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

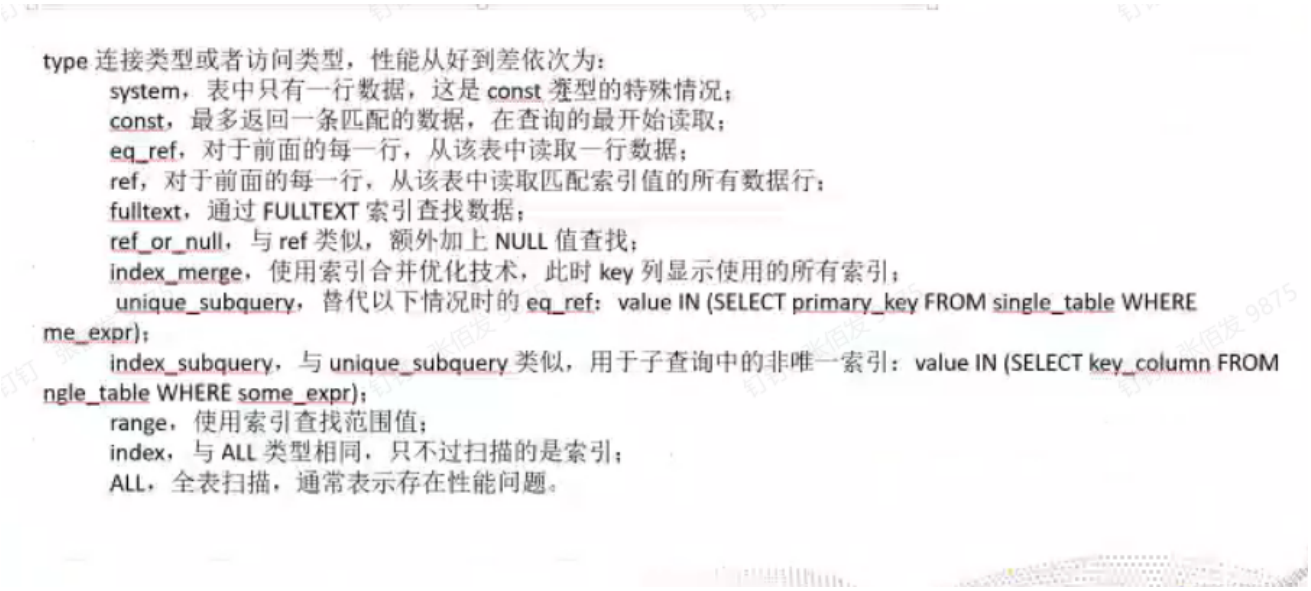
```
EXPLAIN
select * from student
union all
select * from student;
```

