Q1: XML and XPath

The table “users” has been altered such that the “address” attribute’s datatype is XML rather than string. To do so, the address column was dropped, using the following command:

```
ALTER TABLE users DROP COLUMN address;
```

The address attribute was then re-added to the table using the following command, with its type as XML:

```
ALTER TABLE users ADD address xml;
```

Using the fact that “email” is the primary key in the users table, the address value for all the tuples in the table was updated with XML content. Below are the 6 UPDATE queries, and a screenshot of the updated schema:

1)
```
UPDATE users
SET address = '<addr country = "Canada">
 <street> 1115 Sherbrooke Ouest </street>
 <city> Montreal </city>
 <province> Quebec </province>
 <postal> H3C 2F0</postal>
 </addr>'
WHERE email = 'bettina.kemme@mcgill.ca';
```

2)
```
UPDATE users
SET address = '<addr country = "Canada">
                  <street> 3212 Stanley</street>
                  <city> Quebec City </city>
                  <province> Quebec </province>
                  <postal> H3A 1W8 </postal>
                  </addr>'
WHERE email = ‘josephakl95@gmail.com';
```

3)
```
UPDATE users
SET address = ' <addr country = "Canada">
                    <street> 3510 Guy</street>
                    <city> Montreal </city>
                    <province> Quebec </province>
                    <postal> H3C 1M7</postal>
                    </addr>'
WHERE email = 'zeyadhazemsaleh@gmail.com';
```

4)
```
UPDATE users
SET address = '<addr country = "Canada">
                <street> 1201 Clark </street>
                <city> Toronto </city>
                <province> Ontario </province>
                <postal> H4D 2QA</postal>
                </addr>'
WHERE email = 'ossama.samir.ahmed@gmail.com';
```

5)
```
UPDATE users
SET address = '<addr country = "Canada">
                <street> 3208 Drummond </street>
                <city> Montreal </city>
                <province> Quebec </province>
                <postal> H3A 2W9</postal>
                </addr>'
WHERE email = 'lucien.george@mail.mcgill.ca';
```

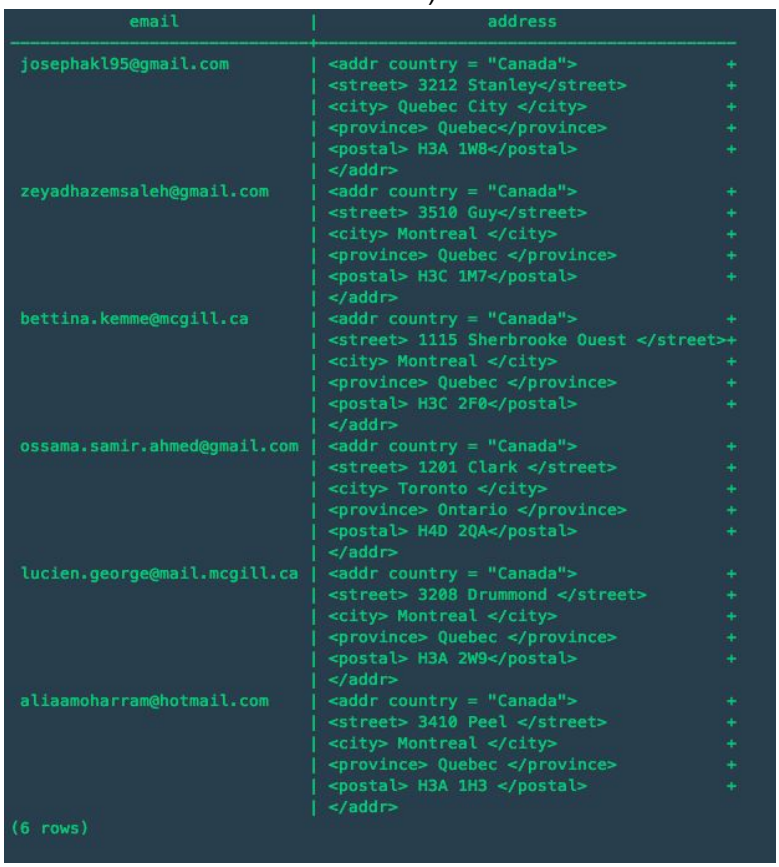
6)
```
UPDATE users
SET address = '<addr country = "Canada">
                <street> 3410 Peel </street>
                <city> Montreal </city>
                <province> Quebec </province>
                <postal> H3A 1H3 </postal>
                </addr>'
WHERE email = 'aliaamoharram@hotmail.com';
```

