

DB2 OS/390 SQL Tuning Tips and Techniques

Jim Wankowski
Quest Software

Agenda

- Introduction
- Understanding Access Paths
 - access paths determination
 - Single/Multi Table access methods
- Fundamentals of SQL Tuning
 - Predicate coding
 - Runstats
 - Interpreting the Plan table
- SQL Tuning Scenarios



Why SQL Tuning?

- Largest performance gains
- Lowest cost-benefit ratio
- Small number of SQL statements consume majority of resources

SQL

- DDL – Used for structure changes
- DCL – Grant and revoke statements
- DML – Select, Insert, Update, and Delete

- This presentation will focus primarily on DML Select statements

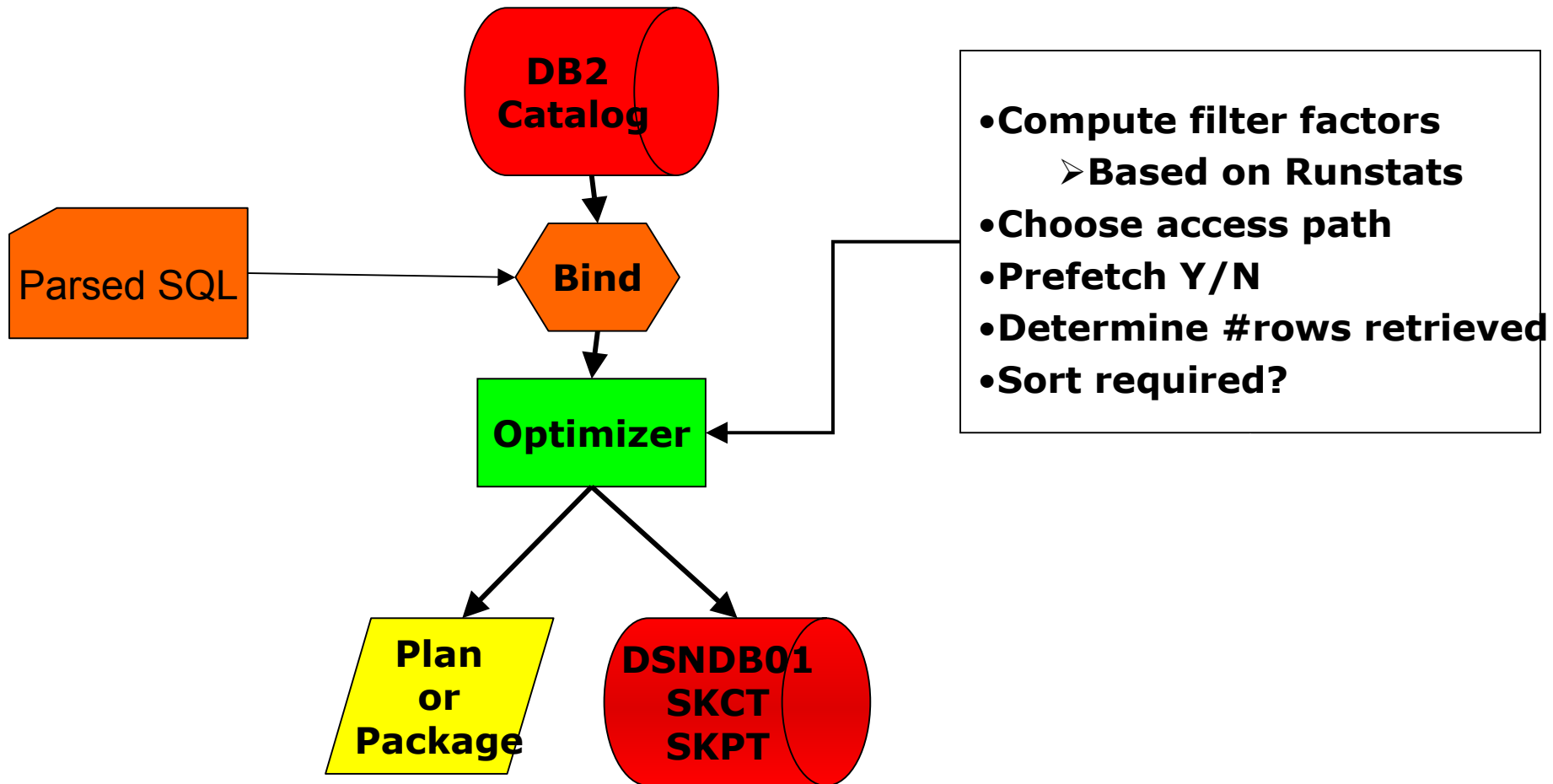
SQL Access Paths

- How is the data being read?
 - Tablespace Scan
 - Index Access
 - Index Only
 - Index Scan
 - Matching Index columns

Prefetching Data

- Sequential Prefetch
 - Generally used for TS scans
 - Sequential sets of pages read into bufferpool with single I/O
 - #pages determined by BP size
 - Can be determined at Bind time
- List Prefetch
 - Matching IX scan
 - Usually low cluster ratio
 - Sequential set of RIDs
 - Data pages retrieved based on RID list
 - Always used with Hybrid joins and MX access

How are access paths determined?



