ACADEMIA DE STUDII ECONOMICE BUCUREŞTI
FACULTATEA DE CIBERNETICĂ, STATISTICĂ ŞI INFORMATICĂ ECONOMICĂ

Admitere 2018

Programul de masterat profesional de specializare

BAZE DE DATE – SUPORT PENTRU AFACERI

Tematica de concurs CSIE3
BAZE DE DATE RELAȚIONALE
PROGRAMAREA ÎN LIMBAJUL SQL

BUCUREŞTI
Iulie 2018

Tematica de concurs CSIE3

NR. CRT	TEMATICA		REFERINȚA BIBLIOGRAFICĂ	PAGINAȚIE
1.	Baze de date relaţionale	Modelul relaţional: structura relaţională a datelor, algebra şi calculul relaţional, restricţii de integritate. Exemplificări în Oracle	[1]	pag. 103-121 pag. 129-143
		Realizarea bazelor de date relaţionale: analiza statică, dinamică şi funcţională; proiectarea structurii conceptuale, logice şi fizice; normalizarea datelor	[1]	pag. 144-186 pag. 197-203
2.	Programarea în limbajul SQL	Actualizarea structurii bazei de date: crearea obiectelor, modificarea proprietăţilor şi ştergerea acestora	[3]	pag. 101-122
		Actualizarea datelor: adăugarea de înregistrări, modificarea valorilor, ştergerea înregistrărilor	[3]	pag. 123-126
		Interogarea datelor: condiţionarea datelor, utilizarea joncţiunilor şi a funcţiilor SQL, gruparea datelor, gestiunea subcererilor	[3]	pag. 127-172
3.	Programarea în limbajul PL/SQL	Elemente de programare procedurală	[2]	pag. 9-52
		Mecanismul de cursor	[2]	pag. 53-77
		Gestiunea subprogramelor: proceduri și funcții	[2]	pag. 103-118

Bibliografie

NR. CRT.	REFERINȚA BIBLIOGRAFICĂ	
[1]	Lungu I., Bâra A., Bodea C., Botha I., Diaconiţa V., Florea A., Velicanu A.	Tratat de baze de date. Vol I. Baze de date. Organizare, proiectare şi implementare, Editura ASE, Bucureşti, 2011, ISBN 978-606-505-472-1, ISBN volum 978-606-505-481-3
[2]	Bâra A., Botha I., Diaconiţa V., Lungu I., Velicanu A.	Baze de date. Limbajul PL/SQL, Editura ASE, Bucureşti, 2009, ISBN 978-606-505-263-5
[3]	Lungu I.	Baze de date Oracle. Limbajul SQL, Editura ASE, Bucureşti, 2005, ISBN 973-594-684-X

2

Agenda

- Baze de date relaţionale
  - Modelul relaţional:
    - structura relaţională a datelor,
    - operatori relaţionali,
    - restricţii de integritate.
  - Realizarea bazelor de date relaţionale:
    - analiza statică, dinamică şi funcţională;
    - proiectarea structurii conceptuale, logice şi fizice;
    - normalizarea datelor
- Programarea în limbajul SQL
  - Actualizarea structurii bazei de date:
  - Actualizarea datelor:
  - Interogarea datelor:

ACADEMIA DE STUDII ECONOMICE BUCUREŞTI
FACULTATEA DE CIBERNETICĂ, STATISTICĂ ŞI INFORMATICĂ ECONOMICĂ

Baze de date relaționale

Modelul relațional - Concepte

- Domeniul - un ansamblu de valori, caracterizat printr-un nume;
- Relaţia/Tabela - un subansamblu al produsului cartezian al mai multor domenii, care este caracterizat printr-un nume şi conţine tupluri cu semnificaţie
- Atributul - coloana unei tabele de date, caracterizată printr-un nume. Numele acestuia exprimă, de regulă, semnificaţia valorilor din cadrul coloanei respective
- Tuplul - linia unei tabele de date şi nu are nume
- Cheia - un atribut sau un ansamblu de atribute care are rolul de a identifica o înregistrare dintr-o tabelă.
- Schema relației - numele relaţiei, urmat de lista atributelor, pentru fiecare atribut precizându-se domeniul asociat
- Bază de date relaţională (BDR) - un ansamblu de relaţii (tabele) de date împreună cu legăturile dintre ele

Operatori relaționali

- Acţionează asupra structurilor de date pentru operaţii de prelucrare: actualizare, consultare, sortare.
- În structura relaţională se implementează cu ajutorul LMD (limbajul de manipulare a datelor).

I. Operatori din algebra relaţională (AR)

- Descriu tipuri de prelucrări asupra relaţiilor, operanzii sunt relaţii, iar rezultatul este, de asemenea, o relaţie
- Operatori de bază:
  - Universali: reuniunea, diferenţa, produsul cartezian
  - Specifici: proiecţia, selecţia şi joncţiunea
- Operatori derivaţi: intersecţia şi diviziunea
- Extensii ale algebrei relaţionale standard: complementarea unei relaţii, spargerea unei relaţii şi închiderea tranzitivă.

Operatori din algebra relațională (AR)

- *Produsul cartezian (R1xR2)* - operație definită pe două relații, R_1 și R_2, pe baza cărora se construiește o nouă relație, R_3, a cărei schemă se obține prin concatenarea schemelor relațiilor R_1 și R_2 și a cărei extensie cuprinde toate combinațiile tuplurilor din R_1 cu cele din R_2

```
SELECT * FROM angajati, departamente
SELECT * FROM angajati CROSS JOIN departamente;
```

- *Reuniunea (R1∪R2)* - operație definită pe două relații, R1 și R2, ambele cu aceeași schemă, prin care se obține o nouă relație R3, cu schema identică cu R1 și R2, dar având ca extensie tuplurile din R1 și R2, luate împreună o singură dată.

```
SELECT nume FROM angajati
      UNION
SELECT nume FROM clienti;
```

