle se lansează fie din ecranul de întâmpinare implicit al *autorun*-ului primului disc al kit-ului, fie direct prin executabilul *setup.exe*. Fereastra „ochioasă” a *Oracle Universal Installer*-ului pe care o veți obține astfel este creată integral în Java, așa încât prima lansarea s-ar putea să dureze puțin până când va fi instalat în prealabil JRE<sup>3</sup>-ul, dacă nu este deja instalat. Normal, primul lucru pe care îl putem face este să părăsim ecranul de „bună-întâmpinare” și să mergem (*Next*) îl al doilea ecran, dar care este primul pas obligatoriu al procesului de instalare:

Etapă de specificare a căilor - **File Locations**. În această etapă ne întâmpină trei rubrici: *source path*, *destination name* și *destination path*. Prima rubrică face legătura cu fișierul ce conține detaliile de instalare a tuturor componentelor care formează software-ul Oracle – pentru o primă instalare de pe discul din unitatea CD-ROM nu este necesar să o modificăm. A doua rubrică va conține propunerea de nume pentru *reședința* (HOME) a produselor nou instalate - aici se cuvin câteva precizări: dacă este prima instalare și nu sunteți pretențioși numele implicit poate fi fără probleme păstrat, dacă însă mai este deja instalat ceva software Oracle atunci ni se va sugera numele primului HOME în ordinea instalării ceea ce însă nu este întotdeauna o soluție fericită fiindcă, în cazul unei versiuni diferite, suprapunerea instalării peste vechiul HOME ar putea să determine nefuncționalități destul de grave ale produselor Oracle existente, în acest caz se recomandă modificarea numelui implicit din rubrica *destination name* și specificarea unei căi separate de calea implicită din rubrica *destination path*. După cum am sugerat deja, ultima rubrică specifică calea de instalare și dacă nu există obiecții (nu mai există produse Oracle instalate în aceeași cale, iar spațiul pe disc este de ajuns) poate fi păstrată.

---

<sup>3</sup> Java Runtime Engine



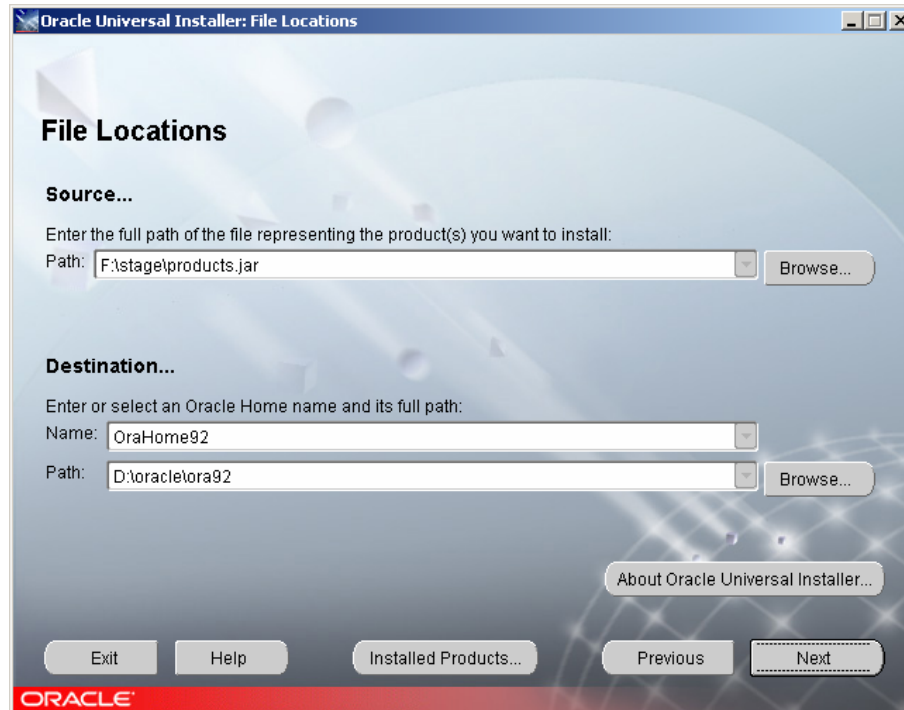


Figura 2.6. Specificarea căilor de instalare

Pasul următor ne pune în fața unei (prime) dileme: *ce mod de instalare să folosim*. Prima distincție pe care o putem face este că există *servere* de baze de date și *clienți* pentru aceste servere, iar a doua că există *instrumente de administrare simplă* doar pentru baza de date și clienții acesteia, și există *un cadru de administrare complex* pentru software-ul de întreprindere în care se integrează aceste servere de baze de date și clienții lor. Deocamdată, ca utilizatori novici ce suntem, ne interesează în primul rând să obținem baza de date, instrumentele minime de administrare și un client simplu (pe aceeași mașină cu serverul BD) cu ajutorul căruia să ne legăm la serverul BD. Acest mod de abordare presupune selectarea primei opțiuni. Eventual, după ce avem o bază de date funcțională și vom dori să ne conectăm de la o altă mașină din rețea, atunci putem invoca și ultima opțiune. Deocamdată rămânem fideli butonului „radio” *Oracle9i Database* pentru a nu avea complicații inutile și bătaie de cap suplimentare.



Figura 2.7. Specificarea instalării unei baze de date locale

O dată cu apăsarea butonului *Next* un nou ecran (cu trei opțiuni) ne zgândărește răbdarea – *Installation Types Oracle 9i Database*. Problema s-ar putea formula astfel: ce fel de bază de date dorim să obținem: una superpotentă (pentru aplicații tranzacționale solicitante, depozite de date sau aplicații Internet-intensive), una mai cuminte pentru aplicații departamentale dar și ceva facilități pentru distribuire, replicare și chiar aplicații Web, sau una și mai cuminte destinată doar testării în condiții mai degrabă single-user a unor aplicații totuși serioase care explorează caracteristicile obișnuite a unui server din clasa Oracle ? Deocamdată nu ne considerăm „zmei”, și vom renunța (prematur poate) la prima opțiune, lucrăm în echipă și ne mândrim totuși cu o mașină-server acceptabilă, deci vom exclude varianta 3 (recomandabilă dacă, de exemplu, avem la dispoziție un laptop de pe care dorim să portăm ulterior munca noastră într-ale bazelor de date pe un server adevărat). Vom opta așadar pentru varianta *standard*, nedorind să riscăm să compromitem (iremediabil) prima instalare folosind varianta *Custom*.

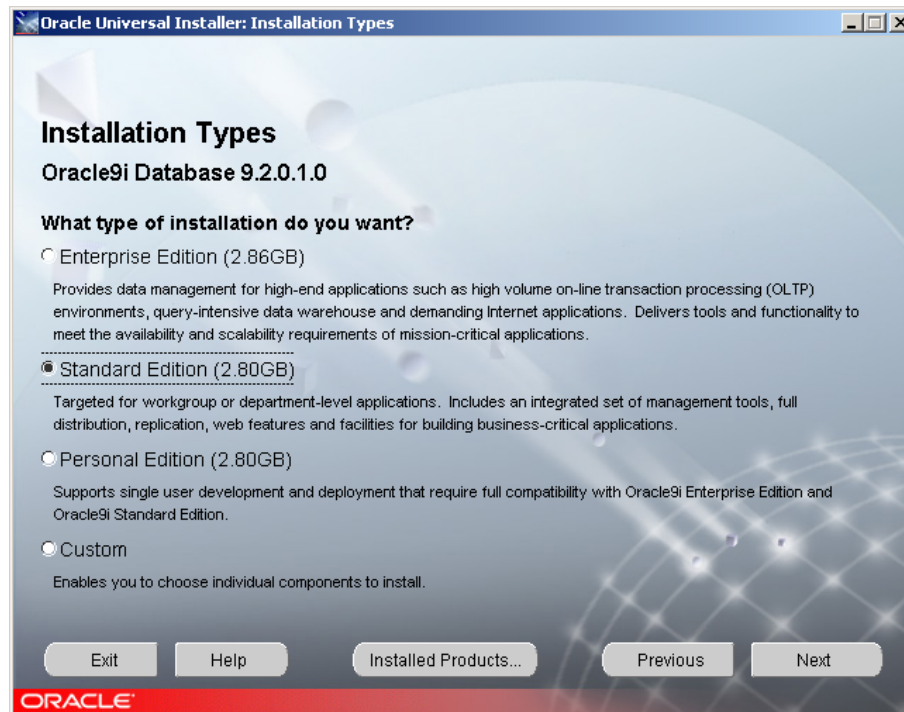


Figura 2.8. Tipurile de instalare

Următorul ecran *Database Configuration* ne permite să mai luăm o decizie: putem obține din momentul instalării și baza de date ale cărei caracteristici să fie determinate de unul dintre modelele pre-fabricate incluse în kit-ul de instalare (opțiunile *General Purpose*, *Transaction Processing*, *Data Warehouse*), sau putem opta pentru crearea unei baze de date în procesul căreia să intervenim eventual în cunoștință de cauză. Dacă dorim însă să amânăm pe mai târziu crearea acesteia putem merge pe varianta *Software Only*. Deocamdată, să fim încrezători și să mergem pe prima opțiune.



Figura 2.9. Selectarea unui model pentru baza de date

Trecând superficial peste pasul *Oracle Services for MTS*, ne grăbim să mai „apăsăm” un *Next* și ... răbdare, baza de date trebuie să aibă un nume – *Global Database Name*, iar instanța trebuie identificată și ea prin rubrica *SID*. Deci nu ignorați pasul *Database Identification*, pentru că, oricum, fără completarea acestor rubrici nici *Installer*-ul, prevăzător, nu merge mai departe. Ca să resolvăm rapid dilema legată de nume să-i spunem pur și simplu *Oracle* (nu este însă un principiu universal recomandabil, puteți să personalizați acest nume).

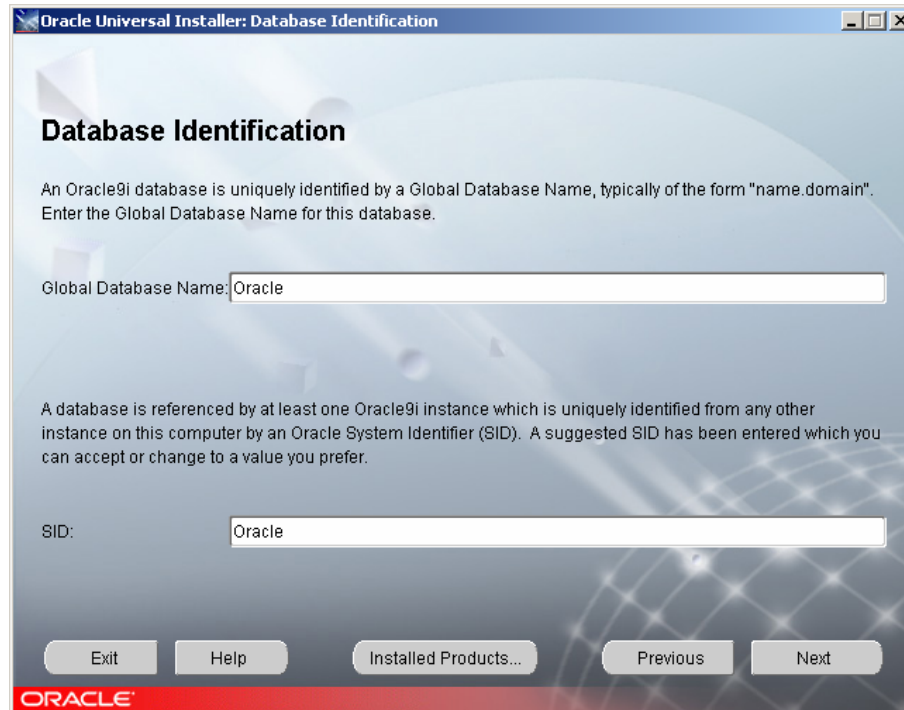


Figura 2.10. Specificarea numelui bazei de date

Pasul următor ne propune o *cale pentru fișierele bazei de date*, de regulă în interiorul *reședinței* (HOME) specificată în pasul *File Locations*. Ca să nu avem ulterior probleme la pornirea bazei de date, e bine să fim de acord cu directorul implicit.

Pasul *Database Character Set* are pentru noi (românii) o anume importanță legată de posibilitatea reprezentării în baza de date a caracterelor diacritice (ă, î, â, ș, ț). Cei care vor dori aplicații (de exemplu pentru Web) care să redea corect textul în românește vor trebui să aprofundeze această problemă a suportului multinațional, dar deocamdată să nu fim „naționaliști” și să acceptăm setul de caractere *default*.

În sfârșit, ajungem în ecranul *Summary* care nu mai conține butonul *Next* ci *Install*, prin urmare, după tegiversări și discuții (în contradictoriu), să lăsăm *Installer*-ul să-și facă treaba și să introducem pe rând, în ordine, atunci când ni se va solicita, celelalte discuri ce formează kit-ul de instalare.

După un timp de așteptare mai mult sau mai puțin îndelungat (depinde de performanțele mașinii-server, dar oricum nu cade sub 30 min), procedura de instalare continuă cu un ecran care merită toată atenția: *specificarea parolelor pentru SYS și SYSTEM*.<sup>4</sup> Acești utilizatori reprezintă conturile cu privilegii de

---

<sup>4</sup> Până la versiunea 9i, bazele de date Oracle erau create implicit cu parolele *manager* pentru *SYSTEM* și *change\_on\_intall* pentru *SYS*. Pentru evitarea problemelor de securitate implicate de

administrare a sistemului și sunt creați implicit. Pe lângă aceștia, Oracle mai furnizează și câteva conturi care conțin scheme BD exemplu, destinate inițierii noilor veniți. În acest sens cea mai cunoscută schema este *SCOTT*, care solicită la conectare parola *tiger*.

