Liszkay Béla – Nagy Elemér Károly

Cinege – bibliográfiai és példányrekordok szűrése*

Egy könyvtár életében a bibliográfiai és példányrekordok formátuma számos alkalommal változik, akarva vagy akaratlanul. Évtizedekig megőrzött, változatlan formátum esetén is előfordulnak formai és tartalmi hibák, nem is beszélve a konverziós hibákról, amelyek a kereshetőséget és a karbantarthatóságot jelentősen megnehezíthetik. A cikkben a BME OMIKK-ban és az Universitätsbibliothek der Freie Universität Berlinben is alkalmazott, a BME OMIKK-ban fejlesztett szabad, Cinege nevű szoftver azon részeit mutatjuk be, amelyek segítségével a bibliográfiai és a példányrekordok aktuálisan alkalmazott formátumától való eltérést szűrhetjük le parancssoros eszközökkel.

Bevezetés

Azokban a könyvtárakban, ahol hosszú időn keresztül dolgoznak saját építésű vagy üzemeltetésű adatbázisokkal, azok formátuma előbb vagy utóbb módosul, egyrészt az adatbázisok technikai hátteréül szolgáló technológiák és szoftverek fejlődése következtében, másrészt az adatformátumot érintő szervezeti, személyi, törvényi[1] változások, valamint szabványok módosulása miatt. Mindkétféle változás következménye, hogy a régi adatokat konvertálni kell, de az adatok mennyisége és a rendelkezésre álló erőforrások miatt erre csak a gépi konverzió alkalmas.

A konverzió problémái

A konverzió során tipikusan előforduló, gyakori probléma a hibás ki-, illetve bemenő rekordok kezelése. Ha a konverziós programunk nincs felkészítve kimondottan a hibás rekordok kezelésére, legjobb esetben is azzal számolhatunk, hogy a hibás bemenő rekordokból hibás kimenő rekordok készülnek. Gyakran előfordulhat az is, hogy ilyenkor a konverziós program hibaüzenettel leáll, elrontja az adatállomány egy részét (ha nem veszszük észre időben, akár visszavonhatatlanul is), vagy szélsőséges esetben letörli az egész adatbázist. Hibás adatok konverziójánál a hibák száma jellemzően közel exponenciálisan nő.

Általánosságban elmondható, hogy minél rugalmasabb egy rekordformátum, azaz minél több értéket, illetve értékkombinációt enged meg, annál több hiba kerülhet, és kerül az adatbázisba. A hibák egy részére léteznek beépített szűrők, amelyek az adott hibás rekordokat nem engedik elmenteni, vagy a beviteli program figyelmeztet a hibára. Az viszont nem várható el egy ellenőrző programtól, hogy minden lehetséges hibakombinációra létezzen ilyen jellegű figyelmeztetés. Mivel a rendszeren kívülről származó konvertáló programok rendszerint más felületen keresztül érintkeznek az adatbázissal, a konverterek csak kivételes esetekben működnek együtt a beépített szűrőkkel.

Az adatállományok konvertálást követő ellenőrzésére szolgálnak az utólagos szűrési eljárások. Az írásunkban bemutatott *Cinege* szoftver az utólagos szűrésekre hatékonyan alkalmazható eszköz.

Adatszűrési megközelítések

Az adatszűrési megközelítések közül esetünkben három szempont szerint osztályozzuk a kézi vezérlést nem igénylő adatszűrőket: adatforrás, hibareakció és optimizmus szerint.

Az adatszűrők *adatforrás* szerint négy csoportra oszthatók, ha az XML-t külön kategóriaként kezeljük, és nem a szövegfájlok közé soroljuk:
- szövegfájlból dolgozó szűrő,
- adatbázisból dolgozó szűrő,
- bináris fájlból dolgozó szűrő,
- XML fájlból dolgozó szűrő.

* A 2006. április 19−21. között Miskolcon megrendezett *Networkshop* konferencián elhangzott előadás szerkesztett változata.

Az adatszűrőket *hibareakció* szerint három csoportba oszthatjuk:
- hiba esetén leálló szűrő,
- hiba esetén automatikus javítást megkísérlő szűrő,
- hiba esetén a hibás rekordot kihagyó szűrő.

Az adatszűrőket *optimizmus* szerint két csoportba oszthatjuk:
- pozitív szemléletű – a rekordokat alapvetően jónak tekintő szűrő,
- negatív szemléletű – a rekordokat alapvetően hibásnak tekintő szűrő.

Természetesen egy-egy adatszűrő a különböző típusú hibákra eltérően reagálhat. Általában a konverter használhatóságát javítja, ha az adatszűrő reakciója az adott hibára beállítható.[2] Például a klasszikus „hullámos õ” és „kalapos ô” automatikusan magyar „ő”-re cserélendő, viszont a több címmel (egynél több 245/a mezővel) rendelkező rekordokat ki kell hagyni a konverzióból. A hibás rekordok azonosítóját mindkét esetben fel kell jegyezni a rekordot utoljára módosító katalogizáló azonosítójával együtt.

