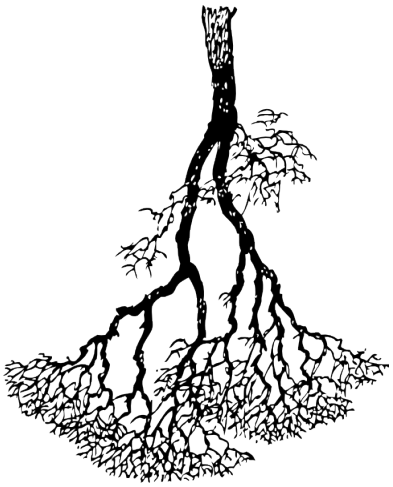
DEPARTEMENT INFORMATIQUE
DE LA FACULTE DES SCIENCES

Quentin Yeche (21520370), Yanis Allouch (21708237)

Rapport du TP Noté N°2 : XPath/XQuery



HMIN103 — Données du web

Référent: Federico Ulliana et Pierre Pompidor

2020

Table des matières

Introduction

L'analyse et le traitement des données consistent à les étudier afin d'en extraire des structures capables de les stocker ainsi que de les manipuler de façon la plus optimale.

Ce sixième TP est une synthèse des précédents, de la conception de structures de données XML via DTD à l'utilisation de *XPATH* et *XQUERY* jusqu'a leur mise en oeuvre dans une base de donnée relationnel.

Ce TP est composé de trois exercices. Le premier concerne une implémentation d'une DTD au format *Monet* sous Oracle, le second propose de faire une modélisation *schema-aware*. Enfin le dernier exercice à pour objectif de développer les concepts de *l'interval-encoding* en s'aidant de l'outil *SaxParser*.

Le TP se déroule sur papier et machine.
Durant ce TP, nous appliquerons les règles qui seront abordées dans les cours référencés DdW4-Rel2XML (partie 1) et DdW4-XML2REL (partie 2).

L'ensemble des réponses est reporté dans ce compte-rendu.

1 Stockage Monet

1. Implémenter sous Oracle le schéma Monet permettant de stocker un document XML pour les bâtiments que vous avez proposé pour le TP 1.

```
DROP TABLE batiment;
DROP TABLE batiment_etage;
DROP TABLE batiment_etage_bureau;
DROP TABLE batiment_etage_bureau_code;
DROP TABLE batiment_etage_bureau_personne;
DROP TABLE batiment_etage_salle;
DROP TABLE batiment_etage_salle_nombrePlaces;



CREATE TABLE batiment ( node varchar(20));

CREATE TABLE batiment_etage ( node varchar(20) , txtval varchar(20));

CREATE TABLE batiment_etage_bureau ( node varchar(20));

CREATE TABLE batiment_etage_bureau_code ( node varchar(20), numval number
    (10));

CREATE TABLE batiment_etage_bureau_personne ( node varchar(20), txtval
    varchar(20));

CREATE TABLE batiment_etage_salle ( node varchar(20));

CREATE TABLE batiment_etage_salle_nombrePlaces ( node varchar(20), numval
     number(10));

INSERT INTO batiment (node)
VALUES ('n1');

INSERT INTO batiment_etage (node, txtval)
VALUES ('n2', 'foobarDescription1');

INSERT INTO batiment_etage_bureau (node)
VALUES ('n3');

INSERT INTO batiment_etage_bureau_code (node, numval)
VALUES ('n4', 'foobarCode1');

INSERT INTO batiment_etage_salle (node)
VALUES ('n5');

INSERT INTO batiment_etage_salle_nombrePlaces (node, numval)
VALUES ('n6', '421');

INSERT INTO batiment_etage (node, txtval)
VALUES ('n7', 'foobarDescription2');

INSERT INTO batiment_etage_bureau (node)
VALUES ('n8');

INSERT INTO batiment_etage_bureau_code (node, numval)
VALUES ('n9', '422');

INSERT INTO batiment_etage_bureau_personne (node, txtval)
VALUES ('n10', 'william');

