# sql基本语句

## insert、update

<**insert** id="insertTBOX" parameterType="com.maxus.tsp.platform.service.model.car.Tbox"

useGeneratedKeys="true" keyProperty="id" >

INSERT INTO tbox (sn, iccid, simno, uuid, status, createDate)

VALUES (#{sn}, #{iccid}, #{simno}, #{uuid}, #{status}, #{createDate})

</insert>

<**update** id="updateTBOX" parameterType="com.maxus.tsp.platform.service.model.car.Tbox">

UPDATE tbox

<set>

<if test=" sn!=null and sn!=''">

sn = #{ sn },

</if>

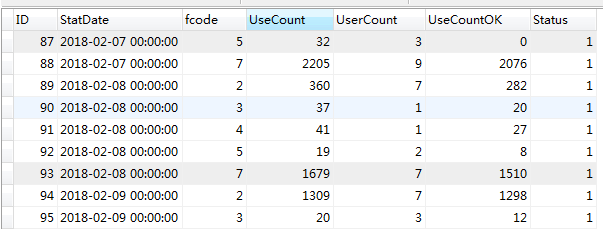
updateDate = NOW()

</set>

WHERE ID = #{id}

</update>

## select各种查询



## 统计 sum、case when

按时间统计每种操作(fcode)的当日使用次数(UseCount)、使用人数(UserCount)、成功次数(UseCountOK)

<select id="getTspUseListExport" resultType="HashMap">

SELECT statDate,DATE\_FORMAT(aa.StatDate,'%Y-%m-%d') AS date,

sum(CASE WHEN aa.fcode='1' THEN aa.UseCount ELSE 0 END ) as useCount\_log,

sum(CASE WHEN aa.fcode='1' THEN aa.UserCount ELSE 0 END ) as userCount\_log,

sum(CASE WHEN aa.fcode='1' THEN aa.UseCountOK ELSE 0 END ) as useCountOK\_log

…

FROM tsp\_use\_stat aa

WHERE 1=1

<if test="startTime != null and startTime != ''">

AND date\_format(aa.statDate,'%Y-%m-%d %h:%M:%s') &gt;= #{startTime}

</if>

<if test="endTime != null and endTime != ''">

AND date\_format(aa.statDate,'%Y-%m-%d %h:%M:%s') &lt;= #{endTime}

</if>

GROUP BY date,statDate

ORDER BY statDate DESC

</select>

## 去重取最大值 联合

每种操作(fcode)的当日使用次数(UseCount)、使用人数(UserCount)的最大值，联合起来成一行

SELECT a.fcode AS fcode, SUM(UseCount) AS UseCount, SUM(UserCount) AS UserCount FROM(

SELECT a.fcode, a.UseCount AS UseCount, 0 AS UserCount

FROM tsp\_use\_stat a

WHERE a.UseCount=(

SELECT MAX(b.UseCount) FROM tsp\_use\_stat b WHERE a.fcode=b.fcode

) GROUP BY a.fcode

UNION ALL

SELECT a.fcode, 0 AS UseCount, a.UserCount AS UserCount

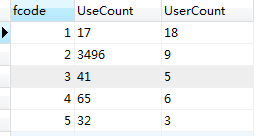
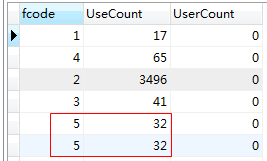
FROM tsp\_use\_stat a

WHERE a.UserCount=(

SELECT MAX(b.UserCount) FROM tsp\_use\_stat b WHERE a.fcode=b.fcode

) GROUP BY a.fcode

) a GROUP BY fcode

如果不加GROUP BY a.fcode，当有两个最大值时，会显示两行，就两个最大值没去重

**distinct**只能返回他的目标字段，而无法返回其他字段,所以我们往往用distinct来返回不重复字段的条数（count(distinct id)），如果要查询不重复的记录，有时候可以用group by。

<if test="startDate != null and startDate != '' and endDate != null and endDate != ''">

AND (DATE(CreatDate) BETWEEN #{startDate} AND #{endDate})

</if>

## 导入csv文件到表语句

load data infile 'D:/LSFD13203HC000963.csv'

into table sunwin\_gps character set gb2312

fields terminated by ',' optionally enclosed by '"' escaped by '"'

lines terminated by '\r\n';

## 视图

视图是虚拟的表，与包含数据的表不一样，视图只包含使用时动态检索数据的查询；不包含任何列或数据

**申沃demo按时间间隔筛选创建一个表gps的视图，在查询的时候减少数据量**

CREATE

ALGORITHM = UNDEFINED

DEFINER = `root`@`localhost`

SQL SECURITY DEFINER

VIEW `maxus\_sunwin`.`v\_sunwin\_gps` AS

SELECT

`a`.`ID` AS `ID`,

`a`.`vin` AS `vin`,

`a`.`licence` AS `licence`,

`a`.`name` AS `name`,

`a`.`CreateDate` AS `CreateDate`,

`a`.`Longitude` AS `Longitude`,

`a`.`Latitude` AS `Latitude`

FROM

`maxus\_sunwin`.`sunwin\_gps` `a`

GROUP BY `a`.`vin` , LEFT(DATE\_FORMAT(`a`.`CreateDate`, '%Y-%m-%d %H:%i:%s'),

16) yyyy-mm-dd hh:mm去掉秒group by

LEFT(str,len) 返回字符串str的最左面len个字符。

## sql

去重取最大值



**oracle方法**：

 在oracle中，我们可以简单地用over(partition by)函数处理（代码示例如下），解释一下就是利用partition by函数按照合同号分区，在每个分区内再利用order by根据回款金额排序，where num=1的意思就是取每个分区中排序第一的那一行