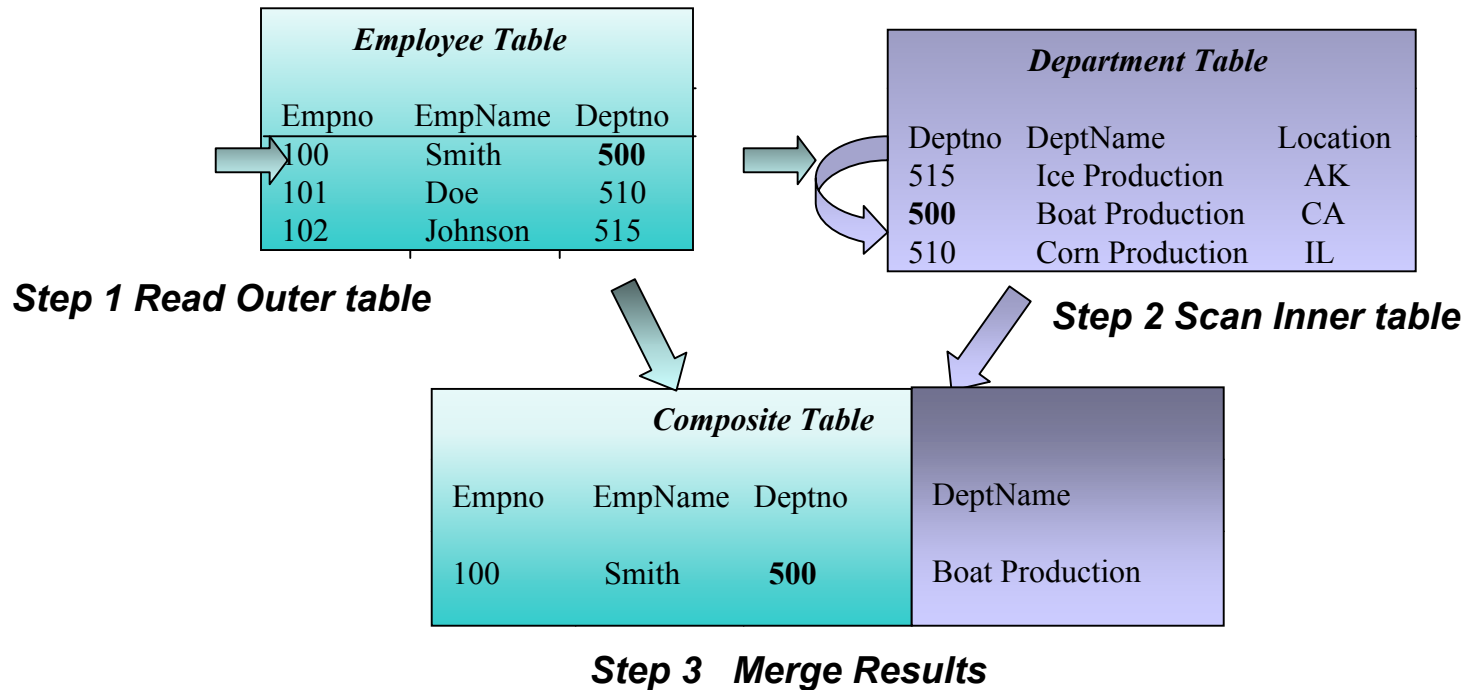
Single Table Access Paths

- **Tablespace Scan** – All pages within tablespace are retrieved
- **Matching Index Scan** – Predicate matches leading column or columns of an index
- **Multiple Index Access** – More than one index used to satisfy data access (RID list unions)
- **In-List Index Access** – Matching index scan where an 'In' predicate is used as a matching equal predicate
- **Non-matching Index Scan** – Predicate does not match leading columns in the index.
- **Index Only Access** – No data page retrieval is necessary
- **One Fetch Access** – Requires only one row be retrieved (Min or Max column function)

Multiple Table Access Methods

- Nested Loop Join
- Merge Scan Join
- Hybrid Join

Nested Loop Join



*Select A.Empno, A.Empname, A. Deptno, B.Deptname
from Employee A, Department B
where A.Deptno = B.Deptno;*

Nested Loop Join

- Recommended for
 - Small number of rows in outer table
 - Predicates filter out high percentage of outer table rows
 - Highly clustered index on inner table join predicate
 - Number of pages in inner table is small

Merge Scan Join

<i>Employee Table</i>		
Empno	EmpName	Deptno
100	Smith	500
101	Doe	510
102	Johnson	515

Step 1 Read Outer in join column order

<i>Department Table</i>		
Deptno	DeptName	Location
515	Ice Production	AK
500	Boat Production	CA
510	Corn Production	IL

Step 2 Sort Inner table by join column

<i>Sorted Department Table</i>		
Deptno	DeptName	Location
500	Boat Production	CA
510	Corn Production	IL
515	Boat Production	AK

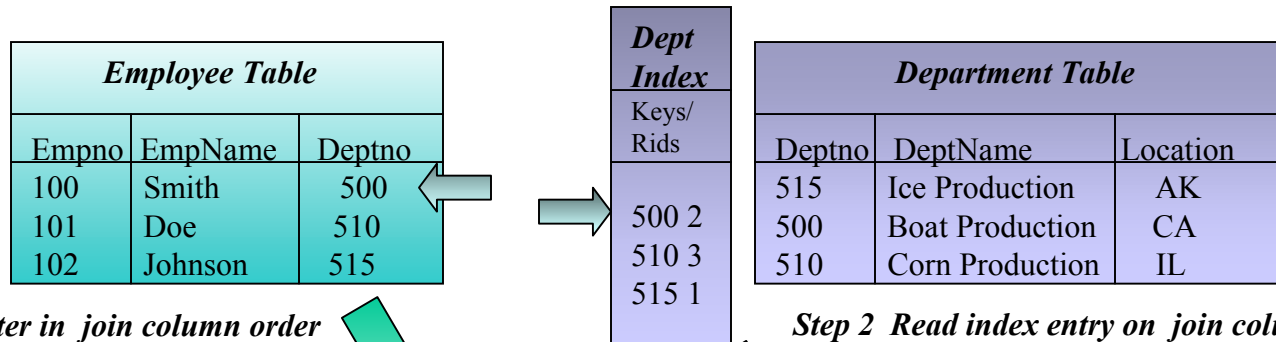
Step 3 Match Outer table to Inner table by join columns

<i>Composite Table</i>			
Empno	EmpName	Deptno	DeptName
100	Smith	500	Boat Production

Merge Scan Join

- Must have:
 - Join predicate on columns with same datatype and length
- Performance implication
 - Generally requires sorting of one or both tables
- Recommended for
 - Large number of qualifying rows in each table (many to many)
 - Large tables with no indexes with matching columns
 - Few columns in inner table selected, makes sort more efficient

Hybrid Join



The diagram illustrates the third step of a hybrid join process: Merge outer table data and inner Index RID. It features a table titled Intermediate Table Phase 1.

Empno	EmpName	Deptno	RIDS
100	Smith	500	2
101	Doe	510	3
102	Johnson	515	1

Step 3 Merge outer table data and inner Index RID

Hybrid Join

Step 4 Sort by RID entry to create a sorted RID list AND a phase 2 intermediate table

<i>Intermediate Table Phase 2</i>			
Empno	EmpName	Deptno	RIDS
102	Johnson	515	1
100	Smith	500	2
101	Doe	510	3

Step 5 Retrieve data from inner table using list prefetch

<i>Department Table</i>		
Deptno	DeptName	Location
515	Ice Production	AK
500	Boat Production	CA
510	Corn Production	IL

List Prefetch

Composite Table

Empno	EmpName	Deptno	DeptName
102	Johnson	515	Ice Production
100	Smith	500	Boat Production
101	Doe	510	Corn Production

Step 6 Merge intermediate phase 2 table with Inner table rows

Hybrid Join

- Must have:
 - Inner Join
 - Index on join column of inner table
- Recommended for
 - Inner tables with no clustering or low ratio clustering indexes on join columns
 - Outer table has many duplicate qualifying rows

Tuning SQL

SQL Tuning

- Elements affecting SQL Performance
 - Predicate design
 - Size of result set
 - Subqueries
 - Accuracy of Statistics
- Understanding the Plan table
- SQL tuning scenario

