Designing Indexes to Improve Query Performance: Part 2

Gail Shaw
TECHNICAL LEAD

@SQLintheWild   http://sqlinthewild.co.za

Overview

Introducing nonclustered indexes

Common query predicates

Indexing for equality

Indexing for inequality

Indexing for ORs

Indexing for joins

Include columns

Filtered indexes

Temp Icons! Replace in editing

Indexing for Joins

Nested loop joins benefit from an index on the inner table

Merge joins may benefit from indexes to provide necessary ordering

Hash joins don’t benefit from indexes

“The aforementioned company names and logos are neither owned or affiliated with Pluralsight, LLC, but their respective owners. They are shown here for illustrative purposes only.”

Demo

Indexing for joins

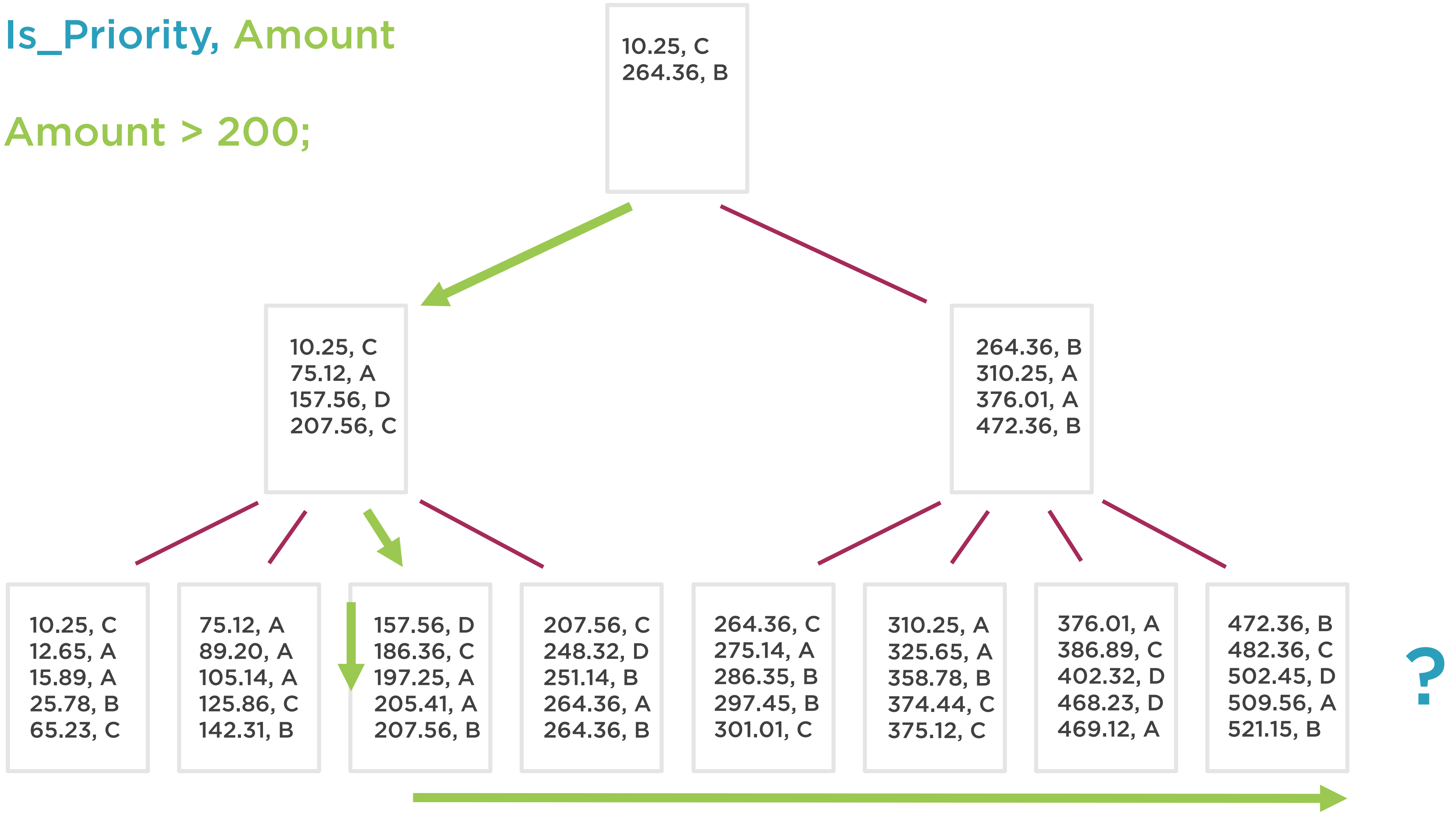
Index Include Columns

Additional columns at the leaf level of the index

Used to avoid expensive key lookups

Include Columns in an Index

```
SELECT Is_Priority, Amount
...
WHERE Amount > 200;
```



