Вопросы по дисциплине СУБД Oracle (15/16) уч. год.

1. Понятие СУБД. Основные категории СУБД. Архитектура ANSI-SPARC.

СУБД — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

Основные функции СУБД:

- Управление данными на дисках.
- Управление данными в ОП (в т.ч., кэширование).
- Журналирование изменений, резервное копирование, восстановление после сбоев.
- Поддержка языков БД (DML + DDL).

Состав СУБД:

- Ядро — отвечает за управление данными.
- Процессор языка БД — транслирует запросы с высокоуровневого языка на низкоуровневый.
- Подсистема поддержки времени исполнения. Сервисные программы.

Классификация СУБД:

1. По модели данных:
   - Иерархические — данные представляются в виде дерева. Пример — LDAP / AD, реестр Windows.
   - Сетевые — используют сетевую модель данных. Частный случай — графовые СУБД. Примеры — HypergraphDB, OrientDB.
   - Объектно-ориентированные — используют ОО-модель данных. Пример — InterSystems Caché.
   - Реляционные и объектно-реляционные — используют реляционную модель данных (возможно, с частичной поддержкой ООП). Примеры — Oracle, MySQL, PostgreSQL.
2. По степени распределённости:
   - локальные;
   - распределённые.
3. По способу доступа к БД.
   - Файл-серверные — данные находятся на файл-сервере, СУБД — на каждом клиентском компьютере. Примеры — M$ Access, dBase, FoxPro.
   - Клиент-серверные — СУБД находятся на сервере вместе с данными. Примеры — Oracle, M$ SQL Server, Caché.
   - Встраиваемые — СУБД встраивается в приложение, хранит только его данные и не требует отдельной установки. Примеры — SQLite, BerkeleyDB.

Архитектура ANSI-SPARC.

Предложена в 1975 г. подкомитетом SPARC (Standards Planning And Requirements Commitee) ANSI.

Архитектура СУБД включает в себя 3 уровня:

- Внешний (пользовательский).
- Промежуточный (концептуальный).
- Внутренний (физический).

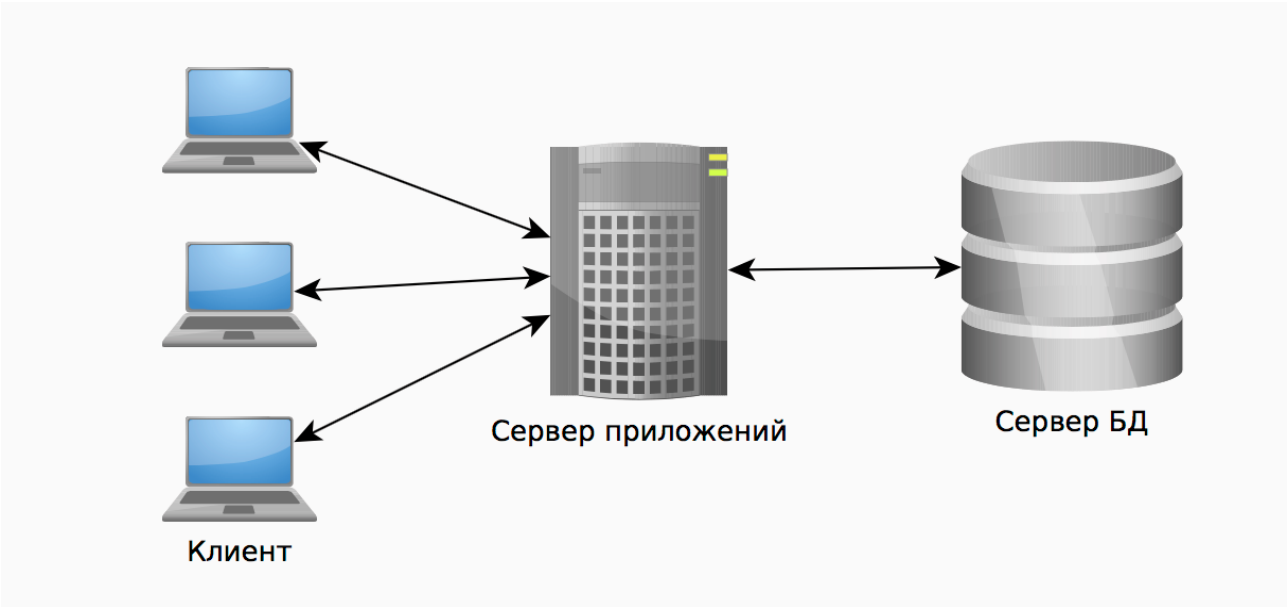
Почти все современные СУБД соответствуют принципам ANSI-SPARC.

2. СУБД Oracle. Архитектура. Подключение, взаимодействие с БД.

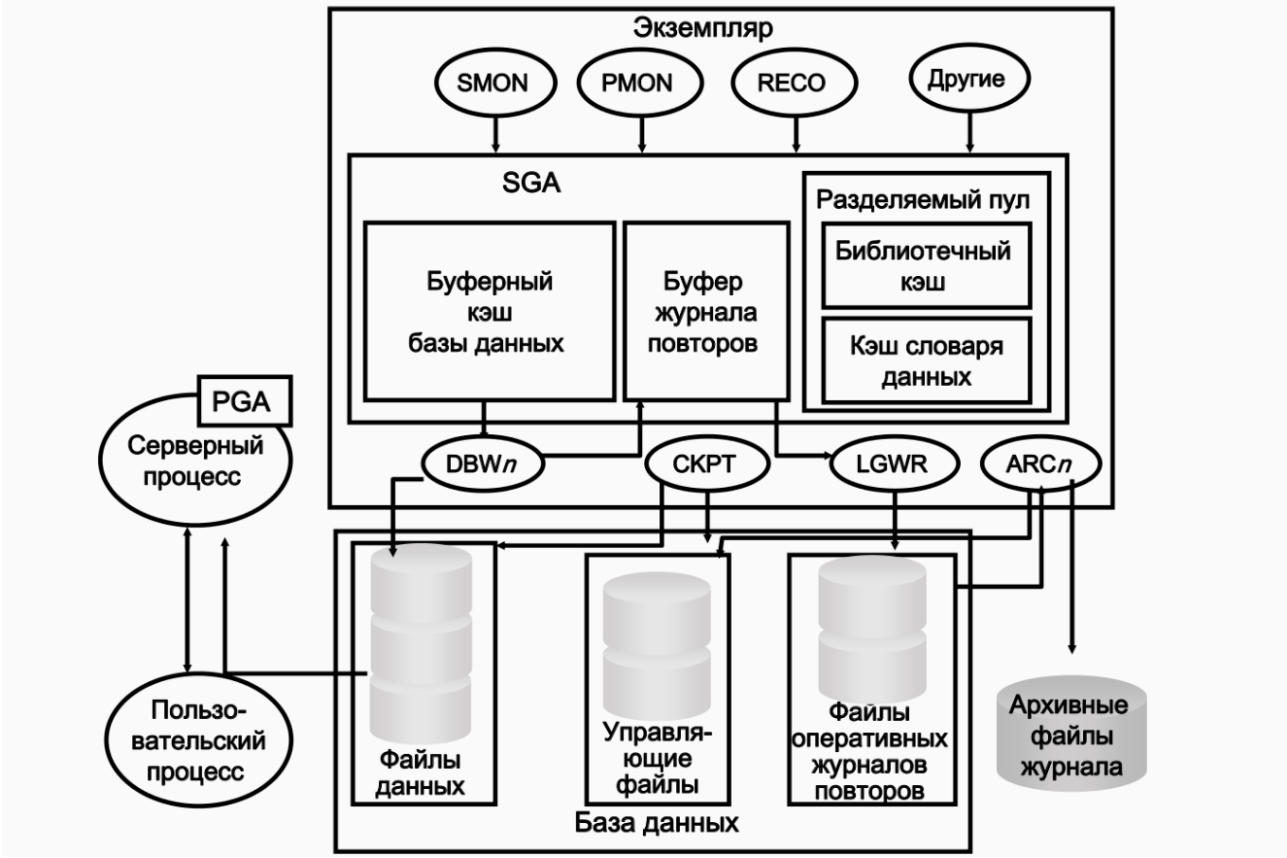
По классификации — объектно-реляционная распределённая клиент-серверная СУБД.
Очень показательный пример архитектуры Enterprise-level решения.
Поддержка транзакций, PL/SQL, ООП, технологии RAC.

Архитектура:

ИС



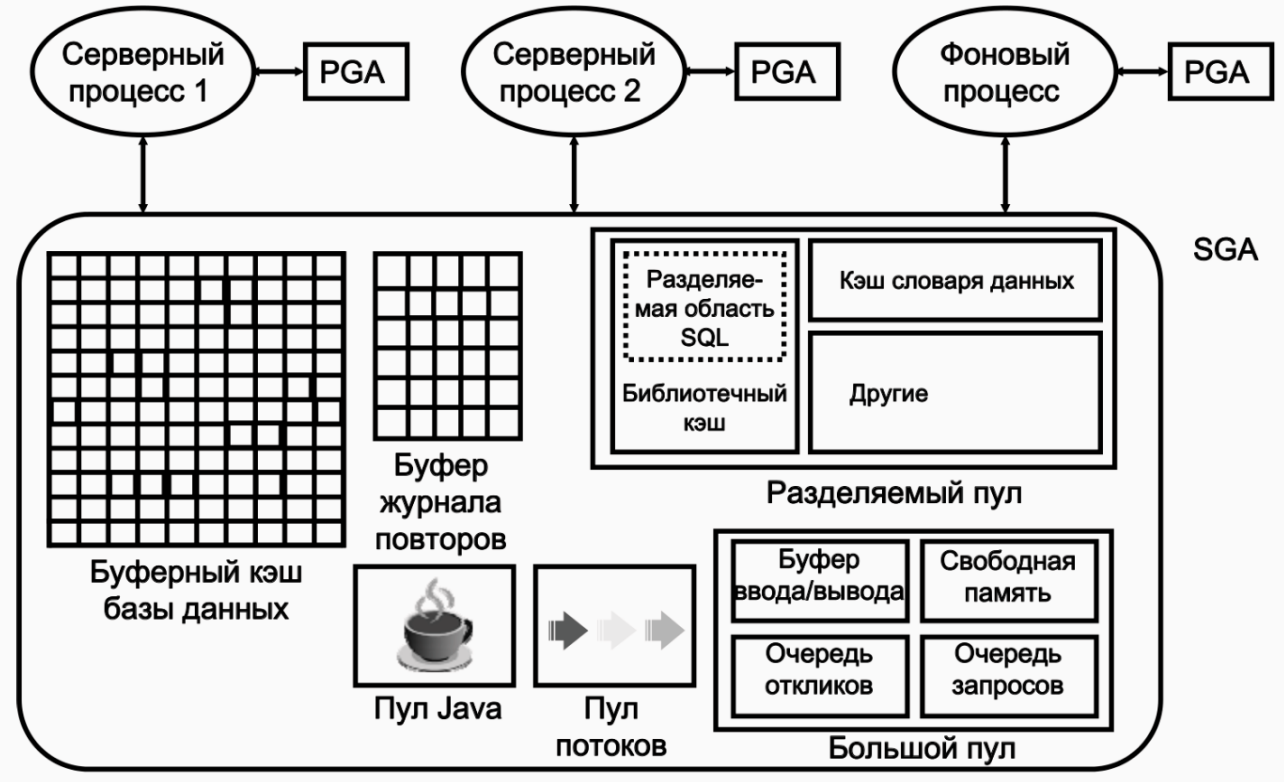
Архитектура Oracle:



Структуру БД можно рассматривать по-разному:

- На уровне структур в основной памяти ЭВМ.
- На уровне процессов в ОС.
- На уровне структуры хранилища данных в ФС

3. Структура памяти БД Oracle.



Буферный кэш:

Является частью SGA.

Хранит копии блоков данных, считанных их файлов данных.

Если нужного блока данных нет в кэше, он читается с диска и помещается в кэш.

Совместно используется всеми параллельно работающими пользователями.

Управляется сложным алгоритмом, основанным на LRU.

Буфер журнала повторов:

Циклический буфер в SGA.

Хранит информацию об изменениях в БД.

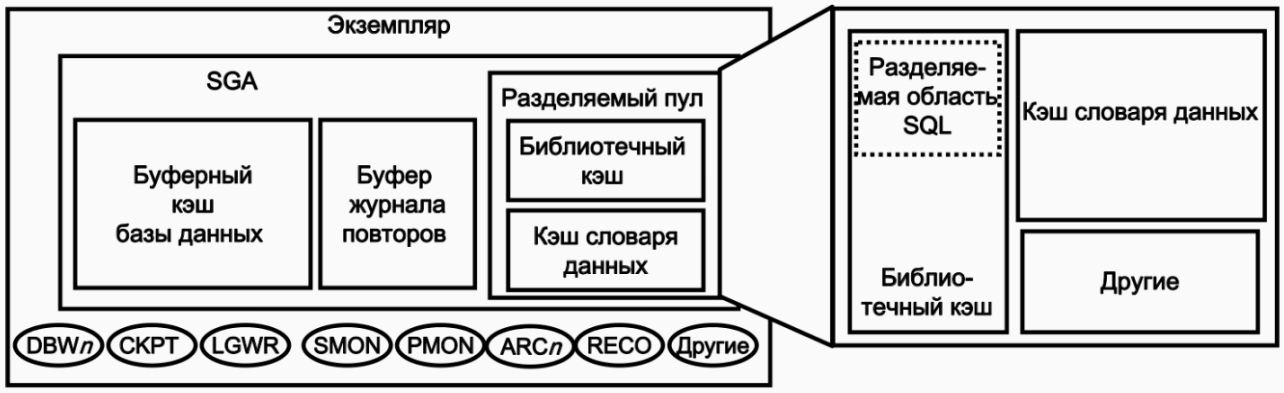
Содержит записи повторов, в которых хранится информация для повторного применения изменений, внесенных операциями DML и DDL.

Записи повторов используются для восстановления базы данных в случае необходимости.

Фоновый процесс LGWR производит запись буфера журнала повторов на диск.

Разделяемый пул:

- Область SGA.
- Структура пула:



Выделение памяти в разделяемом пуле:

Данные вытесняются из пула по алгоритму LRU.

Серверный процесс проверяет разделяемый пул на предмет наличия разделяемой области SQL для идентичного оператора.

Серверный процесс выделяет частную область SQL по запросу сеанса.

В некоторых случаях разделяемая область SQL сбрасывается целиком:

ALTER SYSTEM FLUSH SHARED_POOL;

Большой пул:

Необязательная область SGA.

Выделяется вручную администратором БД.

В отличие от разделяемого пула, нет автоматического освобождения памяти по LRU.

Может быть использован:

- Для операций передачи большого объёма данных между разными БД.
- Для операций резервного копирования / восстановления.

Размер задаётся параметром инициализации LARGE_POOL_SIZE

Каждый буфер может находиться в одном из четырёх возможных состояний:

- Закрепленный (за сеансом). Другим сеансам запрещено одновременно выполнять запись в один и тот же блок — они должны ждать, пока он освободится.
- Очищенный. Буфер является незакрепленным и становится кандидатом на немедленное устаревание, если его текущее содержимое (блок данных) не будет запрошено еще раз.
- Свободный / неиспользуемый. Буфер пуст, так как экземпляр был только что запущен. Отличие от очищенного в том, что буфер еще не использовался.
- Заполненный. Буфер больше не закреплен, однако его содержимое (блок данных) изменилось, и процессу DBWn необходимо сбросить его на диск, прежде буфер можно будет объявить устаревшим.

4. Архитектура процессов.

2 вида процессов:

- Пользовательские процессы.

  Запускаются в момент подключения пользователя к БД.

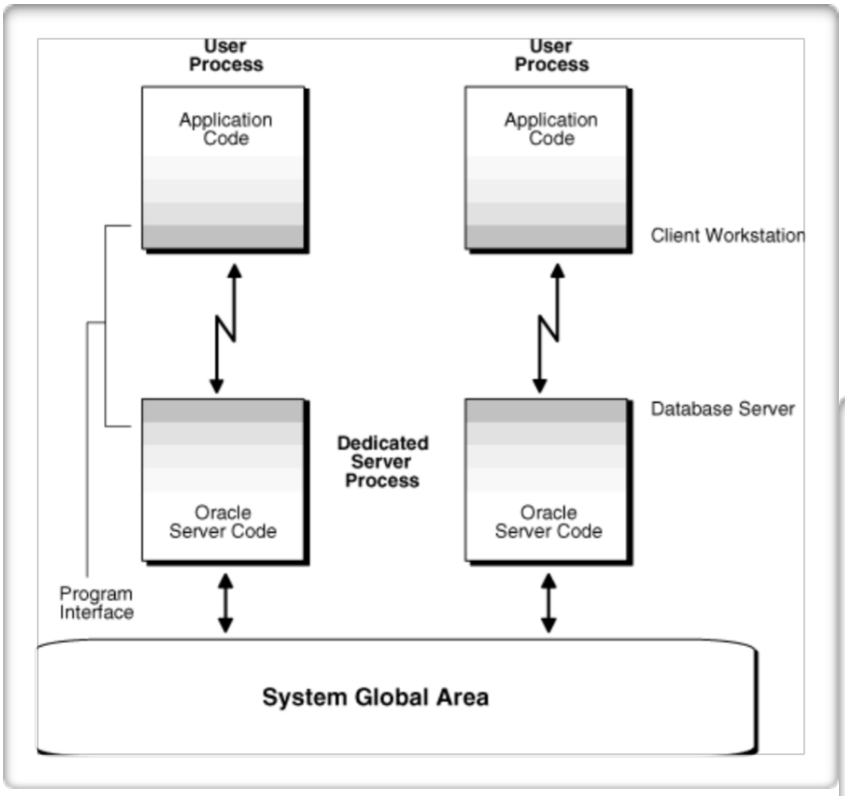
Процессы БД:

Серверный процесс: подключается к экземпляру Oracle и запускается при установлении сеанса пользователем.

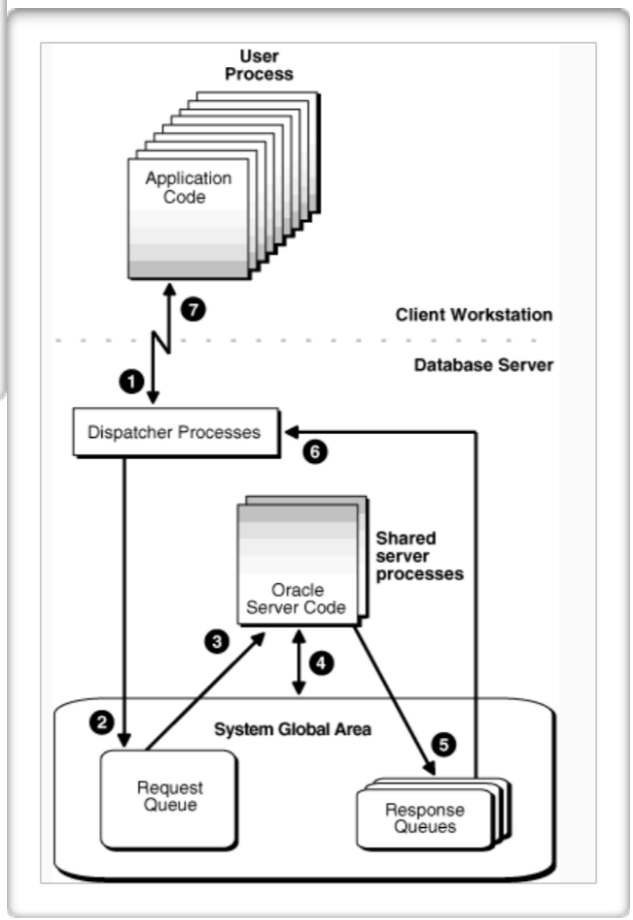
Фоновые процессы: запускаются при запуске экземпляра Oracle.

Режимы:

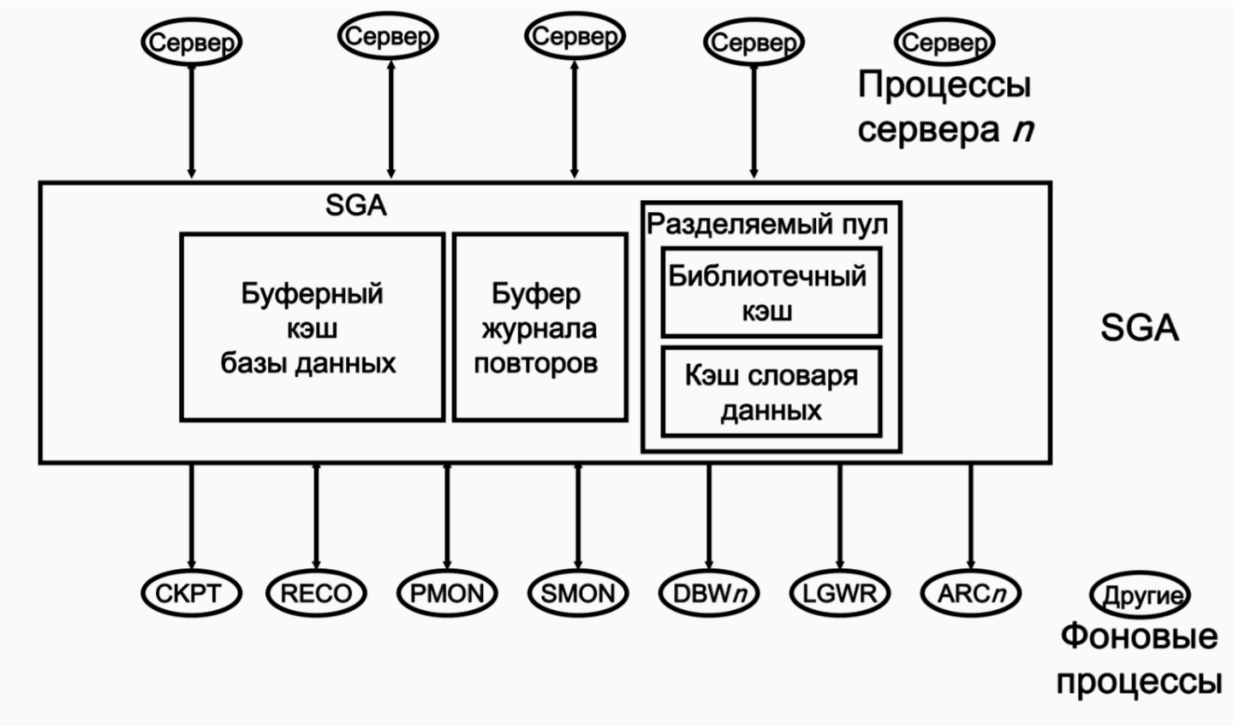
1. Выделенный сервер: каждому пользовательском у процессу создаётся свой «персональный» серверный.



2. Разделяемый сервер: каждому пользовательскому процессу серверные процессы выделяются диспетчером из специального пула.



Структуры процессов:



5. Запись данных в БД, логика работы процесса записи.

Процесс DBWn

Записывает измененные (заполненные) буферы из буферного кэша базы данных на диск:

- асинхронно во время выполнения другой обработки;
- периодически для перехода к следующей контрольной точке.

Администратор может сконфигурировать СУБД на использование до 20 параллельных процессов DBW с помощью параметра инициализации DB_WRITER_PROCESSES.

Логика:

Изменения записываются в файлы в том порядке, в котором они были сделаны (согласно SCN — System Change Number).

Для этого используется LRUW (LRU-Write) — список заполненных буферов в кэше, отсортированный по SCN.

При записи данных в файл DBW одновременно «перемещает» указатель на контрольную точку, с которой будет начато восстановление в случае сбоя (инкрементальная установка контрольных точек).

6. Запись журнала повторов, логика работы процесса записи.

Процесс LogWriter (LGWR)

Записывает буфер журнала повторов в файл журнала повторов на диске.

Запись осуществляется:

- когда пользовательский процесс фиксирует транзакцию;
- когда буфер журнала повторов заполняется на одну треть;
- перед тем как процесс DBWn записывает измененные буферы на диск.

После того, как данные из буфера журнала повторов записаны на диск, серверные процессы могут записать на их место новые данные.

Логика

Данные в файл журнала повторов записываются сразу же после того, как пользователь вызвал оператор COMMIT. Т.е., данные в журнал повторов обычно записываются раньше, чем в файлы данных.

Это называется механизмом *быстрой фиксации транзакции*.

7. Создание контрольной точки, процесс архивирования журнала повторов.

Процесс CKPT

Контрольная точка – это структура данных, определяющая SCN в потоке повторов базы данных.

Контрольные точки записываются в управляющий файл и в заголовок каждого из файлов данных. Эту операцию выполняет процесс CKPT.

Процесс CKPT не записывает блоки на диск, эту операцию выполняет процесс DBWn.

Запись номеров SCN в заголовки файлов гарантирует, что все изменения, внесенные в блоки базы данных до фиксации данного номера SCN уже записаны на диск.

