基本数据类型

n：最多字符数量

p（有效位数）：字段的最大长度

s（标度）：小数点后面的位数

|  |  |  |  |
| --- | --- | --- | --- |
|  | CHARACTER(n) | | 通常用空格填充数量不足的字符。 |
|  | CHARACTER VARYING(n) | |  |
|  | BLOB | |  |
|  | TEXT | |  |
| 数值类型 | BIT(n) | 通用的数值类型NUMERIC(p) |  |
| BIT VARYING(n) |  |
| DECIMAL(p,s) |  |
| INTEGER |  |
| SMALLINT |  |
| BIGINT |  |
| FLOAT(p,s) |  |
| DOUBLE PRECISION(p,s) |  |
| REAL(s) |  |
| 日期和时间类型 | DATE | |  |
| TIME | |  |
| DATETIME | |  |
| TIMESTAMP | |  |
| 布尔值 |  | | 取值： TRUE、FALSE、NULL。 |
| NULL数据类型 |  | | NULL |

自定义类型

|  |  |
| --- | --- |
| CREATE TYPE **数据类型名** AS OBJECT  (**成员名** **数据类型**(,  **成员名** **数据类型**)\*); |  |

域：能够被使用的有效数据类型的集合。

|  |  |
| --- | --- |
| 创建域  CREATE DOMAIN **域名** AS **数据类型**; |  |
| 给域添加约束  ALTER DOMAIN **域名**  ADD CONSTRAINT **约束名**  CHECK(**关于VALUE的表达式**); | 1、**关于VALUE的表达式**的数据类型是布尔值。 |

|  |  |
| --- | --- |
| from子句  FROM **表名**( **别名**)?(,**表名**( **别名**)?)\* |  |
| where子句  WHERE **关于表名**.**列名**(,**表名**.**列名**)\***的表达式**  外连接：在**表名**.**列名**后插入(+)，让比较操作符另一端的字段的本不能出现的所有值各出现一次，得到的所有记录中该字段的值当然都是NULL。 | 1、**关于表名**.**列名**(,**表名**.**列名**)\***的表达式**的数据类型是布尔值。 |
| group by子句  GROUP BY (**表名**.**列名**|**列号**)(,(**表名**.**列名**|**列号**))\*  GROUP BY ROLLUP(**表名**.**列名**(,**表名**.**列名**)\*)  GROUP BY CUBE(**表名**.**列名**(,**表名**.**列名**)\*) | 1、ROLLUP：每次去掉最右边1个字段。按[去之前的所有字段]的汇总记录后紧跟着按[去之后的所有字段]的汇总记录（1条，去掉的字段的值为NULL）。（最后1行为全NULL，相当于树根。从下往上缕就是DFS。）  2、CUBE：去掉若干个字段的所有可能方式都来一遍。（最后1行为全NULL。）  例：GROUP BY CUBE(列1,列2)得到   |  |  |  | | --- | --- | --- | |  | ┏ | 属于下面一行的[根据列1和列2的值得到的]所有组  ... | | ┏ | 根据列2的值得到的第1组（列1的位置是NULL） | | | ┇ |  | | | ┃ | ┏ | 属于下面一行的[根据列1和列2的值得到的]所有组  ... | | ┣ | 根据列2的值得到的最后1组（列1的位置是NULL） | | | ┃ | ┏ | 根据列1的值得到的第1组（列2的位置是NULL） | | ┇ | ┇ |  | | ┣ | ┣ | 根据列1的值得到的最后1组（列2的位置是NULL） | | （列1和列2的位置是NULL） | | | |
| having子句  HAVING **关于汇总函数**(,**汇总函数**)\***的表达式** |  |

数据定义语言（DDL）

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CREATE TABLE语句  CREATE TABLE **表名**  (**列名** **数据类型**( | NULL| NOT NULL)(,  **列名** **数据类型**( | NULL| NOT NULL))\*);  CREATE TABLE 表名;  AS **记录集合**; | 1、   |  |  | | --- | --- | | 无 | 字段可以为NULL。 | | NULL | | NOT NULL | 字段不可以为NULL。 |   2、例：   |  |  |  | | --- | --- | --- | | MySQL、Oracle | CREATE TABLE **表名**  AS **记录集合**; | 1、**记录集合**：**SELECT语句去掉末尾的分号**。 | | Microsoft SQL Server | SELECT (\*|**列名**(,**列名**)\*)  INTO **表名**  FROM **表名**(  **where子句**)?; |  | |
| ALTER TABLE命令  ALTER TABLE **表名**  ( modify)?( column **列名**)?( **数据类型**|NULL|NOT NULL)?  ( | restrict| cascade)  ( drop)?( constriant **约束名**)?  ( add)?( column)? **列名** **数据类型**( | NULL| NOT NULL); |  |
| DROP TABLE **表名**( | restrict| cascade) | 1、   |  |  | | --- | --- | | 无 |  | | restrict | 若表被视图或约束所引用，返回一个错误。 | | cascade | 若表被视图或约束所引用，删除所有引用视图和约束。 | |

