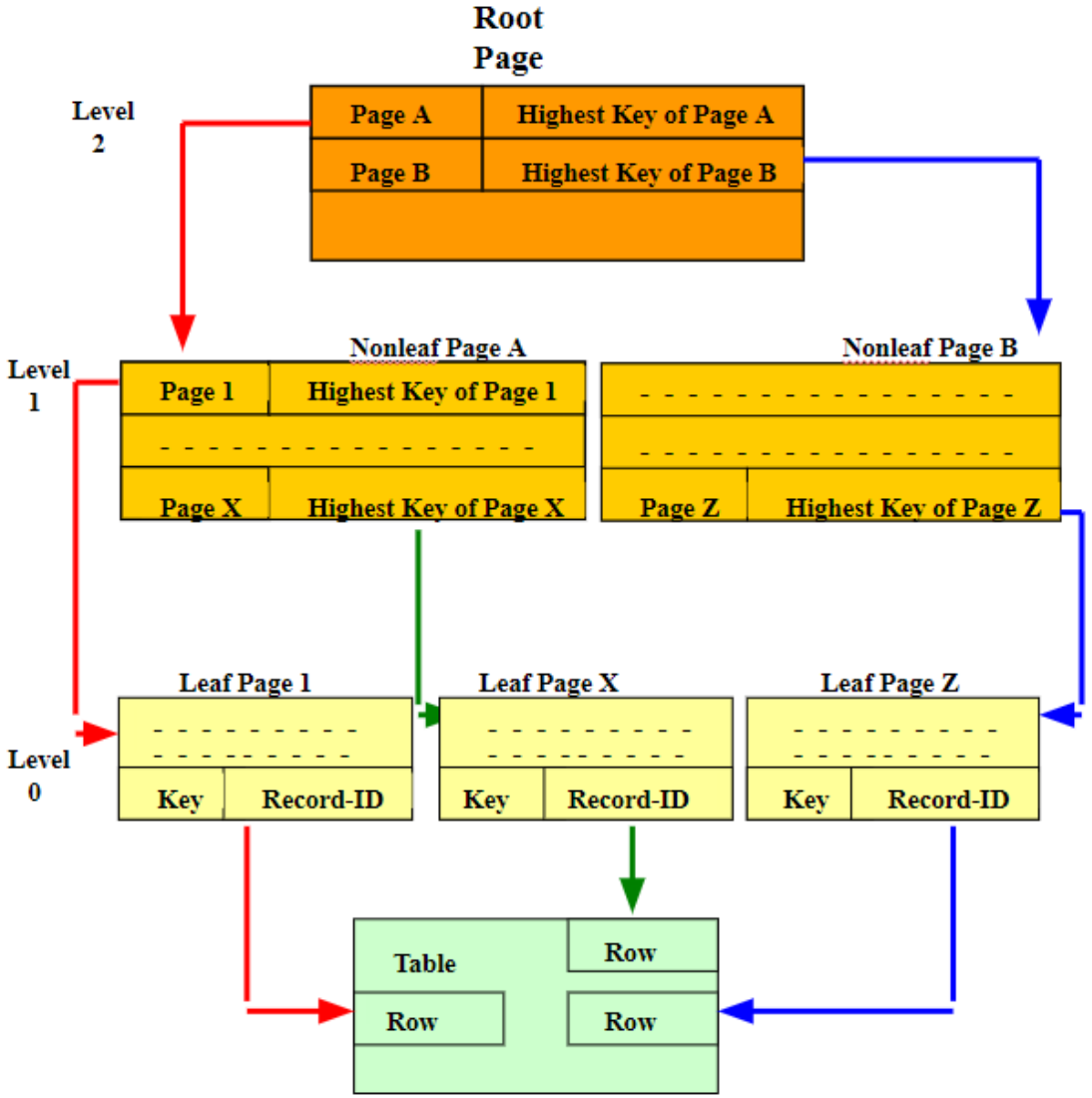
SQL Tuning and Indexes

David Simpson
ProTech / Themis
September 19, 2022
www.protechtraining.com

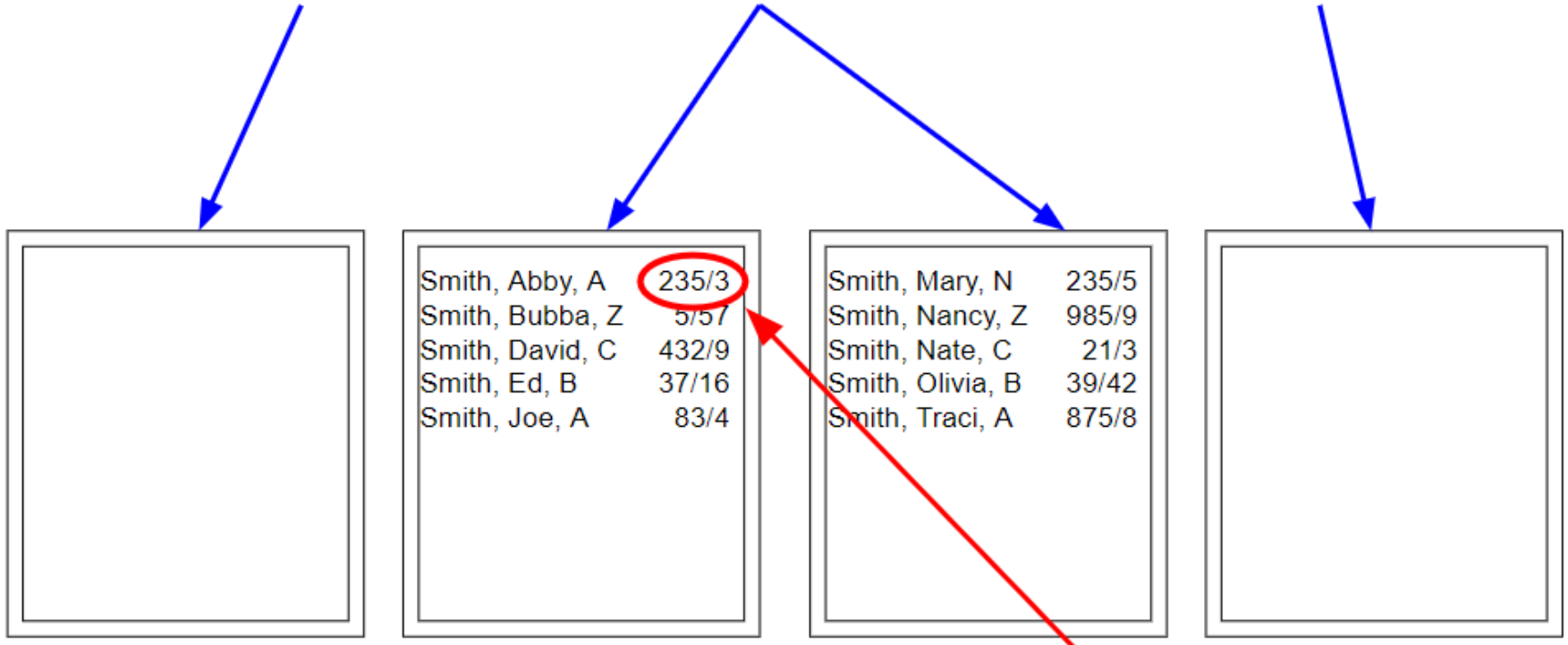
David Simpson

Since 1993 David has worked as a developer and DBA in support of very large transactional and business intelligence systems. David is a certified DB2 DBA on both z/OS and LUW. David was voted Best User