TP1:

```sql
USE Parc;

CREATE TABLE IF NOT EXISTS Segment (
    indIP VARCHAR(45) PRIMARY KEY,
    nomSegment VARCHAR(30) NOT NULL,
    etage INT
) ENGINE=InnoDB;

CREATE TABLE IF NOT EXISTS Salle (
    nSalle VARCHAR(255) PRIMARY KEY,
    nomSalle VARCHAR(30) NOT NULL,
    nbPoste INT,
    indIP VARCHAR(45),
    FOREIGN KEY (indIP) REFERENCES Segment(indIP) ON DELETE SET NULL
) ENGINE=InnoDB;

CREATE TABLE IF NOT EXISTS Poste (
    nPoste VARCHAR(255) PRIMARY KEY,
    nomPoste VARCHAR(30) NOT NULL,
    indIP VARCHAR(45),
    ad INT,
    typePoste VARCHAR(30),
    nSalle VARCHAR(255),
    FOREIGN KEY (indIP) REFERENCES Segment(indIP) ON DELETE SET NULL,
    FOREIGN KEY (nSalle) REFERENCES Salle(nSalle) ON DELETE CASCADE
) ENGINE=InnoDB;

CREATE TABLE IF NOT EXISTS Logiciel (
    nLog VARCHAR(255) PRIMARY KEY,
    nomLog VARCHAR(30) NOT NULL,
    dateAch DATE,
    version VARCHAR(30),
    typeLog VARCHAR(30),
    prix DECIMAL(10,2) CHECK (prix >= 0)
) ENGINE=InnoDB;

CREATE TABLE IF NOT EXISTS Installer (
    numIns INT AUTO_INCREMENT PRIMARY KEY,
    nPoste VARCHAR(255),
    nLog VARCHAR(255),
    dateIns DATETIME DEFAULT CURRENT_TIMESTAMP,
    delai INT,
    FOREIGN KEY (nPoste) REFERENCES Poste(nPoste) ON DELETE CASCADE,
    FOREIGN KEY (nLog) REFERENCES Logiciel(nLog) ON DELETE CASCADE
) ENGINE=InnoDB;
```

```sql
CREATE TABLE IF NOT EXISTS Types (
    typeLP VARCHAR(30) NOT NULL PRIMARY KEY,
    nomType VARCHAR(50) NOT NULL
) ENGINE=InnoDB;

SHOW TABLES;
```

```sql
USE Parc;

DROP TABLE IF EXISTS Installer;
DROP TABLE IF EXISTS Logiciel;
DROP TABLE IF EXISTS Poste;
DROP TABLE IF EXISTS Salle;
DROP TABLE IF EXISTS Segment;
DROP TABLE IF EXISTS Types;

SHOW TABLES;

SELECT nSalle, indIP FROM Salle WHERE indIP NOT IN (SELECT indIP FROM
Segment);
SELECT nLog, typeLog FROM Logiciel WHERE typeLog NOT IN (SELECT typeLP FROM
Types);
```

