BASICS

Please don't share this to your competitor .

I may be wrong .The mother loves all the kids in the world .

But when it comes to competition , the mother always wants only her kid to win .

Oracle Architectural Components

Oracle Server

Oracle instance + Database

Oracle instance

MEMORY STRUCTURES

+

BACKGROUND PROCESSES

User Process

Server Process

Background process

Oracle components

Data

Actual data

Redo

Changes made to the database-Used for recovery

Undo

Used for Read Consistency

MEMORY STRUCTURES

MEMORY STRUCTURES

1. PROGRAM GLOBAL AREA

Process global area

2. SHARED GLOBAL AREA

System global area

Program Global Area

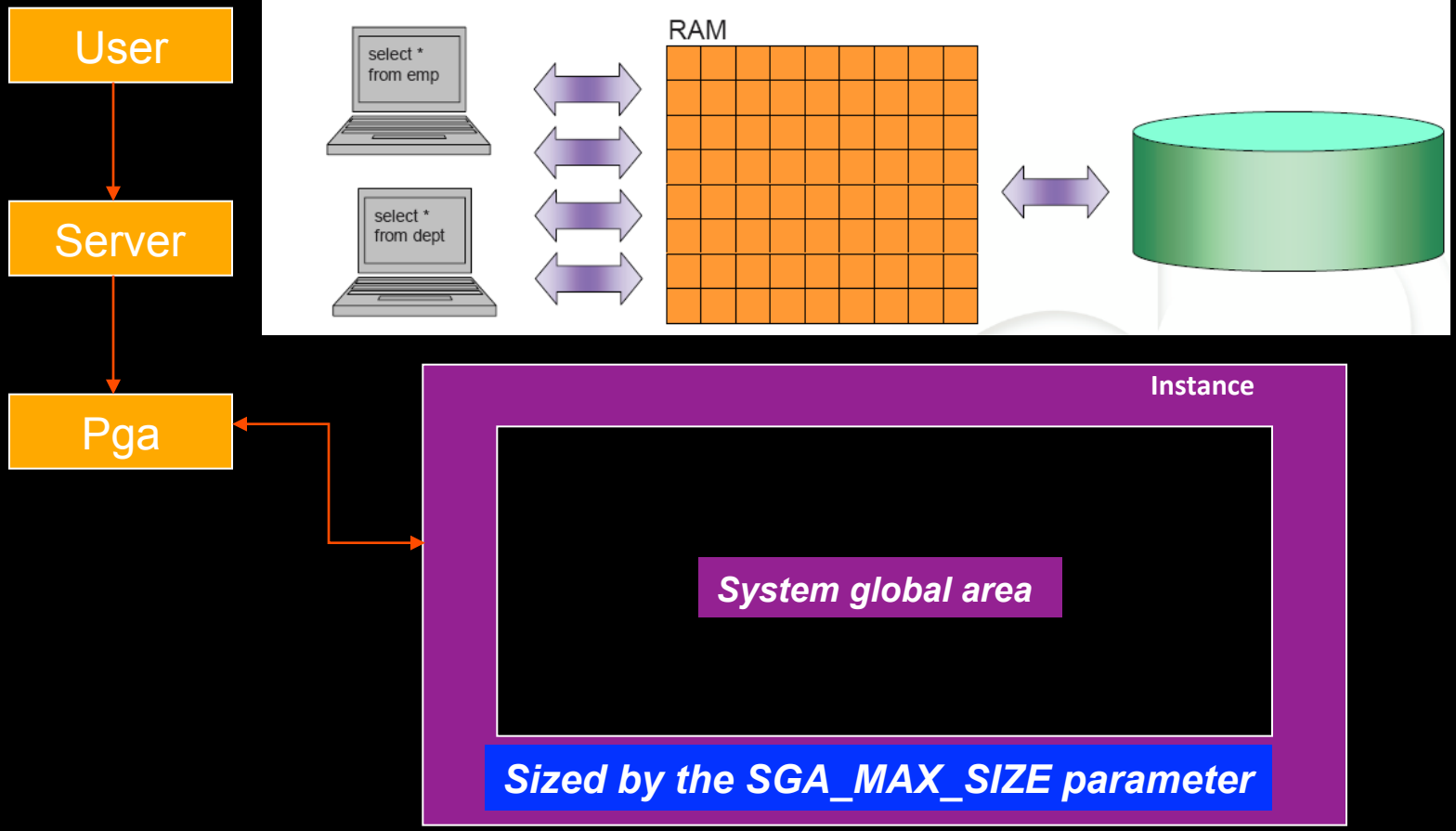
Memory reserved for each user process connecting to an Oracle database

Allocated when a process is created

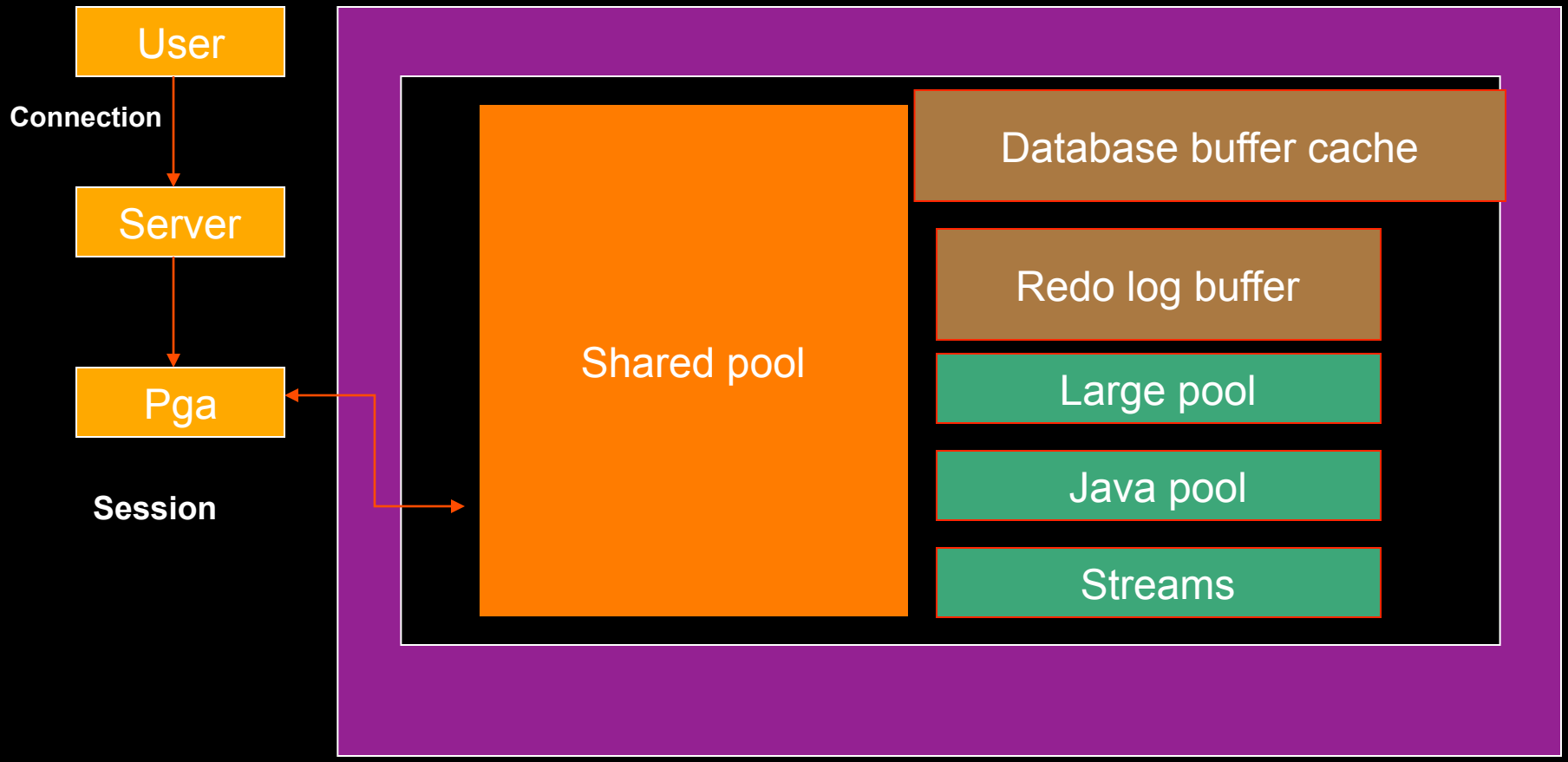
Deallocated when the process is terminated

