Tárolási alapfogalmak, objektumok, adatszótár nézetek (Oracle)

Adatbázis-kezelés

Fizikai és logikai szint

- Az adatbázisok vizsgálatánál elkülönítünk egy fizikai és egy logikai szintet.
- Logikai szinten „fogalmakkal” dolgozunk és közömbös számunkra, hogy ezek miként tárolódnak, illetve miként valósítja meg azokat a rendszer. A logikai szinthez tartoznak például a táblák, indexek, beépített függvények stb.
- A fizikai szinten konkrétan azt vizsgáljuk, hogy az egyes adatbázis-objektumok miként tárolódnak, miként implementálják az egyes adatbázis-műveleteket stb.
- A logikai adattárolás legnagyobb egységei a **tablespace**k, míg a fizikai tárolás **datafile**ok formájában történik.

Táblaterületek és Adatfájlok

- A tablespacek adatait fizikailag egy vagy több datafile tárolja.
- A fájlrendszer további részeit képezik még a visszagörgetéshez használt **Redo log** fájlok, illetve az adatbázis indításához és működéséhez elengedhetetlen **Control** fájlok.
- Az adatbázisunk méretét háromféleképpen növelhetjük:
  - új tablespacet hozunk létre
  - egy már létező tablespacehez új datafilet adunk hozzá
  - egy már létező datafile méretet növeljük (engedélyezhetjük, hogy az adatbázis ezt dinamikus hajtsa végre, amint szüksége van több területre)

Táblaterületek I.

- Általában egy-egy adatbázis több táblaterületből (tablespace) áll.
- A táblaterület a „valamilyen szempontból összetartozó” adatbázis komponenseket gyűjti egybe.
- Minden adatbázishoz hozzátartozik egy SYSTEM táblaterület, amely a rendszerkatalógus-táblákat tárolja.
- Általában az a jó, ha ezen a táblaterületen nem is helyezkedik el más, a katalógusokra ugyanis az Oracle-nek folyamatosan szüksége van. Ha más is használná a táblaterületet, az könnyen töredezetté válhatna, ami rontaná a teljesítményt.
- A SYSAUX táblaterület SYSTEM táblaterület mellett létrehozott segédterület. Több adatbázis komponens is alapesetben itt tárolódik.
- Azok az adatbázisra vonatkozó metaadatok, amelyek nem a SYSTEM táblaterületen tárolódnak, szintén idekerülnek.

Táblaterületek II.

- Az UNDO táblaterület elsősorban a tranzakciók végrehajtásánál használja az Oracle, ha nem akarjuk manuálisan kezelni a visszagörgetési táblaterületeket.
- Ez táblaterület biztosítja, hogy a sikertelen tranzakciókat visszagörgethesse a rendszer.
- Egy éppen változás alatt lévő tábláról szintén ennek segítségével kaphatnak konzisztens nézetet a felhasználók.
- A felhasználóknak egy külön USER táblaterületet szoktak létrehozni.
- Ezenkívül legalább egy ideiglenes (TEMPORARY) táblaterületet is létre szoktak hozni az ideiglenes objektumok számára. Ha ez nincs, az Oracle a SYSTEM táblaterületen tárolja az ideiglenes adatokat.
- Sok esetben érdemes még egy ideiglenes táblaterületet definiálni csak a rendezések számára. Rendezést használnak: a DISTINCT, GROUP BY, ORDER BY műveletek, a halmazműveletek, de a statisztikák gyűjtésénél, indexek felépítésénél szintén szükség van rendezésre.

Séma, objektumok

- Egy-egy séma (**Schema**) az egy-egy felhasználó tulajdonában lévő objektumokat tárolja, neve megegyezik a felhasználó nevével.

- A sémabeli objektumokra <sema_nev>.<objektum_nev> formában lehet hivatkozni.
- Példa: balhal.szeret.

- A sémák és tablespacek között nincs semmilyen összefüggés: egy tablespace tartalmazhat objektumokat több különböző sémából, illetve egy séma objektumai is tárolódhatnak különböző tablespacekben

- Legfontosabb sémában található objektumok:
  - klaszterek; kényszerek; adatbázis hivatkozások; adatbázis triggerek; dimenziók; külső eljáráskönyvtárak; indexek és indextípusok; Java osztályok; materializált nézetek és a hozzájuk tartozó logok; objektum táblák, objektum típusok és objektum nézetek; operátorok; szekvenciák; tárolt függvények, eljárások és csomagok; szinonimák; táblák és indexelt táblák; nézetek.

- A séma és az objektumok a logikai szinthez tartozó fogalmak.

Adat-, vezérlőállományok

- A táblaterületek adatállományokon (data file) helyezkednek el. Egy adatfájl csak egyetlen táblaterülethez tartozhat, ám egy táblaterület több adatfájlon is tárolódhat.

