DB2 SQL Tuning Tips for Developers Webinar

Tony Andrews
tandrews@themisinc.com

Twitter

Follow @ThemisTraining

Themis
Leaders in IT Education

Questions?

I will try my best to get to some questions towards the end of the webinar.

You can submit questions by typing into the questions area of your webinar control panel.

Any questions not answered due to time constraints can be answered afterward via an email.

Presentation will be added to our Themis website under ‘Webinar’ at top of main page. www.themisinc.com

Webinar Objectives

- Learn what makes queries, programs, and applications perform poorly
- Learn what you can do as a developer to improve performance
- Better understand what SQL optimization is
- What to do when you see table scans in a query
- Teach developers the different types of predicates
- Learn the difference between indexable and non- indexable predicates
- Learn why data statistics and 'Knowing Your Data' is so important
- Learn the top steps to tuning a query or program
- Leave with many SQL standards and guidelines for development

Themis
Leaders in IT Education

What are some of the key areas that can cause performance issues within applications, programs, and queries?

- Bad coding practices. Poorly coded SQL
  - Non indexable predicates
  - Stage 2 / Residual predicates
  - SQL doing more than it needs (extra tables, extra sorts, etc.)
- Wrong access path / Poor access path. Watch out for table scans!!
- Poor index design (Low Cardinality, Redundancy, Column order, etc). Know the application's workload!
- Too much synchronous I/O
- Too many calls to DB2 from program logic

Themis
Leaders in IT Education

What are some of the key areas that can cause performance issues within applications, programs, and queries?

- Large sorts. Know your data when you see a sort!
- Unneeded materialization of data
- Too much lock contention
- Statistics out of date (especially in test environments). Need a good test environment with production statistics and enough data to compare performance tests.
- Wrong clustering order of data

What’s Best ?

1). Index Only queries
2). Accurate data distribution statistics
3). Accurate estimates from optimizer on number of rows to be returned
4). Minimal runtime back-and-forth conversation with DB2
5). No functions on columns in Join predicates or Where logic
6). No table scans
7), No index scans
8). No sorts
9). Alternate ways to code SQL logic for (Exists/Not Exists, Summarized
data, Use of Self Joins, etc.).
10). Use of ‘For read Only’ and ‘With UR’ whenever possible
11). Use of ‘Fetch First XX Rows Only’ whenever possible .
12). Correct clustering order of data
13). Know your data! Especially non-uniform distributions for columns.

Bad Coding Practice
SQL Tip

1). Take out any / all Scalar functions coded on columns in predicates.

For example, this is the most common:

```
SELECT EMPNO, LASTNAME
FROM EMPLOYEE
WHERE YEAR(HIREDATE) = 2005
```

Should be coded as:

```
SELECT EMPNO, LASTNAME
FROM EMPLOYEE
WHERE HIREDATE BETWEEN ‘2005-01-01’ and ‘2005-12-31’
```

V9: Can now create indexes on SQL expressions.
V11: Optimizer actually does this date rewrite now (and others!)

Themis
Leaders in IT Education

Bad Coding Practice
SQL Tip

1). Take out any / all Scalar functions coded on columns in predicates.

For example, this is the most common:

```
SELECT EMPNO, LASTNAME
FROM EMPLOYEE
WHERE HIREDATE + 7 DAYS > CURRENT DATE
```

Should be coded as:

```
SELECT EMPNO, LASTNAME
FROM EMPLOYEE
WHERE HIREDATE > CURRENT DATE -  7 days
```

V9: Can now create indexes on SQL expressions.

Bad Coding Practice
SQL Tip

1). Take out any mathematics coded on columns in predicates.

For example, this is the most common:

```
SELECT EMPNO, LASTNAME
FROM EMPLOYEE
WHERE SALARY * 1.1 > ?
```

Should be coded as:

```
SELECT EMPNO, LASTNAME
FROM EMPLOYEE
WHERE SALARY > ?  /  1.1
```

V9: Can now create indexes on SQL expressions.

