Gyakorlati óravázlat 06 – Adatbázis tervezése

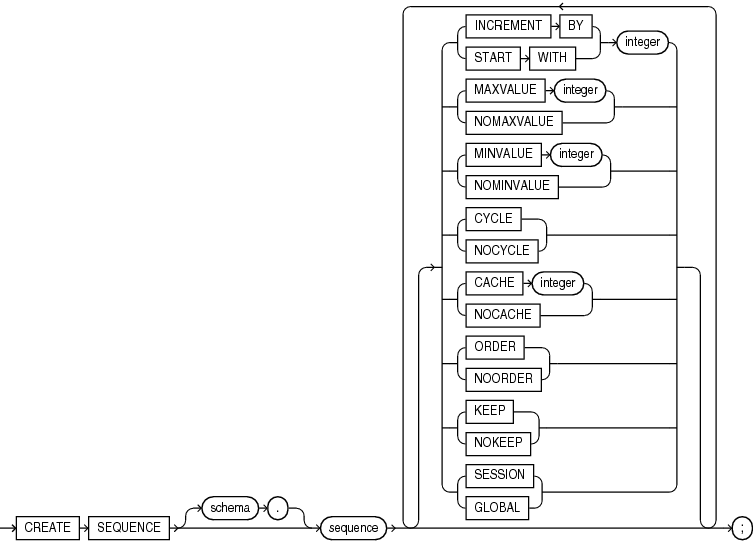
1. Adatbázis létrehozása: sequence, identity, foreign key
2. Adatbázis tervezése: egyedek, kapcsolatok, mezők, adattípusok, kényszerek

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

**Sequence létrehozása**

Sorszámgenerátor objektum. Segítségével biztosítható, hogy mindig különböző értéket tudjunk kiolvasni. Korábban külön létre kellett hozni és egy trigger segítségével lehet megoldani, hogy automatikus sorszámozás jöjjön létre. Az újabb Oracle verziókban erre már nincs szükség, a következő példában majd meglátjuk, hogy mivel lehet kiváltani. Ennek ellenére nézzünk meg egy sequencet, hogy aztán megértsük, hogy mi zajlik a háttérben.

A sorszámgenerátornak kötelezően csak nevet kell adnunk, de opcionlisan megadhatunk rengeteg mást is (dokumentációban lehet bogarászni). A legöbb dolognak egyértelmű a megnevezése, mondjuk a START WITH (kezdőérték), INCREMENT BY (lépésköz). Pár szót mondhatunk ezekről, de abszolút nem szükséges belemenni. Elég sűrű óra lesz enélkül is.



A sorszámgenerátor következő értékét így lehet lekérdezni:

select név.nextval from dual;

Az aktuális értéket is le lehet kérdezni, a nextval helyett currval-lal, de ez minden sessionben csak azután működik, hogy már volt lekérdezve nextval az adott sessionben!

**Feladatok:**

1. Hozzunk létre egy sorszámgenerátort (Sequence) és kérdezzük le a következő értékét, aktuális értékét!
2. Töröljük ki az összes táblánakat és sorozatunkat!

SELECT \* FROM user\_objects

ORDER BY object\_type;

SELECT

'drop '|| object\_type ||' '

||object\_name

||case when lower(object\_type)='table' then ' purge;' else ';' end Lefuttatni

FROM user\_objects

where lower(object\_type) in ('table','sequence') and lower(generated)='n'

ORDER BY timestamp DESC;

