Az Oracle rendszer komponensei

Célok

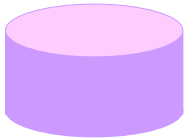
- Az Oracle szerver felépítésének és fő komponenseinek megismerése
- Annak bemutatása, hogy egy felhasználó Oracle példányhoz (instance) kapcsolódása hogy történik
- A következő feladatok végrehajtásának állapotai:
  - Lekérdezések
  - DML utasítások
  - Commit utasítások
- Az állományok, folyamatok és az ORACLE szerver által használt memóriarészek bemutatása

Az Oracle adatbázis architektúra

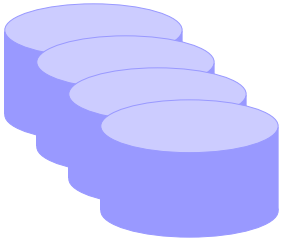
Az Oracle adatbázis két fő részből áll:

1. **Az adatbázis vagyis a fizikai struktúrák rendszere:**
   - A vezérlő fájl (control file), mely az adatbázis konfigurációját tárolja
   - A helyrehozó napló fájlok (redo log files), amikben a helyreállításhoz szükséges információkat tároljuk
   - Az adatfájlok, amelyekben az összes tárolt adat szerepel
   - Paraméterfájl, amelybe olyan paramétereket tárolunk, amelyek befolyásolják egy példány méretét és tulajdonságait
   - Jelszófájl, amelyben a rendszergazdák (SYSDBA) jelszavát tároljuk
2. **A példány (instance) vagyis a memória struktúrák és folyamatok rendszere:**
   - A memóriában lefoglalt System Global Area (SGA) terület és azok a szerverfolyamatok, amelyek az adatbázis-műveletek végrehajtásáért felelősek.

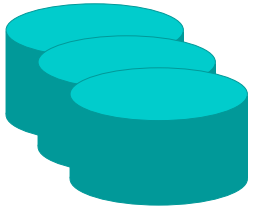
A fizikai adatbázis felépítése



Control files
(metaadatok)

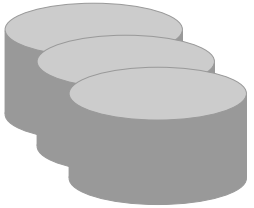


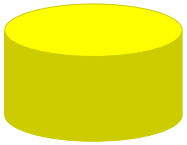
Data files



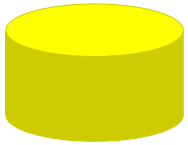
Online redo log files

(a helyreállításhoz szükséges naplófájlok)





Parameter file

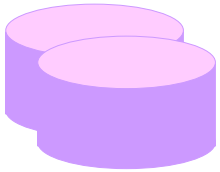


Password file

Archive log files

(régebben mentett naplók)

A vezérlő fájlok (Control files)

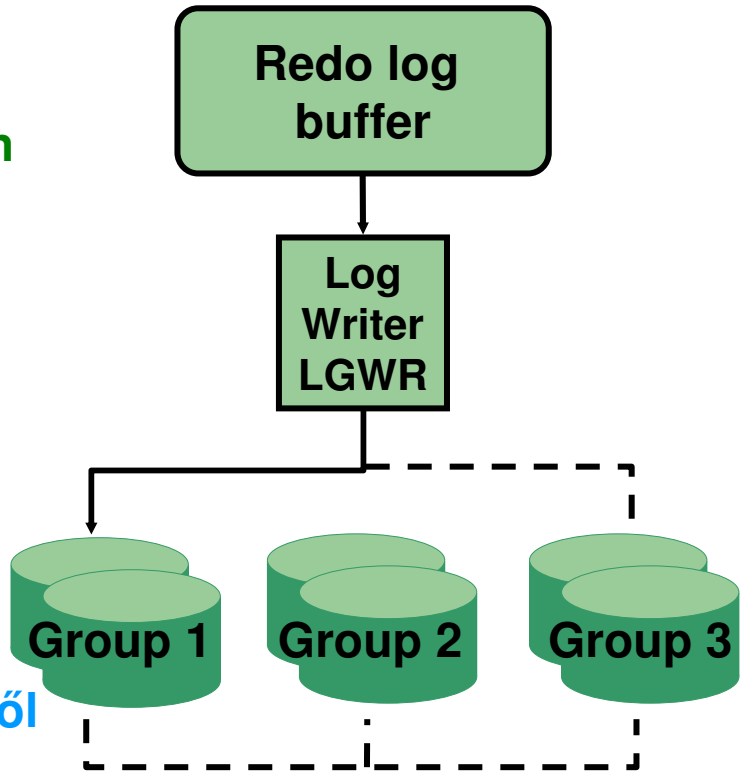


Control files

- A példány indításakor az adatbázis rákapcsolásához (mount) be kell olvasni a vezérlő fájlokat.
- Az adatbázist alkotó fizikai fájlokat határozza meg.
- Ha új fájlt adunk az adatbázishoz, akkor automatikusan módosulnak.
- A vezérlő fájlok helyét az inicializálási paraméterben adjuk meg.
- Adatvédelmi szempontból legalább három különböző fizikai eszközön legyen másolata (**multiplex the control files**). Ez is inicializálási paraméterrel adható meg. Az Oracle szerver elvégzi a többszörös másolatok szinkronizációját.

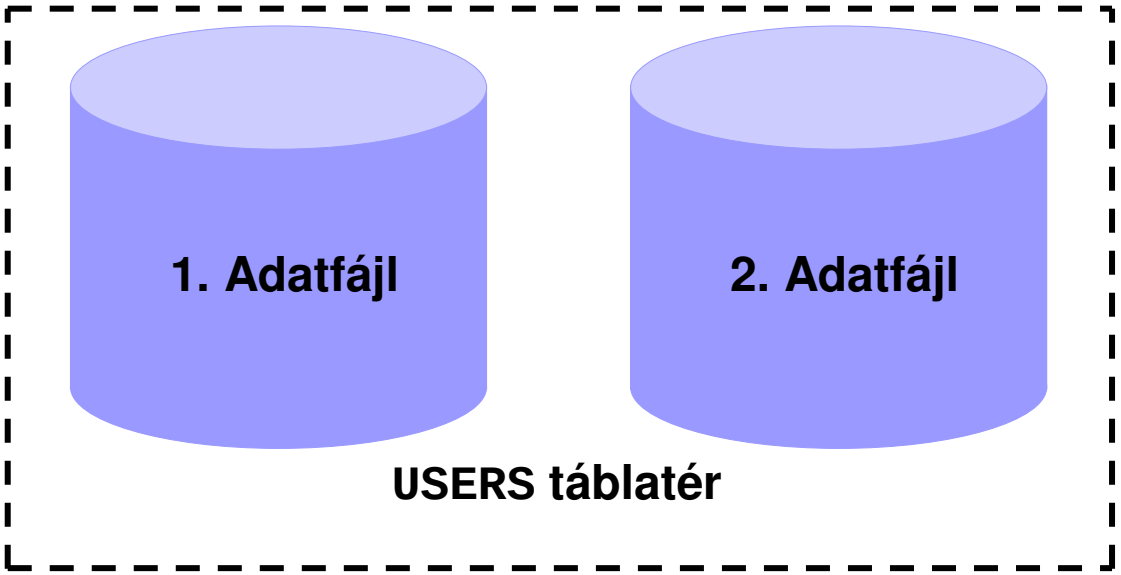
Napló állományok (Redo Log Files)

- Az adatbázis változtatásait rögzíti
- Konzisztens állapot visszaállításhoz szükséges rendszerhiba, áramszünet, lemezhiba, stb. esetén
- Adatvédelmi okokból különböző lemezeken többszörös másolatokat kell belőle szinkronizáltan kezelni.
- A REDO napló REDO fájlcsoportokból áll.
- Egy csoport egy naplófájlból és annak multiplexelt másolataiból áll.
- Minden csoportnak van egy azonosítószáma.
- A naplóíró folyamat (log writer process - LGWR) írja ki a REDO rekordokat a pufferből egy REDO csoportba, amíg vagy tele nem lesz a fájl, vagy nem érkezik egy direkt felszólítás, hogy a következő REDO csoportba folytassa a kiírást.
- A REDO csoportok feltöltése körkörösen történik.