```

```sql
INSERT INTO batiment_etage_salle (node)
VALUES ('n11');

INSERT INTO batiment_etage_salle_nombrePlaces (node, numval)
VALUES ('n12', '422');
```

2. Exprimez en SQL trois requêtes XPath de votre choix sur ce document.

```sql
-- 1 --  //personne[self::node() = 'william']

SELECT bebp.txtval
FROM batiment_etage_bureau_personne bebp
WHERE bebp.txtval = 'william';

-- 2 --  //etage[.//personne='william' or ()]

/*
Exemple de requete qui ne s'exprime pas en SQL sans ajouter des FK et une
      relation de parent sur la table.
*/

-- 3 --  //etage/description

SELECT be.txtval
FROM batiment_etage be;


-- 4 --  //nombrePlaces[. > 421]

SELECT besn.numval
FROM batiment_etage_salle_nombrePlaces besn
WHERE besn.numval > '421';
```

3. Comment pourrait on étendre le schéma Monet pour gérer l'axe de navigation ancestor ?
   — Proposition :
   Comme vu sur la seconde requête XPATH, certaines requêtes ne se traduisent pas en SQL sans étendre le schéma relationnel. La solution consiste à ajouter une colonne parent et/ou ancêtre pour pouvoir y naviguer dans nos tables relationnel comme dans un document XML.

2 Stockage schema-aware

1. À partir de la DTD proposée pour la presse (voir TP 1), définir le schéma de stockage relationnel associé.

   Voici la DTD obtenue lorsque l'on a appliqué les règles d'affaiblissement des expressions régulières :

```
<!DOCTYPE presse [
  <!ELEMENT presse (journal|journalistes) >
  <!ELEMENT journal (nom | directeur | article*) >
  <!ELEMENT article (corps) >
  <!ATTLIST article titre CDATA #IMPLIED >
  <!ATTLIST article auteur IDREF #REQUIRED >
  <!ELEMENT corps (#PCDATA) >
  <!ELEMENT journalistes (journaliste*) > <!-- only is a section-->
  <!ATTLIST journaliste idJ ID #REQUIRED >
  <!ELEMENT journaliste (nom|prenom|pseudonyme) >
  <!ATTLIST journaliste anonymisation (oui|non) "non" > <!-- +check-->
  <!ELEMENT pseudonyme (#PCDATA) >
  <!ELEMENT nom (#PCDATA) >
  <!ELEMENT prenom (#PCDATA) >
  <!ELEMENT directeur (nom|prenom) >
]>

```

   Quelques remarques supplémentaires sont utiles avant de passer à la suite :
   - Tout d'abord nous choisissons ici de ne pas avoir de table directeur et d'intégrer les informations de l'unique directeur dans la table journal. Le journal étant unique la table journal n'aura qu'une seule ligne. On traduira d'ailleurs cette propriété par des contraintes en SQL ;
   - Puisqu'il n'y a qu'un seul journal la table article n'aura pas besoin de la référence à journal que la transformation canonique implique ;
   - De même la balise journalistes est seulement utile à la hiérarchie du document XML, et elle est unique. Cette hiérarchie sera conservée par le simple fait que les instances de journaliste seront toutes dans la table journaliste.
   - Nous essaierons d'ajouter des contraintes SQL afin de récupérer certaines informations que la transformation canonique efface, telle que la présence d'un pseudonyme XOR un couple prénom nom.

