

معماری بانک اطلاعاتی اوراکل

در تصویر مقابل معماری بانک اطلاعاتی اوراکل را مشاهده می کنیم که دو بخش اصلی آن Instance و Database هستند.

Instance از موارد و آیتم های مختلفی تشکیل شده است که در ابتدا در مورد آنها بحث خواهیم نمود. و با تجربه در آینده متوجه خواهید شد که این مفاهیم خیلی مهم هستند.

این معماری ای که مشاهده می کنید در واقع ساختار و اساس اوراکل هست و اوراکل بر مبنای آن کار می کند و با تغییر ورژن ها ، تغییر نمی کند . البته در نسخه های جدیدتر (نسخه 12) مواردی کمی به آن اضافه شده است.

آیتم هایی که در شکل در محدوده instance قرار دارند background Process نامیده می شوند.

در قسمت پایین تصویر هم فضای ذخیره سازی دیتابیس قرار دارد.

تفاوت بین دیتابیس (DATABASE) و INSTANCE

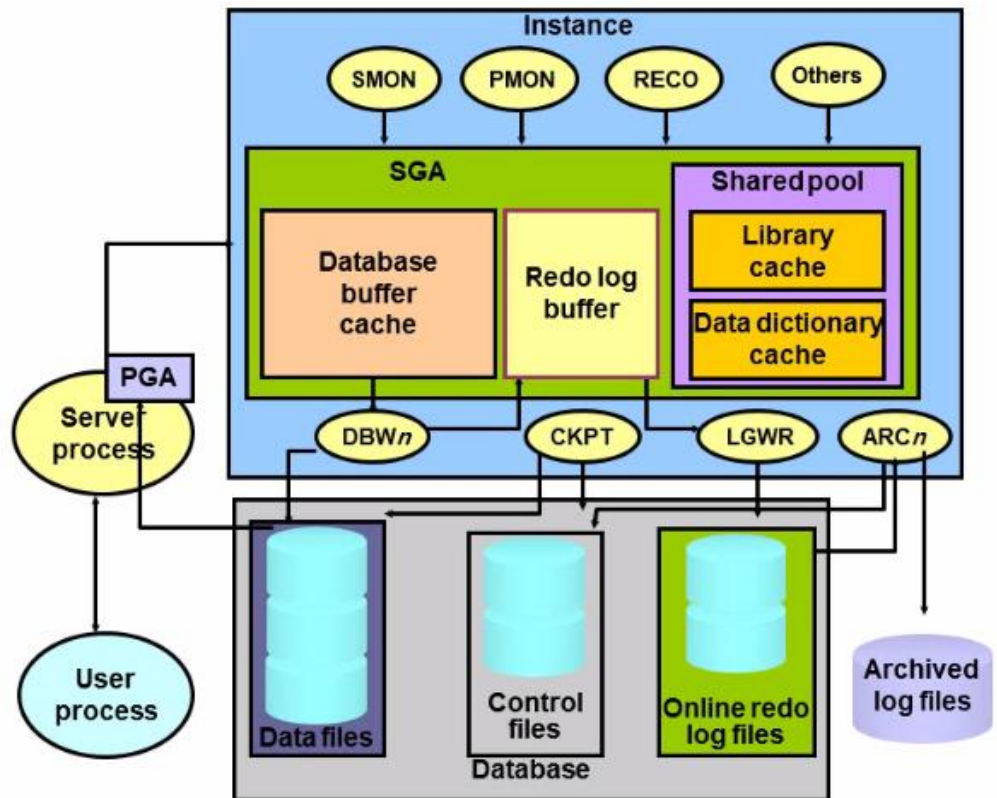
واژه های instance و Database ارتباط خیلی نزدیکی با هم دارند ، اما در واقع یک چیز واحد نیستند و نباید این دو را با هم اشتباه گرفت.

دیتابیس جایی است که مجموعه ای از فایل ها و داده ها (داده های کاربردی یا application data و متا دیتاها یا Meta Data) ذخیره شده است.

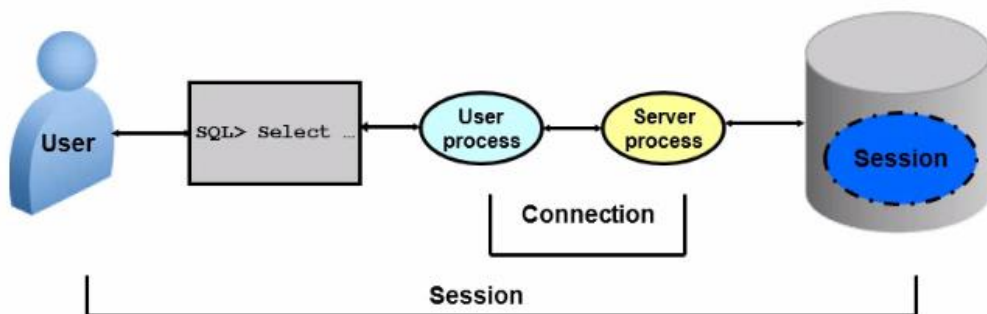
Instance یک نرم افزار (و حافظه ای است) که اوراکل برای استفاده و دستکاری داده های دیتابیس از آن استفاده میکند و برای دستکاری و استفاده داده های دیتابیس مستقیماً نمی توانیم از دیتابیس استفاده کنیم ، برای این منظور اوراکل اشیا مورد نظر ما را در یک فضا و حافظه ای قرار می دهد و در آنجاست که قادر خواهیم بود تا وارد داده های دیتابیس شویم و داده های دیتابیس را دستکاری کنیم.

یک دیتابیس به وسیله یک Instance باز (Open) می شود و قابل استفاده خواهد بود.

نکته ای که وجود دارد این است که هر Instance می تواند به یک دیتابیس متصل شود و آنرا باز کند و داده های آنرا دستکاری کند.



ارتباط با دیتابیس



ارتباط با دیتابیس

اتصال : (connection) ارتباط یک user Process با یک instance را اتصال می نامند.

User process همان کاربری خواهد بود که قصد دارد به دیتابیس متصل شود.

هر user process وقتی که می خواهد به دیتابیس متصل شود ، یک Server Process به آن اختصاص داده می شود. و هر Server process یک PGA خواهد داشت.

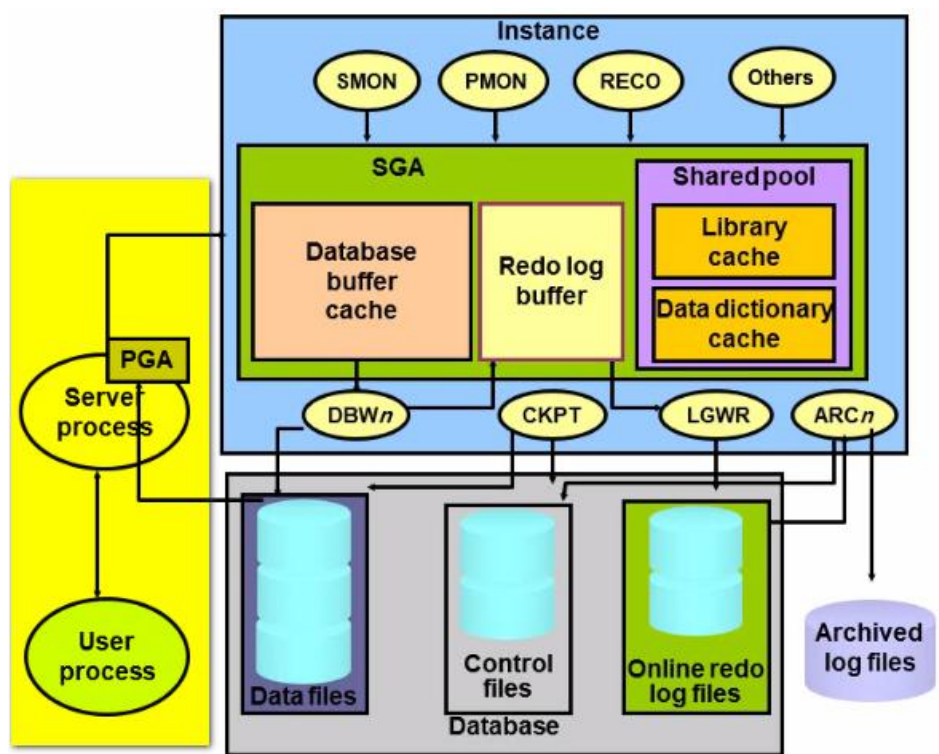
در واقع PGA اطلاعات انواع دسترسی کاربر به بانک اطلاعاتی را از سرور می گیرد و در خود نگه داری می کند. در قسمت Server Process پردازش انجام میشود و نتیجه در PGA قرار می گیرد.

