Monitoring Your MySQL Server's Health Trends

Igor Donchovski, PlanetScale

About me

👨‍💻 Enterprise Support Engineer - Planetscale

🎓 Education: MSc in Software Engineering

📜 Certifications

- MongoDB Certified DBA
- Oracle Professional MySQL 5.7
- Terraform Associate Certified
- GCP Professional Architect

💻 Expertise: 20+ years in IT, 15 years with MySQL, 10 years MongoDB

🌟 Personal: Husband and Father, Avid Traveler, Speaker

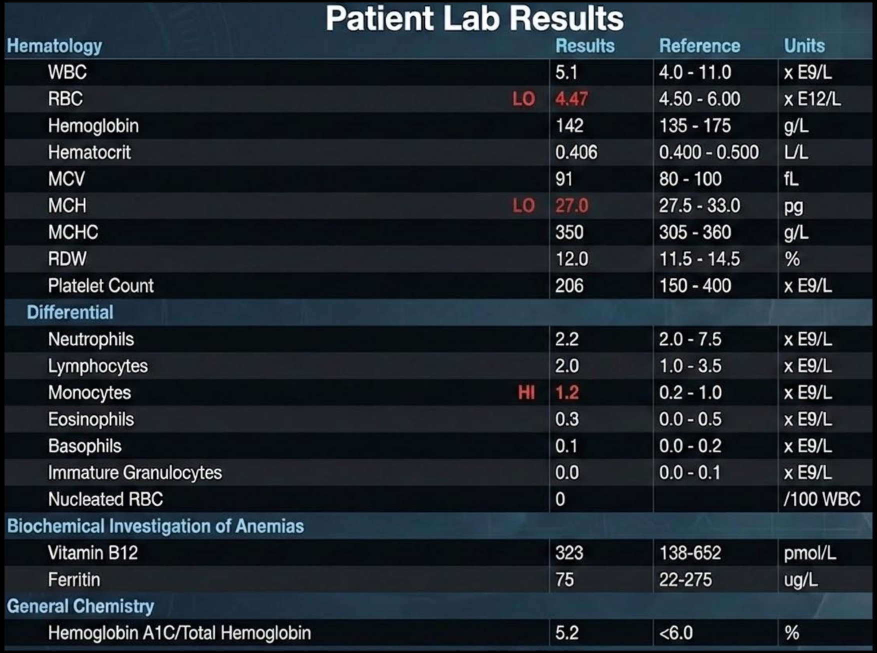
🧩 Playing Linkedin puzzle games on daily basis

Agenda

- Why MySQL databases need regular health checks
- Monitoring the server status
- Less is More
- Understanding normal ranges vs warning signs
- Building a monitoring dashboard
- From metrics to performance improvements
- AI as a supplementary monitoring tool

The Doctor's Office

- You visit the doctor annually for regular blood tests
- Doctor evaluates results against normal ranges (references)
- Early detection prevents serious problems
- Your database needs the same approach



Patient Lab Results

Hematology		Results	Reference	Units
WBC		5.1	4.0 - 11.0	x E9/L
RBC	LO	4.47	4.50 - 6.00	x E12/L
Hemoglobin		142	135 - 175	g/L
Hematocrit		0.406	0.400 - 0.500	L/L
MCV		91	80 - 100	fL
MCH	LO	27.0	27.5 - 33.0	pg
MCHC		350	305 - 360	g/L
RDW		12.0	11.5 - 14.5	%
Platelet Count		206	150 - 400	x E9/L
Differential				
Neutrophils		2.2	2.0 - 7.5	x E9/L
Lymphocytes		2.0	1.0 - 3.5	x E9/L
Monocytes	HI	1.2	0.2 - 1.0	x E9/L
Eosinophils		0.3	0.0 - 0.5	x E9/L
Basophils		0.1	0.0 - 0.2	x E9/L
Immature Granulocytes		0.0	0.0 - 0.1	x E9/L
Nucleated RBC		0		/100 WBC
Biochemical Investigation of Anemias				
Vitamin B12		323	138-652	pmol/L
Ferritin		75	22-275	ug/L
General Chemistry				
Hemoglobin A1C/Total Hemoglobin		5.2	<6.0	%

The Problem

Databases don't become slow by accident - something changed

- Performance degrades when something changes
    - schema, queries, or workload
- Undetected changes compound into major outages
- By the time users complain, multiple changes have accumulated
- Tracking changes early prevents most database problems
- Monitor both metrics AND what's changing

The Medical Analogy

Pretending a MySQL database is like a human body

- Blood tests → System metrics and performance data
- Normal ranges → Expected operating levels
- Warning signs → Degrading trends over time
- Diagnosis → Identifying the underlying cause
- Treatment → Performance tuning and targeted improvements
- Regular checkups → Ongoing monitoring and review