Mivel a programozásban agresszív, illetve defenzív hibakezelést különböztetünk meg, a konverterek és a szűrők is lehetnek agresszívek, illetve defenzívek. Az agresszív hibakezelés lényege, hogy amint hibát találunk, azonnal a lehető legrészletesebb hibaüzenetet produkáljuk, és leállítjuk a programot, vagy legalább az adott rekord konverzióját. A defenzív hibakezelés lényege, hogy a hibát megpróbáljuk elrejteni, illetve behelyettesíteni, remélve, hogy az eredmény így is megfelelő lesz. E két szemléletbeli megközelítés nagyon hasonló a hibareakció szerinti csoportosításhoz, de annál ismertebb a programozók között.

A Cinege bibliográfiai és példányrekord szűrői adatbázisból dolgoznak, a rekordokat alapvetően hibásnak tekintik, és a hibás rekord azonosítóját regisztrálják. Az adatokat más formátumból letöltő programrészek (Downloader.java) hiba esetén ezenfelül még rekordok kihagyását is kezelik, azaz ezerhárom bemeneti (ebből három hibás) rekord esetén a kimeneti adatbázisban csak ezer rekord lesz, de azok mind ellenőrzöttek. A szűrők rugalmasan, parancssorban megadott feltételek alapján működnek, és sokkal nagyobb támogatást kínálnak a hibás rekordokat meghatározó szabályok definiálására, mint a hibátlan rekordok meghatározására. Emiatt a hibásnak nem talált rekordokat tekintik jó rekordoknak. Azaz, a Cinege agresszív és pesszimista típusú alkalmazás.

Bibliográfiai, példány- és egyéb rekordok

A gyakorlatban elterjedt relációs adatbázison[3] alapuló megközelítés szerint az adatbázisban különböző típusú (bibliográfiai, adminisztratív, példány-, olvasói, kölcsönzési, előjegyzési stb.) rekordok (később még lesz róluk szó) találhatók a különböző táblákban, a rekordok közötti kapcsolatot vagy a rekordokban tárolt azonosítók (*1-1, 1-N* kapcsolatok), vagy a kereszthivatkozási táblákban tárolt azonosítók (*1-1, 1-N, M-N* kapcsolatok) létesítik. A rekordok azonosítását kulcsok (a könyvtári gyakorlatban rendszerint azonosító számok vagy vonalkódok), a rekordok közötti kapcsolatokat pedig távoli vagy idegen kulcsok (foreign keys) oldják meg. Ez utóbbi alkalmazásával az adatbázis-kezelő képes automatikusan kitörölni a törölt bibliográfiai rekordhoz tartozó példányadatokat, vagy megtagadni a bibliográfiai rekord törlését, ha van hozzárendelt példányrekord. Gyakran előfordul, hogy ezek alkalmazása jelentősen csökkenti a rendszer rugalmasságát, sok alkalmazás eleve nem is használja ezeket a távoli kulcsokat, és az adatok közötti konzisztenciát az alkalmazás tartja fenn, több-kevesebb sikerrel. Sem a távoli kulcsok, sem az elsődleges kulcsok nem működnek hatékonyan abban az esetben, ha a rekordokat nem az adatbázis által ismert rekordformátumban, hanem pakolt[4] formában tároljuk. Ebben az esetben a kapcsolatokat valamilyen tömörített formában tároljuk, amihez gyorsító tárként funkcionáló, a rekordokat pakolatlan formában tároló táblák is tartoznak, amelyeket időnként újraépítünk. A pakolt mezők tartalmához általában csak külső programokkal, legjobb esetben is csak tárolt eljárásokkal férünk hozzá. A pakolásnak nagy hátránya, hogy mivel az adatbázis-kezelő nem ismeri a pakolás szabályait és feltételeit, nem is tudja ellenőrizni őket. A pakolás nagy előnye, hogy mivel az adatbázis-kezelő nem ismeri a pakolás szabályait és feltételeit, bármit belerakhatunk, nem csak azt, amit az adatbázis-kezelő ismer vagy támogat. Példa a pakolásra: amikor egy időpontot (mondjuk, a kölcsönzés idejét) egyetlen hosszú számként tárolunk (pl. "20050908122339"). Hátránya, hogy ebben a formában az adatbázis-kezelő nem képes a szabálytalan időpontot észrevenni ("20050932122339"), hiszen ez egy létező szám. Előnye, hogy egyszerűbben tudjuk kezelni a rekordokat

("select count(*)/85 as daily_average_number_of_ loans_in_last_semester, substr(loan_time, 9, 2) as hour from loans where substr(loan_time, 1, 6) between '200602' and '200607' group by substr(loan_time, 9, 2);"),
és hogy nincsenek konverziós problémák például a MySQL és PostgreSQL között, illetve hogy tárolhatunk itt relációelméleti szempontból más mezőbe illő adatokat is (pl. az ismeretlen kölcsönzési időpont „0”). A pakolást ügyesen kell alkalmaznunk ahhoz, hogy a lekérdezéseket egyszerűbbé tegyük, és ne nehezebbé vagy lehetetlenné.

