SQL DML: Limbaj de manipulare a datelor

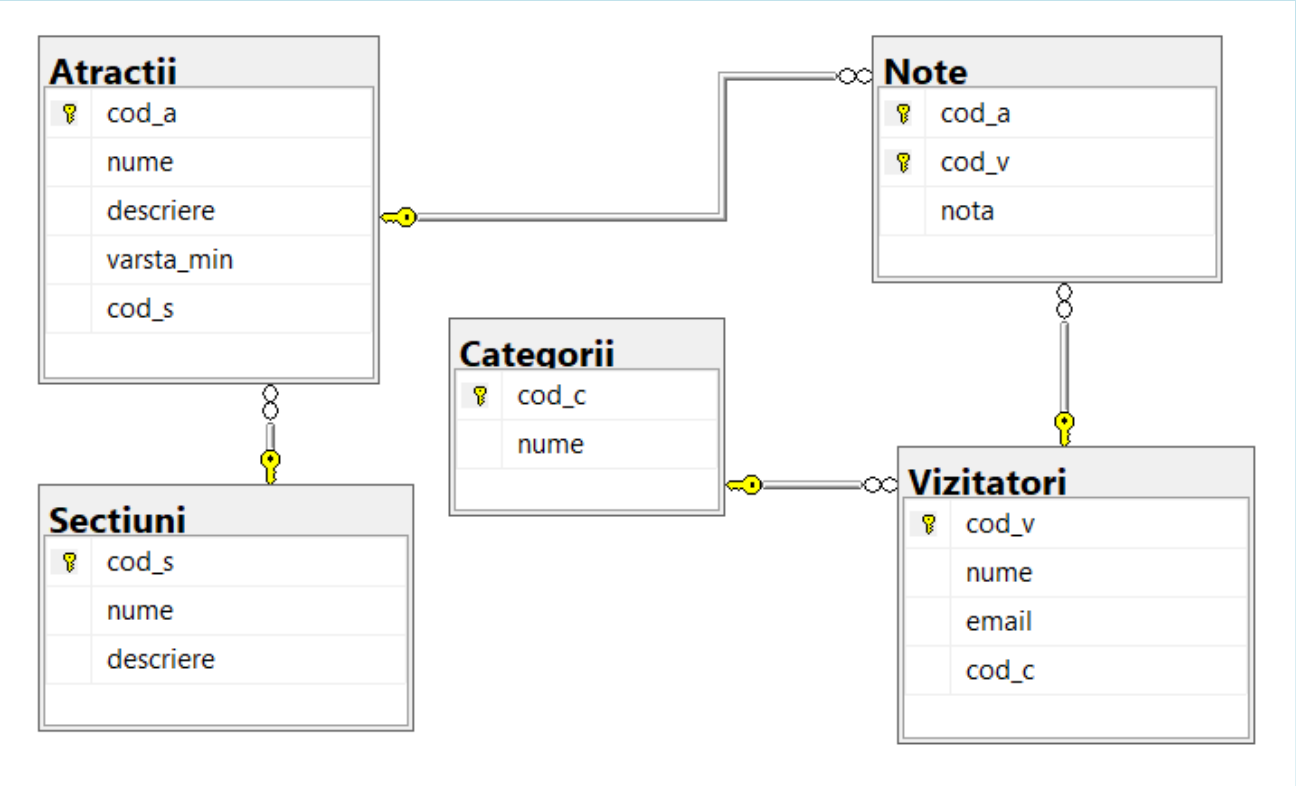
SEMINAR 2

Limbajul SQL: DML

- **DML** (*Data Manipulation Language* - ***Limbaj de manipulare a datelor***) conține instrucțiuni pentru inserare, actualizare, ștergere și interogare a datelor stocate într-o bază de date relațională
- Cele mai folosite **instrucțiuni DML** sunt:
  - **INSERT –** inserează înregistrări noi
  - **UPDATE –** actualizează înregistrări
  - **DELETE –** șterge înregistrări
  - **SELECT –** extrage înregistrări