TP2:

```sql
USE Parc;

INSERT INTO Segment(indIP, nomSegment)
VALUES ("130.120.80", "Brin RDC");
INSERT INTO Segment(indIP, nomSegment)
VALUES ("130.120.81", "Brin 1er étage");
INSERT INTO Segment(indIP, nomSegment)
VALUES ("130.120.82", "Brin 2e étage");
INSERT INTO Segment(indIP, nomSegment)
VALUES ("130.120.83", "Brin Extra");

INSERT INTO Salle (nSalle, nomSalle, nbPoste, indIP)
VALUES ("s01", "Salle 1", 3, "130.120.80");
INSERT INTO Salle (nSalle, nomSalle, nbPoste, indIP)
VALUES ("s02", "Salle 2", 2, "130.120.80");
INSERT INTO Salle (nSalle, nomSalle, nbPoste, indIP)
VALUES ("s03", "Salle 3", 2, "130.120.80");
INSERT INTO Salle (nSalle, nomSalle, nbPoste, indIP)
VALUES ("s11", "Salle 11", 2, "130.120.81");
INSERT INTO Salle (nSalle, nomSalle, nbPoste, indIP)
VALUES ("s12", "Salle 12", 1, "130.120.81");
```

```sql
INSERT INTO Salle (nSalle, nomSalle, nbPoste, indIP)
VALUES ("s21", "Salle 21", 2, "130.120.82");
INSERT INTO Salle (nSalle, nomSalle, nbPoste, indIP)
VALUES ("s22", "Salle 22", 0, "130.120.83");
INSERT INTO Salle (nSalle, nomSalle, nbPoste, indIP)
VALUES ("s23", "Salle 23", 0, "130.120.83");

INSERT INTO Poste (nPoste, nomPoste, indIP, ad, typePoste, nSalle)
Values ("p1", "Poste 1", "130.120.80", 1, "TX", "s01");
INSERT INTO Poste (nPoste, nomPoste, indIP, ad, typePoste, nSalle)
Values ("p2", "Poste 2", "130.120.80", 2, "UNIX", "s01");
INSERT INTO Poste (nPoste, nomPoste, indIP, ad, typePoste, nSalle)
Values ("p3", "Poste 3", "130.120.80", 3, "TX", "s01");
INSERT INTO Poste (nPoste, nomPoste, indIP, ad, typePoste, nSalle)
Values ("p4", "Poste 4", "130.120.80", 4, "PCWS", "s02");
INSERT INTO Poste (nPoste, nomPoste, indIP, ad, typePoste, nSalle)
Values ("p5", "Poste 5", "130.120.80", 5, "PCWS", "s02");
INSERT INTO Poste (nPoste, nomPoste, indIP, ad, typePoste, nSalle)
Values ("p6", "Poste 6", "130.120.80", 6, "UNIX", "s03");
INSERT INTO Poste (nPoste, nomPoste, indIP, ad, typePoste, nSalle)
Values ("p7", "Poste 7", "130.120.80", 7, "TX", "s03");
INSERT INTO Poste (nPoste, nomPoste, indIP, ad, typePoste, nSalle)
Values ("p8", "Poste 8", "130.120.81", 1, "UNIX", "s11");
INSERT INTO Poste (nPoste, nomPoste, indIP, ad, typePoste, nSalle)
Values ("p9", "Poste 9", "130.120.81", 2, "TX", "s11");
INSERT INTO Poste (nPoste, nomPoste, indIP, ad, typePoste, nSalle)
Values ("p10", "Poste 10", "130.120.81", 3, "UNIX", "s12");
INSERT INTO Poste (nPoste, nomPoste, indIP, ad, typePoste, nSalle)
Values ("p11", "Poste 11", "130.120.82", 1, "PCNT", "s21");
INSERT INTO Poste (nPoste, nomPoste, indIP, ad, typePoste, nSalle)
Values ("p12", "Poste 12", "130.120.82", 2, "PCWS", "s21");

INSERT INTO Logiciel (nLog, nomLog, dateAch, version, typeLog, prix)
VALUES ("log1", "Oracle 6", "1995-05-13", "6.2", "UNIX", 3000);
INSERT INTO Logiciel (nLog, nomLog, dateAch, version, typeLog, prix)
VALUES ("log2", "Oracle 8", "1999-09-15", "8i", "UNIX", 5600);
INSERT INTO Logiciel (nLog, nomLog, dateAch, version, typeLog, prix)
VALUES ("log3", "SQL Server", "1998-04-12", "7", "PCNT", 2700);
INSERT INTO Logiciel (nLog, nomLog, dateAch, version, typeLog, prix)
VALUES ("log4", "Front Page", "1997-06-03", "5", "PCWS", 500);
INSERT INTO Logiciel (nLog, nomLog, dateAch, version, typeLog, prix)
VALUES ("log5", "WinDev", "1997-05-12", "5", "PCWS", 750);
INSERT INTO Logiciel (nLog, nomLog, version, typeLog, prix)
VALUES ("log6", "SQL*Net", "2.0", "UNIX", 500);
INSERT INTO Logiciel (nLog, nomLog, dateAch, version, typeLog, prix)
VALUES ("log7", "I. I. S.", "2002-04-12", "2", "PCNT", 810);
INSERT INTO Logiciel (nLog, nomLog, dateAch, version, typeLog, prix)
VALUES ("log8", "DreamWeaver", "2003-09-21", "2.0", "BeOS", 1400);
```

```sql
INSERT INTO Types (typeLP, nomType)
VALUES ("TX", "Terminal X-Window");
INSERT INTO Types (typeLP, nomType)
VALUES ("UNIX", "Système Unix");
INSERT INTO Types (typeLP, nomType)
VALUES ("PCNT", "PC Windows NT");
INSERT INTO Types (typeLP, nomType)
VALUES ("PCWS", "PC Windows");
INSERT INTO Types (typeLP, nomType)
VALUES ("NC", "Network Computer");
INSERT INTO Types (typeLP, nomType) VALUES ("BeOS", "Système Be");

INSERT INTO Installer (NPOSTE, NLOG, DATEINS) VALUES
('p2', 'log1', '2003-05-15'),
('p2', 'log2', '2003-09-17'),
('p6', 'log6', '2003-05-20'),
('p6', 'log5', '2003-05-20'),
('p8', 'log6', '2003-05-19'),
('p8', 'log7', '2003-05-20'),
('p11', 'log4', '2003-04-20'),
('p12', 'log3', '2003-04-20'),
('p11', 'log7', '2003-04-10'),
('p7', 'log7', '2002-04-01');
USE Parc;

SET SQL_SAFE_UPDATES = 0;

SELECT * FROM Segment;
SELECT nLog, typeLog, prix FROM Logiciel;

UPDATE Segment SET etage = 0 WHERE indIP = "130.120.80";
UPDATE Segment SET etage = 1 WHERE indIP = "130.120.81";
UPDATE Segment SET etage = 2 WHERE indIP = "130.120.82";
UPDATE Segment SET etage = 3 WHERE indIP = "130.120.83";

UPDATE Logiciel SET prix = prix * 0.9 WHERE typeLog = 'PCNT';

SELECT * FROM Segment;
SELECT nLog, typeLog, prix FROM Logiciel;

SET SQL_SAFE_UPDATES = 1;
```