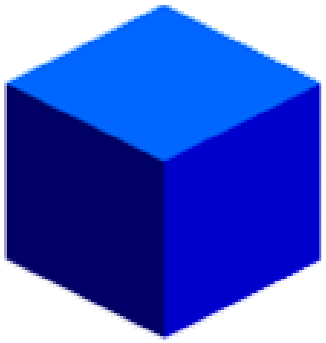
Táblaterek (tablespaces) és adatfájlok

- Minden adatbázis logikailag egy vagy több táblatérre van felosztva.
- A táblaterek egy vagy több fájlból állnak.
- Az adatfájlok mindig csak egy táblatérhez tartoznak.
- A táblatér mérete szerint kétféle lehet:
  - nagy fájlból álló táblatér (bigfile tablespace): ez egyetlen fájl, de 4G blokkot tartalmazhat
  - kis fájlokból álló táblatér (small file tablespace): több kisebb fájlból áll

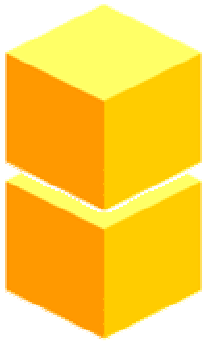


Szegmensek, területek (extents), és blokkok

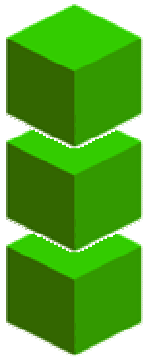
- Egy **táblatér** több **szegmensből** is állhat.
- Az adatbázis objektumokat, táblákat, indexeket a szegmensekben tároljuk.
- A szegmensek **területekből (extents)** állnak.
- A területek (extents) folytonos **adatblokkok** halmazai.
- Az adatblokkok az adatbázis legkisebb írható/olvasható egységei.
- Az adatblokkok operációs rendszerbeli blokkokra képezhetők le.



Szegmens



Területek
Extents



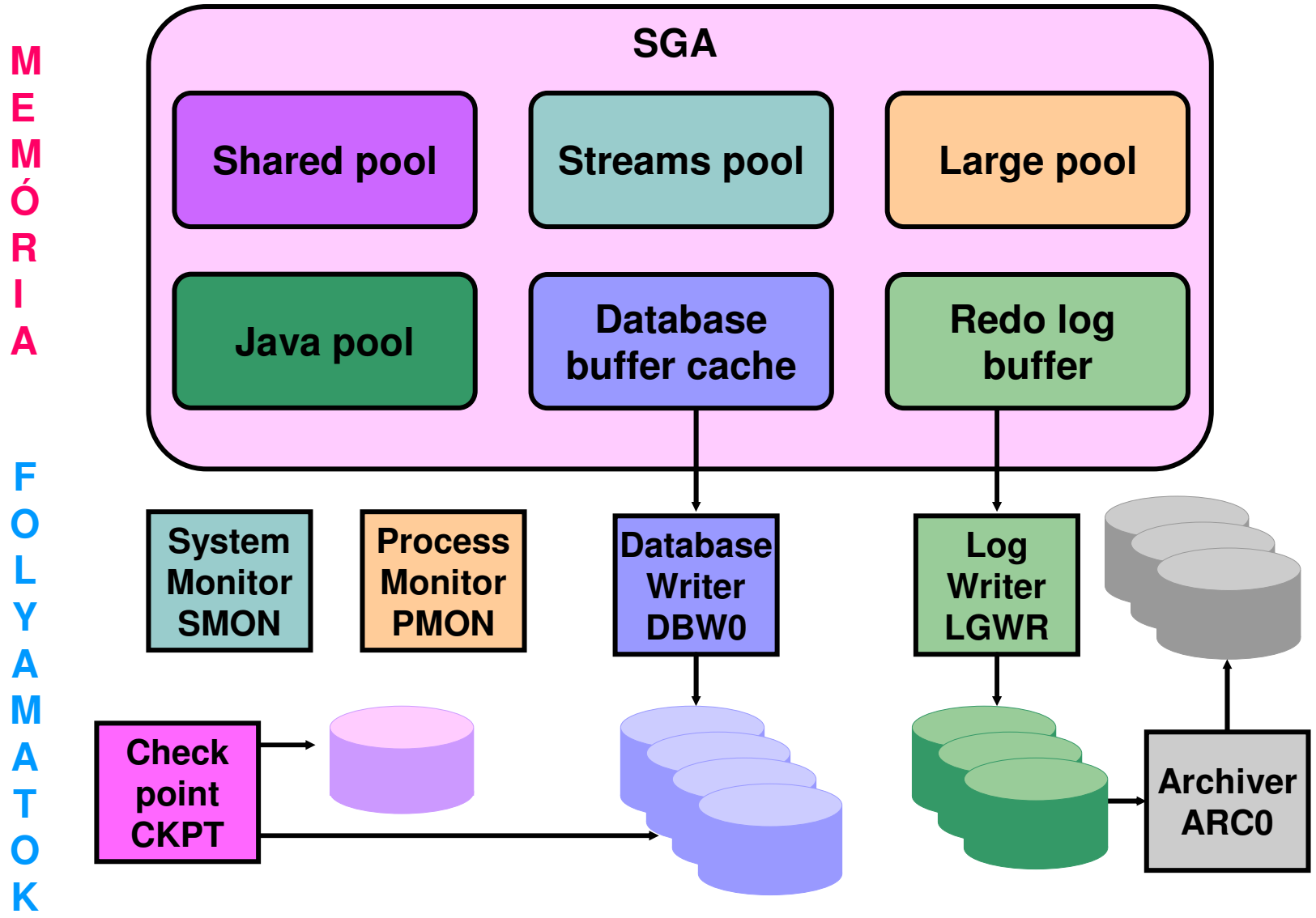
Adatblokkok

OS blokkok

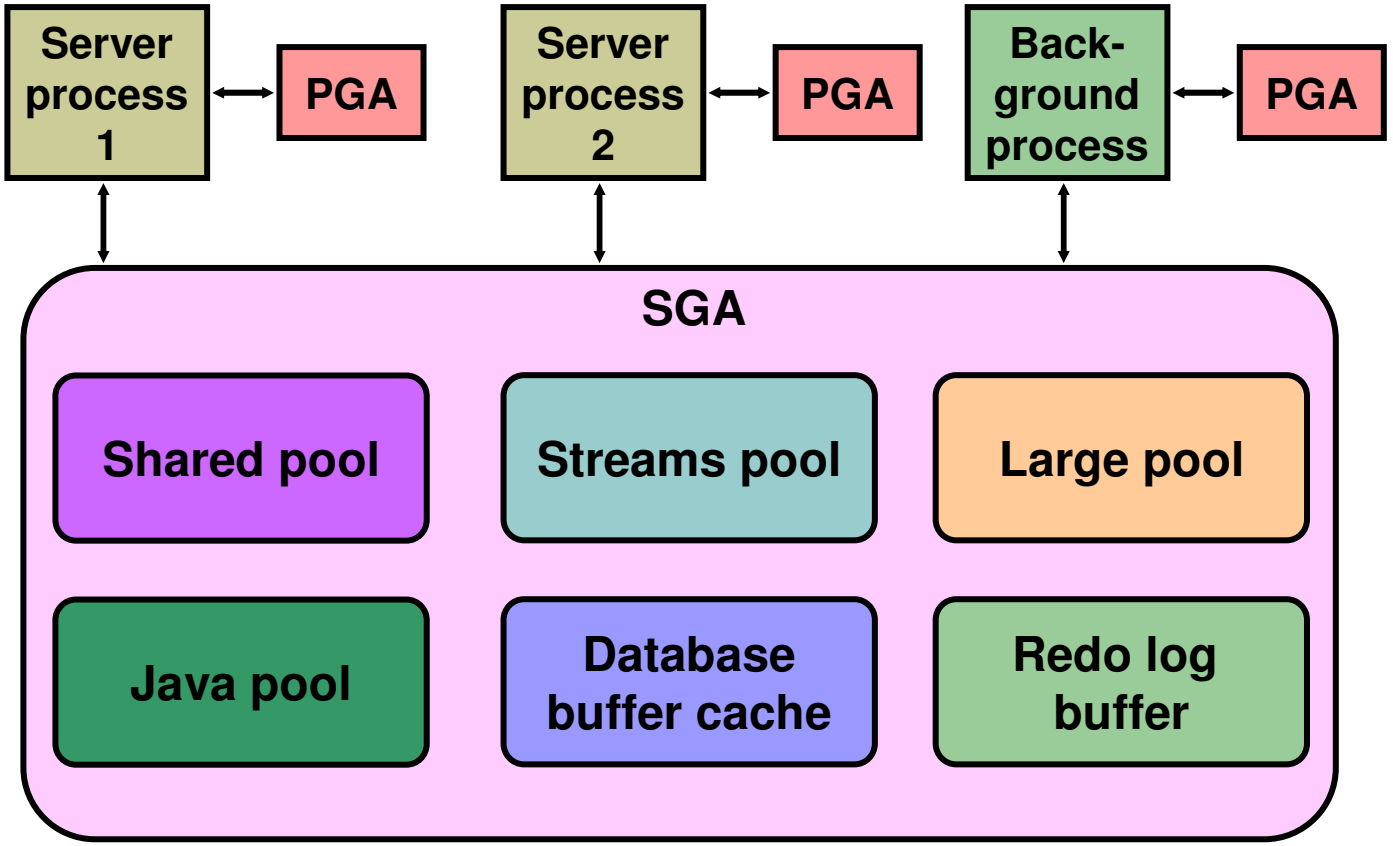
– Az adatblokk mérete alapértelmezésben 8K.

