DBA

1、负责管理和维护Tidb、MySQL、Mongo、Redis、ES、Tidb数据库实例；

2、设计和开发数据库运维管理工具、制定缓存系统的使用方案；

3、持续改进数据库和缓存系统方案，保障数据库的可靠、稳定、高效地运行。

4、处理数据库的日常维护工作，及出现重大问题时的应急恢复；

5、提供数据库开发支持，根据业务需求进行索引优化和调优

6、定期进行系统的健康检查，并对性能问题进行定位和排查。

7、配合开发同事完成数据库一系列优化工作。

8、熟练执行数据库性能调优、监控、备份、异常处理、故障恢复

9、熟悉数据库的高可用和动态扩展方案

10、能提供良好的读写分离方案（IO读/写）

1、5年以上Linux平台数据库管理经验；

2、从事Mysql技术方面的支持和维护实际工作经验在5年以上；

3、精通MySQL管理，数据备份，优化；

4、有NoSQL的管理、优化经验, 熟悉Redis、MongoDB, Elasticsearch管理，数据备份，优化；

5、精通MySQL数据库引擎原理，深入理解数据库引擎的实现机制，能根据相关机制优化数据库；

6、善于学习总结，关注数据库技术发展，了解新技术；

7、具备较强的责任心，良好的沟通能力，积极分享，带头提升整个团队的技术水平；

8、接触过千万级数据库的处理经验

9、有灾难性故障处理经验

10、熟悉Tidb实例管理，备份与优化的优先

MYSQL面试题

MYISAM 和 INNODB 的区别

1.表锁行锁

2.事务处理

应用层的区别：

1，MyISAM不支持外键，InnoDB支持，这是非关系型和关系型数据库的最大区别

2，MyISAM不支持事务，InnoDB支持

3，MyISAM不支持行锁，InnoDB支持，InnoDB也会退化为表锁

4，MyISAM支持全文索引，InnoDB不支持（5.7版本后支持）

5，MyISAM可以没有主键，InnoDB必须有（不设会内置）

设计层的区别：

1，先讲概念，什么是聚簇索引和非聚簇索引

1.1，聚簇索引：主键索引的叶子节点存储主键值；

1.2，非聚簇索引：主键索引的叶子节点不存储主键值，仅存主键值的地址；

仅考虑算法层面的话，从聚簇索引中获取数据通常比在非聚簇索引中查找要快。

2，MyISAM和InnoDB索引都采用B+树结构（这里不讨论hash），区别在于

2.1，MyISAM主键索引是非聚簇索引；

2.2，InnoDB主键索引是聚簇索引；

2.3，InnoDB普通索引（二级索引）是非聚簇索引；

3，数据存储方式的不同

3.1，MyISAM索引文件和数据文件是分离的（.myi索引文件和.myd数据文件）

3.2，InnoDB的数据文件按主键聚集（.idb数据文件）

4，为什么MyISAM查询更快？

4.1，InnoDB要缓存数据块，MyISAM只缓存索引，加载索引可能更快

4.2，InnoDB寻址先到块再到行，MyISAM直接寻址到文件offset

4.3，InnoDB要维护MVCC一致

权限管理

(1) mysql部署依赖的系统用户权限

root: system mysql命令

mysql: mysql系统用户:/data /log

数据安全

(2)mysql访问用户，create user。

数据库权限非常重要:

一般业务需求: select (主,从) , delete、update、insert(主库)足以。千万不要给业务root用户，DBA用root?不要。

root用户管理:权限只限于127.0.0.1/localhost。登录mysql主机才能操作。

dbadmin: all,ddl+dml+select

replica用户: replica slave，replica client，切记权限扩小化

权限IP限制?dbadmin@-%-? DB服务器开放了外网，撞库风险。权限管理平台，可以定到10.10.10.10,10.10.10.%，10.10.%。千万不要%?