   Les tables que nous créons sont donc les suivantes :
   - journal(nom,nomDir,prenomDir)
   - journaliste(<u>idJ</u>, nom, prenom, pseudonympe, anonymisation)
   - article(<u>articleID</u>, titre, *auteur*, corps)

   Enfin, voici les créations de table en SQL :

```
create table journal(
    verrou varchar(1) default '1',
    nom VARCHAR(20),
    nomDir VARCHAR(20),
    prenomDir VARCHAR(20),
    constraint PK_journal PRIMARY KEY (verrou),
    constraint CK_journal_locked CHECK (verrou='1'),
    constraint CK_journal_lock_NN check (verrou is not null)
```

```sql
);

create table journaliste (
  idJ VARCHAR(10) PRIMARY KEY,
  nom VARCHAR(20),
  prenom VARCHAR(20),
  pseudonyme VARCHAR(20),
  anonymisation VARCHAR(3),
  constraint CK_journal_anonymisation
      check (anonymisation in ('oui','non')),
  constraint CHK_journal_pseudoNom
      check ((pseudonyme is null) or (nom is null and prenom is null))
);

create table article (
  articleID NUMBER(10) PRIMARY KEY,
  titre VARCHAR(100),
  auteur VARCHAR(10),
  corps VARCHAR(4000),
  constraint FK_article_journaliste
      foreign key(auteur) references journaliste(idJ)
);

```

Remarque : La clé primaire de la table journal verrou et les contraintes associées garantissent l'unicité de la ligne.

2. Remplissez les tables avec des valeurs correspondants à l'un des documents XML que vous avez proposés lors du TP 1.
   Vous trouverez les insertions, ainsi que l'intégralité du code pour cet exercice, en annexe.
3. Traduisez en SQL trois requêtes XPath de votre choix.

```xpath
//article[not(contains(@titre/data(),"Macron"))
    and not(contains(corps/text(),"Macron"))]

```

```sql
select * from article
where ('Macron' not in titre and 'Macron' not in corps);

```

```xpath
//article[@auteur=//journaliste[@anomymisation/data()="oui"]/@idj]

```

```sql
select titre, pseudonyme, corps from article join journaliste on article.
    auteur = journaliste.idJ
where anonymisation = 'oui';

```

```xquery
for $j in //journaliste
return <resultat>
    {$j/nom/text()},
    {$j/prenom/text()},
    {$j/pseudonyme/text()},
    {count(//article[@auteur=$j/@idj])}
</resultat>

```

```
1 select nom, prenom, pseudonyme, count(*) from journaliste join article on
      article.auteur = journaliste.idJ
2 group by nom, prenom, pseudonyme;
3
```

3 Interval-encoding avec SAX

1. Donnez l'encodage begin/end de l'un des documents XML pour les bâtiments que vous avez proposé lors du TP 1, puis enregistrez-le dans la table NODE.
   — Voici le document XML utilisé pour l'encodage BEGIN/END auquel nous avons ajoutés devant chaque balise, un commentaire contenant le numéro BEGIN associé à la balise quand elle est ouvrante et END quand la balise est fermante.

```
<!--1--><batiment>
  <!--2--><etage>
    <!--3--><description><!--4-->foobarDescription1<!--5-->
            </description><!--6-->
    <!--7--><bureau>
      <!--8--><code><!--9-->foobarCode1<!--10-->
              </code><!--11-->
            </bureau><!--12-->
    <!--13--><salle>
      <!--14--><nombrePlaces><!--15-->421<!--16-->
              </nombrePlaces><!--17-->
            </salle><!--18-->
          </etage><!--19-->
   <!--20--><etage>
    <!--21--><description><!--22-->foobarDescription2<!--23-->
            </description><!--24-->
    <!--25--><bureau>
      <!--26--><code><!--27-->422<!--28-->
              </code><!--29-->
      <!--30--><personne><!--31-->william<!--32-->
              </personne><!--33-->
            </bureau><!--34-->
    <!--35--><salle>
      <!--36--><nombrePlaces><!--37-->422<!--38-->
              </nombrePlaces><!--39-->
            </salle><!--40-->
          </etage><!--41-->
        </batiment><!--42-->
```

