**一、现代数据库怎么管理数据模式的？数据模型怎么影响系统性能？什么是结构化数据，半结构化数据，非结构化数据？**

数据库系统划分为三个层次，称为三级模式，分别为概念模式，外模式，内模式，都存于数据目录中，是数据目录的最基本内容，DBMS通过数据目录，管理和访问数据模式。

数据模型包含数据结构，数据操作，数据完整性约束，影响性能的因素主要是数据结构的复杂度和数据操作的可优化程度 结构化数据：数据整体结构化，通过数据模型描述 半结构化数据：单条记录内部的数据有结构，数据文件间无联系，整体无结构 非结构化数据：数据间，数据文件间都没有结构

**二、简述SQL和关系代数的联系和区别**

答：关系数据模型有三种等价的操作语言:关系代数、SQL、关系演算(元组关系演算和域关系演算)，它们的非过程化程度依次递增，主要应用领域也不同.SQL是关系数据库的标准语言，关系代数和关系演算是它的理论基础

联系：[关系代数](https://www.baidu.com/s?wd=%E5%85%B3%E7%B3%BB%E4%BB%A3%E6%95%B0&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d-Pjmvnvf3rj79rjfzuWcz0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHT4nWbvPj0zrH0Yrjbsnj0Y)是sql的理论基础。区别：sql是[结构化查询语言](https://www.baidu.com/s?wd=%E7%BB%93%E6%9E%84%E5%8C%96%E6%9F%A5%E8%AF%A2%E8%AF%AD%E8%A8%80&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d-Pjmvnvf3rj79rjfzuWcz0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHT4nWbvPj0zrH0Yrjbsnj0Y)，是数据库具体的技术标准和规范。[关系代数](https://www.baidu.com/s?wd=%E5%85%B3%E7%B3%BB%E4%BB%A3%E6%95%B0&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d-Pjmvnvf3rj79rjfzuWcz0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHT4nWbvPj0zrH0Yrjbsnj0Y)是数学理论。

**三、如事务不遵守ACID准则，则对数据库产生何种后果？为什么一般不涉及数据库的程序中不提ACID准则？**

若事务不遵守ACID准则，数据库中会产生脏数据，数据不一致、难以恢复等情况。不涉及数据库的程序不提ACID是因为它们很少需要满足数据库这样的高并发性和一致性需求。

一个事务是由应用程序中对数据库的一组操作序列组成的。如果事务不遵守ACID准则，则数据库中数据的完整性和一致性等就可能会因为事务的执行而遭到破坏。而一般不涉及数据库的程序不存在多用户之间数据的共享问题，所以在一般不涉及数据库的程序中不提ACID准则。

**六、相对层次、网状数据库系统，查询优化对关系型数据库系统更为重要。这句话对么？**

层次和网状数据库的数据模型使用指针表示属性之间的关系，这样的结构也固定了这两种数据库的查询路径，进行优化的空间有限。关系数据理论和关系数据查询语言提供了查询优化的空间。关系型数据库的抽象程度深，其查询语言一般是非过程语言，仅表达查询要求，不说明查询的过程。对于同一个查询语句，对应的关系代数等价的不同表达式的查询效率有着很大的差异；集合操作不同的执行规则和策略也对查询效率有着很大的影响；同时关系数据在物理存储形式和存取方式和路径上都没有限制。因此对于关系数据库系统来说，查询优化就更为重要，它对系统的性能有着很大的影响。

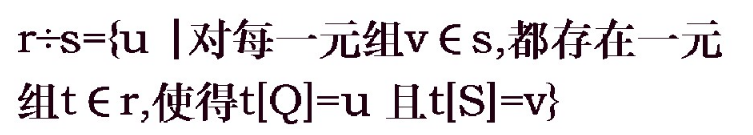
**七、假设运行记录与数据库的存储磁盘有独立失效模式，介质失效恢复时，对运行记录中上一检查点以前已提交事务应该redo否？为什么？**

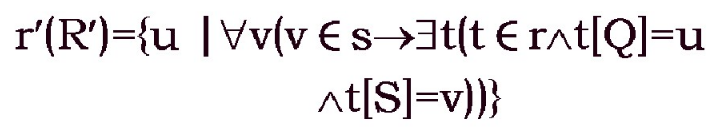
介质失效恢复时，对运行记录中上一检查点以前的已提交的事务应该redo。因为介质失效会丢失存储介质上的所有数据，恢复时需要在加载最近后备副本后，根据运行记录中的后像，重做最近后备副本以后提交的所有更新事务。因此最近一次检查点以前提交的事务也要做redo操作。