Description of XML data inserted:
The updated *address* attribute with XML datatype is used to conveniently store frequently requested information about an address such as: country, city, street, province, and postal code. It makes it easier to extract the city/province/street/postal code/country of a specific user[1].

Description of the Altered Relation:
The altered relation - *users* - stores the exact same information as did the old relation. The only difference is the datatype switch of the *address* attribute from string to XML.

[1] *"Specific user"* instead of all users or multiple users who satisfy a certain predicate, because in PostgreSQL the XPATH statement only works on a single tuple. Thus, within the scope of this deliverable XPATH queries are executed on a single tuple. **This information has been relayed to the Professor and has been verified, and XPATH queries operating on a single tuple have been deemed acceptable as it is a restriction posed by the system used (PostgreSQL).*

Result after updating these 6 tuples (this is the result of a selection and projection statement on the email and address attributes[2]):



```
           email            |                address
----------------------------+---------------------------------------
 josephakl95@gmail.com      | <addr country = "Canada">              +
                            | <street> 3212 Stanley</street>         +
                            | <city> Quebec City </city>             +
                            | <province> Quebec</province>           +
                            | <postal> H3A 1W8</postal>              +
                            | </addr>
 zeyadhazemsaleh@gmail.com  | <addr country = "Canada">              +
                            | <street> 3510 Guy</street>             +
                            | <city> Montreal </city>                +
                            | <province> Quebec </province>          +
                            | <postal> H3C 1M7</postal>              +
                            | </addr>
 bettina.kemme@mcgill.ca    | <addr country = "Canada">              +
                            | <street> 1115 Sherbrooke Ouest </street>+
                            | <city> Montreal </city>                +
                            | <province> Quebec </province>          +
                            | <postal> H3C 2F0</postal>              +
                            | </addr>
 ossama.samir.ahmed@gmail.com | <addr country = "Canada">            +
                            | <street> 1201 Clark </street>          +
                            | <city> Toronto </city>                 +
                            | <province> Ontario </province>         +
                            | <postal> H4D 2QA</postal>              +
                            | </addr>
 lucien.george@mail.mcgill.ca | <addr country = "Canada">            +
                            | <street> 3208 Drummond </street>       +
                            | <city> Montreal </city>                +
                            | <province> Quebec </province>          +
                            | <postal> H3A 2W9</postal>              +
                            | </addr>
 aliaamoharram@hotmail.com  | <addr country = "Canada">              +
                            | <street> 3410 Peel </street>           +
                            | <city> Montreal </city>                +
                            | <province> Quebec </province>          +
                            | <postal> H3A 1H3 </postal>             +
                            | </addr>
(6 rows)
```

XPath Queries

As mentioned in footnote 1, the XPATH statement in PostgreSQL only operates on a *single* tuple, and therefore the queries are restricted by this.