   — La table NODE utilisée.

```
CREATE TABLE NODE (
  begin_ number(10),
  end_ number(10),
  parent number(10),
  tag varchar(20),
  type varchar(20),
  txtval varchar(20),
  CONSTRAINT PK_NODE PRIMARY KEY (begin_)
);
```

   — Voici enfin les insert dans la table NODE étant associé a l'XML précédent. Nous avons ajoutés de l'indentation au formatage du texte pour représenter la structure XML source.

```
INSERT INTO NODE (begin_, end_, parent, tag, type)
VALUES (1,42,'','batiment','ELT');
  INSERT INTO NODE (begin_, end_, parent, tag, type)
  VALUES (2,19,'1','etage','ELT');
    INSERT INTO NODE (begin_, end_, parent, tag, type)
    VALUES (3,6,'2','description','ELT');
```

```
    INSERT INTO NODE (begin_, end_, parent, tag, type, txtval)
    VALUES (4,5,'3','','TXT', 'foobarDescription1');
  INSERT INTO NODE (begin_, end_, parent, tag, type)
  VALUES (7,12,'2','bureau','ELT');
    INSERT INTO NODE (begin_, end_, parent, tag, type)
    VALUES (8,11,'7','code','ELT');
      INSERT INTO NODE (begin_, end_, parent, tag, type, txtval)
      VALUES (9,10,'8','','TXT', 'foobarCode1');
  INSERT INTO NODE (begin_, end_, parent, tag, type)
  VALUES (13,18,'2','salle','ELT');
    INSERT INTO NODE (begin_, end_, parent, tag, type, )
    VALUES (14,17,'13','nombrePlaces','ELT');
      INSERT INTO NODE (begin_, end_, parent, tag, type, txtval)
      VALUES (15,16,'14','','TXT', '421');
INSERT INTO NODE (begin_, end_, parent, tag, type)
VALUES (20,41,'1','etage','ELT');
  INSERT INTO NODE (begin_, end_, parent, tag, type)
  VALUES (21,24,'20','description','ELT');
    INSERT INTO NODE (begin_, end_, parent, tag, type, txtval)
    VALUES (22,23,'21','','TEXT', 'foobarDescription2');
  INSERT INTO NODE (begin_, end_, parent, tag, type)
  VALUES (25,34,'20','bureau','ELT');
    INSERT INTO NODE (begin_, end_, parent, tag, type)
    VALUES (26,29,'25','code','ELT');
      INSERT INTO NODE (begin_, end_, parent, tag, type, txtval)
      VALUES (27,28,'26','','TEXT', '422');
    INSERT INTO NODE (begin_, end_, parent, tag, type)
    VALUES (30,33,'25','personne','ELT');
      INSERT INTO NODE (begin_, end_, parent, tag, type, txtval)
      VALUES (31,32,'30','','TEXT', 'william');
  INSERT INTO NODE (begin_, end_, parent, tag, type)
  VALUES (35,40,'20','salle','ELT');
    INSERT INTO NODE (begin_, end_, parent, tag, type)
    VALUES (36,39,'35','nombrePlaces','ELT');
      INSERT INTO NODE (begin_, end_, parent, tag, type, txtval)
      VALUES (37,38,'35','','TEXT', '422');
```

2. À l'aide de la classe SaxParser (.java sur Moodle), programmez l'encodage par intervalles begin/end. Cette classe permet de parser un document XML en streaming (noeud après noeud). Modifiez les méthodes startElement et endElement qui sont déclenchées à l'ouverture et fermeture d'un noeud de type element, ainsi que la méthode characters qui est déclenchée lorsque on rencontre un noeud de texte. L'output du programme sera une liste de mises à jour pour la table NODE. INSERT INTO NODE (begin,end,parent,tag,nodtyp) VALUES(1,16,null,'racine','element')
3. Testez le programme sur le document XML concernant les Tweets, et réformulez sur la table NODE trois requêtes de votre choix.
   — Le nombre d'utilisateur de la base de donnée.

```
SELECT COUNT(*)
FROM NODE bdd
WHERE bdd.tag = 'user'
AND bbd.type = 'ELT';
```

