# Oracle

## 基础部分

///

### 基础要点

1）环境变量

Oracle主目录：ORACLE\_HOME

2）

3）

4）

5）

///

### 通用知识

1）ROWNUM

2）单引号

①字符串：2个单引号括起来;

②转义：在字符串中，使用一个单引号进行特殊字符转义，使之可以在字符串中表达出来.

3）SID：Oracle实例名(Instance Name)

唯一(sole)标识符(identifier)

sole identifier  
4）Oracle安装之后

实例名:SID

实例\_服务名：OralceServiceSID

监听器\_服务：OracleOraDb11g\_home1TNSListener

5）SQL语言：

DDL-Data Definition Language 数据定义语言

DCL-Data Control Language 数据控制语言

DML-Data Manipulate Language 数据操纵语言

DQL-Data Query Language 数据查询语言

6）每个Db至少有1个实例为其服务，实例中的内存结构称为系统全局区(System Global Area)

7）每个Db实例需要占用非常大的内存空间，因此一般一台服务器只安装一个Db实例，一个Db实例可以有多个用户（模式schema），不同的应用系统使用不同的用户访问即可。

8）安装Oracle完成之后，Oracle为管理员预置了2个用户：sys / system

同时，为程序测试提供了一个普通用户：scott (默认密码是：tigger)。

可以决定普通用户是否被锁定。

系统帐户：sys / system / sysman / dbsnmp

/\* nmp: network management protocol \*/

9）连接身份（该用户连接后拥有的权限）

①sysdba：Db管理员，权限：打开/关闭Db服务器、备份/恢复Db、日志归档、会话限制、管理功能、创建Db。sys用户必须以sysdba的身份才能登录，system用户可以以普通身份登录。

②sysoper：Db操作员，权限：打开/关闭Db服务器、备份/恢复Db、日志归档、会话限制。

③normal：普通用户，权限：查询某些表的数据。

/\* 默认用户是 normal \*/

10）字符串连接： ||

/\* 由于Oracle支持类型自动转换，数值也可以进行连接 \*/

11）空值（NULL值）不等于0或者空格，空值是指未赋值、未知、不可用的值。任何数据类型的列都可以包含空值，除非该列被定义为非空或主键。

NOT NULL / PRIMARY KEY

Oracle将NULL作为无穷大处理，所以排序都放在

之后。

IS NULL

NOT NULL

12）

13）

14）

///

3）

4）

5）

///

///

### 标准规范

1）Oracle中，所有英文单词全部大写.Db名称、表名、列名、索引、表空间、值，全部采用大写的形式存储。

2）表命名：tb\_xx

主键\_约束：pk\_tb\_name\_col\_name

外键\_约束：fk\_tb\_name\_col\_name

索引：idx\_tb\_name\_col\_name

视图：v\_xx

3）sql\_语句\_规范：关键字、函数名大写，表名、字段名小写。  
4）

5）

///

关键字

1）PRIMARY KEY

2）NOT NULL

3）UNIQUE

4）DEFAULT

5）REFERENCE KEY REFERENCE tab\_name(col\_name)

6）

7）

///

Oracle\_Db

Db / DbInstance / DbSchema / DbObject

public class Db {

private Collection<DbInstance> instance;

}

public class DbInstance {

private Collection<DbSchema> schema;

}

public class DbSchema {

private Collection<DbObject> dbObj;

}

DbObject子类: Table / Index / Cluster /

/\* 一般采取一个Db实例下，构建多个Db模式，创建一个Db用户，即对应一个Db模式 \*/

DbObject是Db中很多对象的父类/抽象.

///

## 数据类型

1）数据类型(datatype)是列或存储过程中的一个属性。Oracle支持的数据类型：字符、数字、其他数据类型。

2）NLS: National Language Support本地/国家语言支持.

