SQL – limbajul de manipulare a datelor

Inserarea de inregistrari folosind INSERT

```
INSERT [LOW_PRIORITY | DELAYED | HIGH_PRIORITY] [IGNORE]
    [INTO] tbl_name [(col_name,...)]
    {VALUES | VALUE} ({expr | DEFAULT},...),(...),...
    [ ON DUPLICATE KEY UPDATE
      col_name=expr
        [, col_name=expr] ... ]
```

```
INSERT [LOW_PRIORITY | DELAYED | HIGH_PRIORITY] [IGNORE]
    [INTO] tbl_name
    SET col_name={expr | DEFAULT}, ...
    [ ON DUPLICATE KEY UPDATE
      col_name=expr
        [, col_name=expr] ... ]
```

```
INSERT [LOW_PRIORITY | HIGH_PRIORITY] [IGNORE]
    [INTO] tbl_name [(col_name,...)]
    SELECT ...
    [ ON DUPLICATE KEY UPDATE
      col_name=expr
        [, col_name=expr] ... ]
```

SQL – limbajul de manipulare a datelor

Stergerea de inregistrari folosind DELETE

```
DELETE [LOW_PRIORITY] [QUICK] [IGNORE] FROM tbl_name
    [WHERE where_condition]
    [ORDER BY ...]
    [LIMIT row_count]
```

```
DELETE [LOW_PRIORITY] [QUICK] [IGNORE]
    tbl_name[.*] [, tbl_name[.*]] ...
    FROM table_references
    [WHERE where_condition]
```

```
DELETE [LOW_PRIORITY] [QUICK] [IGNORE]
    FROM tbl_name[.*] [, tbl_name[.*]] ...
    USING table_references
    [WHERE where_condition]
```

SQL – limbajul de manipulare a datelor

Modificarea unor inregistrari folosind UPDATE

```
UPDATE [LOW_PRIORITY] [IGNORE] table_reference
    SET col_name1={expr1|DEFAULT} [, col_name2={expr2|DEFAULT}] ...
    [WHERE where_condition]
    [ORDER BY ...]
    [LIMIT row_count]
```

```
UPDATE [LOW_PRIORITY] [IGNORE] table_references
    SET col_name1={expr1|DEFAULT} [, col_name2={expr2|DEFAULT}] ...
    [WHERE where_condition]
```

SQL – limbajul de manipulare a datelor

Import de inregistrari folosind LOAD DATA INFILE

```
LOAD DATA [LOW_PRIORITY | CONCURRENT] [LOCAL] INFILE 'file_name'
    [REPLACE | IGNORE]
    INTO TABLE tbl_name
    [CHARACTER SET charset_name]
    [{FIELDS | COLUMNS}
        [TERMINATED BY 'string']
        [[OPTIONALLY] ENCLOSED BY 'char']
        [ESCAPED BY 'char']
    ]
    [LINES
        [STARTING BY 'string']
        [TERMINATED BY 'string']
    ]
    [IGNORE number LINES]
    [(col_name_or_user_var,...)]
    [SET col_name = expr,...]
```

SQL – Limbajul de interogare a datelor

Interogarea datelor folosind SELECT

```
SELECT
    [ALL | DISTINCT | DISTINCTROW ]
      [HIGH_PRIORITY]
      [STRAIGHT_JOIN]
      [SQL_SMALL_RESULT] [SQL_BIG_RESULT] [SQL_BUFFER_RESULT]
      [SQL_CACHE | SQL_NO_CACHE] [SQL_CALC_FOUND_ROWS]
    select_expr [, select_expr ...]
    [FROM table_references
    [WHERE where_condition]
    [GROUP BY {col_name | expr | position}
      [ASC | DESC], ... [WITH ROLLUP]]
    [HAVING where_condition]
    [ORDER BY {col_name | expr | position}
      [ASC | DESC], ...]
    [LIMIT {[offset,] row_count | row_count OFFSET offset}]
    [PROCEDURE procedure_name(argument_list)]
    [INTO OUTFILE 'file_name' export_options
      | INTO DUMPFILE 'file_name'
      | INTO var_name [, var_name]]
    [FOR UPDATE | LOCK IN SHARE MODE]]
```

