



Welcome

Janis Griffin  
Senior Sales Engineer

## Oracle 19c Automatic Indexing - Inside & Out

Quest

# Who Am I



Senior Sales Engineer / DBA

[Janis.Griffin@Quest.com](mailto:Janis.Griffin@Quest.com)

Twitter® - @DoBoutAnything

- Current – 30+ Years in Oracle®, DB2®, ASE, SQL Server®, MySQL®

— DBA and Developer —

Specialize in Performance Tuning

Review Database Performance for Customers  
Common Question – How do I tune it?

# Agenda

- 19c Automatic Indexing – What is it?
  - What's new in 21c
- High Level Steps
  - Capture, Verify & Decide
  - Monitor
  - Report
- DBMS\_AUTO\_INDEX Package
- 19c Automatic Indexing – How It Works
  - Several Case Studies
  - How to drop Automatic Indexes

# 19c Automatic Indexing – What is it?

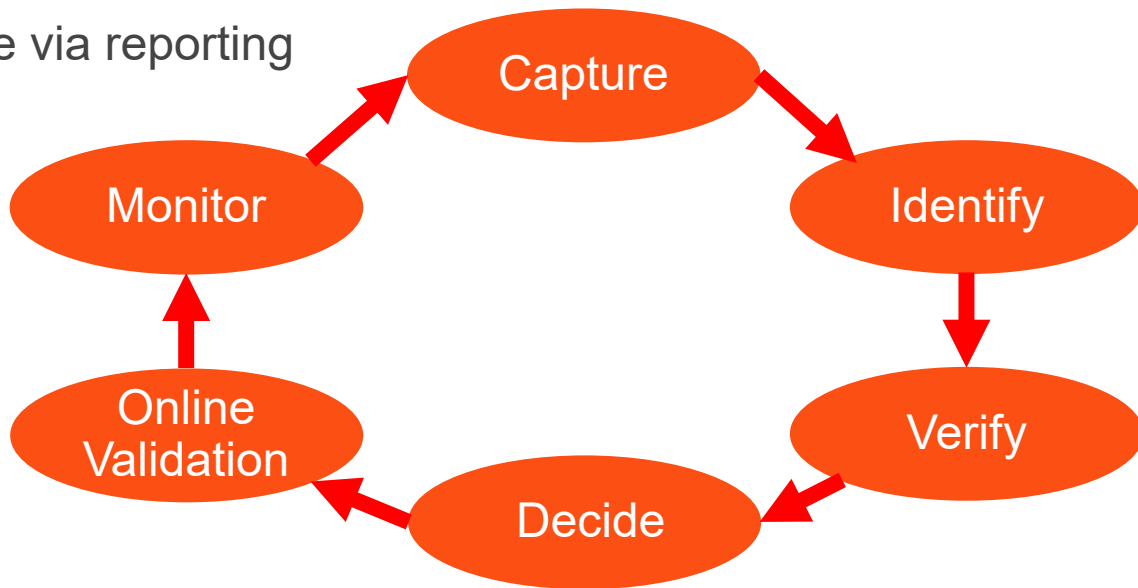
- Implements indexes based expert index tuning knowledge
  - Identifies ‘candidate indexes’ based on table column usage
  - Without DBA involvement
    - Except for DBA can set preferences
      - > View report of indexes and their impact on the application
- Works incrementally
  - Needs to be iterative and continuous
  - Created as invisible
    - Uses ‘SYS\_AI’ as the name prefix
  - Automatic indexes are tested
    - If improved performance – indexes made visible
    - If no improvement – indexes are marked unusable

---

> Later removed

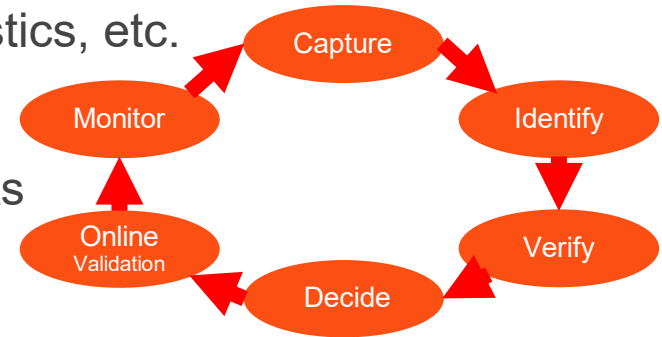
# 19c Automatic Indexing High Level Steps

- No DBA interaction
- All tuning activities
  - Auditable via reporting



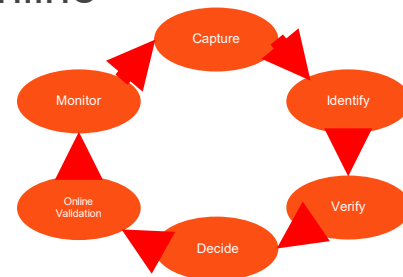
# 19c Automatic Indexing – How It Works

- Capture
  - Captures the application SQL history into a SQL repository
  - Includes SQL, plans, bind values, execution statistics, etc.
- Identify Candidate Indexes
  - That may help the newly captured SQL statements
  - Creates indexes as unusable invisible indexes
    - Metadata only
  - Drops indexes obsoleted by newly created indexes (logical merge)
- Verify
  - Ask optimizer if index candidates will be used for captured SQL statements
  - Materialize indexes and run SQL to validate that performance improved
  - All verification is done outside application workflow



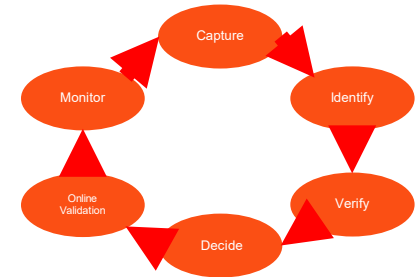
# 19c Automatic Indexing – How It Works

- Decide
  - If performance is better for all statements, indexes are marked visible
  - If performance worse for all statements, indexes remain invisible
  - If performance worse for some statements
    - Indexes are marked visible except for SQL statements that regressed
- Online Validation
  - Validation of new indexes continues for other statements online
  - Only one of the sessions executing a SQL statement
    - is allowed to use the new indexes
- Monitor
  - Index usage is continuously monitored
  - Automatically created indexes will be dropped if not used in a long tim



# 19c Automatic Indexing Benefits

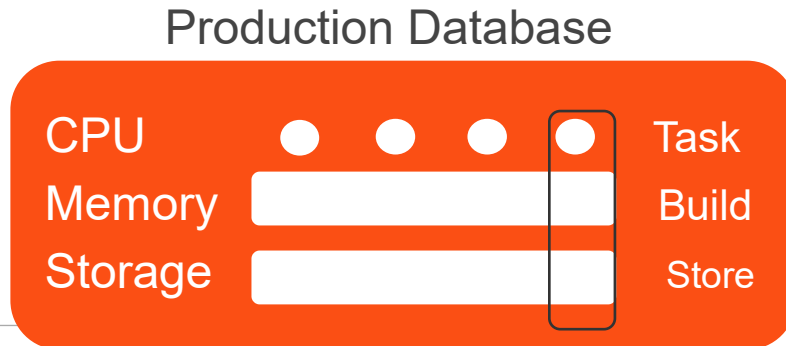
- Great for OLTP, OLAP, mixed workloads but critical for OLTP
- Applies to tuned and un-tuned applications
  - If tuned
    - Existing secondary indexes may be outdated
    - Important indexes are missing
    - Some secondary indexes can be dropped and auto indexes can be added
  - If un-tuned
    - Existing indexes support primary and unique key constraints
- Can be used in all stages of application lifecycle
- Support single and concatenated indexes
  - Function-based indexes      21c – Partitioned tables
  - Compression advanced low





# 19c Automatic Indexing

- Automatic indexing defaults to run in same database as application
- Indexing task consumes CPU, memory and storage
  - Resource manager plan limits task to 1 CPU
  - DBA can control
    - Which temp tablespace is used to build indexes
    - Which tablespace and how much space can be used by auto indexing



# Automatic Indexing Requirements

- Feature is only available to Enterprise Edition on Engineered Systems
  - Exadata only

Feature / Option / Pack	SE2	EE	EE-ES	DBCS SE	DBCS EE	DBCS EE-HP	DBCS EE-EP	ExaCS	Notes
Automatic Indexing	N	N	Y	N	N	N	N	Y	<b>EE-ES:</b> Available on Exadata. Not available on Oracle Database Appliance.

- Workaround for testing
  - In CDB as sysdba
    - Alter system set “\_Exadata\_feature\_on”=true scope=spfile;
    - Shutdown immediate;
    - Startup
- Unfortunately, this is not supported
  - Don't use on real system

# 19c DBMS\_AUTO\_INDEX Controls Auto Indexing

- Automatic indexing procedures
  - CONFIGURE
    - AUTO\_INDEX\_MODE – Turns on, off or report only
      - > IMPLEMENT - Turns on automatic indexing
        - > New indexes that improve performance are made visible & used by optimizer
      - > REPORT ONLY -Turns on automatic indexing
        - > New indexes remain invisible
      - > OFF - Turns off automatic indexing
    - AUTO\_INDEX\_SCHEMA
      - > Can include/exclude schemas using ALLOW parameter
        - > Is case sensitive & can use wildcards
      - > If NULL, all schemas can use auto index

# Configure.Auto\_Index\_Mode Example

```
SQL> SELECT parameter_name, parameter_value
  2 FROM   cdb_auto_index_config
  3* ORDER BY 1, 2;
```

PARAMETER_NAME	PARAMETER_VALUE
AUTO_INDEX_COMPRESSION	OFF
AUTO_INDEX_DEFAULT_TABLESPACE	
AUTO_INDEX_MODE	OFF
AUTO_INDEX_REPORT_RETENTION	31
AUTO_INDEX_RETENTION_FOR_AUTO	373
AUTO_INDEX_RETENTION_FOR_MANUAL	
AUTO_INDEX_SCHEMA	
AUTO_INDEX_SPACE_BUDGET	50

8 rows selected.

```
SQL> EXEC DBMS_AUTO_INDEX.CONFIGURE('AUTO INDEX MODE','IMPLEMENT');
```

PARAMETER_NAME	PARAMETER_VALUE
AUTO_INDEX_COMPRESSION	OFF
AUTO_INDEX_DEFAULT_TABLESPACE	
AUTO_INDEX_MODE	IMPLEMENT
AUTO_INDEX_REPORT_RETENTION	31
AUTO_INDEX_RETENTION_FOR_AUTO	373
AUTO_INDEX_RETENTION_FOR_MANUAL	
AUTO_INDEX_SCHEMA	
AUTO_INDEX_SPACE_BUDGET	50

