GBASE

GBase 8a 分析型数据管理系统

管理员手册

www.gbase.cn

GBase 8a 管理员手册，南大通用数据技术股份有限公司

GBase 版权所有©2004-2017，保留所有权利。

版权声明

本文档所涉及的软件著作权、版权和知识产权已依法进行了相关注册、登记，由南大通用数据技术股份有限公司合法拥有，受《中华人民共和国著作权法》、《计算机软件保护条例》、《知识产权保护条例》和相关国际版权条约、法律、法规以及其它知识产权法律和条约的保护。未经授权许可，不得非法使用。

免责声明

本文档包含的南大通用公司的版权信息由南大通用公司合法拥有，受法律的保护，南大通用公司对本文档可能涉及到的非南大通用公司的信息不承担任何责任。在法律允许的范围内，您可以查阅，并仅能够在《中华人民共和国著作权法》规定的合法范围内复制和打印本文档。任何单位和个人未经南大通用公司书面授权许可，不得使用、修改、再发布本文档的任何部分和内容，否则将视为侵权，南大通用公司具有依法追究其责任的权利。

本文档中包含的信息如有更新，恕不另行通知。您对本文档的任何问题，可直接向南大通用数据技术股份有限公司告知或查询。

未经本公司明确授予的任何权利均予保留。

通讯方式

南大通用数据技术股份有限公司

天津华苑产业区海泰发展六道 6 号海泰绿色产业基地 J 座(300384)

电话：400-013-9696　　邮箱：info@gbase.cn

商标声明

GBASE是南大通用数据技术股份有限公司向中华人民共和国国家商标局申请注册的注册商标，注册商标专用权由南大通用公司合法拥有，受法律保护。未经南大通用公司书面许可，任何单位及个人不得以任何方式或理由对该商标的任何部分进行使用、复制、修改、传播、抄录或与其它产品捆绑使用销售。凡侵犯南大通用公司商标权的，南大通用公司将依法追究其法律责任。

目 录

前言

手册简介

GBase 8a 管理员手册主要从用户使用的角度介绍了 GBase 8a 数据库日常管理维护的知识，通过阅读本手册用户可以学习到 GBase 8a 数据库的常用管理技能和维护技巧，掌握对数据库常用对象的命令管理和图形化管理技能，掌握对数据库状态监控的技能。

第一章对 GBase 8a 做简单介绍，主要描述了 GBase 8a 的功能，特点等方面的内容。

第二章介绍了 GBase 8a 的基本管理，主要描述了服务的启停，登录与退出的内容。

第三章介绍了 GBase 8a 中 DDL 与 DML 的语法，主要描述了基本的 DDL 与 DML 的语法内容。

第四章介绍了 GBase 8a 中的审计日志，主要描述了如何运用审计日志来监视用户所执行的数据库操作的内容。

第五章介绍了 GBase 8a 的权限管理，主要描述了用户管理以及用户权限管理方面的内容。

第六章对于 GBase 8a 的图形化管理工具进行了简单介绍。

公约

下面的文本约定用于本文档：

约　定	说　明
加粗字体	表示文档标题
大写英文（SELECT）	表示 GBase 8a 关键字
等宽字体	表示代码示例
…	表示被省略的内容。

1 系统简介

1.1 系统简介

GBase 8a是一款具有高效统计和分析能力的列存储关系型数据库管理系统，能够管理TB级数据。

GBase 8a面向分析型应用领域，以列为基本存储方式和数据运算对象，结合列数据压缩处理、并行处理、快速智能索引等新型数据处理技术，在查询、统计、分析以及批量加载性能上具备突出的优势。

GBase 8a符合SQL92、ODBC、JDBC、ADO.NET等国际规范，提供完备的数据存储和数据管理功能。

1.2 产品技术特点

GBase 8a技术上的“三高”优势:

- 高性能

列存储在大大减少了I/O的同时,显著提高查询性能;

智能索引大幅提高查询性能;

高速的数据加载性能;

高效的并行 SQL 执行计划。

- 高性价比

采用多种压缩技术,减少存储数据所需的空间,可以将所用空间减少1～20倍，并相应地提高了有效的I/O性能;采用高压缩技术，能显著减少存储开销，从而帮助客户减少了数据库整体投入成本。

- 高易用性

易于实施和管理，只需要传统数据库 1/10 的管理成本，与主要商业智能

工具兼容，如 GBase BI、Cognos、SAP BO、BIEE、SAS、SPSS。

1.3 产品功能简介

GBase 8a 支持的功能如下表所示：

功　能	描　述
结构化查询语言	符合 SQL 92 标准，支持 CREATE、ALTER、DROP 等 DDL 语法，支持 SELECT、INSERT、UPDATE、DELETE 等 DML 语法，支持单表，多表联合查询。
数据类型	INT、TINYINT、SMALLINT、BIGINT、DECIMAL、FLOAT、DOUBLE 数值数据类型； CHAR、VARCHAR TEXT 字符数据类型； DATE、TIME、DATETIME、TIMESTAMP 日期类型； BLOB 二进制数据类型。
数据库对象	提供了数据库，表，索引，视图，存储过程，自定义函数等常用数据库对象的创建，修改和删除操作，支持数据库用户的创建，删除操作，以及用户权限的分配与回收。
行列混合存储	基于创建的物理表，可以实现行存列的创建，修改和删除。
图形化工具	提供了企业管理工具。
接口	符合并支持 C API、ODBC、JDBC、ADO.NET 等接口规范。
外围工具	提供数据加载、导出、备份/恢复等外围工具。

2 基本管理

2.1 系统安装

这部分内容请参见《GBase 8a 安装手册(Linux RHEL6)》。

2.2 服务的启停

当 GBase 8a 安装完毕后，其中的 gbase.server 服务需要手动启动，之后，每当开机和重新启动机器时，gbase.server 服务都需要手动启动。如果用户在使用中，需要手工进行 GBase 服务的启停操作，则要使用安装 GBase 8a 的用户进行操作。具体命令如下:

- 启动命令

```
$ gbase.server start
```

- 停止命令

```
$ gbase.server stop
```

- 重新启动命令

```
$ gbase.server restart
```

更详细的介绍，参见《GBase 8a 安装手册(Linux RHEL6)》。

2.3 修改数据库 root 用户的密码

在安装 GBase 8a 的用户下，可以进行 GBase 8a 的登录。

默认情况下，在进行 GBase 8a 的安装过程中，系统创建数据库超级帐号 root，并且可以为它设置初始密码，如果在这个安装过程中，用户将 root 的密码设置为空，那么，用户还可以在首次登录 GBase 8a 后，为 root 帐号设置一

个安全密码。

示例如下：安装 GBase 8a 时，用户将 root 用户密码设置为空，此时，在第一次登录 GBase 8a 时，可以继续为 root 用户设置密码。

```
$ gbase. -uroot
GBase client 8.5.1.2 build 27952. Copyright (c) 2004-2013, GBase.  All Rights Reserved.

gbase> SET PASSWORD FOR root = PASSWORD('H133%_h');
Query OK, 0 rows affected
```

退出登录的命令为在 gbase>提示符下，键入\q。

```
gbase> \q
Bye
```

修改 root 的口令后，重新登录 GBase 8a。

```
$ gbase -uroot
Enter password:
GBase client 8.5.1.2 build 27952. Copyright (c) 2004-2013, GBase.  All Rights Reserved.

gbase >
```

3 DDL与DML

3.1 DDL

3.1.1数据库

CREATE DATABASE是用给定的名称来创建一个数据库。用户需要获得创建数据库的权限，才可以使用CREATE DATABASE。

如果用户没有指定IF NOT EXISTS并且该数据库存在，则产生一个错误。

创建数据库

```
CREATE DATABASE test;
CREATE DATABASE IF NOT EXISTS test;
```

删除数据库

```
DROP DATABASE test;
DROP DATABASE IF EXISTS test;
```

3.1.2表

用户可以通过CREATE TABLE命令在当前数据库创建一个指定名称的表。

GBase 8a支持两种表类型，普通表、和临时表。

表类型	特点
普通表	就是符合SQL标准，使用CREATE TABLE创建的表。
临时表	使用 TEMPORARY 关键字，临时表被限制在当前连接中，当连接关闭时，临时表会自动地删除。

3.1.2.1 普通表

GBase 8a 支持可以通过下面的语法创建普通表:

CREATE TABLE [IF NOT EXISTS] [*database_name.*]*tbl_name*

[(create_definition,...)] [table_options];

table_options:

[COMMENT 'comment_value']

删除普通表:

DROP TABLE [IF EXISTS] *tbl_name*;

创建普通表

```
CREATE TABLE t3(a int);
```

删除表

```
DROP TABLE t3;
```

3.1.2.2 临时表

GBase 8a 支持临时表，可以通过下面的语法创建临时表:

CREATE TEMPORARY TABLE [IF NOT EXISTS] [*database_name.*]*tbl_name*

[(create_definition,...)] [table_options];

table_options:

[COMMENT 'comment_value']

删除临时表:

DROP [TEMPORARY] TABLE [IF EXISTS] *tbl_name*;

临时表生命周期为会话级，当前会话连接结束后自动删除。

临时表支持普通表支持的所有相关操作，包括：DDL、DML、SHOW CREATE TABLE、DESC、SELECT 及 TEMPORARY TABLE 和普通表之间的关联等操作。

3.1.3视图

GBase 8a 禁止对视图进行 INSERT, DELETE, UPDATE 动作。

创建视图

```
CREATE VIEW v_t AS SELECT a,b FROM t;
CREATE OR REPLACE VIEW v_t AS SELECT b FROM t;
CREATE OR REPLACE VIEW v_t_1 AS SELECT a FROM t;
```

更改视图

```
ALTER VIEW v_t(a, b) AS SELECT * FROM t;
ALTER VIEW v_t(aa, bb) AS SELECT a,b FROM t;
```

删除视图

```
DROP VIEW IF EXISTS v_t, v_t_1;
```

3.1.4存储过程和函数

创建存储过程或函数。

下面介绍创建自定义存储过程的语法，在 GBase 8a 里，我们为用户提供了对存储过程的支持。

- 存储过程 (Procedure)：

CREATE PROCEDURE <*proc_name*>([<*parameter_1*>[,…] [,*parameter_n*]])

[characteristic ...]

下面介绍创建自定义函数的语法，在 GBase 8a 里，我们为用户提供了对自定义函数的支持。

- 函数（Function）：

CREATE FUNCTION <*func_name*>([<*parameter_1*>[,…] [,*parameter_n*]])

RETURNS type

[characteristic ...]

<函数定义>

<参数列表>:

[IN | OUT | INOUT] param_name type

而函数则没有[IN | OUT | INOUT]标记。

type:

Any valid GBASE data type

characteristic:

LANGUAGE SQL

| [NOT] DETERMINISTIC

| { CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA }

| SQL SECURITY { DEFINER | INVOKER }

| COMMENT 'string'

针对 CONTAINS SQL, NO SQL, READS SQL DATA, MODIFIES SQL DATA 的使用，需要注意的是，虽然可以一次性全部指定，但仅最后一个出现的关键字为有效的。

<过程/函数定义>:

这个语法参数是一系列的 SQL 语句的组合，其中包含一些数据操作以完成一定的功能逻辑。

修改存储过程或函数

```
ALTER {PROCEDURE | FUNCTION} sp_name [characteristic ...]

characteristic:

    { CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA }

  | SQL SECURITY { DEFINER | INVOKER }

  | COMMENT 'string'
```

删除存储过程或函数

```
DROP {PROCEDURE | FUNCTION} [IF EXISTS] sp_name
```

3.1.5 哈希索引

使用哈希索引能提高等值精确查询的性能。创建哈希索引时需明确指明创建哈希索引的列及该列所在的表。

哈希索引创建成功后，在数据存储介质上存放数据的位置会产生与索引相关的文件。

哈希索引有两种类型，一种是全局索引，称为全局哈希索引，基于表中整列建立索引；一种是局部索引，称局部哈希索引，基于表中第一个 DC 建立索引。

创建哈希索引后，基于索引列的等值查询的性能会提高，尤其是表中的数据量非常大的情况，在小数据量的情况下，哈希索引对性能的提升效果不明显。

支持创建哈希索引的数据类型，见下表：

可创建索引的数据类型
TINYINT
SAMLLINT
INT
BIGINT
FLOAT
DOUBLE

可创建索引的数据类型
DECIMAL
CHAR
VARCHAR
DATE
DATETIME
TIME

创建全局哈希索引

```
CREATE INDEX IDX_T1_A ON T1(A) USING HASH GLOBAL;
```

创建局部哈希索引

```
CREATE INDEX IDX_T2_A ON T2(A) USING HASH LOCAL;
```

删除索引

```
DROP INDEX IDX_T1_A ON T1;
DROP INDEX IDX_T2_A ON T2;
```

使用限制说明:

- 二进制类型的列不适合使用哈希索引；或者该列数据量较大，但 distinct 值较少时，也不适合使用哈希索引。
- 创建索引时，只能指定单列，不能指定多列创建联合索引。
- 同一表上不能创建相同名称的哈希索引，同一表的同一列上能且只能创建一个哈希索引。

3.2 DML

使用 GBase 8a 批量执行 INSERT、UPDATE、DELETE 等操作的性能，高于单条执行这些操作。

3.2.1 INSERT

GBase 8a 支持向表中插入数据。

INSERT [INTO] [database_name.]tbl_name [(col_name,...)]

VALUES ({expr | DEFAULT},...),(...),...;

或

INSERT [INTO] [database_name.]tbl_name [(col_name,...)]

SELECT ... FROM [*database_name.*] *tbl_name*;

创建表 t1 和 t2

```
CREATE TABLE t1(id int);
CREATE TABLE t2(id int);
```

示例：插入数据

```
gbase> INSERT INTO t1 VALUES(1),(2),(3),(4),(5),(6),(2),(3),(1);
Query OK, 9 rows affected
Records: 9  Duplicates: 0  Warnings: 0

gbase> INSERT INTO t2 SELECT * FROM t1;
Query OK, 9 rows affected
Records: 9  Duplicates: 0  Warnings: 0
```

3.2.2UPDATE

GBase 8a 支持更新表中数据。

语法：

UPDATE [database_name.]tbl_name

SET *col_name1* = expr1 [,*col_name2* = expr2 ...]

[WHERE where_definition]

创建表 t1

```
CREATE TABLE t1(id int);
```

表 t1 中插入数据

```
INSERT INTO t1 VALUES(1),(2),(3),(4),(5),(6),(2),(3),(1);
```

示例：更新表中数据

```
gbase> UPDATE t1 SET t0.id = t0.id+1 WHERE t0.id > 1;
Query OK, 7 rows affected
```

3.2.3 DELETE

GBase 8a 支持删除表中数据。

语法：

```
DELETE [FROM] [database_name.]tbl_name [tbl_alias]
    [WHERE where_definition]
```

创建表 t1

```
CREATE TABLE t(a int);
```

示例 1：标准删除语句。

```
gbase> INSERT INTO t VALUES(1),(2),(3),(4),(5);

gbase> DELETE FROM t WHERE a = 2;
Query OK, 1 row affected
```

示例 2：省略关键字 FROM 的删除。

```
gbase> INSERT INTO t VALUES(1),(2),(3),(4),(5);

gbase> DELETE t WHERE a = 2;
Query OK, 1 row affected
```

示例 3：按照别名删除，同时省略掉 FROM 关键字。

```
gbase> INSERT INTO t VALUES(1),(2),(3),(4),(5);
```

```
gbase> DELETE t tt WHERE tt.a=2;
Query OK, 1 row affected
```

4 审计日志

4.1 简介

审计日志最重要的作用，就是用于监控 SQL 执行性能，当一些 SQL 语句的执行用时大于 long_query_time 设定值的时候，审计日志便将这些 SQL 语句记录下来，方便用户针对这些执行效率低下的 SQL 语句进行分析，优化，改写，从而提高 SQL 语句的执行效率。

审计日志记录的主要内容如下：

- thread_id：线程号，同 processlist 中的 ID；
- taskid：每个 sql 任务编号；
- start_time：开始执行时间；
- end_time：结束执行时间；
- user_host：登录的用户和 IP，显示格式为：priv_user[user]@hostname[ip]；
- user：用户名；
- host_ip：用户登录端 IP 地址；
- query_time：执行语句所用时间；
- rows：行数；
- db：执行语句所针对的数据库；
- table_list：涉及表，格式：`<db>`.`<tb>`[,...]；
- sql_text：记录执行用时大于 long_query_time 设定值的 SQL 语句；
- sql_type：sql 类型，如 DDL、DML、DQL、OTHERS；
- sql_command：sql 命令类型，如 SELECT、UPDATE，INSERT，LOAD 等；

- algorithms：涉及的算子，比如JOIN、WHERE、GROUP、HAVING等；
- status：sql执行状态，如SUCCESS、FAILED、KILLED等；
- conn_type：用户登录方式(CAPI、ODBC、JDBC、ADO.NET、STUDIO)；

其中sql_command的取值范围为：INSERT、DELETE、UPDATE、LOAD、CREATE USER、CREATE DB、CREATE TABLE、CREATE VIEW、CREATE INDEX、CREATE PROCEDURE、CREATE FUNCTION、RENAME USER、ALTER DB、ALTER TABLE、ALTER PROCEDURE、ALTER FUNCTION、ALTER EVENT、DROP USER、DROP DB、DROP TABLE、DROP VIEW、DROP INDEX、DROP PROCEDURE、DROP FUNCTION、DROP EVENT、TRUNCATE、GRANT、REVOKE、SELECT和OTHERS。

algorithms的取值范围为：START_WITH、CONNECT_BY、JOIN、WHERE、GROUP、OLAP_GROUP、HAVING、OLAP_FUNC、DISTINCT、ORDER和LIMIT。若一个SQL涉及多个算子，各算子之间用逗号分割。

