## 引言：一个被说烂了的原则

2:8原则

20%的代码消耗80%的性能

修改那20%的代码，可以优化那80%的性能

# 首先聊聊数据库设计

## 1、环境情况

* 数据库类型：ArteryBase x.x.x，生产环境推荐使用最新版本abase
* ABase、PostgreSQL、GP开发中，访问数据库对象（包括SQL、hibernate配置、mybatis配置等）时，必须指定schema名，如 ywst.t\_ms\_aj。禁止通过修改search\_path隐性指定schema查找路径等手段便捷访问数据库对象。

## 2、整体设计要求

* 业务表不做代码值、代码名称此类简单冗余
* 禁止使用触发器
* 必须做增量设计
* 不允许存在逻辑删除标志位
* 设计逻辑外键，必须定义索引
* 表必须有主键或唯一约束
  + abase表中必须有主键
  + GP特性为：若表设计中存在物理主键或唯一约束，则分布键必须包含对应主键或唯一约束列。所以在特定环境下，可以不设置物理主键或唯一约束
  + sybase ase表中主键必须为“唯一约束+非聚簇索引”，不能存在聚簇索引
* OLTP系统任意表最大列数不能超过80
  + 法院业务的旧设计中AJ、DSR表除外
* 多级子表逻辑外键设计要求
  + 一级子表要有主表的逻辑外键
  + 二级子表要有主表、一级子表逻辑外键
  + 三级子表要有主表、一级子表、二级子表逻辑外键
  + 依次类推
* 超大表建议考虑拆表
  + 大表（业务数据预估超过500W（ABase）、200W（ASE））考虑拆表，视具体业务和架构师讨论决定
* 同一表禁止双向复制

## 3、精度设计要求

* UUID类型主外键，设置为C32
* 数字原则上只有integer和numeric(20,4)
* 金额，单位为元。设置为numeric(20,4)
* VC精度控制
  + 推荐只有300/600/900三种精度。
* 名称汉字设置为VC300
* 外部存储文件的文件显示名称、文件名设置为VC300，存储路径设置为VC600
* 没有精度要求的，不要写精度：int/date/datetime/text等等
* 密码不用二进制类型，用16进制字符串

## 4、业务数据设计要求

* 同含义字段，名称、类型、精度必须统一
* 字段名称、字段中文描述、说明含义必须一致

# 最后一起聊聊我们的项目

## 1、使用32的jdk或不完整的jdk

架构原因、用户操作系统、控件（签章）原因等等都不是根本原因，出了问题早晚要修改。

——公司技术标准不允许32jdk

——没办法调高jvm大小（1.5G左右）；在实际的生产环境中，JVM默认的参数，往往不能满足需要，因此，就出现一个问题：既然操作系统已经安装64位，并且JDK有64位版本存在，生产环境下尽量在64位的os下安装使用64位jdk，从而把性能发挥至最优。

——没办法监控jconsole

## 2、不使用数据库连接池（c3p0\jndi\jdbc\dhcp）

“数据库连接池统一使用c3p0，且初始化、最大、最小连接都配置成同样的数值。”

——公司技术标准

虽然使用连接池，但是使用废弃的方法，不是由spring来管理需要手动关闭数据库连接。或是其他原因导致的连接池未生效。

## 3、sql语句

like、not like、in、not in、join、for循环中大量查询数据库、拆分大sql（逻辑清晰、多cpu并行）、batch DML（合并update）

badSql，IN后括号里的值超过1000以上时会报ora-01795错误。

Select \* from tableX

合同评审申请列表

SELECT

COUNT (\*)

