BME Villamosmérnöki és
Informatikai Kar

Távközlési és Médiainformatikai Tanszék
Adatbázisok Laboratórium

Elosztott adatbázis-kezelő rendszerek

Laboratóriumi gyakorlat feladatlap

Oktatási segédanyag a Hálózatba kapcsolt adatbázisok tárgyhoz

1. változat

Egyetemi belső használatra!

Konzulensek:

Erős Levente
Gajdos Sándor

Szerző:

Sajó Dániel

2014.

1 Néhány szó a gyakorlatról

A gyakorlat célja a MySQL Cluster néhány fontos funkciójának és tulajdonságának bemutatása. A gyakorlat során a szomszédos gépnél ülő hallgatók párban dolgoznak. A feladatsor az együtt dolgozókat 1. hallgató és 2. hallgató néven hivatkozza. Hogy az együtt dolgozók közül ki az első és ki a második, az tetszőleges, de a munka során ne változzon.

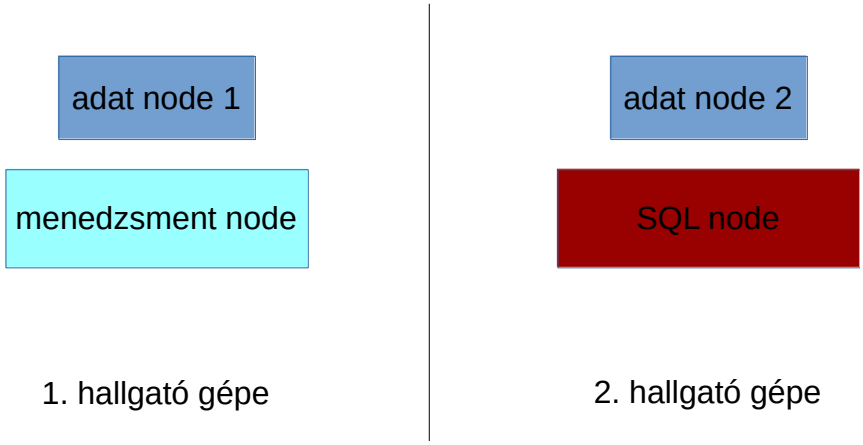
A feladatokat virtuális gépen végezzük. A virtuális gépen Windows XP operációs rendszer fut, a hálózatot a gazdagépen keresztül Bridge üzemmódban érik el. Ez annyit jelent, hogy a virtuális gép a gazdagép hálózatán egyenrangú csomópontként jelenik meg. Ez számunkra azért fontos, mert a feladatok végzése során szükség lesz a virtuális gépek IP címére, amivel elérik egymást. A virtuális gép IP-címét az ipconfig paranccsal lehet megnézni.

A gyakorlat során a MySQL Cluster 7.3. verzióját használjuk, mely az 5.6-os verziójú MySQL-re épül. A program letölthető a http://dev.mysql.com/downloads/cluster/ weboldalról, a virtuális gépeken a C:\mysql könyvtárban található.

2 MySQL Cluster bekonfigurálása, elindítása

A rendszerünket úgy állítjuk be, hogy 2 adat node-ból, 1 SQL node-ból és 1 menedzsment node-ból álljon. A tárolásra kétszeres redundanciát határozunk meg, azaz minden adatból 2 példányt tárolunk.

A node-okat éles rendszerben külön gépen érdemes futtatni, habár lehetőség van akár az összes node-ot is egy gépen futtatni. A node-okat a laborban két gépre telepítjük az 1. ábrának megfelelően.



1. ábra: Rendszerünk felépítése

A rendszer elindításához 2 konfigurációs fájlt kell készítsünk.

2.1 Globális konfigurációs fájl

Az első a rendszer globális konfigurációs fájlja, melyet csak a menedzsment szerver olvas. A menedzsment szerver osztja tovább a benne foglalt információt a többi node-nak. Ebbe a fájlba kerülnek a node-ok azon beállításai, amelyek az egész rendszerre vonatkoznak, a node-

ok együttműködéséhez kellenek. Erre példa a node-ok elérési helye: a fájl tartalmazza azon IP-címeket, amelyeken az egyes node-ok elérhetőek. További információt a globális konfigurációs fájlról a http://dev.mysql.com/doc/refman/5.6/en/mysql-cluster-config-file.html oldalon találhatunk.

