



IBM System i™

## Indexing Strategies for DB2 for i5/OS

Presented by Jarek Miszczyk  
DB2 for i5/OS Technology Team  
IBM Rochester, MN USA

*i want stress-free IT.*  
*i want control.*  
*i want an **i**.*

© Copyright IBM Corporation, 2007. All Rights Reserved.  
This publication may refer to products that are not currently available in your country. IBM makes no commitment to make available any products referred to herein.

IBM System i



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



*i want an **i**.*

© 2007 IBM Corporation

## Indexing Technology within DB2 for i5/OS Review

IBM System i

### DB2 for i5/OS

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

*i want an i.* © 2007 IBM Corporation

## 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	SMALL
001	RED	LARGE
003	BLACK	SMALL
004	GREEN	MEDIUM

...WHERE COLOR = 'BLACK' AND ITEM\_NO = 003

...WHERE SIZE = 'MEDIUM'

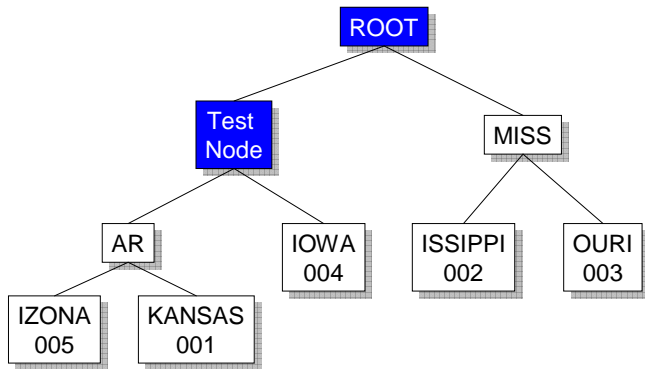
...WHERE ITEM\_NO = 001 AND SIZE = 'LARGE'

## 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**

# 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

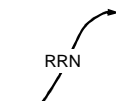
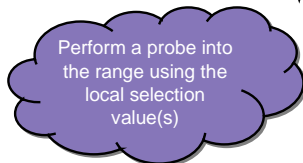
Given an index on table EMPLOYEE keyed on STATE...

```

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

EMPLOYEE Index

STATE
...
<b>IOWA (004)</b>
IOWA (007)
IOWA (010)
IOWA (017)
KANSAS (011)
MISSISSIPPI (002)
MISSISSIPPI (013)
MISSOURI (003)
...



EMPLOYEE Table

RRN	STATE
001	ARKANSAS
002	MISSISSIPPI
003	MISSOURI
<b>004</b>	<b>IOWA</b>
005	ARIZONA
006	MONTANA
007	IOWA
008	NEBRASKA
009	NEBRASKA
010	IOWA
011	KANSAS
012	WISCONSIN
013	MISSISSIPPI
014	WISCONSIN
015	WISCONSIN
016	ARKANSAS
017	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 iSeries Navigator GUI