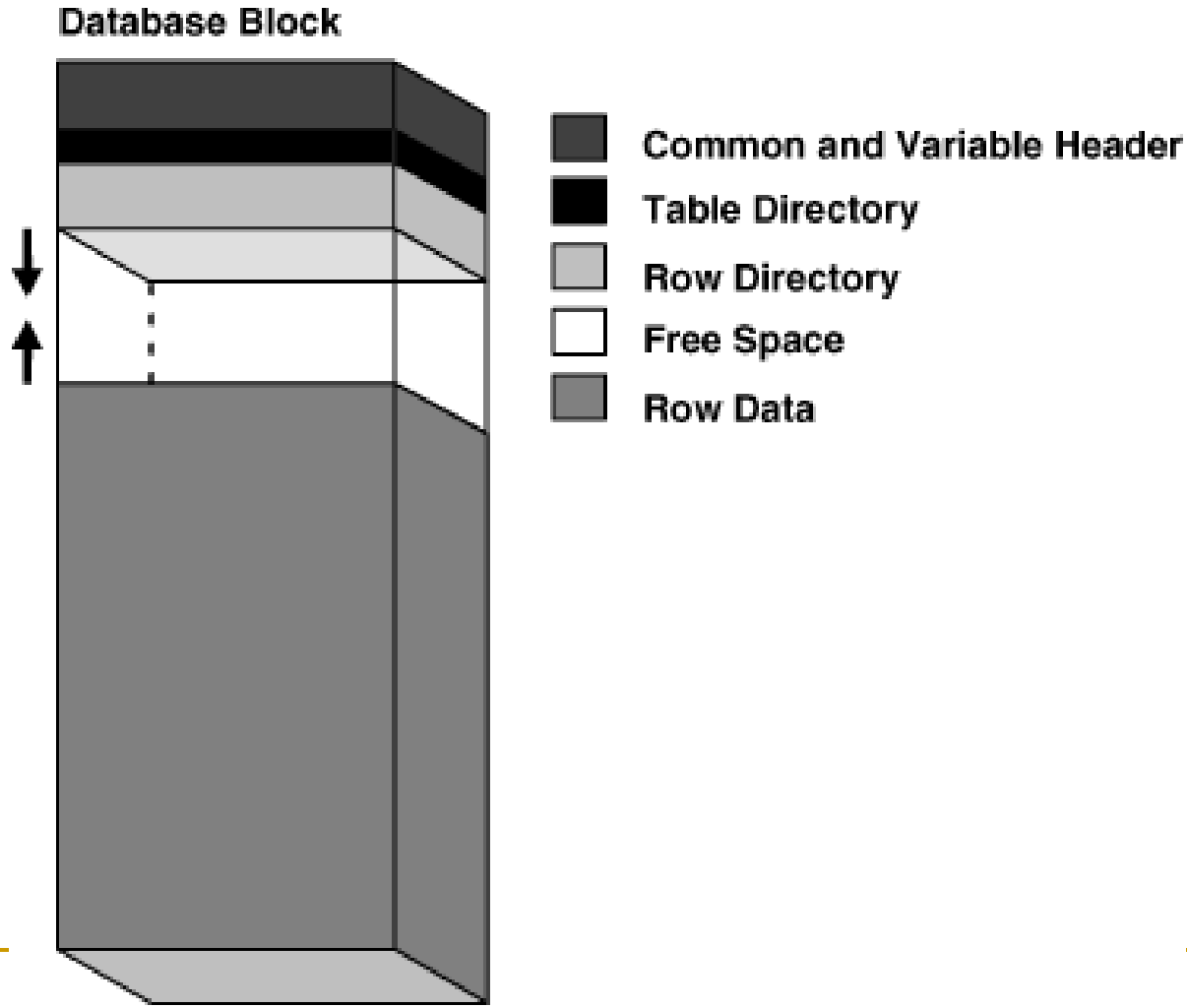
- A vezérlőállomány (control file) egy kis bináris fájl, amely az adatbázis elindításához és működtetéséhez szükséges. Egy vezérlőállomány csak egyetlen adatbázishoz tartozhat.

- Egy vezérlőállományban tárolódik többek között:
  - az adatbázis neve,
  - az adatbázishoz tartozó adatfájlok, redo log fájlok stb. neve, elérési útvonala,
  - archivált log információkkal kapcsolatos adatok,
  - ellenőrzési pontokkal kapcsolatos információk stb.

Adattárolás logikai megvalósítása

- A legkisebb adategység, amit az Oracle kezelni képes a blokk. Egy-egy blokk bizonyos mennyiségű bájtnak felel meg az adattárolón.
- Az Oracle által kezelt blokkok mérete általában különbözik a háttérben működő operációs rendszer blokkméretétől, viszont annak csak többszöröse lehet.
- Az extensek a háttértárolón folytonosan elhelyezkedő adatblokkokból állnak.
- Egy-egy szegmens egy-egy objektumnak felel meg. Egy szegmens egy fizikailag tárolt objektum. Particionált táblák és indexek esetén minden partíció külön szegmensen helyezkedik el.
- Az Oracle a létrehozáskor egyetlen extensen tárolja a szegmenst, ám ahogy az bővül, a rendszer újabb és újabb extenseket foglal le, amelyek az esetek nagyrészében nem folytonosan helyezkednek el.
- Egy szegmens nem tartozhat több táblaterülethez, az viszont előfordulhat, hogy több adatfájlon tárolódik. Ezzel szemben egy extens mindig egy adatfájlba kerül.

Blokkok I.


Database Block
Common and Variable Header
Table Directory
Row Directory
Free Space
Row Data

Blokkok II.