- *Diferența (R1\R2)* - operație definită pe două relații, R1 și R2, ambele cu aceeași schemă, prin care se obține o nouă relație, R3, cu schema identică cu R1 și R2 cu extensia formată din acele tupluri ale relației R1 care nu se regăsesc și în relația R2

```
SELECT * FROM angajati
MINUS
SELECT * FROM angajati where id_departament=80;
```

- *Intersecția (R1∩R2)* - operație definită pe două relații, R_1 și R_2 ambele cu aceeași schemă, prin care se obține o nouă relație, *R3,* cu schema identică cu a relațiilor operand și cu extensia formată din tuplurile din R_1 și R_2

```
SELECT id_angajat FROM angajati
INTERSECT
SELECT id_angajat FROM comenzi;
```

Operatori din algebra relațională (AR)

- *Proiecția* – se definește asupra unei relații R și duce la obținerea unei noi relații P care conține un număr redus de atribute față de relația inițială și toate valorile sau combinațiile distincte de valori ale acestor atribute

```
SELECT DISTINCT nume, prenume, salariul FROM angajati ;
```

- *Selecția* - se definește asupra unei relații R și duce la obținerea unei noi relații S care va conține toate atributele relației inițiale și un număr redus de tupluri. Reducerea se face după o condiție numită condiție de selecție.

```
SELECT * FROM angajati WHERE salariul>5000;
```

- *Joncțiunea* - operație definită pe două relații, R1 și R2, care constă din construirea unei noi relații R3, prin concatenarea unor tupluri din R1 cu tupluri din R2, pe baza unei condiții specificate explicit în cadrul operației
  - Internă (*inner join*)
    - De egalitate
    - Naturală
    - De neegalitate
  - Externă (*outer join*)
    - La stânga
    - La dreapta
    - Completă

Tipuri de joncțiuni

- Joncțiunea de egalitate - definită pe două relaţii, R_1 şi R_2, prin care este construită o nouă relaţie, R_3, a cărei schemă este obţinută prin reuniunea atributelor din relaţiile iniţiale

```
SELECT *
FROM angajati a, comenzi c
WHERE a.id_ang=c.id_ang;
```

sau

```
SELECT *
FROM angajati a JOIN comenzi c
ON a.id_ang=c.id_ang;
```

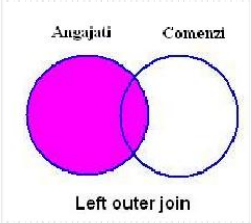
- Joncțiunea naturală - presupune existenţa unor atribute având aceeaşi denumire în ambele relaţii

```
SELECT *
FROM angajati a NATURAL JOIN comenzi c;
```

- Joncțiunea externă - operaţie prin care din două relaţii, R_1 şi R_2, se obţine o nouă relaţie, R_3, prin joncţiunea relaţiilor iniţiale. La noua relaţie R_3 sunt adăugate şi tuplurile din R_1 şi/sau R_2 care nu au participat la joncţiune. Aceste tupluri sunt completate în relaţia R_3 cu valori *NULL* pentru atributele relaţiei corespondente (R_2, respectiv R_1).

Joncțiunea la stânga

	ID_ANG	NUME
1	1	Ion
2	2	Mihai
3	3	Anca
4	4	Sofia
5	5	Mircea



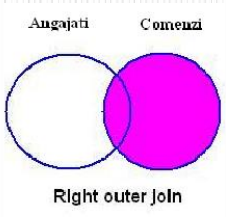
	NRC	DATAC	ID_ANG	ID_CL
1	1	01.01.2011	1	1
2	2	07.01.2011	2	2
3	3	08.01.2011	3	2
4	4	08.01.2011	2	1
5	5	09.01.2011		1

```
SELECT * FROM angajati a, comenzi c where a.id_ang=c.id_ang(+)
SELECT * FROM angajati a left join comenzi c on a.id_ang=c.id_ang
SELECT * FROM angajati a left join comenzi c using(id_ang)
```

	ID_ANG	NUME	NRC	DATAC	ID_ANG	ID_CL
1	1	Ion	1	01.01.2011	1	1
2	2	Mihai	2	07.01.2011	2	2
3	3	Anca	3	08.01.2011	3	2
4	2	Mihai	4	08.01.2011	2	1
5	5	Mircea				
6	4	Sofia				

Joncțiunea la dreapta

	ID_ANG	NUME
1	1	Ion
2	2	Mihai
3	3	Anca
4	4	Sofia
5	5	Mircea



Right outer join

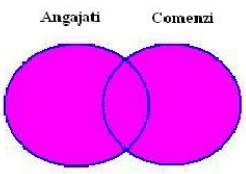
	NRC	DATAC	ID_ANG	ID_CL
1	1	01.01.2011	1	1
2	2	07.01.2011	2	2
3	3	08.01.2011	3	2
4	4	08.01.2011	2	1
5	5	09.01.2011		1

```
SELECT * FROM angajati a, comenzi c where a.id_ang(+)=c.id_ang
SELECT * FROM angajati a right join comenzi c on a.id_ang=c.id_ang
SELECT * FROM angajati a right join comenzi c using(id_ang)
```

	ID_ANG	NUME	NRC	DATAC	ID_ANG	ID_CL
1	1	Ion	1	01.01.2011	1	1
2	2	Mihai	4	08.01.2011	2	1
3	2	Mihai	2	07.01.2011	2	2
4	3	Anca	3	08.01.2011	3	2
5			5	09.01.2011		1

Joncțiunea completă

	ID_ANG	NUME
1	1	Ion
2	2	Mihai
3	3	Anca
4	4	Sofia
5	5	Mircea



Full outer join

	NRC	DATAC	ID_ANG	ID_CL
1	1	01.01.2011	1	1
2	2	07.01.2011	2	2
3	3	08.01.2011	3	2
4	4	08.01.2011	2	1
5	5	09.01.2011		1

```
SELECT * FROM angajati a, comenzi c where a.id_ang(+)=c.id_ang union
SELECT * FROM angajati a, comenzi c where a.id_ang=c.id_ang(+);
SELECT * FROM angajati a full join comenzi c on a.id_ang=c.id_ang;
SELECT * FROM angajati a full join comenzi c using(id_ang);
```