TP3

```sql
ALTER TABLE Segment ADD nbSalle TINYINT(2) DEFAULT 0;
ALTER TABLE Segment ADD nbPoste TINYINT(2) DEFAULT 0;
ALTER TABLE Logiciel ADD nbInstall TINYINT(2) DEFAULT 0;
ALTER TABLE Poste ADD nbLog TINYINT(2) DEFAULT 0;

ALTER TABLE Salle MODIFY COLUMN nomSalle VARCHAR(30);
ALTER TABLE Segment MODIFY COLUMN nomSegment VARCHAR(15);


ALTER TABLE Installer ADD CONSTRAINT unique_poste_logiciel UNIQUE (nPoste,
nLog);

ALTER TABLE Installer ADD CONSTRAINT fk_installer_poste FOREIGN KEY (nPoste)
REFERENCES Poste(nPoste);
ALTER TABLE Installer ADD CONSTRAINT fk_installer_logiciel FOREIGN KEY (nLog)
REFERENCES Logiciel(nLog);


SELECT nSalle, indIP FROM Salle WHERE indIP NOT IN (SELECT indIP FROM
Segment);
SELECT nLog, typeLog FROM Logiciel WHERE typeLog NOT IN (SELECT typeLP FROM
Types);

DELETE FROM Salle WHERE indIP NOT IN (SELECT indIP FROM Segment);

INSERT INTO Types (typeLP, nomType) VALUES ('BeOS', 'système Be');

ALTER TABLE Salle ADD CONSTRAINT fk_salle_segment FOREIGN KEY (indIP)
REFERENCES Segment(indIP);
ALTER TABLE Logiciel ADD CONSTRAINT fk_logiciel_type FOREIGN KEY (typeLog)
REFERENCES Types(typeLP);

DESCRIBE Salle;
DESCRIBE Segment;
DESCRIBE Logiciel;
DESCRIBE Types;

SELECT * FROM Salle;
SELECT * FROM Segment;
SELECT * FROM Logiciel;
SELECT * FROM Types;

SELECT nSalle, indIP FROM Salle WHERE indIP NOT IN (SELECT indIP FROM
Segment);
SELECT nLog, typeLog FROM Logiciel WHERE typeLog NOT IN (SELECT typeLP FROM
Types);
```

TP4:

```sql
SELECT * FROM employes;

SELECT nom, prenom FROM employes;
SELECT nom, prenom, salaire FROM employes WHERE salaire > 2000;

SELECT * FROM employes WHERE service = (
    SELECT id_service FROM services WHERE nom = 'commercial'
);

SELECT * FROM employes WHERE date_embauche > '2010-01-01';

SELECT * FROM employes ORDER BY salaire ASC;

SELECT * FROM employes ORDER BY nom ASC;

SELECT * FROM employes ORDER BY salaire DESC, nom ASC;

SELECT DISTINCT salaire FROM employes;

SELECT * FROM employes WHERE salaire BETWEEN 1500 AND 3000;

SELECT * FROM employes WHERE salaire IN (2000, 2500, 3000);

SELECT * FROM employes WHERE nom LIKE 'B%';

SELECT * FROM employes WHERE prenom LIKE '%a%';

SELECT AVG(salaire) AS salaire_moyen FROM employes;

SELECT service, COUNT(*) AS nb_employes FROM employes GROUP BY service;

SELECT MAX(salaire) AS salaire_max, MIN(salaire) AS salaire_min FROM employes;

SELECT employes.nom, services.nom AS service
FROM employes
JOIN services ON employes.service = services.id_service;

SELECT employes.nom AS Nom, employes.prenom AS Prénom, services.nom AS Service
FROM employes
JOIN services ON employes.service = services.id_service;

SELECT * FROM employes
WHERE salaire > (SELECT AVG(salaire) FROM employes);
```