```
CREATE ENCODED VECTOR INDEX
  SchemaName/IndexName ON SchemaName/TableName
  (ColumnName)
  WITH n DISTINCT VALUES;
```

## Encoded Vector Index (EVI)

Symbol Table				
Key Value	Code	First Row	Last Row	Count
Arizona	1	1	80005	5000
Arkansas	2	5	99760	7300
...				
Wisconsin	49	7	30111	340
Wyoming	50	252	83000	2760

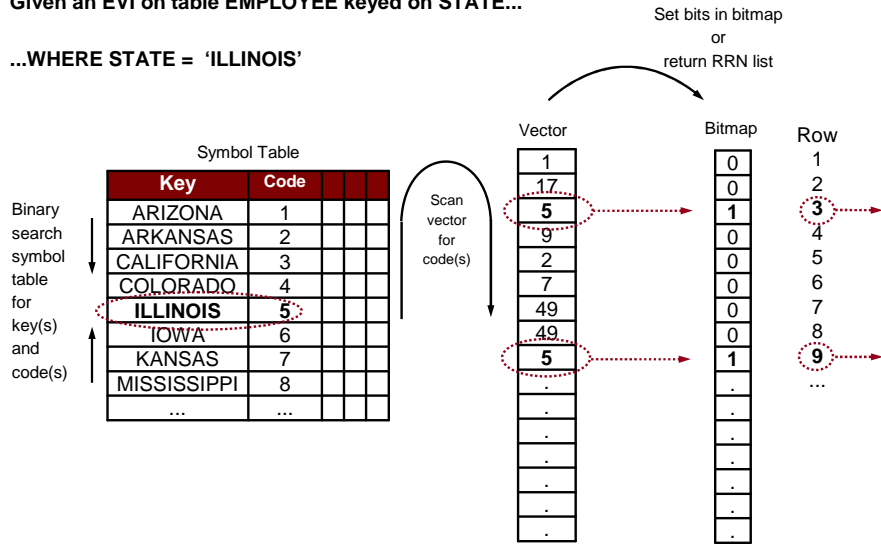
Vector	RRN
1	1
17	2
5	3
9	4
2	5
7	6
49	7
49	8
5	9
...	...

- 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
- Rather than a bit array for each distinct key value, use one array of codes

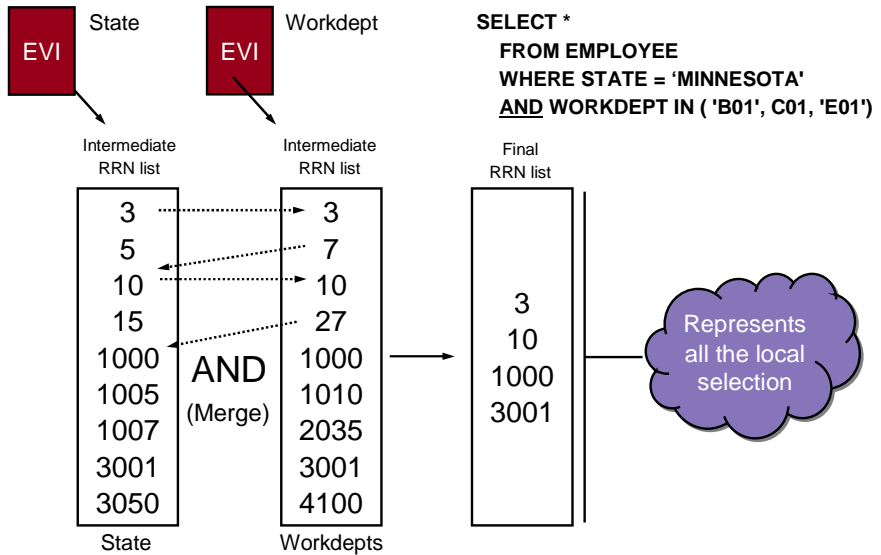
## Bitmap / RRN List Example

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

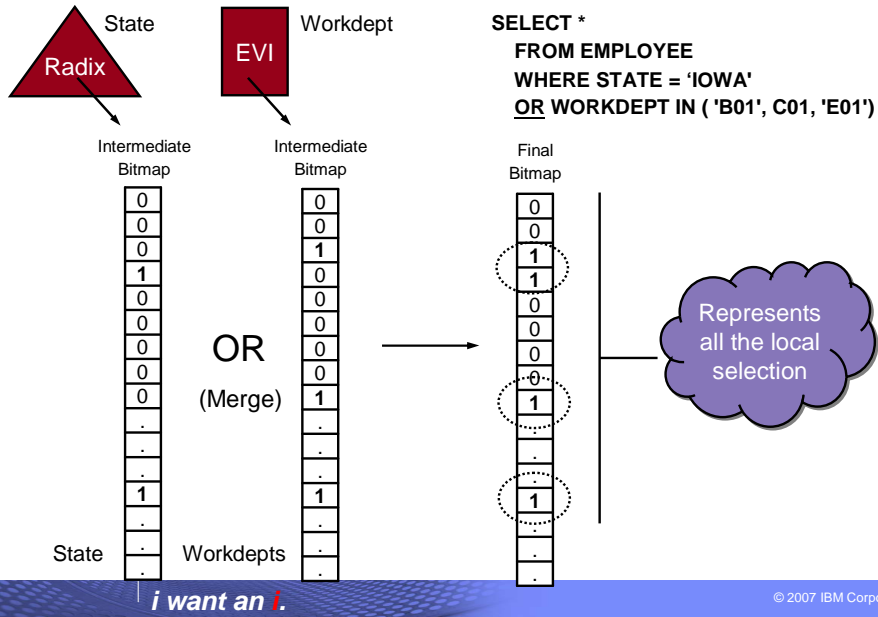
...WHERE STATE = 'ILLINOIS'



## Index ANDing / ORing Example



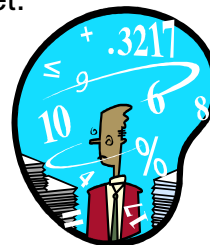
## Index ANDing / ORing Example 1



## DB2 for i5/OS

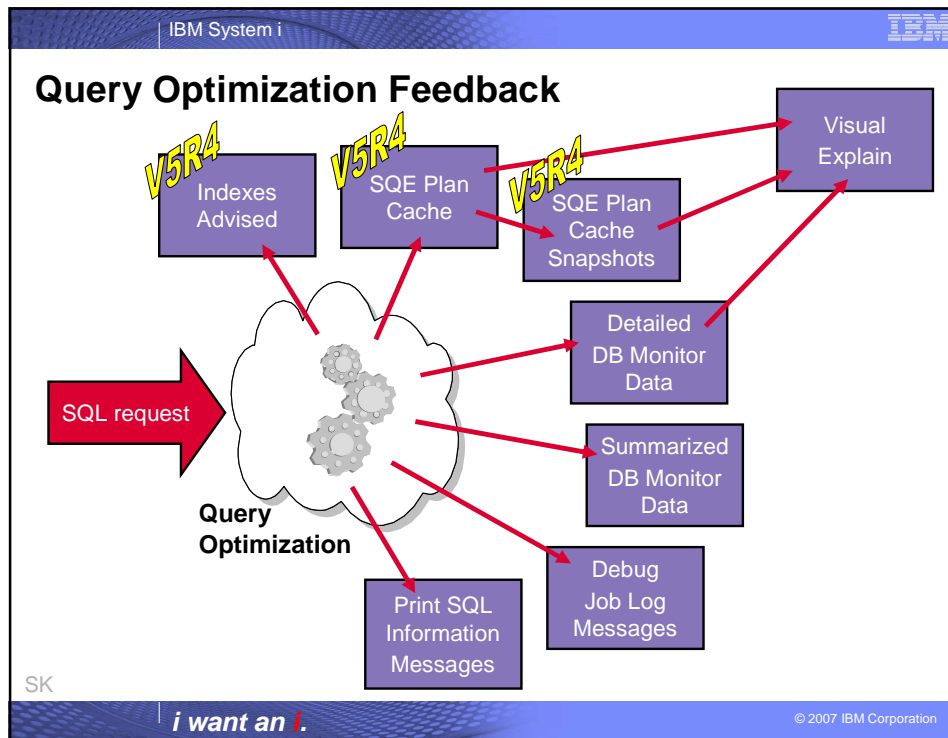
cardinality The number of elements in a set.

- 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

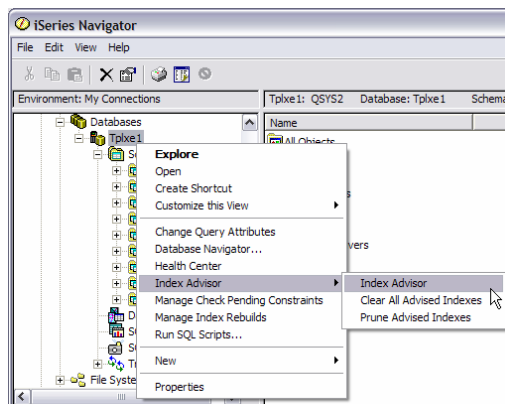


- IBM System i
- ## Indexing Advice from the Optimizer
- Both CQE and SQE provide index creation advice
  - CQE
    - Basic advice
    - Radix index only
    - Based on table scan and local selection columns only
    - Temporary index creation information also provides insight
    - CQE Visual Explain will try and tie pieces together to advice 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
- i want an i.*
- © 2007 IBM Corporation

## Index Advised – System wide

- New V5R4 feature
- System wide index advice
  - Data is placed into a DB2 table (QSYS2/SYSIXADV)
  - Autonomic
  - No overhead
- CQE and SQE support
  - CQE only provides basic advice based on local selection predicates
  - SQE provides complex advice based on all parts of the query
    - Not complete, but much better
