

The Science and Art of Indexing on DB2 for IBM i

Tom McKinley (mac2@us.ibm.com)
DB2 for IBM i Center of Excellence
Rochester Lab Services

Power is performance redefined

Deliver IT services faster,
with higher quality,
and with superior economics

www.ibm.com/power



Gateway/400 01/2012

© 2012 IBM Corporation

Scenario

Find the first occurrence of **“IBM”** in a very large book...

What do you do first?



Turn to the index!

in·dex Something that serves to guide, point out,
or otherwise facilitate efficient reference.

Creating a useful index
is both a *Science* and an *Art*.



Agenda

- DB2 for i Indexing Technology
- Query Optimization using Indexes
- Indexing Strategies
- Case Study
- Appendix
 - OmniFind Text Search Server for DB2 on IBM i

Indexing Technology within DB2 for IBM i

DB2 for i

- Two types of indexing technologies are supported
 - **Radix Index**
 - **Encoded Vector Index**

(OmniFind Text Search Server. See Appendix)
- Each type of index has specific uses and advantages
- Respective indexing technologies compliment each other
- Indexes can be used for statistics and implementation
- Indexes can provide RRNs and/or data
- Indexes are scanned or probed
 - Probe can only occur on contiguous, leading key columns
 - Scan can occur on any key column
 - Probe and scan can be used together

Using Indexes - Probe v Scan

- **Probe** (key positioning) with leading, n contiguous key columns

1

1+2

1+2+3

- **Scan** (test) with any other key columns

2

3

2+3

Index Key Columns (ITEM_NO, COLOR, SIZE)

ITEM_NO	COLOR	SIZE
001	BLUE	LARGE
002	RED	SMALL
003	BLACK	SMALL
004	GREEN	MEDIUM

...WHERE COLOR = 'BLACK' AND ITEM_NO = 003

...WHERE SIZE = 'MEDIUM'

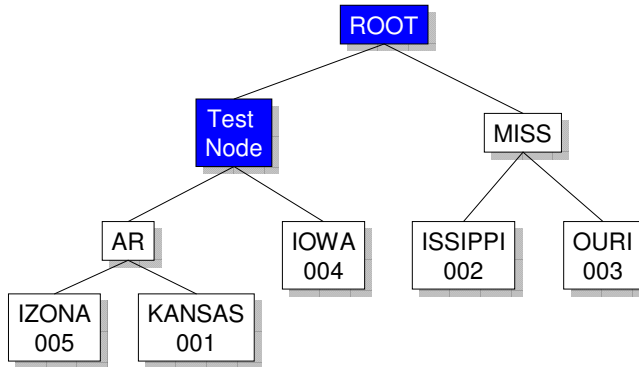
...WHERE ITEM_NO = 001 AND SIZE = 'LARGE'

Radix Index

- Index "tree" structure
- Key values are compressed
 - Common patterns are stored once
 - Unique portion stored in "leaf" pages
 - Positive impact on size and depth of the index tree
- Algorithm used to find values
 - Binary search
 - Modified to fit the data structure
- Maintenance
 - Index data is automatically spread across all available disk units
 - Tree is automatically rebalanced to maintain an efficient structure
- Temporary indexes
 - Considered a temporary data structure to assist the DB engine
 - Maintained temporary indexes available in SQE V5R4 and later

Radix Index

Database Table	
001	ARKANSAS
002	MISSISSIPPI
003	MISSOURI
004	IOWA
005	ARIZONA
...	...



ADVANTAGES:

- Very fast access to a single key value
- Also fast for small, selected range of key values (low cardinality)
- Provides order

DISADVANTAGES:

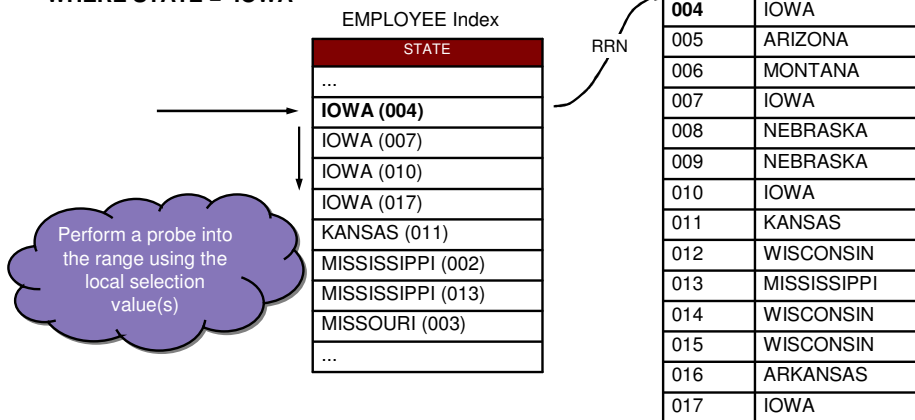
- Table rows retrieved in order of key values (not physical order) which equates to random I/O's
- No way to predict which physical index pages are next when traversing the index for large number of key values

Index Probe Example

Given an index on table EMPLOYEE keyed on STATE...

```

SELECT *
FROM EMPLOYEE
WHERE STATE = 'IOWA'
    
```



Encoded Vector Index (EVI)

- Index for delivering fast data access in analytical and reporting environments
 - Advanced technology from IBM Research
 - Used to produce dynamic bitmaps and RRN lists
 - Fast access to statistics to improve query optimizer decision making
- Not a “tree” structure
- Can only be created through an SQL interface or Navigator for i GUI

```
CREATE ENCODED VECTOR INDEX MySchema.IXName
ON MySchema.TabName(KEY(s))
INCLUDE ( SUM(SomeOtherColName));
```

New in 7.1
Maintained
aggregate

Encoded Vector Index (EVI)

Symbol Table				
Key Value	Code	Count	Include Sum()	Include Sum()
Arizona	1	5000	1500	2005
Arkansas	2	7300	3200	450
...				
Wisconsin	49	340	575	1200
Wyoming	50	2760	210	0

optional

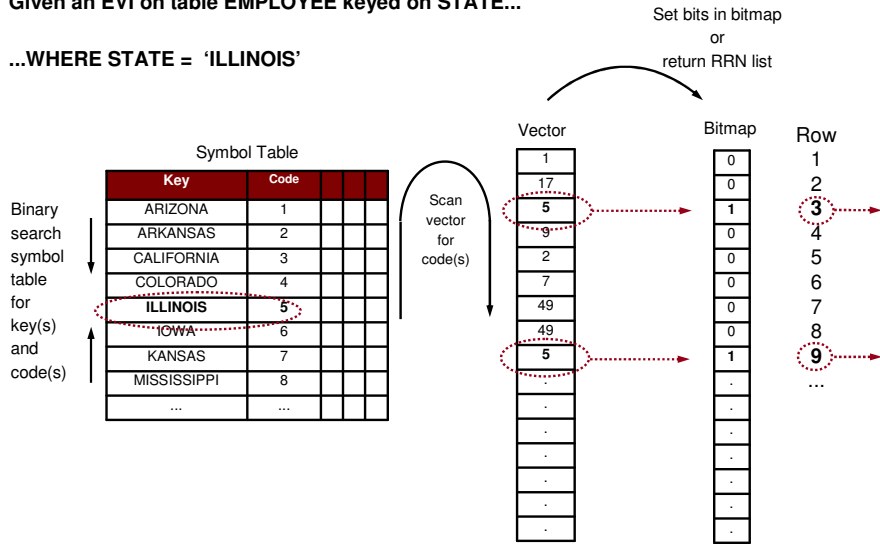
Vector	RRN
1	1
17	2
5	3
9	4
2	5
7	6
50	7
49	8
5	9
...	...

- Symbol table contains information for each distinct key value
 - Each key value is assigned a unique code (key compression)
 - Code is 1, 2, or 4 bytes depending on number of distinct key values
 - **Enhanced in i 7.1 to include SUM and COUNT in the definition**
- Rather than a bit array for each distinct key value, use one array of codes

Bitmap / RRR List Example

Given an EVI on table EMPLOYEE keyed on STATE...

