1) **Stockage schema-unaware : Vertical-Edge vs Monet**

1) **Schéma Vertical-edge et monet en SQL :**

Schéma vertical-edge :

```
CREATE TABLE Vertical-Edge(
        source VARCHAR(255),
        target VARCHAR(255) primary key not null,
        ordinal INT,
        txtval VARCHAR(255),
        numval NUMBER,
        foreign key(source) references Vertical-Edge(target)
);
```

Ca sera la même commande pour chaque table ci-dessous, on modifie juste le nom et la clé étrangère de son parent.

Schéma Monet :

```
CREATE TABLE Monet(
        node VARCHAR(255) primary key not null,
        txtval VARCHAR(255),
        numval NUMBER
)
```

2) **Peupler les tables du TD1 Question 1 :**

1) Vertical-Edge :

Presse

source	target	ordinal	txtval	numval
	n1	1		

```
INSERT INTO presse VALUES (null, 'n1', 1, null, null);
```

Journalistes

source	target	ordinal	txtval	numval
n1	n2	1		

INSERT INTO journalistes VALUES ('n1', 'n2', 2, null, null);

Journaliste

source	target	ordinal	txtval	numval
n2	n12	1		

INSERT INTO journaliste VALUES ('n2', 'n12', 1, null, null);

Journaliste_id

source	target	ordinal	txtval	numval
n12	n13	1		1

INSERT INTO journaliste_id VALUES ('n12', 'n13', 1, null, 1);

Journaliste_anonymisation

source	target	ordinal	txtval	numval
n12	n14	2	“oui”	

INSERT INTO journaliste_anonymisation VALUES ('n12', 'n14', 2, 'oui', null);

Journaliste_nom

source	target	ordinal	txtval	numval
n12	n15	3	TRAN	

INSERT INTO journaliste_nom VALUES ('n12', 'n15', 3, 'Tran', null);

Journaliste_prenom

source	target	ordinal	txtval	numval
n12	n16	4	Thi Tra My	

INSERT INTO journaliste_prenom VALUES ('n12', 'n16', 4, 'My', null);

Journal

source	target	ordinal	txtval	numval
n1	n3	2		

INSERT INTO journal VALUES ('n1', 'n3', 1, null, null);

Journal_nom

source	target	ordinal	txtval	numval
n3	n4	1	“MIDI LIBRE”	

INSERT INTO journal_nom VALUES ('n3', 'n4', 1, 'MIDI LIBRE', null);

Journal_directeur

source	target	ordinal	txtval	numval
n3	n5	2		

INSERT INTO journal_directeur VALUES ('n3', 'n5', 2, null, null);

Journal_article

source	target	ordinal	txtval	numval
n3	n6	3		

INSERT INTO journal_article VALUES ('n3', 'n6', 3, null, null);

Journal_directeur_nom

source	target	ordinal	txtval	numval
n5	n7	1	NGUYEN	

INSERT INTO journal_directeur_nom VALUES ('n5', 'n7', 1, 'Nguyen', null);

Journal_directeur_prenom

source	target	ordinal	txtval	numval
n5	n8	2	Khang	

INSERT INTO journal_directeur_prenom VALUES ('n5', 'n8', 2, 'Huu Khang', null);

Journal_article_titre

source	target	ordinal	txtval	numval
n6	n9	1	Voici le titre	

INSERT INTO journal_article_titre VALUES ('n6', 'n9', 1, 'Voici le titre1', null);

Journal_acticle_auteur

source	target	ordinal	txtval	numval
n6	n10	2		1

INSERT INTO journal_article_auteur VALUES ('n6', 'n10', 2, null, null);

Journal_article_corps

source	target	ordinal	txtval	numval
n6	n11	3	Voici le corps de l'article	

INSERT INTO journal_article_corps VALUES ('n6', 'n11', 3, 'Ceci est le corps', null);

