

Proiect Final

Canuci Robert-Alin

08.08.2024

Partea I

1. Explicați pe scurt ce sunt cerințele de business, la ce ne folosesc și cine le creează.

Cerințele de business reprezintă documentațiile care descriu nevoile și așteptările organizației față de un sistem software sau un proiect. Ele ne folosesc pentru a defini clar obiectivele proiectului, pentru a stabili criteriile de succes și pentru a ghida procesul de dezvoltare și testare. Cerințele de business sunt create de stakeholderi, incluzând manageri de produs, analiști de business, și alți membri ai echipei de proiect, în colaborare cu utilizatorii finali și clienții.

2. Explicați diferența între un test condition și test case.

Test condition: Reprezintă un aspect specific al sistemului software care trebuie testat pentru a verifica dacă funcționează conform specificațiilor. De exemplu, un test condition ar putea fi "verificarea autentificării utilizatorului."

Test case: Este un set de pași specifici, date de intrare, și rezultate așteptate folosite pentru a testa un anumit aspect al aplicației. Un test case derivat din test condition "verificarea autentificării utilizatorului" ar putea include pașii de a introduce un nume de utilizator și o parolă, și verificarea că accesul este acordat sau refuzat în funcție de validitatea credențialelor.

3. Enumerați și explicați pe scurt etapele procesului de testare.

Planificarea testării: Definirea strategiei de testare, a obiectivelor și a resurselor necesare.

Analiza cerințelor: Înțelegerea și evaluarea cerințelor de business și tehnice pentru a identifica ce trebuie testat.

Designul testării: Crearea test case-urilor, test scripts-urilor și a mediului de testare necesar.

Implementarea testării: Pregătirea și configurarea mediului de testare, instalarea și configurarea aplicației.

Executarea testării: Rularea test case-urilor și înregistrarea rezultatelor.

Evaluarea rezultatelor: Analizarea rezultatelor testării pentru a determina dacă cerințele sunt îndeplinite și identificarea defectelor.

Raportarea defectelor: Documentarea și comunicarea defectelor găsite echipei de dezvoltare.

Retesting și regression testing: Verificarea corectării defectelor (retesting) și asigurarea că alte părți ale aplicației nu au fost afectate de schimbările făcute (regression testing).

Încheierea testării: Revizuirea și raportarea activităților de testare și rezultatelor finale, și închiderea procesului de testare.

Monitorizare și control: Activitate continuă care se desfășoară începând cu etapa de planificare și se termină cu etapa de închidere, având rolul de a compara progresul actual cu planul de testare. În cazul în care se observă riscul de a nu ne îndeplini obiectivele se iau măsuri de control.

4. Explicați diferența între retesting și regression testing.

Retesting: Este procesul de retestare a unui defect specific după ce a fost reparat pentru a verifica dacă problema a fost rezolvată.

Regression testing: Este procesul de testare a întregii aplicații sau a unei părți semnificative a acesteia pentru a asigura că noile schimbări nu au introdus alte defecte.

5. Explicați diferența între functional testing și non-functional testing.

Functional testing: Testează funcționalitățile sistemului pentru a verifica dacă acestea funcționează conform specificațiilor. Exemple includ testarea login-ului, gestionarea utilizatorilor, etc.

Non-functional testing: Evaluează aspecte non-funcționale ale sistemului, cum ar fi performanța, securitatea, utilizabilitatea, etc. Exemple includ testarea timpului de răspuns sub sarcină, testarea securității la atacuri, etc.

6. Explicați diferența între blackbox testing și whitebox testing.

Blackbox testing: Testează aplicația fără a cunoaște structura internă a codului. Se concentrează pe input-uri și output-uri și verifică dacă sistemul funcționează conform specificațiilor.

Whitebox testing: Testează aplicația având cunoștințe despre structura internă a codului. Se concentrează pe verificarea logicii interne, a fluxurilor de control, a buclelor, etc.

7. Enumerați tehnicile de testare și grupați-le în funcție de categorie (blackbox, whitebox, experience-based).

Blackbox testing:

- Testare de echivalență a partițiilor
- Analiza valorilor de frontieră
- Testare de tip state transition
- Testare de caz de utilizare

Whitebox testing:

- Testare de acoperire a codului
- Testare de bucle
- Testare de ramuri/decizii
- Testare de fluxuri de control

Experience-based testing:

- Testare exploratorie
- Testare ad-hoc
- Testare bazată pe erori din trecut

8. Explicați diferența între verification și validation.

Verification: Procesul de evaluare a produsului software în stadiile de dezvoltare pentru a se asigura că produsul este construit corect (conform specificațiilor). Exemple includ revizuirea designului, inspectarea codului, etc.

