Universitatea Politehnica din București

Facultatea de Automatică și Calculatoare

Veloce Motors

Sistem de Gestiune pentru Dealer Auto

Documentație Tehnică

Proiect Baze de Date

Student:

Nedelcu Bianca-Nicoleta

Grupa 332AA

Cuprins

Capitolul 1

Introducere

1.1 Context și Motivație

În industria auto, gestionarea eficientă a unui dealer presupune coordonarea unui volum mare de informații: vehicule în stoc, clienți cu preferințe diverse, angajați cu roluri specifice, tranzacții și plăți. Un sistem informatic bine proiectat poate transforma aceste provocări în avantaje competitive.

Veloce Motors reprezintă o soluție software completă pentru digitalizarea operațiunilor unui dealer auto. Aplicația integrează toate aspectele activității comerciale într-o platformă unitară, accesibilă prin browser web.

1.2 Obiectivele Proiectului

Proiectul își propune să demonstreze competențele în domeniul bazelor de date relaționale prin:

- Analiza cerințelor și modelarea domeniului de aplicație
- Proiectarea unei scheme de bază de date normalizate
- Implementarea relațiilor între entități (1:1, 1:N, N:N)
- Definirea constrângerilor pentru asigurarea integrității datelor
- Scrierea interogărilor SQL de complexitate variată
- Integrarea bazei de date într-o aplicație web funcțională

1.3 Tehnologii Utilizate

Stiva tehnologică a fost aleasă pentru robustețe și compatibilitate:

Backend:	Python cu framework-ul Flask
Baza de date:	Microsoft SQL Server 2014
Conectivitate:	PyODBC (driver ODBC pentru SQL Server)
Frontend:	HTML5, CSS3, JavaScript
Template Engine:	Jinja2 (integrat în Flask)

Capitolul 2

Descrierea Cerințelor

2.1 Analiza Domeniului

Un dealer auto operează cu trei categorii principale de utilizatori, fiecare cu nevoi și drepturi de acces distincte. Sistemul trebuie să reflecte această structură organizațională.

2.1.1 Administrator

Administratorul are control complet asupra sistemului. Responsabilitățile includ:

- **Gestiunea personalului** – adăugarea de noi angajați cu generarea automată a conturilor de acces, modificarea datelor existente și eliminarea angajaților care nu mai activează (cu verificarea prealabilă a dependențelor)
- **Gestiunea inventarului** – introducerea vehiculelor noi în stoc cu toate specificațiile tehnice, actualizarea prețurilor și statusurilor, eliminarea vehiculelor din evidență
- **Rapoarte analitice** – accesul la statistici de vânzări, performanțe ale angajaților, analize de profitabilitate pe mărci, identificarea clienților fideli

2.1.2 Vânzător

Vânzătorul gestionează relația cu clienții și procesul de vânzare:

- **Procesarea cererilor** – vizualizarea cererilor de achiziție în așteptare, preluarea unei cereri (devenind responsabil pentru finalizare), negocierea prețului final, înregistrarea vânzării
- **Vânzări directe** – posibilitatea de a înregistra vânzări pentru clienți existenți fără a trece prin sistemul de cereri
- **Monitorizare performanță** – acces la istoricul propriu de vânzări și la totalul valorii generate

2.1.3 Client

Clientul beneficiază de o experiență de tip e-commerce:

- **Explorare catalog** – navigare pe categorii (mărci), vizualizarea specificațiilor tehnice, căutare după multiple criterii

- **Coș de cumpărături** – adăugarea vehiculelor de interes, gestionarea listei, inițierea procesului de achiziție
- **Urmărirea comenzilor** – vizualizarea statusului cererilor de achiziție, posibilitatea de anulare pentru cererile în așteptare
- **Plăți** – efectuarea plății pentru achizițiile finalizate prin Card, Cash sau Transfer Bancar
- **Liste de preferințe** – salvarea modelelor de interes pentru acces rapid ulterior

2.2 Cerințe Funcționale Sintetizate

Rezumat Cerințe

Autentificare: Sistem cu 3 roluri (Admin, Vânzător, Client)
Vehicule: Catalog cu specificații tehnice complete, gestiune stoc
Clienți: Înregistrare, profil, istoric achiziții
Vânzări: Flux complet cerere → aprobare → vânzare → plată
Rapoarte: Analize de performanță și profitabilitate

Capitolul 3

Proiectarea Bazei de Date

3.1 Identificarea Entităților

Din analiza cerințelor au rezultat următoarele entități principale:

1. Tabela_Angajati – personalul dealer-ului (vânzători, administratori)
2. Tabela_Clienti – persoanele fizice sau juridice care achiziționează vehicule
3. Tabela_Utilizatori – conturile de autentificare în sistem
4. Tabela_Modele_Auto – catalogul de modele cu specificații tehnice
5. Tabela_Masini_Stoc – vehiculele individuale disponibile în stoc
6. Tabela_Vanzari – tranzacțiile de vânzare înregistrate
7. Tabela_Plati – plățile efectuate pentru vânzări
8. Tabela_Cereri_Achizitie – solicitările de cumpărare ale clienților
9. Tabela_Cos_Client – coșul virtual de cumpărături
10. Tabela_Clienti_Modele_Auto – tabelă asociativă pentru relația N:N

3.2 Schema Relațională

Schema bazei de date respectă principiile normalizării și asigură integritatea referențială prin chei străine.

Angajati (ID_Angajat, Nume, Prenume, Data_Angajare)
Clienti (ID_Client, Tip_Client, Nume, Prenume, Tara, Oras, Strada, Numar_Strada, Telefon)
Utilizatori (UserID, NumeUtilizator, Parola, Rol, *ID_Angajat_FK*, *ID_Client_FK*)
Modele_Auto (ID_Model, Nume_Model, Marca, Generatie, Tip_Caroserie, Combustibil, Capacitate_Cilindrica, Norma_Poluare, Putere, Transmisie)
Masini_Stoc (VIN, *ID_Model*, Culoare_Exterior, Data_Intrare_Stoc, Cost_Achizitie, Pret, Status_Vanzare)
Vanzari (ID_Vanzare, *VIN_Masina*, *ID_Client*, *ID_Angajat*, Data_Vanzare, Pret_Final)
Plati (ID_Plata, *ID_Vanzare*, Data_Platii, Suma_Platita, Metoda_Plata)

Cereri_Achizitie (ID_Cerere, *ID_Client*, *VIN_Masina*, Data_Cerere, Mesaj_Client, Status_Cerere, *ID_Vanzator_Preluat*, Data_Preluare)
Cos_Client (ID_Cos, *ID_Client*, *VIN_Masina*, Data_Adaugare)
Clienti_Modele_Auto (ID, *ID_Client*, *ID_Model*)

Notație: Cheie primară, Cheie străină

Capitolul 4

Descrierea Detaliată a Tabelelor

4.1 Tabela Angajati

Tabela Angajati stochează informațiile de bază despre personalul dealer-ului auto. Fiecare angajat poate fi asociat ulterior cu un cont de utilizator pentru accesul în sistem.

Coloană	**Tip de Date**	**Constrângeri**	**Descriere**
ID_Angajat	INT	PK, IDENTITY(1,1)	Identificator unic generat automat la inserare
Nume	NVARCHAR(50)	NOT NULL	Numele de familie al angajatului
Prenume	NVARCHAR(50)	NOT NULL	Prenumele angajatului
Data_Angajare	DATE	NOT NULL	Data la care a început contractul de muncă

Observații: Această tabelă este referențiată de Tabela_Utilizatori (pentru asocierea contului) și de Tabela_Vanzari (pentru evidența vânzătorului responsabil). Ștergerea unui angajat cu vânzări înregistrate este blocată la nivel de aplicație pentru păstrarea istoricului.

4.2 Tabela Clienti

Tabela Clienti conține datele despre cumpărătorii de vehicule. Sistemul suportă atât persoane fizice cât și persoane juridice, diferențiate prin câmpul `Tip_Client`.

Coloană	**Tip de Date**	**Constrângeri**	**Descriere**
ID_Client	INT	PK, IDENTITY(1,1)	Identificator unic autogenerat
Tip_Client	NVARCHAR(20)	NOT NULL	"Persoana Fizica" sau "Persoana Juridica"
Nume	NVARCHAR(50)	NOT NULL	Numele clientului sau denumirea firmei
Prenume	NVARCHAR(50)	NOT NULL	Prenumele (sau reprezentant legal)
Tara	NVARCHAR(50)	NULL	Țara de reședință
Oras	NVARCHAR(50)	NULL	Localitatea
Strada	NVARCHAR(100)	NULL	Adresa – strada
Numar_Strada	NVARCHAR(10)	NULL	Numărul imobilului
Telefon	NVARCHAR(20)	NULL	Număr de contact

4.3 Tabela Utilizatori

Tabela Utilizatori gestionează autentificarea și autorizarea în sistem. Fiecare utilizator este asociat fie cu un angajat, fie cu un client, în funcție de rol.