Speaker and Best Overall Speaker at IDUG North America 2006. He was also voted Best User Speaker at IDUG Europe 2006 and is a member of the IDUG Speakers Hall of Fame. David is also an IBM Champion.

Index Structure



Index Structure - Leaf Pages



Matching Index Scan

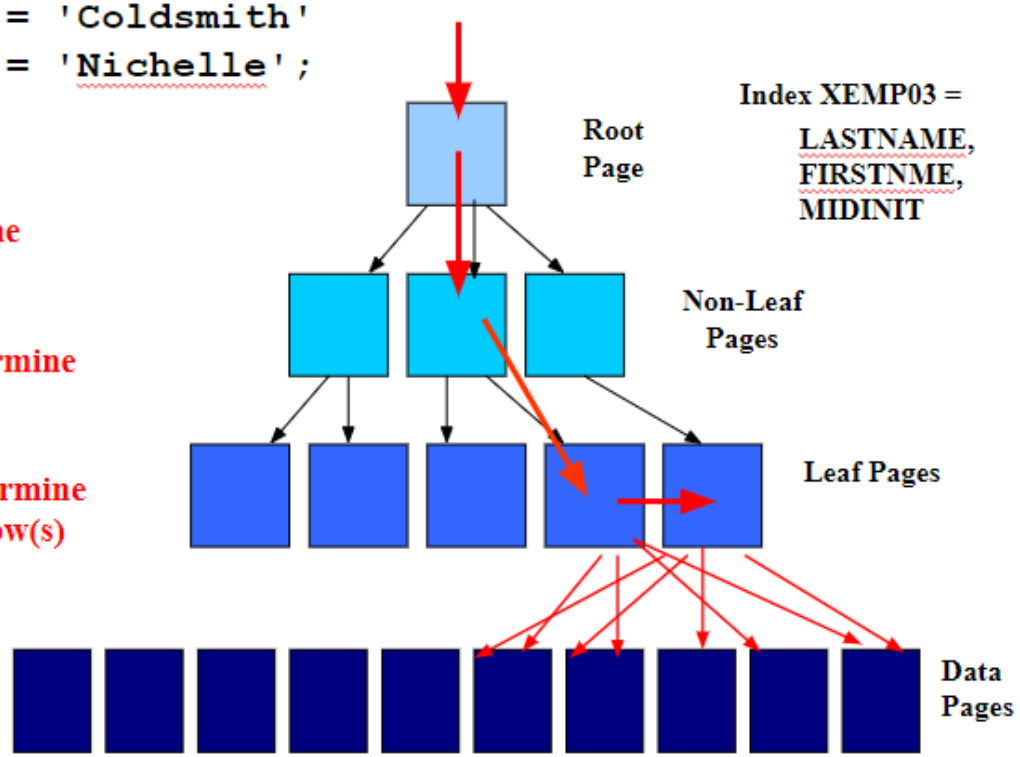
```
SELECT * FROM EMP
 WHERE LASTNAME = 'Coldsmith'
   AND FIRSTNME = 'Nichelle';
```