Validation: Procesul de evaluare a produsului final pentru a se asigura că produsul corespunde nevoilor și cerințelor utilizatorilor finali. Exemple includ testarea funcționalității, testarea utilizabilității, etc.

9. Explicați diferența între positive testing și negative testing și dați câte un exemplu din fiecare.

Positive testing: Testarea sistemului cu input-uri valide pentru a verifica dacă sistemul funcționează conform așteptărilor. De exemplu, testarea login-ului cu un nume de utilizator și o parolă corecte.

Negative testing: Testarea sistemului cu input-uri invalide pentru a verifica dacă sistemul gestionează corect situațiile de eroare. De exemplu, testarea login-ului cu un nume de utilizator sau o parolă incorecte și verificarea că accesul este refuzat.

10. Enumerați și explicați pe scurt nivelurile de testare.

Unit Testing: Testează componente individuale sau module ale aplicației pentru a verifica corectitudinea lor.

Integration Testing: Testează interacțiunile dintre module pentru a verifica dacă funcționează împreună conform specificațiilor.

System Testing: Testează sistemul complet integrat pentru a verifica dacă îndeplinește cerințele specificate.

Acceptance Testing: Testează sistemul din perspectiva utilizatorului final pentru a valida că îndeplinește nevoile și cerințele acestuia.

Partea a II-a

I. Crearea bazei de date si instructiuni DDL

Scenariu: *Vom crea o baza de date HR pentru a gestiona angajatii, departamentele, bonusurile de performanta, sesiunile de training si angajatii care participa la sesiunile de training.*

1. 1 Crearea bazei de date si a tabelelor

```
CREATE DATABASE HRDatabase;
USE HRDatabase;

CREATE TABLE Departments (
    DepartmentID INT PRIMARY KEY,
    DepartmentName VARCHAR(50) NOT NULL
);

CREATE TABLE Employees (
    EmployeeID INT PRIMARY KEY,
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    DepartmentID INT,
    Salary DECIMAL(10, 2),
    PerformanceBonus DECIMAL(10, 2),
    FOREIGN KEY (DepartmentID) REFERENCES
Departments (DepartmentID)
);

CREATE TABLE Projects (
    ProjectID INT PRIMARY KEY,
    ProjectName VARCHAR(100) NOT NULL,
    DepartmentID INT,
    FOREIGN KEY (DepartmentID) REFERENCES
Departments (DepartmentID)
);

CREATE TABLE EmployeeProjects (
    EmployeeID INT,
    ProjectID INT,
    PRIMARY KEY (EmployeeID, ProjectID),
    FOREIGN KEY (EmployeeID) REFERENCES
Employees (EmployeeID),
    FOREIGN KEY (ProjectID) REFERENCES
Projects (ProjectID)
);

CREATE TABLE Training (
    TrainingID INT PRIMARY KEY,
    TrainingName VARCHAR(100) NOT NULL,
    DateConducted DATE NOT NULL,
    TrainerName VARCHAR(50) NOT NULL
);

CREATE TABLE EmployeeTraining (
    EmployeeID INT,
    TrainingID INT,
    PRIMARY KEY (EmployeeID, TrainingID),
    FOREIGN KEY (EmployeeID) REFERENCES
Employees (EmployeeID),
    FOREIGN KEY (TrainingID) REFERENCES
Training (TrainingID)
);
```

1.2 Relații între tabele și chei

Employees și Departaments

Tabele: Employees (Angajați), Departments (Departamente)

Cheie primară în Departaments: DepartmentID

Cheie secundară în Employees: DepartmentID

Relație: (1,n)

Explicație: Tabelul Departments are o cheie primară DepartmentID, care identifică în mod unic fiecare departament. Tabelul Employees are o cheie secundară DepartmentID care face referire la DepartmentID din tabelul Departments. Această relație semnifică faptul că fiecare angajat aparține unui singur departament, dar fiecare departament poate avea mai mulți angajați.