Limbajul SQL - DML

- Se dă o bază de date având următoarea structură:



Limbajul SQL - DML

- Pentru a adăuga date într-un tabel, vom folosi instrucțiunea **INSERT**
- Exemplu:

```
--Adăugarea unei înregistrări noi în tabelul Sectiuni
INSERT INTO Sectiuni (nume, descriere) VALUES
('sectiunea 1', 'cea mai mare sectiune');
--Adăugarea unei înregistrări noi în tabelul Atractii
INSERT INTO Atractii (nume, descriere, varsta_min, cod_s) VALUES
('rollercoaster', 'cel mai rapid din parc', 12, 1);
```

Observații

- Specificarea coloanelor după numele tabelului este opțională
- Prin specificarea coloanelor controlăm asocierile coloană-valoare, deci nu ne bazăm pe ordinea în care apar coloanele atunci când a fost creat tabelul sau când structura tabelului a fost modificată ultima dată
- Dacă **nu** specificăm o valoare pentru o coloană, ***SQL Server*** va verifica dacă există o valoare implicită pentru coloana respectivă iar dacă nu există și coloana nu permite *NULL* atunci inserarea **nu** va avea loc

Limbajul SQL - DML

- Pentru a actualiza înregistrări într-un tabel, vom folosi instrucțiunea **UPDATE**
- Exemplu:

```
--Modificarea unei înregistrări din tabelul Sectiuni
UPDATE Sectiuni SET descriere='cea mai veche sectiune'
WHERE  nume='sectiunea 1';
```

- !!!Omiterea clauzei **WHERE** va rezulta în actualizarea tuturor înregistrărilor din tabel

Limbajul SQL - DML

- Pentru a șterge înregistrări dintr-un tabel, vom folosi instrucțiunea **DELETE**
- Exemplu:

```
--Ștergerea unei înregistrări din tabelul Sectiuni
DELETE FROM Sectiuni WHERE nume='sectiunea 1';
```

- !!!Omiterea clauzei **WHERE** va rezulta în ștergerea tuturor înregistrărilor din tabel

Limbajul SQL - DML

- Dacă dorim să extragem înregistrări din baza de date, vom folosi instrucțiunea **SELECT**, iar **rezultatul interogării** va fi afișat într-un **tabel rezultat** (*result-set*)
- Exemplu:

```
--Returnarea tuturor înregistrărilor din tabelul Sectiuni

SELECT * FROM Sectiuni;

/*Returnarea tuturor înregistrărilor din tabelul Sectiuni, specificând explicit
numele coloanelor*/

SELECT cod_s, nume, descriere FROM Sectiuni;
```

Limbajul SQL - DML

- Dacă dorim să extragem doar valorile distincte dintr-o coloană (sau combinație de coloane) vom folosi cuvântul cheie **DISTINCT**
- Exemplu:

```
--Returnarea tuturor valorilor distincte din coloana varsta_min din tabelul Atractii

SELECT DISTINCT varsta_min FROM Atractii;
```

Limbajul SQL - DML

- Dacă dorim ca o interogare să returneze doar înregistrări care îndeplinesc anumite criterii, vom folosi clauza **WHERE**
- Exemplu:

```
/*Returnarea numelui și descrierii atracțiilor care au vârsta minimă recomandată egală cu 12*/

SELECT nume, descriere FROM Atractii WHERE varsta_min=12;

/*Returnarea numelui și vârstei minime recomandate ale atracțiilor care au numele diferit de 'Castelul Negru'*/

SELECT nume, varsta_min FROM Atractii WHERE nume<>'Castelul Negru';
```

Limbajul SQL - DML

- Dacă dorim să returnăm toate atracțiile care au vârsta minimă recomandată **mai mare sau egală cu 14**, vom executa următoarea interogare:

```
SELECT * FROM Atractii WHERE varsta_min>=14;
```

