常用ORACLE知识浅析

# 数据库是什么

ORACLE实例=内存（SGA+PGA）+进程（pmon、smon、lgwr、ckpt等）

ORACLE数据库= 数据文件+控制文件+联机重做日志文件

Q1：如何查看机器上起了哪些数据库，实例名是什么？

只要查看实例相关的进程即可，例如 ps –ef | grep pmon

Q2:数据库名、实例名、服务名等各种名字是什么？

DB\_NAME:数据库名，建库的时候起的名字

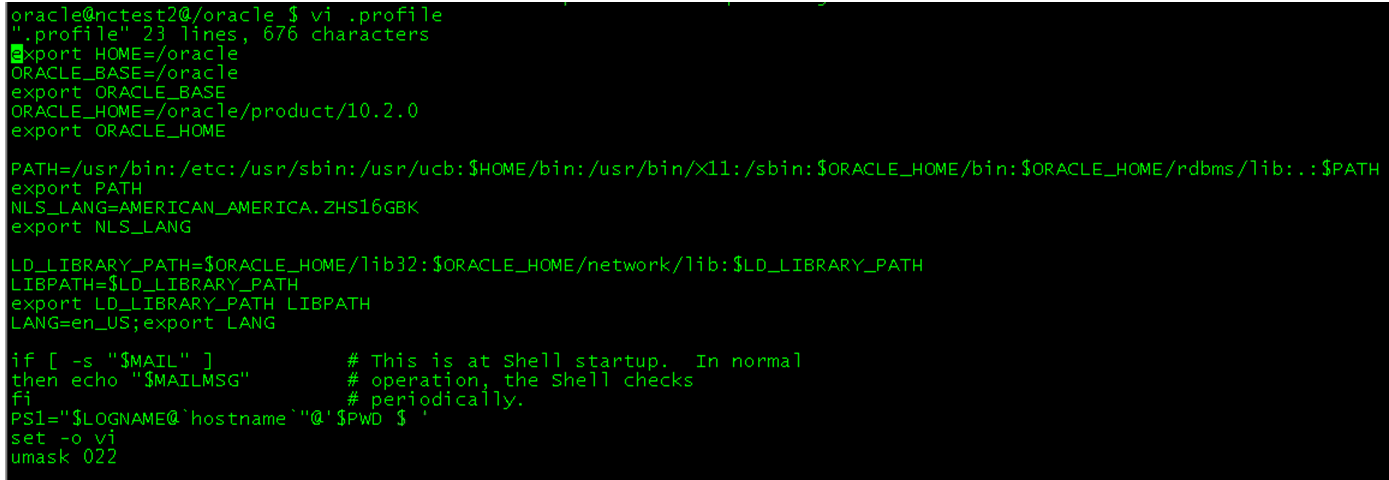
INSTANCE\_NAME:实例名，一般与数据库名相同，也可以不同（常见于RAC）

ORACLE\_SID:操作系统层面的实例名

SERVICE\_NAME:监听程序对外服务的名字，通过lsnrctl status 查看，如果没有显示配置listener.ora,一般是DB\_NAME + DB\_DOMAIN

# ORACLE环境相关

1. 环境变量



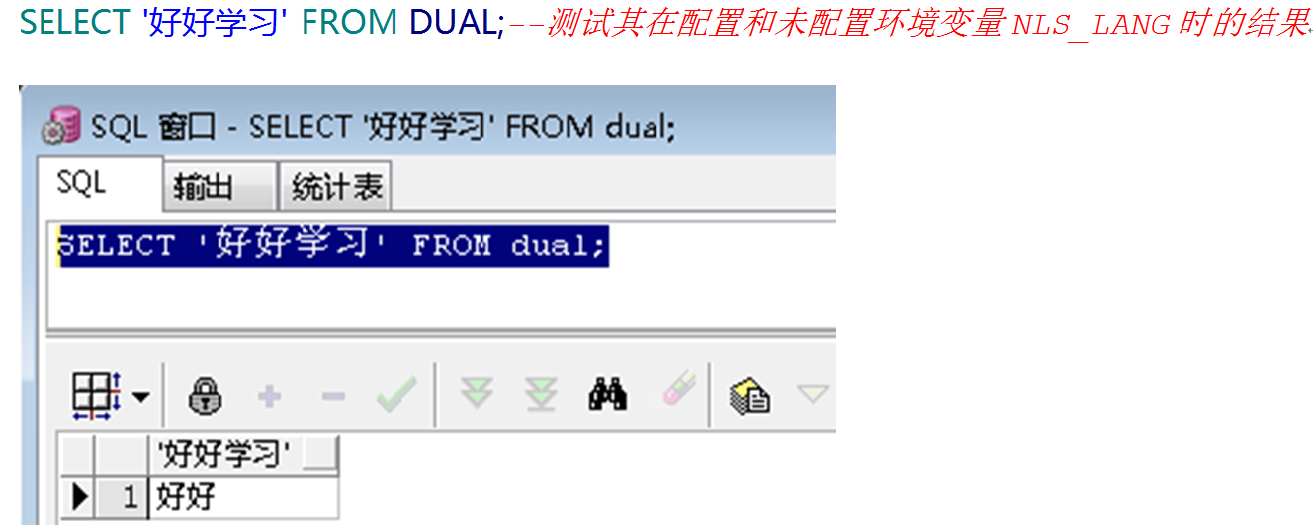
ORACLE\_BASE:ORACLE根目录

ORACLE\_HOME: ORACLE安装的产品目录

PATH:需要包含$ORACLE\_HOME/bin,sqlplus/expdp/impdp等ORACLE相关的可执行程序都在此目录下

NLS\_LANG:字符集，客户端的与数据库的要匹配，否则查询中文有问题

例：



1. 监听与连接

本机连接：sqlplus linc/unix;--不需要tnsnames.ora,只需要指定ORACLE\_SID环境变量即可

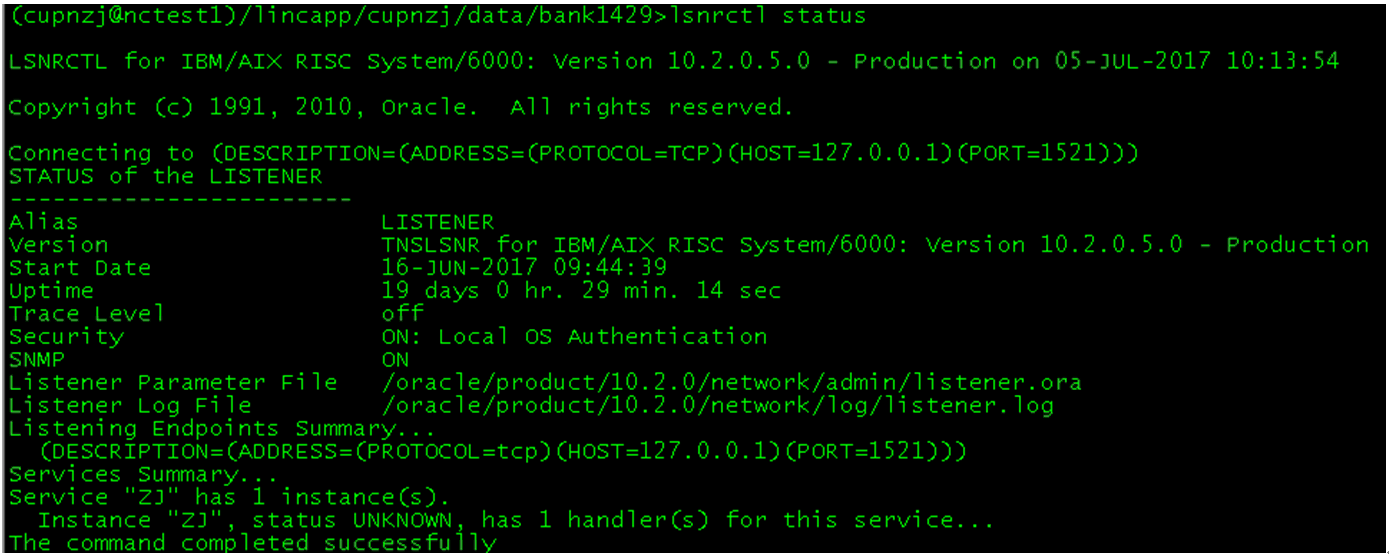
服务连接：sqlplus linc/unix@xxx;--不需要ORACLE\_SID，但需要配置tnsnames.ora和启动监听

