

# What's coming from the Optimizer in DB2 9 for z/OS?

Terry Purcell, IBM Silicon Valley Lab  
 Session 1819  
 Thurs, 19/Oct, 09:30am – 10:45am



**TAKE BACK CONTROL**

**IBM INFORMATION ON DEMAND 2006**  
 October 15 – 20, 2006  
 Anaheim Convention Center  
 Anaheim, California

## Trademarks

2

AIX*	IBM eServer	z/VM*
CICS*	IBM logo*	zSeries*
DB2*	IMS	
DB2 Connect	On Demand Business logo	
DB2 Universal Database	Parallel Sysplex*	
DRDA*	System z	
FICON*	System z9	
GDPS*	WebSphere*	
HiperSockets	z/Architecture	
IBM*	z/OS*	

\* Registered trademarks of IBM Corporation

**The following are trademarks or registered trademarks of other companies.**

Intel is a trademark of the Intel Corporation in the United States and other countries.  
 Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.  
 Java and all Java-related trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc., in the United States and other countries.  
 Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.  
 UNIX is a registered trademark of The Open Group in the United States and other countries.

\* 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.

This presentation and the claims outlined in it were reviewed for compliance with US law. Adaptations of these claims for use in other geographies must be reviewed by the local country counsel for compliance with local laws.



**IBM INFORMATION ON DEMAND 2006**

**TAKE BACK CONTROL**

## Disclaimer

3

The information in this document has not been submitted to any formal IBM review and is distributed on an "as is" basis without any warranty expressed or implied. Use of this information or the implementation of any of these techniques is a user responsibility and depends on the user's ability to evaluate and integrate them into the user's operational environment. While each item may have been reviewed for accuracy in a specific situation there is no guarantee the same or similar results may be achieved elsewhere.



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Agenda

4

- REOPT AUTO
- Histogram Statistics
- Page Range Processing
- Global Query Optimization
- Generalized sparse index and in-memory data cache
- Dynamic Index ANDing
- Indexing Enhancements
- Optimization Service Center
- Misc Optimization Enhancements



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL



## REOPT Auto Based On Parameter Marker Change

TAKE BACK CONTROL



### REOPT enhancement for dynamic SQL <sup>6</sup>

- V8 REOPT options
  - Dynamic SQL
    - REOPT(NONE, ONCE, ALWAYS)
  - Static SQL
    - REOPT(NONE, ALWAYS)
  
- V9 Addition for Dynamic SQL
  - Bind option REOPT(AUTO)



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Dynamic SQL REOPT - AUTO

- For dynamic SQL with parameter markers
  - DB2 will automatically reoptimize the SQL when
    - Filtering of one or more of the predicates changes dramatically
      - Such that table join sequence or index selection may change
    - Some statistics cached to improve performance of runtime check
  - Newly generated access path will replace the current in the statement cache.
  
- First optimization is the same as REOPT(ONCE)
  - Followed by analysis of the values supplied at each execution of the statement



## REOPT(AUTO) and DSC

- Two new columns for DSN\_STATEMENT\_CACHE\_TABLE.

Column Name	Description
BIND_RA_TOT	Total number of rebinds that have been issued for the dynamic statement due to REOPT(AUTO).
BIND_RO_TYPE	'N' - REOPT(NONE) or its equivalent '1' - REOPT(ONCE) or its equivalent 'A' - REOPT(AUTO) 'O' - Current plan is deemed as optimal and no need for further REOPT(AUTO)



# Histogram Statistics

TAKE BACK CONTROL



10

## RUNSTATS Today

- Distribution statistics exist for data skew of a single value:
  - Eg. STATUS Y = 99%, N = 1%
- Provides more information to the Optimizer for literal values
  - Instead of assuming data is evenly distributed
    - Y (50%) & N (50%)
  - Optimizer knows that 99% is Y and 1% is N
    - May have significant effect on choice of index or join sequence/method



## RUNSTATS Today (cont.)

- But what about data skew across a range of values?
  - For example, Sales are highest in the 2 weeks before Christmas
    - SALES\_DATE BETWEEN '2006-12-11' AND '2006-12-24' returns significantly more rows than a 2 week range in March
    - This knowledge could alter the optimizer's access path choice



## RUNSTATS Histogram Statistics

- **RUNSTATS** will produce equal-depth histogram
  - i.e. each quantile (range) will have about the same number of rows (not the same number of values)
  - Another term is range frequency (differs from value frequency)
- Example
  - 1, 3, 3, 4, 4, 6, 7, 8, 9, 10, 12, 15 (sequenced)
  - Lets cut that into 3 quantiles.
    - 1, 3, 3, 4, 4                      6,7,8,9                      10,12,15

Seq No	Low Value	High Value	Cardinality	Frequency
1	1	4	3	5/12
2	6	9	4	4/12
3	10	15	3	3/12



## RUNSTATS Histogram Statistics Notes 13



- RUNSTATS
  - Maximum 100 quantiles for a column
  - Same value columns WILL be in the same quantile
  - Quantiles will be similar size but:
    - Will try and avoid big gaps between quantiles
    - Highvalue and lowvalue may have separate quantiles
    - Null WILL have a separate quantile
  
- Supports column groups as well as single columns



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Histogram Statistics Benefits 14