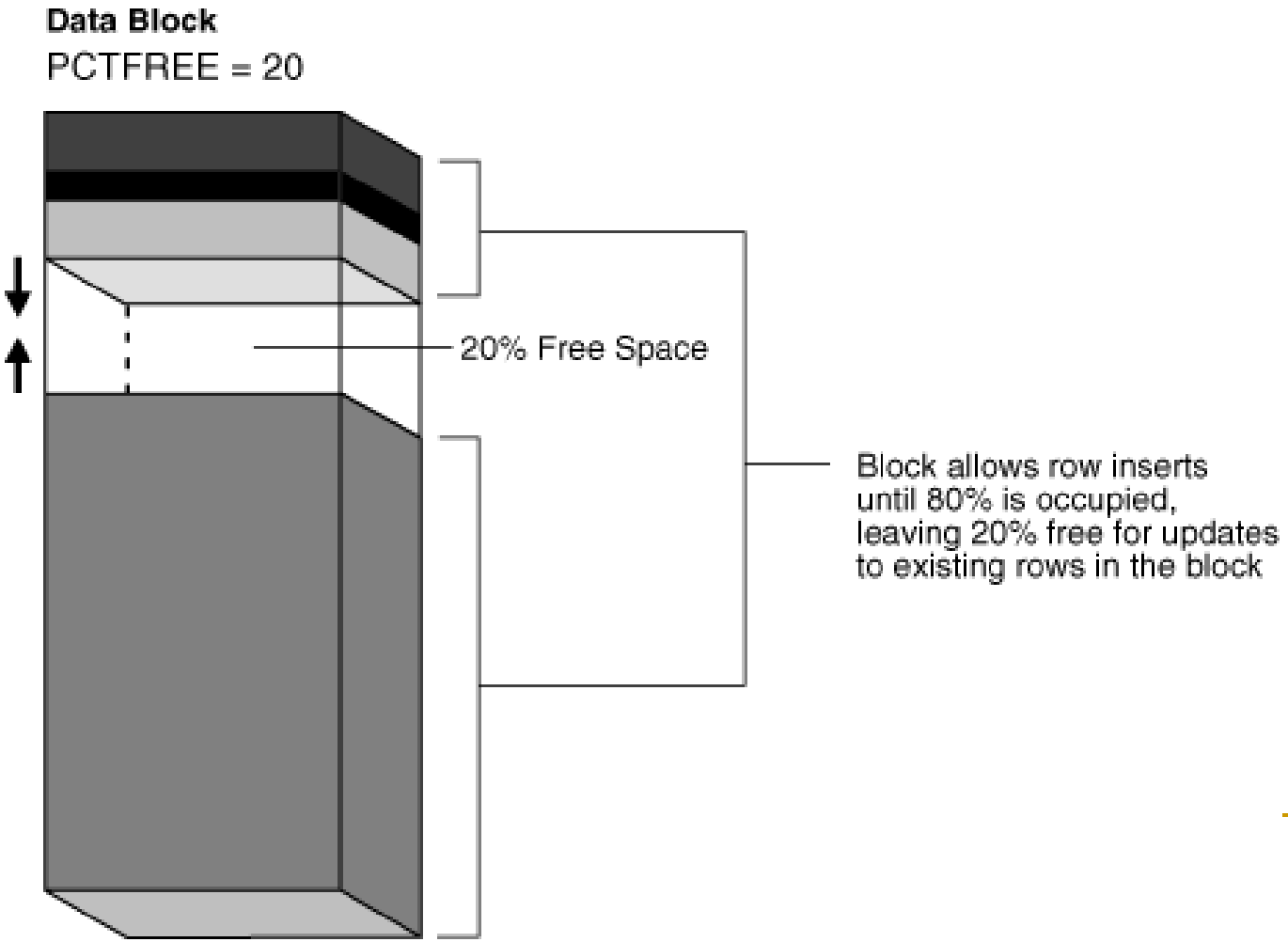
- A fejléc általános adatokat tárol, mint például a blokk címe, annak a szegmensnek a típusa (adat, index stb.), amihez tartozik stb.
- A tábla könyvtár arról a tábláról tartalmaz információkat, amelyeknek sorai a blokkban tárolódnak.
- A sor könyvtár a blokkban található sorokról tárol információkat, például a címüket. Ha törlődnek a blokk sorai, az Oracle nem szabadítja fel ezt a területet. Csak akkor hasznosítja újra, amikor új sorok kerülnek a blokkba.
- Az előző három részre angolul overhead-ként is hivatkoznak.
- A sor adat rész tartalmazza magukat a sorokat. A sorok „átlóghatnak” más adatblokkokba.
- A szabad területet módosításnál, illetve új sorok beszúrásánál használja a rendszer. A tranzakciók a sorok lock-olásánál szintén használják ezt a területet.

Sorok láncolása és vándorlása

- Ha egy sor nem fér be egy adatblokkba, például médiafájlok esetén, az Oracle a szegmenshez tartozó láncolt adatblokkokban tárolja az egyes részeket.
- Ha egy sor módosítás hatására túl nagyra nő, s már a szabad területen sem fér el, akkor a rendszer az egész sort egy új adatblokkba mozgatja. Ezt nevezik sorok vándorlásának (migrating).
- Az eredeti helyen csak a sor új helyének címe tárolódik. A vándorlás növelheti az I/O műveletek számát.