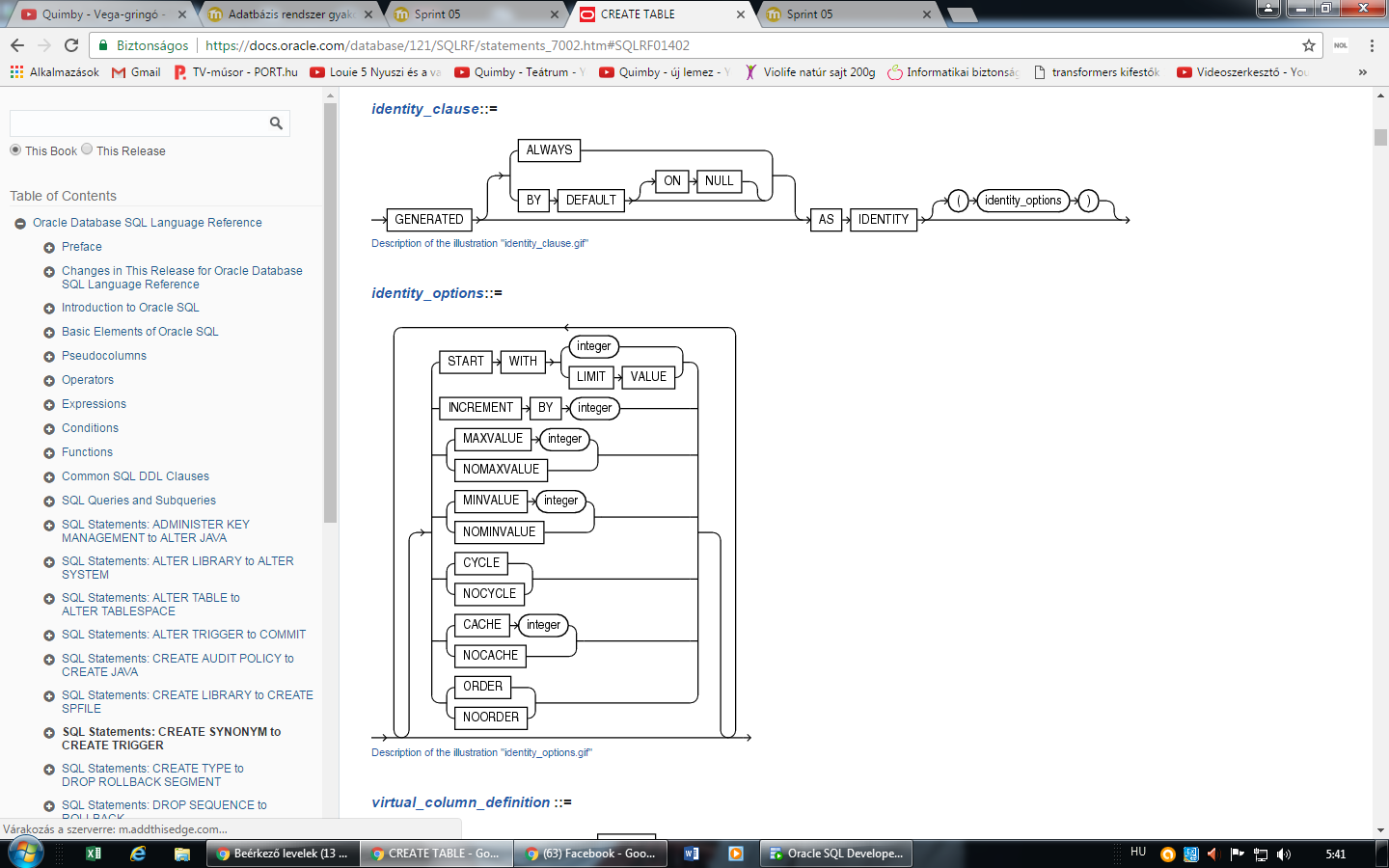
**Magyarázat**: a múlt órán mutatott ALL\_TABLES és párja a USER\_TABLES mellett nagyon hasznos ismerni az ALL\_OBJECTS vagy USER\_OBJECTS katalógustáblát is. Itt nem csak a tábláinkat látjuk, hanem minden objektumot. Többletinformáció: létrehozás és utolsó modosítási ideje (CREATED, LAST\_DDL\_TIME), illetve TIMESTEMP, ami az utolsó módosítás dátuma pontosabban. Ezeken kívül is számos oszlop, érdekes lehet még a GENERATED, hiszen pl. indexünk eddig csak olyan van, amit nem mi hoztunk létre explicit, hanem a rendszer hozta létre egy-kényszer betartatására (primary key, unique). A generált objektumoknak az a tulajdonságuk, hogy explicit nem lehet őket módosítani, vagy eldobni, hanem akkor törlődnek, ha törlődik az objektum, amihez kapcsolódnak.