A létrehozandó globális konfigurációs fájl a következőket tartalmazza:

```
[ndb_mgmd]
# Management process options:
hostname=<menedzsment node-ot futtató gép ip címe> # Hostname or
IP address of MGM node
DataDir= C:/mysql/bin/cluster-logs # Directory for management node
log files
NodeId=5

[ndbd default]
# Options affecting ndbd processes on all data nodes:
NoOfReplicas=2 # Number of replicas
DataMemory=80M # How much memory to allocate for data storage
IndexMemory=18M # How much memory to allocate for index storage

[ndbd]
HostName= <1. adat node-ot futtató gép ip címe>
NodeId=1

[ndbd]
HostName= <2. adat node-ot futtató gép ip címe>
NodeId=2

[mysqld]
HostName= <SQL node-ot futtató gép ip címe> # Hostname or IP
address
NodeId=10
```

A fájl az ini formátumot követi. Erről részletesebben a http://en.wikipedia.org/wiki/INI_file oldalon olvashatunk:

Az első [ndb_mgmd] rész a menedzsment node-ot beállításait tartalmazza. Itt a következőket adjuk meg:

- hostname: a node-ot futtató gép IP-címe
- DataDir: az a könyvtár, ahova a menedzsment szerver által generált fájlok kerülnek. Ilyen fájlok például a rendszer log fájljai.
- NodeId: a menedzsment node azonosítója a rendszerben. Ha node-onként kézzel nem állítjuk be, a node-ok elindulásukkor véletlenszerű kapnak azonosítót. A node

azonosítókat azért állítjuk be kézzel, hogy a későbbi feladatok egyértelműen tudjanak hivatkozni rájuk.

Az [ndbd default] rész közös beállításokat tartalmaz az összes adat node-ra. Itt állítjuk be a kétszeresen redundáns adattárolást (NoOfReplicas), illetve azt, hogy az adat node-ok mennyi memóriát használhatnak az adatok és az indexek tárolására (DataMemory, IndexMemory).

Az [ndbd] részeknél állítjuk be egyesével az adat node-ok elérhetőségét és azonosítóját (HostName, NodeId).

Végül a [mysqld] rész szól az SQL node beállításairól. Itt az [ndbd] résszel megegyező módon állítjuk be a z SQL node elérhetőségét és azonosítóját.

A paraméternevekben szereplő kis- és nagybetűket a menedzsment node nem különbözteti meg.

A menedzsment node-okat, adat node-okat és sql node-okat szabályozó többi paraméterről itt olvashatunk:

- http://dev.mysql.com/doc/refman/5.6/en/mysql-cluster-mgm-definition.html
- http://dev.mysql.com/doc/refman/5.6/en/mysql-cluster-ndbd-definition.html
- http://dev.mysql.com/doc/refman/5.6/en/mysql-cluster-api-definition.html

Következő lépésben hozzuk létre a konfigurációs fájlt azon a gépen, ahol a menedzsment node-ot fogjuk futtatni, a piros szövegeket értelemszerűen átírva! A konfigurációs fájlt érdemes Notepad++ programmal szerkeszteni.

Mentsük a C:\mysql\bin\ helyre config.ini néven a fájlt azon a gépen, amin a menedzsment node-ot fogjuk futtatni (1. hallgató gépe). A fájl karakterkódolása *UTF-8 without BOM* legyen.

2.2 Node-ok saját beállítás fájlja

Szükséges készítsünk az egyes node-ok számára egy-egy külön beállítás fájlt. Ezekben minimálisan annak benne kell lennie, hogy az egyes node-ok a menedzsment node-ot hol érik el. Emellett más, csak az egyes node-okat érintő beállítások is megadhatók itt.

Példát a többi beállítási lehetőségre a http://dev.mysql.com/doc/refman/5.6/en/mysql-cluster-config-example.html oldalon olvashatunk.

Az adat node beállításfájlja a következőképpen néz ki:

```
[mysql_cluster]
# Options for data node process:
ndb-connectstring=<menedzsment node-ot futtató gép ip címe> #
location of management server
```

Az SQL node beállításfájlja pedig a következőképpen:

```
[mysqld]
```

```
# Options for mysqld process:
ndbcluster # run NDB storage engine
ndb-connectstring=<menedzsment node-ot futtató gép ip címe> #
location of management server
```

Itt azt is meg kell adni, hogy az SQL szervert clusterben futtatjuk, erre utal az *ndbcluster* kezdetű sor.

A menedzsment node-nak külön konfigurációs fájlt nem szükséges készítenünk, habár megtehetjük.

A node-ok futtatható állományai a beállításokat a C:\mysql\my.ini fájlban keresik alapértelmezés szerint. Minden node csak a rá vonatkozó részt veszi figyelembe, így ha egy gépen több node-ot is futtatunk, fentiekben leírt beállításaik kerülhetnek ebbe az egyetlen my.ini fájlba.

