

ACID – proprietati la tranzactii

Atomic. Daca la o tranzactie facem mai multe update, insert etc. modificarie la date vor fi aplicate toate sau nici una. Fiecare tranzactie este socotitica ca o unitate care fie e executata, fie nu. Tot ce face tranzactia, o face asupra la toate datele deodata, nu doar la unele.

Consisty – tranzaction manager ne ofera garantia de a avea date consistente pana si dupa la tranzactie, adica nicio data nu se pierde nicaieri

Isolation – izolarea la tranzactii. O tranzactie nu poate vedea ce face alta.

Durability – odata efectuata, modificarile unei tranzactii sunt puse pe memoria permanenta, adica interna, si chiar de sistemul cade, ele mereu vor fi pastrate acolo

* Fenomene la tranzaction izolation level:
* Dirty Read – situatie in care o tranzactie citeste date care inca nu sunt committed. Daca T1 modifica o coloana, dar nu da commit, T2 va putea citit aceste valori modificate, desi T1 nu a dat commit, dar daca T1 nu va da commit, T2 a lucrat cu date inexistente.
* Non repeatable read – situatie cand o tranzactie citeste aceeasi linie de 2 ori, dar primeste rezultate difertie. De ex, T1 citeste o linie, si face ce are de facut. Acum T2 modifica linia data. Daca T1 o citeste din nou, va oferi rezultate diferite
* phantom read – apare cand 2 aceleasi queries sunt executate, dar liniile pe care ambele le iau sunt diferite.
* Sunt 4 nivele de izolation:
* READ UNCOMIMITED – o tranzactie poate citi modificarile uncommited facute de alta tranzactie si permite dirty reads
* READ COMMITED – o tranzactie poate citi doar modificarile commited ale altei tranzactii. Tranzactia tine lock asupra la linia cu care lucreza la momentul dat si nimeni nu o poate accesa pana ea nu elibereaza row.
* REPEATABLE READ – o tranzactie tine lock asupra la toate coloanele si liniile cu care lucreaza, si nicio alta tranzactie nu le poate modifica sau accesa pana prima nu elibereaza lockul.
* SERIALIAZABLE – tranzactiile sunt executate in mod secvential, adica cate una rand pe rand.
* Pentru a modifica nivelul de izolation in Oracle, se foloseste:

SET TRANSACTION ISOLATION LEVEL <LEVEL>

ALTER SESSION SET ISOLATION LEVEL <LEVEL>

* Cel default e READ COMMITTED

**DML STATEMENTS**

* Insert
* Update
* Delete
* Merge – insert si update, adica daca sunt deja prezente, sunt doar updated, daca nu, sunt inserate
* **Atentie! In Oracle nu putem insera mai multe valori folosind Insert!!!**
* **E bine sa scrie commit dupa fiecare tranzactie,sau operatii ce nu depind de altele, desi oricum daca tranzactia odata reuseste, se va da autocommit**

**SELECT**

* ROWNUM – numarul coloanei selectate. Alternativa la limit

Putem folosi ROWNUM pentru a limita cu conditii numarul de tuples, gen WHERE ROWNUM < 5

SELECT \* FROM test WHERE RAWNUM <= 4;

* OFFFSET Nr ROWS – Ignora primele Nr rows

SELECT \* FROM test OFFSET 4 ROWS;

* Oracle are o tabela numita dual, si ea nu are valori sau coloane, e doar penru a testa functii
* Putem pune NULL FIRST sau NULL LAST ca ASC sau DESC
* In Orale SQL server nu putem combina RAWNUM cu OFFSET ROWS, si se pune ROWS FETCH NEXT X ROWs ONLY

SELECT \* FROM test OFFSET 2 ROWS FETCH NEXT 2 ROWS ONLY;

* Nu punem AS dupa numele la tabel:

FROM employees ~~AS~~ E1

* Putem sa Punem AS la numele la coloana

SELECT E1.first\_name,E2.first\_name AS "Manager\_First\_name"

**Function**



* trim(string) – elimina spatiile libere in plus de la inceput si final
* ltrim(string) – elimina spatiile libere de la inceput
* rtrim(string) – elimina spatiil libere de la final
* lpad(string, size, stringToAdd) – adauga in stanga la string stringul stringToAdd de atatea ori ca stringul final sa aiba size stabilit
* rpad() – ca lpad, doar ca la dreapta
* initcap(string) – adauga litera mare la ineput de fiecare cuvant
* instr(stringInitial, stringDeCautat) – returneaza pozitia la stringul gasit
* substr(string, pozitia, nrCaractere) – extrage nr de caractere de la pozitia indicata,
* round(nr, Pozitie) – va aproxima de la pozitie, gen: round(1.996,2) = 1.997 sau round(1.9994,3) = 1.9990
* ceil(numar) – taie cifrele de dupa virgula si aduna cu 1
* floor(numar) – taie cifrele de dupa virgula si scade cu 1



**Variables**

DEFINE variabila = 10;

SELECT &variabila FROM dual;

**Function**

CREATE FUNCTION Functie

(param1 number, param2 number)

return number AS

BEGIN

return param1+param2;

END;

Dupa parametri, pot fi puse urmatoarele cuvinte cheie:

In – parametrul poaet fi citit, dar nu poate fi modificat

OUT – parametrul nu poate fi citit, dar poate sa i se modifice valoarea

IN OUT – parametrul poate fi si ctit si modificat

Procedura

CREATE PROCEDURE multiplica(ident LONG, valoare varchar)

IS

BEGIN

INSERT INTO test VALUES(ident, valoare);

END;

begin

multiplica(50,'test50');

end;

Atentie! Procedura o executam intre

begin

....

end;

**Procedura vs Function**

* O functie mereu returneaza o valoare cu RETURN, dar o procedura poate returna sau poate sa nu returne ceva
* O funtie poate fi folosita cu SEELCT, INSERT etc. dar o procedura nu poate fi invocata in SQL statements

**Join**

* Dupa Join putem sa folosim using

SELECT first\_name, last\_name, salary, job\_title  
FROM employees  
 LEFT JOIN jobs USING (job\_id);

* Dar putem folosi si on, nu e nicio problema

**Key auto increment**

cardNr NUMBER(6) GENERATED ALWAYS AS IDENTITY (START WITH 10000 INCREMENT BY 1) PRIMARY KEY,

**Triggers**

* Pentru a accesa valorile sterse, folosim :old si cele noi :new
* Pentru a afla daca vreo valoare a fost stearsa sau adaugata folosim:

if inserting, if updating sau if deleting