[SQL 基础教程 3](#_Toc23195)

[SELECT，选择 3](#_Toc26712)

[DISTINCT，去重 3](#_Toc16153)

[WHERE，限制条件 3](#_Toc24283)

[ORDER BY，排序 3](#_Toc26544)

[INSERT INTO，插入数据 3](#_Toc22610)

[Update，更新数据 3](#_Toc23223)

[DELETE，删除 3](#_Toc13721)

[SQL 高级教程 4](#_Toc8693)

[规定返回记录数目 4](#_Toc15432)

[LIKE，指定搜索列模式 4](#_Toc3016)

[通配符 4](#_Toc19405)

[IN，规定多个值 4](#_Toc8806)

[BETWEEN，规定值区间 4](#_Toc13332)

[Alias，别名 4](#_Toc29261)

[join，连接两个表 5](#_Toc710)

[join类型 5](#_Toc16674)

[UNION/UNION ALL，合并多个结果集 5](#_Toc804)

[SELECT INTO，创建表的备份复件 5](#_Toc8332)

[CREATE DATABASE，创建数据库 5](#_Toc19970)

[CREATE TABLE，创建表 5](#_Toc25797)

[约束 (Constraints)，限制加入表的数据的类型 6](#_Toc29308)

[NOT NULL，不接受 NULL 值 6](#_Toc1285)

[UNIQUE，唯一标识 6](#_Toc17742)

[PRIMARY KEY，主键 6](#_Toc23538)

[FOREIGN KEY，外键 6](#_Toc5117)

[CHECK，限制列中的值的范围 7](#_Toc12457)

[DEFAULT，向列中插入默认值 7](#_Toc8751)

[CREATE INDEX，创建索引 7](#_Toc29999)

[DROP，删除索引、表和数据库 8](#_Toc19175)

[ALTER TABLE，增删改已有表的列 8](#_Toc19783)

[Auto-increment，插入时生成唯一数字 8](#_Toc11393)

[VIEW，视图 8](#_Toc18678)

[Date 9](#_Toc15773)

[MySQL Date 函数 9](#_Toc22599)

[MySQL Date 数据类型 9](#_Toc23134)

[SQL 日期处理 9](#_Toc27144)

[NULL，遗漏的未知数据 10](#_Toc5288)

[NULL值替换函数 10](#_Toc30106)

[数据类型 10](#_Toc10049)

[SQL 函数 10](#_Toc28677)

[AVG，返回平均值 10](#_Toc11007)

[COUNT，返回统计行数 10](#_Toc28401)

[FIRST/LAST，返回第一个/最后一个 记录的值 10](#_Toc26219)

[MAX/MIN，返回最大值/最小值 11](#_Toc4154)

[SUM，返回总数 11](#_Toc4997)

[GROUP BY，分组结果集 11](#_Toc13101)

[HAVING，过滤聚合函数 11](#_Toc5254)

[UCASE/LCASE，转大写/小写 11](#_Toc14522)

[MID，提取字符 11](#_Toc6100)

[LEN，返回文本字段长度 11](#_Toc27159)

[ROUND，舍入为指定小数位数 11](#_Toc27785)

[NOW，返回当前日期和时间 12](#_Toc25008)

[FORMAT，格式化字段 12](#_Toc5177)

# SQL 基础教程

## SELECT，选择

SELECT colName

SELECT \*

SQL 语句对大小写不敏感；字符串值比对如需判断大小写需配置数据库。

## DISTINCT，去重

SELECT DISTINCT col1，col2 FROM tab1; 应用于多列

SELECT DISTINCT (col1),col2 FROM tab1; 应用于单列

## WHERE，限制条件

运算符：=,<>,>,<,>=,<=,

between,not between,like,not like,in,not in

is null,is not null

and,or,

## ORDER BY，排序

ORDER BY colName

ASC，升序，默认；DESC，降序

## INSERT INTO，插入数据

INSERT INTO tab VALUES (值1, 值2,....)