2) **Monet :**

Presse

node	txtval	numval
n1		

Presse_journalistes

node	txtval	numval
n2		

Presse_journalistes_journaliste

node	txtval	numval
n3		

Presse_journalistes_journaliste_id

node	txtval	numval
n4		1

Presse_journalistes_journaliste_anonymisation

node	txtval	numval
n6	oui	

Presse_journalistes_journaliste_nom

node	txtval	numval
n8	TRAN	

Presse_journalistes_journaliste_prenom

node	txtval	numval
n10	MY	

Presse_journal

node	txtval	numval
n12		

Presse_journal_nom

node	txtval	numval

n13	MIDI LIBRE	

Presse_journal_directeur

node	txtval	numval
n15		

Presse_journal_directeur_nom

node	txtval	numval
n16	NGUYEN	

Presse_journal_directeur_prenom

node	txtval	numval
n18	KHANG	

Presse_journal_article_titre

node	txtval	numval
n20	Voici le titre	

Presse_journal_article_auteur

node	txtval	numval
n22		1

Presse_journal_article_corps

node	txtval	numval
n24	Voici le corps	

3) Requête XPATH sous SQL

1) Tous les noms des journalistes

XPATH :
//presse/journalistes/journaliste/nom/text()

Vertical-Edge :
```
SELECT journaliste_nom.txtval
FROM presse, journalistes, journaliste, journaliste_nom
WHERE presse.target = journalistes.source AND journalistes.target = journaliste.source
AND journaliste.target = journaliste_nom.source;
```

Monet :
```
SELECT txtval
FROM Presse_journalistes_journaliste_nom
```

2) Tous les prenoms des journalistes

XPATH :
//presse/journalistes/journaliste/prenom/text()

Vertical-Edge :
```
SELECT journaliste_prenom.txtval
FROM presse, journalistes, journaliste, journaliste_prenom
WHERE presse.target = journalistes.source AND journalistes.target = journaliste.source
AND journaliste.target = journaliste_prenom.source;
```

Monet :
```
SELECT txtval
FROM Presse_journalistes_journaliste_prenom
```

3) Tous les noms des journals de la presse

XPATH :
//presse/journal/nom/text()

Vertical-Edge :
```
SELECT journal_nom.txtval
FROM presse, journal, journal_nom
WHERE presse.target = journal.source AND journal.target = journal_nom.source;
```

Monet :
```
SELECT txtval
FROM Presse_journal_nom;
```

4) Le nom de l’auteur de l’article du node 6

XPATH :

//presse[journalistes/journaliste/id/@id =
journal/article/auteur/text()]/journalistes/journaliste/nom/text()

Vertical-Edge :

```
SELECT j4.txtval
FROM presse p
        JOIN journalistes j1 ON p.target = j1.source
        JOIN journaliste j2 ON j1.target = j2.source
        JOIN journaliste_id j3 ON j2.target = j3.source
        JOIN journaliste_nom j4 ON j2.target = j4.source
        JOIN journal j5 ON p.target = j5.source
        JOIN journal_article j6 ON j5.target = j6.source
        JOIN journal_article_auteur j7 ON j6.target = j7.source
WHERE j7.target = ‘n6’ AND j7.numval = j3.numval;
```

Monet :

Nous pensons que c’est impossible d’effectuer cette requête avec le Monet-DB

5) Nombre total des journalistes

XPATH : count(//presse/journalistes/journaliste)

Vertical-Edge :

