**Structura bazei de date utilizata ca exemplu la seminar -** Se considera activitatea de evidenta a comenzilor incheiate de o societate comerciala cu diverse firme prin intermediul agentilor angajati in cadrul societatii. Comenzile contin produse aflate in depozitul societatii, iar pentru fiecare produs se cunoaste in permanenta stocul existent.

**FIRME**

| CODFIRMA | DENFIRMA | LOC | CONTBANCA | ZONA |
|---|---|---|---|---|

**AGENTI**

| CODAGENT | NUMEAGENT | DATAANG | DATANAST | ZONA | FUNCTIE | CODSEF |
|---|---|---|---|---|---|---|

**COMENZI**

| NRCOM | CODFIRMA | CODAGENT | DATA |
|---|---|---|---|

**RANDCOM**

| NRCOM | CODPRODUS | CANT | PRET | TERMENLIVR |
|---|---|---|---|---|

**PRODUSE**

| CODPRODUS | DENPRODUS | UM | STOC |
|---|---|---|---|

# RECAPITULARE COMENZI LDD

## *1. CREAREA TABELELOR – COMANDA CREATE TABLE*

> **CREATE TABLE** ***nume_tabelă***
> **(**
> ***.... definirea câmpurilor şi a tipurilor de date aferente***
> ***… definirea restricţiilor de integritate***
> **);**

Definirea restricţiilor – se poate realiza la nivel de câmp (*in-line*) sau la nivelul tabelei (*out-of-line*):
Sintaxa generală:

**Constraint** ***nume_restricţie*** **tip_restricţie** ***[ (câmpurile cărora li se aplică restricţia)]***

Tipuri de restricţii:

1. Restricţia de tip **PRIMARY KEY:**

Constraint *nume_restricţie* PRIMARY KEY *[ (câmpuri care formează cheia primară)]*

2. Restricţia de tip **FOREIGN KEY:**

Constraint *nume_restricţie* FOREIGN KEY *(câmpul cheie externă)* REFERENCES *Tabelă_părinte (câmp cheie primară)*

3. Restricţia de tip **NOT NULL:**

Se defineşte **numai** la nivelul câmpului căruia i se aplică restricţia:

Ex: *nume VARCHAR2(20) NOT NULL*

4. Restricţia de tip **UNIQUE:**

Constraint *nume_restricţie* UNIQUE *[ (câmp cheie unică)]*

5. Restricţia de tip **CHECK:**

Constraint *nume_restricţie* CHECK *[ (condiţie asupra unui câmp)]*

## *2. MODIFICAREA STRUCTURII TABELELOR - COMANDA ALTER*

Realizeaza urmatoarele:

- Modificarea structurii tabelei: ADD, MODIFY, DROP COLUMN, SET UNUSED
- Modificarea restrictiilor de integritate: ADD, MODIFY, DROP, DISABLE CONSTRAINT
- Redenumeste tabela: RENAME

> **ALTER TABLE** ***nume_tabelă***
> - **ADD** ***(definire câmpuri);***

- **MODIFY *(redefinire câmpuri existente);***
- **DROP COLUMN *câmp;***
- **ADD CONSTRAINT *nume_restricţie* TIP_RESTRICŢIE ….;**
- **DROP CONSTRAINT *nume_restricţie;***
- **DISABLE CONSTRAINT *nume_restricţie;***
- **ENABLE CONSTRAINT *nume_restricţie;***
- **RENAME TO *nume_nou_tabelă;***

## *3. STERGEREA TABELELOR – COMANDA DROP*

**DROP TABLE *nume_tabelă* CASCADE CONSTRAINTS;**

# RECAPITULARE COMENZI LMD

## *1. ADAUGAREA DATELOR – COMANDA INSERT*

```
INSERT INTO TABELA VALUES ([LISTA DE VALORI PENTRU FIECARE ATRIBUT]);
```

