SimpleSQL:
Egy relációs réteg SimpleDB-hez

Tanulmány a
André Calil, Ronaldo dos Santos Mello:
SimpleSQL: A Relational Layer for SimpleDB
cikkhez (ADBIS konferencia 2012.)

ELTE IK 2012.
Balázs Barnabás Lóránt
Iván Gergő
Szalona Szandra

Összefoglaló

Ez az előadás a SimpleSQL-t mutatja be, amely egy relációs réteg az Amazon SimpleDB-hez. Az Amazon SimpleDB egy alapvetően rugalmas, dokumentumközpontú, nem-relációs, cloud adattároló, amelyben a fejlesztőknek nem kell az adatbázis adminisztrálásával foglalkozni, csak az adatok betöltésével és lekérdezésével, minden mást az Amazon SimpleDB végez el. A SimpleSQL egy SQL interfészt nyújt, amely magában foglal minden információt az adatmodellezésről (data modeling), adatok perzisztenciájáról (data persistence) és az adatok eléréséről SimpleDB-ben. Bemutatjuk még az architektúra, adat és művelet leképezés módját relációs adatbázisból SimpleDB-be, továbbá kísérleteket, amelyek megbecslik a felhőben található adat lekérdezésének teljesítményét SimpleSQL-t és csak SimpleDB-t használva.

1 Bevezetés

Az utóbbi években megfigyelhető, hogy a szoftverek szolgáltatásként történő értékesítése egy újító ötletből egy abszolút életképes üzleti modellé lépett elő. Belátható, hogy a szolgáltató cégeknek ezen rendszerek beszerzése és fenntartása jelentős költséget jelent, azonban a kereslet is magas, hiszen felhasználók/fejlesztők számára vonzó a szolgáltatásnyújtó által fenntartott szoftver, amelyben a biztos elérhetőséget és adatok titkosságát szerződések biztosítják.

Az adattárolás és menedzsment képessége is elérhető felhő alapú számítógépes platformokon keresztül (cloud computing platform). Ez eltér a már megszokott adatbázis menedzselési rendszerarchitektúráktól, rendelkezik elosztott rendszerekre jellemző tulajdonságokkal, mint a biztos elérhetőség vagy a hibatűrés. Ennek megfelelően új adatmodellek is megjelentek, amelyek megfelelnek a jelenlegi webes alkalmazásoknak és programozási mintáknak, valamint sokkal inkább szöveg vagy objektum és nem rekord központúak.

Ilyen új modellek például a kulcs-érték gyűjtemények (key-value collection), dokumentumorientált (document-oriented) vagy a szuper-oszlop (super-column). Az ilyen modellekre alapozott felhő adatbázis rendszerek Not only SQL (NoSQL) néven váltak ismerté.

Mivel ezek az adatbázisok nem relációsak, az SQL szabvány nem támogatott. Emiatt a relációs adatokra épülő alkalmazások költöztetése és alkalmassá tétele

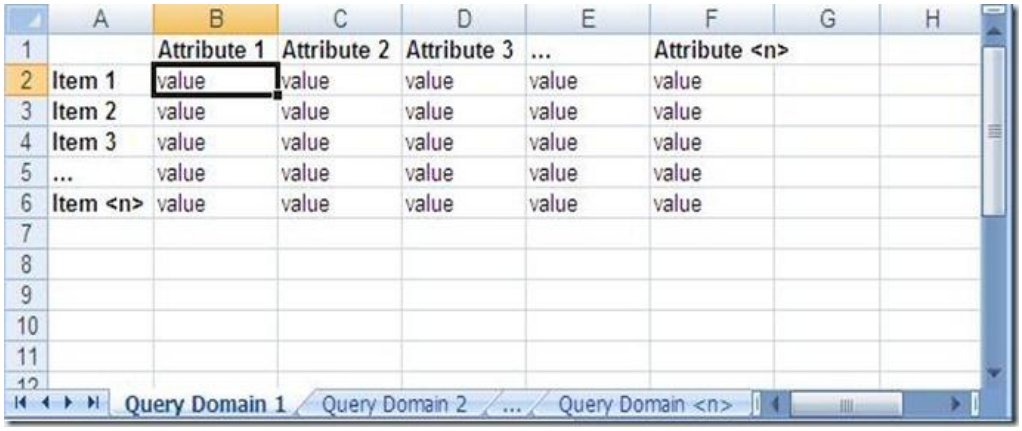
(migrate and adapt) nehézkes. Ennek a problémának a megoldására való a SimpleSQL. A SimpleSQL az ISO/IEC SQL egy egyszerűsített változatát támogatja, amelynek segítségével képesek vagyunk adatmódosító műveletek végrehajtására. Emelett néhány lekérdezéshez kapcsolódó képessége is van. SimpleSQL-nél a kliens alkalmazás elszigetelt a SimpleDB hozzáférési metódusaitól valamint adatmodelljétől és egy megszokott, relációs felületet biztosít a felhőben levő adatok eléréséhez.