sqlplus linc/unix @标识-指tnsnames配置的名字

sqlplus linc/unix @IP/实例名

查看启动实例： ps –ef | grep pmon

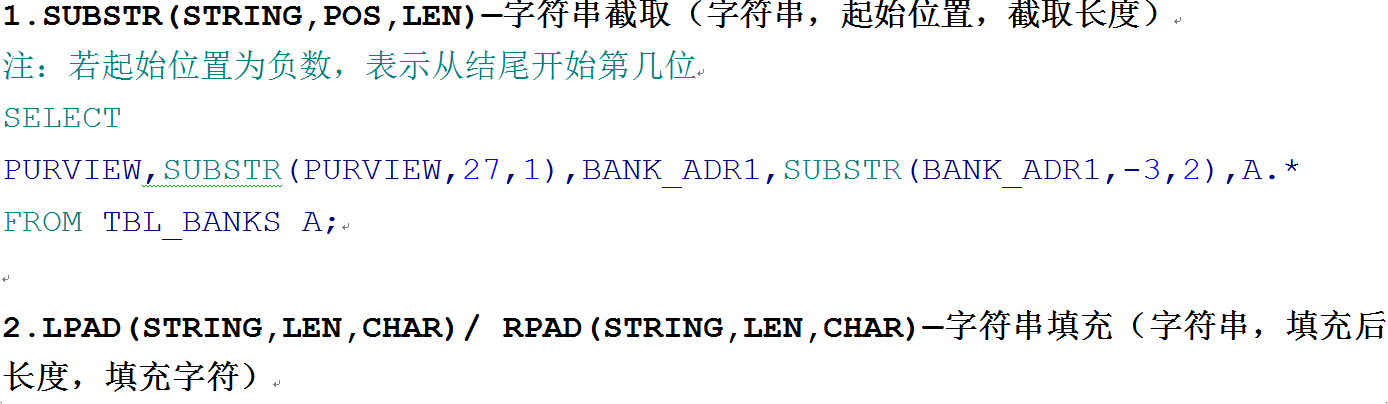
查看监听状态：lsnrctl status

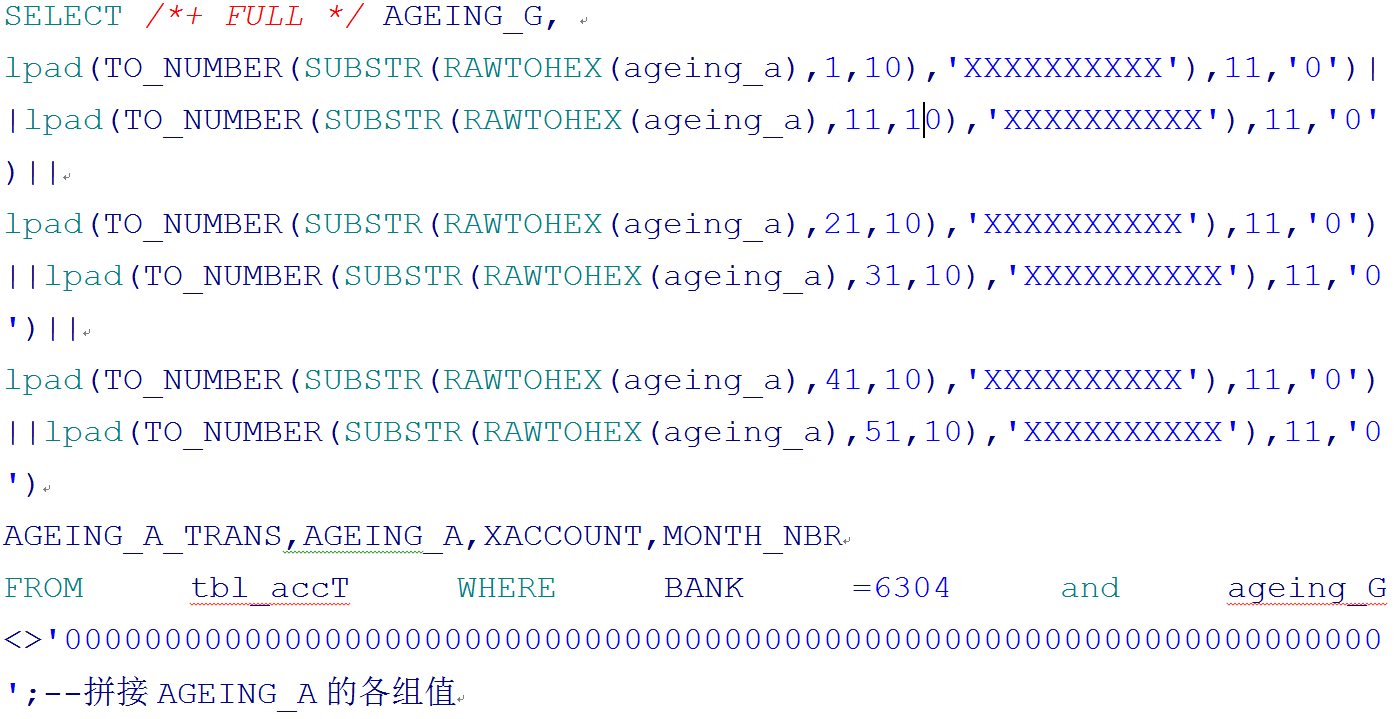


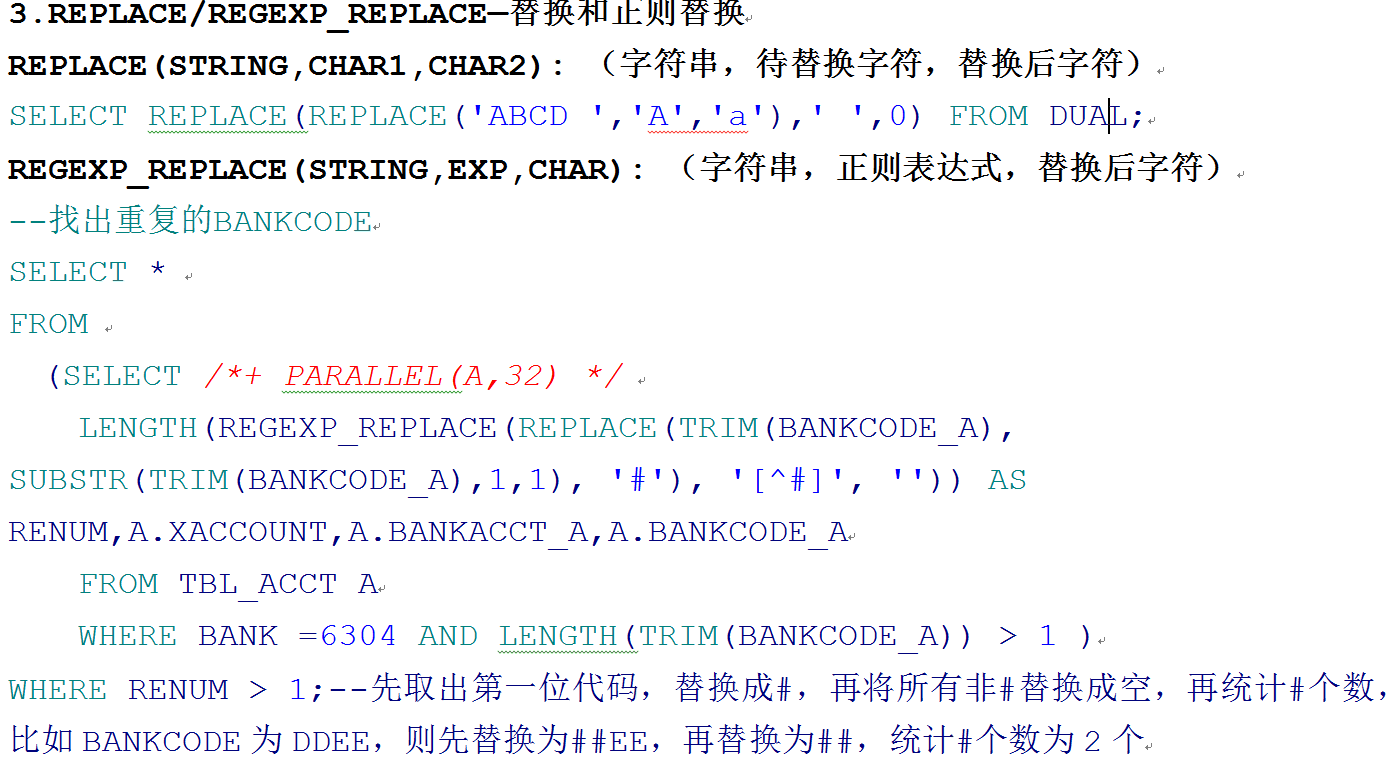
启停监听：lsnrctl start/stop

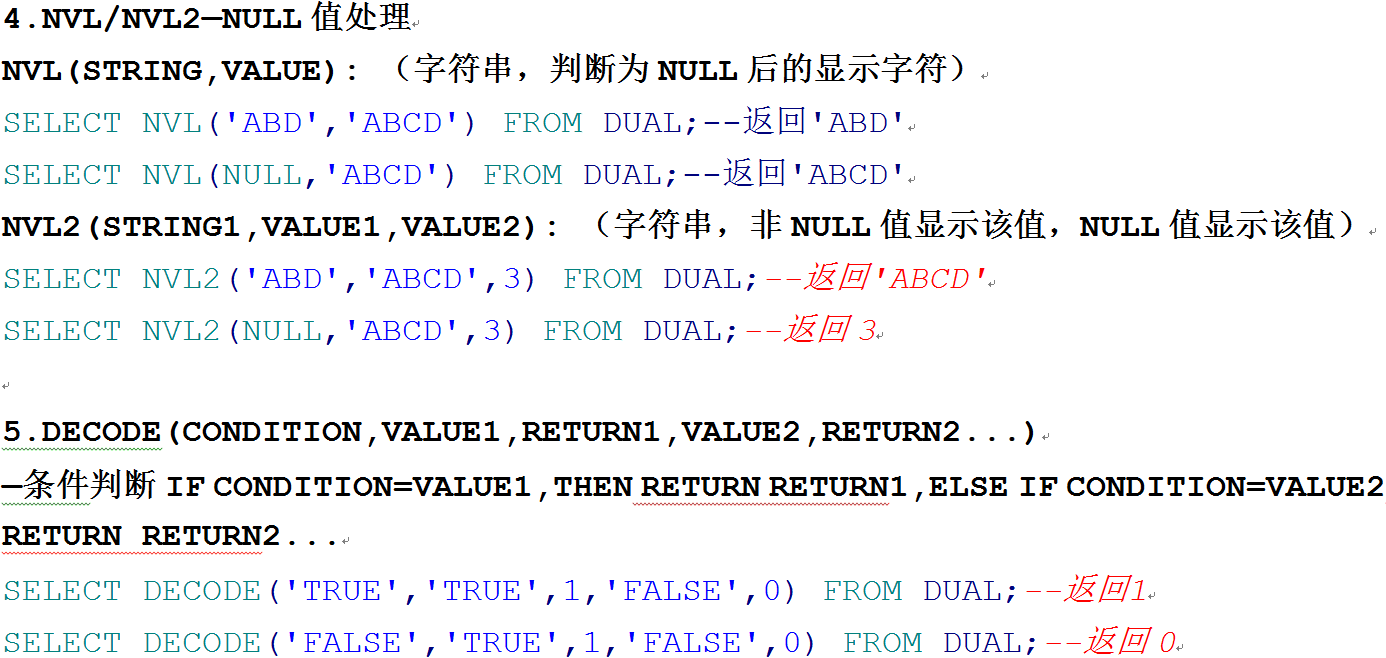
# 简单函数

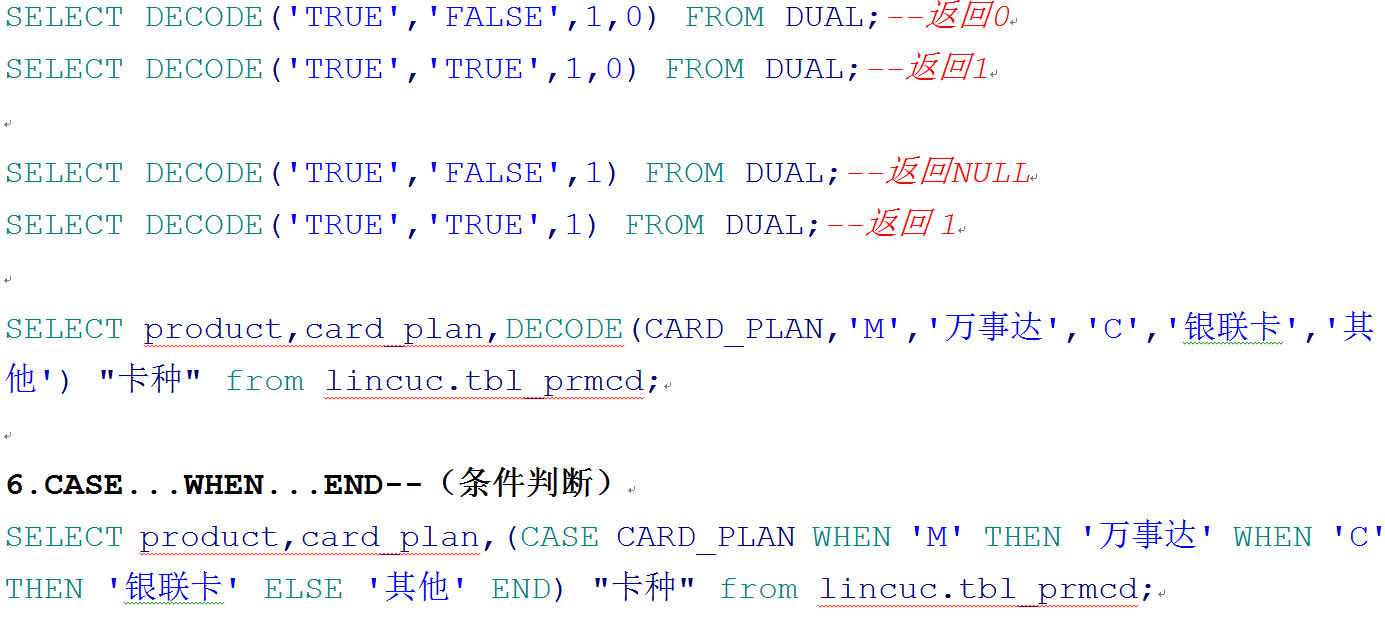
1. 常用函数





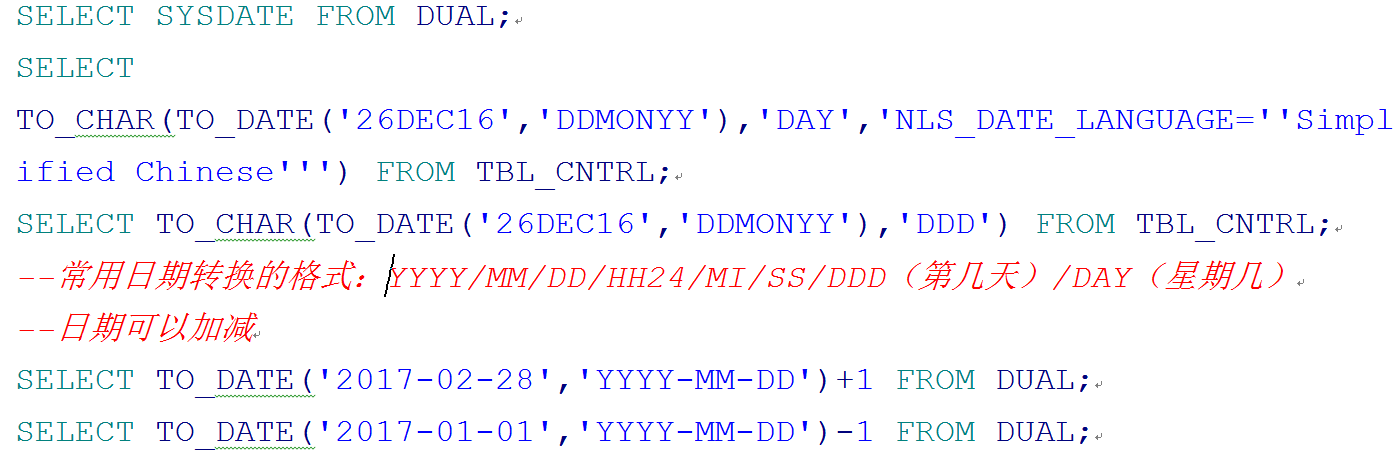






注：SUBSTR是截取个数 SUBSTRB是按字节截取

1. 日期装换



TO\_CHAR函数用法

日期转换格式 to\_char(date,’格式’)；

select to\_char(sysdate,'yyyy-MM-dd HH24:mi:ss') from dual;

数字格式转换 to\_char(number,’格式’)；

select to\_char(88877) from dual;

select to\_char(1234567890,'099999999999999')  from dual;

select to\_char(12345678,'999,999,999,999')  from dual;

select to\_char(123456,'99.999')  from dual;

select to\_char(1234567890,'999,999,999,999.9999')  from dual;

金钱格式转换：to\_char(salary,’$99,99’)；

select TO\_CHAR(123,'$99,999.9') from dual;

进制转换：10进制转换为16进制

select to\_char(4567,'xxxx') from dual;

select to\_char(123,'xxx') from dual;

注意：TO\_NUMBER()函数用法与TO\_CHAR相反

1. CAST转换

CAST()函数可以进行数据类型的转换，参数有两部分，源值和目标数据类型，中间用AS关键字分隔。

用法1：转换列或值

语法：cast( 列名/值 as 数据类型 )

转换列：select cast(empno as varchar2(10)) as empno from emp;

转换值：SELECT CAST('123' AS int) as result from dual;

用法2：转换一个集合

语法：cast( multiset(查询语句) as 数据类型 )

1. REPLACE（）函数的用法/LTRIM（）函数的用法
2. coalesce()函数

返回参数中的第一个非空表达式（从左向右依次类推）

例：select coalesce(null,2,3) from dual; // Return 2

select coalesce(null,null,3) from dual; // Return 3

select coalesce(1,2,3) from dual; // Return 1

实际应用如：coalesce(sum(decimal(amount,13,2)),0.00)用来统计金额，如果为空就显示为0

# SQL必知必会

1. **用通配符进行过滤**

通配符搜索只能用于文本字段（字符串），非文本字段不能使用通配符搜索。

百分号（%）通配符用法：

Select prod\_id,prod\_name from product where prod\_name like ‘Fissh%’; 通配符%看起来像是可以匹配任何东西，但是有个例外就是NULL。子句prod\_name like ‘%’不会匹配产品名称为NULL的行。