FROM

(

SELECT

T\_XSGL\_HTPS.n\_SPZT AS spzt,

T\_XSGL\_HTPS.C\_CJR AS sqr,

T\_XSGL\_HTPS.C\_SPR AS spr,

T\_XSGL\_HTPS.C\_ID AS T\_XSGL\_HTPSC\_ID,

T\_XSGL\_HTPS.C\_HTWJLB AS T\_XSGL\_HTPSC\_HTWJLB,

T\_XSGL\_HTPS.C\_QYFXZ AS T\_XSGL\_HTPSC\_QYFXZ,

T\_XSGL\_HTPS.C\_QYFGSGSGX AS T\_XSGL\_HTPSC\_QYFGSGSGX,

T\_XSGL\_HTPS.C\_KHBH AS T\_XSGL\_HTPSC\_KHBH,

T\_XSGL\_HTPS.C\_CJR AS T\_XSGL\_HTPSC\_CJR,

T\_XSGL\_HTPS.C\_SSGS AS T\_XSGL\_HTPSC\_SSGS,

T\_XSGL\_HTPS.C\_SFZTB AS T\_XSGL\_HTPSC\_SFZTB,

T\_XSGL\_HTPS.N\_SPZT AS T\_XSGL\_HTPSN\_SPZT,

T\_XSGL\_HTPS.C\_QYFMC AS T\_XSGL\_HTPSC\_QYFMC,

T\_XSGL\_HTPS.C\_ZTBXMID AS T\_XSGL\_HTPSC\_ZTBXMID,

T\_XSGL\_HTPS.N\_SPJG AS T\_XSGL\_HTPSN\_SPJG,

T\_XSGL\_HTPS.C\_HTPSMC AS T\_XSGL\_HTPSC\_HTPSMC,

T\_XSGL\_HTPS.C\_FGSFZR AS T\_XSGL\_HTPSC\_FGSFZR,

T\_XSGL\_HTPS.C\_QYZJ AS T\_XSGL\_HTPSC\_QYZJ,

T\_XSGL\_HTPS.C\_ZXS AS T\_XSGL\_HTPSC\_ZXS,

T\_XSGL\_HTPS.C\_KHLXFS AS T\_XSGL\_HTPSC\_KHLXFS,

T\_XSGL\_HTPS.D\_SQRQ AS T\_XSGL\_HTPSD\_SQRQ,

T\_XSGL\_HTPS.C\_LCZT AS T\_XSGL\_HTPSC\_LCZT,

T\_XSGL\_HTPS.C\_KHLXR AS T\_XSGL\_HTPSC\_KHLXR

FROM

T\_XSGL\_HTPS T\_XSGL\_HTPS

) t\_0a84df06bc

算法复杂（np的分案）

## 4、索引（没建索引或是不当索引，差不多占了一半左右）

索引的作用是什么？

没有索引除了导致语句本身运行速度慢外，更是导致大量的磁盘读写操作，使得整个系统性能都受之影响而变差。

组合索引中字段的顺序是非常重要的，越是唯一的字段越是要靠前。

另外，无论是组合索引还是单个列的索引，尽量不要选择那些唯一性很低的字段。

比如说，在只有两个值0和1的字段上建立索引没有多大意义。

SELECT l.task\_list\_id, l.task\_percent, l.title, c.category\_name, l.parent\_task\_id, l.task\_manager, l.task\_performer\_list, l.create\_time, l.view\_user\_list, l.begin\_time, l.plan\_finish\_time, l.task\_content, l.attachment, l.task\_type,l.done\_state, l.done\_time FROM task\_list l LEFT JOIN task\_category c ON l.task\_category\_id = c.task\_category\_id WHERE l.task\_type <> 2 AND l.close\_state\_flg = 0 AND l.task\_performer\_list like '%admin%' AND l.done\_state < 1 order by l.plan\_finish\_time asc limit 0 , 20

优化后

SELECT l.task\_list\_id, l.task\_percent, l.title, c.category\_name, l.parent\_task\_id, l.task\_manager, l.task\_performer\_list, l.create\_time, l.view\_user\_list, l.begin\_time, l.plan\_finish\_time, l.task\_content, l.attachment, l.task\_type,l.done\_state, l.done\_time FROM task\_list l LEFT JOIN task\_category c ON l.task\_category\_id = c.task\_category\_id WHERE l.task\_type <> 2 AND l.close\_state\_flg = 0 AND MATCH(l.task\_performer\_list) AGAINST(';admin;' IN BOOLEAN MODE ) AND l.done\_state < 1 order by l.plan\_finish\_time asc limit 0 , 20