## *2. MODIFICAREA DATELOR – COMANDA UPDATE*

```
UPDATE [TABELA]
SET [COLOANA] = [VALOARE]
WHERE [CONDITIE];
```

## *3. STERGEREA DATELOR – COMANDA DELETE*

```
DELETE FROM [TABELA]
WHERE [CONDITIE];
```

## *4. SELECTIA DATELOR – COMANDA SELECT*

```
SELECT [DISTINCT] { * , tabelă1.câmp1 [alias] , expresii AS ALIAS ...}
FROM tabelă1, tabelă2,....
WHERE {condiţii, precizarea legăturilor dintre tabele}
GROUP BY tabelă .câmp
HAVING {condiţii impuse valorilor de grup}
ORDER BY tabelă .câmp  ASC/DESC;
```

unde:

| | |
|---|---|
| SELECT | specifică atributele selectate; |
| DISTINCT | suprimă valorile duplicate; |
| * | selectează toate atributele; |
| atribut | selectează coloana numită; |
| expresie | permite construirea de expresii si valori noi |
| alias | denumiri pentru atributele selectate; |
| FROM tabele | specifică tabelele ce conţin coloanele selectate. |
| WHERE | clauza permite specificarea conditiilor si a criteriilor de selectie a datelor |
| GROUP BY | se precizeaza campul dupa care vor fi grupate datele in cazul expresiilor si functiilor de grup (SUM(), AVG(), COUNT(), MIN(), MAX()) |
| HAVING | in cazul functiilor de grup conditiile impuse acestora se precizeaza in clauza HAVING |
| ORDER BY | precizeaza ordonarea in functie un anumite campuri ascendent (ASC) –implicit sau descendent (DESC) |

## SELECTIA DATELOR – COMANDA SELECT - continuare

# FUNCTII

- **Funcţii single-row (sau scalare).** O funcţie single-row întoarce un singur rând rezultat pentru fiecare rând al tabelei interogate sau view
- **Funcţii de grup (sau agregate).** O funcţie de grup întoarce un singur rând rezultat pentru un grup de rânduri interogate. Funcţiile de grup pot apare în clauza HAVING

## FUNCŢII DE GRUP

**AVG([DISTINCT|ALL] n)** – calculeaza media elementelor
**COUNT({* | [DISTINCT|ALL] expr})** – intoarce numarul total al elementelor
**MAX([DISTINCT|ALL] expr)** – intoarce elementul maxim
**MIN([DISTINCT|ALL] expr)** – intoarce elementul minim
**SUM([DISTINCT|ALL] n)** - calculeaza suma elementelor

Se utilizeaza urmatoarele clauze:
**GROUP BY** – grupeaza datele in functie de un anumit camp;
**ORDER BY** – ordoneaza datele in functie de un anumit camp;
**HAVING** – permite stabilirea unor criterii de selectie asupra functiilor de grup;

1. Să se afişeze valoarea maximă, valoarea medie, valoarea minimă şi valoarea totală a produselor comandate:

```
SELECT avg(rc.cant * rc.pret), max(rc.cant * rc.pret), min(rc.cant * rc.pret),
sum(rc.cant * rc.pret)
FROM rindcom rc;
```

2. Să se afişeze data primei comenzi încheiate şi data celei mai vechi comenzi încheiate:

```
SELECT min(data), max(data)
FROM comenzi;
```

3. Să se afişeze numărul de produse al căror stoc>200:

```
SELECT count(*) Nr_prod FROM produse WHERE stoc>200;
```

4. Să se afişeze numărul total de comenzi incheiate:

```
SELECT count(nrcom) Numar_Comenzi FROM comenzi;
```

5. Să se afişeze numărul de produse vândute:

```
SELECT count(distinct(codprodus)) Produse_Vandute FROM rindcom;
```

6. Să se afişeze cantitatea medie vândută din fiecare produs şi să se realizeze ordonarea în funcţie de aceasta (se utilizeaza functia AVG() si clauza GROUP BY pt gruparea datelor in functie de produse, iar ordonarea se realizeaza cu ajutorul functiei ORDER BY ).

```
SELECT codprodus, avg(cant) Medie_Prod FROM rindcom
GROUP BY codprodus
ORDER BY avg(cant);
```