Key Lookups

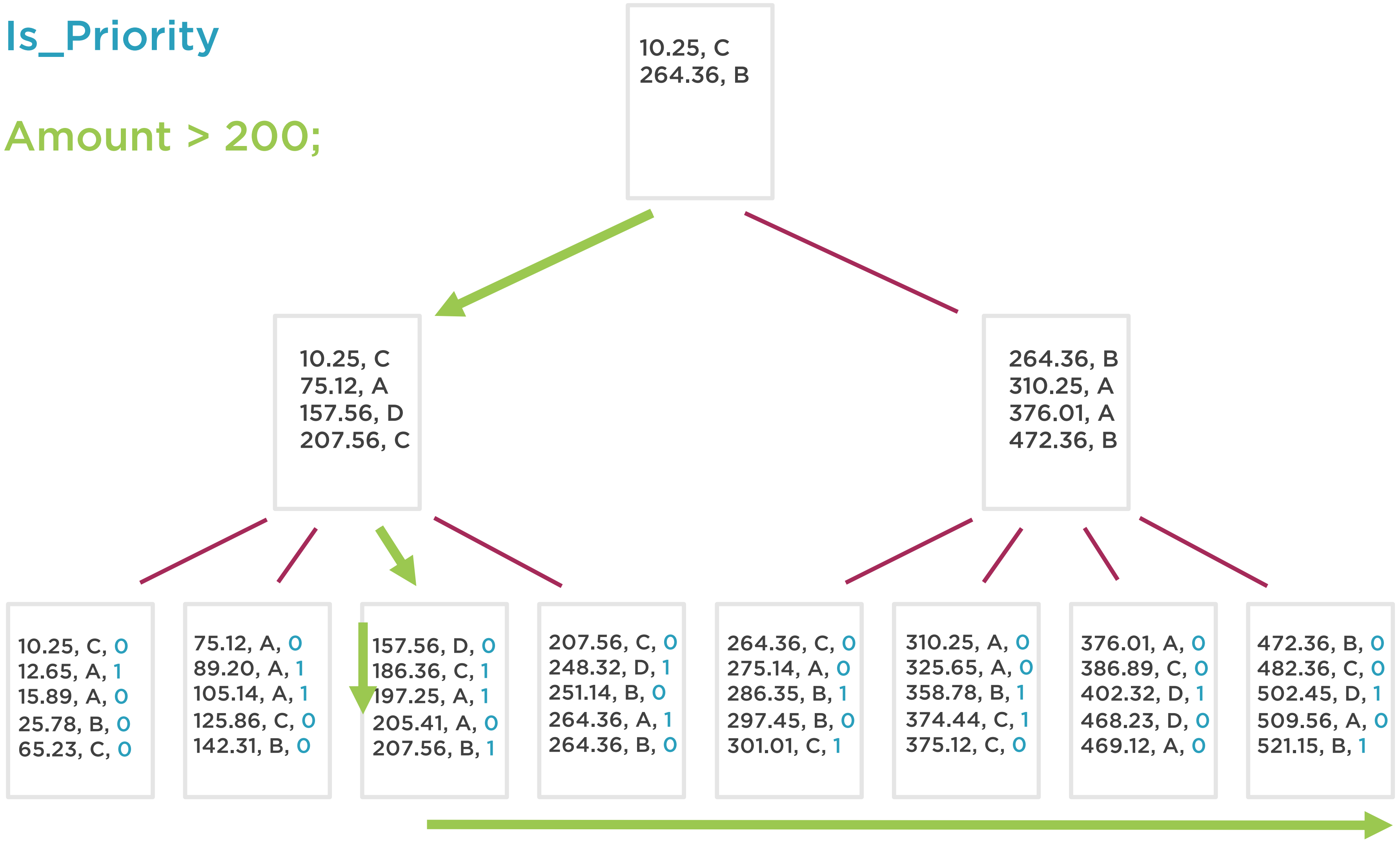
Single-row seek against the clustered index

Fetch columns which are required, but not in the key of the index used

Slow if there are a large number

Include Columns in an Index

```
SELECT Is_Priority
...
WHERE Amount > 200;
```



Demo

Include columns

Filtered Indexes

Indexes on a subset of rows in the table

Can be useful on tables with skewed data

Also useful for complex unique constraints

Don’t work with parameterised queries

Demo

Filtered indexes

How Many Nonclustered Indexes?

✓ As many as you need for the workload

✗ And no more

Summary

Introducing nonclustered indexes

Indexing for equality

Indexing for inequalitry

Indexing for ORs

Indexing for joins

Include columns

Filtered indexes