V11 Stage 1 Predicates Involving Columns in Predicates

New Stage 1 / Indexable predicates

WHERE value BETWEEN COL1 AND COL2

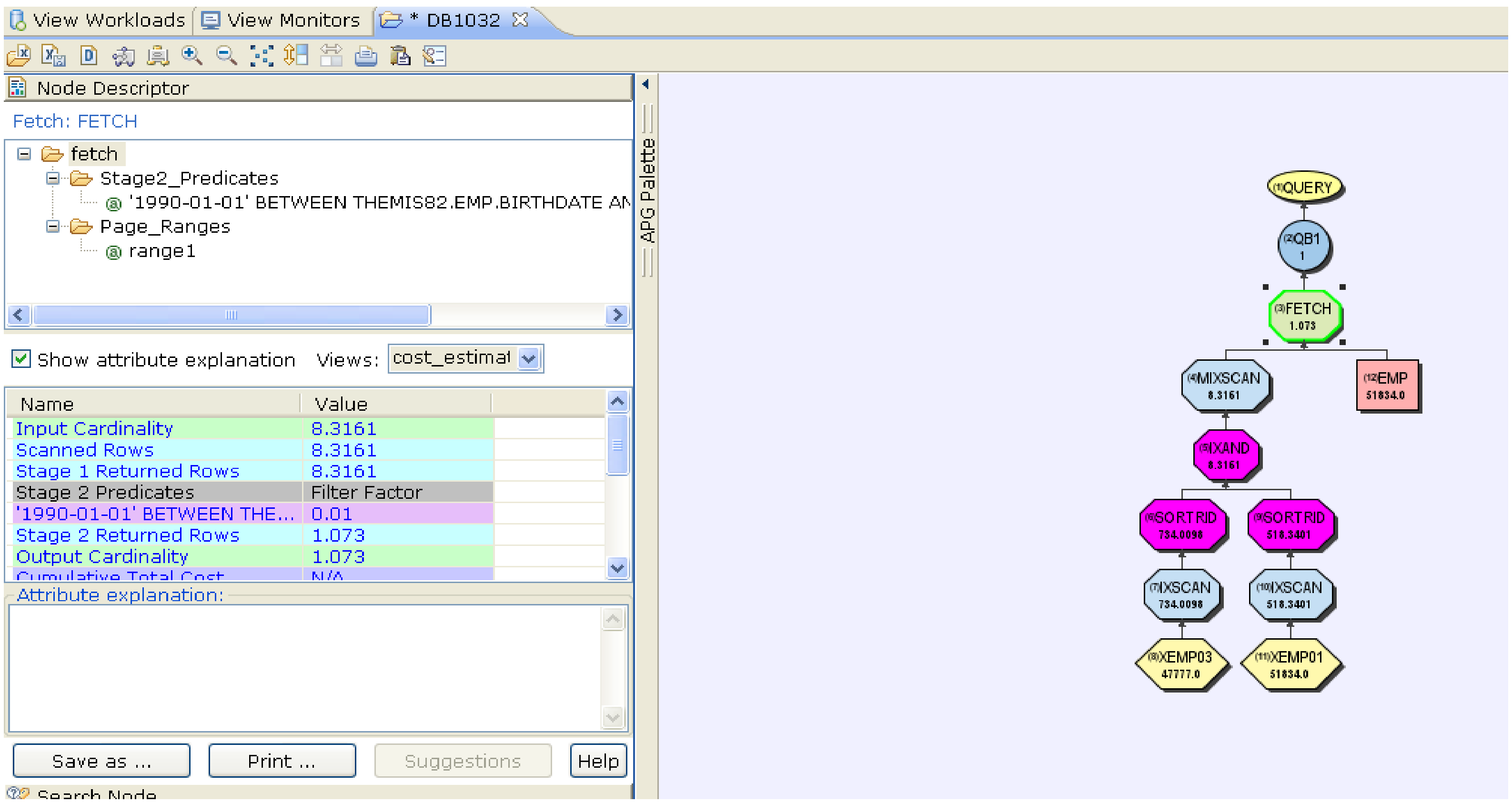
WHERE SUBSTR(COLX, 1, n) = value ➔ From Pos 1 only

WHERE DATE(TS_COL) = value

WHERE YEAR(DT_COL) = value

Bad Coding Practice
Stage 2 Predicates

Use the Visual Explain in IBM Data Studio or query directly the DSN_PREDICAT_TABLE to see any stage 2 predicates. Note the filter factor information also. **WHERE '1900-01-01' BETWEEN DATE_COL1 AND DATE_COL2**



Tuning Approaches

- Explain the Query
- Change the SQL. Rewrite the query or predicates a different way
- Redesign the program flow
- Update / Improve data statistics
- Change Physical Design

Themis
Leaders in IT Education

What Causes a Table Scan?

- The predicate(s) may be poorly coded in a non-indexable way.
- The predicates in the query do not match any available indexes on table.
- The table could be small, and DB2 decides a tablespace scan may be faster than index processing.
- The catalog statistics say the table is small, or maybe there are no statistics on the table.
- The predicates are such that DB2 thinks the query is going to retrieve a large enough amount of rows that would require a tablespace scan. Check the Filter Factor!
- The predicates are such that DB2 picks a non-clustered index, and the number of pages to retrieve is high enough based on total number of pages in the table to require a tablespace scan.
- The tablespace file or index files could physically be out of shape and need a REORG.