企业办公网访问?建议禁止，提供sql查询平台解决，跳转堡垒机进行访问。

数据类型

1、数值 INT TINYINT BIGINT

2、浮点 FLOAT DOUBLE DECIMAL

float:非精度类型(四舍五入), decimal(20，10)

3、字符 CHAR VARCHAR TEXT

工作技巧 宁用varchar(10000) 不用text

4、二进制 BLOB

5、时间 DATE DATETIME TIMESTAMP

timestamp玩法:

(1)存的值固定。时区不同,显示不同

(2)长度4字节

(3)updato-time timestamp not null default current-timestamp on update current-timestamp;业务:1、update name update time;有了上面方式，只更新要更新的内容，update-time自动更新。

datetime 1970之前,2038年之后,就用它。8字节够用。

注意点:千万不用用varchar存储时间varchar (255)，时间范围查询。很慢!

6、布尔

varchar(10) 括号中的数代表的是什么？

代表的是字符长度 可以用 char\_length(table.volumn)进行查询，也就是可以存10个字符包括标号，而不是字节长度。

如果超过会报错字符长度过长，则不会写入。

如果实在想写入，可以修改参数 sql\_mode 为 NO\_ENGINE\_SUBSITITUTION，但写入后也只是依据字符长度截取的内容

datetime timestamp的区别

1.datetime 是8个字节 timestamp 是4个字节

2.timestamp 是数值

MySQL日志介绍

innodb日志:redo(事务流程、MCC) , undo (功能、不同版本undo特性)

server日志:

binlog (复制,格式)，

errorlog(排查问题必备，有时需要ELK)、

general log(审计日志，跟slow log结合优化优先级，问题排查)，

slow log (30% sql 优化，重要数据来源，pt-query-digest,sql advise，web页面+慢SQL报表)。

5.7和8.0新特性

5.7

https://dev.mysql.com/doc/refman/5.7/en/mysql-nutshell.html

(1）数据库安全

引入ALTER USER命令，可用来修改用户密码，密码的过期策略，及锁定用户等表空间加密

mysql.user表中存储密码的字段从password修改为authentication\_string

(2)复制

MGR、多源复制、增强半同步(AFTER\_SYNC)、基于WRITESET的并行复制、在线开启GTID复制、在线设置复制过滤规则

(3)innodb

UNDO表空间的自动回收，InnoDB原生支持分区表，在此之前，是通过ha\_partition接口来实现的，引入了innodb\_deadlock\_detect选项，在高并发场景下，可使用该选项来关闭死锁检测

(4)系统表

优化了Performance Schema，其内存使用减少，集成了SYS数据库，简化了MySQL的管理及异常问题的定位，

(5)操作性

可设置SELECT操作的超时时长(max\_execution\_time)

在线修改Buffer pool的大小 低峰期操作

引入了查询改写插件(Query Rewrite Plugin)，可在服务端对查询进行改写mysql\_install\_db被mysqld --initialize代替，用来进行实例的初始化

可通过SHUTDOwN命令关闭MySQL实例

看阿里的数据库内核月报:

https://www.bookstack.cn/read/aliyun-rds-core/fa26cd1629beddd8.md

https://www.bookstack.cn/read/aliyun-rds-core/eb37d1fcc2b4ce76.md

https://www.bookstack.cn/read/aliyun-rds-core/a3a0f1d936417434.md

https://www.bookstack.cn/read/aliyun-rds-core/615266555fb07b04.md

mysql 8.0新特性

https://dev.mysql.com/doc/refman/8.0/en/mysql-nutshell.html

1、新增降序索引

order by A asc,B asc;order by A desc.B desc;order by A asc,B desc;

2、隐式索引

10个索引，索引优化，合并，干掉。drop index秒。后果??业务force index。月度跑，年度跑SQL，周跑,1天generi log.