Процесс архивирования журнала повторов - процесс ARCn

Копируют файлы журнала повторов на указанное устройство хранения после заполнения журнала.

Могут собирать данные для восстановления транзакций.

Функционируют, только если БД работает в режиме ARCHIVELOG.

Можно изменить максимальное количество процессов архиваторов при помощи параметра инициализации LOG_ARCHIVE_MAX_PROCESSES

8. Установка БД. Основные задачи администратора при установке в среде Unix-подобных систем.

Для корректной работы Oracle настоятельно рекомендуется перед установкой задать значения ряду переменных окружения:

- ORACLE_BASE. Устанавливает путь к «корню» иерархии каталогов Oracle. Например: export ORACLE_BASE=/u01/app/oracle
- ORACLE_HOME. Устанавливает путь к «корневому» каталогу БД. Этот путь свой для каждого экземпляра БД. Пример: export ORACLE_HOME=$ORACLE_BASE/product/11.1.0/db_1
- ORACLE_SID. Задаёт имя экземпляра Oracle. Значение по умолчанию - ORCL. Формат — строка, состоящая из цифр и букв и начинающаяся с буквы.
- NLS_LANG. Устанавливает язык и кодировку БД. Формат - язык_местность.набор символов, например: export NLS_LANG=RUSSIAN_CIS.AL32UTF8

Способы установки:

- С помощью Oracle Universal Installer — в интерактивном режиме с помощью графической утилиты (написанной на Java):
  ```
  ./runInstaller
  ```
- «Silent Mode» — с помощью файла конфигурации, (Response File) заданного в ходе одной из предыдущих установок:
  ```
  ./runInstaller –record -responseFile
  ./runInstaller -silent -responseFile
  responsefilename
  ```

Установка Oracle под *nix:

```
$ su
# password:
# cd /u01/app/oracle/oraInventory
# ./orainstRoot.sh
# cd /u01/app/oracle/product/11.1.0/db_1
# ./root.sh
```

Хорошим тоном считается установка от пользователя oracle, который предварительно добавлен в группы oinstall и dba (OSOPER и OSDBA соответственно).

Основными задачами администратора при установке являются:

- Обеспечение достаточного места и ресурсов для БД
- Установка необходимых переменных окружения
- Грамотное ведение процесса установки
- Устранение возможных ошибок (например, увеличение swap)

9. Архитектура хранения базы данных. Подход OFA.

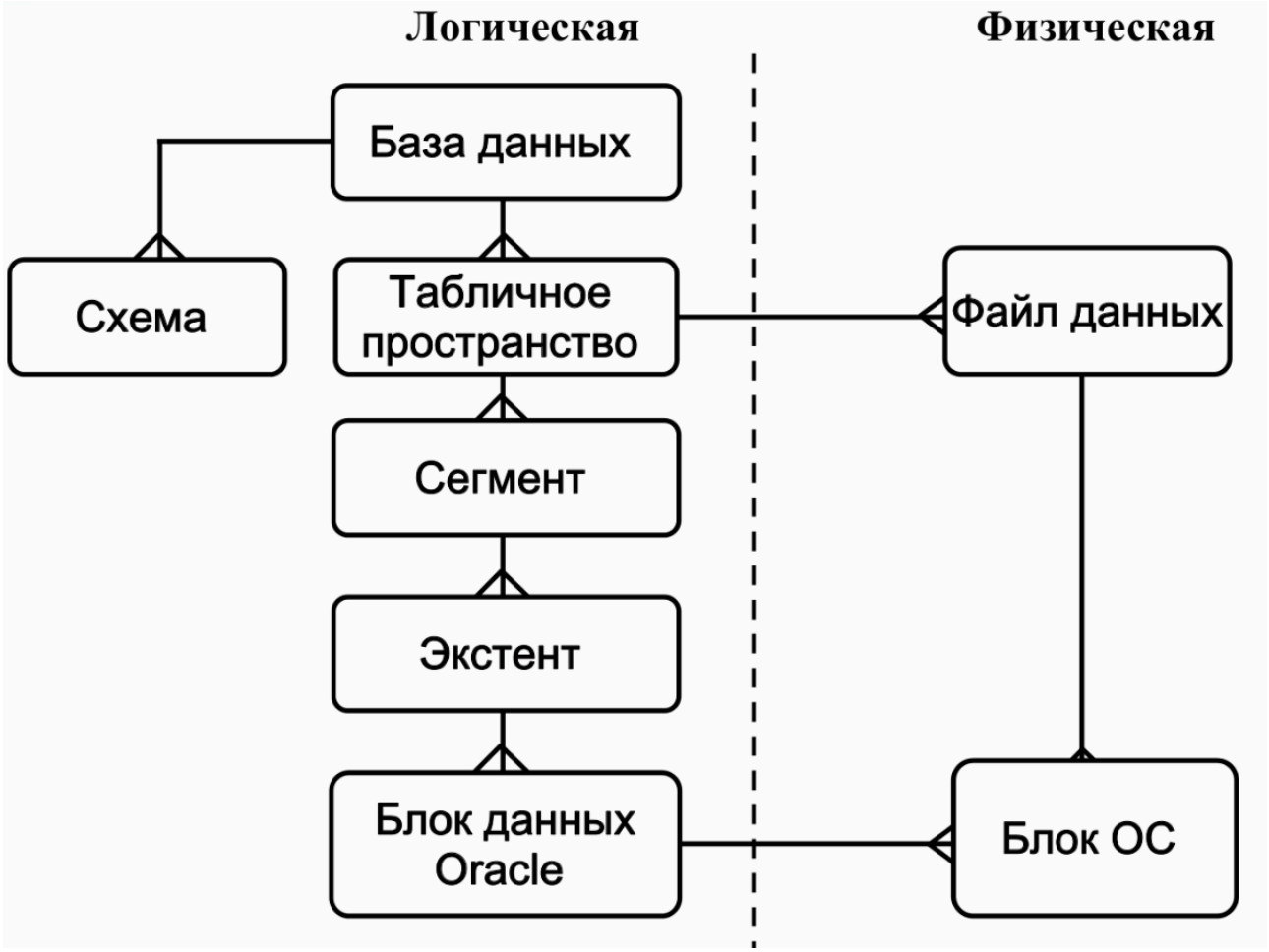
Файлы, из которых состоит БД, делятся на следующие категории:

- **Управляющие файлы.** Содержат данные о самой базе данных (то есть информацию о физической структуре базы данных).
- **Файлы данных.** Содержат данные пользователя или приложения БД, а также метаданные и словарь данных.
- **Оперативные файлы журнала повторов.** Если происходит сбой сервера БД, при котором не теряются файлы данных, экземпляр позволит восстановить базу данных с помощью информации, содержащейся в этих файлах.

Помимо перечисленных, экземпляр БД использует следующие файлы:

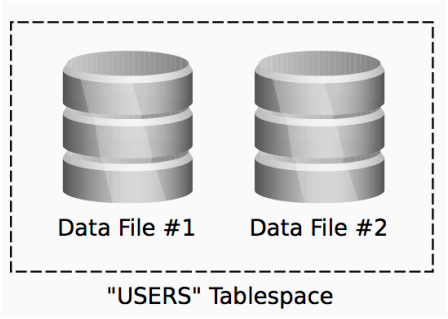
- **Файл параметров**. Используется для определения конфигурации экземпляра для запуска.
- **Файл паролей.** Позволяет удаленно подключаться к БД пользователям sysdba, sysoper и sysasm.
- **Резервные копии файлов**. Используются для восстановления БД.
- **Архивные файлы журнала повторов.** Содержат непрерывную историю изменений данных (повторных операций), которую создает экземпляр. С помощью этих файлов и резервной копии БД можно восстановить утраченные файлы данных.
- **Файлы трассировки.** Любой серверный или фоновый процесс может выполнять запись в определенный файл трассировки. При обнаружении процессом внутренней ошибки он записывает дамп информации об ошибке в свой файл трассировки.
- **Файлы журнала предупреждений.** Журнал предупреждений БД – это хронологический журнал сообщений и ошибок. Каждый экземпляр использует один файл журнала предупреждений.

Логические и физические структуры БД



Табличные пространства и файлы данных

Каждая БД логически состоит из одного или более табличных пространств, являющихся логическими блоками хранения.
Табличные пространства состоят из одного или нескольких файлов данных.
Файл данных может принадлежать только одному табличному пространству.
Табличное пространство может находиться в оперативном (доступном) и автономном (недоступном) режимах.



Табличные пространства SYSTEM и SYSAUX

Любая БД Oracle должна содержать табличные пространства SYSTEM и SYSAUX. Они автоматически создаются вместе с БД и всегда должны находиться в оперативном режиме.

- В табличном пространстве SYSTEM хранятся таблицы, обеспечивающие основные функции базы данных, например, таблицы словаря данных.
- Табличное пространство SYSAUX является вспомогательным. В нем хранится множество компонентов БД, например, репозиторий Enterprise Manager.

- Сегменты существуют в табличном пространстве.
- Сегмент – это набор экстентов.
- Экстент – это набор блоков данных.
- Блоки данных связаны с дисковыми блоками.

Optimal Flexible Architecture(OFA)

Универсальный подход (шаблон) к конфигурированию СУБД (не только Oracle).
Описывает структуру каталогов БД и других ресурсов в ФС.
Как и остальные шаблоны, предназначен для построения максимально гибкой структуры экземпляра БД и избежания возможных «типовых» проблем.
Не является обязательным.

Имена каталогов в соответствии с OFA:

- Точки монтирования — /pm (например, /u01 или /u02).
- Домашние каталоги — /pm/h/u (например, /u01/app/oracle или /u02/home/oracle).
- Каталоги бинарных файлов СУБД — /pm/h/u/product/v (например, /u01/app/oracle/product/11.1.0/db_1).
- Каталоги с конфигурационными и другими административными файлами — /pm/h/u/admin/d/a (например, /u01/app/oracle/admin/db01/arch).
- Файлы БД:
  - Управляющие файлы: /pm/q/d/controln.ctl.
  - Файлы журнала повторов: /pm/q/d/redon.log.
  - Файлы данных: /pm/q/d/tn.dbf.

10. Параметры инициализации экземпляра БД. Файлы параметров.

Считываются экземпляром Oracle при его запуске.

Существует 2 типа файлов параметров:

- Файлы параметров сервера (SPFILE) — двоичный файл, чтение и запись в который осуществляет сервер БД. Самостоятельно изменять этот файл нельзя. Имя по умолчанию — spfile<SID>.ora.
- Текстовый файл параметров инициализации — может быть только считан сервером, но не записан. Настройки параметров инициализации необходимо задавать и изменять вручную (с помощью текстового редактора). Имя по умолчанию — init<SID>.ora. При наличии SPFILE этот файл игнорируется.

Создание бинарного файла параметров инициализации сервера на основании текстового:

SQL> CREATE SPFILE FROM PFILE;

Текстовый файл с параметрами инициализации БД должен содержать как минимум 3 параметра:

- DB_NAME — имя БД (максимум 8 символов).
- CONTROL_FILES — список управляющих файлов БД.
- MEMORY_TARGET — общее количество памяти, которое будет выделено экземпляру БД.

Существует два типа файлов параметров инициализации:

- Статические параметры — могут быть изменены только путем редактирования файлов init.ora и SPFILE. Чтобы изменения вступили в силу, необходимо остановить и перезапустить БД.
- Динамические параметры — могут изменяться при работающей БД. Существует два типа динамических параметров:
  - Параметры уровня сеанса — оказывают влияние только на текущий сеанс. Пример — параметры поддержки национальных языков (NLS).
  - Параметры уровня системы — оказывают влияние на всю базу данных и все сеансы. Пример — параметры, отвечающие за сброс данных разделяемого пула или параметры расположения архивного журнала. Действие данных параметров зависит от настройки SCOPE.x.

Динамические параметры можно изменять с помощью команд ALTER SESSION и ALTER SYSTEM:

SQL> ALTER SESSION SET NLS_DATE_FORMAT ='mon dd yyyy’;

Некоторые дополнительные параметры инициализации БД:

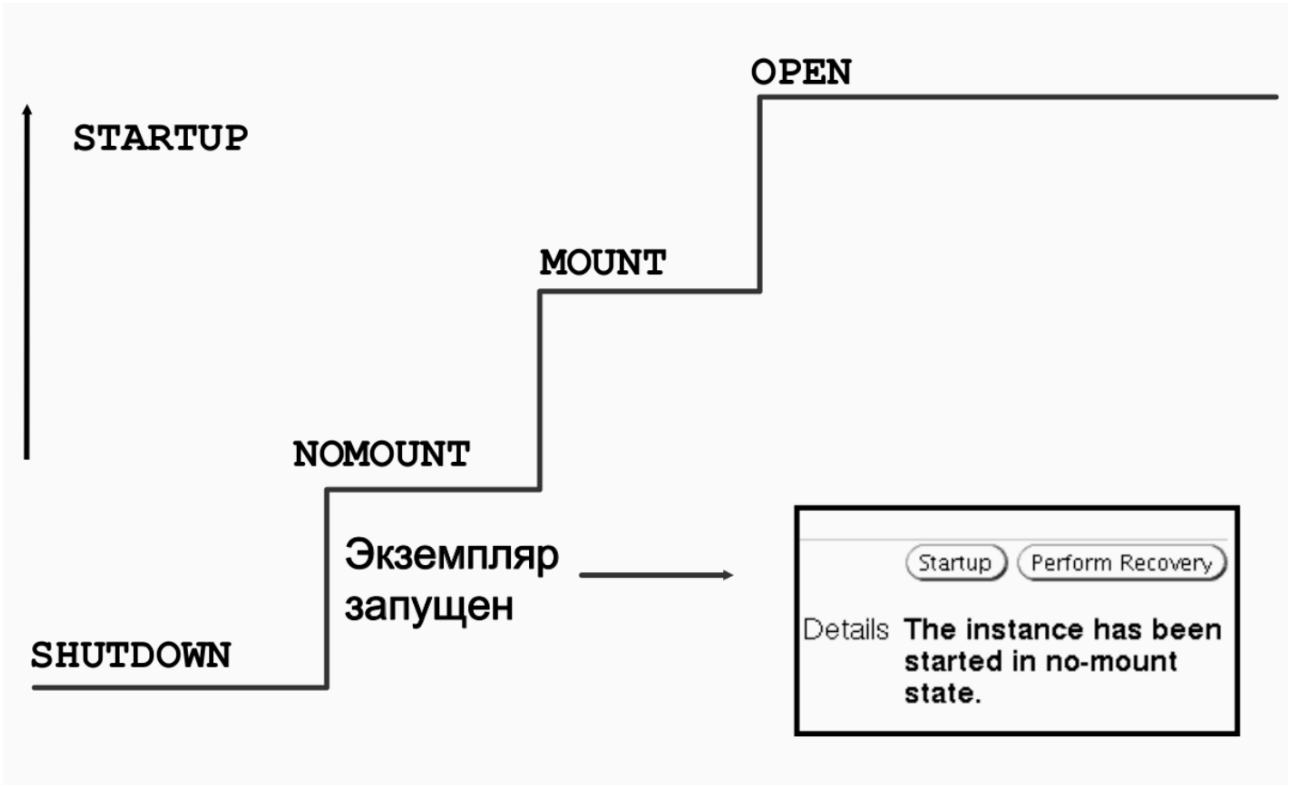
- DB_FILES — максимальное количество файлов БД.
- PROCESSES — максимальное количество параллельных
- пользовательских процессов.
- DB_BLOCK_SIZE — размер блока данных БД (в байтах; по умолчанию — 8 КБ).
- DB_CACHE_SIZE — размер блока буферного кэша БД (в байтах; по умолчанию — 48 МБ для однопроцессорной системы).
- SGA_TARGET — общий размер SGA (в байтах).

Пример файла параметров инициализации:

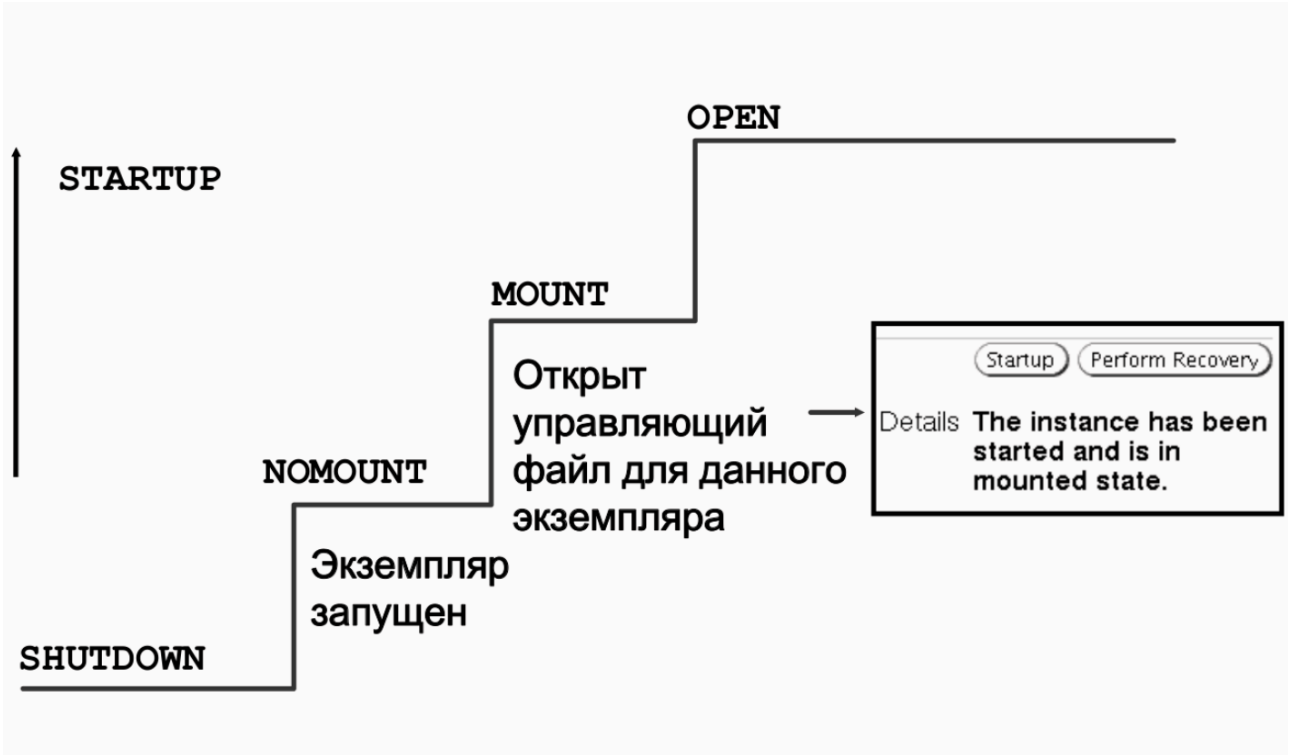
```
db_name='ORCL'
memory_target=1G
processes = 150
db_block_size=8192
db_domain=cs.ifmo.ru
db_recovery_file_dest='<ORACLE_BASE>/flash_recovery_area'
db_recovery_file_dest_size=2G
diagnostic_dest='<ORACLE_BASE>'
dispatchers='(PROTOCOL=TCP) (SERVICE=ORCLXDB)'
open_cursors=300
remote_login_passwordfile='EXCLUSIVE'
undo_tablespace='UNDOTBS1'
# You may want to ensure that control files are created on
separate physical
# devices
control_files = (ora_control1, ora_control2)
compatible =’12.0.0'
```

11. Запуск и остановка экземпляра БД. Режимы запуска и остановки.

NOMOUNT



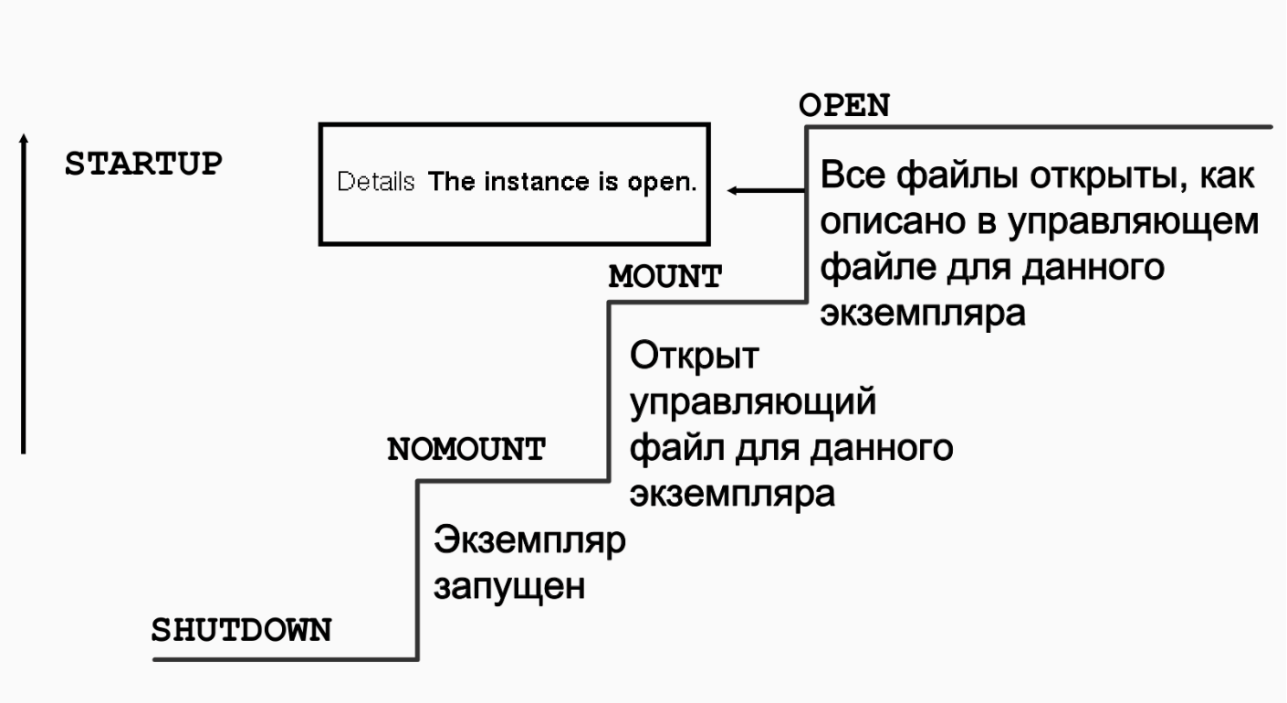
MOUNT



При монтировании БД выполняются следующие операции:

- БД ассоциируется с ранее запущенным экземпляром.
- Выполняется поиск и открытие управляющих файлов, указанных в файле параметров.

- Производится чтение управляющих файлов для получения сведений об именах и состоянии файлов данных и интерактивных файлах журналов повтора.



OPEN
При открытии БД выполняются следующие операции:
- Открытие файлов данных.
- Открытие оперативных файлов журнала повторов.

SQL> startup
Запускает экземпляр, ассоциирует с ним файлы БД, а затем монтирует и открывает БД.
SQL> startup nomount
Запускает экземпляр без монтирования БД.
SQL> alter database mount;
Монтирует и открывает БД, находящуюся в состоянии NOMOUNT.

2 способа:

- Через веб-интерфейс Enterprise Manager.
- «Вручную» — с помощью SQL*Plus:

```
SQL> shutdown
SQL> shutdown transactional
SQL> shutdown immediate
SQL> shutdown abort
```

Режим остановки	A	I	T	N
Разрешены новые подключения	Нет	Нет	Нет	Нет
Ожидание завершения текущего сеанса	Нет	Нет	Нет	Да
Ожидание завершения текущей транзакции	Нет	Нет	Да	Да
Принудительное создание контрольной точки и закрытие файлов	Нет	Да	Да	Да

Режимы остановки:

- **A = ABORT**
- **I = IMMEDIATE**
- **T = TRANSACTIONAL**
- **N = NORMAL**

12. Словарь данных и динамические представления V$ и GV$.

