

# SQL Tuning and Indexes

---

David Simpson  
ProTech / Themis  
September 19, 2022  
[www.protechtraining.com](http://www.protechtraining.com)



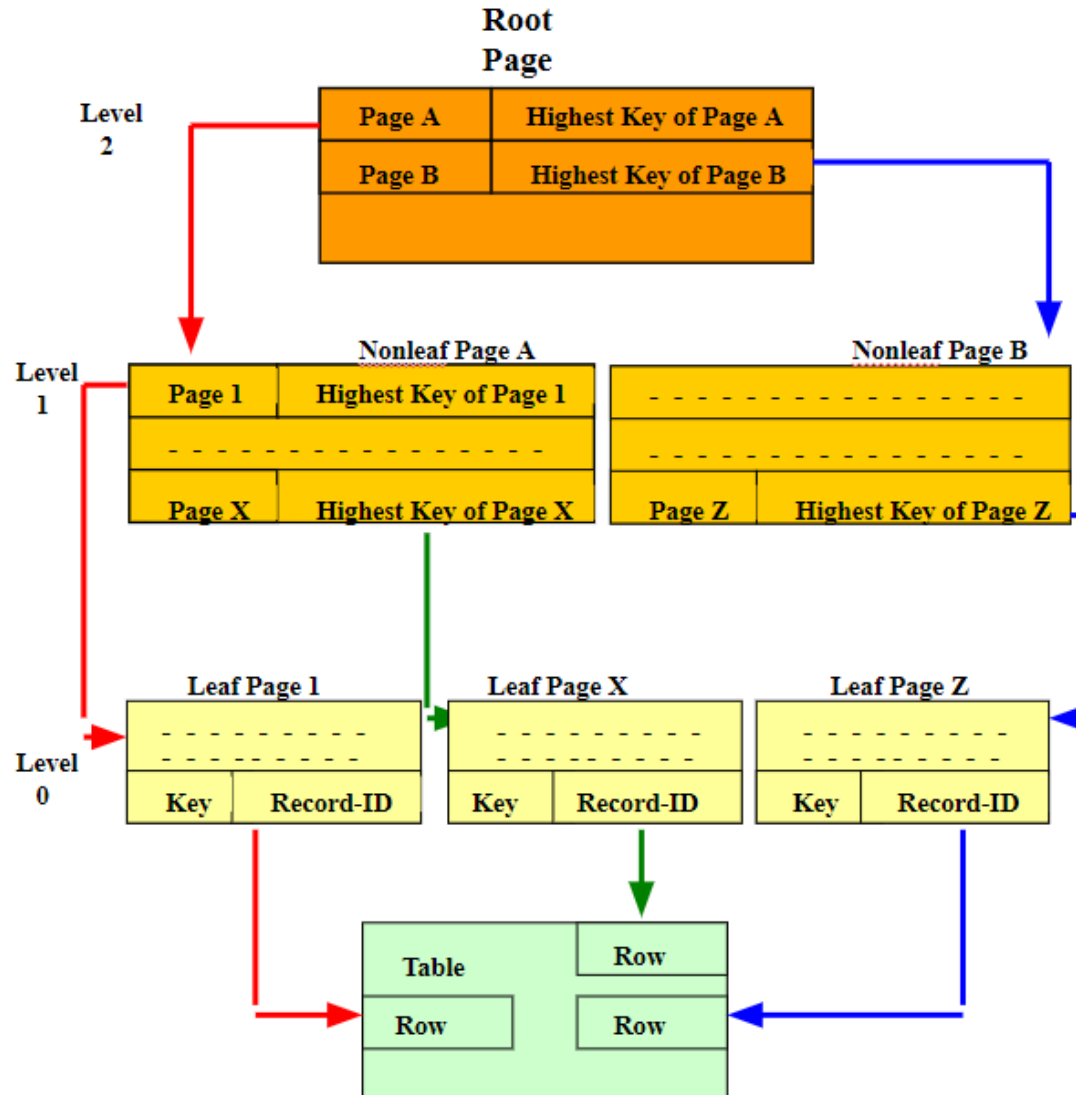
# David Simpson

---

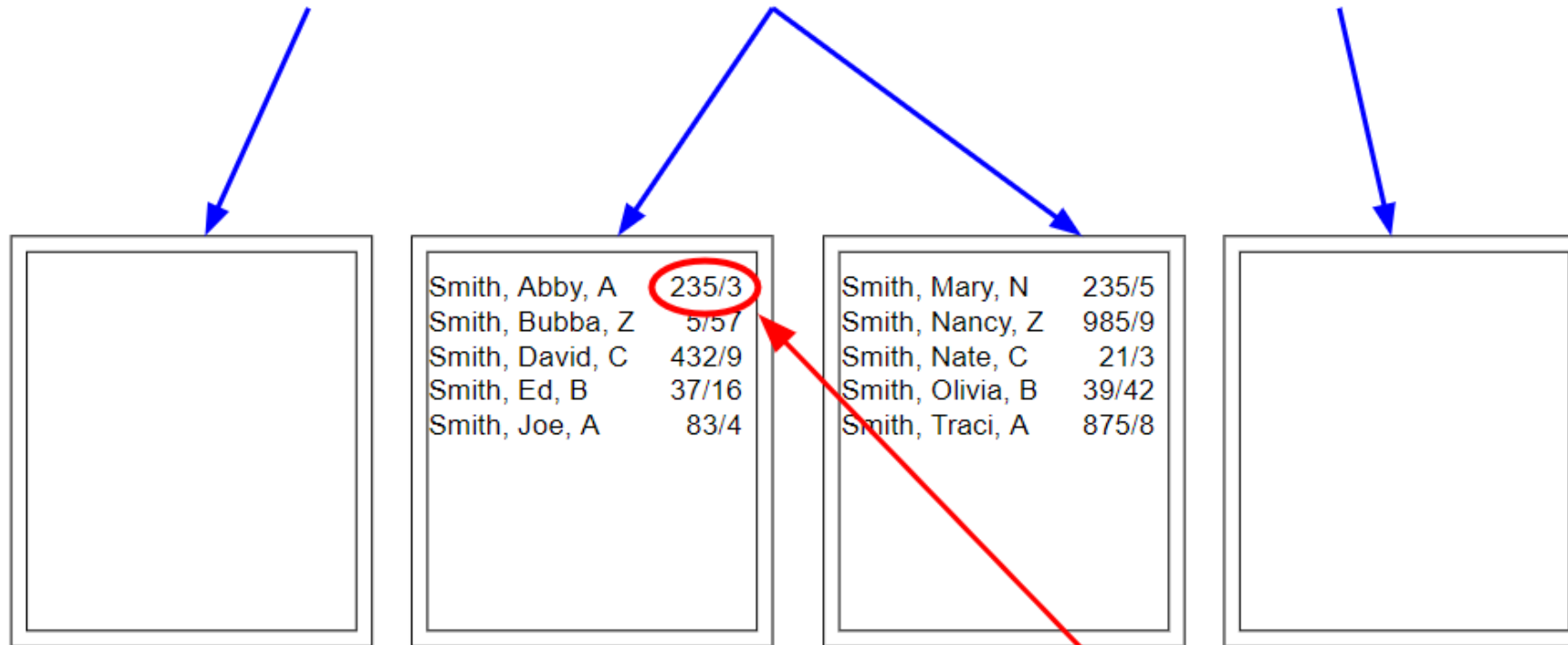


Since 1993 David has worked as a developer and DBA in support of very large transactional and business intelligence systems. David is a certified DB2 DBA on both z/OS and LUW. David was voted Best User Speaker and Best Overall Speaker at IDUG North America 2006. He was also voted Best User Speaker at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