Employees și Projects

Tabele: Employees (Angajați), Projects (Proiecte), EmployeeProjects (ProiecteAngajați)

Cheie primară în Employees: EmployeeID

Cheie primară în Projects: ProjectID

Cheie secundare în EmployeesProjects: EmployeeID, ProjectID

Relație: (n,n)

Explicație: Tabelul Employees are o cheie primară EmployeeID, iar tabelul Projects are o cheie primară ProjectID. Tabelul EmployeeProjects este un tabel de legătură cu două chei secundare: EmployeeID și ProjectID. Acest tabel stabilește o relație de tip (n,n) între Employees și Projects, indicând faptul că fiecare angajat poate lucra la mai multe proiecte și fiecare proiect poate avea mai mulți angajați.

Employees și Training

Tabele: Employees (Angajați), Training (Training), EmployeeTraining (TrainingAngajați)

Cheie primară în Employees: EmployeeID

Cheie primară în Training: TrainingID

Cheii secundare în TrainingEmployees: EmployeeID, TrainingID

Relație: (n,n)

Explicație: Tabelul Employees are o cheie primară EmployeeID, iar tabelul Training are o cheie primară TrainingID. Tabelul EmployeeTraining este un tabel de legătură cu două chei secundare: EmployeeID și TrainingID. Acest tabel stabilește o relație de tip (n,n) între Employees și Training, indicând faptul că fiecare angajat poate participa la mai multe sesiuni de training și fiecare sesiune de training poate avea mai mulți participanți.

2. Alter Table

```
ALTER TABLE Employees ADD DateHired DATE;
```

	EmployeeID	FirstName	LastName	DepartmentID	Salary	PerformanceBonus
*	NULL	NULL	NULL	NULL	NULL	NULL

61 • ALTER TABLE Employees ADD DateHired DATE;

62

<

Result Grid

Filter Rows:

Edit:

Export/Import:

	EmployeeID	FirstName	LastName	DepartmentID	Salary	PerformanceBonus	DateHired
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

3.Drop Table

```
DROP TABLE EmployeeProjects;
```

4.Truncate Table

```
TRUNCATE TABLE EmployeeProjects;
```

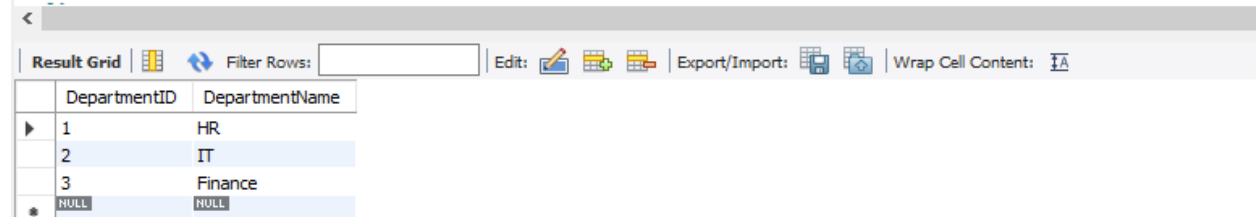
II. Instructiuni DML

Scenariu: *Adaugarea, modificarea si stergerea datelor din tabele.*

1. Adaugarea datelor

```
INSERT INTO Departments (DepartmentID, DepartmentName) VALUES (1, 'HR'), (2, 'IT'), (3, 'Finance');
```

63 • INSERT INTO Departments (DepartmentID, DepartmentName) VALUES (1, 'HR'), (2, 'IT'), (3, 'Finance');



DepartmentID	DepartmentName
1	HR
2	IT
3	Finance
NULL	NULL

```
INSERT INTO Employees (EmployeeID, FirstName, LastName, DepartmentID, Salary, PerformanceBonus, DateHired) VALUES
```

```
(1, 'Marian', 'Mihai', 1, 50000, 1200, '2022-01-15'),  
(2, 'Alina', 'Andreiescu', 2, 60000, 900, '2021-03-20'),  
(3, 'Vasile', 'Cazacu', 3, 55000, 1500, '2023-06-10'),  
(4, 'Alexandru', 'Popescu', 1, 47000, 1100, '2022-02-12'),  
(5, 'Ioana', 'Marinescu', 2, 52000, 950, '2021-07-22'),  
(6, 'Andrei', 'Ionescu', 3, 58000, 1300, '2023-01-11'),  
(7, 'Elena', 'Georgescu', 1, 60000, 1050, '2023-03-30'),  
(8, 'Mihai', 'Dumitrescu', 2, 49000, 800, '2022-05-14'),  
(9, 'Ana', 'Popa', 3, 53000, 1200, '2021-08-19'),  
(10, 'George', 'Stan', 1, 54000, 1400, '2022-09-07'),  
(11, 'Maria', 'Tudor', 2, 61000, 1600, '2023-04-24'),  
(12, 'Vlad', 'Luca', 3, 57000, 1150, '2021-10-13'),  
(13, 'Adriana', 'Nedelcu', 1, 62000, 1800, '2023-05-18'),  
(14, 'Florin', 'Manea', 2, 65000, 1250, '2022-11-11');
```