Coloană	**Tip de Date**	**Constrângeri**	**Descriere**
UserID	INT	PK, IDENTITY(1,1)	Identificator unic auto-generat
NumeUtilizator	NVARCHAR(50)	NOT NULL, UNIQUE	Username pentru autentificare
Parola	NVARCHAR(100)	NOT NULL	Parola contului
Rol	NVARCHAR(20)	NOT NULL	"Admin", "Vanzator" sau "Client"
ID_Angajat_FK	INT	FK → Angajati, NULL	Referință pentru Admin/-Vânzător
ID_Client_FK	INT	FK → Clienti, NULL	Referință pentru Client

Observații: Constrângerea UNIQUE pe `NumeUtilizator` previne duplicarea conturilor. Pentru angajați (Admin, Vânzător), se populează `ID_Angajat_FK`; pentru clienți, se populează `ID_Client_FK`.

4.4 Tabela Modele_Auto

Tabela Modele_Auto reprezintă catalogul de modele auto disponibile, cu specificații tehnice detaliate. Un model poate avea multiple instanțe fizice în stoc.

Coloană	**Tip de Date**	**Constrângeri**	**Descriere**
ID_Model	INT	PK, IDENTITY(1,1)	Identificator unic
Nume_Model	NVARCHAR(50)	NOT NULL	Denumirea modelului (ex: "X5")
Marca	NVARCHAR(50)	NOT NULL	Producătorul (ex: "BMW")
Generatie	NVARCHAR(50)	NULL	Generația (ex: "G05 2018-2023")
Tip_Caroserie	NVARCHAR(50)	NULL	Sedan, SUV, Hatchback, Break, etc.
Combustibil	NVARCHAR(50)	NULL	Benzină, Diesel, Hibrid, Electric
Capacitate_Cilindrica	INT	NULL	Capacitatea motorului în cm^3
Norma_Poluare	NVARCHAR(50)	NULL	Euro 5, Euro 6, etc.
Putere	INT	NULL	Puterea motorului în CP
Transmisie	NVARCHAR(50)	NULL	Manuală sau Automată

Observații: Separarea modelelor de mașinile din stoc evită redundanța datelor. Când

se adaugă o mașină cu un model inexistent, se creează automat și înregistrarea în această tabelă.

4.5 Tabela Masini_Stoc

Tabela Masini_Stoc conține inventarul efectiv de vehicule. Fiecare înregistrare reprezintă un vehicul fizic unic, identificat prin VIN (Vehicle Identification Number).

Coloană	**Tip de Date**	**Constrângeri**	**Descriere**
VIN	NVARCHAR(50)	PK	Seria de șasiu – identificator unic mondial
ID_Model	INT	FK → Modele_Auto, NOT NULL	Referință către specificațiile modelului
Culoare_Exterior	NVARCHAR(50)	NOT NULL	Culoarea caroseriei
Data_Intrare_Stoc	DATETIME	NOT NULL	Momentul achiziției de către dealer
Cost_Achizitie	NUMERIC(10,2)	NOT NULL	Prețul plătit de dealer (€)
Pret	NUMERIC(10,2)	NOT NULL	Prețul de vânzare afișat (€)
Status_Vanzare	NVARCHAR(50)	NOT NULL	“Disponibil”, “Rezervat”, “Vandut”

Observații: VIN-ul este cheie primară naturală deoarece este unic la nivel global. Statusul “Rezervat” se aplică automat când un client trimite o cerere de achiziție.

4.6 Tabela Vanzari

Tabela Vanzari înregistrează tranzacțiile finalizate. Fiecare vânzare leagă un vehicul de clientul cumpărător și angajatul responsabil.

Coloană	**Tip de Date**	**Constrângeri**	**Descriere**
ID_Vanzare	INT	PK, IDENTITY(1,1)	Identificator unic autogenerat
VIN_Masina	NVARCHAR(50)	FK → Masini_Stoc, NOT NULL	Vehiculul vândut
ID_Client	INT	FK → Clienti, NOT NULL	Cumpărătorul
ID_Angajat	INT	FK → Angajati, NOT NULL	Vânzătorul care a procesat
Data_Vanzare	DATETIME	NOT NULL	Momentul finalizării tranzacției
Pret_Final	NUMERIC(10,2)	NOT NULL	Prețul negociat final (€)

Observații: Prețul final poate diferi de prețul listat în stoc, permițând negocieri. La înregistrarea vânzării, statusul mașinii devine automat “Vandut”.

4.7 Tabela Plati

Tabela Plati evidențiază încasările pentru vânzările efectuate. O vânzare are asociată o singură plată (plată integrală).