Tuning Approach: Change the SQL and/or Change the program Design

- Can any predicates be rewritten (and still keep same logic)
- Can the query be rewritten
- Can we combine any queries in the program

Sometimes there can 2,3,4,5,6 different ways to code an SQL statement and return the same results. They do not all optimize the same!

Change the SQL Example 1

Each of these will produce the same results, but operate very differently. Typically one will perform better than the other depending on data distributions. For Example:

Non Correlated Subquery

```
SELECT E.EMPNO, E.LASTNAME
FROM EMP   E
WHERE E.EMPNO IN
        (SELECT D.MGRNO
         FROM DEPT D
        WHERE D.DEPTNO LIKE ‘D%”)
```

Can also be coded as:

```
SELECT E.EMPNO, E.LASTNAME
 FROM EMP E
 WHERE EXISTS
        (SELECT 1
          FROM DEPT D
          WHERE D.MGRNO = E.EMPNO
             AND D.DEPTNO LIKE ‘D%’)
```

Or a 2 table join, but watch out for possible duplicates (if 1 to many relationship)

```
SELECT DISTINCT E.EMPNO, E.LASTNAME
 FROM EMP   E, DEPT D
 WHERE E.EMPNO = D.MGRNO
     AND  D.DEPTNO LIKE ‘D%’
```

Themis
Leaders in IT Education

Change the SQL Example 2

PROBLEM: Find all employees who major in math (MAT) and (CSI).

EMPMAJOR

EMPNO	MAJOR
E1	MAT
E1	CSI
E2	MAT
E3	CSI
E4	ENG

Group By / Having Logic:

```
SELECT EMPNO
  FROM   EMPMAJOR
  WHERE  MAJOR IN ('MAT', 'CSI')
  GROUP BY EMPNO
  HAVING COUNT(*) = 2;
```

Self Join Logic:

```
SELECT EMPNO
FROM EMPMAJOR AS EMP1 JOIN
           EMPMAJOR AS EMP 2
       ON EMP1.EMPNO = EMP2.EMPNO
   WHERE EMP1.MAJOR = 'MAT'
     AND EMP2.MAJOR = 'CSI';
```

Quota Query Logic

```
SELECT DISTINCT EM1.EMPNO
 FROM   EMPMAJOR AS EM1
 WHERE 2 =
     (SELECT  COUNT(*)
      FROM     EMPMAJOR EM2
      WHERE  EM2.EMPNO = EM1.EMPNO
         AND  EM2.MAJOR IN  ('MAT', 'CSI') ;
```

Change the SQL Example 3

PROBLEM: Find the youngest employee out of the EMP table in each department).

Hint: Youngest employee are the ones with highest (max) birthdate.

Correlated Subquery :

```
SELECT E1.EMPNO, E1.LASTNAME
FROM EMP AS  E1
WHERE E1.BIRTHDATE = (SELECT MAX(E2.BIRTHDATE)
                      FROM EMP E2
                     WHERE E2.DEPTNO = E1.DEPTNO)
```

Row Value Expression:

```
SELECT E1.EMPNO, E1.LASTNAME
FROM EMP E1
 WHERE (E1.DEPTNO,E1.BIRTHDATE) IN
            (SELECT E2.DEPTNO,MAX(E2.BIRTHDATE)
             FROM EMP E2
             GROUP BY E2.DEPTNO)
```

Common Table Expression

```
WITH X AS
  (SELECT  DEPTNO, MAX(BIRTHDATE) AS MAX_BIRTHDATE
   FROM EMP GROUP BY DEPTNO)
SELECT E.EMPNO, E.LASTNAME, E.BIRTHDATE
FROM EMP E,  X
WHERE E.BIRTHDATE = X.MAX_BIRTHDATE
     AND E.DEPTNO = X.DEPTNO
ORDER BY E.EMPNO
;
```

Tuning Approach: Redesign the Program Flow

- Know your numbers. How many inserts, updates, deletes, selects, open cursors, and fetches per execution? Can they be cut down?
- Code relationally and not procedurally
- Know the many different ways to code for mass inserts, mass deletes, and mass updates.
- Minimize the number of times your code sends SQL statements to DB2.
- Take advantage of multi row processing, merge, select from insert/update/delete, multi table joins, etc.
- Order incoming data by either primary key, or column(s) of the index selected from DB2.