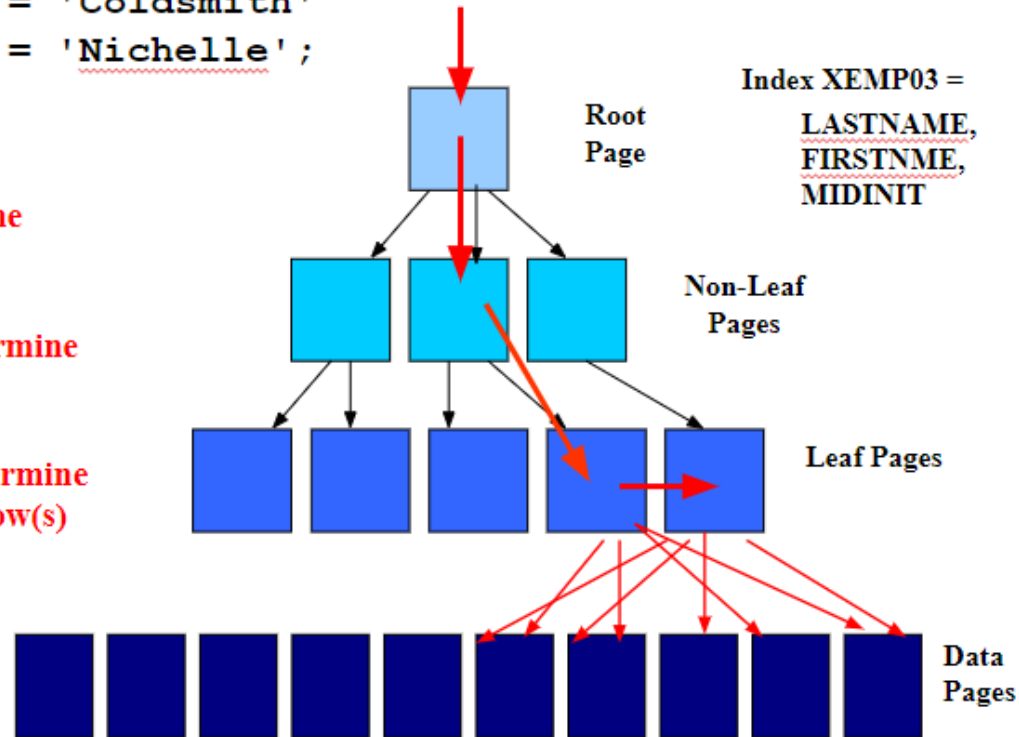
Index on  
(LASTNAME, FIRSTNAME,  
MIDINIT)

Record ID (RID)  
Physical Location  
of the row

# 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

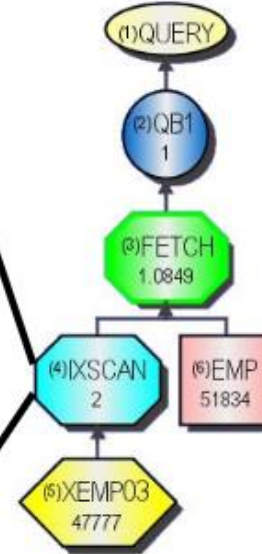


```
SELECT * FROM EMP
WHERE LASTNAME = 'Coldsmith'
AND FIRSTNME = 'Nichelle';
```

## PLAN\_TABLE

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

Name	Value
Input RIDs	51834
Index Leaf Pages	548
Matching Predicates	Filter Factor
THEMIS93.EMP.LASTNAME='Coldsmith'	0.001
THEMIS93.EMP.FIRSTNME='Nichelle'	0.0057
Scanned Leaf Pages	1
Output RIDs	2
Total Filter Factor	2.0930573E-5
Matching Columns	2



# 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';
```

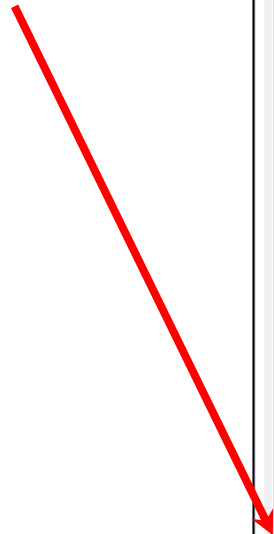
## PLAN\_TABLE

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';
```



Access Plan Diagram

Save Open SQL Statement Warnings Environment & Explain Options Open in New Window Graph view

Description Search

Query

Name	Value
Index Only	N
Type	I
Matching Columns	0
Cost Information	
Input RIDs	51,834
Index Leaf Pages	565
Scanned Leaf Pages	565
Scanned RIDs	51,834
Scanned Rows	9
Output RIDs	9
Matching Filter Factor	1.0
Total Filter Factor	1.6252836E-4