و کاربر توسط بر اساس دسترسی هایی که PGA قرار دارد به instance متصل می شود.

در این لحظه است می گوئیم یک اتصال یا Connection صورت گرفته است.

ولی مجموعه کل موارد از سمت کاربر تا Instance یک Session نامیده می شود.

ممکن است که n کاربر به یک دیتابیس متصل شوند که n تا Session هم خواهیم داشت. که اگر کاربر به مشکلی برخورد کرد یا fail شد باید این Session بسته شود. که در درس های بعدی در مورد ان صحبت خواهد شد.



اطلاعات یک اتصال : وقتی می خواهیم به دیتابیس متصل شویم ، اتصال (Connection) باید اطلاعات زیر را داشته باشد:

نام آن دیتابیس (SID) را داشته باشد ، username و password ای که کاربر از طریق آن قصد دارد به آن دیتابیس وصل شود IP , و Port سروری که دیتابیس بر روی آن قرار دارد.

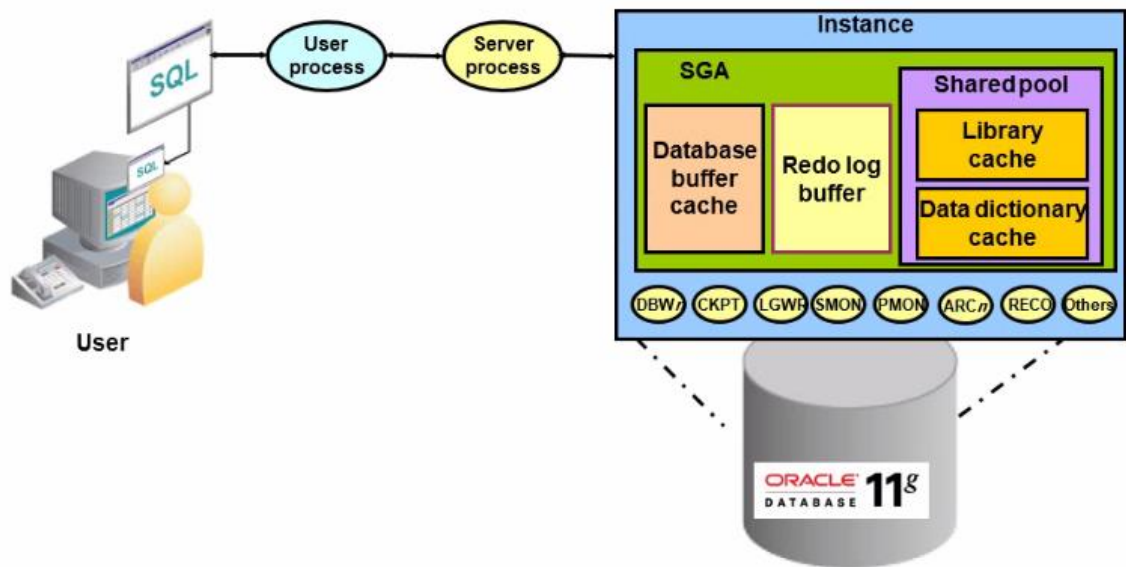
با استفاده از این اطلاعات وارد می شود و PGA بررسی می کند که آیا دسترسی های لازم را دارد؟ اگر داشت مجوز ورود را به کاربر می دهد.

تعریف: Session

یک ارتباط خاص با Instance از طریق یک User Process

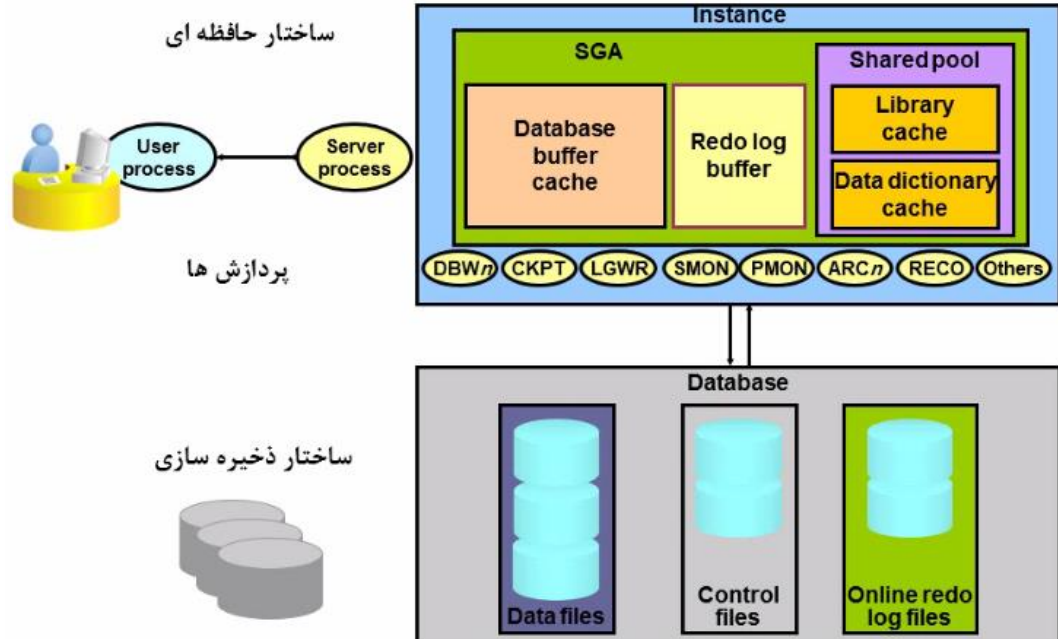
در واقع یک کاربر از زمانی که کار خود را شروع می کند و به دیتابیس متصل شود و کارهایی در آن دیتابیس انجام دهد تا ارتباطش با دیتابیس در آن نشست پابرجا باشد در مجموع به آن یک session گفته می شود.

ارتباط با دیتابیس اوراکل



همانطور که گفته شد کاربر به وسیله یک کلاینتی یک دستور SQL را وارد می کند و قصد دارد که وارد شود. در واقع یک user process می شود و سپس یک Server Process به آن اختصاص داده می شود. هر Server Process یک PGA که به کاربر اعلام می کند که آیا دسترسی دارد یا ندارد. و در صورت داشتن مجوز اتصال، Connection برقرار می شود.

ساختار سرور بانک اطلاعاتی اوراکل

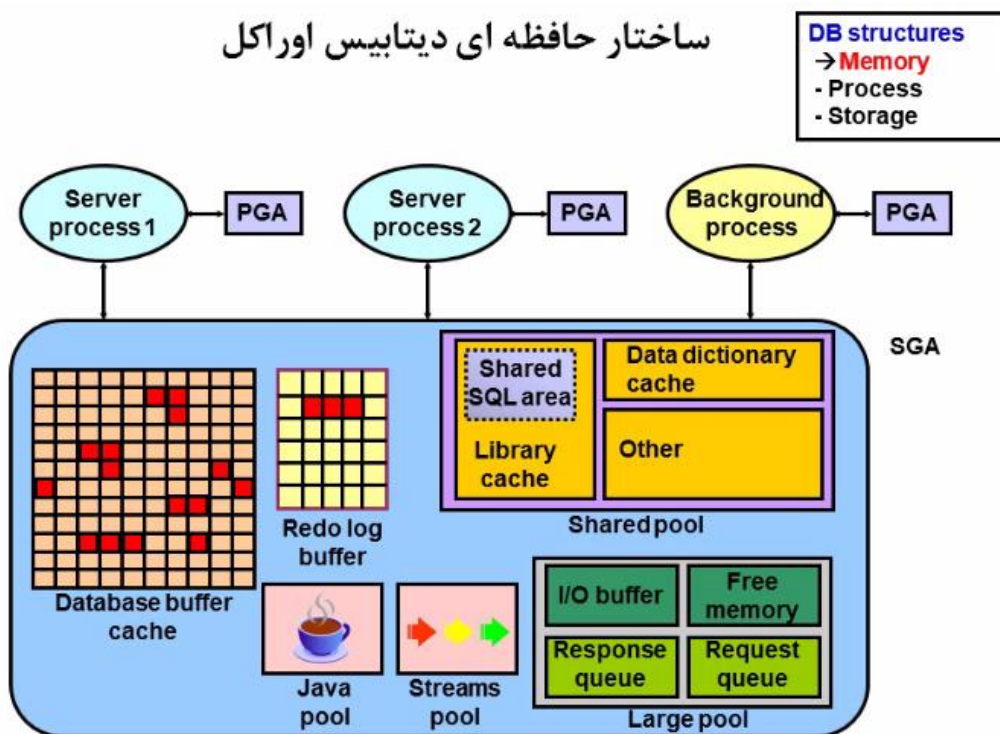


ساختار سرور بانک اطلاعاتی اوراکل

ساختار سرور بانک اطلاعاتی اوراکل : **ساختار حافظه ای** دارد، با استفاده از Background Process ها **پردازش** صورت می گیرد و در نهایت با استفاده از مکانیزم های مختلف در دیتابیس و **ساختار ذخیره سازی**، داده ها ذخیره می شوند. همه داده های اصلی در **Data Files** قرار می گیرند.