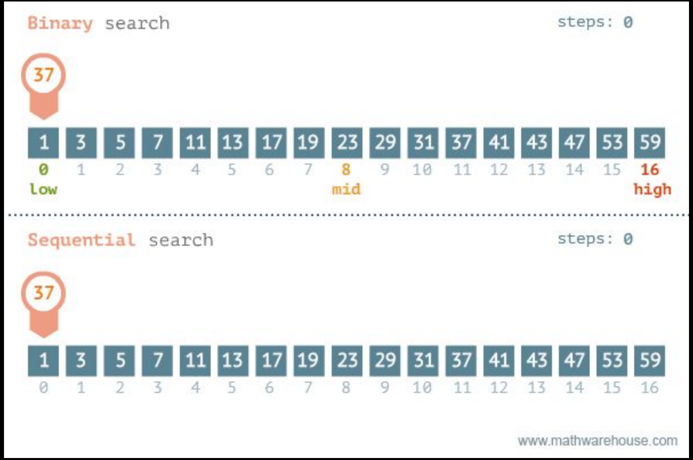
Full Table Scans

Metric: SELECT count(*) FROM sys.schema_tables_with_full_table_scans;

Value greater than 0 indicates there are queries doing full table scans

For detailed view:

```
mysql> SELECT * FROM sys.schema_tables_with_full_table_scans;
+---------------+----------------+--------------------+----------+
| object_schema | object_name    | rows_full_scanned | latency  |
+---------------+----------------+--------------------+----------+
| donchovski    | tables         |           1373502 | 1.53 s   |
+---------------+----------------+--------------------+----------+
```



Index Health - Redundant

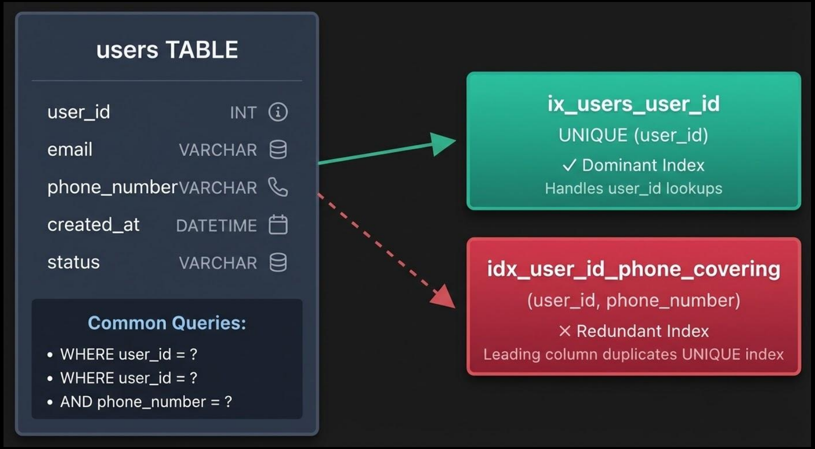
Every index costs memory and slows down writes

Metric: `SELECT count(*) FROM sys.schema_redundant_indexes where table_schema=database();`

Value greater than 0 indicates there are redundant indexes

For detailed view:

```
mysql> SELECT * FROM sys.schema_redundant_indexes where
table_schema=database()\G
              table_schema: donchovski
                table_name: users
     redundant_index_name: idx_user_id_phone_covering
   redundant_index_columns: user_id,phone_number
redundant_index_non_unique: 1
       dominant_index_name: ix_users_user_id
    dominant_index_columns: user_id
 dominant_index_non_unique: 0
            subpart_exists: 0
             sql_drop_index: ALTER TABLE `donchovski`.`users` DROP
INDEX `idx_user_id_phone_covering`
```



Index Health - Unused

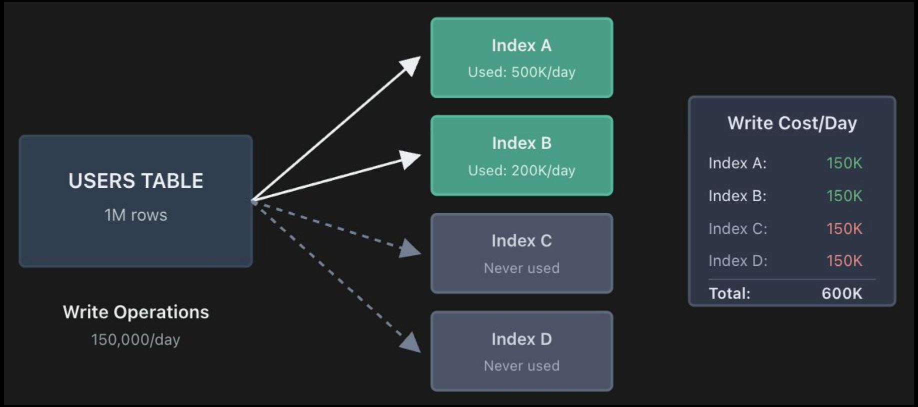
Every index costs memory and slows down writes

Metric: `SELECT count(*)  FROM sys.schema_unused_indexes where object_schema=database();`

Value greater than 0 indicates there are unused indexes

For detailed view:

```
mysql> SELECT *  FROM sys.schema_unused_indexes
where object_schema=database();
+---------------+-------------+----------------------+
| object_schema | object_name | index_name           |
+---------------+-------------+----------------------+
| donchovski    | users       | ix_users_phone_number |
| donchovski    | users       | idx_users_email      |
+---------------+-------------+----------------------+
```



