# 概述

有时会需要将MySQL数据库中的数据导出到外部存储文件中，MySQL数据库中的数据可以导出成sql文本文件、xml文件或者html文件。同样，这些导出文件也可以导入到MySQL数据库中。

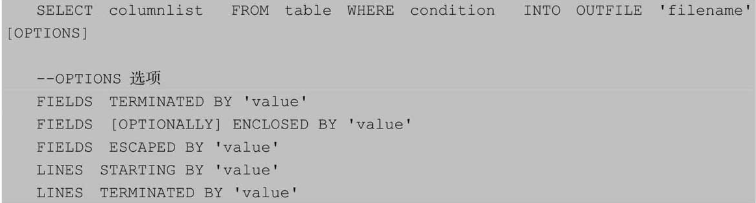
说明：导入导出主要用于异构和同构数据库之间的数据迁移，其作用于备份恢复不同（但是备份恢复的方法在同构数据库间可以起到数据迁移的作用）。

# 导入

## SELECT INTO OUTFILE

MySQL数据库导出数据时，允许使用包含导出定义的SELECT语句进行数据的导出操作。该文件被创建到服务器主机上，因此必须拥有文件写入权限（FILE权限）才能使用此语法。“SELECT...INTO OUTFILE 'filename'”形式的SELECT语句可以把被选择的行写入一个文件中，并且filename不能是一个已经存在的文件。

SELECT...INTO OUTFILE语句的基本格式如下：



可以看到SELECT columnlist FROM table WHERE condition为一个查询语句，查询结果返回满足指定条件的一条或多条记录；INTO OUTFILE语句的作用就是把前面SELECT语句查询出来的结果导出到名称为“filename”的外部文件中。[OPTIONS]为可选参数选项，OPTIONS部分的语法包括FIELDS和LINES子句，其可能的取值有：

●　FIELDS　TERMINATED BY 'value'：设置字段之间的分隔字符，可以为单个或多个字符，默认情况下为制表符‘\t’。

●　FIELDS　[OPTIONALLY] ENCLOSED BY 'value'：设置字段的包围字符，只能为单个字符，若使用了OPTIONALLY则只有CHAR和VERCHAR等字符数据字段被包括。

●　FIELDS　ESCAPED BY 'value'：设置如何写入或读取特殊字符，只能为单个字符，即设置转义字符，默认值为‘\’。

●　LINES　STARTING BY 'value'：设置每行数据开头的字符，可以为单个或多个字符，默认情况下不使用任何字符。

●　LINES　TERMINATED BY 'value'：设置每行数据结尾的字符，可以为单个或多个字符，默认值为‘\n’。

FIELDS和LINES两个子句都是自选的，但是如果两个都被指定了，FIELDS必须位于LINES的前面。

SELECT...INTO OUTFILE语句可以非常快速地把一个表转储到服务器上。如果想要在服务器主机之外的部分客户主机上创建结果文件，不能使用SELECT...INTO OUTFILE。在这种情况下，应该在客户主机上使用比如“MySQL –e "SELECT ..." > file\_name”的命令来生成文件。

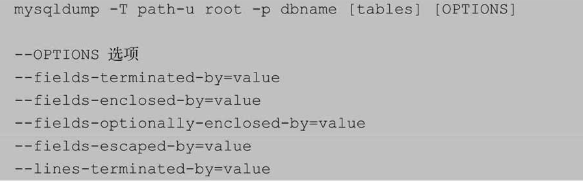
SELECT...INTO OUTFILE是LOAD DATA INFILE的补语。用于语句的OPTIONS部分的语法包括部分FIELDS和LINES子句，这些子句与LOAD DATA INFILE语句同时使用。

## mysqldump

除了使用SELECT… INTO OUTFILE语句导出文本文件之外，还可以使用MySQLdump。MySQLdump工具不仅可以将数据导出为包含CREATE、INSERT的sql文件，也可以导出为纯文本文件。