SQL - Limbajul de interogare a datelor

Sintaxa pentru clauza JOIN

```
table_references:
    table_reference [, table_reference] ...

table_reference:
    table_factor
  | join_table

table_factor:
    tbl_name [[AS] alias] [index_hint]
  | table_subquery [AS] alias
  | ( table_references )
  | { OJ table_reference LEFT OUTER JOIN table_reference
        ON conditional_expr }

join_table:
    table_reference [INNER | CROSS] JOIN table_factor [join_condition]
  | table_reference STRAIGHT_JOIN table_factor
  | table_reference STRAIGHT_JOIN table_factor ON conditional_expr
  | table_reference {LEFT|RIGHT} [OUTER] JOIN table_reference join_condition
  | table_reference NATURAL [{LEFT|RIGHT} [OUTER]] JOIN table_factor

join_condition:
    ON conditional_expr
  | USING (column_list)

index_hint:
    USE {INDEX|KEY} [FOR JOIN] (index_list)
  | IGNORE {INDEX|KEY} [FOR JOIN] (index_list)
  | FORCE {INDEX|KEY} [FOR JOIN] (index_list)

index_list:
    index_name [, index_name] ...
```

SQL – Limbajul de interogare a datelor

Subintoregare

```
SELECT * FROM t1 WHERE column1 = (SELECT column1 FROM t2);
```

```
DELETE FROM t1
WHERE s11 > ANY
 (SELECT COUNT(*) /* no hint */ FROM t2
  WHERE NOT EXISTS
   (SELECT * FROM t3
    WHERE ROW(5*t2.s1,77)=
     (SELECT 50,11*s1 FROM t4 UNION SELECT 50,77 FROM
      (SELECT * FROM t5) AS t5)));
```

Subintoregare rescrisa ca join

```
SELECT * FROM t1 WHERE id IN (SELECT id FROM t2);
```

```
SELECT DISTINCT t1.* FROM t1, t2 WHERE t1.id=t2.id;
```

```
SELECT * FROM t1 WHERE id NOT IN (SELECT id FROM t2);
SELECT * FROM t1 WHERE NOT EXISTS (SELECT id FROM t2 WHERE t1.id=t2.id);
```

```
SELECT table1.*
  FROM table1 LEFT JOIN table2 ON table1.id=table2.id
  WHERE table2.id IS NULL;
```

SQL – Aplicatie

Creare tabele

```
CREATE TABLE IF NOT EXISTS departament
(id int unique auto_increment primary key,
nume char(20),
manager_id int);

CREATE TABLE IF NOT EXISTS masina
id int unique auto_increment primary key,
numar char(10)
marca_id int
FOREIGN KEY (marca_id) REFERENCES marca_masina(id));

CREATE TABLE IF NOT EXISTS marca_masina
id int unique auto_increment primary key,
nume char(10));

CREATE TABLE IF NOT EXISTS angajat
(id int unique auto_increment primary key,
nume char(20),
prenume char(20),
departament_id int,
manager_id int ,
salariu int,
angajare date,
vechime date,
masina_id int,
INDEX (departament_id),
FOREIGN KEY (departament_id) REFERENCES departament(id),
FOREIGN KEY(manager_id) REFERENCES angajat(id),
FOREIGN KEY(masina_id) REFERENCES masina(id));
```

SQL – Aplicatie

Inserare date in tabela – folosind 2 sintaxe diferite pentru INSERT si importul dintr-un fisier extern (local)

```
#inserare nume departamente (folosind sintaxa cu 'VALUES')
INSERT INTO departament (nume, manager_id) VALUES
('R&D', 1), ('QA', 2), ('IT', 3), ('Backend', 4), ('HR', 5);

#inserare nume departamente (folosind sintaxa cu 'set <nume_camp>')
INSERT INTO departament set nume='Finance', set manager_id = 14;
INSERT INTO departament set nume='BI', set manager_id = 15;

#inserare nume departamente (folosind import dintr-un fisier local)
LOAD DATA LOCAL INFILE 'C:/departament.txt' INTO TABLE departament
FIELDS TERMINATED BY ','
LINES TERMINATED BY ';';


#=======================================================
#C:/departament.txt
#=======================================================
#16,'Sales',16;17,'Marketing',18;
#=======================================================
```