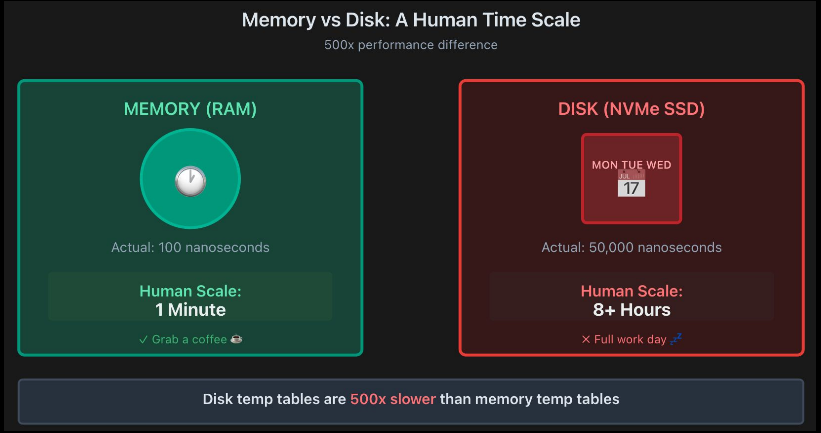
Temporary Tables

The number of internal [on-disk] temporary tables created by the server while executing statements

When memory threshold exceeded MIN(tmp_table_size, max_heap_table_size)

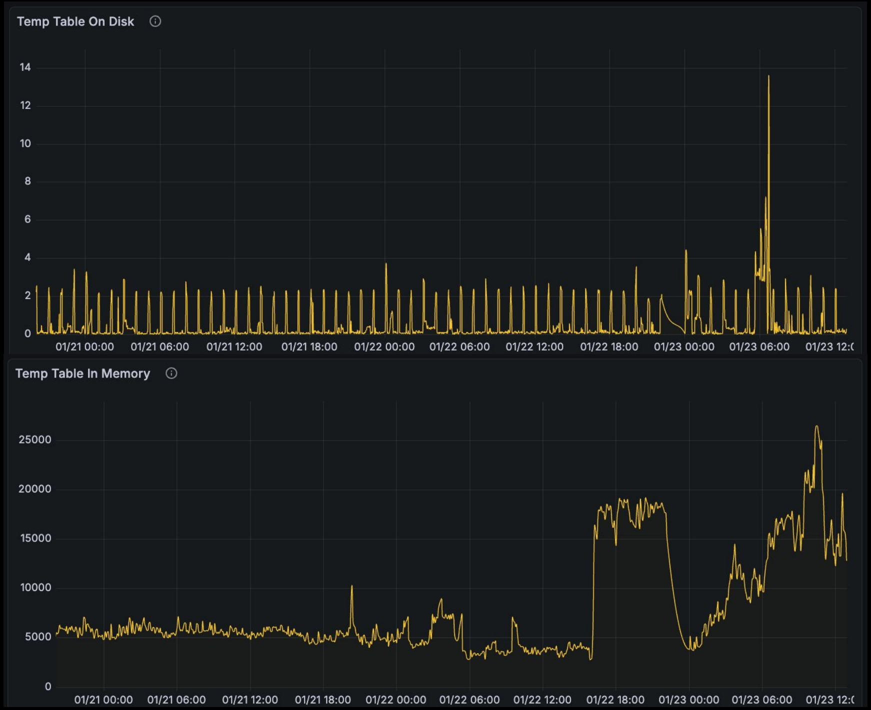
Common causes:

- Missing indexes forcing filesort
- Queries involving: GROUP BY, ORDER BY, DISTINCT, UNION
- Large result sets in subqueries
- Complex JOINs with many tables
- TEXT/BLOB columns forcing disk temp tables


Memory vs Disk: A Human Time Scale
500x performance difference
MEMORY (RAM)
Actual: 100 nanoseconds
Human Scale:
1 Minute
✓ Grab a coffee
DISK (NVMe SSD)
MON TUE WED
17
Actual: 50,000 nanoseconds
Human Scale:
8+ Hours
✕ Full work day
Disk temp tables are 500x slower than memory temp tables