MySQLdump创建一个包含创建表的CREATE TABLE语句的tablename.sql文件和一个包含其数据的tablename.txt文件。

MySQLdump导出文本文件的基本语法格式如下：



只有指定了-T参数才可以导出纯文本文件；path表示导出数据的目录；tables为指定要导出的表名称，如果不指定，将导出数据库dbname中所有的表；[OPTIONS]为可选参数选项，这些选项需要结合-T选项使用。使用OPTIONS常见的取值有：

●　--fields-terminated-by=value：设置字段之间的分隔字符，可以为单个或多个字符，默认情况下为制表符“\t”。

●　--fields-enclosed-by=value：设置字段的包围字符。

●　--fields-optionally-enclosed-by=value：设置字段的包围字符，只能为单个字符，只能包括CHAR和VERCHAR等字符数据字段。

●　--fields-escaped-by=value：控制如何写入或读取特殊字符，只能为单个字符，即设置转义字符，默认值为反斜线“\”。

●　--lines-terminated-by=value：设置每行数据结尾的字符，可以为单个或多个字符，默认值为“\n”。

**提示：**

与SELECT…INTO OUTFILE语句中的OPTIONS各个参数设置不同，这里OPTIONS各个选项等号后面的value值不要用引号括起来。

## mysql

MySQL是一个功能丰富的工具命令，使用MySQL还可以在命令行模式下执行SQL指令，将查询结果导入到文本文件中。相比MySQLdump，MySQL工具导出的结果可读性更强。

如果MySQL服务器是单独的机器，用户是在一个client上进行操作，用户要把数据结果导入到client机器上。可以使用MySQL -e语句。

使用MySQL导出数据文本文件语句的基本格式如下：



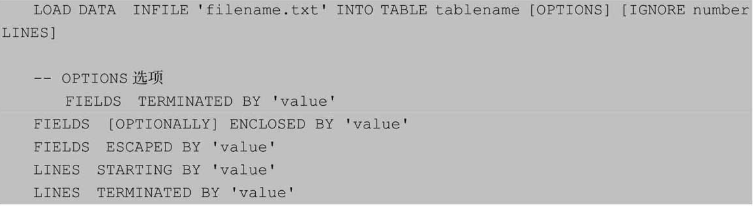
该命令使用--execute选项，表示执行该选项后面的语句并退出，后面的语句必须用双引号括起来，dbname为要导出的数据库名称；导出的文件中不同列之间使用制表符分隔，第1行包含了各个字段的名称。

# 导出

## LOAD DATA FROM

MySQL允许将数据导出到外部文件，也可以从外部文件导入数据。MySQL提供了一些导入数据的工具，包括LOAD DATA语句、source命令和mysql命令。LOAD DATA INFILE语句用于高速地从一个文本文件中读取行，并装入一个表中。文件名称必须为文字字符串。

LOAD DATA语句的基本格式如下：



可以看到LOAD DATA语句中，关键字INFILE后面的filename文件为导入数据的来源；tablename表示待导入的数据表名称；[OPTIONS]为可选参数选项，OPTIONS部分的语法包括FIELDS和LINES子句，其可能的取值有：

●　FIELDS　TERMINATED BY 'value'：设置字段之间的分隔字符，可以为单个或多个字符，默认情况下为制表符“\t”。

●　FIELDS　[OPTIONALLY] ENCLOSED BY 'value'：设置字段的包围字符，只能为单个字符。如果使用了OPTIONALLY，则只有CHAR和VERCHAR等字符数据字段被包括。

●　FIELDS　ESCAPED BY 'value'：控制如何写入或读取特殊字符，只能为单个字符，即设置转义字符，默认值为“\”。

●　LINES　STARTING BY 'value'：设置每行数据开头的字符，可以为单个或多个字符，默认情况下不使用任何字符。

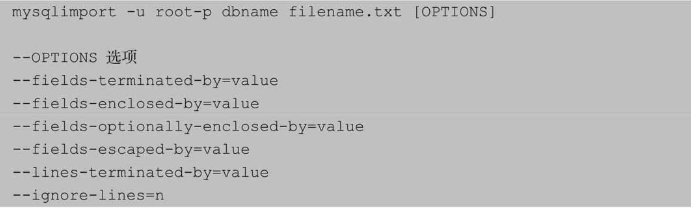
●　LINES　TERMINATED BY 'value'：设置每行数据结尾的字符，可以为单个或多个字符，默认值为“\n”。

