



Fully Automated Snowflake Warehouse Optimization

Below are the Snowflake users, roles, and permissions you will need to create prior to using Keebo's warehouse optimization for Snowflake. These can be done during trial setup.

Keebo only accesses certain usage metadata within WAREHOUSE_METERING_HISTORY, QUERY_HISTORY, and WAREHOUSE_EVENTS_HISTORY from SNOWFLAKE.ACCOUNT_USAGE. Keebo does not access user data. For additional protection, instead of granting access to the above schema, the customer creates a KEEBO_SCHEMA with views that restrict Keebo's access to only the required metadata.

Create KEEBO_USER and KEEBO_ROLE

```
CREATE ROLE KEEBO_ROLE;  
CREATE USER KEEBO_USER PASSWORD = 'password';  
GRANT ROLE KEEBO_ROLE TO USER KEEBO_USER;  
ALTER USER KEEBO_USER SET DEFAULT_ROLE = KEEBO_ROLE;
```

Grant database access

```
GRANT USAGE ON DATABASE ${databaseName} TO ROLE ${userRole};  
CREATE SCHEMA ${databaseName}.${schemaName};  
GRANT ALL ON SCHEMA ${databaseName}.${schemaName} TO ROLE ${userRole};
```

Create query history view

```
CREATE VIEW ${databaseName}.${schemaName}.QUERY_HISTORY  
AS SELECT  
    QUERY_ID,  
    SHA2(QUERY_TEXT, 256) AS HASH_QUERY_TEXT,  
    SHA2(REGEXP_REPLACE(REGEXP_REPLACE(QUERY_TEXT, $$('.*?')$$,  
    $$'{str}'$$), $$-?\d+$$, ' {digit}'), 256) AS HASH_SANITIZED_QUERY,  
    DATABASE_ID,  
    DATABASE_NAME,  
    SCHEMA_ID,
```

SHA2(SCHEMA_NAME, 256) AS HASH_SCHEMA_NAME,
QUERY_TYPE,
SESSION_ID,
SHA2(USER_NAME, 256) AS HASH_USER_NAME,
SHA2(ROLE_NAME, 256) AS HASH_ROLE_NAME,
WAREHOUSE_ID,
WAREHOUSE_NAME,
WAREHOUSE_SIZE,
WAREHOUSE_TYPE,
CLUSTER_NUMBER,
QUERY_TAG,
EXECUTION_STATUS,
ERROR_CODE,
ERROR_MESSAGE,
START_TIME,
END_TIME,
TOTAL_ELAPSED_TIME,
BYTES_SCANNED,
PERCENTAGE_SCANNED_FROM_CACHE,
BYTES_WRITTEN,
BYTES_WRITTEN_TO_RESULT,
BYTES_READ_FROM_RESULT,
ROWS_PRODUCED,
ROWS_INSERTED,
ROWS_UPDATED,
ROWS_DELETED,
ROWS_UNLOADED,
BYTES_DELETED,
PARTITIONS_SCANNED,
PARTITIONS_TOTAL,
BYTES_SPILLED_TO_LOCAL_STORAGE,
BYTES_SPILLED_TO_REMOTE_STORAGE,
BYTES_SENT_OVER_THE_NETWORK,
COMPILATION_TIME,
EXECUTION_TIME,
QUEUED_PROVISIONING_TIME,
QUEUED_REPAIR_TIME,
QUEUED_OVERLOAD_TIME,
TRANSACTION_BLOCKED_TIME,
CREDITS_USED_CLOUD_SERVICES,
QUERY_LOAD_PERCENT,
OUTBOUND_DATA_TRANSFER_CLOUD,

```

        OUTBOUND_DATA_TRANSFER_REGION,
        OUTBOUND_DATA_TRANSFER_BYTES,
        INBOUND_DATA_TRANSFER_CLOUD,
        INBOUND_DATA_TRANSFER_REGION,
        INBOUND_DATA_TRANSFER_BYTES,
        LIST_EXTERNAL_FILES_TIME,
        RELEASE_VERSION,
        EXTERNAL_FUNCTION_TOTAL_INVOCATIONS,
        EXTERNAL_FUNCTION_TOTAL_SENT_ROWS,
        EXTERNAL_FUNCTION_TOTAL_RECEIVED_ROWS,
        EXTERNAL_FUNCTION_TOTAL_SENT_BYTES,
        EXTERNAL_FUNCTION_TOTAL_RECEIVED_BYTES,
        IS_CLIENT_GENERATED_STATEMENT
FROM SNOWFLAKE.ACCOUNT_USAGE.QUERY_HISTORY;
GRANT ALL ON VIEW ${dbName}.${schemaName}.QUERY_HISTORY TO
ROLE KEEBO_ROLE;

```

Create metering history view

```

CREATE VIEW ${dbName}.${schemaName}.WAREHOUSE_METERING_HISTORY
AS SELECT
    CREDITS_USED,
    CREDITS_USED_CLOUD_SERVICES,
    CREDITS_USED_COMPUTE,
    END_TIME,
    START_TIME,
    WAREHOUSE_ID,
    WAREHOUSE_NAME
FROM SNOWFLAKE.ACCOUNT_USAGE.WAREHOUSE_METERING_HISTORY;
GRANT ALL ON VIEW ${dbName}.${schemaName}.WAREHOUSE_METERING_HISTORY TO
ROLE ${userRole};

```

Create warehouse events view

```

CREATE VIEW ${dbName}.${schemaName}.WAREHOUSE_EVENTS_HISTORY
AS SELECT
    CLUSTER_NUMBER,
    EVENT_NAME,
    EVENT_REASON,
    EVENT_STATE,
    QUERY_ID,

```

```
        ROLE_NAME,  
        TIMESTAMP,  
        USER_NAME,  
        WAREHOUSE_ID,  
        WAREHOUSE_NAME  
FROM SNOWFLAKE.ACCOUNT_USAGE.WAREHOUSE_EVENTS_HISTORY;  
GRANT ALL ON VIEW ${databaseName}.${schemaName}.WAREHOUSE_EVENTS_HISTORY TO  
ROLE ${userRole};
```

Grant permissions on the warehouse to optimize

```
GRANT USAGE, MODIFY, OPERATE  
ON WAREHOUSE <WAREHOUSE>  
TO ROLE KEEBO_ROLE;
```