	ID_ANG	NUME	NRC	DATAC	ID_ANG	ID_CL
1	1	Ion	1	01.01.2011	1	1
2	2	Miha	2	07.01.2011	2	2
3	3	Anca	3	08.01.2011	3	2
4	2	Miha	4	08.01.2011	2	1
5	5	Mircea				
6	4	Sofia				
7			5	09.01.2011		1

II. Operatori din calculul relațional (CR)

- Pentru operatorii calcului relaţional operandul poate fi tuplu sau domeniu rezultând astfel:
  - Calcul relaţional orientat pe tuplu
  - Calcul relaţional orientat pe domeniu
- În ambele cazuri operatorii sunt:
  - Conectivele (conectorii):
    - Conjuncţia $\wedge$
    - Disjuncţia $\vee$
    - Negaţia $\neg$
  - Cuantificatorii:
    - Cuantificatorul existenţial $\exists$
    - Cuantificatorul universal $\forall$

Restricții de integritate

- Au rolul de a păstra datele corecte, consistente şi coerente în procesul de culegere, stocare, prelucrare, transmitere şi extragere a acestora.
- Se pot descrie prin LDD, dar nu numai.
- Restricţii structurale (minimale):
  - De unicitate a cheii - într-o tabelă nu trebuie să existe mai multe tupluri cu aceeaşi valoare pentru ansamblul cheie;
  - Referenţială - într-o tabelă T1 care referă o tabelă T2, valorile cheii externe trebuie să figureze printre valorile cheii primare din T2 sau să ia valoarea NULL (neprecizat);
  - Entităţii - într-o tabelă, atributele din cheia primară nu trebuie să ia valoarea NULL
- Restricţii de comportament:
  - De domeniu - domeniul corespunzător unui atribut dintr-o tabelă trebuie să se încadreze între anumite valori:
  - Temporare - valorile anumitor atribute se compară cu nişte valori temporare (rezultate din calcule etc.).

COTATII	
Simbol	
Data_cot	*>data_listarii*
Pret_Inchidere	*>=pret_min, <= pret_maxi,*
Volum	*>=0*
Pret_minim	*>=0*
Pret_maxim	*>=0*

Restricțiile de integritate suportate în SQL-Oracle

- NOT NULL
  - nu permite valori NULL (nespecificate) în coloanele unei tabele;
- UNIQUE
  - nu permite valori duplicat în coloanele unei tabele;
- PRIMARY KEY
  - nu permite valori duplicate sau NULL în coloana sau coloanele definite astfel;
- FOREIGN KEY
  - presupune ca fiecare valoare din coloana sau setul de coloane definit astfel să aibă o valoare corespondentă identică în tabela de legătură, tabelă în care coloana corespondentă este definită cu restricţia UNIQUE sau PRIMARY KEY;
- CHECK
  - elimină valorile care nu satisfac anumite cerinţe (condiţii) logice.

Etape de realizare a unei BDR

- Analiza de sistem
- Proiectarea bazei de date
- Implementarea bazei de date
- Punerea în funcţiune şi exploatarea bazei de date
- Întreţinerea bazei de date

Analiza de sistem

1. *Analiză structurală* (*statică*) - analiza componentelor sistemului (entităţilor) şi a legăturilor (asocierilor) dintre acestea. Conduce la obţinerea *modelului structural* (*static*) al sistemului;

   Cea mai utilizată tehnică de analiză structurală este tehnica entitate-asociere (E-R: *Entity-Relationship*), care permite construirea modelului structural sub forma unei diagrame entitate-asociere.

2. *Analiză temporală* (*de comportament*) - analiza stărilor sistemului şi a tranziţiilor posibile între aceste stări, în raport de anumite evenimente. Conduce la obţinerea *modelului dinamic* (*temporal*) al sistemului; Presupune:
   - identificarea stărilor în care se pot afla componentele sistemului
   - identificarea evenimentelor care determină trecerea unei componente dintr-o stare în alta
   - stabilirea succesiunii (fluxului) de evenimente

3. *Analiză funcţională* - are drept scop determinarea transformărilor de date care se produc în cadrul sistemului în scopul satisfacerii cerinţelor informaţionale aferente acestui sistem. Conduce la obţinerea *modelului funcţional* al sistemului economic;

Proiectarea structurii bazei de date

- Constă din următoarele activităţi:
  - *Proiectarea schemei conceptuale* presupune stabilirea colecţiilor de date şi definirea detaliată a conţinutului acestora; determinarea legăturilor dintre colecţiile de date şi a modului de reprezentare a acestora în cadrul schemei conceptuale; testarea schemei obţinute şi revizuirea acesteia, dacă este cazul.
  - *Proiectarea schemei externe* a bazei de date relaţionale este realizată, în principal, cu ajutorul tabelelor virtuale (*views*) şi al mecanismelor de acordare a drepturilor de acces la BDR.
  - *Proiectarea schemei interne* a bazei de date relaţionale presupune stabilirea modului de organizare fizică a datelor şi a căilor de acces la acestea (prin folosirea de indecşi şi clustere).

Tehnica normalizării
Optimizarea schemei conceptuale a BDR

- Optimizarea schemei conceptuale a unei BDR se realizează printr-o evaluare din aproape în aproape a fiecărei relaţii pe baza unor criterii de evaluare.
- Au fost identificate cinci criterii de evaluare, cărora le corespund cinci forme normale, notate FN1 ÷ FN5
- Trecerea succesivă a relaţiilor unei BDR prin formele normale cunoscute, până la aducerea lor în forma normală stabilită ca fiind optimă în contextul analizat – **tehnica normalizării**.
- Obiectivul normalizării îl constituie optimizarea structurii BDR prin:
  - Eliminarea anomaliilor de actualizare a datelor;
  - Înlăturarea redundanţei datelor.