select

row\_number()over(partition by 合同号 order by 回款金额 desc) as num

from test where num=1

**mysql方法：**

 但是在mysql中并没有这样的函数，我们需要重新写逻辑（代码如下），老规矩解释一下：第二个select函数是关键所在，功能就是，当a.`合同号` = b.`合同号`时，我们取出最大的回款金额，当a.`回款金额`等于这个最大的回款金额时，我们认为这一行符合条件并取出，因此a中的每个合同号我们都要执行一次第二个select函数，来判断当前合同号的这一行是否是最大回款金额的这一行。

SELECT a.`合同号`, a.姓名, a.`回款金额`

FROM test a WHERE a.`回款金额` =

(SELECT MAX(b.`回款金额`) FROM test b WHERE a.`合同号` = b.`合同号`);

## 索引

alter的用法是修改基本表 alter table<表名>add

alter table table\_name add index index\_name (column\_list) ;

alter table table\_name add unique (column\_list) ;

alter table table\_name add primary key (column\_list) ;

CREATE INDEX能够增加普通索引和UNIQUE索引两种, 不能创建PRIMARY KEY

create index index\_name on table\_name (column\_list) ;

create unique index index\_name on table\_name (column\_list) ;

CREATE INDEX idx\_group ON base\_user (state, car\_name) 组合索引

drop index index\_name on table\_name ;

alter table table\_name drop index index\_name ;

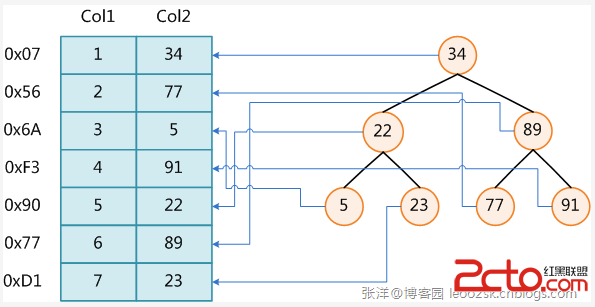
alter table table\_name drop primary key ;

其中，在前面的两条语句中，都删除了table\_name中的索引index\_name。而在最后一条语句中，只在删除PRIMARY KEY索引中使用，因为一个表只可能有一个PRIMARY KEY索引，因此不需要指定索引名。如果没有创建PRIMARY KEY索引，但表具有一个或多个UNIQUE索引，则MySQL将删除第一个UNIQUE索引。

如果从表中删除某列，则索引会受影响。对于多列组合的索引，如果删除其中的某列，则该列也会从索引中删除。如果删除组成索引的所有列，则整个索引将被删除。

数据库索引，是数据库管理系统中一个排序的数据结构，以协助快速查询、更新数据库表中数据。索引的实现通常使用B树及其变种B+树

为表设置索引要付出代价的：一是增加了数据库的存储空间，二是在插入和修改数据时要花费较多的时间(因为索引也要随之变动)。



图展示了一种可能的索引方式。左边是数据表，一共有两列七条记录，最左边的是数据记录的物理地址（注意逻辑上相邻的记录在磁盘上也并不是一定物理相邻的）。为了加快Col2的查找，可以维护一个右边所示的二叉查找树，每个节点分别包含索引键值和一个指向对应数据记录物理地址的指针，这样就可以运用二叉查找在O(log2n)的复杂度内获取到相应数据。虽然这是一个货真价实的索引，但是实际的数据库系统几乎没有使用二叉查找树或其进化品种红黑树（red-black tree）实现的

### Tree

由于B-Tree的特性，在B-Tree中按key检索数据的算法非常直观：首先从根节点进行二分查找，如果找到则返回对应节点的data，否则对相应区间的指针指向的节点递归进行查找，直到找到节点或找到null指针，前者查找成功，后者查找失败

### 唯一索引

### 主键索引

主键索引是唯一索引的特定类型。该索引要求主键中的每个值都唯一。当在查询中使用主键索引时，它还允许对数据的快速访问

### 聚集索引

### 索引优缺点

创建索引可以大大提高系统的性能。

第一，通过创建唯一性索引，可以保证数据库表中每一行数据的唯一性。

第二，可以大大加快数据的检索速度，这也是创建索引的最主要的原因。

第三，可以加速表和表之间的连接，特别是在实现数据的参考完整性方面特别有意义。

第四，在使用分组和排序子句进行数据检索时，同样可以显著减少查询中分组和排序的时间。

第五，通过使用索引，可以在查询的过程中，使用优化隐藏器，提高系统的性能。

也许会有人要问：增加索引有如此多的优点，为什么不对表中的每一个列创建一个索引呢？因为，增加索引也有许多不利的方面。

第一，创建索引和维护索引要耗费时间，这种时间随着数据量的增加而增加。

第二，索引需要占物理空间，除了数据表占数据空间之外，每一个索引还要占一定的物理空间，如果要建立聚簇索引，那么需要的空间就会更大。

第三，当对表中的数据进行增加、删除和修改的时候，索引也要动态的维护，这样就降低了数据的维护速度。