Az adatbázisokat matematikai szempontból megközelítve egy jó adatbázis nem tartalmaz redundáns információkat, sem olyan információkat, amelyeket több különböző útvonalon lehet elérni. Ezeket a kritériumokat normál formáknak hívják, és megtiltják például azt, hogy egy olyan táblában, ahol a bibliográfiai rendszerszám szerepel, az a pakolt HUNMARC rekordban is szerepeljen, jóllehet ez könyvtárosi szempontból logikus és szükséges feltétel. Ismerve a könyvtári adatbázisokat, általánosságban elmondható, hogy nem „szép szerkezetű” vagy „Boyce-Codd normál formájú” adatbázisok, viszont a célnak tökéletesen megfelelnek.

A pakolás önmagában nagyon hasznos eljárás, de megvan az a hátránya, hogy ha nincs megfelelő szoftvertámogatás, nem tudunk érdemben hozzáférni az adatainkhoz. Az indexeket és az egysoros katalóguscédulát (kivonat, rövid dokumentumleírás) éppúgy a pakolt rekordokból kell előállítani, mint a bibliográfiai rekordok közötti kereszthivatkozási táblákat. A pakolt mező kezelésének legnagyobb problémája, hogy – mivel az adatbáziskezelő rendszer nem ismeri a struktúráját – nem képes azt ellenőrizni, vagy abból bonyolultabb feltételek szerint szűrni. Egy jól strukturált, pakolatlan táblából SQL parancsokkal gyorsan és pontosan tudunk információt kinyerni, pakolt táblák esetén ez nem feltétlenül van így. Ideális esetben a pakolással SQL-ben jobban kezelhetővé tesszük az adatbázist, rossz esetben nagyon nehézzé vagy praktikusan lehetetlenné. A jól strukturált, pakolatlan táblába formailag hibás rekordot nem is tudunk beszúrni, míg pakolt mezőbe ez minden további nélkül megtehető. A pakolt mezők tartalmát az alkalmazásnak kell ellenőriznie, tehát egy olyan konverterrel, amely az alkalmazást megkerülve közvetlenül az adatbázisba dolgozik, szinte bármilyen hibás rekordot betölthetünk.

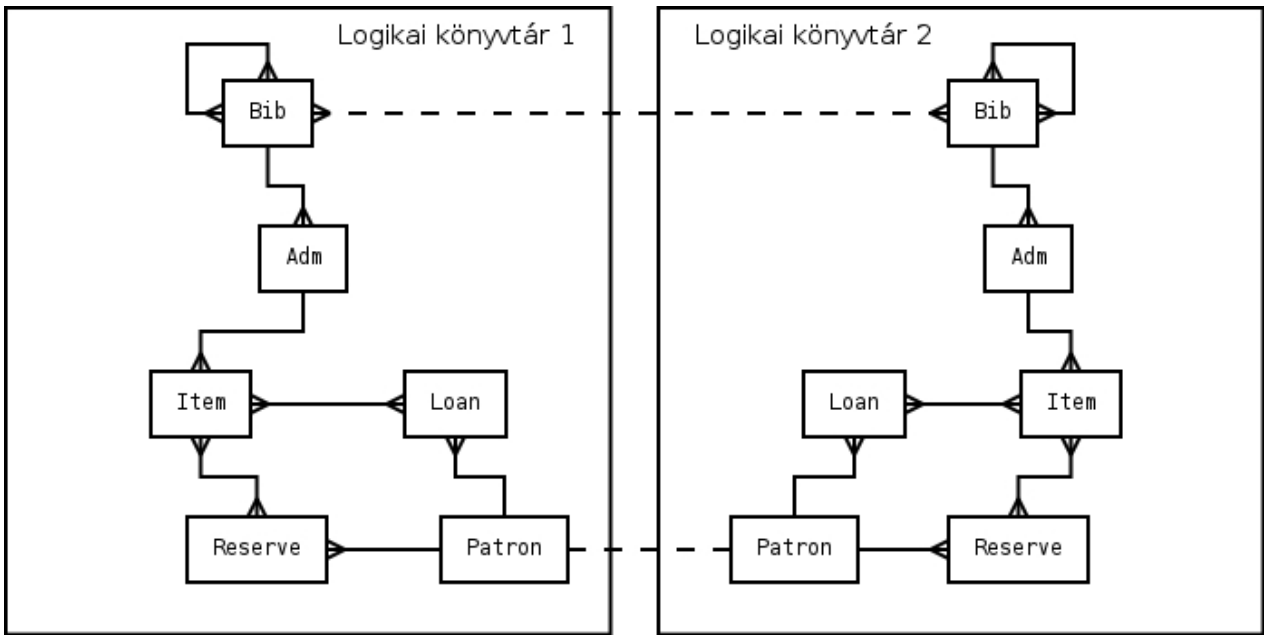
A bibliográfiai rekordokhoz gyakorlatilag tetszőleges számú, tetszőleges mezőkódú, indikátorú, nyelvű, karakterkészletű stb. mező tartozik, ezen belül hasonlóan tetszőleges számú és típusú almező lehetséges, amelyeknek adott esetben a sorrendje is számít. A legtöbb nem bibliográfiai rekordtípus általában csak meghatározott számú és definiált tartalmú mezőt alkalmaz.