- Especially beneficial when gaps exist in ranges
  - Example, SAP uses INTEGER (or worse, VARCHAR) to store YEAR-MONTH data
    - Optimizer does not realize there are no values between 200512 and 200601
      - 200513 – 200600 are valid numeric values, but invalid year/month
    - Optimizer assumes
      - BETWEEN 200512 AND 200601  **90 valid numerics, but only 2 valid dates**
    - Returns more rows than
      - BETWEEN 200501 AND 200512  **12 valid numerics, and 12 valid dates**
  
- Histogram statistics can represent “pockets” of data
  - Allowing more accurate filtering estimates



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## RUNSTATS Syntax 1

15

### COLGROUP-SPEC:

```
>> |-----|
>> | <-----<
>> | <-column-name->-----COLGROUP-STATS-SPEC-----|>><
>> |-----|
```

### COLGROUP-STATS-SPEC:

```
>> |-----|>><
>> |-FREQVAL-COUNT-integer+---MOST---| | -HISTOGRAM- | -NUMQUANTILES-----|
>> |---LEAST---| |---LEAST---| | -NUMQUANTILES-integer- |
>> |-----|
```

- Histogram
  - 1 to 100 quantiles
  - If less than 100 column values, degrades to Distribution Stats (as now)



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## RUNSTATS Syntax 2

16

### CORRELATION-STATS-SPEC:

```
>> |-----|
>> | -KEYCARD- |
>> |-----|
>> | <-----<
>> | -FREQVAL--NUMCOLS--1---COUNT---10---MOST-----|>><
>> |-----|
>> |-----|
>> |-FREQVAL--NUMCOLS---integer-COUNT---integer+---MOST---|
>> |---LEAST---| |---LEAST---| | -NUMQUANTILES-----|
>> |-----|
>> | -HISTOGRAM- | -NUMCOLS--1-----NUMQUANTILES-----|
>> |-----|
>> | -NUMCOLS--integer--NUMQUANTILES--integer- |
>> |-----|
```

- For index with key columns of mixed order
  - histogram stats can only be collected on the prefix columns with same order.
  - If the specified key columns are of mixed order, a warning message DSNU633 is issued
- Note REORG TABLESPACE and LOAD do **NOT** support HISTOGRAMS.



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL



# Page Range Processing

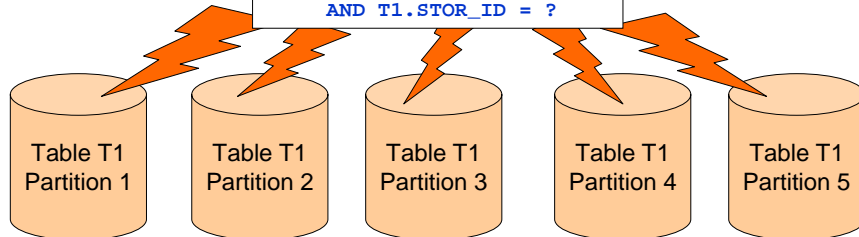
TAKE BACK CONTROL



## Limiting the Partitions Accessed

18

```
SELECT SUM(GROSS_SALES)
FROM T1
WHERE T1.MONTH = ?
AND T1.STOR_ID = ?
```



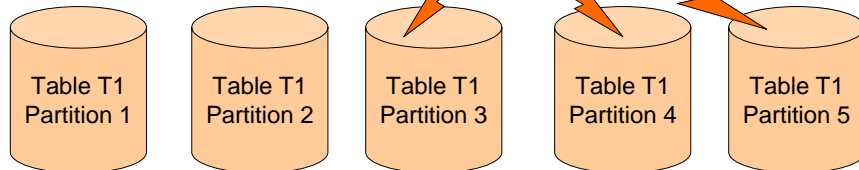
- With DPSIs or tablespace scan of partitioned tablespace
  - it's desirable to avoid accessing partitions with no qualifying rows
- Done using page range screening,
  - if the query has predicates on the leading columns of the partitioning key, DB2 can eliminate partitions



## Page Range Screening Enhancements

19

```
SELECT SUM(GROSS_SALES)
FROM T1
WHERE T1.MONTH = ?
AND T1.STOR_ID = ?
```



- DB2 9 for z/OS introduces two enhancements to the page range screening, resulting in fewer partitions accessed unnecessarily:
  - Join predicates
  - Non-matching predicates



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Page Range Screening with Join Predicates

20

- Using join predicates for page range screening:

```
SELECT * FROM T1, T2
WHERE T1.TRANS_MONTH = T2.TRANS_MONTH
AND T1.CUSTNUM = T2.CUSTNUM
```

The diagram shows the query with two arrows pointing to the join predicates. The arrow for 'T1.TRANS\_MONTH = T2.TRANS\_MONTH' is labeled 'Non-Indexed partition key' and the arrow for 'T1.CUSTNUM = T2.CUSTNUM' is labeled 'DPSI key'.