در سرور بانک اطلاعاتی، ما یک ساختار فیزیکی (Physical) داریم که از طریق سیستم عامل می توانیم به آنها دسترسی داشته باشیم، مثل دیتابیس فایل ها (DB files) و پردازش ها (Process) و یک ساختار منطقی (Logical) داریم که از طریق دسترسی به دیتابیس می توانیم به آنها دسترسی داشته باشیم، مثل آبجکت های دیتابیس (DB objects)، Table و Index و...

ساختار حافظه ای دیتابیس اوراکل



مهمترین قسمت ها در ساختار حافظه ای دیتابیس اوراکل Database buffer cache ، Redo log buffer و Shared pool هستند که در مجموعه ای به نام SGA قرار دارند.

در شکل بالا دو کاربر قصد اتصال به دیتابیس را دارند که دو Server Process ایجاد شده است و به ازای هر Server Process یک PGA ایجاد می شود.

خود Background Process هایی که در ساختار حافظه ای دیدیم ، آنها نیز به صورت تک تک یک PGA دارند) بالاخره باید دسترسی آن کار و پردازش را داشته باشند ، پس قطعا یک PGA دارند).

SGA

SGA یا Shared Global Area کارهای زیر را انجام میدهد:

تخصیص حافظه : حافظه را بین database buffer cache و ... تقسیم می کند.

شامل داده : یکسری داده در خود دارد.

اطلاعات کنترل فایل

ارتباط با یک Instance

PGA

PGA یا Program (Private) Global Area در زمان شروع یک پردازش ایجاد می شوند

شامل یکسری داده است : مثل دسترسی هایی که آن کاربر به آبجکت های مختلف دارد.

اطلاعات کنترل فایل

به طور انحصاری مربوط به یک Server Process یا Background Process است.

UGA

UGA یا User Global Area در زمان شروع پردازش ایجاد می گردد ولی همیشه مورد استفاده قرار نمی گیرد.

شامل موارد زیر می شود:

OLAP Pool: در گذشته درباره OLAP (واکشی بالا) ذکر شد که UGA در آن زمان کاربرد خواهد داشت.

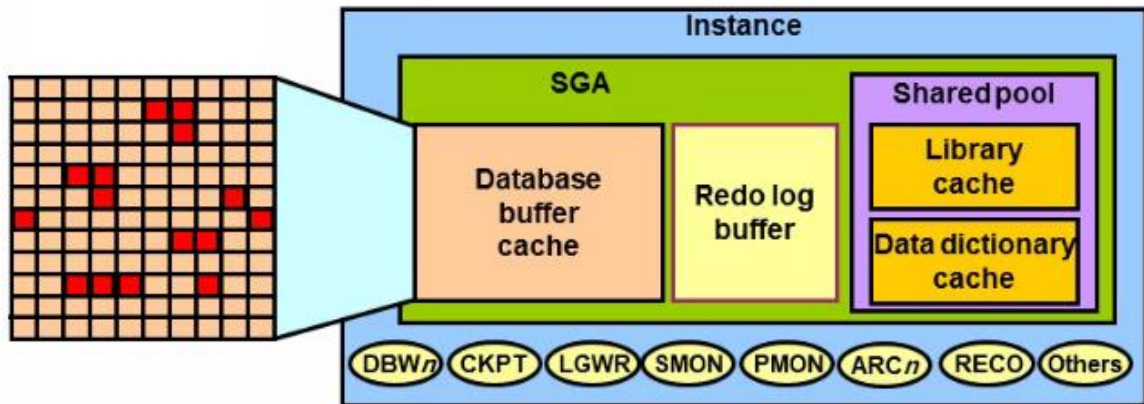
Session Variable: کل آن پروسه ای که برای برقراری دیتابیس صورت میگیرد Session خواهد بود. و وقتی دیتابیس از حجم عظیم داده ها استفاده می کند از این مورد استفاده می کند.

نکته ای که وجود دارد این است که : به طور انحصاری مربوط به هر User Session می باشد . یعنی به یک Session کامل ، یک UGA اختصاص داده میشود. در واقع هر User Process ای فضایی از حافظه ما را میگیرد(اشغال میکند) ، و وقتی ما داریم حجم زیادی از داده ها را واکشی می کنیم قطعا به یک حافظه بیشتری احتیاج داریم ، یعنی حد معمول آن دیگر جوابگو نیست و باید بیشتر از آن باشد و اینجاست که UGA به کار می آید. و برای واکشی حجم عظیمی از داده ها کمک می کند.

اجزای تشکیل دهنده: SGA

در این بخش ابتدا به ساختارهای حافظه در یک Oracle Instance می پردازیم. به طور کلی دو ساختار اصلی از حافظه در اوراکل وجود دارد. اولین قسمت و مهمترین قسمت System Global Area یا SGA میباشد . بیشتر اوقات وقتی از حافظه صحبت میشود منظور همان SGA میباشد. خود SGA تشکیل شده از قسمت های مختلفی از جمله Buffer Cache, Shared Pool, Redo Log Buffer می باشد. البته اجزای دیگری در این قسمت وجود دارد که به شرح آنها نیز خواهیم پرداخت.

Database Buffer cache



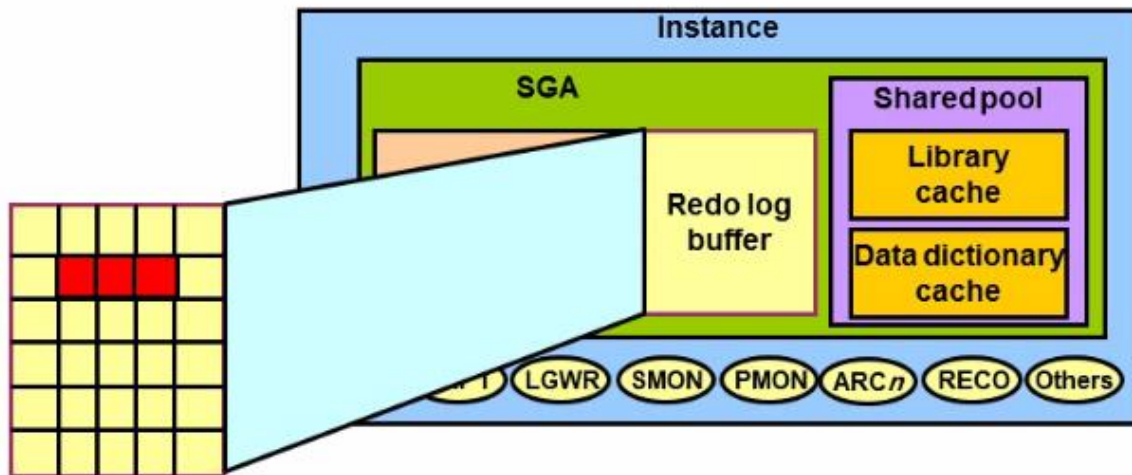
بخشی از SGA می باشد ، و در واقع یک کپی از داده های اصلی (یک کپی از بلوک های داده) که در دیتابیس ذخیره شده است به عنوان کش (Cache) در Database Buffer Cache قرار می دهد برای اینکه می خواهد روی آنها پردازش انجام دهد. و از این اطلاعات برای پردازش استفاده کند. چون منطقی نیست که پردازش روی داده های اصلی صورت بگیرد باید یک کپی از آن داده ها گرفته شود و به حافظه منتقل شود و کارها و پردازش ها بر روی آن انجام شود و در آخر اگر نهایی شد به دیتابیس فرستاده شود و تغییرات بر روی داده های اصلی اعمال شود.

یادآوری : کوچکترین عضو داده در دیتابیس بلاک نامیده می شود.

نگهداری بلوک های داده که از دیتا فایل (دیتافایل محل نگهداری داده های اصلی می باشد) خوانده می شود.