Predicates

Screening Predicates 2

- ▲ Predicate 2 - "THEMIS81"."EMP"."FIRSTNME"='Nic...
- ▲ Predicate 3 - "THEMIS81"."EMP"."MIDINIT"='R'

(1) QUERY  
1,137.27  
122.10

(2) QBI  
8

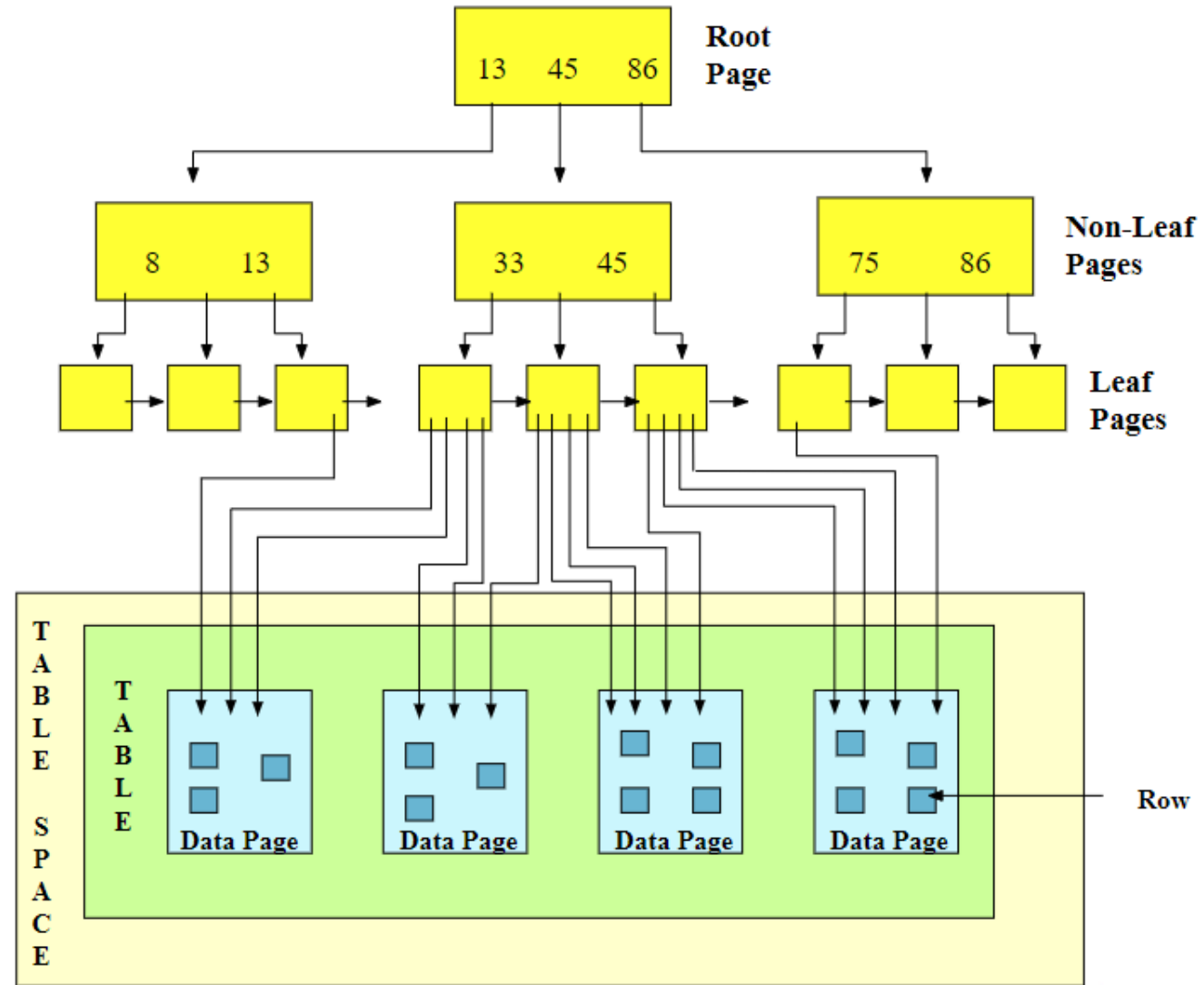
(3) FETCH  
8  
1,137.27  
122.10

(4) IXSCAN  
9

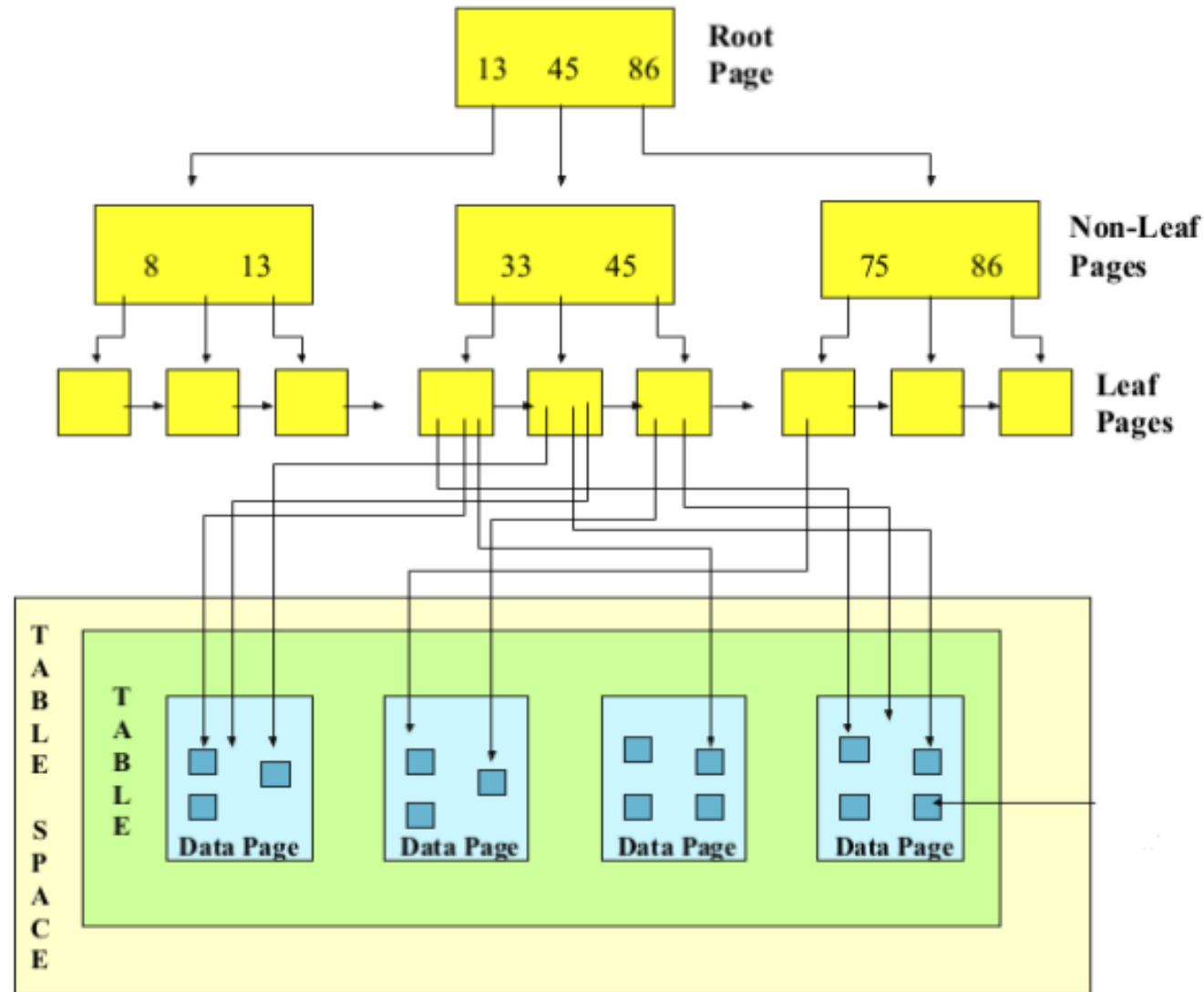