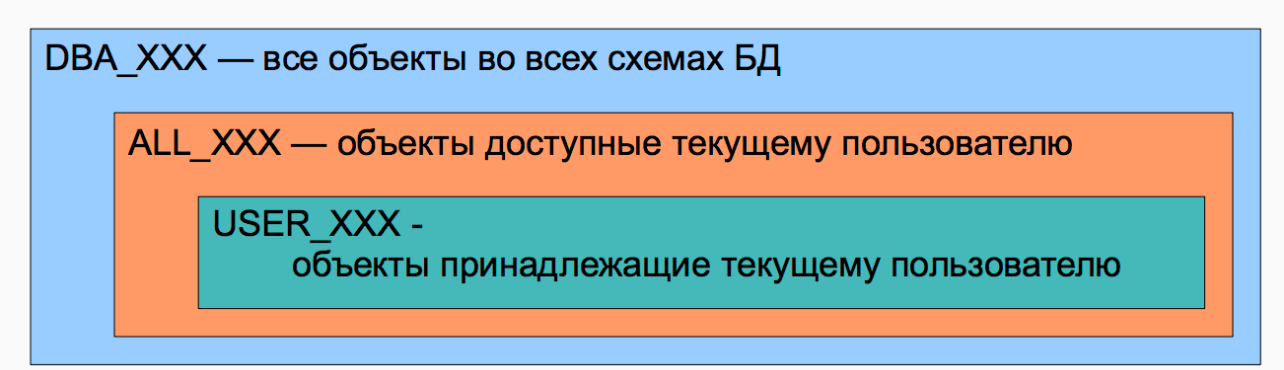
Словарь данных (Data Dictionary, DD) — это набор доступных только для чтения таблиц и представлений, которые содержат различную информацию о БД:

- Информацию обо всех объектах схемы БД (таблицах, представлениях, индексах, кластерах, синонимах, последовательнастях, процедурах, функциях, пакетах, триггерах и т. д.).
- Информацию о том, сколько дискового пространства выделено объектам схемы, и какой процент этого пространства уже использован.
- Информацию о значениях полей таблиц по умолчанию. Информацию о существующих в БД ограничениях целостности.
- Сведения обо всех пользователях БД, их ролях и выданных привилегиях.
- Результаты текущего аудита — кто в данный момент имеет, обращается и/или модифицирует объекты схемы.

Словарь данных состоит из двух «уровней»:

- Системные таблицы — содержат информацию в логичном с точки зрения архитектуры системы виде.
- Представления — содержат ту же самую информацию в более удобном для чтения и обработки формате.

Каждое представление имеет префикс, характеризующий, что за информация в нём содержится:



Популярные представления:

- CAT, USER_CATALOG, ALL_CATALOG, DBA_CATALOG. TAB, TABS, USER_TABLES, ALL_TABLES, DBA_TABLES.
- DICT, DICTIONARY — описание таблиц и представлений словаря.
- IND, USER_INDEXES, ALL_INDEXES, DBA_INDEXES.
- XXX_SNAPSHOTS, XXX_DB_LINKS, XXX_CLUSTERS, XXX_ERRORS, XXX_SOURCES, XXX_DEPENDENCIES...
- Подробный список http://citforum.ru/database/oraclepr/oraclepr_15.shtml

Таблица DUAL:

Таблица в словаре данных, которая состоит из одного столбца с именем DUMMY и одной строки со значением x.

Используется в случаях, когда нужно проверить работоспособность БД — результат запроса к этой таблице всегда заранее известен:

```
SQL> desc dual
Name                          Null?    Type
----------------------------- -------- -----------------
DUMMY                                  VARCHAR2(1)
SQL> select * from dual;
D
-
X
```

Также может использоваться в качестве таблицы-«заглушки», т. к. она всегда существует:

```
SQL> select user from dual;
USER
------------------------------
SCOTT
```

Dynamic Performance Views:
Представление производительности:

- Отражают текущее состояние экземпляра БД.
- Информация динамически генерируется в зависимости от состояния.
- Префикс V$ - представления производительности экземпляра БД.
- Префикс GV$ - глобальные представления узлов кластера RAC.
- Подробный список: http://citforum.ru/database/oraclepr/oraclepr_15.shtml

Полезные представления:

- V$SYSTEM_EVENT — содержит общесистемную информацию о ресурсах, которых ждет весь экземпляр.
- V$SESSION_EVENT — список событий, которые приходилось ждать в каждом сеансе.
- V$SESSION_WAIT — детальная посеансовая информация о ресурсах, которые сеанс ожидает в данный момент или ждал в последний раз.
- V$SESSION — информация о сеансе, в том числе о событии, которое ожидает сеанс в данный момент или ждал в последний раз
- Подробный список: http://citforum.ru/database/oraclepr/oraclepr_15.shtml

13. Методы разрешения имен, настройка псевдонимов.

После создания базы данных и различных ее объектов, а также загрузки необходимых данных следующей серьезной задачей, которую потребуется выполнить, будет установка подключения между сервером БД и пользователями, которые будут ею пользоваться. Oracle Net Services (Сетевые службы Oracle) — набор служб, которые делают это возможным. Компоненты Oracle Net Services должны существовать как на клиенте, так и на сервере, и, как правило, они используют сетевой протокол TCP/IP для установки сетевых подключений между клиентами и сервером базы данных.

Oracle Net поддерживает несколько методов разрешения информации о соединении:

- Разрешение имен Easy Connect: используется строка соединения TCP/IP.
- Локальное разрешение имен: используется локальный файл конфигурации.
- Разрешение имен на сервере каталогов: используется центральный сервер каталогов с поддержкой LDAP.
- Внешнее разрешение имен: используется внешняя служба разрешения имен.

Подробнее - гл. 5 учебника:

Oracle Net поддерживает следующие методы разрешения имен:

- Разрешение имен Easy Connect: при данном методе клиенты могут соединяться с сервером БД Oracle с помощью строки соединения TCP/IP, которая состоит из имени хоста, после которого могут быть также указаны порт и имя службы:
  Данный метод разрешения имен не требует настройки.
- Локальное разрешение имен: при данном методе дескрипторы соединения (различаемые по именам сетевых служб) хранятся на стороне клиента в локальном файле конфигурации tnsnames.ora.
- Разрешение имен на сервере каталогов: для доступа к службе БД данный метод сохраняет идентификаторы соединения для обращения к службе БД на центральном сервере каталогов, который поддерживает протокол облегченного доступа к каталогам (LDAP);
- Внешнее разрешение имен: при данном методе имена служб хранятся в поддерживаемой сторонней службе разрешения имен. Поддерживаемые сторонние службы:
  - Служба внешнего разрешения имен Network Information Service (NIS);
  - Distributed Computing Environment (DCE) Cell Directory Services (CDS).

Ну хз:

Настройка псевдонимов служб

Чтобы создать локальный псевдоним службы Oracle Net, в раскрывающемся списке «Administer» (Администрирование) выберите пункт «Local Naming» (Локальное разрешение имен) и нажмите кнопку «Go» (Перейти). Затем нажмите кнопку «Create» (Создать).

Чтобы настроить псевдонимы служб для разрешения имен в сервере каталогов, выберите пункт «Directory Naming» (Разрешение имен в сервере каталогов), а не «Local Naming» (Локальное разрешение имен).

Примечание. Если разрешение имен на сервере каталогов не настроено, пункт «Directory Naming» (Разрешение имен в сервере каталогов) выбрать невозможно. Разрешение имен на сервере каталогов рассматривается в курсе Управление идентификаторами с помощью Oracle Enterprise, а также в руководстве Oracle Advanced Security Administration.

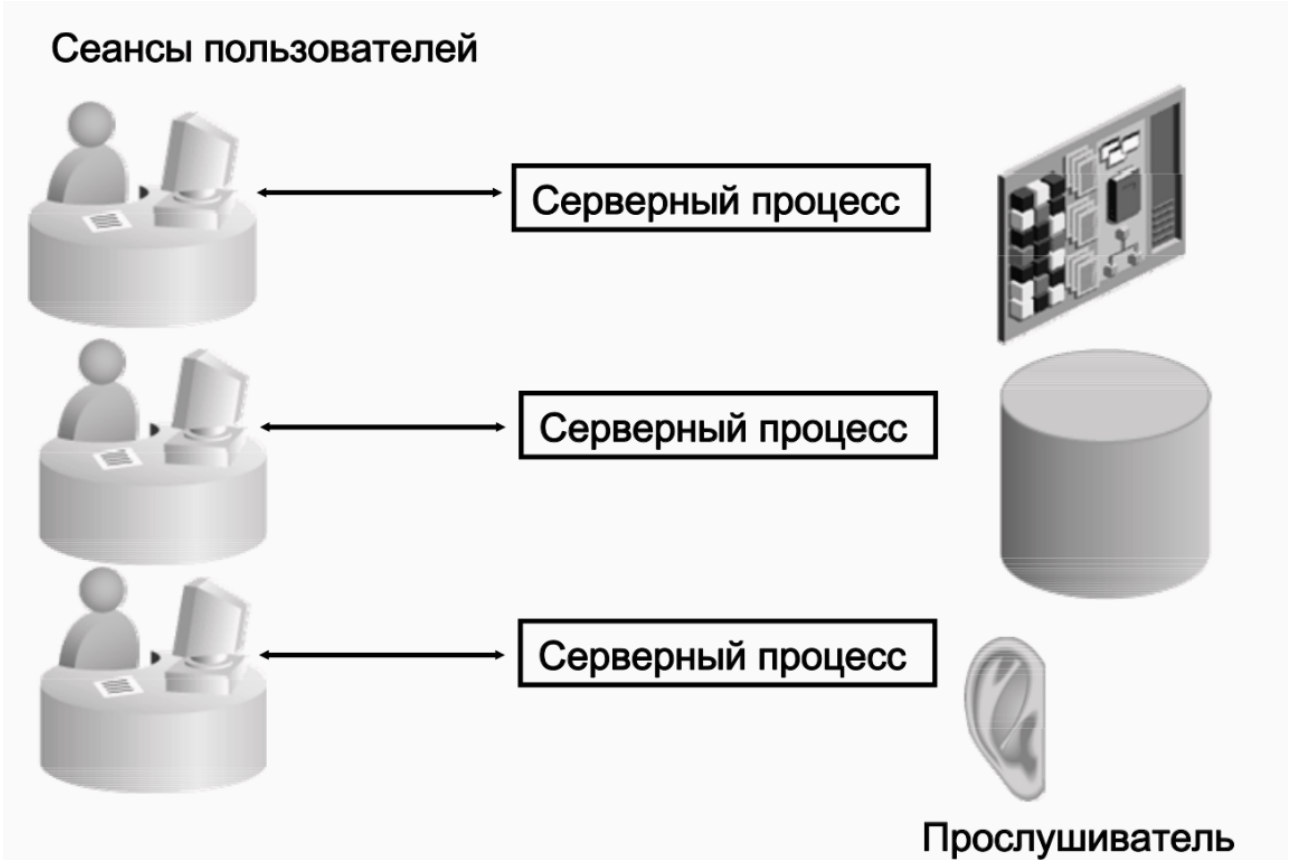
На странице создания имени сетевой службы в поле «Net Service Name» (Имя службы) введите уникальное имя. (Это имя будут вводить пользователи.) Введите имя службы или системный идентификатор (SID) базы данных, с которой устанавливается соединение, затем нажмите кнопку «Add» (Добавить), чтобы добавить адрес для имени службы.

Для введенного адреса укажите протокол, порт и имя хоста, используемого прослушивателем.

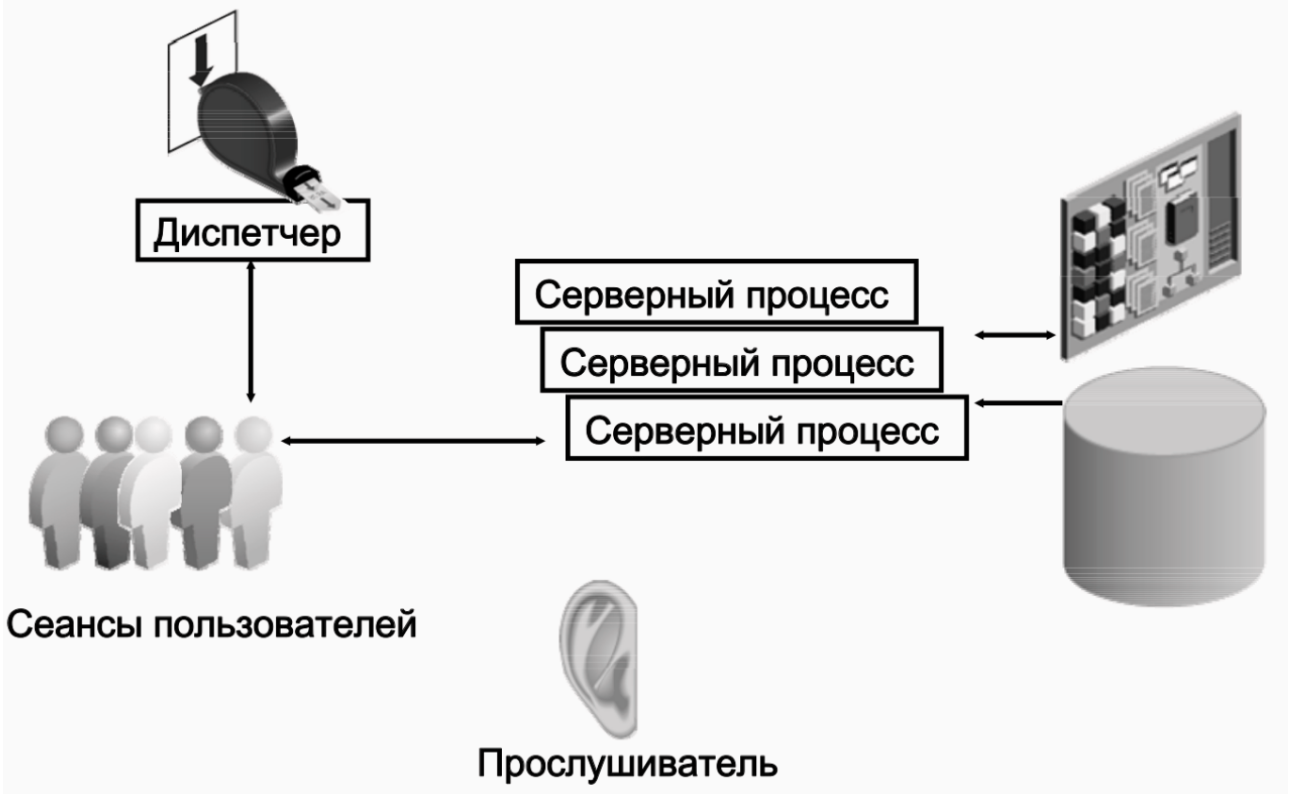
14. Выделенный и разделяемый режим работы сервера. Преимущества и недостатки.

Выделенный сервер

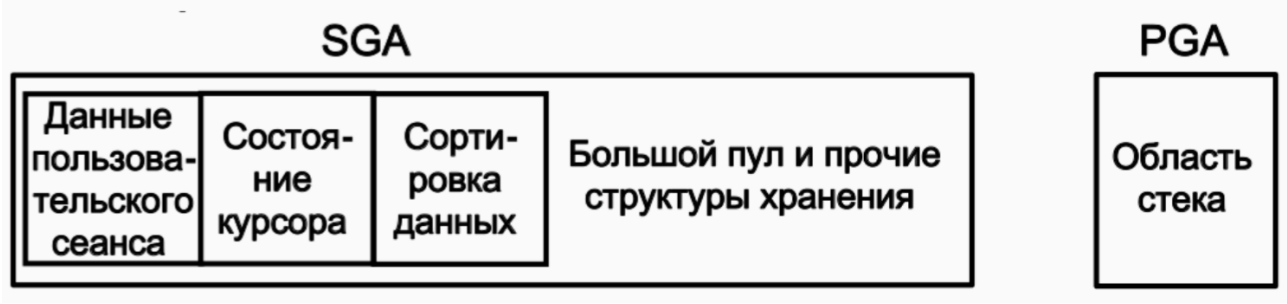
Главный недостаток — плохая масштабируемость.



Разделяемые серверы



Для каждой службы в архитектуре разделяемого сервера используется как минимум один процесс диспетчера (обычно больше).
Слушатель хранит список доступных диспетчеров для каждого имени службы, а также информацию о загрузке соединения (количество одновременных соединений) для каждого диспетчера.
Запрос перенаправляется к наименее загруженному диспетчеру, обслуживающему службу с запрошенным именем.
Во время сеанса пользователь поддерживает соединение с одним и тем же диспетчером (но запросы могут обрабатывать разные серверные процессы).
Один диспетчер может обслуживать сотни сеансов пользователей.
Диспетчеры направляют запросы пользователей в общую очередь, которая размещена в области SGA, выделенной для разделяемого пула.
Т.к. запросы одного пользовательского процесса могут обрабатывать разные серверные, большая часть данных из PGA переносится в SGA:

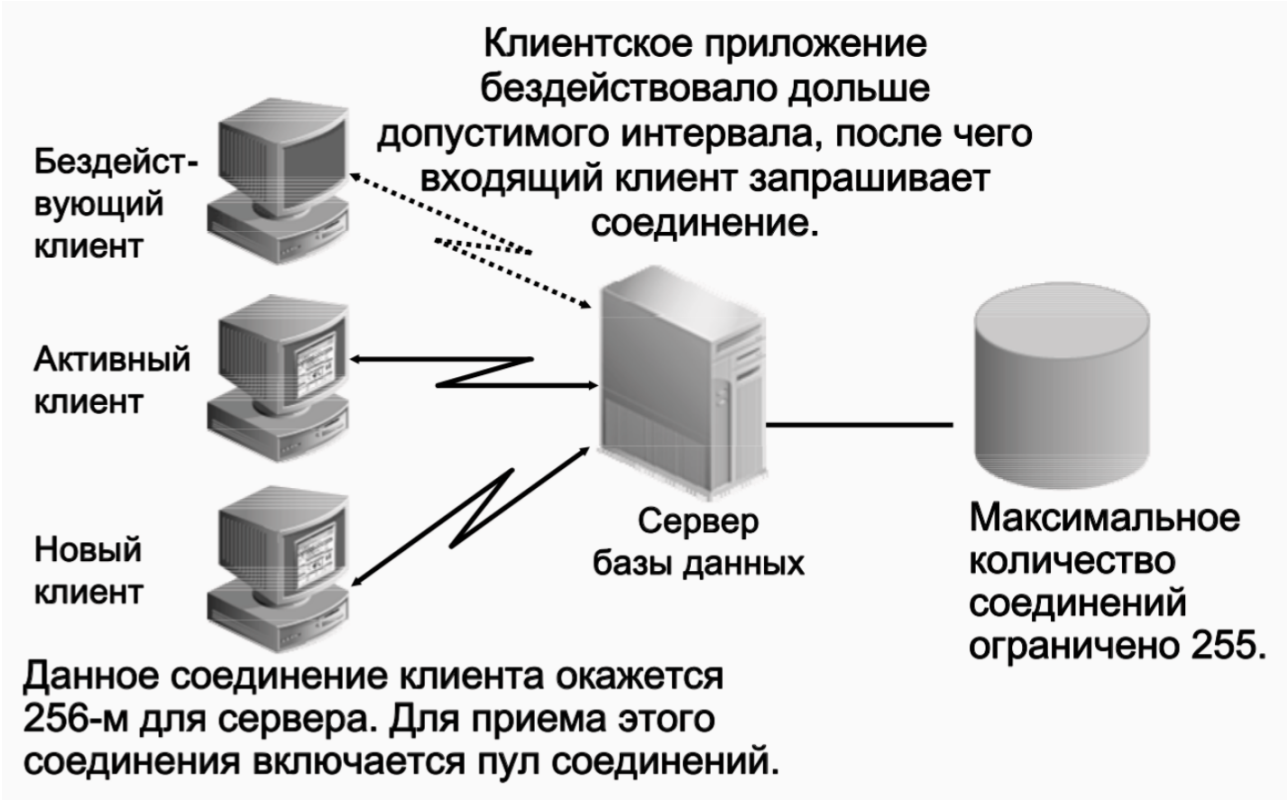


Недостатки: это нужно учитывать при конфигурации размера SGA.
Определённые операции с БД не стоит выполнять с помощью разделяемых серверов:

- Администрирование БД.
- Операции резервного копирования и восстановления.
- Пакетную обработку и операции с массовой загрузкой.
- Операции с хранилищами данных.

Преимущество: хорошая масштабируемость.

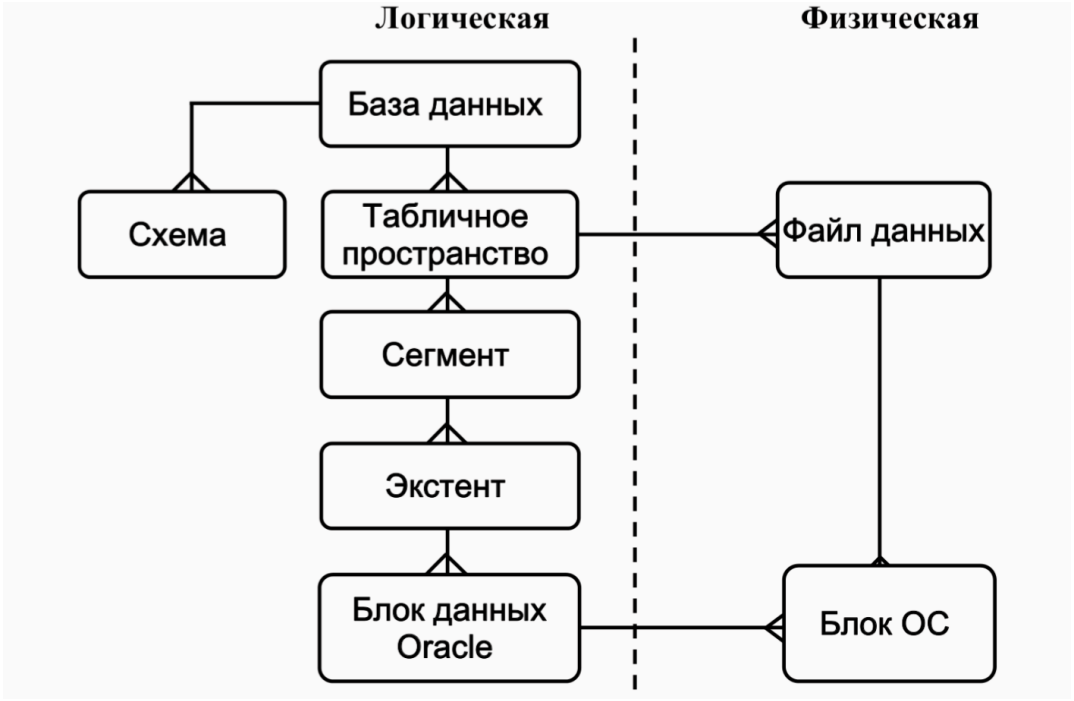
Пул соединений:



В этом примере сервер базы данных Oracle был сконфигурирован на 255 соединений. Один из клиентов был неактивным требуемое количество времени. Создание пула соединений делает это соединение доступным для входящего клиентского соединения, которое является 256-м соединением. Когда у неактивного клиента появляется работа, которую необходимо выполнить, соединение для этого клиента восстанавливается посредством другого неактивного соединения другого клиента.

15. Логические и физические структуры хранения. Представление табличных данных, блок базы данных.

Логическая и физическая структуры хранения БД



Размер блока данных

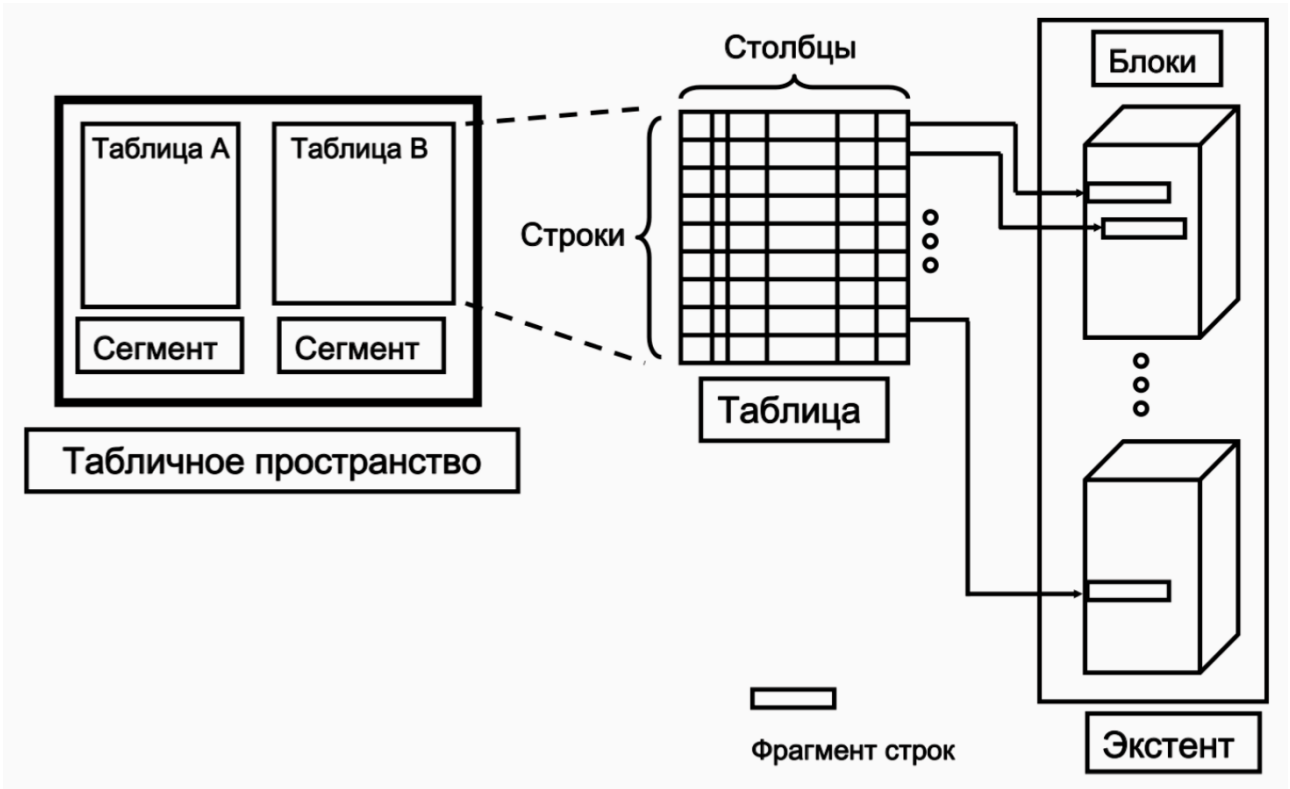
Определяется параметром инициализации DB_BLOCK_SIZE.