- Dacă dorim să returnăm toate atracțiile care au vârsta minimă recomandată **mai mică sau egală cu 16**, vom executa următoarea interogare:

```
SELECT * FROM Atractii WHERE varsta_min<=16;
```

Limbajul SQL - DML

- Dacă dorim să returnăm toate atracțiile care au vârsta minimă recomandată strict mai mare decât 11 și strict mai mică decât 16, vom executa următoarea interogare:

```
SELECT * FROM Atractii WHERE varsta_min>11 AND varsta_min<16;

--Varianta cu operatorul BETWEEN, care specifică un interval închis de valori

SELECT * FROM Atractii WHERE varsta_min BETWEEN 12 AND 15;
```

- Dacă dorim să returnăm toate atracțiile care au vârsta minimă recomandată **în afara intervalului închis [14,18]**, vom executa următoarea interogare:

```
SELECT * FROM Atractii WHERE varsta_min NOT BETWEEN 14 AND 18;
```

- !!!Tipul de date al coloanei *varsta_min* este **int**

Limbajul SQL - DML

- Dacă dorim să returnăm toate atracțiile care au vârsta minimă recomandată **egală** cu 12, 14 sau 16 vom executa următoarea interogare:

```
SELECT * FROM Atractii WHERE varsta_min IN (12,14,16);
```

- Dacă dorim să returnăm toate atracțiile care au vârsta minimă recomandată **diferită** de 12, 14 sau 16 vom executa următoarea interogare:

```
SELECT * FROM Atractii WHERE varsta_min NOT IN (12,14,16);
```

Limbajul SQL - DML

- Pentru a specifica șabloane de căutare într-o coloană, vom folosi operatorul **LIKE**
- Dacă dorim să returnăm toate înregistrările din tabelul *Vizitatori* pentru care numele începe cu litera A, vom executa următoarea interogare:

```
SELECT * FROM Vizitatori WHERE nume LIKE 'A%';
```

- Dacă dorim să returnăm toate înregistrările din tabelul *Vizitatori* pentru care numele se termină cu litera a, vom executa următoarea interogare:

```
SELECT * FROM Vizitatori WHERE nume LIKE '%a';
```

Limbajul SQL - DML

- Dacă dorim să returnăm toate înregistrările din tabelul *Vizitatori* pentru care numele conține ‘ana’, vom executa următoarea interogare:

```
SELECT * FROM Vizitatori WHERE nume LIKE '%ana%';
```

- Dacă dorim să returnăm toate înregistrările din tabelul *Vizitatori* pentru care numele se termină cu ‘na’ și este format din 3 caractere, vom executa următoarea interogare:

```
SELECT * FROM Vizitatori WHERE nume LIKE '_na';
```

- !!!Caracterul _ înlocuiește un singur caracter
- !!!Caracterul % înlocuiește zero sau mai multe caractere

Limbajul SQL - DML

- Dacă dorim să returnăm toate înregistrările din tabelul *Vizitatori* pentru care numele începe cu litera A, B sau C, vom executa următoarea interogare:

```
SELECT * FROM Vizitatori WHERE nume LIKE '[ABC]%';
```

- Dacă dorim să returnăm toate înregistrările din tabelul *Vizitatori* pentru care numele **nu** începe cu litera A, B sau C, vom executa următoarea interogare:

```
SELECT * FROM Vizitatori WHERE nume LIKE '[^ABC]%';
```

Limbajul SQL - DML

- Dacă dorim să extragem toate înregistrările din tabelul *Vizitatori* pentru care coloana *email* are valoarea *NULL*, vom executa următoarea interogare:

```
SELECT * FROM Vizitatori WHERE email IS NULL;
```

- Dacă dorim să extragem toate înregistrările din tabelul *Vizitatori* pentru care coloana *email* are valoarea **diferită** de *NULL*, vom executa următoarea interogare:

```
SELECT * FROM Vizitatori WHERE email IS NOT NULL;
```