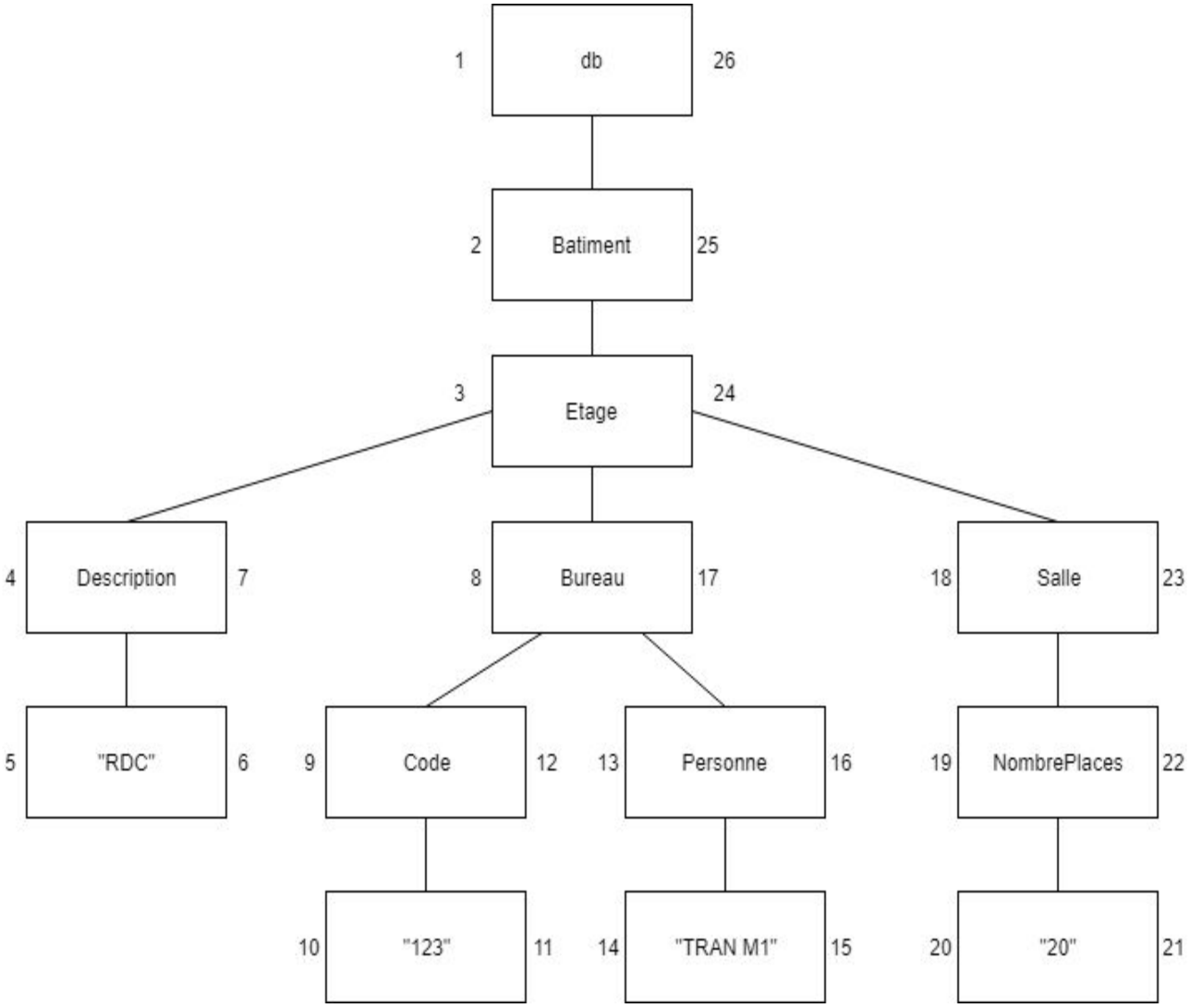
```
SELECT COUNT(*)
FROM presse p
        JOIN journalistes j1 ON p.target = j1.source
        JOIN journaliste j2 ON j1.target = j2.source
        JOIN journaliste_id j3 ON j2.target = j3.source;
```

Monet :

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

2) Stockage schema-aware

1) **Schéma stockage relationnel**



2) **Peupler les tables**

NODE Table

begin	end	par	tag	type
1	24		batiment	ELT
2	23	1	etage	ELT
3	6	2	description	ELT
4	5	3	“RDC”	TEXT
7	16	2	bureau	ELT
8	11	7	code	ELT
9	10	8	“123”	TEXT
12	15	7	personne	ELT
13	14	12	“TRAN M1”	TEXT
17	22	2	salle	ELT
18	21	17	nombrePlaces	ELT
19	20	18	“20”	TEXT

3) **Requête XPATH sous SQL**

1) Le code de l’étage “RDC”

XPATH :
//batiment/etage[description/text() = “RDC”]/code/text()

SQL :
```
SELECT c.tag
FROM NODETable a, NODETable b, NODETable c, NODETABLE d
WHERE a.tag = “batiment” AND b.tag = “etage” AND c.tag = “code” AND d.description
        AND Descendant(a.begin, b.begin)
        AND Descendant(b.begin, c.begin)
        AND Descendant (b.begin, d.begin)
        AND Descendant (d.begin, “description”);
```