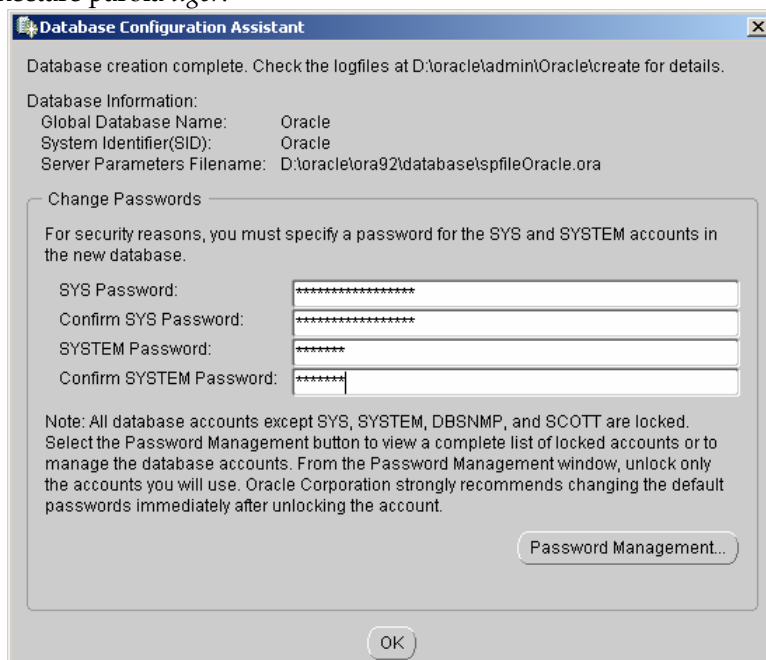


Figura 2.11. Trebuie specificate parolele pentru conturile de administrare SYS și SYSTEM create implicit

Ultimul pas presupune inițializarea și configurarea câtorva instrumente necesare pentru bunul mers și buna gospodărire a bazei de date „nou-născute”. Cele mai importante se referă la *Oracle Net Configuration* pentru că au în vedere gestionarea cererilor eventualilor clienți (de pe alte calculatoare din rețea) ai serverului BD (sarcină delegată serviciului *Listener*) și configurarea clientului local pentru a ne putea conecta la baza de date direct de pe mașina server. De regulă, stratul care asigură conectarea și comunicarea cu serverul BD se bazează pe așa-numite „adaptoare” pentru protocoalele de rețea cele mai răspândite (TCP/IP, SPX/IPX, etc.), însă clientul local se bazează pe un protocol de inter-comunicare între procese, așa încât nu vor fi solicitate (deocamdată) detalii suplimentare legate de rețeaua locală și locul mașinii-server în această rețea.

---

nemodificarea ulterioară a acestor parole, în 9i se solicită expres, din momentul instalării, specificarea noilor parole.

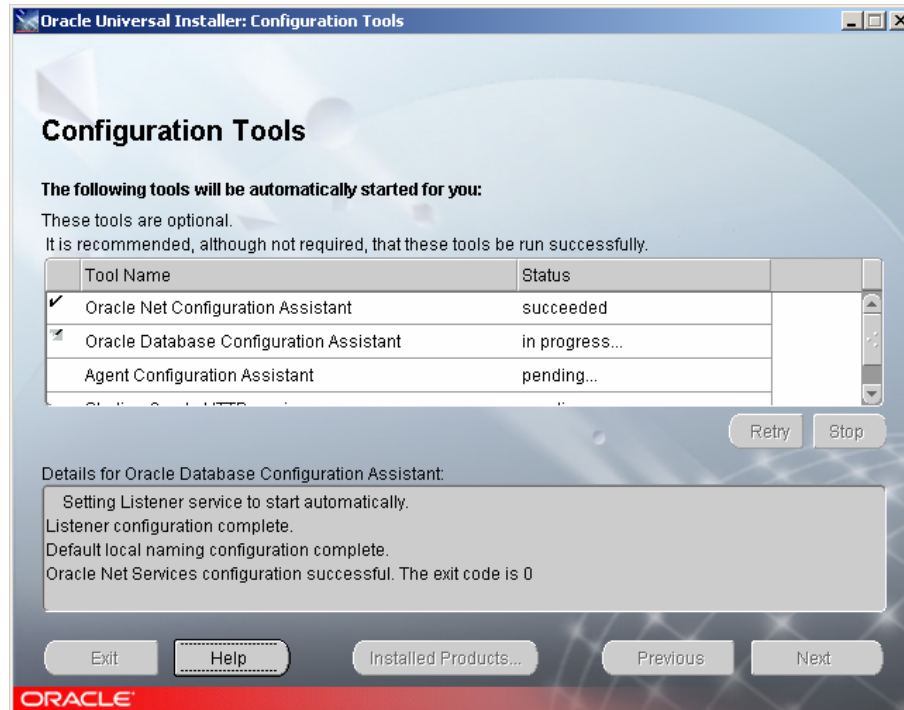


Figura 2.12. În final, configurarea ultimelor instrumente printre care și acelea legate de gestionarea conectării clienților din rețea la serverul bazei de date

Dacă întreaga procedură s-a încheiat cu succes, atunci putem verifica mai întâi din *Control Panel* → *Administrative Tools* → *Services* (pentru MS Windows 2000) serviciile create pentru baza de date și mediul aferent ei pe server.

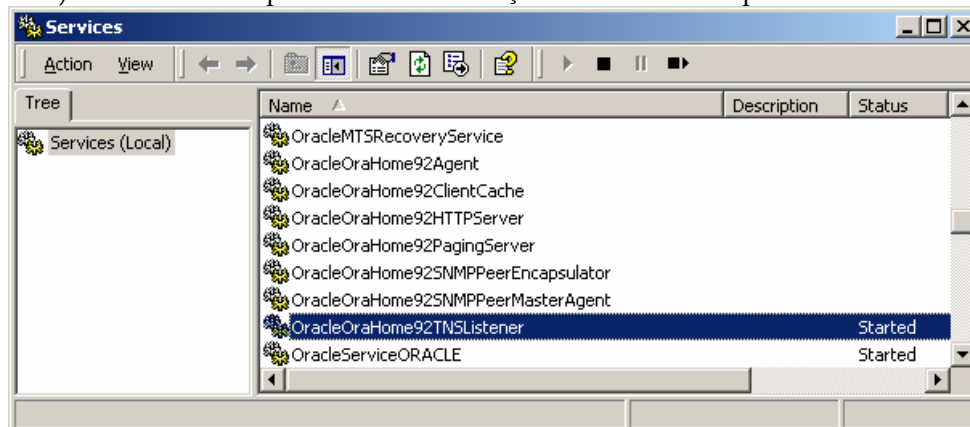


Figura 2.13 Serviciile Oracle

Cele asupra cărora ne vom opri apoi în configurare sunt *OracleTNSListener* și *OracleServiceORACLE*. Primul este gestionarul cererilor clienților din rețea

pentru baza de date, iar cel de-al doilea reprezintă *instanța* bazei de date numită în cazul nostru *ORACLE* (vezi pasul 5). Din acest motiv este recomandabil (în special pentru a cruța o parte din resurse) ca să fie oprite celelalte servicii, iar modul acestora de pornire să fie *Manual*, pentru a nu fi repornite la *reboot*-area calculatorului.

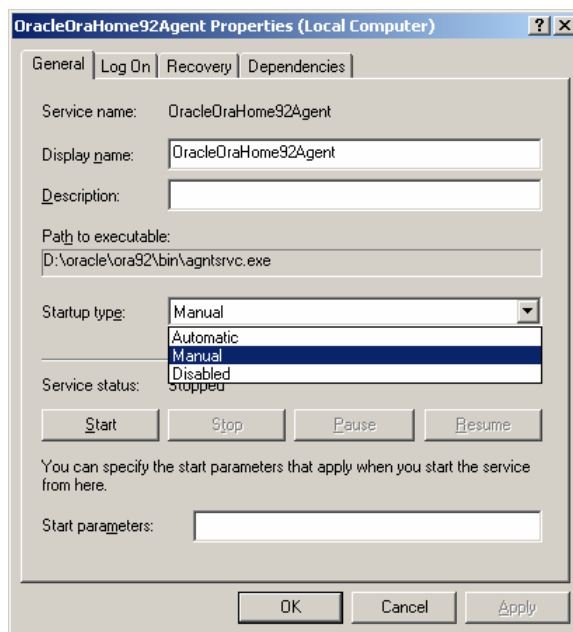


Figura 2.14. Configurarea modului de pornire *Manual* al serviciilor Windows2000

Acestea fiind spuse, putem încerca o conectare la baza de date folosind instrument-ul cel mai cunoscut (și, probabil, cel mai nepopular), vechiul *SQL Plus* din *Start* → *Programs* → *Oracle - Home* → *Application Development*. Ecranul de conectare cuprinde trei rubrici: *nume utilizator* - *username*, *parolă* - *password* și *nume (serviciu) bază de date* - *host string*. Vom specifica *SCOTT* pentru utilizator, *tiger* pentru parolă și vom lăsa necompletată ultima rubrică, ceea ce înseamnă că dorim să ne conectăm la baza de date locală.

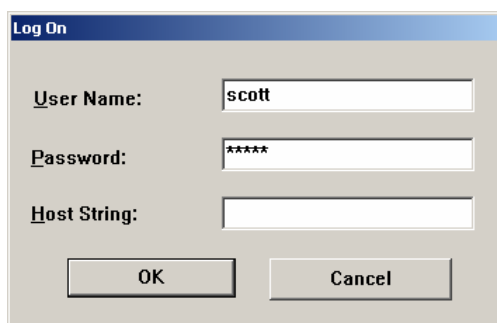
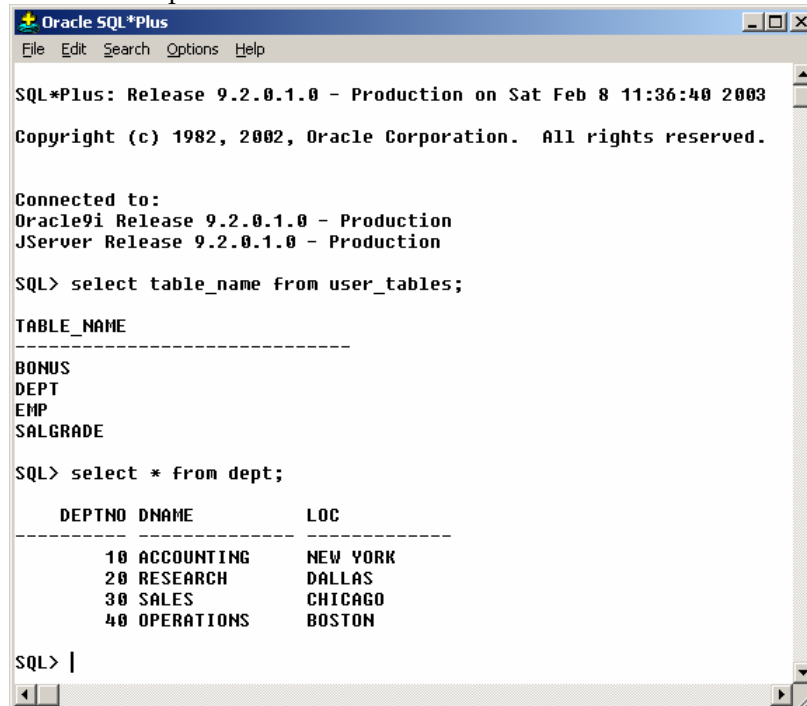




Figura 2.15. Ecranul de întâmpinare SQL\*PLUS. Conectarea la baza de date

Pentru a verifica eventualele structuri de tabele din schema curentă putem interoga dicționarul bazei de date, apoi putem să executăm o frază SELECT-SQL pentru o tabelă specifică.



```

Oracle SQL*Plus
File Edit Search Options Help

SQL*Plus: Release 9.2.0.1.0 - Production on Sat Feb 8 11:36:40 2003

Copyright (c) 1982, 2002, Oracle Corporation. All rights reserved.

Connected to:
Oracle9i Release 9.2.0.1.0 - Production
JServer Release 9.2.0.1.0 - Production

SQL> select table_name from user_tables;

TABLE_NAME
-----
BONUS
DEPT
EMP
SALGRADE

SQL> select * from dept;

  DEPTNO DNAME          LOC
-----
10 ACCOUNTING NEW YORK
20 RESEARCH  DALLAS
30 SALES      CHICAGO
40 OPERATIONS BOSTON

SQL> |

```

Figura 2.16. Utilitarul SQL Plus

### 2.3.6. Crearea unei baze de date Oracle

Am văzut în paragraful anterior cum se poate instala software-ul Oracle într-o variantă care să includă și o bază de dată implicită („clonată” de fapt pe baza unei structuri prefabricate de pe kit-ul de instalare). S-ar putea însă să existe câțiva „cârcotași” care au avut răbdarea să citească paragrafele 2.3.2 și 2.3.3 (despre structurile logice și fizice) și care nu privesc cu „ochi buni” modul în care se obține noua bază de date, sau mai exact modul în care aceasta acaparează în mare secret și fără avertizare resursele, probabil destul de limitate, ale calculatorului de test. Aceștia vor dori un control mult mai amănunțit asupra instanței și celorlalte structuri fizice ale bazei de date. Din acest motiv vor amâna momentul creării bazei de date, selectând în pasul *Database Configuration* din procesul de instalare opțiunea *Software Only*.

### Folosirea mediului de asistență pentru crearea unei baze de date personalizate

Asistentul pentru configurarea (mai) amănunțită a bazei de date îl veți găsi în *Start → Programs → Oracle – Home → Configuration and Migration Tools*.

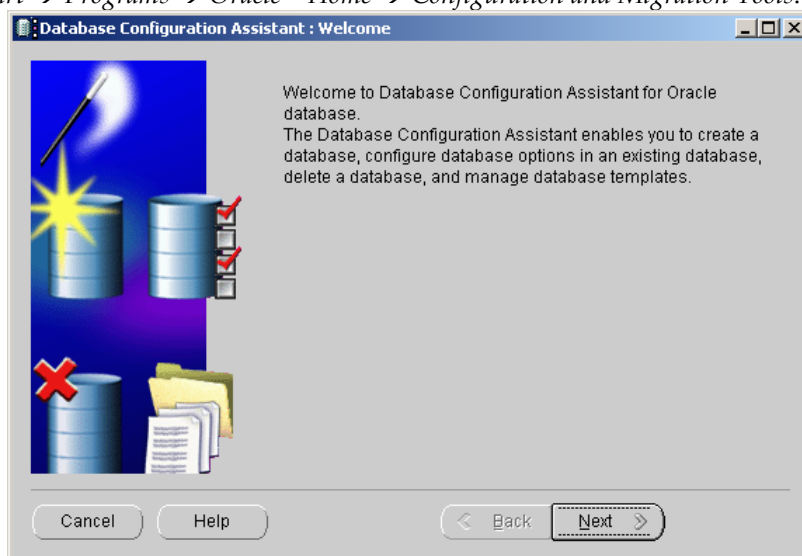


Figura 2.17. Ecranul de întâmpinare al *DB Configuration Assistant*

Acest utilitar poate fi folosit pentru crearea unei noi baze de date însă, el și la (re)configurarea opțiunilor unei baze de date existente, ștergerea unei baze de date sau gestionarea șabloanelor (modelelor) predefinite.