Used by only one User process

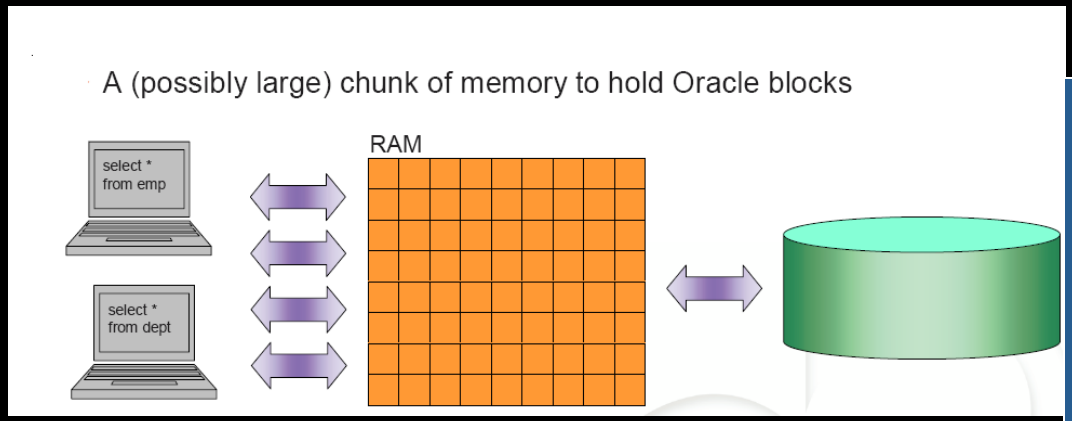
System Global Area

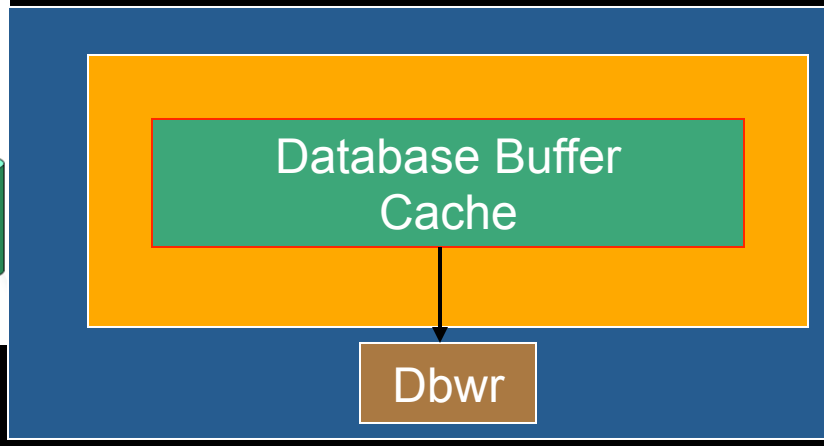


System Global Area



Database Buffer Cache





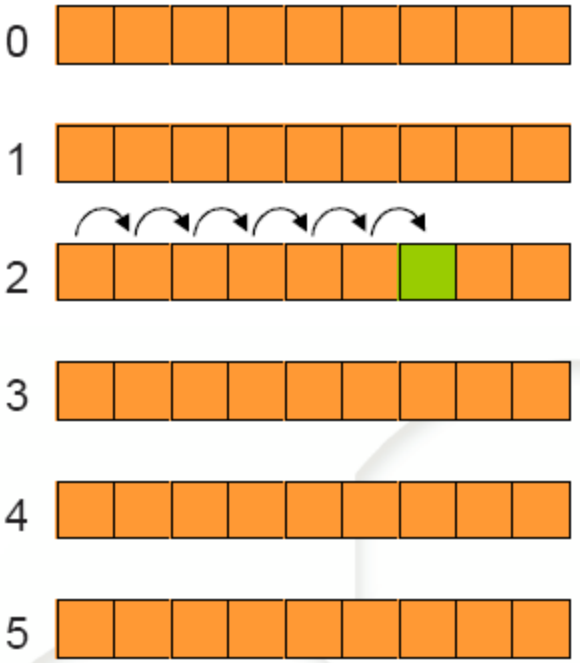
Before any database block can be used ,

it must be physically read from disk
and placed into the database buffer cache

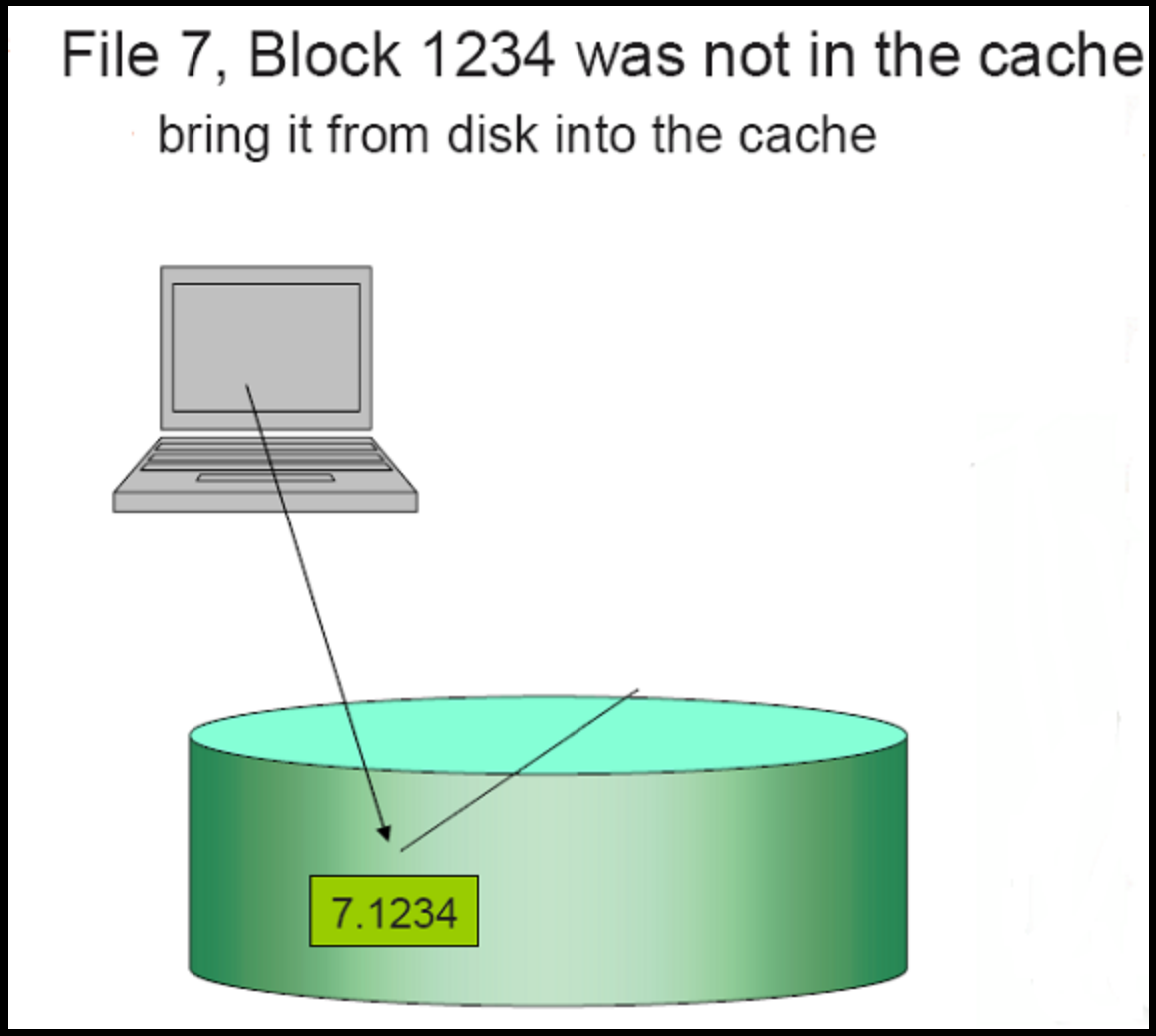
Database Buffer Cache

I need File 7, Block 1234

is it already in the cache ?

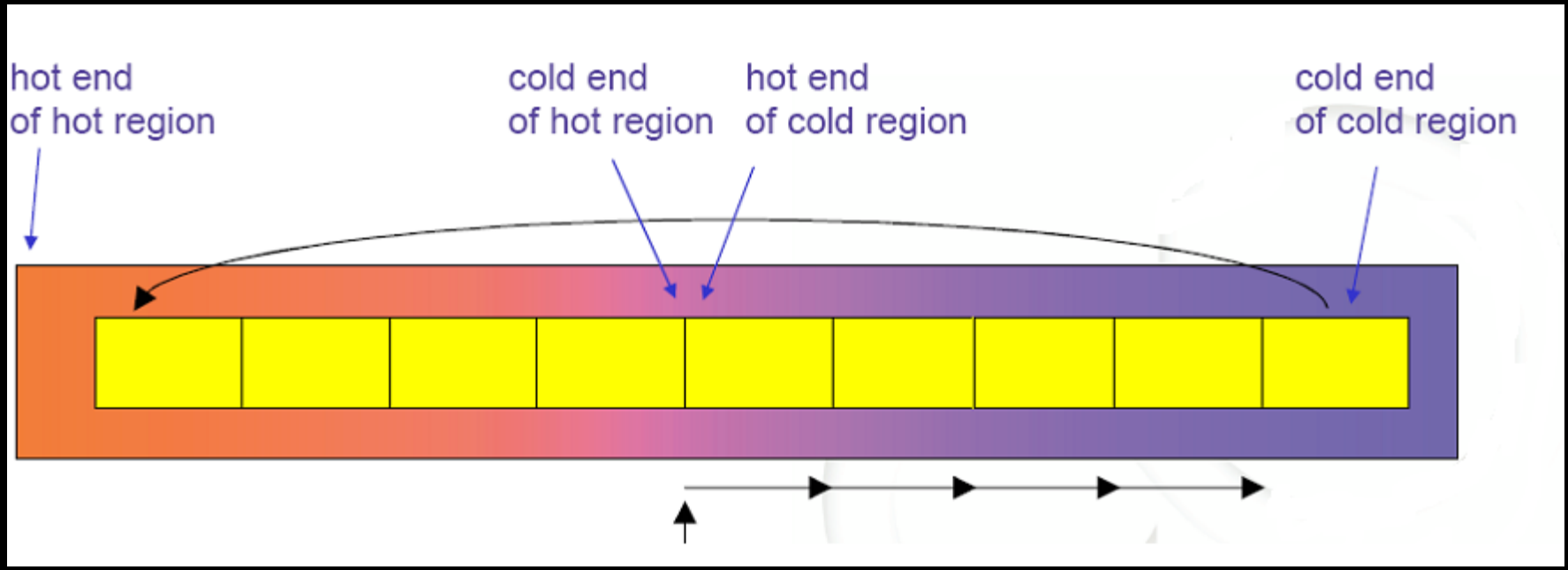


Database Buffer Cache



Database Buffer Cache

Uses MRU—LRU Mechanism



Database Buffer Cache

Database Buffer Cache has 4 different buffers

Free Buffer

Pinned Buffer

Dirty Buffer

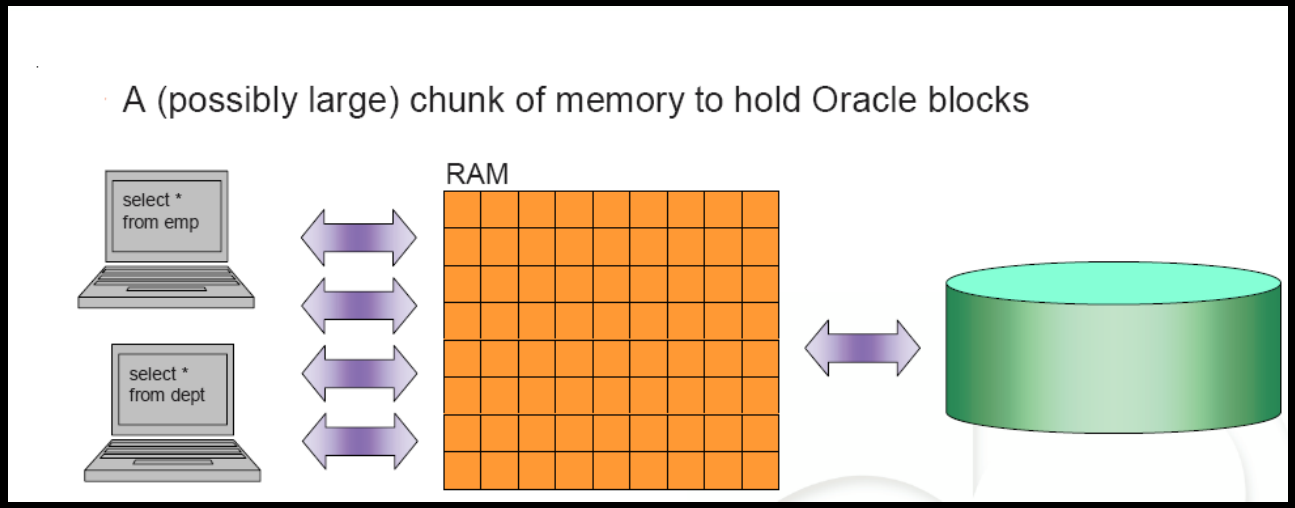
Clean Buffer

DB_CACHE_SIZE

V$BH

Shared pool

Shared pool



Library cache

Data Dictionary cache

SHARED_POOL_SIZE

Helps in Parsing (Only Syntactic)

Stores most recently used SQL and PL/SQL statements

Enables the sharing of commonly used statements Algorithm

Consists of two structures:

Shared SQL area

Private PL/SQL area

V$LIBRARYCACHE

ROW CACHE

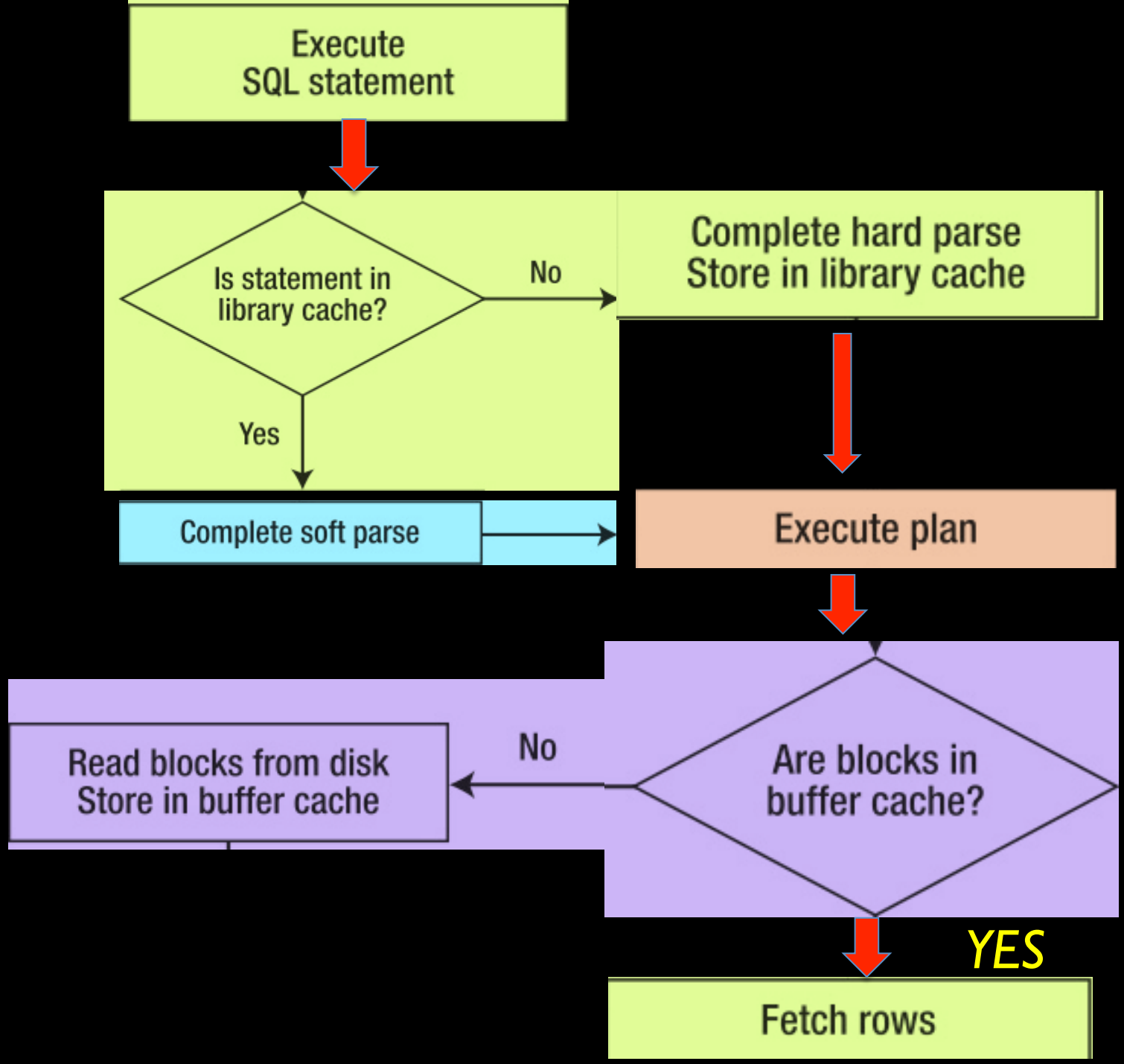
Helps in Parsing (Only Semantic)

Collection of the most recently used definitions in the database

Includes information about database files, tables,indexes, columns, users, privileges, and other database objects

V$ROWCACHE

Steps involved for a sql query


Execute
SQL statement
Is statement in
library cache?
No
Yes
Complete hard parse
Store in library cache
Complete soft parse
Execute plan
Are blocks in
buffer cache?
No
Read blocks from disk
Store in buffer cache
YES
Fetch rows

Redo log Buffer

Records all changes made to the database data blocks

Primary purpose is recovery

Changes recorded within are called redo entries

Redo entries contain information to reconstruct or redo changes

Background processes

Smon

Pmon

Dbwr

Lgwr

Ckpt

THE REAL HERO

Database

How Oracle store Data

Control Files

BRAIN OF THE Database

Control File Contents

Database name and identifier

Names and locations of data files and log files

Checkpoint information

Data files

Data files

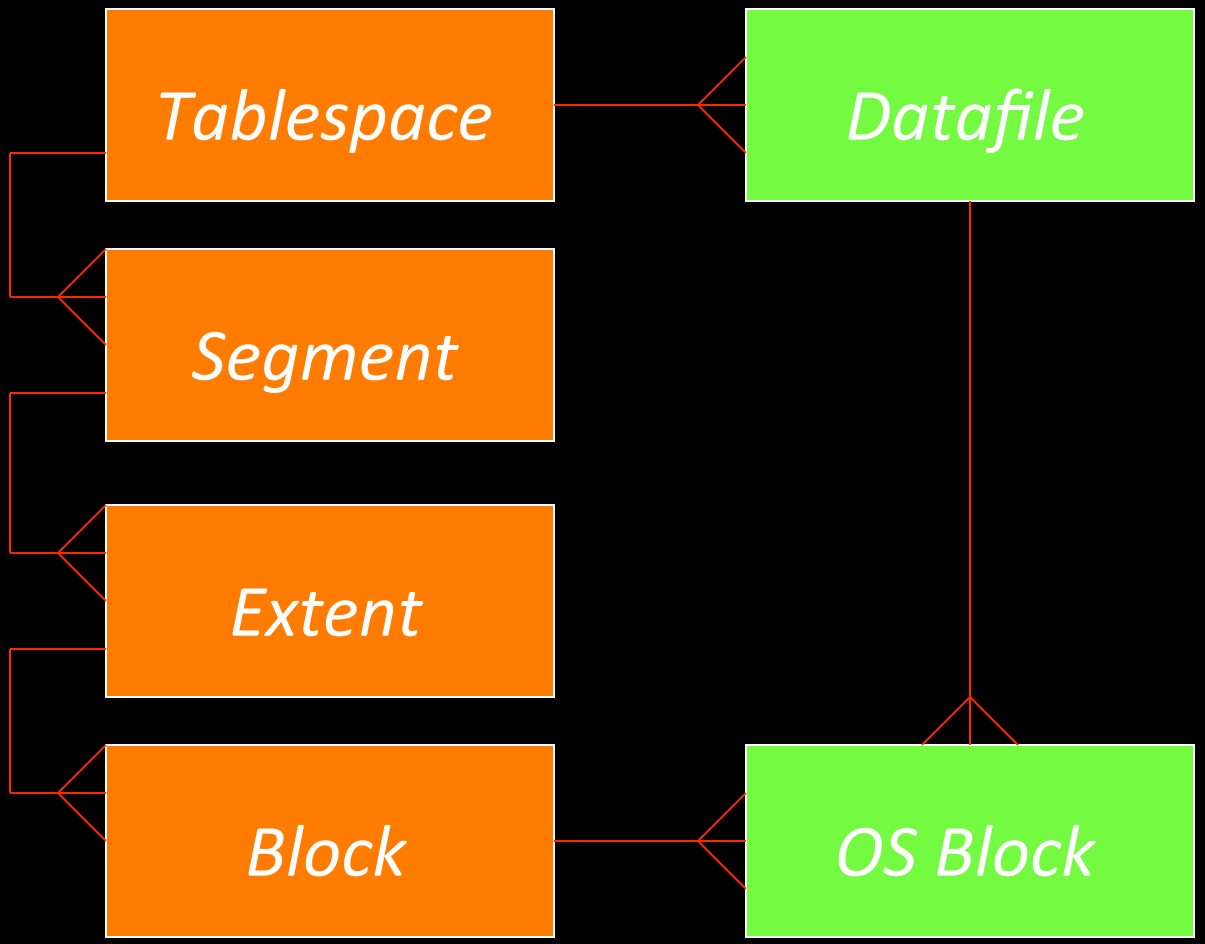
Stores actual Data

Tablespaces

Segments

Extents

Data blocks

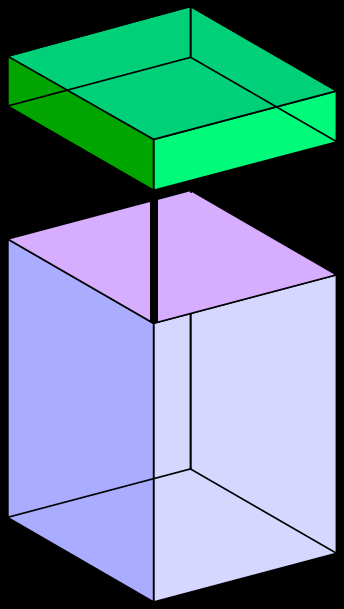

Tablespace
Datafile
Segment
Extent
Block
OS Block

Block Utilization Parameters

Data Block Management

Two methods are available for managing data blocks:

- Automatic Segment-Space Management
- Manual Management



INITRANS

MAXTRANS

PCTFREE

PCTUSED

FREE LISTS

Redo Log Files

How Redo Log Files Work

Record all changes made to data

Used for recovery

Minimum 2 Redo log files are required

Redo log files are used in a cyclic fashion.

How Redo Log Files Work

When a redo log file is full, LGWR will move to the next log group ,called a log switch

Checkpoint operation occurs and Information written to the control file

Archive files

Archive files are copy of Log files

EXPLAIN PLAN / SQL TRACE

SET AUTOTRACE ON

SET AUTOTRACE TRACEONLY EXPLAIN

```
explain plan for select emp_id from emp;
```

```
select * from table(dbms_xplan.display);
```

DBMS_XPLAN. DISPLAY Examples

```
SELECT * FROM TABLE
(dbms_xplan.display('PLAN_TABLE','abc','BASIC'));
```

```
SELECT * FROM TABLE
(dbms_xplan.display('PLAN_TABLE','abc','TYPICAL'));

SELECT * FROM TABLE
(dbms_xplan.display('PLAN_TABLE','abc','ALL'));
```

BASIC – Displays the minimum information

TYPICAL – Displays the relevant information in the plan and predicate information ,parallel

ALL – All of typical including projections, alias

```
EXPLAIN PLAN
set statement_id = 'X1'
FOR  SELECT d.deptno , d.dname  ,
COUNT(*)
AS count_employees  ,
SUM(e.sal) AS
sum_salaries  FROM dept d  , emp e
WHERE d.deptno = e.deptno
GROUP BY  d.deptno  , d.dname;
```

```
SELECT plan_table_output FROM    TABLE

( DBMS_XPLAN.DISPLAY ('PLAN_TABLE',’X1',’ALL') )
```

Tkprof

Initialisation parameters

- timed_statistics = TRUE
- SQL_TRACE = TRUE
- user_dump_dest = '<directory name>‘
- trace_file_identifier = '<string>'

Enabling and Disabling SQL Trace

At the instance level:

```
SQL_TRACE = TRUE
ALTER SYSTEM SET SQL_TRACE=FALSE;
```

At the session level:

```
Alter session set SQL_TRACE = true
Alter session set tracefile_identifier= 'sense'
```

TKPROF performance

TKPROF --Converts the trace file into a readable format.

TKPROF <trace filename> <output filename>

TKPROF

Set auto trace on

call	count	cpu	elapsed	disk	query	current	rows
Parse	1	0.00	0.00	0	0	0	0
Execute	3117	0.83	0.92	0	0	0	3113
Fetch	3117	7.03	9.88	40	55902	0	3105
total	6235	7.86	10.80	40	55902	0	6218

TKPROF output

```
********************************************************************************
*
count    = number of times OCI procedure was executed

cpu      = cpu time in seconds executing

elapsed  = elapsed time in seconds executing

disk     = number of physical reads of buffers from disk

query    = number of buffers gotten for consistent read

current  = number of buffers gotten in current mode (usually for update)

rows     = number of rows processed by the fetch or execute call
********************************************************************************
*
```

Explain plan

set autotrace on

Set autotrace on explain

Set autotrace on statistics

Set autotrace traceonly explain

Set autotrace traceonly statistics

Set autotrace traceonly

Set autotrace off

V$views

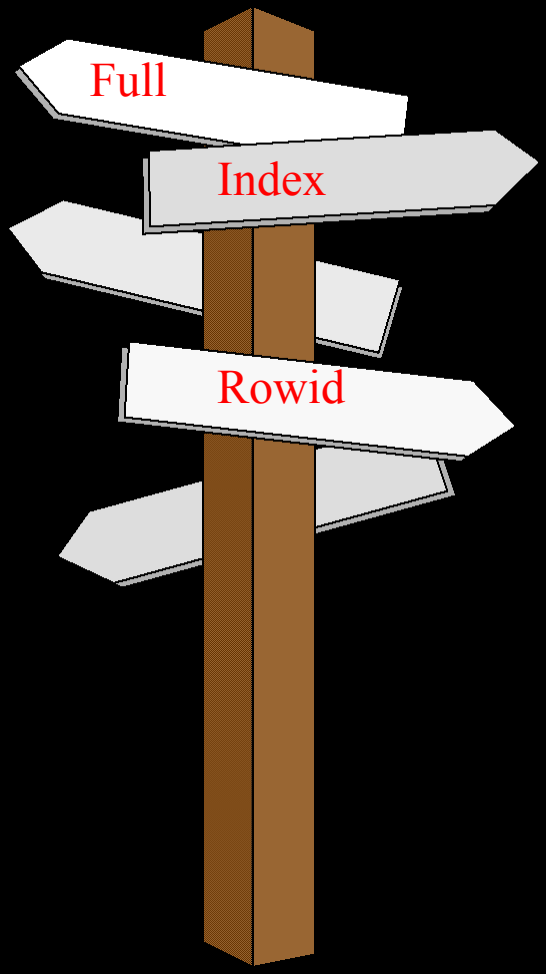
User/Session
V$LOCK
V$OPEN_CURSOR
V$PROCESS
V$SORT_USAGE
V$SESSIONT/P
V$SESSTAT T/P
V$TRANSACTION
V$SESSION_EVENT
V$SESSION_WAIT

V$SQL
V$SQLAREA
V$WAITSTAT
V$SESSION
V$SESS_IO
V$SQL_PLAN
V$SQL_PLAN_STATISTICS
V$OPEN_CURSOR
V$SESSION_LONGOPS
V$SYSSTAT
V$SESSION_EVENT
V$SESSION_WAIT
V$SYSTEM_EVENT
V$FILESTAT

Indexes

Access paths

- *Full table scan*
- *Sample scans*
- *Rowid scans*
- *Index full scans*
- *Index unique scan*
- *Index range scan*
- *Index fast full scans*
- *Index joins*
- *Bitmap joins*
- *Cluster scans*
- *Hash scans*



So what's a full-table scan?

