UNIVERSITATEA DIN BUCUREȘTI

FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

SPECIALIZAREA INFORMATICĂ

**DATA WAREHOUSE & BUSINESS INTELLIGENCE**    
**- ANALIZA -**

Autor

Tiberius Coman

George Banica

Daniela Alexandra Constantin

Sebastian Alexandru Velciu

Profesor titular

Lect. Dr. GABRIELA MIHAI

   
GRUPA 505

ANUL II MASTER, SEMESTRUL I

**Exercitiul 1**

Prezentarea concisă a bazei de date (utilizarea ei) :

Sistemul de plati permite crearea de conturi si carduri aferente pentru persoane fizice pe care ii vom numi generic clienti. Acestia pot face cumparaturi la diferiti comercianti care sunt inregistrati in prealabil in sistem. Comerciantii trebuie sa fie la randul lor clienti ai aplicatiei si necesita inregistrarea unui dispozitiv POS prin care clientii pot face cumparaturi. De asemenea in cazul in care comerciantii nu detin un POS fizic pot inregistra site-ul pe care il folosesc pentru a comercializa produsele. Acest sistem este cu circuit inchis in sensul ca, nu se acepta tranzactii pentru clienti ale caror carduri nu sunt cunoscute sau comercianti ale caror dispozitive nu sunt regasite in sistem.

**Exercitiul 2**

a) Realizarea diagramei entitate-relație (ERD).

In urma analizarii modelului s-au creat urmatoarele relatii:

CLIENT(id\_client#, nume, tip\_client, data\_inscriere, data\_incetare);

CONT(id\_cont#, id\_card#, tip\_card, data\_emitere, data\_expirare, numar\_card, tip\_cont, nume\_cont, sold, data\_creare, data\_inchidere);

TRANZACTII(id\_tranzactie#, suma, cod\_cont\_debitor, cod\_cont\_creditor, data\_initiere, data\_procesare, stare);

COMERCIANT(id\_comerciant#, cod\_client, nume, data\_inscriere, data\_incetare );

CANAL\_PLATA(cod\_comerciant, tip\_echipament, cod\_cont, data\_inceput, data\_incetare, locatie);

b) Transformarea sistemului conceptual într-un design logic, subliniind relaţiile dintre

tabele, cheile primare şi străine (externe).

Diagram

Description automatically generated

c) Implementarea tabelelor în Oracle, folosind chei primare, constrângeri de referinţă şi

domeniu

Diagram

Description automatically generated

**Exercitiul 3**

1. Proiectarea logica de depozitului de date

Business Intelligence pentru sistemele de plată poate reprezenta utilizarea datelor pentru a obține informații despre sistemele de plată și operațiunile acestora. Scopul este de a înțelege performanța și eficiența sistemelor de plată și de a lua decizii bazate pe date care pot îmbunătăți experiența generală pentru utilizatori și pot crește profitabilitatea sistemului.

Unele aplicații comune ale business intelligence în sistemele de plată includ:

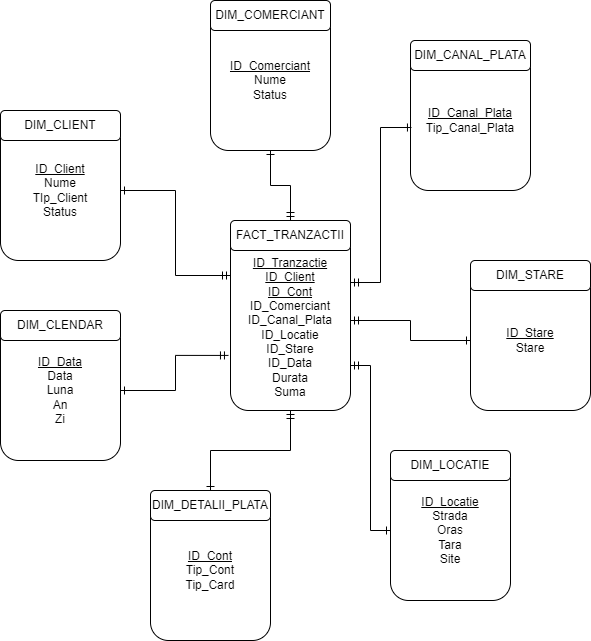
* **Optimizarea plăților**: sistemul de BI poate fi utilizat pentru a analiza datele tranzacțiilor pentru a identifica blocajele și ineficiența proceselor de plată și pentru a face îmbunătățiri bazate pe date care cresc eficiența și reduc timpul de procesare.
* **Analiza comportamentului cliențilo**r: sistemul de BI poate fi utilizat pentru a analiza comportamentul și preferințele clienților, pentru a înțelege nevoile acestora și pentru a face recomandări bazate pe date pentru noi produse și servicii;
* **Gestionarea riscurilor**: Business intelligence poate fi utilizat pentru a analiza datele de plată pentru a identifica și gestiona riscul, inclusiv riscul de credit, riscul de piață și riscul operațional;
* **Tendințele de plată și forecasting**: datele și analizele pot fi utilizate pentru a înțelege tendințele de plată și pentru a face predicții despre plățile și veniturile viitoare.

