# TP1 – Création de la base de données

CreParc.sql :

CREATE TABLE Segment (

indIP VARCHAR(11) PRIMARY KEY,

nomSegment VARCHAR(20) NOT NULL,

etage VARCHAR(30)

);

CREATE TABLE Salle (

nSalle VARCHAR(7) PRIMARY KEY,

nomSalle VARCHAR(20) NOT NULL,

nbPoste TINYINT,

indIP VARCHAR(11),

FOREIGN KEY (indIP) REFERENCES Semgent(indIP)

);

CREATE TABLE Types (

typeLP VARCHAR(9) PRIMARY KEY,

nomType VARCHAR(20)

);

CREATE TABLE Poste (

nPoste VARCHAR(7) PRIMARY KEY,

nomPoste VARCHAR(20) NOT NULL,

indIP VARCHAR(11),

ad TINYINT CHECK (ad BETWEEN 0 and 255),

typePoste VARCHAR(9),

nSalle VARCHAR(7),

FOREIGN KEY (nSalle) REFERENCES Salle(nSalle),

FOREIGN KEY (typePoste) REFERENCES Types(typeLP)

);

CREATE TABLE Logiciel (

nLog VARCHAR(5) PRIMARY KEY,

nomLog VARCHAR(20),

dateAch DATETIME,

version VARCHAR(7),

typeLog VARCHAR(9),

prix DECIMAL(6,2) CHECK (prix >= 0)

);

CREATE TABLE Installer (

nPoste VARCHAR(7),

nLog VARCHAR(5),

numIns INTEGER PRIMARY KEY,

dateIns DATE,

delai SMALLINT,

FOREIGN KEY (nPoste) REFERENCES Poste(nPoste),

FOREIGN KEY (nLog) REFERENCES Logiciel(nLog)

);

descParc.sql :

DESCRIBE Segment;

DESCRIBE Salle;

DESCRIBE Poste,

DESCRIBE Logiciel;

DESCRIBE Installer;

DESCRIBE Types;

dropParc.sql :

DROP TABLE Installer;

DROP TABLE Logiciel;

DROP TABLE Poste;

DROP TABLE Types;

DROP TABLE Salle;

DROP TABLE Segment;

# TP2 – Insertion de données

InsParc.sql :

INSERT INTO Segment (IndIP, nomSegment, etage) VALUES

("130.120.80", "Brin", "RDC"),

("130.120.81", "Brin", "1er etage"),

("130.120.82", "Brin", "2e etage"),

("130.120.83", "Brin", "3e etage");

INSERT INTO Salle (nSalle, nomSalle, nbPoste, indIP) VALUES

("s01", "Salle 1", 3, "130.120.80"),

("s02", "Salle 2", 2, "130.120.80"),

("s03", "Salle 3", 2, "130.120.80"),

("s11", "Salle 11", 2, "130.120.81"),

("s12", "Salle 12", 1, "130.120.81"),

("s21", "Salle 21", 2, "130.120.82"),

("s22", "Salle 22", 0, "130.120.83"),

("s23", "Salle 23", 0, "130.120.83");

INSERT INTO Types (TypeLP, nomType) VALUES

("TX", "Terminal X-Windows"),

("UNIX", "Système Unix"),

("PCNT", "PC Windows NT"),

("PCWS", "PC Windows"),

("NC", "Network Computer");

INSERT INTO Poste (nPoste, nomPoste, indIP, ad, typePoste, nSalle) VALUES

("p1", "Poste 1", "130.120.80", 01, "TX", "s01"),

("p2", "Poste 2", "130.120.80", 02, "UNIX", "s01"),

("p3", "Poste 3", "130.120.80", 03, "TX", "s01"),

("p4", "Poste 4", "130.120.80", 04, "PCWS", "s02"),

("p5", "Poste 5", "130.120.80", 05, "PCWS", "s02"),

("p6", "Poste 6", "130.120.80", 06, "UNIX", "s03"),

("p7", "Poste 7", "130.120.80", 07, "TX", "s03"),

("p8", "Poste 8", "130.120.81", 01, "UNIX", "s11"),

("p9", "Poste 9", "130.120.81", 02, "TX", "s11"),

("p10", "Poste 10", "130.120.81", 03, "UNIX", "s12"),

("p11", "Poste 11", "130.120.82", 01, "PCNT", "s21"),

("p12", "Poste 12", "130.120.82", 02, "PCWS", "s21");

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

("log1", "Oracle 6", "1995-05-13", "6.2", "UNIX", 3000),

("log2", "Oracle 8", "1999-09-15", "8i", "UNIX", 5600),

("log3", "SQL Server", "1998-04-12", "7", "PCNT", 2700),

("log4", "Front Page", "1997-06-03", "5", "PCWS", 500),

("log5", "Win Dev", "1997-05-12", "5", "PCWS", 750),

("log6", "SQL\*Net", NULL, "2.0", "UNIX", 500),

("log7", "I. I. S.", "2002-04-12", "2", "PCNT", 810),

("log8", "Dreamweaver", "2004-09-21", "2.0", "BeOS", 1400);