SQL Tuning - Predicates

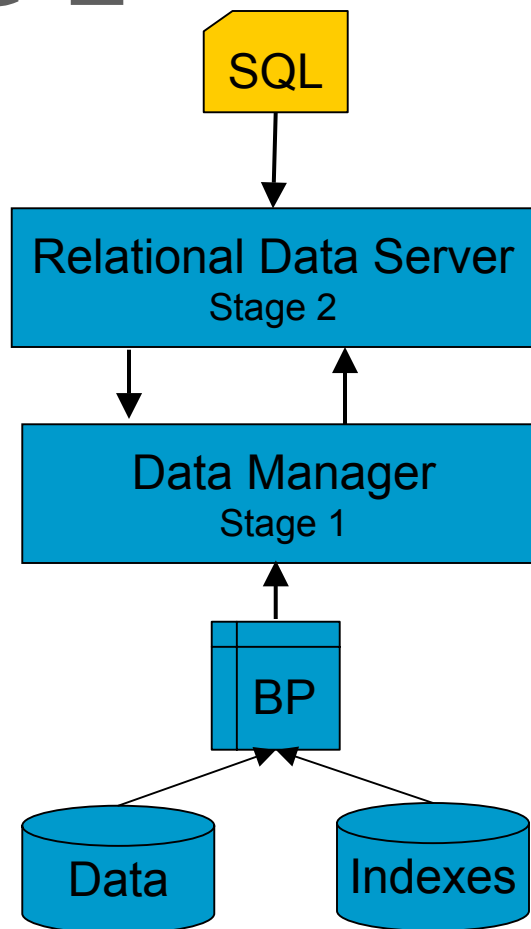
- Predicate Coding “Rules of Thumb”
 - Code most restrictive clauses first
 - Check that the predicate is indexable
 - Check that the predicate is Stage 1 (sargable)
 - Predicate Table included in Application Programming and SQL Guide
 - IN instead of Unions
 - Use Joins instead of Subqueries
 - Avoid unnecessary sorting
 - Host Variables
 - Ensure length and precision match column
 - Consider using REOPT(VARS) to re-optimize at runtime
 - Redundant
 - Remove redundant or unnecessary predicates

SQL Tuning – Result Sets

- Restrictions on result set
 - Only select columns that are required
 - 'Select *'
 - Optimize for 'n' rows
 - Use when only small percentage of rows are fetched
 - Avoids sorts
 - Make Predicates as restrictive as possible

Stage 1 vs. Stage 2

- Use Stage 1 whenever possible
 - Filters data before retrieving
 - Minimizes I/O
 - All indexable predicates are Stage 1
- Stage 1 refers to the DM (data manager)
- Stage 2 refers to RDS (relational data system).
- DM processes pages in the bufferpool and passes them to RDS. The more rows DM must pass to RDS to complete a search the more costly.
- Refer to DB2 Administration Guide for table of Stage 1 indexes.



Subqueries

- Correlated subqueries
 - The subquery refers to a column in the outer query
 - The subquery is executed for each qualified row of the outer query
 - Always Stage 2 processing
- Noncorrelated subqueries
 - The subquery has no reference to column in outer query
 - Executed once when cursor is opened
- Available indexes determine which is most efficient
- Use Joins instead of subqueries whenever possible.

Stats used for Access Path Determination

- **SYSCOLDIST**
 - COLVALUE
 - FREQUENCYF
 - TYPE
 - CARDF
 - COLGROUPOCOLNO
 - NUMCOLUMNS
- **SYSCOLUMNS**
 - COLCARDF
 - HIGH2KEY
 - LOW2KEY
- **SYSINDEXES**
 - CLUSTERING
 - CLUSTERRATIOF
 - FIRSTKEYCARDF
 - FULLKEYCARDFNLEAF
 - NLEVELS
- **SYSINDEXPART**
 - LIMITKEY
- **SYSROUTINES**
 - IOS_PER_INVOC
 - INSTS_PER_INVOC
 - INITIAL_IOS
 - INITIAL_INSTS
 - CARDINALITY
 - SYSTABLES
 - CARDF
 - EDPROC
 - NPAGES
 - PCTROWCOMP
- **SYSTABLESPACE**
 - NACTIVEF
- **SYSTABSTATS**
 - CARDF
 - NPAGES

RUNSTATS

- **When to run RUNSTATS:**
 - ✓ After LOAD, REORG, and REBUILD IX
 - ✓ After creating new index
 - ✓ After heavy insert, update, delete activity
- **Performance Tip:**
 - ✓ Only run RUNSTATS against partitions which have changed.
 - ✓ Consider using the SAMPLE option for column statistics to reduce processor costs
 - Be sure to check access paths when using SAMPLE option

Monitor Runstats

- REBIND after RUNSTATS?
 - ✓ New index created
 - ✓ CLUSTERRATIOF changes $<>$ 80%
 - ✓ Changes more than 20%
 - Cardinality
 - NPAGES
 - NACTIVEF
 - NLEAF
 - Range of HIGH2KEY to LOW2KEY

Interpreting the Plan Table

TNAME	ACCESSTYPE	MATCH COLS	ACCESSNAME	INDEX ONLY	PREFETCH
DB2 Table Name	I = Index	If accesstype = I, I1, N, or MX the number of index key columns used in the index scan	Accesstype = I, I1, N, or MX the name of the index	Y = Yes N = No Whether data access was required	S = sequential data pages L = list of pages Pages are read in advance
	I1 = One fetch index scan				
	N = In list index scan				
	R = Tablespace scan				
	M = Multiple Index scan				
	MX = Index scan				
	MI = Intersection of multiple indexes				
MU = Union of multiple indexes					
Emp	I	2	Employee_ix	N	

*Example – Select * from Emp where empno = '000010' and workdept = 'A01';*

SQL Access Paths

Tname	Accesstype	Matchcols	Accessname	Index only	Prefetch
Emp	R	0		N	S
<i>No index on predicate...tablespace scan with prefetch</i>					
Emp	I	1	XEMP1	N	
<i>Index on predicate column results in index and tablespace access</i>					

Example – Select * from Emp

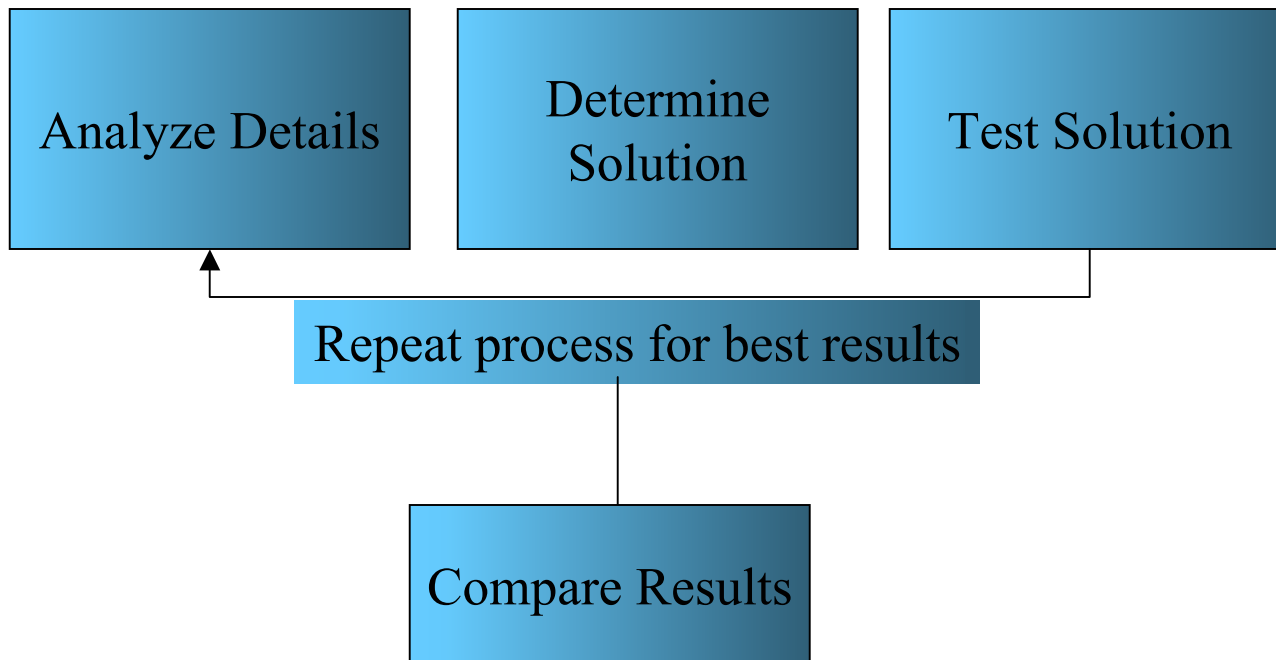
where empno = '000010' and workdept = 'A01';