- *Process of reading all blocks of a database table sequentially*

- *Full-table scans may cause of bad query performance*

Why ---Full table scan

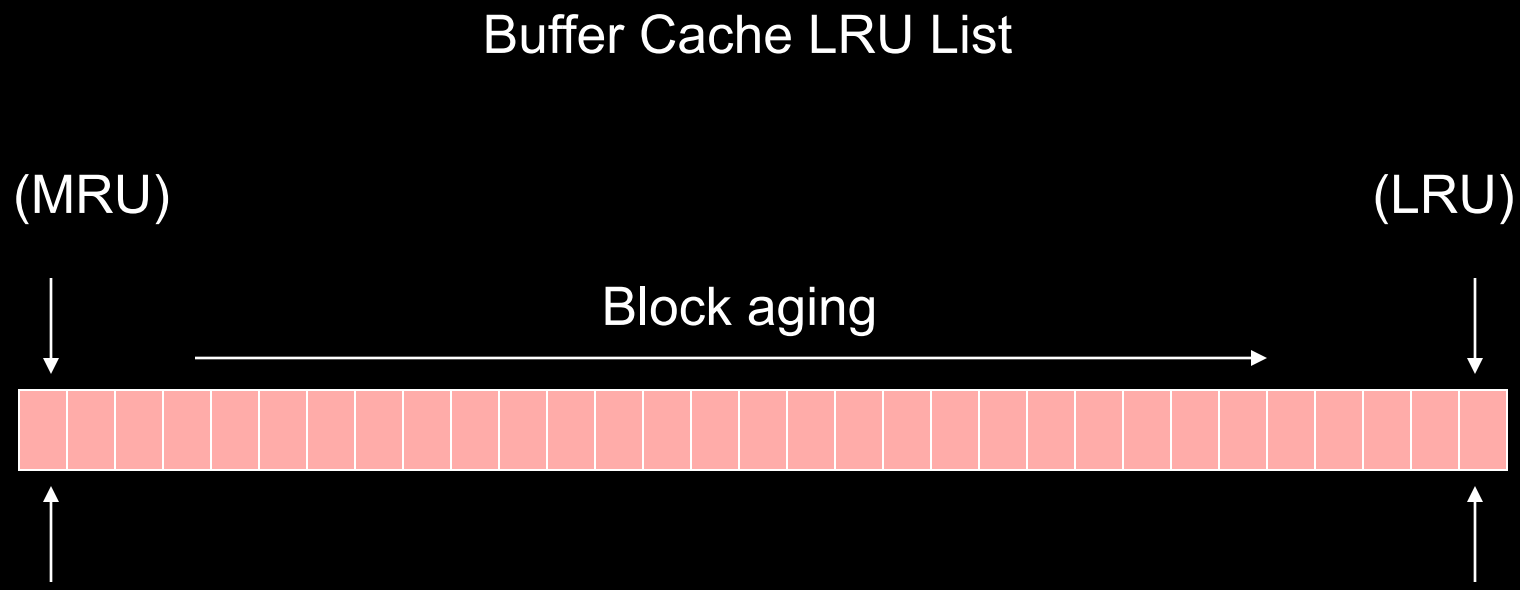
Missing indexes on large tables

A poor indexing strategy on large tables

Missing table or index statistics

Why Full table scan is bad

FTS effects on buffer cache



INDEX *ACCESS*

Last few pages of any book

1. *B* tree Index*
2. *Bitmap Index*
3. *Reverse key Index*
4. *Function Based Index*
5. *Bitmap join Index*
6. *Partitioned Index*
7. *Virtual Index*
8.

- *Invisible Indexes*

Concept of an Index

An index contains a pointer for each table row (rowid)

To access the data rows quickly

Create index emp_idx (eno) ;

Create index emp_idx (eno,ename,sal) ;

Create unique index emp_idx (eno);

Different Index scans

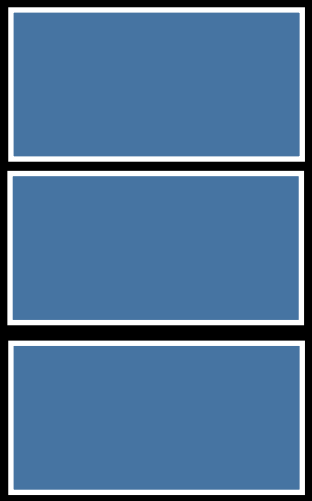
Unique scan	*1 record*
Range Scan	*More than 1 record*
skip scan	
fast full scan	*count (*)*
index-join	*more than 1 table*

How Oracle store Index data

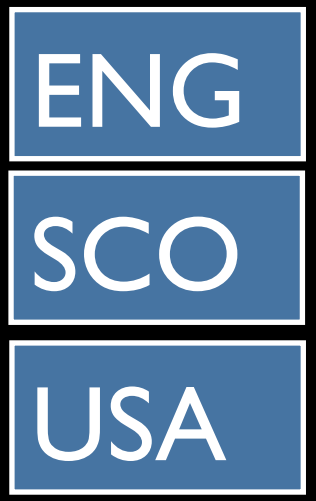
Insertion

Inserting 'ENG','SCO' and 'USA'

Before



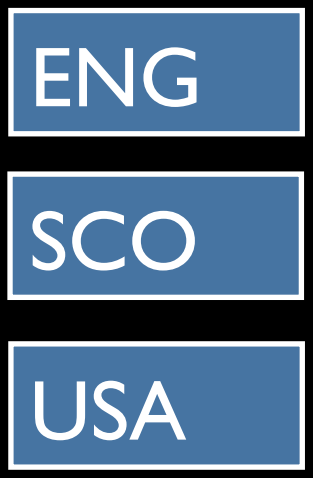
After



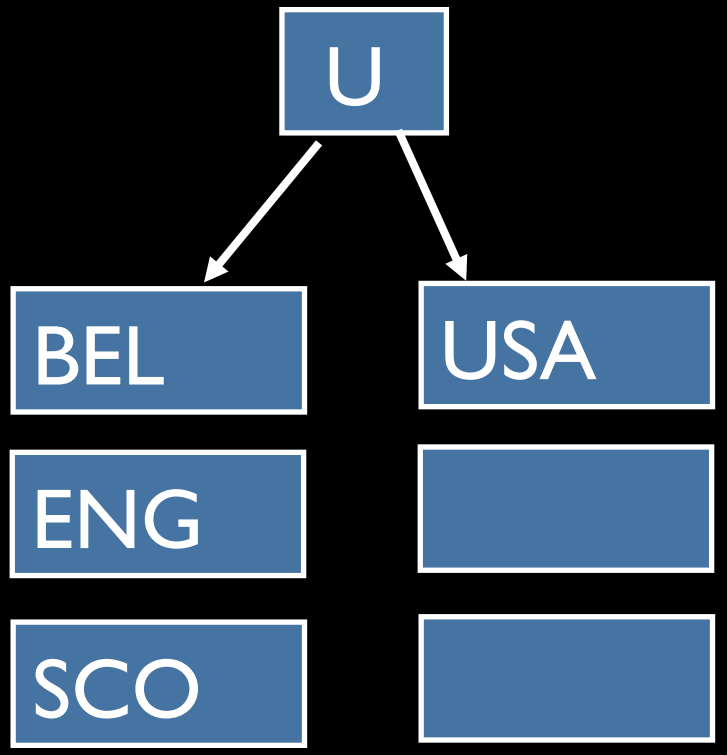
Insertion

Inserting 'BEL'

Before



After



BITMAP INDEX

For column that has a low cardinality
(few distinct values)

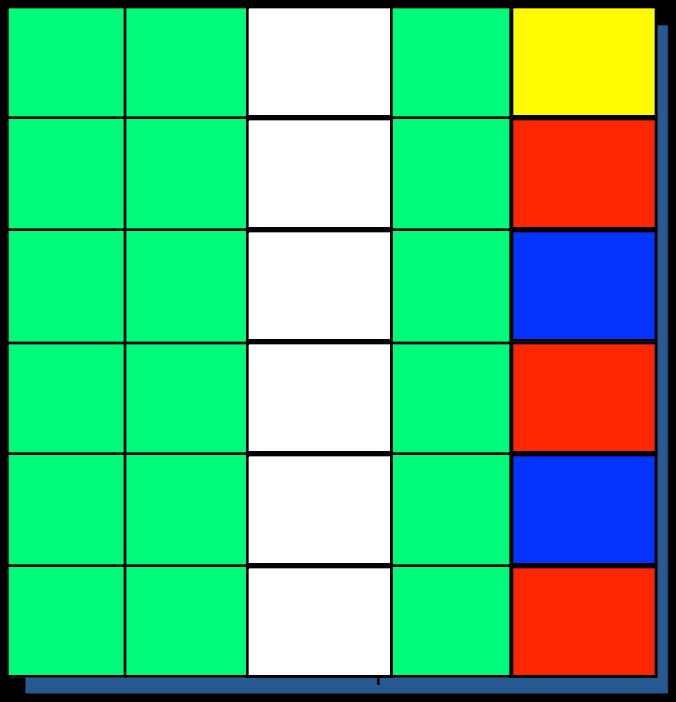
Examples *Male / female*
Yes / No
Good / bad
Colours

BITMAP INDEX

Create bitmap index emp_idx (Gender) ;

Bitmap Indexes

*Select * from emp WHERE Gender = 'M' and Color = 'RED';*



Red: <0,1,0,1,0,1>

Mal <1,0,1,1,0,1>

REVERSE Key Indexes

KEY	ROWID
1257	0000000F.0002.0001
2877	0000000F.0006.0001
4567	0000000F.0004.0001
6657	0000000F.0003.0001
8967	0000000F.0005.0001
9637	0000000F.0001.0001
9947	0000000F.0000.0001
. . .	. . .

