这篇文章可以和我前端时间写的《MySQL开发篇，存储引擎的选择真的很重要吗？》结合参考。

前言

还在为面试官问你处理有过百万数据或者千万级数据的经验而烦恼吗？
还在为想测试大量数据苦于没有环境而烦恼吗？
还在为有了环境，不知如何生成大量数据而烦恼吗？
看了这篇文章，这个小伙子有点东西啊！

正文

一、MySQL or MariaDB随机生成1000w数据

MySQL随机生成1kw条数据相对而言麻烦一点，没有rownum这个字段确实很蛋疼。
提供一个思路当然可以采取曾经火遍半边天的Python去随机生成大量数据，然后使用工具或者命令导入到MySQL数据库。
其次，还可以在Oracle生成大量数据，然后使用迁移工具转数据到MySQL
(当时，偷懒不想写函数和存储过程。我就是用这种方法的，利用DM数据迁移工具即可实现。)

01 使用工具SQLyog

创建表tolove

```
/** 创建表tolove **/
CREATE TABLE test.`tolove` (
  `ID` INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
  `GIRE_NAME` VARCHAR(64) COLLATE utf8_bin DEFAULT NULL,
  `GIRL_AGE` VARCHAR(64) COLLATE utf8_bin DEFAULT NULL,
  `CUP_SIZE` VARCHAR(10) COLLATE utf8_bin DEFAULT NULL
) ENGINE=INNODB DEFAULT CHARSET=utf8 COLLATE=utf8_bin
```

创建函数rand_numbe

```
/** 创建函数rand_number,生成随机数字 **/
DELIMITER $
CREATE FUNCTION rand_number() RETURNS INT
BEGIN
    DECLARE i INT DEFAULT 0;
    SET i= FLOOR(1+RAND()*100);
    RETURN i;
END $
DELIMITER $
```

创建函数rand_name

```
/** 创建函数rand_name,随机字符串 **/
DELIMITER $
CREATE FUNCTION rand_name(n INT) RETURNS VARCHAR(255)
BEGIN
    DECLARE chars_str VARCHAR(100) DEFAULT
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ';
```

```
    DECLARE return_str VARCHAR(255) DEFAULT '';
    DECLARE i INT DEFAULT 0;
    WHILE i < n DO
    SET return_str =
CONCAT(return_str,SUBSTRING(chars_str,FLOOR(1+RAND()*52),1));
    SET i = i+1;
    END WHILE;
    RETURN return_str;
END $
DELIMITER $
```

创建存储过程insert_tolove

```
/** 创建存储过程insert_tolove **/
DELIMITER $
CREATE PROCEDURE insert_tolove(IN max_num INT(10))
BEGIN
    DECLARE i INT DEFAULT 0;
    DECLARE EXIT HANDLER FOR SQLEXCEPTION ROLLBACK;
    START TRANSACTION;
    WHILE i< max_num DO
    INSERT INTO test.`tolove`(ID,GIRL_NAME,GIRL_AGE,CUP_SIZE)
VALUES(NULL,rand_name(5),rand_number(),NULL);
    SET i = i + 1;
    END WHILE;
COMMIT;
END $
DELIMITER $
```

测试验证

```
/** 测试验证 **/
SET GLOBAL event_scheduler=1;
COMMIT;

SELECT * FROM mysql.`event`;

SHOW FUNCTION STATUS;
UPDATE mysql.`proc` SET  Security_type='INVOKER';
```

调用存储过程insert_tolove

```
/** 调用存储过程insert_tolove **/
CALL insert_tolove(1000*10000);
SELECT COUNT(*) FROM test.`tolove` t WHERE t.girl_age='16';
```

验证插入语句

```
/** 验证是否输错 **/
INSERT INTO test.`tolove`(ID,GIRL_NAME,GIRL_AGE,CUP_SIZE)
VALUES(NULL,'1','1','A');
```