INSERT INTO Installer (nPoste, nLog, numIns, dateIns, delai) VALUES

("p2", "log1", 1, "2003-05-15", NULL),

("p2", "log2", 2, "2003-09-17", NULL),

("p4", "log5", 3, "2003-05-20", NULL),

("p6", "log6", 4, "2003-05-20", NULL),

("p6", "log1", 5, "2003-05-15", NULL),

("p8", "log2", 6, "2003-05-19", NULL),

("p8", "log6", 7, "2003-05-20", NULL),

("p11", "log3", 8, "2003-04-20", NULL),

("p12", "log4", 9, "2003-04-20", NULL),

("p11", "log7", 10, "2003-04-20", NULL),

("p7", "log7", 11, "2002-04-01", NULL);

Modification.sql :

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 Logiciel

SET prix = prix \* 0.9

WHERE typeLog = 'PCNT';

SELECT \* FROM Segment;

SELECT nLog, typeLog, prix FROM Logiciel;

# TP3 – Evolution d’un schema

Évolution.sql :

ALTER TABLE Segment

    ADD nbSalle TINYINT(2) DEFAULT 0,

    ADD nbPoste TINYINT(2) DEFAULT 0;

ALTER TABLE Logiciel

    ADD nbInstall TINYINT(2) DEFAULT 0;

ALTER TABLE Poste

    ADD nbLog TINYINT(2) DEFAULT 0;

-- Vérification de la structure et du contenu

DESCRIBE Segment;

DESCRIBE Logiciel;

DESCRIBE Poste;

SELECT \* FROM Segment;

SELECT \* FROM Logiciel;

SELECT \* FROM Poste;

-- Augmenter la taille de nomSalle à VARCHAR(30)

ALTER TABLE Salle

    MODIFY nomSalle VARCHAR(30) NOT NULL;

-- Diminuer la taille de nomSegment à VARCHAR(15)

ALTER TABLE Segment

    MODIFY nomSegment VARCHAR(15) NOT NULL;

-- Tenter de diminuer la taille à VARCHAR(14) (peut échouer si des valeurs > 14 caractères existent)

ALTER TABLE Segment

    MODIFY nomSegment VARCHAR(14) NOT NULL;

-- Vérification de la structure et du contenu

DESCRIBE Salle;

DESCRIBE Segment;

SELECT \* FROM Salle;

SELECT \* FROM Segment;

-- Unicité de l'installation d'un logiciel sur un poste

ALTER TABLE Installer

    ADD CONSTRAINT unq\_Installer\_nPoste\_nLog UNIQUE (nPoste, nLog);

-- Clés étrangères pour l'intégrité référentielle

ALTER TABLE Poste

    ADD CONSTRAINT fk\_Poste\_indIP\_Segment FOREIGN KEY (indIP) REFERENCES Segment(indIP);

ALTER TABLE Poste

    ADD CONSTRAINT fk\_Poste\_nSalle\_Salle FOREIGN KEY (nSalle) REFERENCES Salle(nSalle);

ALTER TABLE Installer

    ADD CONSTRAINT fk\_Installer\_nPoste\_Poste FOREIGN KEY (nPoste) REFERENCES Poste(nPoste);

ALTER TABLE Poste

    ADD CONSTRAINT fk\_Poste\_typePoste\_Types FOREIGN KEY (typePoste) REFERENCES Types(typeLP);

ALTER TABLE Installer

    ADD CONSTRAINT fk\_Installer\_nLog\_Logiciel FOREIGN KEY (nLog) REFERENCES Logiciel(nLog);

-- Ajout de la contrainte entre Salle et Segment (peut échouer)

ALTER TABLE Salle

    ADD CONSTRAINT fk\_Salle\_indIP\_Segment FOREIGN KEY (indIP) REFERENCES Segment(indIP);

-- Ajout de la contrainte entre Logiciel et Types (peut échouer)

ALTER TABLE Logiciel

    ADD CONSTRAINT fk\_Logiciel\_typeLog\_Types FOREIGN KEY (typeLog) REFERENCES Types(typeLP);

-- Extraction des enregistrements problématiques

-- Salles dont indIP n'existe pas dans Segment

SELECT nSalle, indIP FROM Salle

WHERE indIP IS NOT NULL AND indIP NOT IN (SELECT indIP FROM Segment);

-- Logiciels dont typeLog n'existe pas dans Types

SELECT nLog, typeLog FROM Logiciel

WHERE typeLog IS NOT NULL AND typeLog NOT IN (SELECT typeLP FROM Types);

-- Suppression des salles problématiques (exemple pour 's22' et 's23')

DELETE FROM Salle

WHERE indIP IS NOT NULL AND indIP NOT IN (SELECT indIP FROM Segment);

-- Ajout du type de logiciel manquant dans Types (exemple : 'BeOS', 'Système Be')