2) Total des places

XPATH :
sum(//batiment/etage/salle/nombresPlaces)

SQL :
SELECT SUM(nbPlaces.tag)
FROM NODETable batiment, NODETable etage, NODETable salle, NODETable nbPlaces
WHERE batiment.tag = ‘batiment’ AND etage.tag = ‘etage’ AND salle.tag = ‘salle’ AND nbPlaces.tag = ‘nbPlaces’
AND Descendant (batiment.begin, etage.begin)
AND Descendant (etage.begin, salle.begin)
AND Descendant (salle.begin, nbPlaces.begin);

3) Tous les étages du batiment

XPATH :
//batiment/etage/description/text()

SQL :
SELECT description.tag
FROM NODETable batiment, NODETable etage, NODETable description
WHERE batiment.tag = ‘batiment’ AND etage.tag = ‘etage’ AND description.tag = ‘description’
AND Descendant(batiment.begin, etage.begin)
AND Descendant(etage.begin, description.begin)

4) Tous les salles ayant plus de 10 places

XPATH :
//batiment/etage/salle[text() > 10]

SQL :
SELECT salle.begin
FROM NODETable batiment, NODETable etage, NODETable salle
WHERE batiment.tag = ‘batiment’ AND etage.tag = ‘etage’ AND salle.tag = ‘salle’
AND Descendant(batiment.begin, etage.begin)
AND Descendant(etage.begin, salle.begin)
HAVING COUNT (salle.tag) > 10
GROUP BY salle.begin

5) Tous les bureaux

XPATH :
//batiment/etage/bureau

SQL :
SELECT bureau.begin
FROM NODETable batiment, NODETable etage, NODETable bureau
WHERE batiment.tag = ‘batiment’ AND etage.tag = ‘etage’ AND bureau.tag = ‘bureau’’
AND Descendant (batiment.begin, etage.begin)
AND Descendant (etage.begin, bureau.begin)

3) Oracle-XML

5) Requête XPATH et XQUERY

A) avec XPATH

1) Tous les utilisateurs

```
SELECT EXTRACT (colonne_xml, '//utilisateur') FROM tweeter_table_clob;
```

2) Tous les tweets

```
SELECT EXTRACT (colonne_xml, ‘//tweet’) FROM tweeter_table_clob;
```

3) le tweet dont l’id est t16

```
SELECT EXTRACT (colonne_xml, ‘//tweet[@id = “t16”]‘) FROM tweeter_table_clob;
```

4) le tweet dont l’auteur est de l’id 16

```
SELECT EXTRACT (colonne_xml, ‘//tweet[auteur/@idref = “u16”]’) FROM
tweeter_table_clob;
```

5) le tweet contient le hashtag “#heheXD!”

```
SELECT EXTRACT (colonne_xml, ‘//tweet[corps/hashtags/hashtag[contains(.,
“#heheXD!”)]]’) FROM tweeter_table_clob;
```

B) avec XQUERY

1) Tous les utilisateurs

```
SELECT XMLQUERY ('for $x in //tweet return ($x, //utilisateur[@id = $x/auteur/@idref])'
PASSING colonne_xml RETURNING CONTENT) FROM tweeter_table_clob;
```

2) Tous les utilisateur dans l’ordre croissant

```
SELECT XMLQUERY ('for $x in //utilisateur order by $x return $x' PASSING colonne_xml
RETURNING CONTENT) FROM tweeter_table_clob;
```

3) Le date le plus ancien

SELECT XMLQUERY (‘let $y := min(//tweet/Date/text()) return $y’ PASSING colonne_xml RETURNING CONTENT) FROM tweeter_table_clob;

4) affiches tous les tweets avec ses hashtags

SELECT XMLQUERY (‘for $x in //tweet let $y := /$x/corps/hashtags return ($x, $y)’ PASSING colone_xml RETURNING CONTENT) FROM tweeter_table_clob;

