数据库里表太多？快速获取某张表的表定义语句有妙招

当数据库含有几万张表，又需要快速获取到某个表的 DDL 时，如果通过管理工具一行行的找就太不方便了；而且作为一个熟练的技术人员，没有几个小诀窍提高工作效率怎么行？以下介绍两个能够快速轻松地完成本次任务的“小方法”，一起来看看吧。

今天的测试环境

操作系统：中标麒麟 6 64 位

数据库：　达梦数据库 v8.1

```
[dmdba@NeoKylin6-dm8 ~]$ uname -ra
Linux NeoKylin6-dm8 2.6.32-220.el6.x86_64 #1 SMP Tue Apr 10 05:55:10 EDT 2012 x86_64 x86_64 x86_64 GNU/Linux
[dmdba@NeoKylin6-dm8 ~]$ disql
disql V8.1.0.147-Build(2019.03.27-104581)ENT
用户名:sysdba
密码:

服务器[LOCALHOST:5236]:处于普通打开状态
登录使用时间: 6.776(毫秒)
SQL>
```

方法一：调用 DM 系统存储过程 SP_TABLEDEF

存储过程的定义

```
SQL> desc SP_TABLEDEF;

行号     NAME            TYPE$           IO DEF RT_TYPE
---------- --------------- --------------- -- --- -------
1          SP_TABLEDEF     PROC
2            SYS.SCHMNAME  VARCHAR(32767) IN
3            SYS.TABLENAME VARCHAR(32767) IN

已用时间: 7.060(毫秒). 执行号:24.
SQL>
```

功能说明：　获得表的定义

参数说明：

schname：模式名

tablename：表名

这个存储过程，直接 call 调用或者写在匿名块中调用均可。

比如，查询 DMHR 模式下 EMPLOYEE 表定义：

Call sp_tabledef('DMHR','EMPLOYEE');

```
SQL> call sp_tabledef('DMHR','EMPLOYEE');

行号     COLUMN_VALUE
---------- --------------------------------------------------------------------------------
1          CREATE TABLE "DMHR"."EMPLOYEE"  (  "EMPLOYEE_ID" INT NOT NULL,  "EMPLOYEE_NAME"
VARCHAR(20),  "IDENTITY_CARD" VARCHAR(18),  "EMAIL" VARCHAR(50) NOT NULL,  "PHONE_NUM" VARC
HAR(20),  "HIRE_DATE" DATE NOT NULL,  "JOB_ID" VARCHAR(10) NOT NULL,  "SALARY" INT,  "COMMI
SSION_PCT" INT,  "MANAGER_ID" INT,  "DEPARTMENT_ID" INT,  NOT CLUSTER PRIMARY KEY("EMPLOYEE
_ID"),  CONSTRAINT "EMP_EMAIL_UK" UNIQUE("EMAIL"),  CONSTRAINT "EMP_DEPT_FK" FOREIGN KEY("D
EPARTMENT_ID") REFERENCES "DMHR"."DEPARTMENT"("DEPARTMEN
2          T_ID"),  CONSTRAINT "EMP_JOB_FK" FOREIGN KEY("JOB_ID") REFERENCES "DMHR"."JOB"("
JOB_ID"),  CHECK("SALARY" > 0)) STORAGE(ON "DMHR", CLUSTERBTR) ;

已用时间: 49.173(毫秒). 执行号:25.
```

BEIGN

Sp_tabledef('DMHR','EMPLOYEE');