1) Root page is read to determine corresponding non-leaf page

2) Non-leaf page is read to determine corresponding leaf page

3) Leaf page(s) are read to determine RID of corresponding data row(s)

4) Data Pages are returned



Index Structure is utilized by reading some Index Pages and their corresponding Data Pages

Matching Index Scan

```
SELECT * FROM EMP
 WHERE LASTNAME = 'Coldsmith'
   AND FIRSTNME = 'Nichelle';
```

PLAN_TABLE

PLAN NO	METHOD	TNAME	ACCESS TYPE	MATCH COLS	ACCESS NAME	INDEX ONLY	PREFETCH
1	0	EMP	I	2	XEMP03	N	

What About This?

```
SELECT * FROM EMP
 WHERE LASTNAME = 'Coldsmith'
   AND MIDINIT = 'R';
```

INDEX XEMP03 on (LASTNAME, FIRSTNME, MIDINIT)

Index Screening Predicate

PLAN_TABLE

PLAN NO	METHOD	TNAME	ACCESS TYPE	MATCH COLS	ACCESS NAME	INDEX ONLY	PREFETCH
1	0	EMP	I	1	XEMP03	N	

Non-matching Index Scan

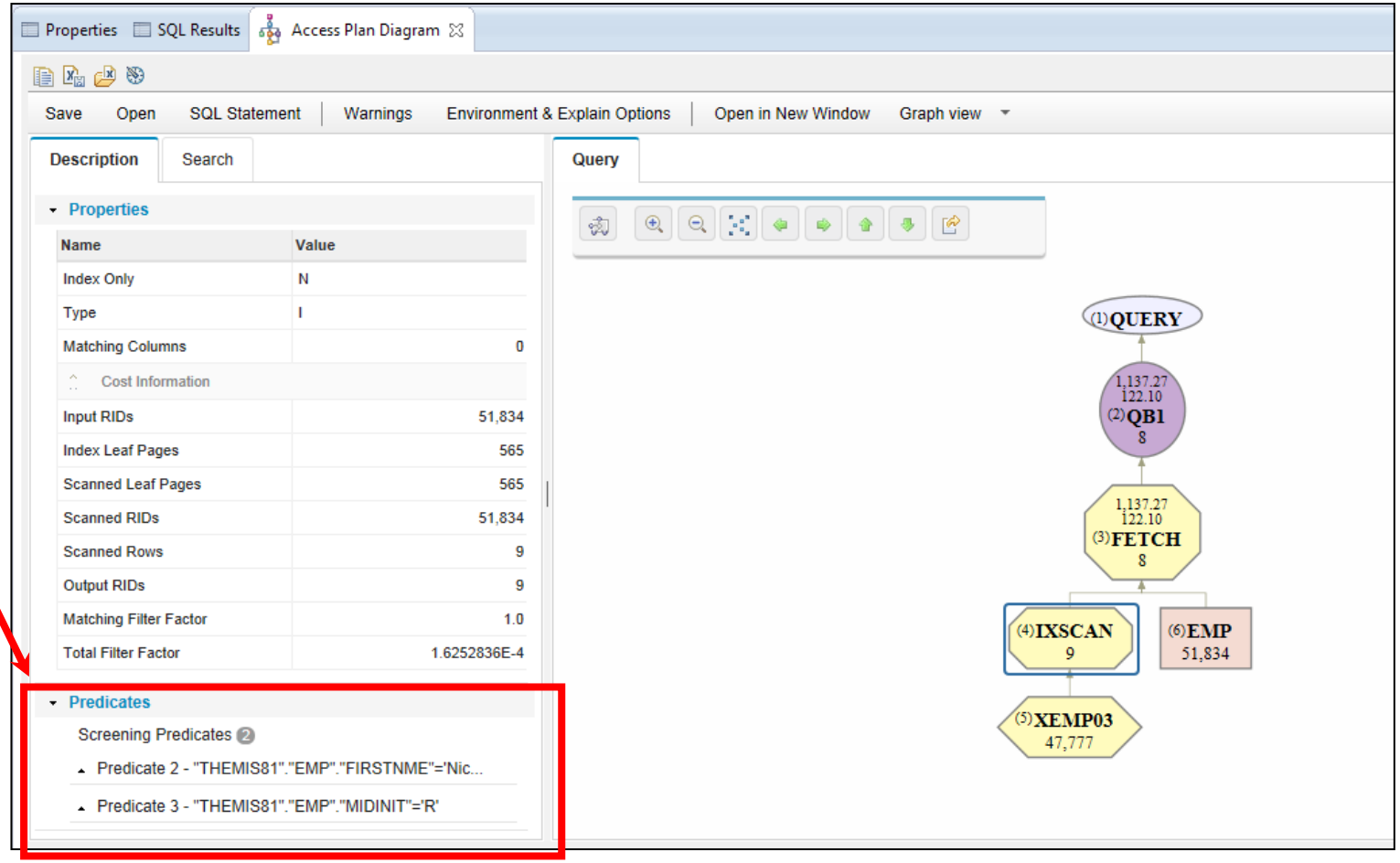
```
SELECT * FROM EMP
 WHERE FIRSTNME = 'Nichelle'
   AND MIDINIT = 'R';
```