Tehnica normalizării
Anomaliile de actualizare

- Anomalia de ştergere = stergând un tuplu dintr-o tabelă, pe lângă informaţiile şterse, se pierd şi informaţiile utile existente în tuplul respectiv;
- Anomaliile de adăugare = nu pot fi incluse noi informaţii necesare într-o tabelă deoarece nu se cunosc şi alte informaţii utile;
- Anomalia de modificare = este dificil de modificat o valoare a unui atribut atunci când ea apare în mai multe tupluri.

Tehnica normalizării

Forma normală unu (FN1)

- BDR se află în FN1 dacă toate relaţiile componente sunt în FN1.
- O relaţie este în FN1 dacă valorile asociate atributelor se află la nivel elementar (atomic) şi dacă nu există atribute generatoare de valori repetitive.

R

Marca	Numep	Adresa			$Prencopil_1$	$Datanaşterec_1$	$Prencopil_2$	$Datanaşaterec_2$	...
		Str	Loc	Cod					

Relaţia *R* are un atribut compus, denumit *Adresa* şi un grup de atribute repetitive, format din atributele *Prencopil* şi *Datanaşterec*

P

Marca	Numep	Strada	Loc	Cod

C

Prencopil	Datanasterec	Marca

Tehnica normalizării

Forma normală doi (FN2)

- BDR se află în FN2 dacă toate relaţiile componente sunt în FN2.
- O relaţie este în FN2 dacă este în FN1 şi oricare dintre atributele non-cheie este dependent funcţional complet de atributele care formează cheia primară a relaţiei.
- FN2 interzice manifestarea unor dependenţe funcţionale parţiale între atributele non-cheie şi cele care formează cheia primară a relaţiei.

R

NrComanda	CodProdus	DenProdus	UM	Cantitate

În relaţia R se manifestă următoarele dependenţe funcţionale parţiale:
CodProdus → DenProdus
CodProdus → UM

R1

NrComanda	CodProdus	Cantitate

R2

CodProdus	DenProdus	UM

Tehnica normalizării

Forma normală trei (FN3)

- BDR se află în FN3 dacă toate relaţiile componente sunt în FN3.
- O relaţie este în FN3 dacă este în FN2 şi atributele non-cheie nu sunt dependente tranzitiv de cheia primară a relaţiei.
- FN3 interzice manifestarea dependenţelor funcţionale tranzitive în cadrul relaţiei.

R

Marca	NumeAng	DataAng	Salariu	CodDep	DenDep	CodFunctie	DenFunctie	SalMin	SalMax

În cadrul relaţiei R se manifestă dependenţele funcţionale tranzitive:
CodDep → DenDep
CodFunctie → DenFunctie
CodFunctie → SalMin
CodFunctie → SalMax

R1

Marca	NumeAng	DataAng	Salariu	CodDep	CodFunctie

R2

CodDep	DenDep

R3

CodFunctie	DenFunctie	SalMin	SalMax

Tehnica normalizării

Forma normală Boyce-Codd (BCNF)

- BDR se află în BCNF dacă toate relaţiile componente sunt în BCNF.
- O relaţie este în BCNF dacă dependenţele funcţionale care se manifestă în cadrul relaţiei conţin în partea stângă (ca determinant) o cheie candidată.
- BCNF interzice manifestarea dependenţelor ale căror determinanţi nu sunt candidaţi cheie (dependenţe non-cheie).

R

IdClient	IdTranzactie	Valoare	ComisionStandard	Piata	Discount

Presupunem că în cadrul relaţiei *R* (aflată în FN3) se manifestă următoarea dependenţă:
Piata → Discount, dependenţă non-cheie (*Piata* nu este o cheie candidată)
Prin aplicarea procedurii de aducere în BCNF se obţin relaţiile:

R1

IdClient	IdTranzactie	Valoare	ComisionStandard	Piata

R2

Piata	TipClient	Discount

Tehnica normalizării

Forma normală patru (FN4)

- BDR se află în FN4 dacă toate relaţiile componente sunt în FN4.
- O relaţie este în FN4 dacă în această relaţie nu se manifestă mai mult de o dependenţă multivaloare.

R

Curs	Profesor	ReferintaBibliografica

În cadrul relaţiei *R* se manifestă următoarele dependenţe multivaloare:
Curs →→ Profesor
Curs →→ ReferintaBibliografica

R1

Curs	Profesor

R2

Curs	ReferintaBibliografica

25

Tehnica normalizării

Forma normală cinci (FN5)

- BDR se află în FN5 dacă toate relaţiile componente sunt în FN5.
- O relaţie este în FN5 dacă fiecare dependenţă joncţiune este generată printr-un candidat cheie al relaţiei.

R

Curs	Profesor	Specializare

În cadrul relaţiei *R* se manifestă următoarele dependenţe multivaloare:
Curs →→ Profesor
Curs →→ Specializare
Profesor →→ Specializare

R1

Curs	Profesor

R2

Curs	Specializare

R3

Profesor	Specializare

Exemple de teste grilă

- 1

(3p) Intr-o baza de date relationala, fie relatia R1. In urma aplicarii unei operatii pe relatia R1 se obtine o noua relatie R2, care contine doar o parte dintre tuplurile relatiei R1 si toate coloanele. Precizati ce operator al algebrei relationale realizeaza aceasta operatie:
a) produsul cartezian
b) diferenta
c) selectia
d) reuniunea
e) proiectia

- 2

(2p) In modelul de date relational pentru baze de date:

a) legaturile intre tabele se descriu in limbajul de descriere a datelor (LDD)
b) atributele care au rol in realizarea legaturilor intre tabele se numesc chei compuse
c) nu exista notiunea de restrictia entitatii
d) nu exista notiunea de schema relatiei
e) selectia si existenta sunt operatori din algebra relationala

Exemple de teste grilă

- 3

(3p) Normalizarea relatiilor din cadrul bazelor de date relationale ofera posibilitatea:

a) cresterii redundantei
b) eliminarii anomaliilor de adaugare de noi coloane
c) eliminarii anomaliilor de stergere
d) eliminarii protectiei datelor
e) regasirii tuplurilor dupa mai multe chei secundare