A SimpleSQL támogatja a joinokkal történő lekérdezést, amely a NoSQL adatbázisok elérésének nem egy alapvető módja és SimpleDB interfészben sincs implementálva. A SimpleSQL képes a számos táblát összekötő lekérdezések olyan módú felbontására, amely lekérdezések egy sorozatát hozza létre. Ezen lekérdezések már csak egy táblára vonatkoznak és az adott táblából szerzik be a megfelelő adatokat. Majd végül ezen megoldásokat egyesíti a végső megoldás eléréséhez.

2 SimpleDB

A NoSQL adatbázisok közül az egyik legjelentősebbek a kulcs-érték típusú adattárolók, amelyek szótárszerű struktúrát alkalmaznak. A felhasználók által definiált kulcsokhoz rendelik az értékeket. A másik, amiről érdemes szót ejteni az a dokumentumorientált, amely az objektumokat dokumentumokként rendezi és indexeket tartalmaz a kereséshez.

A SimpleDB az Amazon dokumentumorientált modellt követő megoldása az adatmenedzsmentre a felhőben. Az adatok automatikusan sokszorosításra kerülnek a felhasználó által kiválasztott régióban található adatcenterekben (data center). A SimpleDB adatmodellje domainek-ből, tételekből (item), attribútumokból (attribute) és értékekből (value) áll.



SimpleDB adatmodellje

A domain egy névből áll és tételek egy csoportjából. Minden tétel rendelkezik attribútumok egy csoportjával, amelyek kulcs-érték párok. Egy felhasználó legfeljebb 250 domain-nel rendelkezhet és a domai-nek maximum 10GB-osak lehetnek, amely a legtöbb alkalmazás számára elegendő.

A SimpleDB nem támogatja az olyan lekérdezéseket, amelyek különböző domainek-ből kapcsolnak össze adatokat. Az ilyen esetekben a felhasználói program által végrehajtatott join műveletekre van szükség. A tételek egy névből és attribútumok egy csoportjából állnak. Csakúgy, mint a domainek-nél ez a név itt is egyedi kell, hogy legyen. Az attribútumoknál egy kulcshoz több érték is tartozhat és egy adott domain-en belüli nem feltétlenül kell minden objektumnak ugyanazokkal az attribútumokkal rendelkezni.

A SimpleDB garantálja, hogy író/módosító művelet végrehajtása során a tételek minden másolata változik, azt azonban nem, hogy olvasás után a tétel legújabb verziója kerül kiolvasásra, mivel a tétel összes másolatának módosítása időbe telik, így ha közben érkezik egy olvasási kérés, akkor előfordulhat, hogy egy régebbi/elavult értékeket kapunk vissza.

A felület, amelyen keresztül elérhetjük a SimpleDB-t HTTP kéréseket továbbit. A legtöbb jelenleg használatos fejlesztői keretrendszer (development framework) el tudja érni a rendszert. Minden olvasó és író művelet, valamint még a domain adminisztrációs feladatok is GET és POST HTTP műveletek segítségével történnek. A SimpleDB csak szolgáltatásként elérhető, lokálisan nem telepíthető.

