SQL语句的概述

SQL语言的分类

数据定义语言（Data Definition Language）主要用于修改、创建和删除数据库对象，其中包括CREATE ALTER DROP语句。

数据查询语言（Data Query Language）主要用于查询数据库中的数据，其主要是SELECT语句，SELECT语句包括五个子句，分别是FROM WHERE HAVING GROUP BY 和WITH语句。

数据操作语言（Data Manipulation Language）主要用于更新数据库里数据表中的数据，包括INSERT UODATE DELETE语句。

数据控制语言（Data Control Language）主要用于授予和回收访问数据库的某种权限。包括GRANT REVOKE等语句。

事物控制语言，主要用于数据库对事物的控制，保证数据库中数据的一致性，包括COMMIT ROLLBACK语句。

常用的数据类型

MYSQL：

整型

|  |  |
| --- | --- |
| 整数类型 | 取值范围 |
| TINYINT | -128至127（1字节） |
| SMALLINT | -32768至32767（2字节） |
| MEDIUMINT | -sqrt(2,23)至sqrt(2,23)-1 |
| INT | -sqrt(2,31)至sqrt(2,31)-1 |
| BIGINT | -sqrt(2,63)至sqrt(2,63)-1 |

浮点型

|  |  |
| --- | --- |
| 浮点数类型 | 取值范围 |
| FLOAT(M,N) |  |
| DOUBLE(M,N) |  |

字符类型

|  |  |
| --- | --- |
| CHAR(N) | 定长1至255个字符 |
| VARCHAR(N) | 可变长1至65535 |
| TINYTEXT(N) | 可变长1至65535 |
| TEXT(N) |  |
| MEDIUMTEXT(N) |  |
| LONGTEXT(N) |  |

SQL语句的书写规范

SQL语句中不区分关键字的大小写

SQL语句中不区分列名和对象名的大小写

SQL语句对数据库中数据的大小写敏感

SQL语句中使用--注释，当使用--注释时，--后面至少有--个空格，多行注释用/\* \*/

数据库的创建与删除

数据库的创建

CREATE DATABASE database\_name;

在MYSQL COMMAND LINE CLIENT中书写SQL语句时，在SQL语句后面都要加上分号

数据库的删除

DROP DATABASE database\_name;

数据表的创建与更新

数据库中的表

数据记录：在数据表中的每一行被称为数据记录

字段：数据表中的每一列被称为字段

主键（PRIMARY KEY）：作为数据表中唯一的表示，保证了每一天数记录的唯一性。主键在关系数据库中约束实体完整性。所谓的实体完整性，是指对数据表中行的约束。

外键（FOREIGN KEY）：外键用来定义表与表之间的关系。在数据表中，如果属性列F是关系B中的一个属性（并不是关系B的主键），并且属性列F是关系A的主键，则F就是B的外键。外键在关系数据库中约束参照完整性。所谓的参照完整性是指表与表之间的约束。

索引：索引是指向数据表中的一个指针，指向索引字段在数据表中的物理位置。一般可以在如下几种情况下建立索引

在主键列中创建索引

多表连接时，在经常使用的连接列上创建索引

在经常使用WHERE子句查询的列上创建索引

在经常进行分组GROUP BY及排序ORDER BY的列上创建索引

约束：为了保证数据的完整需要使用约束

1. 唯一约束（UNIQUE）使用唯一约束的某一列或者某一组中没有相同的值，即保证了值的唯一性。但是唯一约束中可以插入NULL值
2. 主键约束（PRIMARY KEY）保证使用主键约束的列中只能有唯一的值，并且不能包含NULL值，数据表中每一列只能定义一个PRIMARY KEY
3. 外键约束（FOREIGN KEY）保证表参照完整性，确保对一个数据表的数据操作不会对与之关联的表造成不利的影响
4. 检查约束（CHECK）限制列的取值范围或者取值条件，可以为一个列定义多个CHECK约束
5. 非空约束（NOT NULL）用于对列的约束

创建数据表

CREATE TABLE table\_name(

column\_name1 datatype [constraint\_condition1]

[,column name2 datatype [constraint\_condition2]

…)

使用约束

1.唯一约束（UNIQUE）用来保证某一列或者某一组列中没有相同的值，一般为列创建了一个唯一约束后，数据库会自动为该列建立一个唯一索引，其索引名与约束名相同

例如：CREATE TABLE T\_dept(

dept\_ID VARCHAR(15) UNIQUE,

dept\_Name VARCHAR(10)

2.主键约束

CREATE TABLE T\_result(

stuID VARCHAR(15),

cruID VARCHAR(15),

result DOUBLE,

PRIMARY KEY(stuID,curID)

)