7. Să se afişeze produsele şi cantitatea medie vândută numai acele produse a căror cantitate medie este mai mare de 1500 (conditia se specifica in clauza HAVING si nu in clauza WHERE deoarece este utilizata functia de grup AVG si conditia este avg(cant)>800)

```
SELECT codprodus, avg(cant) FROM rindcom
GROUP BY codprodus
HAVING avg(cant)>800;
```

8. Sa se calculeze valoarea totala a fiecarei comenzi si sa se sorteze descrescator in functie de valoare:

```
SELECT comenzi.nrcom, SUM(rindcom.cant * rindcom.pret) Total_Comanda
FROM comenzi, rindcom
WHERE rindcom.nrcom=comenzi.nrcom
GROUP BY comenzi.nrcom
ORDER BY Total_Comanda DESC;
```

9. Sa se afiseze numai comenzile care au valoarea cuprinsa intre 3 si 5 mil (conditia va fi mentionata in clauza HAVING pt ca se utilizeaza o functie de grup - SUM):

```
SELECT comenzi.nrcom, SUM(rindcom.cant * rindcom.pret) Total_Comanda
FROM comenzi, rindcom
WHERE rindcom.nrcom=comenzi.nrcom
GROUP BY comenzi.nrcom
HAVING SUM(rindcom.cant * rindcom.pret) BETWEEN 1000000 AND 3000000
ORDER BY Total_Comanda DESC;
```

## FUNCTII SINGLE-ROW

### →Funcţii de tip caracter

**Operatorul de concatenare** ( || )

10. Să se afişeze denumirea produsului şi stocul disponibil

```
SELECT 'produsul: ' || initcap(denprodus)|| ' are stocul diponibil ' || stoc
FROM produse;
```

**Functia LOWER() , UPPER()**

11. Sa se afiseze firmele din zona ‘muntenia’:

```
SELECT codfirma, upper(denfirma), upper(loc), upper(zona)
FROM firme
WHERE lower(zona)=’muntenia’;
```

**Funcţia CONCAT() , funcţia LENGTH() , funcţia SUBSTR()**

12. Să se afişeze denumirea firmei concatenată cu localitatea şi lungimea atributului denumirea firmei, numai pentru localităţile al căror nume începe cu “C”

```
SELECT denfirma, concat(denfirma,loc), length(denfirma)
FROM firme
WHERE substr(loc,1,1)='C';
```

**Functia** REPLACE()

```
SELECT REPLACE('JACK and JUE','J','BL') "Changes"
     FROM DUAL;

Changes
--------------
BLACK and BLUE
```

**→Funcţii de tip numeric**

**Funcţia ROUND()**

13. Să se afişeze numărul 45,923 rotunjit la două zecimale

```
SELECT round(45.923,2), round(45.923,0) FROM dual;
```

**→Funcţii de tip dată calendaristică**

**Funcţile SYSDATE**

14. Să se afişeze perioada de timp corespunzătoare (în săptămâni) între data încheierii comenzii şi data curentă:

```
SELECT nrcom, round((sysdate-data)/7) saptamani
FROM comenzi;
```

15. Afisati data curenta (se selecteaza data din tabela *DUAL*):

```
SELECT sysdate DATA_CURENTA FROM DUAL;
```

**Funcţiile MONTH_BETWEEN() , ADD_MONTHS() , NEXT_DAY() , LAST_DAY()**

16. Să se afişeze comenzile, data încheierii comenzilor, numărul de luni între data curentă şi data încheierii, următoarea zi de vineri după data încheierii, ultima zi din luna din care face parte data încheierii, precum şi data corespunzătoare după 2 luni de la data încheierii

```
SELECT nrcom, data,
round(MONTHS_BETWEEN(sysdate, data)) luni,
ADD_MONTHS(data,2),
NEXT_DAY(data, 'FRIDAY'), LAST_DAY(data)
FROM comenzi;
```