PLAN_TABLE

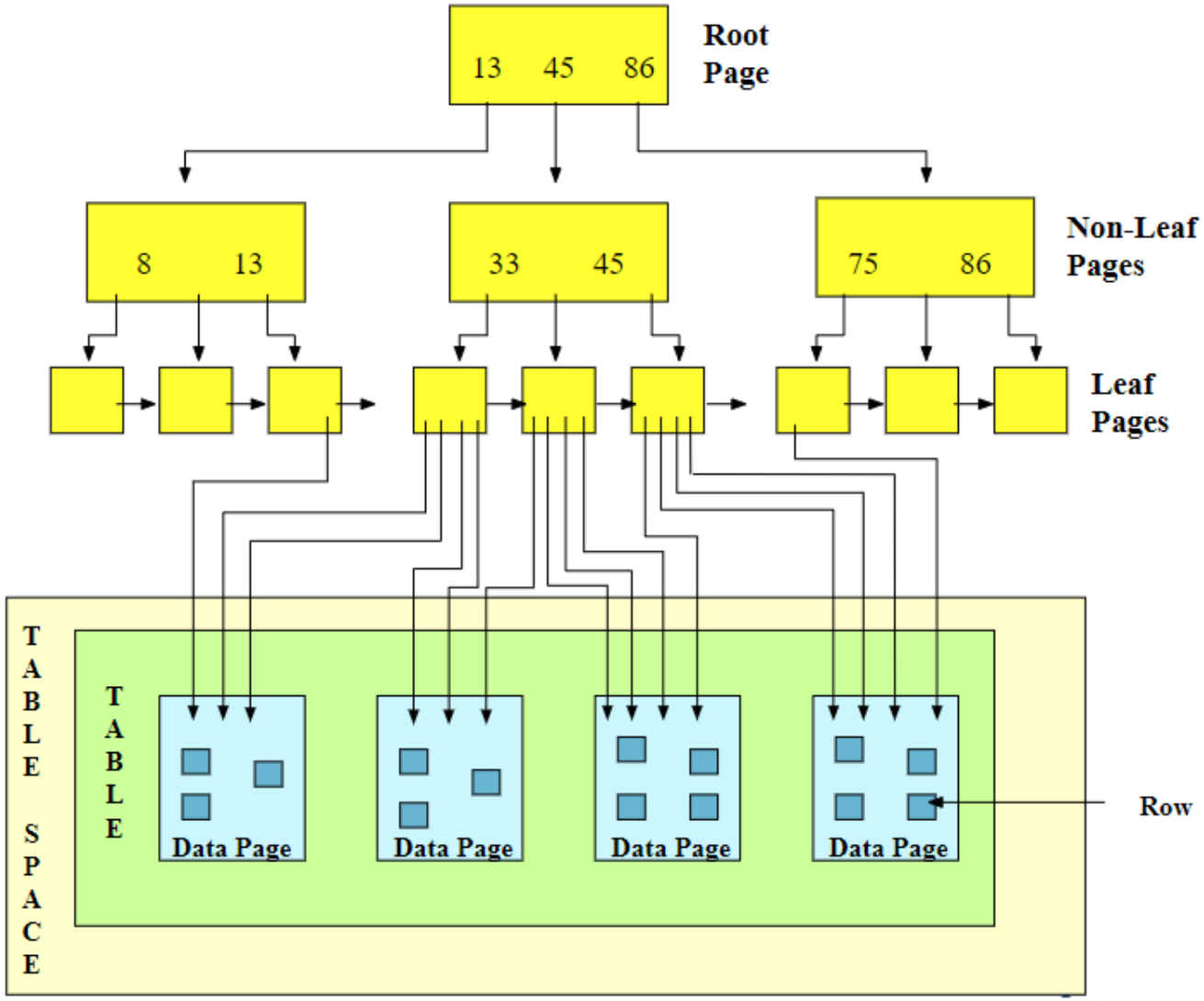
PLAN NO	METHOD	TNAME	ACCESS TYPE	MATCH COLS	ACCESS NAME	INDEX ONLY	PREFETCH
1	0	EMP	I	0	XEMP03	N	