# Configure.Auto\_Index\_Schema Example

```
SQL> exec dbms_auto_index.configure(parameter_name=>'AUTO_INDEX_SCHEMA', parameter_value=>'TEST',allow=> TRUE);
```

```
PL/SQL procedure successfully completed.
```

```
SQL> @dba_auto
```

PARAMETER_NAME	PARAMETER_VALUE
AUTO_INDEX_COMPRESSION	OFF
AUTO_INDEX_DEFAULT_TABLESPACE	
AUTO_INDEX_MODE	IMPLEMENT
AUTO_INDEX_REPORT_RETENTION	31
AUTO_INDEX_RETENTION_FOR_AUTO	373
AUTO_INDEX_RETENTION_FOR_MANUAL	
<u>AUTO_INDEX_SCHEMA</u>	<u>schema IN (TEST)</u>
AUTO_INDEX_SPACE_BUDGET	50

```
COLUMN parameter_name FORMAT A40
```

```
COLUMN parameter_value FORMAT A15
```

```
SELECT parameter_name, parameter_value
```

```
FROM cdb_auto_index_config
```

```
ORDER BY 1, 2;
```

- EXEC DBMS\_AUTO\_INDEX.CONFIGURE('AUTO\_INDEX\_SCHEMA', 'TEST', FALSE);

# DBMS\_AUTO\_INDEX.CONFIGURE - Cont.

## – CONFIGURE

- AUTO\_INDEX\_RETENTION\_FOR\_AUTO
  - > Number of days (default 373) auto indexes retained after last used date
- AUTO\_INDEX\_RETENTION\_FOR\_MANUAL
  - > Number of days (default NULL) manual indexes retained after last used date
- AUTO\_INDEX\_REPORT\_RETENTION
  - > Number of days automatic indexing logs are retained before deletion
    - > Automatic indexing report is based of the logs (Default is 31 days)
- AUTO\_INDEX\_DEFAULT\_TABLESPACE
  - > Tablespace to use to store auto indexes (Default is NULL)
- AUTO\_INDEX\_SPACE\_BUDGET
  - > Percentage of tablespace size to use for auto indexes
  - > Can only be used when using default tablespace is used

# DBMS\_AUTO\_INDEX.CONFIGURE - Cont.

- CONFIGURE additional commands
  - AUTO\_INDEX\_COMPRESSION enables/disables advanced compression
    - > ON for Advanced Low Compression
    - > OFF for no compression (Default)
  - PARAMETER\_VALUE is specific to parameter
    - > If NULL, setting is assigned a default value
  - ALLOW for AUTO\_INDEX\_SCHEMA parameter
    - > TRUE adds the specified schema to the inclusion list
    - > FALSE adds the schema to the exclusion list
    - > NULL remove the schema from the list that it is currently added

# DBMS\_AUTO\_INDEX.CONFIGURE Examples

```
SQL> exec dbms_auto_index.configure('AUTO_INDEX_REPORT_RETENTION','90');
```

PL/SQL procedure successfully completed.

```
SQL> SELECT parameter_name, parameter_value  
       2 FROM cdb_auto_index_config  
       3 ORDER BY 1, 2;
```

PARAMETER_NAME	PARAMETER_VALUE
AUTO_INDEX_COMPRESSION	ON
AUTO_INDEX_DEFAULT_TABLESPACE	
AUTO_INDEX_MODE	IMPLEMENT
AUTO_INDEX_REPORT_RETENTION	90
AUTO_INDEX_RETENTION_FOR_AUTO	373
AUTO_INDEX_RETENTION_FOR_MANUAL	
AUTO_INDEX_SCHEMA	schema IN (TEST)
AUTO_INDEX_SPACE_BUDGET	50

```
SQL> create tablespace auto_idx_ts datafile '/home/oracle/db_home/oradata/ORCL/orclpdb/auto_idx_ts.dbf' size 3g autoextend on;
```

Tablespace created.

```
SQL> exec DBMS_AUTO_INDEX.CONFIGURE('AUTO_INDEX_DEFAULT_TABLESPACE','AUTO_IDX_TS');
```

PL/SQL procedure successfully completed.



# DBMS\_AUTO\_INDEX.CONFIGURE Examples

```
SQL> exec DBMS_AUTO_INDEX.CONFIGURE('AUTO_INDEX_SPACE_BUDGET',20);
```

```
PL/SQL procedure successfully completed.
```

```
SQL> exec DBMS_AUTO_INDEX.CONFIGURE('AUTO_INDEX_RETENTION_FOR_MANUAL',373);
```

```
PL/SQL procedure successfully completed.
```

```
SQL> exec DBMS_AUTO_INDEX.CONFIGURE('AUTO_INDEX_RETENTION_FOR_AUTO',15);
```

```
PL/SQL procedure successfully completed.
```

```
SQL> @dba_auto;
```

PARAMETER_NAME	PARAMETER_VALUE
AUTO_INDEX_COMPRESSION	ON
AUTO_INDEX_DEFAULT_TABLESPACE	AUTO_IDX_TS
AUTO_INDEX_MODE	IMPLEMENT
AUTO_INDEX_REPORT_RETENTION	90
AUTO_INDEX_RETENTION_FOR_AUTO	15
AUTO_INDEX_RETENTION_FOR_MANUAL	373
AUTO_INDEX_SCHEMA	schema IN (TEST)
AUTO_INDEX_SPACE_BUDGET	20

# New (CDB/DBA) Views for Auto Indexes

- `DBA_AUTO_INDEX_CONFIG` \*
  - Display the current configuration of the automation index
- `DBA_AUTO_INDEX_EXECUTIONS`
  - History of Automatic Indexing task executions
- `DBA_AUTO_INDEX_IND_ACTIONS`
  - Actions performed on automatic indexes (e.g create, rebuild, etc...)
- `DBA_AUTO_INDEX_SQL_ACTIONS`
  - Actions performed on SQL to verify automatic indexes
- `DBA_AUTO_INDEX_STATISTICS`
  - Shows statistics related to automatic indexes
- `DBA_AUTO_INDEX_VERIFICATIONS`
  - Shows statistics about `PLAN_HASH_VALUE` (original buffer gets, etc...)

# Additional Views for Auto Indexes

- DBA\_ADVISOR\_TASKS – new tasks

```
SQL> select task_name, description, advisor_name, status from dba_advisor_tasks;
```

TASK_NAME	DESCRIPTION	ADVISOR_NAME	STATUS
SYS_AUTO_INDEX_TASK		SQL Access Advisor	EXECUTING
SYS_AI_VERIFY_TASK		SQL Performance Analyzer	COMPLETED
SYS_AI_SPM_EVOLVE_TASK	Automatic SPM Evolve Task	SPM Evolve Advisor	INITIAL
SYS_AUTO_SPM_EVOLVE_TASK	Automatic SPM Evolve Task	SPM Evolve Advisor	COMPLETED
AUTO_STATS_ADVISOR_TASK		Statistics Advisor	COMPLETED
SYS_AUTO_SPCADV107000614012020	Auto Space Advisor	Segment Advisor	COMPLETED
INDIVIDUAL_STATS_ADVISOR_TASK		Statistics Advisor	INITIAL

8 rows selected.

- DBA\_INDEXES – new column (AUTO)

OWNER	INDEX_NAME	AUT	INDEX_TYPE	TABLE_OWNE	TABLE_TYPE
TEST	PK_STUDENT	NO	NORMAL	TEST	TABLE
TEST	PRODUCT_PK	NO	NORMAL	TEST	TABLE
TEST	SHIPMENTDETAILS_IDX	NO	NORMAL	TEST	TABLE
TEST	SYS_AI_22ty9tc8rvvlx	YES	NORMAL	TEST	TABLE
TEST	SYS_AI_76tdrszhyq6sm	YES	NORMAL	TEST	TABLE
TEST	SYS_AI_7yqmlagd9ffnn	YES	NORMAL	TEST	TABLE
TEST	SYS_AI_8h4g2x5u9jx0v	YES	NORMAL	TEST	TABLE
TEST	SYS_AI_9nr176um7dc3x	YES	NORMAL	TEST	TABLE
TEST	SYS_AI_b7wfmv59u3nx6	YES	NORMAL	TEST	TABLE
TEST	SYS_AI_bbtzahkgk9f9s	YES	NORMAL	TEST	TABLE
TEST	SYS_AI_fyjgc63q5mz1d	YES	NORMAL	TEST	TABLE
TEST	WAGE_ID_PK	NO	NORMAL	TEST	TABLE

```
SELECT owner,  
       index_name,  
       auto,  
       index_type,  
       table_owner,  
       table_type  
FROM dba_indexes  
WHERE table_owner = 'TEST'  
-- WHERE auto='YES'  
ORDER BY owner, index_name;
```

# SMB\$CONFIG Table

- Shows both documented & undocumented settings for Auto Indexes

```
SQL> desc sys.smb$config
```

Name	Null?	Type
PARAMETER_NAME	NOT NULL	VARCHAR2 (128)
PARAMETER_VALUE	NOT NULL	NUMBER
LAST_UPDATED		TIMESTAMP (6)
UPDATED_BY		VARCHAR2 (128)
PARAMETER_DATA		CLOB

```
SQL> select PARAMETER_NAME,PARAMETER_VALUE from sys.smb$config
2 where parameter_name like '%AUTO_INDEX%' order by 1;
```

PARAMETER_NAME	PARAMETER_VALUE
AUTO_INDEX_COMPRESSION	0
AUTO_INDEX_DEFAULT_TABLESPACE	0
AUTO_INDEX_MODE	0
AUTO_INDEX_REPORT_RETENTION	31
AUTO_INDEX_RETENTION_FOR_AUTO	0
AUTO_INDEX_RETENTION_FOR_MANUAL	0
AUTO_INDEX_SCHEMA	0
AUTO_INDEX_SPACE_BUDGET	50
AUTO_INDEX_ABSDIFF_THRESHOLD	100
AUTO_INDEX_CONCURRENCY	1
AUTO_INDEX_CONTROL	0
AUTO_INDEX_DERIVE_STATISTICS	0
AUTO_INDEX_IMPROVEMENT_THRESHOLD	20
AUTO_INDEX_REBUILD_COUNT_LIMIT	5
AUTO_INDEX_REBUILD_TIME_LIMIT	30
AUTO_INDEX_REGRESSION_THRESHOLD	10
AUTO_INDEX_REVERIFY_TIME	30
AUTO_INDEX_SPA_CONCURRENCY	1
AUTO_INDEX_STS_CAPTURE_TASK	0
AUTO_INDEX_TASK_INTERVAL	900
AUTO_INDEX_TASK_MAX_RUNTIME	3600
AUTO_INDEX_TRACE	0