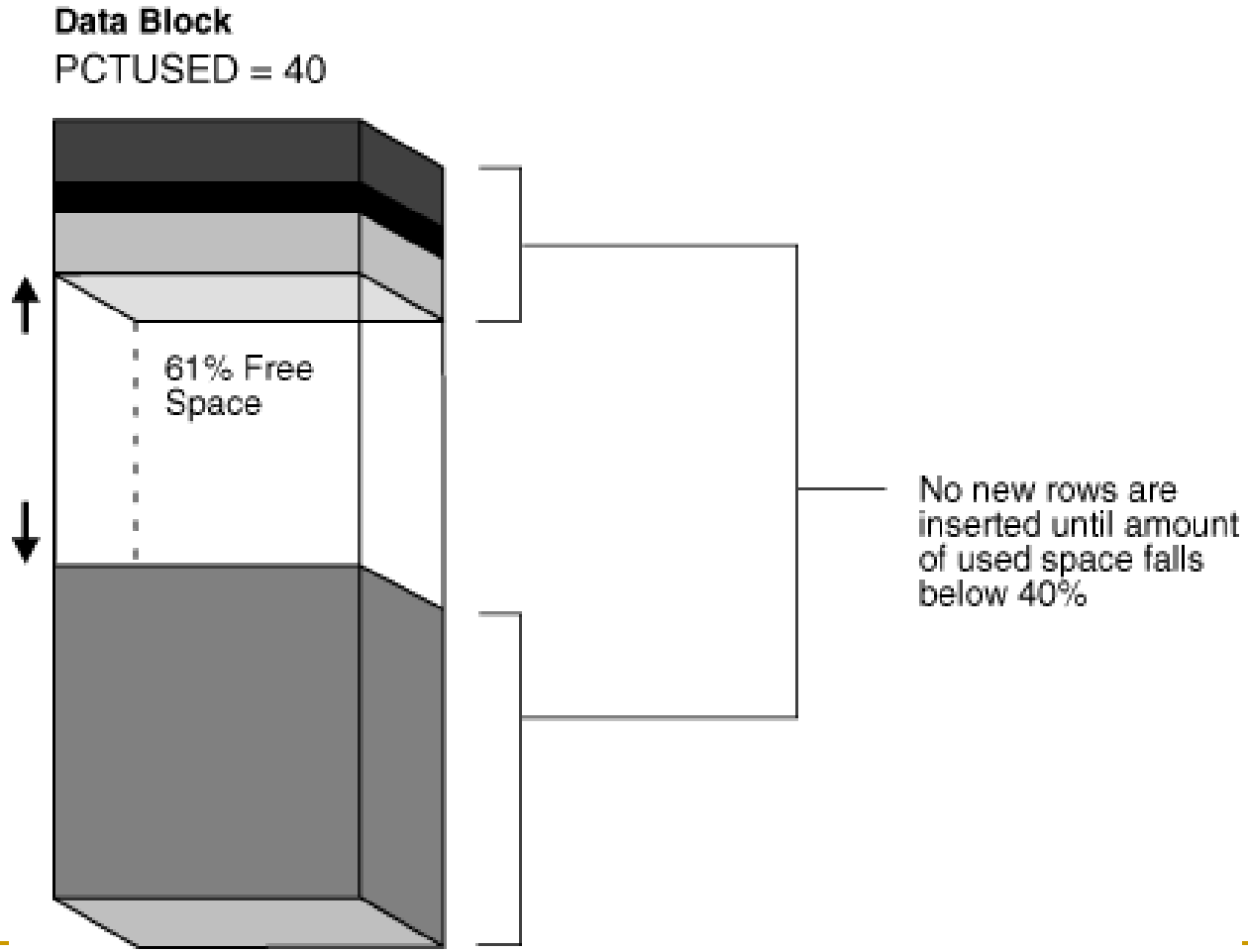
PCTFREE paraméter

Manuálisan felügyelt tablespaceknél két paramétert használhatunk az adatblokkok szabad területéhez történő hozzáférés vezérlésére PCTFREE : -vel beállíthatjuk, hogy az adatblokk hány százalékát akarjuk update-ekre fenntartani

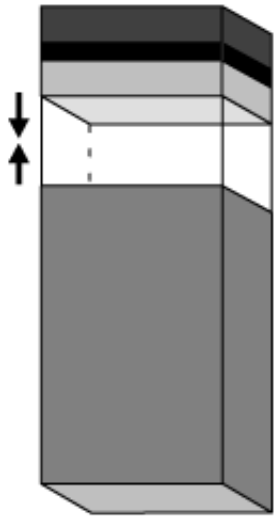


PCTUSED paraméter

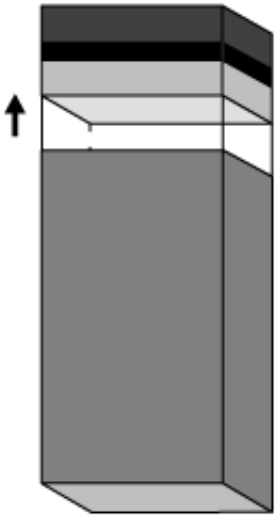
A *PCTUSED* paraméter egy minimum értéket ad a használt területre, amíg nem kezdeményezhetünk új instertet.



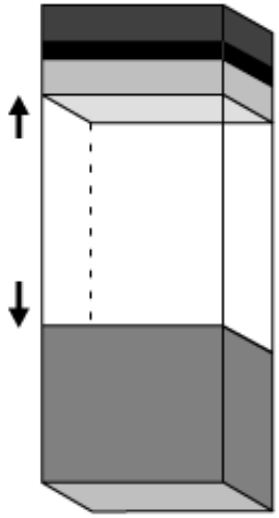
Data Block
PCTFREE = 20, PCTUSED = 40



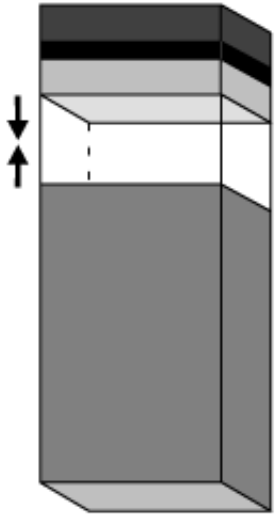
1 Rows are inserted up to 80% only, because PCTFREE specifies that 20% of the block must remain open for updates of existing rows.



2 Updates to exisiting rows use the free space reserved in the block. No new rows can be inserted into the block until the amount of used space is 39% or less.