Non-matching Index Scan

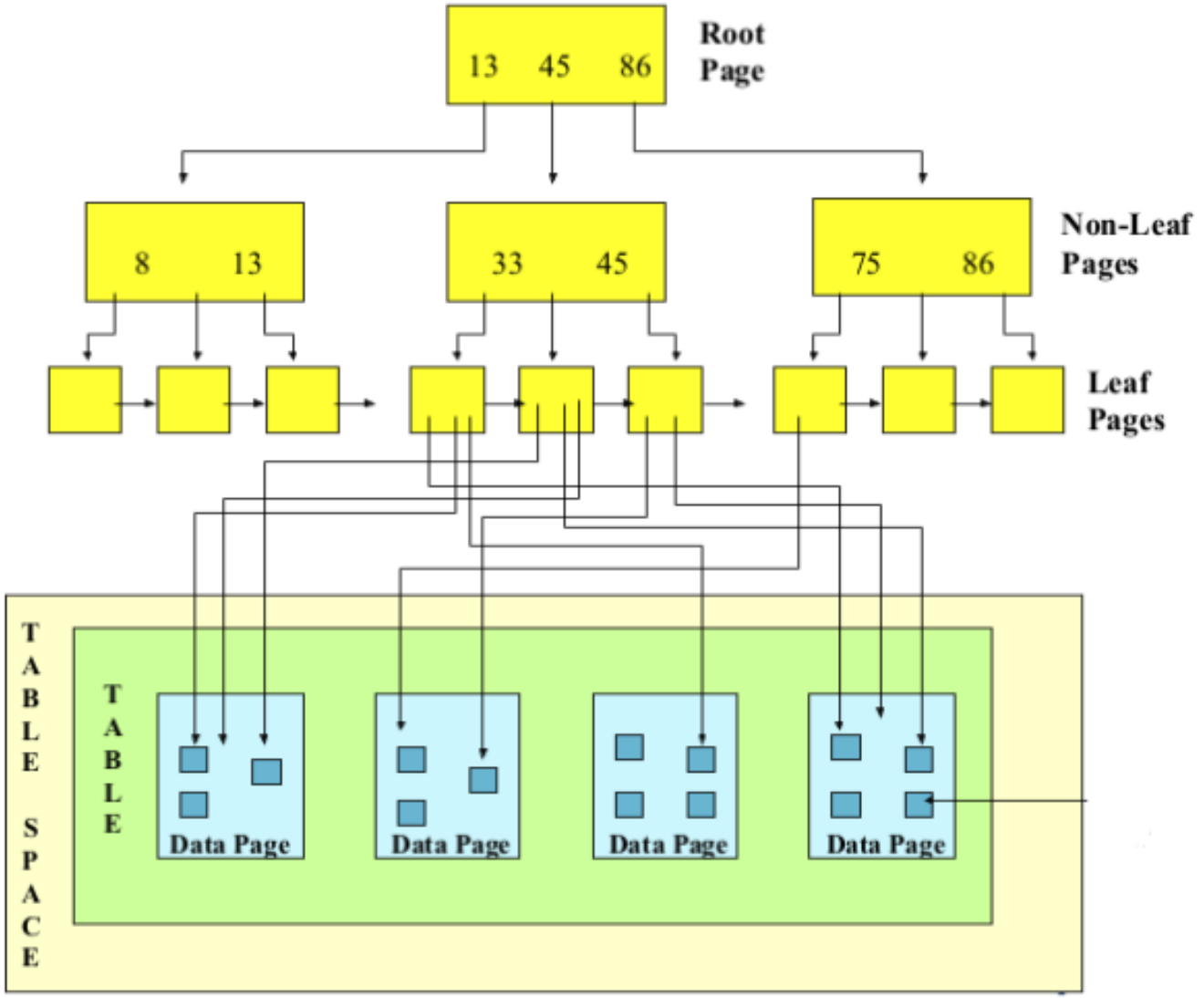
```
SELECT * FROM EMP
 WHERE FIRSTNME = 'Nichelle'
   AND MIDINIT = 'R';
```