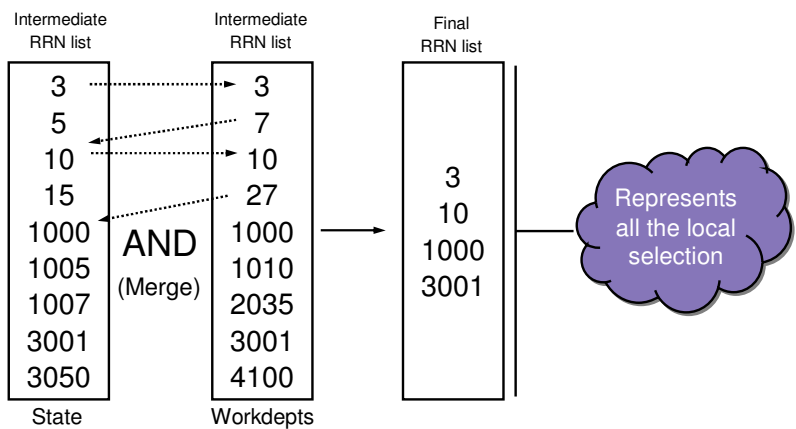
...WHERE STATE = 'ILLINOIS'



Index ANDing Example

EVI State **EVI** Workdept

SELECT *
FROM EMPLOYEE
WHERE STATE = 'MINNESOTA'
AND WORKDEPT IN ('B01', 'C01', 'E01')



Another Symbol Table only Case

- Grouping column(s) in a different table
- Create EVI on Join Column
- Resulting query joins to EVI Symbol table in place of table with sum column

Create ENCODED VECTOR INDEX idx3 ON sales (partid)
 include (SUM(sales), count(*)) ← NEW IN 7.1

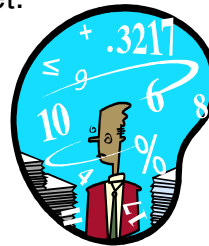
```

Select  p.PartSupplier,
        Sum(s.sales) as totalPartSupplierRev,
        Count(*)
From    Sales s ← Use idx3
        Join Part P on s.partid=p.partid  Symbol table in
Group BY P.PartSupplier                 place of Sales
Order By 2 DESC                          table
    
```

DB2 for IBM i

cardinality The number of elements in a set.

- High cardinality = large distinct number of values
- Low cardinality = small distinct number of values



In general...

- A [radix index](#) is best when accessing a small set of rows and the key cardinality is high
- An [encoded vector index](#) is best when accessing a set of rows and the key cardinality is low
- Understanding the data and query are key

Creating Indexes

- CREATE INDEX SQL statement
CREATE INDEX MY_IX on MY_TABLE (KEY1, KEY2)
- CREATE ENCODED VECTOR INDEX SQL statement
CREATE ENCODED VECTOR INDEX MY_EVI on MY_TABLE (KEY1)
- System i Navigator – Database graphical interface
- CRTPF and CRTLF CL commands
 - Keyed access path within the physical file or logical file
 - Join logical file
- Primary Key, Foreign Key and Unique Key Constraints
 - CREATE TABLE
 - ALTER TABLE
 - ADDPFCST

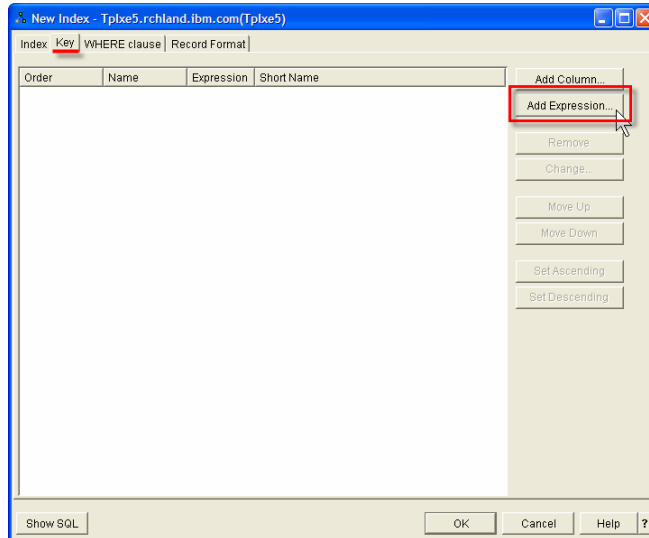
Creating Indexes – 6.1

The screenshot shows a 'New Index' dialog box with the following configuration:

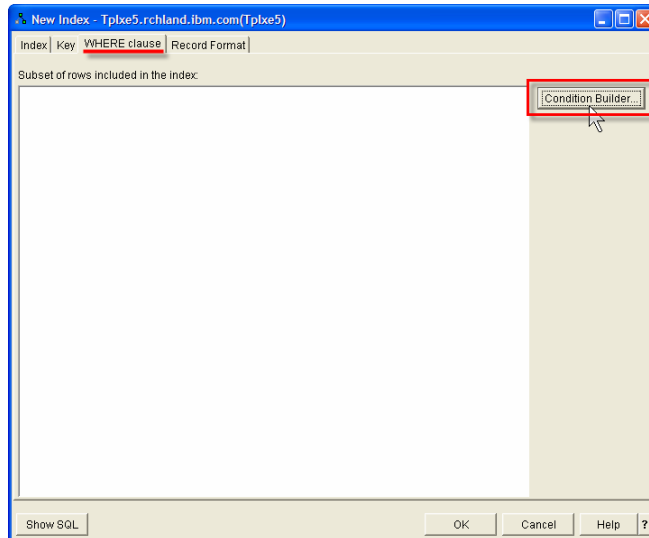
- Index name: (empty)
- Index schema: TPCDS1GB
- System index name: System-generated
- Table schema: TPCDS1GB
- Table name: CATALOG_SALES
- Index type: Not unique
- Number of distinct values: Not specified
- Number of tasks: Use current setting
- Page size: Default
- Sort sequence table: Sort sequence of the job
- Language Identifier: (empty)
- Text: (empty)

Buttons at the bottom: Show SQL, OK, Cancel, Help ?

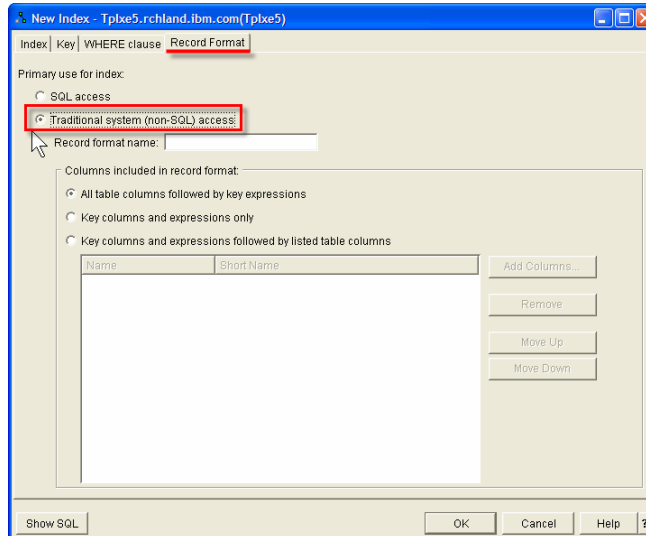
Creating Indexes – 6.1



Creating Indexes – 6.1



Creating Indexes – 6.1



6.1 Creation of Index with Derived Keys

- Creation of indexes with *derived* keys via SQL

```
CREATE INDEX ORDERPRIORITYUPPER ON T1
  (UPPER(ORDERPRIORITY) AS UORDERPRIORITY ASC);
```

```
CREATE ENCODED VECTOR INDEX YEARQTR ON T1
  (YEAR(ORDERDATE) AS ORDYEAR ASC, QUARTER(ORDERDATE)
  AS ORDQTR ASC);
```

```
CREATE INDEX TOTALEXTENDEDPRICE ON T1
  (QUANTITY * EXTENDEDPRICE AS TOTEXTPRICE ASC);
```

NOTE: There are some restrictions on which indexes can be used by the optimizer in 6.1

V6R1 Enhanced Native access to the SQL Index

- Specify format name And table columns to be added to that format.

```
CREATE INDEX STAR100G.SHIPMODE ON STAR100G.ITEM_FACT  
(SHIPMODE ASC, ZIPCODE ASC)
```

```
RCDFMT SHPMODEZIP
```

```
ADD CUSTNAME, SHIPADDR;
```

```
CREATE INDEX STAR100G.SHIPMDMON ON STAR100G.ITEM_FACT  
(SHIPMODE ASC, MONTH(SHIPDATE) AS SHIPMON for COLUMN  
SHIPMON ASC)
```

```
RCDFMT SHPMODEMON
```

```
ADD CUSTNAME, SHIPADDR, COMMITDATE;
```

6.1 Create Sparse Indexes from SQL

- Support of WHERE clause on SQL create index

```
CREATE INDEX FASTDELIVER ON T1  
(SHIPMODE ASC)
```

```
WHERE SHIPMODE = 'NEXTDAYAIR'  
OR SHIPMODE = 'COURIER';
```

NOTE: Index is *NOT* used by the query optimizer (CQE and SQE) prior to 7.1

When to a use Derived Index?