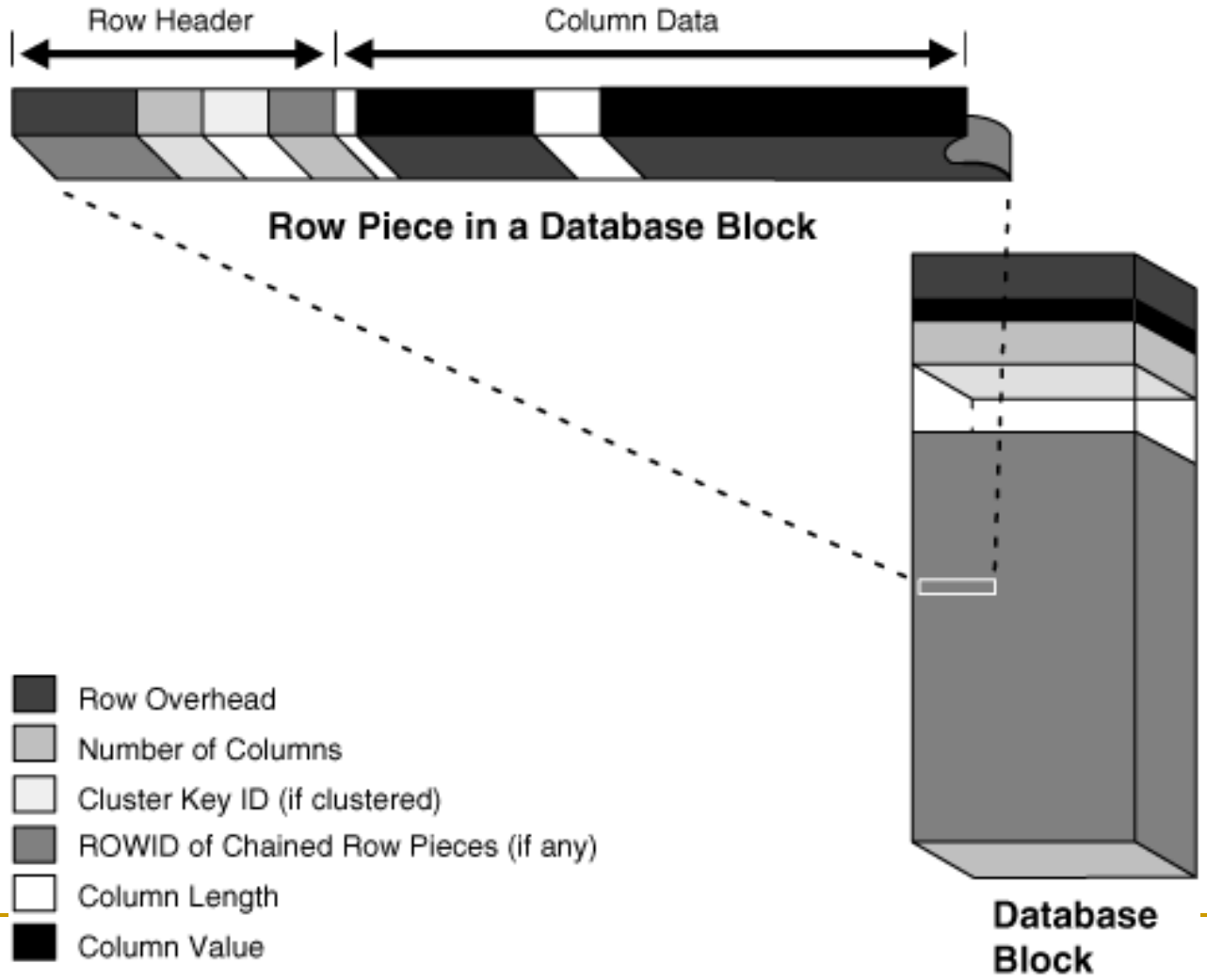


3 After the amount of used space falls below 40%, new rows can again be inserted into this block.



4 Rows are inserted up to 80% only, because PCTFREE specifies that 20% of the block must remain open for updates of existing rows. This cycle continues . . .

Sorok formátuma és mérete I.



Sorok formátuma és mérete II.

- Egy sor legfeljebb 255 mezőt tartalmazhat. Ha ennél több mező szerepel, a rendszer a maradékot igyekszik ugyanabban a blokkban láncolva elhelyezni (infra-block chaining).
- Egy-egy sor fejléce a sorok oszlopairól, az egymáshoz láncolt sordarabokról (row pieces) – ha vannak –, klaszter esetén a klaszter kulcsáról tárol információkat.
- Ha a teljes sort tartalmazza a blokk, a fejléc legalább 3 byte hosszú.
- Ezek után minden egyes oszlopnál tárolódik az oszlop hossza (ha 255 byte-nál rövidebb 1 byte-on, ha hosszabb 3 byte-on) és a mező értéke.
- Ha változó hosszúságú adatról van szó, az adat által elfoglalt hely a változtatásoknak megfelelően módosul.
- NULL érték esetén az Oracle csak az oszlop hosszát tárolja (zero).
- A sor végén szereplő NULL értékek esetén még az oszlop hossza sem tárolódik. Ezt érdemes figyelembe venni a táblák attribútumainak felsorolásakor a CREATE utasításban.

ROWID I.

- A ROWID a sorokat az elhelyezkedése (index-szervezett táblák) vagy címe alapján azonosítja.
- Egy sor mindaddig megőrzi azonosítóját, míg nem törlődik, ezért hasznos lehet SELECT, UPDATE, DELETE utasításokban. A ROWID-n történő hivatkozás a sorok elérésének leggyorsabb módja.
- Az indexekben a kulcsérték(ek) mellett szintén a ROWID tárolódik.
- A kiterjesztett ROWID (extended ROWID) formátuma:

  OOOOOOFFFBBBBBBRRR, itt:
  - OOOOOO: az adatobjektum száma, ami azonosítja a megfelelő szegmenst. Azon objektumoknak, amelyek ugyanahhoz a szegmenshez tartoznak, ugyanaz az azonosítója.
  - FFF: a táblaterületen belüli relatív adatállomány száma.
  - BBBBBB: az adatblokk azonosítója az adatállományon belül.
  - RRR: a blokkon belüli sor.

ROWID II.

```
SELECT ROWID,
        SUBSTR(ROWID,1,6) "OBJECT",
        SUBSTR(ROWID,7,3) "FILE",
        SUBSTR(ROWID,10,6) "BLOCK",
        SUBSTR(ROWID,16,3) "ROW„
 FROM ugyfel;
```

- A SUBSTR(ROWID,1,6) visszaadja a ROWID attribútum értékét az első karaktertől 6 karakter hosszúságban. A lekérdezés segítségével a különböző részeit különíthetjük el a ROWID-nek.

