# Ibear中sql数据缓存功能概要设计

1. 功能详细描述：

|  |  |  |
| --- | --- | --- |
| 阶段 | 功能简介 | 状态 |
| 1 | 调用相同sql时，除第一次外，后续不再通过数据库  ，而是通过第一次sql的缓存结果直接取值； | √ |
| 1 | 可控制哪些sql需要缓存； | √ |
| 1 | 释放缓存的方式和时间点； | √ |
| 2 | 当sql表名相同、条件相同，但是查询字段不同时，会根据查询字段自动进行匹配； | √ |
| 3 | 对sql条件进行合并，减少数据库访问次数，对数据进行缓存； | × |
| 3 | 自动对sql结果进行缓存（该表的全部字段值），而不是局限于sql中的字段值； | × |
| 3 | 对长时间没有变更的数据进行缓存（可选择）； | × |
| 3 | 通过机器学习（MachLearnEnable）的方法，分析程序所执行的sql； | × |

1. 设计思路

增加缓存的全局配置（cacheModelsEnabled）；

增加自动清理配置（autoClearEnabled）：

a、增加 “事务清理TransationClear”：程序commit、rollback后自动清理缓存数据；

b、增加“溢出清理OverflowClear”：超过最大缓存数量（maxOverflaw）后，自动清理最早进入缓存的内容，maxOverflaw默认值128；

c、增加“手工清理**SelfClear**”：需要手动调用清理函数；

新增全部清理：清理全部的缓存数据；

新增模型方式清理：清理某模型的缓存数据；

新增栈模式清理：清理最早缓存的一条数据；

1. 实现方式

解析sql语句（select），获得所有字段名、表名、索引名（where条件中的字段名），生成散列表，记录模型结构明细：

map<表名,Map<索引,字段> > table\_key

再根据查询结果生成散列表，记录模型存储数据明细：

map<表名,Map<索引,List<列值> > > table\_value;

数据结构设计详见下图：



数据结构设计

例：

语句：select b,d from abc where a=? and c=?;

则：columns.insert(ac,bd);

table\_key.insert(abc,columns);

绑定变量值：1，3

查询结果为：2，4

values.insert(13,24);

table\_value.insert(abc,values);

一阶段：

相同sql执行时，解析获得表名、字段名、绑定变量值；（条件必须相同，否则无法找到）

table\_key.find(abc)获得columns，根据ac获得acbd；

在根据table\_value.find(abc)获得values，values.find(13)如果不为空则返回值，再根据columns获得的字段名进行对应；如果为空，继续insert；

二阶段：

sql模型相同，条件相同，对之前没有包含的“查询列”进行合并，并缓存；对已有的列直接从缓存中取值；

详细流程详见下图：



*流程图*