EMPLOYEE_ID	LAST_NAME ...
7499	ALLEN
7369	SMITH
7521	WARD ...
7566	JONES
7654	MARTIN
7698	BLAKE
7782	CLARK

create unique index i1_t1 ON t1(c1) REVERSE ;

alter index i2_t1 REBUILD REVERSE ;

Function based index:

```
Create index emp_idx on emp
(UPPER(ename));
```

Finding index usage

I

select index_name,monitoring,used,start_monitoring, end_monitoring from v$object_usage;

Finding index usage

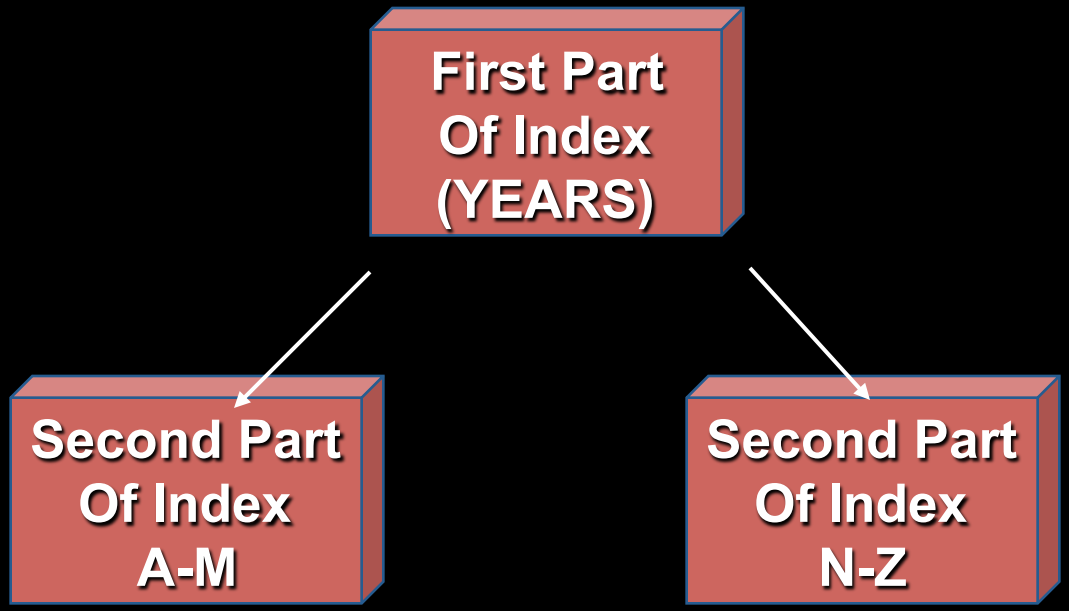
ALTER INDEX <index> MONITORING USAGE. ;

ALTER INDEX <index> NOMONITORING usage ;

Index Skip Scan

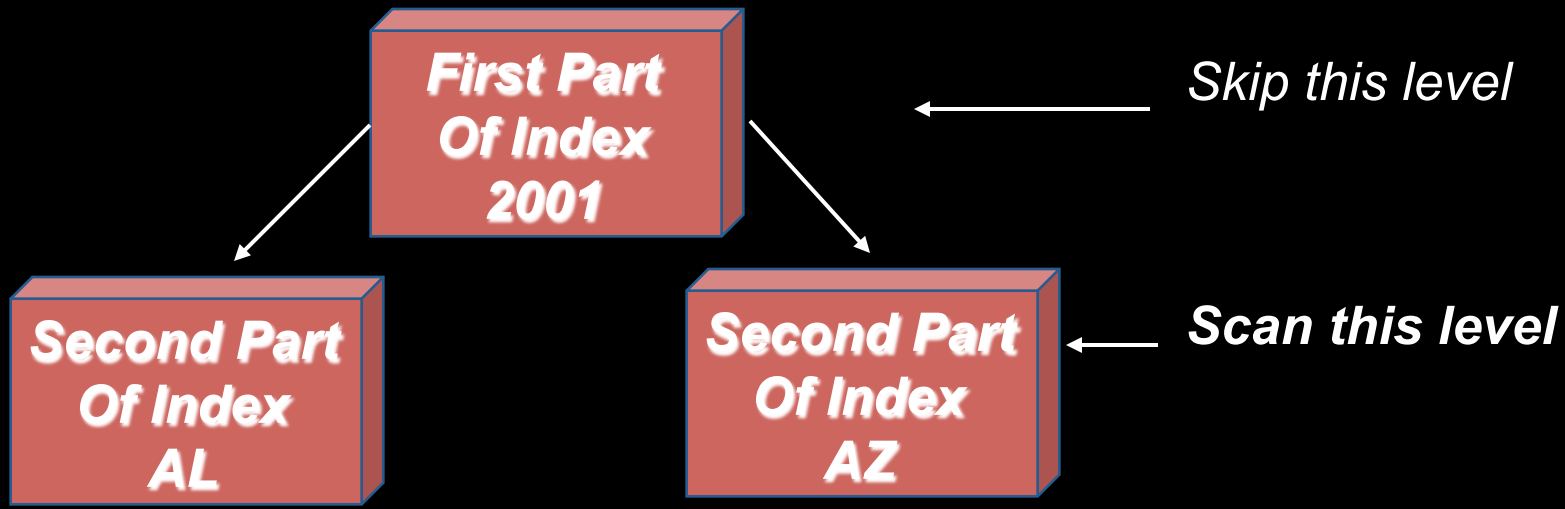
Index Skip Scan

Create index year_state_idx on test2(year, state);



Index Skip Scan

```
SELECT COUNT (*) FROM TEST2
 WHERE STATE= 'AL'
```



```
create index IX_SS on EMP(DEPTNO,SAL);

select /*+ index_ss(EMP IX_SS) */ * from
emp where SAL < 1500;
```

Views for Indexes

User_indexes
User_ind_columns
V$object_usage
Index_stats

Index Maintenance

Alter index emp_idx rebuild ;

WHAT IS

REBUILD

Removing

Un necessary

Why Should I Rebuild the Index

Why should I rebuild Index

Oracle does not delete Index entries of Records that are deleted by the user.

So we rebuild ,to reclaim the space occupied by the deleted rows

Performance will become worse as oracle may choose Full table scan instead of Index scan

Benefits of Rebuilding Index

Space occupied by the deleted rows is reclaimed

Performance will become better as Oracle Chooses Index scan after rebuild .

When Should I Rebuild Index

Height is > 4

Del_lf_rows is > 20 %

Poor clustering factor

Checking Clustering Factor

Execute dbms_stats.gather_table_stats(scott, 'EMP', cascade=>true);

Cascade is

Gather_table_stats + Gather_index_stats
In a single query

```
Select   blocks,  num_rows     from
         user_tables
Where    table_name = 'BTREETABLE';
```

```
Select  index_name,  blevel,  clustering_factor
    from        user_indexes
where table_name = 'BTREETABLE'
```

Blevel ---Btree level

"Good" CF --- If CF is closer to Blocks in table

"Bad" CF --- If CF is closer to Rows in table

Bitmap Join Index

```
CREATE BITMAP INDEX empdept_bji
    ON employees(d.deptno)
FROM employees e, departments d
WHERE d.deptno = e.deptno;
```

Invisible Indexes

```
CREATE INDEX IND1 ON luck(object_id)
 INVISIBLE;

ALTER INDEX ind1 INVISIBLE ;
```

Optimizer does not consider this index:

Optimizer can consider this index:

ALTER INDEX ind1 VISIBLE;

Select /+ index(luck ind1) */ * from luck where object_id <1000;*

Index Organised Tables

Heap-organised table

```
CREATE TABLE team
(
    team_key VARCHAR2(3),
    team_name
    VARCHAR2(50),
    country_key
    VARCHAR2(3)
    CONSTRAINT team_pk
    PRIMARY KEY
(team_key);
)
ORGANIZATION HEAP;
```

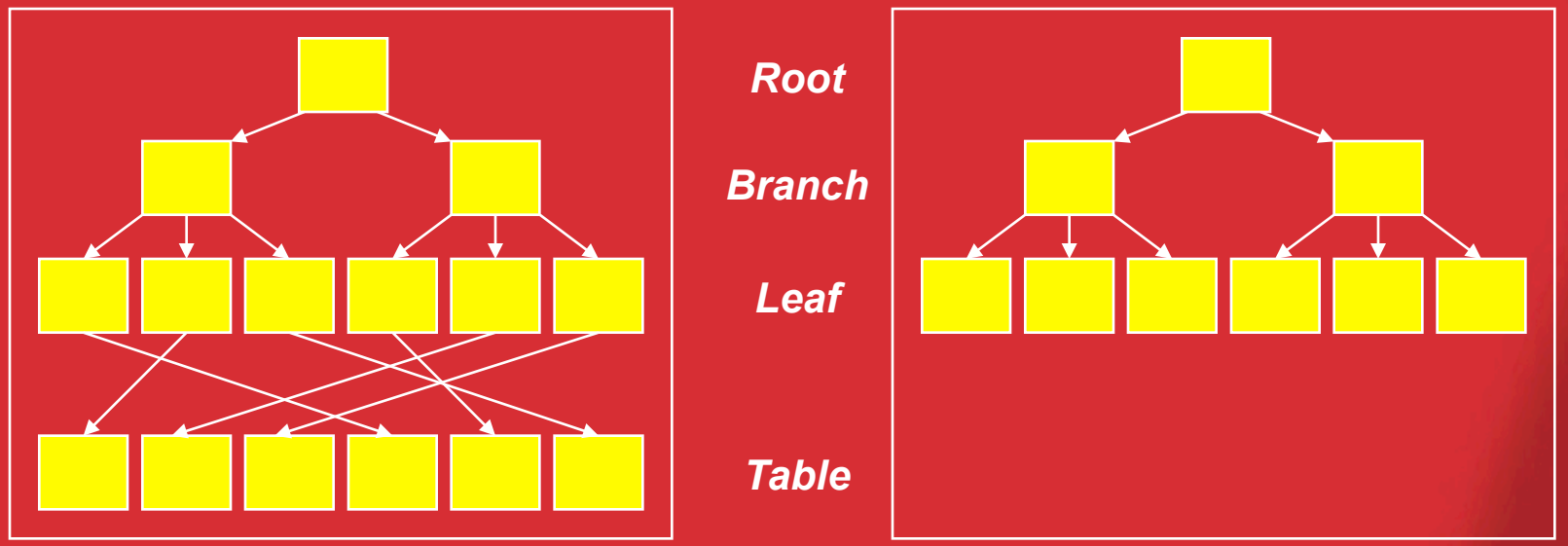
Index-organised table

```
CREATE TABLE team
(
    team_key VARCHAR2(3),
    team_name
    VARCHAR2(50),
    country_key
    VARCHAR2(3)
    CONSTRAINT team_pk
    PRIMARY KEY
(team_key);
)
ORGANIZATION INDEX;
```

Index Organised Tables

Heap-organised table

Index-organised table



Index Organised Tables

You need to have primary key

Don't use IOT if data volume is high

You can also create indexes for other columns

Don't create more than 2 indexes on IOT

Partitioned Indexes

Local Indexes:

single index is created of the partitioned table.

Global Indexes:

– Single Index is created For all Partitions

Partitioned Indexes examples

```
create index idx_local on emp(id) local ;
```

```
create index idx_global on emp(id) global ;
```

Virtual Indexes examples

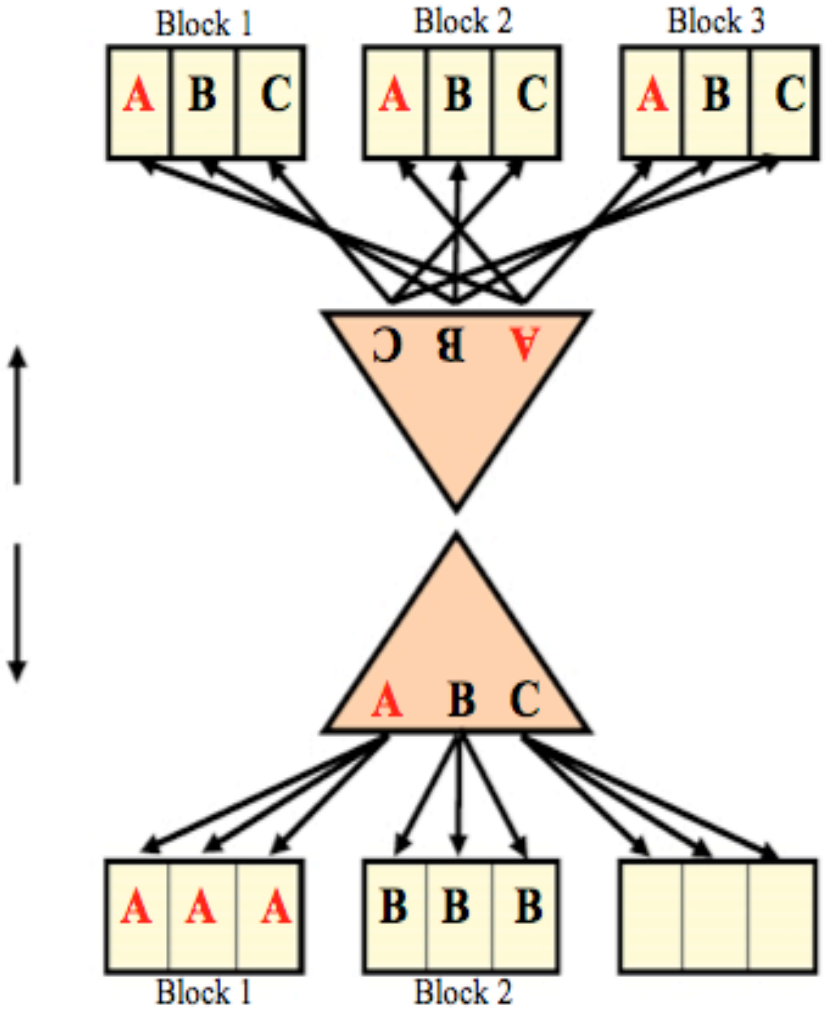
```
create index emp_idx on emp (object_id)
nosegment;
```

```
alter session set "_use_nosegment_indexes"=true;
```

Does not occupy space in the Database

Index Clustering Factor

Must read all blocks to retrieve all As



POOR CF

GOOD CF

Only need to read one block to retrieve all As

“Good” CF generally has value closer to blocks in table

“Bad” CF generally has a value closer to rows in table –Oracle does not use index

What to do in case of poor CF

Create Histograms:

Histogram Stores Data distribution information

exec dbms_stats.gather_table_stats (user,'t l',method_opt=>'for all columns size l' , cascade=>TRUE);

Always specify more than 1 .At least 10

Solution for Poor Clustering Factor

Increase freelists (not more than 12)
Your tablespace should be in MSSM

Optimizer

Why Do You Need an Optimizer?