(5) XEMP03  
47,777

(6) EMP  
51,834

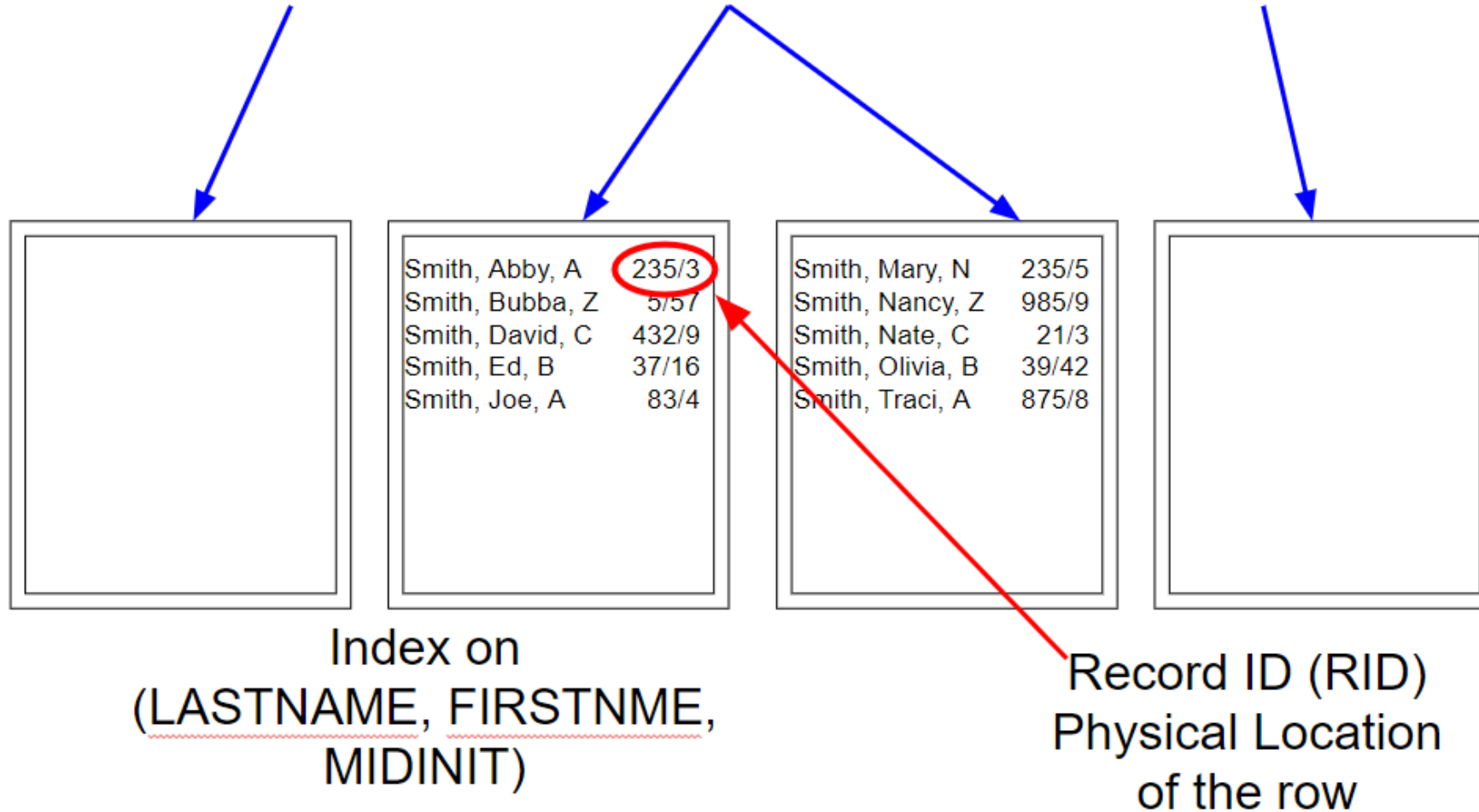
# Clustered Index



# Non-Clustered Index



# Is it Clustered?



# Index Only Access

```

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

Description		Search
<div style="background-color: #f0f0f0; padding: 2px;"> <b>Properties</b> </div>		
Name	Value	
Type	I	
Matching Columns	1	
Prefetch		
Index Only	Y	
... Cost information		
Input RIDs	51,834	
Index Leaf Pages	565	
Scanned Leaf Pages	1	
Scanned RIDs	59	
Scanned Rows	59	
Output RIDs	59	
Cumulative Total Cost	8.82	
Cumulative I/O Cost	1.00	

Query

🔄 🔍 🔍 🌐 👉 👈 👆 👇 📄