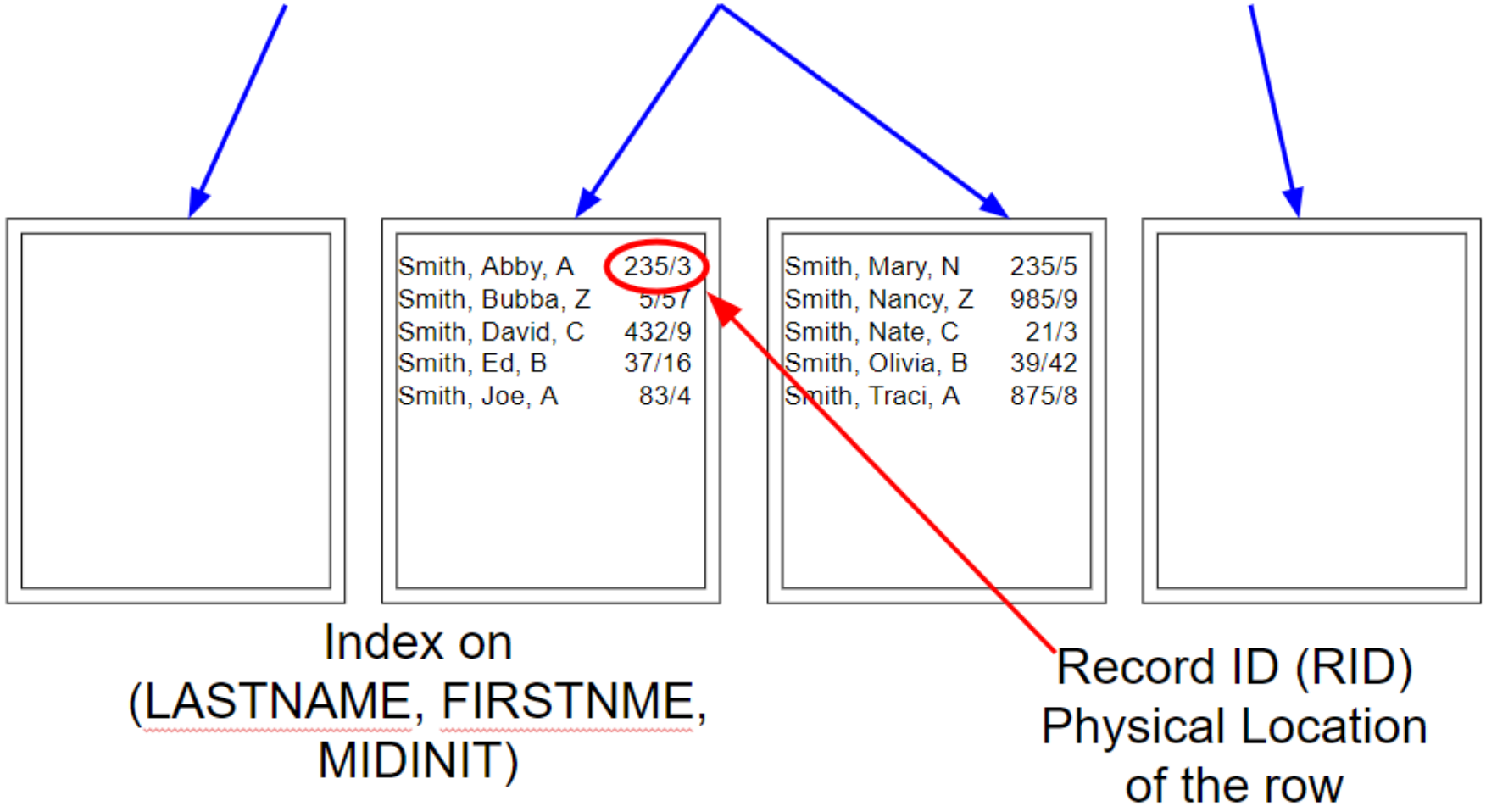
Clustered Index



Non-Clustered Index

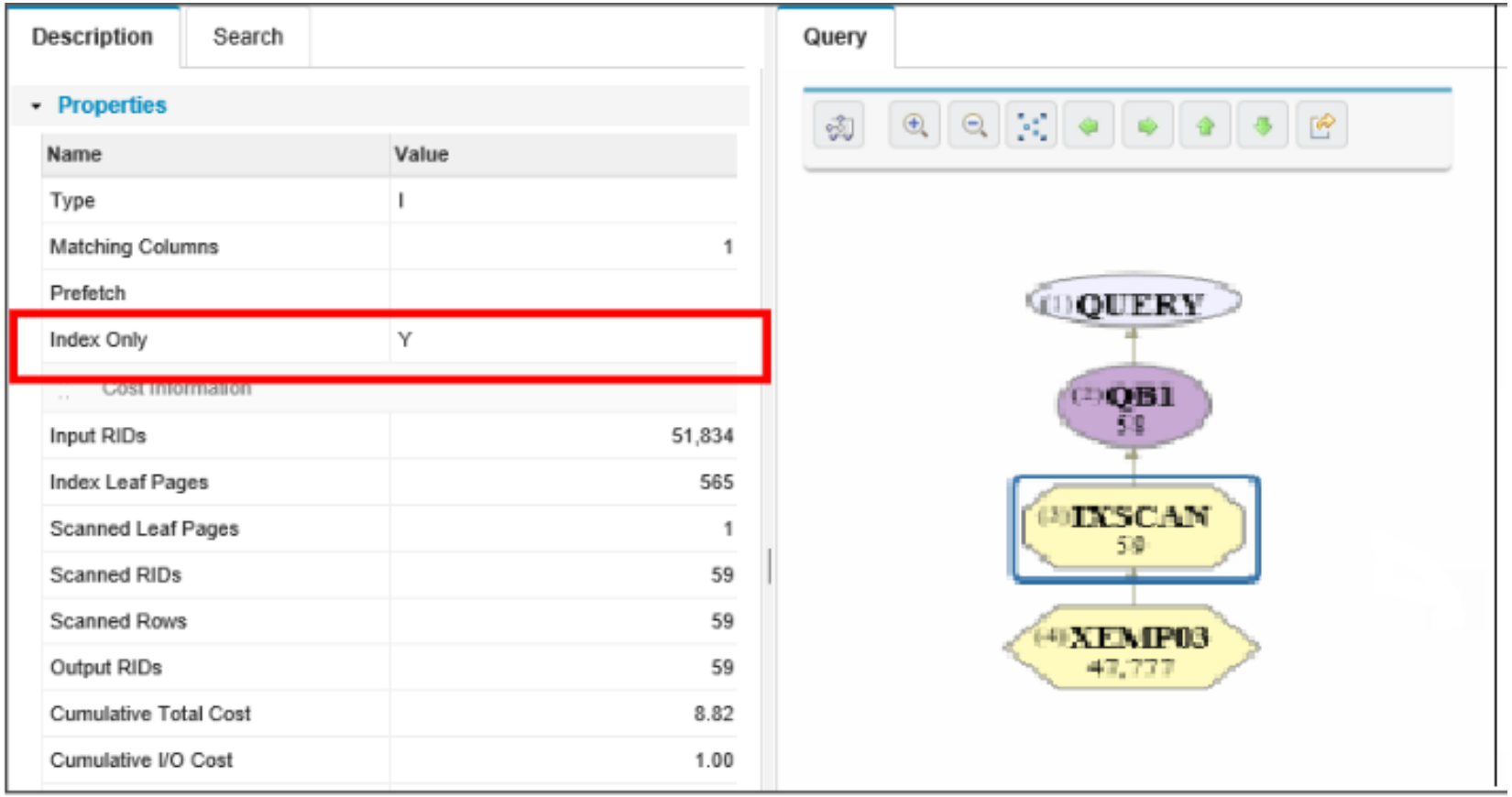


Is it Clustered?



Index Only Access

```
SELECT      LASTNAME,
            FIRSTNME,