IGNORE number LINES选项表示忽略文件开始处的行数，number表示忽略的行数。执行LOADDATA语句需要FILE权限。

## mysqlimport

使用MySQLimport可以导入文本文件，并且不需要登录MySQL客户端。MySQLimport命令提供许多与LOAD DATA INFILE语句相同的功能，大多数选项直接对应LOAD DATA INFILE子句。使用MySQLimport语句需要指定所需的选项、导入的数据库名称以及导入的数据文件的路径和名称。

MySQLimport命令的基本语法格式如下：



dbname为导入的表所在的数据库名称。注意，MySQLimport命令不指定导入数据库的表名称，数据表的名称由导入文件名称确定，即文件名作为表名，导入数据之前该表必须存在。[OPTIONS]为可选参数选项，其常见的取值有：

●　--fields-terminated-by= 'value'：设置字段之间的分隔字符，可以为单个或多个字符，默认情况下为制表符“\t”。

●　--fields-enclosed-by= 'value'：设置字段的包围字符。

●　--fields-optionally-enclosed-by= 'value'：设置字段的包围字符，只能为单个字符，包括CHAR和VERCHAR等字符数据字段。

●　--fields-escaped-by= 'value'：控制如何写入或读取特殊字符，只能为单个字符，即设置转义字符，默认值为反斜线“\”。

●　--lines-terminated-by= 'value'：设置每行数据结尾的字符，可以为单个或多个字符，默认值为“\n”。

●　--ignore-lines=n：忽视数据文件的前n行。

# 分布式数据库实践

## OceanBase

## TiDB

## TDSQL

## GoldenDB

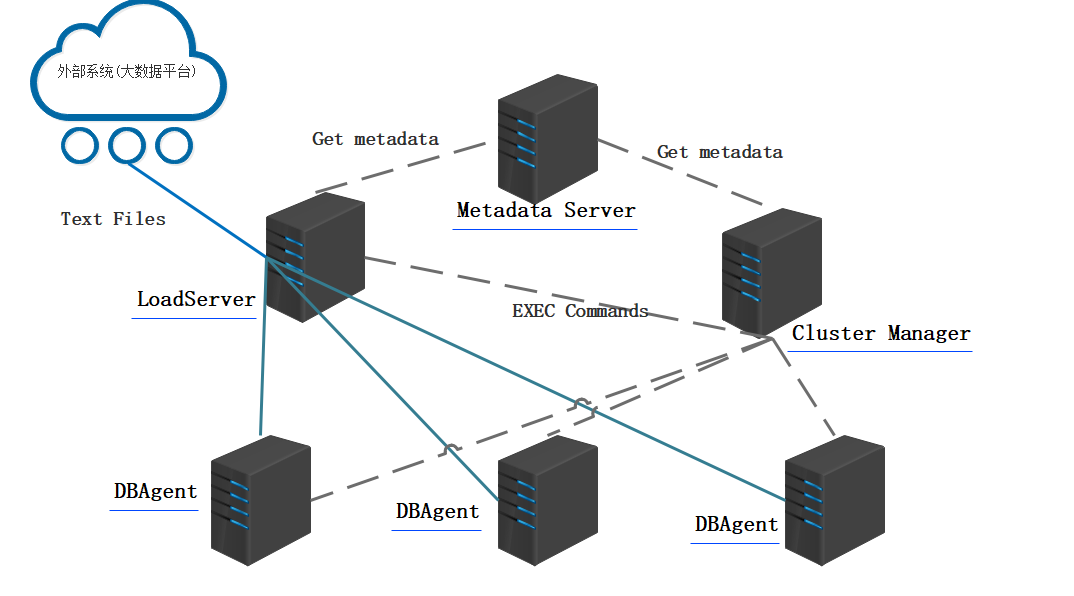
### 背景

支持分布式数据库批量数据导入，应用场景如：异构数据库之间的数据迁移，例如将oracle数据库数据导出为数据文件，可以直接使用该工具导入到GoldenDB数据库中。