# Other DBMS\_AUTO\_INDEX Procedures

- DROP\_SECONDARY\_INDEXES
  - Deletes all the indexes, except the ones used for constraints
    - From a schema or a table
    - Example - `begin dbms_auto_index.drop_secondary_indexes('SH'); end;`
- REPORT\_ACTIVITY
  - Returns a report of automatic indexing operations
    - Executed during a specific period

```
declare
report clob :=null;
begin
report :=DBMS_AUTO_INDEX.REPORT_ACTIVITY (
    activity_start => TO_TIMESTAMP('2020-01-01', 'YYYY-MM-DD'),
    activity_end   => TO_TIMESTAMP('2020-01-31', 'YYYY-MM-DD'),
    type           => 'TEXT',
    section        => 'SUMMARY',
    level          => 'BASIC');
dbms_output.put_line(report);
end;
```

```
SELECT DBMS_AUTO_INDEX.report_activity(
    activity_start => SYSTIMESTAMP-1,
    activity_end   => SYSTIMESTAMP,
    type           => 'TEXT',
    section        => 'ALL')
FROM dual;
```

- REPORT\_LAST\_ACTIVITY - Returns a report of the latest operation

# 19c Automatic Indexing Reporting & Hints

- Each auto index task generates a report
  - Reports can be generated via
    - DBMS\_AUTO\_INDEX.REPORT\_ACTIVITY function
      - > Date/Time range
      - > Format (XML, HTML, Text)
      - > Level (basic, typical, all)
      - > Section
        - > Summary, Index Details,
          - Verification Details, Errors, All
- Use hints to control auto indexes
  - /\*+ USE\_AUTO\_INDEXES \*/
  - /\*+ NO\_USE\_AUTO\_INDEXES \*/

# Report\_Activity Example

```
SQL> get rpt2.sql
 1 SELECT DBMS_AUTO_INDEX.report_activity(
 2         activity_start => SYSTIMESTAMP-1,
 3         activity_end   => SYSTIMESTAMP,
 4         type           => 'TEXT',
 5         section        => 'ALL')
 6* FROM   dual
SQL> /

GENERAL INFORMATION
-----
Activity start       : 20-JAN-2020 00:26:59
Activity end         : 21-JAN-2020 00:26:59
Executions completed : 18
Executions interrupted : 0
Executions with fatal error : 2
-----

SUMMARY (AUTO INDEXES)
-----
Index candidates                : 21
Indexes created (visible / invisible) : 3 (3 / 0)
Space used (visible / invisible)  : 94.5 MB (94.5 MB / 0 B)
Indexes dropped                  : 0
SQL statements verified          : 12
SQL statements improved (improvement factor) : 6 (722.5x)
SQL plan baselines created (SQL statements) : 2 (2)
Overall improvement factor       : 7405x
-----

SUMMARY (MANUAL INDEXES)
-----
[ ] Unused indexes      : 0
Space used              : 0 B
Unusable indexes       : 0
-----
```

# Report\_Activity Cont.

```
INDEX DETAILS
-----
1. The following indexes were created:
-----
| Owner | Table          | Index                               | Key                               | Type   | Properties |
-----|-----|-----|-----|-----|-----|-----|
| TEST  | AUTO_IX        | SYS_AI_bbtzahkgk9f9s | DIST_NO                           | B-TREE | NONE       |
| TEST  | CLASS          | SYS_AI_7yqmlagd9ffnn | NAME                               | B-TREE | NONE       |
| TEST  | REGISTRATION  | SYS_AI_8h4g2x5u9jx0v | CLASS_ID,CANCELLED                | B-TREE | NONE       |
-----

VERIFICATION DETAILS
-----
1. The performance of the following statements improved:
-----
Parsing Schema Name   : TEST
SQL ID                 : 0dshxb6zujc75
SQL Text               : delete from auto_ix where dist_no = 10 and rownum <4950

Improvement Factor    : 48902x

Execution Statistics:
-----

```

	Original Plan	Auto Index Plan
Elapsed Time (s):	364413	67
CPU Time (s):	<u>355589</u>	<u>67</u>
Buffer Gets:	48902	3
Optimizer Cost:	12092	3
Disk Reads:	43758	0
Direct Writes:	0	0
Rows Processed:	4949	5
Executions:	1	1



# Report\_Activity Cont.

```
Parsing Schema Name : TEST
SQL ID              : 26fq0bn6zhkvc
SQL Text            : select * from auto_ix where dist_no=10
Improvement Factor  : 44201x

Execution Statistics:
-----
                        Original Plan                Auto Index Plan
-----
Elapsed Time (s):    8357750                          117
CPU Time (s):        8103571                          117
Buffer Gets:         2696264                          8
Optimizer Cost:      12092                            8
Disk Reads:          463                              0
Direct Writes:       0                               0
Rows Processed:      3050                             5
Executions:          61                              1

PLANS SECTION
-----
- Original
-----
Plan Hash Value : 548828358

-----
| Id | Operation                | Name      | Rows | Bytes | Cost | Time |
-----
| 0  | SELECT STATEMENT          |           |      |      | 12092 |      |
| 1  | TABLE ACCESS FULL        | AUTO_IX   | 5000 | 295000 | 12092 | 00:00:01 |
-----

- With Auto Indexes
-----
Plan Hash Value : 792607439

-----
| Id | Operation                | Name      | Rows | Bytes | Cost |
-----
| 0  | SELECT STATEMENT          |           | 5    | 295   | 8    |
| 1  | TABLE ACCESS BY INDEX ROWID BATCHED | AUTO_IX   | 5    | 295   | 8    |
| * 2 | INDEX RANGE SCAN          | SYS_AI_bbtzahkgk9f9s | 5    |       | 3    |
-----
```

# Other DBMS\_AUTO\_INDEX Procedures

- REPORT\_LAST\_ACTIVITY - Returns a report of the latest operation

```
declare  
report clob := null;  
begin
```

```
select dbms_auto_index.report_last_activity() from dual;
```

```
report := DBMS_AUTO_INDEX.REPORT_LAST_ACTIVITY (  
    type => 'TEXT',  
    section=> 'ALL',  
    level=> 'TYPICAL');  
dbms_output.put_line(report);  
end;
```

- Type can be TEXT (default), HTML or XML
- Section can be SUMMARY, INDEX\_DETAILS, VERIFICATION\_DETAILS, ERROR or ALL
  - Can combined
    - > SUMMARY + INDEX\_DETAILS – shows summary and index\_details
    - > ALL – ERRORS – shows every section except errors
- Level = Basic, Typical or All

# 3 Case Studies

- Tuning Examples
  - Used throughout the years with many Oracle versions – 10 & up
  - Manual results were compared with the Tuning Advisor suggestions
    - Consistent in previous releases
      - > Advisor usually missed the mark or got close but required additional DBA intervention
- Oracle 19C – Test Automatic Indexing
  - Billing Query for a University
  - Sale Order Query
  - Popular Airline Flights in USA

# Billing Query for a University

- Slow performance was reported by a customer
  - Having trouble with their billing system
  - The following query was identified as performing poorly

```
SELECT s.fname, s.lname, r.signup_date
FROM student s
  INNER JOIN registration r ON s.student_id = r.student_id
  INNER JOIN class c ON r.class_id = c.class_id
WHERE c.name = 'SQL TUNING'
AND r.signup_date BETWEEN
to_date(:beg_date,'DD-MON-YY') and to_date(:beg_date,'DD-MON-YY') +1
AND r.cancelled = 'N';
```

- Table sizes

- Registration – 80,000 rows
- Student – 18,000 rows
- Class – 1000 rows

TABLE_NAME	INDEX_NAME	COLUMN_NAME	COLUMN_POSITION
CLASS	PK_CLASS	CLASS_ID	1
REGISTRATION	PK_REGISTRATION	STUDENT_ID	1
REGISTRATION	PK_REGISTRATION	CLASS_ID	2
REGISTRATION	PK_REGISTRATION	SIGNUP_DATE	3
STUDENT	PK_STUDENT	STUDENT_ID	1

# Auto Indexes Enabled for Schema 'Test'

```
SQL> EXEC DBMS_AUTO_INDEX.CONFIGURE('AUTO_INDEX_MODE','IMPLEMENT');
```

PL/SQL procedure successfully completed.

```
SQL> @d_config
```

PARAMETER_NAME	PARAMETER_VALUE
AUTO_INDEX_COMPRESSION	OFF
AUTO_INDEX_DEFAULT_TABLESPACE	AUTO IDX TS
AUTO INDEX MODE	IMPLEMENT
AUTO INDEX REPORT RETENTION	90
AUTO_INDEX_RETENTION_FOR_AUTO	15
AUTO_INDEX_RETENTION_FOR_MANUAL	373
AUTO INDEX SCHEMA	schema IN (TEST)
AUTO_INDEX_SPACE_BUDGET	20

OWNER	INDEX_NAME	AUT	TABLE_NAME
TEST	SYS_AI_76tdrszhyq6sm	YES	CLASS
TEST	SYS_AI_7yqmlagd9ffnn	YES	CLASS
TEST	SYS_AI_8h4g2x5u9jx0v	YES	REGISTRATION
TEST	SYS_AI_9nr176um7dc3x	YES	REGISTRATION
TEST	SYS_AI_b7wfmv59u3nx6	YES	REGISTRATION
TEST	SYS_AI_bbtzahkgk9f9s	YES	AUTO_IX
TEST	SYS_AI_fyjgc63q5mz1d	YES	CUSTOMER