完整性约束

|  |  |
| --- | --- |
| 主键约束  在CREATE TABLE语句中，在( | NULL| NOT NULL)后插入 PRIMARY KEY。  或  在CREATE TABLE语句中，在);前插入,PRIMARY KEY(**列名**(,**列名**)\*)。  或  ALTER TABLE **表名**  ADD CONSTRAINT **约束名** PRIMARY KEY(**列名**(,**列名**)\*); |  |
| 唯一性约束  在CREATE TABLE语句中，在( | NULL| NOT NULL)后插入 UNIQUE。 |  |
| 外键约束  在CREATE TABLE语句中，在);前插入,CONSTRAINT **约束名** FOREIGN KEY(**列名**) REFERENCES **表名**(**列名**)。  或  ALTER TABLE **表名**  ADD CONSTRAINT **约束名** FOREIGN KEY(**列名**) REFERENCES **表名**(**列名**); |  |
| NOT NULL约束 |  |
| 检查约束  在CREATE TABLE语句中，在);前插入,CONSTRAINT **约束名** CHECK(**关于列名的表达式**)。 | 1、**关于列名的表达式**的数据类型是布尔值。 |
| 去除约束  ALTER TABLE **表名**  DROP CONSTRAINT **约束名**; | 1、例：   |  |  | | --- | --- | | MySQL | ALTER TABLE **表名**  DROP PRIMAARY KEY; | |

数据操作语言（DML）

|  |  |
| --- | --- |
| INSERT INTO **表名**  VALUES(**值**(,**值**)\*);  INSERT INTO **表名**(**列名**(,**列名**)\*)  VALUES(**值**(,**值**)\*);  INSERT INTO **表名**  **记录集合**; | 1、没有出现的列没有NOT NULL约束。  2、没有出现的列插入NULL。 |
| UPDATE **表名**  SET **列名**=**值**(,**列名**=**值**)\*(  **where子句**)?; |  |
| DELETE FROM **表名**(  **where子句**)?; |  |

数据查询语言（DQL）

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SELECT (\*|(|ALL |DISTINCT )(**列名**|**关于汇总函数的表达式**)  (,(**列名**|**关于汇总函数的表达式**))\*)  **from子句**(  **where子句**)?(  **group by子句**)?(  ORDER BY (**表名**.**列名**|**列号**)(| ASC| DESC)  (,(**表名**.**列名**|**列号**)(| ASC| DESC))\*)?;  得到记录集合。   |  |  | | --- | --- | | 统计记录数量 | SELECT COUNT(\*)  FROM **表名**; | | 统计字段的值的数量 | SELECT COUNT(**列名**)  FROM **表名**; | | 统计列的值的集合的基数 | SELECT COUNT(DISTINCT **列名**)  FROM **表名**; | | 1、   |  |  | | --- | --- | | 无 | 不去重 | | ALL | | DISTINCT | 去重 |   2、   |  |  | | --- | --- | | 无 | 升序 | | ASC | | DESC | 降序 |   3、列号是某一列在SELECT后列表里的位置，从1开始。  4、被直接选中的字段必须出现在GROUP BY子句里。  5、GROUP BY子句也有排序的功能。优先级比ORDER BY子句低。  ！！！使用GROUP BY子句且不使用汇总函数没有意义。 |
| 给字段创建临时别名  在SELECT语句中，在**列名**后插入 **别名**。 |  |