支持分布式数据库批量数据导出，应用场景除了异构数据库之间的数据迁移，同时应用为数据逻辑备份，应用数据分析、统计、处理等。

### 典型组网

分布式数据库批量导入导出系统典型组网如图所示，包括如下模块：LDS、CM、DBAgent、MDS。



LDS：数据导入导出服务器，负责处理用户dbtool客户端命令，执行数据导入导出任务。

CM：数据库集群管理中心，负责转发LDS命令到DBAgent模块，接收汇总DBAgent命令响应并回复LDS模块。

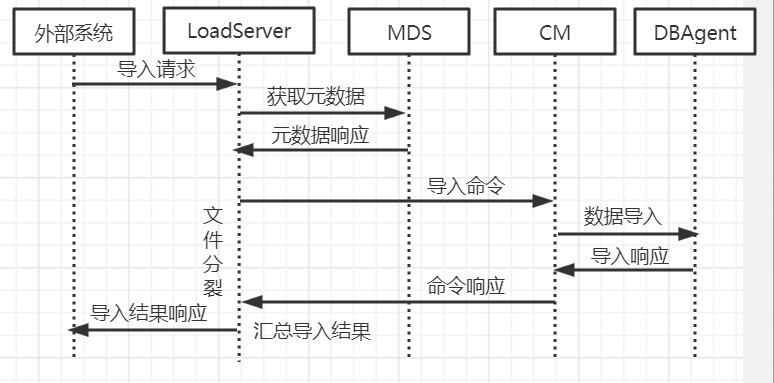
DBAgent：数据库监控代理程序，负责连接数据库节点，执行LDS服务器下发的导入导出命令。

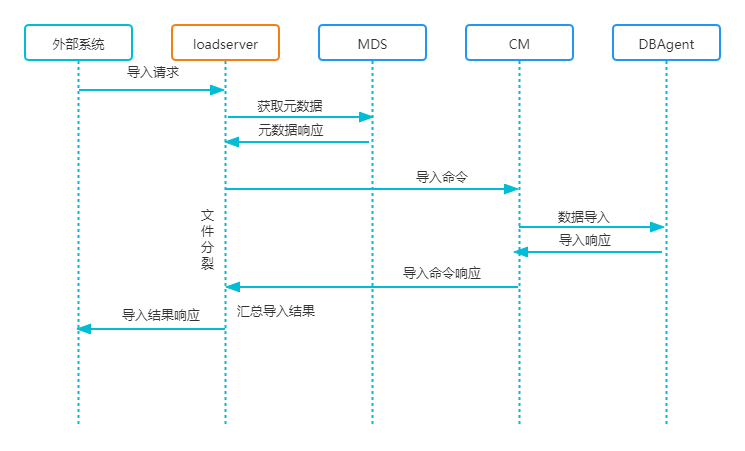
MDS：数据库元数据服务器，负责向LDS服务器提供数据库表的元数据信息，负责提供鉴权服务。

### 流程分析

#### 导入流程

大致流程：接收导入命令请求->SQL命令语法校验->获取元数据->拆分数据文件->下载拆分数据文件->执行导入命令->删除拆分数据文件





具体流程：

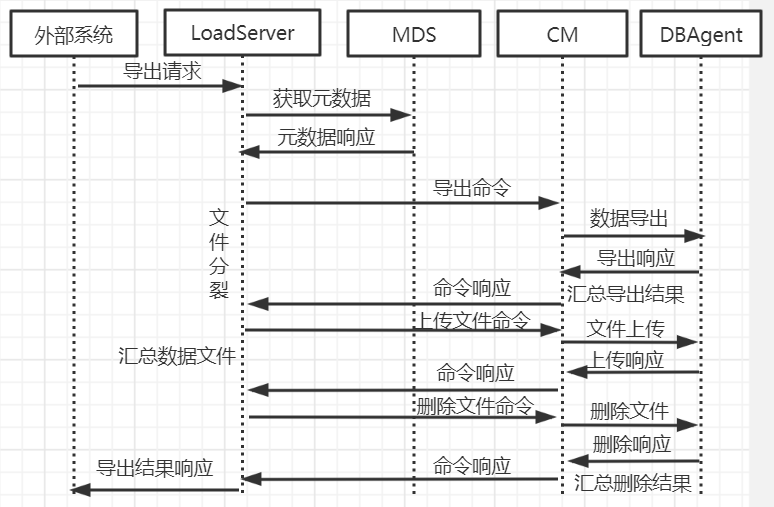
1. 用户使用dbtool客户端，向LDS服务器发送导入请求，LDS解析命令正确后，根据clusterid、database和table name去MDS查询该表的元数据信息，用于获取表结构定义和数据分布信息；
2. LDS使用这些信息（加上数据文件描述信息.frm）来识别数据文件（datafilename）中的每个数据行字段，将各个字段转换成内存Filed对象，将数据行转换成record对象；
3. LDS使用数据分布信息来确定每个数据行应该拆分到哪个DBGroup中，写入该拆分数据文件；
4. LDS请求CM通知各个DBAgent下面管理的DBGroup的拆分数据文件；