ALTER TABLE tALTER INDEX i INVISIBLE;

3、全局参数持久化

修改配置动态生效。100-》500,vim my。cnf:过了，主库切了。

SET PERSIST max\_connections = 5B0;

auto-cnf

4、字符集默认:Latin1->utf8mb4

5、系统表引擎:MyISAM4->Innodb

6、DDL原子化，在MySQL 8.0之前，DDL操作是非原子型操作，在执行过程中遇到实例故障重启，可能导致DDL没有完成也没有回滚。

7、安全:身份认证方式由以前的mysql\_native\_password改为caching\_sha2\_password(就算同样的密码,加密后显示也不一样)

8、binlog保留时间:由expire\_logs\_days => binlog\_expire\_logs\_seconds

9、关闭查询缓存。

10、MGR优化

11、JSON、GIS加强

IO调度算法

权限管理

对于truncate drop 等权限如有人要求开放，需上级审批走流程

慢查询会把数据库打挂（oom)

1.一个大的慢查询-需要更多的内存作为缓存

2.并发多的慢查询可能会遇到锁等等因素,消耗整个系统的资源,包括内存,但还会有cpu

索引的优点

1.减少了服务器需要扫描的数据量

2.帮助服务器避免排序和临时表

3.随机IO变成顺序IO

提高查询速度，降低更新表的速度，如对表进行INSERT、UPDATE和DELETE。索引文件也要更新

建立索引占用磁盘空间(大表多个组合索引),生产不超过8个索引

数据列过滤因子过小(比如 1,0),索引没有什么效果

小表全表扫描更高效

隐式转换

PMM

MySQL常见面试题

DDL

Online DDl的实现原理

Online DDl的适用场景

pt-online-schema-change的实现原理

如何定位 DDL被阻塞的问题

高可用

常用的高可用方案及各自的优缺点

MHA怎么保证数据不丢失

MHA 的故障切换流程

MGR的实现原理

基于分布式协议来实现的

优化

执行计划关注哪些列

分页优化

如何优化多表关联查询

索引失效的常见场景

优化一般从哪几方面着手

InnoDB

MVCC

什么是元数据锁，如何定位

锁的类型

事务的隔离级别

复制

主从延迟的常见原因及解决方法?

主从数据不一致的常见原因及如何修复?

半同步复制的原理，AFTER\_SYNC和 AFTER\_COMMIT 的区别

规避数据丢失

并行复制的原理

基于库级别

基于组提交

LOGICAL\_CLOCK

Seconds\_Behind\_Master的计算方法及局限性

备份

mysqldump和XtraBackup加全局读锁的时机

mysqldump可以保证非InnoDB表的备份一致性吗?

mysqldump的实现原理

XtraBackup的实现原理

DDL对备份的影响

如何备份Binlog

如何进行指定位置点的恢复

监控

常用的监控方案

常用的监控指标

常用的操作系统命令

●ACID是什么意思?

ACID 是数据库管理系统（DBMS）在写入或更新数据过程中，为保证交易（transaction）是正确可靠的，所必须具备的四个特性：

原子性（Atomicity）：原子性是指一个事务中的所有操作要么全部成功，要么全部失败。也就是说，一个事务不能只完成部分操作，而必须完整地执行所有操作。

一致性（Consistency）：一致性是指事务必须使数据库从一个一致状态转换到另一个一致状态。也就是说，事务不能使数据库处于不一致的状态。

隔离性（Isolation）：隔离性是指多个事务并发执行时，每个事务都应该不受其他事务的影响。也就是说，每个事务都应该独立地执行，就像其他事务不存在一样。

持久性（Durability）：持久性是指事务一旦提交，其对数据库所做的更改就应该是永久性的，即使系统发生故障，这些更改也不会丢失。

ACID 特性是数据库系统的重要保证，可以确保数据库数据的准确性和可靠性。

一致性和原子性举个示例来说明

原子性：例如，在银行转账交易中，从原账户扣除金额和向目标账户添加金额这两个操作必须作为一个整体来执行，如果其中任何一个操作失败，则整个交易都应该失败。

一致性：例如，在库存管理系统中，如果一个商品的库存数量为 10，那么在任何情况下，该商品的库存数量都不应该为负数。

隔离性：例如，在多个用户同时访问数据库的情况下，每个用户都应该看到自己的数据，而不会看到其他用户的数据。

持久性：例如，即使数据库系统发生故障，数据库中的数据也应该能够恢复。

●ACID在MYSQL数据库里是怎么实现的

ACID在MySQL数据库里的实现

MySQL数据库通过undo log、redo log、锁机制等机制来实现ACID特性。

1. 原子性

原子性是指事务中的所有操作要么全部成功，要么全部失败。MySQL通过undo log来实现原子性。Undo log记录了事务开始前数据的状态，如果事务执行过程中发生错误，可以通过undo log将数据回滚到事务开始前的状态。