A második select elég összetettnek tűnik, de valójában selectlista egyetlen elemet tartalmaz: szövegösszefűzéssel egy drop utasítást. A drop utasítás változó elemeit a USER\_OBJECTS táblából kérdezzük le, azaz az objektum típusát, nevét, illetve ha a típus tábla, akkor a PURGE kulcsszót is odaírjuk a végére. A where-ben megadjuk, hogy csak tábla vagy sequence típus és csak a nem generáltak. A rendezés jelentősége később fog kiderülni, de érdemes ezt az utasítást megőrizni, mert máskor is praktikus lehet (illetve ennek egy módosított változatával fogok majd néha sémákat takarítani pl.). Szóval a rendezés azt mondja, hogy fordított sorrendben dobjuk el az objektumokat, mint ahogy létrehoztuk őket. Ez azért jó, mert akkor nagy valószínűséggel az egymásra hivatkozó táblákat is jó sorrenben sikerül majd eldobnunk.

Lefuttatás után az eredményül kapott utasításokat ctr+c ctr+v segítségével visszamásolhatjuk a worksheetre és kijelölés után egyben le tudjuk őket futtatni. Csak a tábláinkat és a sorszámainkat dobjuk el (abból is csak azokat, amiket nem mi generáltunk – egyelőre még csak ilyenünk lesz), mégis ha újra lekérdezzük a USER\_OBJECTS-et, akkor azt látjuk, hogy az indexek is eltűntek.

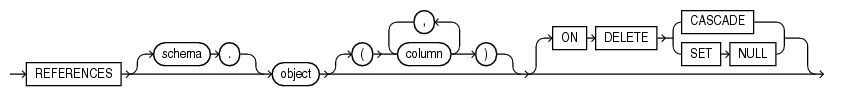
**Adatbázis létrehozása**

Egyrészt nézzük meg az **IDENTITY** használatát, amit a tábla oszlopának definiálásakor tudunk megadni ugyanott, ahol DEFAULT-ot tudnánk (a kettő kizárja egymást).



* Always vagy elhagyás esetén nem is enged semmit beírni az oszlopba, by default esetén csak üresen hagyáskor írja felül. Akár így, akár úgy, automatikusan not null lesz.
* A háttérben egy sequence jön létre, ha törlünk, újra beszúrunk, akkor a számláló automatikusan tovább megy.
* Hogyan döntsük el, hogy ALWAYS vagy BY DEFAULT legyen? Véletlenül törölt felhasználót lehet-e a korábbi kódjával visszaírni például? Az sql fájlban lévő megoldás szerepel így is úgy is.
* Láthatjuk a dokumentációban identity\_options-ként kb. ugyanazokat a tulajdonságokat lehet megadni, amit a sequence-nél. Ezek a háttérben automatikusan létrejövő sequence tulajdonságai lesznek.

Másrészt nézzük az **idegen kulcsot**! Ez egy kényszer. Meg lehet adni inline vagy out-of-line is.

