Laborator nr. 2
Introducere în MySQL *(partea a II-a)*
Comanda SELECT. Funcții și operatori.

Titular disciplină
șef lucr. dr. ing. Cristian Nicolae Buțincu
cristian-nicolae.butincu@academic.tuiasi.ro

Titular aplicații
șef lucr. dr. ing. Cătălin Mironeanu
catalin.mironeanu@academic.tuiasi.ro

Titular aplicații, ing. Sorin Avram
sorin.avram@academic.tuiasi.ro

Titular aplicații, ing. Ionuț-Alexandru Baltariu
ionut-alexandru.baltariu@academic.tuiasi.ro

an universitar 2024 - 2025

Descărcați de pe platforma moodle scriptul cretabs_mysql.sql. Descărcarea implicită se face în directorul: `/home/student/Downloads`

Conectați-vă la baza de date MariaDB `c13_01_db` ... `c14_14_db`, corespunzătoare contului folosit și rulați scriptul descărcat folosind comenzile:

```
SOURCE /home/student/Downloads/cretabs_mysql.sql;
SHOW TABLES;
```

Studiați subcapitolul 15.2.13 SELECT Statement din referința bibliografică [1].

Studiați capitolul 14 Functions and Operators din referința bibliografică [2].

1 Comanda SELECT

Comanda SELECT face parte din categoria comenzilor de manipulare a datelor (eng. *Data Manipulation Language* – DML). Comanda SELECT este utilizată pentru a extrage date dintr-o tabelă sau mai multe tabele dintr-o bază de date. Aceasta permite selectarea unei sau mai multor coloane dintr-un tabel/tabele, cu posibilitatea de a filtra, sorta sau grupa rezultatele în funcție de criteriile specificate. Se pot, de asemenea, combina date din mai multe tabele folosind operații de *join*. De obicei sunt impuse condiții folosind clauza WHERE, iar ordinea rezultatelor poate fi specificată prin ORDER BY. De asemenea, comanda SELECT poate fi însoțită de funcții de agregare, precum SUM(), COUNT() sau AVG(), pentru a realiza calcule pe datele selectate.

Sintaxa completă a comenzii SELECT necesită un studiu mai amplu. Pentru familiarizarea cu această comandă, o sintaxă simplificată este:

```
SELECT [DISTINCT]
    [FROM table_references]
    [WHERE where_condition]
    [GROUP BY col_name]
    [HAVING where_condition]
    [ORDER BY col_name] [ASC | DESC]
    [LIMIT row_count]
```

Exemple de rulare a comenzii SELECT folosind datele importate anterior:

```
#afișarea structurii tabelei departments
DESC departments;
#afișarea tuturor datelor din tabela departments
SELECT * FROM departments;
#afișarea structurii tabelei employees
DESC employees;
#afișarea tuturor datelor din tabela employees
SELECT * FROM employees;
#următoarea comandă folosește o filtrare în clauza WHERE
SELECT * FROM employees WHERE job_id = 'IT_PROG';
#următoarea comandă folosește funcția COUNT()
SELECT COUNT(*) FROM employees WHERE job_id = 'IT_PROG';
```

2 Operatori

Lista completă a operatorilor MySQL o regăsiți în 14.4 Operators din referința bibliografică [2]. Mulți dintre acești operatori se regăsesc și în alte limbaje de programare.

Printre acești operatori, sunt câțiva specifici bazelor de date:

2.1 BETWEEN ... AND ...

Descrierea acestui operator o regăsiți în expr BETWEEN min AND max din referința bibliografică [2].

Acest operator specific bazelor de date permite filtrarea datelor între o valoare minimă și o valoare maximă (interval închis), care produce un rezultat echivalent cu expresia (min <= expr AND expr <= max).

```
SELECT * FROM employees WHERE salary BETWEEN 5000 AND 7000;
#comanda echivalentă folosind operatorii <= și >=
SELECT * FROM employees WHERE salary >= 5000 AND salary <= 7000;
```

Folosirea operatorului NOT în combinație cu operatorul BETWEEN ... AND ...

```
SELECT * FROM employees WHERE salary NOT BETWEEN 5000 AND 7000;
```

2.2 IS NULL

Descrierea acestui operator o regăsiți în IS NULL din referința bibliografică [2].

O valoare NULL nu poate fi comparată cu o altă valoare NULL, de aceea pentru identificarea valorilor NULL se folosește operatorul IS.

```
SELECT * FROM employees WHERE manager_id IS NULL;
```

Folosirea operatorului NOT în combinație cu operatorul IS NULL:

```
SELECT * FROM employees WHERE commission_pct IS NOT NULL;
```

2.3 LIKE

Descrierea acestui operator o regăsiți în expr LIKE pat din referința bibliografică [2].

Cu LIKE se pot folosi următoarele două caractere *wildcard* în model:

- % se potrivește cu orice număr de caractere, chiar și zero caractere.
- _ se potrivește cu exact un caracter.

```
SELECT * FROM employees WHERE first_name LIKE 'El%';
SELECT * FROM employees WHERE phone_decimal LIKE '07______12'; #sunt 6 caractere _
```

Studiați problematica clauzei ESCAPE a operatorului LIKE.