   — Le nombre de tweet de la base de donnée.

```
SELECT COUNT(*)
FROM NODE bdd
WHERE bdd.tag = 'tweet'
AND bbd.type = 'ELT';
```

— Les tweets ayant l'hastag 'I<3XML'

```
SELECT *
FROM NODE bdd
WHERE bdd.txtval LIKE '#I<3XML'
AND bdd.tag = ''
AND bdd.type = 'TXT';
```

4. Testez le programme avec le fichier XML http ://www.ins.cwi.nl/projects/xmark/Assets/standard.gz et reportez le temps d'execution.

   Le programme que nous avons créé ne prend pas en compte les attributs. Les modifications apportées au code donné sont les suivantes :
   — Ajout d'une classe Element. Une instance d'Element correspond à un insert qui devra être créé. Element contient donc des attributs tels que begin, parent, end, tag, text, etc.
   — Ajout d'un compteur statique pour la numérotation des noeuds. Ce compteur est incrémenté dans *startElemen*(), *endElement*(), et deux fois dans *characters*().
   — Ajout d'une pile statique d'Element. Le parcours séquentiel d'un XML est adapté à la structure de pile. Dans la méthode *startElement*() la tête de la pile nous donne la valeur de *parent*. Nous empilons ensuite le nouvel Element créé. Dans *endElement*() le compteur nous donne la valeur de *end* pour la tête de la pile. Nous pouvons donc dépiler et créer l'insertion. La méthode *characters*() utilise seulement la pile pour accéder à la tête pour connaître la valeur de *parent*. L'insertion peut ensuite être directement créée.

   Le temps nécessaire pour produire les 3 millions d'insertions est de 3 secondes environ.

5. Modifier le programme afin d'implémenter l'encodage pre/post du document.
   La modification est assez simple. Nous scindons le compteur précédent en deux compteurs *pre* et *post*. Le compteur *pre* est incrémenté lorsque nous empilons un Element (dans *startElement*()), et le compteur *post* est incrémenté lorsque nous dépilons un Element (dans *endElement*()). Nous incrémentons également les deux compteurs une fois chacun dans *characters*().

6. Modifier le programme afin d'implémenter l'encodage Dewey du document.

   Les modifications pour l'encodage Dewey sont assez différentes. Nous n'avons plus besoin de rencontrer la balise fermante d'un élément pour effectuer son insertion. Nous n'avons donc plus besoin d'une pile d'Element. A la place de cette pile, nous gérons le code Dewey par une pile d'entiers, ce qui réduit considérablement la complexité en espace du programme. Nous gardons également en mémoire un compteur de position et la position du dernier élément commencé *lastBegin* et du dernier élément terminé *lastEnd*. Cela nous permet de distinguer dans *startElement*(), quand il faut ajouter un nouveau nombre à la notation

Dewey (si $position = lastBegin + 1$) de quand il faut incrémenter la notation Dewey (si $position = lastEnd + 1$. Dans $element()$ nous enlevons la tête de la pile de Dewey.

7. Enfin, modifiez votre programme pour qu'il puisse répondre à des requêtes XPath de la forme //a avec a une balise quelconque.

