1. 数据类型

1.1. 字符型

varchar2　　0-4000B　　变长字符串

nvarchar2　0-1000B　　Unicode 变长字符型

char　　0-2000B　　定长字符型

nchar　　0-1000B　　Unicode 定长字符型

long　　0-2GB　　变长字符串

1.2. 数字型

number(p, s) p 精度，最大 38 位十进制，s 小数位数；定长整数和小数

float　　最大 126 位二进制，转化为十进制乘以 0.30103

1.3. 日期型

date　　日期时间，-4712/1/1 至 9999/12/31，精确到秒

timestamp 日期时间，精确到小数秒，可显示上下午

1.4. 其他类型

blob　　最大 4GB，二进制数据

clob　　最大 4GB，字符串数据

bfile　　与 OS 有关，把非结构化二进制数据存储在数据库意外的 OS 文件中

2. 数据库定义语言(DDL)

2.1. 创建数据表

```
Create Table tableName
(
    fieldName1 fieldType [null | not null],      //定义非空约束
```

```
    fieldName2 fieldType Default “Value”      //定义默认值
    fieldName3 fieldType,
    fieldName4 fieldType not null,
    fieldName5 fieldType not null,
    Primary Key (fieldName4),           //定义主键，主键必须非空
    [Primary Key (fieldName4, fieldName5)],
    Foreign Key(fieldName) Reference fTableName(fFieldName)   //
    定义外键
)
```

2.2. 修改数据表

```
Alter Table tableName [Add | Drop |Modify] fieldName fieldType;
Alter Table tableName Drop fieldName [cascade constraints]; //
把有关约束一并删除
```

2.3. 删除数据表

```
Drop Table tableName;     //删除之前需要消除引用
Drop Table tableName;
```

2.4. 约束

```
//添加主键约束
Alter Table tableName Add Constraints constraintsName Primary Key
(fieldName);
//添加外键约束并添加级联删除
[Add] Constraint constrainName Foreign Key(fieldName) Reference
fTableName(fFieldName) On Delete CasCade;
//添加 Check 约束
[Add] Constraint constraintsName Check(condition);//condition 是
条件，如 age>=18 and age<=30
//添加 Unique 约束
[Add] Constraint constraintsName Unique(fieldName);
```

```
//删除约束
Alter Table tableName Drop Constraint constraintsName;
//Not Null 约束
Alter Table tableName Modify fieldName Not Null;
```

3. 数据的增删改（DML）

3.1. 插入数据

```
//约束会限制数据的插入
Insert Into tableName(fieldName1, fieldName2) Values(value1, value2);
//从其他表获取值插入，需列数和类型一致
Insert Into tableNameA(fieldName1, fieldName2) Select fieldName3, fieldName4 From tableNameB;
//创建时插入
Create Table tableNameA As Select fieldName1, fieldName2 From tableNameB;
```

3.2. 更新数据

```
Update tableName Set fieldName1 = value1, fieldName2 = value2;
Update tableName Set fieldName1 = value1 where fieldName2 = value2;
Merge [Into] tableNameA
Using tableNameB On (condition)
When Matched Then merge_update_clause
When Not Matched Then merge_insert_clause;
```

3.3. 删除数据

```
Delete From tableName;//删除表中全部数据
Delete From tableName Where filedName Expression value;
Truncate Table tableName;//全部删除，比 Delete 快
```

4. 数据的检索（DQL）

4.1. 检索

```
Select
   [Distinct | All]
      select_list
   From table_list
   [where_clause]
   [group_by_clause]
   [HAVING condition]
   [order_by_clause]
```

4.2. 显示列

```
*  |
[scheme.] {table | view}.*  |
expr [ [As] c_alias ] //设置别名，As 可省略
```

4.3. 操作列

```
Select '售价: ' || price*1.2 As 售价;
```

4.4. 排序

```
Order By
{ expr | position | c_alias }//表达式，位置，别名
[ASC | DESC]
[NULLS FIRST | NULLS LAST]   //对空字段的处理方式
[ , { expr | position | c_alias }     //多个字段排序
   [ASC | DESC]
   [NULLS FIRST | NULLS LAST]
]
```

4.5. 过滤

//关系操作符

<,>,< >,!=, <=,=, >=

//比较操作符

is Null,

Like, //_代表一个字符，%代表多个字符

Between…And…,

In

//逻辑操作符

and, or, not

4.6. 分组

Group By

{ expr |

{RollUp | Cube}({expr[, expr]…}) //小计，总计

}

没有出现在 Group By 子句中的列不能放到 Select 子句中，不能在 Where 语句中使用。

4.7. Having

给 Group By 添加限制条件

Group By age Having Count(*)>1

```
SELECT Email FROM Person GROUP BY Email HAVING count( * )>1;
```

4.8. 子查询

//子查询返回多行时使用

Any, //同 Some，任何一个，多用于非等

Some, //同 Any 多用于

All

4.9. 连接查询

//最简形式，将得到两张表的笛卡尔积

Select * From tableA, tableB;

//内连接，简单连接（等值连接、不等值连接）

[Inner] Join…On

//自连接

From tableA A, tableA B

//外连接

Left Join…On

Right Join…On

Full Join…On

+放在非主表一方，不推荐

5. 数据的控制（DCL）

6. 内置函数

6.1. 数值型函数

//绝对值

ABS()