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 ;
```



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进而全表扫描；

尽量避免使用in 、 not in ，回导致引擎走全表扫描，用exits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进而全表扫描；

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进而全表扫描；

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

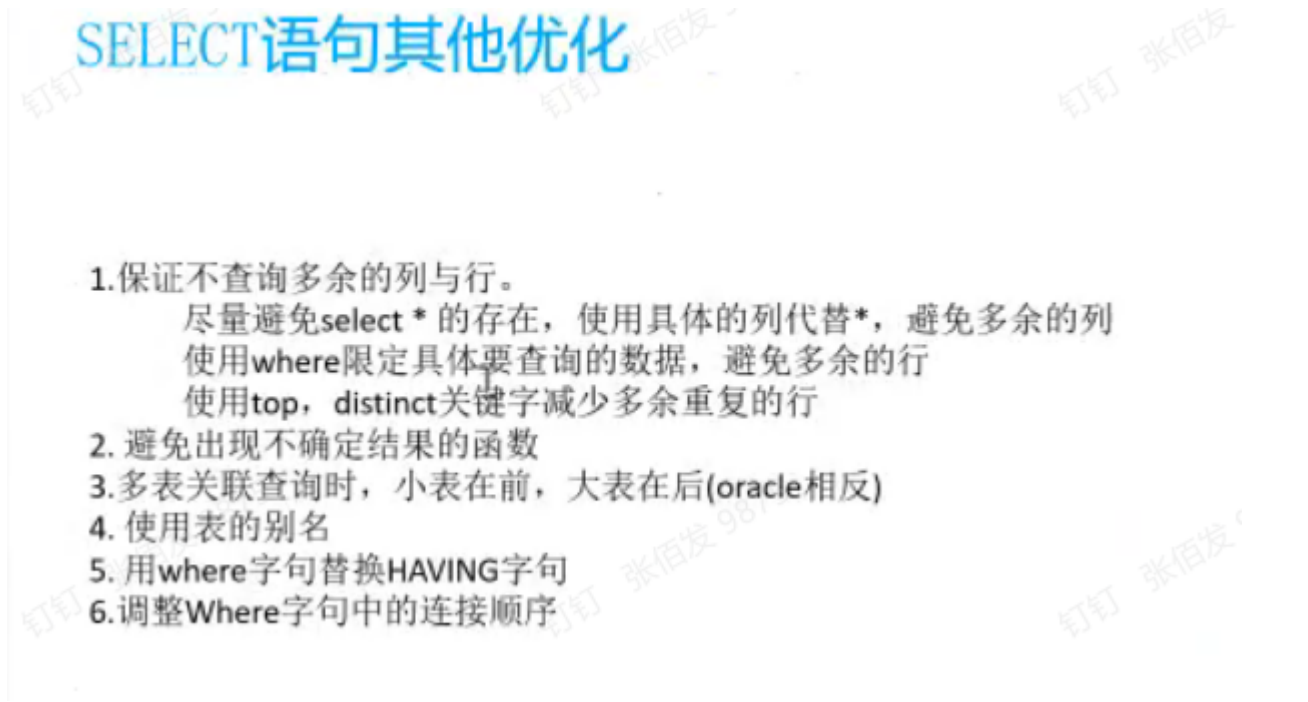
where条件仅包含复合索引非前置列

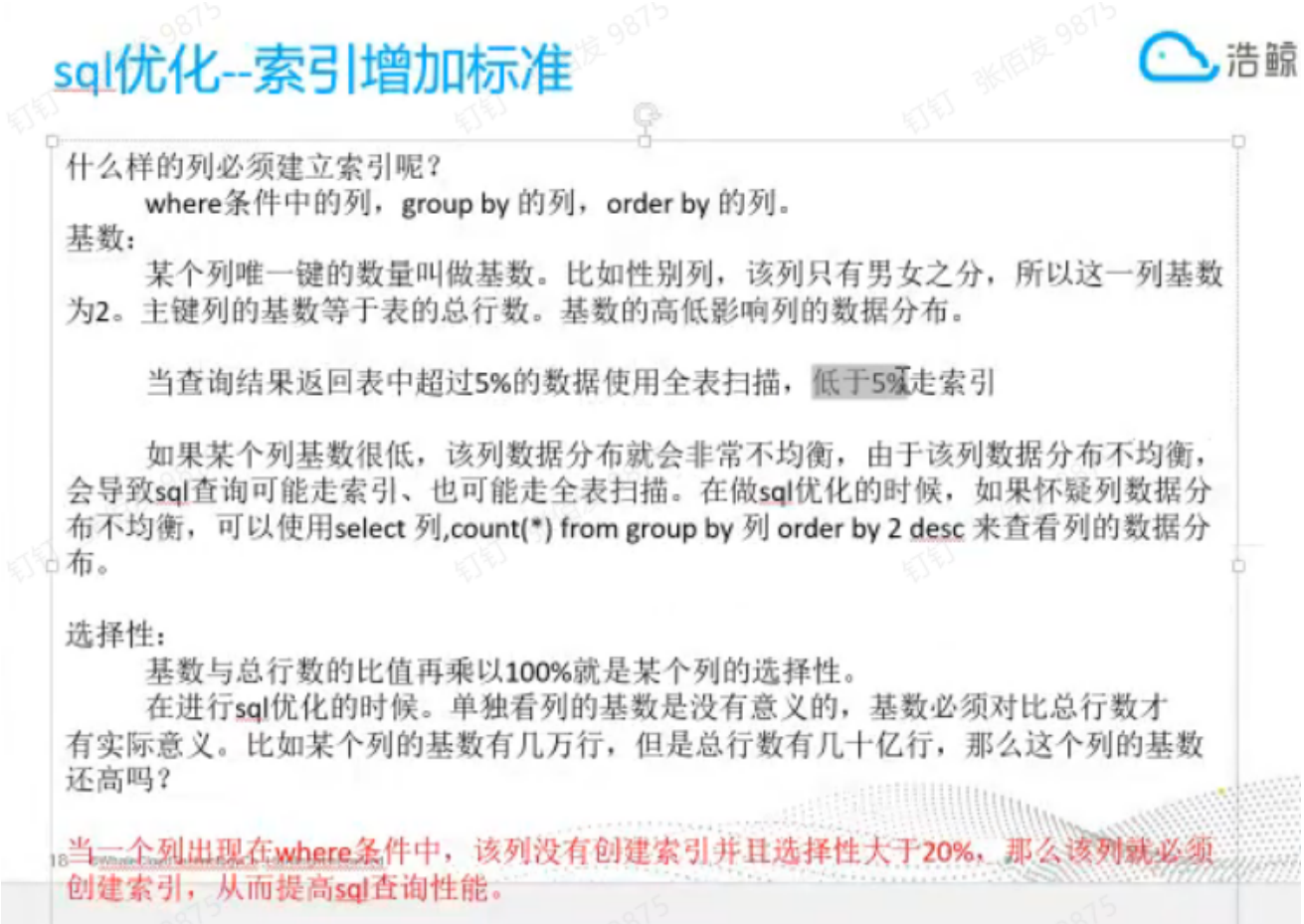
隐式类型转换造成不适用索引

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





9、MySQL正则表达式

```
regexp()
-- 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