Размер—от2до32КБ.

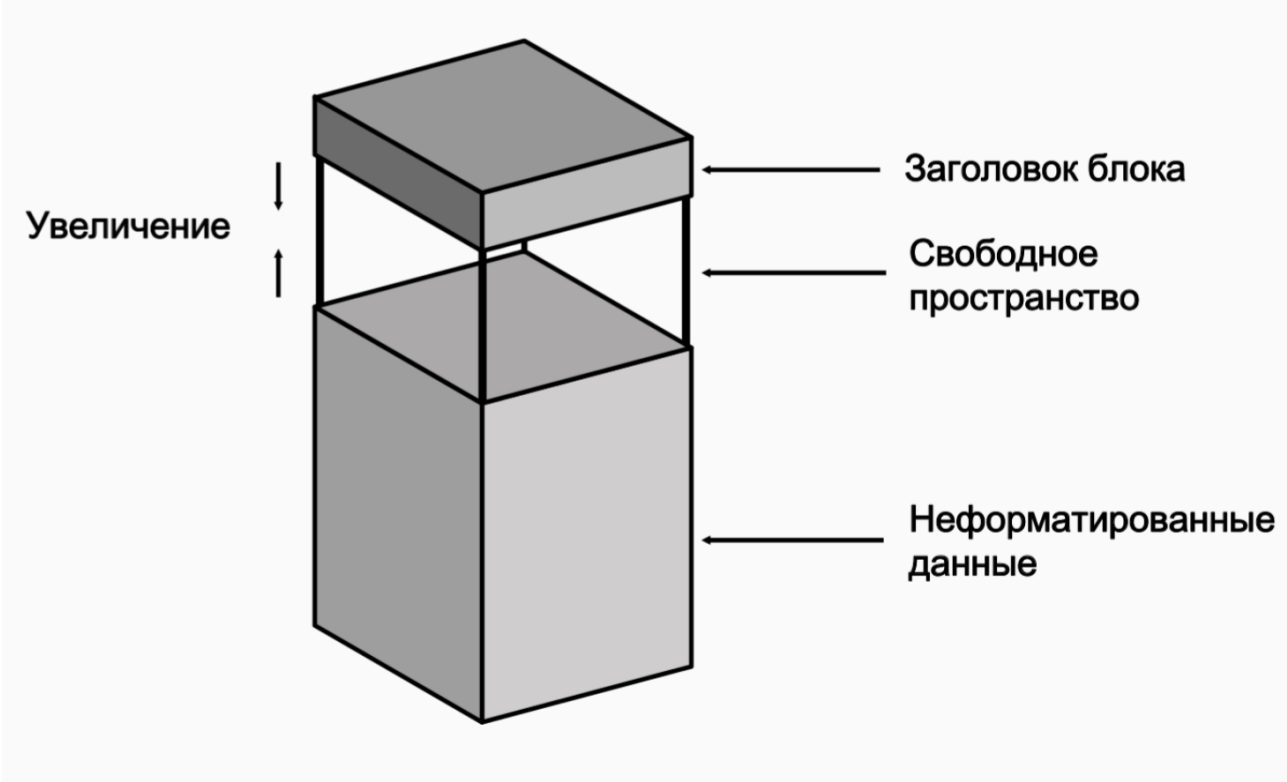
По умолчанию — 8 КБ.

Изменить размер блока данных можно только путём повторного создания БД.

Хранение табличных данных



Содержимое блока данных



Заголовок блока — содержит сведения о типе сегмента (табличный или индексный), адресе блока данных, каталоге таблицы, каталоге строки и транзакционных слотах размером по 24 байта каждый, которые используются при внесении изменений в строки блока. Заполняется сверху вниз.
Неформатированные данные — данные для строк блока. Заполняются снизу вверх.
Свободное пространство внутри блока, за счет которого могут увеличиваться заголовок и пространство неформатированных данных.
События, которые могут увеличить размер заголовка:

- Требуется больше неформатированных записей.
- Требуется большее количестве транзакционных слотов.

Изначально свободное пространство является непрерывным, однако в ходе выполнения операций удаления и обновления может стать фрагментированным.
Свободное пространство может быть дефрагментировано сервером Oracle.

16. Табличные пространства и файлы данных. Управление табличными пространствами.

Табличные пространства:
БД состоит из одного или нескольких логических табличных пространств.
Каждое табличное пространство базы данных Oracle состоит из одного или нескольких файлов данных.
БД должна содержать минимум два табличных пространства — SYSTEM и SYSAUX, каждое из которых представлено минимум одним файлом данных.
Одна БД может иметь до 65534 файлов данных.
Если табличное пространство в течение всего жизненного цикла представлено одним (и только одним) файлом данных, оно называется табличным пространством типа BIGFILE.
Временный файл – это файл, который принадлежит временному табличному пространству:
Создается с параметром TEMPFILE:

```
alter tablespace temp
add tempfile 'c:\oracle\oradata\temp3\temp02.dbf' size 50m
reuse
autoextend on
next 1m
maxsize 500m;
```

Временные табличные пространства используются для операций сортировки и не могут содержать постоянных объектов базы данных, таких как таблицы.
Управление табличными пространствами:
Управляемое локально:

- управление свободными экстентами осуществляется в табличном пространстве;
- для записи свободных экстентов используется битовый образ;
- каждый бит соответствует блоку или группе блоков;
- значение бита описывает экстент: свободен или занят.

Управляемое словарём:

- свободными экстентами управляет Oracle через словарь данных;
- при выделении и освобождении экстентов обновляются соответствующие представления

Управляемые локально:
Экстенты могут выделяться одним из двух способов:

- Автоматически — размерами экстентов управляет система (он всегда кратен 64 КБ). Способ неприменим ко временным табличным пространствам.
- Унифицированно — табличные пространства используют унифицированный размер экстентов указываемый администратором (по умолчанию — 1 МБ). Режим нельзя использовать для табличных пространств отмены операций.

Управление пространством сегментов можно осуществлять:

- Автоматически — используются битовые образы. Битовый образ описывает состояние каждого блока данных в сегменте в соответствии с объемом пространства блока, которое доступно для вставки строк.
- Вручную — используются списки свободных сегментов (списки блоков данных, в которых имеется свободное пространство). Требуется вручную задавать и настраивать значения параметров хранения PCTUSED, FREELISTS и FREELIST GROUPS для объектов схемы.

17. Пользователи БД, учетные записи пользователей. Системные учетные записи.

Формуляр (account) пользователя БД — это способ организации принадлежности и доступа к объектам БД.

Пароль необходим для аутентификации в БД Oracle.

У каждого пользователя БД есть свой уникальный формуляр БД. У каждого формуляра пользователя есть:

- Уникальное имя пользователя. Не может превышать 30 байт, не может содержать специальные символы, должно начинаться с буквы.
- Метод аутентификации. По умолчанию — пароль.
- Табличное пространство по умолчанию.
- Временное табличное пространство.
- Профиль пользователя. Набор ресурсов и ограничений с помощью паролей, присвоенных пользователю.
- Состояние формуляра. Пользователям доступны только «открытые» формуляры.

Системные учетные записи(Предопределенные формуляры SYS и SYSTEM):

Формуляр SYS:

- получает роль администратора БД;
- обладает всеми полномочиями с параметром ADMIN OPTION;
- необходим для запуска, остановки и выполнения некоторых служебных команд;
- является владельцем словаря данных;
- является владельцем репозитория автоматической рабочей нагрузки (AWR).

Формуляру SYSTEM предоставляется роль администратора БД.

Оба этих формуляра не должны использоваться для стандартных операций.

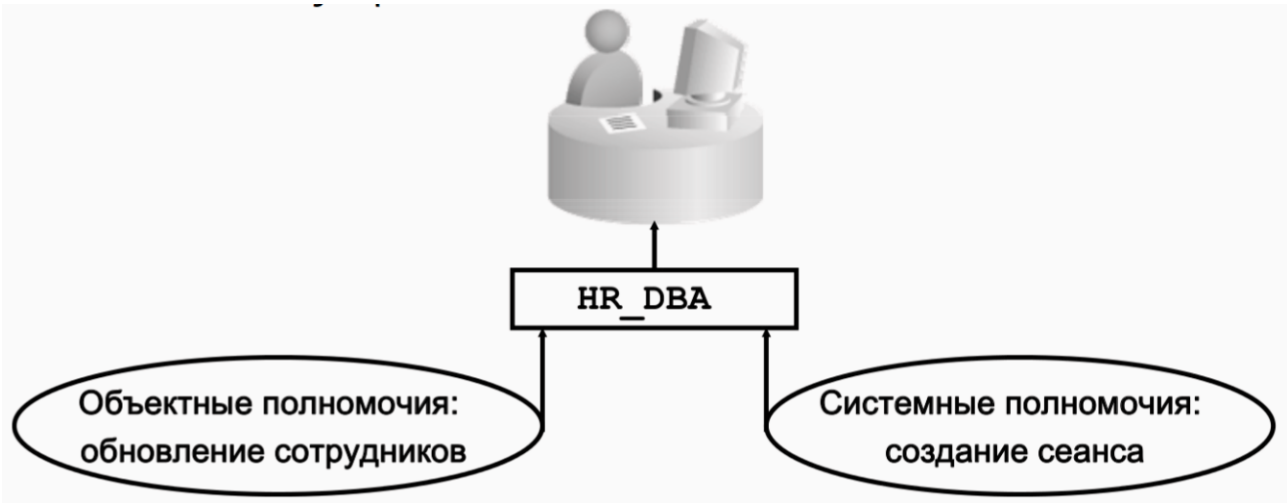
18. Системные и объектные полномочия. Назначение и удаление полномочий.

Полномочие (privilege) — это право на выполнение определенного типа SQL- оператора или на доступ к объекту пользователя.

Существует два типа полномочий пользователя:

Системные: позволяют пользователям выполнять определенные действия с базой данных.

Объектные: позволяют пользователям получать доступ к определенным объектам и манипулировать ими.



Системные полномочия:

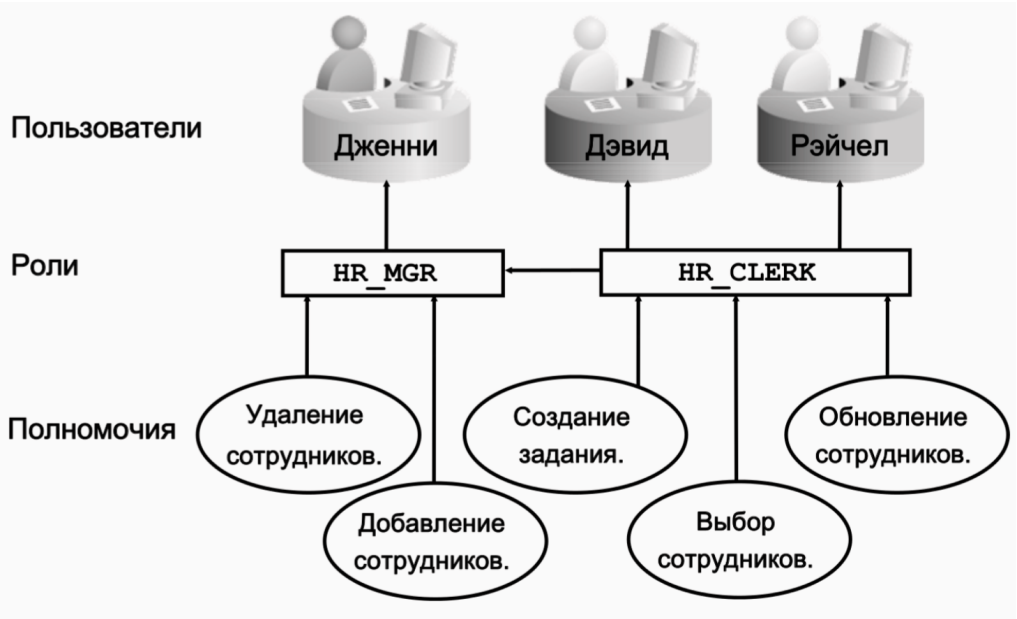
- Предоставление полномочий, которые содержат фразу ANY, означает выход за пределы схемы. Например, обладая полномочиями CREATE TABLE, можно создать таблицу, однако только в собственной схеме. А полномочия SELECT ANY TABLE позволяют выбирать среди таблиц других пользователей.
- Пользователь SYS и пользователи, владеющие ролью DBA, обладают всеми полномочиями ANY.
- SYSDBA и SYSOPER. Эти полномочия позволяют выполнять административные задачи в БД. SYSOPER позволяет пользователю выполнять оперативные задачи, но без возможности просмотра пользовательских данных. В него входят следующие системные полномочия:
  - STARTUP и SHUTDOWN
  - CREATE SPFILE
  - ALTER DATABASE OPEN/MOUNT/BACKUP
  - ALTER DATABASE ARCHIVELOG
  - ALTER DATABASE RECOVER (только полное восстановление).
  - RESTRICTED SESSION
- Системное полномочие SYSDBA дополнительно дает разрешение на неполное восстановление и удаление базы данных.
- SYSASM. Позволяет запустить, остановить экземпляр ASM и управлять им. DROP ANY объект. Позволяет удалять объекты других пользователей схемы.
- CREATE, MANAGE, DROP и ALTER TABLESPACE. Позволяют управлять табличными пространствами.
- CREATE LIBRARY. В БД Oracle имеется возможность создавать и вызывать внешний код (например библиотеку C) с помощью PL/SQL. Такой библиотеке должно быть присвоено имя объектом LIBRARY в базе данных.
- CREATE ANY DIRECTORY. В качестве меры безопасности каталог ОС, в котором хранится код, необходимо связать с виртуальным объектом каталога Oracle. Владелец полномочия CREATE ANY DIRECTORY потенциально может вызвать небезопасные объекты кода.
- GRANT ANY OBJECT PRIVILEGE. Позволяет предоставлять разрешения для объектов, которыми его обладатель не владеет.

- ALTER DATABASE и ALTER SYSTEM. Позволяют изменять базу данных и экземпляр Oracle (например, можно переименовать файл данных или очистить кэш буфера).

Объектные полномочия:
GRANT privileges ON object TO user;
REVOKE privileges ON object FROM user;
Виды полномочий:
- SELECT — возможность выполнять выборку данных из таблицы.
- INSERT — возможность добавлять новые строки в таблицу.
- UPDATE — возможность изменять данные в таблице.
- DELETE — возможность удалять строки из таблицы.
- REFERENCES — возможность создавать ограничения целостности для выбранной таблицы.
- ALTER — возможность выполнять оператор ALTER TABLE применительно к выбранной таблице.
- INDEX — возможность создавать индексы на столбцы выбранной таблицы.
- ALL — все перечисленные выше полномочия.

19. Роли БД. Предопределенные роли. Профили пользователей.

- *Упрощенное управление полномочиями.* Роли используются для упрощенного управления полномочиями. Вместо того чтобы предоставлять нескольким пользователям один и тот же набор полномочий, можно предоставить эти полномочия роли, а затем предоставить эту роль пользователям.
- *Динамическое управление полномочиями.* Если полномочия, связанные с ролью, изменяются, все пользователи, которым предоставлена эта роль, немедленно автоматически получают обновленные полномочия.
- *Выбор доступных полномочий.* Роли можно включать и отключать, чтобы временно активировать или деактивировать полномочия. Таким образом можно контролировать полномочия пользователя в той или иной ситуации.



Предопределенные роли:

Роль	***Полномочия***
CONNECT	CREATE SESSION
RESOURCE	CREATE CLUSTER, CREATE INDEXTYPE, CREATE OPERATOR, CREATE PROCEDURE, CREATE SEQUENCE, CREATE TABLE, CREATE TRIGGER, CREATE TYPE
SCHEDULER_ADMIN	CREATE ANY JOB, CREATE EXTERNAL JOB, CREATE JOB, EXECUTE ANY CLASS, EXECUTE ANY PROGRAM, MANAGE SCHEDULER
DBA	Большая часть системных полномочий; несколько других ролей. Следует предоставлять только администраторам.
SELECT_CATALOG_ROLE	Системных полномочий нет; HS_ADMIN_ROLE и более 1700 объектных полномочий в словаре данных

Назначение ролей:
Используются те же самые операторы GRANT и REVOKE:

```
SQL> CREATE ROLE cust_serv_mgr;
SQL> GRANT cust_serv_clerk TO cust_serv_mgr;
SQL> GRANT insert, update ON customers TO cust_serv_mgr;
SQL> GRANT delete ON issue_track TO cust_serv_mgr;
SQL> GRANT cust_serv_mgr TO mary;
SQL> REVOKE insert ON customers FROM cust_serv_mgr;
SQL> REVOKE cust_serv_mgr FROM mary;
```

Профили:

- Профили (profiles) — это именованные наборы ограничений на использование ресурсов БД и экземпляра.
- Каждому пользователю назначается профиль, причем одновременно только один.
- Если при изменении профиля пользователь уже находился в системе, изменения вступят в силу со следующего сеанса работы пользователя.
- Профиль DEFAULT является базовым для всех остальных профилей.
- Профили не накладывают ограничения на использование ресурсов пользователями, если параметру инициализации RESOURCE_LIMIT не присвоено значение TRUE.
- Если параметру RESOURCE_LIMIT присвоено стандартное значение FALSE, ограничения на использование ресурсов, заданные профилем, игнорируются. Настройки пароля для профиля всегда применяются принудительно.

С помощью профилей администратор может контролировать следующие системные ресурсы:

● **Процессор**. Ресурсы процессора можно ограничивать на основе сеансов или вызовов:
  ● Ограничение процессор/сеанс, равное 1000, означает, что если сеанс потребляет более 10 секунд процессорного времени, то будет вызывана ошибку и выполнен выход:
    ORA-02392: exceeded session limit on CPU usage, you are being logged off
  ● При ограничении вызовов место всего сеанса пользователя контролируется процессорное время, потребляемое отдельными командами. Если команда пользователя превышает его, её выполнение прерывается и пользователь получает сообщение об ошибке:
    ORA-02393: exceeded call limit on CPU usage

● **Сеть/память.** Можно задать следующие параметры:
  ● Время соединения: указывает интервал времени в минутах, в течение которого пользователь может
  быть соединен до того, как он будет автоматически отключен;
  ● Время простоя: указывает интервал времени в минутах, в течение которого сеанс пользователя может бездействовать до того, как он будет автоматически отключен.
  ● Параллельные сеансы: определяет, сколько параллельных сеансов может быть создано с помощью формуляра пользователя БД.

● **Личная область SGA.** Ограничивает размер области, выделяемой в глобальной системной области (SGA) для сортировки, слияния растровых изображений и т.д.

● **Дисковый ввод/вывод.** Ограничивает объем данных, которые пользователь может считать за сеанс или за вызов.

● Кроме того, для профиля можно задать **смешанные ограничения**.

20. Целостность, ограничения, состояние ограничений(уч. 8).

Понимание целостности данных

Для недопущения введения в столбцы тех или иных значений можно использовать следующие ограничения:

• NOT NULL: по умолчанию все столбцы таблицы могут принимать неопределенные значения. Слово null означает отсутствие значения. Ограничение NOT NULL указывает, что столбец таблицы не должен содержать неопределенных значений. Например, можно определить ограничение NOT NULL, чтобы указать, что ввод значений в столбец LAST_NAME обязателен для всех строк таблицы EMPLOYEES.

• Ключ UNIQUE: ограничение целостности для ключа UNIQUE указывает, что все значения в столбце или наборе столбцов (ключе) должны быть уникальными, то есть ни одна пара строк не должна содержать одинаковых значений в этом столбце или наборе столбцов. Например, ограничение ключа UNIQUE определено для столбца DEPARTMENT_NAME в таблице DEPARTMENTS, чтобы запретить строки с одинаковыми названиями отделов. За исключением отдельных случаев это задается с помощью уникального индекса.

• PRIMARY KEY: любая таблица в БД может иметь не более одного ограничения PRIMARY KEY. Значения из группы одного или нескольких столбцов, для которых определено данное ограничение, образуют уникальный идентификатор строки. Фактически, все строки именуются по значению первичного ключа.

Реализация ограничения целостности PRIMARY KEY в сервере Oracle обеспечивает выполнение двух следующих условий:

• Ни одна пара строк таблицы не содержит одинаковых значений в заданном столбце или группе столбцов.

• Столбец первичных ключей не может содержать неопределенных значений. Таким образом, каждая строка такого столбца должна содержать значение.

Обычно ограничения PRIMARY KEY накладываются с помощью индексов. Ограничение первичного ключа, созданное для столбца DEPARTMENT_ID в таблице DEPARTMENTS, наложено посредством явного создания:

- уникального индекса для данного столбца;
- ограничения NOT NULL для данного столбца.

• Ссылочные ограничения целостности: разные таблицы в реляционной базе данных могут быть связаны общими столбцами, поэтому необходимо соблюдать правила отношения столбцов. Ссылочные правила целостности гарантируют сохранение этих отношений.

Ссылочные ограничения целостности накладывают следующее требование: для всех строк таблицы значение во внешнем ключе должно совпадать со значением в первичном ключе.

Например, внешний ключ определен для столбца DEPARTMENT_ID таблицы EMPLOYEES. Это гарантирует, что все значения в столбце совпадают со значениями в первичном ключе таблицы DEPARTMENTS. Следовательно, в столбце DEPARTMENT_ID таблицы EMPLOYEES отсутствуют неправильные номера отделов.

Еще один тип ссылочного ограничения целостности называется рефлексивным ограничением ссылочной целостности. При этом внешний ключ ссылается на родительский ключ в той же таблице.

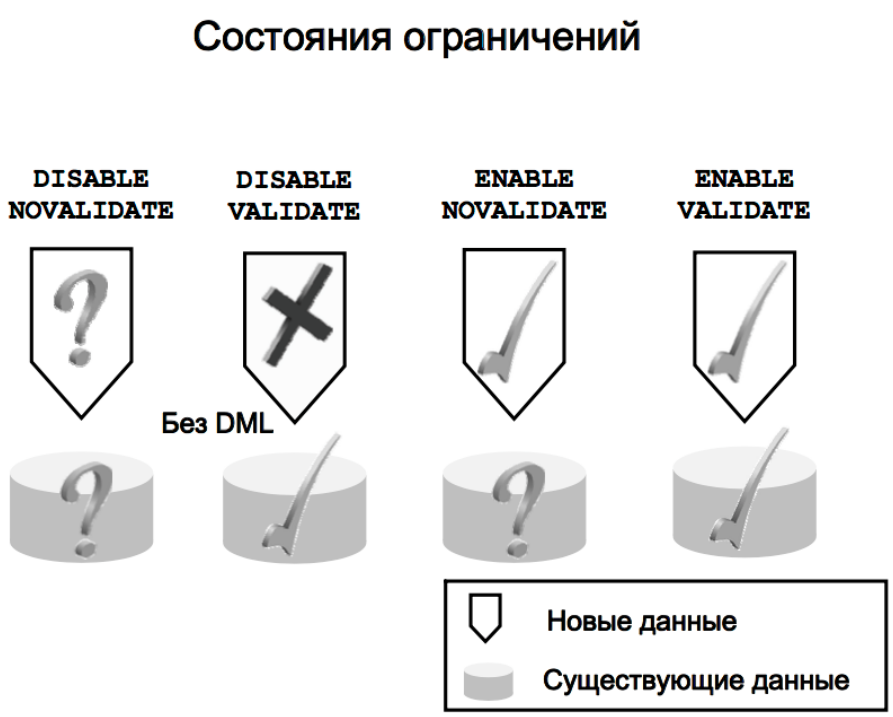
• Ограничения CHECK: ограничение целостности CHECK для столбца или набора столбцов указывает, что для каждой строки таблицы должно выполняться или иметь неизвестное значение какое-либо заданное условие.

Если при выполнении DML-оператора проверки условия ограничения получен результат FALSE, выполняется откат оператора, если наложено ограничение IMMEDIATE. Если ограничение является отложенным (DEFERRED), откат производится при фиксации, а не выполнении DML.