目前，用户每次查看的审计日志为登录节点机器上的日志内容。

4.2 配置参数

可以使用如下配置方式：

- 使用配置audit_log参数的方式开启或关闭审计日志。需要配置为全局级变量（默认为0,即关闭审计日志）。

  SET GLOBAL audit_log = 0;

  或

  SET GLOBAL audit_log = 1;

- 配置long_query_time参数，决定执行时间超过多少时长的操作会被记入审计日志（默认为10秒）。

  SET long_query_time = 5;（当前会话生效）

  或

SET GLOBAL long_query_time = 5; (如果将本参数设置为全局变量，在本配置生效后，所有新建会话生效)

- 配置审计日志输出方式，必须设成表形式（即设为 table，就能以表形式输出审计日志)。

  SET GLOBAL log_output = 'table';

4.3 存储方式

GBase 8a 使用如下方式存储这些内容:

审计日志的信息存储在系统表 gbase.audit_log 中。

4.4 使用约束

审计日志用于记录所有的 SQL 操作，对于包含结果集行数的统计操作，只涉及以下 4 种 DML 操作: SELECT、DELETE、INSERT 和 UPDATE。

清空 audit_log 时，需要使用 TRUNCATE audit_log 语句，一般情况下，建议每 2-3 个月可以 TRUNCATE 一次 audit_log 表，用于清除陈旧日志信息，避免数据过多，影响日后的分析。

4.5 使用示例

示例 1: 使用系统表查看审计日志。

```
$ gbase -uroot -p
Enter password:

GBase client 8.5.1.2 build 37185. Copyright (c) 2004-2013, GBase.  All Rights Reserved.

gbase> SET GLOBAL audit_log = 1;
Query OK, 0 rows affected
```

```
gbase> SET long_query_time = 0;
Query OK, 0 rows affected

gbase> SET GLOBAL log_output = 'table';
Query OK, 0 rows affected

gbase> DROP USER tzt;
Query OK, 0 rows affected

gbase> DROP DATABASE test;
Query OK, 1 row affected

gbase> CREATE USER tzt identified by 'tzt';
Query OK, 0 rows affected

gbase> GRANT ALL ON *.* TO tzt@'%';
Query OK, 0 rows affected

gbase> CREATE DATABASE test;
Query OK, 1 row affected

gbase>  USE test;
Query OK, 0 rows affected

gbase> CREATE TABLE t1(i int);
Query OK, 0 rows affected

gbase> INSERT INTO t1 VALUES (1),(2);
Query OK, 2 rows affected
Records: 2  Duplicates: 0  Warnings: 0

gbase> SELECT start_time,user_host,query_time,rows,LEFT(sql_text, 30),
conn_type FROM gbase.audit_log;
+--------------------+-----------------------------------+
| start_time         | user_host                         |
+--------------------+-----------------------------------+
```

```
| 2015-01-30 15:23:12 | root[root] @ localhost [127.0.0.1] |
| 2015-01-30 15:23:27 | root[root] @ localhost [127.0.0.1] |
| 2015-01-30 15:23:35 | root[root] @ localhost [127.0.0.1] |
| 2015-01-30 15:23:35 | root[root] @ localhost [127.0.0.1] |
| 2015-01-30 15:23:46 | root[root] @ localhost [127.0.0.1] |
| 2015-01-30 15:23:54 | root[root] @ localhost [127.0.0.1] |
| 2015-01-30 15:24:02 | root[root] @ localhost [127.0.0.1] |
| 2015-01-30 15:24:09 | root[root] @ localhost [127.0.0.1] |
| 2015-01-30 15:24:21 | root[root] @ localhost [127.0.0.1] |
| 2015-01-30 15:24:35 | root[root] @ localhost [127.0.0.1] |
+---------------------+------------------------------------+

+-----------------+------+-------------------------------+-----------+
| query_time      | rows | LEFT(sql_text, 30)            | conn_type |
+-----------------+------+-------------------------------+-----------+
| 00:00:00.090623 |    0 | (1)SET GLOBAL log_output = 'ta | CAPI      |
| 00:00:00.000953 |    0 | (1)DROP USER tzt               | CAPI      |
| 00:00:00.109080 |    0 | (1)DROP DATABASE test          | CAPI      |
| 00:00:00.002490 |    1 | (1)SELECT DATABASE()           | CAPI      |
| 00:00:00.000464 |    0 | (1)CREATE USER tzt identified  | CAPI      |
| 00:00:00.000404 |    0 | (1)GRANT ALL ON *.* TO tzt@'%' | CAPI      |
| 00:00:00.009732 |    0 | (1)CREATE DATABASE test        | CAPI      |
| 00:00:00.000102 |    0 | (1)USE test                    | CAPI      |
| 00:00:00.036431 |    0 | (1)CREATE TABLE t1(i int)      | CAPI      |
| 00:00:00.034718 |    2 | (1)INSERT INTO t1 VALUES (1),( | CAPI      |
+-----------------+------+-------------------------------+-----------+
10 rows in set

gbase> INSERT INTO t1 SELECT * FROM t1;
Query OK, 2 rows affected
Records: 2  Duplicates: 0  Warnings: 0

gbase> UPDATE t1 SET i = 3;
Query OK, 4 rows affected
Rows matched: 4  Changed: 4  Warnings: 0

gbase> DELETE FROM t1;
```

```
Query OK, 4 rows affected
```

```
gbase> SELECT start_time,user_host,query_time,rows, LEFT(sql_text, 30),
conn_type FROM gbase.audit_log;
+---------------------+-----------------------------------
| start_time          | user_host
+---------------------+-----------------------------------
| 2015-01-30 15:23:12 | root[root] @ localhost [127.0.0.1]
| 2015-01-30 15:23:27 | root[root] @ localhost [127.0.0.1]
| 2015-01-30 15:23:35 | root[root] @ localhost [127.0.0.1]
| 2015-01-30 15:23:35 | root[root] @ localhost [127.0.0.1]
| 2015-01-30 15:23:46 | root[root] @ localhost [127.0.0.1]
| 2015-01-30 15:23:54 | root[root] @ localhost [127.0.0.1]
| 2015-01-30 15:24:02 | root[root] @ localhost [127.0.0.1]
| 2015-01-30 15:24:09 | root[root] @ localhost [127.0.0.1]
| 2015-01-30 15:24:21 | root[root] @ localhost [127.0.0.1]
| 2015-01-30 15:24:35 | root[root] @ localhost [127.0.0.1]
| 2015-01-30 15:24:45 | root[root] @ localhost [127.0.0.1]
| 2015-01-30 16:20:54 | root[root] @ localhost [127.0.0.1]
| 2015-01-30 16:25:11 | root[root] @ localhost [127.0.0.1]
| 2015-01-30 16:25:29 | root[root] @ localhost [127.0.0.1]
+---------------------+-----------------------------------

+-----------------+------+-------------------------------+-----------+
| query_time      | rows | LEFT(sql_text, 30)            | conn_type |
+-----------------+------+-------------------------------+-----------+
| 00:00:00.090623 |    0 | (1)SET GLOBAL log_output = 'ta | CAPI      |
| 00:00:00.000953 |    0 | (1)DROP USER tzt               | CAPI      |
| 00:00:00.109080 |    0 | (1)DROP DATABASE test          | CAPI      |
| 00:00:00.002490 |    1 | (1)SELECT DATABASE()           | CAPI      |
| 00:00:00.000464 |    0 | (1)CREATE USER tzt identified  | CAPI      |
| 00:00:00.000404 |    0 | (1)GRANT ALL ON *.* TO tzt@'%' | CAPI      |
| 00:00:00.009732 |    0 | (1)CREATE DATABASE test        | CAPI      |
| 00:00:00.000102 |    0 | (1)USE test                    | CAPI      |
| 00:00:00.036431 |    0 | (1)CREATE TABLE t1(i int)      | CAPI      |
| 00:00:00.034718 |    2 | (1)INSERT INTO t1 VALUES (1),( | CAPI      |
| 00:00:00.001751 |   10 | (1)SELECT start_time,user_host | CAPI      |
```

```
| 00:00:00.029832 |    2 | (1)INSERT INTO t1 SELECT * FRO | CAPI       |
| 00:00:00.013553 |    4 | (1)UPDATE t1 SET i = 3         | CAPI       |
| 00:00:00.009300 |    4 | (1)DELETE FROM t1              | CAPI       |
+-----------------+------+--------------------------------+------------+
14 rows in set
```