3）大对象

LOB：Large Object (共有如下3种)

BLOB：Binary Large Object 二进制大对象

CLOB：Character Large Object 字符大对象

NCLOB：Natioanl CLOB 本地语言 CLOB

提示：可以指定将一个LOB数据保存在Oracle数据库内，也可以指向一个包含此数据的外部文件。

4）

5）Oracle支持类型的自动转换/显示转换。

6）NULL

NULL不代表任何值，它表示没有值。

NOT NULL: 非空约束。

IS NULL：判断某个数据是否为NULL。

7）数据类型声明字节长度，字母-1个字节，汉字-2个字节

8）

### 字符类型

1）CHAR：存储固定长度的字符数据。

CHAR() /\* size = 1 \*/

CHAR(int size) /\* [1, 2000]Byte \*/

提醒：如果对CHAR类型变量赋值，其长度小于规定的长度，Oracle将自动在[右边]用空格填充。

2）VARCHAR2：存储可变长度的字符数据。

VARCHAR2() /\* maxSize = 1 \*/

VARCHAR2(int maxSize) /\* [1, 4000]Byte \*/

提醒：对VARCHAR2赋值的最大长度而不是实际长度，不需要用空格填充。最大4000个字节。

3）CLOB

CLOB()/\* 无参 \*/

提示：最大可以存储4GByte数据。

4）BLOB

Binary Large Object 二进制大对象

以二进制格式保存数据。

BLOB() /\* 无参 \*/

5）BINARY\_FLOAT

6）BINARY\_DOUBLE

7）BFILE：指向存储在Oracle数据库以外的文件的指针。

BFILE() /\* 无参 \*/

提醒：Oracle可以读取、查询BFILE,但不能写入，文件的大小由OS决定。

### 数字类型

1）Oracle使用标准的、可变长度的内部格式来存储数据，这个内部格式精度可以高达38位。

2）NUMBER

NUMBER()/\* precision = 38,scale = 0 \*/

NUMBER(int precision) /\* scale = 0 \*/

NUMBER(int precision, int scale)

提示：precision-精度，scale-小数点右边的位数，若scale < 0 则把数字取舍到小数点左边的指定位数。

/\* 小数: 需要将小数点包含在精度值内 \*/

precision: [1, 38] scale: [-84, 127]

用1B存储指数，20B变长字节存储该数字剩下的部分。

3）

4）

5）

///

### 日期类型

1）Oracle标准日期格式：

dd-mon-yy hh:mi:ss

中文格式：

YYYY-MM-DD HH:MI:SS FF1-9

YYYY-MM-DD HH24:MI:SS FF1-9

/\* FF后面跟着数字1-9，是秒的小数部分，最大精度是9 \*/

Oracle用7个字节来存储日期时间数据。

示例：

SELECT TO\_CHAR(SYSTIMESTAMP, 'YYYY-MM-DD HH24:MI:SS FF9') FROM dual;

结果：2013-05-05 10:54:17 187000000

2）DATE

DATE() /\* 返回Db当前日期时间 \*/

3）TIMESTAMP

TIMESTAMP() /\* 返回当前时区的日期时间 \*/

4）TIMESTAMP WITH LOCAL TIME ZONE

/\* 会话的时区 \*/

5）

///

### 其他数类型

1）RAW / LONG RAW / LONG RAW RAW

2）Oracle以bit的形式来存储数据，RAW数据类型一般用于存储有特定格式的对象，例如位图。

3）RAW 最大可占 2KB空间，LONG RAW最大可占2GB空间，都没有默认大小，所以在使用时要指定大小，可以建立索引。

4）ROWID是一种特殊的列类型，称为伪列。Oracle数据中的每张表都有ROWID伪列，ROWID表示行的地址。（ROWID数据类型，ROWID伪列）

5）RAW(int size) size:[1, 2000]

6）

7）

8）

///

## 函数部分

1）字符串\_Fun

String CONCAT(String, String)

/\* String对象中第一个字符的ascii码 \*/

int ASCII(String)

char CHR(int ascii)

/\* 单词\_首字母\_大写,其余小写，单词通过空格、逗号、控制字符来识别 \*/

String INITCAP(String)

int INSTR(String, String)

int INSTR(String, String, int begin)

/\* int appear 出现的位置 \*/

int INSTR(String, String, int, int appear)

int LENGTH(String)

String LOWER(String)

String UPPER(String)

/\* 在列的左边粘贴字符串 \*/

String LPAD(String)

/\* 在列的右边粘贴字符串 \*/

String RPAD(String)

/\* 删除左边出现\_字符串 \*/

String LTRIM(String, String trim\_str)

/\* 删除右边出现\_字符串 \*/

String RTRIM(String, String trim\_str)

String SUBSTR(String, int, int len)

String REPLACE(String, String, String rep\_str)

/\* 返回与给定str读音相同\_字符串 \*/

String SOUNDER(String)

String TRIM(String)

String TRIM(String, String)