Készítsünk el egy-egy my.ini fájlt mindkét gépen a megfelelő tartalommal!

3 A rendszer indítása

3.1 A menedzsment node indítása

Először a menedzsment node-ot kell elindítsuk.

Ehhez az első hallgató gépén adjuk ki a következő parancsot a parancssorban:

```
ndb_mgmd --config-file=C:/mysql/bin/config.ini --
configdir=C:/mysql/bin/cluster-cache --reload
```

A menedzsment node futtatásához az ndb_mgmd.exe programot futtatjuk, mely az Network DataBase Management Daemon-t rövidíti. A *config-file* paraméterrel megadjuk a programnak a globális konfigurációs fájl helyét.

A menedzsment node a globális konfigurációs fájl kiolvasása után arról egy bináris mentést készít a *configdir* paraméterrel megadott mappába, a mentéseket esetenként verziózva is. A node csak akkor olvassa ki a konfigurációs fájlt, ha nem talál mentést, vagy ha arra a *reload* vagy *initial* indítási paraméterrel utasítjuk. Reload paraméter használatával a legutolsó mentést összehasonlítja a konfigurációs fájl tartalmával, ha eltérést talál, akkor új mentést készít. Initial paraméter használata esetén pedig törli a mentéseket és újat készít a konfigurációs fájl alapján. Ha a node csak a mentett bináris fájlt olvassa ki, indulása gyorsabb, a bináris fájl könnyebb feldolgozása miatt. Mivel a node a konfigurációs fájl mentéseit verziózva tárolja, az újabb bináris fájlokat törölve és a node-ot újraindítva, vissza lehet állni a régebbi beállításokra.

A menedzsment node elindítása után várja a rendszer többi node-jának elindulását.

3.2 A menedzsment kliens segédprogram indítása

A menedzsment kliens program nem szükséges a cluster működéséhez. Segítségével azonban lekérdezhető a rendszer állapota, elindítható az adatok mentése, illetve más adminisztratív

műveletek végezhetők vele. A kliens név arra utal, hogy kliens-szerver architektúrában a menedzsment node a szerver és kiszolgálást ad a menedzsment kliensnek, amely hozzá kapcsolódik.

Új parancssor (cmd) ablakban indítsuk el a menedzsment kliens programot:

```
ndb_mgm --connect-string=<menedzsment node-ot futtató gép ip címe>
```

A *connect-string* paraméter csak akkor szükséges, ha nem a menedzsment node-ot futtató gépről akarjuk a klienst indítani. A menedzsment kliens elindulása után a konzolba beírva a *show* parancsot a program kiírja a rendszer állapotát, azaz hogy az egyes node-ok futnak-e, hol találhatók és mi a node-azonosítójuk.

```
ndb_mgm> show
Connected to Management Server at: localhost:1186
Cluster Configuration
---------------------
[ndbd(NDB)] 2 node(s)
id=1 (not connected, accepting connect from localhost)
id=2 (not connected, accepting connect from localhost)

[ndb_mgmd(MGM)] 1 node(s)
id=5 @127.0.0.1 (mysql-5.6.19 ndb-7.3.6)

[mysqld(API)] 1 node(s)
id=10 (not connected, accepting connect from localhost)
```

A laborgépeken a kimenetben a *localhost* helyén a node-ok globális konfigurációs fájlban megadott IP-címei szerepelnek. Az ndbd szekció mutatja az adat node-ok, az ndb_mgmd a menedzsment node-ok és a mysqld a SQL node-ok állapotát. A kimeneten látszik, hogy még csak a menedzsment node fut, az adat node-ok és az sql node még nem.

3.3 Adat node-ok indítása

Új parancssori ablakban indítsunk el mindkét gépen egy-egy adat node-ot a következő paranccsal:

```
ndbd.exe
```

A két adat node elindulása eltarthat néhány percig. Sikeres elindulásukat a menedzsment kliensben ellenőrizhetjük. A *show* parancsot beírva az éledő adat node-ok sorában a *starting* felirat jelenik meg:

```
[ndbd(NDB)] 2 node(s)
id=1 @127.0.0.1 (mysql-5.6.19 ndb-7.3.6, starting, Nodegroup: 0,
*)
id=2 @127.0.0.1 (mysql-5.6.19 ndb-7.3.6, starting, Nodegroup: 0)
```

A @127.0.0.1 helyén (amely minden számítógépen saját magát jelenti) értelemszerűen a laborban érvényes ip címek állnak. Csillaggal az elosztott döntéshozatalban a főnök node van jelölve (ezzel a labor során nem foglalkozunk). Mivel a redundancia szintje 2, node csoportonként ennyi adat node lesz, ami látszik is fent.