下划线（\_）通配符用法：

Select prod\_id,prod\_name from product where prod\_name like ‘F\_s\_h%’;下划线通配符只匹配单个字符（DB2不支持\_通配符）

方括号（[ ]）通配符：

Select prod\_id,prod\_name from product where prod\_name like ‘[FQ]%’;方括号通配符用来指定一个字符集，它必须匹配指定位置的一个字符

注意：通配符很有用，但是这种功能是有代价的，即通配符搜索一般比其他搜索要耗费时间，能用其他方式搜索的尽量避免通配符

1. **创建计算字段**

计算字段是运行时在SELECT语句内创建的

字段拼接用法：

Access和SQL Server使用+号, DB2、Oracle使用||。

Select vend\_name || ‘(‘ || vend\_country || ’)’ as vend\_title from Vendors ;

注意：大多数DBMS都支持RTRIM( ) (去掉字符串右边的空格)、LTRIM( )(去掉字符串左边的空格)以及TRIM( ) (去掉字符串左右两边的空格)。这三个函数可以有参数，具体功能不同于没有参数的情况，可以去掉字符串中特定字符。

1. **数据分组**

创建分组：

分组是使用SELECT语句的GROUP BY子句建立的

例：SELECT vend\_id,count(\*) as num\_prods from Products group by vend\_id;

过滤分组：

HAVING子句过滤分组

例：SELECT vend\_id,count(\*) as num\_prods from Products group by vend\_id having count(\*) >= 2;

分组和排序：

一般在使用GROUP BY子句时，应该也使用ORDER BY 子句。这是保证数据正确排序的唯一方法。

例：SELECT vend\_id,count(\*) as num\_prods from Products group by vend\_id having count(\*) >= 2 order by vend\_id;

注意：GROUP BY子句必须在WHERE子句之后，ORDER BY子句之前。WHERE在数据分组前进行过滤，HAVING在数据分组后过滤

1. **使用子查询**

子查询，即嵌套在其他查询中的查询

利用子查询进行过滤：

SELECT cust\_name,cust\_contact

FROM Customers

WHERE cust\_id IN(SELECT cust\_id

FROM Orders

WHERE order\_num IN(SELECT order\_num

FROM OrderItems

WHERE prod\_id = ‘RGAN01’));

作为计算字段使用子查询

SELECT cust\_name,

cust\_contact

(SELECT COUNT(\*)

FROM Orders

WHERE Orders.cust\_id = Customers.cust\_id) AS orders

FROM Customers

ORDER BY cust\_name;

注意：作为子查询的SELECT语句只能查询单个列，企图检索多个列将返回错误。若果使用子查询时两个表中字段名称相同应该使用完全限定名.（点号）

1. **联结**

关系表的设计就是要把信息分解成多个表，一类数据一个表。各表通过某些共同的值互相关联（所以才叫关系数据库）

使用联结时要保证都有WHERE子句，否则DBMS将返回比想象的多的多的数据（返回笛卡尔积的联结）

内联结（INNER JOIN）分为隐式内连接（没有INNER JOIN形成的中间表为两表的笛卡尔积）和显示内连接（有INNER JOIN形成的中间表为两表经过ON 条件过滤后的笛卡尔积）