- V8, all parts of the DPSI index are accessed
  - page range screening only uses local predicates
- V9, only 1 partition of the DPSI index on T2 needs to be accessed for each row from T1
  - join predicate(s) used for page range screening,



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Page Range Screening with Non-matching Predicates

```
SELECT SUM(GROSS_SALES) FROM T1  
WHERE T1.STATE = 'CA'
```

← 2nd partition key

- Assuming:
  - T1 partitioned by **YEAR, STATE**
  - 10 years X 50 states = 500 partitions
  - STATE is not indexed
- V8, page range screening only applies to leading limit key(s)
  - all (500) partitions must be scanned
- V9, since only STATE = 'CA' is required,
  - page range screening can be applied on 2<sup>nd</sup> limit key,
  - only 10 partitions are scanned (10 years X 1 state)



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

IBM

## Global Query Optimization

TAKE BACK CONTROL



## Problem Scenario 1

23

- **V8, Large Non-correlated subquery is materialized\***

```
SELECT * FROM SMALL_TABLE A
WHERE A.C1 IN
      (SELECT B.C1 FROM BIG_TABLE B)
```

- “BIG\_TABLE” is scanned and put into workfile
- “SMALL\_TABLE” is joined with the workfile

\* Assumes subquery is not transformed to join

- **V9 may rewrite non-correlated subquery to correlated**

- Much more efficient if scan / materialisation of BIG\_TABLE was avoided
- Allows matching index access on BIG\_TABLE

```
SELECT * FROM SMALL_TABLE A
WHERE EXISTS
      (SELECT 1 FROM BIG_TABLE B WHERE B.C1 = A.C1)
```



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Problem Scenario 2

24

- **V8, Large outer table scanned rather than using matching index access\***

```
SELECT * FROM BIG_TABLE A
WHERE EXISTS
      (SELECT 1 FROM SMALL_TABLE B WHERE A.C1 = B.C1)
```

- “BIG\_TABLE” is scanned to obtain A.C1 value
- “SMALL\_TABLE” gets matching index access

\* Assumes subquery is not transformed to join

- **V9 may rewrite correlated subquery to non-correlated**

```
SELECT * FROM BIG_TABLE A
WHERE A.C1 IN
      (SELECT B.C1 FROM SMALL_TABLE B)
```

- “SMALL\_TABLE” scanned and put in workfile
- Allows more efficient matching index access on BIG\_TABLE



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Global Optimization Objectives

25

### Improve Query Performance

- Consider both correlated and non-correlated forms of a given query
- Consider the inter-queryblock combinations
- Select the form / combination with the lowest overall estimated cost



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Virtual Tables

26

- A new way to internally represent subqueries

### Virtual Tables

- A virtual table is simply an abstract representation of a subquery
  - Virtual tables may or may not represent a workfile
  - Allows subquery to be considered in different join sequences
- Virtual tables only apply to subqueries that cannot be transformed to joins,
    - or cases where transformation to join would reduce the choices available to Optimizer (This can occur with Updatable Cursors)



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Virtual Tables – further thoughts

27

- V9 increases the number of cases where subqueries are transformed to join
  - Transformation to join usually allows the most number of choices to be considered by the Optimizer
    - Greatly increasing the chances that the Optimizer will select the most efficient access path
- Allows Optimizer to easily rearrange the order and manner in which the subqueries are processed



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## EXPLAIN Output

28

- Additional row for “Virtual Table” when it is materialised to a workfile
  - Table type for materialized virtual tables is "W" for "Workfile".
    - Name includes an indicator of the subquery queryblock number
      - Example → “DSNVT(02)”
  - Non-materialized Virtual tables will not be shown in EXPLAIN output.
- Additional column PARENT\_PLANNO
  - Used with PARENT\_QBLOCKNO (existing column) to connect child queryblock to parent miniplan
  - Since V8 only contains PARENT\_QBNO, it is not possible to distinguish which plan step the child tasks belong to.



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## EXPLAIN – Non-correlated subquery

**SELECT \* FROM T1 WHERE T1.C2 IN  
(SELECT T2.C2 FROM T2, T3 WHERE T2.C1 = T3.C1)**

QBNO	PLAN-NO	METHOD	TNAME	AC-TYPE	MC	AC-NAME	SC-JN	PAR_QB	PAR_PNO	QB_TYPE	TB_TYPE
1	1	0	DSNVT(02)	R	0		N	0	0	SELECT	W
1	2	1	T1	I	1	T1_IX_C2	Y	0	0	SELECT	T
2	1	0	T2	R	0		N	1	1	NCOSUB	T
2	2	1	T3	I	1	T3_X_C1	N	1	1	NCOSUB	T

- QBNO=2, PLANNO=1 & 2 rows both have PARENT\_PLANNO = 1 and PARENT\_QBNO = 1
  - Thus the row corresponding to QBNO=1, PLANNO=1 is the parent row.



## EXPLAIN – Correlated subquery

Using the same query as on the previous slide, but correlating the subquery:

**SELECT \* FROM T1  
WHERE EXISTS  
(SELECT 1 FROM T2, T3  
WHERE T2.C1 = T3.C1 AND T2.C2 = T1.C2)**

QBNO	PLAN-NO	METHOD	TNAME	AC-TYPE	MC	AC-NAME	PAR_QB	PAR_PNO	QB_TYPE	TB_TYPE
1	1	0	T1	R	0		0	0	SELECT	T
2	1	1	T2	I	1	T2_IX_C2	1	1	CORSUB	T
2	2	1	T3	I	1	T3_IX_C1	1	1	CORSUB	T



## Other Considerations

31

- **INSERT, UPDATE and DELETE**
  - Same support as SELECT in V9
    - Removes V8 limitations for subquery to join transformation for non-SELECT.
  