17. Să se afişeze comenzile incheiate in luna trecuta:

```
SELECT nrcom, data FROM comenzi
WHERE round(MONTHS_BETWEEN(sysdate, data))=1;

--daca nu exista inserati o comanda inregistrata in luna trecuta si rulati din nou query
```

**Funcţia ROUND()**

18. Să se afişeze comenzile incheiate in 2004. Se va rotunji data încheierii la prima zi din luna corespunzătoare dacă data încheierii este în prima jumatate a lunii sau la prima zi din luna următoare:

```
SELECT nrcom, data, ROUND(data, 'MONTH')
FROM comenzi
WHERE data LIKE '%04';
```

**→Funcţii de conversie**

**Funcţia TO_CHAR(d [, fmt ])**

19. Să se afişeze comenzile şi data încheierii in format “MM/YY”

```
SELECT nrcom, TO_CHAR(data, 'MM/YY') data_incheierii
FROM comenzi;
```

**Week day name from date:**

```
SELECT TO_CHAR(date '1982-03-08', 'DAY') day FROM dual;
SELECT TO_CHAR(sysdate, 'DAY') day FROM dual
```

**Funcţia TO_DATE(char [, fmt ])**

**select TO_DATE('2003/07/09', 'yyyy/mm/dd') from dual**

## Funcţia TO_NUMBER(char [,fmt ])

20. Să se afişeze comenzile şi data încheierii în formatul “ Month dd, YYYY”

```
SELECT nrcom, data
FROM comenzi
WHERE data=TO_DATE( 'January 15, 2005', 'Month dd,YYYY');
```

## Funcţia EXTRACT ()

21. Să se afişeze informaţii despre comenzile încheiate în anul 2004

```
SELECT nrcom, data
FROM comenzi
WHERE EXTRACT (YEAR from data) = 2004;
```

```
select EXTRACT(YEAR from sysdate) from dual;
2020

select EXTRACT(MONTH from sysdate) from dual;
4

select EXTRACT(DAY from sysdate) from dual;
24
```

## CREAREA UNEI TABELE PE BAZA CAMPURILOR DIN ALTA TABELA:

```
CREATE TABLE nume_tabela
AS
SELECT [*, nume campuri]
FROM nume_tabela_sursa
[WHERE conditie];
```

**Exemplu**: Tabela firme_buc va contine firmele din Bucuresti

```
CREATE TABLE FIRME_BUC
AS
SELECT * FROM FIRME
WHERE LOC='BUCURESTI';
```

## ADAUGAREA DATELOR PE BAZA VALORILOR DIN ALTE TABELE:

```
INSERT INTO nume_tabela
SELECT [*, nume campuri]
FROM nume_tabela_sursa
[WHERE conditie];
```

**Exemple:**

Sa se creeze tabela STOC_MIN cu aceeasi structura cu a tabelei PRODUSE care sa contina informatii depre produsele cu stocul mai mic decat 1000 unitati.

```
Create table stoc_min as select * from produse where 2=3;

INSERT INTO STOC_MIN
SELECT *
FROM PRODUSE
WHERE STOC<=1000;

select * from stoc_min;
```

## GESTIUNEA ALTOR OBIECTE ALE BAZEI DE DATE

## 1. TABELE VIRTUALE

**CREATE [OR REPLACE] VIEW *nume_view***
**AS**
***subcerere***

Sa se realizeze o tabela virtuala cu toate firmele din Bucuresti:

```
CREATE VIEW firme_buc_v
AS
SELECT * FROM firme
WHERE upper(loc)='BUCURESTI';

SELECT * FROM firme_buc_v;
```

Sa se realizeze o tabela virtuala care sa contina numai produsele pentru care unitatea de masura (um)=’buc’.

```
CREATE VIEW PROD_BUC_V
AS
SELECT codprodus COD_PRODUS, denprodus DENUMIRE, um UNITATE
FROM produse
WHERE um='buc';

SELECT * FROM PROD_BUC_V;
```