Astfel, ne propun ca modelul nostru sa poata permita:

* Analiza sumelor tranzacționate în funcție de clienți, comercianți, modalități de plata și locații sau timp;
* Analiza duratelor de soluționare a transferurilor;
* Analiza numărului de transferuri eșuate.

Pentru a obține abilitatea de a analiza aceste metrici, vom propune următorul model pentru depozitul de date:

* FACT\_TRANZACȚII:
  + **ID\_Tranzactie** - câmp ce provine din tabela TRANZACȚII, ne va ajuta sa identificam tranzacția în cadrul careia s-au realizat plata și încasarea sumelor;
  + **ID\_Client** - câmp ce provine din tabela CLIENT, stabilește o legătură cu DIM\_CLIENT și ne va ajuta sa analizăm datele în funcție de clienți;
  + **ID\_Cont -** câmp ce provine din tabela CONT, stabilește o legătură cu DIM\_DETALII\_PLATA și ne va ajuta sa analizăm datele în funcție tipul conturilor și tipurile de carduri.
  + **ID\_Comerciant** - câmp ce provine din tabela COMERCIANT și stabilește legătura cu DIM\_COMERCIANT și ne va ajuta sa analizăm datele în funcție de comercianti;
  + **ID\_Canal\_Plata** - câmp generat în funcție de modalitatea de plata, codul 1 pentru plata la POS, codul 2 pentru plăți ONLINE pentru coloana Tip\_Echipament din tabela CANAL\_DE\_PLATA, stabilește legătura cu DIM\_CANAL\_DE\_PLATA;
  + **ID\_Locatie** - câmp ce provine din tabla LOCATIE și stabilește legătura cu DIM\_LOCATIE și ne va ajuta sa analizăm datele în funcție de locurile unde s-au făcut transferurile.
  + **ID\_Stare** - câmp generat în funcție de starea transferului, codul 1 pentru transferuri INIȚIATE, codul 2 pentru transferuri PROCESATE, codul 3 pentru transferuri ERONATE, pe baza coloanei STARE din table TRANZACȚII, stabilește legătura cu DIM\_STARE.
  + **ID\_Data** - câmp generat pe baza coloanei DATA\_INIȚIERE din tabela TRANZACȚII, în format “YYYYMMDD”, va fi folosit pentru a stabili legatura cu DIM\_CALENDAR și ne va permite sa analizăm datele în funcție de timp.
  + **Suma** - câmp ce provine din tabela TRANZACȚII, va avea volare negativa pentru plati si valoare pozitiva pentru incasari;
  + **Durata** - câmp calculat pe baza coloanelor DATA\_INITIERE și DATA\_PROCESARE din tabela TRANZACȚII, reprezentand durata în minute de la inițiere pana la procesare.
* DIM\_CLIENT:
  + **ID\_Client** - câmp ce provine din tabela CLIENT, este identificatorul unic pentru un client al sistemului de plata;
  + **Nume** - câmp ce provine din tabela CLIENT, reprezinta numele clientului si va permite construirea unor rapoarte sau vizualizari în funcție de clienți.
  + **Tip\_Client** -câmp ce provine din tabela CLIENT, reprezinta statul juridic al clientului (persoana fizica sau persoana juridica), va permite contruirea de vizualizari in funcție de aceasta clasificare.
  + **Status** - câmp calculat pe baza coloanelor Data\_Inscriere și Data\_Incetare din tabela CLIENT și va oferii informații despre statutul actual al clientului, dacă este client actual sau a încetat utilizarea sistemului de plata.
* DIM\_COMERCIANT:
  + **ID\_Comerciant** câmp ce provine din tabela COMERCIANT, este identificatorul unic pentru un comerciant al sistemului de plata;
  + **Nume** - câmp ce provine din tabela COMERCIANT, reprezinta numele comerciantului si va permite construirea unor rapoarte sau vizualizari în funcție de comercianti;