Нарушение ограничения происходит, если переданный DML-оператор не совместим с ограничением. Нарушения ограничения могут обуславливаться разными условиями. Вот некоторые из них:

• Неуникальность: попытка хранения одинаковых значений в столбце с ограничением UNIQUE (например, если столбец является первичным ключом или используется уникальный индекс);
• Целостность ссылочных данных: нарушение правила, согласно которому у каждой дочерней строки должна быть родительская строка;
• Проверка: попытка сохранить в столбце значение, противоречащее правилам, определенным для столбца. Например, для столбца AGE задано ограничение CHECK, согласно которому все значения столбца должны быть положительными.

Состояние ограничения



Чтобы проще разрешать ситуации, когда данные временно могут нарушать ограничение, можно использовать различные состояния ограничений. Ограничение целостности можно включить (ENABLE) или выключить (DISABLE).
Если ограничение включено, проверка данных происходит при вводе данных или обновлении БД. Данные, противоречащие правилам ограничения, не вводятся.
Если ограничение отключено, в базу данных могут вводиться противоречащие ограничению данные.
Ограничение целостности может находиться в одном из следующих состояний:
• DISABLE NOVALIDATE
• DISABLE VALIDATE
• ENABLE NOVALIDATE
• ENABLE VALIDATE

DISABLE NOVALIDATE: новые или уже существующие данные могут противоречить ограничению, если они не проверены. Этим часто пользуются при обработке данных из уже проверенного источника в режиме таблицы только для чтения, вследствие чего новые данные в таблицу не попадают. Состояние NOVALIDATE используется для хранилищ данных, в которых данные уже прошли проверку. Поэтому повторная проверка не нужна, что экономит время обработки.
DISABLE VALIDATE: при данном состоянии ограничения изменение столбцов, на которых оно наложено, запрещено, так как это нарушит целостность данных – уже существующие данные будут проверены, а новые будут вводиться без проверки. Это состояние часто

используют, когда существующие данные нужно проверить, но не изменить, а индекс для производительности не нужен.
ENABLE NOVALIDATE: новые данные соответствуют ограничению, однако уже существующие данные находятся в неизвестном состоянии. Это состояние часто используется, когда точно известно, что таблица содержит обработанные и проверенные данные, не требующие проверки. При этом все новые вводимые данные не должны противоречить ограничению.
ENABLE VALIDATE: ограничению не должны противоречить ни новые, ни существующие данные. Это самое распространенное состояние ограничения, используемое по умолчанию.

21. Понятие блокировки, уровни блокировки и операторы их использующие.

Блокировки

Перед тем как база данных позволит сеансу изменить данные, он должен заблокировать изменяемые данные. Блокировка дает сеансу монопольные права на работу с данными, благодаря чему другая транзакция не сможет изменить заблокированные данные до тех пор, пока блокировка не будет снята.

Транзакции могут блокировать отдельные строки, наборы строк или даже целые таблицы. База данных Oracle поддерживает как ручную, так и автоматическую блокировку. Автоматически созданные блокировки всегда действуют на самом нижнем уровне, чтобы минимизировать вероятность возникновения потенциальных конфликтов с другими транзакциями.

Уровни блокировки:

Механизм блокирования разработан для обеспечения максимально возможной степени параллельного доступа к данным базы данных. Для транзакций, которые изменяют данные, используются блокировки на уровне строки, а не на уровне блока или таблицы. При изменениях объектов (например, перемещении таблицы) возникают блокировки на уровне объекта, при этом база данных или схема не блокируются.

При запросе данных блокировки не возникают. Кроме того, запрос будет выполнен успешно даже в том случае, если опрашиваемые данные уже заблокированы (будут отображены первоначальные значения, восстановленные из данных отмены операций и существовавшие до создания блокировки).

В ситуациях, когда несколько транзакций пытаются заблокировать один ресурс, блокировку создаст первая транзакция, которая попыталась обратиться к нему. Другие транзакции будут ожидать окончания первой транзакции. Механизм обслуживания очередей работает автоматически и не требует вмешательства администратора.

Все автоматические блокировки снимаются после подтверждения транзакции. Транзакция считается защищенной после выполнения оператора COMMIT или ROLLBACK. Если транзакцию выполнить не удалось, тот же фоновый процесс автоматически отменит все изменения, внесенные незаконченной транзакцией, и снимет все созданные ею блокировки.

Каждая DML-транзакция(INSERT, UPDATE и DELETE) использует две блокировки:

• блокировку EXCLUSIVE на уровне строки для обновляемой строки или строк

• блокировку ROW EXCLUSIVE на уровне таблицы для таблицы, содержащей строки Это не даст другому сеансу заблокировать всю таблицу (возможно, для ее удаления или усечения) при внесении изменений.

Блокировка ROW EXCLUSIVE на уровне таблицы не позволяет DDL изменять метаданные словаря при обработке неподтвержденной транзакции. Это позволяет поддерживать целостность словаря и согласованность данных на протяжении жизненного цикла транзакции.

Кроме того, в Oracle данные блокируются автоматически при выполнении таких операций DDL, как CREATE, ALTER и DROP.

Исключающие блокировки DDL

Для выполнения операторов DDL, создающих, изменяющих и удаляющих объекты базы данных, требуются исключающие блокировки DDL *(**exclusive DDL locks**)* соответствующих объектов. Например, при выполнении оператора ALTER TABLE, если к таблице добавляется новое ограничение целостности, транзакция устанавливает исключающую блокировку DDL этой таблицы. При этом другие пользователи не могут модифицировать или удалить таблицу до тех пор, пока оператор ALTER не будет выполнен.

Разделяемые блокировки DDL

При выполнении некоторых операторов DDL могут устанавливаться разделяемые блокировки DDL объектов базы данных. Такие блокировки обычно нужны операторам DDL, с помощью которых между объектами базы данных устанавливаются различные

взаимосвязи. Предположим, например, что создается новый модуль, при этом процедуры и функции модуля ссылаются на множество различных таблиц базы данных. При создании модуля транзакция устанавливает исключающую блокировку DDL модуля и разделяемую блокировку DDL таблиц, на которые происходит ссылка. Разделяемые блокировки DDL не позволяют другим транзакциям устанавливать исключающие блокировки этих таблиц для их изменения или удаления до тех пор, пока не закончится компиляция модуля. Однако разделяемые блокировки не устраняют возможности того, что другие транзакции также установят разделяемые блокировки DDL таблиц, на которые производится ссылка, например, для создания хранимых процедур, использующих эти таблицы.

Блокировки синтаксического анализа

Oracle использует разделяемый пул для кэширования SQL-операторов и программ PL/SQL, прошедших этапы синтаксического анализа и оптимизации. Это уменьшает время реакции системы для пользователей, которые выполняют одно и то же приложение. Объект, кэшированный в разделяемом пуле, устанавливает блокировки синтаксического анализа для объектов базы данных, на которые ссылается. Блокировка синтаксического анализа (parse lock) — это блокировка DDL уникального типа, которая применяется для управления взаимосвязями между объектом, находящимся в разделяемом пуле, и объектами базы данных, на которые тот ссылается. Если транзакция изменяет или удаляет объект базы данных в то время, когда объект в разделяемом пуле удерживает блокировку синтаксического анализа для объекта базы, то Oracle делает объект, находящийся в разделяемом пуле, недействительным. Затем, когда приложение в следующий раз выполняет SQL-оператор или программу PL/SQL, Oracle знает, что нужно повторить синтаксический анализ и оптимизацию оператора, чтобы учесть изменения, произошедшие с объектами базы данных, на которые производится ссылка.

22. Конфликты блокировок, устранение конфликтов блокировок. Взаимные блокировки(уч. 9).

Конфликты блокировок

Транзакция 1	Время	Транзакция 2
`UPDATE employees SET salary=salary+100 WHERE employee_id=100;` `1 row updated.`	9:00:00	`UPDATE employees SET salary=salary+100 WHERE employee_id=101;` `1 row updated.`

`UPDATE employees SET COMMISION_PCT=2 WHERE employee_id=101;`
Сеанс ожидает постановки в очередь из-за конфликта блокировок.
	9:00:05 	`SELECT sum(salary) FROM employees;`
		`SUM(SALARY)`
		`----------`
		`692634`

Сеанс все еще ожидает!	16:30:00	Множество операций выборки, вставки, обновления и удаления за последние 7,5 часов, но без каких-либо фиксаций или откатов!
`1 row updated.` Сеанс продолжается.	16:30:01	`commit;`

Причины конфликтов блокировок:

- Неподтвержденные изменения
- Транзакции, которые обрабатываются длительное время.
- Неадекватно высокие уровни блокирования.

Устранение конфликтов блокировок:

Чтобы устранить конфликт блокировок, сеанс должен освободить созданную блокировку. Наилучший способ достичь этого – связаться с пользователем и попросить его завершить транзакцию.

В крайнем случае администратор может прервать сеанс, создавший блокировку, нажав кнопку «Kill Session» (Удалить сеанс). Помните, что при таком завершении сеанса все изменения, внесенные данной транзакцией, будут потеряны (будет выполнен откат). Пользователю завершенного сеанса нужно будет повторно подключиться к базе данных и повторить все операции, которые были выполнены после последней операции подтверждения.

Пользователи, сеанс которых был прерван, получат следующее сообщение об ошибке при попытке выполнить новый SQL-оператор:

```
ORA-03135: connection lost contact
```

Примечание. Снайпер сеансов PMON может автоматически прерывать сеансы по истечении заданного времени простоя. Также это можно делать с помощью профилей или диспетчера ресурсов.

Устранение конфликтов блокировок с помощью SQL

Управление сеансами, как и большинство других заданий, можно осуществлять не только в Enterprise Manager, но и с помощью операторов SQL. Таблица v$session содержит

подробные сведения обо всех подключенных сеансах. blocking_session – идентификатор блокирующего сеанса. Если запросить SID и SERIAL# (где SID совпадает с идентификатором заблокированного сеанса), вы получите всю необходимую информацию для выполнения операции kill session (удаления сеанса).
Примечание. Для автоматического отключения сеансов, которые бездействуют и блокируют другие сеансы, можно использовать диспетчер ресурсов базы данных (Database Resource Manager).

Взаимные блокировки:

Транзакция 1		Транзакция 2
`UPDATE employees SET salary = salary x 1.1 WHERE employee_id = 1000;`	9:00	`UPDATE employees SET manager = 1342 WHERE employee_id = 2000;`
`UPDATE employees SET salary = salary x 1.1 WHERE employee_id = 2000;`	9:15	`UPDATE employees SET manager = 1342 WHERE employee_id = 1000;`
`ORA-00060: Обнаружена взаимная блокировка во время ожидания ресурса`	9:16	

Взаимная блокировка – это особый случай конфликта блокировок. Взаимные блокировки возникают при участии двух (или больше) сеансов, когда сеансы ожидают данные, взаимно заблокированные друг другом. Так как каждый из сеансов ожидает данные, заблокированные противоположной стороной, ни один из сеансов не может завершить транзакцию и устранить конфликт.
База данных Oracle автоматически выявляет взаимные блокировки и прерывает выполнение оператора с сообщением об ошибке. Реакцией на ошибку должна быть операция подтверждения или отмены изменений, которая освободит блокировку одного из сеансов, благодаря чему второй сеанс сможет завершить свою транзакцию.
В примере, приведенном в данном слайде, в ответ на ошибку о возникновении взаимной блокировки транзакция 1 должна выполнить операцию подтверждения или отмены изменений. В случае подтверждения потребуется еще раз выполнить второе обновление, чтобы успешно завершить транзакцию. В случае отката потребуется выполнить оба обновления, чтобы успешно завершить транзакцию.

23. Данные отмены операций (UNDO), изменение данных отмены операций и журнала повторов операций во время транзакции (уч. 10).

Данные отмены операции

База данных Oracle сохраняет исходные значения данных (данные отмены операций), когда процесс изменяет данные в базе данных. Данные сохраняются до внесения в них изменений. Сохранение данных отмены операций позволяет выполнить откат неподтвержденных данных. Данные отмены операций поддерживают целостность данных при чтении и запросы с моментальным откатом. Данные отмены операций могут использоваться для отмены (моментального отката) транзакций и таблиц.

Данные отмены операций:

• являются копией оригинальных (неизмененных) данных;
• сохраняются для каждой транзакции, которая изменяет данные;
• сохраняются как минимум до завершения транзакции;
• используются для:
  – операций отката;
  – обеспечения целостности данных при чтении;
  – запросов, транзакций и таблиц моментального отката;
  – восстановления данных после сбойных транзакций.

В случае возникновения сбойной транзакции база данных Oracle отменяет все изменения, которые внес пользователь, и восстанавливает исходные значения данных.

Данные отмены операций сохраняются для всех транзакций как минимум до завершения транзакции по одной из причин:

• пользователь отменил транзакцию (откат транзакции)
• пользователь завершил транзакцию (фиксация транзакции)
• пользователь выполнил DDL-оператор, такой как CREATE, DROP, RENAME или ALTER. Если текущая транзакция содержит DML-операторы, база данных сначала подтвердит транзакцию, а затем выполнит и подтвердит операторы DDL в виде новой транзакции
• неплановое завершение сеанса пользователя (откат транзакции)
• запланированное завершение сеанса пользователя (фиксация транзакции).

Объем хранимых данных отмены операций и время хранения зависят от уровня активности базы данных и ее конфигурации.

Данные отмены операций и данные повторов во время транзакций

Данные отмены операций и данные повторов на первый взгляд похожи, однако они используются для разных целей. Данные отмены операций нужны для того, чтобы иметь возможность отменить внесенные изменения (обычно это требуется для сохранения целостности данных при чтении, а также в операциях отката). Данные повторов нужны для того, чтобы иметь возможность повторно применить изменения, если они будут утеряны по каким-либо причинам. Изменения в блоках данных отмены операций также сохраняются в журнале повторов.

Процесс фиксации сохраняет внесенные транзакцией изменения в файл журнала повторов, который является постоянным файлом на диске, в отличие от данных, хранимых в памяти. Кроме того, файл журнала повторов обычно мультиплексируется. Таким образом, на диске может храниться множество копий данных повторов. Так как изменения могут быть еще не записаны в файлы данных, в которых хранятся блоки таблицы, запись в постоянный журнал повторов является достаточной гарантией сохранения целостности базы данных.

Представим, что произошло отключение питания до того, как подтвержденные изменения были сохранены в файлах данных. Так как транзакция была подтверждена, данные не будут утеряны. При следующем запуске системы можно будет осуществить накат записей повторов для сохранения изменений, существовавших на момент отключения питания, в файлах данных.

24. Автоматическое и ручное управление основной и разделяемой памятью.

Разделяемая память:

В прежних версиях Oracle администраторы тратили довольно много времени на подбор правильного размера SGA. Ничего необычного не было в том, чтобы довольно часто выполнять перекалибровку размера SGA, добиваясь оптимальной настройки экземпляра. В Oracle Database 11g вы можете конфигурировать автоматическое управление памятью, используя новый параметр инициализации MEMORY_TAGRET. Все, что необходимо сделать для этого – это присвоить определенное значение параметру MEMORY_TARGET, и Oracle возьмет на себя автоматическое распределение памяти между компонентами SGA и PGA. Выделение памяти SGA Oracle различным компонентам происходит не статически, а меняется по мере изменения загрузки базы данных. Oracle может автоматически управлять следующими пятью компонентами SGA (соответствующие инициализационные параметры Oracle указаны в скобках):

- Буферный кэш базы данных (DB_CACHE_SIZE);
- Разделяемы пул (SHARED_POOL_SIZE);
- Большой пул (LARGE_POOL_SIZE);
- Пул Java (JAVA_POOL_SIZE);
- Пул потоков (STREAMS_POOL_SIZE);

Как видите, Oracle автоматически настраивать пять компонентов SGA, которые мы называем параметрами SGA с автоматически устанавливаемым размером. Вы должны по-прежнему самостоятельно управлять остальными компонентами SGA, даже при автоматическом управлении памятью.

Ниже приведены настраиваемые вручную компоненты SGA:

- Постоянный буферный кэш (DB_KEEP_CACHE_SIZE);
- Повторно используемый буферный кэш (DB_RECYCLE_CACHE_SIZE);
- Все буферные кэши нестандартного размера блока (DB_nK_CACHE_SIZE);
- Буфер журнала повторного выполнения (LOG_BUFFER).

Обратите внимание, что первые три компонента в этом списке необязательны. Как администратор базы данных, вы должны установить значения всех ручных компонентов SGA.

Основная память:

Все компьютеры используют память, которая в действительности состоит из иерархии различных уровней памяти. В сердце иерархии находится главная память (main memory), которая содержит все исполняемые инструкции и манипуляции данными. Вся главная память представляет собой память произвольного доступа (RAM), что означает возможность чтения байта из любого ее участка за одно и то же время. Обычно вы имеете доступ к данным в главной памяти за 10-100 наносекунд.

Важная часть информации Oracle, хранимой в выделенной ему RAM, составляет программный код, выполняющийся в данный момент или выполнявшийся только что. Если новый пользовательский процесс нуждается в том же коде, он доступен ему в памяти в скомпилированном виде, что значительно ускоряет время его выполнения. Области памяти также содержат информацию о том, какие пользователи заблокировали определенную таблицу, тем самым повышая эффективность коммуникаций между разными сеансами. Но наиболее важно, наверное, то, что области памяти помогают в обработке данных, находящихся в постоянном дисковом хранилище. Oracle не проводит непосредственных изменений в данных на диске: данные всегда читаются с диска, удерживаются в памяти и изменяются там перед тем, как быть переданными обратно на диск.

Для обозначения этих участков памяти принято использовать термин буферы. Буферы памяти – это постраничные области памяти, в которые Oracle передает содержимое дисковых блоков. Если база данных нужно прочесть (выбрать) или обновит данные, она копирует соответствующие блоки с диска в буферы памяти. После проведения необходимых изменений, Oracle переносит содержимое буферов памяти на диск.

Oracle использует два типа структур памяти – общую и относящуюся к процессу. Системная глобальная область (system global area – SGA) – это часть общей памяти,

которую разделяют между собой все серверные процесс (включая фоновые). Специфичная для процессов часть памяти известна как программная глобальная область (program global area - PGA), или принадлежащая процессу память (process-private memory). В последующих разделах мы рассмотрим эти два компонента памяти Oracle более подробно.

25. Ошибки пользователя БД. Область мгновенного восстановления.

Ошибка пользователя: операция выполняется успешно, но сама операция неверна (удаление таблицы или ввод неверных данных).

Причины: случайное удаление или изменение данных. Если изменения не зафиксированы, можно просто сделать откат rollback.

Область мгновенного восстановления:

Область мгновенного восстановления - пространство для размещения архивных журналов, резервных копий, журналов Flashback, зеркальных управляющий и журналов повторов. Упрощает управление хранением бэкапов. Размер: в 2 раза больше. Методика сохранения резервных копий определяет момент устаревания файлов, БД автоматически удаляет ненужные.

- Упрощает управление хранением резервных копий.
- Использует отдельное пространство на диске (отдельно от рабочих файлов БД); рекомендуется использовать отдельный накопитель.
- Расположение задается параметром USE_DB_RECOVERY_FILE_DEST.
- Должна быть достаточного размера.
- Управляется автоматически.

Настройка области мгновенного восстановления предполагает определение ее расположения, размера и методики сохранения.

Чтобы задействовать мгновенную область восстановления, Вы должны установить два параметра инициализации DB_RECOVERY_FILE_DEST_SIZE (который определяет дисковую квоту, или максимальное пространство, используемое для файлов FRA этой базы данных) и DB_RECOVERY_FILE_DEST (который указывает местоположение для FRA).

<u>26. Ошибки экземпляра БД. Процедура и этапы восстановления экземпляра.</u>

Сбой экземпляра: непредвиденное завершение работы экземпляра базы данных.

Автоматическое восстановление после сбоя экземпляра:

- Производится вследствие попыток открыть базу данных, файлы которой не были синхронизированы при завершении работы.
- Не требует от администратора никаких действий, за исключением запуска экземпляра БД.
- Использует информацию, хранящуюся в группах журналов повторов, для синхронизации файлов.
- Состоит из двух отдельных операций:
- накат: восстанавливается состояние файлов данных до момента сбоя экземпляра;
- откат: внесенные, но не зафиксированные изменения возвращаются к исходному состоянию.

Этапы восстановления экземпляра:

1) Файлы данных не синхронизированы.
2) Накат (повтор).
3) Зафиксированные и незафиксированные данные в файлах.
4) Открытие БД.
5) Откат (отмена).
6) Зафиксированные данные в файлах.

- Во время восстановления экземпляра к файлам данных необходимо применить все транзакции между позицией контрольной точки и окончанием журнала повторов.
- Настройка восстановления экземпляра производится путем управления интервалом между позицией контрольной точки и окончанием журнала повторов:

```
SQL> ALTER SYSTEM SET FAST_START_MTTR_TARGET=30;
```

27. Резервное копирование. Полная и инкрементальная резервные копии.

Решения для резервного копирования

Можно использовать:

- Утилиты exp/imp (устарели) для схем пользователей;
- Data pump — архитектура резервного копирования, включающая новые утилиты expdp/impdp
- Диспетчер восстановления (RMAN — Recovery MANager);
- Утилиту Oracle Secure Backup;

Стратегия резервного копирования может охватывать:

- всю базу данных (полная);
- часть базы данных (частичная).

В зависимости от типа резервная копия может содержать:

- все блоки данных в пределах выбранных файлов (полная);
- только ту информацию, которая изменилась с момента последнего резервного копирования (инкрементная):
  - кумулятивная (изменения вплоть до последнего уровня 0);
  - дифференциальная (изменения вплоть до последнего инкрементного резервного копирования).

Существует два режима резервного копирования:

- автономное (согласованное, «холодное»);
- оперативное (несогласованное, «горячее»).

Резервные копии можно хранить как:

- копии образов;
- наборы резервирования.

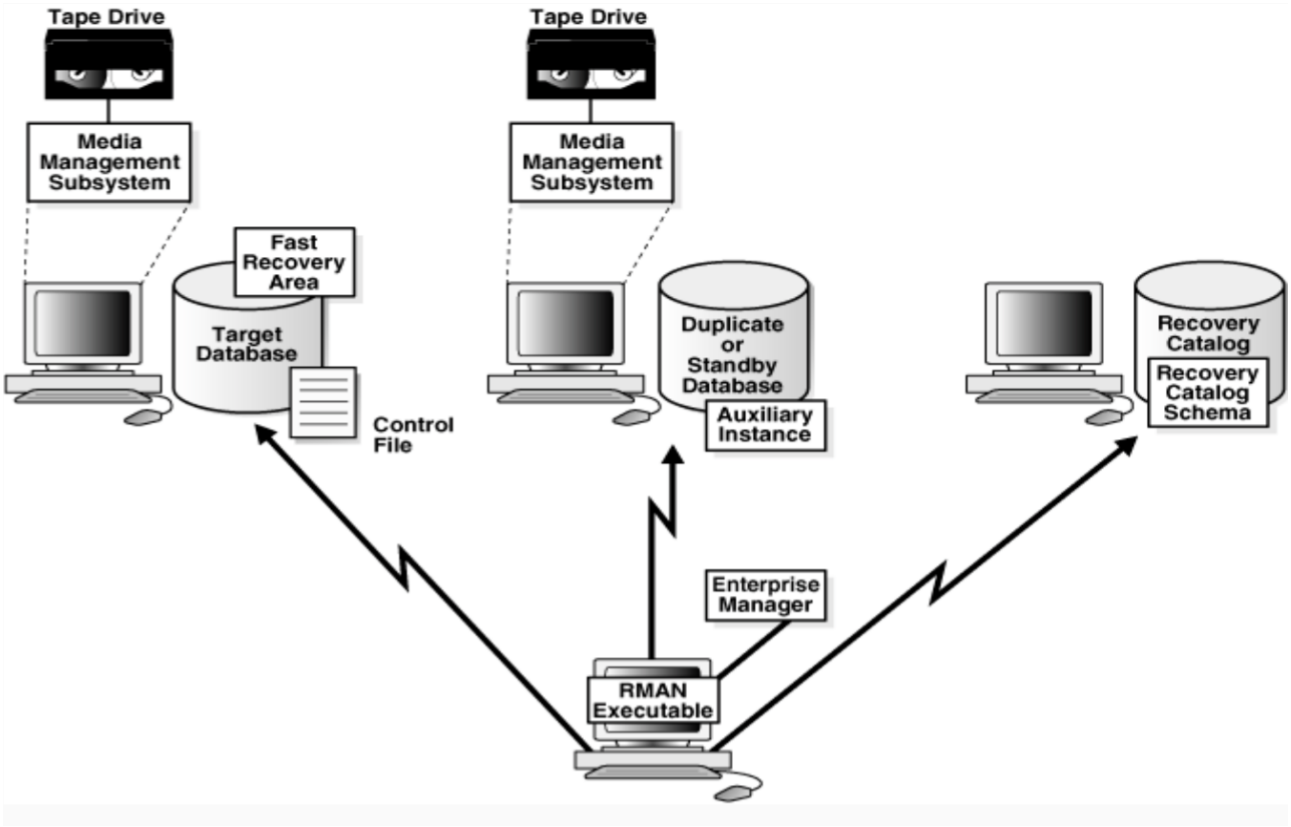
28. Менеджер резервного копирования RMAN

Отдельная утилита для организации сохранения и восстановления БД в автономном или оперативном режимах.