2. 一致性

一致性是指事务执行前后，数据库都处于一致的状态。MySQL通过redo log和锁机制来实现一致性。Redo log记录了事务对数据的修改，当事务提交时，redo log会将修改持久化到磁盘。锁机制可以防止多个事务同时修改同一份数据，从而保证数据的完整性。

3. 隔离性

隔离性是指多个事务并发执行时，彼此之间互不影响。MySQL通过锁机制来实现隔离性。MySQL支持多种锁机制，例如行锁、表锁、页锁等。不同的锁机制提供不同的隔离级别，可以满足不同应用的需求。

4. 持久性

持久性是指事务提交后，对数据的修改即使系统发生故障也不会丢失。MySQL通过redo log来实现持久性。Redo log会将事务对数据的修改持久化到磁盘，即使系统发生故障，也可以通过redo log恢复数据。

总结

MySQL通过undo log、redo log、锁机制等机制来实现ACID特性，从而保证数据库数据的完整性和一致性。

●ACID是四个特性是怎么理解的

ACID是数据库管理系统（DBMS）为了保证事务（transaction）的正确可靠性而必须具备的四个特性：

1. 原子性（Atomicity）

原子性是指事务中的所有操作要么全部成功，要么全部失败。事务是一个不可分割的整体，它对数据库的修改要么全部执行，要么全部不执行，不可能只执行其中一部分。

例如，在银行转账业务中，从A账户向B账户转账100元，这是一个事务。该事务包含两个操作：

从A账户扣除100元

向B账户增加100元

这两个操作必须同时成功，否则就会出现数据不一致的情况。例如，如果只执行了第一个操作，而第二个操作失败了，那么A账户就会少100元，而B账户不会增加100元。

2. 一致性（Consistency）

一致性是指事务执行前后，数据库都处于一致的状态。一致性指的是数据的完整性和逻辑正确性。

例如，在一个学生管理系统中，学生表中有一个字段记录学生的年龄。如果一个事务将某个学生的年龄从18岁改为20岁，那么该事务必须保证在事务执行前后，学生表的完整性和逻辑正确性都保持不变。

完整性：学生表的每个记录都必须包含所有必需的字段，并且每个字段的值都必须有效。

逻辑正确性：学生表的各个字段之间必须保持逻辑上的正确性。例如，学生的年龄不可能是负数。

3. 隔离性（Isolation）

隔离性是指多个事务并发执行时，彼此之间互不影响。隔离性可以防止多个事务同时修改同一份数据，从而保证数据的完整性。

例如，在银行转账业务中，如果两个用户同时向同一个账户转账，那么这两个事务必须相互隔离，否则就会出现数据不一致的情况。例如，如果第一个事务将100元转入该账户，而第二个事务将200元转入该账户，那么最终该账户的余额应该是300元。但是，如果这两个事务没有相互隔离，那么就有可能出现最终该账户的余额不是300元的情况。

4. 持久性（Durability）

持久性是指事务提交后，对数据的修改即使系统发生故障也不会丢失。持久性保证了事务的最终一致性。

例如，在银行转账业务中，如果一个事务将100元从A账户转入B账户，那么该事务必须保证即使系统发生故障，A账户的余额也会减少100元，B账户的余额也会增加100元。