Könyvtári rendszerekben általában a bibliográfiai,[5] példány- és olvasói rekordok mellett külön táblákban tárolják a kölcsönzési, előjegyzési, adminisztratív stb. rekordokat. Optimális esetben a bibliográfiai rekordnak az összes, adott kiadványra egyaránt vonatkozó adatot (cím, szerző, nyelv) kellene tartalmaznia. Az adminisztratív rekordnak az egy bibliográfiai rekordhoz tartozó összes példányra, vagy azok egy csoportjára közösen jellemző, a bibliográfiai rekordban nem szereplő adatokat kellene tartalmaznia, a példányrekordnak a csak az adott példányra jellemző adatokat (egyedi vonalkód, raktári jelzet, kölcsönzési státus, állapot). A könyvtári gyakorlatban nem mindig alkalmazzák ezt a kategorizálást, előfordul például, hogy a példányra vonatkozó raktári jelzetet a bibliográfiai rekordban és a példányrekordban is tárolják, sőt van, ahol nincs adminisztratív rekord.

Az egymással hálózatosan összekötött egyes bibliográfiai rekordokhoz *0-1-N* (nulla, egy vagy több) adminisztratív rekord, az adminisztratív rekordokhoz *0-1-N* példányrekord, a példányrekordokhoz *0-1-N* kölcsönzési rekord, az olvasói rekordokhoz *0-1-N* kölcsönzési rekord, az előjegyzésekhez *1* olvasói rekord és *0-1-N* példányrekord vagy *0-1-N* bibliográfiai rekord tartozik, bonyolult szerkezetet alkotva. Ezt a struktúrát tovább bonyolíthatja, amikor egy könyvtári rendszerben több könyvtár, alkönyvtár, gyűjtemény, virtuális könyvtár, fiók, telephely stb. található, aminek következménye, hogy egy vagy több az említett táblákból több adatbázisban is szerepel, amely adatbázisok között ráadásul további kapcsolatok is lehetnek *(1. ábra)*

Szűrők, konverterek

Egy ilyen, meglehetősen bonyolult rendszerből szeretnénk időnként hibalistákat és munkalistákat kinyerni. A rendszeresen szükséges hibalisták előállítására szolgáló eljárások tipizálhatók, hétről hétre futtathatók. Például a „példány ára” mező tartalma egy szám, és esetleg egy „Ft”, „Ft.” vagy „.-” utótag kell, hogy legyen. Egyszer megírjuk rá a programot, betesszük a szerver időzítőjébe, és az hetente automatikusan postázza a javításra jogosult munkatársnak a hibalistát.



1. ábra **Könyvtári relációs adatbázisok sematikus struktúrája**

Egyszeri esetben és rövid határidővel bonyolultabb a feladat, mert ha nem egészen pontos a specifikáció, az csak a lista elkészülte után derül ki, és gyakorta további eljárások végrehajthatóságát lehetővé tevő kimeneti formátumban várják az eredményt (pl. xls-ben, mert az lehetőséget kínál a további kényelmes rendezésre, elrejtésre, formátumbeállításokra, autoszűrésre).

A munkalisták bizonyos könyvtári munkafolyamatok támogatásához szükségesek, és a legritkább esetben gondoltak rá a könyvtári alkalmazás tervezői. Egy példa: a bibliográfiai rekordokban lévő URL hivatkozások példányonkénti átlagszáma, ETO kód szerinti bontásban. Egy másik példa: az elmúlt három hónapban létrehozott összes bibliográfiai rekord – és a hozzá tartozó összes példányrekord összes mezőjét tartalmazó – listája, kivéve azokat a példányrekordokat, amelyek a „BMEA2” alkönyvtárban vannak, vagy nem tartozik hozzájuk forintszámla (más pénznem megengedett), vagy a státusuk „11”, vagy a gyűjteményük „REF”, vagy ha a bibliográfiai rekorddal azonos című, de más kiadási évű és legalább három hónapos rekord már szerepel az adatbázisban, vagy ha csak olyan példány tartozik az adott bibliográfiai rekordhoz, amit az előbb kiszűrtünk.

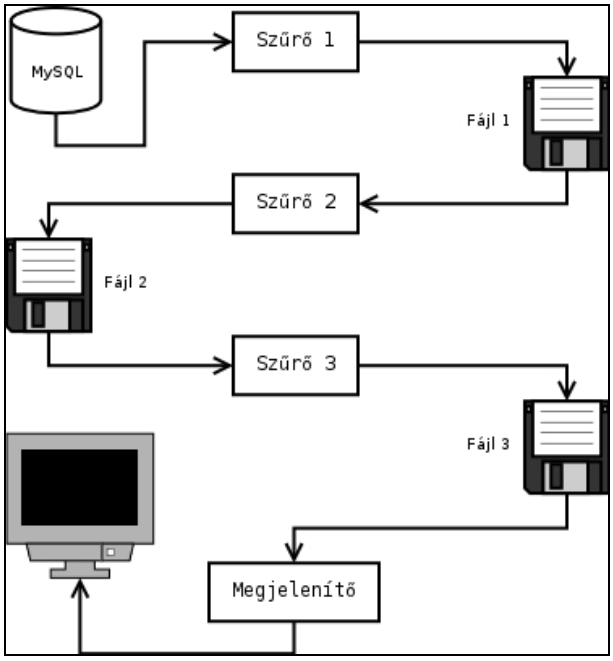
Az adatbázis adatállományának folyamatos minőségjavító karbantartásához elengedhetetlenek a munka- és hibalisták. Szeretnénk olyan konverteket és szűrőket alkalmazni, amelyekkel az egész adatbázist tudjuk – meglehetősen bonyolult feltételek szerint is – szűrni, lehetőleg a szerveren és adott esetben munkaállomásunkon, laptopunkon is, Excel vagy Excelbe konvertálható kimenettel.