String LEADING(String)

String TRAILING(String)

2）数字\_Fun

double ABS(double)

double ACOS(double)

double ASIN(double)

double ATAN(double)

double SIN(double)

double COS(double)

double TAN(double)

double COSH(double)

double SINH(double)

double TANH(double)

double EXP(double)

int FLOOR(double)

int CEIL(double)

double LN(double)

double LOG(double, double)

int MOD(int, int)

double POWER(double, double)

int ROUND(double)

Number ROUND(double, int)

/\* 截取\_数字 \*/

intTRUNC(double)

Number TRUNC(double, int numericSize)

/\* -1, 0, 1 \*/

int SIGN(double)

double SQRT(double)

Date ADD\_MONTHS(int)

Date LAST\_DATE(Date)

int MONTHS\_BETWEEN(Date, Date)

Date NEW\_TIME(Date, String, String)

Date NEXT\_DAY(Date, Date)

Date NEXT\_DAY(String, String)

Date SYSDATE()

3）转换\_Fun

RowId CHARTOROWID(String)

String ROWIDTOCHAR(RowId)

String HEXTOROW(String)

String ROWTOHEXT(String)

String TO\_CHAR(Date, String fmt)

例如:

SELECT TO\_CHAR(SYSDATE, 'yyyy-mm-dd') FROM dual;

SELECT TO\_CHAR(SYSDATE, 'yyyy-mm-dd hh:mi:ss') FROM dual;

SELECT TO\_CHAR(SYSDATE, 'yyyy-mm-dd hh24:mi:ss') FROM dual;

String TO\_CHAR(Number, String fmt)

SELECT TO\_CHAR(123, '999') FROM dual;

Date TO\_DATE(String, String fmt)

SELECT TO\_DATE('2013-04-19', 'yyyy-mm-dd') FROM dual;

Timestamp TO\_TIMESTAMP(String, String fmt)

Number TO\_NUMBER(String, String fmt)

SELECT TO\_NUMBER('123', '999') FROM dual;

String TO\_MULTI \_BYTE(String)

BLOB BFILENAME(String dir, String file)

String CONVERT(????)

String DUMP(String, String fmt, int, int len)

4）

EMPTY\_BLOB() EMPTY\_CLOB()

String GREATEST(String, String...)

Number GREATEST(Number, Number...)

String LEAST(String, String...)

Number LEAST(Number, Number...)

5）

6）

常用属性

1）用户\_

String UID;

String USER;

2）

3）

///

注释:

COMMENT ON TBALE tb\_name IS’comm’;

COMMENT ON COLUMN tb.col IS’comm’;

///

## 表空间

1）

2）

3）步骤

①创建临时表空间

CREATE TEMPORARY TABLESPACE tbs\_name

LOGGING

TEMPFILE’dir.../xx.dbf’

SIZE ?M

AUTOEXTEND ON

NEXT xxM

MAXSIZE ?M

EXTENT MANAGEMENT LOCAL;

②创建表空间

CREATE TABLESPACE tbs\_name

DATAFILE ‘dir../xx.dbf’

SIZE xxM

AUTOEXTEND ON

NEXT

MAXSIZE

EXTENT MANAGEMENT LOCAL;

③创建用户

CREATE USER user\_name IDENTIFIED BY pwd

DEFAULT TABLESPACE xx

TEMPORARY TABLESPACE yy

ACCOUNT LOCK/UNLOCK;

④授权

GRANT CONNECT, SESSION, RESOURCE, DBA TO user\_name;

/\* 有些版本报 SESSION角色不存在错误 \*/

4）修改表空间

ALTER TABLESPACE xx

5）删除表空间

DROP TABLESPACE tb\_space; /\* 删除该表空间在Db内的文件信息，物理文件并不删除 \*/

DROP TABLESPACE tb\_space INCLUDING CONTENTS AND DATAFILES; /\* 删除内容和物理文件 \*/

6）

///

Oracle\_scott\_emp:

INSERT INTO EMP VALUES

(7369,'SMITH','CLERK',7902,to\_date('17-12-1980','dd-mm-yyyy'),800,NULL,20);

INSERT INTO EMP VALUES

(7499,'ALLEN','SALESMAN',7698,to\_date('20-2-1981','dd-mm-yyyy'),1600,300,30);