Tuning Approach: Explain the Query

- Any Table Scans? What's causing it?
- Any Index Scans? What's causing it?
- Any Partition Scans? What's causing it?
- Which Index? Matching columns? Screening?
- Any Sorts? What's causing it? How big is the sort?
- Any Join sorts? What other queries join to that table?
- Any subqueries? Can they be rewritten?
- Any materialization from NTE and CTE's? Can they be rewritten? *(Not saying these are always bad…)*
- Check the predicates? Stage 2 or Residual? Filter factor?

Update / Improve Data Statistics

- Are statistics up to date (or close enough)?
- Do all columns have cardinality statistics?
- Are there any columns used in predicates with skewed distribution of data?
  - Are there statistics to support the data skew?
  - Frequency value vs Histogram
  - Is your code taking advantage of the statistics by either hard coding or re-optimizing at runtime?
- Has data changed in the table (10% or more increase or decrease) since last compile? **KNOW YOUR DATA!!!!**

Themis
Leaders in IT Education

Update / Improve Data Statistics

Statistics in Test vs Production.

- **Just copying statistics is not good enough. Need enough data to see run time differences**

- **Have to test the different code and compare CPU times.**

- **DB2 not always correct in its guestimations**

Program Hard Coding for Performance. Know your data!

STATUS_CODE current values 'A' 90% of data
'I' 6% of data
'T' 4% of data

1)
```
Select ….. From Table
Where ……..
  and ……..
  and Status_Code = 'A'
;
```

2)
```
Select ….. From Table
Where ……..
  and ……..
  and Status_Code = :HV
  and Status_Code <> 'A'
;
```

Themis
Leaders in IT Education

Physical Design

Make sure of the clustering order of data in your tablespaces.

Tables should be physically clustered in the order that they are typically processed by queries processing the most data. This ensures the least amount of 'Getpages' when processing.

Long running queries with 'List Prefetch' and 'Sorts' in many join processes are good indicators that maybe a table is not in the correct physical order.

Application queries that join to a table via the foreign key vs the primary key is a good indicator.

Too many 'Getpages' vs rows returned

Themis
Leaders in IT Education

Change the Physical Design ?
EMP table clustered by EMPNO

000010 HAAS …… A00 **000020 THOMPSON …… B01** **000030 KWAN ………. C01** **000050 GEYER …… E01** **000060 STERN …… D11** **000070 PULASKI …… D21** **000090 HENDERSON ……. E11**	**000100 SPENSER …… E21** **000110 LUCHESI …… A00** **000120 O'CONNELL ..…. A00** **000130 QUINTANA …… C01** **000140 NICHOLLS …… C01** **000150 ADAMSON …… D11** **000160 PIANKA …….. D11**	

Should this table be in EMPNO Primary Key order?

It Depends…..

Themis
Leaders in IT Education

Change the Physical Design ?
EMP table clustered by EMPNO

000010 HAAS …… A00 **000020 THOMPSON …… B01** **000030 KWAN ………. C01** **000050 GEYER …… E01** **000060 STERN …… D11** **000070 PULASKI …… D21** **000090 HENDERSON ……. E11**	**000100 SPENSER …… E21** **000110 LUCHESI …… A00** **000120 O'CONNELL ..…. A00** **000130 QUINTANA …… C01** **000140 NICHOLLS …… C01** **000150 ADAMSON …… D11** **000160 PIANKA …….. D11**	

What happens here?

SELECT *
FROM EMP
WHERE DEPTNO = 'A00'

Where are all the rows that have 'A00' as a DEPTNO value?

IF there were 100 rows that contain this value, they could be on 100 pages of data. Yes?

Thank you for allowing me to share some of my experience and knowledge today!

Tony Andrews

tandrews@themisinc.com

- I hope that you learned something new today
- I hope that you are a little more inspired when it comes to SQL coding and performance tuning

Themis
Leaders in IT Education

The material in this presentation is further developed in the following Themis courses:

DB1032 – DB2 for z/OS Performance and Tuning
DB1041 – DB2 z/OS Advanced SQL
DB1037 – Advanced Query Tuning using IBM Data Studio
DB1051 – High Performance Application Design
DB1006 – DB2 LUW Advanced Query Tuning using IBM Data Studio

Links to these courses may be found at: www.themisinc.com

Tony's Email: tandrews@themisinc.com
Twitter: @ThemisTraining

Themis
Leaders in IT Education

"I have noticed that when the developers get educated, good SQL programming standards are in place, and program walkthroughs are executed correctly, incident reporting stays low, CPU costs do not get out of control, and most performance issues are found before promoting code to production."

Themis
Leaders in IT Education

Education. Check out www.ibmpress.com www.amazon.com

Finally! A book of DB2 SQL tuning tips for developers, specifically designed to improve performance.

DB2 SQL developers now have a handy reference guide with tuning tips to improve performance in queries, programs and applications.

As of DB2 V10.