TABLE_NAME	INDEX_NAME	COLUMN_NAME	COLUMN_POSITION
CLASS	SYS_AI_76tdrszhyq6sm	CLASS_ID	1
CLASS	SYS_AI_76tdrszhyq6sm	NAME	2
CLASS	SYS_AI_7yqmlagd9ffnn	NAME	1
REGISTRATION	SYS_AI_8h4g2x5u9jx0v	CLASS_ID	1
REGISTRATION	SYS_AI_8h4g2x5u9jx0v	CANCELLED	2
REGISTRATION	SYS_AI_9nr176um7dc3x	CANCELLED	1
REGISTRATION	SYS_AI_b7wfmv59u3nx6	STUDENT_ID	1
REGISTRATION	SYS_AI_b7wfmv59u3nx6	CLASS_ID	2
AUTO_IX	SYS_AI_bbtzahkgk9f9s	DIST_NO	1
CUSTOMER	SYS_AI_fyjgc63q5mz1d	CREDIT_CARD	1

# Review Process of Registration (class\_id, canceled)

```
SELECT a.execution_name, a.table_name,
       a.index_name, b.stat_name, a.start_time
FROM dba_auto_index_ind_actions a, dba_auto_index_statistics b
WHERE a.execution_name = b.execution_name
ORDER BY 5,3;
```

EXECUTION_NAME	TABLE_NAME	INDEX_NAME	STAT_NAME	START_TIM
SYS_AI_2020-02-26/21:41:56	REGISTRATION	SYS_AI_8h4g2x5u9jx0v	SQL statements improved	26-FEB-20
SYS_AI_2020-02-26/21:41:56	REGISTRATION	SYS_AI_8h4g2x5u9jx0v	SQL statements managed by SPM	26-FEB-20
SYS_AI_2020-02-26/21:41:56	REGISTRATION	SYS_AI_8h4g2x5u9jx0v	SQL plan baselines created	26-FEB-20
SYS_AI_2020-02-26/21:41:56	REGISTRATION	SYS_AI_8h4g2x5u9jx0v	Improvement percentage	26-FEB-20
SYS_AI_2020-02-26/21:41:56	REGISTRATION	SYS_AI_8h4g2x5u9jx0v	Index candidates	26-FEB-20
SYS_AI_2020-02-26/21:41:56	REGISTRATION	SYS_AI_8h4g2x5u9jx0v	SQL statements verified	26-FEB-20
SYS_AI_2020-02-26/21:41:56	REGISTRATION	SYS_AI_8h4g2x5u9jx0v	Indexes created (invisible)	26-FEB-20
SYS_AI_2020-02-26/21:41:56	REGISTRATION	SYS_AI_8h4g2x5u9jx0v	Indexes dropped	26-FEB-20
SYS_AI_2020-02-26/21:41:56	REGISTRATION	SYS_AI_8h4g2x5u9jx0v	Space used in bytes	26-FEB-20
SYS_AI_2020-02-26/21:41:56	REGISTRATION	SYS_AI_8h4g2x5u9jx0v	Space reclaimed in bytes	26-FEB-20
SYS_AI_2020-02-26/21:41:56	REGISTRATION	SYS_AI_8h4g2x5u9jx0v	Indexes created (visible)	26-FEB-20

## DBA\_AUTO\_INDEX\_VERIFICATIONS

EXECUTION_NAME	SQL_ID	ORIGINAL_PLAN_HASH_VALUE	AUTO_INDEX_PLAN_HASH_VALUE	ORIGINAL_BUFFER_GETS	AUTO_INDEX_BUFFER_GETS	STATUS
SYS_AI_2020-02-26/21:41:56	cqa9shb4n45zq	1244828764	2693604979	334.157974		331 UNCHANGED
SYS_AI_2020-02-27/22:19:47	1m72dnkulam29	309240793	2441908068		9	7 UNCHANGED
SYS_AI_2020-02-27/22:19:47	b461cvfsjcczj	2025025906	3891477460		15	14 UNCHANGED
SYS_AI_2020-02-27/22:19:47	bzc043n9nxt7s	1478357878	2693604979		17087	325 IMPROVED
SYS_AI_2020-02-27/22:19:47	fgday4r6bpf59	309240793	1378088465		9	6 UNCHANGED
SYS_AI_2020-02-27/22:34:49	cqa9shb4n45zq	13237339	2693604979		167.36152	325 REGRESSED

# Auto Indexes Created

- Shows status of indexes
  - 2 indexes are taking up space

```
SQL> select index_name, status, dropped, visibility, segment_created
       2 from user_indexes where auto='YES';
```

INDEX_NAME	STATUS	DRO	VISIBILIT	SEG
SYS_AI_bbtzahkgk9f9s	UNUSABLE	NO	INVISIBLE	NO
SYS_AI_76tdrszhyq6sm	UNUSABLE	NO	INVISIBLE	NO
SYS_AI_7yqmlagd9ffnn	<u>VALID</u>	<u>NO</u>	<u>INVISIBLE</u>	<u>YES</u>
SYS_AI_fyjgc63q5mz1d	UNUSABLE	NO	INVISIBLE	NO
SYS_AI_b7wfmv59u3nx6	UNUSABLE	NO	INVISIBLE	NO
SYS_AI_8h4g2x5u9jx0v	<u>VALID</u>	<u>NO</u>	<u>INVISIBLE</u>	<u>YES</u>
SYS_AI_9nr176um7dc3x	UNUSABLE	NO	INVISIBLE	NO

Class.name

Reg.Class\_id,Canceled

```
select segment_name, bytes from dba_segments
where segment_name in
  (select index_name from dba_indexes where tablespace_name like 'AUTO%');
```

SEGMENT_NAME	BYTES
SYS_AI_7yqmlagd9ffnn	131072
SYS_AI_8h4g2x5u9jx0v	2097152

```
Total size
-----
2228224
```

# With Compression

```
EXEC DBMS_AUTO_INDEX.CONFIGURE('AUTO_INDEX_COMPRESSION','ON');
```

INDEX_NAME	STATUS	VISIBILIT	DRO	COMPRESSION	SEG
SYS_AI_bbtzahkgk9f9s	UNUSABLE	INVISIBLE	NO	ADVANCED LOW	NO
SYS_AI_76tdrszhyq6sm	VALID	INVISIBLE	NO	ADVANCED LOW	YES
SYS_AI_7yqmlagd9ffnn	VALID	VISIBLE	NO	ADVANCED LOW	YES
SYS_AI_fyjgc63q5mz1d	UNUSABLE	INVISIBLE	NO	ADVANCED LOW	NO
SYS_AI_b7wfmv59u3nx6	VALID	INVISIBLE	NO	ADVANCED LOW	YES
SYS_AI_8h4g2x5u9jx0v	VALID	VISIBLE	NO	ADVANCED LOW	YES
SYS_AI_9nr176um7dc3x	UNUSABLE	INVISIBLE	NO	ADVANCED LOW	NO



# Report\_Activity

```
DBMS_AUTO_INDEX.REPORT_ACTIVITY(SYSTIMESTAMP-3,SYSTIMESTAMP,'TEXT','ALL','ALL')
```

---

**GENERAL INFORMATION**

---

```
Activity start      : 25-FEB-2020 19:26:23
Activity end       : 28-FEB-2020 19:26:23
Executions completed : 68
Executions interrupted : 0
Executions with fatal error : 0
```

---

**SUMMARY (AUTO INDEXES)**

---

```
Index candidates      : 14
Indexes created (visible / invisible) : 6 (2 / 4)
Space used (visible / invisible) : 6.68 MB (2.23 MB / 4.46 MB)
Indexes dropped       : 0
SQL statements verified : 6
SQL statements improved (improvement factor) : 1 (52.6x)
SQL plan baselines created : 0
Overall improvement factor : 17.5x
```

---

**SUMMARY (MANUAL INDEXES)**

---

```
DBMS_AUTO_INDEX.REPORT_ACTIVITY(SYSTIMESTAMP-3,SYSTIMESTAMP,'TEXT','ALL','ALL')
```

---

```
Unused indexes      : 0
Space used          : 0 B
Unusable indexes    : 0
```

---

**INDEX DETAILS**

---

1. The following indexes were created:  
\*: invisible

---

Owner	Table	Index	Key	Type
TEST	CLASS	* SYS_AI_76tdrszhyq6sm	CLASS_ID,NAME	B-TREE
TEST	CLASS	SYS_AI_7yqmlagq9ffnn	NAME	B-TREE
TEST	REGISTRATION	SYS_AI_8h4g2x5u9jx0v	CLASS_ID,CANCELLED	B-TREE
TEST	REGISTRATION	* SYS_AI_b7wfmv59u3nx6	STUDENT_ID,CLASS_ID	B-TREE

**VERIFICATION DETAILS**

---

1. The performance of the following statements improved:

---

```
Parsing Schema Name : TEST
SQL ID               : bzc043n9nxt7s
SQL Text             : /* SQL Analyze(41,1) */ SELECT s.fname, s.lname,
r.signup_date FROM student s INNER JOIN registration r ON
s.student_id = r.student_id INNER JOIN class c ON
r.class_id = c.class_id WHERE c.name = 'SQL TUNING' AND
r.signup_date BETWEEN to_date(:beg_date,'DD-MON-YY') and
to_date(:beg_da...
```

---

```
Improvement Factor : 52.6x
```

---

**Execution Statistics:**

---

	Original Plan	Auto Index Plan
Elapsed Time (s):	439090	38276
CPU Time (s):	273210	25351
Buffer Gets:	170870	342
Optimizer Cost:	105	116
Disk Reads:	36	1
Direct Writes:	0	0
Rows Processed:	80	8
Executions:	10	1

# Report\_Activity Cont.

```
PLANS SECTION
-----
- Original
-----
Plan Hash Value : 1478357878
-----
| Id | Operation                                | Name                | Rows | Bytes | Cost |
-----|-----|-----|-----|-----|-----|
| 0  | SELECT STATEMENT                          |                     |      |       | 105  |
| 1  |   FILTER                                  |                     |      |       |      |
| 2  |     NESTED LOOPS                           |                     | 1    | 112   | 105  |
| 3  |       NESTED LOOPS                         |                     | 1    | 47    | 104  |
| 4  |         TABLE ACCESS FULL                 | REGISTRATION        | 1    | 18    | 103  |
| 5  |           TABLE ACCESS BY INDEX ROWID    | STUDENT              | 1    | 29    | 1    |
| 6  |             INDEX UNIQUE SCAN              | PK_STUDENT           | 1    |       | 0    |
| 7  |               TABLE ACCESS BY INDEX ROWID | CLASS                | 1    | 65    | 1    |
| 8  |                 INDEX UNIQUE SCAN          | PK_CLASS             | 1    |       | 0    |
-----
- With Auto Indexes
-----
Plan Hash Value : 2693604979
-----
| Id | Operation                                | Name                | Rows |
-----|-----|-----|-----|
| 0  | SELECT STATEMENT                          |                     | 8    |
| * 1 |   FILTER                                  |                     |      |
| 2  |     NESTED LOOPS                           |                     | 8    |
| 3  |       NESTED LOOPS                         |                     | 8    |
| * 4 |         HASH JOIN                           |                     | 8    |
| 5  |           TABLE ACCESS BY INDEX ROWID BATCHED | CLASS                | 2    |
| * 6 |             INDEX RANGE SCAN                | SYS_AI_7yqmlaqd9ffnn | 2    |
| * 7 |               TABLE ACCESS FULL           | REGISTRATION        | 4183 |
| * 8 |                 INDEX UNIQUE SCAN           | PK_STUDENT           | 1    |
| 9  |                   TABLE ACCESS BY INDEX ROWID | STUDENT              | 1    |
-----
Notes
-----
- Dynamic sampling used for this statement ( level = 11 )
- This is an adaptive plan
-----
ERRORS
-----
No errors found.
```