例：SELECT O.ID,O.ORDER\_NUMBER,C.ID,C.NAME  
FROM CUSTOMERS C INNER JOIN ORDERS O ON C.ID=O.CUSTOMER\_ID;

外联结（OUTER JOIN）分为左外连接（LEFT OUTER JOIN）、右外连接（RIGHT OUTER JOIN）全外连接（FULL OUTER JOIN）

例：SELECT O.ID,O.ORDER\_NUMBER,O.CUSTOMER\_ID,C.ID,C.NAME  
FROM ORDERS O LEFT OUTER JOIN CUSTOMERS C ON C.ID=O.CUSTOMER\_ID AND O.ORDER\_NUMBER<>'MIKE\_ORDER001';

1. **组合查询（UNION）**

UNION必须由两天或两条以上的SELECT语句组成，语句间使用关键字UNION分隔（因此，如果组合四条SELECT语句，将要使用三个UNION关键字）

UNION中每个查询必须包含相同的列、表达式或聚集函数（不过，各个列不需要以相同的次序列出）

列数据类型必须兼容：类型不必完全相同，但必须是DBMS可以隐含转换的类型。

UNION默认是去掉重复的行的，使用UNION ALL可以不取消重复的行。

1. **插入数据（INSERT）**

例：INSERT INTO Customers(cust\_id,cust\_name)

VALUES(‘10086’,

‘Toy’);

插入表要有表名、字段以及字段值。

例：插入检索出的数据

INSERT INTO Customers(cust\_id,cust\_name)

SELECT cust\_id,

cust\_name

FROM CustNew;

例：从一个表插入另一个表

CREATE TABLE CustCopy AS

SELECT \* FROM Customers;

1. **更新和删除数据**

DELETE语句从表中删除行，甚至是删除表中所有行。但是DELETE不删除表本身。更快的删除可使用TRUNCATE TABLE语句，它完成相同的工作，而速度更快（因为不记录数据的变动）。

1. **创建和操纵表**

创建表CRATE TABLE OrderItems

(

Order\_num INTEGER NOT NULL PRIMARY KEY,

Order\_date CHAR NOT NULL DEFAULT SYSDATE

);

更新表 ALTER TABLE Vendors

ADD vend\_phone CHAR(20);

ALTER TABLE Vendors

ADD CONSTRAINT PRIMARY KEY(vend\_id[);

‘

ALTER TABLE Vendors

DROP COLUMN vend\_phone ;

删除表 DROP TABLE Vendors;

1. **约束**

创建表CRATE TABLE OrderItems

(

Order\_num INTEGER NOT NULL PRIMARY KEY,

Order\_date CHAR(8) NOT NULL DEFAULT SYSDATE，

Cust\_id CHAR(10) NOT NULL REFERENCES

Custromers(cust\_id)

);

其中的表定义了REFERENCES关键字，它表示cust\_id的任何值都必须是Custromers表的cust\_id中的值（外键）

修改表：

ALTER TABLE Vendors

ADD CONSTRAINT

FOREIGN KEY(vend\_id) REFERENCES Customers(cust\_id);

主键：

. 任意两行的主键值都不相同

. 每行都具有一个主键值（即列中不允许NULL值）

. 包含主键值的列从不修改或者更新。（大多数DBNS不允许这么做）

唯一约束(UNIQUE)：

. 表中可包含多个唯一约束，但每个表只允许一个主键。

. 唯一约束可包含NULL值

. 唯一约束可修改或更新

. 唯一约束的值可重复使用

. 与主键不一样，唯一约束不能用来定义外键

检查约束(CHECK):

CRATE TABLE OrderItems

(

Order\_num INTEGER NOT NULL PRIMARY KEY,

Order\_date CHAR NOT NULL DEFAULT SYSDATE

Quantity INTEGER NOT NULL CHECK (Quantity > 3)

);

ALTER TABLE Vendors

ADD CONSTRAINT CHECK(gender like ‘[MFH]’)

索引(INDEX):

. 主键数据总是排序的，这是DBMS的工作。因此按主键检索特定行是一种快速有效的操作

. 索引改善检索操作的性能，但降低了数据插入、修改和删除的性能。在执行这些操作时，DBMS必须动态的更新索引。

. 索引数据可能占用大量的存储空间

. 并非索引数据都适合做索引。取值不同的数据（如州）不如具有更多可能只的数据（如姓名），能通过检索得到那么多的好处

CREATE INDEX prod\_name\_ind

ON Products(prod\_name);