- GUI interface via iSeries Navigator
  - Advice for System, or Schema, or Table
- System only adds (summary) rows, user must manage the data
  - Options to clear or prune
- Can create indexes directly from GUI
  - Additional indexing analysis might be required to determine the optimal index

## Index Advised – System wide



IBM System i

## Index Advised – System wide

Index Advisor - Tplxe1

Database: Tplxe1    Advised Indexes for Tplxe1

Table for Which Index was Advised	Schema	Short Name	Partition	Keys Advised	Leading Keys Order Independent
CUST_DIM	IXSTAR10G	CUST_DIM		CUSTKEY	CUSTKEY
CUST_DIM	IXSTAR10G	CUST_DIM		CUSTOMER	CUSTOMER
CUST_DIM	IXSTAR10G	CUST_DIM		CUSTOMER, CUSTKEY	CUSTOMER
CUST_DIM	IXSTAR10G	CUST_DIM		CUSTOMER	CUSTOMER
CUST_DIM	MCSTAR10G	CUST_DIM		CUSTKEY	CUSTKEY
CUST_DIM	MCSTAR10G	CUST_DIM		CUSTOMER	CUSTOMER
CUST_DIM	MCSTAR10G	CUST_DIM		CUSTOMER, CUSTKEY	CUSTOMER
CUSTOMERS	MERS			MKTSEGMENT	
CUSTOMERS	MERS			CUSTOMER, CUSTKEY	CUSTOMER
CUSTOMERS	MERS			COUNTRY, CONTINENT, CUSTKEY	COUNTRY, CON...
CUSTOMERS	MERS			SALESPERSON, CUSTKEY	SALESPERSON

Context menu for selected row:

- Create Index
- Remove from List
- Show Statements...
- Table

Keys Advised	L. K.	Index Type Advised	Last Advised for Query Use	Times Advised for Query Use	Estimated Index Creation Time	Reason Advised
CUSTOMER, CUSTKEY	C..	Binary Radix	2/26/06 8:48:04 PM	53	00:00:29	Record selection
CUSTOMER	C..	Binary Radix	2/26/06 11:23:19 AM	4	00:01:34	Record selection
CUSTKEY	C..	Binary Radix	2/27/06 10:27:15 PM	10	00:00:10	Record selection
CUSTOMER	C..	Binary Radix	2/27/06 10:34:31 PM	3	00:00:26	Record selection
CUSTOMER, CUSTKEY	C..	Binary Radix	2/27/06 10:34:49 PM	33	00:00:29	Record selection
MKTSEGMENT		Binary Radix	2/20/06 4:28:18 PM	17	00:00:01	Ordering/Grouping

*i want an i.* © 2007 IBM Corporation

IBM System i