3 SimpleSQL

A relációs adatbázisokra már lefejlesztett alkalmazások számára akadályt jelent a NoSQL minta. Egy ilyen alkalmazást átírni felhő alapú platformra sok munkát jelentene, ennek feloldására alkalmas egy olyan hozzáférési réteg, amely az SQL kéréseket lefordítja a SimpleDB API-nak és az eredményeket relációs formátumban adja vissza. Ezt nevezzük SimpleSQL-nek. A cikkben prezentált verzió négy szokásos módosító utasítás végrehajtására képest: beillesztés (insert), frissítés (update), törlés (delete) és kiválasztás (select).

Kapcsolat a relációs modell és a SimpleDB modell között:

Relációs	SimpleDB
Séma	Domain
Tábla	-
Tábla sor	Tétel
Attribútum	Attribútum kulcs
Érték	Attribútum érték
Elsődleges kulcs	Tétel név

Ezek az egyenlőségek azért fenntartásokkal kezelendők, kisebb különbségek találhatók. A SimpleSQL parancsaiban nem használhatunk sémaminősítőket. Továbbá ugyan nem találhatunk tábla fogalomnak megfelelőt SimpleDB-ben, azonban a tábla név attribútumként megtalálható, hogy a tételekhez típust tudjunk rendelni.

3.1 Feldolgozási követelmények

Ahhoz, hogy a SimpleDB-hez csatlakozni tudjunk és azonosítsuk a domain-t, a SimpleSQL-nek a következő információkat kell megkapnia a felhasználótól:

- Hozzáférési kulcs (access key): A felhasználó SimpleDB fiókjához tartozó hozzáférési kulcs, amely az Amazon portálra való bejelentkezésnél található meg.
- Titkos hozzáférési kulcs (secret access key): Szintén az Amazon portálon található meg.

- Domain kiosztás: Ha a felhasználó egynél több domain-nel rendelkezik, akkor a SimpleSQL-hez el kell juttatnia egy szótárt, amely a domain nevet kulcsként használja, a domain tábláinak listáját pedig értékként. Ha a felhasználó csak egy domain-nel rendelkezik, akkor az előzőek helyett ennek a domain-nek a nevét továbbíthatja.

3.2 Hozzáférési felület

A SimpleSQL hozzáférési interfésze két metódusból áll: *ExecuteQuery*, amely egy DataTable objektumot ad vissza és az *ExecuteNonQuery*, amely sztringet ad vissza. Mindkettőnek SQL parancsot kell megadni paraméterként.
Ahogy már korábban leírásra került a SimpleSQL négy féle módosító utasítást támogat, azonban ebben az első verzióban ezek rendelkeznek szintaxisra vonatkozó megkötésekkel:

- SELECT: Támogatja az egyetlen táblára vonatkozó lekérdezéseket, illetve a több, inner joinnal összekötött táblákra vonatkozó lekérdezéseket. Join esetében minden deklarált attribútumnak a következő formátumban kell lenni: *tábla.attribútum*
- UPDATE: feltétel nélküli frissítések nem engedélyezettek
- INSERT: parancsonként egy adat beillesztését támogatja
- DELETE: feltétel nélküli részlekérdezések és törlések nem támogatottak

3.3 Parancs felbontás

Az első feldolgozási lépés az SQL parancs felbontása és átkonvertálása a SimpleDB által használt domain-re. Ahhoz, hogy támogatott legyen, minden parancshoz tartozik egy pontos kifejezés, amelynek két célja van: érvényesítse a parancs szintaxisát és kivonatolja a parancs alapelemeit.

3.4 Feldolgozás és visszatérés

Az SQL parancs megalkotása után, a SimpleSQL lefordítja azt egy SimpleDB REST metódushívássá. Az új parancs létrehozásakor, elsődleges a cél SimpleDB domain azonosítása, melyet az SQL parancs megfelelő felbontásával kapunk meg.

A DELETE és UPDATE parancsok visszatérési értéke minden esetben az érintett adatok száma. Hány adatot töröltünk ki, illetve hány adatot frissítettünk. Az INSERT parancs a művelet eredményességével tér vissza (sikerült, vagy nem sikerült), míg a SELECT utasítás a lekérdezésnek megfelelő adatokat szolgáltatja, amiket DataTable struktúrába jelenít meg.