	EmployeeID	FirstName	LastName	DepartmentID	Salary	PerformanceBonus	DateHired
▶	1	Marian	Mihai	1	50000.00	1200.00	2022-01-15
	2	Alina	Andreiescu	2	60000.00	900.00	2021-03-20
	3	Vasile	Cazacu	3	55000.00	1500.00	2023-06-10
	4	Alexandru	Popescu	1	47000.00	1100.00	2022-02-12
	5	Ioana	Marinescu	2	52000.00	950.00	2021-07-22
	6	Andrei	Ionescu	3	58000.00	1300.00	2023-01-11
	7	Elena	Georgescu	1	60000.00	1050.00	2023-03-30
	8	Mihai	Dumitrescu	2	49000.00	800.00	2022-05-14
	9	Ana	Popa	3	53000.00	1200.00	2021-08-19
	10	George	Stan	1	54000.00	1400.00	2022-09-07
	11	Maria	Tudor	2	61000.00	1600.00	2023-04-24
	12	Vlad	Luca	3	57000.00	1150.00	2021-10-13
	13	Adriana	Nedelcu	1	62000.00	1800.00	2023-05-18
	14	Florin	Manea	2	65000.00	1250.00	2022-11-11

```
INSERT INTO Projects (ProjectID, ProjectName, DepartmentID) VALUES (1,
'Project A', 2), (2, 'Project B', 3);
71 • INSERT INTO Projects (ProjectID, ProjectName, DepartmentID) VALUES (1, 'Project A', 2), (2, 'Project B', 3);
```

ProjectID	ProjectName	DepartmentID
1	Project A	2
2	Project B	3
NULL	NULL	NULL

```
INSERT INTO EmployeeProjects (EmployeeID, ProjectID) VALUES (1, 1), (2, 1),
(3, 2);
```

```
73 • INSERT INTO EmployeeProjects (EmployeeID, ProjectID) VALUES (1, 1), (2, 1), (3, 2);
```

EmployeeID	ProjectID
1	1
2	1
3	2
NULL	NULL

```
INSERT INTO Training (TrainingID, TrainingName, DateConducted, TrainerName)
VALUES (1, 'Leadership Training', '2023-05-10', 'Alice Vasilescu'), (2,
'Project Management', '2023-06-15', 'Marian Ivan'), (3, 'Technical Skills',
'2023-07-20', 'Claudia Ionescu');
```

```
75 • INSERT INTO Training (TrainingID, TrainingName, DateConducted, TrainerName)
76 VALUES (1, 'Leadership Training', '2023-05-10', 'Alice Vasilescu'),
77 (2, 'Project Management', '2023-06-15', 'Marian Ivan'),
78 (3, 'Technical Skills', '2023-07-20', 'Claudia Ionescu');
```

TrainingID	TrainingName	DateConducted	TrainerName
1	Leadership Training	2023-05-10	Alice Vasilescu
2	Project Management	2023-06-15	Marian Ivan
3	Technical Skills	2023-07-20	Claudia Ionescu
NULL	NULL	NULL	NULL

```
INSERT INTO EmployeeTraining (EmployeeID, TrainingID) VALUES (1, 1), (2, 2),
(3, 3), (1, 3);
```

```
84 • INSERT INTO EmployeeTraining (EmployeeID, TrainingID) VALUES (1, 1), (2, 2), (3, 3), (1, 3);
```