Extensek I.

- A táblák (table) jelentik az Oracle adatbázisban az alapvető adattárolási egységet.
- Tábla (vagy egyéb szegmens) létrehozásakor a rendszer egy kezdeti extenst (initial extent) allokál a tablespacen belül. Vezérlésére PCTFREE és PCTUSED paramétereket használunk. (Partícionált, Egymásba ágyazott, Ideiglenes, Külső táblák)
- Ha ez betelik, az Oracle újabb extenst allokál. A tárolási paraméterekhez (storage parameters) tartozó PCT_INCREASE paraméter segítségével adható meg, hogy hány százalékkal nőjön a következő extens mérete.
- A MINEXTENTS, MAXEXTENTS paraméterek a minimálisan lefoglalandó, illetve maximálisan lefoglalható extensek számát tárolják.
- A táblaterületek helykezelése történhet az adatszótárak segítségével, illetve lokálisan bitmap-ek felhasználásával. Az Oracle az utóbbit ajánlja nagyobb hatékonysága miatt.
- Általában egy szegmens extensei, ha felszabadulnak sem kerülnek vissza a közös táblatérbe, csak a szegmens törlése után.
- Klaszterek esetén, ha törlünk egyet az objektumok közül, a szegmenshez tartozó extensek még akkor sem „szabadulnak fel”.

Extensek II.

- Temporális szegmensekben, ha az ideiglenes szegmens törlődik, a rendszer automatikusan visszaadja a szabad extenseket a táblaterületnek.
- A speciálisan rendezésre fenntartott szegmensek nem törlődnek egy-egy rendezés után.
- Adatszótárakon alapuló helykezelés esetén az Oracle a különböző extensekből felszabadult folytonos területeket nem vonja össze automatikusan, míg lokális helykezelés esetén igen.

Az adatszótár (**data dictionary**)

- Egy-egy adatbázishoz tartozó adatszótár többek között a következő infomációkat tárolja:
  - az összes adatobjektum definíciója, az általuk felhasznált memória mérete, blokkok száma,
  - a felhasználók neve, jogosultságai,
  - melyik felhasználó módosította az egyes objektumokat stb.

- Az adatszótárhoz tartozó adatok alaptáblákban tárolódnak, az Oracle ezekben követi a változásokat, ám a felhasználók ritkán férnek hozzá ezekhez, mert a rendszer egy belső reprezentációt használ.

- Az adatokat az Oracle által létrehozott nézeteken keresztül lehet megtekinteni. Ezek általunk is olvasható formában jelenítik meg az alaptáblák adatait.

- Az alaptáblák tartalma a felhasználók által nem módosítható.

USER_, ALL_, DBA_ prefixek

- A felhasználói **USER_** prefixű nézetekben a felhasználó csak a saját sémájához tartozó adatokat láthatja.
- A nézetek felépítése hasonló a többi nézetéhez, ám itt általában hiányzik az OWNER oszlop.
- Példa: USER_SEGMENTS.
- A kiterjesztett felhasználói **ALL_** prefixű nézeteken keresztül mindazon adatok megtekinthetők, amelyekhez az aktuális felhasználónak hozzáférése van.
- Az adatbázis adminisztrátori **DBA_** prefixű nézetek az egész adatbázisról adnak átfogó képet.

SYS és SYSTEM felhasználó

- Mindkét felhasználó automatikusan létrejön az adatbázis létrehozásakor, mindkettő adatbázis-adminisztrátori jogkörrel.
- Minden adatszótárhoz tartozó alaptábla és nézet a SYS felhasználó birtokában van a SYS nevű sémán. Döntő fontosságú annak biztosítása, hogy ezekhez más ne férhessen hozzá.
- A SYSTEM felhasználó további adminisztrációhoz szükséges táblákat és nézeteket definiálhat, melyeket az Oracle különböző szolgáltatásai használhatnak.