توسط همه کاربران به طور همزمان به اشتراک گذاشته می شود. هر کاربری که به دیتابیس متصل می شود ، هر اطلاعاتی (بلوک هایی) که نیاز دارد از دیتافایل به database buffer cache منتقل میشود و پردازش صورت میگیرد.

Redo log buffer

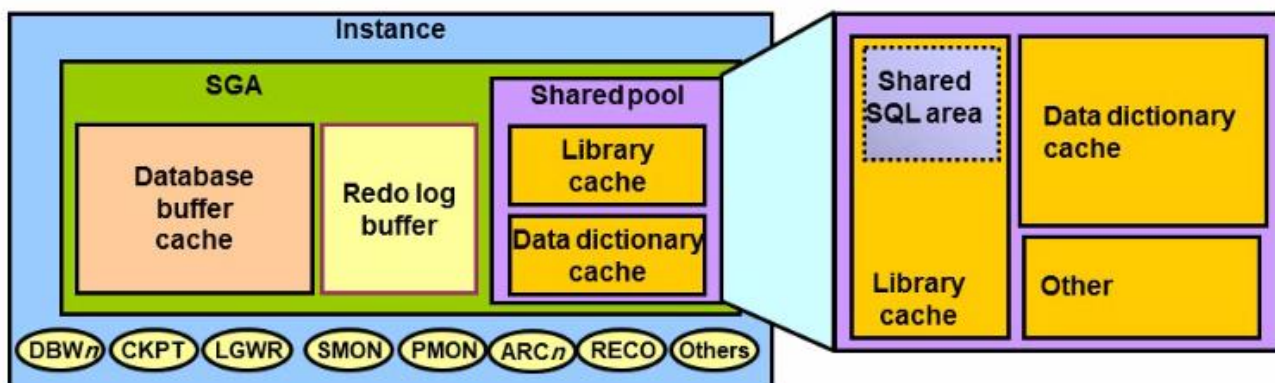


بخشی از SGA می باشد.

هرگونه تغییر در دیتابیس را در خود نگهداری می کند:

تمامی تغییراتی که در دیتابیس انجام میشود(DDL,DML,DCL)ها (در Redo Log Buffer ذخیره می شود. و در فضای فیزیکی هم قسمتی به نام Redo log file وجود دارد که اینها باید در آنجا هم ذخیره شوند، چون اینجا فقط کش (cache)و موقتی هست و اگر دیتابیس Down باشد این قسمت حافظه وجود نخواهد داشت. و هر وقت که دیتابیس بالا باشد Instance هم بالا می آید.

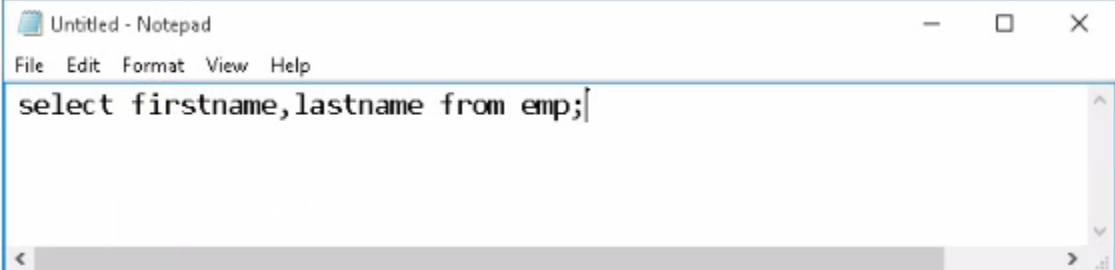
Shared Pool



بخشی از SGA خواهد بود.

شامل موارد زیر می باشد:

به وسیله یک مثال این قسمت توضیح داده میشود ، به مثال زیر دقت نمایید:

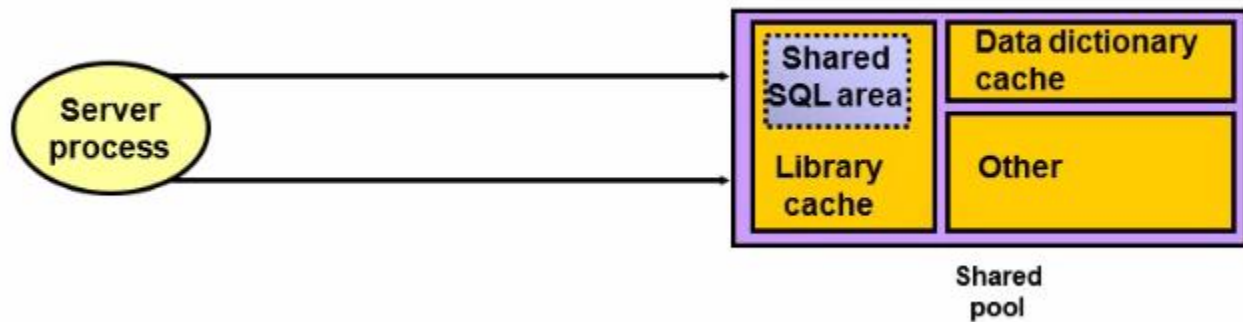


```
select firstname,lastname from emp;
```

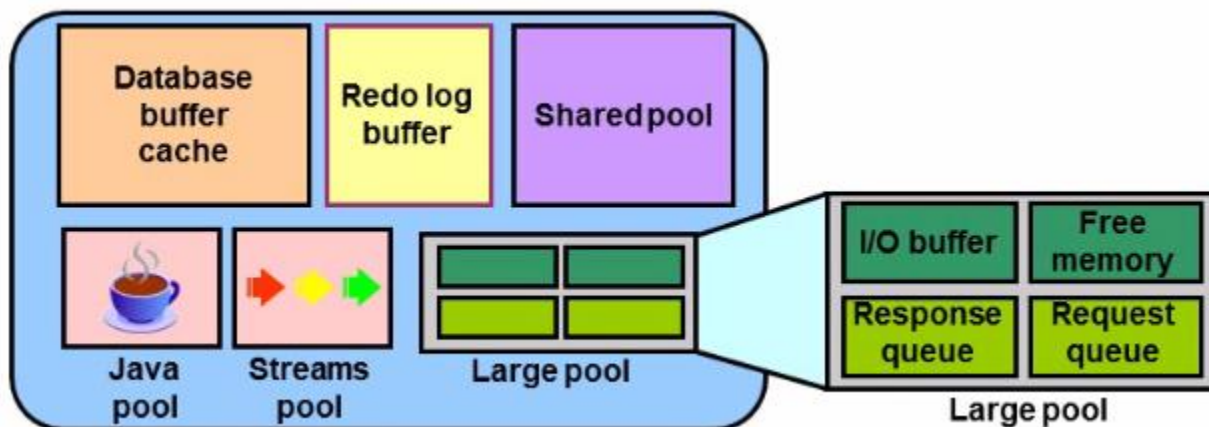
در این دستور SQL در حقیقت کاربر قصد دارد به این ساختار حافظه متصل شود. بلافاصله به User Process یک Server Process اختصاص داده می شود، این Server Process یک PGA خواهد داشت . توسط این PGA سرور پروسس بررسی میکند که آیا کاربری که این Select را بر روی جدول emp زده است ، اصلاً به این جدول دسترسی دارد؟ اگر دسترسی داشت وارد فضای حافظه می شود. و بلوک های داده ای که در دیتافایل مورد احتیاج است (فیلدهای Firstname , lastname از جدول emp) در Database Buffer Cache قرار می گیرد. برای اینکه این اطلاعات را به کاربر نشان دهد ، هر دفعه بخواهیم این اطلاعات را دوباره از دیتابیس بخوانیم و واکنشی کنیم و به کاربر نشان دهیم ، این باعث میشود که سرعت نمایش بسیار کم شود. برای این منظور دیتابیس مکانیزمی دارد که بتواند که اگر قبلاً کاربر به این جدول Select زده باشد ، اگر در کش دیتابیس بود دیگر لازم نیست که دوباره این فرآیند را انجام بدهد. از دیتابیس به کاربر نمایش دهد. که این خیلی سرعت را بالا می برد. پس وارد Shared pool می شود ، و Library Cache این اطلاعات را hash می کند(دلیل هاش کردن : اوراکل یک زبان غیرساخت یافته است و کدهای SQL برای اوراکل غیر نامفهومه باید اینها به یک زبان قابل فهم برای دیتابیس تبدیل شود تا این Select را متوجه شود.)