INSERT INTO tab (列1, 列2,...) VALUES (值1, 值2,....),(...) //逗号隔开插入值对象

## Update，更新数据

UPDATE tab SET col1=v1,col2=v2 WHERE co3=v3; //逗号隔开更新列

## DELETE，删除

DELETE FROM 表名称 WHERE 列名称 = 值;

DELETE \* FROM 表名称 WHERE 列名称 = 值;

只删除表中所有的行；不删除表，表的结构、属性、索引都是完整的。

# SQL 高级教程

## 规定返回记录数目

MySQL：LIMIT，LIMIT number,page

Oracle：ROWNUM，WHERE ROWNUM <= number AND ROWNUM >= number

ROWNUM为条目ID，可用于ORACLE数据库查询同期数据时的表自连接。

## LIKE，指定搜索列模式

LIKE "\_[ab]%[!cd]"

NOT LIKE "heculus"

### 通配符

%，替代一个或多个字符

\_，仅替代一个字符

[charlist]，字符列中的任何单一字符，如[abc]

[^charlist]/[!charlist]，不在字符列中的任何单一字符，如[^abc][!abc]

## IN，规定多个值

WHERE column\_name IN (value1,value2,...)

WHERE column\_name NOT IN (value1,value2,...)

## BETWEEN，规定值区间

选取介于两个值之间的数据范围。这些值可以是数值、文本或者日期。

WHERE column\_name BETWEEN value1 AND value2

WHERE column\_name NOT BETWEEN value1 AND value2

不同的数据库对 BETWEEN...AND 操作符的处理方式有差异。

## Alias，别名

SELECT name NAME from userInfo as user;

通过as设置表、列别名；as可省。

## join，连接两个表

tabA a JOIN tabB b ON a.id=b.id

### join类型

INNER JOIN，与 JOIN 是相同的

JOIN: 如果表中有至少一个匹配，则返回行

LEFT JOIN: 即使右表中没有匹配，也从左表返回所有的行

RIGHT JOIN: 即使左表中没有匹配，也从右表返回所有的行

FULL JOIN: 只要其中一个表中存在匹配，就返回行

## UNION/UNION ALL，合并多个结果集

SELECT column\_name(s) FROM table\_name1

UNION

SELECT column\_name(s) FROM table\_name2

UNION 内部的 SELECT 语句必须拥有相同数量的列。列也必须拥有相似的数据类型。同时，列的顺序必须相同。

UNION 操作符选取不同的值。如果允许重复的值，使用 UNION ALL。

UNION 结果集中的列名总是等于 UNION 中第一个 SELECT 语句中的列名。

## SELECT INTO，创建表的备份复件

SELECT \*

INTO new\_table\_name [IN externaldatabase]

FROM old\_tablename

## CREATE DATABASE，创建数据库

CREATE DATABASE database\_name

## CREATE TABLE，创建表

CREATE TABLE 表名称

(

列名称1 数据类型,

列名称2 数据类型,

......

)

## 约束 (Constraints)，限制加入表的数据的类型

### NOT NULL，不接受 NULL 值

### UNIQUE，唯一标识

表创建时创建 UNIQUE 约束：

MySQL：UNIQUE (Id\_P)

Oracle：Id\_P int NOT NULL UNIQUE

命名约束、多列约束：CONSTRAINT uc\_PersonID UNIQUE (Id\_P,LastName)

表创建后添加UNIQUE 约束：

ALTER TABLE Persons ADD UNIQUE (Id\_P)

ALTER TABLE Persons ADD CONSTRAINT uc\_PersonID UNIQUE (Id\_P,LastName)

撤销 UNIQUE 约束

MySQL：ALTER TABLE Persons DROP INDEX uc\_PersonID

Oracle：ALTER TABLE Persons DROP CONSTRAINT uc\_PersonID

### PRIMARY KEY，主键

表创建时创建 PRIMARY KEY 约束：

MySQL：PRIMARY KEY (Id\_P)

Oracle ：Id\_P int NOT NULL PRIMARY KEY,

命名约束，多列约束：CONSTRAINT pk\_PersonID PRIMARY KEY (Id\_P,LastName)

表创建后添加 PRIMARY KEY 约束：

ALTER TABLE Persons ADD PRIMARY KEY (Id\_P)

ALTER TABLE Persons ADD CONSTRAINT pk\_PersonID PRIMARY KEY (Id\_P,LastName)

