

MySQL Artifact Extraction Prerequisites

Contents

- 1. Introduction3
- 2. Artifact Extraction.....3
 - 2.1 DDL Scripts3
 - 2.2 Query Execution Logs3
 - 2.3 Other Database Objects4
- 3. Getting Help.....7

1. Introduction

LeapLogic's Assessment profiles existing inventory, identifies complexity, performs dependency analysis, and provides recommendations for migration to modern data platform.

2. Artifact Extraction

LeapLogic requires certain artifacts to perform an assessment. As a prerequisite, you need to have a super user privilege to fetch the required data. You can copy them from your LINUX environment, where all the artifacts such as DDL scripts, stored procedures, functions, query execution logs, DML scripts, and other database objects are stored from MySQL. LeapLogic needs all these artifacts in the form of .sql files.

2.1 DDL Scripts

Using a simple Shell script as given below, you can extract all the DDL scripts. To leverage this option, first connect with the MySQL-connector module.

```
mysqldump -h mysql-host -u username -p --no-data --routines --triggers --  
events databasename --result-file /tmp/databasename_db_output.sql
```

Modify the details highlighted in **Yellow** as per your environment. This script exports all the table DDLs. Please run this script for every database in the MySQL cluster.

Note: While running the script using the MySQL client, please provide the password if prompted.

2.2 Query Execution Logs

Use the **MySQL Query Editor** to start extracting the query execution logs. To start exporting the required query execution logs, refer to the below script.

Important Note: Please remember to change the dates marked in **Yellow** in the below script.

The total recommended timeframe is of **three months** for extracting the end-to-end query logs. Next, execute the given script below using the MySQL Query Editor UI and download the output after execution.

```
SELECT  
    "IDWWM" as FILLER,  
    "" as username,  
    "" as app_name,  
    "" as CLIENT_ID,  
    sa.first_seen as StartTime,  
    sa.last_seen as EndTime,  
    "" as TotalIOCount,  
    sa.digest as pid,  
    sa.query,  
    SCHEMA_NAME ,  
    sa.MAX_TOTAL_MEMORY  
    ,QUERY_SAMPLE TIMER WAIT
```

```

,sa.MAX_CONTROLLED_MEMORY
,sew.errors
,sew.warnings
,sa.DB as database
,CPU_latency
,esh.ROWS_SENT
,EVENT_NAME as `type`
,TIMER_WAIT / 1000000000000 AS cpu_time_sec
FROM
sys.statement_analysis sa
Left JOIN
performance_schema.events_statements_summary_by_digest esss
ON
sa.DIGEST = esss.DIGEST
Left join
sys.statements_with_errors_or_warnings sew
ON
sew.DIGEST=sa.DIGEST
left join
performance_schema.events_statements_history esh
ON
esh.DIGEST=sa.digest
where cast(sa.first_seen as date) between '2023-12-01' and '2024-04-30';

```

2.3 Other Database Objects

For rich insights from the assessment of your environment and workloads, we recommend exporting additional database objects as well. Please refer to the below script to export the data as separate delimited files.

```

---Database objects: Execute the below queries for every database using Super user credentials

select db, object_type , count from sys.schema_object_overview soo

---Databases:
select "Databases", count(object_schema) from
sys.innodb_buffer_stats_by_schema ibsbs
where object_schema not in ("mysql","InnoDB System", "sys");

----High Data Volume Tables
SELECT
table_schema AS `Database`,
table_name AS `Table`,
ROUND((data_length + index_length) / 1024 / 1024, 2) AS `Size (MB)`
FROM
information_schema.TABLES
WHERE
table_schema NOT IN ('information_schema', 'performance_schema', 'mysql', 'sys')
ORDER BY
(data_length + index_length) DESC;

---Data for Partitioning / Bucketing
SELECT
TABLE_SCHEMA,
TABLE_NAME,

```

```

PARTITION_NAME,
PARTITION_ORDINAL_POSITION,
PARTITION_METHOD,
SUBPARTITION_NAME,
SUBPARTITION_ORDINAL_POSITION,
SUBPARTITION_METHOD,
PARTITION_EXPRESSION,
SUBPARTITION_EXPRESSION,
TABLE_ROWS
FROM
    information_schema.PARTITIONS
where TABLE_SCHEMA not in ("mysql","InnoDB System",
"sys","performance_schema","information_schema") ;

---Count of stored procedures and Functions
SELECT ROUTINE_TYPE, COUNT(*) AS count
FROM information_schema.ROUTINES
WHERE ROUTINE_SCHEMA not in ("mysql","InnoDB System",
"sys","performance_schema","information_schema")
GROUP BY ROUTINE_TYPE;

---List of stored procedure
select ROUTINE_SCHEMA ,ROUTINE_NAME ,
ROUTINE_DEFINITION,SQL_DATA_ACCESS,ROUTINE_TYPE FROM
information_schema.ROUTINES where ROUTINE_TYPE =" PROCEDURE" and ROUTINE_SCHEMA
not in ("mysql","InnoDB System",
"sys","performance_schema","information_schema");

---Count of external Tables/Views in MySQL

SELECT TABLE_SCHEMA, TABLE_TYPE, COUNT(*) AS "count"
FROM information_schema.TABLES
WHERE TABLE_SCHEMA not in ("mysql","InnoDB System",
"sys","performance_schema","information_schema")
GROUP BY TABLE_TYPE, TABLE_SCHEMA;

---List of External Tables
SELECT TABLE_SCHEMA, TABLE_NAME ,TABLE_TYPE, TABLE_ROWS
FROM information_schema.TABLES
WHERE TABLE_SCHEMA not in ("mysql","InnoDB System",
"sys","performance_schema","information_schema")

---Total I/O Usage by Days
SELECT
    EVENT_NAME,
    SUM(COUNT_READ) AS total_reads,
    SUM(SUM_TIMER_READ) AS total_read_time,
    SUM(COUNT_WRITE) AS total_writes,
    SUM(SUM_TIMER_WRITE) AS total_write_time
FROM performance_schema.file_summary_by_event_name
GROUP BY EVENT_NAME;

----Database Volume:

```

```
SELECT
    table_schema AS 'Database',
    ROUND(SUM(data_length) / 1024 / 1024, 2) AS 'Data Size (MB)',
    ROUND(SUM(index_length) / 1024 / 1024, 2) AS 'Index Size (MB)',
    ROUND(SUM(data_length + index_length) / 1024 / 1024, 2) AS 'Total Size (MB)'
FROM information_schema.tables
GROUP BY table_schema;

-- (15day/1month/6months whatever is possible for extraction)
```

3. Getting Help

Contact LeapLogic technical support at info@leaplogic.io