پس Library cache این کدها را هاش (hash) می کند. و داخل دیتا دیکشنری می رود و بررسی می کند که آیا Select هاش شده از قبل وجود دارد ؟ اگر وجود دارد که نتیجه نهایی را به کاربر نمایش می دهد و در غیر این صورت می رود از دیتافایل موارد را انجام می دهد و به کاربر نشان می دهد و کش آن در دیتابیس ذخیره می شود که اگر دفعه بعد کاربر به دیتابیس Select زد به آن دسترسی داشته باشد.

تخصیص و استفاده مجدد



Large Pool



تخصیص حجم زیادی از حافظه را فراهم می کند:

یک زمانی ما قصد داریم کارهای بزرگ و قدرتمندی انجام دهیم مثلاً در XA Interface که در ورژن Oracle Parallel Server 7.0 فراهم ساخت که **بحث پردازش های موازی** را انجام می دهد و یا پردازش تصاویر و... اوراکل از Large Pool استفاده می کند و Shared Pool را کنار میگذارد. زیرا در Shared Pool فضای حافظه ای محدودی وجود دارد ولی Large Pool تخصیص حجم زیادی از حافظه را فراهم می کند , و باعث می شود که قدرت بیشتری پیدا می کند.

پردازش های ورودی و خروجی سرور

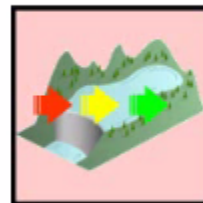
عملیات بک آپ و ریکاوری برای داده های بزرگ:

وقتی که دیتابیس ما از حجم عظیمی داده ها (مثلاً 5 یا 6 ترابایت) تشکیل شده باشد ، در این حالت Backup & Recovery بر طبق روال عادی امکان پذیر نمی باشد ، پس برای Backup های بزرگ هم از Large Pool استفاده می شود.

Java Pool and Stream Pool



Java pool

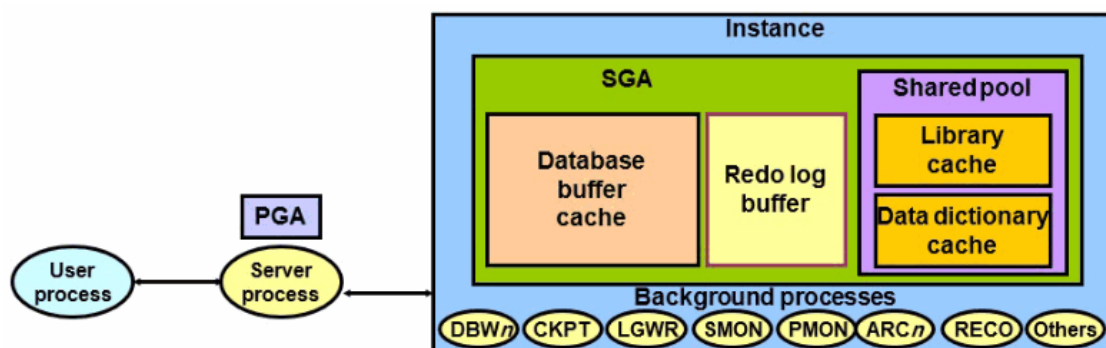


Streams pool

Java Pool زمانی که از کدهای جاوا استفاده می شود کاربرد دارد.

Stream Pool مربوط به پردازش های Oracle Stream (پردازش های موازی) هست. البته بیشتر در ورژن های قدیمی استفاده میشد که اوراکل همچنان آنرا حفظ کرده است.

معماری پردازش



یادآوری: یک User Process یک سرور پروسس می گیرد یک Server Process به آن اختصاص پیدا می کند (، هر Server Process یک PGA خاص خودش را دارد که از طریق آن دسترسی آن کاربر مشخص می شود و به Instance متصل می شود.

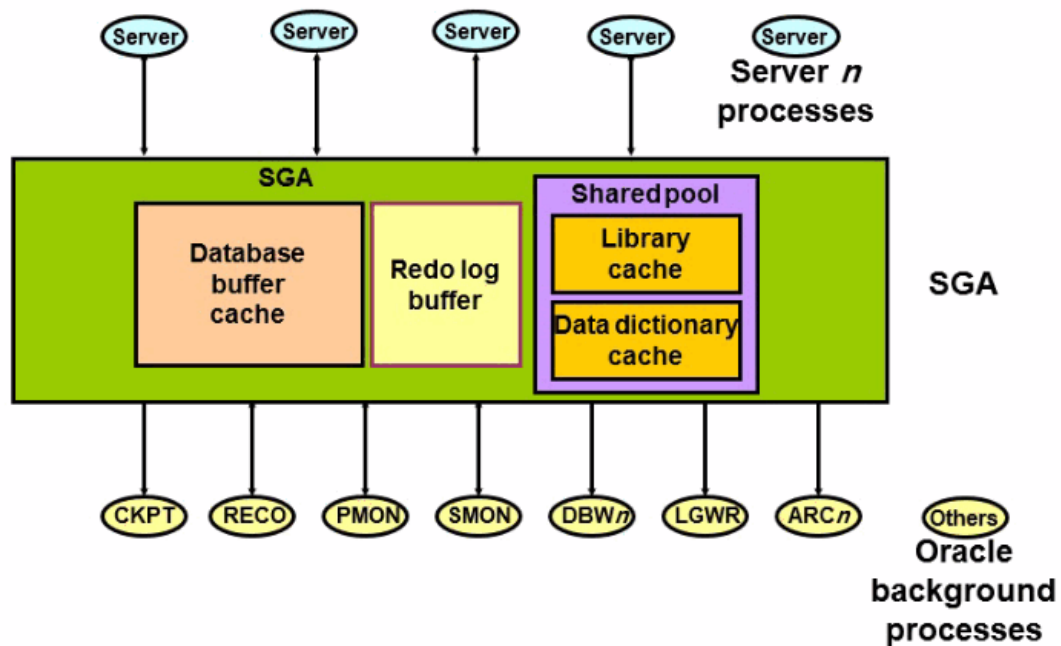
User Process: زمانی که یک کاربر یا قطعه دستور برنامه ای به اوراکل دیتابیس متصل می شود.

• پردازش های دیتابیس:

Server Process: ارتباط بین User Process و Instance را برقرار می کند.

Background Process ها: زمانی که Instance در حال اجرا می باشد ، شروع به فعالیت می کنند.

ساختار پردازش ها



همانطور که در شکل ملاحظه می شود ، **Server Process** ها می توانند **user** هایی باشند که به دیتابیس متصل شده اند (، **user** می توانیم داشته باشیم ، پس **n** تا هم **Server Process** می توانیم داشته باشیم. به **SGA** متصل می شوند و پردازش ها انجام می شود . که در حقیقت همه این پردازش ها توسط **Background Process** ها انجام می شود **Background Process** .ها مثل یکسری کارگر هستند که برای **Instance** کار می کنند.

Database Writer Process (DBW n)

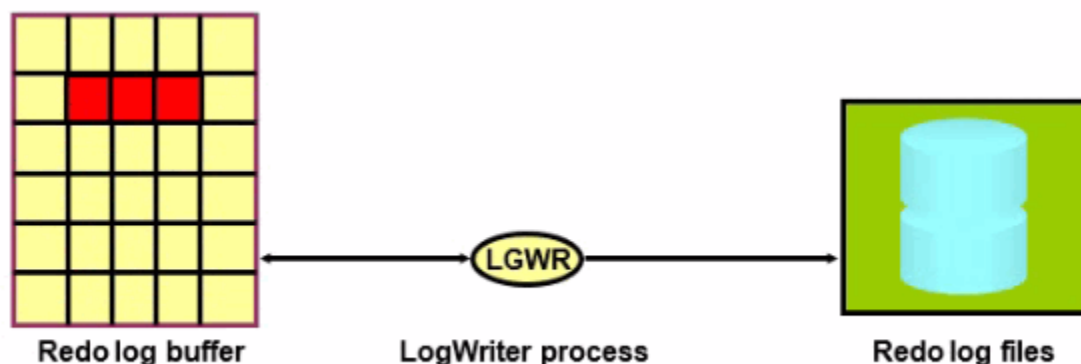


یکی از **Background Process** هایی که وجود دارد به نام **DBW_n** یا **Database Writer Process** نام دارد.

ما در اوراکل چند مدل بلوک داده ای داریم ، یکی از بلوک های داده ای به نام Dirty است DBWn. کارش این است که بافرهایی که در اصطلاح dirty نام دارند را از Database buffer cache بر می دارد و در دیتافایل (data file) ذخیره می کند.