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

-- Réexécution des ajouts de contraintes (devra maintenant réussir)

ALTER TABLE Salle

    ADD CONSTRAINT fk\_Salle\_indIP\_Segment FOREIGN KEY (indIP) REFERENCES Segment(indIP);

ALTER TABLE Logiciel

    ADD CONSTRAINT fk\_Logiciel\_typeLog\_Types FOREIGN KEY (typeLog) REFERENCES Types(typeLP);

-- Vérification : plus d'enregistrements problématiques

SELECT nSalle, indIP FROM Salle

WHERE indIP IS NOT NULL AND indIP NOT IN (SELECT indIP FROM Segment);

SELECT nLog, typeLog FROM Logiciel

WHERE typeLog IS NOT NULL AND typeLog NOT IN (SELECT typeLP FROM Types);

# TP5 – Conception de base de données

Premier Exercice :

* MCD :

Produit 🡨🡪 Vente 🡨🡪 Vente\_Produit

* MLD :

PRODUIT(idProduit, nomProduit, typeProduit)

VENTE(idVente, dateVente)

VENTE\_PRODUIT(idVente, idProduit, poids\_kg, prix\_kg)

* MPD :

CREATE TABLE Produit (

idProduit INT PRIMARY KEY AUTO\_INCREMENT,

nomProduit VARCHAR(50),

typeProduit ENUM('animal', 'fruit', 'légume')

);

CREATE TABLE Vente (

idVente INT PRIMARY KEY AUTO\_INCREMENT,

dateVente DATE

);

CREATE TABLE Vente\_Produit (

idVente INT,

idProduit INT,

poids\_kg DECIMAL(5,2),

prix\_kg DECIMAL(5,2),

PRIMARY KEY (idVente, idProduit),

FOREIGN KEY (idVente) REFERENCES Vente(idVente),

FOREIGN KEY (idProduit) REFERENCES Produit(idProduit)

);

Deuxième exercice :

MCD :

Client 🡨🡪 Materiel 🡨🡪 Intervention 🡨🡪 Composant

1 Intervention 🡨🡪 1 matériel

1 Intervention 🡨🡪 n composants

1 Client 🡨🡪 n matériel

MLD :

CLIENT(idClient, nom, adresse, téléphone)

MATERIEL(idMatériel, marque, modèle, idClient)

INTERVENTION(idIntervention, date, durée, type, tarifHoraire, idMatériel)

COMPOSANT(idComposant, nom, prix)

INTERVENTION\_COMPOSANT(idIntervention, idComposant, quantité)

MPD :

CREATE TABLE Client (

idClient INT PRIMARY KEY AUTO\_INCREMENT,

nom VARCHAR(50),

adresse VARCHAR(100),

téléphone VARCHAR(20)

);

CREATE TABLE Materiel (

idMatériel INT PRIMARY KEY AUTO\_INCREMENT,

marque VARCHAR(50),

modèle VARCHAR(50),

idClient INT,

FOREIGN KEY (idClient) REFERENCES Client(idClient)

);

CREATE TABLE Intervention (

idIntervention INT PRIMARY KEY AUTO\_INCREMENT,

date DATE,

durée DECIMAL(4,2),

type VARCHAR(50),

tarifHoraire DECIMAL(5,2),

idMatériel INT,

FOREIGN KEY (idMatériel) REFERENCES Materiel(idMatériel)

);

CREATE TABLE Composant (

idComposant INT PRIMARY KEY AUTO\_INCREMENT,

nom VARCHAR(50),

prix DECIMAL(6,2)

);

CREATE TABLE Intervention\_Composant (

idIntervention INT,

idComposant INT,

quantité INT,

PRIMARY KEY (idIntervention, idComposant),

FOREIGN KEY (idIntervention) REFERENCES Intervention(idIntervention),

FOREIGN KEY (idComposant) REFERENCES Composant(idComposant)

);

# TP6 – JDBC

public ArrayList<String> getSalles() {

ArrayList<String> result = new ArrayList<>();

try (Connection conn = getConnection();

Statement statement = conn.createStatement();

ResultSet resultSet = stmt.executeQuery("SELECT \* FROM Salle")) {

while (resultSet.next()) {

result.add(resultSet.getString("nSalle") + " - " + resultSet.getString("nomSalle"));

}

} catch (SQLException e) {

e.printStackTrace();

}

return result;

}

public void deleteSalle(String nSalle) {

try (Connection conn = getConnection();

Statement stmt = conn.createStatement(ResultSet.TYPE\_SCROLL\_SENSITIVE, ResultSet.CONCUR\_UPDATABLE);

ResultSet rs = stmt.executeQuery("SELECT \* FROM Salle WHERE nSalle='" + nSalle + "'")) {

if (rs.next()) {

rs.deleteRow();

}

} catch (SQLException e) {

if (e.getErrorCode() == 1451) {

System.out.println("Impossible de supprimer : contrainte référentielle");

} else {

e.printStackTrace();

}

}

}