* 
* Inline simán REFERENCES és megadni az objektumot és annak oszlopát, amire hivatkozunk (ha az oszlopot elhagyjuk, akkor az azonos nevű oszlopra hivatkozik, biztonságosabb, ha úgy szokjuk meg, hogy megadjuk).
* Fontos, hogy a hivatkozó mező és a hivatkozott mező adattípusa azonos legyen (mezőméret mindegy).
* A videóból kifelejtettem, de azért elmondhatjuk, hogy alapértelmezetten hivatkozott értéket nem lehet kitörölni a hivatkozott táblából. Ha viszont az idegenkulcsot ON DELETE klauzulával hozzuk létre, akkor mégis van lehetőség törlésre. A CASCADE tovagyűrűzést jelent, azaz a hivatkozó rekordok is automatikusan törlődnek. Vigyázat! Nem a törléskor tudom eldönteni, hogy akarok-e tovagyűrűzést, hanem az idegen kulcs definiálásakor! A SET NULL azt jelenti, hogy a hivatkozott adat törlésekor a hivatkozó táblában NULL érték lesz. Természetesen ezt csak akkor van értelme beállítani, ha egyébként az adott mezőre nincs NOT NULL feltételünk.
* Idegen kulcs csak másik tábla UNIQUE tulajdonságú mezőjére hivatkozhat (nem feltétlenül kell a hivatkozott mezőnek kulcsnak lenni! azaz a null érték megengedett. Az viszont fontos, hogy az adattípusnak meg kell egyezni.).
* (Viszonylag ritka, de létezik több oszlopos idegen kulcs is, nyilván a hivatkozó és hivatkozott oszlopok darabszámának meg kell egyezni és ebben az esetben az oszlopok kombinációjának kell UNIQUE-nak lenni a hivatkozott táblában – ezt csak out-of-line lehet megadni.)
* Out-of-line megadásnál szükséges a FOREIGN KEY (mezőlista) megadása is, hogy meg tudjuk jelölni, hogy melyik oszlop (esetleg oszlopkombináció) lesz az idegen kulcs. A mintapéldában létrehozom inline és out-of-line is. A végeredmény ugyanaz lesz.

**Feladatok**

1. Hozzunk létre egy KOLCSONZO táblát, azonosító és név legyen benne. Az azonosító automatikusan generálódjon! Szúrjunk be néhány adatot próbaképpen! (Visszavonás után szúrjuk be őket újra és kérdezzük le a táblát, látni fogjuk, hogy a számláló bizony továbbment.) Dobjuk el és hozzuk létre újra BY DEFAULT-tal is. Próbáljuk ki így is az adatbevitelt!
2. Hozzunk létre egy kölcsönzés táblát! Sorszámmal, hivatkozással a kolcsonzo táblára és könyvcímmel!

Kérdezzük le újra a USER\_OBJECTS rekordjait és kérdezzük le a USER\_CONSTRAINTS-et is. Ha felmerülne valakiben, hogy mégis honnan lehet a katalógustáblákat kitalálni két választ is lehet adni: vagy a dokumentációban eggyel feljebbi szintre lépünk (Administration) és ott a Reference pontot választjuk, akkor eljutunk a Data Dictionary leírásokig: <https://docs.oracle.com/database/121/REFRN/GUID-8865F65B-EF6D-44A5-B0A1-3179EFF0C36A.htm> Vagy egyszerűbben az ALL\_OBJECTS-ből lekérdezzük a USER\_ vagy az ALL\_ kezdetű szinonímákat (amelyikre szükségünk van)! Van egy olyan katalógustábla is, ami a táblák kommentjeit tartalmazza. Ha onnan kérdezzük le, akkor leírást is kapunk az egyes táblákról (lehet a lekérdezést szűkíteni like feltételekkel ha van róla elképzelésünk, hogy milyen szó szerepelhet a katalógustábla nevében):

select \* from

all\_tab\_comments

where table\_name like 'USER\_%'

order by 2;

**Adatbázis megtervezése**

Adok néhány szempontot, ami alapján órán a csoportnak közösen kellene megterveznie egy kis adatbázist. Az adatbázist majd minden csoportnak egy külön sémában kell létrehoznia. Az egyes csoportokban egymástól kisebb mértékben eltérő megoldások is születhetnek (pl. nevek egyben vagy külön, igaz hamis mezők jelölése, adattípusok mezőmérete, esetleg check feltétel kontra külön tábla).

Első körben papíron/táblán dolgozunk az órán, aztán jöhet a táblák létrehozása egyesével, a csoportsémába az oktató hozza létre. Az oktató tárolja le az sql-t és majd közzé tesszük a hallgatóknak valamilyen módon, mert most nem lesz elég egyszerűen a táblák lemásolása (az a kényszereket nem hozza át).

