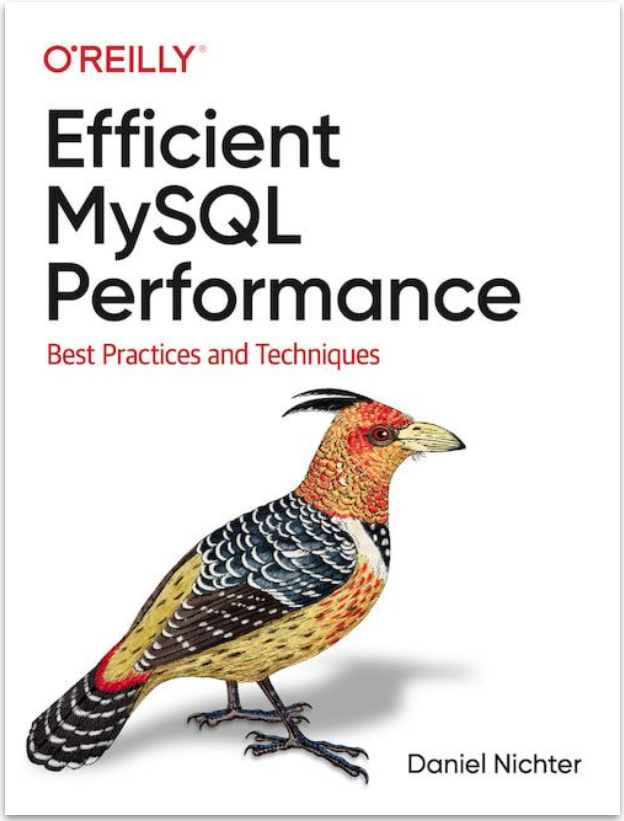
MySQL Performance for Application Developers

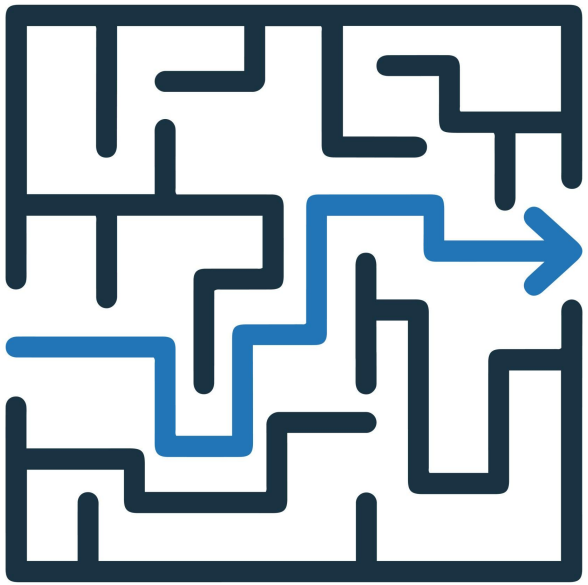
Daniel Nichter @ MySQL and HeatWave Summit 2025

About Me

- 21 years with MySQL
- hackmysql.com
- Open source tools
- Percona Toolkit
- *Efficient MySQL Performance*



An Efficient Path Through the Maze of Performance Topics



Efficient Path

1. Query Response Time
2. Indexes and Indexing
3. Data Storage and Access
4. Sharding
5. Server Metrics
6. Replication
7. Transactions and Data Locking
8. Cloud Performance

~~Efficient Path~~ A 30-minute Pointer to Dereference Later

1. **Query Response Time**
2. **Indexes and Indexing**
3. **Data Storage and Access**
4. Sharding
5. Server Metrics
6. Replication
7. Transactions and Data Locking
8. Cloud Performance

hackmysql.com/path

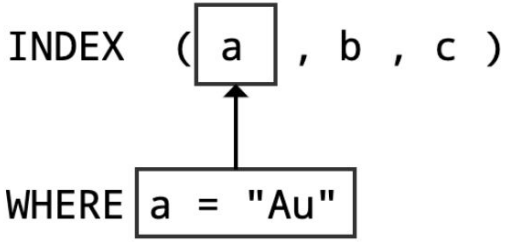
1. Query Response Time

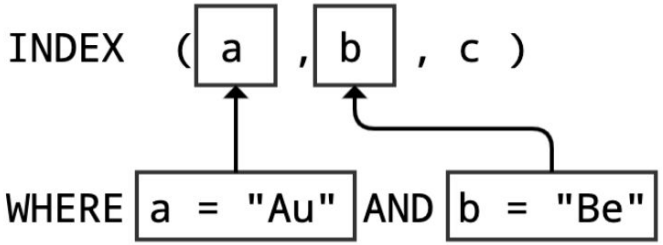
- MySQL does nothing (MySQL isn’t slow, the app makes it slow)
- Performance is query response time
- App developers often don’t know what queries the app executes or how fast they are