– Statikus adatbázis (adattárház) esetén nagyobb méretet érdemes használni, dinamikus adatbázis (tranzakciós adatbázis) esetén kisebbet.

Az Oracle példány (instance) felépítése



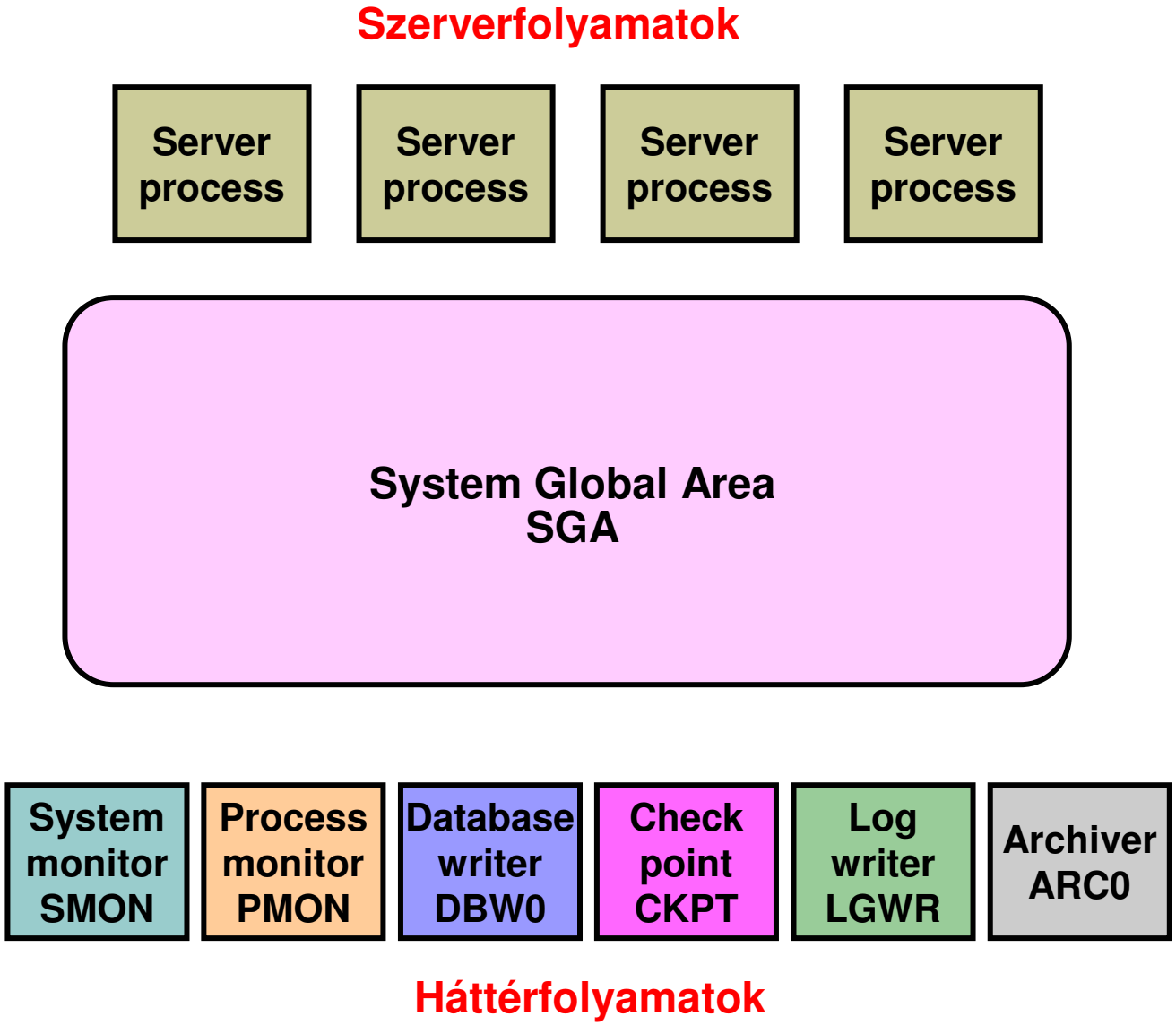
Az Oracle memóriakezelése



- **Egy Oracle példányhoz tartozó Oracle memóriaszerkezet a következő részekből áll:**
  - **System Global Area (SGA): Az összes szerverfolyamat és háttérfolyamat osztozik rajta**
  - **Program Global Area (PGA): Minden szerverfolyamatnak és háttérfolyamatnak saját memóriaterülete (PGA-ja) is van.**
- **Az SGA a példányhoz tartozó adatokat és vezérlő információkat is tartalmazhat.**
- **Az SGA a következő adatszerkezetből áll:**
  1. **Database buffer cache: A beolvasott adatblokkok pufferterülete**
  2. **Redo log buffer: A helyreállításhoz szükséges redo napló pufferterülete, innen íródik ki a napló a lemezen tárolt redo naplófájlokba**
  3. **Shared pool: A felhasználók által használható közös puffer**
  4. **Large pool: A nagyon nagy Input/Output igények kielégítéséhez használható puffer**
  5. **Java pool: A Java Virtuális Gép (JVM) Java kódjaihoz és adataihoz használható puffer**
  6. **Streams pool: Az Oracle Stream-ek pufferterülete**