Efficiency of Access Paths

- Single table Queries
 1. Matching Unique IX(=)
 2. IX only
 3. Multi-IX Access
 4. Matching IX Scan
 5. Non-Matching IX Scan
 6. Segmented TS Scan
 7. Non-Segmented TS Scan
- Multi-Table Queries
 1. Merge Scan Join
 - Very large tables
 2. Hybrid Join
 - Medium size tables
 3. Nested Loop Join
 - Small Tables

SQL Tuning Session

- Let's Tune a SQL Statement



Quest Central's SQL Tuning

Step 1: Analyze Details

- Advanced SQL Tuning Environment
 - Provides Access Path and object statistics
 - In-Context object statistics display
- Select access path step and associated tables, indexes, and columns are highlighted
 - User Selected object statistics display
 - Select table and associated indexes and columns are displayed
 - Select index and associated columns are displayed

The screenshot shows the Quest Central SQL Tuning interface. The main window title is "Quest Central - [SQL Tuning on qsfA-SUBSYS-DSN6]". The menu bar includes File, Edit, View, Tools, Window, and Help. The toolbar contains various icons for file operations and SQL execution. The main area displays a SQL query: `select * from dsn8610.emp, dsn8610.dept;`. Below the query, there are buttons for "Explain", "Execute", and "Both". The "Messages" pane at the bottom shows the results of the "Explain" and "Execute" operations. The "Explain" message is "Explain Successful - 10:10:26 AM Qualifier: DJENSON". The "Execute" message is "Execute Successful - 10:10:27 AM 200 Rows Retrieved (Row Limit: 200) Time (h:m:s.ms) : 0:0:1.683". The status bar at the bottom shows "Qualifier: DJENSON" and "Connection: qsfA-SUBSYS-DSN6 (DSN6)".

SQL – Entered from diagnostics, file, View, SQL Editor, Plan/Package, or free form

Messages | Host Variables

Explain	Execute
Explain Successful - 10:10:26 AM Qualifier: DJENSON	Execute Successful - 10:10:27 AM 200 Rows Retrieved (Row Limit: 200) Time (h:m:s.ms) : 0:0:1.683

Original SQL

Qualifier: DJENSON | Connection: qsfA-SUBSYS-DSN6 (DSN6)

Explorin... | SQL T...

The screenshot displays the Quest Central SQL Tuning interface for a query on the 'qsfa-SUBSYS-DSN6' database. The interface is divided into several panes:

- Explain Access Path:** Shows the execution plan for the query. The plan includes a 'Nested loop join' operation. The access paths are: 1. Table access full DSN8610.EMP, 2. Table access full DSN8610.DEPT.
- Table Statistics:** A table showing statistics for the tables involved in the query.
- Index Statistics:** A table showing statistics for the indexes on the tables.
- Column Statistics:** A table showing statistics for the columns used in the query.

At the bottom of the interface, there is a text box stating: "All rows in the table will be examined to satisfy the query".

Object name	Type	Rows	Pages	Statistics time	CardF	Rec length	PCT rows comp	RI pare	RI childre	Key unique	Key c
DSN8610.EMP	Table	42	2	1999-02-05 09:33:29	42.00	107	0	1	3	0	1
DSN8610.DEPT	Table	14	1	1999-02-05 09:31:35	14.00	76	0	2	3	0	1

Index name	Type	Unique rule	Leaf	Levels	Full keycard	First keycard	Full keycardF	First keycardF	Cluster r
DSN8610.XEMP1	Type 1	Unique (P)	2	2	42	42	0.00	0.00	100
DSN8610.XEMP2	Type 1	Duplicates	0	0	0	0	0.00	1.1556778884	0

Column name	Data type	User type	Foreign Key	Index order	Key seq	Cardinality	ColcardF	Higt
EMPNO	CHAR (6)	No	S	N/A	0	42	42.00	F2F
FIRSTNME	VARCHAR (12)	No	S	N/A	0	-1	-1.00	404
MIDINIT	CHAR (1)	No	S	N/A	0	-1	-1.00	404
LASTNAME	VARCHAR (15)	No	S	N/A	0	-1	-1.00	404
WORKDEPT	CHAR (3)	No	S	N/A	0	0	0.00	000
PHONENO	CHAR (4)	No	S	N/A	0	-1	-1.00	404
HIREDATE					0	-1	-1.00	404

- Step 2: Determine Solution
 - In-Context Advice
 - Provides advice items
 - Provides advice actions in-context to the advice items
 - Allows users to choose an advice action and product rewrites the SQL to incorporate the advice item
 - Options Include:
 - Display object advice
 - Display advice by severity levels

The screenshot displays the Quest Central SQL Tuning interface. The window title is "Quest Central - [SQL Tuning on qsfA-SUBSYS-DSN6]". The menu bar includes File, Edit, View, Tools, Window, and Help. The toolbar contains various icons for file operations and SQL-specific actions. The main interface is divided into several sections:

- SQL:** Contains the query:

```
select *  
from dsn8610 . emp , dsn8610 . dept
```
- Advice:** Lists several advice items:
 - S2 - Qualified Tables
 - S3 - Parallelism disabled
 - S2 - Select "*" (highlighted)
 - S2 - Subselect without predicates
 - S1 - Missing join predicates (highlighted)
- Advice actions:** Contains three checkboxes:
 - DEPT.DEPTNO = EMP.WORKDEPT *Data match
 - DEPT.MGRNO = EMP.EMPNO *Data match
 - DEPT.ADMRDEPT = EMP.WORKDEPT *Data match
- Advice description:** States: "The SQL is performing a join between tables DSN8610.DEPT and DSN8610.EMP. No join predicates exist between these tables, resulting in a cartesian product, which utilizes excessive system resources. Consider adding a join predicate to eliminate the cartesian product."