   L'algorithme que nous proposons est assez simple. Le problème principal quand on souhaite traiter une requête de type $//a$ est qu'un élément a peut tout à fait contenir un autre élément a qui doit également être capturé. Nous traitons donc le problème de la manière suivante :
   — Nous utilisons une pile. Chaque élément de la pile représente un élément a qui est en train d'être reconnu ;
   — Nous disposons également d'un buffer qui contient le dernier élément a qui est en train d'être reconnu ;
   — Lorsque nous rencontrons une balise ouvrante $< a >$ nous ajoutons le buffer à la pile et stockons la nouvelle balise $< a >$ dans le buffer ;
   — Lorsque nous rencontrons une balise fermante $< /a >$ elle vient clore l'élement a du buffer. Nous pouvons donc écrire le buffer en sortie. Mais cet élément a doit également être ajouté au précédent élément a en cours de reconnaissance (la tête de pile). Cette opération revient donc à faire buffer ← depiler(pile) + buffer.
   — Voici le pseudocode de l'algorithme :

Algorithme 1 : expression //a

```
Entrées : a nom d'élément, F fichier XML ouvert en stream, out
          fichier de sortie
Sorties : Le résultat de la requête XPath //a écrit dans out
pile =[] ; buffer=[] ;
 tant que ligne=lire(F) faire
    si ligne est une balise ouvrante de tag a alors
       si buffer non vide alors empiler(pile,buffer);
       buffer ← ligne ;
    fin
    sinon si ligne est une balise fermante de tag a alors
       buffer ← buffer + ligne ;
        ecrire(out,buffer) ;
        si pile non vide alors buffer = depiler(pile)+ buffer;
       sinon buffer ← [];
    sinon si buffer non vide alors buffer ← buffer + ligne;
fin
```

8. Indiquer comment vous devriez étendre votre programme pour supporter n'importe quelle combinaison des axes child et descendant.

   Pour supporter l'axe child il suffit d'avoir un compteur de profondeur. Un axe child est équivalent à permettre la reconnaissance seulement si la profondeur relative est de +1. Pour supporter une combinaison d'axes nous pouvons simplement faire de la récursion. Pour traiter par exemple $//a//b/child :: c$ nous traitons :

a) $//a$

b) $//b$ avec en entrée la sortie de $//a$

c) $/child :: c$ avec en entrée la sortie de $//a//b$

4 Annexe

4.1 document XML pour les bâtiments

```xml
<batiment>
  <etage>
    <description>foobar1</description>
    <bureau>
      <code>foobar1</code>
    </bureau>
    <salle>
      <nombrePlaces>421</nombrePlaces>
    </salle>
  </etage>
  <etage>
    <description>foobar2</description>
    <bureau>
      <code>422</code>
      <personne>william</personne>
    </bureau>
    <salle>
      <nombrePlaces>422</nombrePlaces>
    </salle>
  </etage>
</batiment>
```

4.2 DTD pour les bâtiments

```
< ! DOCTYPE batiment [
  < ! ELEMENT batiment (etage+) >
  < ! ELEMENT etage (description,(bureau+|salle+)) >
  < ! ELEMENT description (#PCDATA) >
  < ! ELEMENT bureau (code, personne*) >
  < ! ELEMENT code (#PCDATA) >
  < ! ELEMENT personne (#PCDATA) >
  < ! ELEMENT salle (nombrePlaces) >
  < ! ELEMENT nombrePlaces (#PCDATA) >
]>
```