Использует свой собственный «язык».

Состоит из следующих компонент:

- Канал(ы) - серверный процесс, возникающий при установлении связи с устройством ввода/вывода.
- Целевая БД (ключевой параметр TARGET).
- Клиент — утилита rman, может выполнятся на отличной от БД системе.
- Media manager — приложение для управления роботами ленточных библиотек.
- Каталог восстановления — отдельная схема БД для хранении информации о резервных копиях.



Пример вызова утилиты из командной строки и соединения с целевой БД:

```
$ rman
RMAN> CONNECT TARGET SYS@prod
target database Password: password
connected to target database: PROD
(DBID=39525561)
```

Пример подключения к целевой БД используя аутентификацию ОС и добавления информации к журналу:

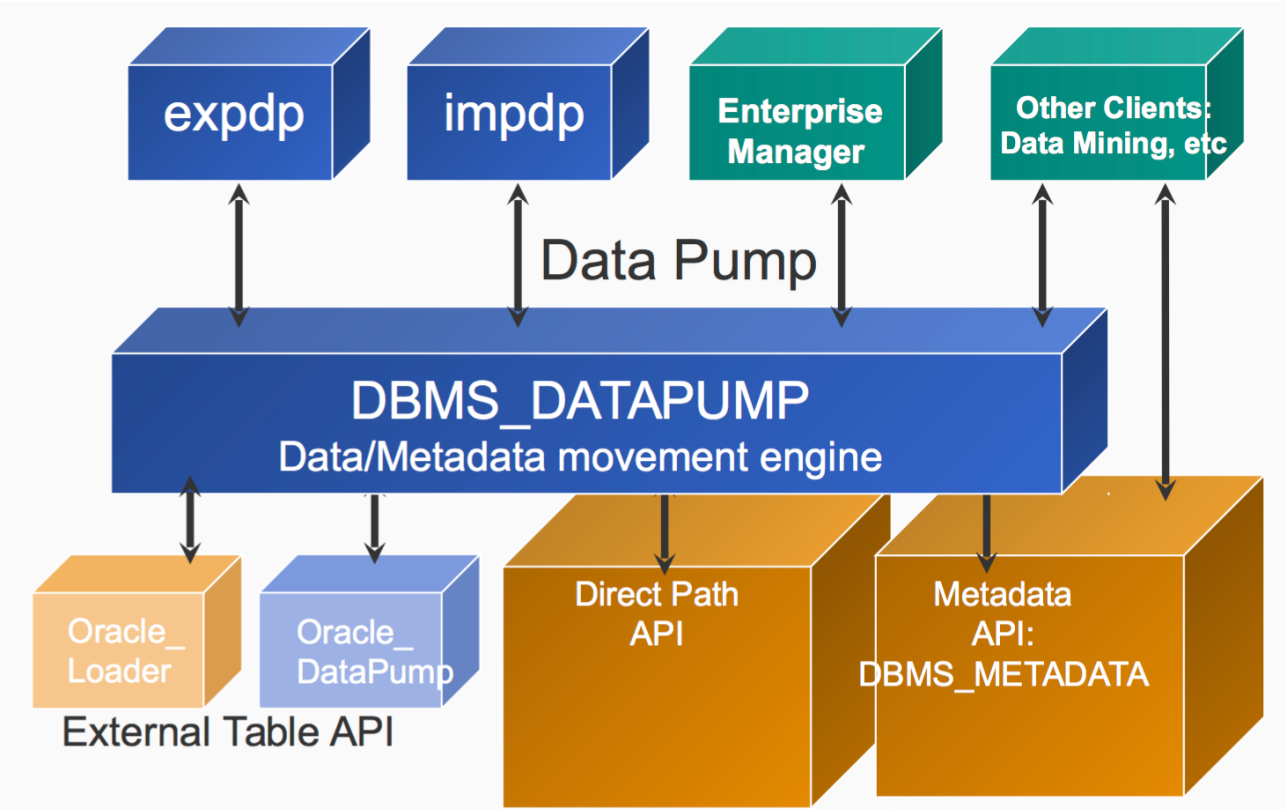
```
$ rman TARGET / LOG /tmp/msglog.log APPEND
$ RMAN NOCATALOG
RMAN> CONNECT TARGET /
RMAN> SHUTDOWN IMMEDIATE
RMAN> STARTUP MOUNT
RMAN> RUN {
2> ALLOCATE CHANNEL d1 TYPE DISK;
3> BACKUP FULL FORMAT '/oracle/oradata/teacher/rman-backup\rman_%d_
%U.bus' DATABASE;
4> }
```

Полное восстановление БД и перевод ее в открытое состояние

```
RMAN> RUN {
2> ALLOCATE CHANNEL d1 TYPE DISK; 3> RESTORE DATABASE;
4> RECOVER DATABASE;
5> ALTER DATABASE OPEN;
6> }
```

29. Data pump — архитектура. Утилиты export и import.

- Высокопроизводительная архитектура на стороне сервера БД
- Загрузка и выгрузка данных и метаданных
- Реализована в пакете DBMS_DATAPUMP.
- Данные передаются в формате потока. Метаданные в виде XML
- Обновленные клиенты expdp and impdp. Синтаксис похож на exp/imp



Особенности архитектуры Oracle Data Pump определяет то, что экспорт/импорт происходит на сервере, dmp-файл формируется на сервере, клиент только управляет процессами экспорта или импорта. Поддерживается параллелизм операций выгрузки и загрузки(только в редакции Oracle Database Enterprise Edition), присутствует возможность предварительной оценки размера dump-файла. Одной из сильных сторон технологии является возможность импорта по сети из одной базы данных в другую, на "лету" без промежуточного файла и экспорт по сети из БД, если та находится в режиме "только чтение".
Data Pump самостоятельно решает, какие методы доступа к данным использовать. Это может быть прямой маршрут или внешние таблицы. Технология использует загрузку или выгрузку по прямому маршруту, когда структура таблицы позволяет сделать это, и в случае если желательна максимальная производительность одиночного потока. Однако при наличие в базе данных кластеризируемых таблиц, ограничений ссылочной целостности, зашифрованных столбцов, или ряда других элементов, Data Pump применит использование внешних таблиц, а не прямой маршрут, чтобы переместить данные.
Все действия Data Pump выполняются множественными задачами (jobs). Эти задачи координируются главным управляющим процессом, который использует расширенную очередь (Advanced Queuing). Во время выполнения, создаётся и используется главным управляющим процессом таблица очереди, которая называется по имени задания. Таблица удаляется после успешного выполнения задания Data Pump. Задание и очередь могут быть названы абсолютно любым именем с использованием параметра "JOB_NAME". Остановка клиентского процесса не останавливает связанное с ним задание. Передача

клиенту комбинации клавиш "Ctrl+C" во время выполнения задания остановит вывод на стандартное устройство вывода и переведёт в командную строку. Ввод "status" в этой командной строке позволить проследить за состоянием текущего задания.

Старые exp/imp:
- Включены в состав СУБД с ранних версий
- Копируют выбранную схему БД в командном режиме на стороне клиента
- Можно копировать отдельно схему и данные
- Загрузка данных в один поток, следовательно невысокая производительность
- Основная проблема — слишком долгое восстановление больших БД.

```
$ EXP SCOTT/TIGER GRANTS=Y TABLES=(EMP,DEPT,MGR)
```

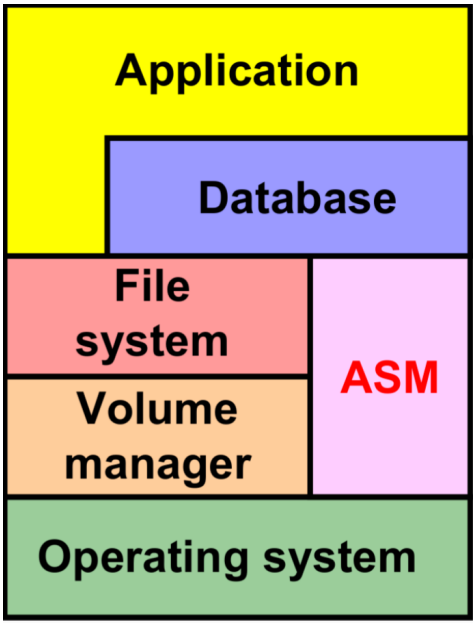
Командный режим expdp/impdp по Ctrl-C:
- ADD_FILE: Добавление файлов к заданию.
- PARALLEL: добавление и удаление обработчиков (workers).
- STATUS: получение статуса обработчиков.
- STOP_JOB{=IMMEDIATE}: Остановка задания с возможностью возобновления работы.
- START_JOB: Рестарт задания.
- KILL_JOB: Уничтожение задания без возможности возобновления. CONTINUE: выход из интерактивного режима, продолжение задания. EXIT: выход из клиента — обработчики продолжают задание.

30. Автоматическое управление хранением (ASM). Назначение, основные возможности.

Automatic Storage Management (ASM) — технология в составе СУБД Oracle, реализующая автоматическое управление хранением данных на уровне менеджера томов.

Особенности ASM:

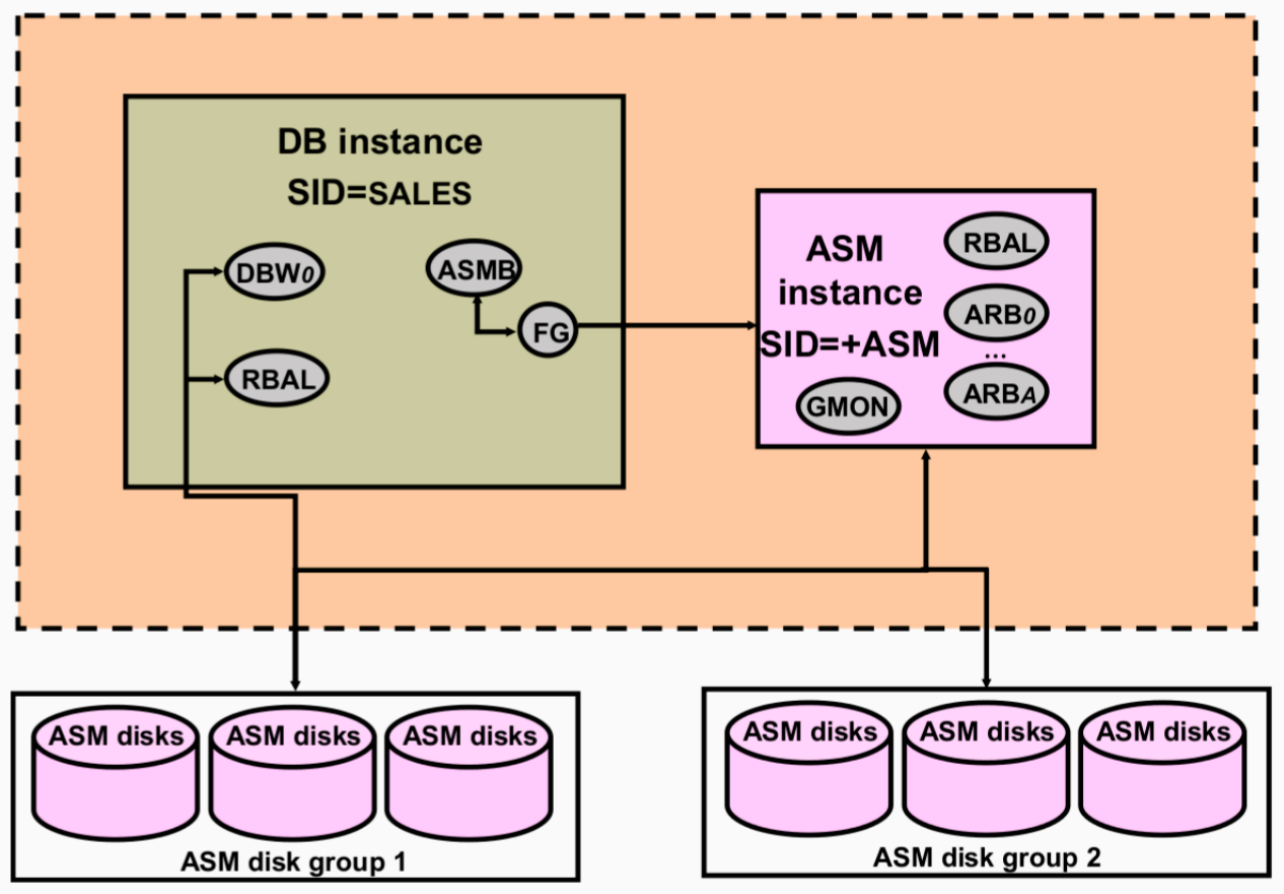
- Управление хранением осуществляется с помощью специальных экземпляров Oracle — ASM Instances.
- ASM может управлять хранением данных как на отдельной машине, так и на уровне кластера RAC в целом.
- Один экземпляр ASM может управлять хранением данных сразу нескольких БД.



Особенности:

- Управление операциями ввода/вывода осуществляется автоматически — «прозрачно» как для приложения, так и для администратора БД.
- Можно расширять хранилище, добавляя в него дополнительные накопители, без остановки БД.
- ASM может управлять созданием резервных копий данных самостоятельно, либо использовать для этого механизмы на уровне ОС и / или системы хранения данных.
- Файлы БД разделены на «блоки распределения» (Allocation Unit — AU). Для того, чтобы определить, где физически находятся файлы блока распределения, используются специальные индексы.
- При изменении ёмкости хранилища (например, при добавлении в него нового диска), файлы внутри AU автоматически перераспределяются пропорционально произошедшему изменению.
- ASM обеспечивает свой механизм зеркалирования, независимый от используемого менеджера томов.
- ASM реализует распределённое хранилище для всех основных файлов в составе экземпляра Oracle — файлов данных, файлов журнала повторов (как оперативных, так и архивных), управляющих файлов и т. д.
- ASM обеспечивает полную поддержку RAC.

31. Экземпляр ASM. Конфигурация, взаимодействие с экземпляром Oracle. Дополнительные процессы в составе экземпляра ASM и экземпляра Oracle.



- Распределённым хранилищем управляет отдельный экземпляр Oracle (ASM Instance), независимый от «основного» экземпляра БД. Этот экземпляр должен быть запущен до запуска экземпляра БД.
- Экземпляр БД получает у экземпляра ASM информацию о расположении необходимых ему файлов, после чего работает с ними напрямую, без участия экземпляра ASM.
- Диски, на которых хранятся данные, объединяются в логические группы — disk groups.
- Экземпляр ASM содержит ряд дополнительных фоновых процессов, которых нет в «обычном» экземпляре БД:
  - RBAL (Rebalance) — управляет перераспределением ресурсов при изменениях в дисковых группах.
  - ARBn (Asm Rebalance Process) — пул процессов, непосредственно осуществляющих перемещение данных AU между дисками.
  - GMON (Group Monitor) — осуществляет мониторинг состояния дисков в группах.
- В случае использования ASM, в экземпляре БД также появляются «дополнительные» процессы:
  - RBAL — управляет доступом к дискам в группах.
  - ASMB (ASM Background Process) — осуществляет взаимодействие с экземпляром ASM.

Конфигурация:

При создании экземпляра ASM используется файл параметров — такой же, как и при создании «обычного» экземпляра Oracle. Тем не менее, он содержит ряд специфичных для ASM параметров:

- INSTANCE_TYPE — должен быт задан как ASM. DB_UNIQUE_NAME — имя сервиса ASM.

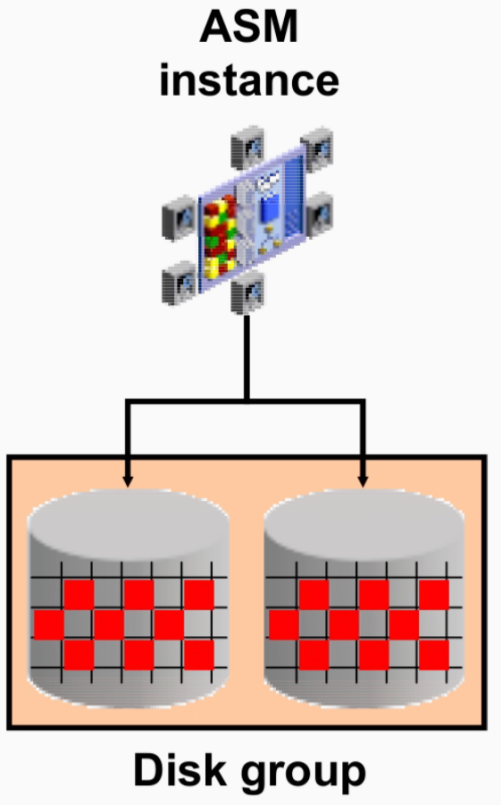
- ASM_POWER_LIMIT — определяет количество ресурсов, которые может использовать ASM при «перебалансировке» БД. Чем выше это значение, тем быстрее ASM выполняет перебалансировку, но тем больше он при этом потребляет ресурсов. Принимает значения от 1 до 11.
- ASM_DISKSTRING — параметр, определяющий набор дисков, которые «видит» ASM.
- ASM_DISKGROUPS — список имён дисковых групп, которые «видит» ASM в момент запуска.

Запуск экземпляра ASM:

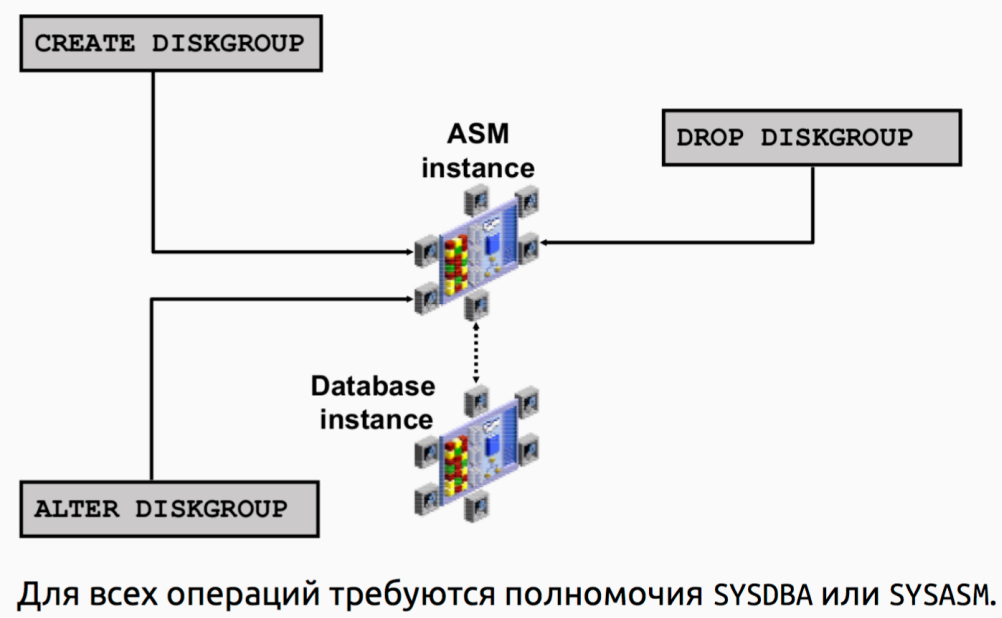
```
$ export ORACLE_SID='+ASM'
$ sqlplus /nolog
SQL> CONNECT / AS sysasm
Connected to an idle instance.
SQL> STARTUP;
Total System Global Area
Fixed Size
Variable Size
ASM Cache
ASM diskgroups mounted
```

32. Дисковые группы. Назначение, особенности конфигурации. Добавление и удаление дисков. Allocation Units. Coarse- & Fine-Grained Striping.

- Дисковая группа — логическое объединение нескольких дисков ASM.
- Может хранить данные нескольких БД.
- Одна БД может хранить свои данные в нескольких дисковых группах.
- Диск может принадлежать только одной дисковой группе.
- Файл ASM может быть сохранён только на одной дисковой группе.
- Файлы хранятся распределённо — сразу на всех дисках, входящих в соответствующую группу.



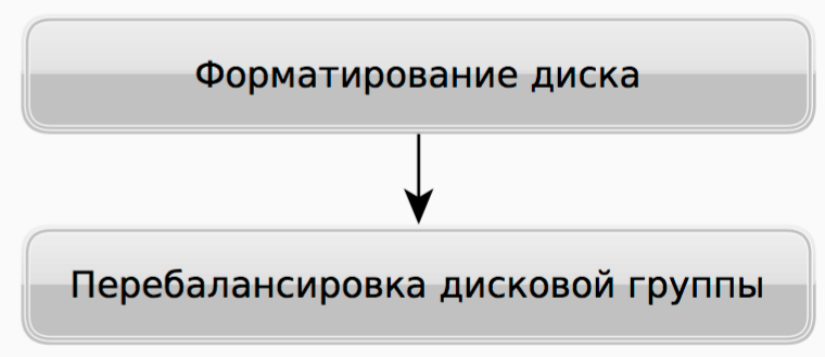
Управление дисковыми группами:



«A» и «B» - разные SCSI-контроллеры — создаём 2 fail groups:

```
CREATE DISKGROUP dgroupA NORMAL REDUNDANCY
FAILGROUP controller1 DISK
    '/devices/A1' NAME diskA1 SIZE 120G FORCE,
    '/devices/A2',
    '/devices/A3'
FAILGROUP controller2 DISK
    '/devices/B1',
    '/devices/B2',
    '/devices/B3';
DROP DISKGROUP dgroupA INCLUDING CONTENTS;

ALTER DISKGROUP dgroupA ADD DISK
    '/dev/rdsk/c0t4d0s2' NAME A5,
    '/dev/rdsk/c0t5d0s2' NAME A6,
    '/dev/rdsk/c0t6d0s2' NAME A7,
    '/dev/rdsk/c0t7d0s2' NAME A8;
ALTER DISKGROUP dgroupA ADD DISK '/devices/A*';
```



Особенностям конфигурации:
Атрибуты дисковых групп

Свойство	Create / Alter	Допустимые значения	Что определяет
au_size	C	1|2|4|8|16|32|64 MB	Размер AU
compatible.rdbms	AC	Валидная версия СУБД	Формат сообщений, которыми обмениваются БД и ASM
compatible.asm	AC	Валидная версия ASM	Формат, в котором хранит свои данные ASM
disk_repair_time	A	От 0 M до 2^{32} D	Время на восстановление, после которого диск переводится в состояние OFFLINE
template.*tname*.redundancy	A	UNPROTECT|MIRROR|HIGH	Способ организации зеркалирования
template.*tname*.stripe	A	COARSE|FINE	Способ размещения данных на дисках

```
CREATE DISKGROUP DATA NORMAL REDUNDANCY
DISK '/dev/raw/raw1','/dev/raw/raw2'
ATTRIBUTE ‘compatible.asm'='11.1';
```

Удаление диска из dgroupA:

```
ALTER DISKGROUP dgroupA DROP DISK A5;
```

Удаление и добавление дисков одной командой:

```
ALTER DISKGROUP dgroupA
    DROP DISK A6
    ADD FAILGROUP fred
DISK '/dev/rdsk/c0t8d0s2' NAME A9;
```

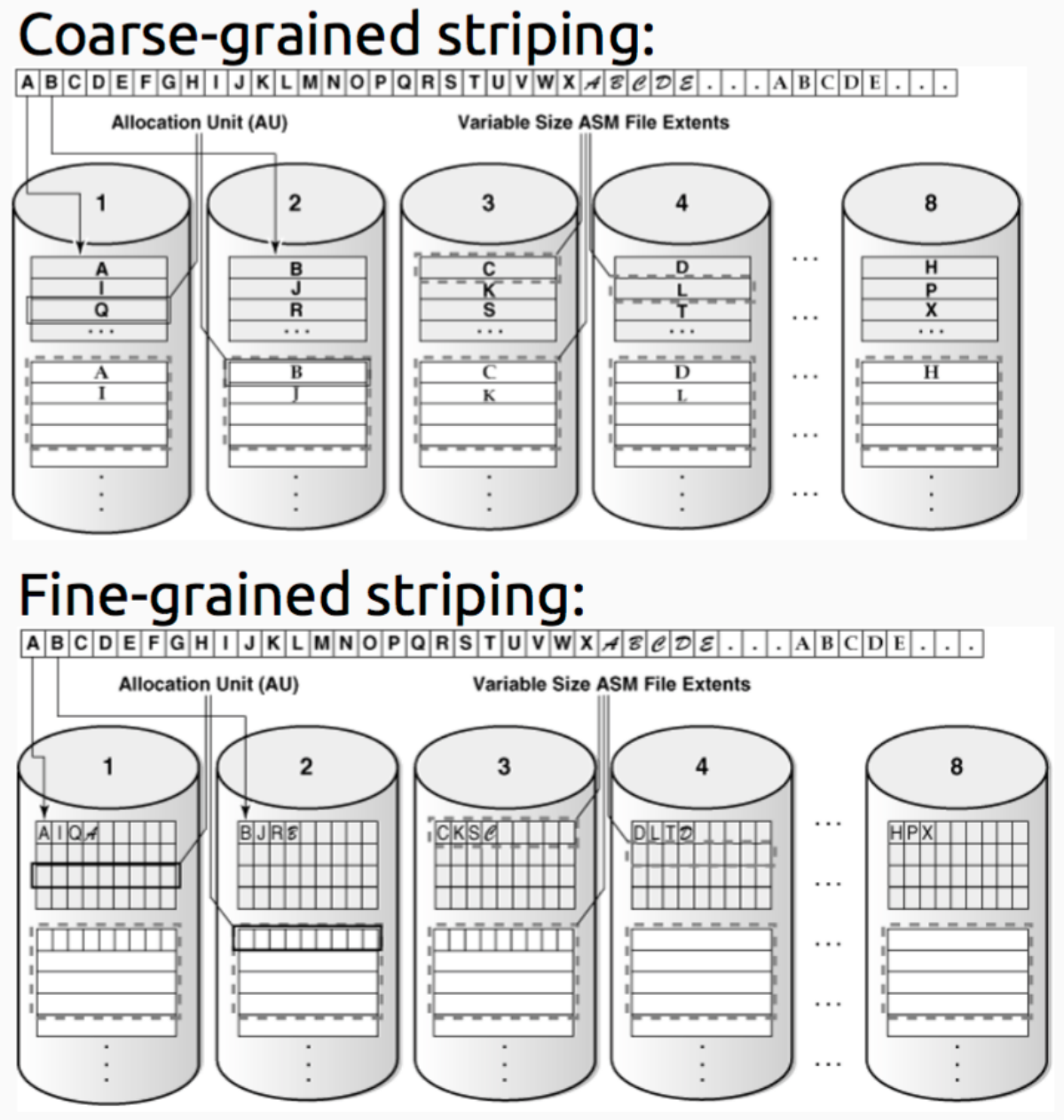
Отмена операции удаления диска:

```
ALTER DISKGROUP dgroupA UNDROP DISKS;
```

Allocation units:

Диски ASM разбиты на блоки распределения (allocation units — Aus):
Размер AU по умолчанию — 1 МБ.
AU — минимальный объём дискового пространства, которым может оперировать ASM.
Один блок данных может храниться только в одном конкретном AU.