EmployeeID	TrainingID
1	1
2	2
1	3
3	3
NULL	NULL

2. Modificarea datelor

```
UPDATE Employees SET Salary = 58000 WHERE EmployeeID = 1;
```

EmployeeID	FirstName	LastName	DepartmentID	Salary	PerformanceBonus	DateHired
1	Marian	Mihai	1	50000.00	1200.00	2022-01-15
2	Alina	Andreiescu	2	60000.00	900.00	2021-03-20
3	Vasile	Cazacu	3	55000.00	1500.00	2023-06-10
4	Alexandru	Popescu	1	47000.00	1100.00	2022-02-12
5	Ioana	Marinescu	2	52000.00	950.00	2021-07-22
6	Andrei	Ionescu	3	58000.00	1300.00	2023-01-11
7	Elena	Georgescu	1	60000.00	1050.00	2023-03-30
8	Mihai	Dumitrescu	2	49000.00	800.00	2022-05-14
9	Ana	Popa	3	53000.00	1200.00	2021-08-19
10	George	Stan	1	54000.00	1400.00	2022-09-07
11	Maria	Tudor	2	61000.00	1600.00	2023-04-24
12	Vlad	Luca	3	57000.00	1150.00	2021-10-13
13	Adriana	Nedelcu	1	62000.00	1800.00	2023-05-18
14	Florin	Manea	2	65000.00	1250.00	2022-11-11

```
80 • UPDATE Employees SET Salary = 58000 WHERE EmployeeID = 1;
```

EmployeeID	FirstName	LastName	DepartmentID	Salary	PerformanceBonus	DateHired
1	Marian	Mihai	1	58000.00	1200.00	2022-01-15
2	Alina	Andreiescu	2	60000.00	900.00	2021-03-20
3	Vasile	Cazacu	3	55000.00	1500.00	2023-06-10
4	Alexandru	Popescu	1	47000.00	1100.00	2022-02-12
5	Ioana	Marinescu	2	52000.00	950.00	2021-07-22
6	Andrei	Ionescu	3	58000.00	1300.00	2023-01-11
7	Elena	Georgescu	1	60000.00	1050.00	2023-03-30
8	Mihai	Dumitrescu	2	49000.00	800.00	2022-05-14
9	Ana	Popa	3	53000.00	1200.00	2021-08-19
10	George	Stan	1	54000.00	1400.00	2022-09-07
11	Maria	Tudor	2	61000.00	1600.00	2023-04-24
12	Vlad	Luca	3	57000.00	1150.00	2021-10-13
13	Adriana	Nedelcu	1	62000.00	1800.00	2023-05-18
14	Florin	Manea	2	65000.00	1250.00	2022-11-11

3. Stergerea datelor

EmployeeID	FirstName	LastName	DepartmentID	Salary	PerformanceBonus	DateHired
1	Marian	Mihai	1	50000.00	1200.00	2022-01-15
2	Alina	Andreiescu	2	60000.00	900.00	2021-03-20
3	Vasile	Cazacu	3	55000.00	1500.00	2023-06-10
4	Alexandru	Popescu	1	47000.00	1100.00	2022-02-12
5	Ioana	Marinescu	2	52000.00	950.00	2021-07-22
6	Andrei	Ionescu	3	58000.00	1300.00	2023-01-11
7	Elena	Georgescu	1	60000.00	1050.00	2023-03-30
8	Mihai	Dumitrescu	2	49000.00	800.00	2022-05-14
9	Ana	Popa	3	53000.00	1200.00	2021-08-19
10	George	Stan	1	54000.00	1400.00	2022-09-07
11	Maria	Tudor	2	61000.00	1600.00	2023-04-24
12	Vlad	Luca	3	57000.00	1150.00	2021-10-13
13	Adriana	Nedelcu	1	62000.00	1800.00	2023-05-18
14	Florin	Manea	2	65000.00	1250.00	2022-11-11

```
DELETE FROM Employees WHERE EmployeeID = 3;
```

EmployeeID	FirstName	LastName	DepartmentID	Salary	PerformanceBonus	DateHired
1	Marian	Mihai	1	50000.00	1200.00	2022-01-15
2	Alina	Andreiescu	2	60000.00	900.00	2021-03-20
4	Alexandru	Popescu	1	47000.00	1100.00	2022-02-12
5	Ioana	Marinescu	2	52000.00	950.00	2021-07-22
6	Andrei	Ionescu	3	58000.00	1300.00	2023-01-11
7	Elena	Georgescu	1	60000.00	1050.00	2023-03-30
8	Mihai	Dumitrescu	2	49000.00	800.00	2022-05-14
9	Ana	Popa	3	53000.00	1200.00	2021-08-19
10	George	Stan	1	54000.00	1400.00	2022-09-07
11	Maria	Tudor	2	61000.00	1600.00	2023-04-24
12	Vlad	Luca	3	57000.00	1150.00	2021-10-13
13	Adriana	Nedelcu	1	62000.00	1800.00	2023-05-18
14	Florin	Manea	2	65000.00	1250.00	2022-11-11
NULL	NULL	NULL	NULL	NULL	NULL	NULL