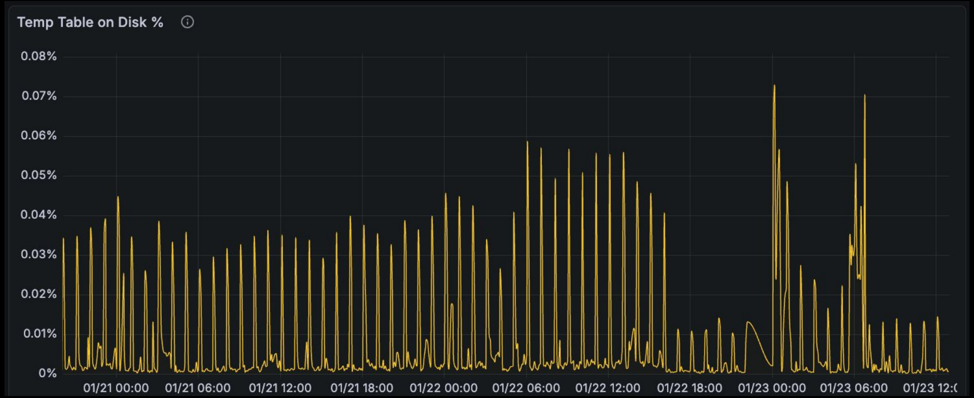
Temporary Tables

Metric: Created_tmp_disk_tables / Created_tmp_tables



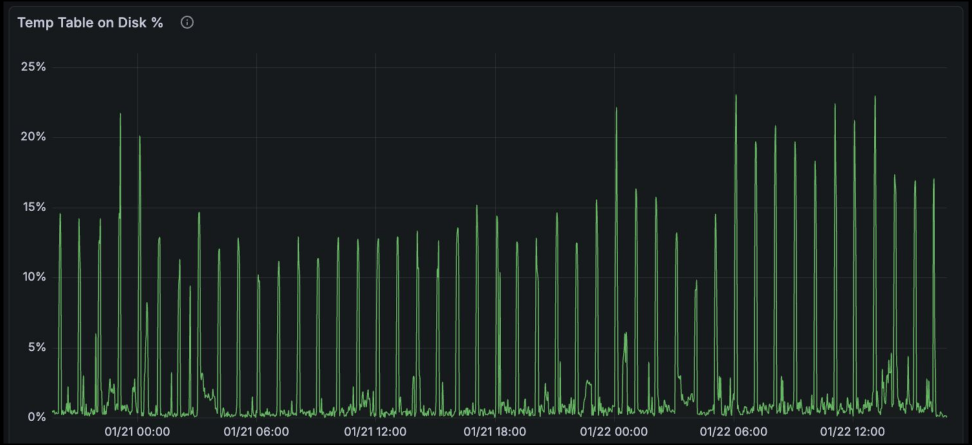
Temporary Tables

Showing how many temporary tables are created on disk compared to the total makes the data easier to understand



Temporary Tables

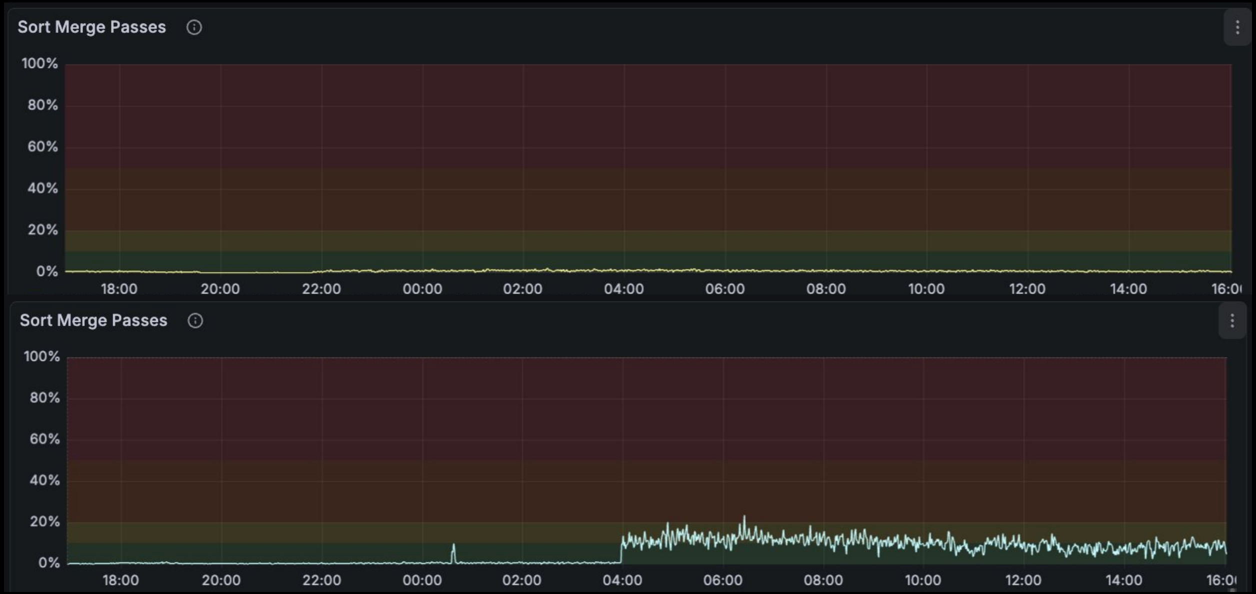
Showing how many temporary tables are created on disk compared to the total makes the data easier to understand



Sort Merge Passes

Metric: Sort_merge_passes / Sort_range + Sort_scans

- Measures how often sort operations require temporary file usage



Lock Waits

Metrics:

- Innodb_row_lock_waits - The number of times operations on InnoDB tables had to wait for a row lock
- Innodb_row_lock_time - The total time (ms) spent in acquiring row locks for InnoDB tables