Az induló adat node-ok egymással és a menedzsment node-dal is együttműködnek. Figyeljük meg a gépek hálózati forgalmát, a *Windows Task Manager* programjával! A program a laborgép asztalán található.

Ahhoz, hogy észrevegyük az adat node-ok elindulását, a menedzsment kliensben nem szükséges a show parancs ismételt kiadása, a kliens magától jelzi az adat node-ok elindulását a következőt kiírva:

```
ndb_mgm> Node 1: Started (version 7.3.6)
Node 2: Started (version 7.3.6)
```

3.4 Az SQL node indítása

A megfelelő gépen (a 2. hallgató gépén) a parancssorba írjuk be a

```
mysqld.exe
```

parancsot, mely elindítja az SQL node-ot., majd ellenőrizzük a node sikeres elindulását a menedzsment kliens *show* parancsával. Az SQL node is elindult, ha a kimeneten a mysqld szekcióban az alábbihoz hasonlót találunk (az IP-cím eltérhet):

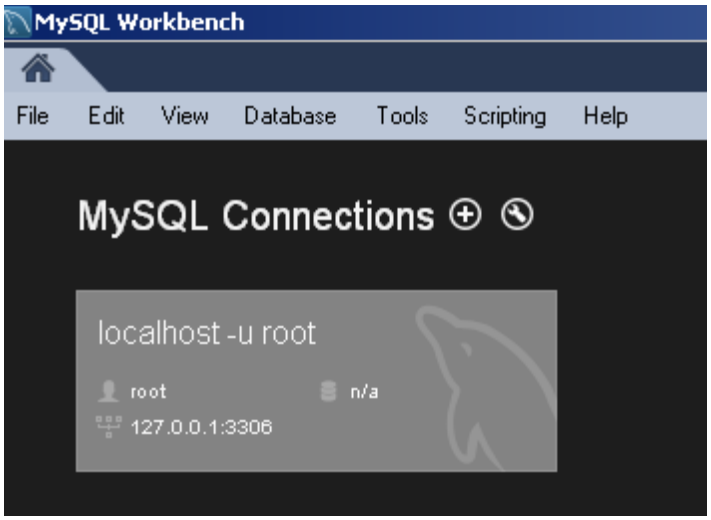
```
id=10 @127.0.0.1 (mysql-5.6.19 ndb-7.3.6)
```

4 Adatok betöltése az adatbázisba

Példaadataink, melyeket az adatbázisba töltünk, a World adatbázisból származnak, mely a MySQL weboldaláról tölthető le tesztelési célokra.

Ebben a feladatban a MySQL Workbench programot használjuk. A program az adatbázis rendszergazdáinak és fejlesztőinek nyújt hasznos funkciókat. Az eszköz többek közt adatmodellezésre és SQL fejlesztésre is alkalmas. Mi most ez utóbbira fogjuk használni. Az ingyenesen elérhető programot az Oracle kifejezetten a MySQL adatbázis-kezelő rendszerhez fejleszti.

Indítsuk el a MySQL Workbench programot azon a gépen, ahol az SQL node is fut (az ikonja kinn van az asztalon. A programban az SQL node elérhetőségére *localhost* van beállítva. Ezt átírva az SQL node gépének IP-címére, akár a másik gépen is futtathatjuk.)

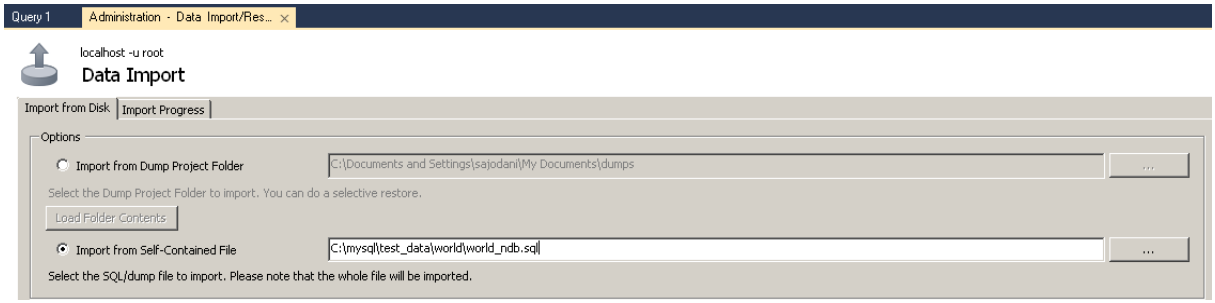


2. ábra Kapcsolódás a MySQL kiszolgálóhoz

A programot elindítva kattintsunk a szürke téglalapra (2. ábra), kiválasztva, hogy melyik SQL node-hoz csatlakozunk. A megjelenő fejlesztőkörnyezetben bal oldalon a Navigator panelen válasszuk a Data Import/Restore opciót (3. ábra).