On the left side, a diagram shows a "Select Statement" node with a green arrow pointing to it, which branches into "Full Select" and "SubSelect" nodes. The "SubSelect" node is highlighted in red.

At the bottom, the "Original SQL" tab is active. The status bar shows "Qualifier: DJENSON" and "Connection: qsfA-SUBSYS-DSN6 (DSN6)".

Advice describes and optionally edits SQL

- Step 3: Test Solution
 - Scenarios
 - Provides an entirely new SQL Tuning environment based on a modified SQL Statement
 - Advantage – allows user to retain original SQL and make one change at a time to allow for comparison of results
 - New scenarios can be created from the toolbar or the scenario tab
 - New scenarios can be generated from within the advice tab

Quest Central SQL Tuning

The screenshot shows the Quest Central for DB2 interface. The main window displays a query plan for a query on the 'qsfa-SUBSYS-DSN6' database. The plan shows a 'Hybrid join' operation (step 5) which involves 'Fetch table data DSN8710.DEPT' (step 4) and 'Fetch table data DSN8710.EMP' (step 2). Step 4 uses a 'Matching index scan DSN8710.XDEPT1', and step 2 uses a 'Non-matching index scan DSN8710.XEMP2'. The execution results table shows the following data:

Object name	Type	Rows	Pages	Statistics time	CardF	Rec length	PC
DSN8710.EMP	Table	42	2	2001-05-09 10:21:39	42.00	107	0
DSN8710.DEPT	Table	14	1	2001-03-16 14:16:09	14.00	76	0

Below the execution results, there is a table showing index statistics:

Index name	Type	Unique rule	Leaf	Levels	Full keycard	First keycard
DSN8710.XEMP1	Type 1	Unique (P)	2	2	42	42
DSN8710.XEMP2	Type 1	Duplicates	1	2	8	8
DSN8710.XEMP3	Type 1	Duplicates	1	2	42	42

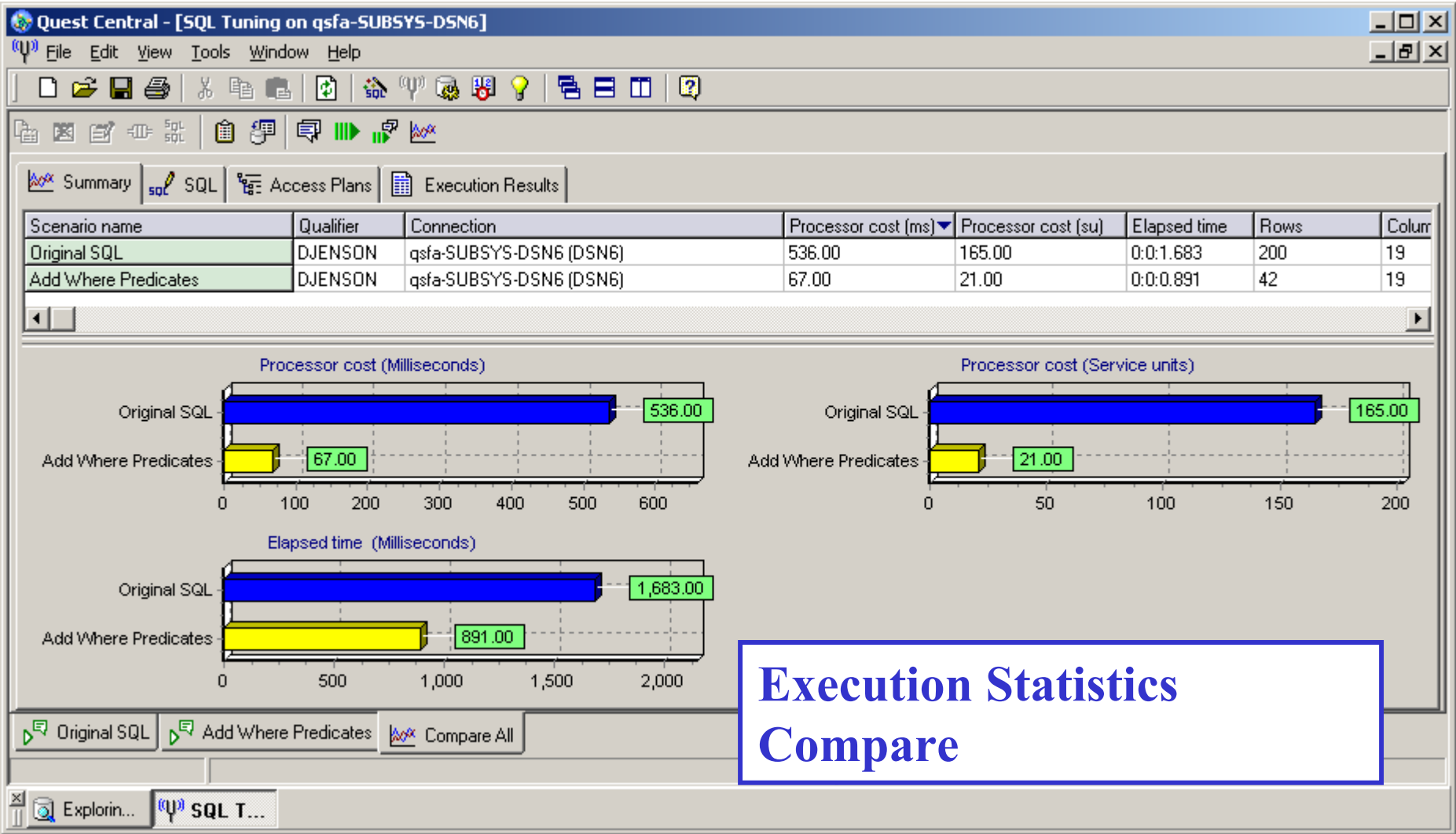
At the bottom of the interface, there is a table showing column statistics:

Column name	Data type	User type	Foreign	Index order	Key seq	CardinalityF	Card
WORKDEPT	CHAR (3)	No	S	Ascending	0	8.00	8
EMPNO	CHAR (6)	No	S	N/A	0	42.00	42
FIRSTNME	VARCHAR (12)	No	S	N/A	0	39.00	39
MIDINIT	CHAR (1)	No	S	N/A	0	20.00	20

The interface also shows a message: 'Index access begins at the leaf pages'. At the bottom, there are buttons for 'Original SQL' and 'Add Where Predicate'. A blue arrow points from the text box to the 'Add Where Predicate' button.

Scenarios allow for modifying original SQL

- Step 4: Compare Results
 - Compare Engine
 - Immediately detect which scenario performs best
 - Compares execution statistics
 - Compares SQL statement text
 - Compares access plans
 - Compare execution results
 - Ensure same amount of data is being returned



Execution Statistics Compare

The screenshot shows the Quest Central for DB2 SQL Tuning interface. The main window is titled "SQL Tuning on qsa-SUBSYS-DSN6". It features a menu bar (File, Edit, View, Tools, Window, Help) and a toolbar with various icons. Below the toolbar are tabs for "Summary", "SQL", "Access Plans", and "Execution Results".