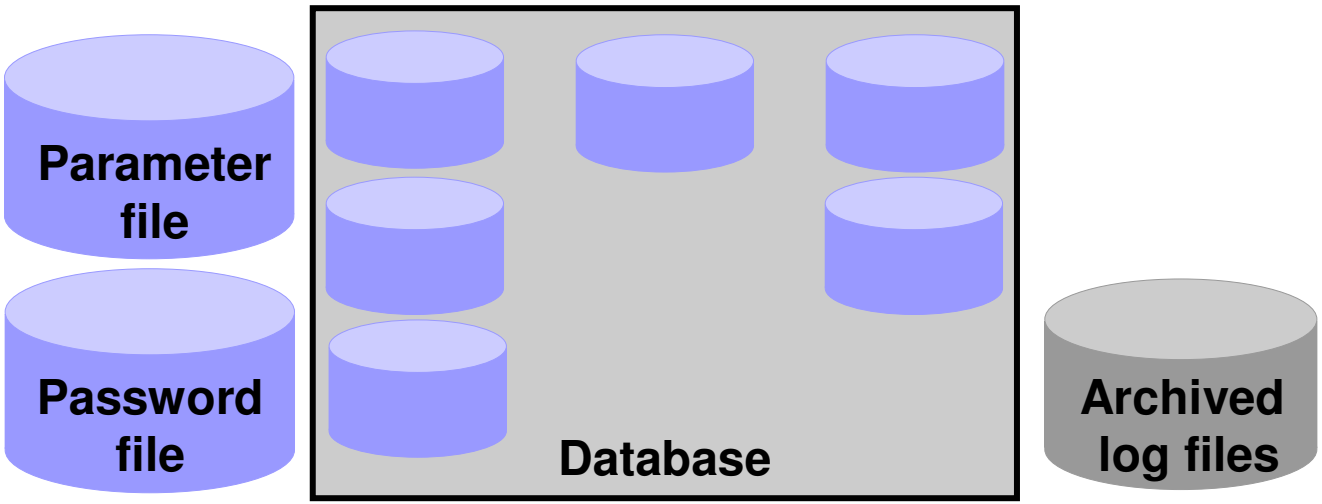
- **Az SGA dinamikus, azaz a pufferek mérete szükség esetén a példány leállítása nélkül változtatható.**
- **A Program Global Area (PGA) a szerverfolyamatok számára van fenntartva.**
- **A PGA mérete és tartalma attól függ, hogy a példány osztott módra (shared server mode) konfiguráltuk e**
- **A PGA általában a következőkből áll:**
  - **Private SQL area: Futási időben szükséges memóriaszerkezeteket, adatokat, hozzárendelési információkat tartalmaz. Minden olyan munkaszakasz (session), amely kiad egy SQL utasítást, lefoglal egy saját SQL területet.**
  - **Session memory: A munkaszakaszhoz (session) tartozó információk, változók tárolásához szükséges terület.**

Az Oracle folyamatok



- Amikor egy alkalmazás vagy Oracle eszköz (mint például az Enterprise Manager) elindul, akkor az Oracle szerver elindít egy szerverfolyamatot, amely lehetővé teszi az alkalmazás utasításainak végrehajtását.
- Az Oracle egy példány indításakor háttérfolyamatokat is elindít, amelyek kommunikálnak egymással és az operációs rendszerrel.
- A háttérfolyamatok kezelik a memóriát, puffereket, végrehajtják az írási, olvasási műveleteket a lemezen, karbantartásokat végeznek.
- A legfontosabb háttérfolyamatok a következők:
  - System monitor (SMON): Katasztrófa utáni indításkor elvégzi a helyreállítást
  - Process monitor (PMON): Ha egy felhasználói folyamat megszakad, akkor elvégzi a szükséges takarítást, karbantartást
  - Database writer (DBW*n*): Az adatpufferből kiírja lemezre, egy fájlba a módosított blokkokat
  - Checkpoint (CKPT): Ellenőrzési pontok esetén elindítja a DBW*n* folyamatokat és frissíti az adatbázis összes adatállományát és vezérlő állámányát
  - Log writer (LGWR): A REDO napló bejegyzéseit írja ki a lemezre
  - Archiver (ARC*n*): A REDO napló állomány másolatait a mentésre kijelölt lemezekre írja ki, mikor egy naplófájl betelik vagy a következő online redo naplóba folytatódik az írás (log switch)

Az adatbázishoz nem tartozó fájlok

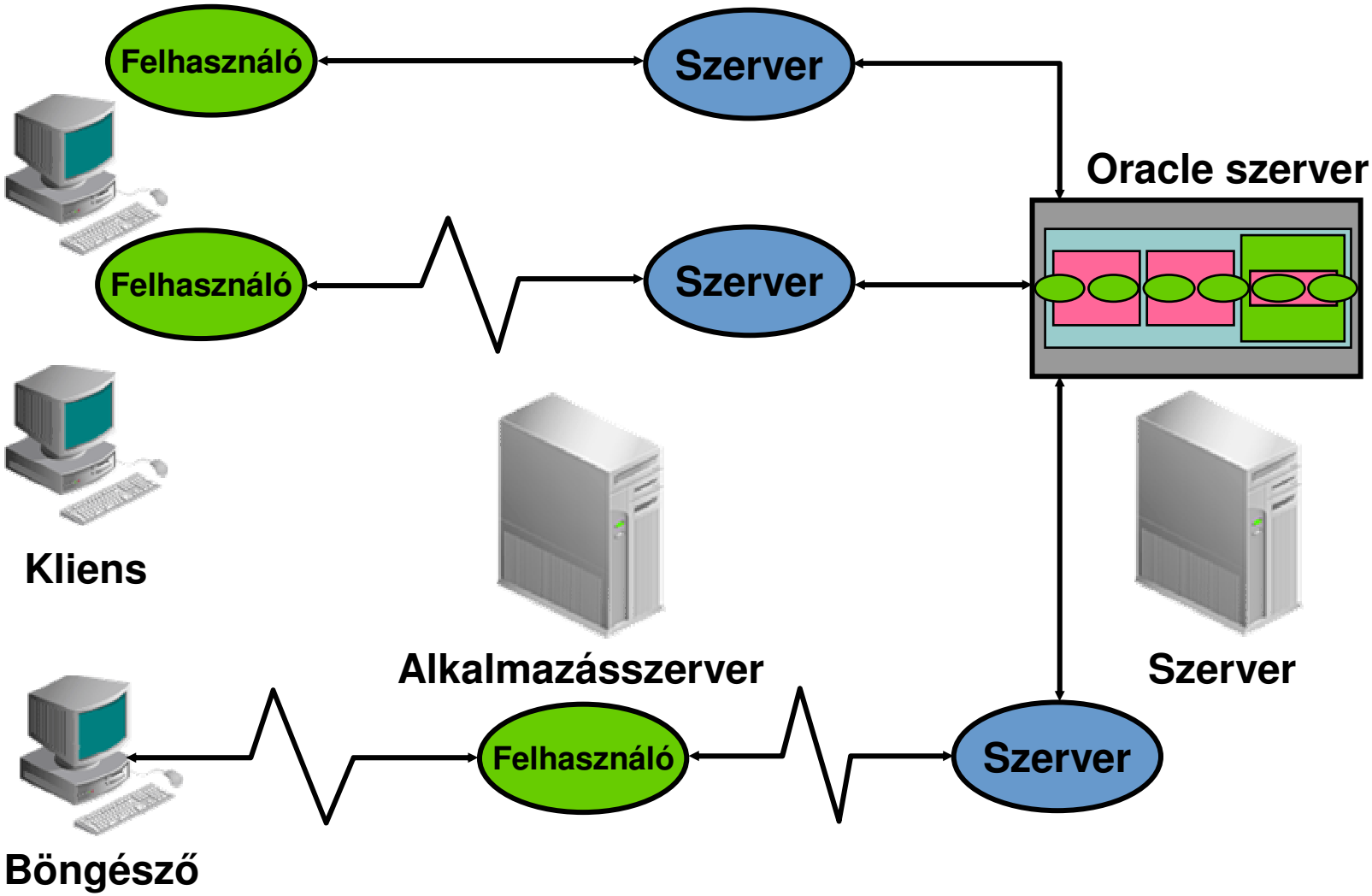


1. A **paraméterfájl** az Oracle példány jellemzőit határozza meg, például az SGA memóriarészeinek méretét.
2. A **jelszófájl**ból derül ki, hogy melyek azok a felhasználók, akik elindíthatnak vagy lekapcsolhatnak egy Oracle példányt.
3. Az **archivált REDO naplófájlok** a naplófájl másolatai, amelyeket lemezhiba után lehet helyreállításhoz használni.

Egy SQL utasítás végrehajtásának folyamata

- Egy példányhoz kapcsolódáshoz (Connect) szükséges:
  - Felhasználói folyamat
  - Szerverfolyamat
- Az SQL utasítás típusától függ, hogy az Oracle szerver milyen komponenseire lesz szükség:
  - A lekérdezések olyan folyamatokat indítanak, amelyek ahhoz kellenek, hogy megkapjuk a kívánt sorokat
  - Az adatmódosító (DML) utasítások naplózó folyamatokat is indítanak, hogy elmentsék a változásokat
  - A véglegesítés (Commit) biztosítja a tranzakciók helyreállíthatóságát
- Az Oracle szerver nem minden komponense vesz részt egy SQL utasítás végrehajtásában.