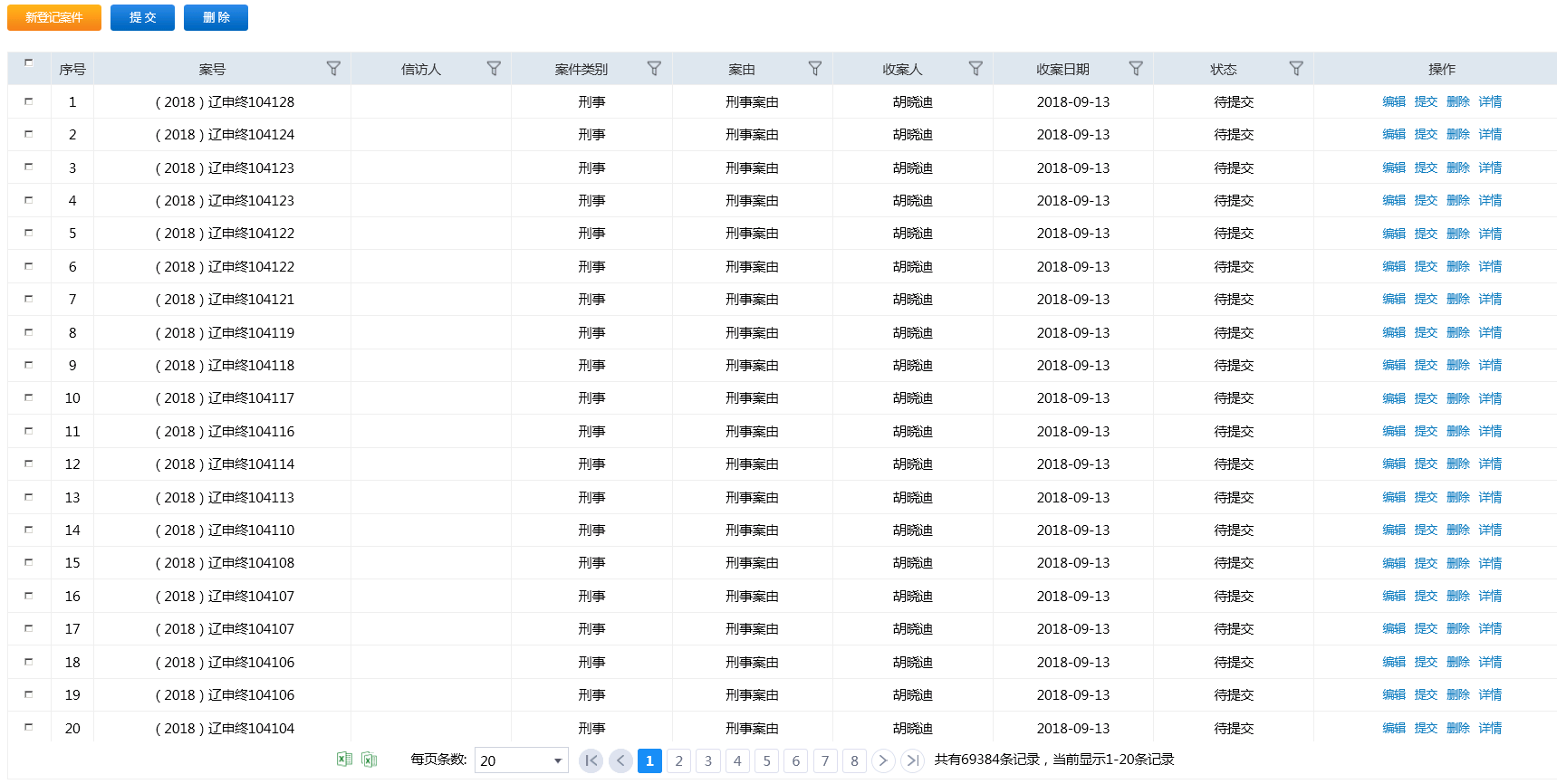
## 5、物理外键

是不是现在项目都是逻辑外键了？

## 6、绑定变量

减少重复解析

——减少sql分析时间、降低cpu使用率



CREATE TABLE "BI"."T\_PERSON" (

"C\_NAME" VARCHAR2(50 BYTE) NOT NULL ,

"N\_ID" NUMBER(8) NOT NULL,

"N\_FLAG" NUMBER(8) NOT NULL

)