# Did it measure up?

```
Plan hash value: 2281644015
```

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		1	112	107 (2)	00:00:01
* 1	FILTER					
2	NESTED LOOPS		1	112	107 (2)	00:00:01
3	NESTED LOOPS		1	112	107 (2)	00:00:01
* 4	HASH JOIN		1	83	106 (2)	00:00:01
5	TABLE ACCESS BY INDEX ROWID BATCHED	CLASS	1	65	2 (0)	00:00:01
* 6	INDEX RANGE SCAN	SYS AI *ffnn	1		1 (0)	00:00:01
* 7	TABLE ACCESS FULL	REGISTRATION	199	3582	104 (2)	00:00:01
* 8	INDEX UNIQUE SCAN	PK STUDENT	1		0 (0)	00:00:01
9	TABLE ACCESS BY INDEX ROWID	STUDENT	1	29	1 (0)	00:00:01

Auto Cost  
107

create index cov\_reg on registration(class\_id, signup\_date, cancelled)

```
Plan hash value: 923900230
```

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		1	112	5 (0)	00:00:01
* 1	FILTER					
2	NESTED LOOPS		1	112	5 (0)	00:00:01
3	NESTED LOOPS		1	112	5 (0)	00:00:01
4	NESTED LOOPS		1	83	4 (0)	00:00:01
5	TABLE ACCESS BY INDEX ROWID BATCHED	CLASS	1	65	2 (0)	00:00:01
* 6	INDEX RANGE SCAN	CL NAME	1		1 (0)	00:00:01
7	TABLE ACCESS BY INDEX ROWID BATCHED	REGISTRATION	1	18	2 (0)	00:00:01
* 8	INDEX RANGE SCAN	COV REG	1		1 (0)	00:00:01
* 9	INDEX UNIQUE SCAN	PK STUDENT	1		0 (0)	00:00:01
10	TABLE ACCESS BY INDEX ROWID	STUDENT	1	29	1 (0)	00:00:01

DBA Tuned  
Cost  
5

# Auto Index on Registration

Plan Details Operation Analysis Object Analysis

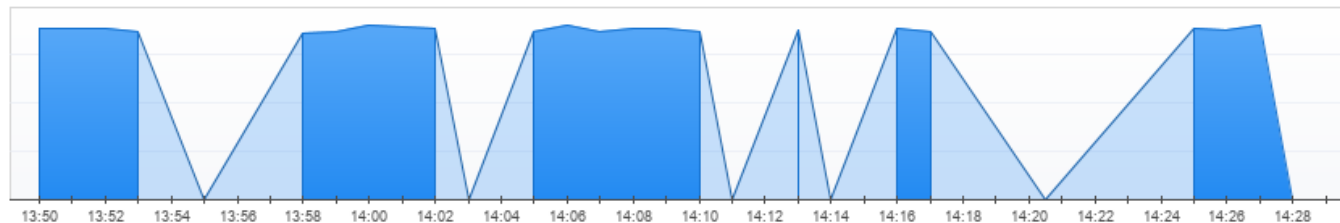
Operation	Object Name	Object Type	Cost	CPU Cost	I/O Cost	Cardinality	Bytes	Time (seconds)	Temp Space	Access Predicates
SELECT STATEMENT			16.84 %	0	0	0	0	0	0	
FILTER			0.00 %	0	0	0	0	0	0	
HASH JOIN			16.84 %	948,135	79	4	448	1	0	"S"."STUDENT_ID"="R"."STUDENT_ID"
NESTED LOOPS			16.84 %	948,135	79	4	448	1	0	
NESTED LOOPS			16.84 %	948,135	79	4	448	1	0	
STATISTICS COLLECTOR			0.00 %	0	0	0	0	0	0	
NESTED LOOPS			15.99 %	911,289	75	4	332	1	0	
TABLE ACCESS FULL	TEST.CLASS	TABLE	1.07 %	312,579	5	1	65	1	0	
TABLE ACCESS BY INDEX ROWID BATCHED	TEST.REGISTRATION	TABLE	14.93 %	598,711	70	4	72	1	0	
INDEX RANGE SCAN	TEST.SYS_AI_8h4g2x5u9jx0v	INDEX	0.21 %	23,971	1	80	0	1	0	"R"."CLASS_ID"="C"."CLASS_ID" AND "R"."CANCELLED"='N'
INDEX UNIQUE SCAN	TEST.PK_STUDENT	INDEX (UNIQUE)	0.00 %	1,900	0	1	0	0	0	"S"."STUDENT_ID"="R"."STUDENT_ID"
TABLE ACCESS BY INDEX ROWID	TEST.STUDENT	TABLE	0.21 %	9,211	1	1	29	1	0	
TABLE ACCESS FULL	TEST.STUDENT	TABLE	0.21 %	9,211	1	1	29	1	0	

# With Auto Index

- DB Users
  - SYS
  - TEST
- SQL Statements
  - SELECT s.fname, s.lname, r.signup\_date FROM
- PL/SQL Blocks
- Programs
- OS Users
- Machines
- Actions
- Modules
- Client Info
- Command Types
- Services
- Consumer Groups
- Sessions
- Client PIDs
- Locked Objects
- Files
- Disks
- Objects I/O
- SOE
- SYSTEM

Overview | Blocking History | Activity Highlights

## Buffer Gets



## Workload related Metrics

Select Metric | View SQL Text | Analyze Plan | Tune SQL | Compare

Metric ▲	Total
Average SQL Response Time	< 0.01
Buffer Gets	4,030,505.00
CPU Usage	6.03
Disk Reads	22.00
Elapsed Time	10.33
Executions	22,596.00
Fetches	45,192.00
Rows Processed	181,844.00

# Auto Index on Registration

Plan hash value: 2023948573

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		1	112	76 (0)	00:00:01
* 1	FILTER					
2	NESTED LOOPS		1	112	76 (0)	00:00:01
3	NESTED LOOPS		1	112	76 (0)	00:00:01
4	NESTED LOOPS		1	83	75 (0)	00:00:01
* 5	TABLE ACCESS FULL	CLASS	1	65	5 (0)	00:00:01
* 6	TABLE ACCESS BY INDEX ROWID BATCHED	REGISTRATION	1	18	70 (0)	00:00:01
* 7	INDEX RANGE SCAN	SYS_AI_8h4g2x5u9jx0v	80		1 (0)	00:00:01
* 8	INDEX UNIQUE SCAN	PK_STUDENT	1		0 (0)	00:00:01
9	TABLE ACCESS BY INDEX ROWID	STUDENT	1	29	1 (0)	00:00:01

# Covered Index on Registration

Plan Details
Operation Analysis
Object Analysis

Operation	Object Name	Object Type	Cost	CPU Cost	I/O Cost	Cardinality	Bytes	Time (seconds)	Temp Space
SELECT STATEMENT			<div style="width: 14.86%; height: 10px; background-color: #4F81BD;"></div> 14.86 %	0	0	0	0	0	0
FILTER			<div style="width: 0.00%; height: 10px; background-color: #4F81BD;"></div> 0.00 %	0	0	0	0	0	0
HASH JOIN			<div style="width: 14.86%; height: 10px; background-color: #4F81BD;"></div> 14.86 %	91,186	11	4	448	1	0
NESTED LOOPS			<div style="width: 14.86%; height: 10px; background-color: #4F81BD;"></div> 14.86 %	91,186	11	4	448	1	0
NESTED LOOPS			<div style="width: 14.86%; height: 10px; background-color: #4F81BD;"></div> 14.86 %	91,186	11	4	448	1	0
STATISTICS COLLECTOR			<div style="width: 0.00%; height: 10px; background-color: #4F81BD;"></div> 0.00 %	0	0	0	0	0	0
HASH JOIN			<div style="width: 9.46%; height: 10px; background-color: #4F81BD;"></div> 9.46 %	54,340	7	4	332	1	0
NESTED LOOPS			<div style="width: 9.46%; height: 10px; background-color: #4F81BD;"></div> 9.46 %	54,340	7	4	332	1	0
STATISTICS COLLECTOR			<div style="width: 0.00%; height: 10px; background-color: #4F81BD;"></div> 0.00 %	0	0	0	0	0	0
TABLE ACCESS BY INDEX ROWID BATCHED	TEST.CLASS	TABLE	<div style="width: 2.70%; height: 10px; background-color: #4F81BD;"></div> 2.70 %	15,833	2	1	65	1	0
INDEX RANGE SCAN	TEST.CL_NAME	INDEX	<div style="width: 1.35%; height: 10px; background-color: #4F81BD;"></div> 1.35 %	8,371	1	1	0	1	0
TABLE ACCESS BY INDEX ROWID BATCHED	TEST.REGISTRATION	TABLE	<div style="width: 6.76%; height: 10px; background-color: #4F81BD;"></div> 6.76 %	38,507	5	4	72	1	0
INDEX RANGE SCAN	TEST.COV_REG	INDEX	<div style="width: 1.35%; height: 10px; background-color: #4F81BD;"></div> 1.35 %	8,971	1	4	0	1	0
TABLE ACCESS FULL	TEST.REGISTRATION	TABLE	<div style="width: 6.76%; height: 10px; background-color: #4F81BD;"></div> 6.76 %	38,507	5	4	72	1	0
INDEX UNIQUE SCAN	TEST.PK_STUDENT	INDEX (UNIQUE)	<div style="width: 0.00%; height: 10px; background-color: #4F81BD;"></div> 0.00 %	1,900	0	1	0	0	0
TABLE ACCESS BY INDEX ROWID	TEST.STUDENT	TABLE	<div style="width: 1.35%; height: 10px; background-color: #4F81BD;"></div> 1.35 %	9,211	1	1	29	1	0
TABLE ACCESS FULL	TEST.STUDENT	TABLE	<div style="width: 1.35%; height: 10px; background-color: #4F81BD;"></div> 1.35 %	9,211	1	1	29	1	0