Kapcsolódás egy példányhoz



- **Mielőtt egy felhasználó küld egy SQL utasítást az ORACLE szervernek, előtte kapcsolódnia kell egy példányhoz.**
- **A felhasználói alkalmazások, vagy például az *i*SQL*Plus felhasználó folyamatokat (*user process*) indítanak el.**
- **Amikor a felhasználó az Oracle szerverre kapcsolódik, akkor készül egy folyamat, amit szerverfolyamatnak hívunk. Ez a folyamat kommunikál az Oracle példánnyal a kliensen futó felhasználó folyamat nevében. A szerver folyamat hajtja végre a felhasználó SQL utasításait.**
- **A kapcsolat egy kommunkációs útvonal a felhasználó folyamat és az Oracle szerver között.**

- **Háromféleképp lehet egy Oracle szerverhez kapcsolódni:**
  1. **OPERÁCIÓS RENDSZEREN KERESZTÜL: A felhasználó belép abba az operációs rendszerbe, ahol az Oracle példány fut és elindít egy alkalmazást, amely eléri az adatbázist ezen a rendszeren. Ekkor a kommunikáció útvonalat az operációs rendszer belső kommunikációs folyamatai hozzák létre.**
  2. **KLIENS-SZERVER KAPCSOLATON KERESZTÜL: A felhasználó a helyi gépén elindít egy alkalmazást, amely a hálózaton keresztül kapcsolódik ahhoz a géphez, amelyen az Oracle példány fut. Ekkor a hálózati szoftvert kommunikál a felhasználó és az Oracle szerver között.**
  3. **HÁROMRÉTEGŰ (three-tiered) KAPCSOLATON KERESZTÜL: A felhasználó gépe a hálózaton keresztül kommunikál egy alkalmazással vagy egy hálózati szerverrel, amely szintén a hálózaton keresztül össze van kötve egy olyan géppel, amelyen az Oracle példány fut. Például a felhasználó egy böngésző segítségével elér egy NT szerveren futó alkalmazást, amely egy távoli UNIX rendszeren futó Oracle adatbázisból gyűjti ki az adatokat.**

- **Kapcsolódáskor egy munkaszakasz (session) kezdődik. A munkaszakasz a felhasználó érvényesítése (validálásakor) esetén kezdődik és a kilépéséig vagy egy abnormális megszakításig tart.**
- **Egy felhasználó több munkaszakaszt is nyithat. Ehhez szükséges, hogy az Oracle szerver elérhető, használható legyen. (Néhány adminisztrációs eszközhöz még ez sem szükséges).**
- **Megjegyzés: Ha fentieknek megfelelően egy-egy értelmű a megfeleltetés a felhasználó és a szerverfolyamat között, akkor dedikált szerverkapcsolatról beszélünk.**

Lekérdezések végrehajtása

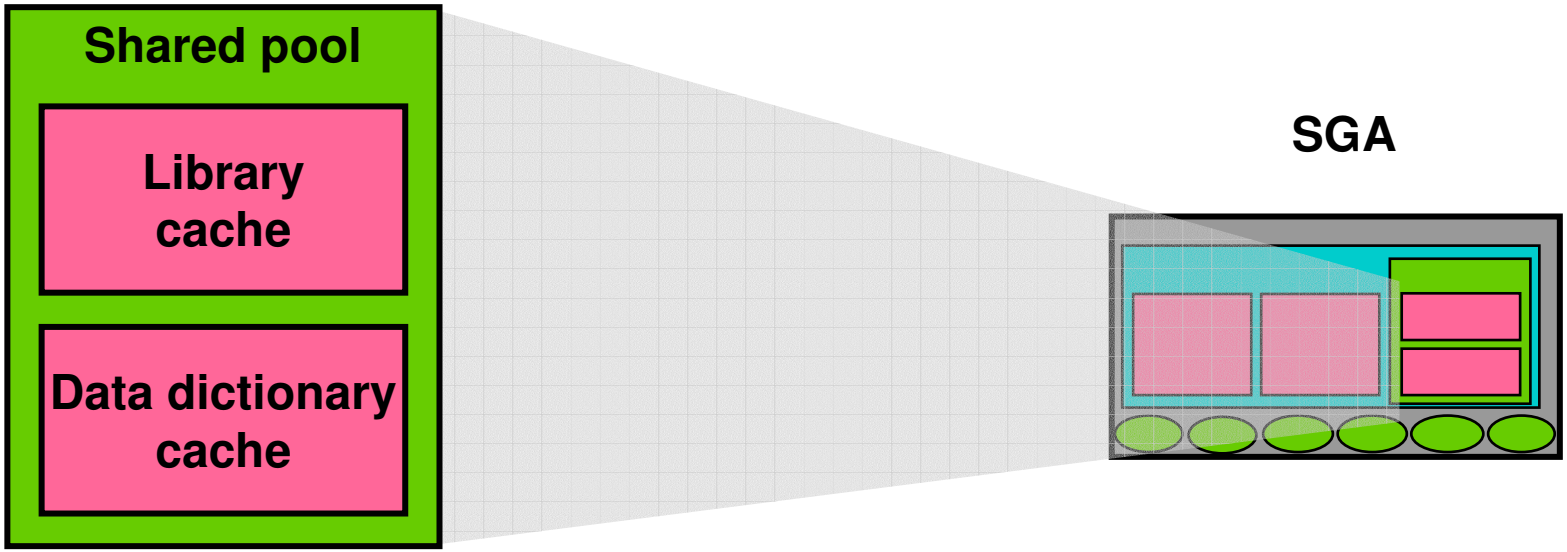
A lekérdezések ellentében más folyamatokkal egy sort, vagy akár több ezer sort is visszaadhatnak eredményképpen.

A lekérdezés végrehajtásának 3 lépése során a szerverfolyamat a következőket végzi el:

1. **Elemzés (Parse):**
   - **A közös SQL pufferben (shared pool) megnézi, hogy szerepel-e ez az utasítás**
   - **Szintaktikus ellenőrzés, léteznek-e az objektumok, rendelkezik-e a megfelelő jogokkal**
   - **Az elemzés alatt zárolja (lock) az objektumokat**
   - **Elkészíti és tárolja az optimális végrehajtási tervet**
2. **Végrehajtás (Execute): Előállítja a keresett sorokat**
3. **Visszaadás (Fetch): Visszaadja a sorokat a felhasználói folyamatnak**

A közös SQL puffer (shared pool)

- A könyvtár (library) cache az SQL utasítás szövegét, elemzett (parsed) kódját, és végrehajtási tervét tartalmazza.
- Az adatszótár (data dictionary) cache a táblák, oszlopok és egyéb objektumok definícióját, jogosultságait tartalmazza.
- A méretét a SHARED_POOL_SIZE inicializáló paraméter állítja be.



A shared pool komponensei

- Az elemzés (parse) alatt a szerver folyamat az SGA-nak ezen a részén fordítja le (compile) az SQL utasítást.
- Két fő komponense van:
  1. Könyvtár (Library) cache
  2. Adatszótár (Data dictionary) cache