4.3 Code presse

```
create table journal(
  verrou varchar(1) default '1',
  nom VARCHAR(20),
  nomDir VARCHAR(20),
  prenomDir VARCHAR(20),
  constraint PK_journal PRIMARY KEY (verrou),
    constraint CK_journal_locked CHECK (verrou='1'),
    constraint CK_journal_lock_NN check (verrou is not null)
);

create table journaliste (
  idJ VARCHAR(10) PRIMARY KEY,
  nom VARCHAR(20),
  prenom VARCHAR(20),
  pseudonyme VARCHAR(20),
```

```
  anonymisation VARCHAR(3),
  constraint CK_journal_anonymisation check (anonymisation in ('oui','non')),
  constraint CHK_journal_pseudoNom check ((pseudonyme is null) or (nom is null
      and prenom is null))
);

create table article (
  articleID NUMBER(10) PRIMARY KEY,
  titre VARCHAR(100),
  auteur VARCHAR(10),
  corps VARCHAR(4000),
  constraint FK_article_journaliste foreign key(auteur) references journaliste
     (idJ)
);


delete from article;
delete from journaliste;
delete from journal;


insert into journal values('1','Times','Williams', 'William');
insert into journaliste values('j01','Poe', 'Edgar', NULL, 'non');
insert into journaliste values('j02', NULL, NULL, 'Anatole France', 'oui');
insert into article values (1, 'Emmanuel Macron ou Le bon sens', 'j02', 'Mes
     chers concitoyens');
insert into article values (2, 'De la gestion de la crise sanitaire', 'j01', '
     Les pizzerias devraient rester ouvertes');
insert into article values (3, 'Les enfants qui s''aiment', 'j02', ' Les
     enfants qui s''aiment s''embrassent debout
Contre les portes de la nuit
Et les passants qui passent les designent du doigt
Mais les enfants qui s''aiment
Ne sont la pour personne
Et c''est seulement leur ombre
Qui tremble dans la nuit
Excitant la rage des passants
Leur rage, leur mepris, leurs rires et leur envie
Les enfants qui s''aiment ne sont la pour personne
Ils sont ailleurs bien plus loin que la nuit
Bien plus haut que le jour
Dans l''eblouissante clarte de leur premier amour.');


select * from article
where ('Macron' not in titre and 'Macron' not in corps);

select titre, pseudonyme, corps from article join journaliste on article.
     auteur = journaliste.idJ
where anonymisation = 'oui';

select nom, prenom, pseudonyme, count(*) from journaliste join article on
     article.auteur = journaliste.idJ
group by nom, prenom, pseudonyme;
```