- Could replace some logical files with SQL indexes for use by RLA native, high level language programs
 - Modernize those objects
 - Big logical page size (8K v 64K)

- Derived indexes may be useful for
 - Case insensitive searches
 - Data extracted from a column (i.e. SUBSTR, YEAR, MONTH...)
 - Derive Common Grouping columns (i.e. YEAR(ORDERDATE))
 - Results of operations (COL1+COL2 , QTY * COST)
 - Might be useful to allow *index only access* in more cases
 - Especially with INCLUDE SUPPORT FOR 7.1
 - **Reduce table scans, index scans and temporary data structures**

EVI's and Grouping

- EVI with A, B, C key fields and INCLUDE(SUM(D)...)
Create encoded vector index GBEVI02 on T1 (A , B , C) **INCLUDE(SUM(D))**

Will be usable for group by ALL Grouping combinations of A,B,C (including Grouping set combinations)

Example:

```
SELECT A,B,C, SUM(D) FROM T1 GROUP BY GROUPING SETS((A), (B), (C))
SELECT A,B,C, SUM(D) FROM T1 GROUP BY GROUPING SETS((A), (B))
SELECT A,B,C, SUM(D) FROM T1 GROUP BY GROUPING SETS((A), (C))
SELECT A,B,C, SUM(D) FROM T1 GROUP BY GROUPING SETS((B), (C))
SELECT A,B,C, SUM(D) FROM T1 GROUP BY GROUPING SETS((A,B), (C))
SELECT A,B,C, SUM(D) FROM T1 GROUP BY GROUPING SETS((A,C), (B))
SELECT A,B,C, SUM(D) FROM T1 GROUP BY ROLLUP(A,B,C)
SELECT A,B,C, SUM(D) FROM T1 GROUP BY CUBE(A,B,C)
```

```
SELECT A,B,C, SUM(D) FROM T1 GROUP BY (A,B,C)
SELECT A,B, SUM(D) FROM T1 GROUP BY (A,B)
SELECT C SUM(D) FROM T1 GROUP BY (C) /* Or A, Or B */
SELECT SUM(D) FROM T1
```

- Experiment with the NEW EVI INCLUDE support on DB2 for IBM i for Grouping and Grouping SET Queries
 - Create EVI with common GB columns and INCLUDE most commonly used sums
 - Always add in COUNT(*) to EVI INCLUDE
 - **Do this only for GB columns that have relatively small cardinality**
 - **EVIs work best if NOT constantly adding new Key values**

Other Index related enhancements

- **In-Memory Table/Index (7.1)**

```
CHGPF FILE(MYSCHEMA/TAB1) KEEPINMEM(*YES)  
CHGLF FILE(MYSCHEMA/IX1) KEEPINMEM(*YES)
```

- **SSD Support for Table/Index (6.1)**

```
CHGLF FILE(MYSCHEMA/IX1) UNIT(*SSD)
```

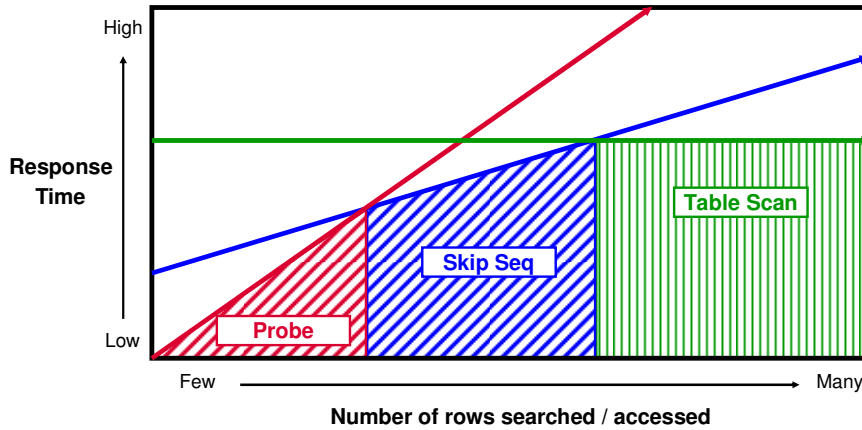
- **Expanded Optimizer use of Sparse and derived Indexes (7.1)**

```
CREATE INDEX cust_act ON CUSTIMERS(cust_id) WHERE  
activCust='Y'
```

Query Optimization
(using indexes)

Data Access Methods

Cost based optimization dictates that the fastest access method for a given table will vary based upon *selectivity* of the query



Strategy for Query Optimization

Query optimization will generally follow this simplified strategy:

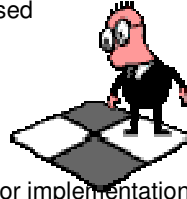
- ✓ Gather meta-data and statistics for costing
 - Selectivity statistics
 - Indexes available to be costing
 - Sort the indexes based upon their usefulness
 - Environmental attributes that may affect the costs
- ✓ Generate default cost
 - Build an access plan associated with the default plan
- ✓ For each index:
 - Gather information needed specific to this index
 - Build an access plan based on this index
 - Cost the use of the index with this access plan
 - Compare the resulting cost against the cost from the current best plan



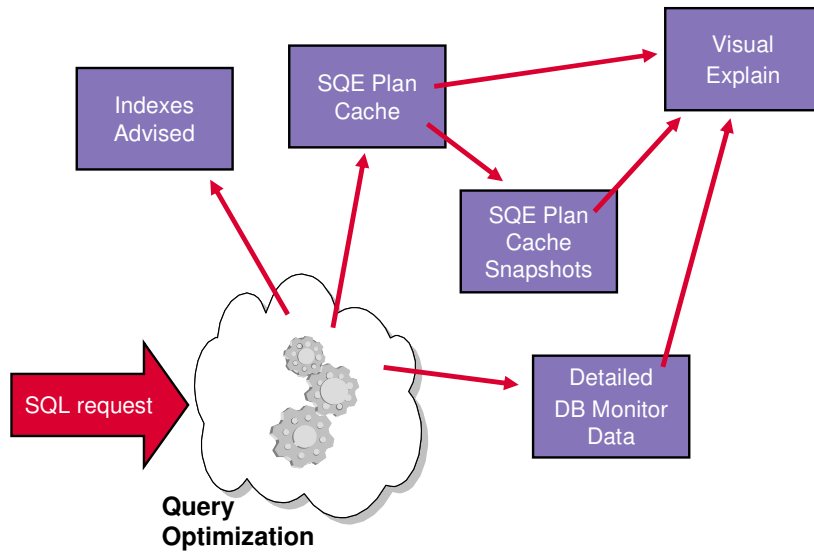
Strategy for Query Optimization

Optimizing indexes will generally follow this simplified strategy:

- Gather list of indexes for statistics and costing
- Sort the list of indexes considering how the index can be used
 - Local selection
 - Joining
 - Grouping
 - Ordering
 - Index only access
- One index may be useful for statistics, and another useful for implementation



Query Optimization Feedback



Indexing Advice from the Optimizer

- Both CQE and SQE provide index creation advice
 - QSYS2/SYSIXADV
 - System i Navigator
- CQE
 - Basic advice
 - Radix index only
 - Based on table scan and local selection columns only
 - Temporary index creation information also provides insight
 - CQE Visual Explain will try and tie pieces together to advise a better index
- SQE
 - Robust advice
 - Radix and EVI indexes
 - Based on all parts of the query
 - Multiple indexes can be advised for the same query
 - Some limitations

Indexing Advice from the Optimizer...