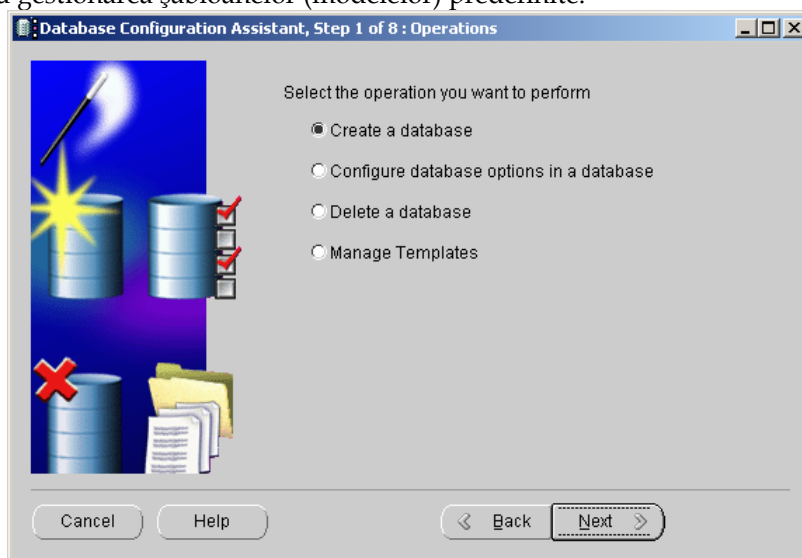


Figura 2.18. Operațiile care se pot realiza cu asistentul de configurare al bazei de date

Megând pe varianta creării asistate a baze de date, în primul pas (important, fiindcă este al doilea în ordinea reală) – **Database Templates** - vor fi propuse mai multe modele (*templates*) predefinite, fiecare având anumite caracteristici preconfigurate, astfel:

*Data Warehouse* – suport pentru interogări complexe efectuate asupra unor mari cantități de date care necesită timp de răspuns rezonabili și acuratețe maximă. Aceste interogări pot varia ca rezultat, de la câteva înregistrări dar cu mari disponibilități de agregare, până la mii de înregistrări rezultate din criterii relativ simple dar provenind dintr-o multitudine de tabele jonctionate;

*General Purpose* – suport pentru o varietate de sarcini, de la tranzacții simple până la interogări complexe. Acest model este recomandabil în special în mediile de testare pre-producție ale unor sisteme de aplicații cu funcționalități complexe și variate;

*Transaction processing* (sau OLTP)- suport în special pentru tranzacții obișnuite; adică actualizarea (inserare, ștergere, modificare) unui număr rezonabil de tabele (eventual și consultarea prealabilă a acestora) în aceeași unitate operațională, însă în medii concurențiale - solicitări simultane deseori asupra acelorași date (acelorași înregistrări, sau altor înregistrări dar din aceleași tabele) provenind de la un număr mare de utilizatori;

*New database* – reprezintă varianta personalizării (*custom-izării*) opțiunilor bazei de date funcție de necesitățile particulare de procesare ale mediului utilizatorilor.

Pentru amănunte, opțiunile predefinite pentru fiecare variantă pot fi obținute prin raportul care este obținut apăsând butonul *Show Details ...*

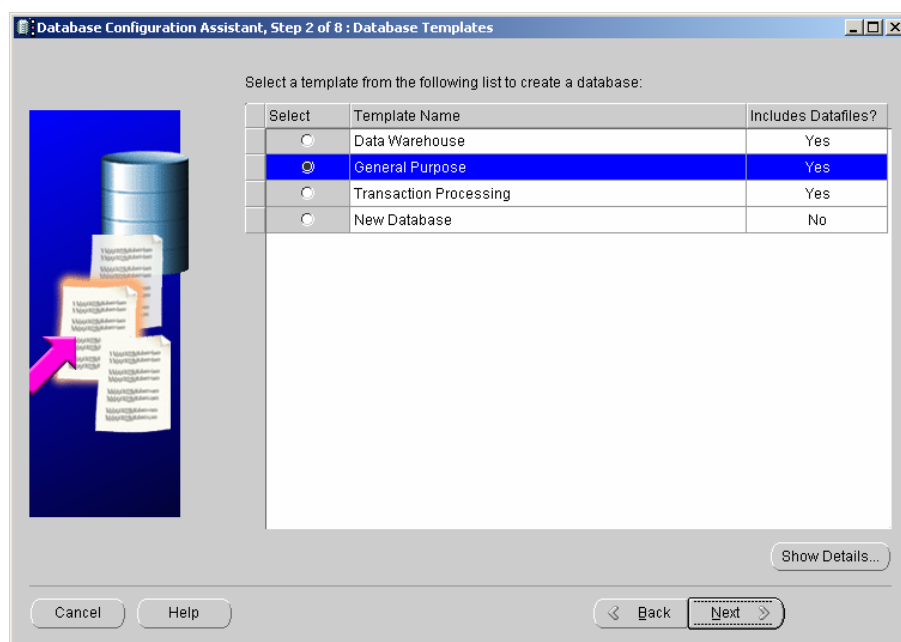


Figura 2.19. Modelele de configurare predefinite

Alegem *General Purpose* și mergem mai departe (*Next*). La fel ca și în cazul instalării implicite, în pasul următor – *Database Identification* – se va solicita numele global al bazei de date și numele instanței care o va deservi. Numele implicit al acestei instanțe va fi sugerat pornind de la numele bazei de date.

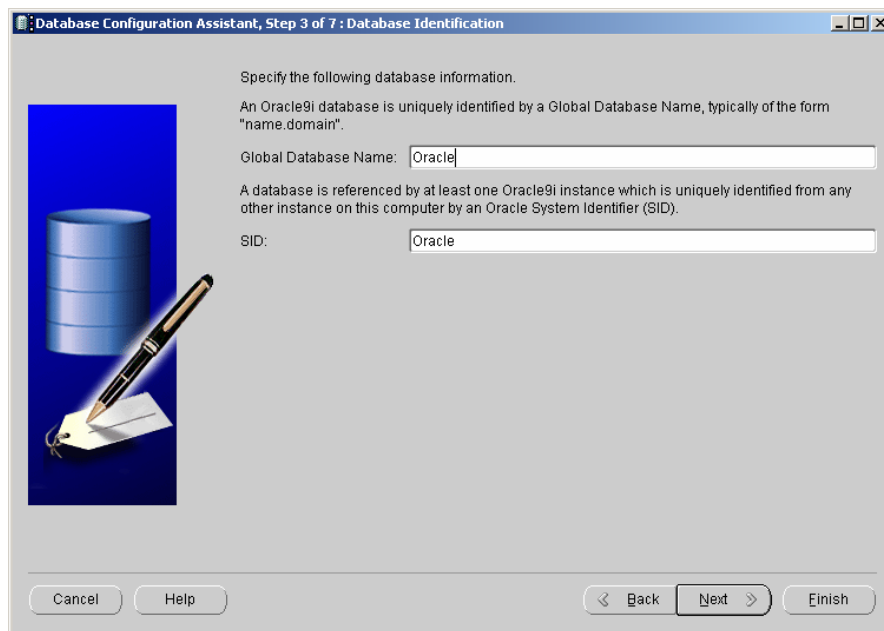


Figura 2.20. Pasul necesar pentru specificarea identității bazei de date

În pasul următor ni se solicită imperativ „modul în care doriți să opereze implicit baza de date” – *Database Connection Options*. Deși formularea este destul de neclară, totuși, pentru fiecare opțiune, există o scurtă descriere. Pe scurt, principiul colaborării între clienți și serverul Oracle se bazează la nivelul acestuia din urmă pe un proces separat care comunică direct cu procesul client și intermediază cererile acestuia asupra bazei de date. Aceste procese intermediare pot fi alocate separat câte unul pentru fiecare conexiune a fiecărui client (modul de dedicat), caz în care comunicarea se face direct, fără cozi de așteptări și riscul unor timpi „morți” la nivelul proceselor client, sau, pentru a „drămui” mai bine resursele serverului (în ceea ce privește RAM-ul necesar), un singur proces server poate deservi simultan mai mulți clienți, caz în care atunci când apar mai multe cereri simultane acestea sunt organizate sub forma unor cozi de așteptare, fiind servite strict în ordinea temporală a apariției.

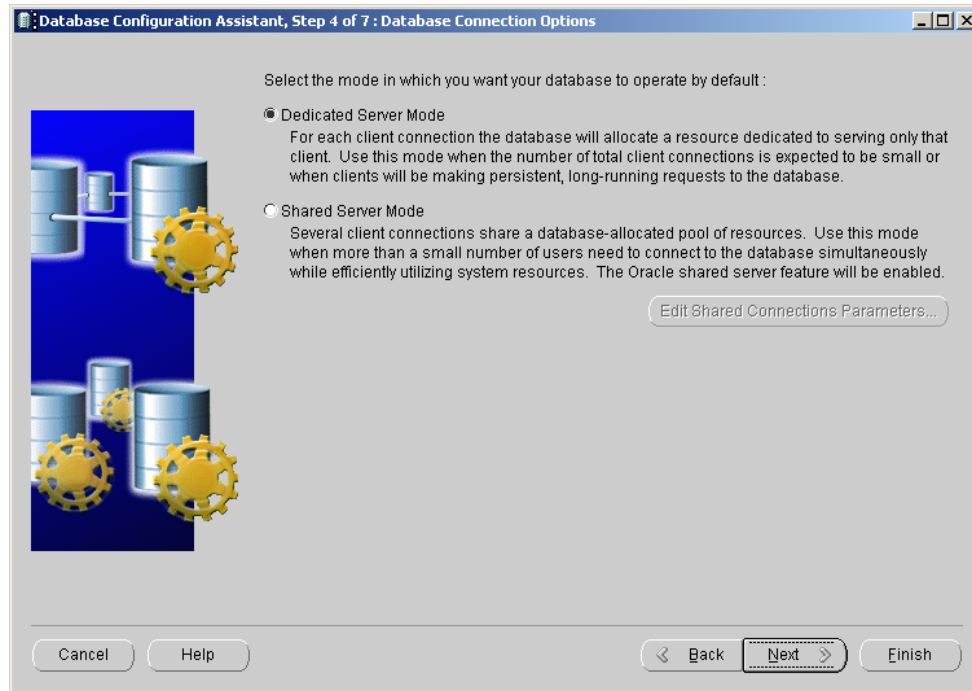


Figura 2.21. Detalii legate de modul de deservire a cererilor clienților bazei de date

În cunoștință (mai mare sau mai mică) de cauză să alegem modul implicit (dedicat) și să mergem mai departe în pasul cel mai complex al acestui proces **Initialization Parameters**. Ecranul corespunzător acestei etape este împărțit în mai multe cadre:

**Memory.** Parametrii corespunzători acestui cadru se referă la modul în care este gestionată memoria internă dedicată instanței care deservește baza de date. Semnificația acestor parametri se leagă în mare parte de descrierea elementelor din paragraful 2.3.2, și anume:

**Shared Pool** – cache-ul destinat datelor consultate din dicționarul bazei de date și informațiilor despre frazele SQL rulate. Pentru o bază de date tranzacțională se recomandă o valoare în jur a 30M, care eventual ar putea fi mărită dacă există dese interogări complexe și variate (aplicații gen DSS);

**Buffer Cache** – buffer-ele Oracle (nu direct ale sistemului de operare) dedicate replicării în memoria internă a blocurilor de date care conțin înregistrările solicitate de interogări sau tranzacții. În general, cu cât baza de date este mai „tranzacționată” (modelul OLTP), cu atât pentru această zonă este rezervat mai mult spațiu. Se recomandă o valoare mai mare (să zicem în jur a 50M), pentru a crește performanțele bazelor de date puternic tranzacționate, valoarea obișnuită pentru bazele de date „normale” în jur de 30M;

**Java Pool** – destinată analizei/interpretării/execuției codului Java stocat în baza de date sub forma procedurilor stocate (opțiune disponibilă de la versiunile 8i).

Există un prag minim pentru dimensiunea acestei zone (pentru un calculator cu 256MB-RAM, automat se recomandă 20M);

*Large Pool* – reprezintă o zonă de memorie relativ opțională, utilă pentru operațiile frecvente de creare a copiilor de siguranță și restaurare a bazei de date. Pentru baze de date în producție se recomandă o dimensiune în jur de 10M, altminteri, doar pentru test, se poate miza pe această valoare;

*PGA* – reprezintă volumul de memorie (maxim) alocat pentru un proces server care colaborează cu celelalte procese din instanță pentru a deservi cererile proceselor-client.

Stabilirea parametrilor anteriori se poate face:

*Typical*, ceea ce înseamnă acceptarea unor valori determinate automat funcție de doi factori: scopul bazei de date (OLTP – tranzacțional, Multipurpose-general, depozite de date – DataWarehousing) și procentul de memorie alocat din memoria RAM a sistemului (nu se recomandă mai puțin de 128MB totuși). Luând în calcul 70% pentru un sistem cu 256MB-RAM se generează următoarele valori:

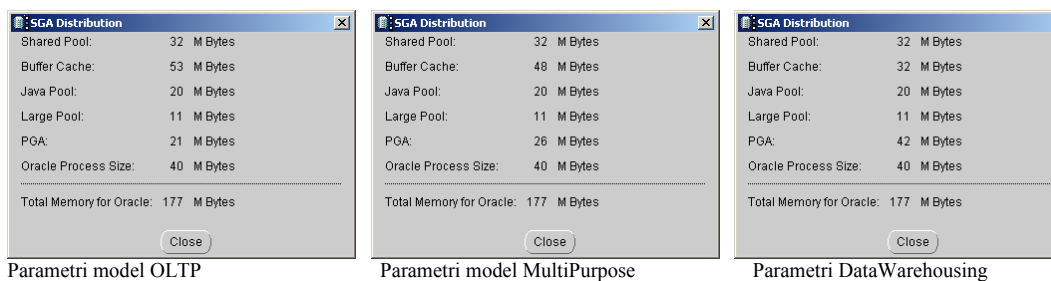


Figura 2.22. Valorile parametrilor determinanți pentru gestiunea memoriei interne specifice pentru diverse modele

*Custom*, variantă în care trebuie precizați toți parametri de mai sus. Pentru a simți că facem ceva totuși important și responsabil, vom merge pe această variantă și vom specifica în ordine valori (în MB): 32, 32, 20, 4, 16 la care se vor adăuga încă 40MB destinați proceselor background ale instanței.