Szűkebb értelemben a szűrők olyan programok, amelyek a bemeneti rekordok egy részét eldobják, a többit pedig változatlan formában kiadják a kimenetükön. Szűkebb értelemben a konverterek az összes bemeneti rekordot egyformán módosítják, majd a kimenetükön továbbítják. Tágabb értelemben a szűrők a bemeneti rekordok egy részét vagy azok elemeit eldobják, az el nem dobott elemeket és rekordokat pedig a kimenetükre továbbítják. Tágabb értelemben a konverterek a rekordok egy részét vagy azok elemeit módosítják vagy eldobják, az el nem dobott és esetleg módosított elemeket és rekordokat pedig a kimenetükre továbbítják.

Szűkebb értelemben tehát a szűrő és a konverter két teljesen különböző fogalom, tágabb értelemben viszont a szűrők olyan konverterek, amelyek az eldobáson kívül más módosítást nem végeznek, azaz az alkalmazásuk eredménye nem egy megváltozott rekord, hanem valamilyen, a rekordból készült kivonat. Természetesen a szó szoros értelmében a kivonat is egy megváltozott rekord, felhasználás szempontjából mégis el tudjuk különíteni a kivonatokat és a megváltozott rekordokat, azaz a szűrőket és a konvertereket, mert az alkal-

mazás szempontjából a kivonat csak a rekord azonosítására szolgál.

A szűrők és konverterek rugalmasságát növeli az egymás után fűzhetőség, azaz a csővezeték (pipeline) szervezés *(2. ábra)*. Ha például a konverterünk csak egyetlen karaktert képes egy másikra kicserélni, de önmagával összefűzhető, akkor ugyanazt a konvertert először kalapos ô-re, majd hullámos õ-re állítva, és a két konvertert összefűzve, egy mindkét karaktert kicserélő konvertert kapunk. Az összefűzés alapfeltétele, hogy az *N*. szűrő kimenete az *N+1.* szűrő bemenete lehessen, ideális esetben az összes konverter kimenete az összes konverter bemenete lehessen. A könyvtári gyakorlatban ez nagyon nehezen valósítható meg, mert elválnak egymástól a bibliográfiai, adminisztratív, kölcsönzési, előjegyzési, példány- stb. rekordok és formátumaik. Az is megoldást kínál, ha az összes, bibliográfiai rekordokkal dolgozó konverter kimenete az összes bibliográfiai rekordokkal dolgozó konverter bemenete is lehet, és van olyan konverter, amely ezt a formátumot adminisztratív, kölcsönzési stb. formátumba képes konvertálni.



2. ábra **Csővezeték szerverezésű szűrők**

Ennek egyik legpraktikusabb módja, ha olyan formátumot használunk, ahol minden rekord első mezője valamely azonosító (például bibliográfiai rendszerszám vagy a példány vonalkódja), és a szűrők ezen rekordok egy részét kiszűrik (nem másolják át a bemenetükről a kimenetükre), a konverterek pedig az első mező után szúrnak be új „oszlopokat”, amelyek hozzáadott információkat tartalmaznak.

Cinege

A *Cinege* tisztán Javában[6] írt, Unicode[7] támogatású, nyílt forráskódú, platformfüggetlen és szabad[8] szoftver. Fejlesztésénél elsősorban a BME OMIKK igényeinek próbáltuk megfelelni, de hamarosan kiderült, hogy más könyvtárak, mint például az UB-FUB[9] vagy a PE-KK[10] is kiválóan tudja használni helyi feladatok elvégzésére. A programrendszer agresszív és pesszimista szemléletmódja lehetővé teszi a könyvtári adatbázisokban évek óta keresett hibák „kibányászását”. Az egyik alkalmazás során egy UTF8-as[11] Oracle adatbázisból „melléktermékként” előkerültek olyan hibás rekordok, amelyek szabálytalan UTF8 karaktereket tartalmaztak. Korábban egy verziófrissítésnél napokig keresték a hibákat eredménytelenül, mivel a verziófrissítő csak az utolsó, ötvenedik hibás rekord számát írta ki, mielőtt leállt.

A Cinege kialakításánál alapvető szempont volt, hogy könnyen lehessen bármilyen adatbázishoz illeszteni (JDBC-t[12] és karakterláncokban tárolt előfordított lekérdezéseket használ), ám mivel a szabad és sok platformon futó MySQL[13] adatbázishoz fejlesztettük, így a legcélszerűbb adatforrás egy MySQL adatbázis. Tartalmazza azt a Downloader-osztályt, amely Aleph[14] 505.14, illetve 505.12 alatt futó Oracle-ből[15] képes automatikusan letölteni a szűrésekhez használt táblákat. A bibliográfiai rekordok a BibUnits tábla BibData mezejében, CSV-be[16] pakolva helyezkednek el, MARC-hoz[17] hasonló (mezőkódokkal, almezőkódokkal és indikátorokkal tagolt, ismétléseket megengedő) formátumban. A többi rekord összes mezője pakolatlanul szerepel, kivéve a dátum és idő együttes tárolására szolgáló mezőket. A példányrekordok például az „Items” táblában helyezkednek el. Fontos jellemző, hogy a Cinegében nincsenek külön táblában az archivált és a kurrens események (mint például az Aleph esetén a Z36 és a Z36H táblákban), és az egyébként tipikusan külön könyvtárban tárolt adatok is egy könyvtárban vannak összeolvasztva. A szerzők meggyőződése, hogy a processzor- és memóriaárak annyira leestek,[18] az indexelés technológiája pedig annyira fejlett, hogy ma már szükségtelen külön adatbázisokra és táblákra bontani a rendszerünket, ha az megfelelően van megszervezve és indexelve. Erre a legegysze-