Local selection, Join, Group By, Order By
Equals first, followed by one inequality (for probing)

No ORed predicates involving different columns

No derived keys

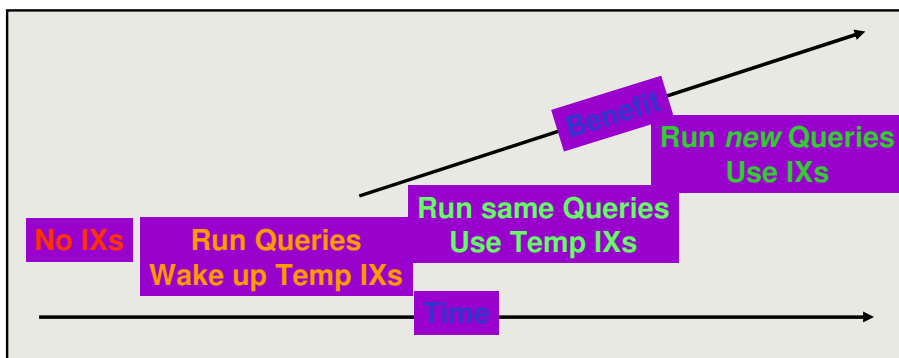
No EVI include columns

Autonomic Index Creation

- Optimizer can have the DB Engine create a temporary index
- Both full and sparse indexes can be created
- Temporary indexes are not used for statistics
- Temporary indexes are *maintained*
- CQE
 - Temporary indexes are not reused and not shared
 - Usually a bottleneck in query performance
 - Can impact overall system performance
 - Can increase the amount of temporary storage used
- SQE
 - Temporary indexes are reused and shared across jobs and queries
 - Creation is based on “watching” the query requests over time
 - Creation is based on optimizer’s own index advice
 - Temporary index maintenance is delayed when all associated cursors closed

Autonomic Index Creation

- CQE temporary indexes represent a good opportunity for tuning
 - Temp indexes are not shared or reused
- SQE temporary indexes represent DB2 self tuning
 - For the same set of queries, temp indexes are about the same as hash tables
 - Temp indexes are shared and reused, providing more benefit



Index Evaluator (Show Indexes)

Environment: My Connections | Tptxe1: Tables Database: Tptxe1 Schema: M

SQL Name | Partitioned | Owner

- CRT_MQT_2 | No | MCAIN
- CUSTOMERS | No | MCAIN
- DBMONITORX | No | MCAIN
- ITE | Edit Contents | IN
- PF1 | View Contents | IN
- PF1 | Definition | IN
- PF1 | Data | IN
- PF1 | Generate SQL... | IN
- PF1 | Index Advisor | IN
- PRC | Journaling | IN
- QA1 | Locked Rows | IN
- QA1 | Permissions | IN
- QA1 | Show Indexes | IN
- QCL | Show Materialized Query Tables | IN
- QCL | Show Related | IN

Indexes for MCAIN.DBMONITORX - Tptxe1

SQL Name	S	C	S	T	N	V	C	I	Last Query Use	Last Query Statistics Use	Query Use Count	Query Statistics Use Count	Last Used Date	Days Used Count
DBMONITORX_IDX1	I	M	M	D					2/3/06 12:58:15 PM	2/3/06 1:12:13 PM	85	157	2/3/06 10:18:15 AM	5
DBMONITORX_IDX2	I	M	M	D					2/2/06 4:26:26 PM	2/2/06 4:26:26 PM	100	118	2/2/06 3:12:56 PM	4
DBMONITORX_QBN1	I	M	M	D					2/3/06 1:12:13 PM	2/3/06 1:12:13 PM	28	653	2/3/06 10:18:15 AM	5
DBMONITORX_QBN2	I	M	M	D					D Y 1 1	2/3/06 1:12:13 PM	0	69		0
DBMONITORX_QBN3	I	M	M	D					D Y 1 1	2/3/06 4:25:55 PM	0	640		0

40 Power is performance redefined Gateway/400 01/2012 © 2012 IBM Corporation

Index Evaluator via Catalog Views

SYSINDEXSTAT	Contains one row for every SQL index. Use this view when you want to see information for a specific SQL index or set of SQL indexes. The information is similar to that returned via Show Indexes in System i Navigator.
SYSPARTITIONINDEXES	Contains one row for every index built over a table partition or table member. Use this view when you want to see index information for indexes built on a specified table or set of tables. The information is similar to that returned via Show Indexes in System i Navigator.
SYSTABLEINDEXSTAT	Contains one row for every index that has at least one partition or member built over a table. If the index is over more than one partition or member, the statistics include all those partitions and members.

Indexing Strategies

What should you do?

Create all advised indexes?

Nothing, let the system handle it?

Monitor, analyze, and tune important tables and queries?

DB2 for IBM i

The goals of creating indexes are:

1. Provide the optimizer the statistics needed to understand the data, based on the query
2. Provide the optimizer implementation choices, based on the selectivity of the query

- ✓ Accurate statistics means accurate costing
- ✓ Accurate costing means optimal query plan
- ✓ Optimal query plans means best performance

The Process of Identifying Indexes

Proactive method

- Analyze the data model, application and SQL requests

Reactive method

- Rely on optimizer feedback and actual implementation methods
- Rely on SQE's ability to auto tune using temporary indexes

Understand the data being queried

- Column selectivity
- Column cardinality

Separating complex queries into individual parts by table

- Selecting
- Joining
- Grouping
- Ordering
- Subquery
- View

Indexing Strategy - Basic Approach

Radix Indexes

- Common local selection columns
- Join columns
- Local selection columns + join columns
- Local selection columns + grouping columns
- Local selection columns + ordering columns

Minimum

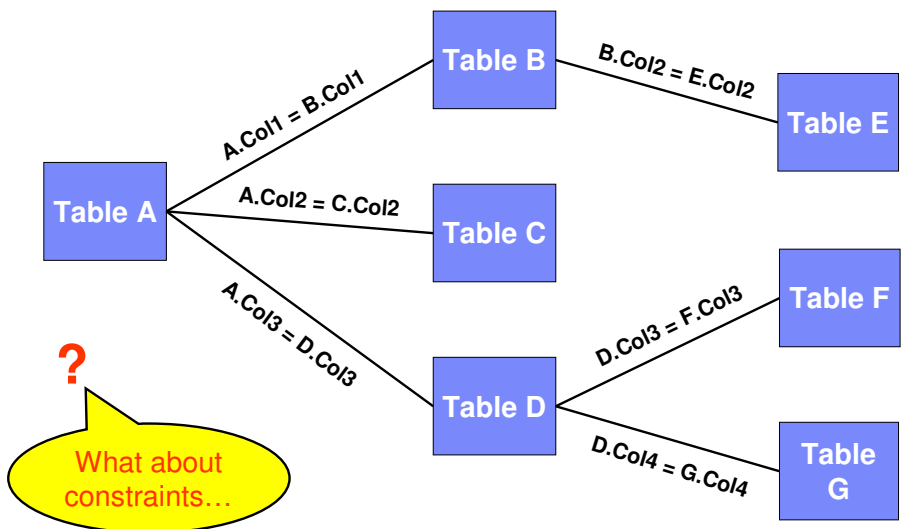
Encoded Vector Indexes

- Local selection column for index ANDing/ORing
- Join columns (star or snowflake schema)
- Index only access
 - DISTINCT, COUNT, COUNT DISTINCT, SUM()