وقتی دیتای ما مشکلی نداشته باشد و پردازش نهایی روی آن انجام شده باشد و در اصطلاح commit (تثبیت نهایی) شده باشد. باید مستقیماً در دیتابیس ذخیره شود. و آنهایی که dirty هستند یعنی تثبیت نهایی شده اند. و توسط database buffer cache باید ذخیره شوند.

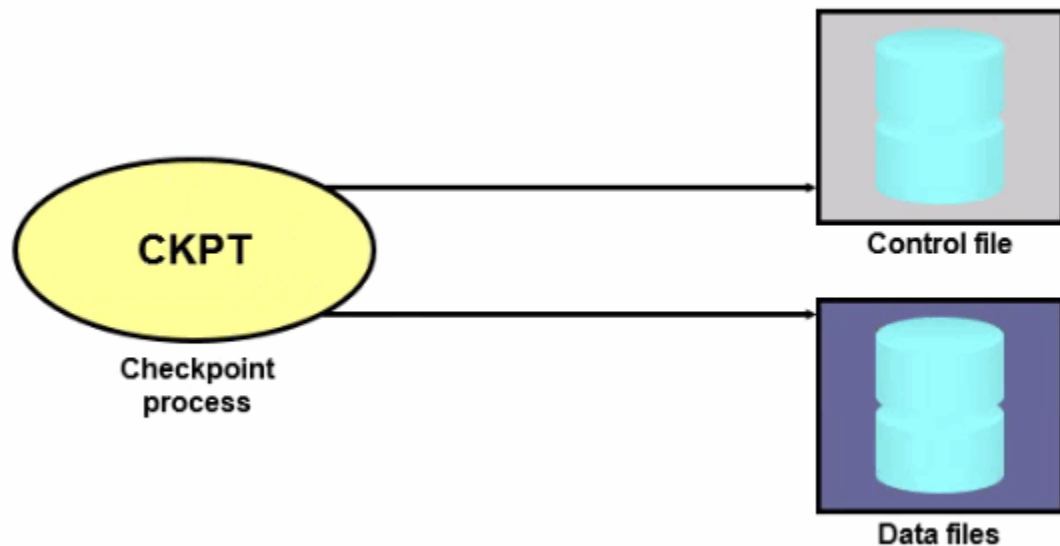
LogWriter Process (LGWR)



تمام Log هایی که داخل Redo log buffer هست را (تغییرات) داخل Redo log files که به صورت فیزیکی هست ذخیره می کند.

این تغییرات در زمان Commit یا زمان پر شدن Redo log buffer نوشته می شود. هر زمانی که Commit ای اتفاق می افتد ، و هم به روش دستی (لاگ سوییچ) می توان این کار را انجام دهیم . و وقتی لاگ سوییچ اتفاق می افتد مطمئن هستیم که تغییرات در Redo log files ذخیره میشود. که اینها در Backup و Recovery مورد استفاده قرار می گیرند. فرض کنید دیتابیس Crash کرده است ، باید دیتابیس بداند که چه زمانی Crash شده است و چه زمانی دیتابیس سالم بوده است. این لاگ ها همه در Redo log files ها ذخیره می شود.

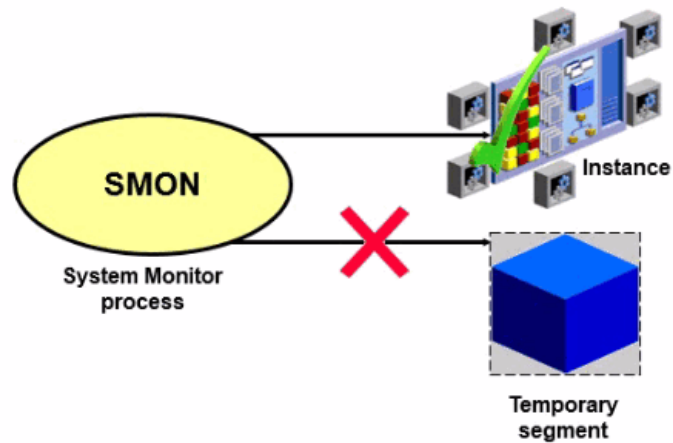
Checkpoint Process (CKPT)



در فضای ذخیره سازی یک دیتافایل و یک کنترل فایل داریم ، دیتافایل ها و کنترل فایل ها یک هدر دارند در آن یکسری اطلاعات ذخیره میشود.

این هدر ها برای این است که اولاً ارتباط بین دیتافایل و کنترل فایل برقرار شود و بتوانند با هم در ارتباط باشند ، و مورد دیگر اینکه اگر هر تغییری انجام شد در هدر دیتافایل یک checkpoint درج میشود. این Checkpoint ها در قسمت backup و Recovery مورد استفاده قرار میگیرند. مثلاً دیتابیس جایی Crash کرده است و می خواهیم بدانیم که تا کجا Checkpoint خورده است ، یعنی تا آنجا دیتابیس سالم بوده است و هیچ مشکلی نداشته است و می توانیم تا آنجا را Recovery کنیم. اما زمانی که checkpoint خورده نشده بود یعنی در آنجا مشکلی به وجود آمده است و دیتابیس در حالت نرمال نبوده است.

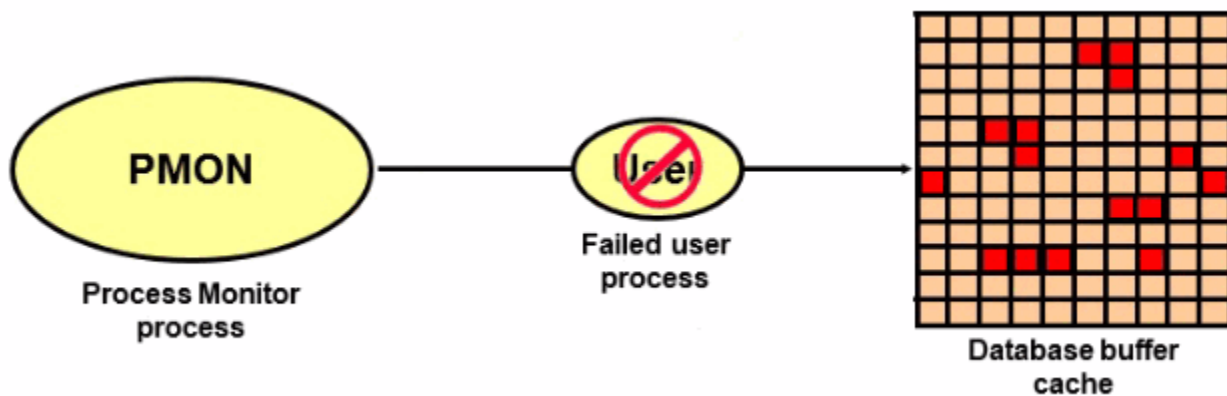
System Monitor Process (SMON)



در SMON زمان بالا آمدن سیستم و زمانی که دیتابیس شروع به کار می کند عمل Recovery را بر روی دیتابیس ما انجام می دهد. مثل اتاق عمل ، عمل جراحی در حالت بیهوشی انجام میشود (=خاموش بودن دیتابیس) ، بعد از عمل جراحی زمانی که بیمار به هوش می آید وارد اتاق ریکاوری می کنند و این عمل ریکاوری برای دیتابیس هم اتفاق می افتد ، برای یکسری اتفاقاتی در دیتابیس افتاده مثلاً بعضی کاربران کارهایی را انجام داده اند و کارهای خود را نبسته اند و ... ، SMON ریکاوری می کند و به صورت نرمال در می آورد و دیتابیس بالا می آید.

عمل ریکاوری را انجام می دهد نه Recovery دیتابیس ، ریکاوری برای بالا آمدن Instance ، این دو ریکاوری با هم کاملاً فرق می کند.

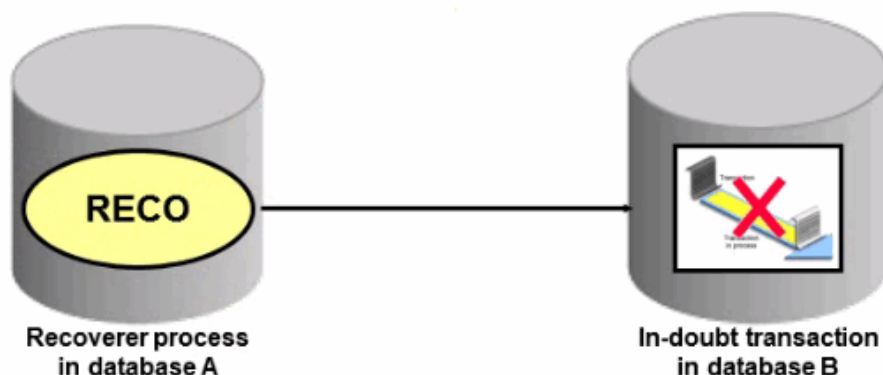
Process Monitor Process (PMON)



PMON مثل یک سرایدار می ماند ، کاربر به عنوان User Process در دیتابیس یکسری کارها را انجام داده است و رها کردی و رفته است و Session آن باز مانده است و این باعث می شود که فضای حافظه گرفته شود. باید مکانیزمی

باشد که این Session ها را ببندد ، مداوم چک کند (یک روز یکبار ، یک هفته یکبار و...) و اینگونه Session ها را می بندد . که در غیر این صورت ممکن است منجر به Crash کردن دیتابیس شود.