- **Optimization Hints support**
  - Information can be fed into the Optimizer using existing OPHINTS
  - Example
    - User can request that a non-correlated subquery be processed in it's correlated form.
    - Or, that a correlated subquery be processed in its de-correlated form.
  - Provides greater control over how a query is processed, without requiring a change to the way in which the query is coded.



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

IBM

## Generalizing Sparse Index and In-Memory Data Cache

TAKE BACK CONTROL





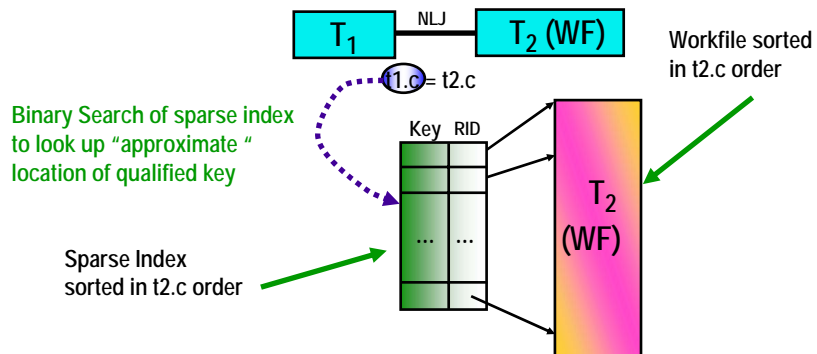
## Pre-V9 Sparse Index & in-memory data cache

- V4 introduced sparse index for non-correlated subquery workfiles
- V7 extended sparse index for the materialized work files within star join
- V8 replaced sparse index with in-memory data caching for star join
  - With runtime fallback to sparse index when enough memory is not available
- **Star schema Sparse indexes**
  - In-memory index occupies up to 240KB
  - Probed through an equal-join predicate
  - Binary search for the target portion of the table
  - Sequential search within the target portion if it is sparse
- **In-memory data caching (also known as in-memory workfile)**
  - Memory pool size controlled by SJMXPOOL zparm
  - Entire workfile is in-memory (and is thus NOT sparse)
  - Searched via binary search (as per sparse index)



## How does Sparse Index work?

- Sparse index may be a subset of workfile (WF)
  - Example, WF may have 10,000 entries
    - Sparse index may have enough space (240K) for 1,000 entries
    - Sparse index is “binary searched” to find target location of search key
    - At most 10 WF entries are scanned



## DB2 V9 Enhancement

35

- In-memory data caching is extended to non-star join
- V9 will use a local pool above the bar
  - Instead of a global pool used in V8 star join
  - Data caching storage management will be associated with each thread
    - Which can reduce the potential storage contention
- New ZPARM MXDTCACH
  - specifies the maximum extent in MB, for data caching per thread.



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Benefit of in memory data caching

36

- In theory, all tables which lack an appropriate index could benefit from sparse index / in-memory data caching :
  - Temporary tables
  - Table expressions
  - Materialized views



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Supporting multi-column Sparse Index

- V8 supports single column only
- V9 supports multi-column Sparse index / in-memory data caching
  - More efficient for > 1 join predicate
- Non-correlated IN subquery with row expression case can benefit from multi-column keys sparse index



## Dynamic Index ANDing for Star Schema



## Introduction

39



- Implementation of a new kind of Star Join methodology
  
- Enhancement consists of:
  - **Pair-Wise Join**
  - **Join Back**
  - **Fall Back plan**
    - If RID overflow occurred
  - **Parallelism support:**
    - Pair-wise Join
    - Join back
  - **Runtime Optimization**



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Some key requirements

40

### Improve and Stabilize Data Warehouse Query Performance

- Provide more predictable query performance
  
- Self-automating and self defending access path
  - Runtime recovery from poor optimizer choice at bind time
  - Good performance with less than perfect statistics
  
- Simplify Index design
  - Better exploitation of single column indexes
  
- More aggressive parallelism
  
- Avoid resource constraints
  - RID pool failures



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## What is Dynamic Index ANDing?

41

- Multi-index access steps are considered independent
- Apply filtering dimensions / snowflakes before fact table
  - Exploiting single and / or multi-column fact table indexes
  - Can be processed concurrently (parallel)
- Runtime determination of filtering
  - Pre-fact dimensions that prove to be poorly filtering can be discarded at runtime and accessed post-fact
- Runtime fallback to workfile for RID processing
  - Avoiding RID pool failures



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Pair-Wise Join with Join Back

42

- Join each dimension table with Fact table through index independently
  - The result of each pair-wise join is a set of Rids of Fact table
- Perform Rid Sort and Rid Merge (ANDing) of the Rid lists
  - Generates the final Fact table rid list
- Final Rid list then used to retrieve data from Fact table
- Join back to dimension table as necessary for obtaining data from dimension tables