TP5:

```sql
-- ========================
-- 1. Création des tables
-- ========================

CREATE TABLE TypeProduit (
    id_type SERIAL PRIMARY KEY,
    libelle VARCHAR(30) NOT NULL
);

CREATE TABLE Produit (
    id_produit SERIAL PRIMARY KEY,
    nom VARCHAR(50) NOT NULL,
    id_type INT NOT NULL REFERENCES TypeProduit(id_type)
);

CREATE TABLE Vente (
    id_vente SERIAL PRIMARY KEY,
    date_vente DATE NOT NULL,
    id_produit INT NOT NULL REFERENCES Produit(id_produit),
    quantite_kg DECIMAL(7,2) NOT NULL,
    prix_kg DECIMAL(7,2) NOT NULL
);

CREATE TABLE Client (
    id_client SERIAL PRIMARY KEY,
    nom VARCHAR(50) NOT NULL,
    prenom VARCHAR(50),
    adresse VARCHAR(100),
    telephone VARCHAR(20),
    email VARCHAR(100)
);

CREATE TABLE Materiel (
    id_materiel SERIAL PRIMARY KEY,
    id_client INT NOT NULL REFERENCES Client(id_client),
    type_materiel VARCHAR(50) NOT NULL,
    marque VARCHAR(50),
    modele VARCHAR(50)
);

CREATE TABLE TypeIntervention (
    id_type_intervention SERIAL PRIMARY KEY,
    libelle VARCHAR(50) NOT NULL,
    prix_horaire DECIMAL(7,2) NOT NULL
);
```

```sql
CREATE TABLE Intervention (
    id_intervention SERIAL PRIMARY KEY,
    id_client INT NOT NULL REFERENCES Client(id_client),
    id_materiel INT NOT NULL REFERENCES Materiel(id_materiel),
    id_type_intervention INT NOT NULL REFERENCES
TypeIntervention(id_type_intervention),
    date_intervention DATE NOT NULL,
    duree_heures DECIMAL(4,2) NOT NULL,
    commentaire TEXT
);

CREATE TABLE Composant (
    id_composant SERIAL PRIMARY KEY,
    nom VARCHAR(50) NOT NULL,
    prix_unitaire DECIMAL(7,2) NOT NULL
);

CREATE TABLE VenteComposant (
    id_vente_composant SERIAL PRIMARY KEY,
    id_intervention INT NOT NULL REFERENCES Intervention(id_intervention),
    id_composant INT NOT NULL REFERENCES Composant(id_composant),
    quantite INT NOT NULL
);

-- =========================
-- 2. Exemples de contenu
-- =========================

INSERT INTO TypeProduit (libelle) VALUES
('Animal'), ('Légume'), ('Fruit');

INSERT INTO Produit (nom, id_type) VALUES
('Lapin', 1), ('Poulet', 1), ('Dinde', 1), ('Veau', 1), ('Cochon', 1),
('Chou', 2), ('Pomme de terre', 2), ('Carotte', 2),
('Fraise', 3), ('Poire', 3), ('Pomme', 3);

INSERT INTO Vente (date_vente, id_produit, quantite_kg, prix_kg) VALUES
('2025-05-01', 1, 5.2, 8.00),
('2025-05-01', 6, 10.0, 2.00),
('2025-05-02', 9, 3.5, 6.00);

INSERT INTO Client (nom, prenom, adresse, telephone, email) VALUES
('Dupont', 'Jean', '1 rue de la Paix', '0600000000', 'jean.dupont@email.com'),
('Martin', 'Sophie', '2 avenue des Lilas', '0611111111',
'sophie.martin@email.com');


INSERT INTO Materiel (id_client, type_materiel, marque, modele) VALUES
(1, 'PC', 'Dell', 'Inspiron 15'),
```