A comparison table is displayed, comparing two scenarios:

Scenario name	Fetches	Filter rows	Generate rows	Group bys	Hash joins	Index ANDing	Index scans	Merge scan joins	Nested loop joins
Original SQL	0	0	0	0	0	0	0	0	1
Add Where Predicate	2	0	0	0	0	0	2	0	0

Below this table are two execution plan panels. The left panel, titled "Original SQL", shows a plan with 5 steps, including "Table access full" for DSN8710.EMP and DSN8710.DEPT, and a "Nested loop join". The right panel, titled "Add Where Predicate", shows a plan with 7 steps, including a "Non-matching index scan" for DSN8710.XEMP2, a "Fetch table data" for DSN8710.EMP, a "Matching index scan" for DSN8710.XDEPT1, and a "Fetch table data" for DSN8710.DEPT. The total cost for the original plan is 536.00, and for the tuned plan, it is 57.00.

A blue-bordered box with the text "Access Path Compare" is overlaid on the bottom center of the screenshot.

At the bottom of the window, there are tabs for "Original SQL", "Add Where Predicate", and "Compare All". The taskbar at the very bottom shows "Explor...", "SQL Edit...", and "SQL T...".

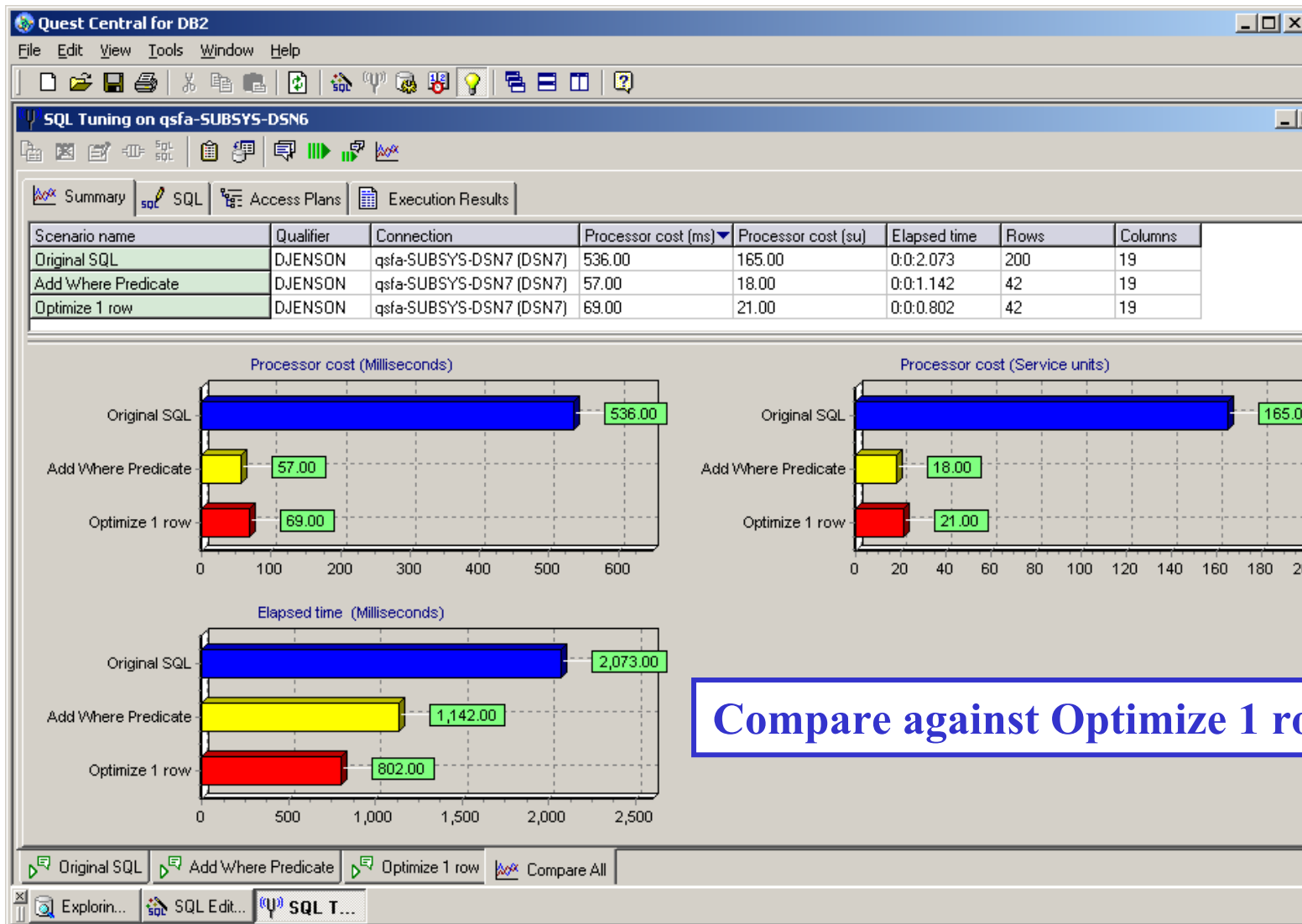
The screenshot shows the Quest Central SQL Tuning interface. The window title is "Quest Central - [SQL Tuning on qsfA-SUBSYS-DSN6]". The menu bar includes File, Edit, View, Tools, Window, and Help. The toolbar contains various icons for file operations and SQL execution. The main area is divided into several sections:

- Summary:** Shows "Original SQL" with 19 columns, 200 rows, and a time of 0:0:1.683. It also shows "Add Where Predicates" with 19 columns, 42 rows, and a time of 0:0:0.891.
- Comparison:** A box indicates "Column Count Same", "Row Count Different", and "Data Different".
- Execution Results Compare:** A blue-bordered box with the text "Execution Results Compare".
- Original SQL Table:** A table with columns EMPNO, FIRSTNME, MIDINIT, LASTNAME, and WORKDEPT. It contains 7 rows, all with EMPNO 000010 and FIRSTNME CHRISTINE.
- Add Where Predicates Table:** A table with the same columns. It contains 7 rows with various employee names and departments.

At the bottom, there are buttons for "Original SQL", "Add Where Predicates", and "Compare All". The taskbar shows "Explorin..." and "SQL T...".