  + **Status** - câmp calculat pe baza coloanelor Data\_Inscriere și Data\_Incetare din tabela COMERCIANT și va oferii informații despre statutul actual al clientului, dacă este client actual sau a încetat utilizarea sistemului de plata.
* DIM\_CANAL\_PLATA:
  + **ID\_Canal\_Plata** - câmp generat pe baza coloanei Tip\_Echipament din tabela CANAL\_PLATA cu scopul de a deveni un identificator unic pentru metodele de plata în DIM\_CANALE\_PLATA. În același mod, pentru campul ID\_Canal\_Plata este populat în FACT\_TRANZACTII, astfel incat sa putem stabilii o legatura dintre cele doua tabele, pentru a permite analiza datelor în funcție de modalitățile de plata;
  + **Tip\_Canal\_Plata** - câmp ce provine din tabela CANAL\_PLATA, reprezinta modalitatea de plata si va permite construirea unor rapoarte sau vizualizari în funcție de aceasta.
* DIM\_STARE:
  + **ID\_Stare** - câmp generat pe baza coloanei Stare din tabela TRANZACȚII cu scopul de a deveni un identificator unic pentru starea transferurilor în DIM\_STARE. În același mod, pentru câmpul ID\_Stare este populat în FACT\_TRANZACTII, astfel incat sa putem stabilii o legatura dintre cele doua tabele, pentru a permite analiza datelor în funcție de stările transferurilor;
  + **Stare** - câmp ce provine din tabela TRANZACTII, reprezinta starea unui transfer si va permite construirea unor rapoarte sau vizualizari în funcție de aceasta.
* DIM\_LOCAȚIE:
  + **ID\_Locatie** - câmp ce provine din tabela LOCAȚIE, este identificatorul unic pentru un comerciant al sistemului de plata;
  + **Oraș, Țara, Strada, Site** - câmpuri ce provin din tabela LOCAȚIE fără a aplica transformări asupra datelor;
* DIM\_DETALII\_PLATA:
  + **ID\_Cont** - câmp ce provine din tabela CONT, este identificatorul unic pentru un cont în cadrul sistemului de plata și ne va permite sa stabilim o legatura cu FACT\_TRANZACTII pentru analiza datelor în funcție de tipurile conturilor și cardurilor folosite pentru transferuri;
  + **Tip\_Cont** - câmp ce provine din tabela CONT, reprezinta tipul contului din / către a fost făcut transferul;
  + **Tip\_Card** - câmp ce provine din tabela CARD, reprezinta tipul cardului din / către a fost făcut transferul.
* DIM\_CALENDAR:
  + **ID\_Data** - câmp generat ce contine data in format “YYYYMMDD”, prin care putem identifica unic o data și prin care putem stabili o legătură cu FACT\_TRANZACTII;
  + **Data** - câmp generat ce contine data în format “DD/MM/YYYY”
  + **Luna** - câmp generat ce contine luna în format “Month”;
  + **An** - câmp generat ce contine anul în format “YYYY”;
  + **Zi** - câmp generat ce contine anul în format “DD”;

****

**Exercitiul 4**

Diagram

Description automatically generated

**Exercitiul 5**

C1: Avem tabela de fapte FACT\_TRANZACTII având următoarele atribute: ID#, ID\_Tranzactie, ID\_Client, ID\_Cont, ID\_Comerciant, ID\_Canal\_Plata, ID\_Stare, ID\_Locatie, ID\_Data, Durata, Suma. Coloanele ID\_Client, ID\_Cont, ID\_Comerciant, ID\_Canal\_Plata, ID\_Stare, ID\_Locatie si ID\_Data identifică în mod unic o înregistrare din această tabelă. Un mod potrivit pentru un depozit de date de a defini aceasta constrângere de unicitate este prezentat in documentul „Sursa”, punctul 5 (cod referință c1).