5 权限管理

5.1 用户管理

用户可以使用 CREATE USER 语句创建一个新的 GBase 8a 帐号。

更多关于用户管理的语法说明，请参见《GBase 8a SQL 参考手册》的相关章节内容。

下面通过两个示例，介绍创建用户和更改用户密码的操作过程。

示例 1：使用超级用户 root 登录，创建一个用户 user1。

```
$ gbase -uroot -p
Enter password:

GBase client 8.5.1.2 build 27952. Copyright (c) 2004-2013, GBase.  All Rights Reserved.

gbase> CREATE USER user1;
Query OK, 0 rows affected
```

示例 2：使用超级用户 root 登录，修改用户 user1 的密码，然后使用 user1 的新密码登录。

```
$ gbase -uroot -p
Enter password:

GBase client 8.5.1.2 build 27952. Copyright (c) 2004-2013, GBase.  All Rights Reserved.

gbase> SET PASSWORD FOR user1 = PASSWORD('H133%_h');
Query OK, 0 rows affected
```

退出登录，使用 use1 用户登录，验证修改口令的正确性。

```
$ gbase -uuser1 -p
Enter password:

GBase client 8.5.1.2 build 27952. Copyright (c) 2004-2013, GBase.  All Rights Reserved.

gbase>
```

5.2 权限管理

在实际的项目中，建议用户规划好不同的数据库用户的职责，并给它赋予相应的操作权限，用以保证数据库的安全操作。

更多关于权限的的语法说明，请参见《GBase 8a SQL 参考手册》的相关章节内容。

下面通过两个示例，介绍权限管理的操作过程。

示例 1：使用超级用户 root，创建一个 user_general 用户，该用户具备 SELECT 操作的权限。

```
$ gbase -uroot -p
Enter password:

GBase client 8.5.1.2 build 27952. Copyright (c) 2004-2013, GBase.  All Rights Reserved.

gbase> CREATE USER user_general;
Query OK, 0 rows affected

gbase> SET PASSWORD FOR user_general = PASSWORD('H%897_@m');
Query OK, 0 rows affected
```

-- 只赋予 SELECT 权限。*.*代表所有数据库的数据库对象,例如：表，视图，存储过程。

```
gbase> GRANT SELECT ON *.* TO user_general;
Query OK, 0 rows affected

gbase> \q
Bye
```

-- 使用 user_general 登录数据库，验证其只具备 SELECT 权限。

-- 存在 test 数据库和一张 t1 表，这只是为演示示例提前创建完毕的。

```
$ gbase -uuser_general -p
Enter password:

GBase client 8.5.1.2 build 27952. Copyright (c) 2004-2013, GBase.  All Rights Reserved.

gbase> USE test;
Query OK, 0 rows affected

gbase> UPDATE t1 SET a = 11 WHERE a = 10;
ERROR 1142 (42000): UPDATE command denied to user 'user_general'@'localhost' for table 't1'

gbase> DELETE FROM t1;
ERROR 1142 (42000): DELETE command denied to user 'user_general'@'localhost' for table 't1'

gbase> SELECT * FROM t1;
+------+
| a    |
+------+
|    1 |
|    2 |
|    3 |
|    4 |
|    5 |
|    6 |
|    7 |
```

```
|     8 |
|     9 |
|    10 |
+-------+
10 rows in set
```

示例 2：使用超级用户 root，创建一个 user_admin 用户，该用户具备超级用户的权限，即全部的权限。

```
$ gbase -uroot -p
Enter password:

GBase client 8.5.1.2 build 27952. Copyright (c) 2004-2013, GBase.  All Rights Reserved.

gbase> CREATE USER user_admin;
Query OK, 0 rows affected

gbase> SET PASSWORD FOR user_admin = PASSWORD('H%897_@m');
Query OK, 0 rows affected

-- 赋予全部权限。*.*代表所有数据库的数据库对象,例如：表，视图，存储过程。
gbase> GRANT ALL ON *.* TO user_admin;
Query OK, 0 rows affected

gbase> \q
Bye

-- 使用 user_admin 登录数据库，验证其只具备 SELECT 权限。
-- 存在 test 数据库和一张 t1 表，这只是为演示示例提前创建完毕的。
$ gbase -uuser_admin -p
Enter password:
```

GBase client 8.5.1.2 build 27952. Copyright (c) 2004-2013, GBase. All Rights Reserved.

```
gbase> USE test;
Query OK, 0 rows affected

-- 可以SELECT数据。
gbase> SELECT * FROM t1;
+------+
| a    |
+------+
|    1 |
|    2 |
|    3 |
|    4 |
|    5 |
|    6 |
|    7 |
|    8 |
|    9 |
|   10 |
+------+
10 rows in set

-- 可以UPDATE数据。
gbase> UPDATE t1 SET a = 11 WHERE a = 10;
Query OK, 1 row affected

gbase> SELECT * FROM t1;
+------+
| a    |
+------+
|    1 |
|    2 |
|    3 |
|    4 |
```

```
|    5 |
|    6 |
|    7 |
|    8 |
|    9 |
|   11 |
+------+
10 rows in set
```

-- 可以DELETE数据。

```
gbase> DELETE FROM t1 WHERE a >= 5;
Query OK, 6 rows affected

gbase> SELECT * FROM t1;
+------+
| a    |
+------+
|    1 |
|    2 |
|    3 |
|    4 |
+------+
4 rows in set
```

-- 可以CREATE用户。

```
gbase> CREATE USER use_test;
Query OK, 0 rows affected
```

-- 可以DROP用户。

```
gbase> DROP USER use_test;
Query OK, 0 rows affected
```

6 图形化管理工具

6.1 企业管理器

GBase8a 企业管理器是 GBase8a 提供的一个图形化集成环境，用于访问、配置、控制、管理 GBase8a 的数据对象。它将一组多样化的图形工具与功能丰富的 SQL 脚本编辑器组合在一起,通过它数据库管理员能够以图形化方式和 SQL 命令行方式与 GBase8a Server 交互。

GBase8a 企业管理器使用 JDBC Driver 与 GBase8a 建立连接，通过 JDBC 接口与 GBase8a 进行交互。

使用 GBase8a 企业管理器可以完成如下工作:

- 管理多台 GBase8a 数据库服务器。
- 可视化管理数据库、表、索引、视图、存储过程和函数等数据对象。
- 可视化创建用户、赋予用户权限、编辑用户和删除用户。
- 可视化浏览、编辑表中的数据记录。
- 编辑、执行 SQL 脚本。
- 使用 SQL 模板编辑、执行 SQL 脚本。
- 可视化查看本地日志。

关于企业管理器的安装，使用的内容介绍，请查看《GBase 8a 管理工具手册》。

7 加载功能

在 V8.6.1.1 版本 GBase 8a 中，加载功能直接集成在 GBase 8a 内部，不需要额外部署外部加载工具。

与 V8.5.1.2 版本单机加载工具相比,新版加载工具具备如下一些特性和优点:

1) 提供面向用户的 SQL 接口，更符合用户的使用习惯;

2) 支持从通用数据服务器拉取数据,支持 ftp/http/hdfs/sftp 等多种协议;

3) 支持普通文本、gzip 压缩、snappy 压缩、lzo 压缩等多种格式数据文件;

4) 支持普通文本与定长文本的加载（format 3 和 format 4），并与 V8.5.1.2 版本格式兼容;

5) 支持错误数据溯源功能，可以准确定位错误数据在源文件中的位置;

7.1语法格式

GBase 8a 支持通过 SQL 接口进行数据加载。

语法:

```
LOAD DATA INFILE 'file_list' INTO TABLE tbl_name [options]


tbl_name:
  [database_name.]table_name
```

options:

[DATA_FORMAT *number* [HAVING LINES SEPARATOR]]

[NULL_VALUE '*string*']

[FIELDS

[TERMINATED BY '*string*']

[ENCLOSED BY '*string*']

[PRESERVE BLANKS]

[AUTOFILL]

[{LENGTH|DEFINER} '*string*']

[TABLE_FIELDS '*string*']

]

[LINES

[TERMINATED BY '*string*']

]

[MAX_BAD_RECORDS *number*]

[DATETIME FORMAT *format*]

[DATE FORMAT *format*]

[TIMESTAMP FORMAT *format*]

[TIME FORMAT *format*]

[TRACE *number*]

[TRACE_PATH '*string*']

[PARALLEL number]

[SKIP_BAD_FILE number]

[SET col_name = value[,...]]

