

# SQL基础

---

## 1、SQL 语言分类

---

-- DQL 数据查询语言 select -- DML 数据操作语言 insert update delete -- DDL 数据定义语言 create alter drop -- DCL 数据控制语言 grant revoke -- TCL 事务控制语言 savepoint、rollback、set transaction、commit

## 2、SQL 书写顺序

---

-- select -> distinct -> from -> join -> on -> where -> group by -> having -> limit

## 3、SQL 执行顺序

---

-- from -> on -> join -> where -> group by -> having -> select -> distinct

## 4、常用函数

---

```
-- 空值转换
select nvl(a.age) from a ; nvl函数是Oracle中的

-- case when 和 decode
select decode(a.age , 10 , '年轻',20,'不年轻') from a ; MySQL 中不存在decode函数?

-- group by 、 group by cube 、 group by rollup 、 grouping 、 group by with rollup

-- 类型转换函数 cast

-- 字符串函数 replace 、 substring 、 len 、 concat

-- 日期函数
-- DATE_SUB 时间减掉
select DATE_SUB('2021-01-20', INTERVAL 1 MONTH) ;

select DATE_SUB('2021-01-20', INTERVAL -1 MONTH) ;

select DATE_SUB('2021-01-20', INTERVAL -1 year) ;

-- DATE_ADD 时间加上来
select DATE_ADD('2021-01-20', INTERVAL 1 MONTH) ;

select DATE_ADD('2021-01-20', INTERVAL -1 MONTH) ;

select DATE_ADD('2021-01-20', INTERVAL -1 year) ;

-- 数学函数 round 两个参数，意思是取小数第几位，一个参数，取整数；
select round(5.123);
select round(5.123 , 1) ;
```

```
select round(5.123 , 2) ;
```

-- 行转列 case when 、 collect\_list 列聚合成数组，不去重、 collect\_set 列聚合成数据，去重、 concat\_ws 拼接字符串

-- 列转行 union all

## 5、窗口函数 MySQL8.0 以后才支持窗口函数，窗口函数不会改变总记录数

```
select * from student ;
select
    card_no as '学号' ,
    class_no as '班级' ,
    grade as '成绩' ,
    rank() over(partition by class_no order by grade desc ) as '排名'
from student ;
```

## 6、SQL 执行计划

```
EXPLAIN
select * from student
union all
select * from student;
```

## 7、load命令使用：

```
-- 启动MySQL 客户端时指定
mysql --local-infile=1 -uroot -proot
-- 执行数据插入
LOAD DATA LOCAL INFILE 'F:\\HJKJ\\奉贤分局数据中台\\cs.txt' INTO TABLE cs fields
TERMINATED BY '---' ;
```

## 8、sql优化

```
explain select * from student
union all select * from student ;
```

type 连接类型或者访问类型，性能从好到差依次为：

system，表中只有一行数据，这是 const 类型的特殊情况；

const，最多返回一条匹配的数据，在查询的最开始读取；

eq\_ref，对于前面的每一行，从该表中读取一行数据；

ref，对于前面的每一行，从该表中读取匹配索引值的所有数据行；

fulltext，通过 FULLTEXT 索引查找数据；

ref\_or\_null，与 ref 类似，额外加上 NULL 值查找；

index\_merge，使用索引合并优化技术，此时 key 列显示使用的所有索引；

unique\_subquery，替代以下情况时的 eq\_ref：value IN (SELECT primary\_key FROM single\_table WHERE some\_expr)；

index\_subquery，与 unique\_subquery 类似，用于子查询中的非唯一索引：value IN (SELECT key\_column FROM single\_table WHERE some\_expr)；