Prove SQL concepts using Quest Central

Quest Central SQL Tuning



Subselect vs. Join

The screenshot shows the Quest Central for DB2 SQL Tuning interface. The main window displays a SQL query with subselects. The query is as follows:

```
SELECT
  A.EMPNO, A.FIRSTNME, A.LASTNAME,
  A.WORKDEPT, B.DEPTNAME
FROM
  DSN8710.EMP A,
  DSN8710.DEPT B
WHERE
  B.DEPTNO = A.WORKDEPT
AND
  A.EMPNO IN
  (SELECT C.EMPNO FROM DSN8710.EMPPROJECT C
   WHERE C.PROJNO = 'AD3100')
AND
  A.EMPNO IN
  (SELECT C.EMPNO FROM DSN8710.EMPPROJECT C
   WHERE C.ACTNO = 10)
```

The interface includes a menu bar (File, Edit, View, Tools, Window, Help), a toolbar, and a tabbed interface with 'SQL' selected. On the right side, there are buttons for 'Explain', 'Execute', and 'Both'. Below the SQL editor, there are two panels: 'Messages' and 'Host Variables'. The 'Messages' panel shows the following output:

Messages	Host Variables
Explain Explain Successful - 1:08:21 PM Qualifier: DJENSON	Execute Execute Successful - 1:08:22 PM 1 Row Retrieved (Row Limit: 200) Time (h:m:s.ms): 0:0:0.661

At the bottom of the window, there are buttons for 'Original SQL', 'Join Predicates', and 'Compare All'. The taskbar at the very bottom shows 'Explorin...', 'SQL T...', and 'SQL Edit...'.

The screenshot shows the Quest Central for DB2 SQL Tuning interface. The main window displays the following SQL query:

```
SELECT A.EMPNO , A.FIRSTNME , A.LASTNAME , A.WORKDEPT , B.DEPTNAME
FROM DSN8710.EMP A , DSN8710.DEPT B
WHERE B.DEPTNO = A.WORKDEPT
AND A.EMPNO IN (
SELECT C.EMPNO
FROM DSN8710.EMPPROJACT C
```

The interface includes a tree view on the left showing the query structure: Select Statement, Full Select, and three SubSelect nodes. The bottom of the window shows tabs for 'Original SQL', 'Join Predicates', and 'Compare All'. The 'SQL T...' tab is active.

The 'Advice' pane on the right lists several recommendations:

- S2 - Qualified Tables
- S3 - Redundant index(es)
- S2 - Unique Index(es) defined as non-unique
- S3 - Parallelism disabled
- S3 - Indexable and sargable predicate(s)
- S1 - Table access without index selectivity
- S2 - Identified join predicates
- S2 - Subquery transformations
- S2 - Subquery transformations
- S2 - Non-Indeatable and Non-Sargable Predi...

The 'Advice actions' pane shows the specific advice: 'A.EMPNO IN (SELECT C.EMPNO FROM DSN8710.EMPPROJACT C WHERE...'. A blue callout box with an arrow pointing to the 'S2 - Subquery transformations' entry contains the text: 'Advice recommends Rewriting as a join'.

The 'Advice description' pane at the bottom explains: 'The statement contains an IN predicate with a subquery. Because none of the columns in the select list are being accessed via a function, you can be able to rewrite the subquery as a join.'

The screenshot displays the Quest Central for DB2 SQL Tuning interface. The main window shows a SQL query being tuned. A blue box labeled "Rewritten Query" is overlaid on the original query text. The original query is as follows:

```
SELECT
  A.EMPNO, A.FIRSTNME, A.LASTNAME,
  A.WORKDEPT, B.DEPTNAME
FROM
  DSN8710.EMP A,
  DSN8710.DEPT B,
  DSN8710.EMPPROJACT C
WHERE
  B.DEPTNO = A.WORKDEPT
  AND A.EMPNO = C.EMPNO
  AND C.PROJNO = 'AD3100'
  AND C.ACTNO = 10
```

The interface includes a menu bar (File, Edit, View, Tools, Window, Help), a toolbar with various icons, and a toolbar for the SQL Tuning window (Access Plan, Plan Statistics, Plan Dependencies, Execution Results, Advice). On the right side, there are buttons for "Explain", "Execute", and "Both". At the bottom, there are tabs for "Messages" and "Host Variables", and a status bar with buttons for "Original SQL", "Join Predicates", and "Compare All".

The "Messages" pane shows the following output:

```
Explain
Explain Successful - 1:15:23 PM
Qualifier: DJENSON
```

The "Execute" pane shows the following output:

```
Execute
Execute Successful - 1:15:24 PM
1 Row Retrieved (Row Limit: 200)
Time (h:m:s.ms): 0:0:0.521
```

Quest Central SQL Tuning

The screenshot displays the Quest Central for DB2 SQL Tuning interface. The main window shows a comparison between two SQL scenarios: 'Original SQL' and 'Join Predicates'. The interface includes a menu bar (File, Edit, View, Tools, Window, Help), a toolbar, and a tabbed view with 'Summary', 'SQL', 'Access Plans', and 'Execution Results' selected. A table at the top provides a detailed comparison of the two scenarios across several metrics. Below the table are three horizontal bar charts: 'Processor cost (Milliseconds)', 'Processor cost (Service units)', and 'Elapsed time (Milliseconds)'. Each chart compares the 'Original SQL' (blue bar) and 'Join Predicates' (yellow bar). The 'Join Predicates' scenario consistently shows significantly lower values across all metrics, indicating better performance. A blue callout box on the right side of the interface contains the text 'Compare join vs. subselect'.

Scenario name	Qualifier	Connection	Processor cost (ms)	Processor cost (su)	Elapsed time	Rows	Columns
Original SQL	DJENSON	qsfa-SUBSYS-DSN7 (DSN7)	122.00	38.00	0:0:0.661	1	5
Join Predicates	DJENSON	qsfa-SUBSYS-DSN7 (DSN7)	5.00	2.00	0:0:0.521	1	5

Processor cost (Milliseconds)

Scenario	Processor cost (ms)
Original SQL	122.00
Join Predicates	5.00

Processor cost (Service units)