1) This query extracts the postal code of the user with email = ‘aliaamoharram@hotmail.com, which is ‘H3A 1H3’:

SELECT (xpath('//postal/text()', (SELECT address FROM users WHERE email = ‘aliaamoharram@hotmail.com’)));

```
cs421=>  SELECT (xpath('//postal/text()', (SELECT address FROM users WHERE email = 'aliaamoharram@hotmail.com')));
     xpath
---------------
 {" H3A 1H3 "}
(1 row)

cs421=>
```

[2] SELECT email, address FROM users;

2) This query extracts the street of the user with email = ‘aliaamoharram@hotmail.com’ which is ‘3410 Peel’ according the inserted data:

SELECT (xpath('//street/text()', (SELECT address FROM users WHERE email = 'aliaamoharram@hotmail.com')));

A screenshot of the complete revised schema is on the following page.

ress	email	fname	lname	dateofbirth	gender	university	rating	picture	phonenumber	add
	josephakl95@gmail.com	Joseph	Akl	1995-04-25	Male	McGill University			+1(514) 445 7873	<addr country = "Ca
nada">	+									
ey</street>	+									<street> 3212 Stanl
</city>	+									<city> Quebec City
province>	+									<province> Quebec</
ostal>	+									<postal> H3A 1W8</p
	zeyadhazemsaleh@gmail.com	Zeyad	Saleh	1994-05-07	Male	McGill University	5		+1(514) 585-6769	</addr>
nada">	+									<addr country = "Ca
street>	+									<street> 3510 Guy</
ity>	+									<city> Montreal </c
/province>	+									<province> Quebec <
ostal>	+									<postal> H3C 1M7</p
	bettina.kemme@mcgill.ca	Bettina	Kemme	2016-01-01	Female	McGill University			Not Available	</addr>
nada">	+									<addr country = "Ca
rooke Ouest </street>+										<street> 1115 Sherb
ity>	+									<city> Montreal </c
/province>	+									<province> Quebec <
ostal>	+									<postal> H3C 2F0</p
	ossama.samir.ahmed@gmail.com	Ossama Samir	Ahmed	1992-10-21	Male	McGill University	4		+1(514) 836-2217	</addr>
nada">	+									<addr country = "Ca
</street>	+									<street> 1201 Clark
ty>	+									<city> Toronto </ci
</province>	+									<province> Ontario
:										

Lucien George: 260571775, Ossama Ahmed: 260549558, Zeyad Saleh: 260556530
Aliaa Moharram: 260572705

Q2) Stored procedure:

Scope:

For this question we decided to implement an “insert and update” procedure, where a user can submit a review about another user, and the procedure will automatically update the other user’s rating. The procedure was implemented using PL/pgSQL.

Creation:

```
CREATE OR REPLACE FUNCTION review_update(_reviewer_email text, _reviewee_email
text, _description text, _rating int) RETURNS void AS $$
DECLARE
user_cursor REFCURSOR;
avg_rating INTEGER;
BEGIN
INSERT INTO reviews VALUES (_reviewer_email, _reviewee_email, _description, _rating);
OPEN user_cursor FOR SELECT AVG(reviews.rating) FROM reviews WHERE
reviews.reviewee_email= _reviewee_email;
FETCH FIRST FROM user_cursor into avg_rating;
CLOSE user_cursor;
UPDATE users
SET rating = avg_rating WHERE users.email = _reviewee_email;
END;
$$ Language plpgsql;
```

Walk through:

The function first inserts the rating, then sets a cursor pointing at the avg(reviews.rating) of the user of interest. It closes the cursor then updates the rating attribute in users accordingly.