Coloană	Tip de Date	Constrângeri	Descriere
ID_Plata	INT	PK, IDENTITY(1,1)	Identificator unic auto-generat
ID_Vanzare	INT	FK → Vanzari, NOT NULL	Vânzarea pentru care se face plata
Data_Platii	DATETIME	NOT NULL	Momentul efectuării plății
Suma_Platita	NUMERIC(10,2)	NOT NULL	Valoarea achitată (€)
Metoda_Plata	NVARCHAR(50)	NOT NULL	"Card", "Cash", "Transfer Bancar"

Observații: Sistemul verifică la nivel de aplicație că nu se poate efectua plată dublă pentru aceeași vânzare.

4.8 Tabela Cereri_Achizitie

Tabela Cereri_Achizitie gestionează fluxul de solicitări ale clienților. O cerere trece prin mai multe stări până la finalizare sau anulare.

Coloană	Tip de Date	Constrângeri	Descriere
ID_Cerere	INT	PK, IDENTITY(1,1)	Identificator unic
ID_Client	INT	FK → Clienti, NOT NULL	Clientul solicitant
VIN_Masina	NVARCHAR(50)	FK → Masini_Stoc, NOT NULL	Vehiculul dorit
Data_Cerere	DATETIME	DEFAULT GETDATE()	Momentul trimiterii cererii
Mesaj_Client	NVARCHAR(500)	NULL	Observații sau cerințe speciale
Status_Cerere	NVARCHAR(20)	DEFAULT "In Asteptare"	Starea curentă a cererii
ID_Vanzator_Preluat	INT	FK → Angajati, NULL	Vânzătorul care a preluat
Data_Preluare	DATETIME	NULL	Momentul preluării

Statusuri posibile: "In Asteptare" → "Preluata" → "Finalizata" sau "Anulata"

4.9 Tabela Cos_Client

Tabela Cos_Client implementează funcționalitatea de coș de cumpărături, permițând clienților să salveze vehiculele de interes înainte de a iniția o cerere.

Coloană	**Tip de Date**	**Constrângeri**	**Descriere**
ID_Cos	INT	PK, IDENTITY(1,1)	Identificator unic
ID_Client	INT	FK → Clienti, NOT NULL	Clientul proprietar al coșului
VIN_Masina	NVARCHAR(50)	FK → Masini_Stoc, NOT NULL	Vehiculul adăugat în coș
Data_Adaugare	DATETIME	DEFAULT GETDATE()	Momentul adăugării

Observații: Când clientul trimite cerere de achiziție pentru un vehicul, acesta este eliminat automat din coș.

4.10 Tabela Clienti_Modele_Auto

Tabela Clienti_Modele_Auto este o tabelă asociativă (de legătură) care implementează relațiaMany-to-Many dintre clienți și modelele de interes.

Coloană	**Tip de Date**	**Constrângeri**	**Descriere**
ID	INT	PK, IDENTITY(1,1)	Identificator unic
ID_Client	INT	FK → Clienti, NOT NULL	Clientul interesat
ID_Model	INT	FK → Modele_Auto, NOT NULL	Modelul marcat ca favorit

Observații: Această tabelă permite clienților să salveze modele preferate. Din pagina de favorite pot adăuga rapid în coș vehicule disponibile din modelele salvate.

Capitolul 5

Relații și Constrângeri de Integritate

5.1 Tipuri de Relații

Baza de date implementează toate cele trei tipuri fundamentale de relații între entități.

5.1.1 Relații 1:1 (One-to-One)

Relațiile 1:1 conectează entități care au o corespondență unică bidirecțională.

Entitate 1	**Entitate 2**	**Descriere**
Angajati	Utilizatori	Fiecare angajat are exact un cont de utilizator pentru acces în sistem. Contul este creat automat la adăugarea angajatului.
Clienti	Utilizatori	Fiecare client are exact un cont pentru autentificare și gestionarea achizițiilor.

Tabela 5.1: Relațiile 1:1 din baza de date

5.1.2 Relații 1:N (One-to-Many)

Relațiile 1:N reprezintă majoritatea conexiunilor din schema bazei de date.