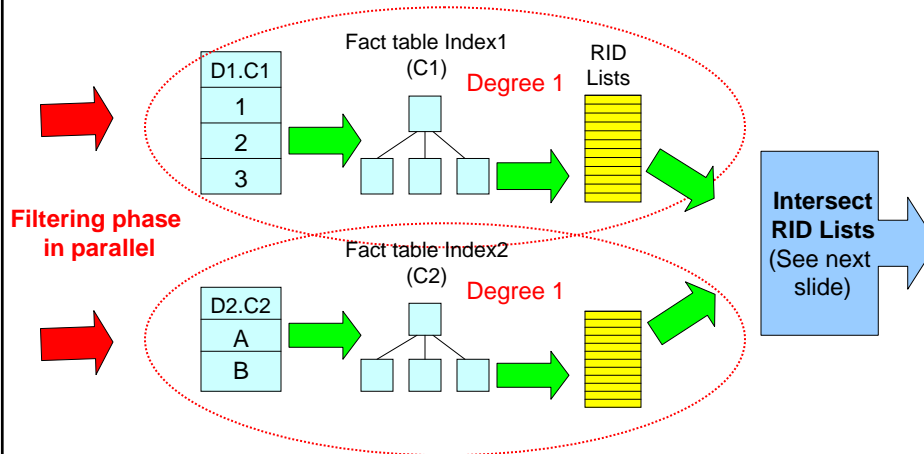


IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

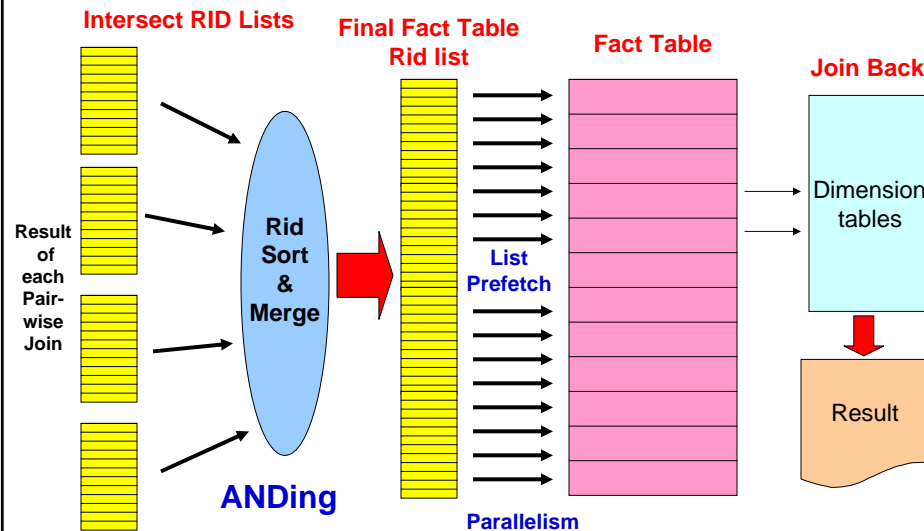
# Solution proposed – Parallel filtering

43



# Intersect Rids, access Fact table and Join Back

44



## RID Pool resource constraint

45

- **Physical constraint**
  - The physical RID pool storage is full
  - No more physical storage space available for storing rids
- **Logical constraint**
  - The RID Map is full
  - No more RID Lists can be created to store rids
- If the RID process hits either of the constraints above, then the fall back plan will write this particular pair of join result rids into a work file.

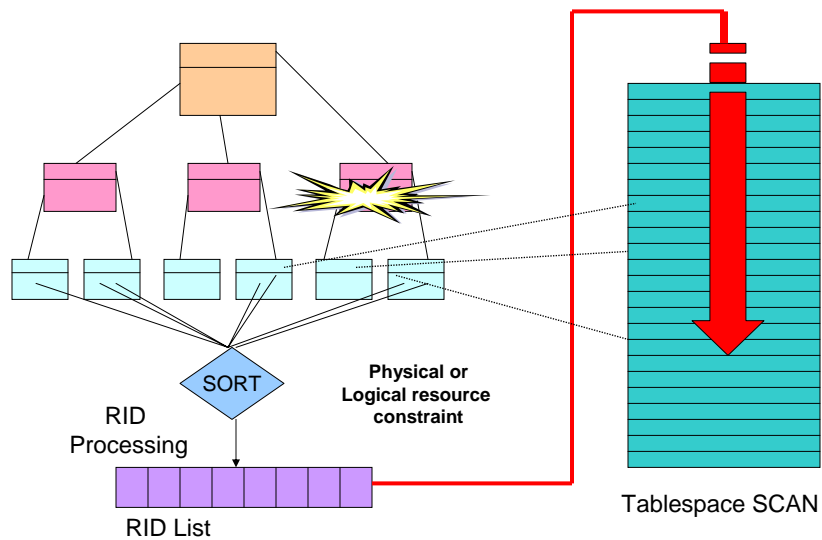


IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## V8 RID Pool failure = TS Scan

46

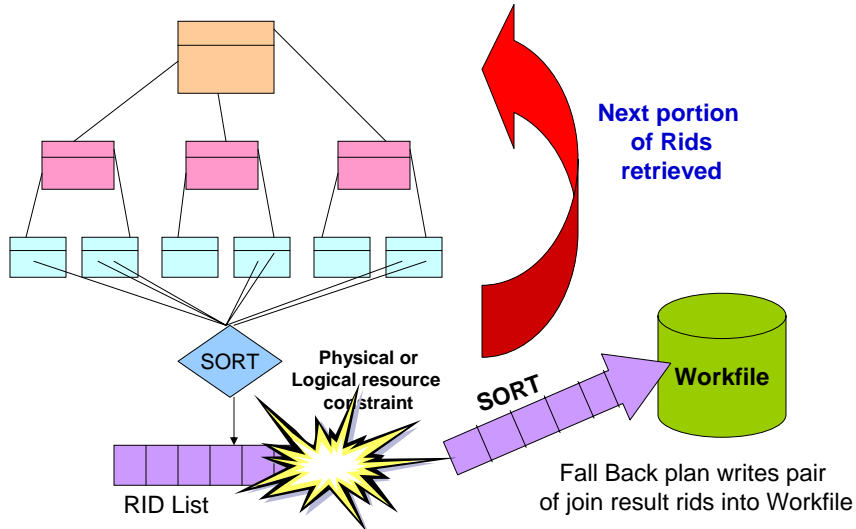


IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## V9 RID Pool Fallback Plan