//取余

Mod(n2, n1) //n2 除以 n1 的余数，n1 为 0 返回 n2

//正负判断

Sign(n) //正数返回 1，0 返回 0，负数返回-1。如果 n 是 Binary 精度，则>=0，=NaN 返回 1

//三角函数

Cos、Acoc、Cosh、Sin、Sinh、Asin、Tan、Tanh、Atan

//准正数函数

Ceil(n)//返回大于等于 n 的最小正数

Floor (n) //返回小于等于 n 的最大整数

//指数函数

Power(n2, n1)//返回 n2 的 n1 次幂，n2 为负时 n1 必须是整数。

Exp(n)//返回 e 的 n 次幂，e=2.71828183…

//对数函数

Sqrt(n)//求 n 的平方根，数字类型 n 不能为负。为 Binary 精度类型时，n<0 将返回 NaN

Log (n1, n2) //返回以 n1 为底 n2 的对数，n1 为除 1 和 0 外的任意正数

Ln(n)//返回 n 的自然对数，n 大于 0

//四舍五入

Round(for number)/ Round(n, integer)//integer 表示小数位数，负数向左

Trunc(for number)/ Trunc(n, integer) //integer 表示小数位数，负数向左只舍不入

6.2. 字符型函数

//ASCII 码与字符转换

CHR(n[Using Nchar_CS])//将给定 ASCII 码转换为字符

ASCII(char) //返回参数首字母的 ASCII 值

//获取字符串长度

Length(string)

//字符串截取

{[SubStr]|[SubStrb]|[SubStrc]|[SubStr2]|[SubStr4]}(char, position[,substring_length])//单位字符、字节、Unicode 字符、UCS2 代码点、UCS4 代码点

//字符串连接

Concat(char1, char2)//同||

//字符串搜索

{[InStr]| [InStrB]| [InStrC]| [InStr2]| [InStr4]}(string, subString[,position[,occurrence]])

//大小写转换

Upper(char)

Lower(char)

InitCap(chat)//首字母大写

//带排序参数的大小写转换

Nls_Upper(char[,nlsparam])// nlsparam 来自 NLS_DATABASE_PARAMETERS

Nls_Lower(char[,nlsparam])//'NLS_SORT = SCHINESE_PINYIN_M'

Nls_InitCap(chat[,nlsparam])

//为指定参数排序

NlsSort(char[,nlsparam])

//替换字符串

Replace(char, search_string[,replacement_string])

//字符串填充

RPad(expr1,n[,expr2])//在 expr1 右边用 expr2 填充，直到长度为 n

LPad(expr1,n[,expr2])//expr2 默认为空格

//删除首尾指定字符

Trim([Leading|Trailing|Both][trim_character From]trim_source)//默认删除前后空格

RTrim[char[,set]]

LTrim[char[,set]]

//字符集和 ID 互换

Nls_Charset_Id(string)//得到字符集名称对应 ID

Nls_Charset_Name(number)//根据字符集 ID 得到对应名称

6.3. 日期型函数

//系统日期、时间

```
To_Char(SysDate,,'YYYY-MM-DD HH24:MI:SS')

To_Char(Current_Date,,'YYYY-MM-DD HH24:MI:SS')// //会话所在时区的当前日期


SysTimestamp

//时区

DbTimeZone//数据库时区

SessionTimeZone//当前会话时区

//日期加月份

Add_Months(date, integer)//data需用To_Date转换

//返回某月最后一天

Last_Day(date)

//一周后的某天

Next_Day(date, char)//char表示星期几

//提取日期的特定部分

Extract(datetime)

/*

Extract(Year From SysDate)

Extract(Minute From SysDate Timestamp '2015-12-30 17:02:00')

Extract(Second From SysDate Timestamp '2015-12-30 17:02:00')

*/

//得到两个日期之间的月份数

Months_Between(date1, date2)

//时区时间转换

New_Time(date,timezone1, timezone2)

//时间四舍五入、截取

Round(date[,fmt])//舍入

Trunc(date[,fmt])//截取
```

6.4. 转换函数

//字符串转 ASCII 类型字符串

AsciiStr(char)

//二进制转十进制

Bin_To_Num(data,[,data…])

//数据类型转换

Cast(expr as type_name)//把 expr 转成 type_name 类型

//字符串与 RowId 互转

CharToRowId(char)//将字符串转成 RowId 类型，RowId 长度为 18 位

RowIdToChar(rowid)//将 RowId 类型转成字符串

RowIdToNChar(rowid)//将 RowId 类型转成字符串

//字符串在字符集间转换

Convert(char, dest_char_sets[,source_char_set])//Char VarChar2 NChar NVarChar2 Clob NClob

//十六进制字符串与 Raw 转换

HexToRaw(char)

RawToHex(raw)

RawToNHex(raw)

//数值转换成字符

To_Char(number[,fmt[,nlsparam]])

To_Char(date[,fmt[,nlsparam]])

//字符转日期

To_Date(char[,fmt[,nlsparam]])

//字符串转数字

To_Number(expr[,fmt[,nlsparam]])

//全角转半角

To_Single_Byte(char)

6.5. NUll 函数

//返回表达式为 Null 的函数

Coalesce//返回表达式中第一个不为 Null 的表达式，全为 Null 则返回 Null

//排除指定条件函数

Lnnvl(condition)//得到除了 condition 要求条件之外的数据

//替换 Null 值

Nvl(exp1, exp2)//expr1!=null?expr1:expr2

Nvl2(exp1, exp2, exp3)// expr1!=null?expr2:expr3

6.6. 集合函数

//求平均值

Avg([Distinct | All] expr)

//求记录数量

Count([* | Distinct | All] expr)

//求最大、最小值

Max([Distinct | All] expr)

Min([Distinct | All] expr)

//求和

Sum([Distinct | All] expr)

6.7. 其他函数

//返回登录名

User

//返回会话和上下文信息

UserEnv(parameter)// parameter = [Language|SessionId|IsDba]

Sys_Context(nameSpace,parameter)//UserEnv, Session_User

//表达式匹配

Decode(expr, search, result[,search1, result1][,default])

7. 索引与约束