INSERT

Egy tábla *tuple*, egy tételnek felel meg a SimpleDB sémában. Így egy INSERT parancs egy tételt generál. Az utasítás végrehajtása előtt a SimpleSQL leellenőrzi, hogy az oszlopok száma egyenlő-e az értékek számával. A megadott attribútumok mellett, a SimpleSQL egy új attribútumot is ad a tételhez, a *SimpleSQL_TableName* formátumban, hogy ezzel megőrizze a céltábla nevét. A tétel neve (amely egy szükséges mező a SimpleDB modellben) egy globális egyedi azonosító egy példányával van kitöltve.

UPDATE és DELETE

Ezen parancsok végrehajtásakor, először az utasításokban szereplő feltételeknek megfelelően szűrjük ki azon elemeket, melyeken a frissítést végrehajtjuk, vagy melyeket törlünk. Ez teljesen hasonló egy egyszerű SELECT utasításhoz (melyben nincs join művelet). DELETE esetében a talált elemek mind törlésre kerülnek. UPDATE esetében meg kellett adni, hogy mely attribútumokat kell frissíteni, illetve azt, hogy ezeknek mi legyen az új értékük.

SELECT

Egy lekérdező utasítás esetén, a SimpleSQL létrehoz egy listát a parancsban szereplő attribútumok, céltáblák, join-ok és szűrők tagjaiból. Ha vannak join műveletek, akkor azokat szétvágja egyszerű lekérdezésekké. Ez azt jelenti, hogy a *tábla.attribútum* jelölés használatával a SimpleSQL azonosítja az összekapcsolt táblák feltételeit és a várható attribútumaikat. Minden egyszerű lekérdezés visszatérésének fogadása után egy adattábla jön létre a várható visszatérés sémájával. A fogadott tételek listája összekapcsolódik a relációs séma idegen kulcsainak segítségével és a visszatérési tábla kitöltésre kerül. Ajánlott, hogy minden

lekérdezés által összekapcsolt tábla legalább egy feltétellel rendelkezzen, így meggátolva, hogy a SimpleSQL-t hatalmas mennyiségű adatot adjon vissza.

Amikor a lekérdezések feldolgozásra kerülnek SimpleDB-ben, akkor a válasz mindig egy tételekből álló gyűjtemény. A SimpleSQL végigiterál minden tétel minden attribútumán. Az attribútum neve, a várható attribútumok alapján jóváhagyásra kerül és a visszatérő attribútumok végső listájához adódik. A visszatérési adattábla a kiválasztott attribútumokkal van feltöltve. Minden visszakapott tételt egy sor reprezentál a visszatérési táblában és a sémája a várható attribútumok uniójával keletkezik. Ha a visszakapott tételek nem ugyanazzal a sémával rendelkeznek, akkor a megfelelő mezőkbe null kerül a visszatérési táblában.

A SimpleDB-ben minden lekérdezésre érkező válasz legfeljebb 1MB méretű lehet. Így bizonyos esetekben nem minden eredményezett tétel kerül visszaadásra az első válaszban. A SimpleDB ilyenkor küld egy *NextToken* értéket, amelynek segítségével a felhasználó újból kiadhatja a lekérdezést ezzel a tokennel együtt, így megszerezve az eredmény következő részét. A SimpleSQL rendelkezik egy rekurzív metódussal, amely addig folytatja ezt a lekérdezést, amíg a teljes válasz a felhasználó birtokába nem kerül.

4 Kísérleti eredmények

A cikk utolsó nagy fejezete egy kísérletet dolgozott fel, melyben végrehajtási idők kerültek elemzésre. A kísérlet az UFSC (Universidade Federal de Santa Catarina) egyetem vizsgaadataiból álló adatokat használta fel. Ezen adatok relációs kapcsolatban állnak és 6 táblát különböztetünk meg egymástól. Többek között a jelentkezők, a kurzusválasztásaik, illetve a vizsgaeredményeik tábláit. Minden tábla több mint 500 000 *tuple*-t tartalmaz. A kísérlet az alábbi adatokkal rendelkező környezetben lett elvégezve:

Dell Vostro 3550 notebook,
Intel Core i5-2430M processzor,
6GB DDR3 1066mHz RAM,
10Mbps ADSL2 internetkapcsolat.

A SimpleDB beállításainak megfelelően minden adat egy domain-ben került elhelyezésre. Az adatok betöltése SimpleDB-be szintén a kísérlet része volt és a következő részben tárgyalt módon történt.

