MySQL Database Capacity Diagnostics and Secure Purge Specification

This technical specification is compiled for system administrators and infrastructure engineers to address storage capacity optimization demands within the core database **sof_db_host_insight** of the Network Access Control (NAC) system. It delivers standardized operating procedures (SOPs) spanning space utilization auditing, high-capacity table localization, historical data purging, and physical disk block reclamation to guarantee long-term database efficiency and architectural stability.

1. Phase I: Storage Capacity Diagnostics & Precision Localization

When database storage volumes encounter heavy performance loads, operations staff must execute a global schema audit to isolate the core tables driving disk allocation. Execute the following analytical query within your database management suite (e.g., HeidiSQL):

1. Retrieve Top 15 High-Capacity Tables within sof_db_host_insight

```
SELECT
    table_name AS 'Table Name',
    ROUND((data_length + index_length) / 1024 / 1024, 2) AS 'Total Capacity (MB)',
    table_rows AS 'Total Rows',
    ROUND(data_length / 1024 / 1024, 2) AS 'Data Capacity (MB)',
    ROUND(index_length / 1024 / 1024, 2) AS 'Index Capacity (MB)',
    ROUND(data_free / 1024 / 1024, 2) AS 'Reclaimable Space (MB)'
FROM
    information_schema.tables
WHERE
    table_schema = 'sof_db_host_insight'
ORDER BY
    (data_length + index_length) DESC
LIMIT 15;
```

2. Storage Metric Diagnostics & Asset Assessment

Based on production baseline analytics, storage consumption is predominantly driven by two primary network illegal event logging tables (collectively accounting for over 80% of total schema allocation):

Table Name	Total Space (MB)	Total Rows	Data Free	Operational Context & Constraints
sof_tb_net_work_illegal_event_record	3,151.19 MB	~7,666,179 rows	5.00 MB	Primary storage consumer. Reclaimable space sits at the baseline allocation threshold, indicating that physical data deletion is required; structural optimization alone will not reclaim blocks.
sof_tb_latest_net_work_illegal_event_record	803.47 MB	~1,966,554 rows	5.00 MB	Secondary storage consumer. Acts as the caching layer for recent violations, but has accumulated significant historical trace data.

2. Phase II: Data Lifecycle Retention Audit

Prior to executing any structural purge, engineers must verify the chronological bounds of the target assets. Because the underlying schema possesses a designated index on the `occurred_date` temporal field, a high-efficiency boundary scan can be performed:

```
SELECT
    MIN(occurred_date) AS 'Earliest Event',
    MAX(occurred_date) AS 'Latest Event'
FROM sof_db_host_insight.sof_tb_net_work_illegal_event_record;
```

Analytical Baseline: Active records span from **2023-08-23 to 2026-06-16**, accumulating roughly 3 years of historical telemetry. Referencing corporate compliance standards and statutory data retention mandates (typically 180 days or 365 days), logs older than 1 year lack immediate operational utility and are approved for data pruning.

3. Phase III: Detailed HeidiSQL Execution Steps & Secure Batch Purging

When manipulating multi-million row datasets, executing large-scale, unrestricted `DELETE` statements is **strictly prohibited**. Such actions cause protracted table locking (Table Lock), Undo Log exhaustion, and disk I/O starvation. Operations must follow this batch workflow:

1. Environment Setup & Targeting

1. Launch HeidiSQL and connect to the designated production MySQL cluster.
2. From the left-hand database tree navigator, left-click to focus the **`sof_db_host_insight`** schema.
3. Click the **[Query]** tab in the upper workspace navigation panel to initialize a clean SQL script viewport.

2. Execution of Chunked Secure Deletion (1-Year Retention Baseline)

Input the following query into the script viewport to prune obsolete records prior to **2025-06-16** in increments of 20,000 rows:

```
-- Chunks purges into 20k rows to protect active transactional throughput
DELETE FROM sof_db_host_insight.sof_tb_net_work_illegal_event_record
WHERE occurred_date < '2025-06-16 00:00:00'
LIMIT 20000;
```

- **Execution Method:** Click the blue **[Run] triangle icon** in the primary toolbar, or leverage the **`F9`** keyboard shortcut.
- **State Monitoring:** Following execution, audit the bottom "Log" panel output. A success state returns *"/* Affected rows: 20,000 ... */"*.
- **Iterative Loop:** Continuously trigger **`F9`** to cycle through batches until the console returns **`Affected rows: 0`**, confirming all data older than the retention boundary has been safely decoupled.

3. Secondary Storage Pruning Sync

Apply identical lifecycle truncation logic to the secondary table, looping the **`F9`** directive until affected rows reach zero:

```
DELETE FROM sof_db_host_insight.sof_tb_latest_net_work_illegal_event_record
WHERE occurred_date < '2025-06-16 00:00:00'
LIMIT 20000;
```

4. Mandatory Operational Step: Table Optimization for Physical Block Reclamation

> **Core Storage Concept:** Under the MySQL InnoDB engine architecture, running a `DELETE` routine does not cause immediate reduction in the filesystem's underlying physical data file (.ibd). The database engine merely tags those specific blocks as "vacant unallocated space." Engineers must commit an explicit optimize instruction to return physical sectors back to the host Windows/Linux OS storage pool.

Enter the table optimization commands and press **`F9`** to execute:

```
OPTIMIZE TABLE sof_db_host_insight.sof_tb_net_work_illegal_event_record;
OPTIMIZE TABLE sof_db_host_insight.sof_tb_latest_net_work_illegal_event_record;
```

⚠️ **Operational Safety Warning:** The `OPTIMIZE TABLE` command forces structural rebuilding and table locking. Given the scale of the remaining dataset, this process will require several minutes. **This step must be deferred to a designated low-traffic maintenance window (e.g., night-shift or weekend maintenance blocks)**. Never execute this command during peak operational hours.

4. Long-Term Maintenance Framework & Preventive Policies

1. **Scheduled Capacity Audits:** Systems infrastructure teams must institute a semi-annual inspection cadence using Phase I guidelines to track `sof_db_host_insight` growth vectors.
2. **Automated Retention Pipelines (Event Scheduler):** It is recommended to implement a native database event 排程 (MySQL Event Scheduler) to automate monthly batch prunings, ensuring logs never pool past compliance thresholds.