3. ábra Adatbetöltő funkció kiválasztása

A megjelenő ablakban az Import from disk fül Import from self-contained file részében válasszuk ki a következő fájlt: C:\mysql\test_data\world\world_ndb.sql (4. ábra)



4. ábra Betöltendő adat fájl kiválasztása

A world_ndb.sql fájl hozza létre a betöltendő három tábla struktúráját, és tölti fel azokat adatokkal.

Ugyanezen a felületen lejjebb a target schema résznél hozzunk létre egy új adatbázist world néven. Tesztadataink ebbe az új adatbázisba kerülnek (5. ábra).

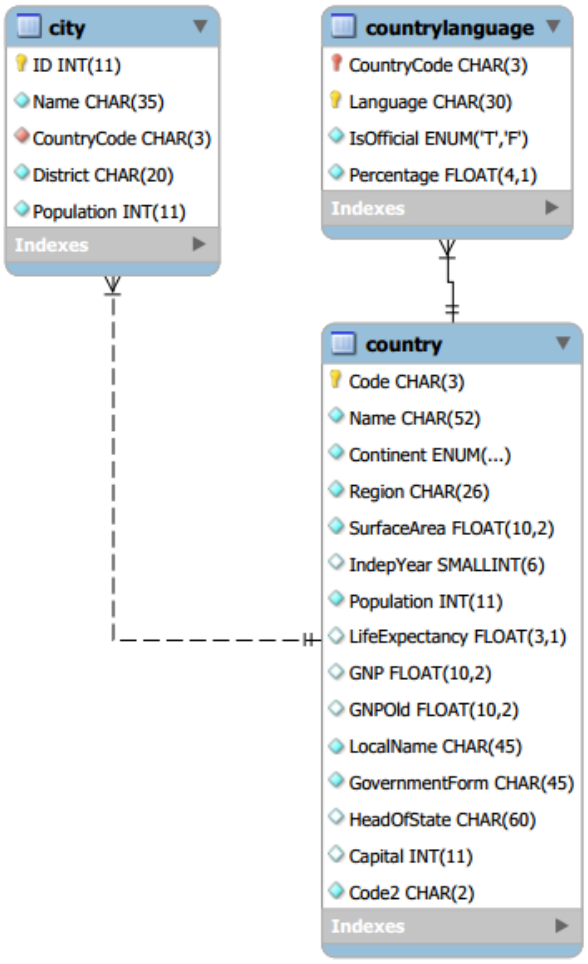
5. ábra Cél adatbázis beállítása

Itt felmerül a kérdés, hogy a MySQL rendszerben mit jelent az adatbázis (database) szó. Az összetartozó adatbázis-objektumok egy halmazát a MySQL adatbázisnak, másnéven sémának (schema) hívja. Adatbázis-objektumok például a táblák, az indexek, stb. Bővebb leírást a fogalomról a http://dev.mysql.com/doc/refman/5.6/en/glossary.html#glos_schema anyagban olvashatunk:

A jobb alsó sarokban kattintsunk a Start Import gombra, majd várjuk meg míg a program végez az adatok betöltésével.

4.1 Lekérdezés készítése

A 6. ábrán a betöltött adattáblák szerkezete látható.