Find the most efficient mechanism for executing any SQL statement

JOB OF OPTIMIZER

Two main components:

Query Transformations

Access Path Selection

Example:

```
select * from dept
where deptno in (select deptno from emp
   where job = 'CLERK')


select d.* from dept d,
(select distinct deptno from emp where job
   = 'CLERK') e
where d.deptno = e.deptno
```

Access Path Selection

For each table, choose the access path (table scan, index scan, etc)

For each join, choose the join method (nested-loop, sort-merge, hash, etc)

Choose the join order for the tables

Takes care of

Selectivity

Cardinality

COST

Optimizer Parameters

CURSOR_SHARING: SIMILAR, EXACT, FORCE
DB_FILE_MULTIBLOCK_READ_COUNT
PGA_AGGREGATE_TARGET
STAR_TRANSFORMATION_ENABLED
RESULT_CACHE_MODE: MANUAL, FORCE
RESULT_CACHE_MAX_SIZE
RESULT_CACHE_MAX_RESULT
RESULT_CACHE_REMOTE_EXPIRATION
OPTIMIZER_INDEX_CACHING
OPTIMIZER_INDEX_COST_ADJ

Optimizer Parameters

OPTIMIZER_FEATURES_ENABLED
OPTIMIZER_MODE: ALL_ROWS, FIRST_ROWS, FIRST_ROWS_n
OPTIMIZER_CAPTURE_SQL_PLAN_BASELINES
OPTIMIZER_USE_SQL_PLAN_BASELINES
OPTIMIZER_DYNAMIC_SAMPLING
OPTIMIZER_USE_INVISIBLE_INDEXES
OPTIMIZER_USE_PENDING_STATISTICS

Gathering Statistics

Used to determine:

Table access cost
Join cardinality
Join order

Table Statistics:

Number of rows & blocks & empty blocks

Average row length

Average free space per block

(Number of chained rows (CHAIN_CNT)

Index Statistics:

B-tree level*
Distinct keys

No of leaf blocks
Clustering factor

(AVG_LEAF_BLOCKS_PER_KEY)
(AVG_DATA_BLOCKS_PER_KEY

Column statistics

No of distinct values, no of nulls,
average length,
min, max,
Histograms
(data distribution when the column data is skewed)

user_tables

user_all_tables

user_indexes

user_XXX_partitions

user_XXX_subpartitions

user_tab_cols

user_tab_col_statistics

user_(sub)part_col_statistics

GATHER_TABLE_STATS

GATHER_INDEX_STATS

GATHER_SCHEMA_STATS

```
Execute dbms_stats.gather_table_stats( scott,
'EMP', cascade=>true );

Execute dbms_stats.gather_index_stats
( winner, 'emp_idx' );

Execute dbms_stats.gather_schema_stats
(ownname => 'WINNER');
```

Gathering statistics for Partitioned Table

```
exec dbms_stats.gather_table_stats
( owner=>'winner',Tabname=>'emp',
partname=>'23_May_2008',
granularity=>'auto');
```

```
exec dbms_stats.gather_table_stats('user',
'emp', GRANULARITY => 'PARTITION');
```

```
exec dbms_stats.gather_table_stats('user',
'emp', GRANULARITY => 'all');
```

```
BEGIN
DBMS_STATS.delete_table_stats ('winner','emp');
DBMS_STATS.lock_table_stats ('winner','emp');
END;
```

Inform Oracle to use to one day old statistics

```
dbms_stats.restore_schema_stats
(ownname=> 'winner', as_of_timestamp =>
systimestamp – 1,
force       => TRUE)
```

Hints

Hints

Syntax must be correct or the Hint will be ignored, and no error message is issued.

There is a 255 character limit to Hints.

When using an alias for a table in the statement, the alias needs to be in the Hint.

Hints Look Like?

DELETE /+ hint [text] */*

INSERT /+ hint [text] */*

SELECT /+ hint [text] */*

UPDATE /+ hint [text] */*

SELECT /+ FULL(table_name) */ * from emp ;*

SELECT /+ INDEX(table_name index_name1 index_name2...) */*

ORDERED - Force the driving table

SELECT /+ ORDERED */ column1, column2..FROM table1, table2*

ALL_ROWS - best throughput.

Select /+ ALL_ROWS */ .....*

The ALL_ROWS hint usually suppresses an index

FIRST_ROWS - best response time.

Select /+ FIRST_ROWS */ .....*

The FIRST_ROWS hint usually forces an index

/+ROWID*/ Use ROWID access method*

/+USE_NOCACHE*/*

Don't put the data in the buffers

/+USE_CACHE*/*

Don't put the data in the buffers

```
select /*+ parallel(e, 4) parallel(b, 4) */
e.ename,hiredate,b.comm
from   emp e, bonus b
where    e.ename =b.ename
```

Hints for Joins

Nested loop	*Use_nl*
Sort merge	*Use_merge*
Hash	*Use_hash*

When Oracle ignores hint

SELECT /+ APPEND */ …*

SELECT /+ FIRST_ROWS */ zip_code from emp group by zip_code;*

SELECT /+ USE_HASH(X) */ * FROM inv_master WHERE inv_type = 'G';*

```
SELECT /*+ INDEX_DESC(TM TEAM_PK) */
player_name,coach, date_joined from roster
JOIN team t USING (tm_id)
where t.effdt = (select max(effdt) From  team
TM where  tm.tm_id = t.tm_id)
```

```
SELECT /*+ INDEX(books xisbn) */ * FROM
 books b WHERE b.isbn = :ISBN;
```

INDEX_JOIN

informs to use an index join as an access path

INDEX_FFS *Does a fast full index scan*

INDEX_SS *Does an index skip scan*

NO_INDEX

Does not allow index scan and does full table scan

USE_NL

Joins the specified table using a nested loop join

NO_USE_NL

Does not use nested loops to perform the join

USE_HASH

Joins the specified table using a hash join

NO_USE_HASH

Does not use HASH JOIN to perform the join

USE_MERGE

Joins the specified table using a merge join

NO_USE_MERGE

Does not use MERGE JOIN to perform the join

Append

Direct path insert

NOAPPEND

Regular insert

Parallel processing

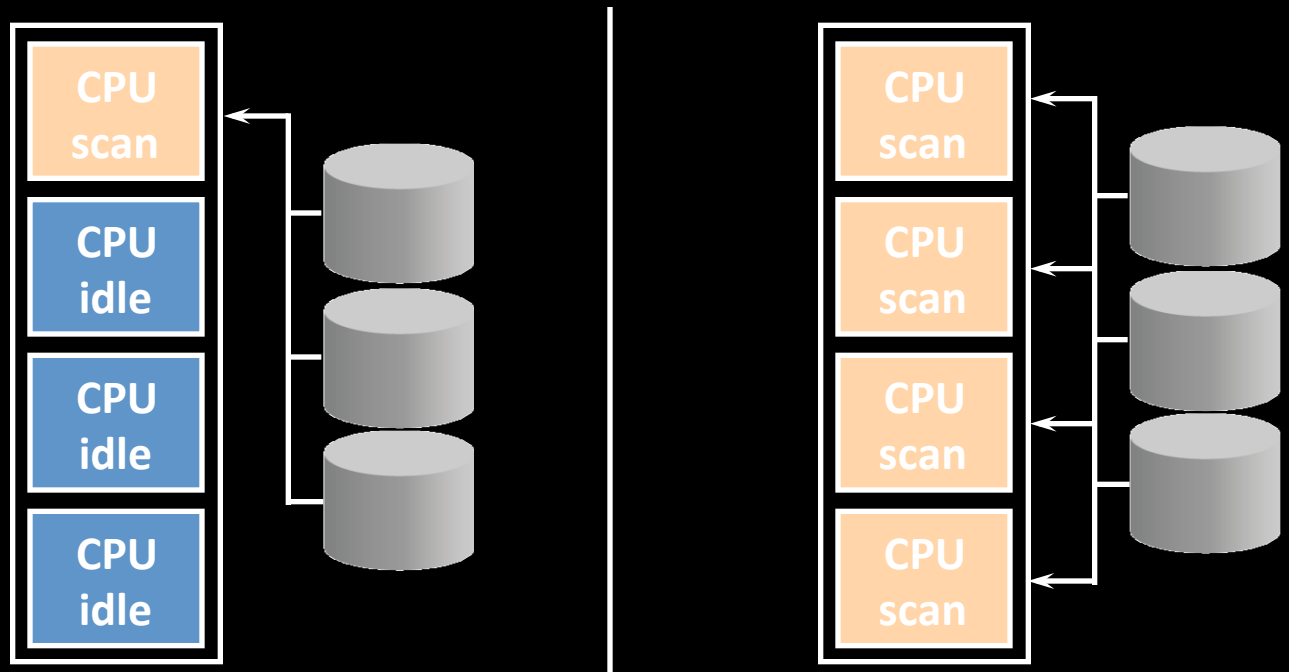
The Parallel Query Option

Multiple Server Processes can work together

Improves data-intensive and CPU intensive operations.

Available only when a Full table scan or a Sort operation is being performed.