参数说明:

FILE_LIST: 待加载文件列表。支持以本地文件方式和 URL 方式指定数据文件路径，以英文逗号(',')作为多个文件的分隔符。

scheme://[user:password@]host[:port]/path,

scheme://[user:password@]host[:port]/path

同时文件名、目录部分均支持使用通配符，具体如下表所示。默认对路径及文件进行匹配。当关闭目录、文件通配功能时，对于 SQL 中出现的通配符按 7.2 节的使用约束当做特殊字符处理。

通配符	含义	说明
*	匹配 0 或多个字符	a*b a 与 b 之间可以有任意长度的任意字符，也可以一个也没有，如 aabcb，axyzb，a012b，ab。
?	匹配任意一个字符	a?b a 与 b 之间必须也只能有一个字符，可以是任意字符，如 aab，abb，acb，a0b。
[list]	匹配 list 中的任意单一字符	a[xyz]b a 与 b 之间必须也只能有一个字符，但只能是 x 或 y 或 z，如：axb，ayb，azb。
[!list]	匹配 除 list 中的任意单一字符	a[!0-9]b a 与 b 之间必须也只能有一个字符，但不能是阿拉伯数字，如 axb，aab，a-b。
[c1-c2]	匹配 c1-c2 中的任意单一字符 如：[0-9] [a-z]	a[0-9]b 0 与 9 之间必须也只能有一个字符 如 a0b，a1b... a9b。
{string1,string2,...}(此功能不支持，{}作为普通字符处理)	匹配 sring1 或 string2 (或更多)其一字符串，string 也可以是通配符	a{abc,xyz,123}b a 与 b 之间只能是 abc 或 xyz 或 123 这三个字符串之一。(匹配三次，得到三次的结果，即最多可以有三个重复的结果) {1..3}.txt 这种情况就只可能是 1.txt 或 2.txt 或 3.txt a{1,2}b{3,4} 这样会依次匹配输出结果是 a1b3 a1b4 a2b3 a2b4

例如：

http://10.10.1.1/data/??????/*.tbl

OPTIONS：

DATA_FORMAT：用来指定使用哪种方式解析数据文件并加载。指定为 3，表示使用文本方式加载。指定为 4，表示使用定长方式加载。如果某列数据可能包含了行分隔符，则需要在 SQL 中输入'HAVING LINES SEPARATOR'子句。指定为 5，表示使用文本文件宽松模式，即数据源文件为包围符中含有换行符和包围符文本文件，或多列少列文本文件。

NULL_VALUE：用于指定空值字符，支持不超过 15 个任意字符的组合，参数值以引号包围，指定方式与字段包围符一样。

FIELDS 子段：

TERMINATED BY：用于指定字段分隔符，支持不超过 15 个任意字符的组合，支持任意字符，参数值以单引号包围，仅当使用文本方式加载时有效。可使用字符本身(仅限可见字符，如：″|″)、C 风格转义字符(如：″\a″)、\xhh 十六进制(如：″\xFF″)或 x''十六进制(如：″x'09'″)四种方式指定。例如：'|'，表示用|作为分隔字符。

ENCLOSED BY：用于指定字段包围符，支持任意单字符，参数值以单引号包围,仅当使用文本方式加载时有效。可使用字符本身(仅限可见字符,如:″|″)、C 风格转义字符(如：″\a″)、\xhh 十六进制(如：″\xFF″)或 x''十六进制(如：″x'09'″)四种方式指定。

PRESERVE BLANKS：用于设定是否保留字段内容两端的空格，默认不保留空格。

AUTOFILL：用于设定是否启用缺失列自动补齐功能，启用该参数后，对缺失分割符的字段数据按照 default 值或者 NULL 值进行加载，默认不自动补齐。

LENGTH|DEFINER：在使用定长模式加载时，用于设定字段长度的参数。定长格式数据导入时，设置每个字段的长度，有多个字段时，用逗号分隔。

TABLE_FIELDS：用于指定列加载，对于日期时间类型可以设置每一列的格

式。

SET：指定列值加载，加载系统将待加载文件和指定加载列值加载到集群系统的表中。输入的类型应为常量，包括字符串、整数值、浮点值和 NULL。

1、 支持指定所有列类型加载值；

2、 指定列值为常量值（包括 NULL），包括字符串（单引号包围）、十进制数值（10）、浮点值（10.9）、NULL、16 进制表示的字符串（0xbac3）、科学计数法（10e4）；

3、 支持多列同时指定加载值。最多可 SET 表列数-1 ，如果设置的列数与表定义中的列数一致将报错：Specified all fields .

4、 支持 format=3、format=4 以及 format=5；

使用限制说明：

1、 输入除常量值外的其他值，如列名、表达式等会报错，报错信息为 Column 'addr' should be const value；

2、 指定的列不能存在于 TABLE_FIELDS 中，否则报错；

3、 如果没有指定 AUTOFILL，指定值的列数+数据中列数之和必须等于表定义或者 TABLE_FIELDS（若指定了 TABLE_FIELDS）中的列数，否则会产生错误数据；如果指定了 AUTOFILL，则可以小于表定义的列数，缺少的列会自动补全。如果 TABLE_FIELDS 列数+SET 列数小于表定义的列数，能够正常加载，没有涉及的列按照 default 值补齐；

4、 同一列在 SQL 中不能重复指定，否则报错。

LINES 子段：

TERMINATED BY：行分隔符，支持任意单字符，参数值以引号包围。指定方式与包围符一样。默认行分隔符为'\n'。

MAX_BAD_RECORDS：在每次加载的任务中，设定错误数据行数的上限。当本次加载任务产生的错误数据行数大于 max_bad_records 设定的值时，加载任务回滚，加载工具报错退出。此参数取值范围为：[0, 4294967295]。此参数为可选参数，默认不限制错误条数。0 表示只要有错误数据就报错退出。

DATE FORMAT：用来指定 date 列类型的默认格式，默认为'%Y-%m-%d'。

DATETIME FORMAT：用来指定 datetime 列的默认格式，默认为'%Y-%m-%d %H:%i:%s'。

TIMESTAMP FORMAT：用来指定 timestamp 列的默认格式，默认为'%Y-%m-%d %H:%i:%s'。

TIME FORMAT：用来指定 time 列的默认格式，默认为'%H:%i:%s'。

TRACE：用来指示本次加载是否保存错误数据溯源。如果指定为 0，则不溯源。如果指定为 1，则进行溯源。默认值为 1。

溯源信息包括：错误数据所在的文件，所在行号。

TRACE_PATH：用来指定本次加载过程中产生的错误数据和日志存放路径。缺省情况下，错误数据和溯源日志记录在加载机节点的 /opt/gnode/log/gbase/loader_logs 下。

PARALLEL：用来控制加载并行度，取值范围[0,1024]。默认值为 0，表示并行度取值是线程池最大可用线程数。

SKIP_BAD_FILE：用来指定本次加载任务中是否忽略不存在或没有读取权限的数据文件继续加载。如果指定为 0，则加载报错终止。如果指定为 1，则忽略异常文件继续加载。默认值为 0。

7.2使用约束

1. 当使用定长加载模式时，必须指定 FIELDS LENGTH 的值。

2. 当使用文本加载模式时，NULL_VALUE 的默认值为'\N'。

3. 当使用文本加载方式时，行分隔符默认为'\n'。

4. 当使用文本加载方式时，如果某列数据可能包含了行分隔符，则需要在 SQL 中输入'HAVING LINES SEPARATOR'子句，同时需要输入入'ENCLOSED BY'指定字段包围符

5. 当在加载文件列表的 URL 中的用户名（user）、密码（password）、主机名（host）或文件路径（path）中包含下表所列的特殊字符时，对特殊字符需要用百分号编码代替。

URL: scheme://[user:password@]host[:port]/path

百分号编码 = % + 特殊字符的两字符十六进制值

特殊字符	百分号编码	说明
%	%25	要求百分号编码
:	%3A	标准 gen-delims 要求百分号编码
/	%2F	
?	%3F	
#	%23	
[	%5B	
]	%5D	
@	%40	
!	%21	标准 sub-delims 建议百分号编码
$	%24	
&	%26	
'	%27	
(	%28	
)	%29	
*	%2A	
+	%2B	
,	%2C	
;	%3B	

=	%3D	
\	%5C	标准中未列字符 建议百分号编码
"	%22	
<	%3C	
>	%3E	
空格	%20	

以下斜体字内容引用自标准 RFC-3986，虽然可能部分保留字符也不会引起 URI 解析问题，但仍建议对所有非保留字符外的字符均使用百分号编码。更详尽的 URI 编码规则请参阅标准 RFC-3986 文档。

2.1. Percent-Encoding

pct-encoded = "%" HEXDIG HEXDIG

2.2. Reserved Characters

reserved = gen-delims / sub-delims

gen-delims = ":" / "/" / "?" / "#" / "[" / "]" / "@"

sub-delims = "!" / "$" / "&" / "'" / "(" / ")"

/ "" / "+" / "," / ";" / "="*

2.3. Unreserved Characters

unreserved = ALPHA / DIGIT / "-" / "." / "_" / "~"

示例：FTP 用户名为 test，密码为 abc/def

错误：gbase> load data infile 'ftp://test:abc/def@192.168.0.1/data/*.tbl' into table t data_format 3;