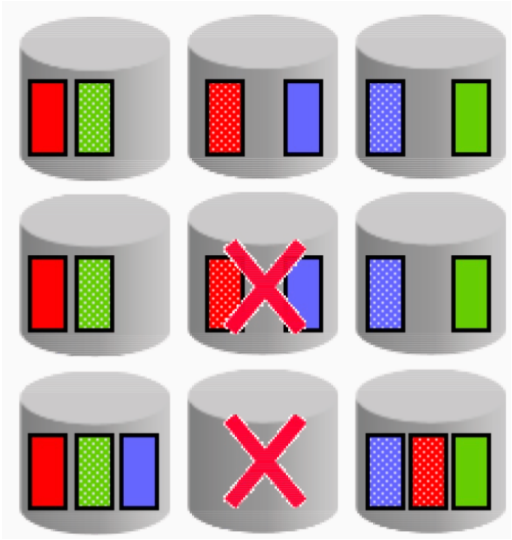
Coarse- & Fine-Grained Striping



33. Избыточность в Oracle ASM. Виды избыточности, конфигурация зеркалирования. Failure-группы.

Зеркалирование дисковых групп:

- Реализовано на уровне AU (а не на уровне дисков).
- На одном и том же диске могут храниться оригиналы и резервные копии различных AU.
- Если «оригинал» AU хранится на одном диске, то его «зеркало» будет храниться на другом диске в рамках той же группы.
- 3 режима организации избыточности данных:
  - «Внешняя» (External Redundancy) — используется аппаратная реализация зеркалирования.
  - «Стандартная» (Normal Redundancy):
    - Двукратное зеркалирование.
    - Как минимум, 2 failure groups.
  - «Высокая» (High Redundancy):
    - Тройное зеркалирование.
    - Как минимум, 3 failure groups.



Конфигурация зеркалирования:

```
create diskgroup dgroupA normal redundancy
disk
        '/export/home/oracle/lab/ko0' name ko0,
        '/export/home/oracle/lab/ko1' name ko1;
```

Failure-группы:

- Набор дисков внутри конкретной группы, использующий общий ресурс (например, контроллер), от отказа которого должна быть обеспечена защита.
- По умолчанию ASM помещает каждый диск в свою собственную failure group. Если вариант «по умолчанию» не устраивает администратора, диски можно перегруппировать.
- ASM автоматичес- ки располагает данные так, чтобы отказ разделяе- мого ресурса failure group не привёл к потере данных.

34. Роли ASM, отличия в уровне привилегий между ними.

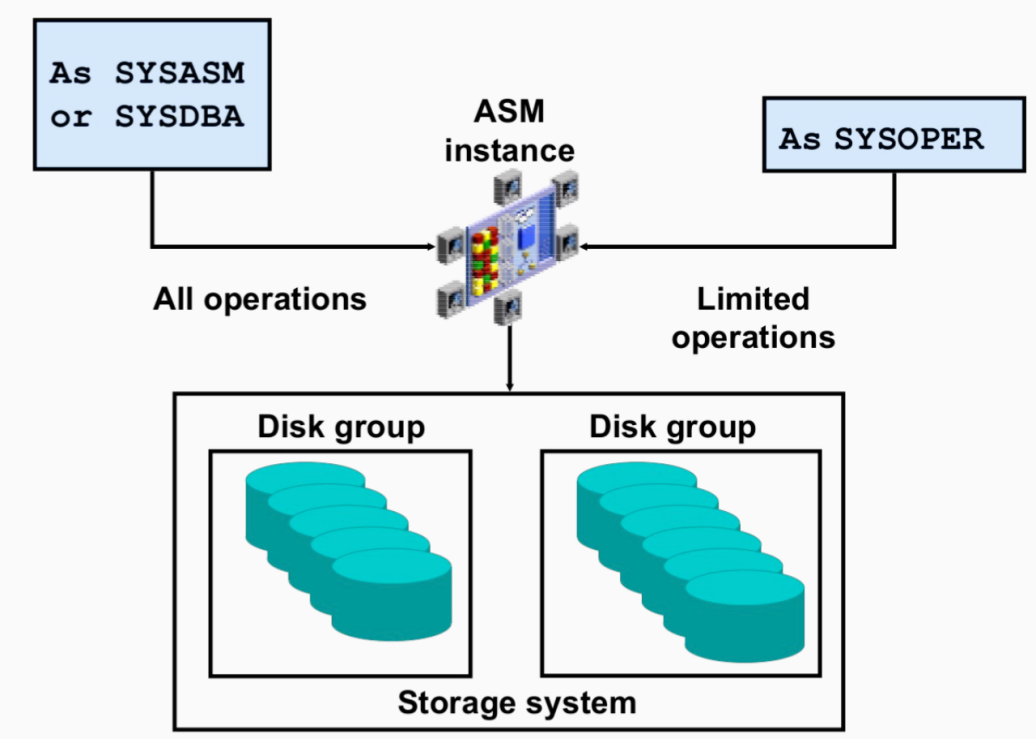
SYSASM — администратор экземпляра ASM:

```
SQL> CONNECT / AS SYSASM
SQL> CREATE USER ossysasmusername IDENTIFIED by passwd;
SQL> GRANT SYSASM TO ossysasmusername;
SQL> CONNECT ossysasmusername / passwd AS SYSASM;
SQL> DROP USER ossysasmusername;
```

По своим возможностям аналогична SYSDBA. В будущих версиях Oracle роль SYSDBA в экземплярах ASM будет недоступна.

- У экземпляров ASM нет своего словаря данных, поэтому к ним можно подключаться только с использованием аутентификации на уровне ОС или файла паролей.
- Существует 3 роли, под которыми пользователь может подключиться к ASM:
- SYSASM и SYSDBA — административный доступ без каких-либо ограничений.
- SYSOPER — набор доступных SQL-команд ограничен минимально необходимым для обслуживания уже сконфигурированной системы:
  - STARTUP/SHUTDOWN
  - ALTER DISKGROUP MOUNT/DISMOUNT ALTER DISKGROUP ONLINE/OFFLINE DISK ALTER DISKGROUP REBALANCE
  - ALTER DISKGROUP CHECK
  - SELECT all V$ASM_* views

Все остальные команды, в частности, CREATE DISKGROUP, ADD/DROP/RESIZE DISK и т. д., требуют наличия привилегий SYSASM или SYSDBA.



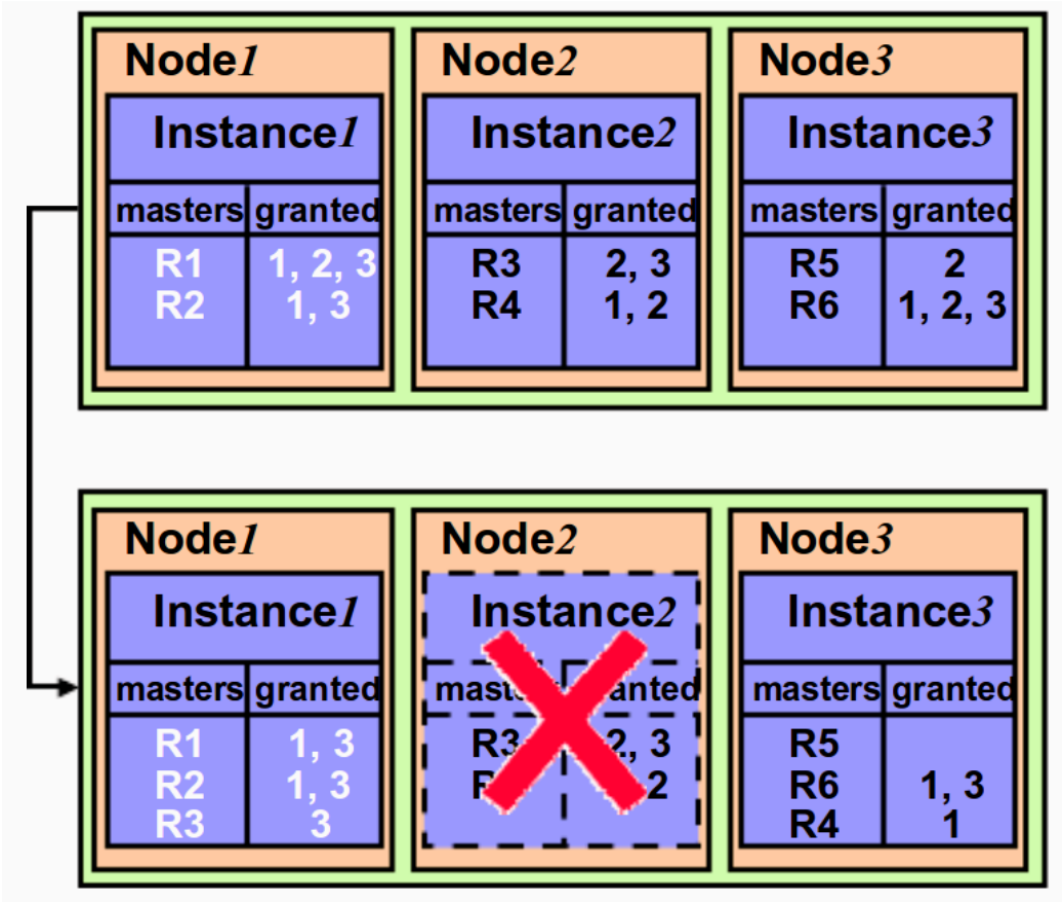
35. Архитектура быстрого восстановления с использованием сервера горячего резерва и кластерная архитектура RAC.

wiki

Кластер высокой доступности обозначаются аббревиатурой HA (англ. *High Availability* — высокая доступность). Создаются для обеспечения высокой доступности сервиса, предоставляемого кластером. Избыточное число узлов, входящих в кластер, гарантирует предоставление сервиса в случае отказа одного или нескольких серверов. Типичное число узлов — два, это минимальное количество, приводящее к повышению доступности. Создано множество программных решений для построения такого рода кластеров.

С **горячим резервом** или **активный/активный.** Все узлы выполняют запросы, в случае отказа одного нагрузка перераспределяется между оставшимися. То есть кластер распределения нагрузки с поддержкой перераспределения запросов при отказе.

Динамическая реконфигурация кластера:



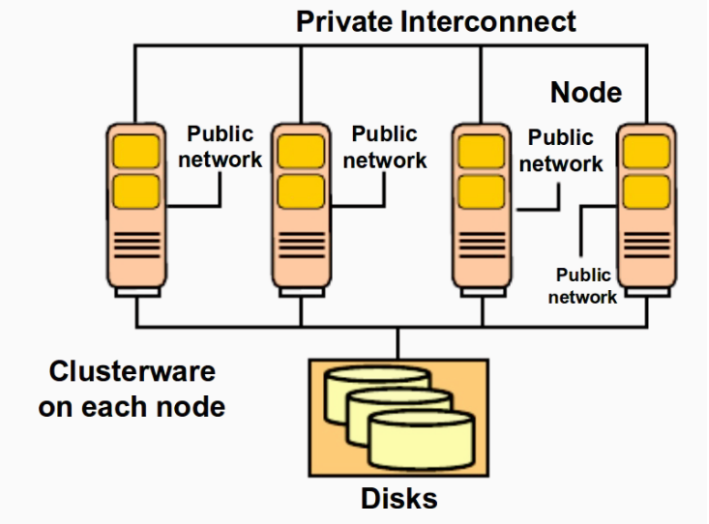
Вдруг какой-то узел не ответил на heartbeat, процесс CSSD на узле, который первым это обнаружил, «бьет тревогу» и докладывает master узлу (если тот еще на связи, в противном случае придется самому брать на себя обязанности master'а). Master инициирует на всех узлах процедуру «голосования», уцелевшие узлы кластера начинают отмечаться на voting disk. Если пропавший узел и тут не оставляет следов, то master начинает процесс исключения пропавшего из кластера. Redo log файл будет читаться дважды: один раз по записям redo, в второй раз (заново) уже по записям undo, чтобы сделать базу доступной для запросов как можно раньше.

1. Часть GRD таблицы с ресурсами упавшего узла «замораживается».
2. Не вышедший на связь узел помечается как «пропавший», чтобы оставшиеся узлы к нему не обращались зря по interconnect.

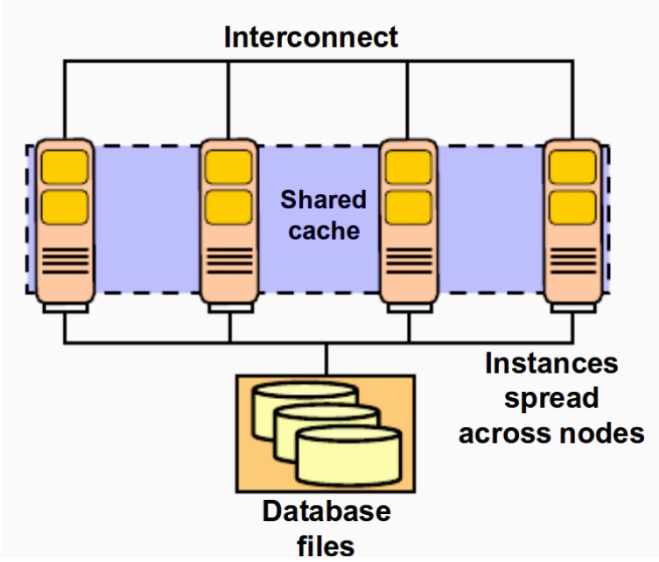
3 Узел, который первым обнаружил пропажу, начинает восстановление информации, которая обрабатывалась на исчезнувшем узле:
   - Понижает темпы обслуживания собственных транзакций, бросая вычислительные ресурсы на восстановление
   - Обращается к общему файловому хранилищу (datastorage), и на себе начинает применять online redo logs, принадлежавшие пропавшему узлу. С учетом порядкового номера SCN блоков, merge их с тем, что хранится в буфере, и «накатывает» (roll-forward) в своем кэше. При этом узел пропускает те устаревшие записи блоков (PI), более поздние версии которых, уже были сброшены на диск. Если у считанных блоков в кластере присутствует master соответствующего ресурса, то узел сообщает список считанных блоков, и master на этих ресурсах выставляет блокировку, чтобы узлы к ним не обращались (пока они восстанавливаются).
   - После чего, вторым прочтением по redo log, учитывая уже undo записи, откатывает (roll-back) незафиксированные транзакции. Происходит это по технологии fast-recovery, т.е. откат транзакций будет производиться отдельным background процессом. Oracle вернет заблокированные незавершенными транзакциями (uncommitted) блоки в согласованное состояние (consistent), к прежним значениям, как только придет запрос на эти блоки. Либо они уже к тому времени будут восстановлены этим самым параллельным background процессом. Таким образом, уже в кластере снимаются блокировки и могут выполняться новые запросы пользователей.
4 Часть таблицы GRD, принадлежавшая упавшему узлу, размораживается на восстанавливающем узле (теперь он master ресурса). Таким образом, в кластере восстанавливается состояние обрабатываемых транзакций на пропавшем узле на момент «падения».

36. Понятие кластера. Аппаратная и программная реализация кластера. Масштабируемость и отказоустойчивость кластерных решений.

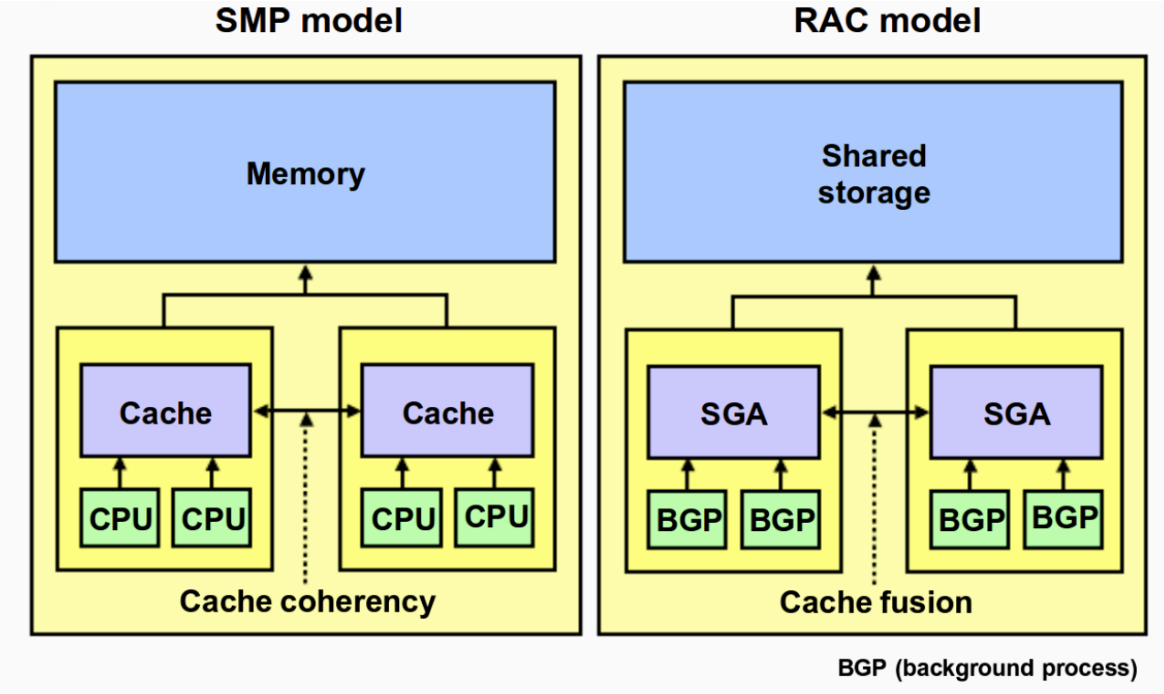
Кластер состоит из двух или более взаимосвязанных машин — узлов кластера.
Узлы кластера «видны» снаружи как единое целое.
Внутренняя структура кластера «скрывается» кластерным ПО — все кластеры «выглядят» снаружи как обычные серверы БД.
Все диски доступны для чтения и записи всем узлам кластера.
На всех узлах кластера используется одна и та же версия ОС.



Oracle RAC
Множество экземпляров Oracle составляют единую БД.
На каждой машине запущено по одному экземпляру Oracle.
Файлы БД — «общие» для всех экземпляров Oracle внутри кластера.
Доступом к данным и взаимодействием между экземплярами Oracle управляет специальное инфраструктурное ПО.



Кластер в сравнении с SMP



Масштабируемость:
Для того, чтобы использование кластера было эффективным, требуется обеспечить масштабируемость на всех уровнях:

- Аппаратный — скорость чтения / записи на диски.
- Взаимодействие между узлами — пропускная способность сети и время отклика.
- Операционная система — возможность работы на многопроцессорных машинах.
- СУБД — синхронизация при параллельном доступе к данным.
- Приложение — особенности архитектуры.

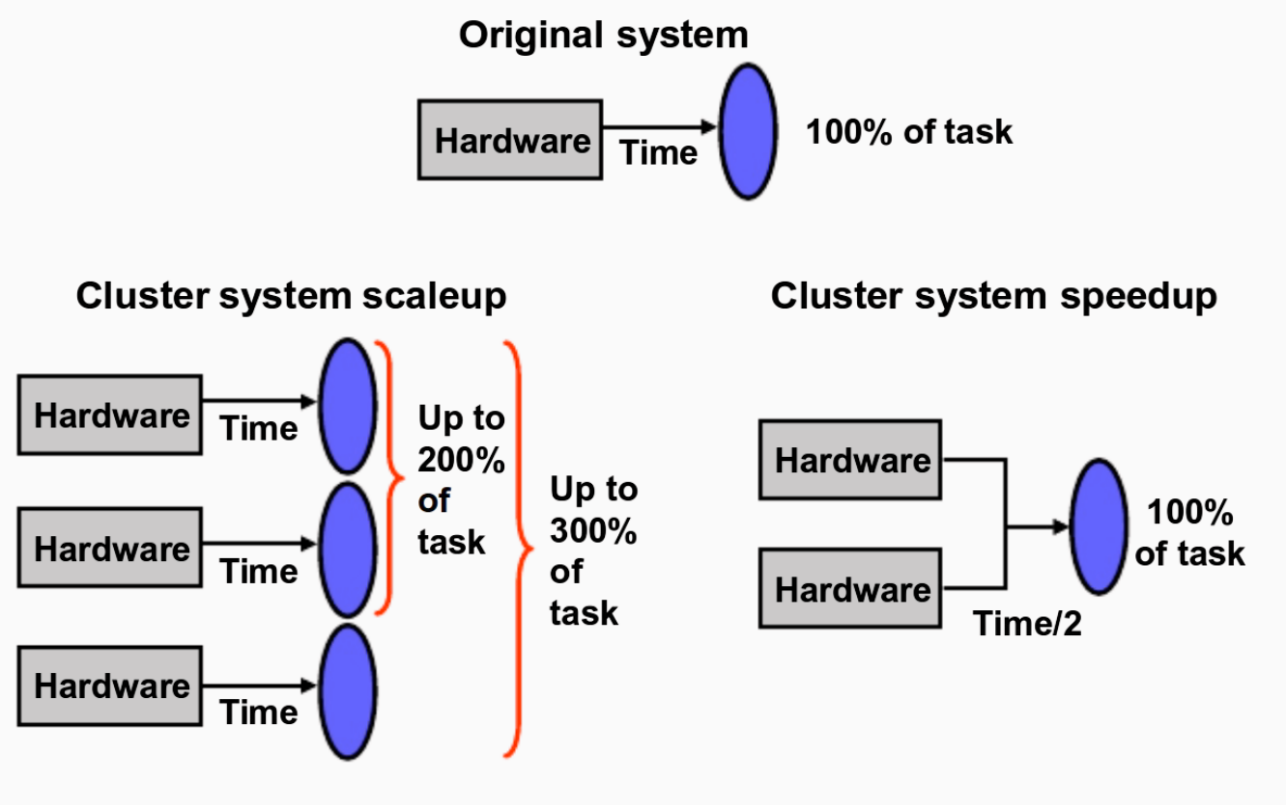
Отказоустойчивость(wiki):
Отказоустойчивый кластер — кластер (группа серверов), спроектированный в соответствии с методом обеспечения высокой доступности и гарантирующий минимальное время простоя за счёт аппаратной избыточности. Без кластеризации сбой сервера приведёт к тому, что поддерживаемые им приложения или сетевые сервисы будут недоступны до восстановления его работоспособности. Отказоустойчивая кластеризация исправляет эту ситуацию, перезапуская приложения на другой системе без вмешательства администратора в случае обнаружения ошибок аппаратного или программного обеспечения.

37. Основные принципы построения масштабируемых приложений. Speedup & Scaleup.