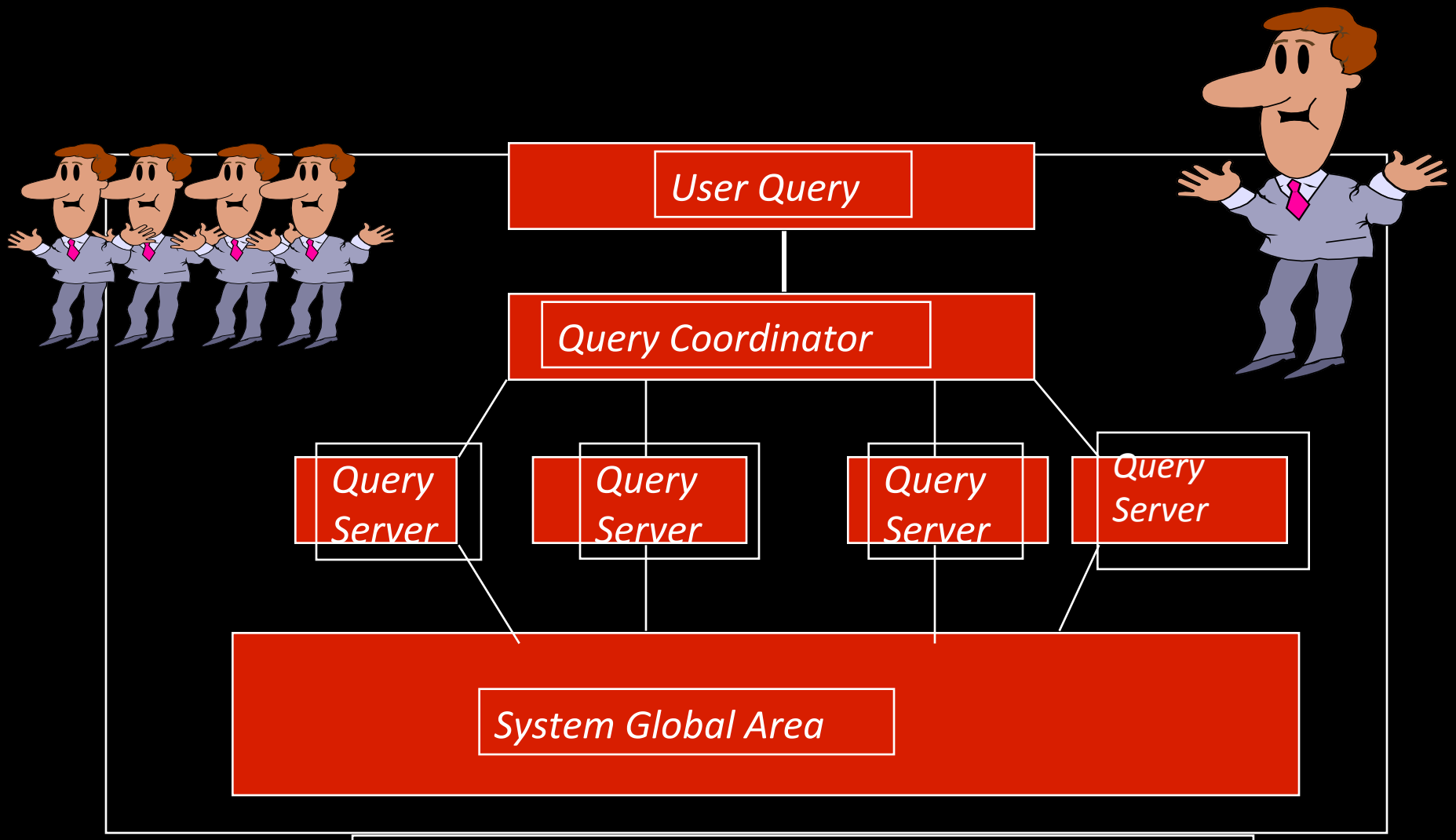
Benefits of Parallel Execution



Server without Parallelism

Server with Parallelism

Parallel Query Process

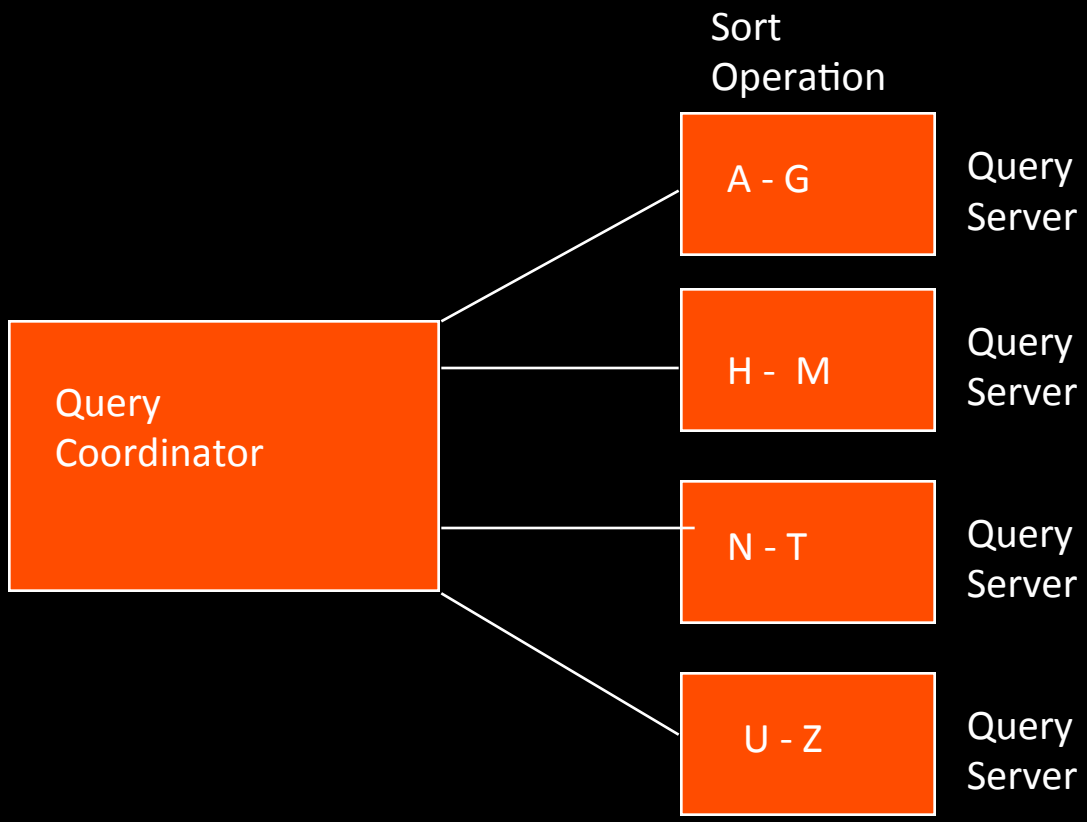


Query coordinator—Server process

Degree of Parallelism

Select degree from user_tables where table_name='EMP'

An example of a sort with the degree of parallelism set to 4



Parallel Query Operations

Access methods:

Index Range Scans

Various SQL operations:

Group by ,Order by, Not in, Exists, In, Distinct, Union,Union all, MinusTable Scans, Index Full Scans Partitioned ,Intersect,Cube, Rollup, Aggregates

Join methods:

Nested loop, Sort Merge,Hash, Star Transformation,

partition-wise join

Parallel DDL Operations

The parallel DDL statements for non-partitioned tables and indexes are:

CREATE INDEX
CREATE TABLE … AS SELECT
ALTER INDEX … REBUILD

The parallel DDL statements for partitioned tables and indexes are:

- *CREATE INDEX*
- *CREATE TABLE ... AS SELECT*
- *ALTER TABLE ... MOVE PARTITION*
- *ALTER TABLE ... SPLIT PARTITION*
- *ALTER TABLE ... COALESCE PARTITION*
- *ALTER INDEX ... REBUILD PARTITION*
- *ALTER INDEX ... SPLIT PARTITION*

Parallel Query Use

With Hints

```
Select /*+ Full(table) Parallel(table 3) */
```

Creating tables to use the Parallel Option

```
Create Table emp  (eno  Number, tname
Varchar2(10))

Parallel 2 ;
```

Parallel DML - V8

Example (Parallel in subquery):

```
insert /*+ PARALLEL (t1,2) */ into   t1
(select /*+ PARALLEL (t2, 6) */   c1, c2, sum
(c3)   from t2   group by c1,c2);
```

If you don't need parallel

Alter table emp noparallel ;

Select /+ noparallel */ * from emp ;*

JOINS

A join defines the relationship between two row sources.

A join is a method of combining data from two data sources.

It is controlled by join predicates, which define how the objects are related.

Nested loop

Sort_merge

Hash join

Nested loop join

Cost of accessing outer table

+

(cardinality of outer table

*

cost of accessing inner table)

Sort merge join

(Cost of accessing outer table
+
outer sort cost)
+
(Cost of accessing Inner table

+
Inner sort cost)

Hash join

(Cost of accessing outer table)

+

(Cost of building hash table)

+

(Cost of accessing Inner table)

What is a Driving Table?

- *Oracle refers to the driving table as the outer table in a join*

- *This is the table that dictates how the data is retrieved from the database*

Select T1.A, T1.B, T2.X, T2.Y from T1, T2

Where T1.A = 10 and

T1.B = T2.X;

T1 ----Driving or Outer table

T2 ----Driven or Inner table

Which Table Should be the Driving Table?

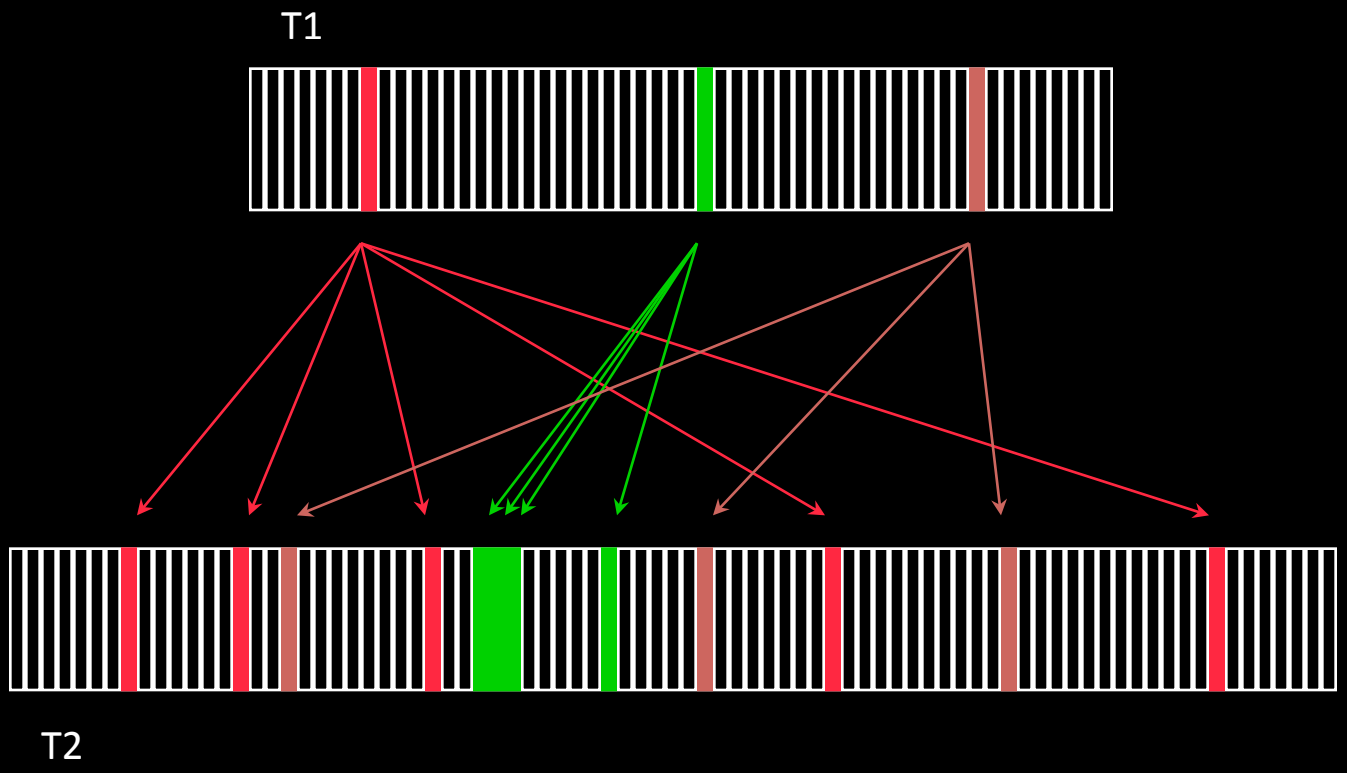
- *The table that returns the fewest number of rows the fastest*

- *If there are three or more tables, the driving table should be the intersection table*

Nested loop join

Oracle compares each row of an inner set
with Each row of the outer set
and returns those rows That satisfy the condition