III. Instructiuni DQL

Scenariu: *Selectia si analiza datelor din tabele.*

1. Afisarea tuturor datelor din tabelul Employees.

```
SELECT * FROM Employees;
```

	EmployeeID	FirstName	LastName	DepartmentID	Salary	PerformanceBonus	DateHired
▶	1	Marian	Mihai	1	50000.00	1200.00	2022-01-15
	2	Alina	Andreiescu	2	60000.00	900.00	2021-03-20
	4	Alexandru	Popescu	1	47000.00	1100.00	2022-02-12
	5	Ioana	Marinescu	2	52000.00	950.00	2021-07-22
	6	Andrei	Ionescu	3	58000.00	1300.00	2023-01-11
	7	Elena	Georgescu	1	60000.00	1050.00	2023-03-30
	8	Mihai	Dumitrescu	2	49000.00	800.00	2022-05-14
	9	Ana	Popa	3	53000.00	1200.00	2021-08-19
	10	George	Stan	1	54000.00	1400.00	2022-09-07
	11	Maria	Tudor	2	61000.00	1600.00	2023-04-24
	12	Vlad	Luca	3	57000.00	1150.00	2021-10-13
	13	Adriana	Nedelcu	1	62000.00	1800.00	2023-05-18
	14	Florin	Manea	2	65000.00	1250.00	2022-11-11
✱	NULL	NULL	NULL	NULL	NULL	NULL	NULL

2. Afisarea anumitor coloane

```
SELECT FirstName, LastName, Salary FROM Employees;
```

	FirstName	LastName	Salary
▶	Marian	Mihai	50000.00
	Alina	Andriescu	60000.00
	Alexandru	Popescu	47000.00
	Ioana	Marinescu	52000.00
	Andrei	Ionescu	58000.00
	Elena	Georgescu	60000.00
	Mihai	Dumitrescu	49000.00
	Ana	Popa	53000.00
	George	Stan	54000.00
	Maria	Tudor	61000.00
	Vlad	Luca	57000.00
	Adriana	Nedelcu	62000.00
	Florin	Manea	65000.00

3. Filtrarea cu WHERE

```
SELECT * FROM Employees WHERE DepartmentID = 1;
```

[illegible]

4. Filtrarea cu LIKE

```
SELECT * FROM Employees WHERE LastName LIKE 'M%';
```

	EmployeeID	FirstName	LastName	DepartmentID	Salary	PerformanceBonus	DateHired
▶	1	Marian	Mihai	1	50000.00	1200.00	2022-01-15
	5	Ioana	Marinescu	2	52000.00	950.00	2021-07-22
	14	Florin	Manea	2	65000.00	1250.00	2022-11-11
✱	NULL	NULL	NULL	NULL	NULL	NULL	NULL

5. Filtrarea cu AND si OR

```
SELECT * FROM Employees WHERE Salary > 50000 AND PerformanceBonus > 1000;
```

	EmployeeID	FirstName	LastName	DepartmentID	Salary	PerformanceBonus	DateHired
▶	6	Andrei	Ionescu	3	58000.00	1300.00	2023-01-11
	7	Elena	Georgescu	1	60000.00	1050.00	2023-03-30
	9	Ana	Popa	3	53000.00	1200.00	2021-08-19
	10	George	Stan	1	54000.00	1400.00	2022-09-07
	11	Maria	Tudor	2	61000.00	1600.00	2023-04-24
	12	Vlad	Luca	3	57000.00	1150.00	2021-10-13
	13	Adriana	Nedelcu	1	62000.00	1800.00	2023-05-18
	14	Florin	Manea	2	65000.00	1250.00	2022-11-11

```
SELECT * FROM Employees WHERE DepartmentID = 2 OR DepartmentID = 3;
```