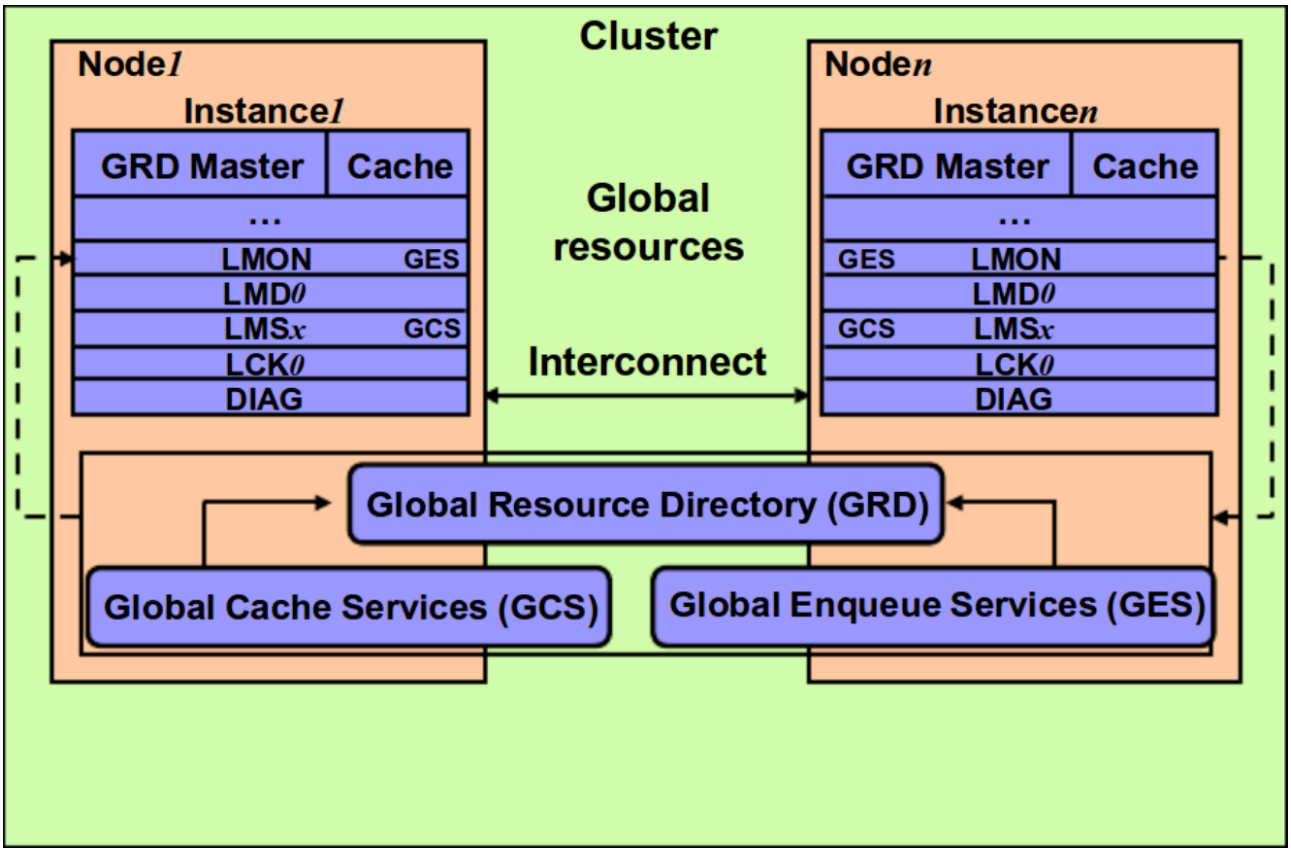
Wiki:

Для запуска в высокодоступной кластерной среде, приложение должно соответствовать, как минимум следующим технические требованиям, последние два из которых имеют решающее значение для его надежной работы в кластере, и которые наиболее трудно в полной мере удовлетворить:

- Должен быть относительно простой способ для запуска, остановки, принудительной остановки, и проверки состояния приложения. В практическом плане это означает, что приложение должно иметь интерфейс командной строки или сценарии для управления приложением, включая поддержку нескольких экземпляров приложения.
- Приложение должно иметь возможность использовать общее хранилище данных (NAS / SAN).
- Очень важно, что приложение должно хранить на энергонезависимом общем хранилище столько, от ее состояния, сколько возможно. Не менее важным является способность перезапуска на другом узле в последнем состоянии до разрушения, с помощью сохраненного состояния из общего хранилища.
- Приложение не должно повреждать данные, если оно выходит из строя, или перезапускается с сохраненного состояния.



38. Глобальные ресурсы Oracle RAC, особенности управления ими. Глобальные DPV.
Управление глобальными ресурсами Oracle RAC:



Самым узким местом любой БД являются дисковый ввод-вывод. Поэтому все базы данных стараются как можно реже обращаться к дискам, используя отложенную запись. В RAC все так же, как и для single-instance БД: у каждого узла в RAM располагается область SGA (System Global Area), внутри нее находится буферный кэш (database buffer cache). Все блоки, некогда прочитанные с диска, попадают в этот буфер, и хранятся там как можно дольше. Но кэш не бесконечен, поэтому, чтобы оценить важность хранимого блока, используется TCA (Touch Count Algorithm), считающий количество обращений к блокам. При первом попадании в кэш, блок размещается в его cold-end. Чем чаще к блоку обращаются, тем ближе он к hot-end. Если же блок «залежался», он постепенно утрачивает свои позиции в кэше и рискует быть замещенным другой записью. Перезапись блоков начинается с наименее используемых. Кэш узла – крайне важен для производительности узлов, поэтому для поддержания высокой производительности в кластере кэшем нужно делиться (как завещал сами-знаете-кто). Блоки, хранимые в кэше узла кластера, могут иметь роль локальных, т.е. для его собственного пользования, но некоторые уже будут иметь пометку глобальные, которыми он, поскрипев ~~зубами~~ дисками, будет делится с другими узлами кластера.

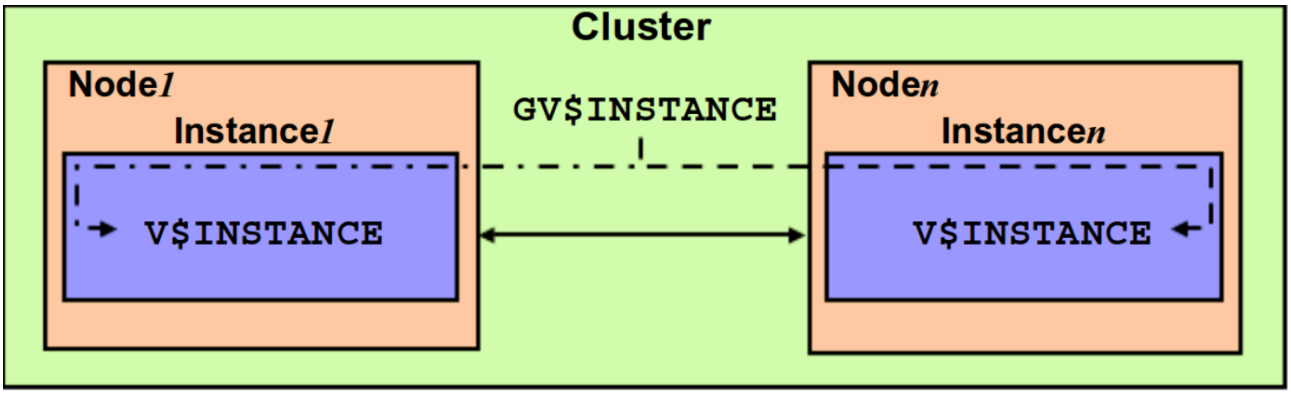
Технология общего кэша в кластере называется Cache-fusion (синтез кэша). CRS(**CRS** (**Cluster-Ready Services**) – набор сервисов, обеспечивающий совместную работу узлов, отказоустойчивость, высокую доступность системы, восстановление системы после сбоя. CRS выглядит как «мини-экземпляр» БД (ПО) устанавливаемый на каждый узел кластера.) на каждом узле порождает синхронные процессы LMSn, общее их название как сервиса — **GCS (Global Cache Service).** Эти процессы копируют прочитанные на этом экземпляре блоки (глобальные) из буферного кэша к экземпляру, который за ними обратился по сети, и также отвечают за откат неподтвержденных транзакций. За их координацию отвечает сервис **GES (Global Enqueue Service)**, представленный на каждом узле процессами LMON и LMD. LMON отслеживает глобальные ресурсы всего кластера,

обращается за блоками к соседним узлам, управляет восстановлением GCS. Когда узел добавляется или покидает кластер, он инициирует реконфигурацию блокировок и ресурсов. LMD управляет ресурсами узла, контролирует доступ к общим блоками и очередям, отвечает за блокировки запросов к GCS и управляет обслуживанием очереди запросов LMSn. В обязанности LMD также входит устранение глобальных взаимоблокировок в рамках нескольких узлов кластера.

Но особая роль в координации ресурсов кластера и объединения кэша отведена таблице **GRD (Global Resource Directory)**, в которой постоянно фиксируется, на каком экземпляре, какой блок (или его копия) доступен, режим, в котором экземпляр удерживает блок, роль блока, SCN. В single-instance варианте SCN – это просто инкрементный счетчик вносимых в БД изменений.
SCN в RAC регулярно синхронизируются до самого большого SCN, известного в кластере. SCN требуется для того, чтобы записи о вносимых изменениях в блоках выстраивались в хронологическом порядке, что необходимо при восстановлении транзакций (roll-forward).

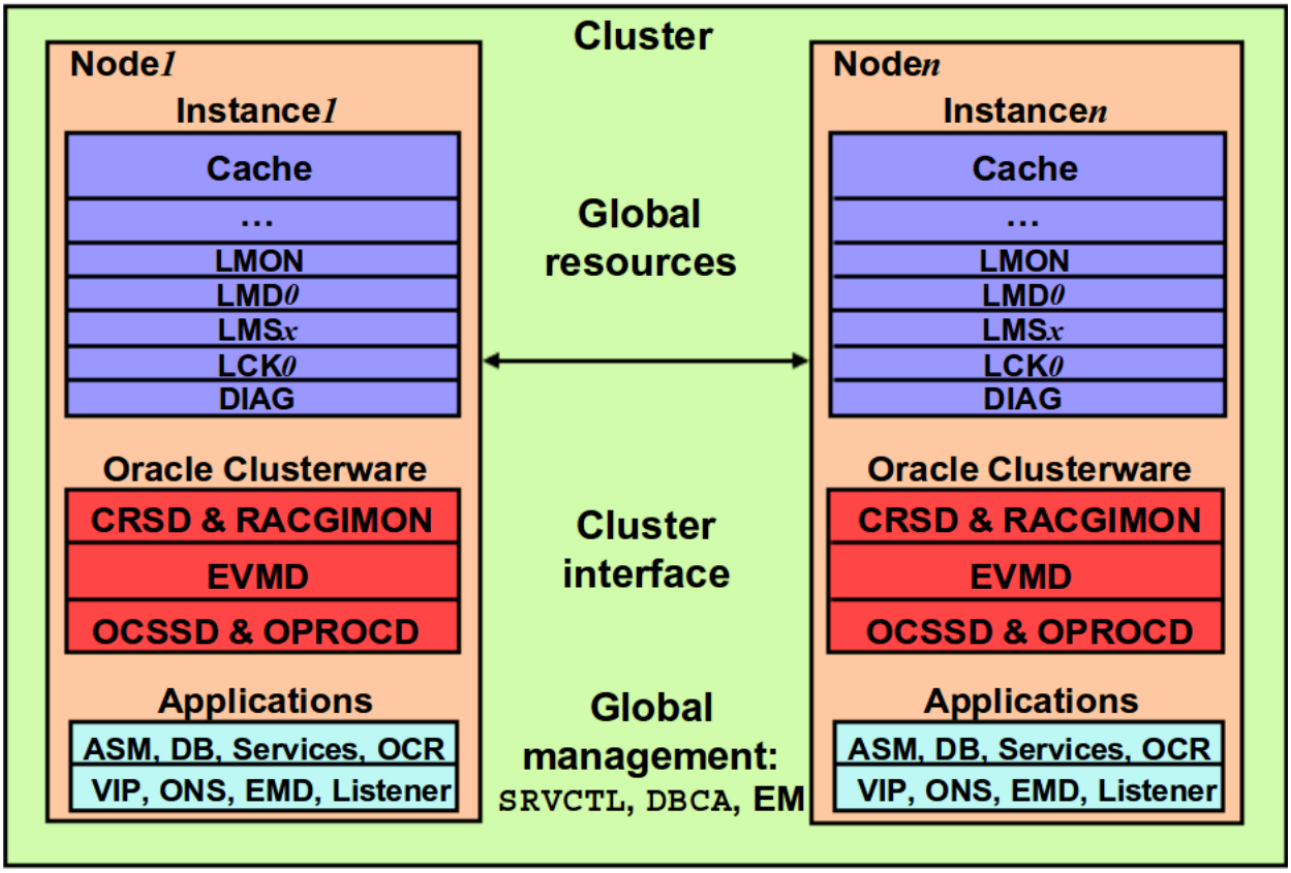
Таблица GRD распределена между узлами кластера. Каждый узел принимает участие в распределении ресурсов кластера, обновляя свою часть GRD. Часть таблицы GRD относится к ресурсам – объектам: таблицы, индексы и.т.п. Она постоянно синхронизируется (обновляется) между узлами.
Когда узел прочел блок данных с диска, он становится master-ом этого ресурса и делает соответствующую отметку в своей части таблицы GRD. Блок помечается как локальный, т.к. узел пока использует его в одиночку. Если же этот блок потребовался другому узлу, то процесс GCS пометит этот блок в таблице как глобальный («опубликован» для кластера) и передаст затребовавшему узлу.

Глобальные DPV(представления производительности):
- Отражают текущее состояние экземпляра БД.
- Информация динамически генерируется в зависимости от состояния.
- Префикс V$ - представления производительности экземпляра БД.
- Префикс GV$ - глобальные представления узлов кластера RAC.
- Содержат информацию обо всех запущенных экземплярах в составе кластера.
- У каждого локального представления (V$) есть соответствующее ему глобальное представление (GV$).
- Исполняются параллельно на всех узлах кластера — «ведущий» запрос на узле, к которому осуществляется обращение и «ведомые» запросы к V$ на остальных узлах.
- Параллелизмом управляет специальный сервис — координатор параллельного исполнения (Parallel Execution Coordinator, PEC).



39. Архитектура Oracle RAC: процессы, конфигурационные файлы, файлы БД.

Процессы:



CRS (Cluster-Ready Services) – набор сервисов, обеспечивающий совместную работу узлов, отказоустойчивость, высокую доступность системы, восстановление системы после сбоя. CRS выглядит как «мини-экземпляр» БД (ПО) устанавливаемый на каждый узел кластера. CRS состоит из 3-х основных компонент:

- CSSD – Cluster Synchronization Service Daemon
- CRSD – Cluster Ready Services Daemon
- EVMD – Event Monitor Daemon

x	Назначение (вкратце)	С какими правами работает	При смерти процесса, перезагружается:
CSSD	Механизм синхронизации для взаимодействия узлов в кластерной среде.	user	процесс
CRSD	Основной «движок» для поддержки доступности ресурсов	root	хост
EVMD	Процесс оповещения о событиях, происходящих в кластере	user	процесс

CRSD (**Cluster Ready Services Daemon**). В его обязанности входит: запуск, остановка узла, генерация failure logs, реконфигурация кластера в случае падения узла, он также отвечает за восстановление после сбоев и поддержку файла профилей OCR. Если демон падает, то узел целиком перезагружается. CRS управляет ресурсами OCR: Global Service Daemon (GSD), ONS Daemon, Virtual Internet Protocol (VIP), listeners, databases, instances, and services.

В обязанности сервиса **CSSD** (**Cluster Synchronization Service Daemon**) входит координация взаимодействия узлов кластера, синхронизация узлов и ресурсов между ними, определение их доступности через следующие функции:

- **Node Membership (NM).**Каждую секунду проверяет heartbeat между узлами. NM также показывает остальным узлам, что он имеет доступ к так называемому **voting disk** (если их несколько, то хотя бы к большинству), делая регулярно туда записи. Если узел не отвечает на heartbeat или не оставляет запись на voting disk в течение нескольких секунд (10 для Linux, 12 для Solaris), то master узел исключает его из кластера.
- **Group Membership (GM).** Функция отвечает за своевременное оповещение при добавлении / удалении / выпадении узла из кластера, для последующей реконфигурации кластера.

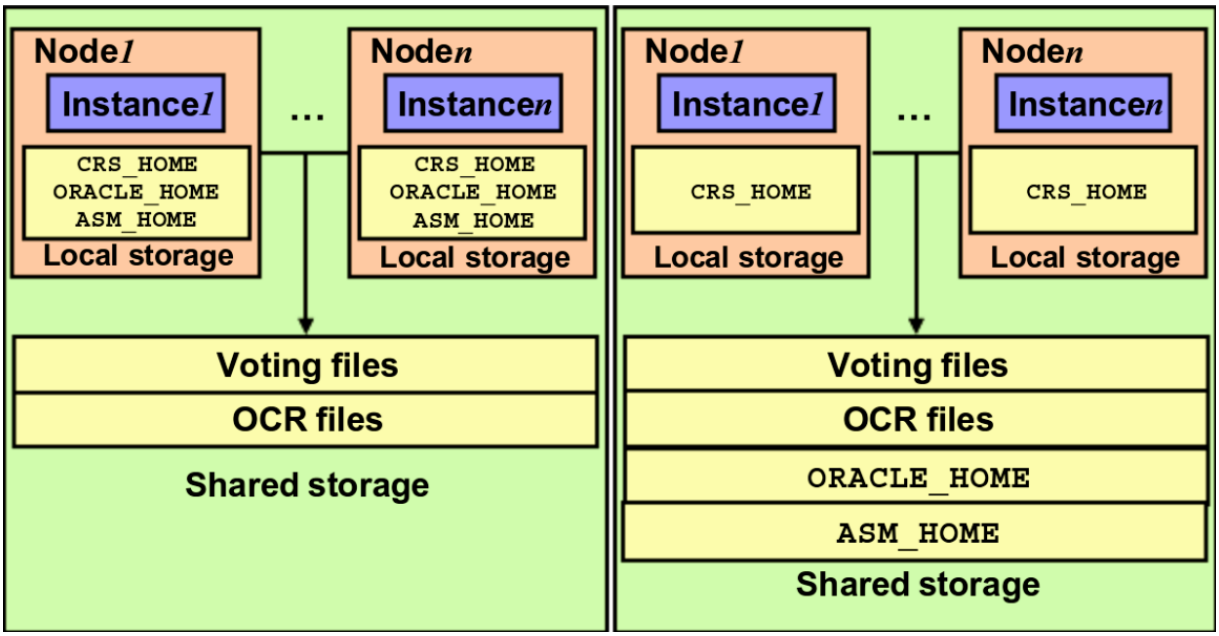
CSSD предоставляет динамическую информацию о узлах и экземплярах, которые являются частью его на текущий момент, и отвечает за блокировки ресурсов в кластере.

Информатором в кластере выступает **EVMD** (**Event Manager Daemon**), который оповещает узлы о событиях: о том, что узел запущен, потерял связь, восстанавливается. Он выступает связующим звеном между CRSD и CSSD. Оповещения также направляются в ONS (Oracle Notification Services), универсальный шлюз Oracle, через который оповещения можно рассылать, например, в виде SMS или e-mail.

Конфигурационные файлы:
Появляются 2 новых категории файлов:

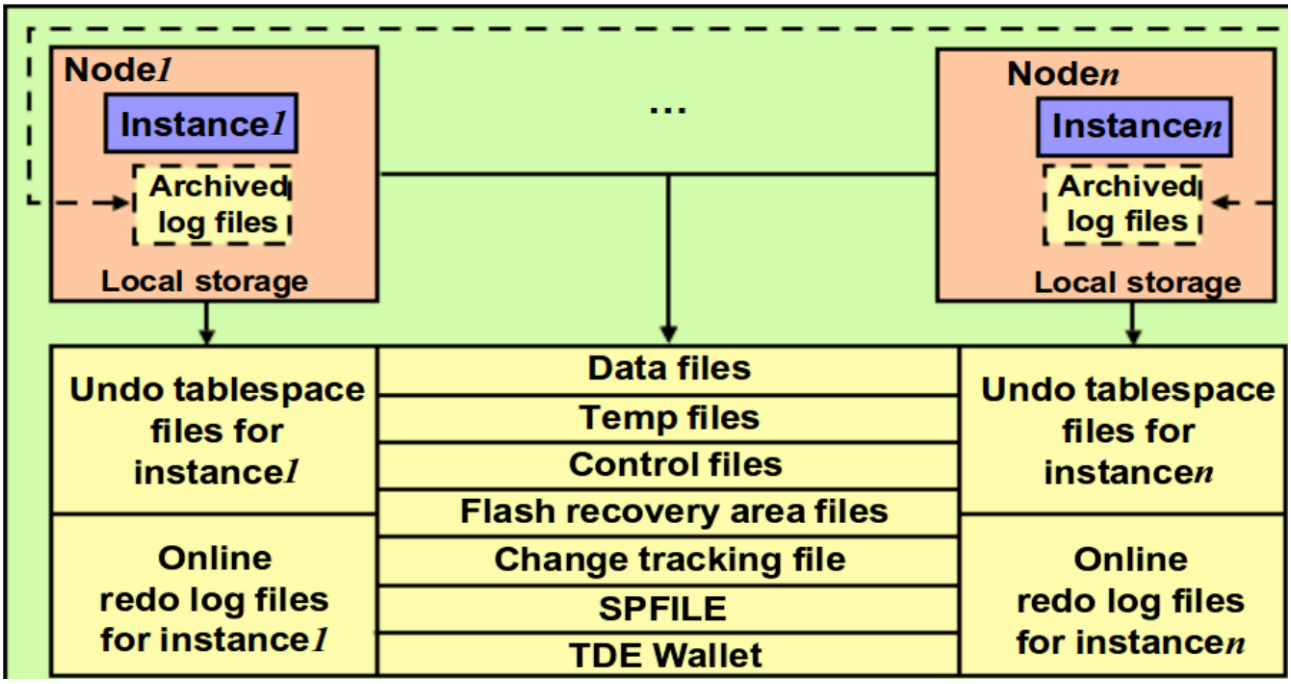
- Файлы с результатами мониторинга состояния кластера.
- Файлы OCR — Oracle Cluster Registry.

Файлы OCR не могут храниться в ASM, т. к. они должны быть доступны ещё до запуска кластера.



Настройки кластера хранятся в **OCR** (**Oracle Cluster Registry**). OCR – это специальный файл профилей узлов базы данных, хранящий их текущую конфигурацию: доступность узлов, распределение сервисов (несколько БД могут поддерживаться различными группами узлов в кластере), сетевые настройки и.т.п. Физически OCR хранится в общем datastorage. При работе кластера каждый узел хранит в памяти OCR, и только один узел (master) производит непосредственное обновление OCR на диске.

Файлы БД:


Node1
Instance1
Archived log files
Local storage
...
Noden
Instancen
Archived log files
Local storage
Undo tablespace files for instance1
Data files
Temp files
Control files
Flash recovery area files
Change tracking file
SPFILE
TDE Wallet
Undo tablespace files for instancen
Online redo log files for instance1
Online redo log files for instancen

40. Варианты построения системы хранения в Oracle RAC, их преимущества и недостатки.

2 варианта: Raw и CFS

Oracle Cluster File System (OCFS)

Файловая система, специально разработанная для хранения разделяемых ресурсов RAC.

Особенности:

- Позволяет всем узлам кластера использовать общий ORACLE_HOME.
- Каждый том OCFS может размещаться на одном или нескольких физических дисках.

В файловом хранилище на базе OCFS можно разместить:

- Установленные бинарные файлы Oracle.
- Файлы экземпляра Oracle (конфигурационные, файлы данных и т. д.).
- Файлы параметров инициализации (spfile).
- Временные файлы, созданные экземпляром Oracle в время работы.
- Voting & OCR files.

Преимущества CFS:

- Проще администрировать.
- Ставится вместе с Oracle, не нужна дополнительная конфигурация.
- Автоматически расширяется по мере возрастания объёма данных.
- Можно использовать для хранения файлов архива журнала повторов.

Недостатки CFS:

- Сложность начальной конфигурации
- возможно уменьшается производительность
- в стек БД добавляется новый продукт.

RAW devices (сырые устройства) - специальные символьные (character) устройства, предназначенные для организации доступа к блочным устройствам без кэширования.

Преимущества raw:

- Потенциально быстрее.
- Ниже требования к инфраструктуре.
- Можно использовать ASM для расширения возможностей.

Недостатки raw:

- Для Oracle 1 raw устройство == 1 файлу => нужно много файлов,
- средствами ОС (df, ls -l) не оценить используемое дисковое пространство и не сделать бэкап/восстановление
- можно хранить только файлы БД
- не поддерживаются с Oracle 12g.