2.4 CASE

Descrierea acestui operator o regăsiți în CASE value WHEN compare_value THEN result [WHEN compare_value THEN result ...] [ELSE result] END din referința bibliografică [2].

```
SELECT department_id, department_name,
       CASE
           WHEN department_name LIKE 'A%' THEN 'Canada'
           WHEN department_name LIKE 'S%' THEN 'Australia'
           ELSE 'America'
       END "Regiunea"
FROM departments;
```

În exemplul anterior se afișează pe lângă id-ul și numele departamentului, o informație despre regiunea în care se află departamentele, funcție de inițiala lor.

3 Funcții

În MySQL, o funcție reprezintă o subrutină stocată care poate primi parametri de intrare și returna o singură valoare. Aceasta este utilizată pentru a încapsula operații frecvent utilizate într-o singură comandă, permițând astfel o manipulare mai eficientă a datelor. Funcțiile pot fi folosite în expresii SQL, precum SELECT sau WHERE.

3.1 Funcții cu valori numerice

Lista completă a funcțiilor care prelucrează valori numerice o regăsiți în 14.6 Numeric Functions and Operators din referința bibliografică [2].

În cadrul laboratorului vor fi discutate și vor fi date exemple pentru funcțiile: ROUND(), TRUNCATE(), FLOOR().

3.2 Funcții cu date calendaristice

Lista completă a funcțiilor care prelucrează date calendaristice o regăsiți în 14.7 Date and Time Functions din referința bibliografică [2].

În cadrul laboratorului vor fi discutate și vor fi date exemple pentru funcțiile: ADDDATE() / DATE_ADD(), DATEDIFF(), LAST_DAY(), DAY() / DAYOFMONTH(), MONTH() și YEAR().

3.3 Funcții cu șiruri de caractere

Lista completă a funcțiilor care prelucrează șiruri de caractere o regăsiți în 14.8 String Functions and Operators din referința bibliografică [2].

În cadrul laboratorului vor fi discutate și vor fi date exemple pentru funcțiile: CONCAT(), INSTR(), LENGTH(), LOWER() / LCASE() și UPPER() / UCASE(), SUBSTR().

```
SELECT first_name, last_name,
       CONCAT(first_name, ' ', last_name) AS "nume complet",
       SUBSTR(CONCAT(first_name, ' ', last_name), 1, INSTR(CONCAT(first_name, ' ',
        ↪  last_name), ' ')) AS "nume extras din nume complet",
       SUBSTR(CONCAT(first_name, ' ', last_name), INSTR(CONCAT(first_name, ' ',
        ↪  last_name), ' ')) AS "prenume extras din nume complet"
FROM employees;
```

3.4 Funcții de agregare

Lista completă a funcțiilor care prelucrează șiruri de caractere o regăsiți în 14.19 Aggregate Functions din referința bibliografică [2].

În cadrul laboratorului vor fi discutate și vor fi date exemple pentru funcțiile: MIN() și MAX(), COUNT(), AVG(), SUM().

4 Activități practice

1. Să se afișeze toți angajații a căror nume conține literele ’a’ și ’o’ (coloana `last_name` din tabela `employees`), în această ordine (mai întâi ’a’ și apoi ’o’).
2. Să se afișeze toți angajații a căror prenume conține literele ’ll’ consecutiv (coloana `first_name` din tabela `employees`).
3. Folosind tabela `employees`, afișați numele, prenumele, numele și prenumele concatenat și lungimea numelui concatenat.
4. Folosind tabela `employees`, afișați numele, salariul și o a treia coloana folosind operatorul CASE care va afișa șirul de caractere "mic", "mediu" sau "mare" în funcție de valoarea salariului. (sub 5000 se va afișa "mic", 5000–7000 se va afișa "mediu", peste 7000 se va afișa "mare"). Cea de-a treia coloană va fi afișată cu numele `tip_sal`.
5. Folosind tabela `employees`, afișați salariul minim, salariul maxim și media salarială utilizând funcțiile de agregare.
6. Folosind tabela `employees`, afișați numele, salariul, salariul mărit cu 23.456% nerotunjit, respectiv rotunjit la a doua zecimală, doar pentru angajații care au manager. Practic, se vor afișa 4 coloane: nume, salariu, salariu mărit nerotunjit și salariu mărit rotunjit.

5 Activități suplimentare

1. Să se afișeze numele departamentului, numele orașului pentru departamentele a căror denumire conține literele ’pp’ consecutiv și se află în orașele care conțin litera ’u’ pe poziția a 3-a și ’h’ pe poziția 5-a (coloana `department_name` din tabela `departments` și coloana `city` din tabela `locations`).
2. Să se afișeze din tabela `employees`, în coloane separate: prenumele, numele și o adresă de e-mail de forma: prima literă din prenume, urmat de caracterul ’.’, urmat de nume, urmat de caracterul ’@’, urmat de primele caractere până la caracterul ’_’ din id-ul job-ului (coloana `job_id`), urmat de ’.acme.com’. Afișarea se va face cu toate literele mici.
   De exemplu, pentru Peter Vargas va rezulta p.vargas@st.acme.com

Bibliografie

[1] “MySQL Docs – Chapter 15 SQL Statements.”
https://dev.mysql.com/doc/refman/9.0/en/sql-statements.html.

[2] “MySQL Docs – Chapter 14 Functions and Operators.”
https://dev.mysql.com/doc/refman/9.0/en/functions.html.