1. A könyvtár (Library) Cache

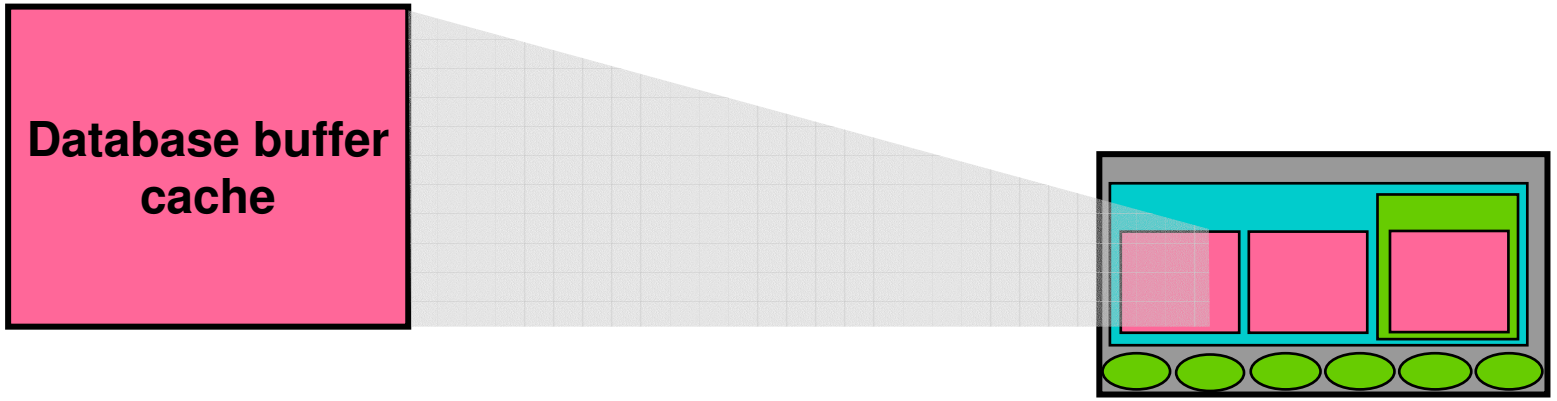
- Az utoljára kiadott SQL utasításokról tartalmaz információt a közös SQL puffer (shared SQL area) nevű memóriaszerkezetben:
  - Az SQL utasítás szövege
  - Az elemző fa (parse tree): Az utasítás lefordított verziója
  - A végrehajtási terve: Milyen lépésekkel kell végrehajtani az utasítást
- Az optimalizátor az Oracle szervernek az a része, amely meghatározza az optimális végrehajtási tervet.
- Ha egy SQL utasítást újra végrehajtunk és a közös SQL terület már tartalmazza a végrehajtási tervet, akkor nem kell újra lefordítania a szerverfolyamatnak az utasítást. Ezzel időt és memóriát lehet spórolni.
- Ha sokáig nem használják fel az SQL utasítást, akkor kikerül a Cashe-ből.

2. Adatszótár (Data Dictionary) Cache

- Az adatszótár utoljára használt definícióit tartalmazza. Így tartalmazhat információkat adatbázisfájlokról, táblákról, indexekről, oszlopokról, felhasználókról, jogosultságokról és más objektumokról is.
- Elemzés (parse) alatt a szerverfolyamat a nevek feloldásához, jogosultságok megállapításához először itt keresi az információt. Ha itt nem találja meg, akkor kezdeményezi a szükséges információk beolvasását az adatfájlokból.

Az adatbázis puffer (Database Buffer Cache)

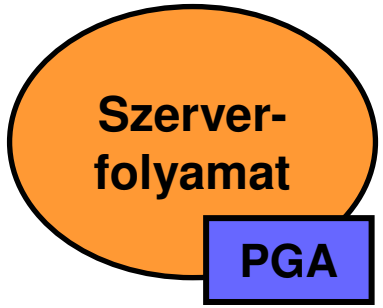
- Az utoljára beolvasott blokkokat tárolja.
- A mérete a `DB_BLOCK_SIZE` inicializáló paramétertől függ.
- A pufferek számát a `DB_BLOCK_BUFFERS` adja meg.



– Amikor egy lekérdezést kell végrehajtani, a szerverfolyamat először megnézi, hogy a keresett blokk nincs-e itt. Ha nem találja a pufferben, csak akkor olvassa be a blokk másolatát az adatfájlból.

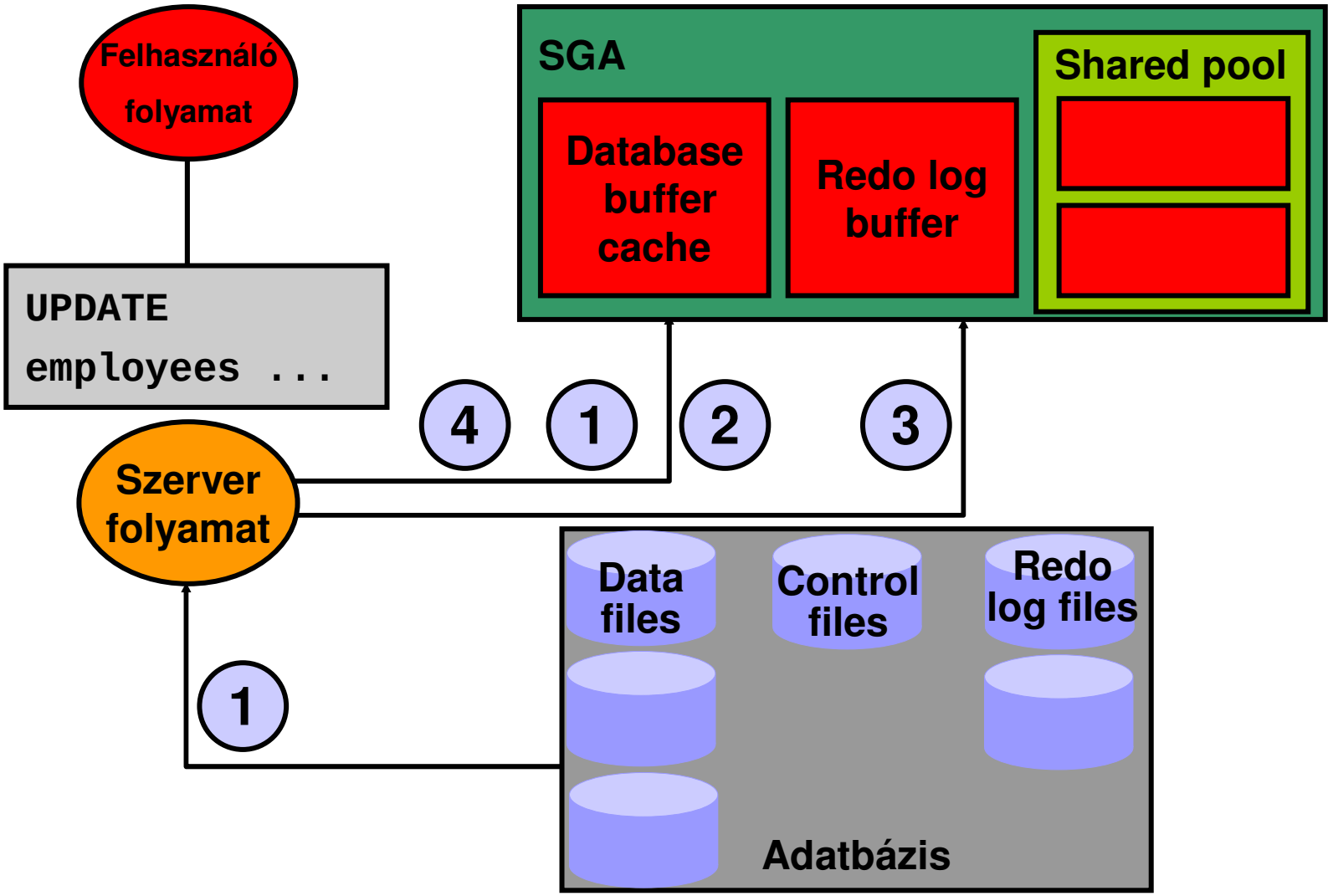
– Ha már nincs hely a pufferben, akkor a legrégebben használt adatblokk helyére olvassa be az újat.

Kizárólagos memóriaterületek: Program Global Area (PGA)



- Nincs megosztva
- Csak a szerverfolyamat írhatja
- A következőket tartalmazza:
  - Rendezési terület (sort area)
  - Információ a munkaszakaszról (session):
    - jogosultságok, statisztikák
  - A kurzorok állapota (cursor state)
  - A munkaszakaszhoz tartozó változók (Stack space)
- A szerver folyamat indulásakor foglalja le ezt a területet, befejezéskor pedig felszabadítja.