3.外键约束

FOREIGN KEY[表名1]（列名1） REFERENCES 表名2 （列名2）

[ON UPDATE [CASCADE]|[SET NULL]|[RESTRICT]]

[ON DELETE [CASCADE]|[SET NULL]|[RESTRICT]]

其中，FOREIGN KEY是关键字，表名1是可选的，列名1是指定数据表中用于外键约束的外键，表名2表示主表的名字，列名2主表中与从表中列名1对应列的名字，后面的ON UPDATE ON DELETE表示对表中的数据的修改或者删除，主从表之间采取什么样的操作方式，是可选的

CASCADE:级联删除，如果主表中一条数据记录被删除，从表中的数据也将删除

SET NULL:置空删除，如果主表中的一条数据记录南北删除，从表中与之相连的数据也将置空

RESTRICT:受限删除，如果主表中的一条数据被删除，则在执行DELETE语句时系统会报错，通知用户与主表相对应的数据在从表中仍然存在，但是与主表相对应的数据在从表中不被删除，它是默认的方式

CREATE TABLE T\_result(

stuID VARCHAR(15),

curID VARCHAR(15),

result DOUBLE,

FOREIGN KEY (stuID) REFERCES T\_student (stuID) ON DELETE CASCADE,

PRIMARY KEY(stuID,curID)

)

3.检查约束CHECK

CREATE TABLE T\_curriculum(

curID VARCHAR(15) PRIMARY KEY,

curName VARCHAR(10),

credit INT,

CHECK(credit BETWEEN 3 AND 8)

)

4.非空约束NOT NUL

使用索引

1. 唯一索引：在数据表中使用UNIQUE可以为一个数据列定义一个唯一索引，唯一索引中的每一个索引值只对应数据表中的一条记录，保证了数据列中记录的唯一性
2. 主索引：在数据表中使用PRIMARY KEY可以为一个数据列定义一个主索引，所谓的主索引就是在主键列中建立索引
3. 单列索引：定义在数据表中一个数据列上的索引就是单列索引，一般在数据查询时，如果WHERE子句中经常用到的数据表中的某一列作为查询条件，就可以把该列创建为单列索引
4. 复合索引：索引可以定义在一个数据表的多个数据列上，像这样的索引被称为复合索引
5. 聚簇索引：

创建与删除索引

CREATE [UNIQUE]|[CLUSTER] INDEX index\_name

ON table\_name(column\_name [排序方式]…)

排序：关键字ASC表示升序排列，关键字DESC表示降序排列

修改数据库中的表

1. 向表中增加一列

ALTER TABLE table\_name ADD(column\_name datatype [constraint\_condition])

例如：ALTER TABLE T\_teacher ADD salary INT NOT NULL

1. 增加一个约束

ALTER TABLE table\_name ADD constraint\_type (column\_name)

例如：ALTER TABLE dept ADD PRIMARY KEY (dept\_ID)

1. 增加一个索引

ALTER TABLE table\_name ADD INDEX(column\_name1[,column\_name2]…)

例如：ALTER TABLE T\_curriculum ADD INDEX i\_credit(credit)

1. 修改表中的某一列

ALTER TABLE table\_name MODIFY column\_name datatype

例如：ALTER TABLE T\_student MODIFY sex CHAR(2)

1. 删除表中的某一列

ALTER TABLE table\_name DROP column\_name

1. 删除一个约束条件

ALTER TABLE table\_name DROP constraint\_type

1. 删除数据库中的表

DROP TABLE table\_name[CASCADE CONSTRAINTS]

删除表时要使用CASCADE CONSTRAINTS级联删除从表中的外键约束

数据的查询操作

查询全部列的数据记录

SELECT \* FROM table\_name or view\_name[,table\_name1 or view\_name1…]

查询表中指定的列

SELECT 目标列[,目标列…]

FROM 表名或者视图名[,表名或者视图名…]

在SELECT语句中查询数据表或者视图指定的列时，在SELECT语句中指定的列名必须是指定的数据表或者视图中存在的列

查询表中不重复的记录

SELECT DISTINCT 目标列[,目标列…]

FROM 表名或者视图名[,表名或者视图名…]

使用列别名查询

SELECT 目标列 [AS] 列别名 [,目标列 [AS] 列别名…]

FROM 表名或者视图名 [,表名或者视图名…]

在SELECT语句中使用列别名的形式对数据表或者视图查询时，可以再在查询的列名后面使用一个空格键来代替AS，空格后面再跟上列别名的名字。如果列别名对字母大小写敏感，或者在使用包含有空格或者特殊字符的列别名，则必须使用单引号或者双引号将其引起来，否则系统将报错