5) pour chaque tweet, afficher son auteur

SELECT XMLQUERY (‘for $x in //tweet return ($x, //utilisateur[@id = $x/auteur/@idref])’ PASSING colonne_xml RETURNING CONTENT) FROM tweeter_table_clob;

4) Interval-encoding avec SAX

1) L’encodage begin/end et dewey de l’XML

a) l’encodage begin/end

```
<batiment begin="1" end="24">
  <etage begin="2" end="23">
    <description begin="3" end="6">
      <texte begin="4" end="5">
        RDC
      </texte>
    </description>
    <bureau begin="7" end="16">
      <code begin="8" end="11">
        <texte begin="9" end="10">
          123
        </texte>
      </code>
      <personne begin="12" end="15">
        <texte begin="13" end="14">
          TRAN M1
        </texte>
      </personne>
    </bureau>
    <salle begin="17" end="22">
      <nombrePlaces begin="18" end="21">
        <texte begin="19" end="20">
          20
        </texte>
      </nombrePlaces>
    </salle>
  </etage>
```

</batiment>

b) l'encodage dewey

```
<db>
   <batiment nodeID="1">
      <etage nodeID="1.1">
         <description nodeID="1.1.1">
            <texte nodeID="1.1.1.1">
               RDC
            </texte>
         </description>
         <bureau nodeID="1.1.2">
            <code nodeID="1.1.2.1">
               <texte nodeID="1.1.2.1.1">
                  123
               </texte>
            </code>
            <personne nodeID="1.1.2.2">
               <texte nodeID="1.1.2.2.1">
                  TRAN M1
               </texte>
            </personne>
         </bureau>
         <salle nodeID="1.1.3">
            <nombrePlaces nodeID="1.1.3.1">
               <texte nodeID="1.1.3.1.1">
                  20
               </texte>
            </nombrePlaces>
         </salle>
      </etage>
   </batiment>
</db>
```

2) Créer et peupler le schéma de stockage

CREATE TABLE Node (begin int, end int, par int, tag VARCHAR(255), nodtyp VARCHAR(255), primary key (begin, end));

INSERT INTO NODE VALUES (1,26, null, 'db', 'ELT');
INSERT INTO NODE VALUES (2, 25, 1, 'batiment', 'ELT');
INSERT INTO NODE VALUES (3, 24, 2, 'etage', 'ELT');
INSERT INTO NODE VALUES (4, 7, 3, 'description', 'ELT');
INSERT INTO NODE VALUES (5,6, 4, 'RDC', 'TEXT');

```
INSERT INTO NODE VALUES (8, 17, 3, 'bureau', 'ELT');
INSERT INTO NODE VALUES (9, 12, 8, 'code', 'ELT');
INSERT INTO NODE VALUES (10, 11, 9, '123', 'TEXT');
INSERT INTO NODE VALUES (13, 16, 8, 'personne', 'ELT');
INSERT INTO NODE VALUES (14, 15, 13, 'TRAN M1', 'TEXT');
INSERT INTO NODE VALUES (18, 23, 2, 'salle', 'ELT');
INSERT INTO NODE VALUES (19, 22, 18, 'nombrePlaces', 'ELT');
INSERT INTO NODE VALUES (20, 21, 19, '20', 'TEXT');
```