SQL Text

```

SELECT s.fname, s.lname, r.signup_date
FROM student s
     INNER JOIN registration r
           ON s.student_id = r.student_id
     INNER JOIN class c
           ON r.class_id = c.class_id
WHERE   c.name = 'SQL TUNING'
       AND r.signup_date BETWEEN TO_DATE (:beg_date, 'DD-MON-YY')
                          AND TO_DATE (:beg_date, 'DD-MON-YY') + 1
       AND r.cancelled = 'N'

```

# Covered Index on Registration

```
SELECT * FROM (SELECT t.tablespace_name table
SELECT /*+ OPT_PARAM('_parallel_syspls_obey_f
/* SQL Analyze(283,1) */ SELECT c_last, c_first, c
SELECT /* D5_SVC */ /*+ dynamic_sampling(0) no
SELECT POS+1 POS, VAL, NONNULLS, NDV, SPLIT
select count(*) from dba_scheduler_windows w, d
```

- PL/SQL Blocks
- Programs
- OS Users
- DB Users
- SOE
- SYS
- TEST

- SQL Statements
- SELECT s.fname, s.lname, r.signup\_date FROM
- select /\*+ connect\_by\_filtering index(sysautl
- select SYS\_CONTEXT('USERENV', 'SERVER\_HO

Overview | Blocking History | Activity Highlights

### Buffer Gets

### Workload related Metrics

Select Metric | View SQL Text | Analyze Plan | Tune SQL | Compare

Metric	Total	
Average SQL Response Time	< 0.01	< 0.01
Buffer Gets	658,622.00	4,030,505.00
CPU Usage	3.96	6.03
Disk Reads	0.00	22.00
Elapsed Time	6.87	10.33
Executions	18,465.00	22,596.00
Fetches	36,931.00	45,192.00
Rows Processed	148,610.00	181,844.00

40

quest.com | confidential



# Covered Index on Registration Wins

- Auto Index on Class(name) cost 107
- Auto Index on Registration(class\_id, canceled) cost 76
- DBA Index on Class(name), Registration(class\_id, signup\_date, cancelled)
  - Cost 5

```
Plan hash value: 923900230
```

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		1	112	5 (0)	00:00:01
* 1	FILTER					
2	NESTED LOOPS		1	112	5 (0)	00:00:01
3	NESTED LOOPS		1	112	5 (0)	00:00:01
4	NESTED LOOPS		1	83	4 (0)	00:00:01
5	TABLE ACCESS BY INDEX ROWID BATCHED	CLASS	1	65	2 (0)	00:00:01
* 6	INDEX RANGE SCAN	CL_NAME	1		1 (0)	00:00:01
7	TABLE ACCESS BY INDEX ROWID BATCHED	REGISTRATION	1	18	2 (0)	00:00:01
* 8	INDEX RANGE SCAN	COV_REG	1		1 (0)	00:00:01
* 9	INDEX UNIQUE SCAN	PK_STUDENT	1		0 (0)	00:00:01
10	TABLE ACCESS BY INDEX ROWID	STUDENT	1	29	1 (0)	00:00:01

# Sale Order Query

- HammerDB load utility – Slow running query

```
SELECT c_last, c_first, c_street_1, c_city, c_state, c_zip,  
       c_phone, o_entry_d, d_name, ol_delivery_d, ol_quantity, ol_amount  
FROM order_line, orders, district, customer, stock  
WHERE o_id = ol_o_id  
AND o_c_id=c_id  
AND s_i_id = ol_i_id  
AND d_id = ol_d_id  
AND ol_w_id = :B2  
AND ol_d_id = :B4  
AND (ol_o_id < :B3 )  
AND ol_o_id >= (:B3 - 20)  
AND s_w_id = :B2  
AND s_quantity < :B1  
AND d_id = :B4  
AND c_last like :B5 ;
```

Order_line	60,461,709
Orders	6,046,215
District	50
Customer	150,000
Stock	500,000

# Existing Indexes

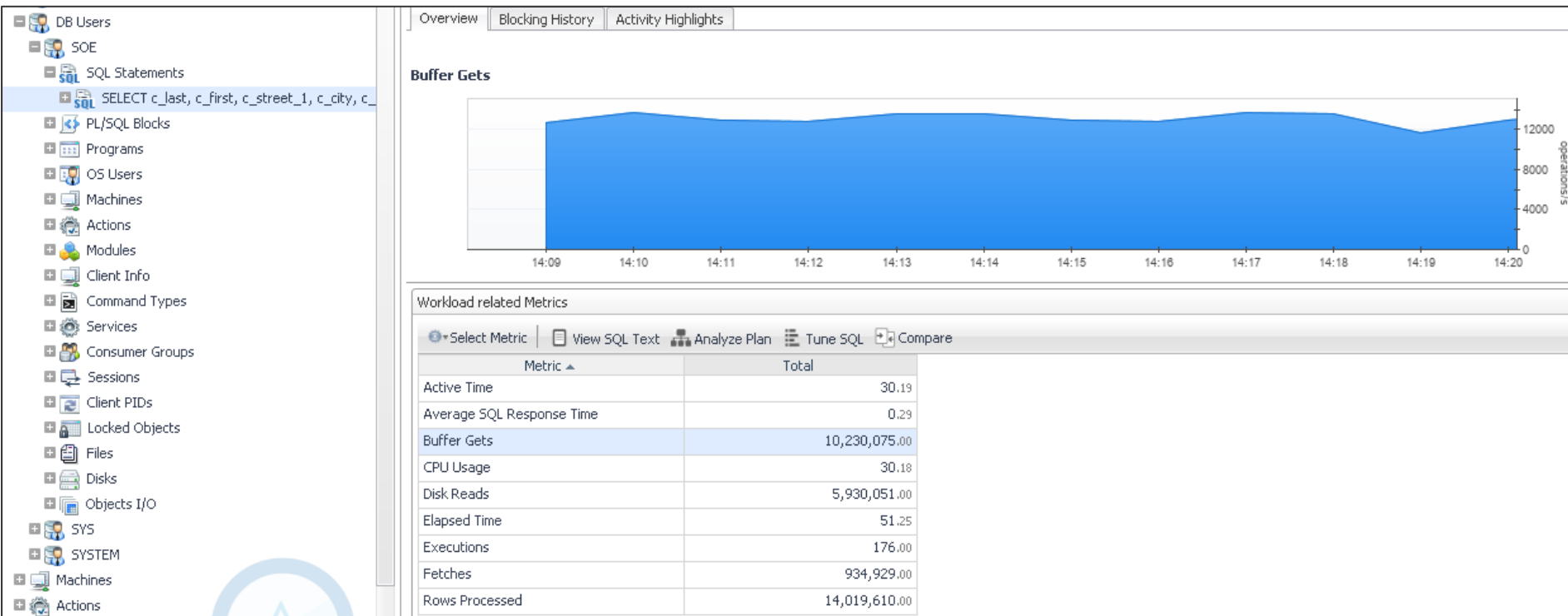
TABLE_NAME	INDEX_NAME	COLUMN_NAME	COLUMN_POSITION
CUSTOMER	CUSTOMER_I1	C_W_ID	1
CUSTOMER	CUSTOMER_I1	C_D_ID	2
CUSTOMER	CUSTOMER_I1	C_ID	3
DISTRICT	DISTRICT_I1	D_W_ID	1
DISTRICT	DISTRICT_I1	D_ID	2
ORDERS	ORDERS_I1	O_W_ID	1
ORDERS	ORDERS_I1	O_D_ID	2
ORDERS	ORDERS_I1	O_ID	3
ORDER_LINE	IORDL	OL_W_ID	1
ORDER_LINE	IORDL	OL_D_ID	2
ORDER_LINE	IORDL	OL_O_ID	3
ORDER_LINE	IORDL	OL_NUMBER	4
STOCK	STOCK_IDX	S_I_ID	1
STOCK	STOCK_IDX	S_W_ID	2
WAREHOUSE	WAREHOUSE_I1	W_ID	1

# Original Execution Plan

```
Execution Plan
-----
Plan hash value: 1040961599
-----
```

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		4010	650K	15981 (1)	00:00:01
* 1	FILTER					
* 2	HASH JOIN		4010	650K	15981 (1)	00:00:01
* 3	HASH JOIN		1594	112K	12687 (1)	00:00:01
4	NESTED LOOPS		1542	84810	4631 (1)	00:00:01
5	NESTED LOOPS		1542	84810	4631 (1)	00:00:01
* 6	HASH JOIN		1512	66528	94 (0)	00:00:01
7	TABLE ACCESS BY INDEX ROWID BATCHED	DISTRICT	5	60	6 (0)	00:00:01
* 8	INDEX SKIP SCAN	DISTRICT_I1	5		1 (0)	00:00:01
* 9	INDEX RANGE SCAN	IORDL	3023	96736	88 (0)	00:00:01
* 10	INDEX RANGE SCAN	STOCK_IDX	1		2 (0)	00:00:01
* 11	TABLE ACCESS BY INDEX ROWID	STOCK	1	11	3 (0)	00:00:01
* 12	TABLE ACCESS FULL	ORDERS	15116	250K	8055 (1)	00:00:01
* 13	TABLE ACCESS FULL	CUSTOMER	7500	688K	3294 (1)	00:00:01

# Original Performance



# Include SOE Schema for Auto Indexing

PARAMETER_NAME	PARAMETER_VALUE
AUTO_INDEX_COMPRESSION	ON
AUTO_INDEX_DEFAULT_TABLESPACE	AUTO_IDX_TS
AUTO_INDEX_MODE	IMPLEMENT
AUTO_INDEX_REPORT_RETENTION	90
AUTO_INDEX_RETENTION_FOR_AUTO	15
AUTO_INDEX_RETENTION_FOR_MANUAL	373
AUTO_INDEX_SCHEMA	schema IN (TEST, SOE)
AUTO_INDEX_SPACE_BUDGET	20