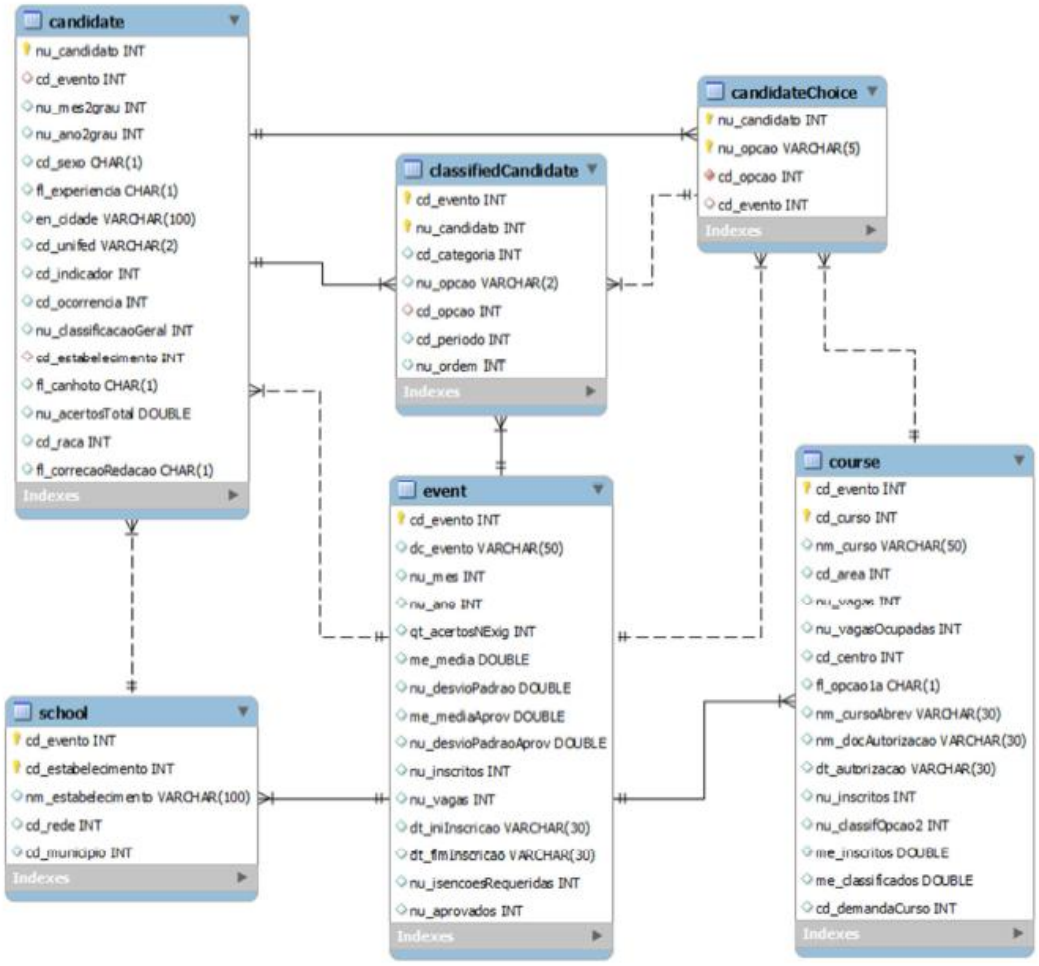


Fig. 4. Relational schema used in the experiments

A kísérlet két művelet teljesítményét értékelte: INSERT és SELECT. Az INSERT utasítás hatékonyságát a nagy adatmennyiség betöltésekor értékelték, míg a SELECT utasítást a különböző bonyolultságú lekérdezések végrehajtásakor. Mindkét utasítás esetén a kísérletben összehasonlításra került a műveletek végrehajtási ideje a SimpleSQL réteget használva, illetve csak a SimpleDB.NET API – t felhasználva.

INSERT

Az INSERT utasítás arra lett felhasználva, hogy az adatokat 2 táblába töltsük be vele. A következő táblázat foglalja össze a feldolgozási időt SimpleSQL, illetve SimpleDB API esetén, valamint a percenkénti átlagosan beillesztett *tuple* számot.