1. **创建视图**

视图是虚拟的表。与包含数据的表不一样，视图只包含使用时动态检索数据的查询

为什么使用视图

. 重用SQL语句

. 简化复杂的SQL操作，在编写查询后可以方便的重用它而不必知道基本的细节

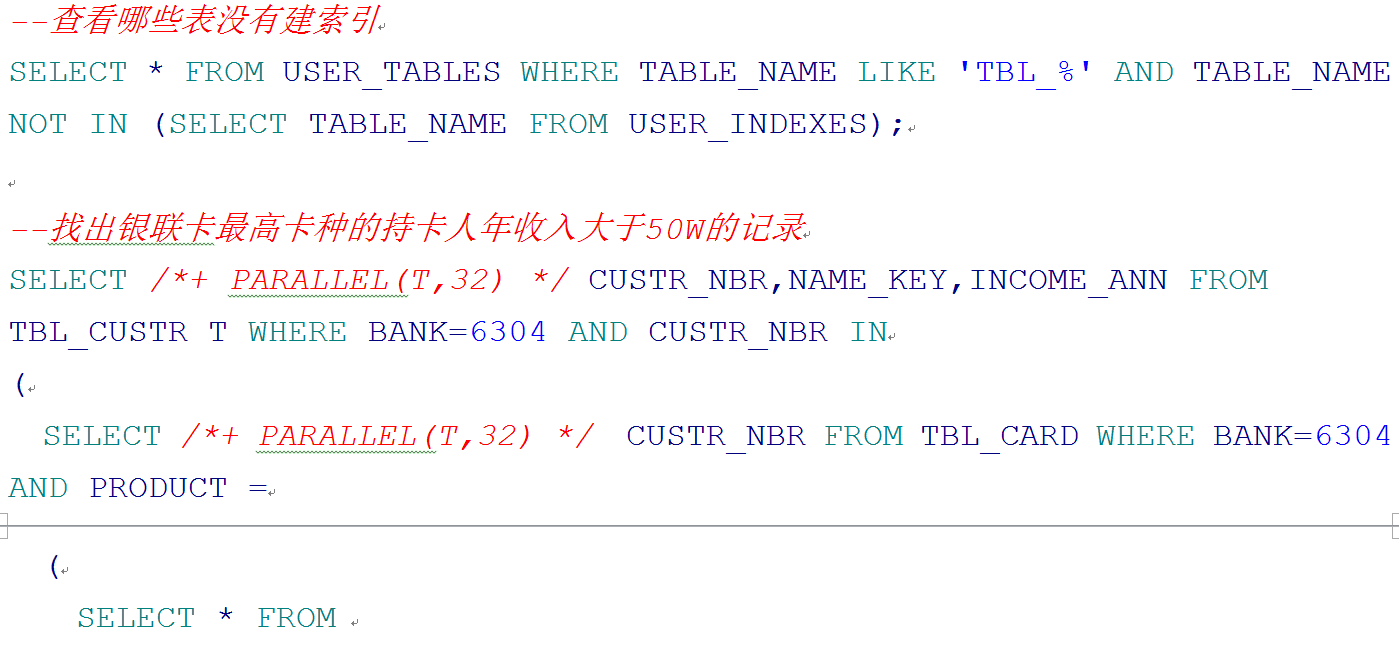
. 使用表的一部分而不是整个表

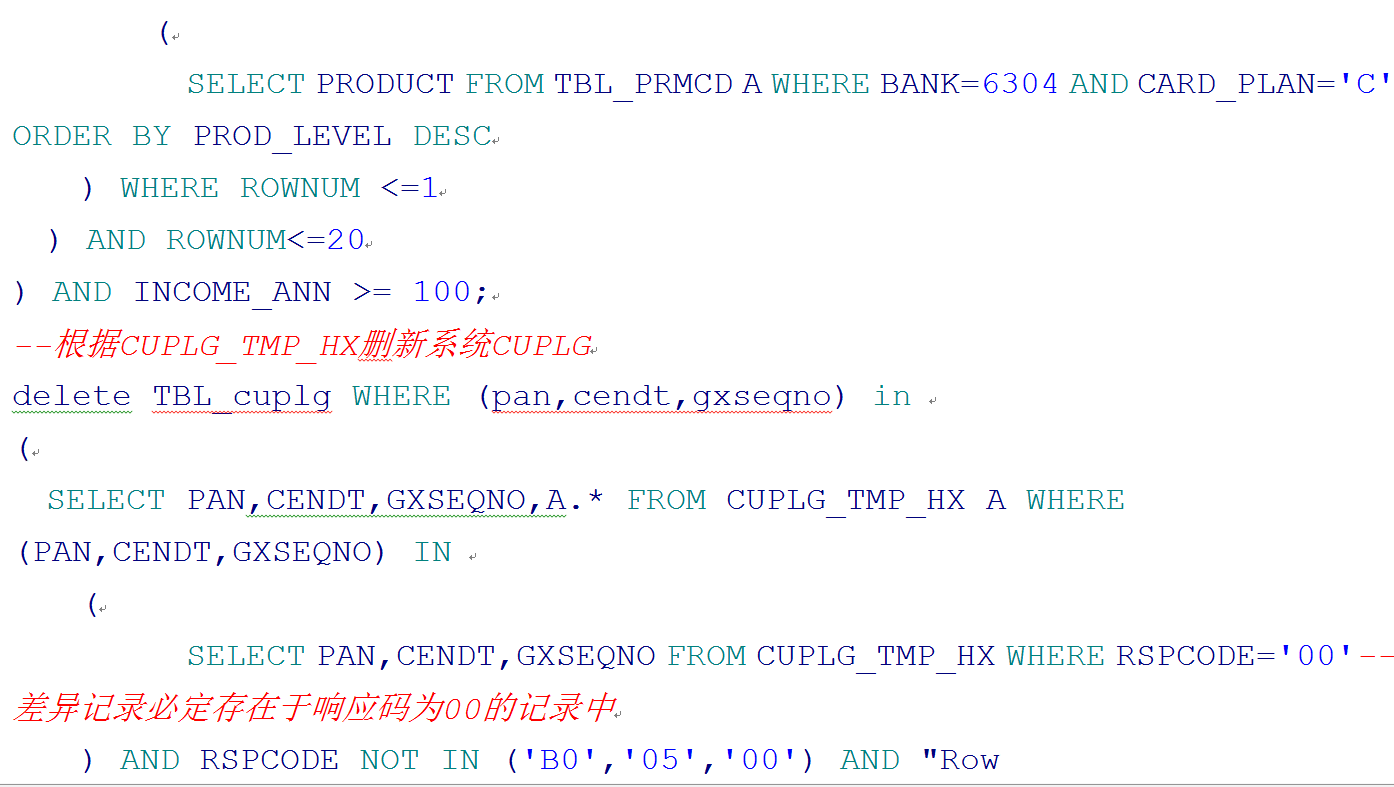
. 保护数据。可以授予用户访问表的特定部分的权限，而不是整个表的访问权限

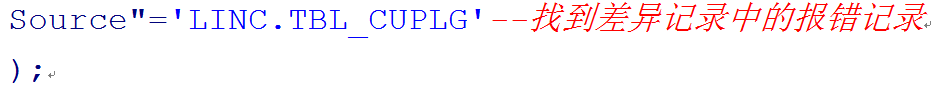
. 更改数据格式和表示。视图可返回与底层表的表示和格式不同的数据。

# 复杂SQL

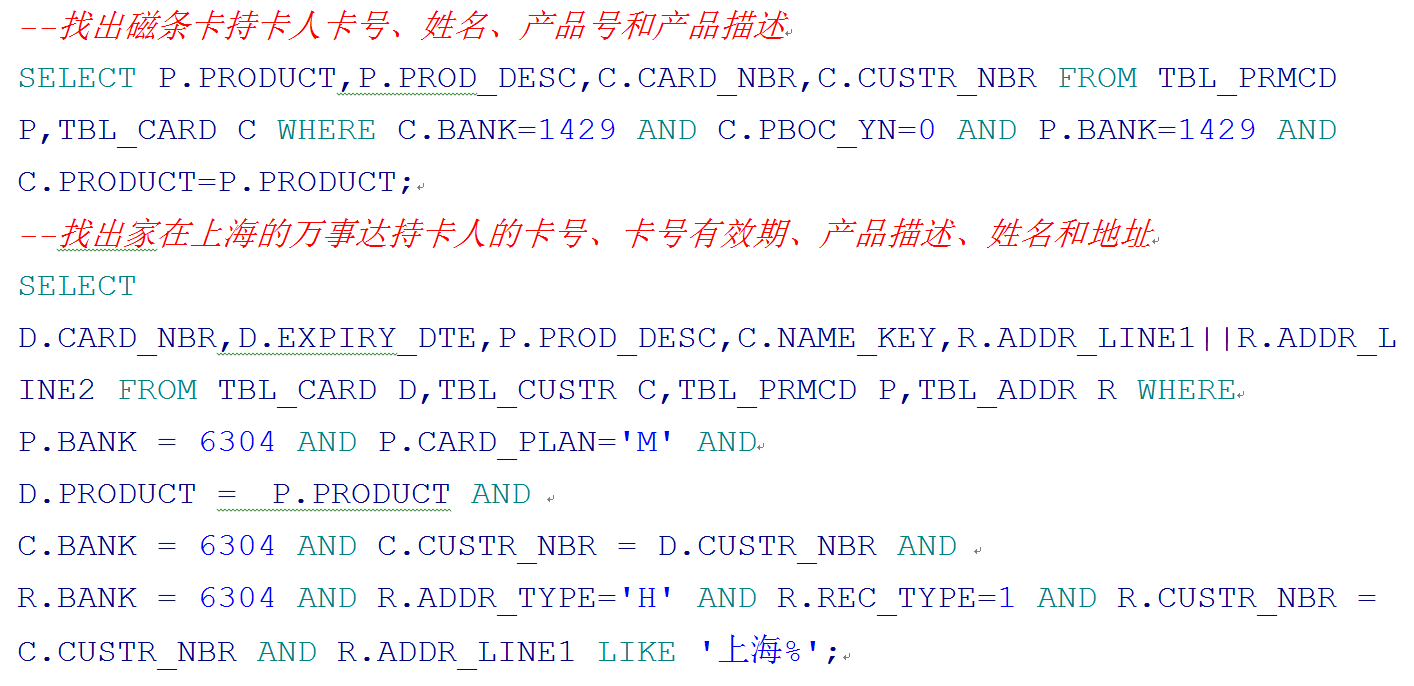
1. 子查询（需要过滤的字段在另外一张表中）