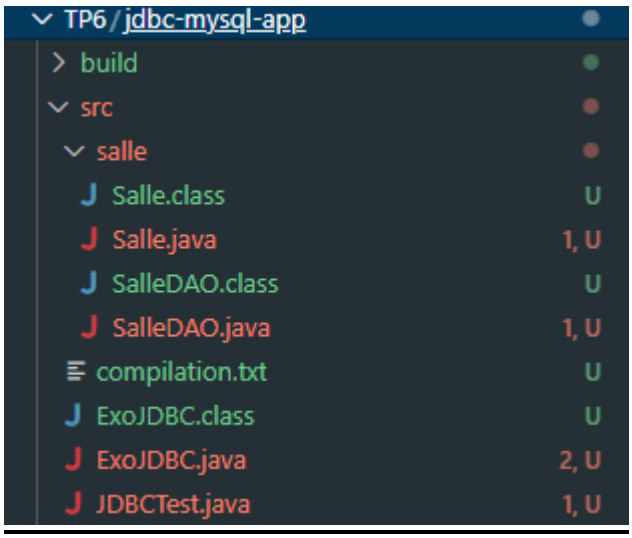
```sql
(2, 'Imprimante', 'HP', 'DeskJet 2700');


INSERT INTO TypeIntervention (libelle, prix_horaire) VALUES
('Dépannage simple', 30.00),
('Réparation complexe', 50.00);

INSERT INTO Intervention (id_client, id_materiel, id_type_intervention,
date_intervention, duree_heures, commentaire) VALUES
(1, 1, 1, '2025-05-10', 1.5, 'Nettoyage virus'),
(2, 2, 2, '2025-05-12', 2.0, 'Remplacement tête impression');

INSERT INTO Composant (nom, prix_unitaire) VALUES
('Barrette RAM 8Go', 40.00),
('Tête impression HP', 35.00);


INSERT INTO VenteComposant (id_intervention, id_composant, quantite) VALUES
(2, 2, 1);

-- =========================
-- 3. Requêtes exemples
-- =========================

SELECT tp.libelle AS type_produit, SUM(v.quantite_kg * v.prix_kg) AS
total_vente
FROM Vente v
JOIN Produit p ON v.id_produit = p.id_produit
JOIN TypeProduit tp ON p.id_type = tp.id_type
WHERE EXTRACT(MONTH FROM v.date_vente) = 5 AND EXTRACT(YEAR FROM v.date_vente)
= 2025
GROUP BY tp.libelle;

-- Récapitulatif mensuel par produit
SELECT p.nom AS produit, SUM(v.quantite_kg * v.prix_kg) AS total_vente
FROM Vente v
JOIN Produit p ON v.id_produit = p.id_produit
WHERE EXTRACT(MONTH FROM v.date_vente) = 5 AND EXTRACT(YEAR FROM v.date_vente)
= 2025
GROUP BY p.nom;

-- =========================
-- 4. Explications MCD/MLD
-- =========================
-- Entités : TypeProduit, Produit, Vente
-- Associations : Un Produit appartient à un TypeProduit, une Vente concerne
un Produit
-- Attributs : nom, libelle, quantite_kg, prix_kg, date_vente
```

```sql
-- ========================
-- Exercice 2 : Auto-entrepreneur informatique
-- ========================
-- 1. Création des tables

CREATE TABLE Client (
    id_client SERIAL PRIMARY KEY,
    nom VARCHAR(50) NOT NULL,
    prenom VARCHAR(50),
    adresse VARCHAR(100),
    telephone VARCHAR(20),
    email VARCHAR(100)
);

CREATE TABLE Materiel (
    id_materiel SERIAL PRIMARY KEY,
    id_client INT NOT NULL REFERENCES Client(id_client),
    type_materiel VARCHAR(50) NOT NULL,
    marque VARCHAR(50),
    modele VARCHAR(50)
);

CREATE TABLE TypeIntervention (
    id_type_intervention SERIAL PRIMARY KEY,
    libelle VARCHAR(50) NOT NULL,
    prix_horaire DECIMAL(7,2) NOT NULL
);

CREATE TABLE Intervention (
    id_intervention SERIAL PRIMARY KEY,
    id_client INT NOT NULL REFERENCES Client(id_client),
    id_materiel INT NOT NULL REFERENCES Materiel(id_materiel),
    id_type_intervention INT NOT NULL REFERENCES
TypeIntervention(id_type_intervention),
    date_intervention DATE NOT NULL,
    duree_heures DECIMAL(4,2) NOT NULL,
    commentaire TEXT
);

CREATE TABLE Composant (
    id_composant SERIAL PRIMARY KEY,
    nom VARCHAR(50) NOT NULL,
    prix_unitaire DECIMAL(7,2) NOT NULL
);

CREATE TABLE VenteComposant (
    id_vente_composant SERIAL PRIMARY KEY,
    id_intervention INT NOT NULL REFERENCES Intervention(id_intervention),
```

```sql
    id_composant INT NOT NULL REFERENCES Composant(id_composant),
    quantite INT NOT NULL
);

-- 2. Exemples de contenu

INSERT INTO Client (nom, prenom, adresse, telephone, email) VALUES
('Dupont', 'Jean', '1 rue de la Paix', '0600000000', 'jean.dupont@email.com'),
('Martin', 'Sophie', '2 avenue des Lilas', '0611111111',
'sophie.martin@email.com');

INSERT INTO Materiel (id_client, type_materiel, marque, modele) VALUES
(1, 'PC', 'Dell', 'Inspiron 15'),
(2, 'Imprimante', 'HP', 'DeskJet 2700');

INSERT INTO TypeIntervention (libelle, prix_horaire) VALUES
('Dépannage simple', 30.00),
('Réparation complexe', 50.00);

INSERT INTO Intervention (id_client, id_materiel, id_type_intervention,
date_intervention, duree_heures, commentaire) VALUES
(1, 1, 1, '2025-05-10', 1.5, 'Nettoyage virus'),
(2, 2, 2, '2025-05-12', 2.0, 'Remplacement tête impression');

INSERT INTO Composant (nom, prix_unitaire) VALUES
('Barrette RAM 8Go', 40.00),
('Tête impression HP', 35.00);

INSERT INTO VenteComposant (id_intervention, id_composant, quantite) VALUES
(2, 2, 1);

-- 3. Explications MCD/MLD
-- Entités : Client, Materiel, TypeIntervention, Intervention, Composant,
VenteComposant
-- Associations :
--    - Un Client possède plusieurs Matériels
--    - Une Intervention concerne un Client, un Matériel, un Type
d'Intervention
--    - Une Intervention peut donner lieu à la vente de plusieurs Composants
-- Attributs : nom, prenom, adresse, type_materiel, marque, modele, libelle,
prix_horaire, date_intervention, duree_heures, commentaire, prix_unitaire,
quantite
```

TP6 :

Arborescence :



Salle.java :

```java
package salle;

public class Salle {
    private String nSalle;
    private String nomSalle;
    private int nbPoste;
    private String indIP;

    public Salle(String nSalle, String nomSalle, int nbPoste, String indIP) {
        this.nSalle = nSalle;
        this.nomSalle = nomSalle;
        this.nbPoste = nbPoste;
        this.indIP = indIP;
    }

    public String getNSalle() { return nSalle; }
    public void setNSalle(String nSalle) { this.nSalle = nSalle; }

    public String getNomSalle() { return nomSalle; }
    public void setNomSalle(String nomSalle) { this.nomSalle = nomSalle; }

    public int getNbPoste() { return nbPoste; }
    public void setNbPoste(int nbPoste) { this.nbPoste = nbPoste; }

    public String getIndIP() { return indIP; }
    public void setIndIP(String indIP) { this.indIP = indIP; }

    @Override
```

```
    public String toString() {
        return "Salle{" +
                "nSalle='" + nSalle + '\'' +
                ", nomSalle='" + nomSalle + '\'' +
                ", nbPoste=" + nbPoste +
                ", indIP='" + indIP + '\'' +
                '}';
    }
}
```