INDEX_NAME	TABLE_NAME	AUT	VISIBILIT	COMPRESSION	SEG	STATUS
SYS_AI_8k0xma30nayxn	CUSTOMER	YES	INVISIBLE	ADVANCED	LOW	YES VALID
SYS_AI_0jfsy72532qv3	CUSTOMER	YES	INVISIBLE	ADVANCED	LOW	YES VALID
SYS_AI_a3tc4dj87650q	CUSTOMER	YES	INVISIBLE	ADVANCED	LOW	NO UNUSABLE
SYS_AI_gj2prfsytzu50	CUSTOMER	YES	INVISIBLE	ADVANCED	LOW	YES VALID
SYS_AI_18pkdxrps0j2m	ORDERS	YES	INVISIBLE	ADVANCED	LOW	YES VALID
SYS_AI_97ya3cug4hxpq	ORDERS	YES	INVISIBLE	ADVANCED	LOW	YES VALID
SYS_AI_3ys7c39vs247p	ORDERS	YES	INVISIBLE	ADVANCED	LOW	NO UNUSABLE
SYS_AI_81dnzcja2qhpq	ORDERS	YES	INVISIBLE	ADVANCED	LOW	NO UNUSABLE
SYS_AI_fdbazxb641kwv	STOCK	YES	INVISIBLE	ADVANCED	LOW	NO UNUSABLE

```
SELECT index_name,table_name,
       auto,visibility, compression,
       segment_created, status
FROM user_indexes
WHERE auto='YES';
```

# Automatic Indexes

SEGMENT_NAME	BYTES
SYS_AI_18pkdxrps0j2m	109051904
SYS_AI_97ya3cug4hxp	117440512
SYS_AI_8k0xma30nayxn	3145728
SYS_AI_0jfsy72532qv3	2097152
SYS_AI_gj2prfsytzu50	4194304

Total Space: 225m

Visible: 9m

TABLE_NAME	INDEX_NAME	COLUMN_NAME	COLUMN_POSITION
CUSTOMER	SYS_AI_0jfsy72532qv3	C_LAST	1
CUSTOMER	SYS_AI_8k0xma30nayxn	C_ID	1
CUSTOMER	SYS_AI_8k0xma30nayxn	C_D_ID	2
CUSTOMER	SYS_AI_8k0xma30nayxn	C_W_ID	3
CUSTOMER	SYS_AI_a3tc4dj87650q	C_W_ID	1
CUSTOMER	SYS_AI_gj2prfsytzu50	C_D_ID	1
CUSTOMER	SYS_AI_gj2prfsytzu50	C_W_ID	1
CUSTOMER	SYS_AI_gj2prfsytzu50	C_LAST	1
ORDERS	SYS_AI_18pkdxrps0j2m	O_ID	1
ORDERS	SYS_AI_18pkdxrps0j2m	O_W_ID	1
ORDERS	SYS_AI_18pkdxrps0j2m	O_D_ID	1
ORDERS	SYS_AI_3ys7c39vs247p	O_D_ID	1
ORDERS	SYS_AI_81dnzcja2qhp	O_W_ID	1
ORDERS	SYS_AI_81dnzcja2qhp	O_D_ID	1
ORDERS	SYS_AI_81dnzcja2qhp	O_C_ID	1
ORDERS	SYS_AI_97ya3cug4hxp	O_C_ID	1
ORDERS	SYS_AI_97ya3cug4hxp	O_ID	2
STOCK	SYS_AI_fdbazxb641kwv	S_W_ID	1

```

1 select index_name,table_name, auto,visibility,segment_created
2 from user_indexes
3 where auto='YES'
4* and visibility = 'VISIBLE'
SQL> /

```

INDEX_NAME	TABLE_NAME	AUT	VISIBILIT	SEG
SYS_AI_8k0xma30nayxn	CUSTOMER	YES	VISIBLE	YES
SYS_AI_0jfsy72532qv3	CUSTOMER	YES	VISIBLE	YES
SYS_AI_97ya3cug4hxp	ORDERS	YES	VISIBLE	YES

# New Execution Plan

## Plan Analysis

**Total cost:** 79,718 | **Total I/O cost:** 79,357 | **Total CPU cost:** 10,765,972,870

Plan Details

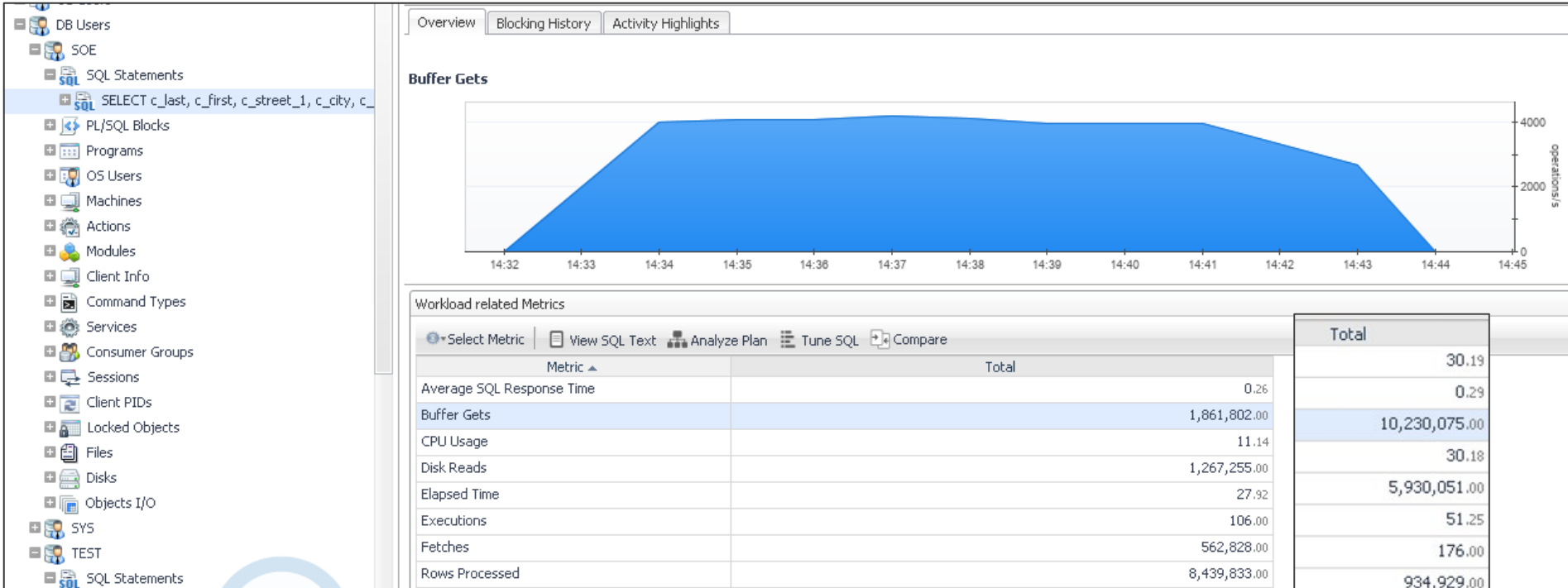
Operation Analysis

Object Analysis

Operation	Object Name	Object Type	Cost	CPU Cost	I/O Cost	Cardinality	Bytes
NESTED LOOPS			5.81 %	37,065,794	4,630	1,542	84,810
NESTED LOOPS			5.81 %	37,065,794	4,630	1,542	84,810
STATISTICS COLLECTOR			0.00 %	0	0	0	0
HASH JOIN			0.12 %	2,663,115	94	1,512	66,528
NESTED LOOPS			0.12 %	2,663,115	94	1,512	66,528
STATISTICS COLLECTOR			0.00 %	0	0	0	0
TABLE ACCESS BY INDEX ROWID BATCHED	SOE.DISTRICT	TABLE	0.01 %	44,979	6	5	60
INDEX SKIP SCAN	SOE.DISTRICT_I1	INDEX (UNIQUE)	0.00 %	8,121	1	5	0
INDEX RANGE SCAN	SOE.IORDL	INDEX (UNIQUE)	0.11 %	1,715,087	88	302	9,664
INDEX RANGE SCAN	SOE.IORDL	INDEX (UNIQUE)	0.11 %	1,715,087	88	3,023	96,736
INDEX RANGE SCAN	SOE.STOCK_IDX	INDEX	0.00 %	15,293	2	1	0
TABLE ACCESS BY INDEX ROWID	SOE.STOCK	TABLE	0.00 %	22,753	3	1	11
TABLE ACCESS FULL	SOE.STOCK	TABLE	0.00 %	22,753	3	1	11
TABLE ACCESS FULL	SOE.ORDERS	TABLE	10.10 %	2,086,361,577	7,985	15,116	256,972
INDEX RANGE SCAN	SOE.SYS_AI_8k0xma30nayxn	INDEX	0.01 %	298,486	4	1,350	0
TABLE ACCESS BY INDEX ROWID	SOE.CUSTOMER	TABLE	1.66 %	10,219,651	1,327	3	282
TABLE ACCESS BY INDEX ROWID BATCHED	SOE.CUSTOMER	TABLE	1.66 %	10,219,651	1,327	7,500	705,000
INDEX RANGE SCAN	SOE.SYS_AI_0jfsy72532qv3	INDEX	0.01 %	298,486	4	1,350	0



# Performance



# DBA Fine Tunes the Query

- create index orders\_i2 on orders(o\_id,o\_c\_id, o\_entry\_d);

PLAN_TABLE_OUTPUT							
Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time	
0	SELECT STATEMENT				64 (100)		
* 1	FILTER						
* 2	HASH JOIN		1	166	64 (0)	00:00:01	
* 3	HASH JOIN		1	72	17 (0)	00:00:01	
* 4	HASH JOIN		1	61	15 (0)	00:00:01	
* 5	HASH JOIN		300	13200	10 (0)	00:00:01	
6	TABLE ACCESS BY INDEX ROWID BATCHED	DISTRICT	5	60	6 (0)	00:00:01	
* 7	INDEX SKIP SCAN	DISTRICT_I1	5		1 (0)	00:00:01	
* 8	INDEX RANGE SCAN	IORDL	60	1920	4 (0)	00:00:01	
* 9	INDEX RANGE SCAN	ORDERS_I2	587	9979	5 (0)	00:00:01	
* 10	INDEX FAST FULL SCAN	STOCK_IDX1	1	11	2 (0)	00:00:01	
* 11	TABLE ACCESS FULL	CUSTOMER	4	376	47 (0)	00:00:01	