Results:

```
cs421=> select * from reviews;
       reviewer_email        |       reviewee_email        |                                                      description                                                      | rating
-----------------------------+-----------------------------+-----------------------------------------------------------------------------------------------------------------------+--------
 lucien.george@mail.mcgill.ca | ossama.samir.ahmed@gmail.com | Ossama was honest about the condition of the item he was selling and the transaction was pretty simple. He was a little bit late though to our meeting. |      4
 aliaamoharram@hotmail.com    | zeyadhazemsaleh@gmail.com    | Zeyad described perfectly the condition of the item and made the transaction process really easy                      |      5
 zeyadhazemsaleh@gmail.com    | lucien.george@mail.mcgill.ca | Lucien was on time                                                                                                    |      5
 zeyadhazemsaleh@gmail.com    | lucien.george@mail.mcgill.ca | terrible experience                                                                                                   |      1
 zeyadhazemsaleh@gmail.com    | aliaamoharram@hotmail.com    | good experience overall                                                                                               |      4
 josephakl95@gmail.com        | aliaamoharram@hotmail.com    | excellent transaction                                                                                                 |      5
(6 rows)

cs421=> select email, rating from users;
            email             | rating
------------------------------+--------
 josephakl95@gmail.com        |
 bettina.kemme@mcgill.ca      |
 zeyadhazemsaleh@gmail.com    |      5
 ossama.samir.ahmed@gmail.com |      4
 lucien.george@mail.mcgill.ca |      3
 aliaamoharram@hotmail.com    |      5
(6 rows)

cs421=> select review_update('ossama.samir.ahmed@gmail.com', 'zeyadhazemsaleh@gmail.com', 'he was a little late', 3);
 review_update
---------------

(1 row)

cs421=> select * from reviews;
       reviewer_email        |       reviewee_email        |                                                      description                                                      | rating
-----------------------------+-----------------------------+-----------------------------------------------------------------------------------------------------------------------+--------
 lucien.george@mail.mcgill.ca | ossama.samir.ahmed@gmail.com | Ossama was honest about the condition of the item he was selling and the transaction was pretty simple. He was a little bit late though to our meeting. |      4
 aliaamoharram@hotmail.com    | zeyadhazemsaleh@gmail.com    | Zeyad described perfectly the condition of the item and made the transaction process really easy                      |      5
 zeyadhazemsaleh@gmail.com    | lucien.george@mail.mcgill.ca | Lucien was on time                                                                                                    |      5
 zeyadhazemsaleh@gmail.com    | lucien.george@mail.mcgill.ca | terrible experience                                                                                                   |      1
 zeyadhazemsaleh@gmail.com    | aliaamoharram@hotmail.com    | good experience overall                                                                                               |      4
 josephakl95@gmail.com        | aliaamoharram@hotmail.com    | excellent transaction                                                                                                 |      5
 ossama.samir.ahmed@gmail.com | zeyadhazemsaleh@gmail.com    | he was a little late                                                                                                  |      3
(7 rows)

cs421=> select email, rating from users;
            email             | rating
------------------------------+--------
 josephakl95@gmail.com        |
 bettina.kemme@mcgill.ca      |
 zeyadhazemsaleh@gmail.com    |      4
 ossama.samir.ahmed@gmail.com |      4
 lucien.george@mail.mcgill.ca |      3
 aliaamoharram@hotmail.com    |      5
(6 rows)

cs421=>
```

Q4) Indexing

Index Query 1: CREATE INDEX cind ON item(category);

The category attribute of the item relation has been chosen as an index because in the application domain, certain required information involves the category attribute, such queries include:

- Getting the average bid done by a specific user on a certain category
    - select avg(temp2.price::numeric::float8) from (select * from (select aid,price from bid where email='aliaamoharram@hotmail.com') as temp1 join auction on temp1.aid=Auction.aid) as temp2 join item on temp2.itemid = item.itemid and item.category='Furniture';

- Getting the categories that the user bids on the most
    - select item.category from
    (select bid.aid, bid.email, auction.itemid from bid inner join auction on bid.aid = auction.aid)x
    inner join item on x.itemid = item.itemid
    where email = 'aliaamoharram@hotmail.com' group by item.category order by count(*) desc;

    *These queries would be sped up as a result of the indexing.

Index Query 2: CREATE INDEX bpind ON bid(price);

The index on bidid combined with the index on its price are strategic for speeding up certain queries involved in a wide scope of applications, in particular point and range queries. Point queries are particularly useful for finding the highest and lowest prices in a certain bid. Range queries however are very useful as they allow for a triage of the seller by bid price and speeds up price sorting. (On a side not, creating an index on the initialprice of an auction wouldn't have been a bad idea either, as the buyers might be looking for a certain item in a certain price range).