Salle.DAO :

```
package salle;

import java.sql.*;
import java.util.ArrayList;

public class SalleDAO {
    private Connection connection;

    public SalleDAO(Connection connection) {
        this.connection = connection;
    }

    public ArrayList<Salle> getSalles() {
        ArrayList<Salle> salles = new ArrayList<>();
        String query = "SELECT * FROM Salle";
        try (Statement stmt = connection.createStatement();
             ResultSet rs = stmt.executeQuery(query)) {
            while (rs.next()) {
                Salle salle = new Salle(
                    rs.getString("nSalle"),
                    rs.getString("nomSalle"),
                    rs.getInt("nbPoste"),
                    rs.getString("indIP")
                );
                salles.add(salle);
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return salles;
    }

    public void addSalle(Salle salle) {
        String query = "INSERT INTO Salle (nSalle, nomSalle, nbPoste, indIP)
VALUES (?, ?, ?, ?)";
        try (PreparedStatement pstmt = connection.prepareStatement(query)) {
```

```
            pstmt.setString(1, salle.getNSalle());
            pstmt.setString(2, salle.getNomSalle());
            pstmt.setInt(3, salle.getNbPoste());
            pstmt.setString(4, salle.getIndIP());
            pstmt.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void updateSalle(Salle salle) {
        String query = "UPDATE Salle SET nomSalle = ?, nbPoste = ?, indIP = ?
WHERE nSalle = ?";
        try (PreparedStatement pstmt = connection.prepareStatement(query)) {
            pstmt.setString(1, salle.getNomSalle());
            pstmt.setInt(2, salle.getNbPoste());
            pstmt.setString(3, salle.getIndIP());
            pstmt.setString(4, salle.getNSalle());
            pstmt.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void deleteSalle(String nSalle) {
        String query = "DELETE FROM Salle WHERE nSalle = ?";
        try (PreparedStatement pstmt = connection.prepareStatement(query)) {
            pstmt.setString(1, nSalle);
            pstmt.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void deleteSalleWithCursor(String nSalle) {
        String query = "SELECT * FROM Salle";
        try (Statement stmt = connection.createStatement(
                ResultSet.TYPE_SCROLL_SENSITIVE,
                ResultSet.CONCUR_UPDATABLE)) {
            ResultSet rs = stmt.executeQuery(query);
            while (rs.next()) {
                if (rs.getString("nSalle").equals(nSalle)) {
                    rs.deleteRow();
                    System.out.println("Salle supprimée (nSalle=" + nSalle +
")");
                    return;
                }
            }
```

```
            System.out.println("Salle non trouvée (nSalle=" + nSalle + ")");
        } catch (SQLException e) {
            if (e.getErrorCode() == 1451) {
                System.out.println("Erreur 1451 : contrainte référentielle
!");
            } else {
                e.printStackTrace();
            }
        }
    }
}
```

ExoJDBC.java

```
package src;

import java.awt.*;
import java.awt.event.*;
import java.sql.*;
import salle.Salle;
import salle.SalleDAO;

public class ExoJDBC extends Panel implements ActionListener {
    TextField nomDriver;
    TextField urlConnection;
    TextField nomLogin;
    TextField motPasse;
    Button boutonConnection;
    TextField requeteSQL;
    List resultatRequete;
    Button boutonExecuter;

    // 1. Ajout de la variable Connection
    private Connection connection = null;

    public ExoJDBC() {
        Panel haut;
        Panel bas;
        haut = new Panel();
        bas = new Panel();
        boutonConnection = new Button("Connection");
        boutonConnection.addActionListener(this);
        boutonExecuter = new Button("Execution");
        boutonExecuter.addActionListener(this);
        Panel p1 = new Panel();
        p1.setLayout(new GridLayout(4, 2));
        p1.add(new Label("Driver :"));
        p1.add(nomDriver = new TextField(32));
```

```
        p1.add(new Label("URL jdbc :"));
        p1.add(urlConnection = new TextField(32));
        p1.add(new Label("login :"));
        p1.add(nomLogin = new TextField(32));
        p1.add(new Label("password :"));
        p1.add(motPasse = new TextField(32));
        haut.setLayout(new BorderLayout());
        haut.add(p1, BorderLayout.NORTH);
        haut.add(boutonConnection, BorderLayout.SOUTH);
        Panel p2 = new Panel();
        p2.setLayout(new BorderLayout());
        p2.add(new Label("requete"), BorderLayout.WEST);
        p2.add(requeteSQL = new TextField(32), BorderLayout.CENTER);
        Panel p3 = new Panel();
        p3.setLayout(new BorderLayout());
        p3.add(p2, BorderLayout.NORTH);
        p3.add(boutonExecuter, BorderLayout.SOUTH);
        bas.setLayout(new BorderLayout());
        bas.add(p3, BorderLayout.NORTH);
        bas.add(resultatRequete = new List(20));
        setLayout(new BorderLayout());
        add(haut, BorderLayout.NORTH);
        add(bas, BorderLayout.CENTER);
        nomDriver.setText("com.mysql.cj.jdbc.Driver");
        urlConnection.setText("jdbc:mysql://localhost:3306/Parc");
        nomLogin.setText("root");
        motPasse.setText("316MIQq~+X"); // Mets ton mot de passe si besoin
    }

    public void actionPerformed(ActionEvent evt) {
        if (evt.getSource() == boutonConnection) {
            // 2. Connexion à la base
            try {
                String driver = nomDriver.getText().trim();
                String url = urlConnection.getText().trim();
                String user = nomLogin.getText().trim();
                String pass = motPasse.getText().trim();
                Class.forName(driver);
                connection = DriverManager.getConnection(url, user, pass);
                resultatRequete.add("Connexion réussie !");
            } catch (Exception e) {
                resultatRequete.add("Erreur connexion : " + e.getMessage());
            }
        } else if (evt.getSource() == boutonExecuter) {
            // 3. Exécution de la requête SQL
            if (connection == null) {
                resultatRequete.add("Pas de connexion à la base !");
                return;
```

```
            }
            String req = requeteSQL.getText().trim();
            try (Statement stmt = connection.createStatement()) {
                boolean hasResultSet = stmt.execute(req);
                resultatRequete.removeAll();
                if (hasResultSet) {
                    ResultSet rs = stmt.getResultSet();
                    ResultSetMetaData meta = rs.getMetaData();
                    int colCount = meta.getColumnCount();
                    // Affiche les noms de colonnes
                    StringBuilder header = new StringBuilder();
                    for (int i = 1; i <= colCount; i++) {
                        header.append(meta.getColumnName(i)).append("\t");
                    }
                    resultatRequete.add(header.toString());
                    // Affiche les lignes
                    while (rs.next()) {
                        StringBuilder row = new StringBuilder();
                        for (int i = 1; i <= colCount; i++) {
                            row.append(rs.getString(i)).append("\t");
                        }
                        resultatRequete.add(row.toString());
                    }
                } else {
                    int count = stmt.getUpdateCount();
                    resultatRequete.add(count + " ligne(s) affectée(s).");
                }
            } catch (SQLException e) {
                resultatRequete.add("Erreur requête : " + e.getMessage());
            }
        }
    }

    public static void main(String[] args) {
        Frame f = new Frame("ExoJDBC - Consultation Base Parc Informatique");
        ExoJDBC panel = new ExoJDBC();
        f.add(panel);
        f.setSize(800, 600);
        f.setVisible(true);
        f.addWindowListener(new java.awt.event.WindowAdapter() {
            public void windowClosing(java.awt.event.WindowEvent e) {
                System.exit(0);
            }
        });
    }
}
```

