Gyakorlati óravázlat 12 – Hierarchikus lekérdezések, illesztés gyakorlása

1. Hierarchia ábrázolása relációban
2. Hierarchikus lekérdezések logikája, pszeudo oszlopai
3. Illesztések

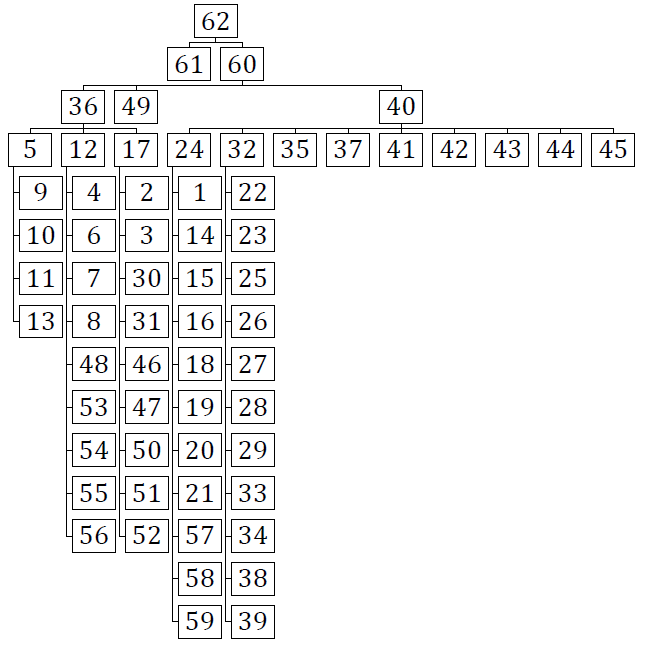
\_\_\_\_\_\_\_\_\_\_\_\_

**Koncepció**

Előadáson már esett szó arról, hogy hierarchikus szerkezet is ábrázolható, tárolható relációban. Gyakorlatilag egy 2 oszlopos táblázatról van szó, ahol az egyik oszlop egyedileg tartalmazza a hierarchia minden egyedét (annak azonosítóját), a másik oszlop pedig a közvetlen felette álló egyednek az azonosítóját. (Előfordulhatna olyan gráf is, amiben van loop, azokra is vannak megoldások, de azokkal a kurzus keretében nem foglalkozunk!)

Fontos, hogy a két oszlopban ugyanolyan adatok szerepelnek, pl dolgozók azonosítója: ki kinek a főnöke, vagy földrajzi helyek, melyik hely melyiknek a része, vagy szervezeti egységek, melyik szervezeti egység melyiknek a része, vagy termékkategóriák amelyek egymás főkategóriái. A lefelső szinten lévő egyedeknek nyilván nincsen felettese. Ennek ábrázolása kétféleképpen lehetséges: vagy null érték szerepel a második oszlopban, vagy egyáltalán nem szerepeltetjük őket az első oszlopban. Mi az utóbbit fogjuk alkalmazni.

Az órai feladatokhoz a MINTA séma DOLGOZO és FONOK tábláit fogjuk használni (MDOLGOZO, MFONOK szinonímák is használhatóak). Előbbiben 62 dolgozó adatait találjuk, utóbbiban azt, hogy ennek a 62 dolgozónak ki a főnöke. A FONOK tábla által létrehozott hierarchia képét odaadom több formátumban, hogy fel tudjátok használni órán a magyarázathoz. Az első két feladatban le is másoljuk magunkhoz a táblákat, azért hogy egyrészt gyakoroljuk az ALTER TABLE utasítással utólag létrehozott kulcs és idegen kulcs létrehozását. Másrészt azért, hogy a létrehozott idegen kulcsokkal tudatosítsuk, hogy mindkét oszlopban ugyanúgy a dolgozók azonosítói szerepelnek.



A gyakorló feladatokhoz a főnökökön kívül a minta séma kereskedelmi adatbázisát egészítem ki egy kategória táblával. Az adatbázis adatait is módosítottam hozzá kicsit, hogy több szintet tudjak létrehozni a kategóriákban. Ehhez rajz sem lesz. Az a megértés egy magasabb foka, hogy csak a tábla ismeretében is tudják értelmezni a feladatokat.