Partea "1"	**Partea "N"**	**Descriere**
Modele_Auto	Masini_Stoc	Un model poate avea multiple exemplare în stoc (aceeași mașină în culori diferite, etc.)
Clienti	Vanzari	Un client poate efectua mai multe achiziții în timp
Angajati	Vanzari	Un vânzător poate procesa multiple tranzacții
Vanzari	Plati	O vânzare are asociată o plată (relație 1:1 funcțională în context 1:N structural)
Clienti	Cereri_Achizitie	Un client poate trimite cereri pentru mai multe vehicule
Masini_Stoc	Cereri_Achizitie	O mașină poate primi cereri de la mai mulți clienți (procesate secvențial)
Clienti	Cos_Client	Un client poate avea multiple vehicule în coș
Masini_Stoc	Cos_Client	O mașină poate fi în coșul mai multor clienți simultan

Tabela 5.2: Relațiile 1:N din baza de date

5.1.3 Relații N:N (Many-to-Many)

Relațiile N:N sunt implementate prin tabele asociative.

Entitate 1	**Entitate 2**	**Tabelă Legătură**	**Descriere**
Clienti	Modele_Auto	Clienti_Modele_Auto	Un client poate fi interesat de mai multe modele; un model poate interesa mai mulți clienți

Tabela 5.3: Relația N:N din baza de date

5.2 Constrângeri de Integritate

5.2.1 Chei Primare

Fiecare tabelă are definită o cheie primară care garantează unicitatea înregistrărilor:

```
-- Cheie primara cu auto-increment
ID_Angajat INT IDENTITY(1,1) PRIMARY KEY
-- Cheie primara (identificator real unic)
VIN NVARCHAR(50) PRIMARY KEY
```

5.2.2 Chei Străine

Cheile străine asigură integritatea referențială între tabele, împiedicând înregistrări orfane:

```
-- Exemple de definitii FK
CONSTRAINT FK_Masini_Model
    FOREIGN KEY (ID_Model) REFERENCES Modele_Auto(ID_Model)

CONSTRAINT FK_Vanzari_Client
    FOREIGN KEY (ID_Client) REFERENCES Clienti(ID_Client)

CONSTRAINT FK_Vanzari_Angajat
    FOREIGN KEY (ID_Angajat) REFERENCES Angajati(ID_Angajat)

CONSTRAINT FK_Plati_Vanzare
    FOREIGN KEY (ID_Vanzare) REFERENCES Vanzari(ID_Vanzare)
```

5.2.3 Constrângeri NOT NULL și UNIQUE

```
-- Campuri obligatorii
Nume NVARCHAR(50) NOT NULL
Pret NUMERIC(10,2) NOT NULL
Data_Vanzare DATETIME NOT NULL
-- Unicitate pentru prevenirea duplicatelor
NumeUtilizator NVARCHAR(50) NOT NULL UNIQUE
```

5.2.4 Valori Implicite

```
-- Populare automata la inserare
Data_Cerere DATETIME DEFAULT GETDATE()
Status_Cerere NVARCHAR(20) DEFAULT 'In Asteptare'
Data_Adaugare DATETIME DEFAULT GETDATE()
```

5.2.5 Validări la Nivel de Aplicație

Pe lângă constrângerile SQL, aplicația implementează validări suplimentare:

- Verificarea existenței vânzărilor înainte de ștergerea unui angajat
- Blocarea ștergerii mașinilor care au fost vândute
- Prevenirea cererilor duplicate pentru același vehicul de la același client
- Blocarea plății duble pentru o vânzare deja achitată
- Verificarea disponibilității vehiculului înainte de creare cerere

Capitolul 6

Funcționarea Aplicației

6.1 Arhitectura Sistemului

Aplicația este construită pe arhitectura **MVC** (Model-View-Controller), adaptată pentru framework-ul Flask:

- **Model** – Baza de date SQL Server, accesată prin PyODBC
- **View** – Template-uri HTML procesate cu Jinja2
- **Controller** – Rutele Flask definite în `app.py`

6.2 Conectarea la Baza de Date

Conexiunea este gestionată printr-un modul dedicat care utilizează connection string-ul ODBC:

```
import pyodbc
CONNECTION_STRING = (
    r"DRIVER={ODBC Driver 11 for SQL Server};"
    r"SERVER=(local)\SQLEXPRESS;"
    r"DATABASE=Vanzare Masini Dealer Auto;"
    r"Trusted_Connection=yes;"
)
def get_connection():
    try:
        conn = pyodbc.connect(CONNECTION_STRING)
        return conn
    except pyodbc.Error as ex:
        error_details = ex.args[0]

        print(f"\nEROARE CRITICA DE CONEXIUNE LA BAZA DE DATE!")
        print(f"Server: LAPTOP-BIANCA\\SQLEXPRESS, Driver: ODBC
           Driver 11")
        print(f"Detaliu eroare SQL: {error_details}")

        return None
```

Fiecare rută deschide o conexiune nouă, execută operațiile necesare și închide conexiunea în blocul `finally` pentru a evita scurgerile de resurse.