JDBCTest.java

```
// Source de l'interface graphique
import java.awt.*;
import java.awt.event.*;
import java.sql.*;
public class JDBCTest extends Panel
implements ActionListener
    {
    TextField nomDriver;
    TextField urlConnection;
    TextField nomLogin;
    TextField motPasse;
    Button boutonConnection;
    TextField requeteSQL;
    List resultatRequete;
    Button boutonExecuter;
    public JDBCTest()
        {
            Panel haut;
            Panel bas;
            haut = new Panel();
            bas = new Panel();
            boutonConnection = new Button("Connection");
            boutonConnection.addActionListener(this);
            boutonExecuter = new Button("Execution");
            boutonExecuter.addActionListener(this);
            Panel p1 = new Panel();
            p1.setLayout(new GridLayout(4, 2));
            p1.add(new Label("Driver :"));
            p1.add(nomDriver = new TextField(32));
            p1.add(new Label("URL jdbc :"));
            p1.add(urlConnection = new TextField(32));
            p1.add(new Label("login :"));
            p1.add(nomLogin = new TextField(32));
            p1.add(new Label("password :"));
            p1.add(motPasse = new TextField(32));
            haut.setLayout(new BorderLayout());
            haut.add(p1, BorderLayout.NORTH);
            haut.add(boutonConnection, BorderLayout.SOUTH);
            Panel p2 = new Panel();
            p2.setLayout(new BorderLayout());
            p2.add(new Label("requete"), BorderLayout.WEST);
            p2.add(requeteSQL = new TextField(32), BorderLayout.CENTER);
            Panel p3 = new Panel();
            p3.setLayout(new BorderLayout());
            p3.add(p2, BorderLayout.NORTH);
            p3.add(boutonExecuter, BorderLayout.SOUTH);
            bas.setLayout(new BorderLayout());
```

```
            bas.add(p3, BorderLayout.NORTH);
            bas.add(resultatRequete = new List(20));
            setLayout(new BorderLayout());
            add(haut, BorderLayout.NORTH);
            add(bas, BorderLayout.CENTER);
        }
    public void actionPerformed(ActionEvent evt)
        {
            //System.out.prinln("Not implemented !!");
            resultatRequete.add("au boulot les gars !!!");
        }
    public static void main(String[] arg)
        {
        JDBCTest test;
        Frame f = new Frame();
        f.setSize(500, 400);
        test = new JDBCTest( );
        f.add(test, BorderLayout.CENTER);
        f.addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        } );
        f.setVisible(true);
        }
    }
```

Compilé avec la commande : javac -cp lib/mysql-connector-java-9.3.0.jar -d build $(find src -name "*.java")

Lancé avec la commande : java -cp build:/usr/share/java/mysql-connector-j-9.3.0.jar src.ExoJDBC

TP7 :

En 3 fichiers :

- **Exo6_1_Extraction_Donnee_stockees.sql**
- **Exo6_2_Variables_de_session.sql**
- **Exo6_3_Transaction.sql**