Advanced
Requires knowledge of query optimization and data lifecycle

Note: Columns used with equal conditions are first in key list

Indexing Strategy - Examples



Indexing Strategy - Examples

```
-- Query 1
SELECT      A.CUSTOMER_NO, A.ORDER_DATE, A.QUANTITY
FROM        ORDERS A
WHERE       A.CUSTOMER_NO = 0112358;

CREATE INDEX ORDERS_IX1 ON ORDERS (CUSTOMER_NO);

-- Query 2
SELECT      A.CUSTOMER_NO, A.ORDER_DATE, A.QUANTITY
FROM        ORDERS A
WHERE       A.CUSTOMER_NO = 0112358
AND         A.ITEM_ID = 'ABC123YXZ';

CREATE INDEX ORDERS_IX2 ON ORDERS (CUSTOMER_NO, ITEM_ID);
```

Indexing Strategy - Examples

```
-- Query 3
SELECT      A.CUSTOMER_NO, A.CUSTOMER, A.ORDER_DATE
FROM        ORDERS A
WHERE       A.CUSTOMER_NO IN (0112358, 1321345, 5891442)
AND         A.ORDER_DATE > '2005/06/30'
ORDER BY    A.ORDER_DATE;

CREATE INDEX ORDERS_IX3a ON ORDERS (CUSTOMER_NO, ORDER_DATE);
CREATE INDEX ORDERS_IX3b ON ORDERS (ORDER_DATE, CUSTOMER_NO);

-- Query 4
SELECT      A.CUSTOMER_NO, A.CUSTOMER, A.ORDER_DATE
FROM        ORDERS A
WHERE       A.CUSTOMER_NO = 0112358
OR          A.ORDER_DATE = '2005/06/30';

CREATE INDEX ORDERS_IX4 ON ORDERS (CUSTOMER_NO);
CREATE ENCODED VECTOR INDEX ORDERS_EVI4
ON ORDERS (ORDER_DATE);
```

Indexing Strategy - Examples

```
-- Query 5
SELECT      A.CUSTOMER_NO, B.CUSTOMER, A.ORDER_DATE, A.QUANTITY
FROM        ORDERS A,
            CUSTOMERS B,
            ITEMS C
WHERE       A.CUSTKEY = B.CUSTKEY
AND         A.ITEMKEY = C.ITEMKEY
AND         A.CUSTOMER_NO = 0112358;

CREATE INDEX ORDERS_IX5a ON ORDERS (CUSTOMER_NO, CUSTKEY);
CREATE INDEX ORDERS_IX5b ON ORDERS (CUSTOMER_NO, ITEMKEY);
CREATE INDEX CUSTOMERS_IX5 ON CUSTOMERS (CUSTKEY);
CREATE INDEX ITEMS_IX5 ON ITEMS (ITEMKEY);
```

Indexing Strategy - Examples

```
-- Query 6
SELECT      YEAR(A.ORDER_DATE),SUM(A.QUANTITY), COUNT(*)
FROM        ORDERS A
GROUP BY    YEAR(A.ORDER_DATE);

CREATE ENCODED VECTOR INDEX ORDERS_IX6A
ON ORDERS (YEAR(ORDER_DATE))
INCLUDE (SUM(QUANTITY), COUNT(*));
```

Indexing Strategy - Examples

-- Query 7

```
SELECT      YEAR(A.ORDER_DATE),QUARTER(A.ORDER_DATE),
            MONTH(ORDER_DATE), SUM(A.QUANTITY), COUNT(*)
FROM        ORDERS A
WHERE       QUARTER(A.ORDER_DATE) = 4
GROUP BY   YEAR(A.ORDER_DATE), QUARTER(A.ORDER_DATE),
            MONTH(ORDER_DATE)
ORDER BY   YEAR(A.ORDER_DATE),QUARTER(A.ORDER_DATE),
            MONTH(ORDER_DATE),
```

```
CREATE ENCODED VECTOR INDEX ORDERS_IX6A
      ON ORDERS (YEAR(ORDER_DATE), QUARTER(A.ORDER_DATE),
                MONTH(ORDER_DATE) )
      INCLUDE (SUM(QUANTITY), COUNT(*));
```

Indexing Strategy - Examples

If the optimizer feedback indicates:

Full table scan → Create an index on local selection columns

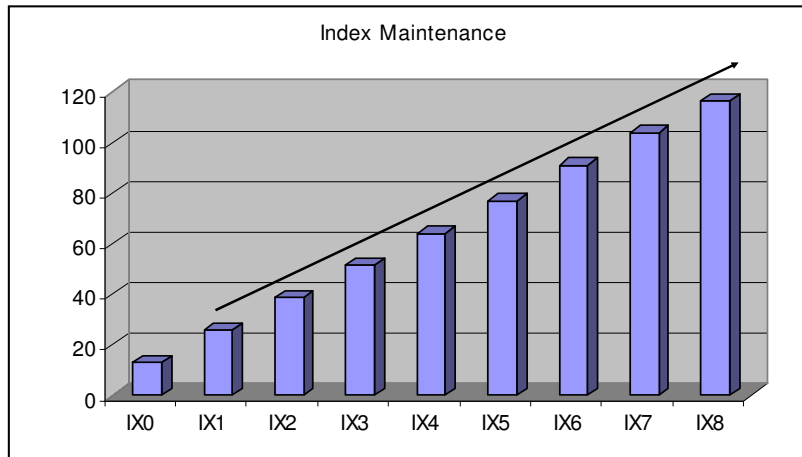
Full index scan → Create an index that is allows probe

Temporary index → Create an index on join columns
 → Create an index on grouping columns
 → Create an index on ordering columns

Hash table → Create an index on join columns
 → Create an index on grouping columns

“Perfect”, multiple key column radix indexes are usually best

Indexing Strategy – Maintenance



In general - index maintenance costs grows linearly

Indexing Strategy – Maintenance v Query Access

- For best query performance, create the appropriate indexes
- Eliminating table scans and temporary data structures will more than make up for index maintenance overhead
- Consider the number of indexes when doing *high* volume batch operations
- Consider parallel index maintenance for INSERTs
 - DB2 SMP feature installed and enabled
- Drop indexes when inserting into an empty table
- Consider dropping indexes when adding, changing or deleting more than 50% of the rows
 - Use SMP to create indexes in parallel
 - (INSERT + INDEX CREATION) < (INSERT + INDEX MAINT)

Modernize your indexes

▪ Look for Max 4G indexes

- Recreate to avoid failure
- Improve performance

To find logical and Physical file indexes that have a maximum size of 4 GB run the following query.

```
Select *
from qsys2.syspartitionindexstat
  where (table_schema not like 'Q%' or table_schema = 'QUSRSYS') and
        ACCPTH_TYPE=1 Order by INDEX_SIZE DESC;
```

CHGLF FILE(DTALIB/PF1) ACCPHTSIZ(*MAX1TB)

CHGPF FILE(DTALIB/PF1) ACCPHTSIZ(*MAX1TB)

Modernize your indexes

• Look for highly used small page indexes

- Recreate these with bigger logical pagesize to reduce total number of Index page fault

Run the following to find LFs that have the small page Size.

```
Select * from qsys2.syspartitionindexstat where (
table_schema not like 'Q%' or table_schema = 'QUSRSYS')
and LOGICAL_PAGE_SIZE < 65535 and INDEX_TYPE
='LOGICAL'
```

Look at most time consuming statements

Statements:

Start...	Most Expensi...	Total Processing Time ▼	Total Times Run	Average ...	Statement
8/19/1...	1.8772	470942.5765	2269470672	0.0002	SET : H : H
8/19/1...	63.41	89556.0975	2320	38.1707	UPDATE E
8/16/1...	0.84	52140.2131	2491657	0.0209	SELECT *
8/19/1...	11926.6278	51529.4332	14	3680.6738	select surr
8/19/1...	63.5648	19866.3376	839	23.6785	UPDATE J
8/19/1...	0.3173	11471.3181	2269473634	0.0000	SELECT I
8/19/1...	0.1502	9054.7384	2269470601	0.0000	SET : H : H
8/19/1...	2.5497	7906.1122	202476	0.0279	SELECT R
8/19/1...	18.7649	7255.7783	1781	4.0739	delete from
8/15/1...	3657.8846	7249.1975	2	3624.5987	select a.HI
8/14/1...	7.4970	4824.6271	1130	4.2695	update EM
8/19/1...	1.1439	4531.3062	10623485	0.0004	SELECT G
8/19/1...	0.1640	4512.6404	2269615350	0.0000	SET : H : H
8/18/1...	15.1699	4126.1909	954	4.3251	UPDATE P
8/19/1...	15.0886	3491.6234	965258	0.0036	UPDATE W
8/19/1...	2.9306	3127.2694	425880	0.0073	SELECT T
8/19/1...	408.9050	2744.2981	7	392.0425	SELECT G
8/16/1...	2540.9420	2540.9420	1	2540.9420	select a.HI
8/19/1...	6.5276	2326.4863	2803	0.8299	select oh.v
8/19/1...	467.3870	1265.5461	3	421.8487	SELECT G
8/19/1...	1.3598	1190.0457	16785	0.0708	select wee
8/19/1...	0.2208	1147.4386	9939517	0.0001	SELECT G
8/19/1...	40.4942	1113.2438	1121	0.9930	SELECT R
8/19/1...	11.5668	1081.1018	1257	0.8600	select SUF
8/19/1...	35.0887	1038.8169	96	10.8210	SELECT R
8/19/1...	1.8694	1001.2396	131408	0.0076	select aba
8/19/1...	211.6538	993.1426	5	198.6285	SELECT T

Status: Complete

Columns... Save Results... Refresh

Close Help ?

Explain from this list

Statements:

Start...	Most Expensi...	Total Processing Time ▼	Total Times Run	Average ...	Statement
8/19/1...	1.8772	470942.5765	2269470672	0.0002	SET : H : H
8/19/1...	63.4109	89556.0975	2320	38.1707	UPDATE E
8/16/1...	0.8467	52140.2131	2491657	0.0209	SELECT *
8/19/1...	11926.6278	51529.4332	14	3680.6738	select surr
8/19/1...	63.5648	19866.3376	839	23.6785	UPDATE J
8/19/1...	0.3173	11471.3181	2269473634	0.0000	SELECT I
8/19/1...	0.1502	9054.7384	2269470601	0.0000	SET : H : H
8/19/1...	2.5497	7906.1122	202476	0.0279	SELECT R
8/19/1...	18.7649	7255.7783	1781	4.0739	delete from
8/15/1...	3657.8846	7249.1975	2	3624.5987	select a.HI
8/14/1...	7.4970	4824.6271	1130	4.2695	update EM
8/19/1...	1.1439	4531.3062	10623485	0.0004	SELECT G
8/19/1...	0.1640	4512.6404	2269615350	0.0000	SET : H : H
8/18/1...	15.1699	4126.1909	954	4.3251	UPDATE P
8/19/1...	15.0886	3491.6234	965258	0.0036	UPDATE W
8/19/1...	2.9306	3127.2694	425880	0.0073	SELECT T
8/19/1...	408.9050	2744.2981	7	392.0425	SELECT G
8/16/1...	2540.9420	2540.9420	1	2540.9420	select a.HI
8/19/1...	6.5276	2326.4863	2803	0.8299	select oh.v
8/19/1...	467.3870	1265.5461	3	421.8487	SELECT G
8/19/1...	1.3598	1190.0457	16785	0.0708	select wee
8/19/1...	0.2208	1147.4386	9939517	0.0001	SELECT G
8/19/1...	40.4942	1113.2438	1121	0.9930	SELECT R
8/19/1...	11.5668	1081.1018	1257	0.8600	select SUF
8/19/1...	35.0887	1038.8169	96	10.8210	SELECT R
8/19/1...	1.8694	1001.2396	131408	0.0076	select aba
8/19/1...	211.6538	993.1426	5	198.6285	SELECT T

Status: Complete

Columns... Save Results... Refresh

Close Help ?

Visual Explain
Work with SQL Statement
Work with SQL Statement and Variables
Save to New...

Look for I/O intensive statements

Visual Explain - sess01 - Ctcdsv7r1(Mcv7r1)

File View Actions Options Help

Final Select
1,183
And
1,183
Logic
Correlated List Clean
1,183
Temporary Sorted List
1,183
Table Primitives
1,183
Index Probe
1,183

Attribute	Value
Total Estimated Row Time (ms)	282,153
Actual Runtime Information	
Optimization Time (ms)	116
Run Time (ms)	Not Available
Statement Open Time (ms)	
Statement Fetch Time (ms)	
Statement Close Time (ms)	
Rows Fetched	
Total Times Query Was Run	
Total Time For All Runs (hrs)	
CPU I/Os	315,008
Synchronous Database Reads	32,032
Asynchronous Database Reads	372,831
Page Faults	35,962
Temporary Storage Used (MB)	3
Total Times Temporary Results ...	0.0
Reason Temporary Result Was ...	Table Contents Changed
Information about SQL statements...	
Statement Number	Not Available
Statement Function	
Statement Operation	
Statement Type	
Statement Name	
Statement Outcome	
SQL Statement Text	

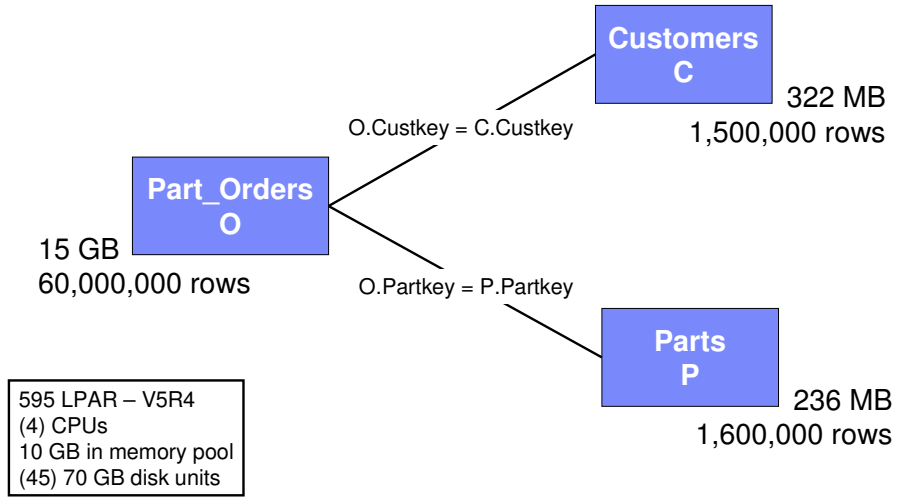
Use the Arrow View to see actual record counts

This index did not have all of the local select selection Columns

Use these stats to understand the I/O cost of the STMT

Indexing Case Study

Indexing Strategy – Case Study



Indexing Strategy – Case Study

- 80 SQL requests from a single JDBC connection...
 - 2 SETs
 - 53 SELECTs
 - 15 INSERTs
 - 5 UPDATEs
 - 15 DELETEs
 - 73 via SQE
 - 5 via CQE
- Scenarios...
 - No indexes
 - Indexes on join columns only
 - 4 radix indexes
 - Indexes for selecting, joining, grouping, ordering
 - 13 radix indexes
 - 2 encoded vector indexes

Indexing Strategy – Case Study

- Indexes on join columns only
 - ✓ create index part_orders_ix1 on part_orders (custkey);
 - ✓ create index part_orders_ix2 on part_orders (partkey);
 - ✓ create index customers_ix1 on customers (custkey);
 - ✓ create index parts_ix1 on parts (partkey);
- Index for selecting, joining, grouping, ordering
 - ✓ create index part_orders_ix3 on part_orders (returnflag, custkey);
 - ✓ create index part_orders_ix4 on part_orders (shipmode, custkey);
 - ✓ create index part_orders_ix5 on part_orders (orderkey, linenum, custkey);
 - ✓ create index part_orders_ix6 on part_orders (orderkey, custkey);
 - ✓ create index part_orders_ix7 on part_orders (returnflag, partkey);
 - ✓ create index part_orders_ix8 on part_orders (shipmode, partkey);
 - ✓ create index part_orders_ix9 on part_orders (orderkey, linenum, partkey);
 - ✓ create index customers_ix2 on customers (customer, custkey);
 - ✓ create index parts_ix2 on parts (part, partkey);
- ✓ create encoded vector index part_orders_evi1 on part_orders (returnflag);
- ✓ create encoded vector index part_orders_evi2 on part_orders (shipmode);