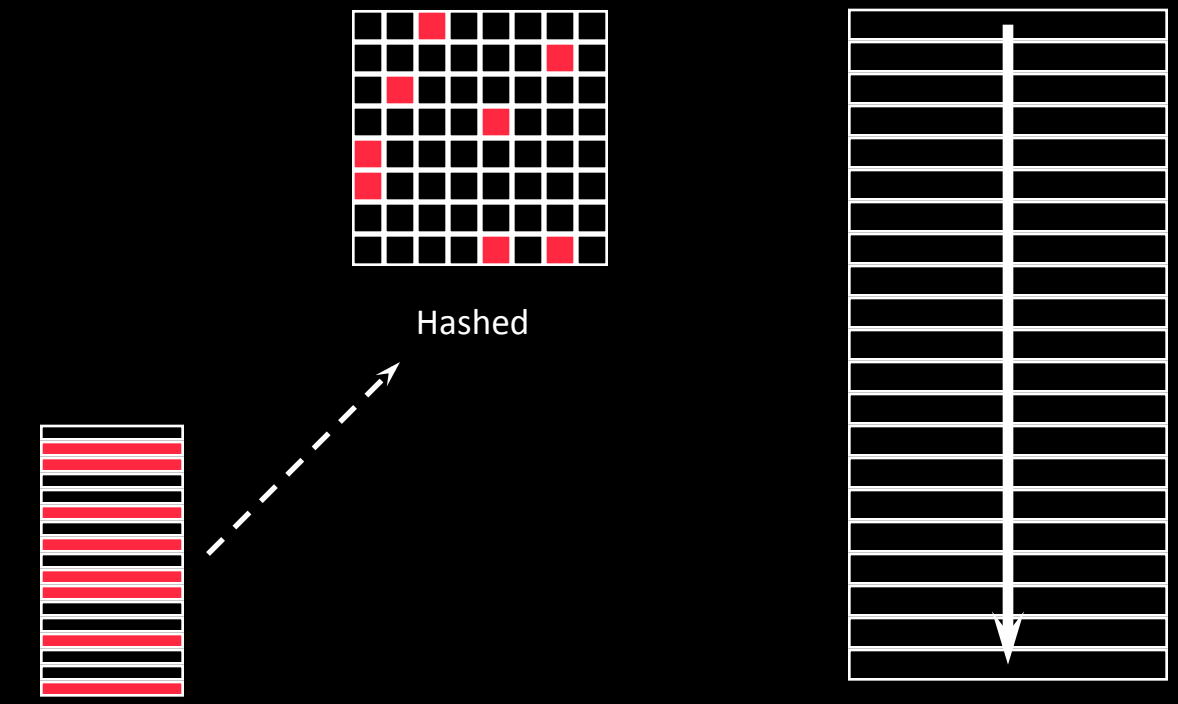
Nested Loop Algorithms ...



Hash Join ... Algorithms ...

- *The two tables are read and split into partitions.*
- *A hash table is built for each partition that fits into memory.*
- *Partitions that don't fit into memory are placed onto disk.*
- *The join is performed by taking a partition from the second table and probing the hash table.*

Hash Join Diagram



The first table is hashed in memory, the second table is used to probe the hash (build) table for matches. In simple cases the cost is easy to calculate.

Hash Join … Tips

HASH_JOIN_ENABLED =TRUE.

HASH_AREA_SIZE

Increase SORT_AREA_SIZE

Requires More Cpu

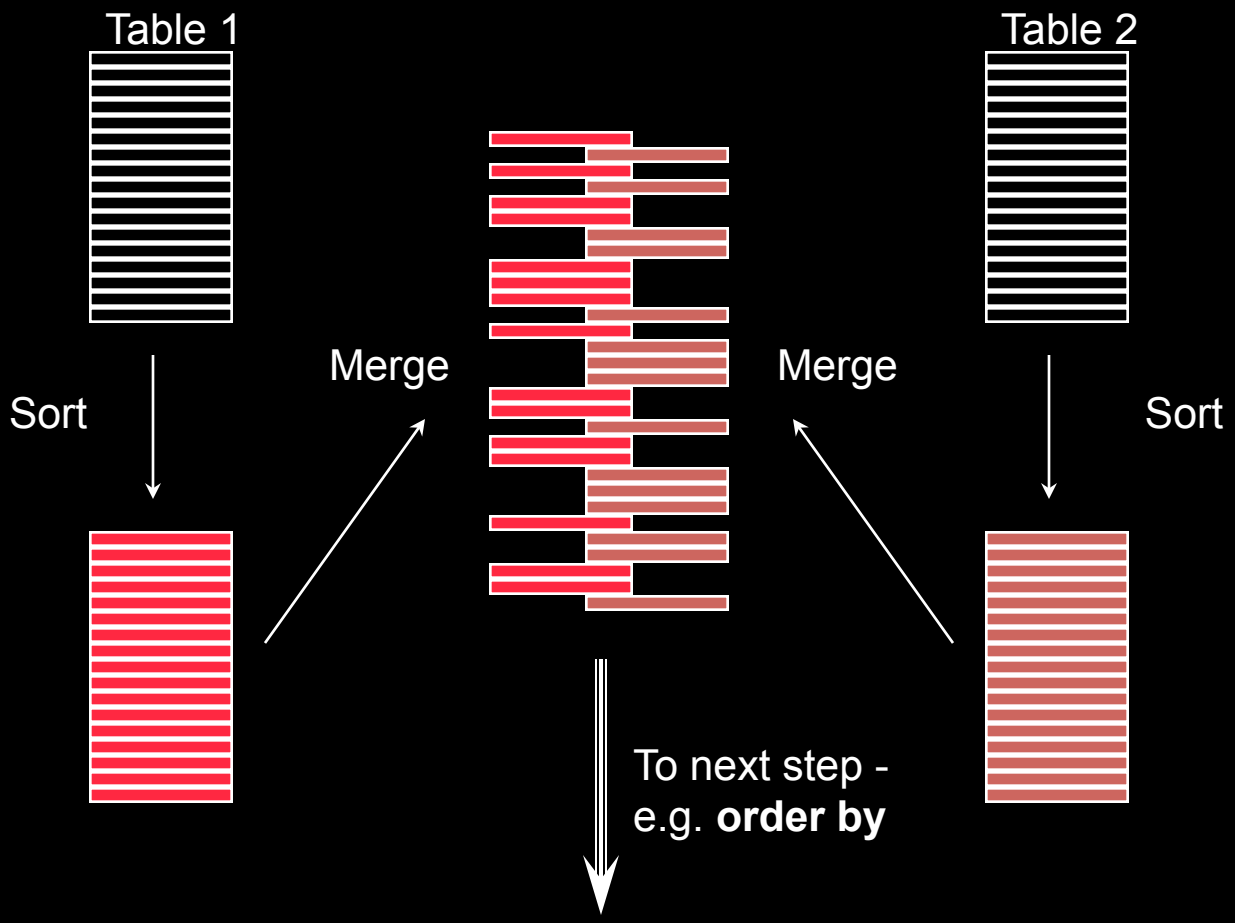
Sort / Merge Algorithms ...

1. select count(t1.v1),count(t2.v2) from big1 t1,
   big2 t2 Where t2.n2 = t1.n1;

Oracle reads, filters, and usually sorts, the qualifying rows in each of the tables independently.

The two intermediate result tables are sorted in the same sequence based on the join predicate column(s).

Sort-Merge Join Diagram



Sort merge Join … Tips

Larger tables that don't generate small result sets.

When indexes don't exist on the join predicates.

Requires more Sort area size and Temp segments

Require more Memory and Disk I/O

TUNING

Basics


Time
Money


“ Speed kills competitor ”

TARGET

User Satisfaction

Problem *Difference in expectation and reality*

Satisfaction -- *Small difference*

CHECK

User Experiences and Business Expectations

WHAT IS PERFORMANCE

It is a measure of the responsiveness of a system

WHAT IS PERFORMANCE PROBLEM

It is a lack of response within an expected time frame.

Performance improvement means

doing the same work in less time.

WHICH HAVE TO BE FASTER

Which have to be faster

1. *User Response time problems*

2. *Inefficiencies that aren't yet noticeable as user response time problems.*

Which requires Tuning

- Is business critical
- Executes frequently
- Runs a long time
- Consumes more resources

WHEN TO TUNE

- *Application design and programming*
- *Database configuration*
- *Adding a new application*
- *Ongoing Tuning*

HOW MUCH TO TUNE

GOAL *MUST BE CLEARLY DEFINED*

QUERY X RUNS 30 MINUTES AND NEED TO RUN IN LESS THAN 2 MINUTES

ONCE YOU *TARGET*

HOW TO TUNE

ELIMINATION OF WORK STEPS :

REDUCING SERVICE TIME

ELIMINATION OF WAITERS

REDUCTION OF QUEUING DELAY

Causes of performance problems

1 : OVERLOAD

2 : YOU ARE DOING IT WRONG

What is my Target

Method

Where is the time spent? *Identification*

How is the time spent?

How to reduce the time spent? *Tuning*

Causes of CPU Usage

- *Logical I/O*
- *Parsing*
- *Spinning (active waiting)*

Causes of Waiting

- *Physical I/O*
- *Lock contention*
- *Latch contention*

Have a Target *(Reducing consistent gets)* ***

Design and develop with performance in mind

Use Bulk Methods

Reduce parsing ***

*** 100 % important

Don't look for short cuts, keep it simple

Keep clear records of what has been changed

Have a sound reason for changing something

Change one thing at a time (Newton's law)

Keep proper plans for reverting back

- *Use Proper table types*
- *Use Proper index types*
- *Use Parallel Processing*
- *Use proper Partition methods*
- *Use Analytic functions*

Build your own test cases

Don't do redundant work

Know what database is capable of doing

Don’t tune a query, tune a question

Definitions which may help you for the interview

What is explain plan

Text or graphical representation of SQL execution flow

Data Skew

A non-uniform distribution of data, generally on a per column, per value basis

Row Migration - *When the amount of data for a record exceeds the space within the block, the row is migrated to another block*

Solution:

Increase the PCTFREE for the segment

Row Chaining - When the amount of data for a record exceeds the block size, the row is chained with another block

Solution:

Increase the block size

Selectivity (P)

P = r / R

R =The total number of rows in the table (R):

r = number of rows returned by the filter in where clause

Cardinality

*Selectivity * no of rows*

USER_INDEXES

BLEVEL: Height of index between root block and leaf block

LEAF_BLOCKS: Number of leaf blocks in index

DISTINCT_KEYS: Number of distinct index values

AVG_LEAF_BLOCKS_PER_KEY: Average number of leaf blocks required to store an indexed value.

USER_INDEXES

AVG_DATA_BLOCKS_PER_KEY: Average number of table blocks that contain rows referenced by indexed key value

NUM_ROWS: Number of leaf row entries

CLUSTERING_FACTOR:
Indicates how well ordered the rows in the table are in relation to the index

Fast full index scan –

This is an alternative to a full table scan when the index contains all the columns that are needed for the query.

At least one column in the index key has the NOT NULL constraint..

Uses multi block reads, unlike a full index scan.

Full Index scan –

processes all of the leaf blocks of an index, but only enough of the branch blocks to find the first leaf block.

It uses single block reads.

Parsing

Syntactic check

Syntax, keywords

Semantic check

Whether objects referenced exist, are accessible (by permissions) and are usable (status)

How to handle single user problem

1. *Understand the problem/experience from the user*
2. *Note down the current response time of the query*
3. *Understand user expectations (from x seconds to y seconds)*
4. *Check with the user was there any increase /decrease in data volume*
5. *Check with the user was there any increase /decrease in user volume*
6. *Check with the user any changes were made in the query / Logic*
7. *Check whether statistics are gathered till date*
8. *Answer yourself for our 3 questions (use explain plan , v$views)*
9. *Check reparsing , consistent gets*
10. *Write down solutions / benefits /efforts you will make for this issue*
11. *Map the right solution to the problem (Take more care here)*
12. *Do have a sound reason for changing some thing .*
13. *Cross check your solution will not bring issues in another area.*
14. *Have rollback plan if any thing goes wrong after your solution*
15. *Document all that you did for this issue .*
    1. *(For your reference)*
    2. *(You have a proof if anyone questions you)*

Thanks for your support

Thanks for your patience

Thanks for your Love

I respect and loved Your Hard work

I will never forget this crowd in life time

DONE