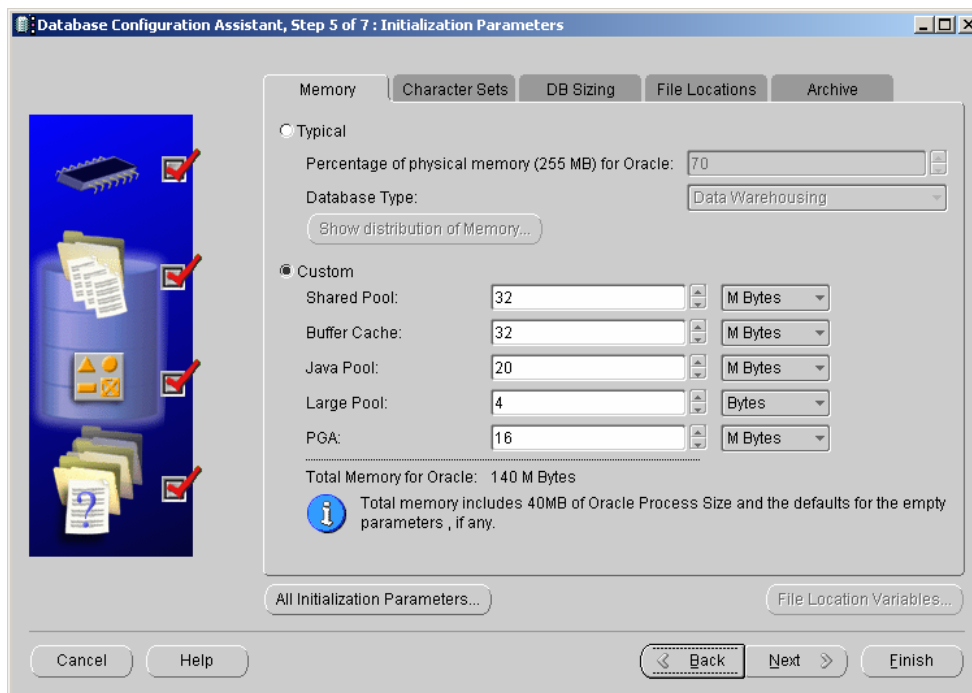


Figura 2.23. Specificarea modului în care va fi repartizată memoria destinată instanței bazei de date

*Character Sets* – setul de caractere are importanță după cum spuneam și în paragraful anterior pentru „naturalizarea” bazei de date Oracle l-a caracterele naționale specifice;

*DB Sizing* – sună cam impropriu, pentru că se referă de fapt la zona de memorie internă care va fi dedicată de fapt proceselor de sortare datorate executării un fraze SELECT ce implică principii de ordonare (ORDER BY, GROUP BY, operatori de agregare) sau creării unor indecși specifici;

*File Locations* – se referă în primul rând la localizarea fizică a fișierului de parametri al instanței relativ *ORACLE\_BASE* adică o locație cum ar fi *D:\Oracle* (nu *D:\Oracle\Ora92*, care înseamnă *ORACLE\_HOME*) și a fișierului de parametri dinamic *SPFILE*;

*Archive* – modul arhivare (bifarea checkbox-ului *ArchiveLogMode*) reprezintă cel mai obișnuit mod de a realiza o copie de siguranță a datelor tranzacționate prin arhivarea fișierelor de jurnalizare (log). În cazul unei căderi, starea bazei de date se poate reface dacă mai există o copie a fișierelor de date și arhivele fișierelor jurnalizate de la momentul efectuării respectivei copii de siguranță. Pentru bazele de test nu este necesară activarea modului de arhivare a jurnalelor, însă pentru o bază de date în producție trebuie realizată o politică de backup/recovery care va include și arhivarea automată a jurnalelor tranzacțiilor din fișierele log.



Pasul următor are în vedere *structurile fizice de stocare*, mai exact fișierele de control, de jurnalizare și de date – **Database Storage**. Ecranul este organizat în două zone distincte: zona din stânga prezintă structurile fizice împărțite pe categorii legate printr-o arborescență, iar în dreapta sunt prezentate detaliile fiecărei categorii.

Prima ramură *Controlfile* prezintă detalii despre fișierele de control, adică localizarea și multiplicarea lor (cadrul *General*), precum și modul de repartizare a spațiului funcție de intrările care vor reflecta structura bazei de date – numărul de fișiere de date, de grupuri redo-log (jurnalizare), de membri per fiecare grup (cadrul *Options*).

A doua ramură *Datafiles* prezintă detaliile legate de localizarea fișierelor de date care vor forma tablespace-urile bazei de date. De exemplu *SYSTEM01.DBF* corespunde tablespace-ului *SYSTEM*, singurul care trebuie creat în mod obligatoriu fiindcă va găzdui dicționarul sau catalogul bazei de date.

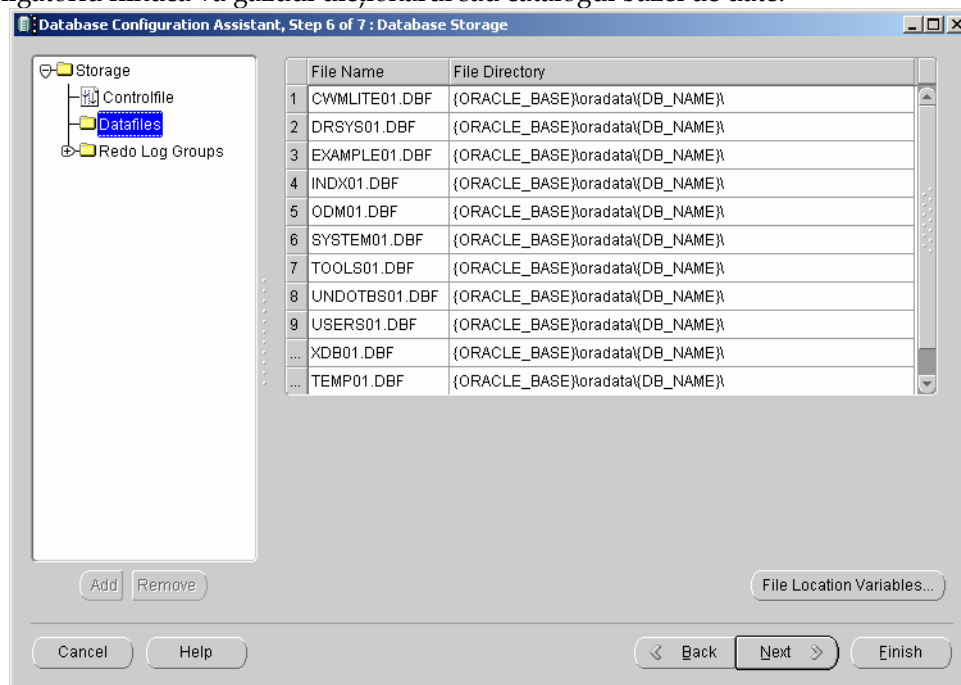


Figura 2.24. Numele și plasarea fișierelor de date care vor forma baza de date

A treia ramură *Redo Log Groups* se referă la fișierele de jurnalizare, organizate în grupuri care, pentru baze de date în producție, se recomandă a avea mai mulți membri multiplexați pe discuri fizice separate pentru a se evita „colapsul” bazei de date în urma unor căderi. Grupurile redo-log trebuie să fie cel puțin două (implicit sunt trei), fiecare conținând cel puțin un membru.

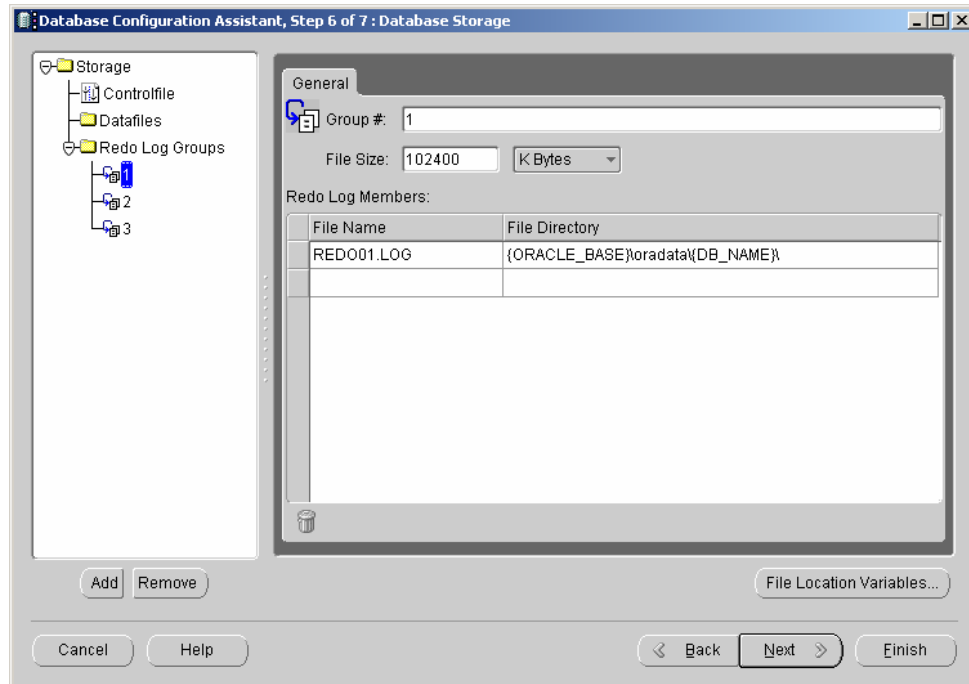


Figura 2.25. Repartizarea și plasarea în structura fizică a fișierelor de jurnalizare

Căile specificate pentru fiecare dintre fișierele discutate anterior sunt încryptate folosind anumite variabile, ale căror valori reale le putem afla apăsând butonul *File Location Variables ...*

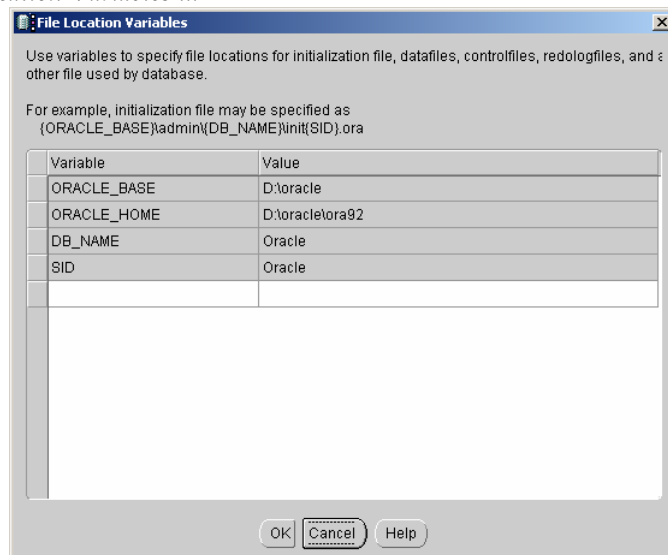


Figura 2.26. Valorile variabilelor care determină căile fizice de localizare ale fișierelor bazei de date

Ca urmare putem deduce structura de directoare care va rezulta în urma acestui proces (vezi paragraful următor).

Ultimul pas *Creation Options* prezintă două opțiuni în legătură cu finalizarea acestui proces: crearea (fizică) a bazei de date și/sau salvarea acestei configurații sub forma unui model (template) care ar putea fi refolosit ulterior pentru specificarea parametrilor unei alte baze de date.

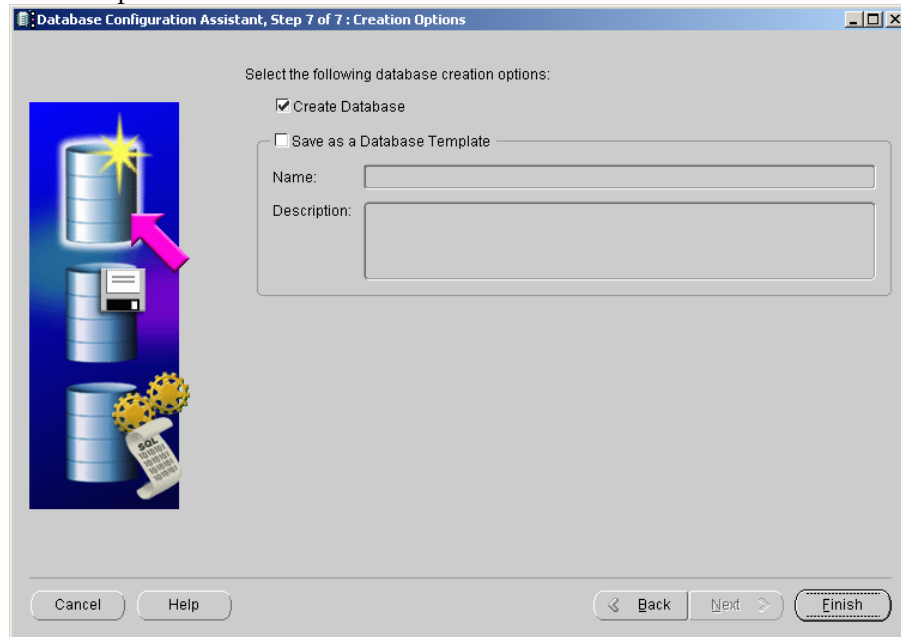


Figura 2.27. Finalizarea procesului de configurare: creare unei noi baze de date și/sau a unui nou model (template)

În fine butonul *Finish* marchează sfârșitul procesului de configurare și începerea procesului efectiv de stocare care se va încheia cu crearea și pornirea serviciilor Oracle descrise în paragraful anterior și cu crearea structurii de directoare descrise în paragraful următor.

De asemenea, spre deosebire de versiunile anterioare, în Oracle9i procesul de creare a bazei de date mai are un efect: înregistrarea (în fișierul de configurare *listener.ora*, secțiunea *SID\_LIST\_LISTENER*) de către procesul *Listener* a noii baze de date, cererile de conectare către instanța acesteia fiind deservite automat, plus înregistrarea unui serviciu-descriptor pentru noua baza de date automat pe clientul local (fișierul *tnsnames.ora*). Pentru detalii suplimentare privitoare la modul de colaborare între clienții și serverele Oracle vezi paragraful 2.5.

### Structura de directoare

După cum se poate concluziona de altfel și din figura care prezintă variabilele *ORACLE\_BASE*, *ORACLE\_HOME*, *DB\_NAME* și *SID*, structura de directoare implicată se prezintă astfel (pentru MS Windows 2000):

rădăcina o reprezintă directorul *Oracle* părinte pentru toate fișierele suport sau ale bazelor de date stocate pe server. Acest director prezintă trei subramuri (subdirectoare):

directorul *Ora92* (pentru Oracle9i2) destinat fișierelor suport ale software-ului Oracle (serverul BD și utilitare) plus fișierele de configurare ale clientului local etc.

directorul *admin* în care va fi creat câte un subdirector specific pentru fiecare bază de date locală. Un astfel de subdirector destinaat unei BD locale prezintă, la rândul lui, alte cinci subdirectoare *bdump*, *cdump*, *create*, *pfile*, *udump*. Dintre acestea, spre exemplu fișierul de parametri *Init.ora* este plasat în *pfile*;

directorul *oradata* care va conține la rândul său câte un director dedicat fiecărei BD locale. Într-un astfel de subdirector sunt stocate implicit fișierele bazei de date (fișierele de date, de control, de jurnalizare).