未绑定变量

declare

begin

for i in 1 .. 999999 loop

UPDATE T\_PERSON SET N\_FLAG = 1 WHERE N\_ID = i;

commit;

end loop;

end;

绑定变量

declare

begin

for i in 1 .. 999999 loop

execute immediate 'UPDATE T\_PERSON SET N\_FLAG = 0 WHERE N\_ID = :a' using i;

commit;

end loop;

end;

## 7、业务表中的大字段（text、blob、clob）

———窄表（将遍历字段放入“窄表”）；短表（根据访问条件分区，例如按地区分表、按时间分表）；临时表（预统计，避免遍历大量原始数据）

———分库（生产库与只读库分离的目的）、np数据是按业务分库

## 8、分页和数据全部加载到内存

讨论下深度分页。

SELECT

T\_XSGL\_HTPS.n\_SPZT AS spzt,

T\_XSGL\_HTPS.C\_CJR AS sqr,

T\_XSGL\_HTPS.C\_SPR AS spr,

T\_XSGL\_HTPS.C\_ID AS T\_XSGL\_HTPSC\_ID,

T\_XSGL\_HTPS.C\_HTWJLB AS T\_XSGL\_HTPSC\_HTWJLB,

T\_XSGL\_HTPS.C\_QYFXZ AS T\_XSGL\_HTPSC\_QYFXZ,

T\_XSGL\_HTPS.C\_QYFGSGSGX AS T\_XSGL\_HTPSC\_QYFGSGSGX,

T\_XSGL\_HTPS.C\_KHBH AS T\_XSGL\_HTPSC\_KHBH,

T\_XSGL\_HTPS.C\_CJR AS T\_XSGL\_HTPSC\_CJR,

T\_XSGL\_HTPS.C\_SSGS AS T\_XSGL\_HTPSC\_SSGS,

T\_XSGL\_HTPS.C\_SFZTB AS T\_XSGL\_HTPSC\_SFZTB,

T\_XSGL\_HTPS.N\_SPZT AS T\_XSGL\_HTPSN\_SPZT,

T\_XSGL\_HTPS.C\_QYFMC AS T\_XSGL\_HTPSC\_QYFMC,

T\_XSGL\_HTPS.C\_ZTBXMID AS T\_XSGL\_HTPSC\_ZTBXMID,

T\_XSGL\_HTPS.N\_SPJG AS T\_XSGL\_HTPSN\_SPJG,

T\_XSGL\_HTPS.C\_HTPSMC AS T\_XSGL\_HTPSC\_HTPSMC,

T\_XSGL\_HTPS.C\_FGSFZR AS T\_XSGL\_HTPSC\_FGSFZR,

T\_XSGL\_HTPS.C\_QYZJ AS T\_XSGL\_HTPSC\_QYZJ,

T\_XSGL\_HTPS.C\_ZXS AS T\_XSGL\_HTPSC\_ZXS,

T\_XSGL\_HTPS.C\_KHLXFS AS T\_XSGL\_HTPSC\_KHLXFS,

T\_XSGL\_HTPS.D\_SQRQ AS T\_XSGL\_HTPSD\_SQRQ,

T\_XSGL\_HTPS.C\_LCZT AS T\_XSGL\_HTPSC\_LCZT,