INSERT INTO EMP VALUES

(7521,'WARD','SALESMAN',7698,to\_date('22-2-1981','dd-mm-yyyy'),1250,500,30);

INSERT INTO EMP VALUES

(7566,'JONES','MANAGER',7839,to\_date('2-4-1981','dd-mm-yyyy'),2975,NULL,20);

INSERT INTO EMP VALUES

(7654,'MARTIN','SALESMAN',7698,to\_date('28-9-1981','dd-mm-yyyy'),1250,1400,30);

INSERT INTO EMP VALUES

(7698,'BLAKE','MANAGER',7839,to\_date('1-5-1981','dd-mm-yyyy'),2850,NULL,30);

INSERT INTO EMP VALUES

(7782,'CLARK','MANAGER',7839,to\_date('9-6-1981','dd-mm-yyyy'),2450,NULL,10);

INSERT INTO EMP VALUES

(7788,'SCOTT','ANALYST',7566,to\_date('13-07-87','dd-mm-rr')-85,3000,NULL,20);

INSERT INTO EMP VALUES

(7839,'KING','PRESIDENT',NULL,to\_date('17-11-1981','dd-mm-yyyy'),5000,NULL,10);

INSERT INTO EMP VALUES

(7844,'TURNER','SALESMAN',7698,to\_date('8-9-1981','dd-mm-yyyy'),1500,0,30);

INSERT INTO EMP VALUES

(7876,'ADAMS','CLERK',7788,to\_date('13-07-87', 'dd-mm-rr')-51,1100,NULL,20);

INSERT INTO EMP VALUES

(7900,'JAMES','CLERK',7698,to\_date('3-12-1981','dd-mm-yyyy'),950,NULL,30);

INSERT INTO EMP VALUES

(7902,'FORD','ANALYST',7566,to\_date('3-12-1981','dd-mm-yyyy'),3000,NULL,20);

INSERT INTO EMP VALUES

(7934,'MILLER','CLERK',7782,to\_date('23-1-1982','dd-mm-yyyy'),1300,NULL,10);

///

## 序列

1）

CREAT SEQUENCE seq\_xx

START WITH xx

INCREMENT BY xx

MINVALUE / NOMINVALUE

MAXVALUE / NOMAXVALUE

CACHE xx / NOCACHE

ORDER / NOORDER

CYCLE / NOCYCLE

2）调用序列

seq\_xx.CURRVAL

seq\_xx.NEXTVAL

3）修改序列

ALTER SEQUENCE

/\* 除了不能修改 START WITH 起始值外，其他都能修改 \*/

4）

5）

DDL

///

DCL

## Sql

1）SQL：Structured Query Language

第四代语言(4GL)，其特点是非过程化，即不用指明执行的具体方法和路径，而是简单地调用相应的语句来直接取得结果即可。

2）Sql包括：ddl/dml/dql/dcl/tcl

Data Definition Language

Data Manipulate Language

Data Query Language

Data Control Language

Transaction Control Language

3）结果集（ResultSet）

SELECT查询语句返回0-n条由多个表组成的记录。

4）commission：佣金、奖金...

5）

6）

7）

///

### DDL

1）CREATE/DROP/ALTER/TRUNCATE/RENAME//

2）创建表

①自定义创建表：

CREATE TABLE tb\_name (

col1 datatype...,...

);

②根据结果创建表：

CREATE TABLE tb\_name AS SELECT 语句;

根据结果集方式创建表，复制了表结构之后，根据条件可以复制表中的记录；复制表结构，即不要记录，只要表，需要加上无条件为false的条件即可。

③复制表结构：

CREATE TABLE tb\_name AS SELECT 语句

WHERE 1 = 2 或者 1 <> 1；

3）TRUNCATE

删除表的所有记录（保留表结构）：

/\* TRUNCATE TABLE属于ddl操作，删除的数据不能恢复，而DELETE FROM属于dml操作，删除的数据可以通过日志文件恢复；如果一个表中记录较多，TRUNCATE的速度比DELETE FROM快，因为TRUNCATE不用做日志操作，所以速度快。 \*/

TRUNCATE TABLE tb\_name;

4）

5）

///

### DCL

1）GRANT / DENY / REVOKE

2）

3）

4）

5）

///

### DML

1）DML：Data Manipulate Language

数据操纵语言

2）INSERT

INSERT INTO tb\_name(col...) VALUES(val...);