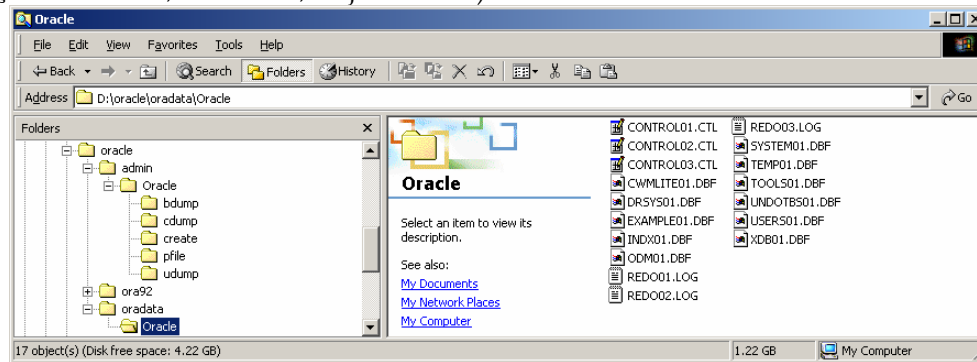


Figura 2.28. Structura locală de fișiere specifice bazei de date

### 2.3.7. Consola Oracle Enterprise Manager. Configurare și utilitate

Consola Oracle Enterprise Manager (OEM) reprezintă instrumentul de administrare al bazelor de date furnizat direct de Oracle. Acesta funcționează fie independent (standalone) fie în colaborare cu un server de administrare dedicat (Oracle Management Server), care poate asigura automatizarea sarcinilor de administrare pentru o rețea concertată de baze de date (în special pentru BD distribuite).

Deocamdată, pentru sarcinile de rutină dar esențiale în administrarea bazei de date pe care am obținut-o, ne vom (foarte) mulțumi cu varianta simplă.



Figura 2.29. Ecranul de întâmpinare al OEM

Fereastra principală OEM cuprinde o suprafață de lucru structurată în două zone: în stânga se găsește o arborescență, nodul *Databases* cuprinzând câte o ramură pentru fiecare bază de date administrată. Aceste noduri reprezintă sau se bazează de fapt pe serviciile-descriptor de pe clientul local. Ținând cont de precizările făcute în finalul paragrafului 2.3.6.1, ar trebui ca să existe implicit o astfel de ramură pentru baza de date creată local.

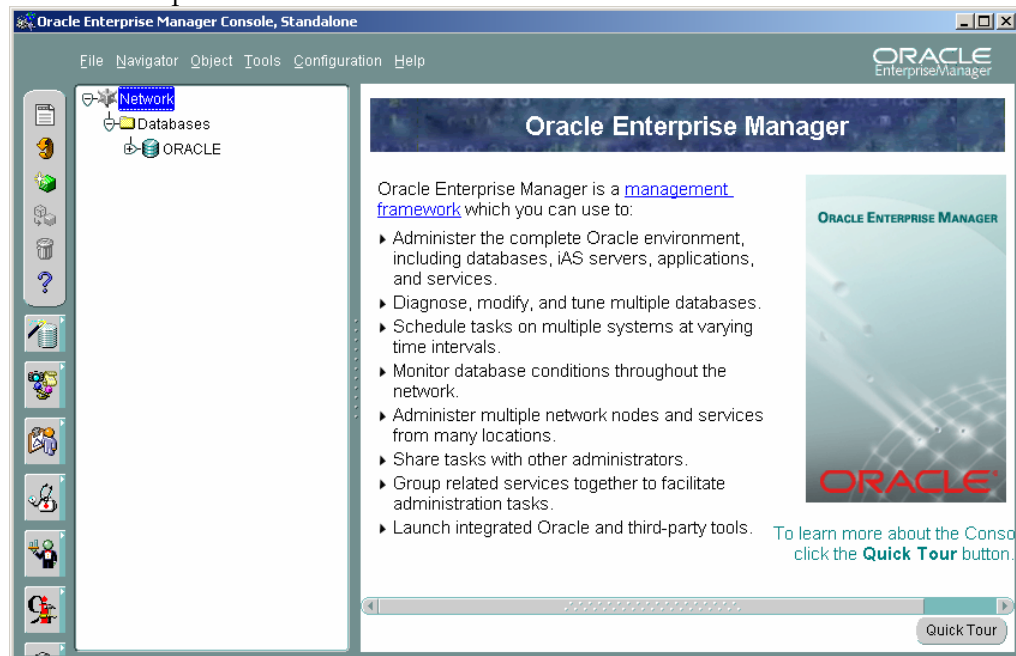


Figura 2.30. Organizarea suprafeței de lucru a OEM

În caz contrar se poate crea un nod (de fapt descriptor-serviciu) din *Navigator* → *Add Database To Tree*.

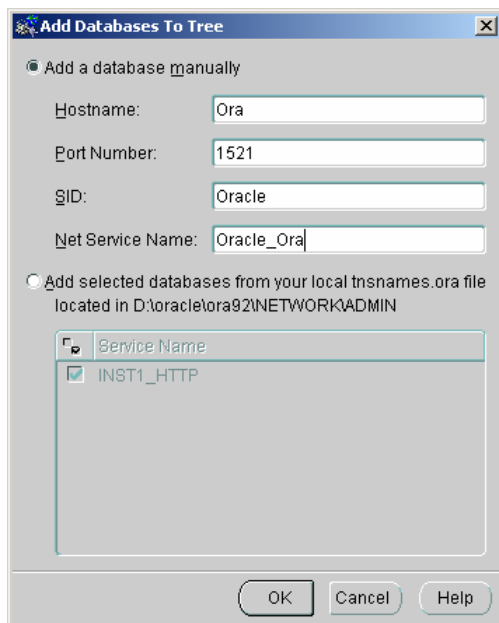


Figura 2.31. Fereastra pentru specificarea descriptorilor-servicii pentru bazele de date ce se doresc a fi administrate din consola OEM

Prima opțiune din fereastra de mai sus ne permite introducerea parametrilor de configurare a unui nou descriptor-serviciu, dacă acesta nu există deja. Dacă acesta este definit deja (de exemplu prin rularea anterioară a utilitarului *Net Configuration Assistant* – vezi paragraful 2.5), atunci poate fi selectat din lista din partea inferioară care devine disponibilă prin selectarea celei de a doua opțiuni *Add ... from your local tnsnames.ora file ...*.

După aducerea în OEM a bazelor de date dorite, încercarea de deschidere a unui nod automat va duce la deschiderea dialogului de conectare:

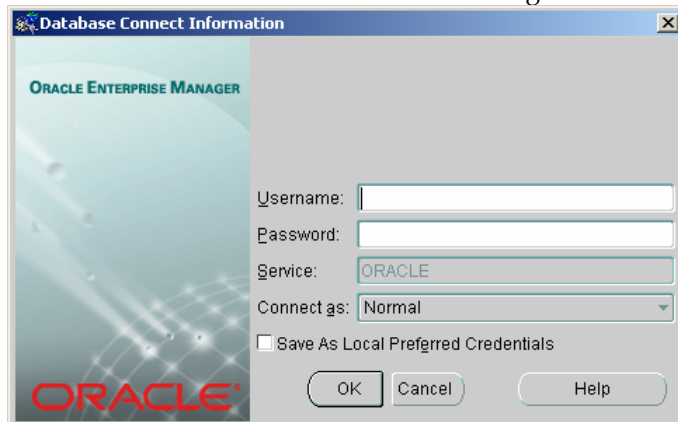


Figura 2.32. Fereastra de conectare

De obicei, conectarea se face printr-un cont administrator, de regulă SYSTEM, însă dacă se dorește și efectuarea unor operații mai deosebite cum ar fi închiderea/deschiderea bazei de date atunci este necesară conectarea cu un cont cu privilegii SYSDBA. Să luăm în considerare acest ultim caz: dacă com dori să ne conectăm la baza de date nou creată (Oracle) vom face click pe nodul de intersecție al acestuia (sau click-dreapta și *Connect*) după care specificăm ca *Username* SYS, parola va fi cea specificată pentru acesta la crearea bazei de date, iar în subrica *Connect as* vom selecta SYSDBA.

Nodul corespunzător bazei de date selectate se va deschide, fiecare ramură reprezentând o sarcină distinctă de administrare, dintre care cele mai reprezentative:

Din *Instance* putem porni/opri baza de date (dacă suntem conectați ca SYSDBA), sau putem să-i schimbăm parametrii de inițializare;

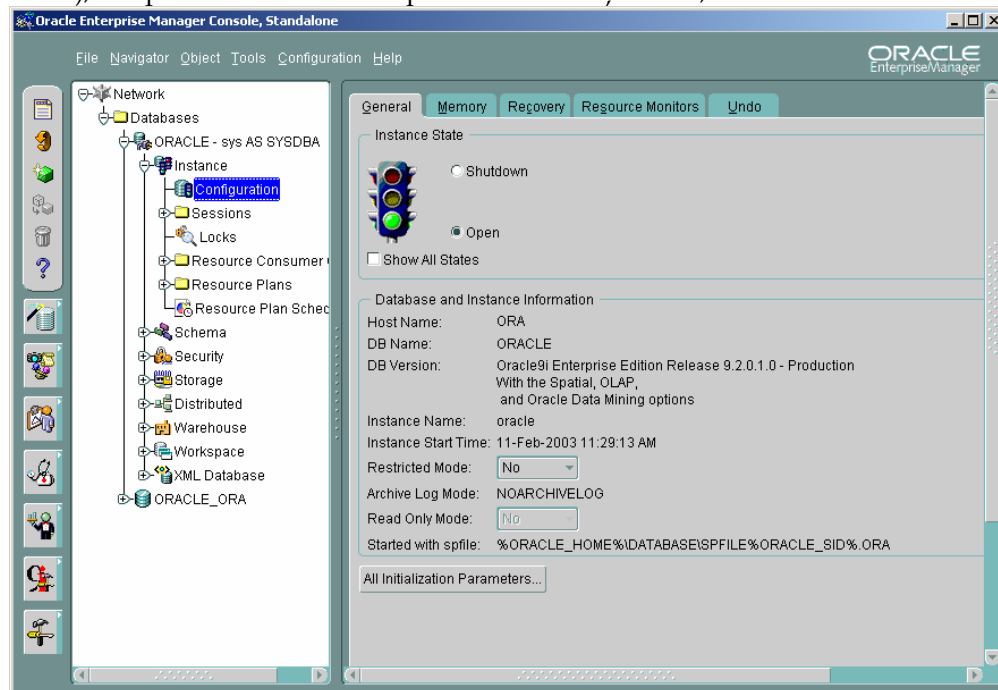


Figura 2.33. Administrarea instanței bazei de date

Din nodul *Schema* avem posibilitatea verificării tuturor obiectelor din baza de date grupate sub schemele fiecărui utilizator;

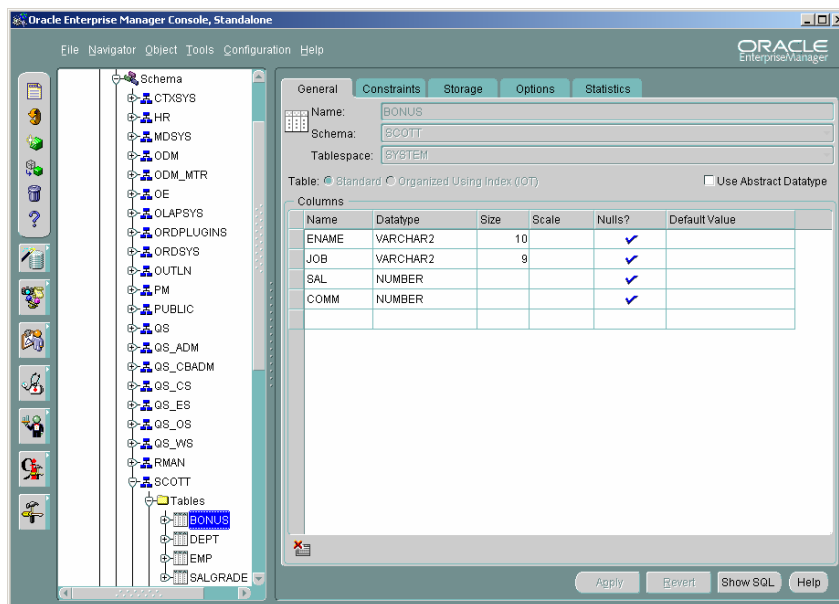


Figura 2.34. Administrarea schemelor de obiecte din OEM

Din nodul *Storage* avem posibilitatea să administrăm structurile fizice de stocare, adică să creăm noi tablespace-uri, să creăm noi fișiere în noi tablespace-uri sau în tablespace-urile existente, să redimensionăm sau chiar să îndepărtăm din baza de date aceste fișiere (ceea ce nu implică de regulă și ștergerea fizică la nivelul sistemului de operare);

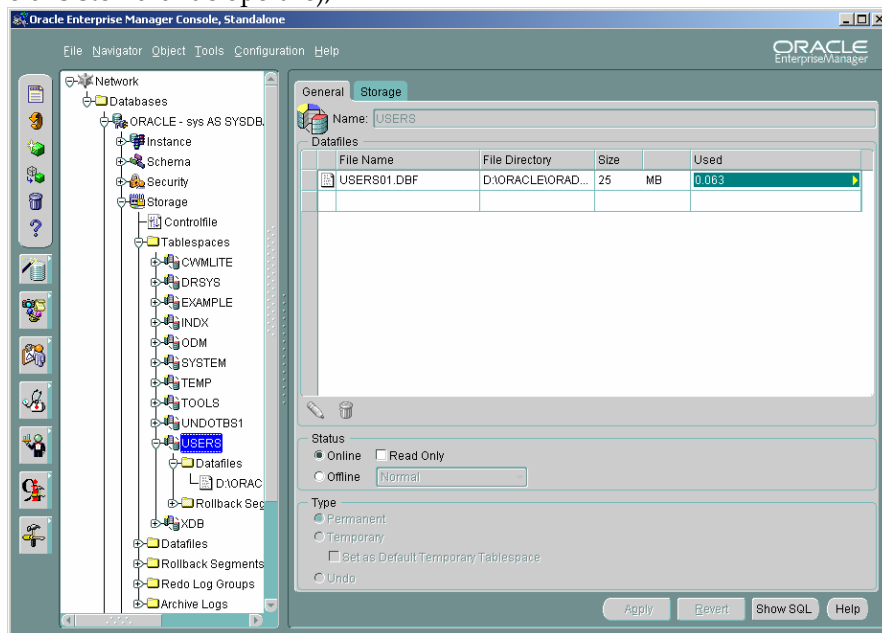




Figura 2.35. Administrarea structurilor fizice din OEM

Din nodul *Security* se pot crea noi conturi, roluri sau profile și se pot acorda și revoca privilegii.