## Index Advised – System wide – Show Statements

SQL Plan Cache Statements - Tplxe1(Tplxe1)

Filters for statement list:

- Minimum runtime for the longest execution:  Seconds
- Queries run after this date and time: Feb 28, 2006 8:58:45 AM
- Top 'n' most frequently run queries:
- Top 'n' queries with the largest total accumulation:

List of statements:

Last Time Run	Most Exp...	Total Pro...	Total Tim...	Statement
Feb 27, 2006 1...	1.415	1.415	1	create table cust_subset
Feb 27, 2006 1...	0.126	0.126	1	create table cust_subset
Feb 27, 2006 1...	0.1256	0.1256	1	create table cust_subset
Feb 27, 2006 1...	0.1255	0.1255	1	create table cust_subset
Feb 27, 2006 1...	0.1254	0.1254	1	create table cust_subset
Feb 28, 2006 6...	0.0288	0.0288	1	select * from part_orders
Feb 28, 2006 6...	0.0255	0.0255	1	select * from part_orders
Feb 28, 2006 6...	0.0247	0.0247	1	select * from part_orders
Feb 28, 2006 6...	0.0166	0.0166	1	select * from part_orders
Feb 28, 2006 6...	0.0096	0.0096	1	select * from part_orders

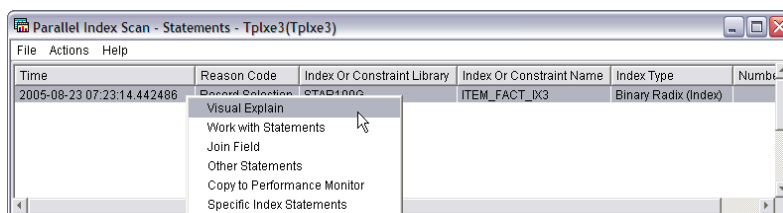
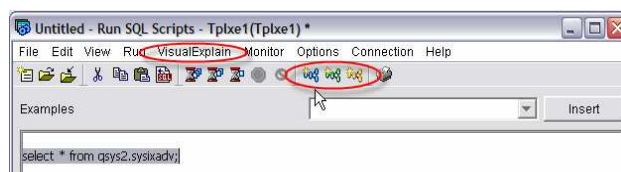
Queries processed by SQE only

*i want an i.* © 2007 IBM Corporation

## Visual Explain

- Enhanced in V5R4
- Graphical representation of query plan
  - Representation of the DB objects and data structures
  - Representation of the methods and strategy
  - Associated environmental information
  - **Advice on indexes and column statistics**
  - Highlighting of specific query rewrites
  - Highlighting of expensive methods
- CQE and SQE support
- GUI interface via iSeries Navigator
- Based on detailed optimizer information
  - SQE Plan Cache
  - SQE Plan Cache Snapshots
  - Detailed Database Monitor Data

## Visual Explain



# Visual Explain – Index Advisor

Visual Explain - Tpxe1 (Tpxe1)

File View Actions Options Help

Statistics and Index Advisor

Time Information