Adatmódosító (DML) utasítások végrehajtása



DML utasítások végrehajtása

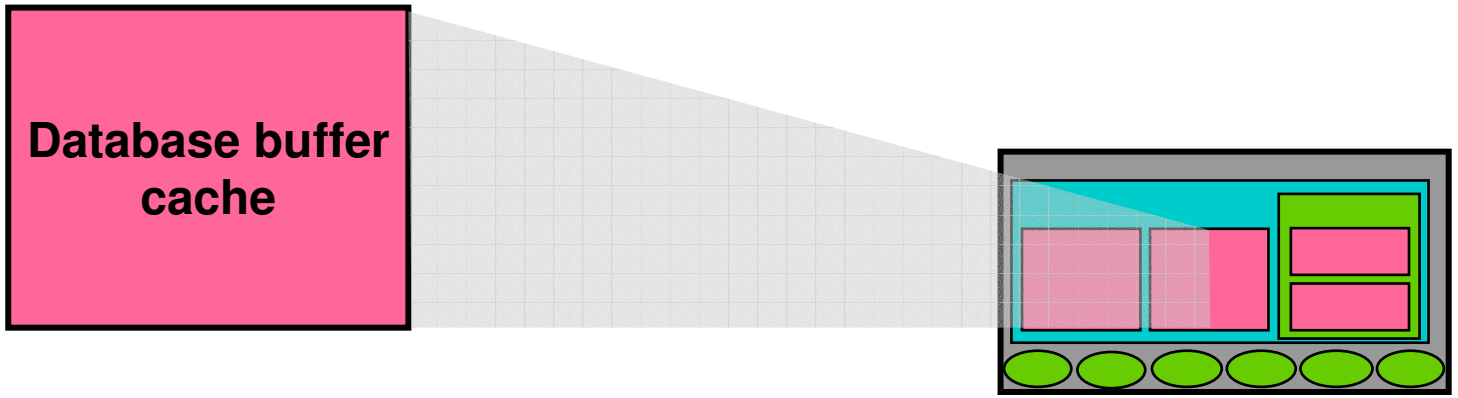
- **Az adatmódosítás végrehajtása kétfázisú:**
  - **A fordítás (Parse) ugyanolyan mint a lekérdezés esetében.**
  - **A végrehajtási most további folyamatokat igényel, mert az adatváltoztatást biztonságosan, visszaállíthatóan kell elvégezni.**
- **A DML végrehajtási fázisa:**
  1. **Ha az adatblokk és a rollback block nincs a pufferben, akkor a szerverfolyamat beolvassa az adatfájlból a pufferbe.**
  2. **A szerverfolyamat zárakat (locks) helyez azokra a sorokra, amelyeket módosítani készül.**
  3. **A REDO naplóba (redo log buffer) a szerverfolyamat beírja a változásokat, hogy majd vissza lehessen állítani szükség esetén az adatokat.**
  4. **A szerverfolyamat az adatok régi értékét bejegyzi a rollback blokkba.**
  5. **Az új értékeket rögzíti az adatblokkban.**

DML utasítások végrehajtása

- A szerverfolyamat a változás (UPDATE) előtti értékeket (before image) értékeket jegyzi fel és csak utána módosítja az adatblokkot. Ezek a változások a pufferben (database buffer cache) hajtódnak végre. A pufferben minden ki nem írt, változtatott blokk kap egy piszkos blokkra utaló bejegyzést, mivel ezek a blokkok különböznek a lemez megfelelő blokkjaitól.
- A törlés (DELETE) vagy beszúrás (INSERT) hasonló lépésekből áll. A "before image" DELETE esetén a törölt sor oszlopértékeit tartalmazza, INSERT esetén pedig a sor helyének címét.
- A késleltetett lemezreírás miatt katasztrófa esetén az SGA-val együtt elveszhetnek ezek a változások is.

A REDO napló puffer (Redo Log Buffer)

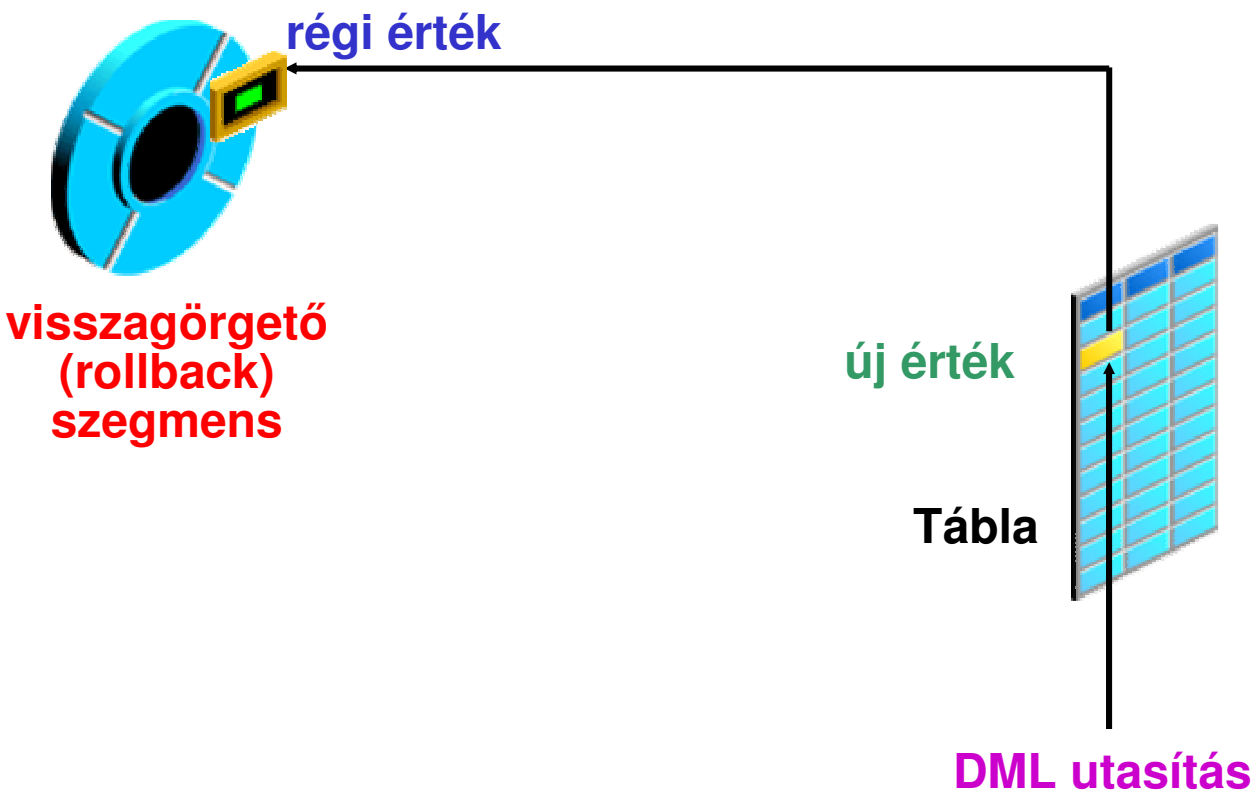
- A méretét a `LOG_BUFFER` paraméter definiálja.
- Az összes változtatást feljegyzi, ami a példányban történik.
- A feltöltése folytonos sorrendben történik.
- Ha betelik, elölről kezdi.



A REDO napló puffer (Redo Log Buffer)

- A REDO naplóbejegyzés feljegyzi, hogy
  - melyik blokk változik,
  - a változás helyét,
  - és az új értéket.
- A REDO bejegyzés nem tesz különbséget a blokkok típusa között, csak azt nézi, hogy mely bájtok változnak meg a blokkban.
- A REDO napló puffer folytonosan töltődik fel, különböző tranzakciók miatti bejegyzések átlapolódhatnak benne.
- Ha betelik, akkor az elejéről töröl helyet magának, de csak akkor, ha a régi REDO bejegyzés már a lemezre íródott.