rűbb bizonyíték, hogy a Pentium III-as laptopon dolgozni lehet a teljes BME OMIKK adatbázissal.

A reguláris kifejezések

A reguláris kifejezések (regular expressions) az 1940-es évek óta vannak jelen a műszaki tudományok egy részében, és kb. 1960 óta az informatikában. Az unixos világban elterjedt „grep" parancs talán a leghíresebb reguláris kifejezést támogató program.[19] Segítségével egyszerűen megfogalmazhatunk például olyan feladatokat, mint a dupla szóközt tartalmazó mezők „[ ][ ]", a kötőjelet vagy aláhúzást tartalmazó mezők „[-_]" vagy a „Szerk"-kel kezdődő és nem szóközre végződő mezők „^Szerk.*[^]$", vagy az elmúlt ötvenhat év valamelyikét tartalmazó mezők „((19[5-9][0-9])|(200[0-5]))" kiszűrése. Bár alkalmazásukat nem egyszerű megtanulni, és nehéz felfedezni a hibás kifejezéseket, tömörségük és nagy leíró erejük miatt kifejezetten alkalmasak a parancssoros szűrők, illetve konverterek paraméterezésére. Nem véletlen, hogy a talán mindmáig legnépszerűbb „C" programnyelv mellett például a PERL-ben,[20] a Javában és az AWK-ban[21] is megtalálhatók. Azok a reguláris kifejezésekhez hasonló, jóval kisebb leíró erejű és egyszerűbb kifejezések, amelyek az AnyX[22] és DOS rendszerhéjakban[23] is értelmezve vannak, mint például a kérdőjel és a csillag, kisebb leíró erejűek, mégis nagyon elterjedtek és ismertek.

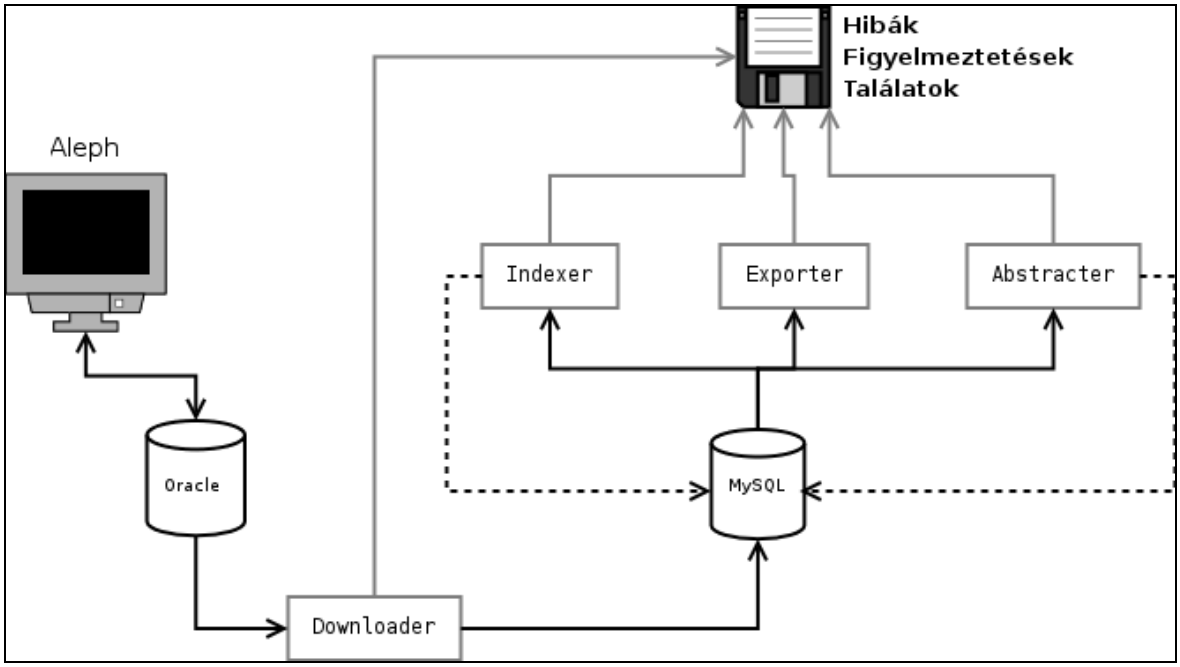
Ellenőrzés Cinegével

A Cinegében az ellenőrzésre jelenleg négy osztály használható, a *Downloader*, az *Exporter*, az *Indexer* és az *Abstracter (3. ábra)*. Az első az adatbázis letöltésére, a második az adatbázis bizonyos elemeinek a kitöltésére, a harmadik az indexek frissítésére, a negyedik pedig az egysoros katalóguscédulák létrehozására szolgál. Ezekről az osztályokról részletesebb információt, illetve példákat a Cinege angol nyelvű dokumentációjában találunk a *data/cinege/docs* alkönyvtárban.