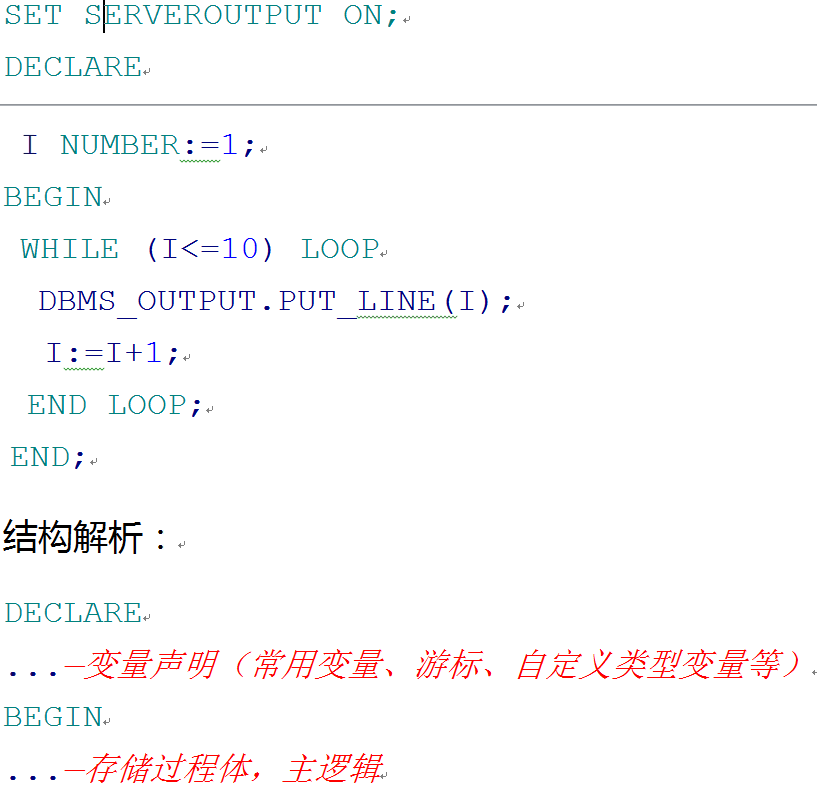




1. 连接（除了过滤字段，还要返回其他字段）



1. 存储过程（用于复杂SQL语句、需要参数的语句或者指定笔数提交的时候）



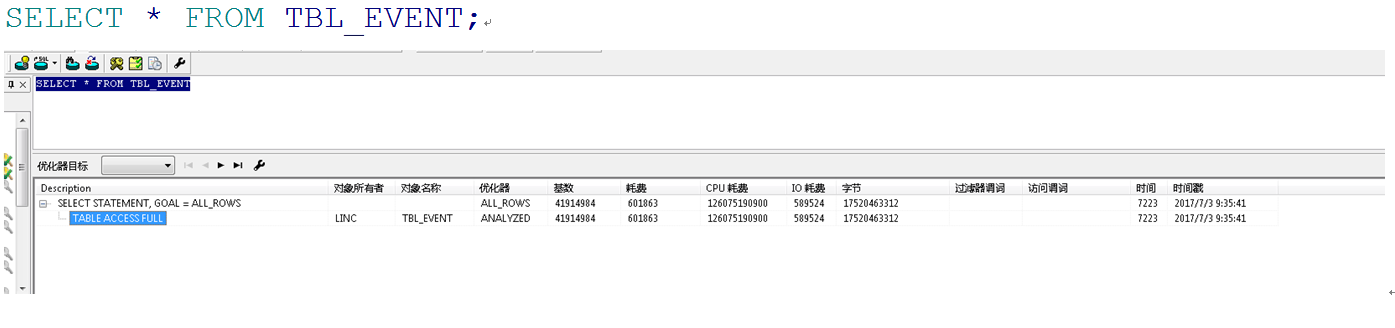


# SQL调优

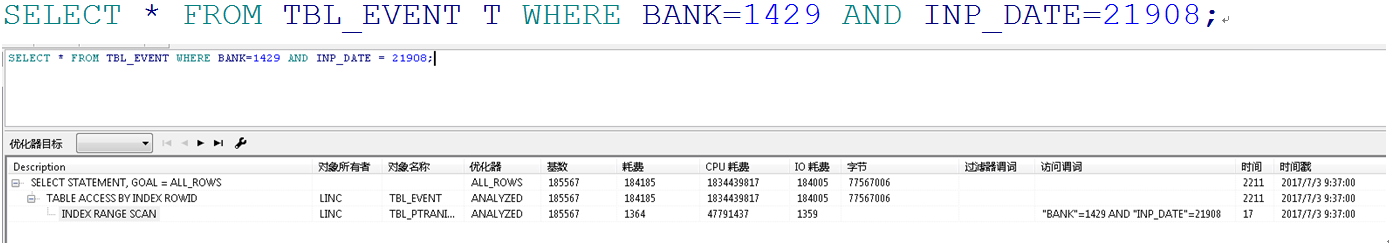
1. 执行计划

PL/SQL Developer里按F5，即可看到SQL语句的解释计划，可以通过设置选取需要的显示项。

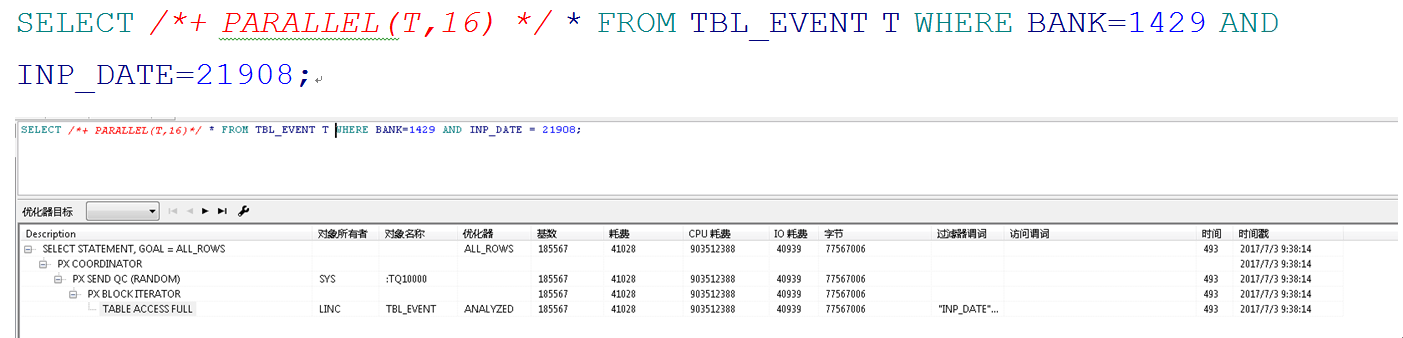
1. 全表扫描



1. 索引

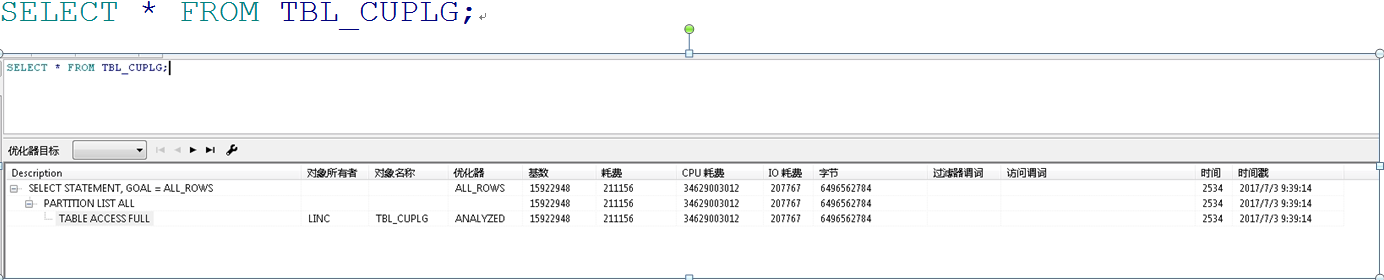


1. 并行

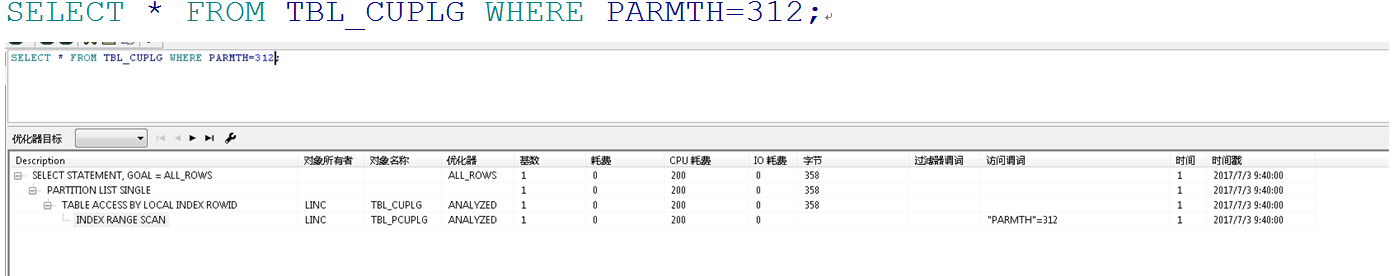