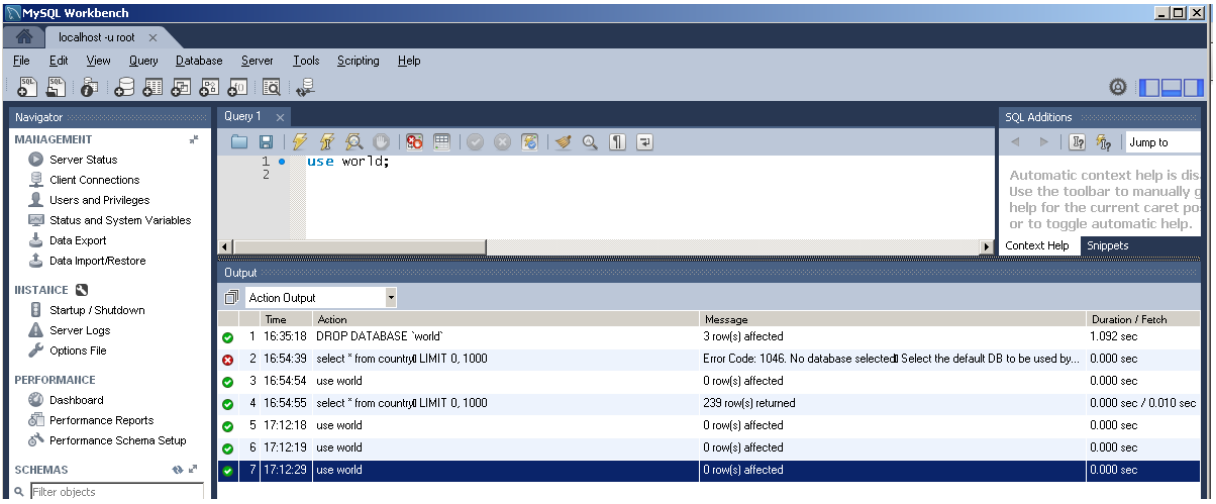


6. ábra A példaadat szerkezete

Annak érdekében, hogy SQL lekérdezéseinkben ne kelljen a tábla neve elé minden alkalommal kiírni az adatbázis nevét (world), a fejlesztőkörnyezet lekérdezés-szerkesztő részébe írjuk be a

```
use world;
```

parancsot, majd nyomjuk meg a control + enter billentyűkombinációt, úgy, hogy a kurzor a soron belül legyen, ezzel lefuttatva az utasítást (7. ábra).



7. ábra SQL utasítás sikeresen lefuttatva

A parancs sikeres lefutásáról alul az Output részben győződhetünk meg: ha a kiadott parancs megjelenik, és mellette zöld pipa ikon van, az utasítás futása sikeres.

Ezek után hajtsunk végre néhány lekérdezést! Például:

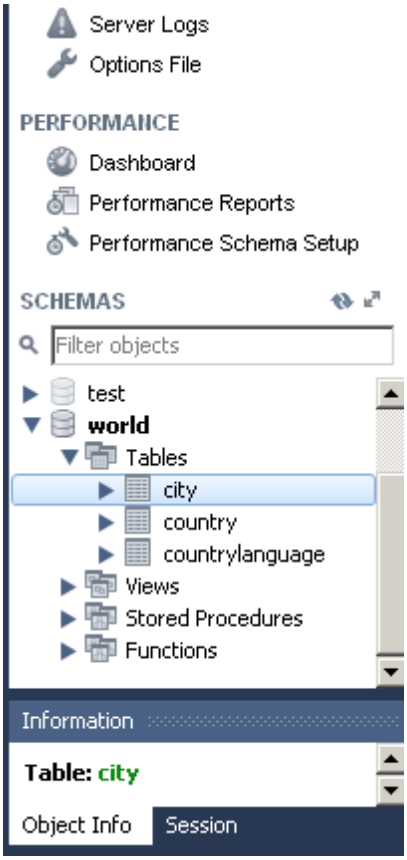
```
select
     count(*)
from
     city
where
     population > 100000
;
```

Vegyük észre, hogy a kiszolgálás nem tér el attól, amit egy nem elosztott rendszertől kapnánk. (Például nem kell tudnunk, hogy az adatok hol és hány példányban tárolódnak. Az elosztott rendszer működését a felhasználónak nem kell ismernie ahhoz, hogy a kérései kiszolgálásra kerüljenek.)

4.2 Adatok partícionálása

Ahogy az az elméleti dokumentumban is szerepel, az adatok tárolása rendszerünkben több helyen (elosztottan) és példányban (többszörözve) történik.

Következő lépésben az elosztott tárolást vizsgáljuk. Mivel 2 adatkiszolgálónk van, 2 partícióra kerülnek az adatok. Ellenőrizzük is le ezt. Kattintsunk az egyik tábla nevére jobb gombbal, majd válasszuk a *Table Inspector* menüt (8. ábra).



8. ábra Az adatbázis tábláinak listája

Ugorjunk a megjelenő felület *partitions* fülére. Itt láthatjuk, hogy a tábla adatai valóban 2 partíción kerülnek tárolásra. A megjelenő táblázat *Table Rows* oszlopában látható, hogy mindkét partícióban a tábla körülbelül ugyanannyi sora tárolódik, melynek oka a partícionálási szabályban, a táblák sorainak tárolási helyét meghatározó algoritmusban keresendő. A rendszer jelen esetben az alapértelmezett partícionálási szabályt, a *Key*-t alkalmazza. Ez az algoritumus a táblasor kulcsértékeiből egy számot képez, melyet eloszt a partíciók számával, és az osztási maradék határozza meg a tárolásra kijelölt partíciót.