3) Programmation l'encodage begin/end

```
public void startElement(String namespaceURI, String localName,
                         String qName, Attributes atts)
          throws SAXException
   {
      System.out.println("starting an element "+localName);

      if(qName.equalsIgnoreCase("dtexte")){
         bdtexte = true;
         tabDescriptionTexte[0] = atts.getValue("begin");
         tabDescriptionTexte[1] = atts.getValue("end");
         tabDescriptionTexte[2] = atts.getValue("par");
      }
      else if(qName.equalsIgnoreCase("ctexte")){
         bctexte = true;
         tabCodeTexte[0] = atts.getValue("begin");
         tabCodeTexte[1] = atts.getValue("end");
         tabCodeTexte[2] = atts.getValue("par");
      }
      else if(qName.equalsIgnoreCase("ptexte")){
         bptexte = true;
         tabPersoTexte[0] = atts.getValue("begin");
         tabPersoTexte[1] = atts.getValue("end");
         tabPersoTexte[2] = atts.getValue("par");
      }
      else if(qName.equalsIgnoreCase("nptexte")){
         bnptexte = true;
         tabNbPlacesTexte[0] = atts.getValue("begin");
         tabNbPlacesTexte[1] = atts.getValue("end");
         tabNbPlacesTexte[2] = atts.getValue("par");
      } else{
         System.out.println("INSERT INTO NODE (begin, end, par, tag, nodtyp) VALUES (" +
atts.getValue("begin") + ", " + atts.getValue("end") + ", " + atts.getValue("par") + ", '" +
localName + "', Element);");
      }
```

```
   }

public void characters(char[] ch, int start, int length) throws SAXException
   {
      String nodeTexte = new String(ch, start, length);
       if(bptexte){
           System.out.println("INSERT INTO NODE (begin, end, par, tag, nodtyp) VALUES (" +
tabPersoTexte[0] + ", " + tabPersoTexte[1] + ", " + tabPersoTexte[2] + ", '" + nodeTexte.trim()
+ "', Texte);");
           bptexte = false;
       }else if (bdtexte){
           System.out.println("INSERT INTO NODE (begin, end, par, tag, nodtyp) VALUES (" +
tabDescriptionTexte[0] + ", " + tabDescriptionTexte[1] + ", " + tabDescriptionTexte[2] + ", '" +
nodeTexte.trim() + "', Texte);");
           bdtexte = false;
       }else if (bctexte){
           System.out.println("INSERT INTO NODE (begin, end, par, tag, nodtyp) VALUES (" +
tabCodeTexte[0] + ", " + tabCodeTexte[1] + ", " + tabCodeTexte[2] + ", '" + nodeTexte.trim()
+ "', Texte);");
           bctexte = false;
       }else if (bnptexte){

           System.out.println("INSERT INTO NODE (begin, end, par, tag, nodtyp) VALUES (" +
tabNbPlacesTexte[0] + ", " + tabNbPlacesTexte[1] + ", " + tabNbPlacesTexte[2] + ", '" +
nodeTexte.trim() + "', Texte);");
           bnptexte = false;
       }
   }
```

6) Programme pour l'encodage Dewey

```
public void startElement(String namespaceURI, String localName,
                         String qName, Attributes atts)
           throws SAXException
   {
       if(qName.equalsIgnoreCase("dtexte")){
           bdtexte = true;
           descriptionIndex = atts.getValue("nodeID");
       }
       else if(qName.equalsIgnoreCase("ctexte")){
           bctexte = true;
           codeIndex = atts.getValue("nodeID");
       }
       else if(qName.equalsIgnoreCase("ptexte")){
           bptexte = true;
```

```
            persoIndex = atts.getValue("nodeID");
        }
        else if(qName.equalsIgnoreCase("nptexte")){
            bnptexte = true;
            nbPlacesIndex = atts.getValue("nodeID");
        } else{
            System.out.println("INSERT INTO NODE (nodeID, tag, type) VALUES (" +
atts.getValue("nodeID") + ", '" + localName + "', " + "', Element);");
        }
    }


public void characters(char[] ch, int start, int length) throws SAXException
    {
        String nodeTexte = new String(ch, start, length);
        if(bptexte){
            System.out.println("INSERT INTO NODE (nodeID, tag, type) VALUES (" +
persoIndex + ", '" + nodeTexte.trim() + "', " + "', Texte);");
            bptexte = false;
        }else if (bdtexte){
            System.out.println("INSERT INTO NODE (nodeID, tag, type) VALUES (" +
descriptionIndex + ", '" + nodeTexte.trim() + "', " + "', Texte);");
            bdtexte = false;
        }else if (bctexte){
            System.out.println("INSERT INTO NODE (nodeID, tag, type) VALUES (" + codeIndex
+ ", '" + nodeTexte.trim() + "', " + "', Texte);");
            bctexte = false;
        }else if (bnptexte){
            System.out.println("INSERT INTO NODE (nodeID, tag, type) VALUES (" +
nbPlacesIndex + ", '" + nodeTexte.trim() + "', " + "', Texte);");
            bnptexte = false;
        }
    }
```