T\_XSGL\_HTPS.C\_KHLXR AS T\_XSGL\_HTPSC\_KHLXR

FROM

T\_XSGL\_HTPS T\_XSGL\_HTPS

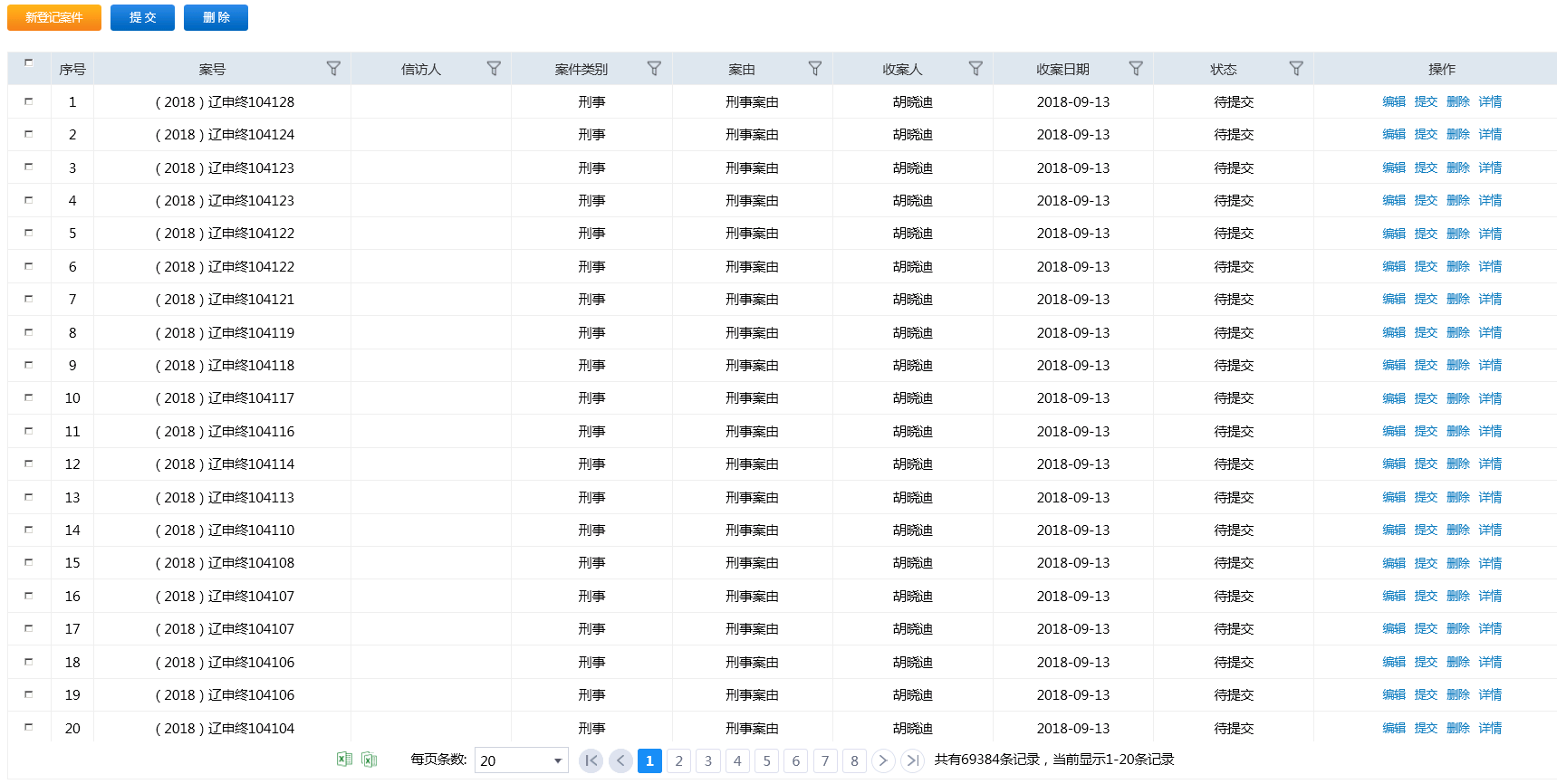
ORDER BY

T\_XSGL\_HTPS.d\_sqrq DESC

LIMIT 10 OFFSET 10280

OFFSET 子句忽略的行仍然需要在服务器内部计算；因此，一个很大的 OFFSET 可能还是不够有效率的，并且会消耗数据库buffer。

如非必要建议去掉列表选择“最后一页”的功能。





## 9、定时任务和轮询方式

要特别注意开机（确切的说是服务启动）的定时任务。曾经发现某项目开机定时任务执行2000多秒，涉及表数据量300W（从目前公司项目的发展趋势看数据量不大），

问题产生的原因是低效率的sql（like、无分页），该任务执行结束前，其他对该表的操作都比较慢，会出现等待和锁的现象。

还有就是数据抽取类的定时任务，要合理实施，分时分区进行，避免同一时刻并发。

媒资轮询科技法庭；

开庭公告轮询科技法庭；

业务系统轮询np；

不易在测试中发现问题，测试一般都是针对本系统；

缺点：不符合业务逻辑（就像手机欠费）、非实时、给轮询和被轮询系统造成巨大压力（np的性能问题不仅仅是np本身的问题）。

## 10、无效的页面请求以及重复的页面请求

登陆之前输入用户名判断所在法院。

## 11、默认参数（数据库和中间件）

应用服务器特别想说说tas。

TAS 在win7、win2008 server R2，win2012 上，如果connector使用nio的方式，在并发时有bug，导致系统无法响应。所以在以上环境中，要将connector设置为bio方式。

这个需要手动配置，因为TAS默认的配置是nio模式的。具体设置的位置为 TAS的ect/jetty.xml 文件中。

Jvm合理设置。

## 12、上传无限制包括文件格式、个数、大小。

即使用户没提出明确要求，开发时也要考虑做限制。

## 13、重视硬件条件

目前比较突出的是音视频项目的硬件设备（手持终端、摄像头）和带宽问题（wifi、3g网、4g网）。

## 14、重视加压开发和加压测试环节，防洪不能只靠一道堤坝，那是在赌人品。

加压数据不能只准备数据量，还要考虑数据分布。Group by。Order by。

## 15、数据库死锁，表锁、页锁和行锁。

数据库有锁怎么办？

Sybase数据库默认的锁是什么？

## 16、加密和解密。选择合适的算法。（安全性与性能的矛盾，平衡是一门艺术。）

HTTP Status-Code=500 (Exception thrown executing [AnnotatedAction@602085d0 targetAction = com.thunisoft.summer.component.ssoserver.login.TuscAuthenticationViaFormAction@25fbd252, attributes = map['method' -> 'submit']] in state 'submit' of flow 'login-webflow' -- action execution attributes were 'map['method' -> 'submit']'; nested exception is java.lang.ArrayIndexOutOfBoundsException: too much data for RSA block)