	EmployeeID	FirstName	LastName	DepartmentID	Salary	PerformanceBonus	DateHired
▶	2	Alina	Andreiescu	2	60000.00	900.00	2021-03-20
	5	Ioana	Marinescu	2	52000.00	950.00	2021-07-22
	8	Mihai	Dumitrescu	2	49000.00	800.00	2022-05-14
	11	Maria	Tudor	2	61000.00	1600.00	2023-04-24
	14	Florin	Manea	2	65000.00	1250.00	2022-11-11
	6	Andrei	Ionescu	3	58000.00	1300.00	2023-01-11
	9	Ana	Popa	3	53000.00	1200.00	2021-08-19
	12	Vlad	Luca	3	57000.00	1150.00	2021-10-13
✱	NULL	NULL	NULL	NULL	NULL	NULL	NULL

6. Functii agregate

```
SELECT AVG(Salary) AS AverageSalary FROM Employees;
```

	AverageSalary
▶	56000.000000

```
SELECT SUM(PerformanceBonus) AS TotalBonus FROM Employees;
```

	TotalBonus
▶	15700.00

7. Filtrarea pe functii agregate

```
SELECT DepartmentID, SUM(Salary) AS TotalSalary FROM Employees GROUP BY  
DepartmentID HAVING SUM(Salary) > 50000;
```

	DepartmentID	TotalSalary
	1	273000.00
	2	287000.00
▶	3	168000.00

8. Join-uri

-- Inner Join

```
SELECT e.FirstName, e.LastName,  
d.DepartmentName  
FROM Employees e  
INNER JOIN Departments d ON  
e.DepartmentID = d.DepartmentID;
```

	FirstName	LastName	DepartmentName
▶	Marian	Mihai	HR
	Alexandru	Popescu	HR
	Elena	Georgescu	HR
	George	Stan	HR
	Adriana	Nedelcu	HR
	Alina	Andreiescu	IT
	Ioana	Marinescu	IT
	Mihai	Dumitrescu	IT
	Maria	Tudor	IT
	Florin	Manea	IT
	Andrei	Ionescu	Finance
	Ana	Popa	Finance
	Vlad	Luca	Finance

-- Left Join

```
SELECT e.FirstName, e.LastName, d.DepartmentName  
FROM Employees e  
LEFT JOIN Departments d ON e.DepartmentID = d.DepartmentID;
```

	FirstName	LastName	DepartmentName
▶	Marian	Mihai	HR
	Alina	Andreiescu	IT
	Alexandru	Popescu	HR
	Ioana	Marinescu	IT
	Andrei	Ionescu	Finance
	Elena	Georgescu	HR
	Mihai	Dumitrescu	IT
	Ana	Popa	Finance
	George	Stan	HR
	Maria	Tudor	IT
	Vlad	Luca	Finance
	Adriana	Nedelcu	HR
	Florin	Manea	IT

-- Right Join

```
SELECT e.FirstName, e.LastName, d.DepartmentName
FROM Employees e
RIGHT JOIN Departments d ON e.DepartmentID = d.DepartmentID;
```

	FirstName	LastName	DepartmentName
►	Marian	Mihai	HR
	Alexandru	Popescu	HR
	Elena	Georgescu	HR
	George	Stan	HR
	Adriana	Nedelcu	HR
	Alina	Andreiescu	IT
	Ioana	Marinescu	IT
	Mihai	Dumitrescu	IT
	Maria	Tudor	IT
	Florin	Manea	IT
	Andrei	Ionescu	Finance
	Ana	Popa	Finance
	Vlad	Luca	Finance

-- Cross Join

```
SELECT e.FirstName, e.LastName, p.ProjectName
FROM Employees e
CROSS JOIN Projects p;
```

	FirstName	LastName	ProjectName
►	Marian	Mihai	Project B
	Marian	Mihai	Project A
	Alina	Andreiescu	Project B
	Alina	Andreiescu	Project A
	Alexandru	Popescu	Project B
	Alexandru	Popescu	Project A
	Ioana	Marinescu	Project B
	Ioana	Marinescu	Project A
	Andrei	Ionescu	Project B
	Andrei	Ionescu	Project A
	Elena	Georgescu	Project B
	Elena	Georgescu	Project A
	Mihai	Dumitrescu	Project B
	Mihai	Dumitrescu	Project A
	Ana	Popa	Project B
	Ana	Popa	Project A
	George	Stan	Project B
	George	Stan	Project A
	Maria	Tudor	Project B
	Maria	Tudor	Project A
	Vlad	Luca	Project B
	Vlad	Luca	Project A
	Adriana	Nedelcu	Project B
	Adriana	Nedelcu	Project A
	Florin	Manea	Project B
	Florin	Manea	Project A