Indexing Strategy – Case Study

– Sample of SQL Requests

- select *
 - from part_orders
 - where custkey = 1
 - and orderkey = 303008;

Highly Selective
- select *
 - from part_orders o, customers c
 - where o.custkey = c.custkey
 - and c.customer = 'Customer#000000001';

Highly Selective
2way Join
- select *
 - from part_orders o, customers c, parts p
 - where o.custkey = c.custkey
 - and o.partkey = p.partkey
 - and c.customer = 'Customer#000000001'
 - and o.orderkey = 303008
 - order by o.linenum;

Highly Selective
3way Join
Ordering

Indexing Strategy – Case Study

– Sample of SQL Requests

- select distinct shipmode
- from part_orders
- order by shipmode;

- select shipmode, count(*)
- from part_orders
- group by shipmode
- order by 2 desc;

No Local Selection
Distinct
Ordering

No Local Selection
Grouping
Ordering

Indexing Strategy – Case Study

– Sample of SQL Requests

- update part_orders set expander = 'UPDATED'
- where custkey = 1;

- delete from part_orders
- where custkey = 1;

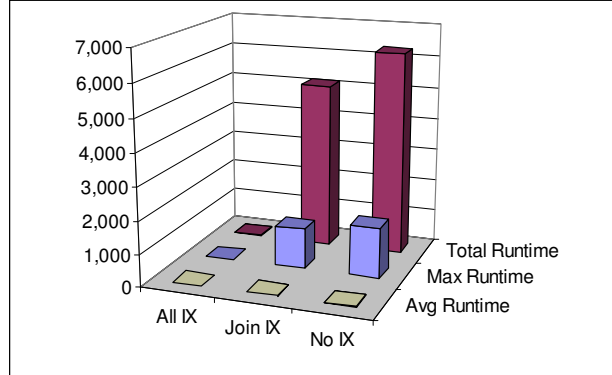
- insert into part_orders
- select * from item_subset1;

Searched Update
Highly Selective

Searched Delete
Highly Selective

Select Small Set
And Insert

Indexing Strategy – Case Study Results



	Total Time	Max Time	Avg Time
All Indexes	23.547	2.493	0.076
Join Indexes	5,138.851	1,249.081	20.975
No Indexes	6,302.275	1,533.910	20.265

Indexing Strategy – Case Study Results

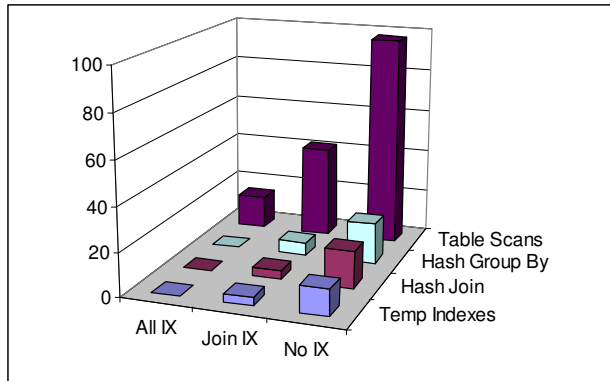
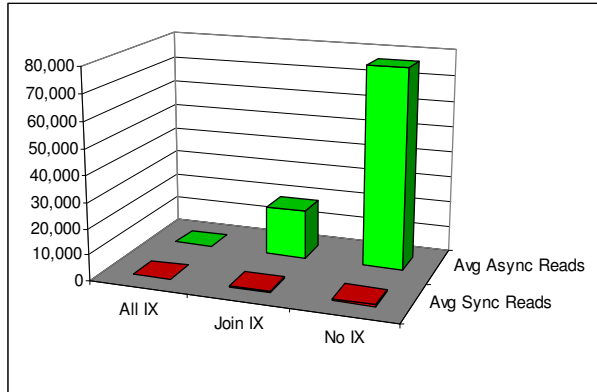


	Table Scans	Hash GroupBy	Hash Join	Temp Indexes
All Indexes	15	0	0	0
Join Indexes	42	6	4	4
No Indexes	97	19	17	12

Indexing Strategy – Case Study Results



	Avg Async Reads	Avg Sync Reads
All Indexes	15	0
Join Indexes	42	6
No Indexes	97	19

Additional Information

- Indexing and Statistics strategies Whitepaper

- <http://www-304.ibm.com/partnerworld/wps/servlet/ContentHandler/servers/enable/site/bi/strategy/index.html>
- This Paper was recently updated to include 6.1 and 7.1 content.

- DB2 for i SQL & Query Performance Tuning and Monitoring Workshop

- <http://www-03.ibm.com/systems/i/software/db2/db2performance.html>

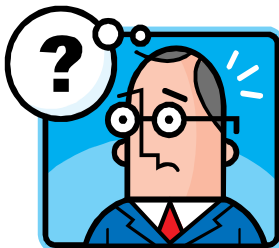
- Text Search indexing technology

- https://www-304.ibm.com/partnerworld/wps/servlet/ContentHandler?contentId=a7QzcNSQBe_4MDADcnt&roadMapId=IbOtoNReUYN4MDADrdm&roadMapName=Education+resources+for+IBM+i+systems&locale=en_US

Additional Information

- DB2 for i Websites
 - Home Page: ibm.com/systems/i/db2
 - DeveloperWorks Zone: ibm.com/developerworks/db2/products/db2i5OS
 - Porting Zone: ibm.com/partnerworld/i/db2porting
- Newsgroups & Forums
 - USENET: comp.sys.ibm.as400.misc, comp.databases.ibm-db2
 - DeveloperWorks: <https://www.ibm.com/developerworks/forums/forum.jspa?forumID=292>
 - System i Network DB2 Forum: <http://forums.systeminetwork.com/isnetforums/>
- Education Resources - Classroom & Online
 - ibm.com/systemi/db2/gettingstarted.html
 - ibm.com/partnerworld/wps/training/i5os/courses
- DB2 for i Publications
 - White Papers: ibm.com/partnerworld/wps/whitepaper/i5os
 - Online Manuals: ibm.com/systems/i/db2/books.html
 - DB2 for i Redbooks (<http://ibm.com/redbooks>)
 - [Getting Started with DB2 Web Query for System i \(SG24-7214\)](#)
 - [OnDemand SQL Performance Analysis ... in V5R4 \(SG24-7326\)](#)
 - [Preparing for and Tuning the SQL Query Engine on DB2 for i5/OS \(SG24-6598\)](#)
 - [Modernizing iSeries Application Data Access \(SG24-6393\)](#)

- Are you experiencing performance problems?
- Are you using SQL?
- Are you getting the most out of DB2 for i?



Need help?