6.3 Sistemul de Autentificare

Autentificarea se bazează pe sesiunile Flask. La login, se verifică credențialele în Tabela Utilizatori și se salvează în sesiune informațiile relevante:

- `session['logged_in']` – flag boolean
- `session['username']` – numele utilizatorului
- `session['rol']` – Admin / Vanzator / Client
- `session['id_angajat']` sau `session['id_client']` – identificatorul entității asociate

Fiecare rută protejată verifică existența sesiunii și rolul corespunzător.

6.4 Fluxul Principal de Achiziție

Procesul de cumpărare a unui vehicul parcurge următoarele etape:

1. **Navigare** – Clientul explorează catalogul pe mărci sau prin căutare
2. **Adăugare în coș** – Vehiculele de interes sunt salvate în TabelaCos_Client
3. **Trimitere cerere** – Clientul inițiază cererea; statusul mașinii devine "Rezervat"
4. **Preluare cerere** – Un vânzător preia cererea și contactează clientul
5. **Finalizare vânzare** – Se stabilește prețul final și se înregistrează în TabelaVanzari
6. **Actualizare stoc** – Statusul mașinii devine "Vandut"
7. **Plată** – Clientul efectuează plata din secțiunea Tranzacții

Capitolul 7

Interogări Simple (JOIN)

Interogările simple utilizează clauze JOIN pentru a combina date din două sau mai multe tabele. Sunt prezentate cele 6 interogări cele mai relevante din aplicație.

7.1 Vehicule Disponibile cu Specificații Complete

Scop: Afișarea catalogului de mașini disponibile pentru vânzare, cu toate detaliile tehnice preluate din tabela de modele.

Tabele: Masini_Stoc, Modele_Auto

```
SELECT
    MS.VIN,
    M.Marca,
    M.Nume_Model,
    M.Tip_Caroserie,
    M.Combustibil,
    M.Putere,
    M.Transmisie,
    MS.Culoare_Exterior,
    MS.Pret,
    MS.Status_Vanzare
FROM Masini_Stoc MS
INNER JOIN Modele_Auto M ON MS.ID_Model = M.ID_Model
WHERE MS.Status_Vanzare = 'Disponibil'
ORDER BY M.Marca, M.Nume_Model;
```

Explicație: INNER JOIN combină fiecare mașină din stoc cu specificațiile modelului său. Clauza WHERE filtrează doar vehiculele disponibile.

7.2 Istoric Complet al Vânzărilor

Scop: Raport detaliat cu toate vânzările, incluzând informații despre vehicul, client și vânzător.

Tabele: Vanzari, Masini_Stoc, Modele_Auto, Clienti, Angajati (5 tabele)

```
SELECT
    V.ID_Vanzare,
```

```
    V.Data_Vanzare,
    M.Marca + ' ' + M.Nume_Model AS Masina,
    MS.Culoare_Exterior,
    C.Nume + ' ' + C.Prenume AS Client,
    A.Nume + ' ' + A.Prenume AS Vanzator,
    V.Pret_Final
FROM Vanzari V
INNER JOIN Masini_Stoc MS ON V.VIN_Masina = MS.VIN
INNER JOIN Modele_Auto M ON MS.ID_Model = M.ID_Model
INNER JOIN Clienti C ON V.ID_Client = C.ID_Client
INNER JOIN Angajati A ON V.ID_Angajat = A.ID_Angajat
ORDER BY V.Data_Vanzare DESC;
```

Explicație: Interogarea îmbină 5 tabele printr-un lanț de JOIN-uri pentru a construi o imagine completă a fiecărei tranzacții.

7.3 Conținutul Coșului de Cumpărături

Scop: Afișarea vehiculelor salvate de un client în coș, cu specificații tehnice și disponibilitate.

Tabele: Cos_Client, Masini_Stoc, Modele_Auto

```
SELECT
    CC.ID_Cos,
    MS.VIN,
    M.Marca,
    M.Nume_Model,
    M.Tip_Caroserie,
    M.Combustibil,
    M.Putere,
    MS.Culoare_Exterior,
    MS.Pret,
    MS.Status_Vanzare,
    CC.Data_Adaugare
FROM Cos_Client CC
INNER JOIN Masini_Stoc MS ON CC.VIN_Masina = MS.VIN
INNER JOIN Modele_Auto M ON MS.ID_Model = M.ID_Model
WHERE CC.ID_Client = @id_client
ORDER BY CC.Data_Adaugare DESC;
```

Explicație: Parametrul `@id_client` filtrează coșul pentru utilizatorul curent. Se afișează și statusul pentru a semnala dacă vehiculul a fost rezervat între timp.