47



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Parallelism support

48

- Parallelism can apply to both Pair-wise join and Join back phase
  - Primary assumption is the Rid pool resource is always available when running on parallelism
- Parallelism support for the Pair-Wise Join
  - Each pair-wise join leg is executed in parallel
  - Within each pair-wise join leg, parallel degree 1
- Parallelism applies in Join Back Phase
  - Once the pair-wise join process is complete, the min and max RID are stored in the pair-wise join structure
    - Optimizer can use the min and max RID to set the parallel partition key value



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL



## Runtime Optimization

49

- Occasionally due to unavailability of the statistics, the plan built at bind time may not reflect the optimal access plan
- Runtime optimization allows a long running star join query a second chance to re-evaluate the join plan
  - Perhaps to skip a pair join if a sufficient reduction has been obtained.
- **Example – Consider a 3 leg join**
  - If the first 2 leg join result (rid sets) after the index ANDing is small enough (say, less than 1% of rids) and leg 3 is not finished yet
    - Optimizer may decide to abort the leg 3 pair-wise join and continue the join back phase with the result rids of index ANDing of leg 1 and leg 2



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Example of a STAR JOIN Query

50

```
SELECT PRODNAME, SUM(SALES), ...  
FROM F, PROD P, CUST C, TIME T, STORES S  
WHERE F.PID = P.ID  
AND F.TID = T.ID  
AND F.SID = S.ID  
AND T.MONTH IN ('JAN', 'FEB') ~~ 17%  
AND S.LOCATION IN ('SAN JOSE', 'DALLAS') ~~ 2%  
AND P.TYPE IN ('FOOD', 'SODA') ~~ 6%  
GROUP BY ...
```

and the indexes declared on Fact table are:

**IDX1: (PID)**

**IDX2: (TID)**

**IDX3: (SID)**

....



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

# EXPLAIN – New Join type

51

	TABLE	JOIN TYPE	ACCESS TYPE	INDEX ONLY
Pair-Wise Join	T	P	I,R,T	Y / N
	F	P	I	Y
	S	P	I,R,T	Y / N
	F	P	I	Y
	P	P	I,R,T	Y / N
	F	P	I	Y
Join Back	F	P	I (with L prefetch) *	
	T			
	S			
	P			
	C			

\* In miniplan the access type is "B"

Please note – EXPLAIN display not finalised as per this version



# Indexing Enhancements

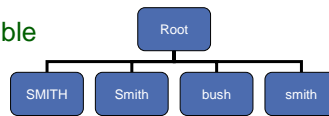


## Stage 2 Predicate Challenge

53

- Consider the following table:

Customer_ID	Integer	Row 1:	Brian	Smith
			i =	
		Row 2000:	John	smith
			i =	
		Row 6000:	Kyle	SMITH
- To search customers with
  - UPPER(Lastname) = 'SMITH'**
    - Predicate is Stage 2
    - Matching index access is not possible



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

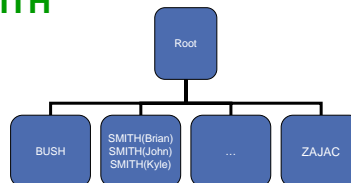
## Stage 2 Predicate Challenge

54

- Solution: Index on Expression:**

```
CREATE INDEX IX_LastName ON CUSTOMER  
( UPPER (Lastname, 'EN_US'));
```

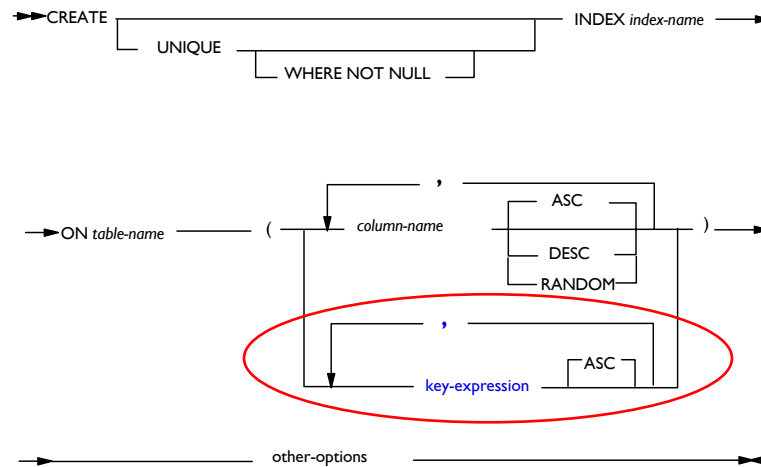
- NOW, to search customers with**
  - UPPER(Lastname) = 'SMITH'**
    - Predicate is Index Matching