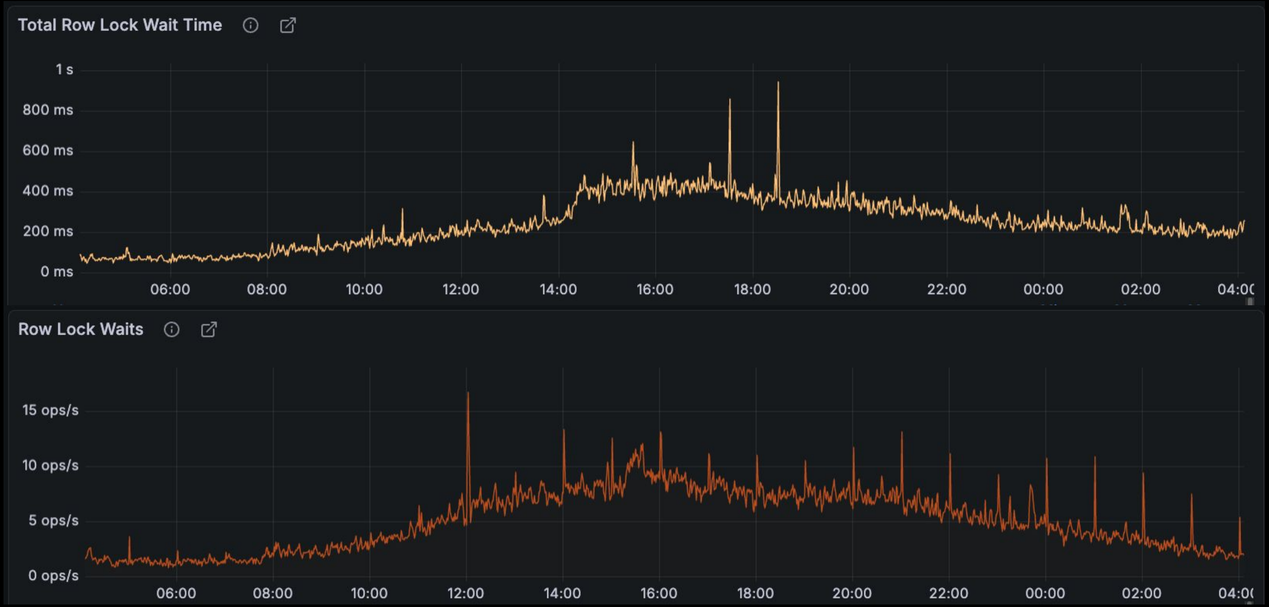
Measures transaction contention and locking

Single slow transaction can block hundreds of queries

Inefficient queries or high concurrency might also increase the lock waits

Lock Waits

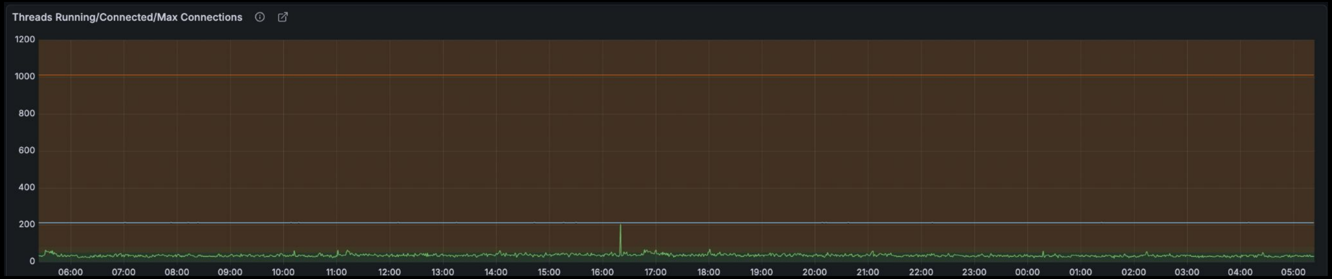
Smaller, faster transactions hold locks for less time and fewer users are blocked



Connection Utilization

Metrics: Threads_connected, Threads_running, max_connections

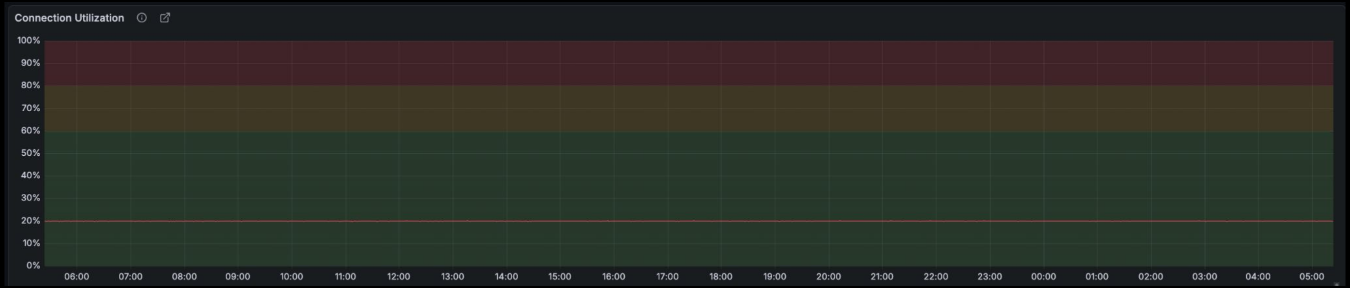
- Threads_connected = Total connections (idle + active)
- max_connections (default 151, max 100k) maximum permitted number of simultaneous client connections