7.4 Cererile de Achiziție cu Status și Vânzător

Scop: Urmărirea cererilor trimise de un client, cu informații despre vehicul și vânzătorul care a preluat cererea.

Tabele: Cereri_Achizitie, Masini_Stoc, Modele_Auto, Angajati

```
SELECT
    CA.ID_Cerere,
    CA.VIN_Masina,
    M.Marca,
    M.Nume_Model,
    MS.Culoare_Exterior,
    MS.Pret,
    CA.Data_Cerere,
    CA.Status_Cerere,
    CA.Mesaj_Client,
    A.Nume + ' ' + A.Prenume AS Vanzator,
    CA.Data_Preluare
FROM Cereri_Achizitie CA
INNER JOIN Masini_Stoc MS ON CA.VIN_Masina = MS.VIN
INNER JOIN Modele_Auto M ON MS.ID_Model = M.ID_Model
LEFT JOIN Angajati A ON CA.ID_Vanzator_Preluat = A.ID_Angajat
WHERE CA.ID_Client = @id_client
ORDER BY CA.Data_Cerere DESC;
```

Explicație: Se folosește LEFT JOIN pentru Angajati deoarece cererile în așteptare nu au încă un vânzător asignat (valoare NULL).

7.5 Plățile Efectuate cu Detalii Vehicul

Scop: Istoricul plăților unui client, cu identificarea vehiculului pentru care s-a făcut plata.

Tabele: Plati, Vanzari, Masini_Stoc, Modele_Auto

```
SELECT
    P.ID_Plata,
    P.Data_Platii,
    P.Suma_Platita,
    P.Metoda_Plata,
    M.Marca + ' ' + M.Nume_Model AS Masina
FROM Plati P
INNER JOIN Vanzari V ON P.ID_Vanzare = V.ID_Vanzare
INNER JOIN Masini_Stoc MS ON V.VIN_Masina = MS.VIN
INNER JOIN Modele_Auto M ON MS.ID_Model = M.ID_Model
```

```
WHERE V.ID_Client = @id_client
ORDER BY P.Data_Platii DESC;
```

Explicație: Lanțul de JOIN-uri traversează de la plată la vânzare, apoi la mașină și model, pentru a afișa denumirea completă a vehiculului.

7.6 Lista Angajaților cu Conturile Asociate

Scop: Vizualizarea personalului și verificarea că fiecare angajat are cont de acces creat.

Tabele: Angajati, Utilizatori

```
SELECT
    A.ID_Angajat,
    A.Nume,
    A.Prenume,
    A.Data_Angajare,
    CASE
        WHEN U.NumeUtilizator IS NOT NULL THEN U.NumeUtilizator
        ELSE 'Fara cont'
    END AS ContUtilizator
FROM Angajati A
LEFT JOIN Utilizatori U ON A.ID_Angajat = U.ID_Angajat_FK
ORDER BY A.Nume ASC;
```

Explicație: LEFT JOIN asigură că toți angajații sunt afișați, chiar dacă nu au cont. Expresia CASE afișează "Fara cont" pentru valorile NULL.

Capitolul 8

Interogări Complexe (Subcereri)

Interogările complexe utilizează subcereri pentru a realiza analize avansate care nu pot fi exprimate doar prin JOIN-uri.

8.1 Angajați cu Performanțe Peste Medie

Scop: Identificarea vânzătorilor ale căror vânzări totale depășesc media generală a angajaților.

Tip subcerere: Subcerere în clauza HAVING, cu o subcerere imbricată

```
SELECT
    A.Nume + ' ' + A.Prenume AS Angajat,
    COUNT(V.ID_Vanzare) AS NrVanzari,
    SUM(V.Pret_Final) AS TotalVanzari
FROM Angajati A
INNER JOIN Vanzari V ON A.ID_Angajat = V.ID_Angajat
GROUP BY A.ID_Angajat, A.Nume, A.Prenume
HAVING SUM(V.Pret_Final) > (
    SELECT AVG(SubTotal)
    FROM (
        SELECT SUM(Pret_Final) AS SubTotal
        FROM Vanzari
        GROUP BY ID_Angajat
    ) AS TotaluriPerAngajat
)
ORDER BY TotalVanzari DESC;
```

Explicație:

1. Subcererea cea mai internă calculează totalul vânzărilor pentru fiecare angajat individual
2. Subcererea exterioară calculează media acestor totaluri
3. Clauza HAVING filtrează doar angajații care depășesc această medie

8.2 Vehicule Premium (Peste Media Mărcii)

Scop: Identificarea mașinilor cu preț superior mediei prețurilor pentru marca respectivă.