Sa se stearga inregistrarile din tabela virtuala PROD_BUC_V pentru produsele care au stocul mai mare de 1000.

```
DELETE FROM PROD_BUC_V
WHERE STOC>1000;
```

Sa realizeze o tabela virtuala cu toti agentii din Muntenia. Tabela virtuala nu va putea fi actualizata:

```
CREATE VIEW agenti_zona_v
AS SELECT * FROM agenti
WHERE upper(zona)='MUNTENIA'
WITH READ ONLY;

SELECT * FROM firme_buc_v;
```

Sa se stearga tabela virtuala PROD_BUC_V:

```
DROP VIEW PROD_BUC_V;
```

Vizualizarea informatiilor despre tabelele virtuale:

SELECT VIEW_NAME, TEXT FROM USER_VIEWS;

## 2. INDECSI

- Permit accesul rapid la date prin sortarea logica a inregistrarilor.
- Sunt gestionati automat de catre serverul Oracle.
- Se creaza automat la introducerea unei restrictii de cheie primara sau de unicitate sau manual de catre utilizator.

**CREATE INDEX *nume_index* ON *Nume_tabelă (câmp);***

**DROP INDEX *nume_index;***

Exemple:
Sa se creeze un index pe tabela agenti pe coloana numeagent:
CREATE INDEX AGENTI_NUMEAG_IDX ON AGENTI(NUMEAGENT);

Vizualizarea inecsilor unui anumit utilizator:
Select * from user_indexes;

Sa se strearga indexul creat anterior:
DROP INDEX AGENTI_NUMEAG_IDX;

## 3. SECVENTE

- Sunt utilizate pentru asigurarea unicitatii cheilor primare sau a valorilor pentru care s-a impus o restrictie de tip UNIQUE.
- Pot fi utilizate pentru mai multe tabele.
- Pentru fiecare secventa se va preciza valoarea de inceput, pasul de incrementare si valoarea maxima generate.

**CREATE SEQUENCE *nume_secvenţă***
**START WITH *valoare_iniţială***
**INCREMENT BY *pasul_de_incrementare***
**MAXVALUE *valoare_maximă***
**NOCYCLE;**

**ALTER SEQUENCE *nume_secvenţă …..;***

**DROP SEQUENCE *nume_secvenţă;***

Exemple:

Sa se creeze o secventa pentru asigurarea unicitatii cheii primare din tabela Comenzi.

Create sequence seq_nrcomanda
start with 500 Increment by 10
Maxvalue 1000 nocycle;

Insert into comenzi values (seq_nrcomanda.nextval, '10', '3', to_date('oct 12,05', 'mon dd,yy'));

Sa se afiseze valoarea curenta a secventei:
Select seq_rcomanda.currval from dual;

Sa se modifice pasul de incrementare pentru secventa anterioara:
Alter sequence seq_nrcomanda increment by 100;

Sa se strearga secventa seq_rcomanda:
Drop sequence seq_rcomanda;

Sa se vizualizeze informatiile depre secventele utilizatorilor:
Select * from user_sequences;

## 4. SINONIME

- Sunt nume alternative utilizate pentru referirea obiectelor unei baze de date
- Pot fi sinonime publice (accesibile tuturor utilizatorilor) sau private.
- Sinonimele publice pot fi create numai de administratorul bazei de date

**CREATE SYNONYM *nume_sinonim* FOR *nume_tabelă*;**

**DROP SYNONYM *nume_sinonim;***

Exemple:
Sa se creeze un sinonim pentru tabela rindcom:
Create synonym detalii_comanda for rindcom;

Sa se strearga sinonimul creat anterior:
Drop synonym detalii_comanda;

Vizualizarea sinonimelor se realizeaza astfel
Select * from user synonyms;

https://www.w3schools.com/sql/exercise.asp?filename=exercise_groupby2