Mysql版本：

5.7整合了第三方公司新的存储引擎

Mysql逻辑分层：

BS架构一般都是客户端服务端

客户端发出请求🡪

连接层：提供与客户端连接的服务

服务层：提供各种用户使用的接口CURD

还提供了SQL优化器功能：MYSQL Query Optimizer

mysql会优化你写的过于复杂的语句，

引擎层：提供了各种存储数据的方式InnoDB，MyISAM

重点区别

InnoDB:设计原则事务优先（适用于高并发，行锁）

MyISAM:性能优先，（表锁）

存储层：存储数据

SQL优化：

优化原因：性能低，执行时间长，等待时间长，SQL语句欠佳（连接查询），索引失效，服务器参数设置不合理（缓冲，线程数）

SQL:

编写过程：

select distinct ..from ..join .. on ..where ..group by ..having ..order by .. limit ..

解析过程：

from ..on ..join ..where ..group by ...having ...select distinct...order by ...limit..

主要在优化索引

MYSQL语句:

去重：select distinct [字段] from [表];

条件查询：

条件运算符：> < = != <> >= <=

逻辑运算符：and or not && || !

模糊查询:like between and in is null

等于号不能判断null值，需要使用is null来判断

函数：

单行函数：

分组函数：

分组查询：

查询列表必须特殊，要求是分组函数和group by后出现的字段

按多个字段分组查询

连接查询：

笛卡尔乘积现象：表一有m行，表二有n行，结果=m\*n

按年底啊查询：

sql192标准\*1

sql199标准\*2

按功能分类：

内连接：

等值连接

非等值连接

自连接

外连接：

左外连接

右连接

全外连接

交叉连接：

\*1sql192：

select 【字段】 from 【表1】 ，【表2】 where 【条件】

执行逻辑通过第一张表的查出的结果筛选第二张表

等值查询，where之后添加条件，笛卡尔积后的所有结果，筛选出符合条件的结果

\*2sql199:

join默认就是inner join 的省略

select 【字段】from 【表1】 join 【表2】 on 【连接条件】 where 【筛选条件】

交叉连接就是笛卡尔积

子查询：

出现在其他语句中的select 语句我们都称为子查询

分类：

按照子查询位置不同：

select 后面

标量子查询

from 后面

表子查询

where或having后面

标量子查询

列子查询

行子查询：格式select \* from 【表】where (【字段1】,【字段2】) = (【子查询语句】)

exists后面

表子查询

按照结果集的行列数不同：

标量子查询（结果集为一行一列）

列子查询（结果集只有一行多列）

行子查询（结果集可以有多行多列）

表子查询（结果集不限定）

标量查询，一般使用=,>,<,<=,>=,<>操作符

列子查询，一般搭配多行操作符使用in,any/some,all in等于集合里任意一个值，any范围比较有任意一个值，all必须范围符合所有值

MYSQL

系统变量：由系统提供

全局变量：所有用户都生效

会话变量：作用于每个会话

自定义变量

用户变量

全局变量

explain:

id：id越大，执行顺序优先，id相同，最上面先执行

select\_type:显示查询的类型

simple简单查询，没有子查询或联合查询

primary查询中包含任何复杂部分，最外层标记

subqurey在select或where中包含了子查询

drived在from中子查询被标记为drived（衍生），会放到临时表中

union联合查询第二个查询被标记为连表查询

type：访问类型：system,const,eq\_ref,range,index,all

system:表中只有一条记录

const:常量，主键或唯一键，只通过一次索引就返回结。

eq\_ref:唯一性索引扫描

ref:非唯一性索引扫描，返回匹配某个单独值的所有行

range:使用了索引，只检索给定范围的行

index:全索引扫描，所需要的数据在索引上，所以不需要扫描表，直接扫描索引

all：全表扫描

ref:查看哪些字段被索引用到了

rows:实际扫描行数