**数据库规范**

# 命名规范

## 表名称

·T\_[模块名]\_[具体业务名]

·以’T’开头，单词之间用’\_’分隔，如：T\_ORDER\_PAYFEE

·遵循达意原则

## 字段名称（不能用关键字。例如：level）

·单词之间用’\_’分隔，如：fee\_sum

·表的id命名:表名去掉t\_以外的部分+**\_id。 例子：**ORDER\_PAYFEE\_ID

**·\_**time 到时分秒 \_date 到天 \_week 到周 \_month 到月 \_year 到年

\_code 字典码 \_code\_name 到字典码名字

\_by 操作人,必须是id \_by\_name 到操作人姓名

## 函数命名

·F\_[表简称]\_[操作名]\_[字段]

如：

1. F\_PayFee\_Get\_Paystatus(),表示获取缴费的支付状态
2. F\_Pay\_Get\_Paystatus(),表示获取结算单的支付状态
3. F\_AG\_Get\_Status()，表示获取合同状态

## 视图命名

·以’V’开头，单词之间用’\_’分隔，如：V\_ORDER\_PAYFEE

## 存储过程命名

·以’P’开头，单词之间用’\_’分隔，如：P\_Balance执行对账

## 触发器

·以’TG’开头，单词之间用’\_’分隔，如：TG\_ORDER\_PAYFEE

# 索引使用

## 应该建索引的字段

·经常作为查询条件的字段

·外键

·经常需要排序的字段

·分组排序的字段。

## 应该少建或者不建索引的字段

·经常需要修改的字段

·表中数据重复且分布平均的字段

## 一些SQL的写法会限制索引的使用

· where子句中如果使用in、or、like、!= <>，均会导致索引不能正常使用；

将"<>"换成">and<";将"is not null "换成">=chr(0)";

·使用函数时，该列就不能使用索引，+号属于函数所以会停用索引。

·比较不匹配数据类型时，该索引将会被忽略。

# sql优化

## 多表查询

·如果from是双表的查询时，大表放在前面，小表放在后面(基础表)。最后面的表是基础表。(只在基于规则的优化器中有效)

·如果三表查询时，选择交叉表(intersection table)作为基础表.(只在基于规则的优化器中有效)

## 条件语句优化

·写where条件时，有索引字段的判断在前，其它字段的判断在后；

·如果where条件中用到复合索引，按照索引列在复合索引中出现的顺序来依次写where条件；

·ORACLE采用自下而上的顺序解析WHERE子句,那些可以过滤掉最大数量记录的条件必须写在WHERE子句的末尾

## 其他关键字优化

查询数量较大时，使用表连接代替IN，EXISTS，NOT IN，NOT EXISTS等。