对查询的记录进行运算

SELECT语句还可以使用算数运算符对指定的列进行运算，其中算术运算符包括加减乘除以及模除，在SELECT语句中对指定的列进行算术运算符进行运算时，只会改变显示的结果，并不会改变数据表中列的原有值

使用连接符（||）连接字段

在ORACLE中，可以使用||连接符连接多个字段，在使用连接符时，连接的数据类型应该是相同的，如果不同系统将会报错，如果需要在连接的字段中加入字符或者是日期类型的值，需要将该字符值或者日期类型的值使用单引号引起来。注意，在MYSQL和MS SQL SERVER中并不支持连接符||，在MYSQL中可以使用CONCAT函数，在MS SQL SERVER中可以使用+来连接

关于NULL值

在数据库中，如果没有为该列赋值，而且该列也没有默认值，此时查询的结果就为空，即NULL

使用WHERE子句查询表中满足条件的记录

比较查询

比较运算符分为三类，算术比较运算符，BETWEEN AND ，IN

算术比较运算符

SQL语句中的算术比较运算符主要包括 = >= <= > < != <>（不等于） !>（不大于） !< （不小于），在SELECT 语句的WHERE子句中可以使用算术比较运算符对指定的列进行比较，应指出的是，在MS SQL SERVER中，<>不是不等于的意思

WHERE 字段1 比较运算符 值

在SQL语句中，如果在WHERE子句中比较的时整数型数据，则可以不用使用单引号，如果比较的时其他的数剧类型，如字符串、日期等，则必须用单引号引起来，另外，WHERE 子句中比较运算符的左侧和右侧的数据类型必须是类型兼容的

BETWEEN AND运算符

字段1 BETWEEN 值1 AND 值2

在SQL中使用BETWEEN AND 运算符可以查询指定条件范围的记录，也可以使用

NOT BETWEEN AND运算符来排除一些记录

IN运算符查询与列表匹配的记录

字段1 IN （属性值1，属性值2……）

在SQL语句中也可以使用NOT IN来排除一些记录

SELECT teaID,teaName,age,sex,dept

FROM T\_teacher

WHERE dept IN(‘计算机系’,’数学系’)

字符串比较

使用比较运算符对字符串进行比较

在使用比较运算符对字符串进行查询和比较时，有的数据库需要区分大小写，如ORACLE，但是在MS SQL SERVER 和MYSQL中不区分，为了避免由于不注意字符串大小写而查询不到正确的结果，可以使用干将字符串转换为大小的函数UPPER LOWER对字符串进行转换

使用BINARY关键字对字符串进行二进制比较

BINARY可将字符串转换为二进制对象再进行比较

SELECT string1 比较运算符 BINARY string2

或者SELECT BINARY string1 比较运算符 string2

逻辑查询

在SQL中逻辑查询主要包括AND OR NOT三种

AND

SELECT stuID,stuName,age,sex

FROM T\_student

WHERE age>20 AND sex=’男’

SQL语句中AND运算符两侧的条件必须都为TRUE，否则将显示FALSE

使用OR运算符查询满足任一条件的记录

SELECT teaID,teaName,age,sex,dept,profession

FROM T\_teacher

WHERE age>’45’ OR profession=’教授’

使用NOT运算符查询满足相反条件的记录

在 MS SQL SERVER中，使用 ~ 代替NOT

空值查询

当使用NULL查询数据表中的记录是否为空时，不能使用比较运算符，只能是用IS NULL或者使用IS NOT NULL，如果使用运算符对NULL进行比较其查询条件返回的结果始终是FALSE，也就是说使用比较运算符对NULL值进行比较不会检索出任何结果

使用LIKE操作符实现模糊查询

匹配任意单个字符

在SQL中，通配符“\_”表示单个字符，一个“\_”只能表示一个字符

SELECT stuID,stuName,age,sex,birth

FROM T\_student

WHERE stuID LIKE ‘s\_\_\_\_\_3’

(WHERE stuID NOT LIKE ‘s\_\_\_\_\_3’)

MYSQL数据库中的LIKE操作符进行匹配时，对字母的大小写不敏感

匹配0个或者多个字符

通配符“%”表示0个或者多个字符

SELECT stuID,stuName,age,sex,birth

FROM T\_student

WHERE stuName LIKE ‘王%’

LIKE关键字后面的匹配符必须使用单引号，对字母进行匹配时，还要注意区分大小写

使用转义字符

如果在查询时数据表中的某个字段中的字符值本身就包含了“%”和“\_”，这时就要用到转义字符，用关键字ESCAPE来转义，步骤：

1. 在需要转义的“%” “\_”字符前加一个转义符，该转义符可以是任意字符
2. 在ESCAPE关键字后指定该转义字符的名称

SELECT deptID,deptName

FROM T\_dept

WHERE deptID LIKE ‘%$\_%’ ESCAPE ‘$’