向表中插入一个结果集：

/\* 要求结果集中每一列的数据类型必须与表中的每一列的数据类型一致，并且两者数量一致 \*/

INSERT INTO tb\_name SELECT语句;

3）DELETE

DELETE FROM tb\_name [WHERE ..];

删除所有记录(保留表结构)：

方法1：DELETE FROM tb\_name;

/\* TRUNCATE TABLE 属于ddl操作 \*/

方法2：TRUNCATE TABLE tb\_name;

/\* TRUNCATE TABLE属于ddl操作，删除的数据不能恢复，而DELETE FROM属于dml操作，删除的数据可以通过日志文件恢复；如果一个表中记录较多，TRUNCATE的速度比DELETE FROM快，因为TRUNCATE不用做日志操作，所以速度快。 \*/

4）UPDATE

UPDATE tb\_name SET col=value,...

[WHERE ...];

5）

6）

///

### DQL

1）DQL：Data Query Language

2）SELECT col\_name,... FROM tb\_name

查询单行记录，使用系统表dual:

SLECT ... FROM dual;

查询常量：

SELECT constant1, constant2,... FROM dual;

3）LIKE（模糊查询）

通配符：% \_

%：匹配0-n个任意字符

\_：匹配1个任意字符

LIKE String ESCAPE char 转义字符

示例：

/\* 搜索含有30%的字符串，\做转义字符 \*/

LIKE ’%30\%%’ ESCAPE ’\’

4）GROUP BY 、 HAVING 、 ORDER BY

5）CASE WHEN

CASE WHEN boolean THEN .. ELSE .. END

示例：

SELECT

COUNT(CASE WHEN price < 13 THEN 1 ELSE NULL END) low,

COUNT(CASE WHEN price BETWEEN 13 AND 15 THEN 1 ELSE NULL END) med,

COUNT(CASE WHEN price > 15 THEN 1 ELSE NULL END) high

FROM products;

/\* COUNT(List<Object> cols)作为聚合方法，在没有分组的情况下，将视全表的记录为1个组，然后执行聚合运算 \*/

6）

7）

///

### TCL

1）COMMIT / SAVEPOINT / ROLLBACK

2）

3）

4）

5）

///

### 高级查询

#### 集合运算

1）集合运算就是将2个或者2个以上结果集组合成为一个结果集。

2）集合运算：交集、并集、补集

INTERSECT （交集）

UNION ALL （并集）：包括重复记录。

UNION （并集）：不包括重复记录。

MINUS（补集）：第一个查询结果集减去第二个查询结果集。

注意：查询返回的结果集的列数以及列的类型必须一一对应，列名可以不同。

3）示例：

查出dept表中哪个部门下没有员工，只需求出dept部门中的部门号与emp表中的部门补集即可：

(SELECT deptno FROM dept)

MINUS

(SELECTDISTINCT deptno FROM emp);

4）

5）

///

#### 表连接

1）交叉连接

/\* 无条件的连接，作笛卡儿积 \*/

tb1 CROSS JOIN tb2

2）内连接

/\* 有条件的连接，2张表行一一匹配 \*/

tb1 [INNER] JOIN tb2 ON(condition...)

3）外连接

/\* 可以省略关键字 OUTER (+)出现在WHERE子句中，标志[补充]值\*/

左外连接：

tb1 LEFT [OUTER] JOIN tb2 ON(xx)

WHERE tb1.xx = tb2.xx(+)

右外连接：

tb1 RIGHT [OUTER] JOIN tb2 ON(xx)

WHERE tb1.xx(+) = tb2.xx

全外连接：

tb1 FULL [OUTER] JOIN tb2 ON(xx)

WHERE tb1.xx(+) = tb2.xx(+)

4）

5）

///

#### 子查询

1）在INSERT / DELETE / UPDATE / SELECT语句内部都可以出现SELECT语句，内部的SELECT语句结果可以作为外部语句中[条件子句]的一部分，也可以作为外部查询的临时表。

/\* 子查询用括号括起来，里面写SELECT语句，如果做临时表可以加表别名 \*/

2）子查询类型：

单行子查询：返回0-1行结果。

多行子查询：返回0-n行结果。

3）关系分析

ANY-任意一个

/\*大于任意一个,即大于子查询的最小值即可 \*/

> ANY(多行子查询)

/\* 小于任意一个,即小于子查询最大值即可 \*/

< ANY(多行子查询) 比子查询最小值还要小

= ANY(多行子查询)等价于 IN (多行子查询)

