Dokumentace úlohy XTD: XML2DDL v PYTHON 3 do IPP 2012/2013

Jméno a Příjmení: Martin Krippel

Login: xkripp00

Úvod

Tento dokument popisuje návrh a způsob implementace projektu (zadání XML2DDL) do předmětu IPP v skriptovacím jazyce Python 3.

Návrh

Návrh se zabývá zpracováním vstupního XML souboru a jeho transformováním na SQL příkazy, respektive výstupní XML soubor, dle zadaných parametrů.

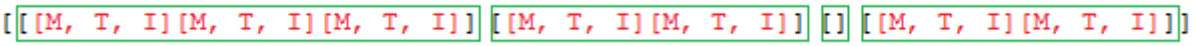
Vstupní XML je nejprve zpracováno pomocí knihovny `ElementTree`. Pomocí této knihovny vznikne stromová struktura popisující XML. Z této struktury jsou následně získávány potřebné údaje pomocí funkci z knihovny `ElementTree` a ukládány do polí. Ze získaných údajů se dají vytvořit SQL příkazy i XML soubory, proto je zpracování vstupního XML stejné pro oba případy.

Výsledný výstup závisí od přepínače `-g`, podle něj se vytváří XML soubor nebo SQL příkazy, zároveň i od přepínače `--output=filename`, který určuje kam se vypíše výsledek.

Implementace

Program začíná načtením a zpracováním parametrů, využívá funkci `getopt`. Po načtení parametrů, zkontroluje případné kolize mezi nimi.

Následně zpracuje vstupní XML soubor pomocí funkce `parse_xml`. Tato funkce ze stromové struktury XML souboru, nejprve získá jména všech tabulek, které je třeba vytvořit. Potom pomocí tohoto pole prochází všechny uzly stromu s danými jmény a získává údaje potřebné ze zadání. To znamená, že vytvoří pole pro atributy elementů, pole jmen podelementů (dále o něm budu mluvit jen jako o poli podelementů) a pole textových elementů daných XML elementů (dále jen pole hodnot). Jednotlivá pole mají strukturu:



Obrázek č. 1 - struktura pole pro data z XML souboru

kde, *M* je jméno sloupce, *T* jeho typ a *I* je index, který se nachází jen v poli podelementů. Celé pole (obr. 1) se skládá z takového počtu prvků (obr. 1 - části označené zeleným obdélníkem), kolik má být tabulek. Umístění těchto prvků odpovídá pořadí jmen v poli jmen tabulek. A každý tento prvek se skládá z polí (obr. 1 - červený text) obsahujících *M*, *T*, případně *I* (samozřejmě, pokud daný element nemá žádný z požadovaných údajů, tak je prvek dané tabulky prázdný (obr. 1 - třetí zelený prvek)). Funkce `parse_xml` vrací čtyři pole: pole jmen tabulek, pole atributů, pole podelementů a pole hodnot.

Tato pole se předají jako parametry funkcím `prep_g` (v případě zadaného prepínača `-g`) nebo `vytvor_sql`. Tyto funkce mají i další parametry v závislosti na zadání.

Funkce `vytvor_sql` se rozvětvuje na dvě hlavní větve podle přepínačů `-b` a `--etc`. Na základě zadání podle těchto parametrů se zpracovávají pole s údaji a vytvářejí se SQL příkazy. Po vytvoření všech SQL příkazů se vypíší a program se ukončí.

Funkce `prep_g` pracuje pouze s polem podelementů. V případě přepínače `--etc` pole upraví na základě hodnoty. Následně je společná část pro přepínače `-b` a `--etc`, kde se odstraní prvky se stejným jménem. Poté, pomocí více funkcí, se vytvoří tabulka závislostí, a naplní se. Naplnění tabulky závislostí: pro každou tabulku se program podívá, zda má tabulka nějaký cizí klíč. Pokud ano, vytvoří se vztah N:1 mezi těmito tabulkami a zároveň se vytvoří symetrický vztah 1:N. Pokud tabulka, na kterou ukazuje cizí klíč má také nějaký cizí klíč, tak se postup opakuje. Toto provádí rekurzivní funkce `rek` pro každou tabulku a každý cizí klíč. Vztahy 1:1 se doplnily při inicializáci a vztahy N: M doplní místo ostatních nevytvořených vztahů. Nakonec se vypíše nový XML soubor a program se ukončí.

V případě nějaké chyby se program ukončuje se zadanými chybovými kódy a na standardní chybový výstup vypíše funkce `chyba` chybové hlášení.

Testování

Testování probíhalo nejprve na jednoduchém XML souboru ze zadání. Pro pokročilejší testy byly použity zveřejněné referenční testy, při kterých se výsledky shodovaly. Vlastní testy na XML souboru s vnořenými elementy byly také uspokojivé, ale bez referenčního řešení se na ně nedalo spoléhat.