如使用 ALTER TABLE 语句添加主键，必须在表首次创建时把主键列声明为不包含 NULL 值。

撤销 PRIMARY KEY 约束：

MySQL：ALTER TABLE Persons DROP PRIMARY KEY

Oracle：ALTER TABLE Persons DROP CONSTRAINT pk\_PersonID

### FOREIGN KEY，外键

用于预防破坏表之间连接的动作，防止非法数据插入外键列。

表创建时创建 FOREIGN KEY 约束：

MySQL：FOREIGN KEY (Id\_P) REFERENCES Persons(Id\_P)

Oracle ：Id\_P int FOREIGN KEY REFERENCES Persons(Id\_P)

命名约束，多列约束：CONSTRAINT fk\_PerOrders FOREIGN KEY (Id\_P) REFERENCES Persons(Id\_P)

表创建后添加 FOREIGN KEY 约束：

ALTER TABLE Orders ADD FOREIGN KEY (Id\_P) REFERENCES Persons(Id\_P)

ALTER TABLE Orders ADD CONSTRAINT fk\_PerOrders FOREIGN KEY (Id\_P) REFERENCES Persons(Id\_P)

撤销 FOREIGN KEY 约束：

MySQL：ALTER TABLE Orders DROP FOREIGN KEY fk\_PerOrders

Oracle：ALTER TABLE Orders DROP CONSTRAINT fk\_PerOrders

### CHECK，限制列中的值的范围

对单个列定义 CHECK 约束，该列只允许特定的值。

对一个表定义 CHECK 约束，中特定列的值进行限制。

表创建时创建 CHECK KEY 约束：

MySQL：CHECK (Id\_P>0)

Oracle：Id\_P int NOT NULL CHECK (Id\_P>0)

命名约束，多列约束：CONSTRAINT chk\_Person CHECK (Id\_P>0 AND City='Sandnes')

表创建后添加 CHECK KEY 约束：

ALTER TABLE Persons ADD CHECK (Id\_P>0)

ALTER TABLE Persons ADD CONSTRAINT chk\_Person CHECK (Id\_P>0 AND City='Sandnes')

撤销 CHECK KEY 约束：

MySQL：ALTER TABLE Persons DROP CHECK chk\_Person

Oracle：ALTER TABLE Persons DROP CONSTRAINT chk\_Person

### DEFAULT，向列中插入默认值

表创建时创建 DEFAULT KEY 约束：

City varchar(255) DEFAULT 'Sandnes'

OrderDate date DEFAULT GETDATE() //插入函数值作为默认值

表创建后添加 DEFAULT KEY 约束：

MySQL：ALTER TABLE Persons ALTER City SET DEFAULT 'SANDNES'

Oracle：ALTER TABLE Persons ALTER COLUMN City SET DEFAULT 'SANDNES'

撤销 DEFAULT KEY 约束：

MySQL：ALTER TABLE Persons ALTER City DROP DEFAULT

Oracle：ALTER TABLE Persons ALTER COLUMN City DROP DEFAULT

## CREATE INDEX，创建索引

用户无法看到索引，它们只能被用来加速搜索/查询。更新表时会更新索引。

创建索引：

CREATE INDEX index\_name ON table\_name (column\_name) //索引列的值可重复

CREATE UNIQUE INDEX index\_name ON table\_name (column\_name) //值不可重复

CREATE INDEX PersonIndex ON Person (LastName DESC) //降序

CREATE INDEX PersonIndex ON Person (LastName, FirstName) //多列索引

## DROP，删除索引、表和数据库

删除索引：

MySQL：ALTER TABLE table\_name DROP INDEX index\_name

Oracle：DROP INDEX index\_name

删除表：

DROP TABLE 表名称

TRUNCATE TABLE 表名称 //只删除数据，不删除表

删除数据库：

DROP DATABASE 数据库名称

## ALTER TABLE，增删改已有表的列

添加列：

ALTER TABLE table\_name ADD column\_name datatype

删除列：

ALTER TABLE table\_name DROP COLUMN column\_name

更改列数据类型：

ALTER TABLE Persons ALTER COLUMN Birthday year

## Auto-increment，插入时生成唯一数字

MySQL：

P\_Id int NOT NULL AUTO\_INCREMENT

Oracle：

创建sequence：

CREATE SEQUENCE seq\_person

MINVALUE 1

START WITH 1

INCREMENT BY 1

CACHE 10

通过sequence.nextval获取值插入：

INSERT INTO Persons (P\_Id,FirstName,LastName)