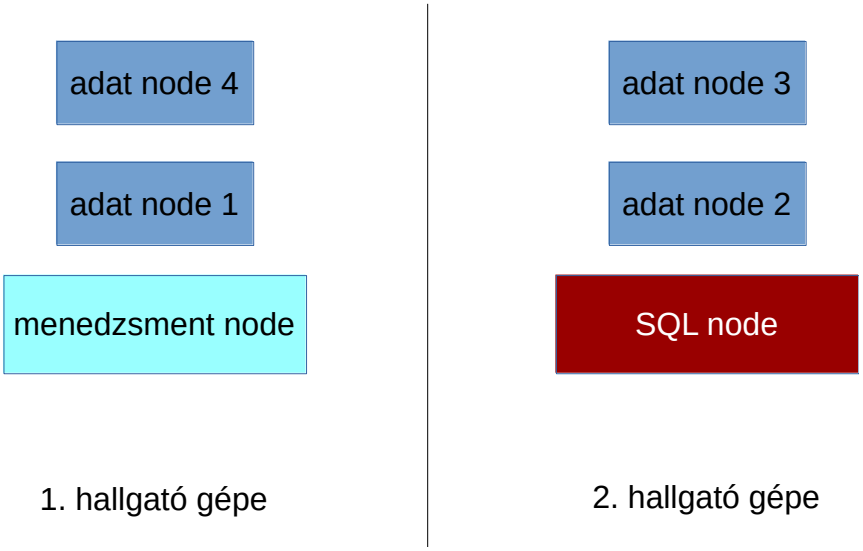
A többi partícionálási algoritmusról a http://dev.mysql.com/doc/refman/5.6/en/partitioning-types.html oldalon olvashatunk.

5 Rendszer bővítése adat node-okkal, leállás nélkül

Ebben a szakaszban a futó rendszerhez adat node-okat adunk, növelve az adattárolásra alkalmas erőforrásokat, és megvizsgáljuk a rendszer teljesítményének skálázhatóságát. A bővítés közben megfigyeljük, hogy a működés folytonos marad.

5.1 A globális konfigurációs fájl szerkesztése

Első lépésként a menedzsment node-ot futtató gépen a config.ini globális konfigurációs állomány átírása szükséges. Írjunk bele két új [ndbd] szekciót, jelezvén a rendszernek, hogy két új adat node-dal bővítjük a rendszert. A rendszer új felépítése a 9. ábrának megfelelően nézzen ki:



9. ábra Adatbázis node-jaink elosztása a 2 hallgatói gép közt

A rendszer adat node-okkal való bővítése közben a kiszolgálás nem áll le. Kivétel ez alól az SQL node újraindításának ideje, mivel ez fogadja az SQL kéréseket. Ha több SQL node is lenne rendszerünkben, és nem egyszerre indítanánk újra őket, akkor a futó sql node-on a másik újraindításának ideje alatt is kapnánk kiszolgálást.

Az újraindítás közben (kivéve az SQL node újraindítását) teszteljük a kiszolgálás folytonosságát a MySQL Workbench programban, lekérdezések végzésével!

5.2 A menedzsment node újraindítása

Következő lépésként indítsuk újra a menedzsment kiszolgálót, a szerkesztett globális konfigurációs állomány ismételt beolvasása érdekében.

Node-ot leállítani a menedzsment kliens következő utasításával kell:

```
<node azonosító> stop
```

Melyre ezt a kimenetet kapjuk:

```
ndb_mgm> <node azonosító> stop
Node <node azonosító> has shutdown.
```

A menedzsment node leállítása azzal is jár, hogy a menedzsment klienssel való kapcsolata megszakad. Előfordulhat, hogy újraindítása után a menedzsment klienst is újra kell indítani.

A korábban (3.1. fejezetben) leírt módon indítsuk el újra a menedzsment node-ot. Ügyeljünk a *--reload* paraméter használatára.

5.3 Az adat node-ok újraindítása

Következő lépésünk az adat node-ok sorban egymás után történő újraindítása. Fontos, hogy egyszerre csak egy node lehet üzemen kívül. Várjuk ki, amíg az egyik újraindul és csak utána indítsuk újra a következőt. A menedzsment node esetében kénytelenek voltunk a node-ot leállítani majd újra elindítani. Az adat node-ok esetében lehetőség van a két lépés helyett

egyet végezni:

```
<node azonosítója> restart
```

5.4 Az SQL node újraindítása

Azon a gépen, ahol az SQL node fut egy parancssorba gépeljük a következő utasítást:

```
mysqladmin -u root shutdown
```

Ennek hatására az SQL node leáll.

Az elindítás most is a mysqld.exe program futtatásával történik ugyanazon a gépen, ahol idáig is futott (a 2. hallgató gépén).