02 通过第三方工具迁移数据

DM8迁移工具DTS

```
/** 通过DM数据库迁移工具迁移的数据 **/
/** student 100w**/
SELECT COUNT(*) FROM test.`student`;
UPDATE test.`student` s SET s.`stu_age`='18' WHERE s.`stu_name` LIKE 'A%';
UPDATE test.`student` s SET s.`stu_sex`='女';

SELECT COUNT(*) FROM test.student t WHERE t.`stu_name` LIKE 'A%'

/** test 1000w **/
SELECT COUNT(*) FROM test.`test`;
```

二、Oracle生成1kw数据大表

01 使用工具plsql developer

```
/** Oracle11g R2 for Windows10 测试随机生成200w数据的表 **/
/** 此次测试对DM8数据库同样适用 **/
--创建表
CREATE TABLE test.student
(
    ID NUMBER not null primary key,
    STU_NAME VARCHAR2(60) not null,
    STU_AGE NUMBER(4,0) NOT NULL,
    STU_SEX VARCHAR2(2) not null
)
```

```
--学生表随机生成200w数据
insert into test.student
select
rownum,dbms_random.string('*',dbms_random.value(6,10)),dbms_random.value(14,16),
'女' from dual
connect by level<=2000000
```

```
select count(*) from test.student t where t.stu_age='16';
```

```
select t.*, t.rowid from TEST.STUDENT t where t.stu_age<16

update test.student set stuid=rownum where 1=1
--修改年龄随机14-16岁之间
update test.student set stu_age=dbms_random.value(14,16) where 1=1
```

提交

```
--提交
commit;
```

```
/** Oracle11g R2 for Windows10 测试随机生成1000w数据的大表 **/
/** 此次测试对DM8数据库同样适用 **/
/** 测试插入1kw数据 ---begin **/
--创建表
CREATE TABLE test.test
(
    id NUMBER not null primary key,
    stu_name NVARCHAR2(60) not null,
    score NUMBER(4,0) NOT NULL,
    createtime TIMESTAMP (6) not null
)
```

```
--模拟插入200w级数据(36s)
--模拟插入1kw数据出现提示connect by内存不足。也许是机器性能不够,也许是需要做参数调整。
--DM8直接改成10000000，毫无压力
insert into test.test
select
rownum,dbms_random.string('*',dbms_random.value(6,20)),dbms_random.value(0,20),
sysdate from dual
connect by level<=2000000
```

```
--统计(初始200w数据)
select count(*) from test.test;
```

```
--提交
commit;
```

```
-- 将test表结构以及数据复制到test01中
create table test.test01 as select * from test.test;
```

```
--执行4次，分分钟变出一张1kw级的数据大表，这个小伙子有点东西啊!
insert into test.test01 as select * from test.test
```

```
--统计
select count(distinct id) from test.test01
```

```
--统计(生成1000w数据完毕)
select count(*) from test.test01;
```

```
--优化id
update test.test01 set id=rownum where 1=1
--设置id为主键
ALTER TABLE test.test01 ADD CONSTRAINT constraint_test01 PRIMARY KEY (id);
/** 测试插入1kw数据 ---end **/
/** Oracle11g R2 for Windows10测试 **/
```