Limbajul SQL - DML

- Dacă dorim să extragem numele categoriei, numele vizitatorului și adresa de email pentru toți vizitatorii care aparțin unei categorii, vom executa următoarea interogare:


```
SELECT C.nume AS categorie, V.nume, V.email FROM Categorii C,
Vizitatori V WHERE C.cod_c=V.cod_c;
```

- Interogarea poate fi rescrisă utilizând **INNER JOIN**:

```
SELECT C.nume AS categorie, V.nume, V.email FROM Categorii C INNER
JOIN Vizitatori V ON C.cod_c=V.cod_c;
```

Limbajul SQL - DML

- Dacă dorim să afișăm numele categoriilor și adresa de email a vizitatorilor, **incluzând și categoriile care nu au vizitatori asociați** (dar **nu** și vizitatorii care nu aparțin unei categorii), vom executa următoarea interogare:

```
SELECT C.nume, V.email FROM Categorii C LEFT JOIN Vizitatori V ON
C.cod_c=V.cod_c;
```

- Dacă dorim să afișăm numele categoriilor și adresa de email a vizitatorilor, **incluzând și vizitatorii care nu aparțin unei categorii** (dar **nu** și categoriile care nu au vizitatori asociați), vom executa următoarea interogare:

```
SELECT C.nume, V.email FROM Categorii C RIGHT JOIN Vizitatori V ON
C.cod_c=V.cod_c;
```

Limbajul SQL - DML

- Dacă dorim să afișăm numele categoriilor și adresa de email a vizitatorilor, **incluzând atât vizitatorii care nu aparțin unei categorii, cât și categoriile care nu au vizitatori asociați**, vom executa următoarea interogare:

```
SELECT C.nume, V.email FROM Categorii C FULL JOIN Vizitatori V
ON C.cod_c=V.cod_c;
```

Limbajul SQL - DML

- Dacă dorim să realizăm un calcul pe o mulțime de valori și să returnăm o singură valoare, vom utiliza **funcții de agregare**
- Funcțiile de agregare se utilizează de obicei împreună cu clauzele **GROUP BY** și **HAVING**
- Exemple de funcții de agregare: **COUNT()**, **SUM()**, **AVG()**, **MIN()**, **MAX()**
- Dacă dorim să returnăm numărul total de înregistrări din tabelul *Categorii*, vom executa următoarea interogare:

```
SELECT COUNT(*) FROM Categorii;
```

Limbajul SQL - DML

- Dacă dorim să afișăm numele categoriilor și numărul de vizitatori pentru fiecare categorie care are **cel puțin un vizitator**, vom executa următoarea interogare:

```
SELECT C.nume, COUNT(cod_v) nr_vizitatori FROM Categorii C INNER JOIN
Vizitatori V ON C.cod_c=V.cod_c GROUP BY C.cod_c, C.nume;
```

- Dacă dorim să afișăm numele categoriilor și numărul de vizitatori pentru fiecare categorie, vom executa următoarea interogare:

```
SELECT C.nume, COUNT(cod_v) nr_vizitatori FROM Categorii C LEFT JOIN
Vizitatori V ON C.cod_c=V.cod_c GROUP BY C.cod_c, C.nume;
```

Limbajul SQL - DML

- Dacă dorim să afișăm numele atracției și media aritmetică a notelor primite pentru toate atracțiile care au primit note, vom executa următoarea interogare:

```
SELECT A.nume, AVG(nota) medie_note FROM Atractii A INNER JOIN Note N
ON A.cod_a=N.cod_a GROUP BY A.cod_a, A.nume;
```

- Dacă dorim să afișăm numele atracției și media aritmetică a notelor primite pentru toate atracțiile care au primit note și au media aritmetică strict mai mare decât 9, vom executa următoarea interogare:

```
SELECT A.nume, AVG(nota) medie_note FROM Atractii A INNER JOIN Note N
ON A.cod_a=N.cod_a GROUP BY A.cod_a, A.nume HAVING AVG(nota)>9;
```