Table 3. Results of INSERT operations

Table	Mode	# tuples	Duration	Average (tuples/min)
candidate	SimpleSQL	50000	03:29:24	238.78
	SimpleDB		03:19:24	250.75
candidate Choice	SimpleSQL	100000	07:37:08	218.75
	SimpleDB		07:07:32	233.90

A táblázat a SimpleSQL esetén egy kicsit magasabb értékeket mutat, de erre számítani is lehetett. Viszont ez az eltérés kevesebb, mint 5% mindkét futás esetén. Emellett pedig a percenkénti átlagosan beilleszett *tuple*-ok száma közti különbség is elhanyagolható. A kísérletből kiderült, hogy a SimpleSQL réteg bevezetése nem veszélyezteti a SimpleDB API teljesítményét és skálázhatóságát.

SELECT

A SimpleSQL a SELECT utasítás futtatására is értékelve lett. Külön – külön került megvizsgálásra egy egyszerű, join nélküli lekérdezés, illetve egy bonyolultabb lekérdezés, mely már 3 kapcsolatot és 4 táblát tartalmazott.
Az első táblázat mutatja be az egyszerű lekérdezés szintaktikáját SimpleSQL és SimpleDB API esetén, valamint a található *tuple*-ok számát, míg a második táblázat összefoglalja a futási időket minden lekérdezés esetén. Minden lekérdezés háromszor került végrehajtásra.

Table 4. Evaluation of simple queries

Query	SimpleSQL	SimpleDB	# tuples
1	`SELECT nu candidate, cd race FROM candidate WHERE en city like 'FLORIAN%'`	`SELECT nu candidate, cd race FROM domain1 WHERE en city like 'FLORIAN%'`	34678
2	`SELECT nu_candidate, cd_race FROM candidate WHERE cd_gender = 'F'`	`SELECT nu_candidate, cd_race FROM domain1 WHERE cd_gender = 'F'`	58410
3	`SELECT * FROM curse WHERE cd area = 1 and nm curse like 'ENGE%' and nu places >= 100 AND nu applicants > 1000`	`SELECT * FROM domain1 WHERE cd area = '1' and nm curse like 'ENGE%' and nu places >= '100' AND nu applicants > '1000'`	58

Table 5. Average duration time for each query

Query	SimpleSQL	SimpleDB
1	00:02:22	00:01:34
2	00:03:09	00:02:33
3	00:00:03	00:00:02

Az adatok alapján a SimpleSQL esetén a feldolgozási idő mindhárom lekérdezés esetén nagyobb volt, mint a SimpleDB API felhasználásakor. Egyúttal ez a magasabb érték nem haladta meg a 40%-ot. Ezen adatokat a kísérletben elfogadhatónak találták, figyelembe véve a nagy adatmennyiséget, melyhez hozzá kellett férni. A *Candidate* tábla esetén ez kb. 150 000 *tuple*.

Bonyolultabb lekérdezések esetén figyelembe kell venni, hogy a SimpleDB API nem ismeri a tételek típusának (tábla) fogalmát, illetve nem támogatja a join operátort sem. Emiatt a SimpleDB API képes arra, hogy csak a vonatkozó adatokat szűrje ki minden egyes táblából és meghagyja az alkalmazói rendszernek a join-ok végrehajtását.

Másrészről a SimpleSQL arra lett tervezve, hogy támogassa a komplex lekérdezéseket. Az utasítás feldolgozási lépései a következők:

- *Vágás (Split):* a parancs szétvágása egyszerű SELECT utasításokká (join-ok nélkül). A szükséges vágások végrehajtásakor a SimpleSQL azonosítja az attribútumokat és feltételeket minden egyes táblához.
- *Hozzáférés (Access):* az összes SELECT utasítás átadásra kerül a SimpleDB-nek
- *Átalakítás (Transform):* minden egyes parancs eredményének transzformálása a relációs sémába
- *Csatlakozás (Join):* a transzformált táblák kombinálása a join feltételek szerint. Így megkapjuk a végső eredménytáblát.