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Create Index Syntax



## Index on Expression Examples

- `CREATE INDEX IX1 ON T1`  
– `(HEX(C1), BINARY(LTRIM(C2)))`;
- `CREATE INDEX IX2 ON T1`  
– `(SUBSTR(C2,1,20), CONCAT(C2, C3))`;
- `CREATE INDEX IX3 ON T2`  
– `(SALARY, BONUS/SALARY, BONUS+SALARY)`;
- `CREATE INDEX IX4 ON T2`  
– `(DAYOFYEAR(ENDSHIP) – DAYOFYEAR(STARTSHIP))`;
- `CREATE INDEX IX5 ON T3`  
– `(GRAPHIC (C3))`;



## Index on Expression Limitations

57

- Key expression can NOT be
  - CLUSTERing
  - PARTITION BY
  - DESCending
- Column of which Index on Expression depends cannot be ALTERed
- Resultant encoding scheme must be same as the table
- Not available for
  - Temporary tables
  - Primary, Foreign Keys or Unique constraints



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Index on Expression Limitations

58

- Key Expression must
  - Be scalar (single result)
  - Reference a table column, cannot be a constant
  - Contain at least one column expression
- Key expression must not contain
  - A subquery
  - An aggregate function (MAX, MIN, SUM, AVG etc)
  - A non-deterministic function (RAND())
  - A user-defined or function with external actions
  - A host variable, parameter marker, special register
  - A sequence reference
  - A case expression
  - Repeating expressions
  - LOBS, XML, DECFLOAT data types
  - Reference to SECURITY LABELS, columns with FIELDPROCs



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Index Enhancement - Tracking Usage

59



- Additional indexes require overhead for
  - Utilities
    - REORG, RUNSTATS, LOAD etc
  - Data maintenance
    - INSERT, UPDATE, DELETE
  - Disk storage
  - Optimization time
    - Increases optimizer's choices
- But identifying unused indexes is a difficult task
  - Especially in a dynamic SQL environment



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Tracking Index Usage

60

- Realtime Statistics records the index last used date.
  - SYSINDEXSPACESTATS.LASTUSED
- "Used", as defined by DB2 is the following.
  - As an access path for query or fetch.
  - For searched UPDATE / DELETE SQL statement.
  - As a primary index for referential integrity.
  - To support foreign key access.



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## RTS Update for last used

- RTS is updated once in a 24 hour period
  - if the index is used by DB2, update occurs.
  - If the index was not used, no update.
- RTS service task performs the update at the 1st externalization interval (set by STATSINT) after 12PM.



## Optimization Service Center



## Optimization Service Center (OSC)

- V8 introduced “Intelligent Visual Explain”
  - Externalizing “hidden plan table” optimizer cost details
  - Stats Advisor enhancement recommends when to collect stats
  - Limited to single query only
  
- V9 provides a more extensive “Optimization Service Center”
  - All the features of Visual Explain / Statistics Advisor
  - Single query or workload
  - Plus query monitor
  - Advisors – Statistics, Index and Query Design



## OSC Welcome Page

**WELCOME**

Welcome! To get started with the Optimization Service Center (OSC), you must first configure a connection to a DB2 for z/OS subsystem. Then you can create a new project to tune a problem query or an entire query workload.

<p><b>Configure DB2 Subsystems</b> Add DB2 subsystems, enable OSC and grant EXPLAIN authorizations.</p>	<p><b>View Workloads</b> View the status of all workloads on a subsystem. Open a workload that has already been created, archive workloads.</p>
<p><b>View Query Activity</b> View dynamic and static queries. Sort and filter queries to find potential problems.</p>	<p><b>Tune Workload</b> Use OSC advisors and advanced tools to analyze a query workload for performance improvement.</p>
<p><b>Tune Single Query</b> Use OSC advisors and advanced tools to analyze a query for performance improvement.</p>	<p><b>View Monitor Profiles</b> View the status of all monitor profiles on a subsystem. Create new monitor profiles or open a monitor that has already been created.</p>

For details see sessions  
1641/1642 Automated Query Tuning in DB2 9 for z/OS Part 1 & 2





## OSC - Problem Query Resolution

65

- Problem Query Identification
  - ✓ Snapping queries from various sources
  - ✓ Monitoring performance exceptions
  
- Problem Query Resolution
  - ✓ Statistics Advisor to provide suggestions on statistics collection
  - ✓ Index Advisor to provide suggestions on index design
  - ✓ Query Advisor to provide suggestions on query rewrite
  - ✓ Query format to present a readable query
  - ✓ Annotation of optimizer rewritten query to embed critical information
  - ✓ Query report to show the underlying physical design with critical information
  - ✓ Visual explain to show the access path choice
  - ✓ Visual optimization hints to implement emergency solution
  - ✓ Service SQL to send relevant doc to IBM for diagnosis



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

IBM

## Misc Optimization Enhancements

TAKE BACK CONTROL



## Sort Improvements

67

- Improved Sort avoidance for DISTINCT
  - Previously, GROUP BY had an advantage over DISTINCT
    - GROUP BY could use duplicate index to avoid sort
    - DISTINCT required unique index to avoid sort
  - From V9, DISTINCT can avoid sort using duplicate index
- Reduced workfile usage for very small sorts
  - Final sort step requiring 1 page will NOT allocate workfile



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Sort Improvements (cont.)