IBM DB2 for i Center of Excellence

- ✓ Database modernization
- ✓ DB2 Web Query
- ✓ Database architecture and design
- ✓ DB2 SQL performance analysis and tuning
- ✓ Data warehousing and Business Intelligence
- ✓ DB2 for i education and training

Contact: Tom McKinley mac2@us.ibm.com
 IBM Systems and Technology Group
 Rochester, MN USA



IBM Systems Lab Services and Training

Our Mission and Profile

- Support the IBM Systems Agenda and accelerate the adoption of new products and solutions
- Maximize performance of our clients' existing IBM systems
- Deliver technical training, conferences, and other services tailored to meet client needs
- Team with IBM Service Providers to optimize the deployment of IBM solutions (GTS, GBS, SWG Lab Services and our IBM Business Partners)

Our Competitive Advantage

- Leverage relationships with the IBM development labs to build deep technical skills and exploit the expertise of our developers
 - Combined expertise of Lab Services and the Training for Systems team
 - Skills can be deployed worldwide to assure all client needs can be met
- Successful worldwide history:**
 17 years in Americas, 9 years in Europe/Middle East/Africa,
 5 years in Asia Pacific

Mainframe Systems

Power Systems

System x & BladeCenter

System Storage

IT Infrastructure Optimization

Data Center Services

Training Services

www.ibm.com/systems/services/labservices stgls@us.ibm.com



IBM Systems Lab Services and Training Power Services

Key Offerings

- High Availability Services on Power Systems (including Advanced Copy Services for PowerHA™ on IBM i)
- Systems Director Services
- PowerCare Services
- Performance and Scalability services (including system, application, and database tuning)
- Virtualization Services for AIX® on Power Systems™
- Application and database modernization consulting (SOA implementation)
- Linux® on Power consulting, custom application development, implementation, and optimization services
- Security on Power consulting and implementation services
- System consolidation and migration service
- High Performance Computing consulting and implementation services
- SAP® on IBM i consulting
- Power Blades on BladeCenter (including VIOS on i and blades running IBM i implementation)
- Smart Analytics services (including DB2® Web Query implementation and consulting)
- Public, private, customized and self-paced virtual training
- Power Systems Technical University

Americas, WW Contacts

Mark Even

even@us.ibm.com, 507-253-1313
 IBM i

Frank Kriss

kriss@us.ibm.com, 507-253-1354
 IBM i, High Availability

Karen Anderson

kanders@us.ibm.com, 972-561-6337
 IBM i Vouchers

Stephen Brandenburg

sbranden@us.ibm.com, 301-803-6199
 PowerVouchers, Virtualization Program, AIX

George Henningsen

gehenni@us.ibm.com, 516-349-3530
 SWOT/SWAT, AIX

Allen Johnston

allenr@us.ibm.com, 704-340-9165
 PowerCare

Dawn May

dmmay@us.ibm.com, 507-253-2121
 Power Performance and Scalability Center

www.ibm.com/systems/services/labservices stgls@us.ibm.com

Summary

- Pay Attention to New indexing Capabilities in DB2 for IBM i in 6.1 and 7.1

- Need to monitor, analyze and create the most beneficial indexes
 - Its an iterative process
 - Don't just create. Drop indexes that are not used

- The right set of indexes can:
 - have a significant positive impact on the performance of your application
 - Improve overall system health



Thank You

Appendix

IBM OmniFind Text Search Server for DB2 for i

- New IBM i product offering: 5733-OMF
 - *No-charge* offering
 - Requires IBM i 6.1

- Delivers common DB2 Family text search technology
 - Advanced, linguistic high-speed searches
 - Support enabled for any character-based column
 - Search technology also supports Rich Text document formats
 - Example: LOB columns containing PDF or Microsoft® Word documents
 - IFS documents can be indexed with extra programming
 - Includes support for 26 different languages

OmniFind Database Requirements

- Table must have primary key, unique key constraint or ROWID column
 - Physical files are supported
 - Unique-keyed physical file requires usage of ADDPFCST to register key as constraint

- Supported column data types:
 - BINARY, VARBINARY
 - BLOB
 - CHAR, VARCHAR
 - CLOB
 - DBCLOB
 - GRAPHIC, VARGRAPHIC
 - XML

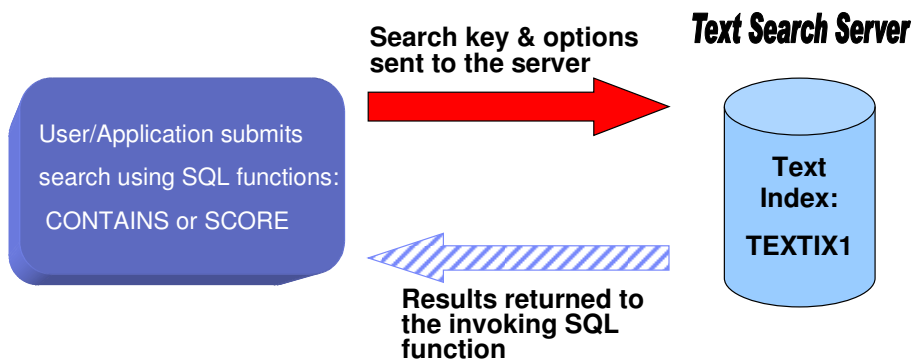
OmniFind Database Requirements

- Some of the supported document format types:
 - Plain text
 - XML
 - HTML
 - Adobe PDF
 - Rich Text Format (RTF)
 - JustSystems Ichitaro
 - Microsoft Excel
 - Microsoft PowerPoint
 - Microsoft Word
 - Lotus® 123
 - Lotus Freelance®
 - Lotus WordPro
 - Lotus Symphony
 - OpenOffice 1.1 & 2.0
 - OpenOffice Calc
 - Quattro Pro
 - StarOffice Calc

Text Index vs DB2 Index

- Text indexes stored outside of DB2 in IFS
- Text indexes have delayed maintenance
 - Changes logged to staging table
 - Index maintenance is scheduled
- Text indexes not protected by IBM i SMAPP
- Text indexes utilize different indexing methods
 - Table with VARCHAR(32000) column and 175,000 rows
 - DB2 Index object size: 1.7 GB
 - Text Index object size: 0.1 GB

OmniFind Search Processing



Note: CONTAINS and SCORE functions only supported by the SQL Query Engine (SQE)

CONTAINS function

- **CONTAINS(*column-name, search-argument, options*)**
 - Column-name: column over which the text search index is built
 - Search-argument: text being searched for
 - Options: optional parameter, can be used to modify the query language or activate synonym matching
- Function Output (integer value):
 - 1 - match was found
 - 0 - no match found

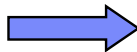
CONTAINS example

- **Find matches on exact string 'New product interest' in the COMMENT column, use a host variable to pass search string**

```
char search_arg[100];
```

```
...
```

```
EXEC SQL DECLARE C1 CURSOR FOR
  SELECT custkey
  FROM customers
```



```
WHERE CONTAINS(comment, :search_arg) = 1
ORDER BY CUSTKEY;
```

```
EXEC SQL SET :search_arg = "New product interest";
```

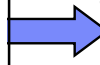
```
EXEC SQL OPEN C1;
```

SCORE Function

- **SCORE**(*column-name, search-argument, options*)
 - Column-name: column over which the text search index is built
 - Search-argument: text being searched for
 - Options: optional parameter, can be used to modify the query language or activate synonym matching
- Function Output:
 - Value between 0 and 1, up to 3 decimal points
 - Higher value indicates a better match on the specified search

SCORE Example

- Find those thesis reports that discuss programming from a performance or parallel perspective along with the normalized score rating



```
SELECT projID, projAuthor,  
INTEGER(SCORE(thesis,  
'programming AND (parallel OR performance)') * 100)  
AS relevance  
FROM projects  
WHERE CONTAINS(thesis, 'programming AND (parallel OR  
performance)')=1  
ORDER BY relevance DESC
```



For More info on Omnifind for i

▪ Get PDF at:

- <http://publib.boulder.ibm.com/infocenter/iseri/v6r1m0/index.jsp?topic=/rzahg/rzahgdb2textsearchpdf.htm>

▪ Or Google "OmniFind Text Search Server for DB2 for i"



Trademarks and Disclaimers

Adobe, Acrobat, PostScript and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce. ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Cell Broadband Engine and Cell/B.E. are trademarks of Sony Computer Entertainment, Inc., in the United States, other countries, or both and are used under license therefrom.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Information is provided "AS IS" without warranty of any kind.

The customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer.

Information concerning non-IBM products was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by IBM. Sources for non-IBM list prices and performance numbers are taken from publicly available information, including vendor announcements and vendor worldwide homepages. IBM has not tested these products and cannot confirm the accuracy of performance, capability, or any other claims related to non-IBM products. Questions on the capability of non-IBM products should be addressed to the supplier of those products.

All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Some information addresses anticipated future capabilities. Such information is not intended as a definitive statement of a commitment to specific levels of performance, function or delivery schedules with respect to any future products. Such commitments are only made in IBM product announcements. The information is presented here to communicate IBM's current investment and development activities as a good faith effort to help with our customers' future planning.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Prices are suggested U.S. list prices and are subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.