**Hierarchikus lekérdezés felépítése, pszeudooszlopok, hasznos függvények**

Hierarchikus lekérdezés létrehozásához a CONNECT BY feltételt kell megadnunk. Ennek helye a SELECT utasításban a WHERE rész után van. A CONNECT BY után kell megadnunk, hogy milyen irányban kell bejárni a táblázatot. A PRIOR szóval adjuk meg a bejárás irányát. Azaz a főnökös példánál maradva

* CONNECT BY fonok\_id = PRIOR id  
  veszi az adott rekordot, aztán folytatja az(ok)kal a rekord(ok)kal, ahol az adott rekordban id-ként szereplő érték főnök\_id-ként szerepel.
* CONNECT BY PRIOR fonok\_id = id  
  veszi az adott rekordot, aztán folytatja az(ok)kal a rekord(ok)kal, ahol az adott rekordban fonok\_id-ként szereplő érték id-ként szerepel.

Az első lekérdezéseinket érdemes kiegészíteni két plusz oszloppal és egy sorbarendezéssel, hátha úgy jobban érthetővé válik a lekérdezés működése:

* LEVEL pszeudo oszlop: megmutatja, hogy az adott rekord egy kiinduláshoz képest hányadik szinten jelenik meg.
* sys\_connect\_by\_path függvény: megmutatja az utat az adott kiinduló rekordtól kezdve az aktuálisan kiírt rekordig

**Feladatok**

1. Hozzuk létre a MINTA séma DOLGOZO és FONOK tábláinak másolatát a saját sémánkban!
2. Módosítsuk a táblákat úgy, hogy a DOLGOZO táblának legyen kulcsa, a FONOK tábla pedig idegen kulcsokkal hivatkozzon a DOLGOZO tábla kulcsára!
3. Kérdezzük le az egyes beosztottak összes főnökét (nem csak a közvetleneket)! Írjuk ki azt is, hogy az adott főnök hány szinttel van feljebb és azt is, hogy kik a köztes főnökök!

SELECT sys\_connect\_by\_path(to\_char(ID,'00'), '/' ) AS kapcsolati\_ut

,LEVEL,ID,fonok\_id

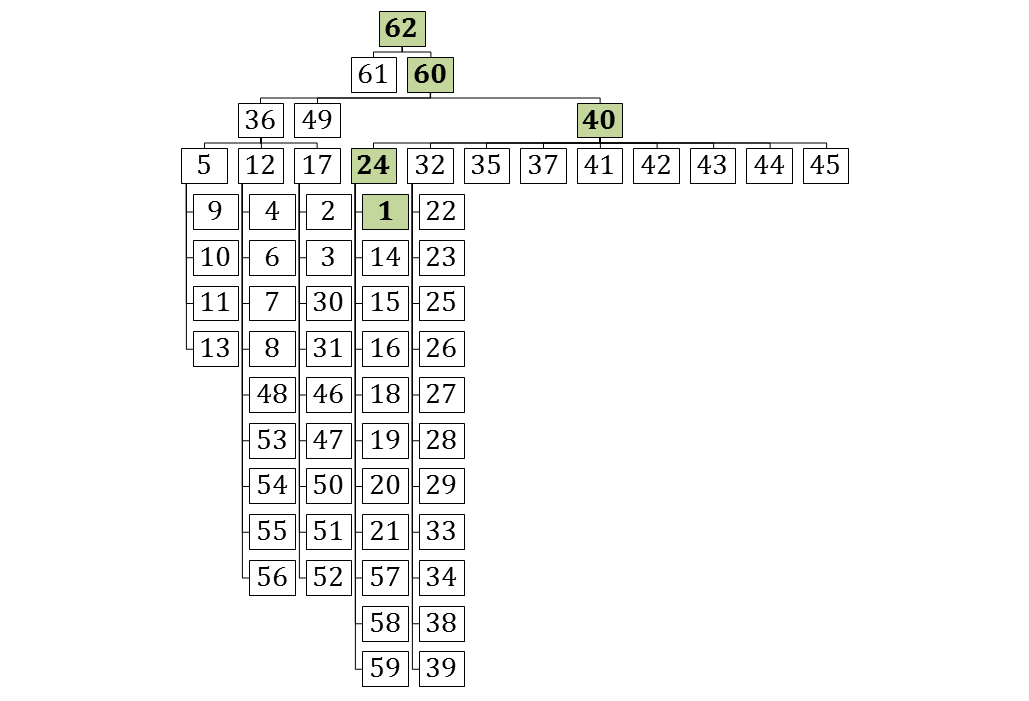
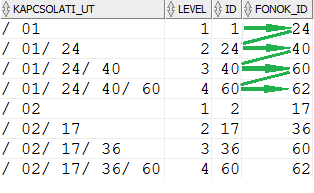
FROM fonok

CONNECT BY ID = PRIOR fonok\_id

ORDER BY kapcsolati\_ut,LEVEL,ID,fonok\_id

;

Idemásolom az eredménytáblázat egy darabját is, és azt is meg lehet mutatni a kivetített hierarchián, hogy ez mit jelent pontosan:



Tehát az 1-es dolgozónak 4 főnöke van, a közvetlen a 24-es, neki a főnöke a 40-es és így tovább.