68

- Early termination of sort with FETCH FIRST clause
  - Previously,
    - sort would continue to completion
  - From V9,
    - Sort will terminate as soon as FIRST 'n' returned when FETCH FIRST n ROWS ONLY specified



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Sequential Access

69

- Does high clusterratio = clustering?
  - For clustering index
    - High clusterratio = high clustering
  - For non-clustering indexes, high clusteratio means
    - Keys align with clustering
    - OR, Data is sequential but not clustered
  - V9 adds new statistic collected by RUNSTATS
    - DATAREPEATFACTOR helps optimizer differentiate clustering from sequential data pattern



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Sequential Access (cont.)

70

- Sequential prefetch only used for tablespace scan in V9
  - Dynamic prefetch used instead for other access paths
    - Dynamic prefetch tracks sequential access at runtime
    - Sequential prefetch is based upon bind/prepare prediction
      - At runtime, data may not be page sequential



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Parallelism Enhancements

71

- V8 implementation
  - Optimizer chooses the lowest cost “sequential” plan
  - Then determines how to “parallelize” the access path
- V9 implementation
  - Multiple sequential plans will be considered for parallelism
  - Lowest cost, after parallelism, is the winner



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Additional Parallelism Enhancements

72

- V8 implementation
  - Non-star join access paths have parallel degree determination based upon leading table
    - Limitations exist such that parallelism cannot be chosen when
      - leading table is a 1-row table, workfile, or is randomly accessed
- V9 implementation
  - Parallelism degree can cut on non-leading table
    - Beneficial for 1-row table, workfile, or DPSI on fact table
- Parallelism also uses histogram statistics for more even distribution of parallel degrees



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## ORDER BY & FETCH FIRST in subqueries

73

- Query containing an ORDER BY can be wrapped inside additional SQL
- ORDER BY and FETCH FIRST n ROWS ONLY in *subselect* / *fullselect*, provides ability to select the top *n* rows

- Examples:

```
SELECT EMP_ACT.EMPNO, PROJNO
FROM EMP_ACT
WHERE EMP_ACT.EMPNO IN
  (SELECT EMPLOYEE.EMPNO
   FROM EMPLOYEE
   ORDER BY SALARY DESC
   FETCH FIRST 3 ROWS ONLY )
(SELECT * FROM T1 ORDER BY C1)
UNION ALL
(SELECT * FROM T2 ORDER BY C2)
```



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Merge (not really optimization)

74

- MERGE
  - A new SQL DML statement in DB2 for z/OS V9
  - Combine Update and Insert operations into one statement
- SELECT FROM MERGE
  - a SELECT statement
  - Show updated/inserted rows
  - Including DB2 generated values
- SELECT FROM UPDATE/DELETE
  - V8 has SELECT FROM INSERT



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Intersect/Except (also not optimization) <sup>75</sup>

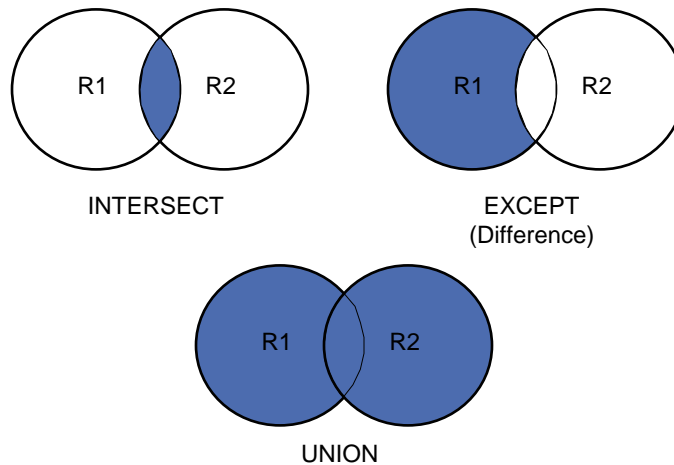
- Add new SQL syntax options
  - INTERSECT/EXCEPT set operators
  - Similar syntax to UNION
- Provides enhanced DB2 family compatibility



IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Intersect/Except/Union semantics <sup>76</sup>



*\*There are some variations and restrictions*

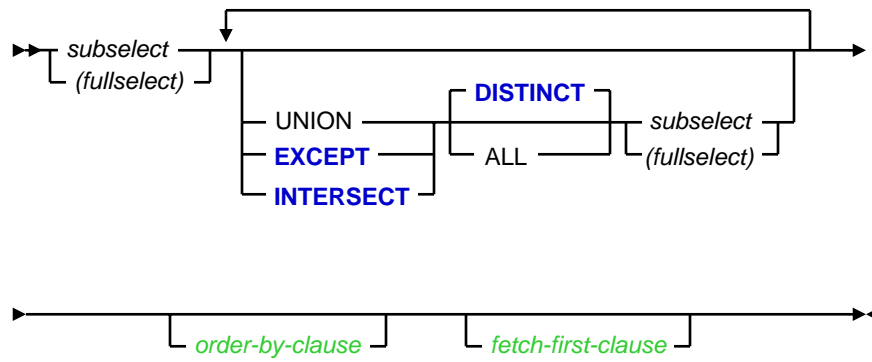


IBM INFORMATION ON DEMAND 2006

TAKE BACK CONTROL

## Intersect/Except Syntax

*fullselect :*



## What's coming from the Optimizer in DB2 9 for z/OS?

Terry Purcell, IBM Silicon Valley Lab

[tpurcel@us.ibm.com](mailto:tpurcel@us.ibm.com)

Session 1819