1. 分区

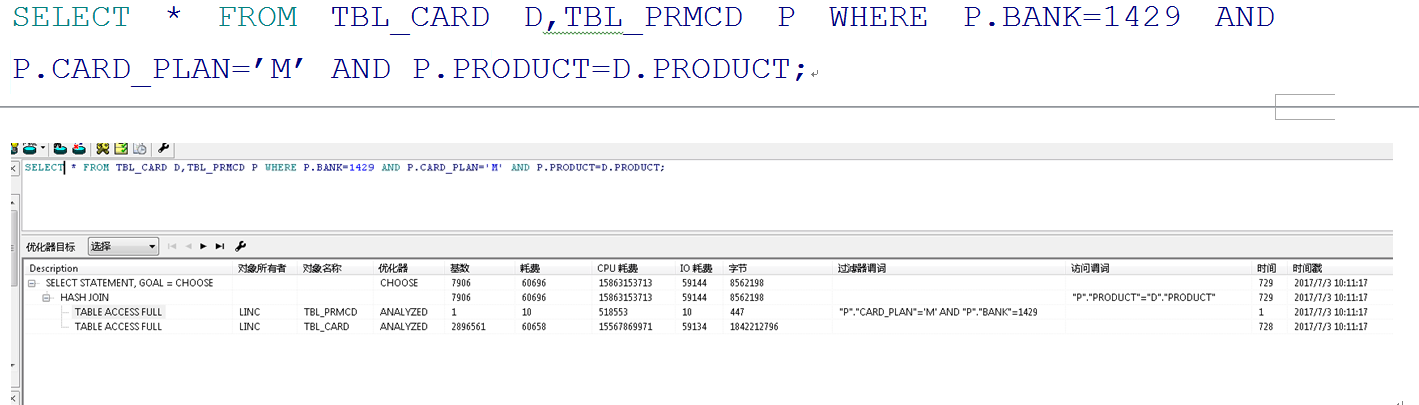
列出所有分区（全表扫描）



列出单个分区



1. 联接

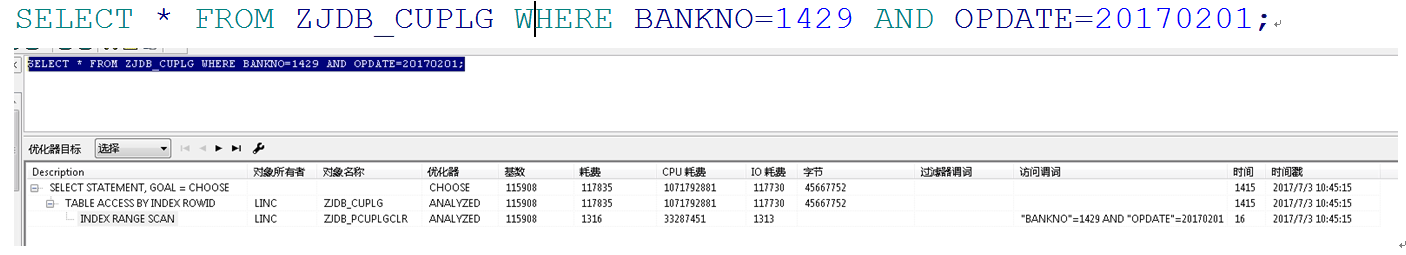


1. 并行与索引

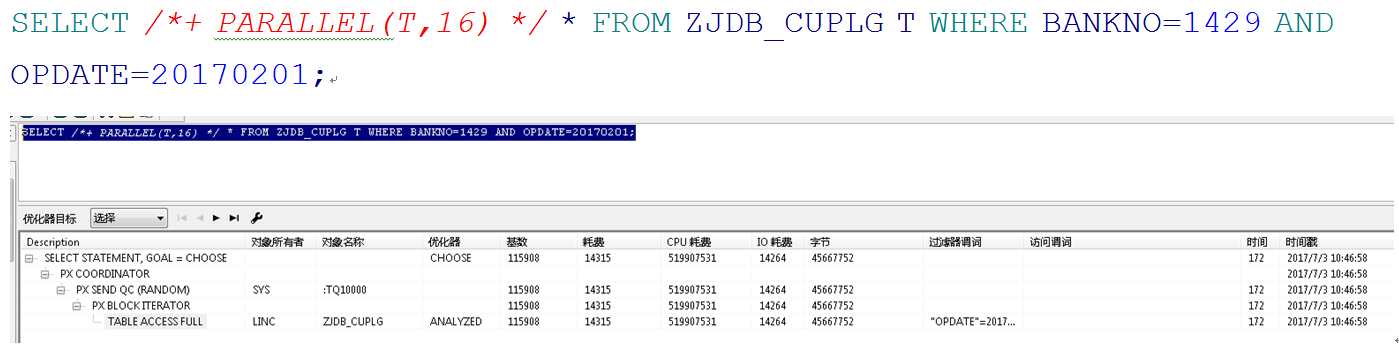
并行快还是索引快？

例1：

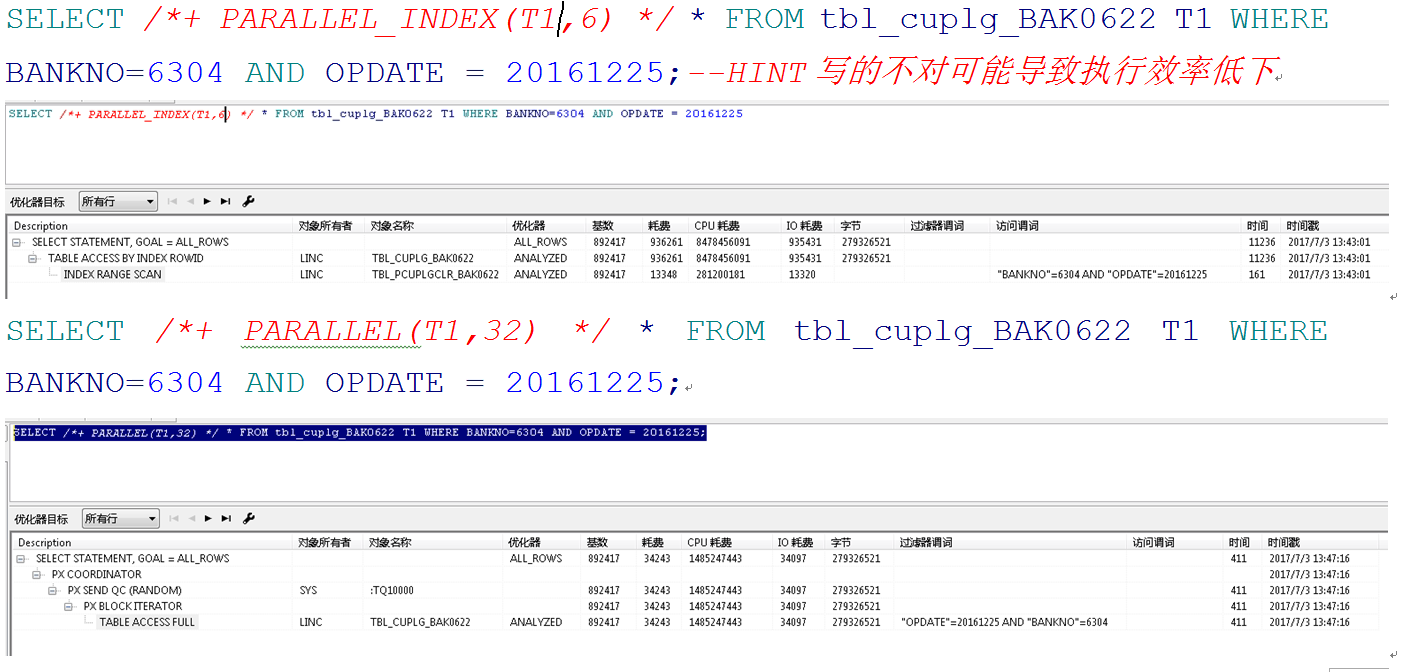
--使用索引方式查询OPDATE=20170102的记录



--使用并行方式查询OPDATE=20170102的记录

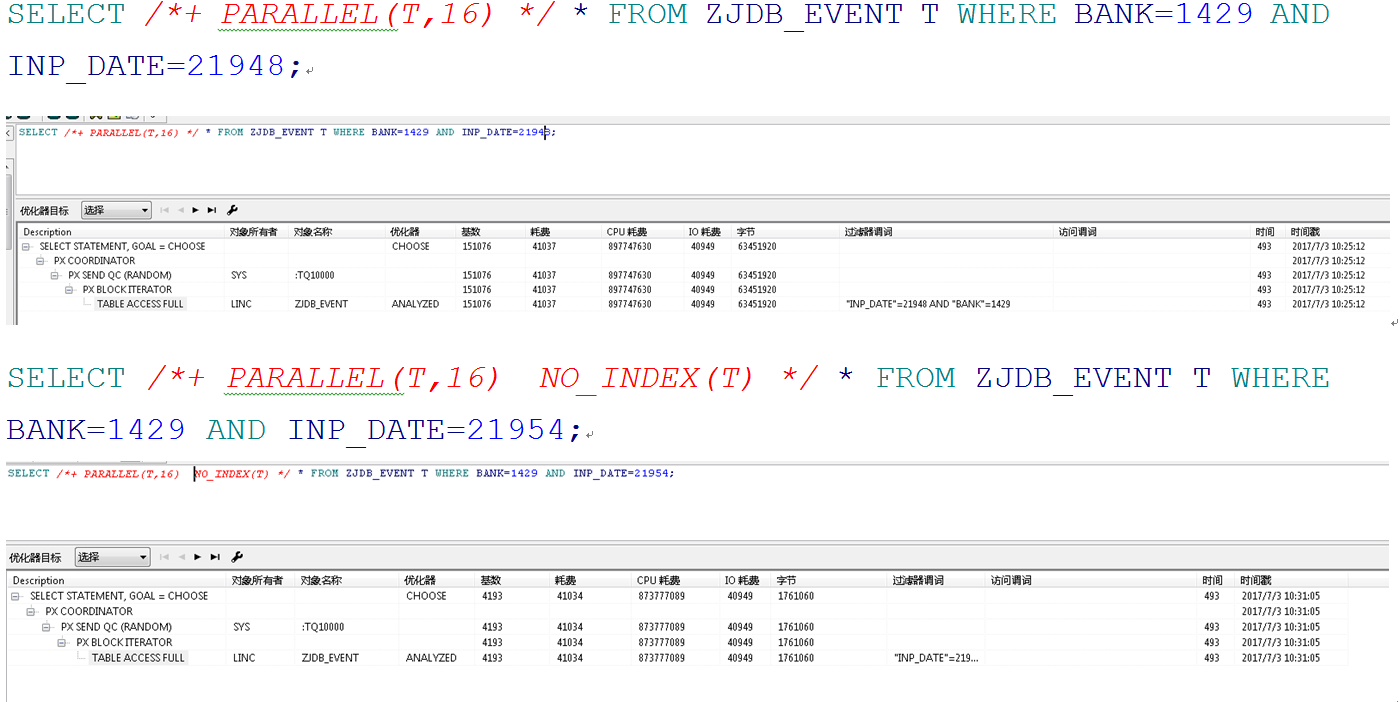


例2：

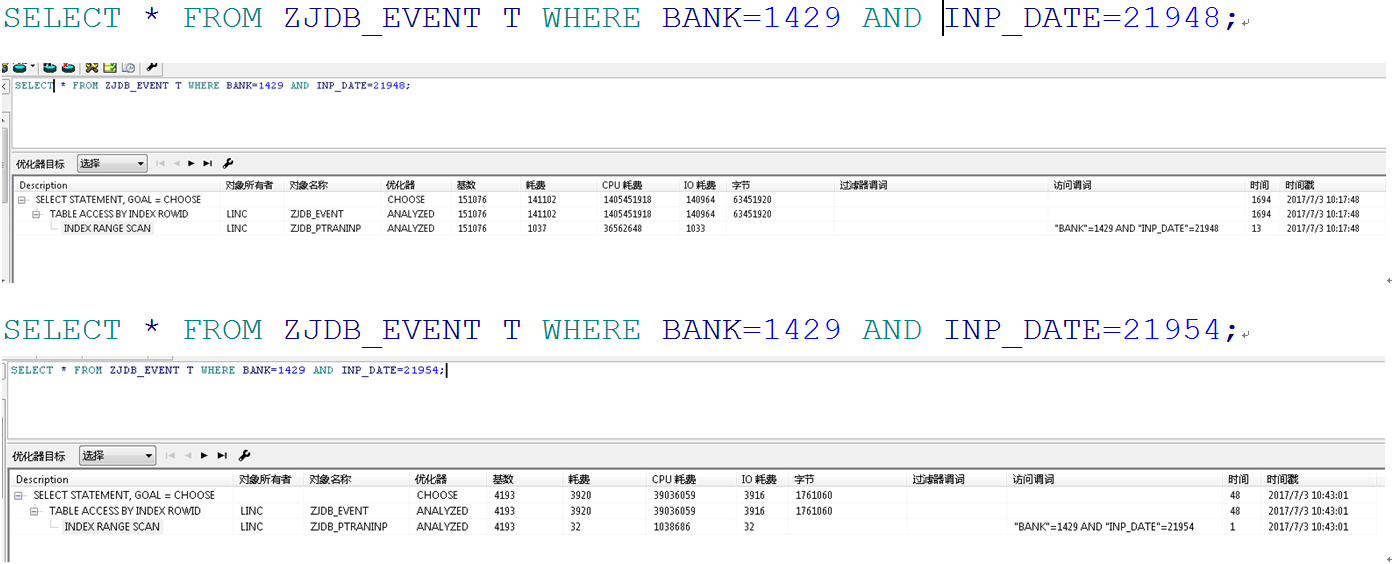


例3：

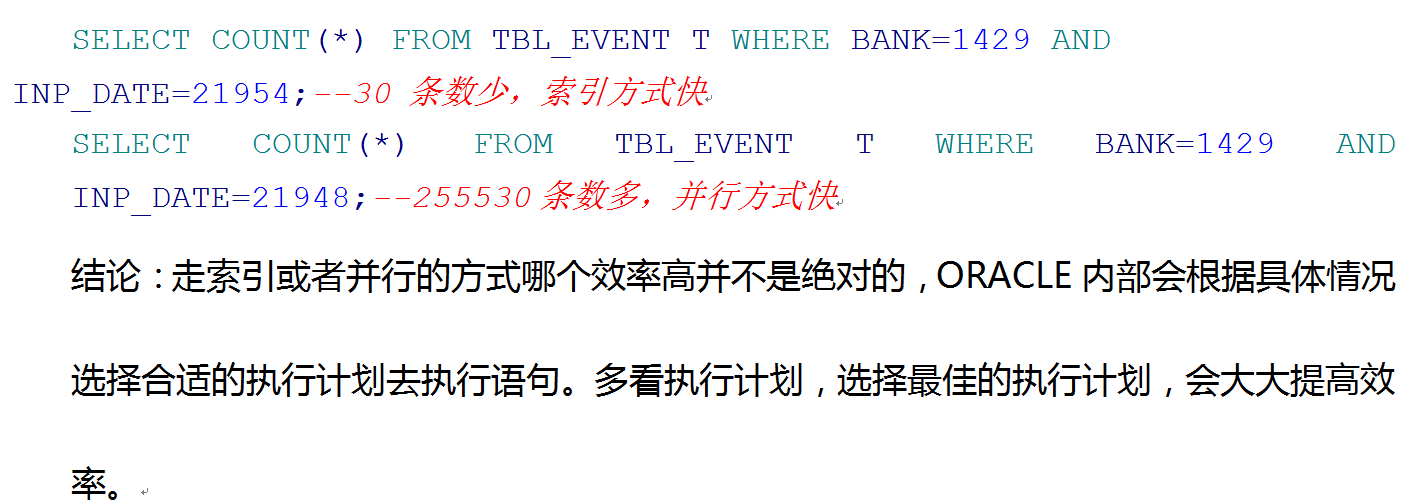
