【第21话：MySQL里面常见的几个面试题】

Hello 小伙伴们，这节课给大家讲解一下MySQL里面几个小的常见问题

1.MySQL支持的表连接类型有几种

MySQL支持的表连接类型分为分为：内连接、左外连接、右外连接。Oracle中支持全外连接，但MySQL不支持。

给定测试表及测试数据

```
create table tb_teacher(
    t_id int primary key auto_increment,
    t_name varchar(20)
);

insert into tb_teacher values(default,'老师1');
insert into tb_teacher values(default,'老师2');

create table tb_student(
    s_id int primary key auto_increment,
    s_name varchar(20),
    s_t_id int
);
insert into tb_student values(default,'学生1',1);
insert into tb_student values(default,'学生2',null);
```

笛卡尔积：多表连接查询出来的结果，不去除任何未关联数据。例如表A有N条数据，表B有M条数据，查询出来的结果有N*M条。

```
select * from tb_student s ,tb_teacher t;
```

内连接:使用inner join，表示查询出来两个具有关联的数据，无关联性的数据不会查询出来。也可以直接使用多表查询

```
select * from tb_student s inner join tb_teacher t on s.s_t_id=t.t_id;
```

左外连接使用left outer join，表示即使左表存在未关联数据，也被查询出来。在left outer join左侧的表叫做左表,右侧的表叫做右表。

```
select * from tb_student s left outer join tb_teacher t on s.s_t_id=t.t_id;
```

右外连接使用right outer join，表示即使右表存在未关联数据，也被查询出来。在left outer join右侧的表叫做右表。

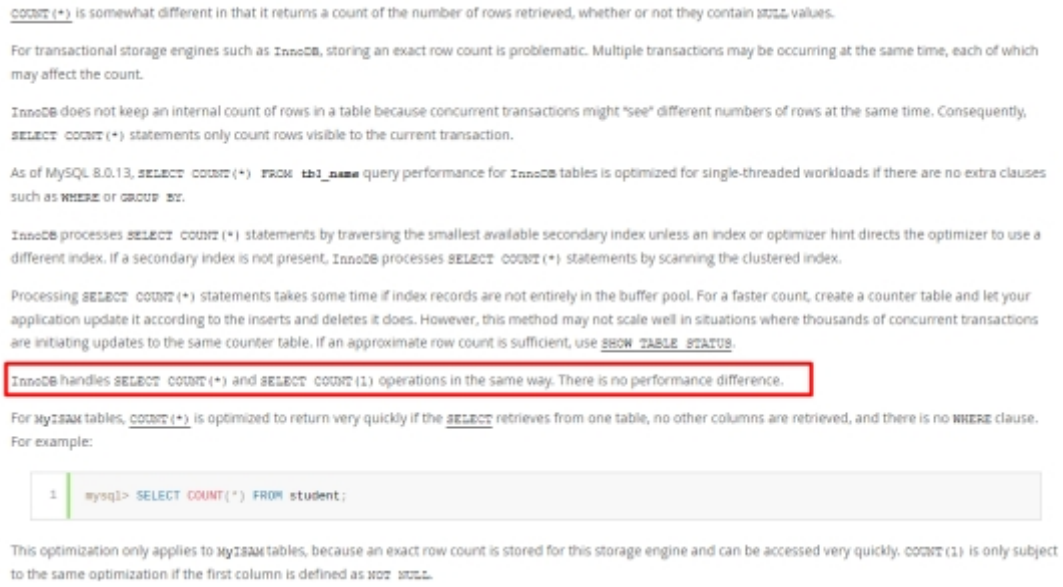
select * from student s right outer join teacher t on s.stid=t.tid;select * from teacher t right outer join student s on s.stid=t.tid;

```
select * from tb_student s right outer join tb_teacher t on s.s_t_id=t.t_id;
```

2.请说一下count(*),count(列),count(1)的区别

对于InnoDB引擎count(*)和count(1)没有性能的区别，都是自动寻找主键列或唯一索引。统计的时候都是统计包含null的行，所以效果都是返回表中数据的行数。

官方文档参考地址： https://dev.mysql.com/doc/refman/8.0/en/aggregate-functions.html



`COUNT(*)` is somewhat different in that it returns a count of the number of rows retrieved, whether or not they contain `NULL` values.

For transactional storage engines such as `InnoDB`, storing an exact row count is problematic. Multiple transactions may be occurring at the same time, each of which may affect the count.

`InnoDB` does not keep an internal count of rows in a table because concurrent transactions might "see" different numbers of rows at the same time. Consequently, `SELECT COUNT(*)` statements only count rows visible to the current transaction.

As of MySQL 8.0.13, `SELECT COUNT(*) FROM tbl_name` query performance for `InnoDB` tables is optimized for single-threaded workloads if there are no extra clauses such as `WHERE` or `GROUP BY`.

`InnoDB` processes `SELECT COUNT(*)` statements by traversing the smallest available secondary index unless an index or optimizer hint directs the optimizer to use a different index. If a secondary index is not present, `InnoDB` processes `SELECT COUNT(*)` statements by scanning the clustered index.

Processing `SELECT COUNT(*)` statements takes some time if index records are not entirely in the buffer pool. For a faster count, create a counter table and let your application update it according to the inserts and deletes it does. However, this method may not scale well in situations where thousands of concurrent transactions are initiating updates to the same counter table. If an approximate row count is sufficient, use `SHOW TABLE STATUS`.

`InnoDB` handles `SELECT COUNT(*)` and `SELECT COUNT(1)` operations in the same way. There is no performance difference.

For `MyISAM` tables, `COUNT(*)` is optimized to return very quickly if the `SELECT` retrieves from one table, no other columns are retrieved, and there is no `WHERE` clause. For example:

```
mysql> SELECT COUNT(*) FROM student;
```

This optimization only applies to `MyISAM` tables, because an exact row count is stored for this storage engine and can be accessed very quickly. `COUNT(1)` is only subject to the same optimization if the first column is defined as `NOT NULL`.

而count(列) 统计列中包含多少行数据，不包含NULL值。

总体说明：

在InnoDB引擎中count(*)和count(1)性能没有什么差别

count(列)需要看列和 `count(*)` 优化后的列情况，如果count(列)使用了非索引列，而表中包含索引列则 `count(*)` 更快。如果count(列)和 `count(*)` 优化后的是同一个列则性能没有什么差别。如果表中没有索引则count(列)和 `count(*)` 性能也没有什么差别。

3.聚集索引和非聚集索引的区别

聚集和非聚集最主要的区别就是索引是否和数据在一起。

聚集索引（聚簇索引）之所以叫"聚集"是因为索引和数据一起存储。叶子节点中存储索引和数据。因为数据顺序和索引顺序是一样的，且顺序存储，所以查询速度比较快。当然了因为有顺序的所以新增需要重新排序会影响速度。简单记忆：主键索引就是聚集索引。

非聚集索引之所以叫"非聚集"是因为叶子节点中只存储主键和索引列。所以在非聚集索引查询时需要通过主键再去查询想要的数据。在叶子节点上通过主键查询数据的过程就是所谓的回表。