Scenario	Processor cost (su)
Original SQL	38.00
Join Predicates	2.00

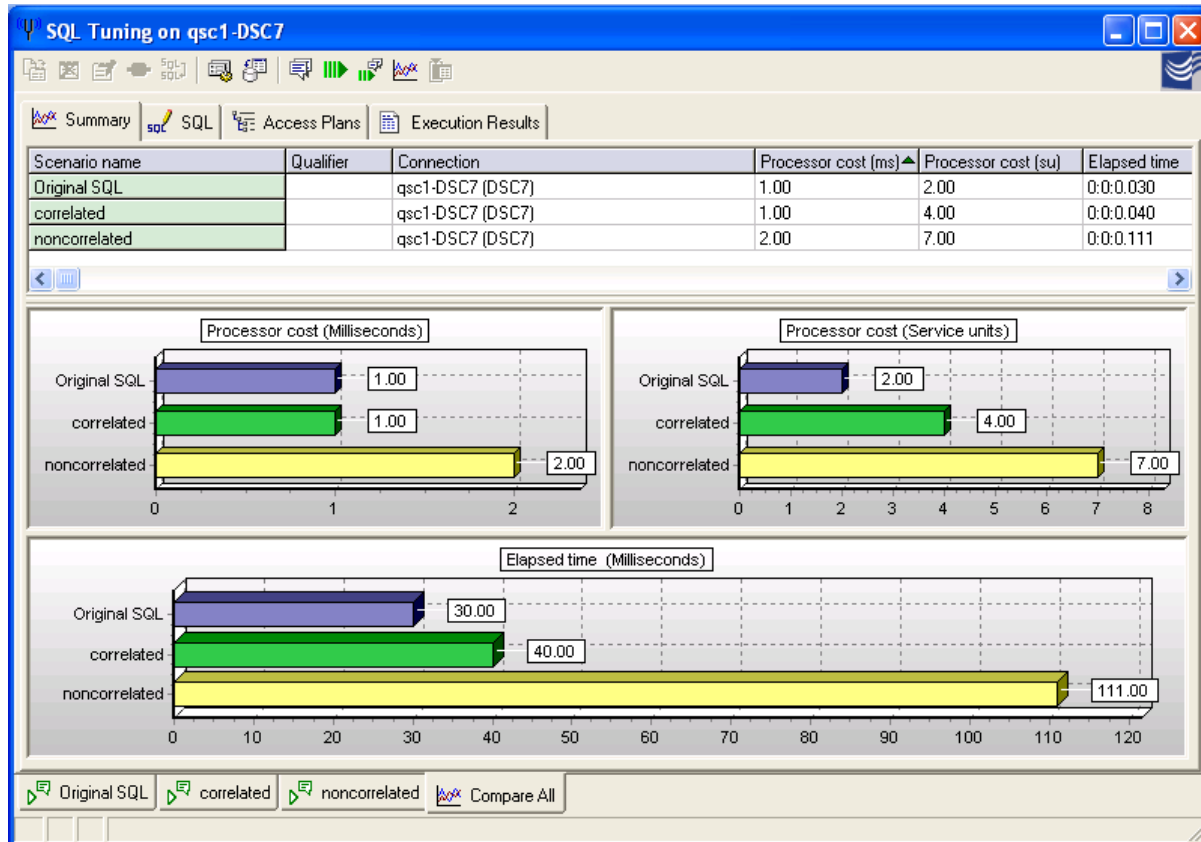
Elapsed time (Milliseconds)

Scenario	Elapsed time (ms)
Original SQL	661.00
Join Predicates	521.00

Compare join vs. subselect

Quest Central SQL Tuning

Join vs. Correlated vs. Noncorrelated subquery



Union vs. IN vs. Between

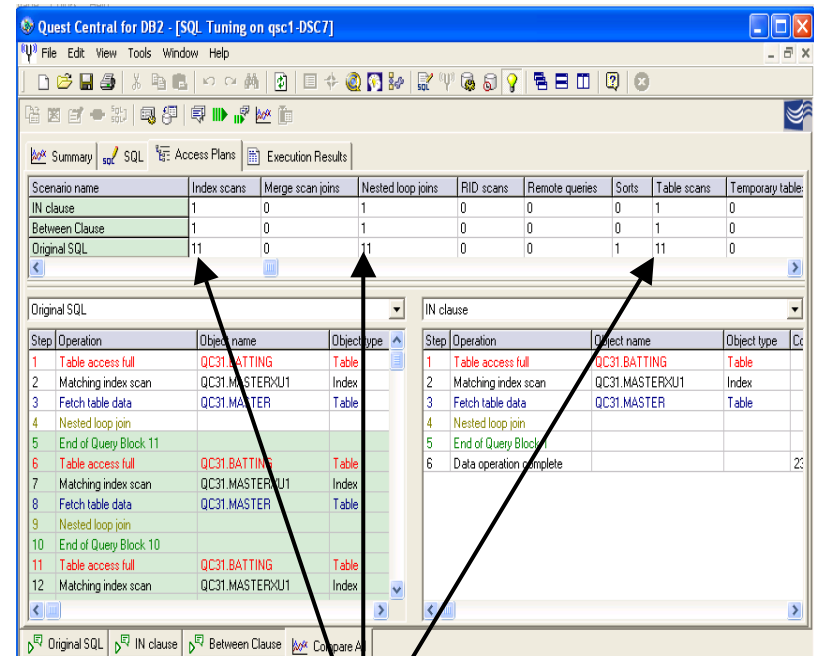
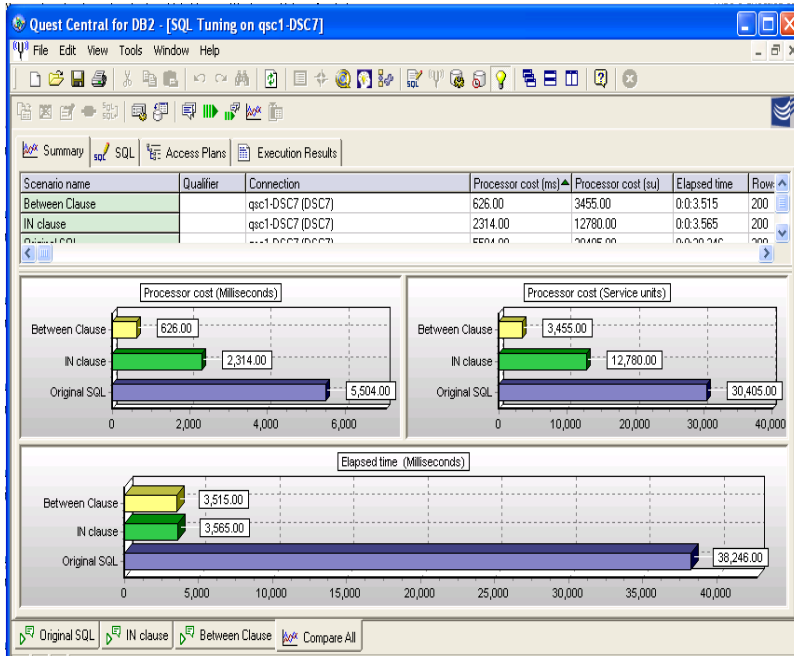
The screenshot shows the Quest Central for DB2 interface. The main window displays a query execution plan for a query on qsc1-DSC7. The plan shows a series of nested loop joins (39, 41, 42, 43, 44, 46, 47, 48, 49, 51, 52, 53, 54, 55) involving tables QC31.MASTER and QC31.BATTING. The plan indicates that data is sorted and that full table access is used for QC31.BATTING. The status bar at the bottom shows the connection is qsc1-DSC7 (DSC7).

Object name	Type	Rows	Pages	Status
QC31.MASTER	Table	15350	222	2003
QC31.BATTING	Table	78881	1830	2003

Index name	Type	Unique rule	Leaf	Levels	Full
QC31.BATIX	Type 2	Duplicates	474	3	733

Column name	Data type	User type	Foreign key	Index
RECNUM	INTEGER (4)	No		N/A
LAHMANID	VARCHAR (12)	No	S	N/A
DATE YYYY	SMALLINT (2)	No		N/A

Compare Results



Clearly determine that UNION is the least Efficient way to write this query

Quest Central for DB2 S/390

- Summary of Quest Central 3.1.1 and OS/390
 - Database Administration
 - Space Management
 - Performance Diagnostics
 - SQL Tuning
- Who says managing DB2 on OS/390 can't be fun?!

THANK YOU FOR LISTENING

Questions? Contact Jim Wankowski.
jim.wankowski@quest.com

Download free trial of Quest Central, whitepapers, and
technical presentations at:

WWW.Quest.Com/DB2