------

ALL-所有

/\* 大于所有,大于子查询最大值 \*/

> ALL(多行子查询)

/\* 小于所有,小于子查询最小值 \*/

< ALL(多行子查询)

= ALL(多行子查询) 等价于 IN (多行子查询)

------

SOME-某些

SOME与ANY的功能相同。

4）

5）

///

#### 伪列

1）伪列就像表中的列一样，但在表中并不存储。

伪列只能查询，不能进行增删改操作。

2）

3）

4）

5）

///

## PlSql

1）PL：Procedural Language 过程化语言

属于第三代语言，关注细节的处理，可以用来实现复杂的业务逻辑。

PL/SQL：联合Db语言和过程语言。

由于SQL属于第四代语言，不能处理程序细节，因此PL作为第三代语言，解决这个问题。

PL/SQL\_组成：数据库PL/SQL、工具PL/SQL

2）PL/SQL的基本单位：区段。

区段：声明、可运行、排除/构建部分。

PL/SQL区段只被编译一次，并且以可运行的形式存储，以降低响应时间。

3）PLSQL允许混合SQL声明和过程结构。

4）调试动作

跳入：Step In

跳过：Step Over

跳出：Step Out

5）控制结构：

顺序、条件、循环结构

6）PlSql程序以块（block）为基本单位，整个PlSql块分为3部分：

声明部分：DECLARE

执行部分：BEGIN

异常处理：EXCEPTION

END /\* 在块的最后以END结束 \*/

7）Db管理系统\_输出

DBMS\_OUTPUT.PUT\_LINE(String msg);

8）以分号结尾

语句;

END;

9）

10）

///

基本要义

1）相等判断 =

2）赋值 :=

var\_name datatype := val;

示例：

v\_empno NUMBER(4, 0) := 888;

3）

///

### 函数

1）CREATE OR REPLACE FUNCTION

fun\_name(param...)

RETURN retval

AS / IS

plsql statement;

2）参数传递方法

①位置表示法：

argument\_value, ...

②名称表示法：

argument => parameter,...

③混合表示法：同时采用位置表示、名称表示。

使用位置表示法所传递的参数必须放置在名称表示法的前面，只要有一个参数使用名称表示法，其他的所有参数都必须采用名称表示法传参。

3）实际参数与形式参数数据传递方法：

①传址法：将实际参数的地址指针传递给形式参数，使形式参数和实际参数指向内存中的同一区域。

/\* 参照法 \*/

②传值法：将实际参数的数据拷贝到形式参数，而不是传地址。

4）可以使用DEFAULT关键字为输入参数指定默认值；不能为输入/输出参数指定默认值。

5）

///

### 过程

CREATE OR REPLACE PROCEDURE

proc\_name

AS

plsql statement;

///

游标

1）游标\_操作\_步骤

声明游标：

CURSOR cur\_name IS selectSql;

打开游标：OPEN cur\_name;

使用游标：FETCH cur\_name INTO ...;

关闭游标:CLOSE cur\_name;

2）

3）

4）

///

### Developer

1）快捷键

F6：代码助手

2）

3）

4）

5）

///

## 通用命令

( Cmd-Sql环境下 )

1）环境说明

进入Oracle的SQL环境，支持Oracle的SQL命令操作。

SQL> xx\_order...

2）用户\_操作:

创建用户:

CREATE USER user\_name

IDENTIFIED BY pwd

PROFILE DEFAULT

DEFAULT TABLESPACE tb\_space

TEMPORARY TABLESPACE temp\_tb\_space

ACCOUNT UNLOCK/LOCK;

修改用户:

ALTER USER user\_name IDENTIFIED BY new\_pwd REPLACE old\_pwd; /\* 修改密码 \*/

ALTER USER user\_name ACCOUNT LOCK/UNLOCK; /\* 用户锁定/解锁 \*/

通过ALTER USER

3）Oracle安装完毕之后，锁定所有的帐户，只开放sys / system 2个帐户, 以dba身份登录。

登录之后，可以开放其他帐户，或者创建新帐户。

4）

5）show命令

show all /\* 查看所有的系统变量 \*/

show user /\* 显示当前连接用户 \*/

show error /\* 显示错误 \*/

6）set命令

7）TNS：Transparent Network Substrate

透明网络底层

8）

9）

///

Linux环境下

/\* 切换到 Oracle 用户 \*/

Linux > su - oracle

###########

监听器 listener control

/\* 启动监听 \*/

SQL > lsnrctl start;