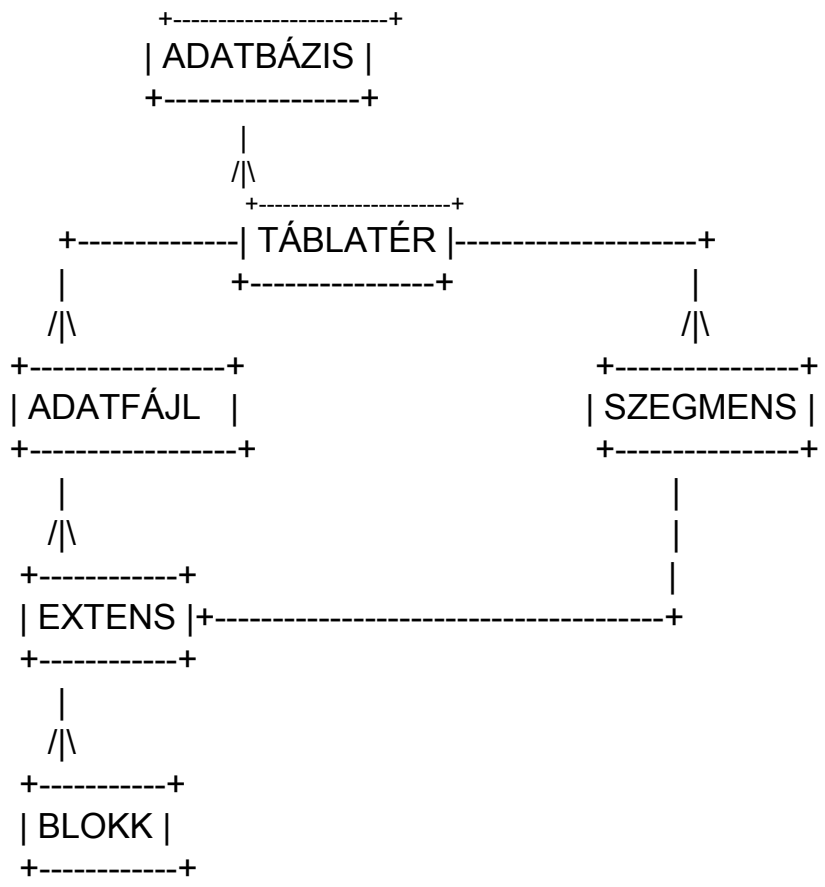
Memória architektúra

- Az Oracle adatbázis memóriájában a következő komponenseket tároljuk:
  - Programkód
  - információ a csatlakozott active és inactive sessionökről
  - programvégrehajtás közben szükséges információk, állapotok
  - az adatbázis processzei között megosztott információk (pl. zárak)
  - cachelt adatok, mint pl. adatblokkok és redo log bejegyzések, amik ugyanakkor természetesen tárolva vannak a merevlemezeken is

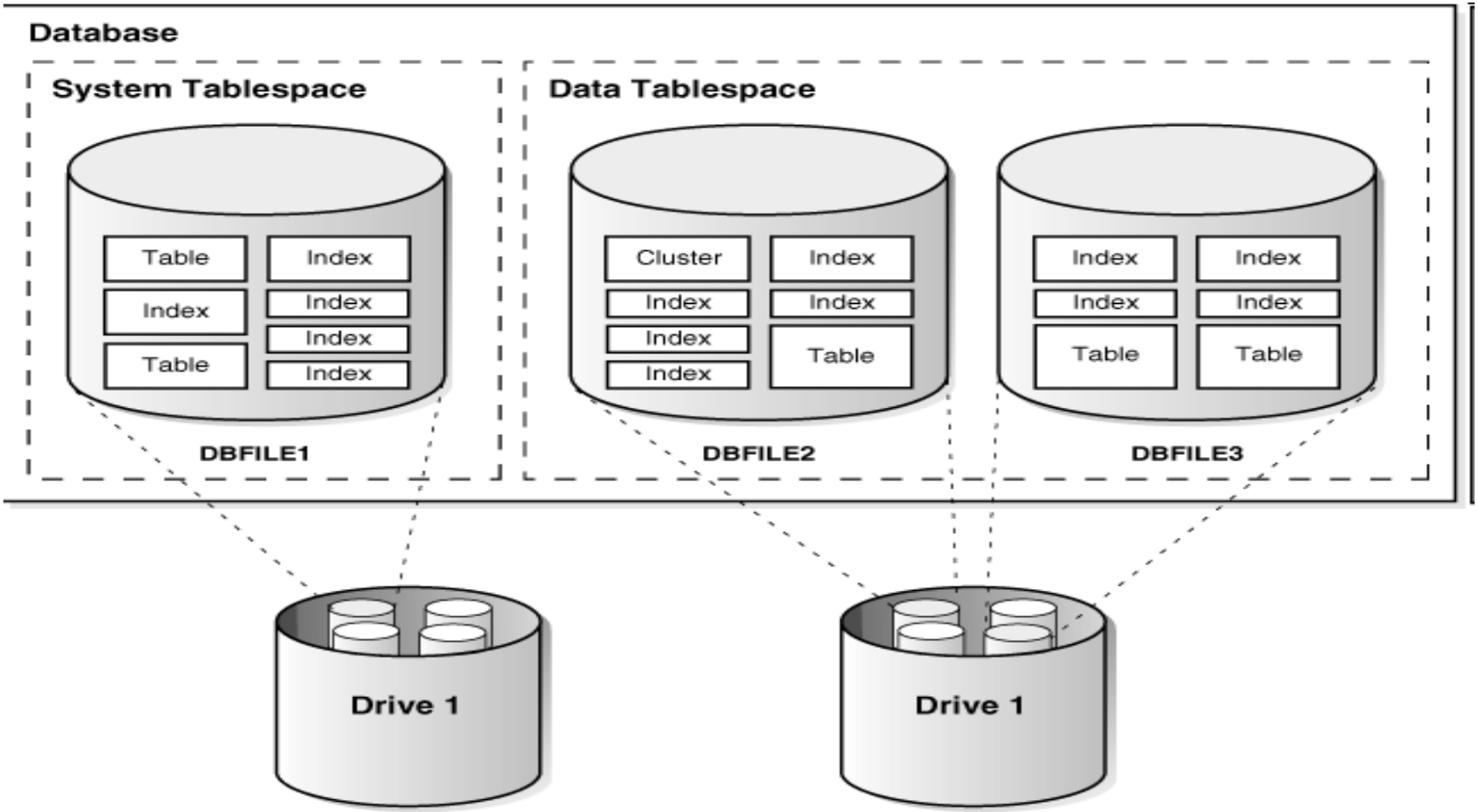
- A **memória struktúrája** három része osztható:
  - *szoftver kód területek:* az éppen futó vagy futtatható kódokat tartalmazza.
  - *System Global Area (SGA):* az összes szerver- és háttérfolyamat (processz) között megosztott memória struktúrák. Adat és vezérlési információkat tartalmaz.
  - *Program Global Area (PGA):* egy bizonyos szerver processzhez tartozó adat és vezérlési információkat tartalmazó memóriaterület, mely a szerverprocessz indításakor jön létre, s csak ő fér hozzá.