VALUES (seq\_person.nextval,'Lars','Monsen')

## VIEW，视图

视图是可视化的表。数据库的设计和结构不受视图影响。查询视图时，数据库重新查询表来重建数据。

CREATE VIEW，创建视图：

CREATE VIEW view\_name AS

SELECT column\_name(s)

FROM table\_name

WHERE condition

CREATE OR REPLACE VIEW，更新视图列：

CREATE OR REPLACE VIEW view\_name AS

SELECT column\_name(s)

FROM table\_name

WHERE condition

DROP VIEW，删除视图：

DROP VIEW view\_name

## Date

### MySQL Date 函数

函数 描述

NOW() 返回当前的日期和时间

CURDATE() 返回当前的日期

CURTIME() 返回当前的时间

DATE() 提取日期或日期/时间表达式的日期部分

EXTRACT() 返回日期/时间按的单独部分

DATE\_ADD() 给日期添加指定的时间间隔

DATE\_SUB() 从日期减去指定的时间间隔

DATEDIFF() 返回两个日期之间的天数

DATE\_FORMAT() 用不同的格式显示日期/时间

### MySQL Date 数据类型

DATE - 格式 YYYY-MM-DD

DATETIME - 格式: YYYY-MM-DD HH:MM:SS

TIMESTAMP - 格式: YYYY-MM-DD HH:MM:SS

YEAR - 格式 YYYY 或 YY

### SQL 日期处理

只有日期部分时可直接比较：

SELECT \* FROM Orders WHERE OrderDate='2008-12-26'

带时间部分，需转换后比较：

SELECT \* FROM Orders WHERE DATE(OrderDate)='2008-12-26'

## NULL，遗漏的未知数据

IS NULL、IS NOT NULL，筛选空值、非空值

### NULL值替换函数

Oracle：

SELECT ProductName,NVL(Price,0) FROM Products

MySQL：

SELECT ProductName,IFNULL(Price,0) FROM Products

SELECT ProductName,COALESCE(Price,0) FROM Products

## 数据类型

<http://www.w3school.com.cn/sql/sql_datatypes.asp>

# SQL 函数

SELECT function(列) FROM 表

## AVG，返回平均值

SELECT AVG(column\_name) FROM table\_name

## COUNT，返回统计行数

SELECT COUNT(column\_name) FROM table\_name //NULL值不计入

SELECT COUNT(\*) FROM table\_name

SELECT COUNT(DISTINCT column\_name) FROM table\_name //统计不同值

## FIRST/LAST，返回第一个/最后一个 记录的值

可使用 ORDER BY 语句对记录进行排序。

SELECT FIRST(column\_name) FROM table\_name

## MAX/MIN，返回最大值/最小值

MIN 和 MAX 也可用于文本列，以获得按字母顺序排列的最高或最低值。

SELECT MAX(column\_name) FROM table\_name

## SUM，返回总数

SELECT SUM(column\_name) FROM table\_name

## GROUP BY，分组结果集

使用GROUP BY的SELECT语句，列必须在GROUP BY子句、聚合函数中。

GROUP BY Customer,OrderDate

## HAVING，过滤聚合函数

HAVING avg(price) >100

## UCASE/LCASE，转大写/小写

SELECT UCASE(column\_name) FROM table\_name

## MID，提取字符

SELECT MID(column\_name,start[,length]) FROM table\_name

column\_name 必需。要提取字符的字段。

start 必需。规定开始位置（起始值是 1）。

length 可选。要返回的字符数。如果省略，则 MID() 函数返回剩余文本。

## LEN，返回文本字段长度

SELECT LEN(column\_name) FROM table\_name

## ROUND，舍入为指定小数位数

SELECT ROUND(column\_name,decimals) FROM table\_name

## NOW，返回当前日期和时间

SELECT NOW() FROM table\_name

## FORMAT，格式化字段

SELECT FORMAT(column\_name,format) FROM table\_name

column\_name 必需。要格式化的字段。

format 必需。规定格式。

实例：

SELECT ProductName, UnitPrice, FORMAT(Now(),'YYYY-MM-DD') as PerDate

FROM Products