三、 DM8数据库生成1kw数据大表

01 使用工具，DM管理客户端

```
/**
1、dm8数据库创建用户test
2、创建表student
3、插入1000w测试数据
4、commit提交
5、count统计数据
**/
```

```
/** 创建学生表begin  **/
CREATE TABLE test.student
(
    ID NUMBER not null primary key,
    STU_NAME VARCHAR2(60) not null,
    STU_AGE NUMBER(4,0) not null,
    STU_SEX VARCHAR2(2) not null
)
```

```
--学生表随机生成1000w数据大约8s，测试插入1kw条数据花了90s左右
insert into test.student
select
rownum,dbms_random.string('*',dbms_random.value(6,10)),dbms_random.value(14,16),
'女' from dual
connect by level<=10000000
```

```
/** 统计数据 **/
select count(*) from test.student t where t.stu_age='16';
```

```
select t.*, t.rowid from test.student t where t.stu_age<16

update test.student set stuid=rownum where 1=1
--修改年龄随机14-16岁之间
update test.student set stu_age=dbms_random.value(14,16) where 1=1
```

```
--(1000w)
select count(*) from test.student;
--优化id
update test.student set id=rownum where 1=1
--设置id为主键
ALTER TABLE test.student ADD CONSTRAINT constraint_student PRIMARY KEY (id);
```

```
--提交
commit;
/** 创建学生表---end  **/
/** DM8 for Windows10测试 **/
```

```
/**DM8 for Windows10测试 **/
/** 测试插入1kw数据 ---begin **/
--创建表test
CREATE TABLE test.test
(
    id NUMBER not null primary key,
    stu_name NVARCHAR2(60) not null,
    score NUMBER(4,0) NOT NULL,
    createtime TIMESTAMP (6) not null
)
```

```
--学生表随机生成200w数据
insert into test.test
select
rownum,dbms_random.string('*',dbms_random.value(6,10)),dbms_random.value(14,16),
'女' from dual
connect by level<=2000000
```

```
/** 分次数创建1kw数据 **/
--创建test
create table test.test as select * from users.student;

--执行4次。每次插入200w，算上初始的200w，刚好1kw数据
insert into test.test as select * from users.student
--统计
select count(distinct id) from test.test
```

四、优化篇

目前只提供优化的思路。

01 Oracle11g

提供思路：优化SQL语句
1、建议：不使用"*"代表所有列名。
2、使用TRUNCATE代替DELETE
3、在确保完整的情况下多用commit语句
4、尽量减少表的查询次数
5、使用NOT EXISTS代替NOT IN
6、表连接优化
7、合理使用索引
8、避免全表扫描大表
9、优化器的使用：比如EXPLAIN PLAN执行计划
10、SQL调优顾问

02 MySQL

同样可以使用大家熟知的EXPLAIN执行计划

```
/** sakila是MySQL官网自己出的示例，类似于Oracle的scott以及
DM的（PERSON、PRODUCTION、PURCHASING、RESOURCES、SALES）**/

-- 新增唯一索引
alter table sakila.`customer` add unique index uk_email(email);
-- 执行计划，type=const
EXPLAIN SELECT * FROM  sakila.`customer`
sc WHERE sc.`email`='MARY.SMITH@sakilacustomer.org';
```

```
/** 列举其中的部分 **/
/**
type：ALL index range ref eq_ref const,system NULL,效率从左自有递增
    type=ALL:全表扫描
    type=index:索引全扫描
    type=range:索引范围扫描,常见于<、<=、>、>=、between、等操作符
    type=ref:使用非唯一索引扫描或者唯一索引的前缀扫描，返回匹配某个单独值得记录行
    type=eq_ref:类似ref,区别在于使用的索引是唯一索引,对于每个索引键值,表中只有一条记录匹配;
    type=eq_ref,简述：多表连接中使用primary key或者unique index作为关联条件。
    type=const,system:单表中最多有一个匹配行，查询起来非常迅速。所以这个匹配行中的其它列的值
可以
    被优化器在当前查询中当做常量处理，例如：根据主键primary key或者唯一索引unique index进行
的查询
    type=NULL:MySQL不用访问表或索引，直接就能得到结果。
    **/
```

03 DM8

参考Oracle数据库优化，国产达梦数据库和Oracle有很多相似之处，基本兼容的。

开心一刻

创作乐无边，学而思有境。**你会发现，自己的知识宝库越来越丰富。**好了，到此为止就是此篇文章的全部内容了，**能看到这里的都是帅哥靓妹啊**！！！善于总结，其乐不穷。**好记性不如烂笔头**，多收集自己第一次尝试的成果，收获也颇丰。

by 龙腾万里sky 原创不易，白嫖有瘾