            MIDINIT
  FROM EMP
WHERE LASTNAME LIKE 'Jo%'
```



List Prefetch

```
SELECT *
  FROM EMP
 WHERE LASTNAME LIKE 'S%';
```

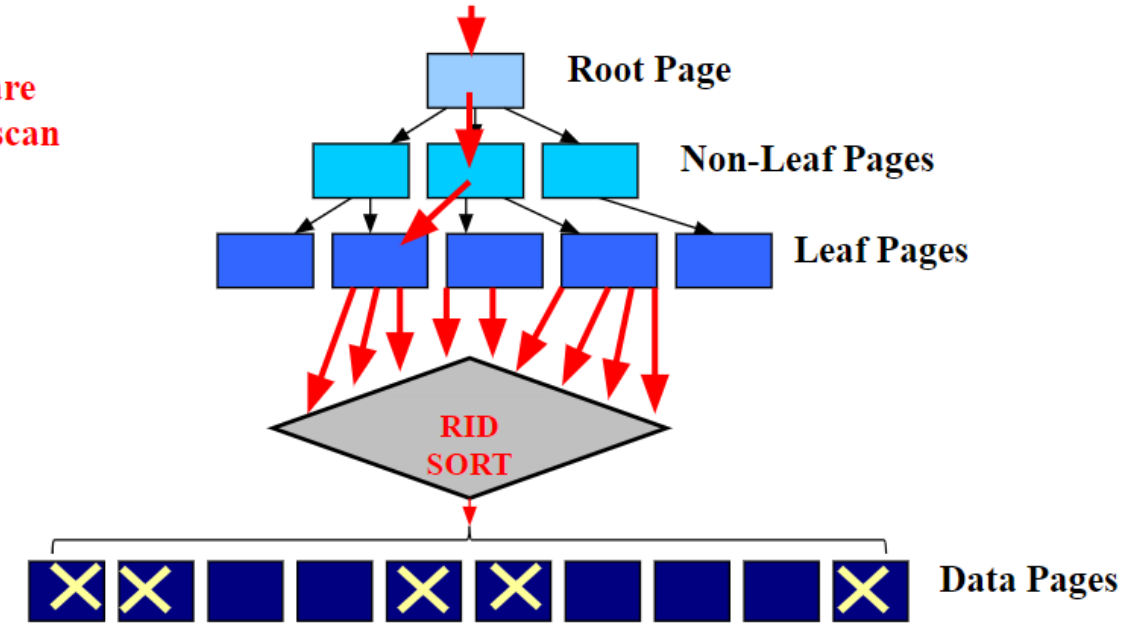
Index XEMP03 =
(LASTNAME, FIRSTNME, MIDINIT)