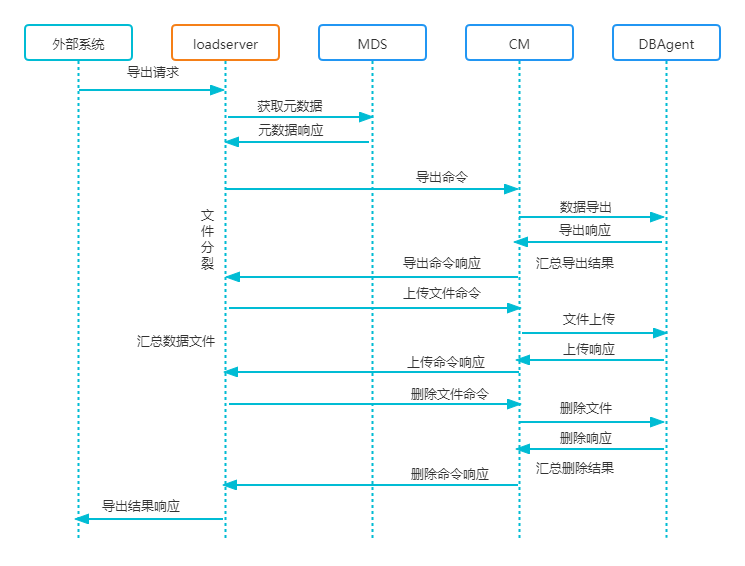
即，LDS服务器本地分裂数据文件，当分裂文件达到配置的文件下发数目时向CM发送指令。

1. 下载成功之后，LDS再请求CM去通知各个 DBAgent执行真正的LOAD DATA INFILE命令；
2. LOAD DATA INFILE命令执行成功后，再请求CM去通知各个DBAgent删除拆分文件；
3. 汇总结果并通知LOAD Client

#### 导出流程

大致流程：接收导出命令请求->SQL命令语法校验->获取元数据->执行导出命令->上传数据文件->汇总数据文件->删除导出数据文件





LDS接收到导出命令后，具体操作：

1. LDS根据clusterid、database和table name去MDS查询该表的元数据信息，用于获取表结构定义和数据分布信息；
2. LDS使用这些信息来判断应该到哪些DBGroup上执行SELECT INTO OUTFILE命令，并根据数据文件描述符拼接成完整的SELECT INTO OUTFILE命令字符串；
3. LDS把命令字符串发送给CM，请求它去通知各个DBAgent执行该SQL命令；
4. SQL命令执行成功之后，LDS请求CM去通知各个DBAgent上传导出来的数据文件到LDS服务器中；
5. 汇总执行结果，如果全部成功则合并这些数据文件；
6. LDS请求CM通知各个DBAgent删除数据文件；
7. 汇总结果并通知LOAD Client。

### 性能

导出性能：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 类型 | 数据量 | 数据文件大小 | 时间 | 备注 |
| LDS导出 | 4亿 | 152G | 1069s | 4分片range |
| LDS导出 | 4亿 | 152G | 1082s | 4分片hash |

导入性能：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 类型 | 数据量 | 数据文件大小 | 时间 | 备注 |
| LDS导入 | 4亿 | 152G | 4927s | 4分片range |
| LDS导入 | 4亿 | 152G | 4883s | 4分片hash |