## 2.4.Utilizatori, privilegii, roluri, într-un cuvânt *securitate*

Cei care se hotărăsc să investească într-o infrastructură pentru date având în centru un server de baze de date de „categorie medie-grea” se așteaptă bineînțeles și la un suport corespunzător de securitate. Datele trebuie să fie „sigure” din punctul de vedere al accesului: fiecare utilizator să poată „vedea” și „modifica” numai ceea ce este autorizat. În sistemele de aplicații cu funcționalitate complexă care reclamă anumite aspecte legate de confidențialitate dictate de „regulile de afaceri” implementate, problema devine critică în condițiile unui mediu cu mai mulți utilizatori fiecare având responsabilități specifice și regăsimu-se sub tutela unui grup formal (departamental) obligat să respecte un anumit set de politici și reguli interne.

Conceptele de bază pentru a stabili autentificarea unui utilizator pentru a avea acces la o bază de date și pentru a stabili relațiile de apartenență între obiectele stocate în baza de date și creatorul lor sunt *contul* și *schema*.

Astfel pentru accesarea datelor dintr-o bază de date Oracle trebuie utilizat un cont utilizator care are asociate anumite drepturi (privilegii) cu privire la acțiunile sale față de server-ul Oracle sau față de obiectele stocate în baza de date. În mod implicit printr-un cont utilizator devin accesibile toate obiectele din schema asociată.

O *schemă* reprezintă o colecție de obiecte formată din tabelele, view-urile, cluster-ele, procedurile stocate, package-urile ș.a. deținute de un anumit utilizator. Aceasta este creată implicit la crearea contului respectiv, de aceea numele ei și numele utilizatorului vor fi utilizate ca fiind echivalente.

### 2.4.1. Crearea utilizatorilor și obținerea schemelor de obiecte

După cum am văzut într-un paragraf anterior, procesul de creare a bazei de date Oracle produce și doi utilizatori privilegiați „împuternici” să efectueze sarcinile de administrare care necesită drepturi speciale, adică **SYS** care deține schema în care se găsesc toate obiectele ce formează catalogul sau dicționarul bazei de date plus responsabilitatea de a porni „manual” baza de date aflată în stadiul „shutdown” și **SYSTEM**, administratorul de facto (chiar dacă și SYS poate efectua o parte din sarcinile sale), care are prin urmare și „puterea” de a crea noi conturi în baza de date, sau de a modifica detaliile celor deja existente.

Crearea unui cont utilizator înseamnă implicit și asigurarea condițiilor pentru formarea schemei de obiecte asociate acestui cont.

Practic, la crearea unui utilizator trebuie specificate cel puțin următoarele elemente:

*numele* utilizatorului care va fi de fapt numele schemei de obiecte;

*parola* ca modalitate standard de autentificare în baza de date (mai există și alte posibilități de autentificare „externally” ținând seama de conturile utilizatorilor sistemului de operare local sau de rețea);

*tablespace-ul default* în care vor fi stocate implicit obiectele din schema asociată contului;

*tablespace-ul pentru segmentele temporare* din care va fi utilizat spațiul de memorie suplimentar necesar operațiilor de sortare costisitoare.

De asemenea, tot în momentul definirii unui cont se poate determina și modul în care utilizatorul poate folosi spațiul de stocare dedicat bazei de date, prin precizarea cotelor de spațiu în diferitele tablespace-uri.

Modul în care un utilizator (sau operațiile inițiate de acesta) poate solicita resursele disponibile serverului bazei de date poate fi reglementat prin așa-numitele *profile*. Acestea includ clauze specifice referitoare la timpul de procesor alocat, numărul de sesiuni concurente, timpul de conectare ș.a. dar și clauze privind utilizarea parolelor, adică perioada de valabilitate, perioada de grație pentru schimbarea parolei, numărul de încercări eșuate de conectare și blocarea contului etc.

Cel mai simplu mod de a obține un nou utilizator și implicit o nouă schemă în baza de date constă în folosirea consolei grafice *Oracle Enterprise Manager*. Primul pas este, bineînțeles, conectarea prin consola OEM la serverul bazei de date (local sau remote) folosind un cont de administrator (SYSTEM, SYS sau un alt utilizator deținând rolul DBA). Fereastra de conectare pe care o obțineți de regulă făcând click pe nodul corespunzător bazei de date sau din meniul *Navigator* → *Connect* vă va surprinde (probabil) cu următoarele rubrici:

*username* – unde puteți specifica numele SYSTEM;

*password* – parola lui SYSTEM așa cum ați precizat-o la crearea bazei de date;

*service* – numele (serviciului) bazei de date corespunzător nodului curent din zona *Navigator* a ferestrei OEM;

*connect as* – modul de conectare. Cum SYSTEM nu poate face operațiuni de genul opririi/pornirii bazei de date singurul mod valabil este *Normal*.

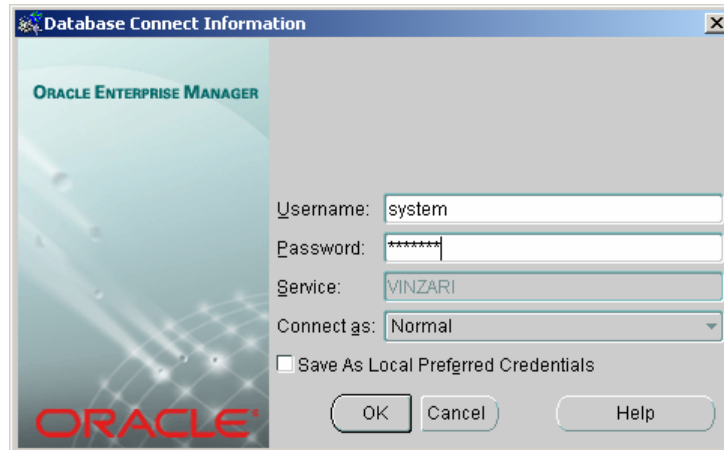
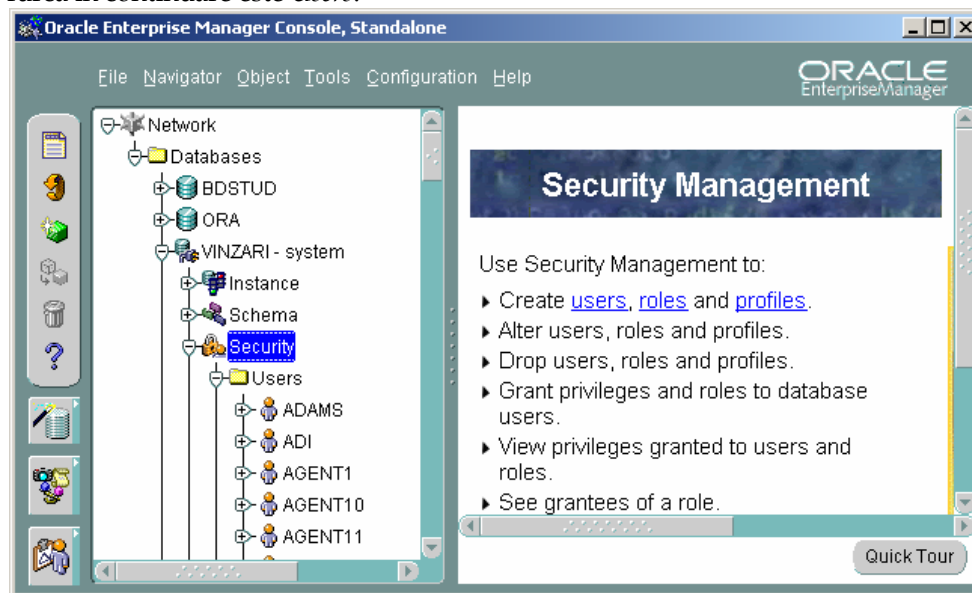


Figura 2.36. Conectați-vă ca SYSTEM

În urma unei conectări reușite (sperăm), nodul corespunzător bazei de date se va expanda și veți putea regăsi astfel un subnod intitulat sugestiv *Security*. Acesta conține trei ramuri *Users*, *Roles*, *Profiles*. După cum vă închipuiți, nodul cu care veți de furcă în continuare este *Users*.

Figura 2.37. Nodul *Security*

Fie prin click pe link-ul „create users” din dreapta, fie pe butonul *Create* din stânga, selectând apoi, din noua fereastră afișată, rubrica *users*, puteți obține fereastra *Create User*. Această fereastră conține de fapt un cadru cu mai multe pagini:

primul cadru, *General*, implică specificarea numelui, profilului (implicit cel *Default*), parolei cu confirmare (pentru modul de autentificare *password*), tablespace-urile *default* și pentru segmentele temporare;

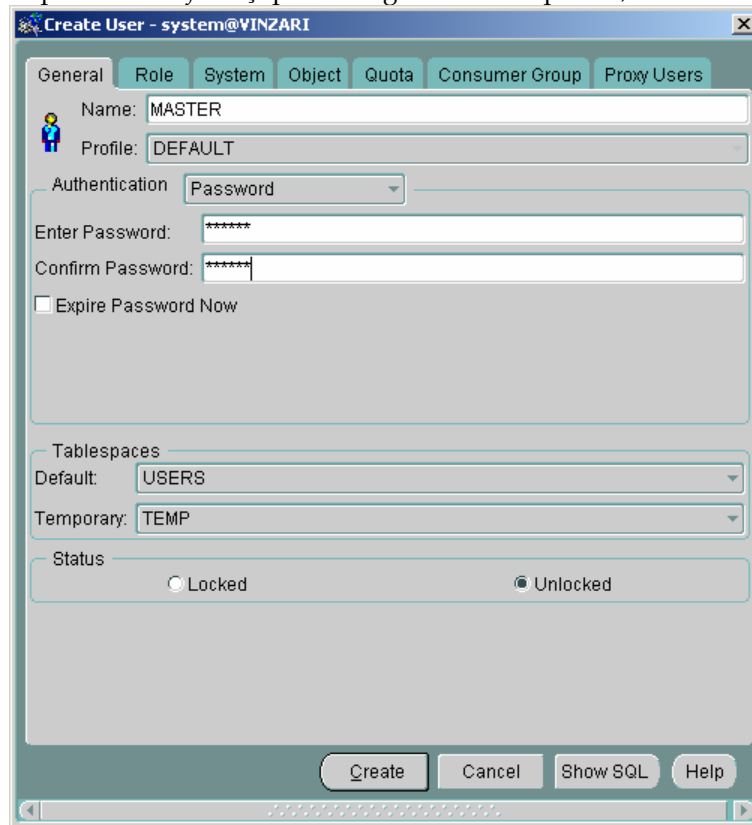


Figura 2.38. Cadrul *General* a ferestrei *Create User*

al doilea cadru, etichetat *Role*, implică și specificarea câtorva drepturi prin intermediul structurilor de tip rol (vom vedea imediat că *rol*-urile reprezintă o modalitate simplă de organizare în grupuri a privilegiilor). Implicit, consola vă invită să vă alegeți *Connect* care asigură un minim de privilegii cu privire la deschiderea unei sesiuni pe server și permisiunea creării câtorva obiecte de bază. Pentru utilizatorii care sunt implicați în dezvoltarea bazei de date și aplicațiilor care utilizează baza de date atunci se mai obișnuiește să se asigure și rolul *Resource*. Coloana *Default* din dreapta semnifică faptul că aceste roluri vor fi activate automat la conectare.

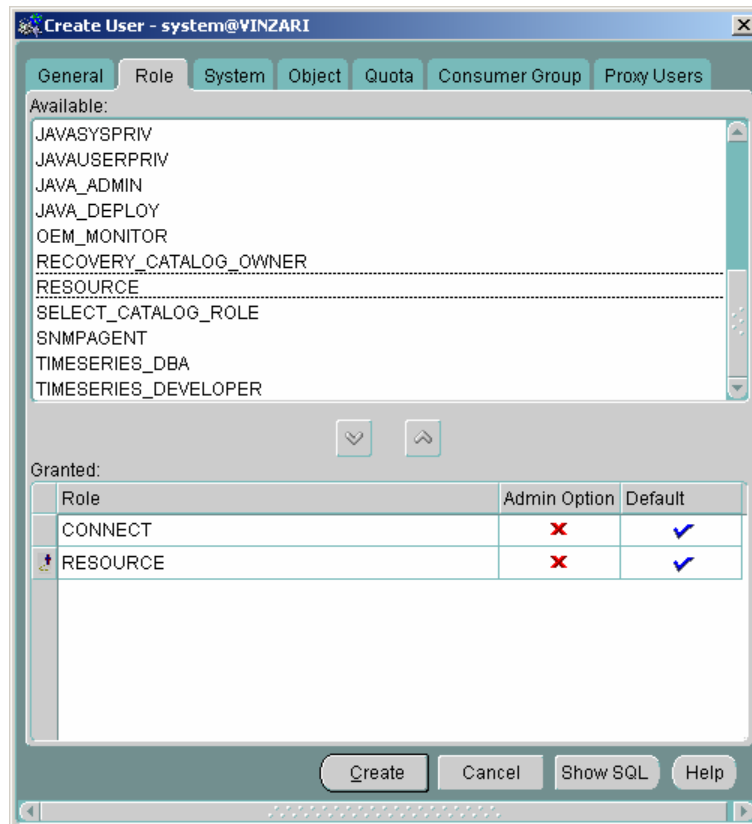


Figura 2.39. Cadrul *Role* al ferestrei *Create User*

Dacă se dorește o restricționare mai detaliată a modului în care un utilizator poate folosi spațiul rezervat pentru date se poate utiliza cadrul *Quota* în care se găsește un tabel ale cărui rubrici reprezintă fiecare tablespace în parte. Rolul *Resource* presupune implicit că utilizatorul va dispune de un privilegiu special prin care va putea crea obiecte în baza de date nerestricționat din punct de vedere al spațiului.

Butonul *Create* va determina crearea unui nou utilizator folosind specificațiile din diferitele cadre, specificații care de fapt se materializează sub forma comenzilor `CREATE USER` și `GRANT` necesare. Aceste comenzi pot fi vizualizate prin apelarea butonului *Show SQL*.

General Role System Object Quota Consumer Group Proxy Users

Name: MASTER

Profile: DEFAULT

Authentication: Password

Enter Password: \*\*\*\*\*

Confirm Password: \*\*\*\*\*

☐ Expire Password Now

Tablespaces

Default: USERS

Temporary: TEMP

Status

☐ Locked ☒ Unlocked

Create Cancel Hide SQL Help

SQL Text

```
CREATE USER "MASTER" PROFILE "DEFAULT"
IDENTIFIED BY "maitre" DEFAULT TABLESPACE "USERS"
TEMPORARY TABLESPACE "TEMP"
ACCOUNT UNLOCK;
GRANT "CONNECT" TO "MASTER";
GRANT "RESOURCE" TO "MASTER";
```

Figura 2.40. Comenzile SQL (DDC) generate pentru a „obține” un nou utilizator

Dacă în final veți obține următorul mesaj

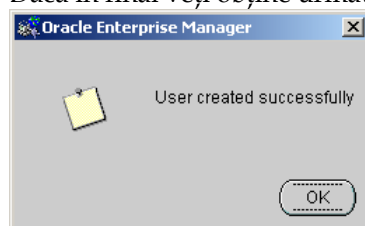


Figura 2.41. Confirmarea succesului în crearea unui nou utilizator  
puteți considera că v-ați făcut treaba acceptabil.