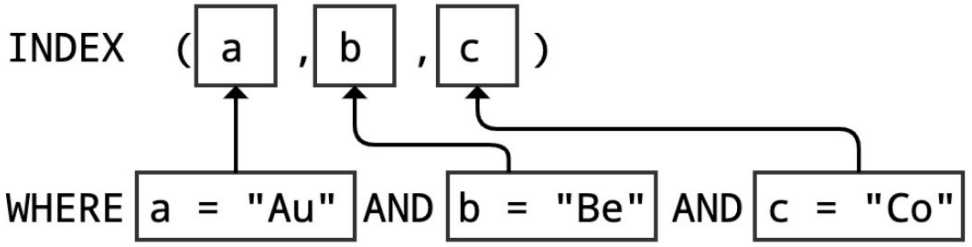
1. Choose a tool
2. Enable and configure MySQL query metrics
3. Analyze (and fix) slow queries
4. Teach others

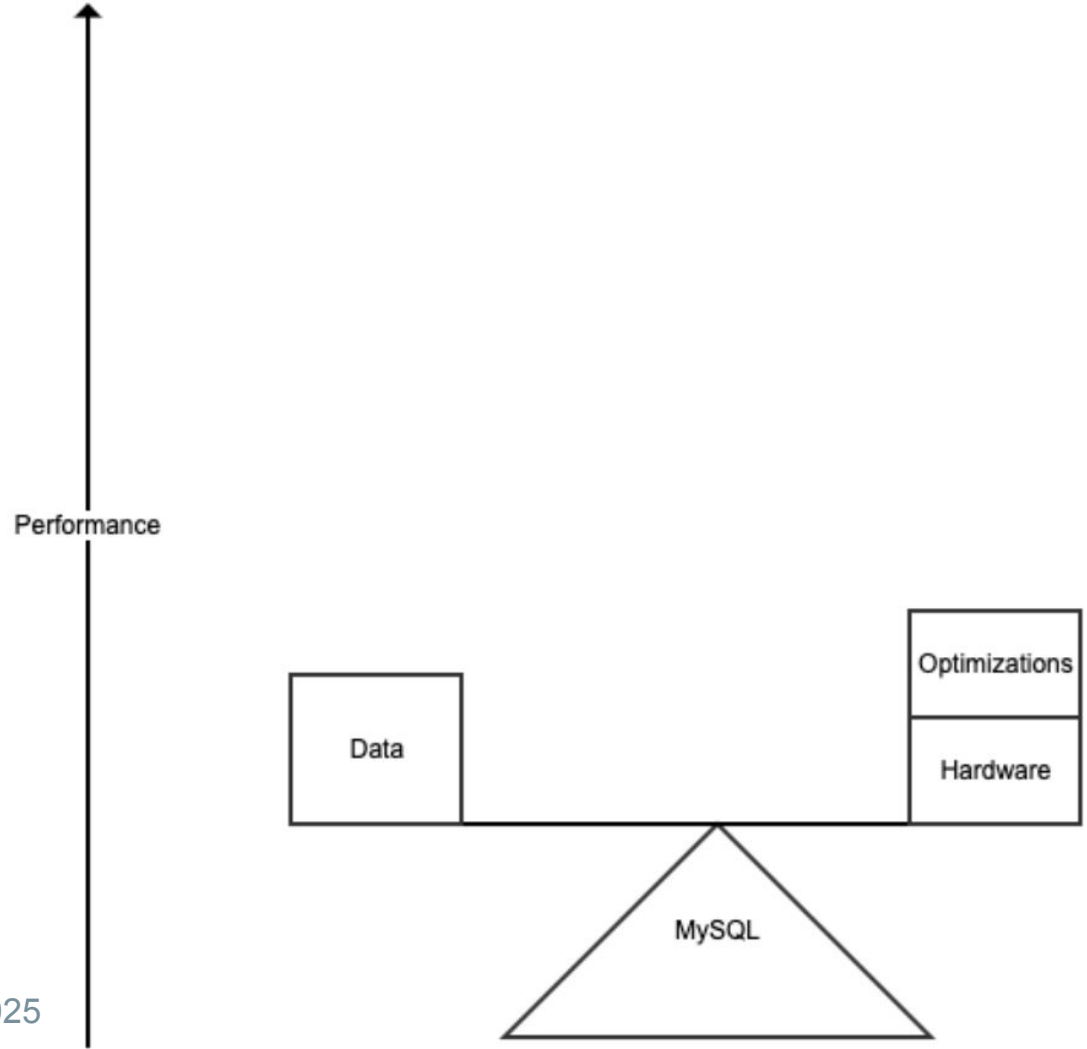
2. Indexes and Indexing

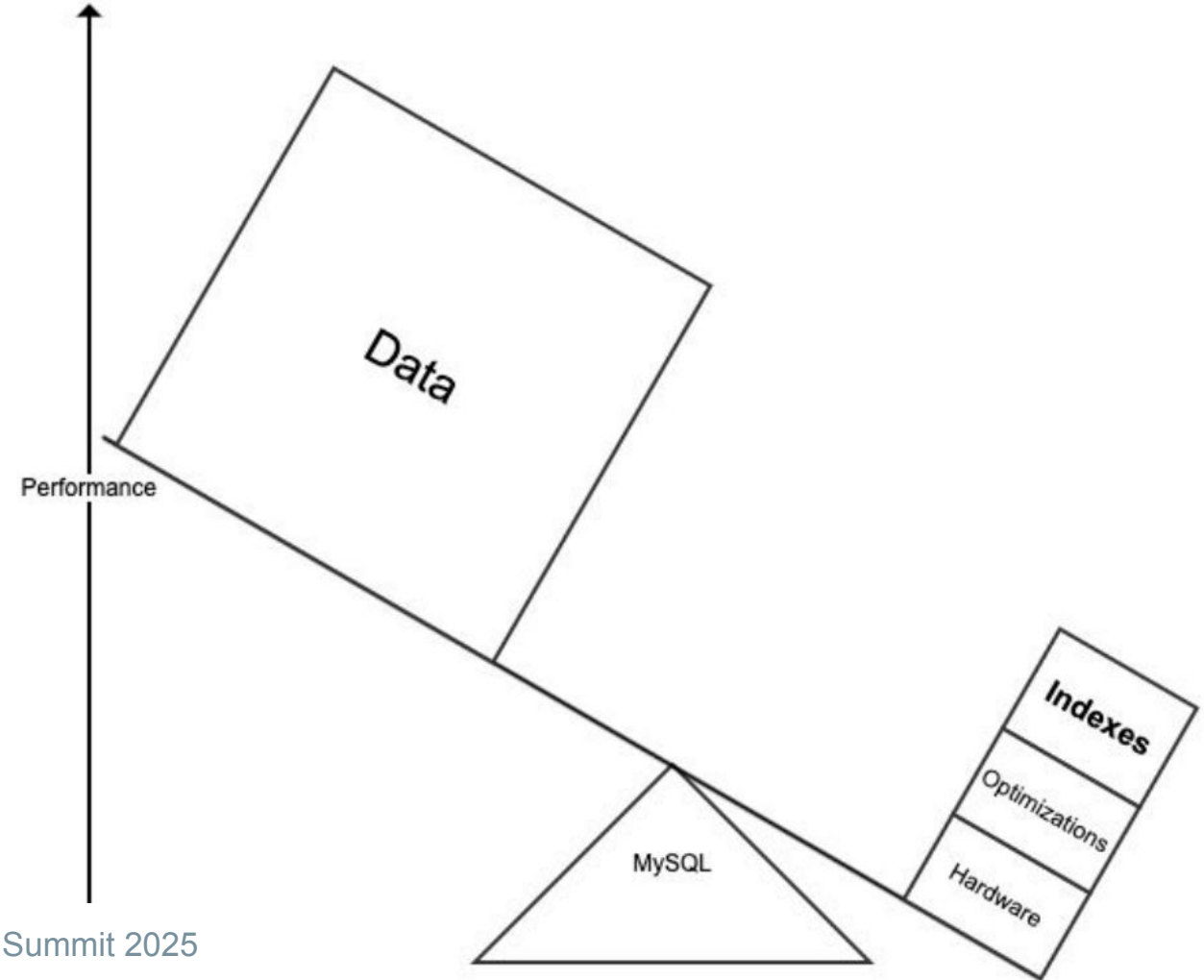
- Leftmost prefix requirement
- Five index uses:
  - WHERE
  - GROUP BY
  - ORDER BY
  - Covering Index
  - Join tables
- Indexes are leverage vs. data and access


INDEX (a , b , c)
WHERE a = "Au"


INDEX (a , b , c)
WHERE a = "Au" AND b = "Be"


INDEX (a , b , c)
WHERE a = "Au" AND b = "Be" AND c = "Co"


Performance
Optimizations
Data
Hardware
MySQL


Performance
Data
MySQL
Indexes
Optimizations
Hardware

2. Indexes and Indexing

- Leftmost prefix requirement
- Five index uses:
  - WHERE
  - GROUP BY
  - ORDER BY
  - Covering Index
  - Join tables
- Indexes are leverage vs. data and access

1. EXPLAIN slow queries
2. Analyze the leftmost prefix of those slow queries
3. Run `pt-duplicate-key-checker`
4. Read section “Optimizing SELECT Statements” in the MySQL manual

3. Data Storage and Access

- Data is dead weight
- Developers celebrate “less”; they cope with “more”
- Access patterns and working set size frame performance

1. Archive (delete) old data–*carefully!*
2. Review access patterns of slow queries
3. Consider other data stores or products

~~Efficient Path~~ A 30-minute Pointer to Dereference Later

1. **Query Response Time**
2. **Indexes and Indexing**
3. **Data Storage and Access**
4. Sharding
5. Server Metrics
6. Replication
7. Transactions and Data Locking
8. Cloud Performance

Thank You

hackmysql.com/path