Timestamp for Creation of Monitor Entry	2006-01-24-13.29.59.655023
Statement Start Timestamp	2006-01-24-13.29.59.630325
Statement End Timestamp	2006-01-24-13.29.59.655008
Optimization Time, in Milliseconds	6
Total Time, in Microseconds	24688
Statement Open Time, in Microseconds	24688
Statement Fetch Time, in Microseconds	Not Available
Statement Close Time, in Microseconds	Not Available

Information about SQL statement executed

Statement Number	53
Statement Function	Select
Statement Operation	Open
Statement Type	Dynamic
Statement Name	STMTO040
Statement Outcome	Unsuccessful
SQL Return Code	-666
SQLSTATE	57005
Cursor Name	CRSR0040
Package Name	
Package Library	

select \* from qsys2.sysixadv where table\_schema = 'MCAIN'

Statement text | Optimizer messages

# Visual Explain – Index Advisor

Index and Statistics Advisor - Tpxe1 (Tpxe1)

Index Advisor | Statistics Advisor

It is recommended that the following indexes be created:

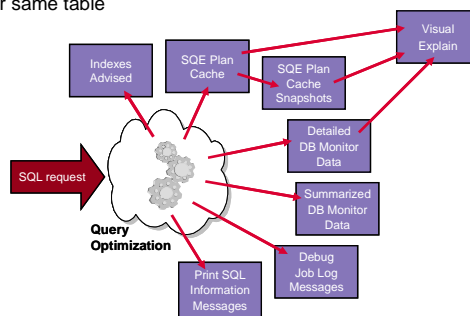
Create	Table Name	Schema	Index Type	Columns
<input checked="" type="checkbox"/>	SYSIXADV	QSYS2	Binary Radix	TABLE_SCHEMA

Create ...

OK Help ?

## Indexed Advised from other Mechanisms

- SQE Plan Cache (V5R4)
  - No direct index advice
  - Index advice via Snapshot data or Visual Explain
- SQE Plan Cache Snapshot (V5R4)
  - Enhanced SQE index advised
  - "3020" records to show multiple indexes for same table
  - Temporary index created
- Detailed Database Monitor (V5R4)
  - Enhanced SQE index advised
  - "3020" records to show multiple indexes for same table
  - Temporary index created
- Summary Database Monitor
  - No enhanced SQE index advised
  - Basic index advice
  - Temporary index created
- Debug Messages in Job Log
  - No enhanced SQE index advised
  - Basic index advice
  - Temporary index created
- Print SQL Information
  - No index advice