- 4

(3p) Coerenta datelor este asigurata intr-o baza de date, printre altele, prin:

a) operatorii relationali
b) structurarea datelor pe trei niveluri
c) reducerea redundantei datelor
d) jonctiunea datelor
e) restrictiile de integritate

Exemple de teste grilă

- 5

(3p) In modelul de date relational pentru baze de date:

a) exista notiunea de tip arbore
b) exista notiunea de tabela relatiei
c) legaturile intre tabele se descriu in limbajul de manipulare a datelor (LMD)
d) proiectia, negatia si existenta sunt operatori din algebra relationala
e) restrictia referentiala contribuie la asigurarea integritatii structurale

- 6

(3p) O tabela este in FN3 daca:
a) este in FN2 si are cel putin o dependenta functionala completa intre atributele non-cheie si cheia primara a tabelei
b) este in FN1 si fiecare atribut cheie primara depinde tranzitiv de atributele non-cheie
c) este in FN2 si atributele non-cheie nu sunt dependente tranzitiv de cheia primara a relatiei
d) este in FN2 si are dependente complete
e) este in FN1 si are dependente functionale incomplete

ACADEMIA DE STUDII ECONOMICE BUCUREŞTI
FACULTATEA DE CIBERNETICĂ, STATISTICĂ ŞI INFORMATICĂ ECONOMICĂ

Programarea în limbajul SQL

Limbajul SQL

- SQL (*Structured Query Language*) este un limbaj de descriere şi manipulare acceptat de toate sistemele de gestiune a bazelor de date relaţionale.
- Atât ANSI (*American National Standards Institute*), cât şi ISO (*International Standards Organization*) îl consideră drept un standard pentru limbajele de interogare a bazelor de date relaţionale.

Limbajul SQL-Oracle
Categorii de comenzi SQL

Categorie comenzi SQL	Comenzi SQL
Limbajul de definire a datelor (LDD)	CREATE
	ALTER
	DROP
Limbajul de manipulare a datelor (LMD)	SELECT
	INSERT
	UPDATE
	DELETE
Limbajul de gestiune a tranzacţiilor	COMMIT
	ROLLBACK
	SAVEPOINT
Limbajul de control al datelor	GRANT
	REVOKE

Limbajul SQL-Oracle
Actualizarea structurii bazei de date

Comandă SQL	Descriere
CREATE	creează structura unui obiect al BD
ALTER	modifică structura unui obiect existent al BD
DROP	şterge un obiect al BD

33

Limbajul SQL-Oracle.
Crearea tabelelor

- Tabela reprezintă o structură de date care conţine datele unei BDR.
- În general, crearea unei tabele constă din:
  - definirea coloanelor
  - definirea restricţiilor de integritate
  - specificarea parametrilor de stocare
  - definirea *cluster*-ului în care este inclusă tabela etc.

34

Limbajul SQL-Oracle.
Crearea tabelelor

```
CREATE TABLE [nume_schema.] nume_tabelă
(
nume_coloana_1 tip_date [DEFAULT expresie],
…
nume_coloana_2 tip_date [DEFAULT expresie]
);
```


Limbajul SQL-Oracle.
Crearea tabelelor

- O tabela poate fi creată în următoarele moduri:
  1) fără indicarea restricţiilor de integritate
  2) cu indicarea restricţiilor la nivel de coloană (*inline*)
  3) cu indicarea restricţiilor la nivel de tabelă (*out-of-line*)
  4) prin copiere din altă tabelă

Notă: variantele 2 şi 3 pot fi utilizate împreună pentru crearea unei tabele. Restricţiile pot fi adăugate si ulterior creări tabelei.

Limbajul SQL-Oracle. Crearea tabelelor

- Restricţiile de integritate în Oracle:
  - PRIMARY KEY
  - FOREIGN KEY
  - UNIQUE
  - NOT NULL
  - CHECK

37

Limbajul SQL-Oracle. Crearea tabelelor

1) fără indicarea restricţiilor de integritate

```
CREATE TABLE angajati
(
marca NUMBER(4),
nume VARCHAR2(20),
prenume VARCHAR2(20),
email VARCHAR2(20),
data_angajare DATE DEFAULT SYSDATE,
salariu NUMBER(8,2),
id_departament NUMBER(3)
);
```

38

Limbajul SQL-Oracle. Crearea tabelelor

- cu indicarea restricţiilor de integritate la nivel de coloană

```
CREATE TABLE angajati
(
marca NUMBER(4) CONSTRAINT pkAng PRIMARY KEY,
nume VARCHAR2(20) NOT NULL,
prenume VARCHAR2(20) NOT NULL,
email VARCHAR2(20) CONSTRAINT uqMail UNIQUE
      CONSTRAINT ckMail CHECK (email LIKE '%@%.%'),
data_angajare DATE DEFAULT SYSDATE,
salariu NUMBER(8,2),
id_departament    NUMBER(3)    CONSTRAINT    fkDep    REFERENCES
  departamente (id_departament) ON DELETE CASCADE
);
```

39

Limbajul SQL-Oracle. Crearea tabelelor

- cu indicarea restricţiilor de integritate la nivel de tabelă

```
CREATE TABLE angajati (
marca NUMBER(4),
nume VARCHAR2(20),
prenume VARCHAR2(20),
email VARCHAR2(20),
data_angajare DATE DEFAULT SYSDATE,
salariu NUMBER(8,2),
id_departament NUMBER(3),
CONSTRAINT pkAng PRIMARY KEY (marca),
CONSTRAINT uqMail UNIQUE (email),
CONSTRAINT ckMail CHECK (email LIKE '%@%.%'),
CONSTRAINT nnNume CHECK (nume IS NOT NULL),
CONSTRAINT nnPrenume CHECK (prenume IS NOT NULL),
CONSTRAINT    fkDep    FOREIGN    KEY(id_departament)    REFERENCES
  departamente (id_departament) ON DELETE CASCADE );
```

40

Limbajul SQL-Oracle.
Crearea tabelelor

4) prin copiere din altă tabelă

```
CREATE TABLE angajati_noi
AS
SELECT * FROM angajati
WHERE data_angajare>TO_DATE ('01.01.2010', 'DD.MM.YYYY');
```