Justificare: Am ales aceasta abordare deoarece opțiunea DISABLE ne scutește de costurile unui index unic (care nu a fost creat), dar in același timp asigura unicitatea datelor. Având in vedere faptul ca tabela FACT\_TRANZACTII este actualizata printr-un proces ETL, perechea de constrângeri DISABLE VALIDATE nu incomodează: nu mai sunt permise operații LMD, constrângerea nu este activa, dar s-a realizat validarea datelor.

C2: Într-o schemă stea, constrângerile de cheie externă validează relațiile dintre tabela de fapte și tabelele dimensiune. Tabela de fapte FACT\_TRANZACTII conține atributele ID\_Client, ID\_Cont, ID\_Comerciant, ID\_Canal\_Plata, ID\_Stare si ID\_Locatie care referă atributele cu același nume din tabelele dimensiune DIM\_CLIENT, DIM\_DETALII\_PLATA, DIM\_COMERCIANT, DIM\_CANAL\_PLATA, DIM\_STARE, respectiv DIM\_LOCATIE. Pentru depozitele de date, o maniera potrivita de a defini constrângerile de cheie externa este exemplificata in documentul „ Sursa ”, punctul 5 (cod referință c2-c7).

Justificare: Atunci când se încărca un depozit de date, de obicei nu este nevoie de validarea constrângerile, deoarece datele au fost încărcate în bulk dintr-o zonă de pregătire în care a fost efectuată toată validarea datelor. Astfel, am salvat timp prețios declarând constrângerile fără a le valida. Crearea unei constrângeri în NOVALIDATE este imediată și nu depinde de dimensiunea tabelului. Constrângerea garantează că nu se vor introduce rânduri in tabela de fapte fără un rând corespunzător în dimensiune. Cu toate acestea, deoarece Oracle nu a validat rezultatul în sine, putem seta atributul RELY. Pe lângă RELY NOVALIDATE, implicit, constrângerea este definită cu opțiunea ENABLE, adică constrângerea este activă și va determina verificarea integrității datelor în cazul operațiilor LDM realizate asupra tabelelor implicate.

C3: Tabela de fapte FACT\_TRANZACTII conține atributul ID\_Data care referă atributul cu același nume din tabela dimensiune DIM\_CALENDAR. Pentru depozitele de date, o alternativă de definire a constrângerii de cheie externă între cele două tabele se găsește in documentul „ Sursa ”, punctul 5, sub codul de referința c8.

Justificare: Opțiunea RELY permite optimizatorului sa utilizeze constrângerea pentru a determina un plan optim, fără consum de resurse.

**Exercitiul 6**

Crearea indexului global:

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generatedConsulatare dictionarul datelor pentru verificarea crearii partitiilor pe index:

Folosire index :

EXPLAIN PLAN

SET STATEMENT\_ID = 's1\_index\_global' FOR

select /\*+ index(dim\_calendar calendar\_idx) \*/ \* from dim\_calendar where luna= 'June';

SELECT plan\_table\_output

FROM

table(dbms\_xplan.display('plan\_table', 's1\_index\_global','serial'));

Plan hash value: 2382551087

---------------------------------------------------------------------------------------------------------------------

| Id | Operation | Name | Rows | Bytes | Cost (%CPU)| Time | Pstart| Pstop |

---------------------------------------------------------------------------------------------------------------------

| 0 | SELECT STATEMENT | | 167 | 5344 | 8 (0)| 00:00:01 | | |

| 1 | PARTITION HASH SINGLE | | 167 | 5344 | 8 (0)| 00:00:01 | 7 | 7 |

| 2 | TABLE ACCESS BY INDEX ROWID BATCHED| DIM\_CALENDAR | 167 | 5344 | 8 (0)| 00:00:01 | | |

|\* 3 | INDEX RANGE SCAN | CALENDAR\_IDX | 167 | | 1 (0)| 00:00:01 | 7 | 7 |

---------------------------------------------------------------------------------------------------------------------

Creare index local :

Graphical user interface, text, application, email

Description automatically generated

Folosire index:

EXPLAIN PLAN

SET STATEMENT\_ID = 's1\_index\_local' FOR

select \* from dim\_locatie where tara = 'Romania';

SELECT plan\_table\_output

FROM

table(dbms\_xplan.display('plan\_table', 's1\_index\_local','serial'));

Plan hash value: 1361974882

--------------------------------------------------------------------------------------------------------------------------

| Id | Operation | Name | Rows | Bytes | Cost (%CPU)| Time | Pstart| Pstop |

--------------------------------------------------------------------------------------------------------------------------

| 0 | SELECT STATEMENT | | 4 | 124 | 8 (0)| 00:00:01 | | |

| 1 | PARTITION LIST ALL | | 4 | 124 | 8 (0)| 00:00:01 | 1 | 5 |

| 2 | TABLE ACCESS BY LOCAL INDEX ROWID BATCHED| DIM\_LOCATIE | 4 | 124 | 8 (0)| 00:00:01 | 1 | 5 |

|\* 3 | INDEX RANGE SCAN | TARA\_IDX | 4 | | 4 (0)| 00:00:01 | 1 | 5 |

--------------------------------------------------------------------------------------------------------------------------

Graphical user interface, text, application, email

Description automatically generatedBitmap index:

Folosire index:

analyze index dim\_client\_bmp compute statistics;

EXPLAIN PLAN

SET STATEMENT\_ID = 's1\_index\_bmp' FOR

select /\*+ index(dim\_client dim\_client\_bmp) \*/ \* from dim\_client where tip\_client = 'PF';

SELECT plan\_table\_output

FROM

table(dbms\_xplan.display('plan\_table', 's1\_index\_bmp','serial'));

Plan hash value: 2457481617

------------------------------------------------------------------------------------------------------

| Id | Operation | Name | Rows | Bytes | Cost (%CPU)| Time |

------------------------------------------------------------------------------------------------------

| 0 | SELECT STATEMENT | | 6 | 156 | 7 (0)| 00:00:01 |

| 1 | TABLE ACCESS BY INDEX ROWID BATCHED| DIM\_CLIENT | 6 | 156 | 7 (0)| 00:00:01 |

| 2 | BITMAP CONVERSION TO ROWIDS | | | | | |

|\* 3 | BITMAP INDEX SINGLE VALUE | DIM\_CLIENT\_BMP | | | | |

Bitmap join index:

create bitmap index bmp\_join\_idx on fact\_tranzactii (suma) from fact\_tranzactii f, dim\_comerciant d

where f.id\_comerciant = d.id\_comerciant local;

analyze index bmp\_join\_idx compute statistics;

alter session set star\_transformation\_enabled = true;

Folosire bitmap join index:

EXPLAIN PLAN

SET STATEMENT\_ID = 's1\_index\_join\_bmp' FOR

select /\*+ STAR\_TRANSFORMATION \*/

/\*+ FACT(fact\_tranzactii) \*/

f.suma, d.nume

from fact\_tranzactii f, dim\_comerciant d

where f.id\_comerciant = d.id\_comerciant

and suma = 50;

SELECT plan\_table\_output

FROM

table(dbms\_xplan.display('plan\_table', 's1\_index\_join\_bmp','serial'));

Plan hash value: 3558226070

--------------------------------------------------------------------------------------------------------------------------------

| Id | Operation | Name | Rows | Bytes | Cost (%CPU)| Time | Pstart| Pstop |

--------------------------------------------------------------------------------------------------------------------------------

| 0 | SELECT STATEMENT | | 1 | 37 | 1 (0)| 00:00:01 | | |

| 1 | NESTED LOOPS | | 1 | 37 | 1 (0)| 00:00:01 | | |

| 2 | NESTED LOOPS | | 1 | 37 | 1 (0)| 00:00:01 | | |

| 3 | PARTITION RANGE ALL | | 1 | 26 | 1 (0)| 00:00:01 | 1 |1048575|

|\* 4 | TABLE ACCESS BY LOCAL INDEX ROWID BATCHED| FACT\_TRANZACTII | 1 | 26 | 1 (0)| 00:00:01 | 1 |1048575|

| 5 | BITMAP CONVERSION TO ROWIDS | | | | | | | |

|\* 6 | BITMAP INDEX SINGLE VALUE | BMP\_JOIN\_IDX | | | | | 1 |1048575|

|\* 7 | INDEX UNIQUE SCAN | SYS\_C008471 | 1 | | 0 (0)| 00:00:01 | | |

| 8 | TABLE ACCESS BY INDEX ROWID | DIM\_COMERCIANT | 1 | 11 | 0 (0)| 00:00:01 | | |

--------------------------------------------------------------------------------------------------------------------------------

**Exercitiul 7**

Pentru a putea stabili care vor fi obiectele dimensiune pe care le vom creea, vom incepe prin a mention ace sunt acestea si la ce ne pot ajuta.