```

graph TD
    Q1((1) QUERY) --> QBI2((2) QBI 59)
    QBI2 --> IXSCAN3{{(3) IXSCAN 59}}
    IXSCAN3 --> XEMP034{{(4) XEMP03 47,777}}
    style IXSCAN3 stroke:#0000FF,stroke-width:2px
    
```

# List Prefetch



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

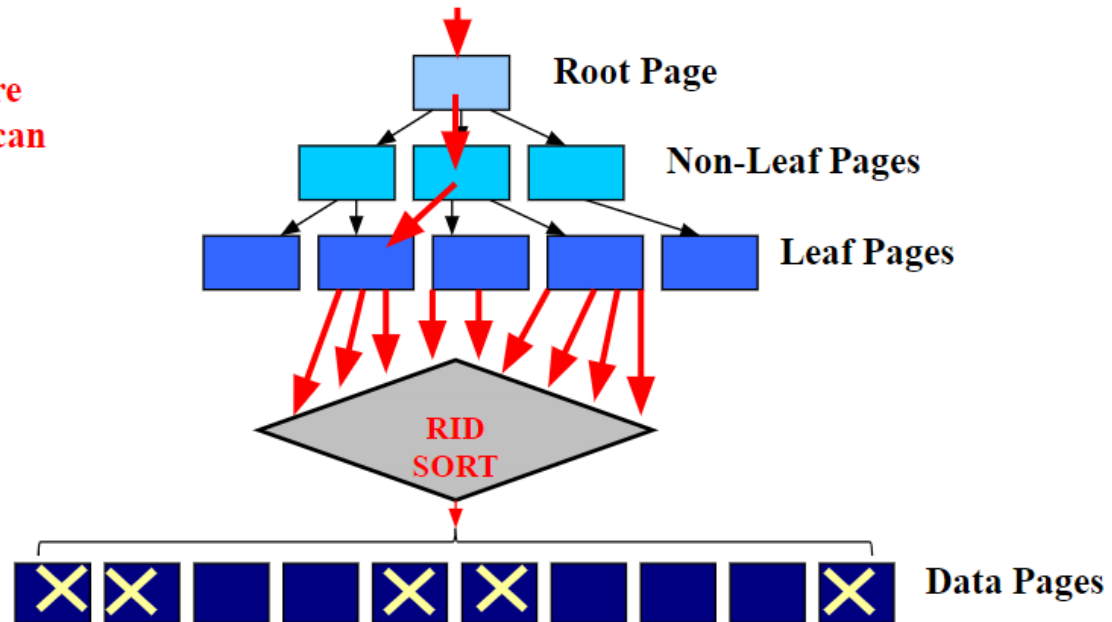
Index XEMP03 =  
(LASTNAME, FIRSTNAME, MIDINIT)

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

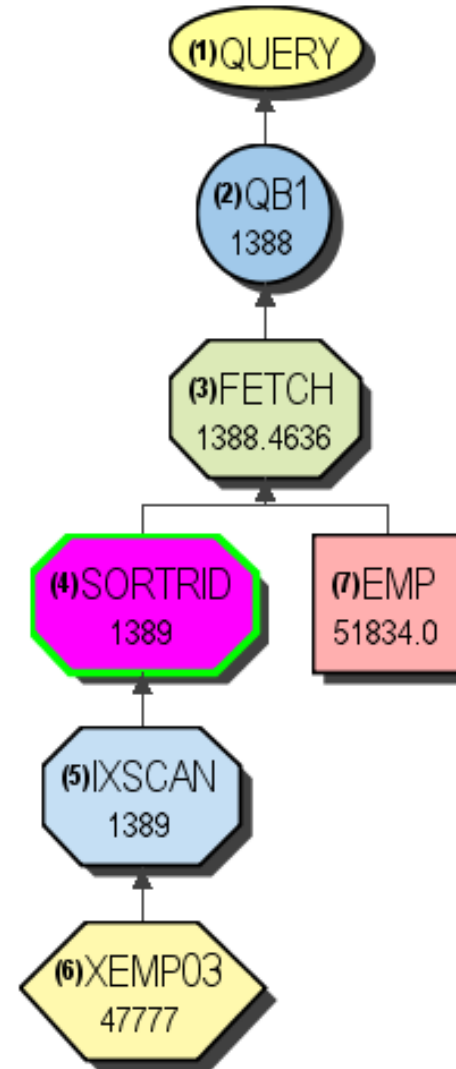
