

AMPS Statistics Database (SQLite 3) Reference



AMPS Statistics Database (SQLite 3) Reference

Publication date Oct 29, 2015

Copyright © 2015

All rights reserved. 60East, AMPS, and Advanced Message Processing System are trademarks of 60East Technologies, Inc. All other trademarks are the property of their respective owners.

Table of Contents

1. Introduction	1
1.1. About this Reference	1
1.2. Prerequisites	1
1.3. Configuration of AMPS	1
2. Introduction to SQLite3	2
2.1. Starting sqlite3	2
2.2. Simple SQLite3 commands	2
3. Admin Interface	4
3.1. Table Naming Scheme	4
3.2. Example Queries	4
4. SQLite Tips and Troubleshooting	5
4.1. Converting AMPS statistics time to an ISO8601 Datetime	5
4.2. Troubleshooting "Database Disk Image is Malformed"	5

Chapter 1. Introduction

1.1. About this Reference

The goal of this reference is to give a user insight into the statistics collected by the AMPS administration interface, and to demonstrate how those statistics can be used to give an overview of the health and efficiency of their AMPS instance.

1.2. Prerequisites

To run the commands in this reference, you will need the `sqlite3` package installed on your local computer. While you may be able to run the SQL examples in this guide using other packages, this guide will assume that all SQL commands will be executed with `sqlite3`.

1.3. Configuration of AMPS

By default AMPS is configured to store the statistics file in a `sqlite` database which is in memory. To configure AMPS to record the statistics to a file, the following configuration options are available in the AMPS configuration file to update the location and frequency of the statistics database file.

```
<AMPSConfig>
  <Name>AMPS-Sqlite</Name>
  <Admin>
    <InetAddr>localhost:9090</InetAddr>
    <FileName>./stats.db</FileName>
    <Interval>5s</Interval>
  </Admin>
  <!-- [snip] -->
</AMPSConfig>
```

In the example listed above, the AMPS administration interface is set to collect statistics every 5 seconds as indicated by the `<Interval>` tag. In the example, the AMPS administration interface is additionally configured to save the statistics in the `stats.db` file which will be created in the directory where AMPS was started.

Chapter 2. Introduction to SQLite3

This chapter is a quick reference to sqlite3. It is intended to help get started in examining the statistics provided by AMPS. While this guide will be sufficient to execute the examples listed, a more comprehensive guide of the sqlite3 command line tool is available at <http://www.sqlite.org/sqlite.html>.

2.1. Starting sqlite3

To start sqlite3 with the stats.db file simply type: `sqlite`.

```
$> sqlite3 ./stats.db
```

This will create a command prompt that looks like the following:

```
$> sqlite3 ./stats.db
SQLite version 3.7.3
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite>
```

To exit the sqlite3 prompt at any time, use the Ctrl+d sequence.

2.2. Simple SQLite3 commands

Tables

To get a listing of all available tables in the sqlite database type the `.table` command.

```
sqlite> .table
HCPUS_DYNAMIC      IMEMORY_CACHES_STATIC
HCPUS_STATIC       IMEMORY_DYNAMIC
HDISKS_DYNAMIC     IMEMORY_STATIC
HDISKS_STATIC      IPROCESSORS_DYNAMIC
HMEMORY_DYNAMIC    IPROCESSORS_STATIC
HMEMORY_STATIC     IREPLICAS_DYNAMIC
HNET_DYNAMIC       IREPLICAS_STATIC
HNET_STATIC        IREPLICATIONS_DYNAMIC
ICLIENTS_DYNAMIC  IREPLICATIONS_STATIC
ICLIENTS_STATIC    ISOW_DYNAMIC
ICONSOLE_LOGGERS_DYNAMIC  ISOW_STATIC
ICONSOLE_LOGGERS_STATIC  ISTATISTICS_DYNAMIC
ICPUS_DYNAMIC      ISTATISTICS_STATIC
ICPUS_STATIC       ISUBSCRIPTIONS_DYNAMIC
```

```
IFILE_LOGGERS_DYNAMIC      ISUBSCRIPTIONS_STATIC
IFILE_LOGGERS_STATIC      ISYSLOG_LOGGERS_DYNAMIC
IGLOBALS_DYNAMIC          ISYSLOG_LOGGERS_STATIC
IGLOBALS_STATIC           ITRANSPORTS_DYNAMIC
IMAPS_DYNAMIC             ITRANSPORTS_STATIC
IMAPS_STATIC              IVIEWS_DYNAMIC
IMEMORY_CACHES_DYNAMIC    IVIEWS_STATIC
```