Dimensiunea este o structură care clasifică datele pentru a le permite utilizatorilor să răspundă la întrebările de afaceri. Dimensiunile utilizate în mod obișnuit sunt clienții, produsele și timpul. De exemplu, fiecare canal de vânzări al unui comerciant poate colecta și stoca date privind vânzările. Managementul lanțului de vânzare poate construi un depozit de date pentru a analiza vânzările produselor sale în toate magazinele de-a lungul timpului și pentru a răspunde la întrebări precum:

* Care este efectul promovării unui produs comparative cu unul care nu este promovat?
* Care sunt vânzările unui produs înainte și după o promoție?
* Cum afectează o promovare diferitele canale de distribuție?

În baza de date Oracle, informațiile dimensionale în sine sunt stocate într-un tabel de dimensiuni. În plus, obiectul dimensiune ajută la organizarea și gruparea informațiilor dimensionale în ierarhii. Aceasta reprezintă relații naturale 1:n între coloane sau grupuri de coloane (nivelurile unei ierarhii) care nu pot fi reprezentate cu condiții de constrângere. Urcarea unui nivel în ierarhie se numește acumularea datelor și coborârea la un nivel în ierarhie se numește detaliere a datelor.

Obiectele dimensiune nu trebuie definite neaparat. Cu toate acestea, creearea lor poate aduce beneficii semnificative, deoarece ajută la rescrierea interogărilor să efectueze tipuri mai complexe de rescrire.

In cazul acestui proiect au fost identificate 2 obiecte dimensiune: locatie, timp. Ambele dimensiuni sunt normalizate(nu este stocata in mai mult de un table, sau altfel zis, nu are referinte la alte tabele).

Aceste obiecte vor avea urmatoarele nivele si ierarhie:

* Locatie: locatie\_id -> strada -> oras -> tara
* Timp: data\_id -> data -> anul

Pentru dimensiunea locatie, clauza ATTRIBUTE ... DETERMINES se referă la a pune in legatura locatie\_id cu celelalte campuri, si anume: strada, oras, tara si site. Aceasta este o determinare unidirecțională. Este doar garantat că pentru oun anumit id, o sa fie găsita exact o valoare potrivită pentru strada, oras, tara si site. Insa nu se poate determina un id pentru un anumit oras spre exemplu.

Similar este si in cazul dimensiunii timp, un singur lucru difera aici, si anume faptul ca putem creea mai multe relatii prin atribute de aceasta data. Motivul pentru care putem face acest lucru este faptul ca atat data\_id cat si data sunt unice in tabelul dim\_calendar. Prin urmare, campurile: id\_data, data, ziua, luna, anul pot fi determinate in mod direct atat prin data\_id cat si prin data.

Nu in ultimul rand, informațiile unui obiect dimensiune sunt doar declarative și nu sunt impuse de baza de date. Dacă relațiile descrise de dimensiuni sunt incorecte, pot apărea rezultate incorecte. Prin urmare, ar trebui să fie verificațe periodic relațiile specificate de CREATE DIMENSION utilizând procedura DBMS\_DIMENSION.VALIDATE\_DIMENSION.

**Exercitiul 8**

Definire partitie pe tabela de fapte :

Graphical user interface, text, application, email

Description automatically generated

Partitionarea te tip interval este o extensie a partitionarii de tip range. Partitiile noi sunt create autmomat de catre SGBD. atunci cand coloana pe care s-a definit regula de partitionare “depaseste” toate valorile. Cel putin o partitie initiala trebuie creata. Valoarea cheii de partitie interval determină valoarea superioara a partitiilor interval, care se numește punct de tranziție, iar serverul de baze de date creează partiții de interval noi pentru datele care au o valoare mai mare decat punctul de tranziție.