A Downloader csak azokat a hibákat szűri ki, amelyek a letöltést lehetetlenné teszik, így a hibás UTF8 karaktereket,[24] a hibásan csomagolt bibliográfiai rekordokat,[25] a dupla vonalkódokat,[26] az irreális visszavételi dátumokat,[27] a tényleges kölcsönzési határidő (dátum) előtti visszavételeket. A paraméterei határozzák meg, hogy melyik táblákat töltse le.

Az Indexer azokat a hibákat szűri ki, amelyek az indexépítést zavarják, így például a nem magyar[28] karaktereket, illetve a kettőnél több címet tartalmazó rekordokat. Pontos működése attól függ, hogy a konfigurációosztályban milyen indexeket állítunk be.[29]

Az Abstracter nagyon hasonlóan működik az Indexerhez, csak indexek helyett egysoros katalóguscédulákat épít.



3. ábra **A Cinege szűrésre használt osztályainak működése**

Az Exporter célja bizonyos rekordok bizonyos mezőinek exportálása az adatbázisból. Erre kilenc parancs szolgál, amelyek közül némelyik paraméterezése nagyon egyszerű, némelyiké meglehetősen összetett. Egyszerű például a BIB2ADM, amely bibliográfiai rekordszámokhoz adminisztratív rekordszámokat rendel, és egyetlen paramétere a bemeneti CSV fájl neve (az exporter.propertires fájlban beállítottak is hatással vannak a működésére).

Összetettebb példa az ADM2LoanNumber, amelynek három paramétere (bemeneti fájlnév, mettől, meddig) és három kimeneti oszlopa van (kölcsönzések száma, kölcsönzött példányok száma, példányok jelenlegi száma).

A legbonyolultabb példa a DB2BIB, amely után szinte tetszőleges számú alparancs állhat, ezek logikai ÉS kapcsolata alkotja a végleges parancsot. Tíz alparancsa van, amelyek mindegyikének három-hat paramétere van. Segítségével olyan hibalisták készíthetők, amelyek a „Ha a rekordnak pontosan egy darab '100'-as hívójelű, 'a' almezőjű mezője van, akkor nem lehet egyetlen '104'-es hívójelű mezője sem" típusú rekordokat tartalmazzák.

A parancsok és alparancsok tömör összefoglalása a dokumentációban három oldalt tesz ki, szintúgy a példák. A szűrést megtanulni csak a gyakorlati alkalmazás során lehet, teljes leírásuk angolul a *data/cinege/docs/exporter.pdf* és a *data/cinege/docs/examples.pdf* fájlokban található.

Példa

Tegyük fel, hogy könyvtárunkban létrehoznak egy Európai Olvasótermet, ahol az Európai Unió három nyelvén (angol, francia, német) írott könyveket szeretnénk szabadpolcra helyezni, mégpedig elsősorban a gyakran kölcsönzött példányokat. Nyújtsunk e feladat megvalósításához hatékony informatikai támogatást:

- Először letöltjük az adatbázis releváns részeit:[30]
  java Downloader download_from_
  z00 download_from_z103 download_from_
  z30 download_from_z36_z36h
- Lekeressük azokat a rekordokat, ahol nincs nyelvkód (ezeket külön kezeljük):
  java Exporter DB2BIB:w1.csv:
  IFMULTIPLICITYCHEdCKSUCCEEDS#041#.*#a#
  false#0#0
- Lekeressük ezekhez a könyvekhez a címet és a kiadót:
  java Exporter BIB2LIST:w1.csv:FIRST#245#
  a:FIRST#260#a
- Megformázzuk OpenOffice-szal[31] a listát (ha túl nagy, a "split" paranccsal feldaraboljuk):
  oocalc data/cinege/work/w1.csv.out.csv
- Lekeressük azokat a rekordokat, ahol van nyelvkód és értéke "eng" vagy "ger" vagy "fre":
  java Exporter "DB2BIB:work1.csv:
  IFMULTIPLICITYCHECKFAILS#041#.*#a#false#
  0#0:\IFANYMATCHES#POST_ALPHANUMERIC#
  041#a#^(eng)|(ger)|(fre)$"
- Lekeressük ezekhez a könyvekhez a címet és a kiadót:
  java Exporter BIB2LIST:work1.csv:FIRST#245#a:
  FIRST#260#a
- Lekeressük az ezekhez tartozó Adminisztratív rendszerszámokat:
  java Exporter BIB2ADM:work1.csv.out.csv
- Lekeressük az elmúlt 12 hónap kölcsönzéseit és kölcsönzött példányok számát:
  java Exporter ADM2LoanNumber:work1.csv.out.
  csv.out.csv:200503010000:200603012359
- Megszűrjük a példánytalan adminisztratív rekordoktól a fájlt:
  grep -v "NULL" data/cinege/work/work1.csv.out.
  csv.out.csv.out.csv >data/cinege/ work/work2.csv
- Lekeressük azokat az Adm rekordokat, ahol van olyan példány, amelyik alkönyvtára "BMEA1":
  java Exporter DB2ADM:work3.csv:ifinfile#work2.
  csv:ifitemsdatamatches#NO_FILTER#SUBLIBRAR
  Y#^BMEA1$
- Hozzáadjuk a példányadatokat:
  java Exporter ADM2Item:work3.csv
- Megformázzuk OpenOffice-szal a listákat, kitörölve a felesleges oszlopokat:[32]
  oocalc data/cinege/work/work2.csv
  oocalc data/cinege/work/work3.csv.out.csv