--使用并行方式查询INP\_DATE=21948/21954



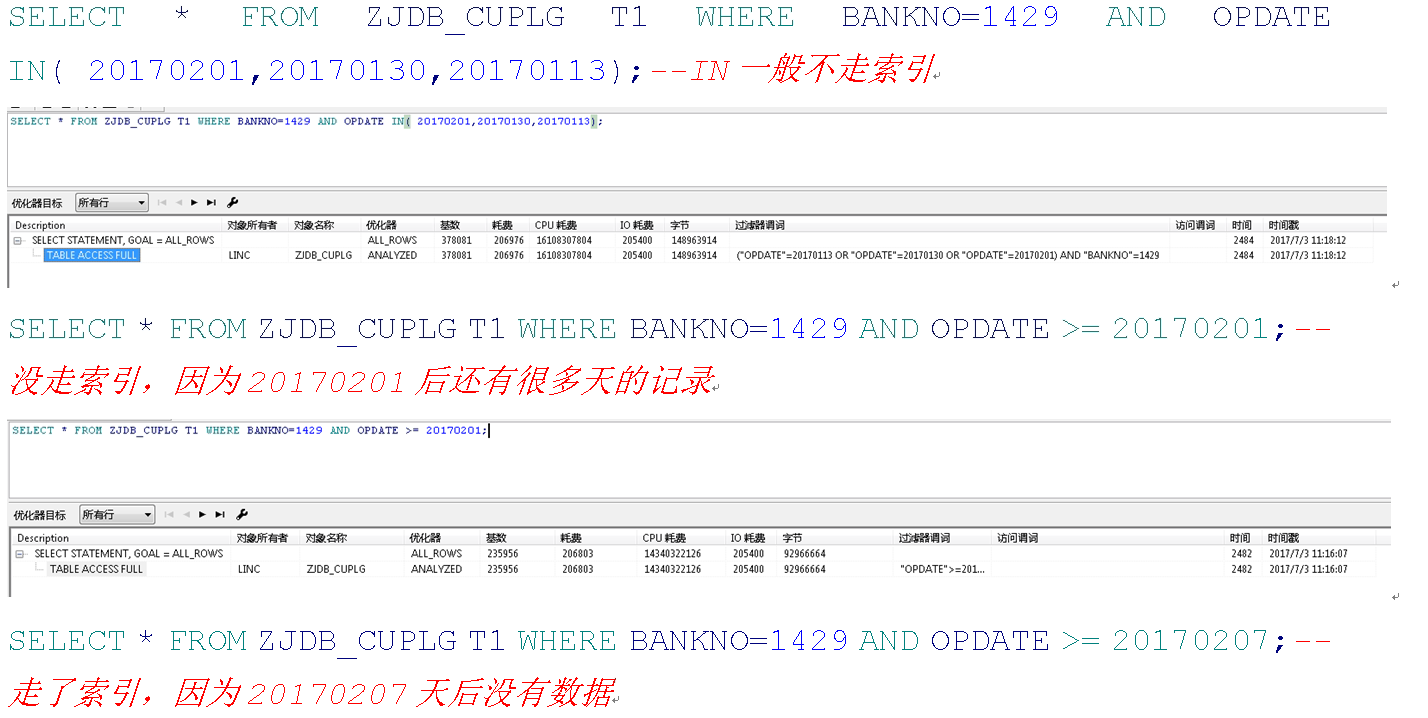
--使用索引方式查询INP\_DATE=21948/21954

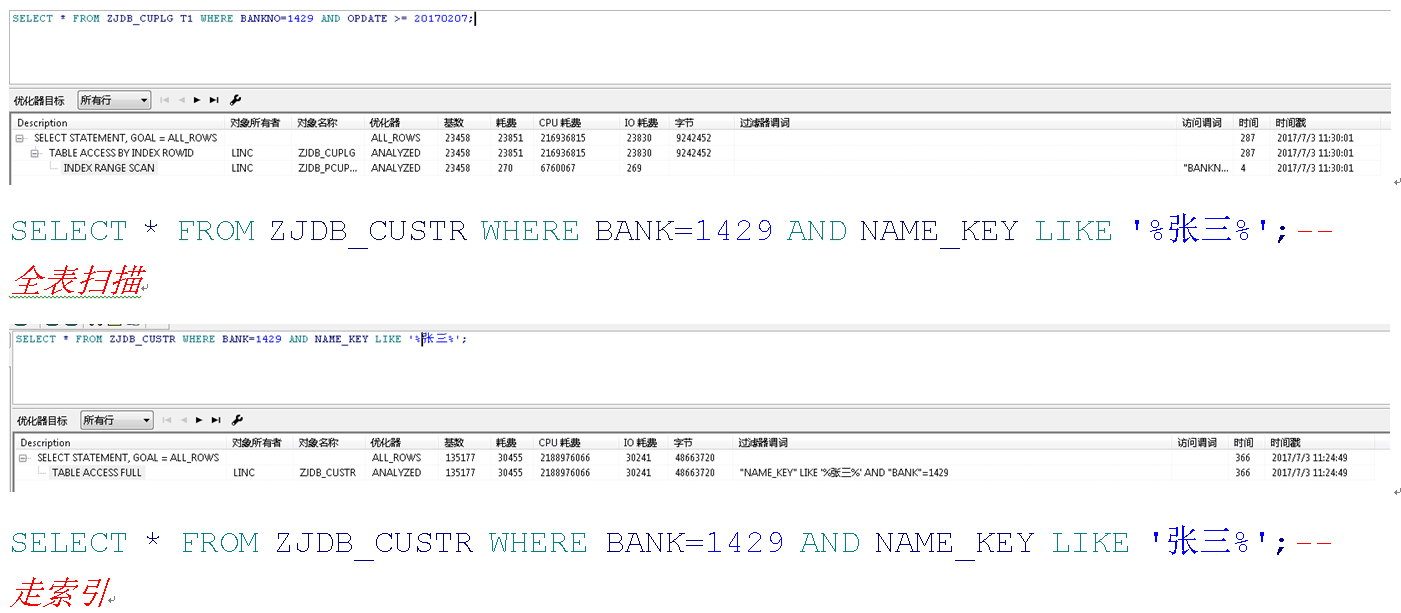


原因：



1. 索引失效情况



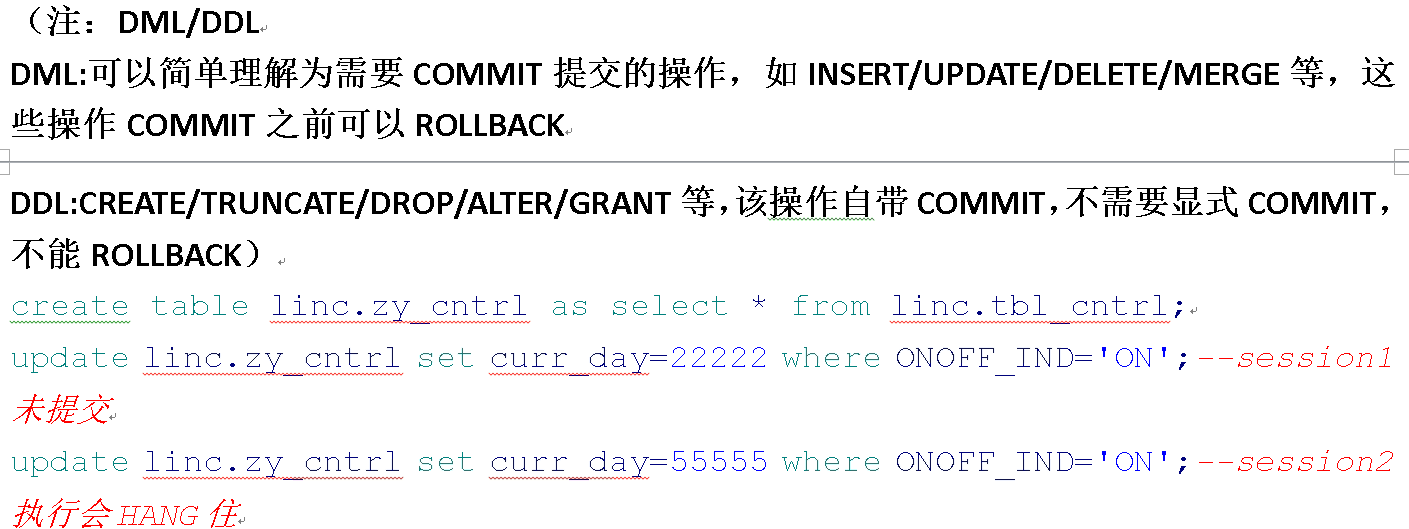




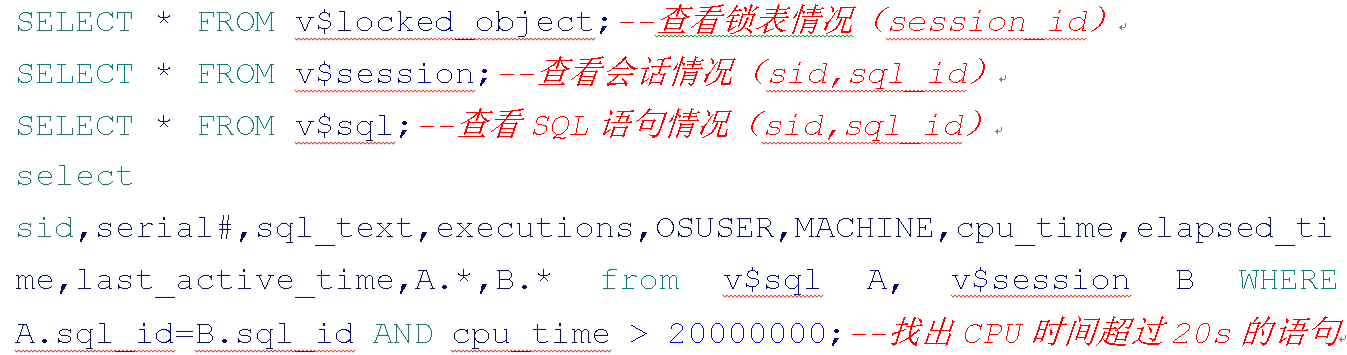
# 锁表监控

1. 锁表情况

最常见的就是失误未提交，比如DML之后没有COMMIT，影响其他会话对该表或该行的事物操作



1. 常见视图



1. 杀掉进程

ALTER SYSTEM KILL SESSION ‘SID,SERIAL#’ immediate;