Notă: restricţiile de integritate existente în tabela *angajaţi* nu se păstrează şi în noua tabelă (cu exceptia lui NOT NULL definit *inline*)

41

Limbajul SQL-Oracle.
Modificarea structurii tabelelor

- Modificarea structurii unei tabele constă din:
  - adăugarea unor coloane noi într-o tabelă existentă (eventual cu indicarea de restricţii sau de valori implicite)
  - modificarea coloanelor unei tabele
  - specificarea unor restricţii pentru coloane existente
  - activarea, dezactivarea sau suprimarea unor restricţii de integritate
  - redenumiri ale coloanelor sau redenumirea tabelei

42

Limbajul SQL-Oracle.
Modificarea structurii tabelelor

```
ALTER TABLE nume_tabela
ADD (nume_coloana_1 tip_data restrictie,
... nume_coloana_2 tip_data restrictie);

ALTER TABLE nume_tabela
MODIFY (nume_coloana_1 tip_data restrictie,
... nume_coloana_2 tip_data restrictie);

ALTER TABLE nume_tabela
RENAME COLUMN nume_coloana TO nume_nou_coloana;

ALTER TABLE nume_tabela
DROP COLUMN nume_coloana;

ALTER TABLE nume_tabela
DROP (nume_coloana_1, nume_coloana_2);
```

43

Limbajul SQL-Oracle.
Modificarea structurii tabelelor

```
ALTER TABLE nume_tabela
ADD CONSTRAINT nume_restrictie tip_restrictie ….;

ALTER TABLE nume_tabela
DROP CONSTRAINT nume_restrictie;

ALTER TABLE nume_tabela
DISABLE CONSTRAINT nume_restricţie;

ALTER TABLE nume_tabela
ENABLE CONSTRAINT nume_restricţie;

ALTER TABLE nume_tabela
RENAME TO nume_nou_tabelă;
```

Limbajul SQL-Oracle.
Ştergerea tabelelor

- Ştergerea unei tabele presupune:
  - ştergerea definiţiei sale din dicţionarul BD
  - ştergerea indecşilor asociaţi tabelei
  - ştergerea privilegiilor conferite în legătură cu tabela
  - eliberarea spaţiului de memorie ocupat
  - invalidarea funcţiilor, procedurilor, tabelelor virtuale, sinonimelor referitoare la tabelă

```
DROP TABLE nume_tabela;
```

- Pentru a se permite ştergerea unei tabele referite într-o altă tabelă se utilizează comanda DROP cu opţiunea CASCADE CONSTRAINTS, în scopul suprimării restricţiilor de referenţialitate:

```
DROP TABLE nume_tabela CASCADE CONSTRAINTS;
```

Limbajul SQL-Oracle
Actualizarea datelor

Comandă	Descriere
INSERT	adaugă o înregistrare nouă într-o tabelă
UPDATE	modifică valori asociate coloanelor unei tabele
MERGE	realizează fie modificări ale datelor, fie adăugări de înregistrări dintr-o altă tabelă, în funcţie de o condiţie de potrivire
DELETE	şterge înregistrări dintr-o tabelă

Limbajul SQL-Oracle
Interogarea datelor

Comandă SQL	Descriere
SELECT	regăseşte date din una sau mai multe tabele

Clauze în comanda SELECT	Descriere
WHERE *condiţie*	restricţionează liniile care se returnează pe baza unui criteriu specificat în condiţia de selecţie
clauze de ordonare ierarhică	structurează rezultatul într-o manieră ierarhică (asemănător cu o organigramă)
GROUP BY *coloane_grupare*	grupează liniile în scopul identificării valorilor comune grupurilor (valori agregate calculate prin funcţii de grup)
HAVING *condiţie*	restricţionează grupurile create prin clauza GROUP BY pe baza unei condiţii asupra funcţiilor de grup
ORDER BY *coloane_ordonare*	sortează liniile (implicit ascendent)

Limbajul SQL-Oracle
Precedența operatorilor

Operator
* , / , - , +
= , >= , > , <= , < , <> , != , IS , LIKE , IN
BETWEEN
NOT
AND
|| , OR

Limbajul SQL-Oracle
Funcții SQL

Funcţii SQL care manipulează şiruri de caractere

Sintaxă	Rezultat
UPPER (s) / LOWER (s)	şir de caractere
INITCAP (s)	şir de caractere
CONCAT (s1,s2)	şir de caractere
LPAD (s1,n,s2) / RPAD (s1,n,s2)	şir de caractere
LTRIM (s1,s2) / RTRIM (s1,s2)	şir de caractere
TRIM (info s1 FROM s2)	şir de caractere
LENGTH (s)	număr
INSTR (s1,s2, poz,n)	număr
SUBSTR (s,poz,n)	şir de caractere
REPLACE (s1,s2,s3)	şir de caractere

Limbajul SQL-Oracle
Funcții SQL

Funcţii SQL care manipulează valori numerice

Sintaxă	Rezultat
ROUND (n,i)	număr
TRUNC (n,i)	număr
MOD (n1,n2)	număr

Limbajul SQL-Oracle
Funcții SQL

Funcţii SQL care manipulează date calendaristice

Sintaxă	Rezultat
SYSDATE	dată calendaristică
ROUND (d,i)	dată calendaristică
TRUNC (d,i)	dată calendaristică
NEXT_DAY (d,s)	dată calendaristică
LAST_DAY (d)	dată calendaristică
ADD_MONTHS (d,n)	dată calendaristică
MONTHS_BETWEEN (d1,d2)	număr
EXTRACT (DAY FROM d) EXTRACT (MONTH FROM d) EXTRACT (YEAR FROM d)	număr

Limbajul SQL-Oracle
Funcții SQL

Funcţii SQL speciale

Sintaxă
NVL (*e1,e2*)
NULLIF (*e1,e2*)
DECODE (*e,expresii_de_căutare,d*)
CASE *expr* WHEN *cond* THEN *rez* ... ELSE *rez* END