SQL – Aplicatie

Inserare date in tabela – folosind 2 sintaxe diferite pentru INSERT

```
#inserare date angajat
INSERT INTO angajat
(nume, prenume, departament_id, manager_id, salariu, angajare)
VALUES
('Popa',   'Ion',         1, NULL, 8000, '2000-1-12'),
('Popescu',    'Maria',   1, 1, 3000, '2003-5-6'),
('Marinescu',  'Vasile', 1, 1, 5000, '2004-6-3'),
('Ionescu',    'Andrei', 1, NULL, 3000, '2002-1-1'),
('Vasilescu',  'Ana',       2, NULL, 2000, '2006-3-3'),
('Dragan',     'Dinu',   2, 5, 2000, '2004-11-12'),
('Mihailescu', 'Adrian', 5, NULL, 2500, '2006-10-12'),
('Teodorescu',     'Matei',  3, NULL, 2000, '2005-1-12'),
('Popescu',    'Vasile', 3, 8, 3000, '2005-9-9'),
('Mateescu',   'Dumitru',3, 8, 3000, '2007-2-5'),
('Calinescu',  'Alin',   4, NULL, 3200, '2005-8-2'),
('Popescu',    'Mihaela',4, 12, 1500, '2005-4-8'),
('Ionescu',    'Diana',  5, NULL, 5000, '2001-1-12');

#inserare date angajat - folosind sintaxa cu 'set <nume_camp>'
INSERT INTO angajat set nume='Marian', set prenume='Maria', set departament_id=6,
                set manager_id=NULL, set salariu=12000, set angajare='2000-3-5';
INSERT INTO angajat set nume='Antonescu', set prenume='Andrei', set departament_id=7,
                set manager_id=NULL, set salariu=14500, set angajare='2009-3-1';
INSERT INTO angajat set nume='Mitu', set prenume='Iulia', set departament_id=8,
                set manager_id=NULL, set salariu=8000, set angajare='2010-6-8';
INSERT INTO angajat set nume='Popescu', set prenume='Adina', set departament_id=9,
                set manager_id=NULL, set salariu=9500, set angajare='2012-12-10';

#inserare masini si marci masini
INSERT INTO marca_masina (nume) VALUES ('Renault','VW','BWM','Audi');
INSERT INTO masina (numar, marca_id) VALUES ('1-B-121',1),('2-B-121',1),('3-B-122',2),('1-B-121',3);
```

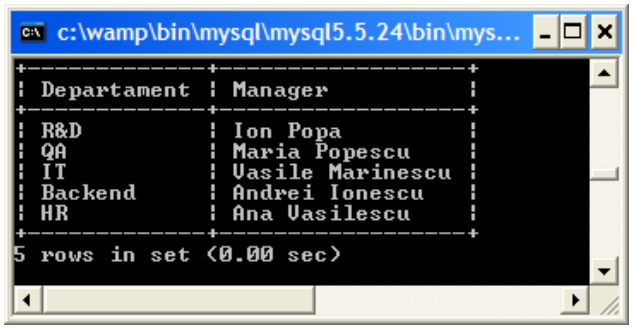
SQL – Aplicatie

Modificare date – UPDATE si stergere date - DELETE

```
#modificare marca masina pentru masina cu un anume numar
UPDATE masina set marca_id = 2 WHERE numer = '2-B-121';
#modificare nume marca masina
UPDATE marca_masina set nume = 'Skoda' WHERE nume = 'Audi';

#stergere marca masina Skoda
DELETE marca_masina WHERE nume = 'Skoda';

#stergere marca masina nefolosita
DELETE marca_masina WHERE id not in (SELECT  marca_id from masina);

#stergere departament 'BI'
DELETE FROM departament WHERE nume='BI';

#crestere salarii (cu 20%) celor care nu au manager
UPDATE angajat set salariu = salariu * 1.20 WHERE manager_id IS NULL;
```

Selectie din doua tabele folosind SELECT si JOIN

```
#Afisati departamentele cu numele managerului fiecarui departament
SELECT d.nume as 'Departament', CONCAT(a.prenume, " ", a.nume) as 'Manager'
FROM departament as d JOIN angajat as a
ON d.manager_id = a.id;
```



SQL – Aplicatie

Selectia rezultatelor, folosind subquery, filtrare cu includerea criteriului de join si join (exemplu cu join intre 2 si 4 tabele)

```
#selectati numele angajatilor care lucreaza la R&D
#(folosind subquery)
SELECT nume, prenume
FROM angajat
WHERE departament_id in (SELECT id  from departament WHERE nume = 'R&D');

#(criteriul de filtrare include conditia de join)
SELECT a.nume AS Nume, prenume AS Prenume
FROM angajat AS a, departament AS d
WHERE a.departament_id = d.id
AND d.nume = 'R&D';

#(folosind join)
SELECT a.nume AS Nume, prenume AS Prenume
FROM angajat AS a JOIN departament AS d
ON a.departament_id = d.id
WHERE d.nume = 'R&D';

#selectati numele, prenumele, id-ul managerului, numele departamentului, masina (cu marca si numarul) asignata
SELECT a.nume as Nume, a.prenume as Prenume, a.manager_id as Manager,
d.nume as Departament,
m.numar as 'Numar masina', mm.nume as 'Marca masina'
FROM angajat as a, departament as d, masina as m, marca as mm
WHERE a.masina_id = m.id and m.marca_id = mm.id
AND a.departament_id = d.id;

SELECT a.nume AS Nume, a.prenume AS Prenume, a.manager_id AS Manager,
d.nume as Departament,
m.numar AS 'Numar masina', mm.nume AS 'Marca masina'
FROM angajat AS a join departament AS d join masina AS m join marca AS mm
ON a.masina_id = m.id and m.marca_id = mm.id
AND a.departament_id = d.id;
```