正确：gbase> load data infile 'ftp://test:abc%2Fdef@192.168.0.1/data/*.tbl' into table t data_format 3;

6. 当在加载文件列表的 URL 中的用户名（user）、密码（password）、主机名（host）或文件路径（path）中包含下表所列的特殊字符时，对特殊字符需要用转义字符代替。

特殊字符	转义字符	说明
\	\\	要求用转义字符
’	\’	要求用转义字符

注：如果上表所列特殊字符已用百分号编码，则无需再用转义字符代替。

示例：FTP 用户名为 test，密码为 abc\def

错误：gbase> load data infile 'ftp://test:abc\def@192.168.0.1/data/*.tbl' into table t data_format 3;

正确：gbase> load data infile 'ftp://test:abc\\def@192.168.0.1/data/*.tbl' into table t data_format 3;

7. 宽松模式处理规则与文本方式加载处理规则不一致的有：

- ✓ 行分隔符、列分隔符、包围符仅支持单字符（单字节），指定多字符报错；
- ✓ 数据中有空值时，入库数据为 null，不是 default 值，设定 default 值对加载结果没有影响；
- ✓ 支持超宽列自动截断；
- ✓ 数据文件的包围符、列分隔符与设置的不一致，如果第一列为字符型，数据截断入库，后面的字段都为空值；如果第一列为数值型，则都为错误数据。
- ✓ 指定 auto_fill_column，在少列的时候自动补齐，无论列定义是否有 default 值，都会用 null 值补齐缺失列，而不是 default 值。

7.3使用示例

1. 以文本方式加载位于 FTP 服务器上的 a.tbl 文件，使用默认行分隔符和默认列分隔符。

LOAD DATA INFILE ’ftp://127.0.0.1/data/a.tbl’ INTO TABLE test.t

DATA_FORMAT 3;

2. 以文本方式加载位于FTP服务器上的a.tbl文件,使用默认行分隔符,并指定'|'为列分隔符。

LOAD DATA INFILE 'ftp://127.0.0.1/data/a.tbl' INTO TABLE test.t DATA_FORMAT 3 FIELDS TERMINATED BY '|';

3. 以文本方式加载位于FTP服务器上data目录下的所有扩展名为tbl文件,使用默认行分隔符和默认列分隔符。

LOAD DATA INFILE 'ftp://127.0.0.1/data/*.tbl' INTO TABLE test.t DATA_FORMAT 3;

4. 以文本方式加载位于FTP服务器上的a.tbl文件,字段内容中有默认行分隔符'\n',指定字段包围符'"'。

LOAD DATA INFILE 'ftp://127.0.0.1/data/a.tbl' INTO TABLE test.t DATA_FORMAT 3 HAVING LINES SEPARATOR FIELDS ENCLOSED BY '"';

5. 以文本方式加载位于FTP服务器上的a.tbl文件,指定空值'\N',保留字段两端空格,并自动补齐缺失列。

LOAD DATA INFILE 'ftp://127.0.0.1/data/a.tbl' INTO TABLE test.t DATA_FORMAT 3 NULL_VALUE '\\N' FIELDS PRESERVE BLANKS AUTOFILL;

6. 以文本方式加载位于FTP服务器上的a.tbl文件,指定行分隔符'|',使用默认列分隔符。

LOAD DATA INFILE 'ftp://127.0.0.1/data/a.tbl' INTO TABLE test.t DATA_FORMAT 3 LINES TERMINATED BY '|';

7. 以定长方式加载位于HTTP服务器上的b.tbl文件,使用默认行分隔符。

```
LOAD DATA INFILE 'http://127.0.0.1/data/b.tbl' INTO TABLE test.t
DATA_FORMAT 4 FIELDS LENGTH '20,11,10' TABLE_FIELDS
'name,mphone,birthday date "%m-%d-%Y"';
```

8. 以文本方式加载位于 HTTP 服务器上的 b.tbl.gz 压缩文件，使用默认行分隔符和默认列分隔符。

```
LOAD DATA INFILE 'ftp://127.0.0.1/data/b.tbl.gz' INTO TABLE test.t
DATA_FORMAT 3;
```

9. 以文本方式加载位于 HDFS 服务器上的 a.tbl.snappy 压缩文件，使用默认行分隔符和默认列分隔符。

```
LOAD DATA INFILE
'hdp://gbase:gbase@127.0.0.1:50070/data/a.tbl.snappy' INTO TABLE
test.t DATA_FORMAT 3;
```

10. 以文本方式加载位于 FTP 服务器上的 a.tbl 文件，指定错误数据溯源，并指定错误数据和日志存放路径。

```
LOAD DATA INFILE 'ftp://127.0.0.1/data/a.tbl' INTO TABLE test.t
DATA_FORMAT 3 TRACE 1 TRACE_PATH '/home/gbase/loader_logs'
```

11. 以文本方式加载位于 FTP 服务器上的 a.tbl 文件，指定默认日期时间格式。

```
LOAD DATA INFILE 'ftp://127.0.0.1/data/a.tbl' INTO TABLE test.t
DATA_FORMAT 3 DATETIME FORMAT '%H:%i:%s %Y-%m-%d';
```

12. 以文本方式加载位于 FTP 服务器上的 a.tbl 文件，指定最大允许错误条数。

```
LOAD DATA INFILE 'ftp://127.0.0.1/data/a.tbl' INTO TABLE test.t
DATA_FORMAT 3 MAX_BAD_RECORDS 0
```

13. 以文本方式加载位于FTP服务器上的test.tbl.lzo压缩文件,使用默认行分隔符和默认列分隔符。

```
LOAD DATA INFILE 'ftp://127.0.0.1/data/test.tbl.lzo' INTO TABLE
test.t DATA_FORMAT 3;
```

14. 以文本方式加载位于SFTP服务器上的a.tbl文件,使用默认行分隔符和默认列分隔符,用sftp://user:password@host/path方式指定SFTP服务器的用户名和密码。

```
LOAD DATA INFILE 'sftp://gbase:gbase@127.0.0.1/data/a.tbl' INTO
TABLE test.t DATA_FORMAT 3;
```

15. 以文本方式加载位于FTP服务器上的a.tbl文件,指定忽略异常文件。

```
LOAD DATA INFILE 'ftp://127.0.0.1/data/a.tbl' INTO TABLE test.t
DATA_FORMAT 3 SKIP_BAD_FILE 1;
```

16. 以文本方式加载位于FTP服务器上的a.tbl文件，使用默认行分隔符和默认列分隔符,用ftp://user:password@host/path方式指定FTP服务器的用户名和密码。

```
LOAD DATA INFILE 'ftp://gbase:gbase@127.0.0.1/data/a.tbl' INTO
TABLE test.t DATA_FORMAT 3;
```

17. 以文本方式加载位于HTTP服务器上的a.tbl文件,使用默认行分隔符和默认列分隔符,用http://user:password@host/path方式指定HTTP服务器的用户名和密码。

LOAD DATA INFILE 'http://gbase:gbase@127.0.0.1/data/a.tbl' INTO TABLE test.t DATA_FORMAT 3;

7.4数据服务器配置

V8.6.1.1 版本加载功能支持从通用数据服务器拉取数据，支持 ftp/http/hdfs 等多种协议。

以下简要描述 Red Hat Enterprise Linux 6.2 平台上 FTP、HTTP 和 HDFS 三种通用文件服务器的配置方法。

7.4.1FTP 服务器配置

使用 vsftp 搭建 FTP 服务器

1. 查看是否已安装 vsftpd

```
# rpm -qa vsftpd
vsftpd-2.2.2-6.el6_0.1.x86_64
```

2. 安装 vsftpd

```
# rpm -ivh vsftpd-2.2.2-6.el6_0.1.x86_64.rpm
```

3. 修改 FTP 服务器默认配置

```
# vim /etc/vsftpd/vsftpd.conf
```

```
# 表示允许匿名用户登录（默认为 YES）.
anonymous_enable=YES

# 表示允许本地用户登录（默认为 YES）.
local_enable=YES

# 表示开放对本地用户的写权限（默认 YES，如仅用作加载文件服务器，可改为 NO）.
write_enable=NO
```

```
# 设置本地用户的文件生成掩码（默认对本地用户的文件生成掩码是 077，可改为 022）
local_umask=022

# 允许匿名 FTP 用户上载文件（默认为 NO）.
#anon_upload_enable=YES

# 允许匿名 FTP 用户创建目录（默认为 NO）.
#anon_mkdir_write_enable=YES

# 启用 FTP 数据端口的连接请求（默认为 YES）.
connect_from_port_20=YES

# 使用 PAM 认证的配置文件名，文件位于/etc/pam.d 目录下
pam_service_name=vsftpd

# 是否使用 userlist 文件控制访问 FTP 服务器
userlist_enable=YES

# 设置禁止访问的文件或目录
#deny_file={*.mp3,*.mov,.private}

# 设置隐藏的文件或目录
#hide_file={*.mp3,.hidden,hide*,h?}

# 设置 FTP 被动模式开放端口范围
pasv_min_port=20001
pasv_max_port=21000

# 设置非匿名登录用户的主目录
#local_root=/var/ftp/pub

更多的配置可查看 vsftpd.conf 文档
# man vsftpd.conf
```