Processz architektúra

- Minden Oracle adatbázishoz csatlakozott felhasználónak két kód-modult kell futtatnia ahhoz, hogy hozzáférhessen egy adatbázis példányhoz.
  - Valamilyen adatbázis *alkalmazás* (pl. előfordító program) vagy *Oracle eszköz* (pl. SQL*Plus) amely SQL utasításokat továbbít egy Oracle adatbázishoz.
  - Adatbázis *szerver kód*, amely értelmezi és végrehajtja az SQL utasításokat.
- Ezeket a kód-modulokat futtatják a processzek. Ennek megfelelően a processzeket is két fő csoportra oszthatjuk: *felhasználói processz*ek, melyek az alkalmazás kódját futtatják, illetve *Oracle adatbázis processz*ek, ahová a szerver- és háttérfolyamatok tartoznak.


ADATBÁZIS
TÁBLATÉR
ADATFÁJL
SZEGMENS
EXTENS
BLOKK

Kapcsolat objektumok, tablespacek és datafile-ok között.



- CREATE TABLE tipus_proba(...)
- TABLESPACE users
- PCTUSED 50 PCTFREE 20 INITRANS 1 MAXTRANS 255
- STORAGE (INITIAL 32K MINEXTENTS 1 MAXEXTENTS 200 PCTINCREASE 0
- FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT);

- STORAGE(...): hogyan viselkedjenek az extensek, mekkorák legyenek, hogyan bővüljenek stb.
- - INITIAL: első extens mérete
- - NEXT: következő extens mérete
- - PCTINCREASE: milyen mértékben növekedjenek az extensek (%-ban) az előzőhöz képest. (50 azt jelenti, hogy másfélszerese lesz a következő)
- - MINEXTENTS: minimális extens darabszám (a tábla létrehozásakor ennyit automatikusan létrehoz)
- - MAXEXTENTS: maximális extens darabszám
- stb.

Feladatok I.

- Adattárolással kapcsolatos fogalmak (DBA_TABLES, DBA_DATA_FILES, DBA_TABLESPACES, DBA_SEGMENTS, DBA_EXTENTS, DBA_FREE_SPACE):

1. Adjuk meg az adatbázishoz tartozó adatfájlok nevét és méretét méret szerint csökkenő sorrendben.
2. Adjuk meg, hogy milyen táblaterek lettek létrehozva az adatbázisban, az egyes táblaterek hány adatfájlból állnak és mekkora az összméretük. (tablater_nev, fajlok_szama, osszmeret)
3. Mekkora a blokkok mérete a USERS táblatéren?
4. Melyik a legnagyobb méretű tábla szegmens az adatbázisban és hány extensből áll? (A particionált táblákat most ne vegyük figyelembe.)
5. Melyik a legnagyobb méretű index szegmens az adatbázisban és hány blokkból áll? (A particionált indexeket most ne vegyük figyelembe.)

Feladatok II.

6. Melyik a legnagyobb méretű LOB szegmens az adatbázisban és mekkora az első extensének mérete?
7. Melyik a legnagyobb méretű tábla partíció az adatbázisban és mekkora lesz a következő extensének a mérete?
8. Adjuk meg adatfájlonként, hogy az egyes adatfájlokban mennyi hely lett lefoglalva összesen.
9. Melyik felhasználó objektumai foglalnak összesen a legtöbb helyet az adatbázisban?
10. Van-e a NIKOVITS felhasználónak olyan táblája, amelyik több adatfájlban is foglal helyet?
11. Melyik táblatéren van az ORAUSER felhasználó dolgozo táblája?

Feladatok III.

- A táblák oszlopai (DBA_TAB_COLUMNS):

12. Adjuk meg azoknak a tábláknak a tulajdonosát és nevét, amelyeknek van 'Z' betűvel kezdődő oszlopa.
13. Adjuk meg azoknak a tábláknak a nevét, amelyeknek legalább 8 darab dátum típusú oszlopa van.
14. Adjuk meg azoknak a tábláknak a nevét, amelyeknek 1. és 4. oszlopa is VARCHAR2 típusú.

Feladatok IV.

- ROWID adattípus formátuma és jelentése:

- A ROWID megjelenítéskor 64-es alapú kódolásban jelenik meg. Az egyes számoknak (0-63) a következő karakterek felelnek meg: A-Z -> (0-25), a-z -> (26-51), 0-9 -> (52-61), '+' -> (62), '/' -> (63)
- Pl. 'AAAAAB' -> 000001

15. Írjunk PL/SQL függvényt, ami a fenti 64-es kódolásnak megfelelő számot adja vissza. A függvény paramétere egy karakterlánc, eredménye pedig a kódolt numerikus érték legyen. (Elég ha a függvény maximum 6 hosszú, helyesen kódolt karakterláncokra működik, hosszabb karakterláncra, vagy rosszul kódolt paraméterre adjon vissza -1-et.)
16. Ennek a fv-nek a segitségével adjuk meg egy táblabeli sor pontos fizikai elhelyezkedeset. (Melyik fájl, melyik blokk, melyik sora.) Például az ugyfel tábla azon sorára, ahol az ügyfél neve 'MELAK'.

Feladatok V.

17. A NIKOVITS felhasználó CIKK táblájának adatai hány blokkban helyezkednek el?
18. Az egyes blokkokban hány sor van?

Feladatok VI.

- Adatbázis objektumok (DBA_OBJECTS):

19. Kinek a tulajdonában van a DBA_TABLES nézet illetve a DUAL tábla?
20. Kinek a tulajdonában van a DBA_TABLES illetve a DUAL szinonima?
21. Milyen típusú objektumai vannak az ORAUSER nevű felhasználónak az adatbázisban?
22. Hány különböző típusú objektum van nyilvántartva az adatbázisban? Melyek ezek? (két külön lekérdezés)
23. Kik azok a felhasználók, akiknek több mint 10-féle objektumuk van?
24. Kik azok a felhasználók, akiknek van triggere és nézete is?
25. Kik azok a felhasználók, akiknek van nézete, de nincs triggere?
26. Kik azok a felhasználók, akiknek több mint 40 táblájuk, de maximum 37 indexük van?