Subinterogări

- O subinterogare este o interogare încorporată într-o altă interogare
- Se poate folosi o subinterogare în clauza **WHERE** pentru a găsi înregistrările dintr-un tabel care se potrivesc cu înregistrările din alt tabel **fără** a folosi **JOIN**
- Dacă dorim să afișăm numele tuturor categoriilor care au cel puțin un vizitator, vom executa următoarea interogare:

```
SELECT nume FROM Categorii

WHERE cod_c IN (SELECT cod_c FROM Vizitatori);
```

Subinterogări

- Dacă dorim, putem rescrie interogarea folosind **INNER JOIN**:

```
SELECT DISTINCT C.nume FROM Categorii C INNER JOIN Vizitatori V ON
C.cod_c=V.cod_c;
```

- Putem rescrie interogarea folosind operatorul **EXISTS**:

```
SELECT C.nume FROM Categorii C

WHERE EXISTS(SELECT * FROM Vizitatori V WHERE V.cod_c=C.cod_c);
```

- !!!Operatorul **EXISTS** returnează valoarea **TRUE** dacă rezultatul subinterogării conține cel puțin o înregistrare

Subinterogări

- O subinterogare în clauza **WHERE** poate fi folosită și pentru a găsi înregistrările din primul tabel care **nu** au potriviri în cel de-al doilea tabel
- Dacă dorim să afișăm numele tuturor categoriilor care **nu** au niciun vizitator, vom executa următoarea interogare:

```
SELECT nume FROM Categorii WHERE cod_c NOT IN (SELECT cod_c FROM
Vizitatori WHERE cod_c IS NOT NULL);
```

- Dacă dorim, putem rescrie interogarea folosind operatorul **NOT EXISTS**:

```
SELECT C.nume FROM Categorii C WHERE NOT EXISTS (SELECT * FROM
Vizitatori V WHERE V.cod_c=C.cod_c);
```

Limbajul SQL - DML

- Dacă dorim să afișăm numele atracțiilor care au primit **cel puțin o dată** nota 9, vom executa următoarea interogare:

```
SELECT nume FROM Atractii

WHERE cod_a = ANY(SELECT cod_a FROM Note WHERE nota=9);
```

- Dacă dorim, putem să rescriem interogarea folosind operatorul **IN**:

```
SELECT nume FROM Atractii

WHERE cod_a IN (SELECT cod_a FROM Note WHERE nota=9);
```

Limbajul SQL - DML

- Dacă dorim să afișăm numele atracțiilor care **nu** au primit nota 9 (dar au primit cel puțin o notă):

```
SELECT nume FROM Atractii WHERE cod_a <> ALL(SELECT cod_a FROM
Note WHERE nota=9) AND cod_a IN (SELECT cod_a FROM Note);
```

- Dacă dorim, putem să rescriem interogarea folosind operatorul **NOT IN**:

```
SELECT nume FROM Atractii WHERE cod_a NOT IN (SELECT cod_a FROM
Note WHERE nota=9) AND cod_a IN (SELECT cod_a FROM Note);
```

Limbajul SQL - DML

- Dacă dorim să afișăm numele atracțiilor care **nu** au primit **niciodată** nota 9, dar au primit **cel puțin o dată** nota 10, vom executa următoarea interogare:

```
SELECT nume FROM Atractii
WHERE cod_a IN (SELECT cod_a FROM Note WHERE nota=10)
EXCEPT
SELECT nume FROM Atractii
WHERE cod_a IN (SELECT cod_a FROM Note WHERE nota=9);
```

Limbajul SQL - DML

- Dacă dorim să afișăm numele atracțiilor care au primit **cel puțin o dată** nota 9 sau **cel puțin o dată** nota 10, vom executa următoarea interogare:

```
SELECT nume FROM Atractii
WHERE cod_a IN (SELECT cod_a FROM Note WHERE nota=10)
UNION [ALL]
SELECT nume FROM Atractii
WHERE cod_a IN (SELECT cod_a FROM Note WHERE nota=9);
```

