

# Arhitectura internă a bazei de date Oracle

Arhitectura internă Oracle descrie modul în care componentele software și hardware colaborează pentru a asigura stocarea, accesarea, procesarea și protejarea datelor. Această arhitectură este construită pe mai multe niveluri bine definite: fișiere, memorie și procese, toate coordonate de instanța Oracle. Înțelegerea acestor mecanisme este esențială pentru administrarea, optimizarea și depanarea bazelor de date Oracle.

## Instanța Oracle și baza de date

Oracle face distincția clară între: Baza de date Oracle - colecția de fișiere fizice (data files, control files, redo logs) și Instanța Oracle – structura de memorie + procesele care accesează baza de date. O bază de date poate exista fără instanță, dar instanța este cea care permite utilizarea bazei de date.

## Nivelul fișierelor

Aici sunt stocate fișierele care nu fac parte din baza de date. Deși nu conțin date propriu-zise, aceste fișiere sunt esențiale pentru funcționarea sistemului:

### 1. Fișiere de parametri

- init.ora sau spfile;
- conțin setările instanței;
- sunt citite la pornirea instanței.

### 2. Fișiere de parole

- permit autentificarea administratorilor;
- utilizate la pornirea bazei fără acces la dicționar.

### 3. Fișiere de arhivă (Archived Redo Logs)

- copii ale redo log-urilor;
- utilizate pentru recuperare completă.

### 4. Fișiere de alertă și trace

- loghează evenimente și erori;
- esențiale pentru diagnosticare.

## Procesele Oracle

Procesele sunt componentele active care execută operațiile necesare funcționării bazei de date.

### Procesele utilizator (User Processes)

- sunt inițiate de aplicații sau utilizatori;
- gestionează interacțiunea cu utilizatorul;
- trimit cereri SQL către Oracle.
- Procesele utilizator nu accesează direct fișierele bazei de date.

### Procesele Oracle (Server Processes)

Procesele Oracle intermediază comunicarea dintre utilizatori și baza de date. Tipuri:

- Dedicated Server – un proces per sesiune;
- Shared Server – mai multe sesiuni folosesc procese comune.

Responsabilități:

- parsează instrucțiuni SQL;

- citesc datele din memorie sau disc;
- returnează rezultatele.

## Procesele de fundal (Background Processes)

Aceste procese lucrează în fundal pentru a menține integritatea și performanța bazei de date. Cele mai importante:

- DBWn (Database Writer). Scribe blocurile modificate din buffer cache în fișierele de date.
- LGWR (Log Writer). Scribe informațiile de redo în fișierele redo log.
- CKPT (Checkpoint). Sincronizează datele scrise pe disc.
- SMON (System Monitor). Recuperează instanța după o cădere.
- PMON (Process Monitor). Curăță procesele eșuate și eliberează resursele.
- ARCn (Archiver). Arhivează redo log-urile.

## Arhitectura memoriei Oracle

Memoria este organizată în două zone distincte:

### 1. Zona de memorie de sistem (SGA – System Global Area)

SGA este o zonă partajată de memorie, accesibilă tuturor proceselor Oracle. Componentele principale ale SGA:

- Database Buffer Cache. Stochează blocurile de date citite sau modificate.
- Shared Pool. Conține: library cache (SQL parse), data dictionary cache.
- Redo Log Buffer. Stochează temporar modificările înainte de scrierea în redo logs.
- Large Pool. Utilizat pentru operații mari (RMAN, paralelism).
- Java Pool / Streams Pool. Suport pentru Java și replicare.

## 2. Zona de memorie de program (PGA – Program Global Area)

PGA este alocată individual fiecărui proces server. Conține:

- informații despre sesiune,
- zone pentru sortări,
- stări de cursori,
- variabile locale.

Diferența esențială: SGA este partajată, PGA este privată.

## Crearea unei baze de date Oracle

Procesul implică:

- Crearea fișierelor de parametri.
- Pornirea instanței în modul NOMOUNT.
- Crearea fișierelor de control.
- Crearea fișierelor de date.
- Inițializarea dicționarului datelor.

Se realizează cu comanda:

CREATE DATABASE ...

## Pornirea unei baze de date Oracle

Pornirea se face în etape:

1. NOMOUNT
  - se pornește instanța,
  - se încarcă SGA și procesele.
2. MOUNT
  - se atașează fișierele de control,
  - baza nu este încă accesibilă.

### 3. OPEN

- fişierele de date şi redo logs sunt deschise,
- baza este accesibilă utilizatorilor.

STARTUP;

## Oprirea unei baze de date Oracle

Moduri de oprire:

- SHUTDOWN NORMAL – aşteaptă utilizatorii.
- SHUTDOWN IMMEDIATE – recomandat.
- SHUTDOWN TRANSACTIONAL – finalizează tranzacţiile.
- SHUTDOWN ABORT – oprire forţată.

## Exerciţii

- a) Avem mai mulţi alergători care participă la diferite curse de maraton. Creaţi tabelele RUNNERS respectiv RACES care să stocheze informaţii despre participanţi respectiv evenimente
- b) Introduceţi tabelul asociativ PARTICIPATIONS care contorizează pentru fiecare sportiv rezultatul la fiecare cursă. Introduceţi un trigger care să prevină ca 2 sportivi diferiţi să termine pe aceeaşi poziţie la aceeaşi cursă. Dacă un sportiv a terminat deja pe o poziţie ce vreţi să fie inserată, inseraţi N/A în locul acesteia.
- c) Populaţi tabelele cu 5/6 intrări fiecare
- d) Implementaţi o funcţie care să calculeze un scor pentru fiecare sportiv activ. Scorul este calculat astfel: (victorie într-o cursă: 10 puncte, loc 2: 6 puncte, loc 3: 4 puncte, loc 4: 2 puncte, loc 5: 1 punct).
- e) Sortaţi toţi alergătorii după numărul de puncte şi afişaţi clasamentul rezultat.
- f) Implementaţi o excepţie predefinită care să apară atunci când încercăm să calculăm scorul unui sportiv care nu a participat la nicio cursă dar este trecut ca activ.

## Problems

- a) We have several runners who participate in different marathon races. Create the tables RUNNERS and RACES to store information about the participants and the events, respectively.
- b) Introduce the associative table PARTICIPATIONS which records, for each athlete, the result obtained in each race. Create a trigger that prevents two different athletes from finishing in the same position in the same race. If an athlete has already finished in the position that is about to be inserted, insert N/A instead.
- c) Populate the tables with 5–6 entries each.
- d) Implement a function that calculates a score for each active athlete. The score is calculated as follows: race win: 10 points, 2nd place: 6 points, 3rd place: 4 points, 4th place: 2 points, 5th place: 1 point
- e) Sort all runners by the number of points and display the resulting ranking.
- f) Implement a predefined exception that is raised when attempting to calculate the score of an athlete who is marked as active but has not participated in any race.