Se pot defini pana la 1,048,757 partitii la nivel de tabel. Avand in vedere ca partitionarea este la nivel de zi se pot stoca intr-un tabel 2, 873 de ani.

Table

Description automatically generatedVerificarea crearii partitiilor prin consultarea dictionarului datelor:

analyze table fact\_tranzactii compute statistics;

Table FACT\_TRANZACTII analyzed.

EXPLAIN PLAN

SET STATEMENT\_ID = 'st\_fact\_partition'

FOR

select \* from fact\_tranzactii where id\_data = 20221231;

SELECT plan\_table\_output

FROM

table(dbms\_xplan.display('plan\_table','st\_fact\_partition','serial'));

Plan hash value: 3811362893

----------------------------------------------------------------------------------------------------------

| Id | Operation | Name | Rows | Bytes | Cost (%CPU)| Time | Pstart| Pstop |

----------------------------------------------------------------------------------------------------------

| 0 | SELECT STATEMENT | | 24 | 648 | 274 (0)| 00:00:01 | | |

| 1 | PARTITION RANGE SINGLE| | 24 | 648 | 274 (0)| 00:00:01 | 1132 | 1132 |

|\* 2 | TABLE ACCESS FULL | FACT\_TRANZACTII | 24 | 648 | 274 (0)| 00:00:01 | 1132 | 1132 |

----------------------------------------------------------------------------------------------------------

Predicate Information (identified by operation id):

---------------------------------------------------

2 – filter("ID\_DATA"=20221231)

EXPLAIN PLAN

SET STATEMENT\_ID = 'st\_fact\_partition\_pruning'

FOR

select \* from fact\_tranzactii where id\_data between 20230123 and 20230128;

SELECT plan\_table\_output

FROM

table(dbms\_xplan.display('plan\_table','st\_fact\_partition\_pruning','serial'));

Plan hash value: 370717546

------------------------------------------------------------------------------------------------------------

| Id | Operation | Name | Rows | Bytes | Cost (%CPU)| Time | Pstart| Pstop |

------------------------------------------------------------------------------------------------------------

| 0 | SELECT STATEMENT | | 6 | 168 | 820 (1)| 00:00:01 | | |

| 1 | PARTITION RANGE ITERATOR| | 6 | 168 | 820 (1)| 00:00:01 | 10024 | 10029 |

|\* 2 | TABLE ACCESS FULL | FACT\_TRANZACTII | 6 | 168 | 820 (1)| 00:00:01 | 10024 | 10029 |

------------------------------------------------------------------------------------------------------------

Predicate Information (identified by operation id):

---------------------------------------------------

2 - filter("ID\_DATA">=20230123 AND "ID\_DATA"<=20230128)

Definire partitie pe tabela de dimensiune locatie :

Graphical user interface, text, application, chat or text message

Description automatically generated

Verificarea crearii partitiilor prin consultarea dictionarului datelor.Graphical user interface, text, application

Description automatically generated

analyze table dim\_locatie compute statistics;

Table DIM\_LOCATIE analyzed.

EXPLAIN PLAN

SET STATEMENT\_ID = 'st\_dim\_partition'

FOR

select \* from dim\_locatie where oras = 'Timisoara'

SELECT plan\_table\_output

FROM

table(dbms\_xplan.display('plan\_table','st\_dim\_partition','serial'));

Plan hash value: 3299633729

| Id | Operation | Name | Rows | Bytes | Cost (%CPU)| Time | Pstart| Pstop |

| 0 | SELECT STATEMENT | | 1 | 36 | 274 (0)| 00:00:01 | | |

| 1 | PARTITION LIST SINGLE| | 1 | 36 | 274 (0)| 00:00:01 | KEY | KEY |

| 2 | TABLE ACCESS FULL | DIM\_LOCATIE | 1 | 36 | 274 (0)| 00:00:01 | 1 | 1 |

EXPLAIN PLAN

SET STATEMENT\_ID = 'st\_dim\_partition\_pruning'

FOR

select \* from dim\_locatie where oras in ('Timisoara', 'Constanta')

SELECT plan\_table\_output

FROM

table(dbms\_xplan.display('plan\_table','st\_dim\_partition\_pruning','serial'));