Tip subcerere: Două subcereri corelate – în SELECT și în WHERE

```
SELECT
    M.Marca,
    M.Nume_Model,
    MS.Pret,
    (SELECT AVG(MS2.Pret)
     FROM Masini_Stoc MS2
     INNER JOIN Modele_Auto M2 ON MS2.ID_Model = M2.ID_Model
     WHERE M2.Marca = M.Marca) AS PretMediuMarca
FROM Masini_Stoc MS
INNER JOIN Modele_Auto M ON MS.ID_Model = M.ID_Model
WHERE MS.Pret > (
    SELECT AVG(MS3.Pret)
    FROM Masini_Stoc MS3
    INNER JOIN Modele_Auto M3 ON MS3.ID_Model = M3.ID_Model
    WHERE M3.Marca = M.Marca
)
ORDER BY M.Marca, MS.Pret DESC;
```

Explicație: Ambele subcereri sunt *corelate* – depind de valoarea `M.Marca` din query-ul principal. Pentru fiecare rând, se recalculează media specifică mărcii respective.

8.3 Modele Favorite cu Verificare Disponibilitate

Scop: Afișarea modelelor din lista de interes a clientului, cu verificarea dacă există exemplare disponibile în stoc.

Tip subcerere: Subcerere corelată în clauza SELECT

```
SELECT
    M.ID_Model,
    M.Marca,
    M.Nume_Model,
    M.Tip_Caroserie,
    M.Combustibil,
    M.Putere,
    (SELECT TOP 1 MS.VIN
     FROM Masini_Stoc MS
     WHERE MS.ID_Model = M.ID_Model
       AND MS.Status_Vanzare = 'Disponibil') AS VIN_Disponibil
FROM Clienti_Modele_Auto CM
INNER JOIN Modele_Auto M ON CM.ID_Model = M.ID_Model
WHERE CM.ID_Client = @id_client
ORDER BY M.Marca, M.Nume_Model;
```

Explicație: Pentru fiecare model favorit, subcererea caută o mașină disponibilă în stoc. Dacă nu există, returnează NULL. Clientul poate astfel vedea instant ce modele pot fi achiziționate.

8.4 Modele Disponibile pentru Adăugare la Favorite

Scop: Popularea dropdown-ului cu modele care nu sunt încă în lista de interes a clientului.

Tip subcerere: Subcerere cu operatorul NOT IN

```
SELECT
    M.ID_Model,
    M.Marca + ' ' + M.Nume_Model AS NumeComplet
FROM Modele_Auto M
WHERE M.ID_Model NOT IN (
    SELECT ID_Model
    FROM Clienti_Modele_Auto
    WHERE ID_Client = @id_client
)
ORDER BY M.Marca, M.Nume_Model;
```

Explicație: Subcererea returnează toate ID-urile modelelor deja salvate de client. Operatorul NOT IN exclude aceste modele din rezultatul final, afișând doar opțiunile disponibile pentru adăugare.

Capitolul 9

Concluzii

9.1 Realizări

Proiectul **Veloce Motors** demonstrează implementarea completă a unui sistem de gestiune bazat pe o bază de date relațională. Principalele realizări includ:

- **Bază de date normalizată** cu 10 tabele (9 principale + 1 asociativă)
- **Relații complete:** 2 relații 1:1, 8 relații 1:N, 1 relație N:N
- **Constrângeri de integritate** la nivel de bază de date și aplicație
- **Interogări SQL diverse:** 6 interogări simple cu JOIN și 4 interogări complexe cu subcereri
- **Aplicație web funcțională** cu 3 roluri de utilizatori și flux complet de achiziție

9.2 Competențe Demonstrate

Prin realizarea acestui proiect au fost aplicate cunoștințe din domeniile:

- Analiza cerințelor și modelarea conceptuală a datelor
- Proiectarea schemelor relaționale normalizate
- Implementarea constrângerilor de integritate în SQL Server
- Scrierea interogărilor SQL (SELECT, INSERT, UPDATE, DELETE)
- Utilizarea JOIN-urilor pentru combinarea datelor din multiple tabele
- Aplicarea subcererilor pentru analize complexe
- Integrarea bazelor de date în aplicații web (Python/Flask)
- Gestionarea sesiunilor și autentificării utilizatorilor

Veloce Motors