### 2.4.2. Privilegii și roluri

Am menționat mai sus că în procesul de creare a unui nou utilizator, pentru a-și putea realiza în baza de date operațiile pe care și le propune, acesta trebuie să dispună de anumite drepturi. Astfel că securitatea unui server de baze de date trebuie să asigure, pe lângă autentificarea utilizatorilor, și securitatea accesului la obiectele bazei de date, plus securitatea acțiunilor generice de manipulare a obiectelor sau a bazei de date în sine. În cazul Oracle *privilegiile* de care poate dispune un utilizator pentru buna desfășurare a responsabilităților sale sunt împărțite în două categorii:

*privilegii la nivel de obiecte*, care permit executarea unor comenzi specifice (UPDATE, DELETE, SELECT) asupra obiectelor existente în baza de date;

*privilegii la nivel de sistem* care se referă la permisiunea creării, ștergerii sau modificării proprietăților globale ale obiectelor (prin comenzi precum CREATE, ALTER, DROP etc.), la permisiunea conectării la serverul bazei de date (deschiderea unei sesiuni) sau, mai mult, (pentru SYS de exemplu) închiderea/deschiderea instanței bazei de date.

Responsabilitățile legate de o anumită funcțiune a unei aplicații se dovedesc, uneori, foarte complexe datorită tranzacțiilor pe care le presupune, fapt care poate implica foarte multe privilegii la nivel de obiect (n SELECT-uri pentru consultarea a n-tabele, n-UPDATE-uri sau INSERT-uri pentru actualizarea a n-tabele etc.). Din acest motiv, necesitatea organizării privilegiilor în structuri suplimentare devine imperativă. Elementul cheie în acest context sunt *rolurile*. Astfel că, pentru a se evita acordarea repetată a unei structuri coerente de privilegii, acestea pot fi grupate în roluri, iar acordarea acestora se face o singură dată pentru un utilizator. De asemenea, folosirea autorizată a acestor structuri de privilegii se poate realiza prin mecanismele de activare predefinite la nivelul rolurilor. Astfel că, după conectare, un utilizator poate dispune de anumite roluri (care i-au acordate în prealabil), iar pe parcurs, funcție de contextul responsabilităților sale în baza de date, poate solicita activarea altor roluri (care pot presupune chiar și o re-autentificare suplimentară prin asocierea unor parole specifice la nivelul lor).

Rolurile mai pot avea un rol esențial. La nivelul SGBD-ului Oracle nu există explicit noțiunea de grup de utilizatori. Grupurile de utilizatori sunt necesare însă pentru gestiunea drepturilor comune pe categorii. Pentru a avea un echivalent al acestor categorii specifice de utilizatori se pot defini tot roluri. Astfel că în funcție de rolurile acordate unui utilizator, se poate stabili din ce categorie face parte. Acestor roluri-categorii de utilizatori li se pot acorda privilegii specifice prin acordare rolurilor care delimitează diversele funcțiuni ale aplicațiilor.

Prin urmare în definiția *rolurilor* intră privilegii (obiect sau sistem), dar și alte roluri. De exemplu, rolul CONNECT conține, în primul rând, privilegiul ALTER SESSION, dar și CREATE TABLE, CREATE INDEX etc. care permit obținerea și manipularea obiectelor din schema proprie. Rolul RESOURCE conține, pe lângă privilegii asupra obiectelor proprii scheme (inclusiv proceduri și trigger-e), și privilegiul UNLIMITED TABLESPACE care îi permite să folosească spațiul din

oricare din tablespace-urile pentru date. Rolul DBA, care se acordă administratorilor bazei de date, conține atât privilegii speciale care îi permit manipularea obiectelor indiferent de proprietarul lor (independent de schemă) cum ar fi `SELECT ANY TABLE`, cât și o serie de alte roluri care îi asigură o serie de drepturi esențiale în munca de administrare cum ar fi `SELECT CATALOG ROLE` care îi permite consultarea tuturor tabelor dicționarului bazei de date.

Creare asistată a unui rol se poate face analog utilizatorilor, dintr-o opțiune *Create...* a Oracle Enterprise Manager (OEM) după care se selectează rubrica *role*. Pentru crearea, spre exemplu, a unui rol *GESTFACT* care să cuprindă privilegiile obiecte pentru consultarea tabeli *CLIENTI* și inserare în tabele *FACTEM* și *LINIIFE* se procedează astfel:

mai întâi, în cadrul *General*, al ferestrei *Create role* va fi specificat un nume pentru noul rol și eventual, dacă se dorește o autentificare suplimentară la activarea lui, atunci se poate alege din rubrica *Autentification* un mod de autentificare, cum ar fi *password*;

apoi, în cadrul *Object*, se selectează din arborele din stânga schema și apoi obiectele ale căror privilegii sunt necesare. După selectarea unui anumit obiect, în stânga va apare o listă cu privilegiile disponibile care vor fi selectate și consemnate în tabelul din partea de jos.

Comenzile necesare care vor fi generate în urma acestui proces sunt `CREATE ROLE` și `GRANT`.



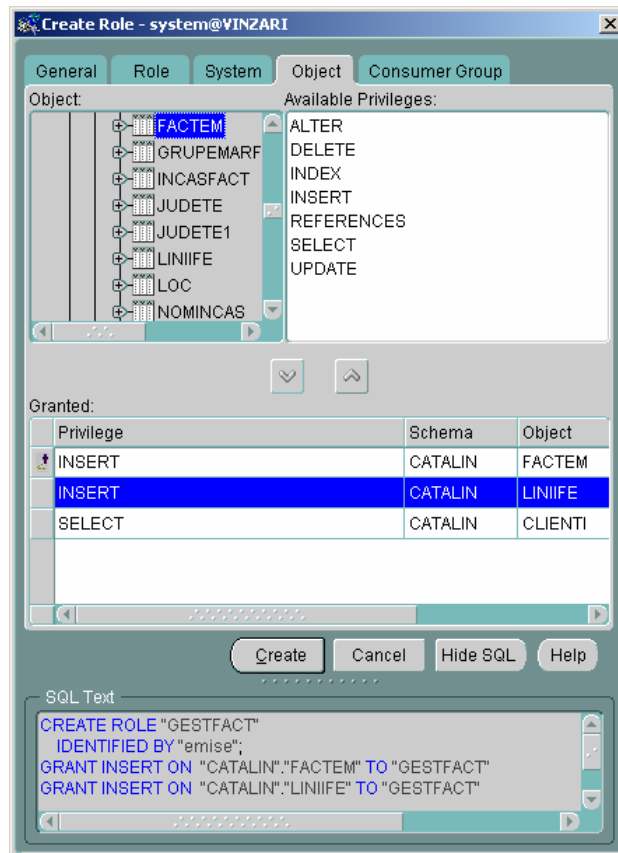


Figura 2.42 Definirea unui rol

Trebuie menționat că, pentru o astfel de operațiune, sunt necesare următoarele drepturi:

- privilegiul sistem CREATE ROLE;
- privilegiile obiect pe schema care conține obiectele *FACTEM*, *LINIIFE* și *CLIENTI*.

De asemenea, pentru ca un rol să fie util, trebuie acordat utilizatorilor care au nevoie de el. Acest lucru se realizează în cadrul *Role* al ferestrei de (re)definire a utilizatorului (sau rolului) respectiv:

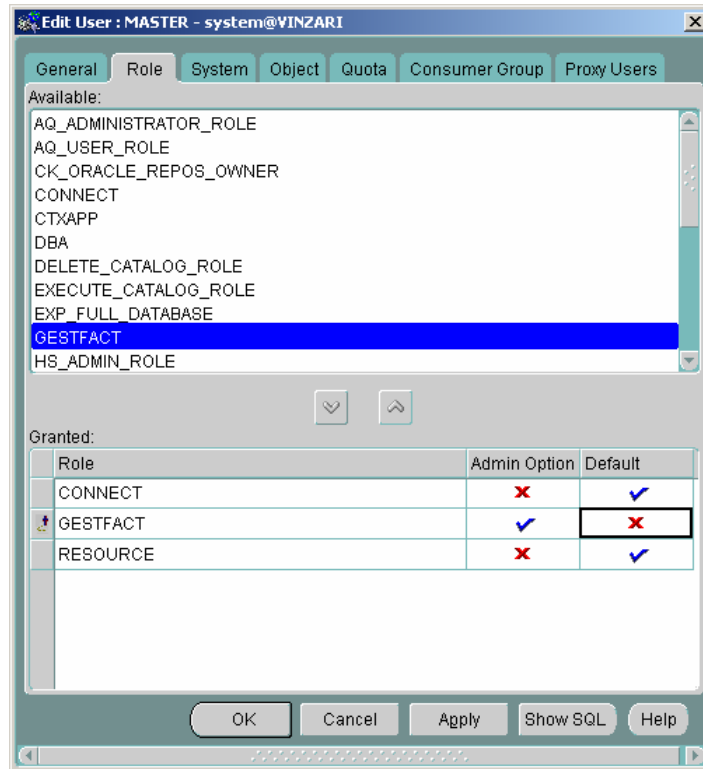


Figura 2.43. Acordarea unui Rol

O mare importanță în acest cadru o are coloana (rubrica) *Default*. „Bifarea” ei semnifică activarea implicită a respectivului rol la conectarea utilizatorului la baza de date. În caz contrar este necesară activarea rolului printr-o comandă SET ROLE:

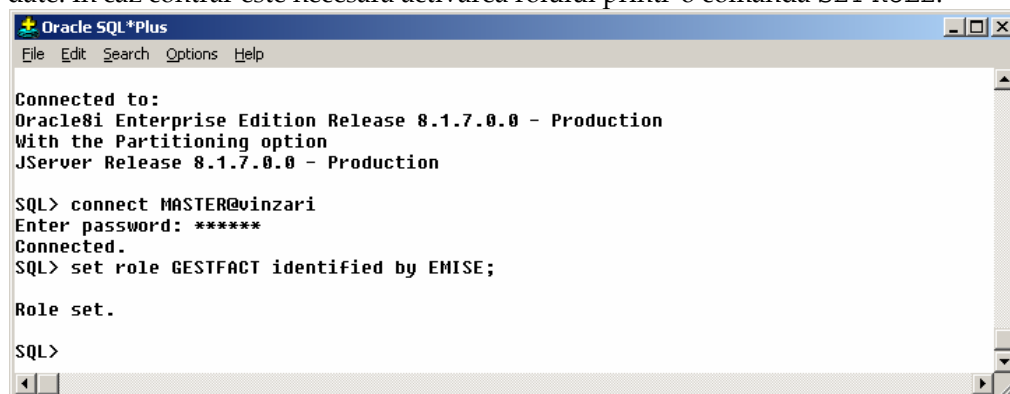


Figura 2.44. Activarea unui rol

Clauza *identified by* este necesară numai pentru rolurile care cer autentificare suplimentară (prin parolă).

## 2.5. Clienți și servere Oracle

Până acum nu am făcut considerații foarte explicite asupra modului în care trebuie configurați clienții din rețea pentru a se putea conecta la serverul pe care este rezidentă baza de date. Singurele detalii pe care le știm se referă la faptul că cererile acestora sunt gestionate inițial de un serviciu numit *Listener* și că, pentru clientul local, nu sunt necesare configurări suplimentare în afara celor implicite de la instalarea software-ului Oracle. Însă, pentru o aplicație client/server serioasă, tocmai posibilitatea separării și fizice a clienților constituie un atu.

Pentru a gestiona cadrul conectării clienților la serverele de baze de date sau, mai exact, pentru a crea posibilitatea construirii aplicațiilor pe două (sau mai multe straturi), Oracle a creat un protocol specific numit (acum, pentru 9i) *Oracle Net*. În decursul istoriei sale acest protocol a mai fost cunoscut ca *NET8* (pentru Oracle 8.0 sau Oracle 8i) sau chiar *SQL\*NET* (pentru Oracle 7, versiunea 7.3.4).

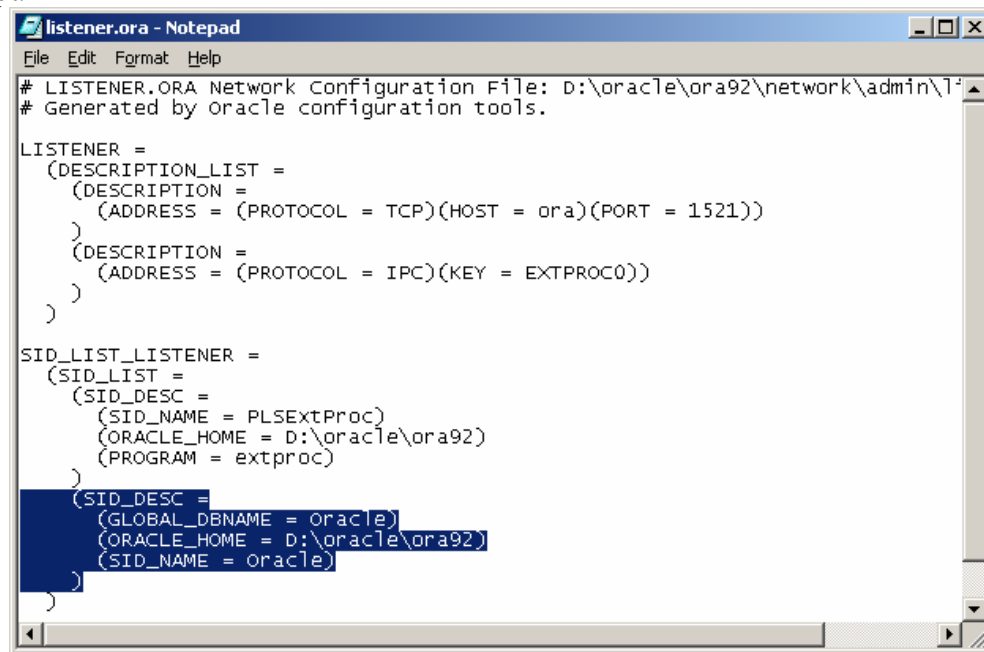
*Oracle Net* permite conectarea clasică a clienților (grei – *fat client*) la servere de baze de date, colaborarea între serverele de baze de date în arhitecturi distribuite la nivelul BD, sau conectarea serverelor de aplicații la serverele de baze de date în cadrul arhitecturilor pe trei straturi cu clienți ușori (*thin client*). Acest protocol permite, de fapt, transparența infrastructurii de rețea, făcând posibil lucrul doar cu nume, sinonime (sau pseudonime) care să nu implice detalii legate de adresele fizice ale mașinilor pe care sunt distribuite aplicațiile client și serverele.