9. Limite si Order By

```
SELECT * FROM Employees ORDER BY Salary DESC LIMIT 5;
```

	EmployeeID	FirstName	LastName	DepartmentID	Salary	PerformanceBonus	DateHired
►	14	Florin	Manea	2	65000.00	1250.00	2022-11-11
	13	Adriana	Nedelcu	1	62000.00	1800.00	2023-05-18
	11	Maria	Tudor	2	61000.00	1600.00	2023-04-24
	2	Alina	Andreiescu	2	60000.00	900.00	2021-03-20
	7	Elena	Georgescu	1	60000.00	1050.00	2023-03-30
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

IV. Subquery-uri

```
SELECT * FROM Employees  
WHERE Salary > (SELECT AVG(Salary) FROM Employees);
```

	EmployeeID	FirstName	LastName	DepartmentID	Salary	PerformanceBonus	DateHired
►	2	Alina	Andreiescu	2	60000.00	900.00	2021-03-20
	6	Andrei	Ionescu	3	58000.00	1300.00	2023-01-11
	7	Elena	Georgescu	1	60000.00	1050.00	2023-03-30
	11	Maria	Tudor	2	61000.00	1600.00	2023-04-24
	12	Vlad	Luca	3	57000.00	1150.00	2021-10-13
	13	Adriana	Nedelcu	1	62000.00	1800.00	2023-05-18
	14	Florin	Manea	2	65000.00	1250.00	2022-11-11
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Selectia angajatilor cu bonusul de performanta mai mare de \$1000

```
SELECT FirstName, LastName FROM Employees  
WHERE PerformanceBonus > 1000;
```

	FirstName	LastName
►	Marian	Mihai
	Alexandru	Popescu
	Andrei	Ionescu
	Elena	Georgescu
	Ana	Popa
	George	Stan
	Maria	Tudor
	Vlad	Luca
	Adriana	Nedelcu
	Florin	Manea

Exemple de Utilizare a Funcțiilor Agregate și Filtrare

Exemplul 1: Calcularea Numărului de Angajați pe Departament

Această interogare va calcula numărul de angajați în fiecare departament, folosind funcția COUNT(EmployeeID) pentru a număra câți angajați sunt în fiecare grup determinat de DepartmentID.

Exemplul 2: Filtrarea Departamentelor cu Mai Mult de 3 Angajați

Această interogare adaugă clauza HAVING pentru a filtra doar departamentele care au mai mult de 3 angajați. HAVING COUNT(EmployeeID) > 3 asigură că sunt selectate doar înregistrările unde numărul de angajați este mai mare de 3.

Exemplul 3: Calcularea Salariului Mediu pe Departament

Această interogare calculează salariul mediu pentru fiecare departament folosind funcția AVG(Salary) în combinație cu GROUP BY DepartmentID pentru a grupa rezultatele după DepartmentID.

Exemplul 4: Filtrarea Departamentelor cu Salariul Mediu Mai Mare de \$50000

În această interogare, HAVING AVG(Salary) > 50000 filtrează departamentele pentru care salariul mediu este mai mare de \$50,000.

```
-- Exemplul 1: Numărul de Angajați  
pe Departament  
SELECT DepartmentID,  
COUNT(EmployeeID) AS NumEmployees  
FROM Employees  
GROUP BY DepartmentID;
```

```
-- Exemplul 2: Departamentele cu  
Mai Mult de 3 Angajați  
SELECT DepartmentID,  
COUNT(EmployeeID) AS NumEmployees  
FROM Employees  
GROUP BY DepartmentID  
HAVING COUNT(EmployeeID) > 3;
```

```
-- Exemplul 3: Salariul Mediu pe  
Departament  
SELECT DepartmentID, AVG(Salary) AS  
AvgSalary  
FROM Employees  
GROUP BY DepartmentID;
```

```
-- Exemplul 4: Departamentele cu  
Salariul Mediu Mai Mare de $50000  
SELECT DepartmentID, AVG(Salary) AS  
AvgSalary  
FROM Employees  
GROUP BY DepartmentID  
HAVING AVG(Salary) > 50000;
```

Aceste interogări demonstrează utilizarea funcțiilor agregate (COUNT, AVG) împreună cu GROUP BY pentru a grupa datele în funcție de un criteriu specific (de exemplu, DepartmentID). Apoi, utilizarea clauzei HAVING permite filtrarea grupurilor bazate pe rezultatele funcțiilor agregate.

Multumesc pentru timpul acordat!