Recovery Process



در زمان backup و Recovery کاربرد دارد.

Archiver Processes (ARCn)



جایی در اوراکل همه لاگ ها به صورت فیزیکی ذخیره میشود که در قسمت backup و Recovery از آن استفاده می شود.

حالا این Redo log فایلها چه کاربردی دارند؟ سالم بودن دیتابیس را نشان می دهد ، و می فهمیم که تا کجا خطا بوده است . و در ضمن تمام کش ها در داخل Redo log هست.

به غیر از اینکه این لاگ ها را به صورت فیزیکی داشته باشیم ، یک جایی وجود دارد که یک کپی از آن گرفته می شود . برای این منظور یک کپی از Archive ها گرفته میشود و در جایی مثل صندوق نگه داری می کنیم و برای این منظور می توانیم یک مقصد و مسیر تعیین نماییم . اوراکل جایی را به نام Flash Recovery Area تعبیه کرده است

که یک کپی از Redo Log ها را در آن فضای ذخیره سازی قرار می دهد تا در مواردی که برای سیستم مشکلی پیش آمد بتوان از آن استفاده نمود.

Archive log ها به صورت پیش فرض در اوراکل فعال نمی باشد.

دیگر پردازش ها:

MMON

مدیریت Background Task ها (مدیریت وظایف) ، مثلاً Schedule ها ، که به آنها اعلام می کنیم که در بازه های زمانی مشخص یک کار مشخص را انجام دهد.

MMNL

به MMON کمک می کند MMON. مثل رییس MMNL می ماند ، که دستورات MMON را اجرا می کند

MMAN

در زمان تخصیص اتوماتیک حافظه کاربرد دارد.

وقتی اوراکل را نصب می کنیم باید حافظه ای به آن داده شود ، مثلاً Ram سیستم ما 10 گیگابایت است ، باید یک مقدار از Ram را برای سیستم عامل نگه داریم و مقداری از Ram را هم به دیتابیس تخصیص بدهیم. و MMAN این حافظه ای که ما به اوراکل تخصیص داده ایم را بین تمام عناصر SGA تقسیم می کند. البته می شود این کار را به صورت استاتیک(دستی) هم انجام دهیم که این کار بسیار کار دشوار و خطرناکی است . البته در نسخه های قدیمی اوراکل باید به صورت دستی این کار انجام می گرفت و اوراکل در نسخه 10G این مورد را اصلاح کرد و می توانیم این کار را به خود اوراکل واگذار کنیم. تمام مدیران دیتابیس (DBA) ها (علاقه مند هستند که این امر به صورت اتوماتیک توسط اوراکل مدیریت شود.

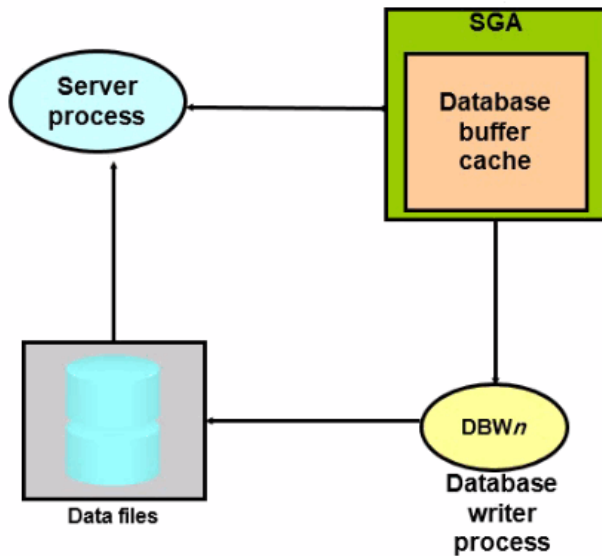
CJQ0

برای اجرای یک Job است.

QMNC

در بحث Stream ها کاربرد دارد

ارتباط Database Buffer Cache و Server Process



دیتابافر کش با استفاده از سرور پروسس اطلاعاتشان را در Database Writer ذخیره می کنند.

قبلاً درباره حالت های بلوک داده در داخل Database Buffer Cache اشاره ای کردیم ، که این بلوک ها می توانند حالت های مختلف زیر را داشته باشند:

Pinned: یعنی این بلوک داده ای در اختیار یک تراکنشی است که سیستم با آن در حال کار است.

Clean: یعنی این بلوک داده پر است و هیچ مشکلی هم ندارد و این بلوک داده می تواند در دیتافایل شود.

Free or unused: یعنی اینکه این بلوک داده ای خالی است و چیزی داخل آن نیست و می شود از آن استفاده کرد تا داده ای در داخل آن قرار بگیرد.

Dirty: بلوک های Dirty پر هستند ولی هنوز نهایی نشده است ، هنوز بلا تکلیف است و منتظر تثبیت است.

معماری ذخیره سازی دیتابیس

معماری ذخیره سازی دیتابیس

ساختار دیتابیس

- حافظه

- پردازش

- ذخیره سازی



Control files



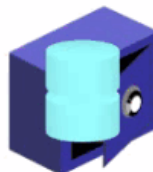
Data files



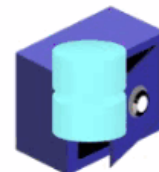
Online redo log files



Parameter file



Backup files



Archived redo log files



Password file



Alert log and trace files

Control Files در آینده در مورد این مبحث مفصل بحث خواهد شد.

Data files تمامی داده های فیزیکی در داخل Data File ها قرار دارند.

Online Redo log files تمام تغییرات در دیتابیس در این قسمت ذخیره می شود.

Parameter files یکسری پارامترها هستند که خیلی ضروری هستند و اوراکل از آن استفاده می کند (در بحث

های بعدی مثل نصب اوراکل و مدیریت حافظه این پارامترها بررسی خواهند شد)

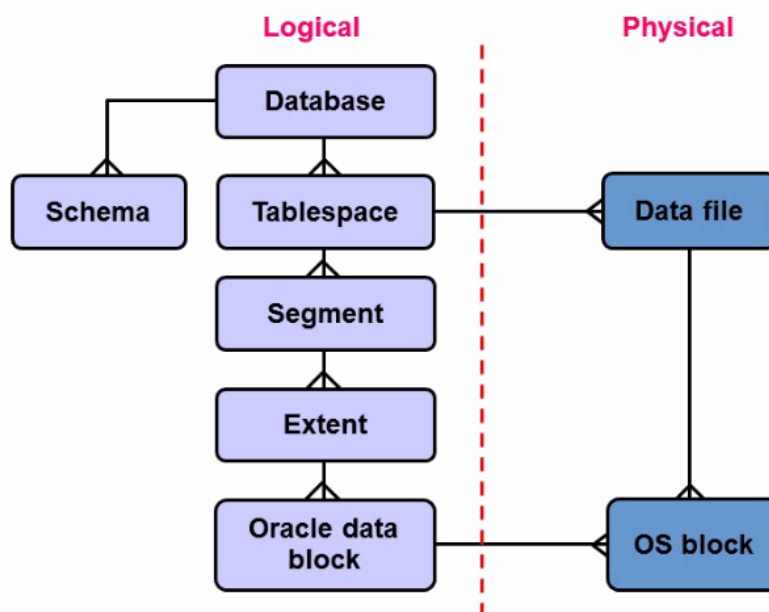
Backup File ها و Archive Redo log file ها هم قبلاً مختصراً بررسی شدند. جهت یادآوری Archive file : ها از فایل های Redo log ها یک آرشیو می گیرند ، به دلیل اینکه ما High Availability یا میزان بالا بودن در دسترسی بودن دیتابیس را بالا ببریم و دیتابیس ما همیشه در حال فعالیت باشد و Down نشود.

Password File ها: پسوردهای ادمین و غیره در این قسمت قرار می گیرد.

Alert log And Trace files: Alert log
قسمت ذخیره خواهد شد.