4. 配置允许或禁止访问 FTP 服务器的用户列表（可跳过）

```
# vim /etc/vsftpd/user_list
```

当/etc/vsftpd/vsftpd.conf 中配置如下时，禁止/etc/vsftpd/user_list 中的所有用户访问 FTP 服务器

userlist_enable=YES

userlist_deny=YES（缺省为 YES）

当/etc/vsftpd/vsftpd.conf 中配置如下时，允许/etc/vsftpd/user_list 中的所有用户访问 FTP 服务器

userlist_enable=YES

userlist_deny=NO

5. 配置禁止访问 FTP 服务器的用户列表（可跳过）

```
# vim /etc/vsftpd/ftpusers
```

6. 关闭 SELINUX 功能或更改其配置（两种方式二选一即可）

1）关闭 SELINUX 功能

```
# vim /etc/selinux/config

# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=disable
# SELINUXTYPE= can take one of these two values:
#     targeted - Targeted processes are protected,
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

重启或者执行

```
# setenforce 0
```

2）更改SELINUX配置

```
# setsebool ftp_home_dir 1
```

注：当用浏览器访问FTP服务器遇到“500 OOPS: cannot change directory:/home/...”时，可能为此问题。

7. 关闭或配置防火墙

1）关闭防火墙

停止防火墙服务

```
# service iptables stop
iptables: 清除防火墙规则:                                  [确定]
iptables: 将链设置为政策 ACCEPT: filter                    [确定]
iptables: 正在卸载模块:                                    [确定]
```

查看防火墙是否在开机时自动启动

```
# chkconfig --list iptables
iptables    0:关闭  1:关闭  2:启用  3:启用  4:启用  5:启用  6:关闭
```

禁止防火墙在开机时自动启动

```
# chkconfig iptables off
```

或

```
# chkconfig iptables off --level 2345
```

设置后防火墙在开机时自动启动状态

```
# chkconfig --list iptables
iptables    0:关闭  1:关闭  2:关闭  3:关闭  4:关闭  5:关闭  6:关闭
```

2）配置防火墙

设置默认规则

```
# iptables -A INPUT -j DROP（注：添加此条规则会阻止未处理的传入数据包，如在此规则之前未添加允许规则将会阻止远程连接）
# iptables -A FORWARD -j ACCEPT
```

开放 FTP 端口

```
# iptables -I INPUT -p tcp —dport 21 -j ACCEPT

# iptables -I OUTPUT -p tcp --sport 21 -j ACCEPT

# iptables -I INPUT -p tcp —dport 20 -j ACCEPT

# iptables -I OUTPUT -p tcp --sport 20 -j ACCEPT

# iptables -I INPUT -p tcp —dport 20001:21000 -j ACCEPT

# iptables -I OUTPUT -p tcp --sport 20001:21000 -j ACCEPT
```

保存防火墙设置

```
# iptables-save > /etc/sysconfig/iptables
```

8. 启动 vsftpd 服务并设置为开机启动项

```
# service vsftpd start
为 vsftpd 启动 vsftpd:                                    [确定]

# chkconfig vsftpd on
```

9. 复制文件到 FTP 目录

1) 如未设置 local_root=/var/ftp/pub 时，复制文件到/home/xxxx（用户的 home 目录）

2) 如已设置 local_root=/var/ftp/pub 时，复制文件到/var/ftp/pub

3) 如已设置 anonymous_enable=YES 时，复制文件到/var/ftp 或

/var/ftp/pub（匿名登录的主目录）

7.4.2HTTP 服务器配置

使用 apache 搭建 HTTP 文件服务器

1、安装 apr 和 httpd

```
# rpm -ivh
apr-1.3.9-3.el6_1.2.x86_64.rpm
apr-util-1.3.9-3.el6_0.1.x86_64.rpm apr-util-ldap-1.3.9-3.el6_0.1.x86_64.rpm
# rpm -ivh
httpd-2.2.15-15.el6.x86_64.rpm
httpd-manual-2.2.15-15.el6.noarch.rpm  httpd-tools-2.2.15-15.el6.x86_64.rpm
```

2、修改 HTTP 服务器默认配置

```
# vim /etc/httpd/conf/httpd.conf
```

修改服务器名称

```
# If your host doesn't have a registered DNS name, enter its IP address here.
# You will have to access it by its address anyway, and this will make
# redirections work in a sensible way.
#ServerName www.example.com:80
ServerName 192.168.10.114:80
```

修改以下位置，将其中的"/var/www/html"修改为"/var/www/files"

也可直接使用"/var/www/html"作为文件存储位置，跳过这一步

```
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
#DocumentRoot "/var/www/html"
```

```
DocumentRoot "/var/www/files"


#
# This should be changed to whatever you set DocumentRoot to.
#
#<Directory "/var/www/html">
<Directory "/var/www/files">
```

3、编辑默认欢迎页配置

```
# vim /etc/httpd/conf.d/welcome.conf
```

注释掉以下几行（默认如果 html 下没有默认页面将显示 403 错误页面）

```
#<LocationMatch "^/+$">
#    Options -Indexes
#    ErrorDocument 403 /error/noindex.html
#</LocationMatch>
```

4、关闭或配置防火墙

1）关闭防火墙

停止防火墙服务

```
# service iptables stop
iptables: 清除防火墙规则:                                  [确定]
iptables: 将链设置为政策 ACCEPT: filter                    [确定]
iptables: 正在卸载模块:                                    [确定]
```

查看防火墙是否在开机时自动启动

```
# chkconfig —list iptables
Iptables    0:关闭  1:关闭  2:启用  3:启用  4:启用  5:启用  6:关闭
```

禁止防火墙在开机时自动启动

```
# chkconfig iptables off
```

或

```
# chkconfig iptables off --level 2345
```

设置后防火墙在开机时自动启动状态

```
# chkconfig —list iptables
Iptables    0:关闭  1:关闭  2:关闭  3:关闭  4:关闭  5:关闭  6:关闭
```

2）配置防火墙

设置默认规则

```
# iptables -A INPUT -j DROP
# iptables -A FORWARD -j ACCEPT
```

开放 HTTP 端口

```
# iptables -I INPUT -p tcp -m tcp —dport 80 -j ACCEPT
# iptables -I OUTPUT -p tcp -m tcp --sport 80 -j ACCEPT
```

保存防火墙设置

```
# iptables-save > /etc/sysconfig/iptables
```

5、启动 httpd 服务并设置为开机启动项

```
# service httpd start
正在启动 httpd:                                          [确定]

# chkconfig httpd on
```

6、将数据文件复制到/var/www/files（或/var/www/html）下

7、用浏览器访问 http://192.168.10.114 即可看到文件列表(前面配置的 ServerName 192.168.10.114:80)

7.4.3HDFS 服务器配置

使用 Apache Hadoop 2.6.0 搭建 HDFS 服务器

1、 Hadoop 集群环境准备

操作系统用户：gbase

集群各节点间的 ssh 互信已建立。

集群已配置 C3 工具。

开源产品版本：

Apache Hadoop 2.6.0

JVM 1.6 或 1.7 版本

集群节点功能规划示例：

IP	主机名	功能
192.168.10.114	ch-10-114	NameNode, DataNode
192.168.10.115	ch-10-115	DataNode
192.168.10.116	ch-10-116	DataNode

2、 主机名配置

各节点的主机名需要正确配置，以 192.168.10.114 节点示例如下，其他节点直接拷贝该配置即可。

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1     localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.10.114 ch-10-114
192.168.10.115 ch-10-115
192.168.10.116 ch-10-116
```

注意：第一行如果配置成如下形式是错误的，安装完成后会出现 Hadoop 的 Datanode 无法连接 Namenode 的情况。

```
127.0.0.1          ch-10-114    localhost    localhost.localdomain    localhost4
```

```
localhost4.localdomain4
```

如果没有为 GBase 8a 配置 DNS 服务器，为保证 GBase 8a 可以解析 Hadoop 的 Namenode 和 Datanode 的主机名，需要配置/etc/hosts 文件，在其中加入如上 Hadoop 的 Namenode 和 Datanode 的 IP 地址和主机名映射。如未配置/etc/hosts 文件，执行加载 HDFS 服务器上文件时，会报类似“Couldn't resolve host name”字样的错误。

检查方法：

1）通过 jps 查看，发现 DataNode 已经启动，但是检查 DataNode 上的日志，发现 DataNode 在不断尝试连接 NameNode 节点的 9000 端口（HDFS 的 RPC 端口）。