2) Rid list is sorted in ascending sequence by data page number

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

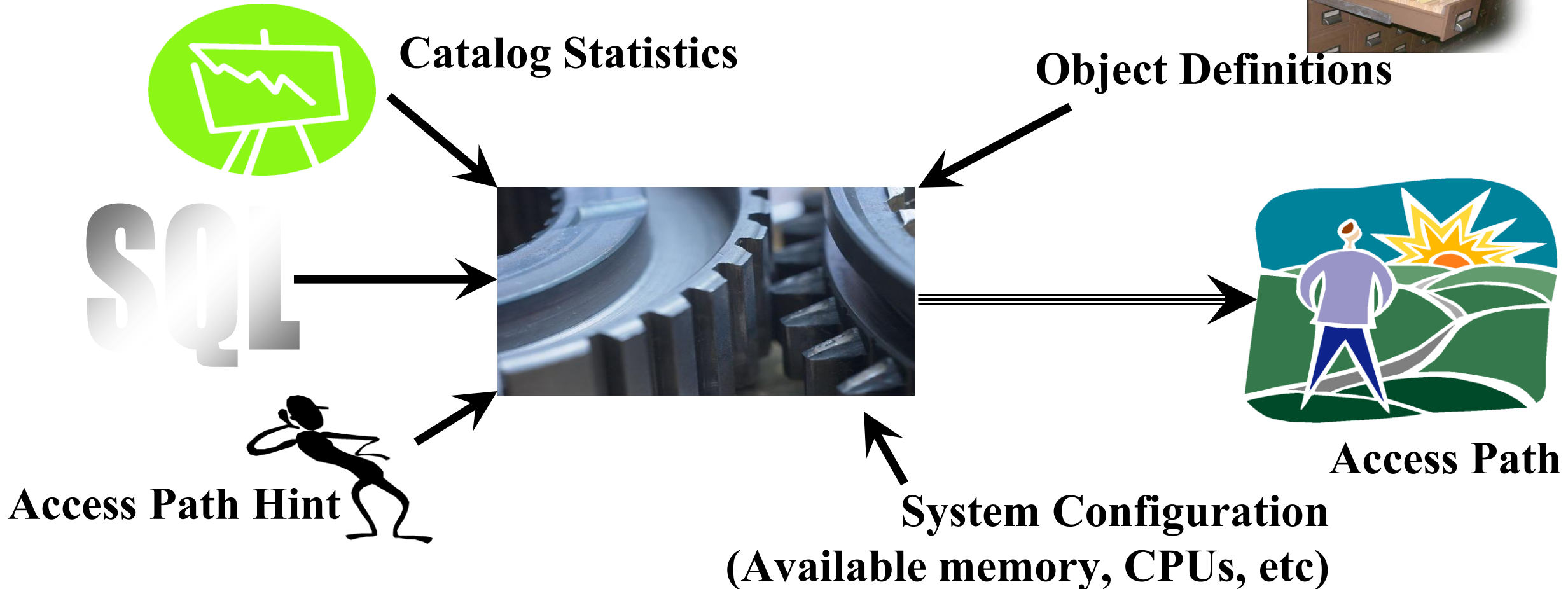


# List Prefetch

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





# 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 (COLCARD) 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](mailto:support@protechtraining.com)

One of our instructors will answer your technical question soon!