  - Temporary index created

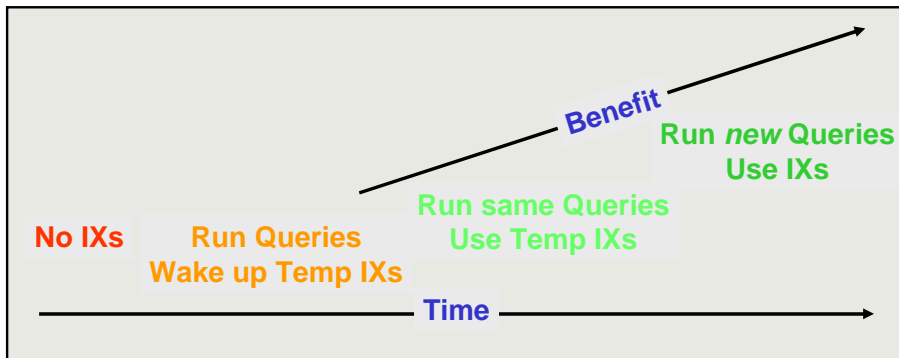


## Autonomic Index Creation

- 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
  - New feature in V5R4
  - 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

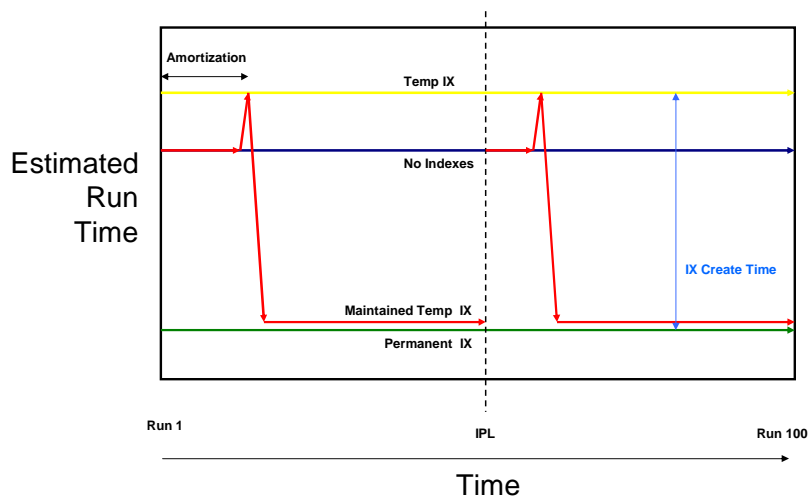
## Autonomic Index Creation

- 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



## Autonomic Index Creation – SQE in Action

Same query run 100 times...



IBM System i

## Index Evaluator (Show Indexes)

Meta data for perm and temp indexes...

SQL Name	S	I	C	S	T	I	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		D	Y	1	1	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		D	Y	1	1	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_QBNK1	I	M	M	D		D	Y	1	1	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_QBNK2	I	M	M	D		D	Y	1	1	2/2/06 4:25:55 PM	2/2/06 4:25:55 PM	0	69		0
DBMONITORX_QBNK3	I	M	M	D		D	Y	1	1	2/3/06 4:36:26 PM	2/3/06 4:36:26 PM	0	640		0

© 2007 IBM Corporation

# Indexing Strategies

MC

## DB2 for i5/OS

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

## DB2 for i5/OS

To be successful...

You must create some indexes!

## 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

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

} Minimum

### Encoded Vector Indexes

- Local selection column (single key)
- Join column (data warehouse - star or snowflake schema)

## Indexing Strategy - Examples

If the optimizer feedback indicates:

**Full table scan** → Create an index on local selection columns

**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

More information and examples at:

[ibm.com/servers/enable/site/education/abstracts/indxng\\_abs.html](http://ibm.com/servers/enable/site/education/abstracts/indxng_abs.html)

## Indexing Case Study

## 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';
```