## 17、谈谈接口

正常的情况下要考虑数据量和并发的问题。（测试时要关闭缓存）

还有考虑异常的情况（返回null值判断、返回非唯一结果、输入参数中文转码、断网后重新执行策略）

## 18、重视用户环境

实体机还是虚拟机、服务器还是PC机、ssd还是普通硬盘、局域网还是外网、专网还是vpn

## 19、首页面的问题

超多统计查询（必要？异步？）、页面大小

## 20、配置文件

日志模式对tas中间件影响很大，正式发布时一定要关闭控制台和日志文件的debug等级。

config.properties中js加载类型选项loadType = 1时，程序会单独加载每个js文件，导致磁盘使用率过高。将选项修改为loadType = 2，程序统一加载js文件，磁盘使用率过高的问题得到解决。

## 21、针对数据库特性

**Update怎么办？**

update test1 set name = (select name from test2 where test1.id = test2.id)；

这个语句的功能？效率怎么样

merge into test1 using test2

on (test1.id = test2.id)

when matched then

update set test1.name = test2.name;

一次扫描，效率高

**Insert into怎么办？**

INSERT INTO table\_name VALUES (值1, 值2,....) 不符合规范为什么？

INSERT INTO table\_name (列1, 列2,...) VALUES (值1, 值2,....)

INSERT INTO Table2(field1,field2,...) select value1,value2,... from Table1；

INSERT INTO Table3(field1,field2,...) select value1,value2,... from Table1；

INSERT INTO Table4(field1,field2,...) select value1,value2,... from Table1；

……

insert all

into Table2 values(value1, value2,...)

into Table3 values(value1, value2,...)

into Table4 values(value1, value2,...)

select value1,value2,...

from Table1;

insert all

when (value1= '条件1') then

into Table2 values(value1, value2,...)

when (value1= '条件2') then

into Table3 values(value1, value2,...)

when (value1= '条件3') then

into Table4 values(value1, value2,...)

select value1,value2,...

from Table1;

## 结语：纸上得来终觉浅 绝知此事要躬行