Adok adatokat majd mindegyik feladathoz a jövő héten. Valószínű feltöltöm a közös sémákat egy tábla kivételével, amit majd órán imortálunk be.

A következő gyakorlaton lesznek az alter utasítások mindenféle módosításra. Fogunk hozzá új táblát létrehozni, azt idegen kulccsal kapcsolni. Egy oszlopot átkódolni, esetleg adattípust váltani (azt ugye nem lehet, hanem hozzáadjuk a másik típusú oszlopot, beírjuk az átkódolt értéket, majd töröljük az eredeti oszlopot és ha szükséges, visszanevezzük az új oszlopot az eredeti névre)

Indexek is a következő gyakorlaton lesznek. Aztán a nézetek majd később, ha már lesznek bonyolultabb lekérdezéseink.

Az egyes gyakorlati csoportok sémaneve:

|  |  |  |  |
| --- | --- | --- | --- |
| G01 | K:09:50-11:20(S.2.220); | Haskó József | G01\_Szallashely |
| G12 | K:09:50-11:20(S.3.302); | Dr. Baksa-Haskó Gabriella | G12\_Szallashely |
| G02 | K:13:40-15:10(S.2.217); | Varga-Dudás Zita, Nagy Ádám | G02\_Szallashely |
| G11 | K:13:40-15:10(S.2.219); | Haskó József | G11\_Szallashely |
| G03 | K:15:30-17:00(S.2.219); | Haskó József | G03\_Szallashely |
| G04 | K:15:30-17:00(S.2.220); | Sütöri Kitti | G04\_Egyetem |
| G06 | K:17:20-18:50(S.2.219); | Kiss András Károly | G06\_Egyetem |
| G08 | K:19:10-20:40(S.2.219); | Kiss András Károly | G08\_Egyetem |
| G13 | K:19:10-20:40(S.2.220); | Steixner Balázs | G13\_Egyetem |
| G05 | CS:09:50-11:20(S.2.218); | Haskó József | G05\_Webshop |
| G07 | CS:11:40-13:10(S.2.218); | Haskó József | G07\_Webshop |
| G14 | CS:11:40-13:10(S.3.302); | Dr. Baksa-Haskó Gabriella | G14\_Webshop |
| G09 | CS:13:40-15:10(S.2.217); | Dr. Baksa-Haskó Gabriella | G09\_Webshop |
| G10 | CS:15:30-17:00(S.2.219); | Haskó József | G10\_Webshop |

**(Ha elkészülnek a sémák, akkor meg fogom adni a jogot az adott csoport hallgatóinak insert-delete-update utasításokra valamelyik táblára és akkor lehet majd tranzakciókezelést tesztelni valamelyik következő órán!)**

Tanácsok a tervezéshez, bármelyik téma is legyen: mindenképpen legyen egy kész tervetek, amihez tudtok igazodni. Ez akkor is jól jön, ha nagyon passzív a társaság. Akkor irányítottabb kérdésekkel lehet azt (vagy ahhoz nagyon közelit) létrehozni, amit magatok kigondoltatok. Meg akkor is jól jön, ha nagyon aktív a társaság és nagyon el akarnak kalandozni. Éredems őket most időkorlát miatt is több egyszerűsítésre rábeszélni. Szóban lehet szárnylani, megbeszélhetjük, hogy valójában ezt így kellene, de most egyszerűsítésnek pl. nem szedjük szét a cím részeit (legalább lehet majd rá jó kis szövegillesztéses lekérdezéseket írni). Már csak azért se bonyolítsátok túl a modlleket, mert aztán nekem fel kell ezeket töltenem adatokkal.

**A Szálláshely séma instrukciói:**

Egy szálláshely közvetítő oldal egyszerűsített adatbázisát kell megterveznünk, amelyben tároljuk a szálláshelyek, az ügyfelek és a foglalások adatait. Az ügyfeleink magánszemélyek, akik regisztrációval kerülnek az adatbázisunkba, ahol meg kell adniuk logint (jelszóval most nem foglalkozunk, titkosítás is kellene hozzá), e-mail címet, nevet, születési adatokat, számlázási címet. Nem lehet két regisztráció azonos e-mail címmel.