Exo6_1_Extraction_Donnee_stockees.sql

```
DROP PROCEDURE IF EXISTS Exo6_1;
DELIMITER //
CREATE PROCEDURE Exo6_1()
```

```
BEGIN
  DECLARE v_numSalle VARCHAR(10);
  DECLARE v_numPoste VARCHAR(10);
  DECLARE v_nomLog   VARCHAR(100);
  DECLARE v_dateInst DATETIME;

  SELECT p.nSalle
    INTO v_numSalle
    FROM Installer i
    JOIN Poste p ON i.nPoste = p.nPoste
    ORDER BY i.dateIns DESC
    LIMIT 1;

  SELECT CONCAT(
    '+---------------------------------------+\n',
    '| Resultat1 exo 1                       |\n',
    '+---------------------------------------+\n',
    '| Derniere installation en salle : ', v_numSalle, ' |\n',
    '+---------------------------------------+'
  ) AS Resultat1;

  SELECT i.nPoste, l.nomLog, i.dateIns
    INTO v_numPoste, v_nomLog, v_dateInst
    FROM Installer i
    JOIN Logiciel l ON i.nLog = l.nLog
    ORDER BY i.dateIns DESC
    LIMIT 1;

  SELECT CONCAT(
    '+-------------------------------------------------------------------
+\n',
    '| Resultat2 exo
1                                                  |\n',
    '+-------------------------------------------------------------------
+\n',
    '| Poste : ', v_numPoste,
    ' Logiciel : ', v_nomLog,
    ' en date du ', DATE_FORMAT(v_dateInst, '%Y-%m-%d %H:%i:%s'), ' |\n',
    '+-------------------------------------------------------------------+'
  ) AS Resultat2;
END //
DELIMITER ;
```

Exo6_2_Variables_de_session.sql

```
DROP PROCEDURE IF EXISTS Exo6_2;
DELIMITER //
CREATE PROCEDURE Exo6_2()
```

```
BEGIN
  DECLARE v_countPostes INT DEFAULT 0;
  DECLARE v_countInst   INT DEFAULT 0;

  SELECT COUNT(*)
    INTO v_countPostes
    FROM Poste
    WHERE nSalle = @numSalle;

  SELECT COUNT(*)
    INTO v_countInst
    FROM Installer i
    JOIN Poste p ON i.nPoste = p.nPoste
    WHERE p.typePoste = @typePoste;

  SELECT CONCAT(
    '-----------------------------------------------------\n',
    '| Resultat exo2                                      |\n',
    '-----------------------------------------------------\n',
    '| ',
    v_countPostes, ' poste(s) installé(s) en salle ', @numSalle,
    ', ', v_countInst, ' installation(s) de type ', @typePoste, ' |\n',
    '-----------------------------------------------------'
  ) AS ResultatExo2;
END //
DELIMITER ;
```

Exo6_3_Transaction.sql

```
DROP PROCEDURE IF EXISTS Exo6_3;
DELIMITER //
CREATE PROCEDURE Exo6_3()
BEGIN
  DECLARE v_dateAchat    DATETIME;
  DECLARE v_dateInstall  DATETIME;
  DECLARE v_resultSleep  INT;

  START TRANSACTION;

  INSERT INTO Logiciel (
        nLog,
        nomLog,
        version,
        typeLog,
        Prix,
        dateAch
      )
  VALUES (
```

```
        @numLog,
        @nomLog,
        @versionLog,
        @typeLog,
        @prixLog,
        NOW()
      );
  SET v_dateAchat = NOW();

  SELECT 'Logiciel insere dans la base' AS message1;
  SELECT CONCAT('Date achat : ', DATE_FORMAT(v_dateAchat, '%Y-%m-%d
%H:%i:%s')) AS message2;

  SELECT SLEEP(5) INTO v_resultSleep;

  INSERT INTO Installer (
        nPoste,
        nLog,
        dateIns
      )
  VALUES (
        @posteCode,
        @numLog,
        NOW()
      );
  SET v_dateInstall = NOW();

  UPDATE Installer
    SET delai = TIMEDIFF(v_dateInstall, v_dateAchat)
    WHERE nPoste = @posteCode
      AND nLog = @numLog
      AND dateIns = v_dateInstall;

  SELECT CONCAT('Date installation : ', DATE_FORMAT(v_dateInstall, '%Y-%m-%d
%H:%i:%s')) AS message3;
  SELECT 'Logiciel installe sur le poste' AS message4;

  COMMIT;
END //
DELIMITER ;
```

Lancement du program:

Création des procédures:

```
SOURCE Exo6_1_Extraction_Donnee_stockees.sql;
SOURCE Exo6_2_Variables_de_Session.sql;
SOURCE Exo6_3_Transaction.sql;
```

Lancement du program :

```
CALL Exo6_1();
SET @numSalle = 's01';
SET @typePoste = 'UNIX';
CALL Exo6_2();
SET @numLog     = 'log15';
SET @nomLog     = 'MySQL Query';
SET @versionLog = '1.4';
SET @typeLog    = 'PCWS';
SET @prixLog    = 95;
SET @posteCode  = 'p7';
CALL Exo6_3();
```