- !!!**UNION ALL** include înregistrări duplicate; **UNION nu** include înregistrări duplicate

Limbajul SQL - DML

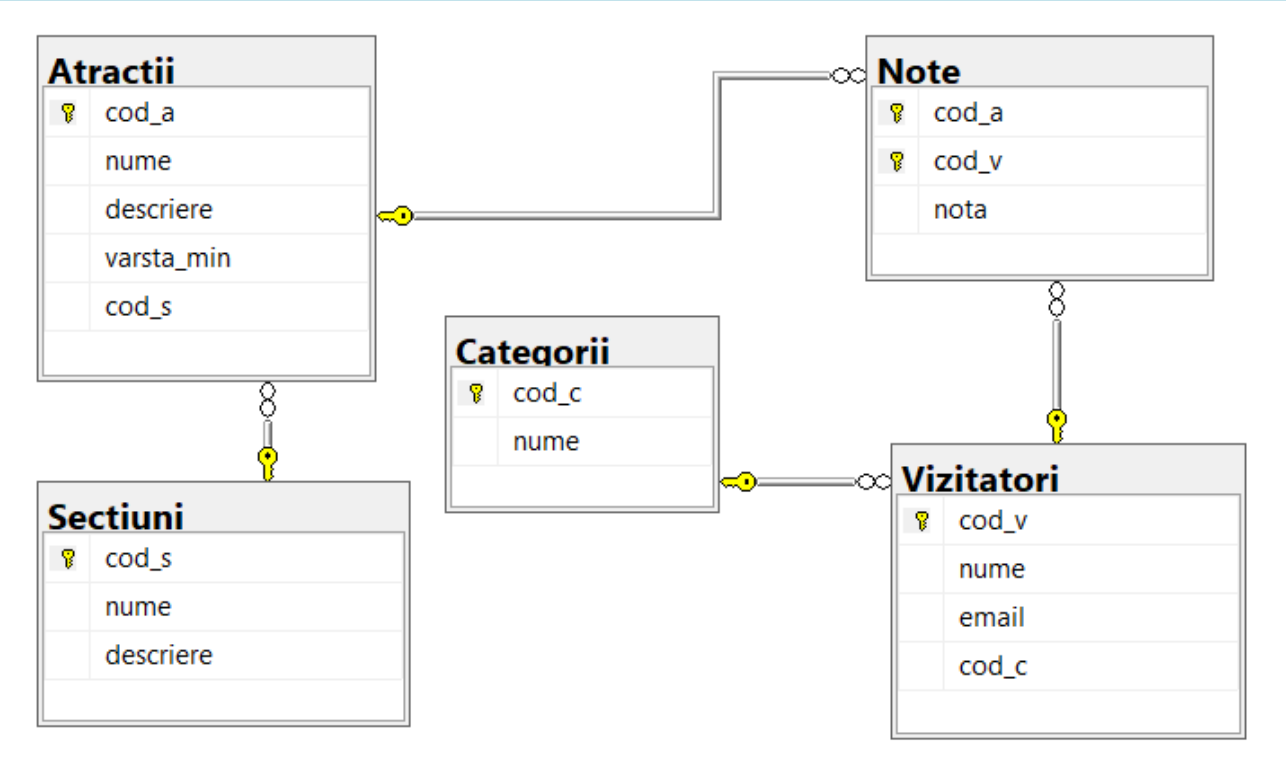
- Dacă dorim să afișăm numele atracțiilor care au primit **cel puțin o dată** nota 9 și **cel puțin o dată** nota 10, vom executa următoarea interogare:

```
SELECT nume FROM Atractii
WHERE cod_a IN (SELECT cod_a FROM Note WHERE nota=10)
INTERSECT
SELECT nume FROM Atractii
WHERE cod_a IN (SELECT cod_a FROM Note WHERE nota=9);
```