Várjuk meg míg a menedzsment kliensen azt látjuk, hogy az SQL node is üzemképes.

5.5 Új adat node-ok elindítása

A megfelelő gépeken indítsuk el az új adat node-okat az ndbd.exe futtatásával. Figyeljük meg a menedzsment kliensen, ahogy elindulnak. A node-ok egyszerre is indíthatók. Az új node-ok elindulását követően a *show* parancsot kiadva az adat node-okat leíró rész így néz ki:

```
[ndbd(NDB)] 4 node(s)
id=1 @192.168.1.2 (mysql-5.6.19 ndb-7.3.6, Nodegroup: 0, *)
id=2 @192.168.1.2 (mysql-5.6.19 ndb-7.3.6, Nodegroup: 0)
id=3 @192.168.1.2 (mysql-5.6.19 ndb-7.3.6, starting, no nodegroup)
id=4 @192.168.1.2 (mysql-5.6.19 ndb-7.3.6, starting, no nodegroup)
```

(Az IP címek értelemszerűen a megfelelő laborgépek IP címei.)

Látszik, hogy a két új adat node még nem tartozik node csoporthoz.

Ahhoz, hogy ezeket is használni tudjuk, új node csoportot kell létrehozzunk a számukra, és az új adat node-okat hozzá kell rendelni a nekik létrehozott csoporthoz. Ez a következő utasítással tehető meg a menedzsment kliensben (a parancs alatt a kliens visszajelzése is olvasható):

```
ndb_mgm> CREATE NODEGROUP 3,4
Nodegroup 1 created
```

A CREATE NODEGROUP utasítást a node csoporthoz hozzárendelendő adat node-ok azonosítóaó követi vesszővel elválasztva. A show paranccsal láthatjuk, hogy az új node csoport létrejött, és a 3. és 4. node hozzárendelése is megtörtént.

5.6 Az adatok újraelosztása

Vizsgáljuk meg, hogy melyik adat node tárolókapacitása milyen mértékig van kihasználva. Ezt a menedzsment kliens *ALL REPORT MEMORY* utasításával tehetjük meg.

Azt tapasztaljuk, hogy az új adat node-ok kapacitása egyáltalán nincs kihasználva, hiszen a

két új adat node-hoz tartozó partíciók tárterület-kihasználtsága 0%. Ennek oka, hogy az adatok újraelosztása a partíciók közt nem automatikus. Végezzük el hát kézzel minden egyes adattáblánkra (city, country, countrylanguage) az adatok újraelosztását! A MySQL Workbench sql kliensből adjuk ki következő parancsot:

```
ALTER ONLINE TABLE <tábla neve> REORGANIZE PARTITION;
```

Hogy egy tábla sorai hány partíció illetve mely partíciók közt kerülnek szétosztásra, a tábla létrehozásakor állíthatjuk be. Alapértelmezés szerint az éppen a rendszerben lévő partíciókra kerülnek a sorai. Az adatok újraelosztását követően sort hozzáadva a táblához már a négy partíció valamelyikén kerül tárolásra az új sor.

6 Adat node-ok leállítása

Vizsgáljuk meg, hogy mennyire sérülékeny a rendszerünk, hány adat node eshet ki úgy, hogy a rendszer működőképes marad. Korábbi ismereteinkből tudjuk, hogy ha node csoportonként legalább egy adat node működik, minden adat hozzáférhető-, a rendszer működőképes marad. Ha viszont egy node csoport valamennyi node-ja leáll, a rendszerünk nem tud tovább működni.

Próbáljuk ki ezt a gyakorlatban!

Ahogy korábban is tettük a node-okat most is a menedzsment kliensből állítsuk le a

```
<node azonosító> stop
```

paranccsal.

Próbáljuk meg leállítani egy node csoport valamennyi node-ját, és vegyük észre, hogy a rendszer nem engedi ezt. Miközben leállítunk node-okat, teszteljük az adatbázis működőképességét SQL lekérdezések készítésével!

A már leállított node igény esetén a 3.3. fejezetben bemutatott módon újra elindítható.

7 A rendszer leállítása

Állítsuk le az SQL node-ot a megfelelő gépen:

```
mysqldamin -u root shutdown
```

Eztán következhet az adat node-ok és a menedzsment node leállítása a menedzsment klienssel. Megtehetnénk, hogy egyesével leállítjuk a node-okat, de van egyszerűbb megoldás is:

```
shutdown
```

A parancs az összes node-ot leállítja az SQL node kivételével. Az SQL node-ot ezért kapcsoltuk ki kézzel.

Végül az

```
exit
```

utasítással zárjuk be a menedzsment klienst is!