使用REGEXP关键字进行模式匹配

1. ^ 用来匹配字符串的开始
2. $ 用来匹配字符串的结尾
3. [ ] 在方括号中的任意字符都可以匹配
4. + 表示用于匹配的该字符在匹配的字符串中出现至少一次或多次
5. \* 表示用于匹配的该字符在匹配的字符串中出现零次或者多次
6. () 括号中的内容将被看成是一个整体
7. { m} 整数m表示花括号中的字符串需要出现的次数

表中数据的排序与分组

使用GROUP BY 子句对数据记录进行排序

指定表中的一列进行排序

使用ORDER BY可对查询的结果进行排序，ASC表升序，DESC表降序

ORDER BY column1 [ASC|DESC]

SELECT teaID,teaName,dept,profession,salary

FROM T\_teacher

ORDER BY salary ASC

如果指定要排序的列中存在空值，则在升序排列中显示在最前，降序排列中显示在最后

ORDER BY也可以放在WHERE子句后面

SELECT teaID,teaName,dept,profession,salary

FROM T\_teacher

WHERE dept=’计算机系’

ORDER BY salary ASC

指定表中列的位置序号进行排序

SELECT teaID,teaName,dept,profession,salary

FROM T\_teacher

ORDER BY 5 ASC

salary的位置是5，可以用5来代替

对SELECT语句中的非选择列进行排序

使用ORDER BY子句中也可以对没有在SELECT中出现的选择列进行排序操作

SELECT teaID,teaName,dept,profession

FROM T\_teacher

WHERE dept=’计算机系’

ORDER BY salary ASC

不过一般的情况下都会使用在SELECT中出现的列进行排序，这样显得很直观

指定数据表中的多列进行排序

ORDER BY可以指定数据表中的多列进行排序操作，这时指定排序的列之间使用分号隔开

ORDER BY column1,column2 [ASC|DESC]

SLEECT teaID,teaName,dept,profession,salary

FROM T\_teacher

ORDER BY dept,salary

使用ORDER BY 进行多列排序时，遵循以下规则：

1. 根据ORDER BY 指定的第一列，按照指定的升序或者降序进行排序
2. 当ORDER BY 子句中指定的第一列中出现相同的数据时，再根据ORDER BY指定的第二列的升序或者降序进行排列
3. 当ORDER BY 子句中指定的第二列中出现相同的数据时，再根据ORDER BY指定的第三列的升序或者降序进行排列，依次类推

常用的聚合函数

|  |  |
| --- | --- |
| 函数 | 功能 |
| COUNT（列|表达式） | 计算给定列或者表达式中的非空行数 |
| COUNT(\*) | 计算数据表中的总行数，包括空值 |
| MAX(列|表达式) | 计算给定列或者表达式中的最大值 |
| MIN() |  |
| SUM(列|表达式) |  |
| AVG() |  |

使用MYSQL数据库限制结果集行数

LIMIT n LIMIT offset,n

连接查询与集合查询

内连接查询

等值连接

SELECT table\_name1.column,table\_name2.column…

FROM table\_name1,table\_name2

WHERE table\_name1.column=table\_name2.column

使用连接查询时可以使用可以采用列别名的方式简化书写

SELECT A.column,B.column

FROM table\_name1 A,table\_name2 B

WHERE A.column=B.column;

非等值连接，是指在连接查询时使用>= <= < > != <> !< !> BETWEEN AND

使用ON子句建立相等连接

在SQL语句中除了使用=建立等值连接外，也可以使用ON子句建立等值连接

SELECT 表名1.字段，表名2.字段

FROM 表名1，表名2

ON 表名1.字段=表名2.字段

交叉连接

交叉连接的结果是一个笛卡尔积

SELECT 表名1.字段，表名2.字段

FROM 表名 CROSS JOIN 表名

自连接查询

自连接查询用在同一表中的连接查询操作

SELECT A.字段，B.字段

FROM 表名1 A，表名2 B

WHERE A.字段=B.字段

由于自连接使用的是同一张表，所以在FROM中需要为表定义不同的列别名

外连接查询

外连接查询可以查询到不满足连接条件的部分记录

左外连接

SELECT 表名1.字段，表名2.字段

FROM 表名1 LEFT JOIN 表名2

ON 表名1.字段=表名2.字段

右外连接 RIGHT JOIN

全外连接 FULL JOIN

集合查询 UNION INTERSECT MINUS

SELECT 语句1

UNION （INTERSECT OR MINUS）

SELECT 语句2

在进行并操作时两个SELECT语句中要查询的列对应的属性的个数和数据类型必须是相同的，否则系统将报错，在MYSQL中不支持INTERSECT 和MINUS