Limbajul SQL-Oracle
Funcții SQL

Funcţii SQL de conversie între tipuri de date

Sintaxă
TO_NUMBER(*s,format*)
TO_CHAR (*n,format*)
TO_CHAR (*d,format*)
TO_DATE (*s,format*)
TO_TIMESTAMP (*s,format*)

53

Limbajul SQL-Oracle
Funcții SQL

Funcţii SQL de grup

Sintaxă
COUNT(*e*)
SUM(*e*)
MIN(*e*)
MAX(*e*)
AVG(*e*)

54

Limbajul SQL-Oracle
Agregarea datelor

Clauze în SELECT	Descriere
GROUP BY *coloane_grupare*	grupează liniile în scopul identificării valorilor comune grupurilor (valori agregate calculate prin funcţii de grup)
HAVING *condiţie*	restricţionează grupurile create prin clauza GROUP BY pe baza unei condiţii asupra funcţiilor de grup

55

Limbajul SQL-Oracle
Joncțiuni

Oracle	Standard SQL
Joncţiune de egalitate	
SELECT t1.a, t2.c FROM tabela1 t1, tabela2 t2 WHERE t1.b=t2.b;	SELECT t1.a, t2.c FROM tabela1 t1 JOIN tabela2 t2 ON t1.b=t2.b;
	SELECT t1.a, t2.c FROM tabela1 t1 NATURAL JOIN tabela2 t2
	SELECT t1.a, t2.c FROM tabela1 t1 JOIN tabela2 t2 USING (b);

56

Limbajul SQL-Oracle
Joncțiuni

Oracle	Standard SQL
Joncţiune externă stânga	
SELECT t1.a, t2.c FROM tabela1 t1, tabela2 t2 WHERE t1.b=t2.b (+);	SELECT t1.a, t2.c FROM tabela1 t1 LEFT JOIN tabela2 t2 ON t1.b=t2.b;
Joncţiune externă dreapta	
SELECT t1.a, t2.c FROM tabela1 t1, tabela2 t2 WHERE t1.b (+)=t2.b;	SELECT t1.a, t2.c FROM tabela1 t1 RIGHT JOIN tabela2 t2 ON t1.b=t2.b;

57

Limbajul SQL-Oracle
Joncțiuni

Oracle	Standard SQL
Joncţiune externă completă	
SELECT t1.a, t2.c FROM tabela1 t1, tabela2 t2 WHERE t1.b=t2.b (+) *UNION* SELECT t1.a, t2.c FROM tabela1 t1, tabela2 t2 WHERE t1.b (+)=t2.b;	SELECT t1.a, t2.c FROM tabela1 t1 FULL JOIN tabela2 t2 ON t1.b=t2.b;
Joncţiunea tabelei cu ea însăşi	
SELECT t1.a, t2.b FROM tabela1 t1, tabela1 t2 WHERE t1.a=t2.b;	SELECT t1.a, t2.b FROM tabela1 t1 JOIN tabela1 t2 ON t1.a=t2.b;

58

Limbajul SQL-Oracle
Interogarea datelor. Subcereri.

Subcerere (cerere imbricată) – comandă SELECT inclusă într-o altă comandă SQL, care poate returna una sau mai multe linii.

Tipuri de subcereri:

Subcereri	Descriere
subcereri *single-row*	returnează o singură linie (conţinând valori pentru una sau mai multe coloane)
subcereri *multiple-row*	returnează 0, 1 sau mai multe linii
subcereri *multiple-column*	returnează mai mult de o coloană ca rezultat al unei subcereri *single-row* sau *multiple-row*
subcereri corelate	subcereri de oricare dintre tipurile anterioare, care asigură legătura dintre tabele, prin referirea de coloane ale cererii părinte
subcereri scalare	returnează o singură coloană rezultat; pot fi utilizate ca orice expresie care apare într-o comandă SQL

Limbajul SQL-Oracle
Interogarea datelor. Subcereri.

```
SELECT id_angajat, nume
FROM angajati
WHERE id_functie = (SELECT id_functie FROM angajati
WHERE UPPER(nume)='KING')
AND UPPER(nume)!='KING';
```

```
ORA-01427: single-row subquery returns more than one row
```

```
SELECT id_angajat, nume
FROM angajati
WHERE id_functie IN (SELECT id_functie FROM angajati
WHERE UPPER(nume)='KING')
AND UPPER(nume)!='KING';
```

Limbajul SQL-Oracle
Interogarea datelor. Subcereri.

Operatori de comparaţie utilizaţi în cadrul subcererilor *multiple-row*:

Operatori	Descriere
IN	• compară cu o listă de valori • verifică dacă valoarea căutată se regăseşte în listă
NOT	• utilizat cu operatorul IN
ANY	• utilizat în combinaţie cu operatorii de comparaţie (=, <,>) • verifică dacă valoarea căutată îndeplineşte condiţia de comparaţie cu oricare dintre liniile returnate de subcerere
SOME	• la fel ca operatorul ANY
ALL	• utilizat în combinaţie cu operatorii de comparaţie (=, <,>) • verifică dacă valoarea căutată îndeplineşte condiţia de comparaţie cu toate liniile returnate de subcerere

Limbajul SQL-Oracle
Interogarea datelor. Subcereri.

- ANY compară valoarea cu oricare valoare returnată de interogare
- ALL compară valoarea cu toate valorile returnate de interogare

Semnificaţia operatorilor în combinaţie cu operatorii de comparaţie:

- <ANY() – mai mic decât maximul
- >ANY() – mai mare decât minimul
- =ANY() – echivalent cu operatorul IN
- >ALL() – mai mare decât maximul
- <ALL() – mai mic decât minimul

Limbajul SQL-Oracle
Interogarea datelor. Subcereri.

```
SELECT nume, prenume
FROM angajati
WHERE (nume, prenume)=(SELECT nume, prenume FROM
  angajati WHERE id_angajat=100);


SELECT id_angajat, nume
FROM angajati a
WHERE salariul > (
    SELECT AVG(salariul)
    FROM angajati
    WHERE id_departament = a.id_departament);
```

63

Exemple grile

- 1.