**Feladatok**

1. Kérdezzük le csak az 1-es kódú dolgozó főnökeit!

A CONNECT BY elé kell beírni az indító feltételt. Minden olyan rekordtól be fogja járni a fát, ami megfelel a feltételnek. Ha a feltétel egytlen azonosítót határoz meg, akkor az adott azonosítójú dolgozók főnökeit fogjuk megkapni. Ha csak a főnökökre vagyunk kiváncsiak, akkor nem szükséges kiírni az útvonalat, szintet sőt az egyes lépésekben az alkalmazottak id-jét sem, hiszen azok mindig megjelennek előtte főnökként is.

1. Írjuk ki az 1-es kódú dolgozó főnökeinek azonosítóját és a nevét!

SELECT fonok\_id, vnev||' '||knev bana\_árpádné\_főnökei

FROM fonok JOIN dolgozo

ON (fonok.fonok\_id=dolgozo.ID) --!!

START WITH fonok.ID = 1

CONNECT BY fonok.ID = PRIOR fonok\_id

ORDER BY LEVEL

;

Ahhoz, hogy az azonosítókhoz egyéb adatokat is tudjunk kapcsolni, természetesen a táblák illesztésére van szükség. Mivel a FONOK tábla két idegen kulccsal is kapcsolódik a DOLGOZO táblához, mindig figyeljünk nagyon oda, hogy épp melyik kapcsolatot akarjuk megvalósítani (esetleg mind a kettőt?)

1. Kérdezzük le az egyes főnökök összes beosztottját (nem csak a közvetlen beosztottakat)! Írjuk ki azt is, hogy hányadik szintű a beosztott és azt is, hogy kik a köztes főnökök! (Fordítsuk meg a 3. feladatot!)
2. írjuk ki csak a 40-es kódú dolgozó beosztottjait!
3. Írjuk ki azokat a főnököket, akiknek van 30 évesnél fiatalabb női alkalmazottjuk! Írjuk ki, hogy hány ilyen alkalmazottjuk van (csak azokat írjuk ki, akiknek egyáltalán van.

SELECT \* FROM dolgozo

WHERE nem=1

AND szuldat>SYSDATE-30\*365.25;

SELECT DISTINCT d2.ID fonok\_id, d2.vnev||' '||d2.knev fonok\_nev

FROM

dolgozo d1 JOIN fonok ON d1.ID=fonok.ID

JOIN dolgozo d2 ON fonok.fonok\_id=d2.ID

START WITH d1.nem=1 AND d1.szuldat>SYSDATE-30\*365.25

CONNECT BY fonok.ID = PRIOR fonok.fonok\_id

;

SELECT d2.ID fonok\_id, d2.vnev||' '||d2.knev fonok\_nev,

count(\*) alkalmazottak\_száma

FROM

dolgozo d1 JOIN fonok ON d1.ID=fonok.ID

JOIN dolgozo d2 ON fonok.fonok\_id=d2.ID

START WITH d1.nem=1 AND d1.szuldat>SYSDATE-30\*365.25

CONNECT BY fonok.ID = PRIOR fonok.fonok\_id

GROUP BY d2.ID, d2.vnev||' '||d2.knev

ORDER BY alkalmazottak\_száma DESC;

Mivel a feltételünk a beosztottakra vonatkozik, kiírni meg a főnökök nevét akarjuk, ezért a DOLGOZO táblára kétszer is szükségünk lesz, egyszer az id-n keresztül, egyszer pedig a fonok\_id-n keresztül van kapcsolat. El kell nevezni a táblázatot kétféleképpen, hogy egyértelműen meg tudjuk mondani, hogy melyik táblázat oszlopaira hivatkozunk. Az, hogy 30 évnél fiatalabb nők főnökeit keressük a START WITH feltételben jelenik meg, azaz azoknál a rekordoknál indítjuk a bejárást, amelyekre ez az összetett feltétel igaz. A DISTINCT azért kell, mert pl. a vezérigazgató nyilván mindegyiküknek a főnöke, felesleges minden bejárás eredményeképpen kiírni. Ugyanezért könnyű a feladat második felét megoldani. Minden főnök annyiszor fog szerepelni a kiírásban, ahány alkalmazott főnöklistájában megjelenik, tehát a count függvény azt fogja megadni, hogy hány alkalmazottjuk van, tehát pont megkapjuk a választ a kérdésünkre.

**Önálló feladatok:**

1. !