PLAN NO	METHOD	TNAME	ACCESS TYPE	MATCH COLS	ACCESS NAME	INDEX ONLY	PREFETCH
1	0	EMP	I	1	XEMP02	N	L

1) A list of RIDS for data pages are accessed by a matching index scan

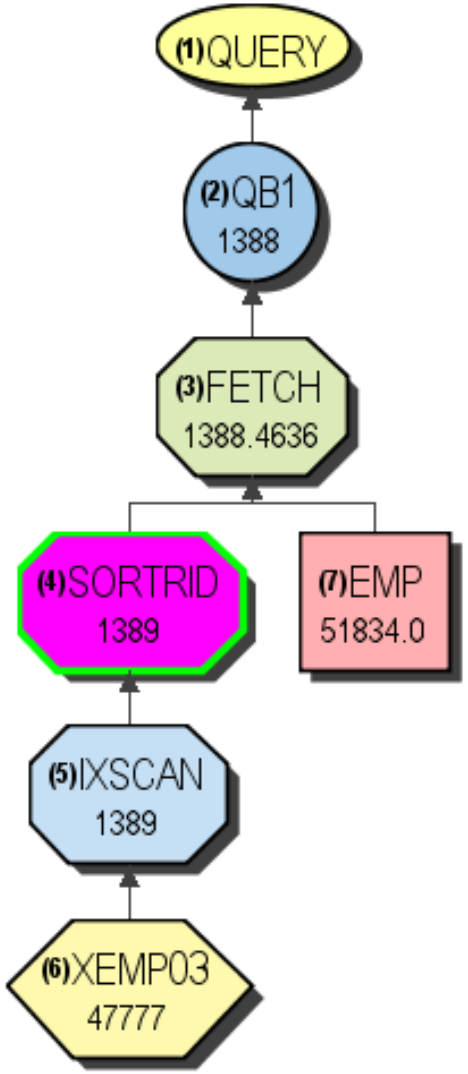
2) Rid list is sorted in ascending sequence by data page number

3) Data Pages are prefetched in order of the sorted RID list



List Prefetch

Input Cardinality	1389	
Scanned Rows	1389	
Stage 1 Returned Rows	1388.4639	
Stage 2 Returned Rows	1388.4639	
Output Cardinality	1388.4639	
Cumulative Total Cost	793.7297	
Cumulative IO Cost	216.9514	
Cumulative CPU Cost	5622297	
Stage 1 Columns	14	
Page Range		
Prefetch	L	



The Db2 Optimizer

Catalog Statistics

Object Definitions

SQL

Access Path

Access Path Hint

System Configuration
(Available memory, CPUs, etc)

Case Study #1

```
SELECT    ...
  FROM T1 JOIN T2
    ON T1.C1 = T2.C1
   AND T1.C2 = T2.C2
 WHERE T1.C4 = 'Something'
   AND T1.C6 > 27
```

Only available interesting index on T2 is a single column index on C1.

Table T2 has 2.4 Billion Rows
Column Cardinality (COLCARDF) Statistics on T2:
C1 = 250
C2 = 10,000,000

btw… cast these as BIGINT when querying SYSCOLUMNS or you get floating point decimals that are hard to read.

Case Study #2

```
SELECT    C3
  FROM T1
 WHERE T1.C4 = 'Something'
   AND T1.C6 > 27
```

Only available interesting index on T1 is a composite index on C4, C6 and is NOT clustered.

Table T1 has 1 Billion Rows
The 2 predicates shown narrow the result to about 200,000 rows.

Case Study #3

```
SELECT ...
  FROM T1
 WHERE T1.C4 = 'Something'
   AND T1.C6 > 27
```

Only available interesting index on T1 is a composite index on C6, C4

Table T1 has 1 Billion Rows

Case Study #4

```
SELECT ...
  FROM T1
 WHERE T1.C4 = 'Something'
   AND T1.C6 <> 27
```

Only available interesting index on T1 is a composite index on C6, C4

Table T1 has 1 Billion Rows

Case Study #5

```
SELECT ...
  FROM T1
 WHERE T1.C4 = 'Something'
   AND T1.C6 = 27
```

In this case let's assume 2 separate indexes on T1:

IX1 is a single column index on C4
IX2 is a single column index on C6

How will optimizer decide which index to choose?

What are the access path options?

Case Study #5 (cont)

```
SELECT ...
  FROM T1
 WHERE T1.C4 = 'Something'
   AND T1.C6 = ?
```

How is this different?

Will the access path always be the same as the previous example?

Think about advantages / disadvantages of parameterized queries…

Thank You for Attending!

"There is always time for an Explain"

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

Thanks and last questions?

Thanks very much for your time and I hope you found this course valuable. **Still have questions after the presentation?**

Contact: support@protechtraining.com

One of our instructors will answer your technical question soon!