ساختار منطقی و فیزیکی دیتابیس

ساختار منطقی و فیزیکی دیتابیس



همانطور که در تصویر مشاهده می کنید ، بخش منطقی و فیزیکی به هم متصل هستند ، ولی بخش فیزیکی قابل لمس است و از طریق سیستم عامل می توانیم آن را ببینیم.

OS block: ما وقتی در سطح سیستم عامل هستیم ، فضای ذخیره سازی ما Hard هست که توسط داده ها در داخل آن ذخیره می شود. وقتی که اوراکل را نصب می کنیم در Hard می توانیم یک فضای جداگانه و اختصاصی را برای دیتابیس در نظر بگیریم که در اینجا OS block به Oracle Data Block تبدیل می شود. که داده های دیتابیس در داخل این بلاک ها قرار می گیرند.

در شکل بالا به نوع ارتباط بین اجزا دقت کنید که ارتباط یک به چند بین آنها حکم فرماست ، مثلاً یک دیتابیس می تواند متعلق به چند اسکیمای باشد مثلاً کاربران Admin , Ali, Hadi و ... که هر کدام از این اسکیمای یکسری جدول ، پکیج ، ایندکس و... و موارد خاص خودشان را دارند ، اما یک اسکیمای فقط متعلق به یک دیتابیس می باشد.

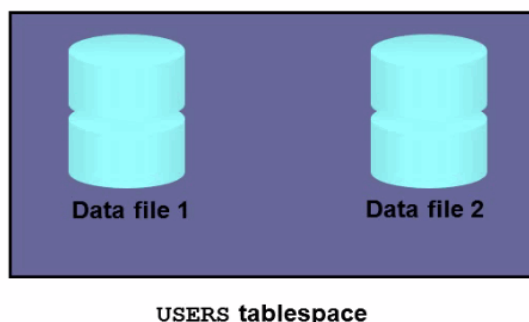
دیتابیس از یک یا چند Tablespace تشکیل شده است. و Tablespace ها از چند Segment تشکیل شده اند . Segment ها معادل همان Table ها هستند.

و هر Segment می تواند چند Extent داشته باشد و مجموعه ای از Extent ها تشکیل Segment می دهند.

و در نهایت Extent ها n تا اوراکل دیتابلاک (بلوک داده ای) دارند. بلوک های داده ای در کنار هم بسته بندی می شوند و در کنار هم تشکیل Extent می دهند و Extent ها هم در کنار هم تشکیل یک Segment می دهند

ساختار فیزیکی دیتابیس اوراکل

Tablespaces and Data Files



ساختار فیزیکی از موارد زیر تشکیل شده است:

Tablespace and Data files :

هر Tablespace شامل یک یا چند Data file هست.

Tablespace ها به صورت منطقی در اوراکل مطرح می شوند و جایی به صورت فیزیکی وجود ندارند. اوراکل دیتافایل هایی که به صورت فیزیکی دارد را به صورت منطقی در مجموعه هایی به نام Tablespace قرار می دهد.

پس هر Data File فقط متعلق به یک Tablespace می باشد.

نکته مهمی که در بحث اوراکل وجود دارد این است که این Tablespace ها فقط یک Data file داشته باشند ؟ و یا چند تا دیتافایل داشته باشند ؟ از آنجایی که این موارد روی کارایی دیتابیس تاثیر می گذارد باید دقیق بررسی کنیم که تعداد دیتافایل ها چه تاثیری در ذخیره سازی می گذارد.

Tablespace پیش فرضی که در اوراکل وجود دارد Tablespace ای به نام **Users** است که اطلاعات User های پیش فرضی که خود اوراکل تعریف کرده است داخل این Tablespace قرار میگیرد.

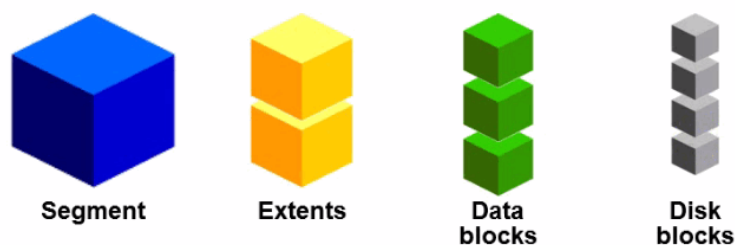
دو Tablespace مهم دیگری که در اوراکل وجود دارد و اوراکل از آنها استفاده می کند:

System Tablespace این Tablespace شامل Data Dictionary Table ها ، می باشد.

یادآوری shared pool : از library cache و Data Dictionary (کش ها و هش ها را در داخل خود نگه می دارد) تشکیل شده است.

SYSAUX Tablespace این Tablespace شامل اطلاعات Enterprise Manager می باشد.

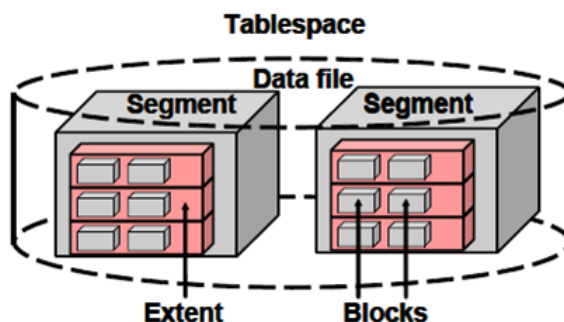
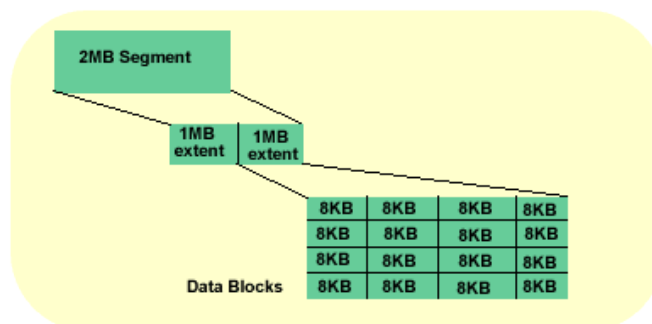
Segments, Extents, and Blocks



کوچکترین عضو اوراکل "دیسک بلاک" هست (این دیسک بلاک ها در سطح سیستم عامل هستند) و اوراکل برای ذخیره سازی داده ای خود بلاک های خود را دارد با اندازه هایی متفاوت از بلاک های دیسک . دیتا بلاک هم در کنار هم تشکیل Extent را می دهند. مثلاً یک Extent می تواند 1 تا 0 دیتابلاک داشته باشد و Extent بعدی 20 تا دیتابلاک داشته باشد.

چند تا Extent هم در کنار هم می توانند قرار بگیرند و تشکیل یک Segment یا Table را می دهند.

در آینده در مورد اینکه مقدار پیش فرض یک بلوک داده به تنهایی چقدر می تواند باشد ؟ بحث خواهد شد. مثلاً 8 کیلو بایت هست که با توجه به تجارت و Business و نرم افزار و... در هنگام طراحی دیتابیس تمایل داریم که دیتابلاک های ما چه اندازه ای داشته باشند). در Performance Tuning اینها مفصل بحث خواهد شد).



زبان دستکاری داده DML

DML مخفف Data Manipulation Language عملیات پردازشی و دستکاری اشیای پایگاه داده مانند insert ، select ، update را پشتیبانی می کند.

DML به عنوان زبان پرس و جو هم شناخته می شود و اغلب دارای قابلیت انجام محاسبات ریاضی و آماری است که عملیات گزارش گیری از پایگاه داده را آسان تر می کند.

زبان کنترل داده DCL –

DCL مخفف Data Control Language امکان تعیین نوع استراتژی های دستیابی، تعریف شاخص ها و مرتب سازی داده های پایگاه داده را می دهد.

دو دسته زبان DSL وجود دارد:

- **رویه ای (Procedural)** کاربر داده ای که نیاز دارد و نحوه دریافت آن را تعیین می کند.
- **غیررویه ای (nonprocedural) یا (Declarative)** کاربر تعیین می کند چه داده ای مورد نیاز است ولی نحوه حصول آن را بیان نمی کند.

هر سیستم پایگاه داده DSL خاص خود را دارد به عبارت دیگر هر مدل داده زبان فرعی خاص دارد. یک DSL خاص که توسط اغلب سیستم های فعلی پشتیبانی می شود SQL است SQL. یک زبان غیر رویه ای است.

سطوح داخلی، ادراکی و خارجی هریک DSL خاص خود را دارند. شیمای هر سطح توسط DSL مربوطه نوشته می شود.