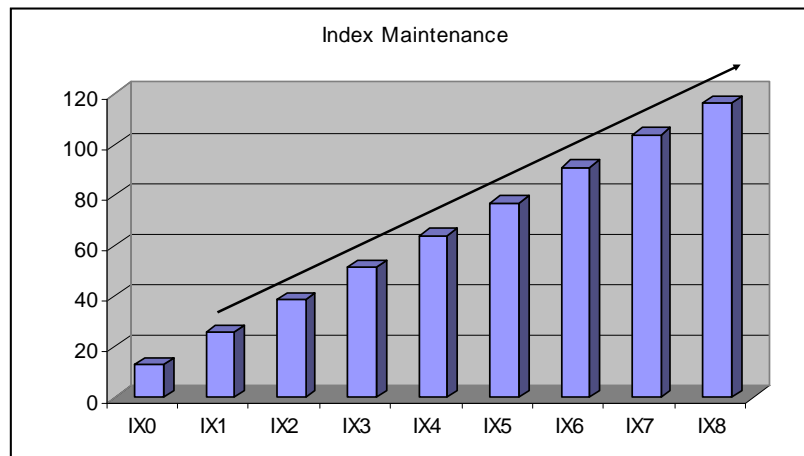
```
CREATE INDEX ORDERS_IX4 ON ORDERS (CUSTOMER_NO);
CREATE ENCODED VECTOR INDEX ORDERS_EVI4
ON ORDERS (ORDER_DATE);
```

## 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 – Maintenance



In general - index maintenance cost 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)
- Consider and watch out for access path protection (SMAPP)

Thank You

## Trademarks

### Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries. For a complete list of IBM Trademarks, see [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml): AS/400, DBE, e-business logo, ESCO, eServer, FICON, IBM, IBM Logo, iSeries, MVS, OS/390, pSeries, RS/6000, S/30, VM/ESA, VSE/ESA, Websphere, xSeries, z/OS, zSeries, z/VM

The following are trademarks or registered trademarks of other companies

Lotus, Notes, and Domino are trademarks or registered trademarks of Lotus Development Corporation  
Java and all Java-related trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries  
Linux is a registered trademark of Linux Torvalds  
UNIX is a registered trademark of The Open Group in the United States and other countries.  
Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.  
SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.  
Intel is a registered trademark of Intel Corporation  
\* All other products may be trademarks or registered trademarks of their respective companies.

### NOTES:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput 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 improvements equivalent to the performance ratios stated here.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

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

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

References in this document to IBM products or services do not imply that IBM intends to make them available in every country.

Any proposed use of claims in this presentation outside of the United States must be reviewed by local IBM country counsel prior to such use.

The information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.