4.4 XML Base de donnée : tweets

```
<bddTweet>
  <tweet idT="t6666" idRefUser="u1">
    <date>2020-10-15T18:53:25</date>
```

```
  <body>
    <formatting>
      <fontsize></fontsize>
      <fontcolor></fontcolor>
      <font></font>
    </formatting>
    <language></language>
    <retweets>1</retweets>
    <author idAuthor="u1">
      <name></name>
      <userref></userref>
    </author>
    <content>
      .
      <userref>@exemple</userref>
      absolutely smashed it at
      <hashtag>#mtvlivelockdown</hashtag>
      ! Catch hhim at the official
      <userref>@clubmtvuk</userref>
      after party tonight @ 10pm
    </content>
  </body>
    <reponses>
      <tweet idT="t5665" idRefUser="u2">
      <date>2020-10-16T13:48:32</date>
      <body>
        <formatting>
          <fontsize></fontsize>
          <fontcolor></fontcolor>
          <font></font>
        </formatting>
        <language></language>
        <retweets>0</retweets>
        <author idAuthor="u2">
          <name></name>
          <userref></userref>
        </author>
        <content>
          Nooope!
        </content>
      </body>
      </tweet>
</reponses>
</tweet>
  <tweet idT="t6226" idRefUser="u1">
  <date>2020-10-10T07:06:17</date>
  <body>
    <formatting>
      <fontsize></fontsize>
      <fontcolor></fontcolor>
      <font></font>
    </formatting>
    <language></language>
    <retweets>0</retweets>
    <author idAuthor="u1">
      <name></name>
      <userref></userref>
    </author>
    <content>
      I can assure
      <userref>@Alxxwi</userref>
      Looking around me,
```

```
      <hashtag>#I&lt;3XML</hashtag>
      We use a collection of XML
      <userref>@Cristophe</userref>
      . Its mission is to provide superior technology and expertise
    </content>
  </body>
</tweet>
  <tweet idT="t6336" idRefUser="u3">
  <date>2020-10-01T12:14:53</date>
  <body>
    <formatting>
      <fontsize></fontsize>
      <fontcolor></fontcolor>
      <font></font>
    </formatting>
    <language></language>
    <retweets>0</retweets>
    <author idAuthor="u3">
      <name></name>
      <userref></userref>
    </author>
    <content>
      .
      <userref>@Cristophe</userref>
      Then I summarize the reasons for which it is an absolutely abominable
   film?
      <hashtag>#mtvlivelockdown</hashtag>
      Everything has to be absolutely above-board
      <userref>@Cristophe</userref>
      I needed to talk with someone who was very smart after party tonight @
    10pm
    </content>
    </body>

    <reponses>
      <tweet idT="t7226" idRefUser="u2">
        <date>2020-10-10T08:06:17</date>
        <body>
          <formatting>
            <fontsize></fontsize>
            <fontcolor></fontcolor>
            <font></font>
          </formatting>
          <language></language>
          <retweets>0</retweets>
          <author idAuthor="u1">
            <name></name>
            <userref></userref>
          </author>
          <content>
            This is just to say
          </content>
        </body>
      </tweet>
      <tweet idT="t7227" idRefUser="u3">
        <date>2020-10-10T09:06:17</date>
        <body>
          <formatting>
            <fontsize></fontsize>
            <fontcolor></fontcolor>
            <font></font>
          </formatting>
```

```
            <language></language>
            <retweets>0</retweets>
            <author idAuthor="u1">
              <name></name>
              <userref></userref>
            </author>
            <content>
              I have eaten
              the <hashtag>#plums</hashtag>
              that were in
              the icebox
            </content>
          </body>
        </tweet>
        <tweet idT="t7228" idRefUser="u1">
          <date>2020-10-10T10:06:17</date>
          <body>
            <formatting>
              <fontsize></fontsize>
              <fontcolor></fontcolor>
              <font></font>
            </formatting>
            <language></language>
            <retweets>0</retweets>
            <author idAuthor="u1">
              <name></name>
              <userref></userref>
            </author>
            <content>
               and which
               you were probably
               saving
               for breakfast
            </content>
          </body>
        </tweet>
      </reponses>
  </tweet>
  <tweet idT="t6446" idRefUser="u4" idRetweet="t6666">
    <date>2020-10-17T00:42:35</date>
    <body>
      <formatting>
        <fontsize></fontsize>
        <fontcolor></fontcolor>
        <font></font>
      </formatting>
      <language></language>
      <retweets>0</retweets>
      <author idAuthor="u4">
        <name></name>
        <userref></userref>
      </author>
      <content>
        She was quite aware of her own limitations
        <userref>@Jean</userref>
        Scotland coach Matt Williams is absolutely right
        <hashtag>#howTOuseAbsolutely</hashtag>
        All I know is what I read in the paper,
        <userref>@Alxxwi</userref>
        because of the authority he brings to it
      </content>
    </body>
```

```
  </tweet>
  <tweet idT="t6556" idRefUser="u4">
    <date>2020-10-13T11:23:46</date>
    <body>
      <formatting>
        <fontsize></fontsize>
        <fontcolor></fontcolor>
        <font></font>
      </formatting>
      <language></language>
      <retweets>0</retweets>
      <author idAuthor="u4">
        <name></name>
        <userref></userref>
      </author>
      <content>
        .
        <userref>@Jean</userref>
        I know where to go when
        <hashtag>#COVID19</hashtag>
        need new news. What about all those words and expressions
        <userref>@Cristophe</userref>
        ?
      </content>
    </body>
  </tweet>
  <user idU="u1">
    <nom>Dupont</nom>
    <prenom>Jean</prenom>
  </user>
  <user idU="u2">
    <nom>Dupont</nom>
    <prenom>Christophe</prenom>
  </user>
  <user idU="u3">
    <nom>Mazrie</nom>
    <prenom>Emilie</prenom>
  </user>
  <user idU="u4">
    <nom>Alxxwi</nom>
    <prenom>Dupont</prenom>
  </user>
</bddTweet>
```