Connection Utilization

Metrics: Threads_connected, Threads_running, max_connections

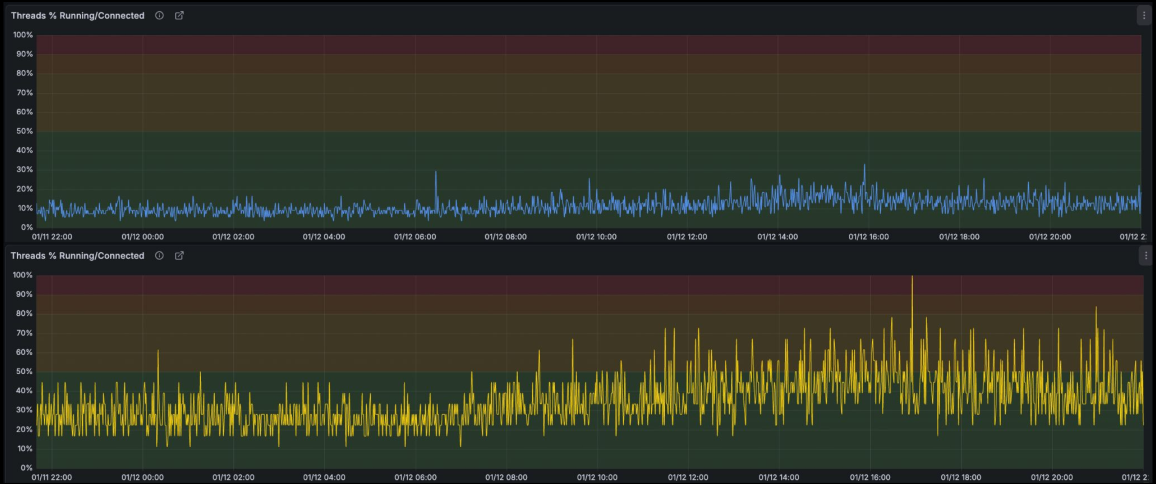
- Threads_connected = Total connections (idle + active)
- max_connections (default 151, max 100k) = maximum permitted number of simultaneous client connections



Thread Activity

Metrics: Threads_running, Threads_connected

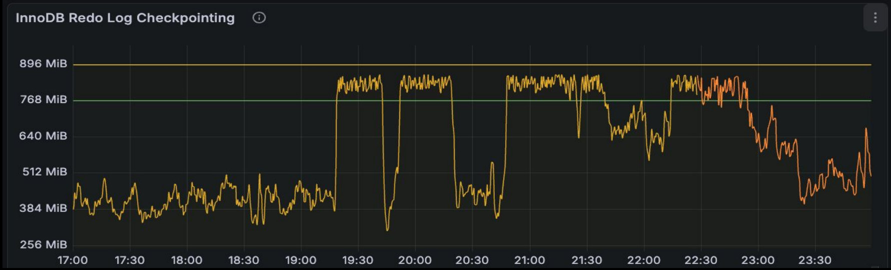
- Threads_connected = Total connections (idle + active)
- Threads_running = Actively executing queries
- High running threads = CPU contention, queueing
- Monitor the thread cache ratio, thread cache misses

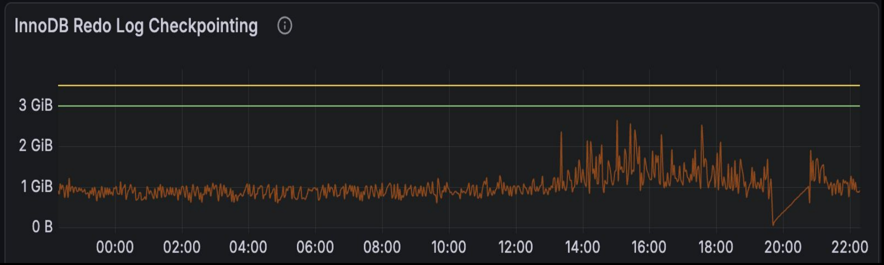


Redo Log Capacity

Metric: Innodb_redo_log_capacity_resized (MySQL 8.0.30+)

- Log Sequence Number (LSN) growth rate
- Checkpoint age relative to redo log size
- Async flush point (75% Redo log capacity)
- Sync flush point (87.5% Redo log capacity)