总结

ACID是数据库管理系统的重要特性，它保证了数据库数据的完整性和一致性。

●ACID是四个特性在数据库里又和哪些特性是密切结合在一起的？

ACID 是数据库管理系统 (DBMS) 的四个重要特性，它们保证了数据库数据的完整性和一致性。在数据库中，ACID 与以下特性密切结合在一起：

1. 并发控制

并发控制是指多个事务同时执行时，DBMS 采取的措施来保证数据的完整性和一致性。ACID 的隔离性特性依赖于并发控制来实现。

2. 锁机制

锁机制是 DBMS 用于实现并发控制的一种重要手段。锁机制可以防止多个事务同时修改同一份数据，从而保证数据的完整性。ACID 的原子性和一致性特性依赖于锁机制来实现。

3. 日志

日志是 DBMS 用于记录数据库操作的一种重要手段。日志可以用于恢复数据库故障，保证数据的持久性。ACID 的持久性特性依赖于日志来实现。

4. 恢复

恢复是指 DBMS 从故障中恢复的能力。恢复可以保证数据库数据的完整性和一致性。ACID 的所有特性都依赖于恢复来实现。

以下是 ACID 与其他特性结合的具体示例:

原子性和锁机制: 当一个事务对数据进行修改时，DBMS 会对该数据加锁，防止其他事务同时修改该数据。这保证了事务的原子性。

一致性和并发控制: DBMS 通过并发控制机制来保证多个事务并发执行时，数据库数据的完整性和一致性。例如，DBMS 可以使用两阶段提交协议来保证事务的一致性。

隔离性和锁机制: DBMS 通过锁机制来实现隔离性。例如，DBMS 可以使用行锁来保证每个事务只能看到自己对数据的修改。

持久性和日志: DBMS 通过日志来记录事务对数据的修改，即使系统发生故障，也可以通过日志恢复数据，保证数据的持久性。

总结

ACID 是数据库管理系统的关键特性，它们与并发控制、锁机制、日志、恢复等特性密切结合在一起，共同保证了数据库数据的完整性和一致性。

●一条SQL查询语句是如何执行的？

一条SQL查询语句的执行过程可以分为以下几个步骤：

1. 连接

客户端应用程序向MySQL服务器发送连接请求，服务器验证用户身份并建立连接。

1. 查询缓存

如果查询缓存功能开启，服务器会先检查查询缓存中是否存在该查询语句的结果。如果存在，则直接返回结果。

3. 解析

服务器对查询语句进行词法分析和语法分析，生成语法树。

4. 优化

服务器对语法树进行优化，生成执行计划。执行计划决定了服务器如何访问数据。

5. 执行

服务器根据执行计划，访问数据并返回结果。

6. 关闭

客户端应用程序断开与服务器的连接。

以下是各个步骤的详细说明:

1. 连接

客户端应用程序使用TCP/IP协议连接到MySQL服务器。连接建立后，客户端应用程序可以向服务器发送SQL语句。

2. 查询缓存

查询缓存是一个存储查询语句结果的内存区域。如果查询缓存功能开启，服务器会在执行查询语句之前先检查查询缓存中是否存在该查询语句的结果。如果存在，则直接返回结果，无需再执行查询语句。

3. 解析

服务器对查询语句进行词法分析和语法分析，生成语法树。词法分析将查询语句分解为一个个的token，语法分析检查查询语句的语法是否正确。

4. 优化

服务器对语法树进行优化，生成执行计划。执行计划决定了服务器如何访问数据。优化器会考虑表的结构、索引的使用、查询条件等因素，选择最优的执行方案。

5. 执行

服务器根据执行计划，访问数据并返回结果。服务器会使用索引来快速找到所需的数据。

6. 关闭

客户端应用程序断开与服务器的连接。

主从延迟的常见原因及解决方法?

主从数据不一致的常见原因及如何修复?

增强半同步和原生半同步有什么区别？

半同步复制的原理，AFTER\_SYNC和 AFTER\_COMMIT 的区别

规避数据丢失

并行复制的原理

基于库级别

基于组提交

LOGICAL\_CLOCK

Seconds\_Behind\_Master的计算方法及局限性