END;

```
SQL> BEGIN
   SP_TABLEDEF('DMHR','EMPLOYEE');
END;
2   3   4   /

行号     COLUMN_VALUE
---------- --------------------------------------------------------------------------------
1          CREATE TABLE "DMHR"."EMPLOYEE"  (  "EMPLOYEE_ID" INT NOT NULL,  "EMPLOYEE_NAME"
VARCHAR(20),  "IDENTITY_CARD" VARCHAR(18),  "EMAIL" VARCHAR(50) NOT NULL,  "PHONE_NUM" VARC
HAR(20),  "HIRE_DATE" DATE NOT NULL,  "JOB_ID" VARCHAR(10) NOT NULL,  "SALARY" INT,  "COMMI
SSION_PCT" INT,  "MANAGER_ID" INT,  "DEPARTMENT_ID" INT,  NOT CLUSTER PRIMARY KEY("EMPLOYEE
_ID"),  CONSTRAINT "EMP_EMAIL_UK" UNIQUE("EMAIL"),  CONSTRAINT "EMP_DEPT_FK" FOREIGN KEY("D
EPARTMENT_ID") REFERENCES "DMHR"."DEPARTMENT"("DEPARTMEN
2          T_ID"),  CONSTRAINT "EMP_JOB_FK" FOREIGN KEY("JOB_ID") REFERENCES "DMHR"."JOB"("
JOB_ID"),  CHECK("SALARY" > 0)) STORAGE(ON "DMHR", CLUSTERBTR) ;

已用时间: 0.570(毫秒). 执行号:47.
SQL>
```

这个存储过程是 DM 数据库独有的，Oracle 并没有这个存储过程。但是它存在一个缺点，只能查询表，不能查询其他类型的对象。

所以这里推荐的第二种方法，也是笔者最常用的查询表定义的方法。

方法二：DBMS_METADATA.GET_DDL

DBMS_METADATA 系统包包含了许多实用的函数，有兴趣可以查看 DM 官方文档慢慢了解，这里就不一一赘述了，介绍一下 GET_DDL

```
DBMS_METADATA.GET_DDL (
object_type      IN VARCHAR2,
name             IN VARCHAR2,
schema           IN VARCHAR2 DEFAULT NULL,
version          IN VARCHAR2 DEFAULT 'COMPATIBLE',
model            IN VARCHAR2 DEFAULT 'ORACLE',
transform        IN VARCHAR2 DEFAULT 'DDL')
RETURN CLOB;
```

我们在测试一下这个包：

```
disql V8.1.0.147-Build(2019.03.27-104581)ENT
SQL> SELECT DBMS_METADATA.GET_DDL('TABLE','EMPLOYEE','DMHR');

行号     DBMS_METADATA.GET_DDL('TABLE','EMPLOYEE','DMHR')
---------- --------------------------------------------------------------------------------
1          CREATE TABLE "DMHR"."EMPLOYEE"
(
"EMPLOYEE_ID" INT NOT NULL,
"EMPLOYEE_NAME" VARCHAR(20),
"IDENTITY_CARD" VARCHAR(18),
"EMAIL" VARCHAR(50) NOT NULL,
"PHONE_NUM" VARCHAR(20),
"HIRE_DATE" DATE NOT NULL,
"JOB_ID" VARCHAR(10) NOT NULL,
"SALARY" INT,
"COMMISSION_PCT" INT,
"MANAGER_ID" INT,
"DEPARTMENT_ID" INT,
NOT CLUSTER PRIMARY KEY("EMPLOYEE_ID"),
CONSTRAINT "EMP_EMAIL_UK" UNIQUE("EMAIL"),
CONSTRAINT "EMP_DEPT_FK" FOREIGN KEY("DEPARTMENT_ID") REFERENCES "DMHR"."DEPARTMENT"("DEPARTMEN
T_ID"),
CONSTRAINT "EMP_JOB_FK" FOREIGN KEY("JOB_ID") REFERENCES "DMHR"."JOB"("JOB_ID"),
CHECK("SALARY" > 0)) STORAGE(ON "DMHR", CLUSTERBTR) ;

已用时间: 62.733(毫秒). 执行号:79.
SQL>
```

GET_DDL 最先是在 Oracle 中使用，因为其功能非常强大：可以用于获取数据库

对表、视图、索引、全文索引、存储过程、函数、包、序列、同义词、约束、触发器等的 DDL 语句。于是 DM 数据库中借鉴 Oracle，做了很好的兼容，只需要知道对象类型、对象名称、对象用户就可以完成上述对象的 DDL 语句。

若碰到需要查询某对象 DDL 语句的场景，不妨试试看!