# Popular Airline Flights in USA

```
SELECT
o.carrier, uc.description AS carrier_name
,ao.description AS origin_airport,co.Description AS origin_city
,o.fl_date,o.fl_num,o.tail_num
,ad.description AS destination_airport
,cd.Description AS destination_city ,w.Description Day_of_Week
FROM t_ontime o
    INNER JOIN L_UNIQUE_CARRIERS uc ON uc.Code = o.UNIQUE_CARRIER
    INNER JOIN L_AIRPORT_ID ao ON ao.Code = o.ORIGIN_AIRPORT_ID
    INNER JOIN L_AIRPORT_ID ad ON ad.Code = o.DEST_AIRPORT_ID
    INNER JOIN L_CITY_MARKET_ID co ON co.Code = o.ORIGIN_CITY_MARKET_ID
    INNER JOIN L_CITY_MARKET_ID cd ON cd.Code = o.DEST_CITY_MARKET_ID
    INNER JOIN L_WEEKDAYS w ON w.Code = o.DAY_OF_WEEK
WHERE to_date(fl_date,'YYYY-MM-DD') BETWEEN &beg_date AND &end_date
AND co.Description = &city
AND w.Description = &day_of_week;
```

L\_UNIQUE\_CARRIERS: 1620  
L\_AIRPORT\_ID: 6438  
L\_CITY\_MARKET\_ID: 5823  
L\_WEEKDAYS: 8  
T\_ONTIME: 6784044

[US DOT - On-time Performance](#)

# No Other Option but Full Table Scans

Plan hash value: 633429076

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT				31176 (100)	
* 1	HASH JOIN		204	45696	31176 (1)	00:00:02
* 2	HASH JOIN		204	36924	31163 (1)	00:00:02
* 3	HASH JOIN		204	28152	31150 (1)	00:00:02
* 4	HASH JOIN		204	23256	31141 (1)	00:00:02
* 5	HASH JOIN		204	18156	31136 (1)	00:00:02
* 6	TABLE ACCESS FULL	L_WEEKDAYS	1	10	3 (0)	00:00:01
* 7	HASH JOIN		1426	110K	31133 (1)	00:00:02
* 8	TABLE ACCESS FULL	L_CITY_MARKET_ID	1	24	9 (0)	00:00:01
* 9	TABLE ACCESS FULL	T_ONTIME	429K	22M	31122 (1)	00:00:02
10	TABLE ACCESS FULL	L_UNIQUE_CARRIERS	1620	40500	5 (0)	00:00:01
11	TABLE ACCESS FULL	L_CITY_MARKET_ID	5823	136K	9 (0)	00:00:01
12	TABLE ACCESS FULL	L_AIRPORT_ID	6438	270K	13 (0)	00:00:01
13	TABLE ACCESS FULL	L_AIRPORT_ID	6438	270K	13 (0)	00:00:01

# Automatic Indexes

TABLE_NAME	INDEX_NAME	COLUMN_NAME	COLUMN_POSITION
L_AIRPORT_ID	SYS_AI_53zguxmr3ss0t	CODE	1
L_CITY_MARKET_ID	SYS_AI_f9bygtwdqmxm	CODE	1
L_CITY_MARKET_ID	SYS_AI_113vdqswmftr3	DESCRIPTION	1
L_UNIQUE_CARRIERS	SYS_AI_91yyf2dwquw7p	CODE	1
T_ONTIME	SYS_AI_d7c062aqxyz1v	ORIGIN_AIRPORT_ID	1
T_ONTIME	SYS_AI_76tkhqzqyhffq	ORIGIN_CITY_MARKET_ID	1
T_ONTIME	SYS_AI_a0y78qnzu4qrc	DEST_AIRPORT_ID	1
T_ONTIME	SYS_AI_4mdzc0pu2gk6p	DEST_CITY_MARKET_ID	1
T_ONTIME	SYS_AI_2qhg8k60a9gd3	DAY_OF_WEEK	1
T_ONTIME	SYS_AI_1jpp5cssdf0kr	UNIQUE_CARRIER	1

- Visible Indexes

L_CITY_MARKET_ID	SYS_AI_113vdqswmftr3	DESCRIPTION	1
L_AIRPORT_ID	SYS_AI_53zguxmr3ss0t	CODE	1
T_ONTIME	SYS_AI_76tkhqzqyhffq	ORIGIN_CITY_MARKET_ID	1
L_UNIQUE_CARRIERS	SYS_AI_91yyf2dwquw7p	CODE	1
L_CITY_MARKET_ID	SYS_AI_f9bygtwdqmxm	CODE	1

# Automatic Index Report

```
select DBMS_AUTO_INDEX.REPORT_LAST_ACTIVITY('TEXT','ALL','ALL') from dual
```

INDEX DETAILS			
1. The following indexes were created: *: invisible			
Owner	Table	Index	Key
TEST	L_AIRPORT_ID	SYS_AI_53zguxr3ss0t	CODE
TEST	L_CITY_MARKET_ID	SYS_AI_113vdqswmftr3	DESCRIPTION
TEST	L_CITY_MARKET_ID	SYS_AI_f9bygtwdqpxxm	CODE
TEST	L_UNIQUE_CARRIERS	SYS_AI_91yyf2dwquw7p	CODE
TEST	T_ONTIME	SYS_AI_76tkhqzqyhffq	ORIGIN_CITY_MARKET_ID
VERIFICATION DETAILS			
Parsing Schema Name : TEST			
SQL ID	: 429sumt3yahs9		
SQL Text	: SELECT O.CARRIER, UC.DESCRPTION AS CARRIER_NAME ,AO.DESCRPTION AS ORIGIN AIRPORT,CO.DESCRPTION AS ORIGIN_CITY ,O.FL_DATE,O.FL_NUM,O.TAIL_NUM ,AD.DESCRPTION AS DESTINATION AIRPORT ,CD.DESCRPTION AS DESTINATION_CITY ,W.DESCRPTION DAY OF WEEK FROM T_ONTIME O INNER JOIN L_UNIQUE_CARRIERS UC ON UC....		
Improvement Factor	: 1.2x		
Execution Statistics:			
	Original Plan	Auto Index Plan	
Elapsed Time (s):	195334030	885494	
CPU Time (s):	186004680	839174	
Buffer Gets:	12456957	36005	
Optimizer Cost:	4506	4506	
Disk Reads:	64548	0	
Direct Writes:	0	0	
Rows Processed:	927986	2317	
Executions:	284	1	

# New Plan

Plan hash value: 4160115658

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT				4506 (100)	
* 1	HASH JOIN		8	1792	4506 (1)	00:00:01
* 2	HASH JOIN					
* 3	HASH JOIN					
* 4	HASH JOIN					
* 5	HASH JOIN					
6	NESTED LOOPS					
7	NESTED LOOPS					
8	TABLE ACCESS BY INDEX ROWID BATCHED	L_CITY_MARKET_ID				
* 9	INDEX RANGE SCAN	SYS_AI_113vdqswmftr3				
* 10	INDEX RANGE SCAN	SYS_AI_76tkhqzqyhffq				
* 11	TABLE ACCESS BY INDEX ROWID	T_ONTIME				
* 12	TABLE ACCESS FULL	L_WEEKDAYS				
13	TABLE ACCESS FULL	L_UNIQUE_CARRIERS				
14	TABLE ACCESS FULL	L_CITY_MARKET_ID				
15	TABLE ACCESS FULL	L_AIRPORT_ID				
16	TABLE ACCESS FULL	L_AIRPORT_ID				

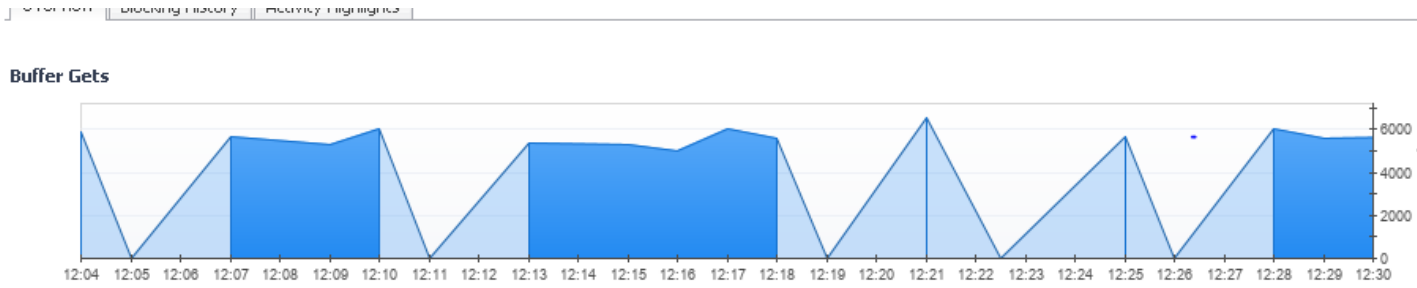
# Original Performance





# Auto Index Performance

- DB Users
- TEST
  - SQL Statements
  - SELECT O.CARRIER, UC.DESCRPTION AS C
  - SELECT xmlelement("plans", xmlelement("plan
  - PL/SQL Blocks
  - Programs
  - OS Users
  - Machines
  - Actions
  - Modules
  - Client Info
  - Command Types
  - Services
  - Consumer Groups
  - Sessions
  - Client PIDs
  - Locked Objects
  - Files
  - Disks
  - Objects I/O
  - SYS



## Workload related Metrics

Select Metric | View SQL Text | Analyze Plan | Tune SQL | Compare

Metric ▲	Total
Active Time	51.02
Average SQL Response Time	0.46
Buffer Gets	4,770,806.00
CPU Usage	51.02
Disk Reads	0.00
Elapsed Time	51.47
Executions	113.00
Fetches	22,790.00
Rows Processed	341,052.00

Total
1,274.08
10.69
19,096,132.00
1,251.74
13,800,356.00
1,347.11
126.00
26,899.00
402,643.00

Original

# Summary

- Automatic Indexing can speed up performance
  - 19c Optimizer has come along way
- Beware of just turning it on blindly
  - Especially in production
  - Watch out for baselines
  - Invisible versus Visible
- Consider using in development / test
  - Be cautious using in production
- Control at schema level (21c – allows control at table level)
- Turn on compression for space savings