Task documentation XTD: **XML2DDL in Python 3** for IPP 2014/2015

Name and surname: **Daniel Dusek**

Login: **xdusek21**

Whole script uses only python built-in libraries. Namely `sys`, `os`, `getopt` and `xml.etree.ElementTree`. Script doesn't use any other supportive library/class written by author.

Design of XTD filter

As the task requires, the filter takes XML input file, parses through it and based on its structure, creates a responding DDL file on user specified output. The content of the DDL file contains SQL queries that can be used to generate table structure in database - based solely on XML file's structure.

Library getopt and parameter processing

Imaginary first base of the project is to handle and process parameters that can be given on filter calling by user. For this very purpose served the standard `getopt` library which simplified the parameter processing and took care of possibly false variations of parameters.

Libraries os, sys

Two the most basic libraries used in this project were library `os` that provides script with various operating system functions and interfaces. Notable function is definively `os.path.realpath()` that significantly simplifies creating a path to file that is on input or the output. Function takes filename as a parameter and returns the realpath from the root, independently on the type (*relative* or *absolute*) path that has been given to it.

Second mentioned library `sys` was in the project very helpful when it came to working with *parameters* or *program terminations*, since it brings system-specific parameters and functions to the script.

Datatype recognition - implementation

Since task specification requires proper recognition of data types that can be found between two elements (as *text value*) or in the attribute, it had to be implemented. There are two pairs of functions in the script, both named `makeValuenumber` and `returnTypeName` with postfixes `Text` and `Attr`. As the names in this moment suggest, they are used to identify data type within text or attribute.

The algorhitm on which are recognition functions based is quite simple. At first, both of them receives string which they instantly convert to lower-case. Then they compare the string with values acceptable by type BIT and if case it fits, they return code number signalising that type was recognized. If the recognition fails in the first step, functions proceed to step number too, therefore trying to parse as a `FLOAT`. In case of failure, trial to parse to `FLOAT` is done, in case of failure, it is considered string (NVARCHAR or `NTEXT`, depends on function).

Second pair of functions then recognizes the name of data type, which is printed on the output.

Implementation of -a and –header switches functionability

Switch `-a` is implemented as one-conditional addition to the rest of working code. If the parameter -a is set, it skips the part where data from attributes are extracted and based on that are created columns.

As it goes for `--header` switch it is once again a one conditional additing to the output string generation, where data from `--header` switch are included in the output before the rest of it.

Implementation of -b and –etc switches functionability

Switches `-b` and `--etc` were a bit more complicated in ways of implementation.

Given by task specification, `--etc` switch reduces the maximal count of columns that will be created from same name subelements to be less than `n` that is specified by user that runs the filter. This is in the code realized by condition and loop that every iteration over child elements checks if the `n` number was exceeded or not. In case it was, loop deletes all the generated previous elements of given name in parent element and creates a new one named as is specified by task specification.

Similar scenario works for `-b`, script completely ignores other occurences of the element with same name.

Conflicts

Since it is specified that any column generated from subelement will be name in format `subelement_id`, if attribute named `subelement_id` occurs, conflict occurs as well. This is handled by program termination with exit code `90`.

Parameter -g implementation

Switch `-g` that was supposed to convert DDL to XML representing relations between them was not impemented because of poor time management reasons.