/\* 停止监听 \*/

SQL > lsnrctl stop;

/\* 查看监听状态 \*/

SQL > lsnrctl status;

/\* 进入 sqlplus 环境 \*/

xx> sqlplus /nolog /\* nolog 表示不登录 \*/

/\* 本地：以 [管理员] 模式进入 \*/

SQL > conn /as sysdba;

/\* 启动 Oracle \*/

SQL > startup;

/\* 停止 Oracle \*/

SQL > shutdown normal /\* 默认参数（可以不加）、等待会话结束、等待事务结束、启动时不需要实例恢复 \*/

SQL > shutdown transactional /\* 不等待会话结束、等待事务结束、启动时不需要实例恢复 \*/

SQL >shutdown immediate

SQL > shutdown abort /\* 关闭 Oracle 时Cache的内容没有写入数据文件，没有提交的事务也没有回滚 \*/

/\* 远程连接 Oracle \*/

SQL > conn <username>/<password> @<Addr>:1521/<OracleInstanceSID> as [sysdba | sysopr | normal];

/\* 执行脚本 \*/

SQL > dbstart /\* 启动 Oracle 脚本 \*/

SQL > dbstop /\* 停止 Oracle 脚本 \*/

/\* 打开Oracle，允许数据库的访问 \*/

SQL > startup open;

/\* 启动Oracle，并开始介质恢复 \*/

SQL > startup open recover;

/\* 指定初始化文件，而不采用默认初始化文件 \*/

SQL > startup pfile=<init\_file\_path>;

/\* 终止当前 Oracle 的运行，重新正常启动 Oracle \*/

SQL > startup force;

/\* 查看表结构 \*/

SQL > desc <tb\_name>;

/\* 清屏-clear screen \*/

SQL > clear scr;

/\* 显示当前用户 \*/

SQL > show user;

/\* 清除输出端所有行-clear line \*/

SQL > cl; /\* 功能类似于 clear scr; \*/

/\* 插入（insert）指定text文本 \*/

SQL > i <xx>;

/\* 将缓冲区中的old字符串改为（change）new字符串 \*/

SQL > c /<old\_str>/<new\_str>;

/\* 删除缓冲区m 至 n行的内容 \*/

SQL > del m, n;

/\* 列出（list）第 n 行的内容 \*/

SQL > l <n>;

/\* 创建文件，并把缓冲区的内容保存到该文件 \*/

SQL > sav <file> cre;

/\* 如果文件存在，则追加，不存在则新建 \*/

SQL > sav <file> app;

/\* 如果文件存在，则覆盖，不存在则新建 \*/

SQL > sav <file> rep;

/\* \*/

SQL > sav <file> replace;

/\* 打开文件并列出缓冲区的内容 \*/

SQL > get <file> list;

/\* 打开文件但并不列出缓冲区的内容 \*/

SQL > get <file> nolist;

/\* 运行指定文件 \*/

SQL > start <file>;

/\* 屏幕\_滚动-spool \*/

/\* 保存输出结果 \*/

SQL > spo <file> cre

/\* 向文件追加输出结果 \*/

SQL > spo <file> app;

/\* 替换 \*/

SQL > spo <file> rep;

/\* 开启 spool \*/

SQL > spo out; /\* ../out.lst \*/

SQL > spo on; /\* ../on.lst \*/

/\* 关闭 spool \*/

SQL > spo off; /\* 关闭out / on 执行一次关闭一个 \*/

/\* 开启原变量值 \*/

SQL > set verify on;

/\* 关闭原变量值 \*/

SQL > set verify off;

/\*

Oracle 日志模式：归档日志模式（archivelog），非归档日志模式（noarchivelog）

\*/

/\* 设置Oracle日志模式：设置为归档模式 \*/

ALTER DATABASE archivelog;

/\* 设置Oracle日志模式：设置为非归档模式 \*/

ALTER DATABASE noarchivelog;

/\* 列出 Oracle 当前日志归档模式 \*/

SQL > archive log list;