Fie tabela:
ANGAJATI(id_angajat number(5) primary key, nume varchar2(32), prenume varchar2(32), salariul number(10,2), id_departament number(3)).
Pentru a determina de numarul de angajati din fiecare departament, ordonat decrescator in functie de acest numar, se foloseste urmatoarea comanda SQL in Oracle:

a) SELECT id_departament, COUNT (*) FROM angajati GROUP BY id_departament HAVING COUNT (*) desc;
b) SELECT id_departament, COUNT (*) FROM angajati HAVING SUM(*)>1;
c) SELECT id_departament, COUNT(*) FROM angajati GROUP BY id_departament ORDER BY COUNT(*) desc;
d) SELECT id_departament, COUNT (*) FROM angajati ORDER BY COUNT(*) desc;

- 2.

In SQL din Oracle, pentru a finaliza tranzactia si a face modificarile permanente se foloseste comanda:

a) SET TRANSACTION OFF
b) SUBSTR
c) COMMIT
d) ROLLBACK

Exemple grile

- 3

Se considera tabela: PRODUSE(cod_produs number(3), den_produs varchar2(25), categorie varchar2(13)).
Precizati care este efectul comenzii SQL in Oracle:
ALTER TABLE produse ADD CONSTRAINT ck_categorie CHECK (categorie in ('hardware', 'software'));

a) adaugarea unei restrictii de domeniu pentru coloana CATEGORIE
b) adaugarea unor noi inregistrari la sfarsitul tabelei pentru produsele din categoriile hardware si software
c) adaugarea unei noi coloane in tabela
d) adaugarea unei noi inregistrari dupa inregistrarea curenta

Exemple grile

- 4

(3p) Alegeti varianta corecta referitoare la urmatoarea comanda SQL:
UPDATE produse SET pret=100 WHERE extract(year from data_introducerii) = extract(year from sysdate);
a) se actualizeaza, in sens de modificare, preturile produselor cu data introducerii in anul curent
b) se modifica preturile produselor a caror data a introducerii este necunoscuta
c) se actualizeaza, in sens de stergere, preturile produselor care au fost introduse astazi
d) se modifica preturile produselor ce au fost introduse anul trecut
e) se modifica preturile produselor care au codul 100

- 5

(2p) In SQL, comanda DELETE are ca efect:
a) actualizarea structurii unei tabele
b) stergerea unor campuri din structura unei tabele
c) efectuarea unei operatii de actualizare a datelor dintr-o tabela
d) stergerea unei tabele dintr-o baza de date
e) stergerea logica a unei tabele

Exemple grile

- 6

(3p) SQL este:
a) un limbaj de programare structurata
b) un limbaj de interogare inclus in orice SGBD
c) un limbaj procedural relational
d) un limbaj pentru interogari structurate
e) un limbaj specific pentru programarea web

- 7

(3p) Se considera tabela: PRODUSE(codp number(3), denp varchar2(25), ump varchar2(3))
Precizati care este efectul comenzii SQL in Oracle:
ALTER TABLE produse ADD (stoc number(9));
a) adaugarea unei noi coloane in tabela
b) adaugarea unei noi relatii la sfarsitul tabelei
c) adaugarea tipului si lungimii pentru coloana stoc existenta deja
d) adaugarea unei noi linii dupa inregistrarea curenta
e) adaugarea unui nou tuplu la sfarsitul tabelei

Exemple grile

- 8

(3p) Fie tabela angajati (id_angajat number(6), nume varchar2(32), salariul number(8,2), id_departament number(6)) avand peste 100 de randuri.
Ce afiseaza urmatoarea comanda SQL-Oracle:
SELECT sum(decode(id_departament,50,salariul,0)),sum(decode(id_departament,80,salariul,0))
FROM angajati;

a) suma salariilor angajatilor din departamentul 50 respectiv 80
b) comanda este eronata deoarece nu se foloseste corect functia decode
c) comanda este eronata deoarece nu se folosesc aliasuri pentru cele doua coloane
d) doua coloane cu valoarea NULL
e) comanda este eronata deoarece nu contine group by

- 9

(2p) Fie tabela ANGAJATI(id_angajat number(5) primary key, nume varchar2(32), prenume varchar2(32), salariul number(10,2)) avand 107 randuri (inregistrari) si interogarea SQL Oracle:
SELECT count(id_angajat+1) FROM angajati;
Care din urmatoarele afirmatii este adevarata?

a) interogarea va afisa valoarea 214
b) interogarea va afisa valoarea 108
c) interogarea va afisa valoarea 107
d) interogarea va afisa valoarea NULL
e) interogarea contine o eroare de sintaxa si nu va rula

Exemple grile

- 10

Se considera tabela: CLIENTI(cod_client number(3), den_client varchar2(25), cnp varchar2(13)).
Ce se intampla cand se foloseste urmatoarea comanda SQL in Oracle?
DELETE * FROM clienti WHERE cod_client IN (102, 103, 104);

a) se sterg clientii cu codurile 102, 103, 104
b) se sterg toate coloanele din tabela CLIENTI
c) nimic, sintaxa e gresita
d) se sterge tabela si indecsii corespunzatori

- 11

Fie tabelele RAND_COMENZI (nr_comanda NUMBER(6), id_produs NUMBER(8), cantitate NUMBER(7), pret NUMBER(7,2)) si PRODUSE (id_produs NUMBER(8), denumire VARCHAR2(32)) precum si interogarea SQL Oracle:
SELECT p.id_produs,denumire_produs, sum(pret*cantitate) val from PRODUSE p join RAND_COMENZI r on p.id_produs=r.id_produs GROUP BY p.id_produs, denumire_produs HAVING sum(pret*cantitate)>5000;
Care din urmatoarele afirmatii este adevarata?

a) se realizeaza o jonctiune externa
b) interogarea va afisa produsele pentru care valoarea totala comandata este mai mare de 5000
c) interogarea contine o eroare si nu va rula
d) interogarea va returna mereu un singur rand

Succes!