A szálláshelyekről tárolunk különböző adatokat: legyen nevük, típusuk, kapacitásuk (szobaszám, összes férőhely), címük, területi elhelyezkedésük (itt különböző szintű földrajzi helyet is megadhatnak, akár városrészt, vagy országon belüli területi egységet). (Később, ha belefér a kurzusba, majd lesz egy hierarchikus adatokat tartalmazó táblánk a földrajzi egységekről – mi minek a része). A szálláshelyeket munkatársaink rögzítik az adatbázisban, így legyen egy mező, ami a felvivő azonosítóját tárolja és a felvitel dátumát (mindkettőt default).

A szálláshelyek szobáit is tároljuk le külön-külön. Itt volna lehetőség brutálisan sok mező használatára, de az órai egyszerűsítés érdekében csak alap adatok legyenek, azonosító, szobaszám, ágyszám, max két eldöntendő tulajdonság (pl. pótágyazható, légkondi).

Tároljuk még a foglalások adatait is. Melyik felhasználó, mikor, mikorra hány fő felnőtt, hány fő gyerekre foglalta. Lehet itt egy megjegyzés rovat is.

**Az Egyetem séma instrukciói:**

Egy egyetem egyszerűsített adatbázisát kell megterveznünk, amelyben tároljuk a hallgatók adatait, valamint a tárgyakat és az abból elért eredményeket. Az eredmények félévekre oszlanak. A tárgyak különböző szervezeti egységekhez tartoznak, melyek hierarchiába szerveződhetnek (ezt csak utólag hozzuk majd létre külön táblaként, ha belefér majd a kurzusba).

A hallgatókról tároljuk az egyedi azonosítójukat, nevüket, születésüket. Ezen kívül a tanulmányaik kezdésének dátumát (ami valamelyik év szeptember 1. vagy február 1. lehet), illetve a szakját.

A tantárgyaknak van egyedi kódja, nem feltétlenül egyedi neve, kreditértéke és besorolása (kötelező, kötelezően választható, választható), tantárgyfelelőse és egy szervezeti egység, amihez tartozik. Ezen kívül van egy leírás mező is a tantárgyleírás számára.

A hallgatók a tárgyakat különböző félévekben vehetik fel és ott különböző eredményeik lehetnek. Egy tárgyat többször is fel lehet venni és egy felvétel alkalmával is több eredmény születhet. Az eredmény lehet egy osztályzat, vagy egyéb (nem vizsgázott, nem jelent meg, igazoltan nem jelent meg). Az eredményeket munkatársaink rögzítik az adatbázisban, így legyen egy mező, ami a felvivő azonosítóját tárolja és a felvitel dátumát (mindkettőt default).

**A Webáruház séma instrukciói:**

Egy webáruház egyszerűsített adatbázisát kell megterveznünk, amelyben tároljuk a termékeket, az ügyfeleket és a rendeléseiket.

A termékekről tárolunk egy egyedi termékkódot, megnevezést, listaárát kategóriát (a kategóriákat hierarchiába sorolva egy külön táblában tároljuk), leírást, melyik raktárunkban van és mekkora készletünk van belőle. Legyen egy hosszabb leírás a termékről. A termékeket munkatársaink rögzítik az adatbázisban, így legyen egy mező, ami a felvivő azonosítóját tárolja és a felvitel dátumát (mindkettőt default).

A vevőink regisztrációval kerülnek az adatbázisunkba, ahol meg kell adniuk logint (jelszóval most nem foglalkozunk, titkosítás is kellene hozzá), e-mail címet, nevet, születési adatokat, nemet, számlázási címet. Nem lehet két regisztráció azonos e-mail címmel.

Egy időben egy rendelést lehet leadni egyféle kiszállítási dátummal, móddal és címmel és számlázási címmel, de egy rendelési kosárban több tétel is szerepelhet, ahol az ár eltérhet a listaártól.