mysql索引以及执行计划

# 执行计划

mysql如何执行一条sql语句，包括sql查询的顺序、是否使用索引、以及使用的索引信息等内容。(<https://juejin.im/post/5a52386d51882573443c852a>)

## 基本语法

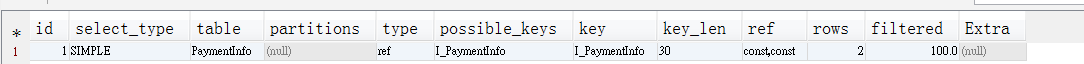
--基本语法 explain sql语句

explain select \* from PaymentInfo;

--其他

explain extended select \* from PaymentInfo where Status = 0 and CreatTime='20190324';

执行计划信息结构



1. Id

查询中各个子查询的执行顺序；

id相同执行顺序由上至下;

id不同，id值越大优先级越高，越先被执行；

id为null时表示一个结果集，不需要使用它查询，常出现在包含union等查询语句中；

1. select\_type

# 索引

索引是一种特殊的查询表，可以被数据库搜索引擎用来加速数据的检索。

注：索引能够提高 SELECT 查询和 WHERE 子句的速度，但是却降低了包含 UPDATE 语句或 INSERT 语句的数据输入过程的速度。

索引创建语法: create index 索引名称 on 表名 (字段名….)。

若创建唯一索引需在create后加上unique。

常用索引的分类：单列索引、唯一索引、复合索引。

1. 单列索引

仅包含一个字段名的索引，称单列索引。例如：

/\*==============================================================\*/

/\* Index: I\_PaymentInfo1 \*/

/\*==============================================================\*/

create index I\_PaymentInfo1 on PaymentInfo

(

TransNo

);

1. 唯一索引

唯一索引也是一种特殊的索引。即不允许被索引的数据列包含重复的值。

/\*==============================================================\*/

/\* Index: UK\_PaymentInfo2 \*/

/\*==============================================================\*/

create unique index UK\_PaymentInfo2 on PaymentInfo

(

TransNo,

Status

);

在表中，若插入表中的TransNo和Status值在数据库中已存在时，此时报错，无法插入。

1. 复合索引

即索引中的字段至少两个或两个以上。例如：

/\*==============================================================\*/

/\* Index: I\_PaymentInfo \*/

/\*==============================================================\*/

create index I\_PaymentInfo on PaymentInfo

(

Status,

CreatTime

);

## 最左前缀原则

顾名思义，最左优先原则。例如已建立复合索引UK\_PaymentInfo2 index(TransNo, Status){注:index(cloumname,…)，表示建立索引}两个字段。其实已经包含了index(TransNo)，和index(TransNo, Status)两个索引。

若建立索引index(TransNo, Status,WorkDate),就已经包含了index (TransNo), index(TransNo, Status), index(TransNo, Status,WorkDate)三个索引。

## 避免使用索引情况

# 例子

首先建立表 缴费信息表 PaymentInfo

drop index I\_PaymentInfo1 on PaymentInfo;

drop index I\_PaymentInfo on PaymentInfo;

drop index UK\_PaymentInfo on PaymentInfo;

drop index UK\_PaymentInfo2 on PaymentInfo;

drop table if exists PaymentInfo;

/\*==============================================================\*/

/\* Table: PaymentInfo \*/

/\*==============================================================\*/

create table PaymentInfo

(

Id int not null auto\_increment,

AcceptId varchar(32) not null,

TransNo varchar(32) not null,

Status int not null,

CreatTime varchar(8) not null,

Extent varchar(32),

primary key (Id)

);

/\*==============================================================\*/

/\* Index: UK\_PaymentInfo \*/

/\*==============================================================\*/

create unique index UK\_PaymentInfo on PaymentInfo

(

AcceptId

);

/\*==============================================================\*/

/\* Index: UK\_PaymentInfo2 \*/

/\*==============================================================\*/

create unique index UK\_PaymentInfo2 on PaymentInfo

(

TransNo,

Status

);

/\*==============================================================\*/

/\* Index: I\_PaymentInfo \*/

/\*==============================================================\*/

create index I\_PaymentInfo on PaymentInfo

(

Status,

CreatTime

);

/\*==============================================================\*/

/\* Index: I\_PaymentInfo1 \*/

/\*==============================================================\*/

create index I\_PaymentInfo1 on PaymentInfo

(

TransNo

);

---插入几条测试记录

insert into PaymentInfo (AcceptId,TransNo,Status,CreatTime) values ("A000001", "T000001", 0, "20190323");

insert into PaymentInfo (AcceptId,TransNo,Status,CreatTime) values ("A000002", "T000002", 0, "20190323");

insert into PaymentInfo (AcceptId,TransNo,Status,CreatTime) values ("A000003", "T000003", 0, "20190323");

insert into PaymentInfo (AcceptId,TransNo,Status,CreatTime) values ("A000011", "T000011", 0, "20190323");

insert into PaymentInfo (AcceptId,TransNo,Status,CreatTime) values ("A000012", "T000012", 0, "20190324");

insert into PaymentInfo (AcceptId,TransNo,Status,CreatTime) values ("A000013", "T000013", 0, "20190324");