2）在 NameNode 节点执行 netstat -an，看到如下信息：

```
$ netstat -an | grep 9000
tcp   0   0 127.0.0.1:9000     0.0.0.0:*         LISTEN
```

错误原因：TCP 监听的 IP 是 127.0.0.1，导致只有本机能够连接到 9000 端口。原因是 NameNode 的/etc/hosts 文件配置错误。

解决办法：去掉第一行的红色字体（ch-10-114），或者将第一行内容后置均可。

```
192.168.10.114 ch-10-114
192.168.10.115 ch-10-115
192.168.10.116 ch-10-116
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1     localhost localhost.localdomain localhost6 localhost6.localdomain6
```

重启 HDFS，再次用 netstat -an | grep 9000 查看，端口和 IP 正确。

```
$ netstat -an | grep 9000
tcp   0   0 192.168.10.114:9000     0.0.0.0:*         LISTEN
```

3、 目录规划

目录	用途
/home/gbase/bin	放置 Hadoop 生态系统，包括 Hadoop 等
/home/gbase/hdfs	放置 HDFS 文件，包括 tmp、name、data

添加环境变量${HADOOP_HOME}

```
$ echo "export HADOOP_HOME=/home/gbase/bin/Hadoop-2.6.0">> ~/.bashrc

$. ~/.bashrc
```

注：下文中的${HADOOP_HOME}指/home/gbase/bin/Hadoop-2.6.0

4、 准备 Hadoop 2.6.0

把 hadoop-2.6.0.tar.gz 解压到各节点的/home/gbase/bin。

```
$ tar xfz hadoop-2.6.0.tar.gz -C /home/gbase/bin
```

5、 配置 hadoop-env.sh

文件路径：${HADOOP_HOME}/etc/hadoop/hadoop-env.sh

```
$ cd ${HADOOP_HOME}

$ vi etc/hadoop/hadoop-env.sh
```

Name node 和 Data node 均按如下配置。

把 export JAVA_HOME=$JAVA_HOME 修改为：

export JAVA_HOME=/usr/lib/jvm/jre-1.6.0-openjdk.x86_64

把 export HADOOP_CONF_DIR=${HADOOP_CONF_DIR:-"/etc/hadoop"}修改为：

export HADOOP_CONF_DIR=/home/gbase/bin/hadoop-2.6.0/etc/hadoop

6、 配置 core-site.xml 文件

文件路径：${HADOOP_HOME}/etc/hadoop/core-site.xml

```
$ cd ${HADOOP_HOME}

$ vi etc/hadoop/core-site.xml
```

Name node和Data node均按如下配置

```
<configuration>
    <property>
        <name>fs.default.name</name>
        <value>hdfs://ch-10-114:9000</value>
    </property>
    <property>
        <name>hadoop.tmp.dir</name>
        <value>file:/home/gbase/hdfs/tmp</value>
    </property>
</configuration>
```

7、 配置hdfs-site.xml

文件路径：${HADOOP_HOME}/etc/hadoop/hdfs-site.xml

```
$ cd ${HADOOP_HOME}
$ vi etc/hadoop/hdfs-site.xml
```

Name node配置

```
<configuration>
    <property>
        <name>dfs.replication</name>
        <value>2</value>
    </property>
    <property>
        <name>dfs.name.dir</name>
        <value>file:/home/gbase/hdfs/name</value>
        <description>name node dir </description>
    </property>
    <property>
        <name>dfs.permissions</name>
```

```
            <value>false</value>
        </property>
    </configuration>
```

Data Node 配置

```
    <configuration>
        <property>
            <name>dfs.data.dir</name>
            <value>file:/home/gbase/hdfs/data</value>
            <description>data node dir</description>
        </property>
    </configuration>
```

8、 配置 Masters 和 Slaves

文件路径: ${HADOOP_HOME}/etc/hadoop/masters

${HADOOP_HOME}/etc/hadoop/slaves

只需要在 NameNode 节点配置即可。

```
$ cd ${HADOOP_HOME}
$ vi etc/hadoop/masters
```

${HADOOP_HOME}/etc/hadoop/masters 文件内容

```
    ch-10-114
```

```
$ cd ${HADOOP_HOME}
$ vi etc/hadoop/slaves
```

${HADOOP_HOME}/etc/hadoop/slaves 文件内容

```
    ch-10-114
    ch-10-115
    ch-10-116
```

9、 格式化NameNode

NameNode的格式化需要在启动HDFS之前进行。

```
$ cexec rm -fr /home/gbase/hdfs/*
$ cd ${HADOOP_HOME}
$ bin/hdfs namenode -format
```

10、 启动HDFS

```
$ cd ${HADOOP_HOME}
$ sbin/start-dfs.sh
```

启动完毕后，通过jps检查各节点的进程，如下即为正确启动了。

```
$ cexec jps
************************* test *************************
--------- 192.168.10.114---------
31318 SecondaryNameNode
31133 NameNode
31554 Jps
--------- 192.168.10.115---------
10835 DataNode
11000 Jps
--------- 192.168.10.116---------
10145 DataNode
10317 Jps
```

11、 停止HDFS

```
$ cd ${HADOOP_HOME}
$ sbin/stop-dfs.sh
```

7.4.4SFTP 服务器配置

SFTP 服务无需布署额外的软件包，开启 sshd 服务即可。

1. 查看是否已启动 sshd 服务

```
# service sshd status
openssh-daemon (pid  2243) is running...
```

2. 默认配置的目录访问。使用默认配置时，sshd 不限制用户的目录访问。用户通过 sftp 登录后，可以在有访问权限的任何目录间跳转。在这种情况下,以下加载语句的 URL 中的文件路径为系统的绝对路径:

load data infile

'sftp://gbase:gbase@192.168.10.114/opt/data/test.tbl'into table test.t

data_format 3;

3. 修改 sshd 默认配置。

当并发加载任务数较大时，会出现 sftp 文件加载失败的情况，此时可按以下方式修改 sshd 配置文件。

编辑/etc/ssh/sshd_config 文件

```
# vi /etc/ssh/sshd_config
```

按以下加粗字体内容修改配置文件

```
# MaxStartups 的值表示为“start:rate:full”，默认值为 10:30:100，当未认证连接数达到 start(10)时，新的连接尝试有“rate/100” (30%)的可能会被 sshd 拒绝，当未认证连接数达到 full(100)时，所有新的连接尝试都会被拒绝。
# 在最大并发加载任务数为 N 时 MaxStartups 的推荐值为(N+10):30:(N*2)
# MaxStartups 10:30:100
MaxStartups 20:30:100
```

出于安全原因，希望对 sftp 登录用户的访问权限进行限制，只能让用户在自己的 home 目录下活动。

启用 sshd 目录锁定功能需要使用到 chroot，openssh 4.8p1 以后都支持 chroot，可用以下命令检查当前系统的 openssh 版本。

```
# ssh -V
OpenSSH_5.3p1, OpenSSL 1.0.0-fips 29 Mar 2010
```

编辑/etc/ssh/sshd_config 文件

```
# vi /etc/ssh/sshd_config
```

按以下加粗字体内容修改配置文件

```
# override default of no subsystems
# 修改默认子系统为 internal-sftp
#Subsystem      sftp    /usr/libexec/openssh/sftp-server
Subsystem       sftp    internal-sftp

# Example of overriding settings on a per-user basis
# Match User sftp 表示以下规则仅匹配于名称为 sftp 的用户，如果需要匹配多个用户名，多个用户名间用逗号分隔。也可用 Match Group sftp 来匹配名称为 sftp 的组，同样如果需要匹配多个组，多个组名间用逗号分隔
Match User sftp
#       X11Forwarding no
#       AllowTcpForwarding no
# 强制执行进程内 sftp server，忽略~/.ssh/rc 文件中的命令
ForceCommand internal-sftp
# 用 chroot 将用户的根目录指定到%h，%h 代表用户的 home 目录，可选的参数还有%u，代表用户名
ChrootDirectory %h
```

注意：sftp 目录的权限配置要点：

1）由 ChrootDirectory 指定的目录开始一直向上到系统根目录为止的目录拥有者都只能是 root

2）由 ChrootDirectory 指定的目录开始一直向上到系统根目录为止都不可有群组写入的权限，即权限值不能高于 755

4. 配置完成后重启 sshd 服务

```
# service sshd restart
```

在已配置目录锁定的情况下,以下加载语句的 URL 中的文件路径为系统的相对路径，test.tbl 的绝对路径应为/home/gbase/opt/data/test.tbl

load data infile

'sftp://gbase:gbase@192.168.10.114/opt/data/test.tbl'into table test.t

```
data_format 3;
```

GBASE

南大通用数据技术股份有限公司
General Data Technology Co., Ltd.

微博二维码　微信二维码

技术支持热线：400-013-9696