运算符

|  |  |  |
| --- | --- | --- |
| 比较操作符 | < |  |
| > |  |
| <= |  |
| >= |  |
| = |  |
| <> |  |
| 逻辑操作符 | IS NULL | **字段名** IS NULL |
| BETWEEN | **字段名** BETWEEN **实数** AND **实数** |
| IN | **字段名** IN (**值**(,**值**)\*) |
| LIKE | **字段名** LIKE **字符串**  **字符串**中%表示至少0个字符，\_表示1个字符。 |
| EXISTS |  |
| UNIQUE |  |
| ALL | **字段名** **比较操作符** ALL **记录集合** |
| ANY | **字段名** **比较操作符** ANY **记录集合** |
| 连接操作符 | NOT |  |
| AND |  |
| OR |  |
| 算术操作符 | \* |  |
| / |  |
| + |  |
| - |  |

汇总函数

|  |  |  |  |
| --- | --- | --- | --- |
| COUNT | COUNT((\*|(|ALL |DISTINCT )**列名**(,**列名**)\*)) | 得到以表达式为字段名的一个字段。 | 1、包括NULL。 |
| SUM | SUM((|ALL |DISTINCT )**列名**) |  |  |
| MAX | MAX((|ALL |DISTINCT )**列名**) |  | 1、不包括NULL。 |
| MIN | MIN((|ALL |DISTINCT )**列名**) |  | 1、不包括NULL。 |
| AVG | AVG((|ALL |DISTINCT )**列名**) |  |  |

字符串函数

|  |  |  |  |
| --- | --- | --- | --- |
| 串接 |  |  |  |
| TRANSLATE | TRANSLATE(**字段名**,**字符串1**,**字符串2**) | 得到用字符替换字符后的字符串。 | 1、字符串1和字符串2长度相等。 |
| REPLACE | REPLACE(**字段名**,**字符串1**,**字符串2**) |  |  |
| UPPER | UPPER(**字段名**) |  |  |
| LOWER | LOWER(**字段名**) |  |  |
| SUBSTR | SUBSTR(**字段名**,**开始位置**,**长度**) |  | 1、第1个字符位置是1。 |
| INSTR | INSTR(**字段名**,**字符串**,**开始位置**,**第几次出现**) | 得到字符出现的位置。 | 1、第1个字符位置是1。 |
| LTRIM | LTRIM(**字段名**,**字符串**) | 得到去掉左边的1个[**字符串**或**字符串的后缀**]后的字符串 |  |
| RTRIM | RTRIM(**字段名**,**字符串**) | 得到去掉右边的1个[**字符串**或**字符串的前缀**]后的字符串 |  |
| DECODE | DECODE(**字段名**,**查找字符串1**,**返回字符串1**((,**查找字符串**,**返回字符串**)\*,**缺省返回字符串**)?) |  |  |
| LENGTH |  |  |  |
| IFNULL | IFNULL(**字段名**,**替代值**) |  |  |
| COALESCE | COALESCE(**字段名**(,**字段名**)\*) | 若有非NULL值，得到第1个非NULL值，否则得到NULL。 |  |
| LPAD | LPAD(**字段名**,**总字符数**,**字符**) |  |  |
| RPAD | RPAD(**字段名**,**总字符数**,**字符**) |  |  |
| ASCII | ASCII(**字符**) | 得到字符的ASCII码。 |  |

算术函数

|  |  |
| --- | --- |
| ABS |  |
| ROUND |  |
| SQRT |  |
| SIGN |  |
| POWER |  |
| CEIL |  |
| FLOOR |  |
| EXP |  |
| SIN |  |
| COS |  |
| TAN |  |

转换函数

|  |  |
| --- | --- |
| 字符串转换为数字 |  |
| 数字转换为字符串 |  |
| 日期转换为字符串 |  |
| 字符串转换为日期 |  |

数据控制语言（DCL）

数据管理命令

事务控制命令

事务：若干个DML语句（INSERT、UPDATE、DELETE）。

|  |  |
| --- | --- |
| COMMIT( WORK)?; | 把事务所做的修改保存到数据库，清空临时回退区域。 |
| ROLLBACK( WORK)?; | 撤销上一个COMMIT或ROLLBACK命令之后的事务。 |
| ROLLBACK TO **保存点名**; | 回退到一个保存点。 |
| SAVEPOINT **保存点名**; | 1、保存点名互不相同。  2、例：   |  |  | | --- | --- | | Microsoft SQL Server | SAVE TRANSACTION**保存点名**; | |
| RELESE SAVEPOINT **保存点名**; | 删除保存点。 |
| SET TRANSACTION (READ WRITE|READ ONLY); | 指定事务的特性。READ ONLY避免对数据库对象进行加锁，可以提高事务的性能。 |