- !!!Cu ajutorul operatorilor **UNION**, **INTERSECT** și **EXCEPT** se pot îmbina rezultatele a două sau mai multe interogări într-un singur *result-set* (fiecare interogare trebuie să conțină același număr de coloane, iar tipurile coloanelor trebuie să fie compatibile)

Problemă propusă

- Se dă baza de date cu următoarea structură:



Problemă propusă

Cerințe:

1) Populați fiecare tabel cu câte 7 înregistrări

2) Actualizați câte o înregistrare din fiecare tabel

3) Ștergeți o înregistrare din tabelul *Note*

4) Scrieți o interogare care afișează toate înregistrările din tabelul *Categorii* al căror nume este egal cu ‘pensionari’ sau ‘copii’

5) Scrieți o interogare care afișează toate înregistrările din tabelul *Sectiuni* al căror nume începe cu litera C

Problemă propusă

6) Scrieți o interogare care afișează toate înregistrările din tabelul *Sectiuni* al căror nume se termină cu litera n și au cel puțin două caractere

7) Scrieți o interogare care afișează toți vizitatorii care nu au dat nicio notă niciunei atracții

8) Scrieți o interogare care afișează numele vizitatorilor, nota și numele atracției evaluate

9) Scrieți o interogare care afișează numele vizitatorilor și numărul de note pe care l-au dat atracțiilor (se vor include și numele vizitatorilor care nu au dat nicio notă)

10) Scrieți o interogare care afișează valorile distincte ale notelor date atracțiilor

Problemă propusă

11) Scrieți o interogare care afișează numele secțiunii, numele și descrierea atracțiilor pentru toate secțiunile care au cel puțin o atracție asociată (se vor include și atracțiile care nu au o secțiune asociată)

12) Scrieți o interogare care afișează numele și vârsta minimă recomandată a atracției și numărul de note primite pentru toate atracțiile care au primit cel puțin 2 note

13) Scrieți o interogare care afișează numele categoriei, numele vizitatorului, nota, numele și descrierea atracției pentru toate categoriile care au numele diferit de ‘adult’ și au vizitatori asociați care au dat cel puțin o notă unei atracții

Problemă propusă

14) Scrieți o interogare care afișează nota maximă primită de către fiecare atracție și numele atracției (se vor selecta doar acele atracții care au primit cel puțin o notă)

15) Scrieți o interogare care afișează nota minimă primită de către fiecare atracție și numele atracției (se vor selecta doar acele atracții care au primit cel puțin o notă)

16) Scrieți o interogare care afișează numele și adresa de email a vizitatorilor care nu aparțin niciunei categorii

17) Scrieți o interogare care afișează numele și descrierea atracțiilor care aparțin unei categorii (valoarea codului de categorie să fie diferită de *NULL*)

Bibliografie

- https://learn.microsoft.com/en-us/sql/t-sql/queries/select-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/queries/select-clause-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/queries/select-examples-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/queries/select-group-by-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/queries/select-having-transact-sql?view=sql-server-ver16

Bibliografie

- https://learn.microsoft.com/en-us/sql/t-sql/queries/from-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/queries/where-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/queries/is-null-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/language-elements/like-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/language-elements/between-transact-sql?view=sql-server-ver16

Bibliografie

- https://learn.microsoft.com/en-us/sql/t-sql/language-elements/exists-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/language-elements/in-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/language-elements/not-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/language-elements/or-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/language-elements/some-any-transact-sql?view=sql-server-ver16

Bibliografie

- https://learn.microsoft.com/en-us/sql/t-sql/language-elements/all-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/language-elements/and-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/language-elements/set-operators-union-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/language-elements/set-operators-except-and-intersect-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/statements/insert-transact-sql?view=sql-server-ver16

Bibliografie

- https://learn.microsoft.com/en-us/sql/t-sql/queries/update-transact-sql?view=sql-server-ver16
- https://learn.microsoft.com/en-us/sql/t-sql/statements/delete-transact-sql?view=sql-server-ver16