A visszagörgető (Rollback) szegmens

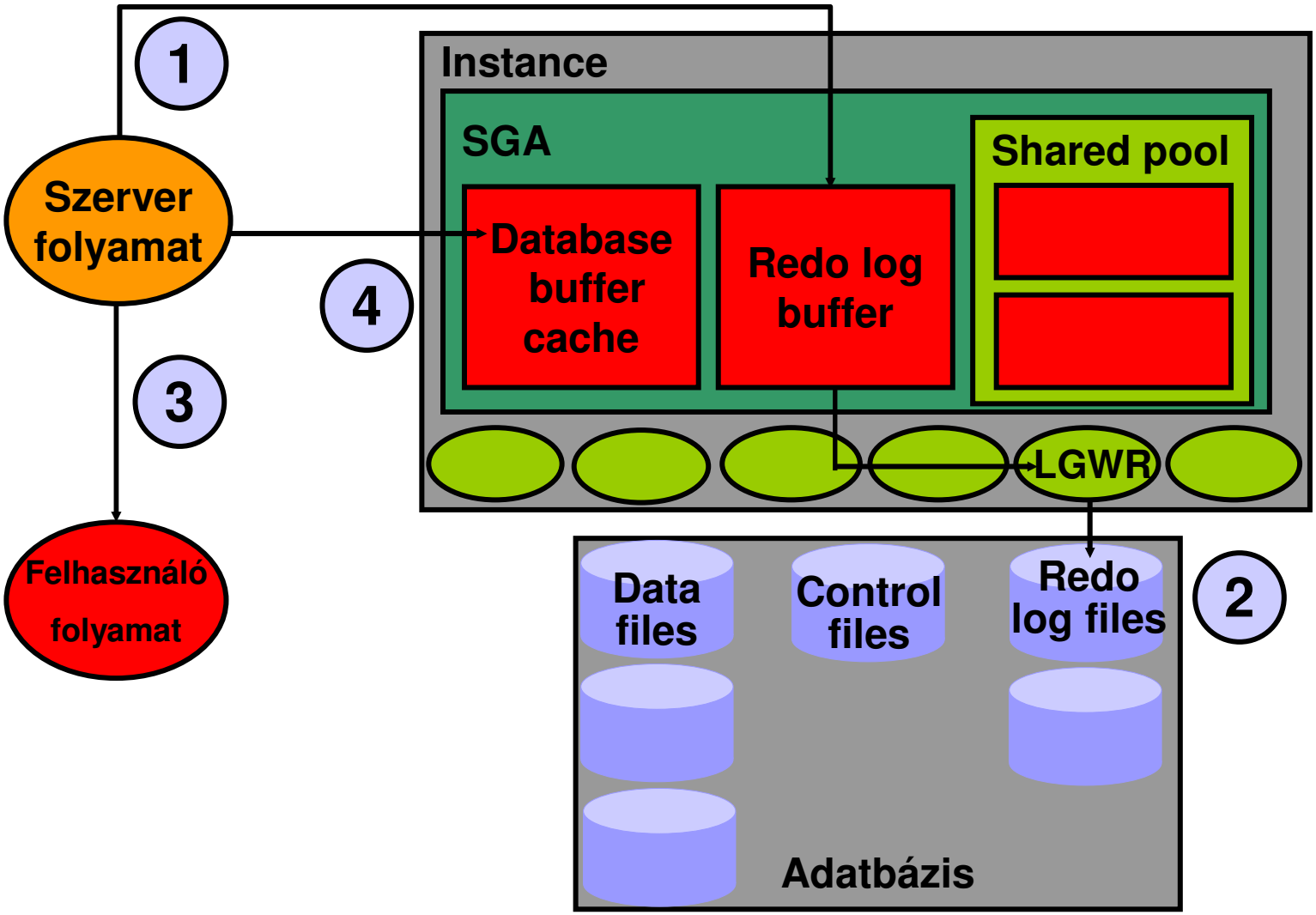


A visszagörgető (Rollback) szegmens

Minden változtatás előtt a szerverfolyamat elmenti az adatok régi értékét egy visszagörgető (rollback) szegmensbe.

- Ennek segítségével:
  - Meg lehet semmisíteni a változtatásokat (UNDO), ha egy tranzakciót visszagörgetünk.
  - Biztosítja az olvasási konzisztenciát (read consistency), ami azt jelenti, hogy más tranzakciók nem látják a DML utasítás hatását, ha az még nincs véglegesítve (commit).
  - Helyre lehet állítani az adatbázis konzisztens állapotát egy katasztrófa után.
- A visszagörgető (Rollback) szegmensek, ugyanúgy mint a táblák, indexek, adatfájlokban tárolódnak.
- A visszagörgető (rollback) blokkokat ugyanúgy be kell olvasni az adatbázis pufferbe (database buffer cache), mint a közönséges adatblokkokat, mikor szüksége van rájuk.
- A visszagörgető szegmenset a DBA készíti el.
- A szegmens változtatásai a REDO napló pufferben lesznek feljegyezve.

A véglegesítés `COMMIT` végrehajtása



Gyors véglegesítés (Fast commit)

- Az Oracle szerver gyors véglegesítést (fast commit) használ, ami biztosítja, hogy a példány sérülése esetén a véglegesített változtatásokat vissza lehessen állítani.
- Változási sorszám (System Change Number)
  - Amikor egy tranzakció véglegesítése következik, az Oracle szerver hozzárendel egy sorszámot (system change number - SCN) a tranzakcióhoz.
  - A SCN olyan mint egy belső időbélyegző, monoton növekedő, és egyedi az adatbázison belül.
  - Az SCN segítségével lehet megoldani az adatok szinkronzációját.
  - Az SCN segítségével lehet biztosítani az olvasási konzisztenciát.
  - Az SCN segítségével az Oracle szerver ellenőrizni tudja a konzisztenciát anélkül, hogy fel kellene használnia az operációs rendszer rendszeridejét.

Gyors véglegesítés (Fast commit)

- **A véglegesítési folyamat (COMMIT) lépései:**

– **A szerverfolyamat egy COMMIT bejegyzést és egy SCN számot ír a REDO napló pufferbe.**

– **A naplókiíró folyamat (LGWR) egyszerre kiírja a REDO napló puffer teljes tartalmát beleértve a commit bejegyzést is a REDO naplófájlokba. Ezután már biztos, hogy a változtatás nem vész el a példány sérülése esetén sem.**

– **A felhasználó értesítést kap, hogy a COMMIT végre lett hajtva.**

– **A szerverfolyamat feljegyzi, hogy a tranzakció rendben befejeződött, és elengedi a tranzakció által kiadott zárakat (locks).**

– **Minden piszkos (dirty) puffert kiír az adatfájlba. Ez a DBW0 folyamattól független, és mindegy, hogy a COMMIT előtt vagy utána történik.**

Gyors véglegesítés (Fast commit)

- **A gyors véglegesítés előnyei:**
  - **A naplófájl szekvenciális írása gyorsabb, mintha az adatfájl különböző blokkjaiba kellene írni.**
  - **A napló fájlba csak azt a minimális információt írjuk be, ami ahhoz kell, hogy a változásokat feljegyezhessük, míg ha az adatfájlokba írnánk, akkor teljes adatblokkokat kellene kiírni.**
  - **Ha több tranzakció akar egyszerre COMMIT-ot kiadni, akkor a példány az összes REDO naplórekordot egy írással végre tudja hajtani.**
  - **Ha a REDO napló puffer nincs nagyon tele, akkor egy szinkronizált írás elég tranzakciónként. Ha ráadásul nagyjából egyszerre akarnak befejeződni a tranzakciók, akkor átlagban egynél kevesebb szinkronizált írás elég tranzakciónként.**
  - **Mivel a REDO napló puffer tartalmát a COMMIT előtt is ki lehet írni, így nem szükséges várni a COMMIT-ra, ha hosszú lenne a tranzakció futása.**
- **Egy tranzakció visszagörgetésekor a naplóíró folyamat (LGWR) nem fog elindulni, így nem kerül ki a lemezre a napló.**
- **Az Oracle szerver helyreállításkor a COMMIT-tal be nem fejezett tranzakciók hatását megsemmisíti.**
- **Ha visszagörgetés után hiba történik, de a rollback bejegyzés még nem került rá a lemezre, akkor a COMMIT hiánya mutatja, hogy egy tranzakció nem fejeződött be, így vissza lett vonva.**

Összefoglalás

- Megismertük az adatbázis fájlait:
  - adatfájlok,
  - vezérlő fájlok,
  - online REDO naplók
- Megismertük az SGA memóriaszerkezetét:
  - DB buffer cache,
  - shared SQL pool,
  - redo log buffer
- Megismertük a legfontosabb háttérfolyamatokat:
  DBW0, LGWR, CKPT, PMON, SMON, ARC0
- Megismertük az SQL utasítások végrehajtásának lépéseit:
  - fordítás (parse),
  - végrehajtás (execute),
  - eredmény visszaadása (fetch)