A *Hozzáférés (Access)* lépés az egyetlen olyan lépés, mely érvényes a SimpleDB API esetében. A többi lépés csak a SimpleSQL esetében fordul elő. Ezek alapján a kísérletben minden lépés feldolgozási ideje mérésre került és különösen az *Access* lépés esetén kell az adatokat összehasonlítani.

Az alábbi táblázat mutatja a komplex lekérdezést, eredeti formában, illetve a SimpleDB szintaxisában. Ez a lekérdezés is 3 alkalommal került végrehajtásra.

Table 6. SELECT command in the original form and in SimpleDB syntax

SimpleSQL	SimpleDB
SELECT classifiedCandi- date.nu order, candida- te.en city, school.nm school, event.dc event FROM candi- date INNER JOIN school ON candida-te.cd school = school.cd school INNER JOIN classifiedCandidate ON can- didate.nu candidate = clas- sifiedCandi- date.nu candidate INNER JOIN event ON classified-	SELECT nu order FROM do- main1 WHERE cd event = '25' SELECT en city FROM do- main1 WHERE cd event = '25' SELECT nm school FROM do- main1 WHERE cd event = '25' SELECT dc event FROM do- main1 WHERE cd event =

Candidate.cd event = event.cd event WHERE event.cd event = 25 AND school.cd event = 25 AND classifiedCandi-date.cd event = 25 AND can-didate.cd evente = 25	'25'

A kísérlet eredményeképpen láthatjuk, hogy az *Access* lépés a legdrágább költségű. A feldolgozási ideje a SimpleSQL esetében nagyobb értéket mutat, hisz minden egyszerű lekérdezés külön-külön adódik át. Azonban azok a lépések, melyeket csak a SimpleSQL hajt végre a remélt szinten gyorsak és az összegük nem haladja meg az *Access* lépés feldolgozási idejét.

Table 7. Average duration time for each processing step for the complex query

Step	SimpleSQL	SimpleDB
Split	00:00:03	
Access	00:18:17	00:18:05
Transform	00:02:23	
Join	00:04:08	

5 Összegzés

Az ilyen adatbázis-kezelő rendszerek, mint szolgáltatások, számos előnnyel járnak. Ilyen előnyöknek tekinthetők a kisebb költségek, valamint, hogy nem kell túl sok aggodalmat fordítani az adatbázis adminisztrálására. Mindamellett, hogy a legtöbb jelenlegi, adatközpontú alkalmazás relációs adatbázisokra épül, javaslott kiépíteni egy hidat, mely lehetővé teszi a relációs adatelérést a felhőben tárolt adatokra.

A cikk szerzői annyiban járultak hozzá a fent említett problémához, hogy bemutatták a cikkben a SimpleSQL-t, ami egy speciális megoldás relációs sémák és relációs parancsok lefordítására a SimpleDB számára, ami egy dokumentumközpontú adatbázis. Annak ellenére, hogy a cikkben a szerzők a SimpleDB adatbázist vizsgálták, állítják, hogy a megoldásuk egy általános megoldás minden relációs és felhő alapú adatbázis közti átmenetre. De erre majd csak egy jövőbeni cikkben térnek ki részletesebben.

A cikkben elvégzett kísérlet alapján a SimpleSQL egy kicsit hosszabb feldolgozási időt igényel, mint ha csak tisztán a SimpleDB-t használnánk. De ez a többlet nem jelenti azt, hogy ne legyen tanácsos adoptálni a megoldást. Az INSERT műveletre a többlet kevesebb volt, mint 5%. Az egyszerű lekérdezésekre viszont 40%, ami még elfogadható, ha figyelembe vesszük, hogy a SimpleSQL feladata, hogy az adatokat átkonvertálja a relációs sémának megfelelő alakra. Az eredmények fényében kijelenthető, hogy a SimpleSQL használata nem fog a teljesítmény kárára válni. A bonyolultabb lekérdezések esetében a feldolgozási idő jelentősen több SimpleSQL-t használva, ez jelzi, hogy ezt a végrehajtást még optimalizálni kell.

Egyelőre még nem foglalkozik több cikk azzal, hogy bemutasson egy relációs interfészt a nem relációs adatbázis rendszerekhez. A szerzők további céljai közé tartozik, hogy további kísérleteket végezzenek különböző nagyságú adathalmazokon, a SimpleSQL optimalizálásának céljából.