/\* 运行.sql文件（sql脚本） \*/

SQL > @<sql\_file>; /\* 当前目录下，或指定全路径，或则是通过SQLPATH环境变量搜寻到的脚本文件 \*/

SQL > start <sql\_fiile>; /\* 当前目录下，或指定全路径，或则是通过SQLPATH环境变量搜寻到的脚本文件 \*/

/\* 在 Oracle 客户端创建一个环境变量 SQLPATH，指定多个.sql文件所在的路径，然后重启客户端（重新建立session），

就可以调用该环境变量了；@<sql\_file>如果在当前路找不到该文件，就回去SQLPATH设置的变量值去寻找相应的.sql文件 \*/

/\* 对当前的输入进行编辑 \*/

SQL > edit;

/\* 在.sql文件中，使用@@执行的文件在当前.sql文件在同一目录下，无需在另外指定路径，直接写文件名即可 \*/

@@<sql\_file\_name>

/\*\*

所有SQL语句，都可以在　SQL命令行模式下运行。

但是 sqplus 的 .sql脚本文件中，只能运行sql语句，不支持 oracle 命令；在.pdc文件中开启 SQL命令行模式。

\*/

/\* 执行缓冲区最后一行的内容，注意，缓冲区中的内容不能以分号结束 \*/

SQL > /;

///

1）Oracle客户端启动：

Cmd下

方式1：sqlplus

方式2：sqlplus /nolog

2）连接Oracle服务器：

SQL>conn 用户名/密码服务器连接字符串 as 连接身份

连接身份：sysdba / sysoper / normal

服务器连接字符串：@本地网络服务名，它代表了远程Db服务器的信息，也叫做TNS服务名称。

TNS： transparent network substrate

透明网络底层，它是内建于Oracle客户端的一个程序，用于进行远程连接，并将@本地网络服务名翻译成有用的连接信息。

/\* 一般来说，客户端都有一个名称为tnsnames.ora的文件，连接Oracle Db服务器时加载该文件，并将对应的tns服务名称转换层连接信息连接Db。因此，客户端知道该如何连接Db服务器，服务端的监听程序也在运行，监听到客户端的连接后就会接收请求，并检查连接信息是否正确与合法，连接正确则接收该请求，否则拒绝。 \*/

连接示例：

> sqlplus /nolog

sql> conn sys/oracle @orcl as sysdba

已连接



或者



3）

4）客户端无法连接Db服务端\_错误\_信息：

[ ORA-12514: TNS: 监听程序当前无法识别连接描述符中请求的服务 ]

5）环境：Oracle服务端-Windows

启动Oracle相关服务：（以11g为例）

/\* 启动监听器 \*/

net start OracleOraDb11g\_home1TNSListener

/\* 启动服务实例 xx-要启动的实例名 \*/

net start OracleServicexx

6）Oracle数据库角色：

①CONNECT：使用Oracle简单权限，可以与服务器建立连接会话（Session-客户端对服务器连接）。

②RESOURCE：更可靠和正式的用户Db用户可以授予此角色，可以创建表、序列、过程、触发器、索引等Db对象。

③DBA：拥有所有的系统权限，包括无限制的空间限额和给其他用户授予各种权限。

/\* 一般情况下，普通用户拥有CONNECT和RESOURCE两个角色即可进行常规的Db开发工作 \*/

7）Oracle权限:

系统权限、Db对象权限

系统权限：只能由DBA用户授权。

对象权限：由拥有该对象的用户授权。

授予权限：

GRANT 角色|权限 TO 角色|用户;

回收权限：

REVOKE 角色|权限 FROM 用户|角色;

修改用户密码：

ALTER USER user\_name IDENTIFIED BY new\_pwd REPLACE old\_pwd;

8）

9）

///

## Db结构

### 物理数据库

1）物理数据库结构包括各种文件:

数据文件、控制文件、参数文件、备份文件、归档文件、重做日志文件、报警和跟踪日志文件。

2）数据文件：存放数据库数据。Oracle数据库由一个或多个数据文件组成，数据文件结合在一起形成表空间。

3）

4）

5）

### 逻辑数据库

1）Oracle为所有的数据分配逻辑数据库空间。

数据库空间分配的单元是：数据块（data block）、盘区、段。

2）数据块（Data Block）

在最好的粒度层次上，Oracle将数据存储在数据块中，数据块也称为逻辑块、Oracle块、页。

3）盘区（Extent）

盘区由连续的数据块组成。

4）段（Segment）

由一组范围组成；

段\_类型：数据段、索引段、临时段、回滚段。

5）