Fészek

A Cinege a méltán világhírű nemzetközi forrástárházon, a *SourceForge*-on fészkel, mégpedig a *http://sourceforge.net/projects/cinege* címen. Célszerű lehet az ismerkedést a Networkshop 2006 videotárházában[33] és a *data/cinege/docs* alkönyvtárban kezdeni. Tanácsokat, segítséget a szerzők elektronikus levélben szívesen nyújtanak, és a hibajelzéseket, valamint a fejlesztési javaslatokat is levélben várják. Bár a Cinege szabad szoftver, és GPL licenccel érhető el, a terméktámogatásról és a lekérdezések fejlesztésének részleteiről a szerzők tudnak felvilágosítást nyújtani.

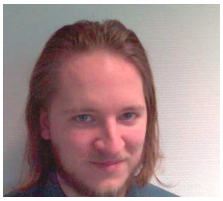
Megjegyzések: a bejegyzett nevek (Aleph, MySQL, Oracle, Java, Windows XP stb.) a bejegyző cégek tulajdonában vannak, itt csupán technikai hivatkozásként szerepelnek. A Cinege részét nem képező szoftvereszközök nem feltétlenül GPL licenccel rendelkeznek.

Jegyzetek

[1] Bár nem tipikus, előfordul. Például adott esetben egy elektronikus folyóiratot vagy könyvet csak a forrás URL megjelölésével emelhetünk legálisan az adatbázisba a szerzői jogok miatt.
[2] Testre szabható, konfigurálható.
[3] Lásd RDBMS, azaz Relational DataBase Management System.
[4] A magyar „pakolt” szót az angol „packed” szó szinonimájaként használjuk.
[5] A besorolási rekordokkal itt nem foglalkozunk, részben azért, mert az adatbázis szintjén sokszor speciális bibliográfiai rekordként vannak megvalósítva, és így a megfelelő bibliográfiai szűrővel jól szűrhetők, részben pedig azért, mert szinte soha nem kerülnek be a mintalistákba.
[6] *http://java.sun.com*
[7] *http://www.unicode.org*
[8] GNU Public License alatt elérhető.
[9] Universitätsbibliothek der Freie Universität Berlin
[10] Pannon Egyetem Központi Könyvtár
[11] 8-bit Unicode Transformation Format
[12] Java Database Connectivity
[13] *http://www.mysql.com*
[14] *http://www.exlibris.co.uk/aleph.htm*
[15] *http://www.oracle.com*
[16] Comma Separated Values
[17] Machine Readable Cataloging
[18] 2 GHz 64 bit CPU, 2 GB RAM, 200 GB HDD, 500 euró
[19] Forrás: *www.wikipedia.org*
[20] Practical Extraction and Reporting Language
[21] *http://en.wikipedia.org/wiki/Awk*
[22] Szűkebb értelemben Unix és Linux rendszerek, tágabb értelemben POSIX rendszereket.
[23] Shellekben.
[24] Amely, mint korábban említettük, a verziófrissítést tette kínszenvedéssé.
[25] Beállítható, hogy az UTF8-ként adott ISO-8859-2 karaktereket átkonvertálja-e tényleges UTF8 karakterekre, illetve hogy a HUNMARC mezőben a mezőhosszokat UTF8 vagy ISO-8859-2 karakterszámban értelmezze-e. Sajnos már mindkettőre volt szükség, remélhetőleg csak az Oracle NLS és az SQLplus beállításai miatt.
[26] Nem ez a megdöbbentő, hanem az, hogy ezekre rákeresve sem vette észre a forrásrendszer, hogy duplikátum.
[27] Felülbírálható ugyan a felkínált dátum, de a "20991231" azért kicsit túlzás.
[28] Természetesen más szűrőket is létrehozhatunk, illetve beállíthatunk.
[29] Beállítás után újra kell fordítani a rendszert, leállítani az indexelőt, dobni az indextáblákat, és elindítani az indexelőt.
[30] Lehetőleg hajnalban, miután a mentés lefutott.
[31] *http://www.openoffice.hu*
[32] Természetesen egy listában is le lehet keresni mindezt, de az bonyolultabb.
[33] *http://vod.niif.hu/index.php?lg=hu&mn=archive&eid=42&sm=listevent*

Beérkezett: 2006. VI. 23-án.

Liszkay Béla
a BME OMIKK
főigazgató-helyettese.
E-mail:
bliszkay@omikk.bme.hu



Nagy Elemér Károly
a BME OMIKK informatikai
csoportjának munkatársa.
E-mail: *eknagy@omikk.bme.hu*