Elementele esențiale ale arhitecturii *Oracle Net* presupun, la nivelul clientului, folosirea de către aplicații a interfeței OCI (*OCI.dll*) care asigură, faptic, stabilirea legăturii cu serverele BD. Localizarea acestor servere se face prin intermediul unor *descriptori* specifici localizați pe stratul client (vezi `%ORACLE_HOME%\network\admin\tnsnames.ora`). Sarcina lor este să „încapsuleze” detaliile de localizare fizică prin protocoalele de transport gen TCP/IP, precum și să transmită comenzile SQL către servere care au în grijă bazele de date. Legătura cu protocolul de transport al rețelei se face prin așa-numitele adaptoare, furnizate de Oracle, și care asigură independența aplicațiilor client de infrastructura de rețea existentă.

La nivelul serverului, elementul esențial în constituie serviciul *Listener* care „ascultă” pe un port specific (1521) cererile clienților, interpretează numele bazelor de date solicitate consultând o listă de identificatori interni (vezi fișierul `%ORACLE_HOME%\network\admin\listener.ora`) și, dacă pentru serviciul BD solicitat există o instanță disponibilă, creează un *proces server* care va prelua sarcina gestiunii mai departe a dialogului cu clientul care a transmis cererea inițială.

Configurarea *Listener*-ului se face de regulă implicit la instalare dacă se alege inclusiv și crearea unei baze de date. Dacă însă se alege varianta instalării doar a software-ului necesar serverului pentru administrarea BD, dar se amână crearea bazei de date, atunci în pasul explicit al configurării *Oracle Net* se cer și câteva detalii legate de *Listener* (nume, IP-ul mașinii server, portul serviciului *Listener* – implicit 1521).

Este util să verificați, la nivelul serverului BD, în *Control Panel* → *Administrative Tools* → *Services* existența serviciului *Oracle{HOME}TNSListener*. Dacă nu există creați-l cu *Net Configuration Assistant* opțiunea *Listener Configuration*. De asemenea, pentru că serviciul *listener* trebuie să fie „conștient” de existența noii baze de date, în fișierul său de configurare trebuie să existe o intrare care să consemneze acest fapt.



```

listener.ora - Notepad
File Edit Format Help
# LISTENER.ORA Network Configuration File: D:\oracle\ora92\network\admin\listener.ora
# Generated by Oracle configuration tools.

LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = TCP)(HOST = ora)(PORT = 1521))
    )
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC0))
    )
  )

SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = D:\oracle\ora92)
      (PROGRAM = extproc)
    )
    (SID_DESC =
      (GLOBAL_DBNAME = oracle)
      (ORACLE_HOME = D:\oracle\ora92)
      (SID_NAME = oracle)
    )
  )

```

Figura 2.45. Fișierul de configurare al serviciului *listener*, la nivelul serverului

Dacă această informație nu există, puteți să o adăugați (respectând sintaxa sugerată în figura de mai sus), după care va trebui repornit serviciul *Oracle{HOME}TNSListener* din utilitarul *Services* al sistemului de operare.

Configurarea clienților se face:

- fie la instalarea software-ului serverului Oracle pentru clientul local;
- fie la instalarea software-ului client pentru o mașină din rețea care va găzdui o aplicație locală (sau un server de aplicații);

fie prin invocarea utilităților *Net Configuration Assistant*, sau *Net Manager* (mai complet dar mult mai complex) din meniul *Configuration and Migration Tools* al grupului de programe legate de Oracle 9i.

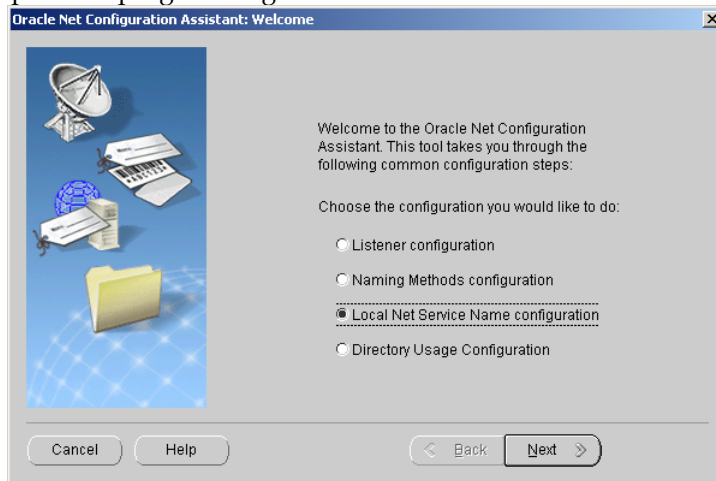


Figura 2.46. Fereastra de întâmpinare a utilitarului *Oracle Net Configuration Assistant*

Pentru configurarea unui descriptor de pe un client care să puncteze la o bază de date instalată pe un server din rețea se alege din fereastra de întâmpinare a *Oracle Net Configuration Assistant* opțiunea *Local Net Service Name configuration*. În pasul următor se optează pentru crearea unui descriptor-serviciu nou opțiunea *Add*. Pasul al treilea presupunea specificarea versiunii bazei de date care urmează să fie accesată, compatibilitatea cu versiunile anterioare fiind asigurată după toate aparențele până la nivelul 8.0. Dacă am optat pentru *Oracle 8i or later database or service* (știind că baza noastră de date este 9i), atunci în ecranul următor ni se va solicita numele acesteia așa cum l-a specificat în rubrica *Global Database Name* din pasul 5 al secțiunii de instalare (paragraful 2.3.5).

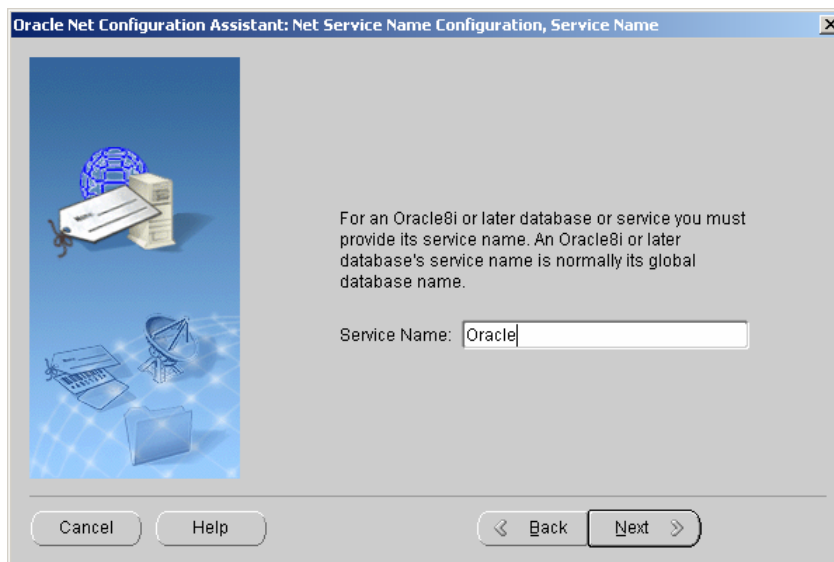


Figura 2.47. Specificarea numelui bazei de date

Apoi vom specifica (de fapt, vom alege dintr-o listă) numele protocolului de transport al rețelei – *TCP* în majoritatea cazurilor astăzi.

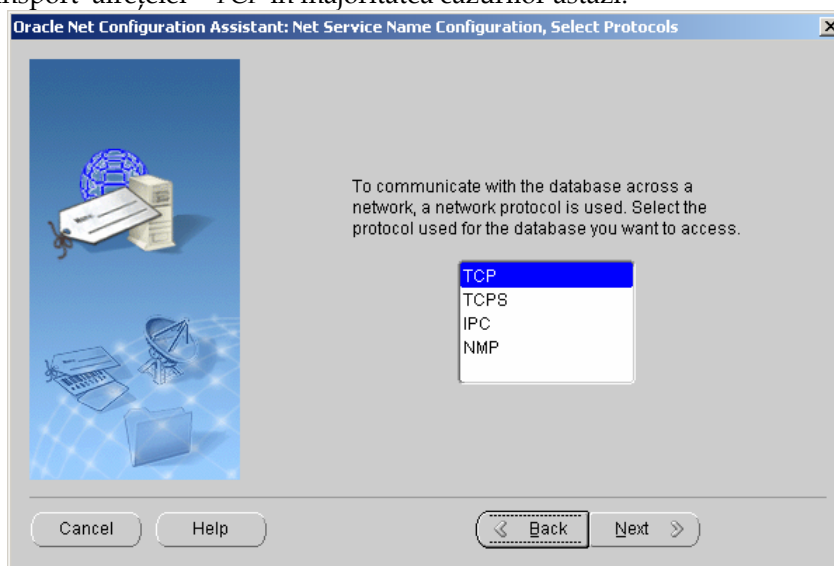
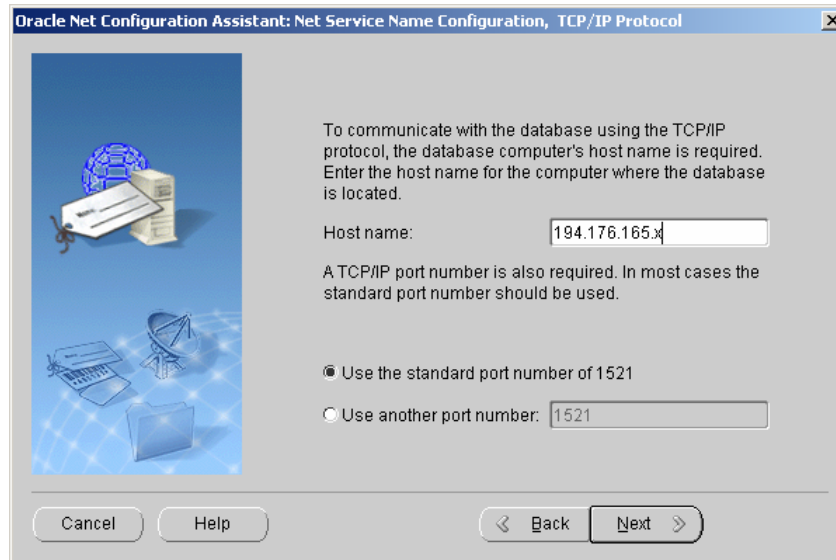


Figura 2.48. Specificarea TCP ca protocol al rețelei

În pasul următor (în funcție de protocolul de rețea ales) va fi solicitată adresa mașinii-server (adresa IP sau numele DNS) pe care se găsește serviciul *Listener* și portul pe care ascultă acesta (implicit 1521):

Figura 2.49. Specificarea adresei și portului *Listener*-ului

În fine, putem eventual testa mai întâi conexiunea, iar în ultimul pas ni se va cere un *nume* pentru descriptorul-serviciu creat care poate fi același cu numele bazei de date (se recomandă), sau ... nu. Dacă numele serviciului pe mașina clientului va fi altul decât numele bazei de date, atunci la *host name* vom specifica întotdeauna numele serviciului local ale cărui detalii le-am precizat mai înainte.

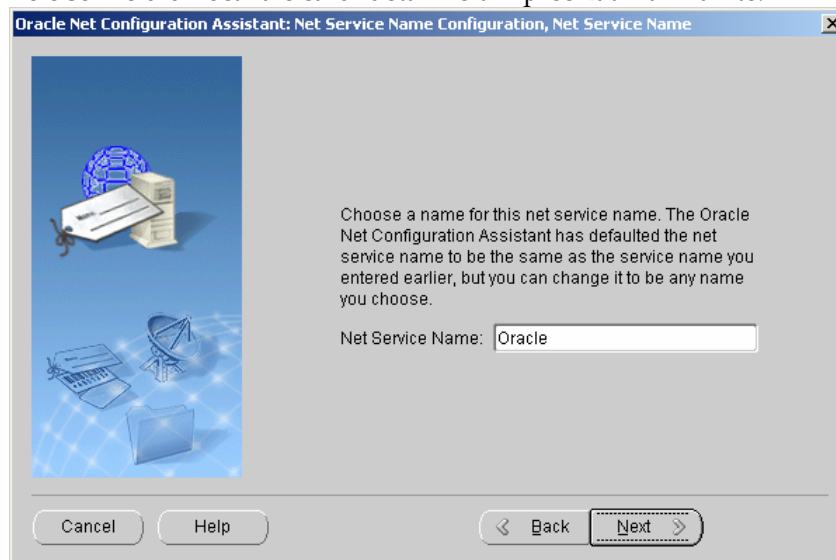
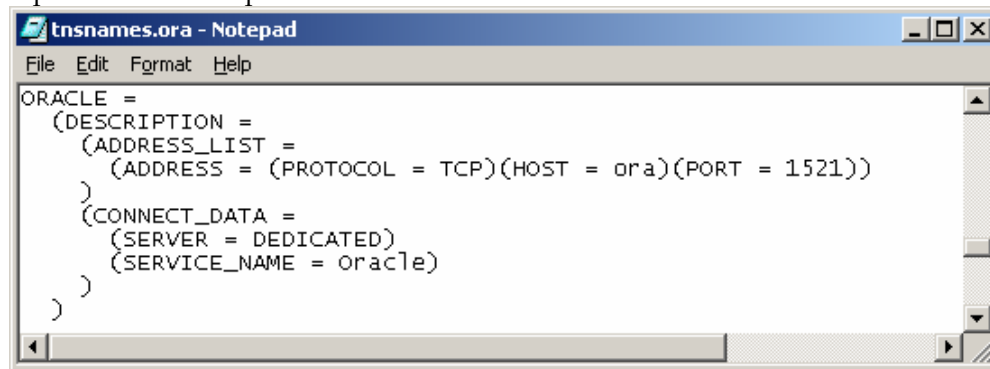


Figura 2.50. Specificarea numelui descriptorului-serviciu care poate să fie același cu numele bazei de date sau nu

După parcurgerea acestui proces, aplicațiile instalate local vor putea avea acces la baza de date folosind numele serviciului *Oracle*, fără a avea astfel nevoie de detalii suplimentare despre configurarea rețelei.

Ca rezultat al secvenței de configurare anterioare, informația privitoare la conectarea clienților este păstrată în %ORACLE\_HOME%\network\admin\tnsnames.ora, în care vom găsi și intrarea corespunzătoare descriptorului *Oracle*:



```
tnsnames.ora - Notepad
File Edit Format Help
ORACLE =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = ora)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = oracle)
    )
  )
```

Figura 2.51. Fișierul de configurare *tnsnames.ora*, la nivelul clienților