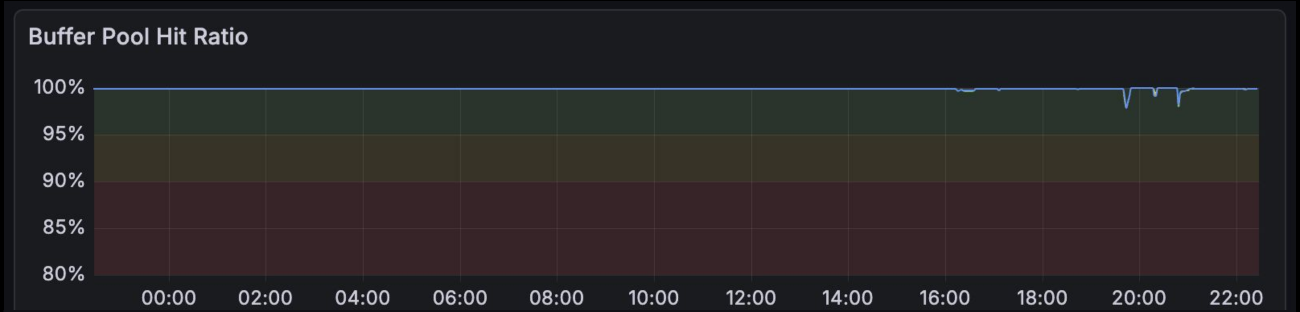


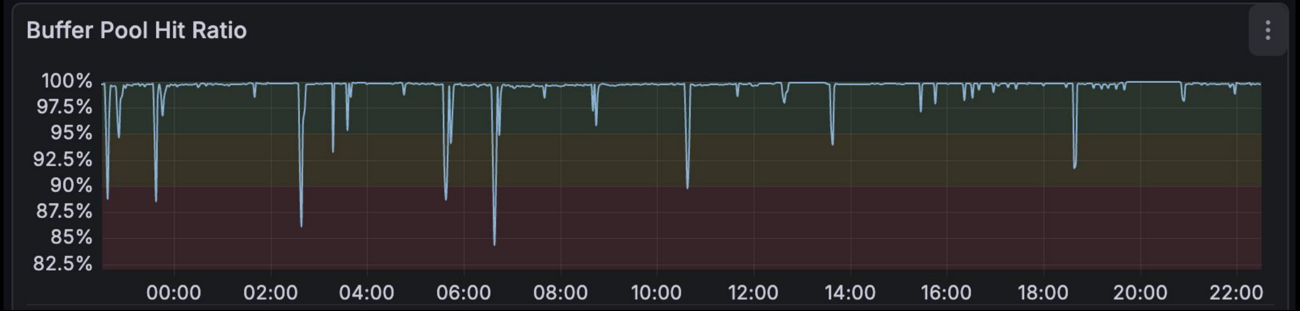


Buffer Pool Efficiency

Metric: Innodb_buffer_pool_read_requests / Innodb_buffer_pool_reads

- Measures cache hit ratio. Low hit rate = More disk I/O
- Hit ratio = (logical reads - physical reads) / logical reads





Buffer Pool Efficiency

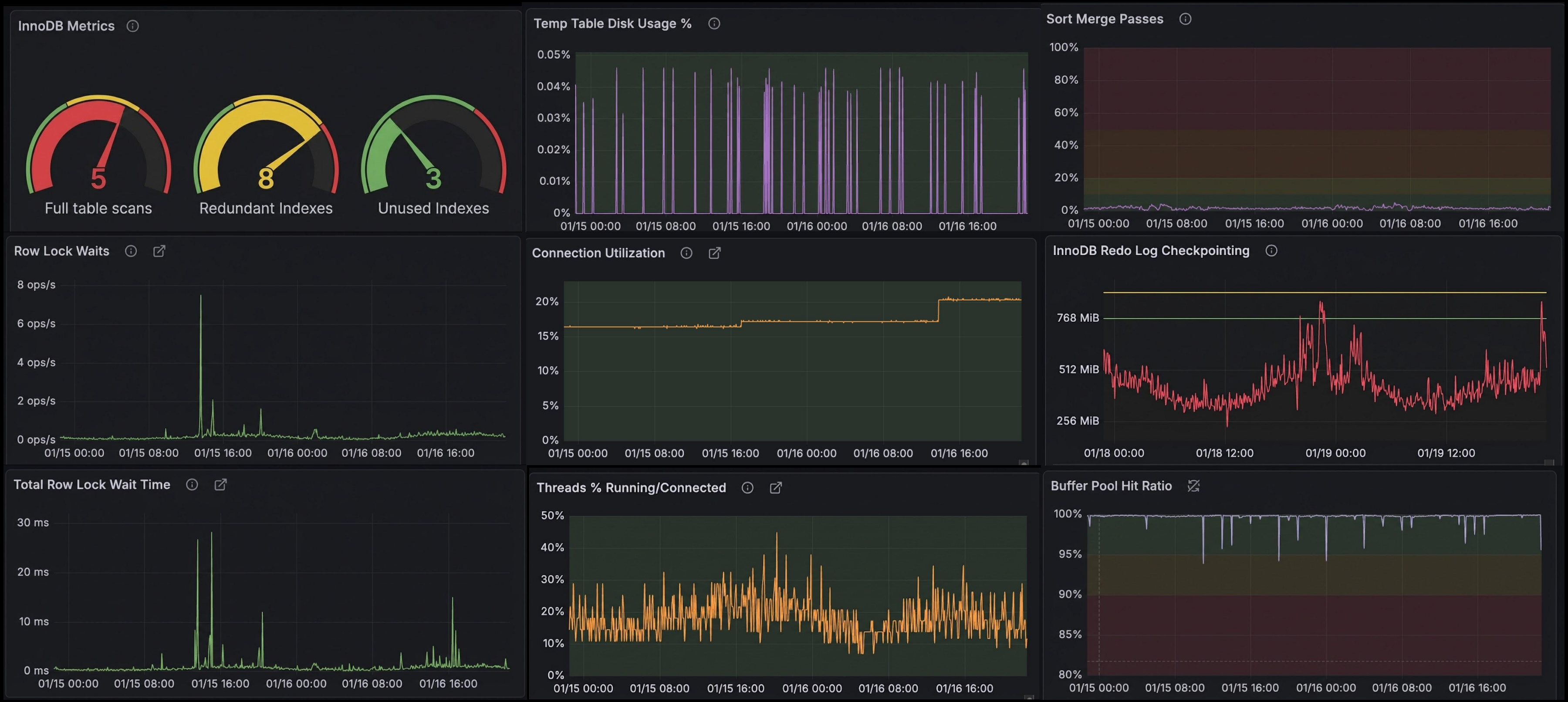
Working set size vs buffer pool size relationship

Causes of Low Hit Rate:

- Buffer pool too small for working set
- Full table scans evicting hot data
- Sudden query pattern change
- Data growth exceeding memory capacity

Actions: Review and optimize queries, reporting or batch workloads, changes in application