Plan hash value: 1889622376

-----------------------------------------------------------------------------------------------------

| Id | Operation | Name | Rows | Bytes | Cost (%CPU)| Time | Pstart| Pstop |

-----------------------------------------------------------------------------------------------------

| 0 | SELECT STATEMENT | | 2 | 54 | 547 (0)| 00:00:01 | | |

| 1 | PARTITION LIST INLIST| | 2 | 54 | 547 (0)| 00:00:01 |KEY(I) |KEY(I) |

| 2 | TABLE ACCESS FULL | DIM\_LOCATIE | 2 | 54 | 547 (0)| 00:00:01 |KEY(I) |KEY(I) |

-----------------------------------------------------------------------------------------------------

**Exercitiul 9**

Pentru acest exercitiu vom declara urmatoarea cerere in limbaj natural:

“Sa se afiseze pentru fiecare client suma medie a tuturor tranzactiilor efectuate la fiecare comerciant in parte.”

Planul este sa cream o vizualizare materializata cu join-uri si agregate care sa corespunda cererii noastre. Mai apoi vom colecta statisticile si vom seta parametrii pentru rescrierea cererii.

**Exercitiul 10**

1. Sa se afle care sunt primele 3 luni ale anului 2022, in functie de volumul tranzactiilor(doar debitariile).

SELECT dim\_calendar.luna, SUM (fact\_tranzactii.suma) AS suma\_totala

FROM fact\_tranzactii

JOIN dim\_calendar ON fact\_tranzactii.id\_data = dim\_calendar.id\_data

WHERE dim\_calendar.anul = 2022 AND fact\_tranzactii.suma > 0

GROUP BY dim\_calendar.luna

ORDER BY suma\_totala DESC

FETCH FIRST 3 ROWS ONLY;

2. Sa se afle primele 3 luni si sumele aferente tranzactiilor effectuate dupa numarul de tranzactii efectuate doar cu POS din anul 2022(doar debitariile).

SELECT COUNT(\*) AS nr\_plati, luna, SUM (fact\_tranzactii.suma) AS suma\_totala

FROM fact\_tranzactii

JOIN dim\_calendar ON fact\_tranzactii.id\_data = dim\_calendar.id\_data

JOIN dim\_canal\_plata ON fact\_tranzactii.id\_canal\_plata = dim\_canal\_plata.id\_canal\_plata

WHERE dim\_canal\_plata.id\_canal\_plata = 1 AND fact\_tranzactii.suma > 0 AND dim\_calendar.anul = 2022

GROUP BY dim\_calendar.luna

ORDER BY nr\_plati DESC

FETCH FIRST 3 ROWS ONLY;

3. Sa se afle tipul canalului de plata a primelor 10 tranzactii dupa suma(doar debitariile).

SELECT dim\_canal\_plata.tip\_canal\_plata, fact\_tranzactii.suma

FROM fact\_tranzactii

JOIN dim\_canal\_plata ON fact\_tranzactii.id\_canal\_plata = dim\_canal\_plata.id\_canal\_plata

WHERE fact\_tranzactii.suma > 0

ORDER BY fact\_tranzactii.suma DESC

FETCH FIRST 10 ROWS ONLY;

4. Sa se afle comerciantul cu media cea mai mare a duratei tranzactiilor(doar debitariile).

SELECT dim\_comerciant.nume, AVG(fact\_tranzactii.durata) as medie\_durata

FROM dim\_comerciant

JOIN fact\_tranzactii ON fact\_tranzactii.id\_comerciant = dim\_comerciant.id\_comerciant

WHERE fact\_tranzactii.suma > 0

GROUP BY dim\_comerciant.nume

ORDER BY medie\_durata DESC

FETCH FIRST 3 ROWS ONLY;

5. Sa se afle numele clientului care a facut cea mai mare achizitie din toti anii(doar debitariile).

SELECT dim\_client.nume, MAX(fact\_tranzactii.suma) as suma\_maxima

FROM fact\_tranzactii

JOIN dim\_client ON fact\_tranzactii.id\_client = dim\_client.id\_client

WHERE fact\_tranzactii.suma > 0

GROUP BY dim\_client.nume

ORDER BY suma\_maxima DESC

FETCH FIRST 3 ROWS ONLY;