range，使用索引查找范围值；

index，与 ALL 类型相同，只不过扫描的是索引；

ALL，全表扫描，通常表示存在性能问题。

## 8.1、最大化利用索引

## 8.2、尽可能避免全表扫描

尽量避免在字段开头模糊查询，会导致数据库引擎放弃索引而全表扫描；

尽量避免使用 in 、 not in ，会导致引擎走全表扫描，用 exists 替换 in ；

尽量避免使用 or ，会导致数据库引擎放弃索引进行全表扫描，优化手段用 union all 替换；

尽量避免进行 null 值得判断，会导致数据库引擎放弃索引而全表扫描；

尽量避免在 where 条件中等号左侧进行表达式、函数操作，会导致数据库引擎放弃索引而全表扫描；

当数据量较大时，避免使用 where 1=1

查询条件不能使用 <> 或者 != 或者 not in

where 条件仅包含复合索引非前置列

隐式类型转换造成不适用索引

order by 条件要与 where 条件一致，否则 order by 不会利用索引进行排序

## 8.3、减少无效数据的查询

sql 其他优化

## SELECT语句其他优化

### 1. 保证不查询多余的列与行。

尽量避免select \* 的存在，使用具体的列代替\*，避免多余的列

使用where限定具体要查询的数据，避免多余的行

使用top, distinct关键字减少多余重复的行

### 2. 避免出现不确定结果的函数

### 3. 多表关联查询时，小表在前，大表在后(oracle相反)

### 4. 使用表的别名

### 5. 用where字句替换HAVING字句

### 6. 调整Where字句中的连接顺序

## sql优化--索引增加标准



什么样的列必须建立索引呢？

where条件中的列，group by 的列，order by 的列。

基数：

某个列唯一键的数量叫做基数。比如性别列，该列只有男女之分，所以这一列基数为2。主键列的基数等于表的总行数。基数的高低影响列的数据分布。

当查询结果返回表中超过5%的数据使用全表扫描，低于5%走索引

如果某个列基数很低，该列数据分布就会非常不均衡，由于该列数据分布不均衡，会导致sql查询可能走索引、也可能走全表扫描。在做sql优化的时候，如果怀疑列数据分布不均衡，可以使用select 列,count(\*) from group by 列 order by 2 desc 来查看列的数据分布。

选择性：

基数与总行数的比值再乘以100%就是某个列的选择性。

在进行sql优化的时候。单独看列的基数是没有意义的，基数必须对比总行数才有实际意义。比如某个列的基数有几万行，但是总行数有几十亿行，那么这个列的基数还高吗？

当一个列出现在where条件中，该列没有创建索引并且选择性大于20%，那么该列就必须创建索引，从而提高sql查询性能。

## 9、MySQL正则表达式

regex()

-- 详见: <https://www.cnblogs.com/ccstu/p/12182324.html>

选项	说明	例子	匹配值示例
^	匹配文本的开始字符	'^b' 匹配以字母 b 开头的字符串	book、big、banana、bike
\$	匹配文本的结束字符	'st\$' 匹配以 st 结尾的字符串	test、resist、persist
.	匹配任何单个字符	'b.t' 匹配任何 b 和 t 之间有一个字符	bit、bat、but、bite
*	匹配零个或多个在它前面的字符	'f*n' 匹配字符 n 前面有任意个字符 f	fn、fan、faan、abcn
+	匹配前面的字符 1 次或多次	'ba+' 匹配以 b 开头，后面至少紧跟一个 a	ba、bay、bare、battle
<字符串>	匹配包含指定字符的文本	'fa'	fan、afa、faad
[字符集合]	匹配字符集合中的任何一个字符	'[xz]' 匹配 x 或者 z	dizzy、zebra、x-ray、extra
[^]	匹配不在括号中的任何字符	'[^abc]' 匹配任何不包含 a、b 或 c 的字符串	desk、fox、f8ke
字符串{n,}	匹配前面的字符串至少 n 次	b{2} 匹配 2 个或更多的 b	bbb、bbbb、bbbbbbb
字符串{n,m}	匹配前面的字符串至少 n 次，至多 m 次	b{2,4} 匹配最少 2 个，最多 4 个 b	bbb、bbbb