Monitoring Dashboard



Health Score with AI

Performance Issue	Points Deducted	Severity	Remaining Score
Starting Score	-	-	**10.0**
Full table scans identified	-0.4	High	9.6
Redundant indexes identified	-0.2	Medium	9.4
Unused indexes identified	-0.2	Medium	9.2
Temporary tables on disk > 25%	-0.6	Critical	8.6
InnoDB row lock wait 20+ ops	-0.4	High	8.2
InnoDB row lock wait time > 200ms	-0.2	Medium	8.0
Connection utilization at 70% of max_connections	-0.2	Medium	7.8
Threads_running at 60% of threads_connected	-0.4	High	7.4
Redo log checkpoint age near sync flush point	-0.6	Critical	6.8
Buffer pool hit ratio < 95%	-0.4	High	**6.4**

Scoring System with AI

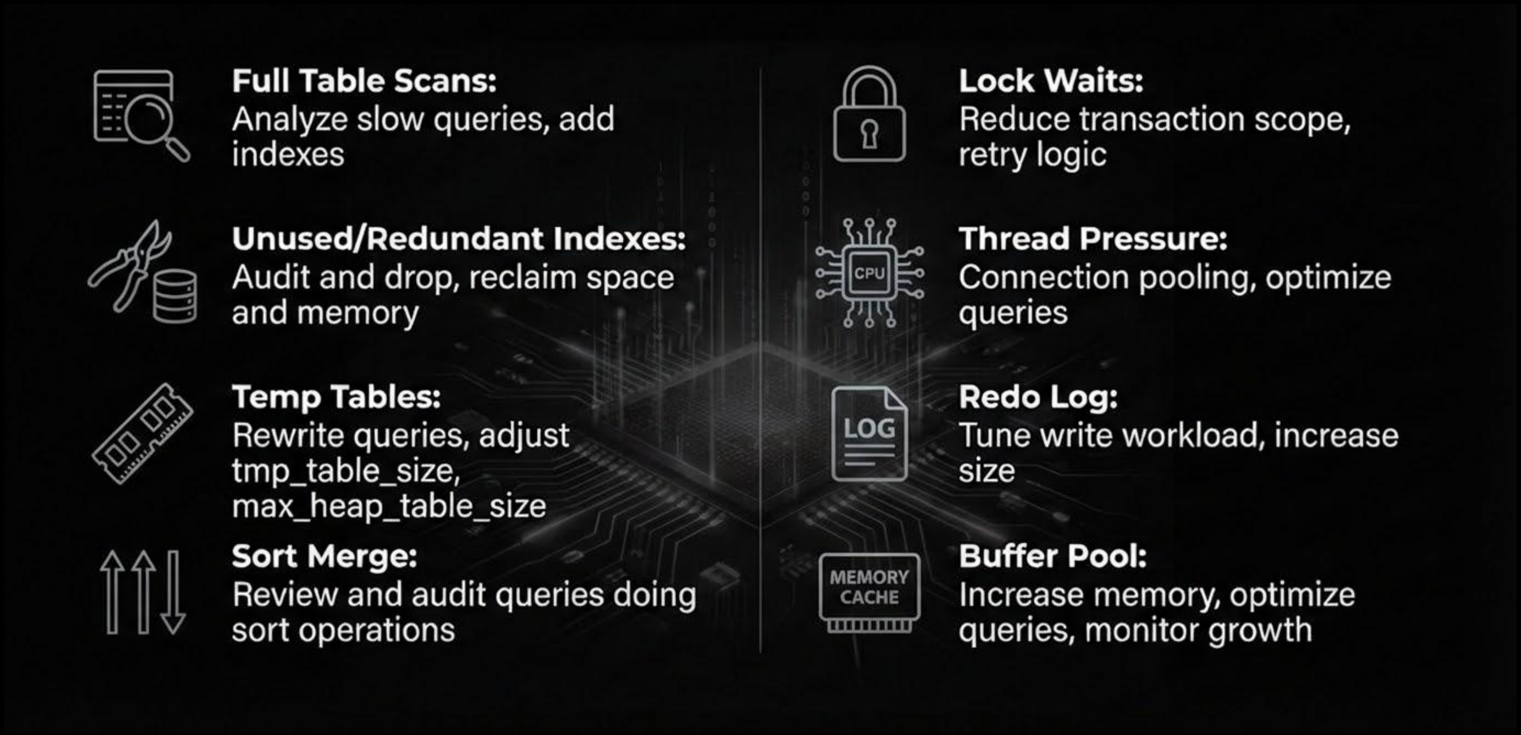
Scoring Interpretation

- **10**: Perfect Health
- **9.0-10.0**: Excellent - minor issues
- **8.0-8.9**: Good - specific areas need attention
- **7.0-7.9**: Fair - noticeable health issues
- **6.0-6.9**: Poor - immediate action required
- **Below 6.0**: Critical - emergency intervention needed

Point Deduction

- **Medium (0.2 pts)**: Optimization opportunities, not urgent
- **High (0.4 pts)**: Performance impact visible, needs planning
- **Critical (0.6 pts)**: Severe impact, requires immediate attention

Taking Action



Full Table Scans: Analyze slow queries, add indexes

Unused/Redundant Indexes: Audit and drop, reclaim space and memory

Temp Tables: Rewrite queries, adjust tmp_table_size, max_heap_table_size

Sort Merge: Review and audit queries doing sort operations

Lock Waits: Reduce transaction scope, retry logic

Thread Pressure: Connection pooling, optimize queries

Redo Log: Tune write workload, increase size

Buffer Pool: Increase memory, optimize queries, monitor growth

Thank You