Schema

To view the schema for any table type: `.schema <table name>`, where `<table name>` is the name of the table to inspect.

```
sqlite> .schema IFILE_LOGGERS_DYNAMIC

CREATE TABLE IFILE_LOGGERS_DYNAMIC( timestamp integer,
static_id integer, bytes_written integer, PRIMARY
KEY( timestamp, static_id ) );
```

Chapter 3. Admin Interface

This chapter describes the philosophy of how the AMPS tables are designed within the statistics database. This chapter also includes some examples of some useful queries which can give an administrator more information than just the raw data would normally give them. Such information can be a powerful tool in diagnosing perceived problems in AMPS.

3.1. Table Naming Scheme

```
<I|H><STAT>_<STATIC|DYNAMIC>
```

Where:

I = AMPS instance stat

H = Host stat

STAT = The stat that is collected (MEMORY, CPUS, SUBSCRIPTIONS, etc)

STATIC = attributes that rarely change for an object (i.e. client name)

DYNAMIC = stats that are expected to change on every sample (rates, counters, etc)

3.2. Example Queries

To view which clients have fallen behind at one time, run:

```
sqlite> SELECT s.client_name, MAX(d.queue_max_latency),  
MAX(queued_bytes_out) FROM ICLIENTS_DYNAMIC d  
JOIN ICLIENTS_STATIC s ON (s.static_id=d.static_id)  
GROUP BY s.client_name;
```

To view clients that are behind in the latest sample:

```
sqlite> SELECT s.client_name, d.queue_max_latency,  
queued_bytes_out FROM ICLIENTS_DYNAMIC d  
JOIN ICLIENTS_STATIC s ON (s.static_id=d.static_id)  
WHERE d.timestamp = (SELECT MAX(d.timestamp)  
FROM ICLIENTS_DYNAMIC d) AND d.queue_max_latency > 0;
```

Chapter 4. SQLite Tips and Troubleshooting

This chapter includes information on SQLite tasks that may not be immediately obvious, and troubleshooting information on SQLite.

4.1. Converting AMPS statistics time to an ISO8601 Datetime

This function converts an AMPS timestamp to an ISO8601 datetime.

```
def iso8601_time(amps_time):  
    """  
    Converts AMPS Stats time into an ISO8601 datetime.  
    """  
    pt = float(amps_time)/1000 - 210866803200 # subtract the unix epoch  
    it = int(pt)  
    ft = pt-it  
    return time.strftime("%Y%m%dT%H%M%S",time.localtime(it)) + ("%06f" %  
        ft)[1:]
```

4.2. Troubleshooting "Database Disk Image is Malformed"

To repair this error, you need to extract the data from the SQLite datastore and create a new datastore. To do this:

1. Open the sqlite datastore. For example, if the database store is named `stats.db`, the command would be:

```
:sqlite3 stats.db
```

2. Dump the data into a SQL script.

```
.mode insert  
.output stats_data.sql  
.dump  
.exit
```

This creates a series of SQL commands that recreate the data in the database.

3. Now create a new database file using the SQL commands.


```
sqlite3 good.db < stats_data.sql
```

Finally, adjust the configuration of the Admin server to use the new database (in this example, `good.db`) or copy the new database over the old database.