第四次实验

> 22375080
>
> 杨佳宇轩

Q1

- 范围查询

  通过实验结果可以发现对于等值查询，B+树索引的查询速度非常快，但范围查询时和普通查询效果相当，因为即使找到了区域，也需要进行遍历

  - B+树查询

    ```
    select * from bdcopy1 where v>10000;
    ```

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	bdcopy1	(Null)	ALL	index_v	(Null)	(Null)	(Null)	3192192	50.00	Using where

  - 普通查询

    ```
    select * from bdcopy2 where v>10000;
    ```

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	bdcopy2	(Null)	ALL	(Null)	(Null)	(Null)	(Null)	3192192	33.33	Using where

    ```
    select * from bdcopy1 where v>10000
    > OK
    > 查询时间: 0.006s


    select * from bdcopy2 where v>10000
    > OK
    > 查询时间: 0.001s


    explain select * from bdcopy1 where v>10000
    > OK
    > 查询时间: 0s


    explain select * from bdcopy2 where v>10000
    > OK
    > 查询时间: 0s
    ```

- 等值查询

  - B+树查询

```
Select * from bdcopy1 where v=3535353;
```

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	bdcopy1	(Null)	ref	index_v	index_v	9	const	1	100.00	(Null)

- 普通查询

```
Select * from bdcopy3 where v=3535353;
```

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	bdcopy3	(Null)	ALL	(Null)	(Null)	(Null)	(Null)	3192192	10.00	Using where

```
Select * from bdcopy1 where v=3535353
> OK
> 查询时间: 0s


Select * from bdcopy3 where v=3535353
> OK
> 查询时间: 6.059s


EXPLAIN Select * from bdcopy1 where v=3535353
> OK
> 查询时间: 0s


EXPLAIN Select * from bdcopy3 where v=3535353
> OK
> 查询时间: 0s
```

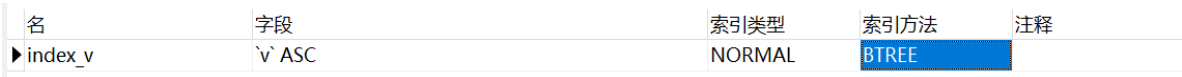
Q2

- 哈希查询速度

```
Select * from bdcopy1 where v=19997
> OK
> 查询时间: 0.012s


Select * from bdcopy3 where v=19997
> OK
> 查询时间: 0.004s


EXPLAIN Select * from bdcopy1 where v=19997
> OK
> 查询时间: 0s


EXPLAIN Select * from bdcopy3 where v=19997
> OK
> 查询时间: 0s
```

可以发现，两张表的查询时间基本相同

重新打开 `bdcopy3` 可以看到实际建立的是 B+ 树索引



名	字段	索引类型	索引方法	注释
index_v	`v` ASC	NORMAL	BTREE	

- 建立memory引擎

```
CREATE TABLE bdcopy4 (bid int, v bigint, s smallint, INDEX USING
HASH (v)) engine = memory;
insert into bdcopy4 select * from bdtable;
create table bdcopy5 like bdtable;
insert into bdcopy5 select  * from bdcopy4;
```

insert into bdcopy4 select * from bdtable	1114 - The table 'bdcopy4' is full	3.87s

创建时报错，可知由于 Memory 存储引擎将表数据存储在计算机的内存中，而不是存储在磁盘上，由于内存的成本相对较高，通常只有有限的内存可供使用

对比 `bdcopy4` 和 `bdcopy5` 的查询速度

```
Select * from bdcopy4 where v=19997
> OK
> 查询时间: 0s


Select * from bdcopy5 where v=19997
> OK
> 查询时间: 0.505s
```

可以发现，没有使用 Hash 索引的 `bdcopy5` 的查询速度明显低于 `bdcopy4`

由此可得，哈希查询速度之快

Q3

- 等值查询

```
select * from bdcopy1(23)
where v = 399998
```

`bdcopy1`：88ms

类型	文本	持续时间 (毫秒)	行	结果
SQL / User	select * from bdcopy1¶where bid = 399998¶LIMIT 0, 800000	88	1	成功

`bdcopy2`：89ms

类型	文本	持续时间 (毫秒)	行	结果
SQL / User	select * from bdcopy2¶where bid = 399998¶LIMIT 0, 800000	89	1	成功

- 范围查询

```
select * from bdcopy1(2)
where bid between 10000 and 120000
```

`bdcopy1`：1213ms

文本	持续时间 (毫秒)	行	结果
select * from bdcopy1¶where bid between 10000 and 12000...	1,213	110001	成功

`bdcopy2`：911ms

类型	文本	持续时间 (毫秒)	行	结果
SQL / User	select * from bdcopy2¶where bid between 10000 and 12000...	911	110001	成功

由此可见，等值查找时，聚簇索引和B+树索引时间差距不大，而在进行范围查找时，由于聚簇索引数据按照物理顺序排放，在聚簇索引的列上进行范围查找时只需要查找少数几个页，使得聚簇索引速度显著大于B+树索引

Q4

```
select * from bdcopy2 where v=54167 and s=3;
select * from bdcopy2 where v=54167;
select * from bdcopy2 where s=3;
EXPLAIN select * from bdcopy2 where v=54167 and s=3;
EXPLAIN select * from bdcopy2 where v=54167;
EXPLAIN select * from bdcopy2 where s=3;
```

名	字段	索引类型	索引方法	注释
index_vs	`v` ASC, `s` ASC	NORMAL	BTREE	

运行结果：

第一个查询：

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	bdcopy2	(Null)	ref	index_vs	index_vs	12	const,const	1	100.00	(Null)

第二个查询：

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	bdcopy2	(Null)	ref	index_vs	index_vs	9	const	26	100.00	(Null)

第三个查询：

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	bdcopy2	(Null)	ALL	(Null)	(Null)	(Null)	(Null)	3192192	10.00	Using where

查询时间：

```
select * from bdcopy2 where v=54167 and s=3
> OK
> 查询时间: 0.006s


select * from bdcopy2 where v=54167
> OK
> 查询时间: 0.003s


select * from bdcopy2 where s=3
> OK
> 查询时间: 0.006s


EXPLAIN select * from bdcopy2 where v=54167 and s=3
> OK
> 查询时间: 0.001s


EXPLAIN select * from bdcopy2 where v=54167
> OK
> 查询时间: 0s


EXPLAIN select * from bdcopy2 where s=3
> OK
> 查询时间: 0s
```

可以发现，三者运行时间基本相同

但只有在两个键都存在或者使用第一个键的时候会使用联合索引

Q5

```
Explain select * from bdcopy1 where bid/2=30000;
Explain select * from bdcopy1 where bid=30000*2;
```

对于第一种查询

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	bdcopy1	(Null)	ALL	(Null)	(Null)	(Null)	(Null)	3192192	100.00	Using where

对于第二种查询

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	bdcopy1	(Null)	ref	index_bid	index_bid	5	const	1	100.00	(Null)

可以发现，对于当查询键单独属于一侧的时候可以调用该字段的查询方式，否则为普通索引查询