Question 5. [8 marks]

Suppose I have a file called nonsense.ddl containing this:

```sql
DROP SCHEMA IF EXISTS rp CASCADE;
CREATE SCHEMA rp;
SET SEARCH_PATH TO rp;

CREATE TABLE Things (  
    A INT PRIMARY KEY,  
    B INT,  
    C INT UNIQUE  
);

CREATE TABLE Junk (  
    G INT PRIMARY KEY,  
    H INT,  
    I INT,  
    FOREIGN KEY (I) REFERENCES Things(A) ON UPDATE CASCADE ON DELETE CASCADE  
);

CREATE TABLE Stuff (  
    D INT,  
    E INT,  
    F INT PRIMARY KEY,  
    FOREIGN KEY (E) REFERENCES Things(C) ON UPDATE RESTRICT ON DELETE SET NULL,  
    FOREIGN KEY (E) REFERENCES Junk(G) ON UPDATE SET NULL ON DELETE CASCADE  
);
```

Part (a) [2 marks]

Suppose I imported this file into PostgreSQL using the command `\i nonsense.ddl` and then a few weeks later the following happened when I tried to access table `Junk`.

```
dbsrv1% psql csc343h-dianeh
psql (9.1.15, server 9.1.14)
Type "help" for help.

csc343h-dianeh=> SELECT * FROM Junk;
ERROR: relation "junk" does not exist
LINE 1: SELECT * FROM Junk;
```

Modify my interaction above so that the `SELECT` statement works.
Solutions:

The table Junk is still defined, but we haven’t referred to it successfully. We can either give a fully qualified name for it:

```
csc343h-dianeh=> select * from rp.Junk;
g | h | i
---+---+---
3 | 2 | 9
6 | 2 | 8
8 | 5 | 9
4 | 1 | 1
(4 rows)
```

or we can set the search path so that we don’t have to:

```
csc343h-dianeh=> set search_path to rp;
SET
csc343h-dianeh=> select * from Junk;
g | h | i
---+---+---
3 | 2 | 9
6 | 2 | 8
8 | 5 | 9
4 | 1 | 1
(4 rows)
```

Part (b)  [2 MARKS]

What is the most important thing that is the same about PRIMARY KEY and UNIQUE?

Solutions:

For both, there can be no duplicates. That is, whether a set of attributes $a_1, a_2, \ldots, a_n$ is PRIMARY KEY or UNIQUE, there can be no two tuples with the same value for $a_1$, and the same value for $a_2$ and \ldots and the same value for $a_n$.

What is one important difference between PRIMARY KEY and UNIQUE?

Solutions:

A table can declare any number of sets of attributes UNIQUE, but it can only have one PRIMARY KEY. Another difference is that the DBMS will / is highly likely to make an index on a PRIMARY KEY, but may not choose to do so for a set of attributes that is merely declared to be UNIQUE.
Part (c) [2 MARKS]
Suppose the tables have been populated as shown below. Modify the data to show the contents of the three tables after this command is executed:

**UPDATE Things SET C = 20 WHERE A = 8;**

<table>
<thead>
<tr>
<th>Things:</th>
<th>Stuff:</th>
<th>Junk:</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>c</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>9</td>
</tr>
</tbody>
</table>

**Solutions:**

There is no change to the tables, because the update is rejected:

csc343h-dianeh=> UPDATE Things SET C = 20 WHERE A = 8;
ERROR: update or delete on table "things" violates foreign key constraint "stuff_e_fkey" on table "stuff"
DETAIL:  Key (c)=(6) is still referenced from table "stuff".

Page 13 of 29
### Part (d) [2 MARKS]

Suppose we began with the same original tables, shown below, but ran a different command. Modify the data to show the contents of the three tables after this command is executed:

```sql
DELETE FROM Things WHERE C = 3;
```

**Things:**

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>9</td>
</tr>
</tbody>
</table>

**Stuff:**

<table>
<thead>
<tr>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>9</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

**Junk:**

<table>
<thead>
<tr>
<th>g</th>
<th>h</th>
<th>i</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**Solutions:**

```sql
csc343h-dianeh=> DELETE FROM Things WHERE C = 3;
DELETE 1
```

```sql
csc343h-dianeh=> select * from Things;
   a | b | c
  ---+---+---
   4 | 2 | 5
   8 | 2 | 6
   1 | 5 | 4
   9 | 8 | 7
   2 | 2 | 9
(5 rows)
```

```sql
csc343h-dianeh=> select * from Stuff;
   d | e | f
  ---+---+---
   3 | 4 | 1
   1 | 6 | 3
   2 | 9 | 5
   2 | 3 | 4
(3 rows)
```

```sql
csc343h-dianeh=> select * from Junk;
   g | h | i
  ---+---+---
   3 | 2 | 9
   6 | 2 | 8
   8 | 5 | 9
   4 | 1 | 1
(4 rows)
```