**八、从查询优化角度分析，为什么SQL查询where子句应尽量避免使用OR？**

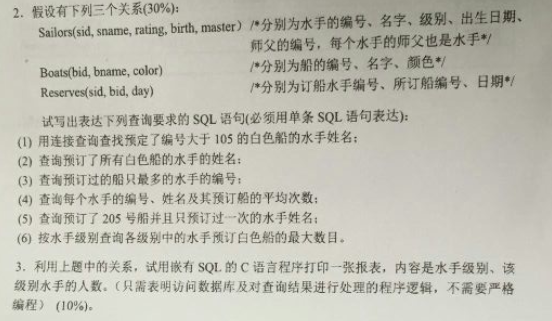
为了加快查询速度，优化查询效率，主要原则就是应尽量避免全表扫描，尽量在where及order by 涉及的列上建立索引。然而在where子句中使用or来当连接条件时，会导致引擎放弃使用索引而进行全表扫描，改用Union后，性能会大大提高。

**九、R/S的元组关系演算**





**十、嵌入式SQL**

****

2、

(1)SELECT S.sname FROM Sailors S, Boats B, Reserves R

WHERE S.sid=R.sid AND B.bid=R.bid AND R.bid>105 AND B.color=’white’

(2)SELECT S.sname FROM Sailors S

WHERE NOT EXISTS

(SELECT B.bid FROM Boats B WHERE NOT EXISTS

(SELECT R.bid FROM Reserves R WHERE R.bid=B.bid AND R.sid=S.sid))

(3) SELECT R.sid FROM Reserves R

GROUP BY R.sid HAVING COUNT(\*)=

(SELECT MAX(count) FROM

(SELECT COUNT(\*) AS count FROM Reserves R1 GROUP BY R1.sid))

(4) (平均次数是什么鬼)

SELECT R.sid, S.sname, avgcount=COUNT(R.bid)/(MAX(day)-MIN(day))

FROM Reserves R, Sailors S

WHERE S.sid=R.sid

GROUP BY R.sid

(5)

SELECT S.sname FROM Reserves R, Sailors S

WHERE S.sid=R.sid

GROUP BY R.bid, R.sid, S.sname

HAVING R.bid=205 AND COUNT(\*)=1

(6)

SELECT rating, MAX(count) FROM

(SELECT S.rating AS rating, R.sid, COUNT(\*) AS count

FROM Reserves R, Sailors S, Boats B

WHERE S.sid=R.sid AND B.bid=R.bid

GROUP S.rating, R.sid, B.color

HAVING B.color=’white’)

GROUP BY rating

3.// 声明SQL通信区

EXEC SQL INCLUDE SQLCA;

// 定义宿主变量

EXEC SQL BEGIN DECLARE SECTION;

float RATING;

int NUM;

EXEC SQL END DECLARE SECTION;

// 连接数据库

EXEC SQL CONNECT :uid IDENTIFIED BY :pwd;

// 使用游标读取数据

EXEC SQL DECLARE C1 CURSOR FOR

SELECT rating, COUNT(sid)

FROM Sailors

GROUP BY rating

EXEC SQL OPEN C1;

while (TRUE)

{

EXEC SQL FETCH C1 INTO :RATING, :NUM;

if (SQLCA.SQLCODE == 10)

break;

if (SQLCA.SQLCODE < 0)

break;

（循环打印数据）

……

}

EXEC SQL CLOSE C1;

**11.举例说明什么是数据模式？什么是数据模型？数据模式和数据模型的关系和区别是什么？**

答：数据模型是用来描述数据的一组概念和定义，以文件系统为例，他所用的数据模型包含文件，记录和字段。 数据模式是对某一类数据的结构，联系和约束的描述。例如学生信息记录可以定义为姓名、学号、性别等属性和关系的形式，就是数据模式。

数据模型是对数据进行概念层次上的描述，是计算机中用来描述现实世界的方法。而数据模式是对特定数据集中数据的结构和联系的描述。数据模式是用给定的数据模型对具体数据的描述。

**12.试简述索引对关系型数据库系统查询优化的意义；应该什么时候用索引？是不是什么时候用索引都能获得益处，试举例说明。**

答：关系型数据库查询优化的途径之一是依赖于存取路径的优化，而在关系型数据库中索引是用得最多的一种存取路径，建立合适的索引是实现查询优化的首要前提。索引提供了对数据的快速访问，根据操作建立合适的索引能够很大程度上优化存取路径，从而提高查询效率。创建唯一性索引，保证数据库表中每一行数据的唯一性等等

一般在数据量较大的时候适合使用索引，在需要频繁进行查询、排序或分组操作的属性性建立合适的索引。并非在任何情况下使用索引都能优化查询效率，应该根据实际操作需求考虑。例如对于小文件上的顺序查找使用堆文件的形式的开销不大，使用索引反而可能会增加开销。因为索引本身需要占用一定的存储空间，而且维护索引也需要一定的开销。对于增删改操作频繁的属性上建立索引可能会起到反作用。

**13.判断并发事务运行正确性的标准是什么？封锁法的基本思想是什么？它是怎样保证并发事务正确运行的？采用封锁法后必须解决的问题是什么？**

答： 判断并发事务运行正确性的标准是对并发事务运行的调度是否可串行化。封锁法的基本思想并发事务对同一数据对象操作前，向系统发出请求对操作对象进行加锁。事务对数据对象的加锁请求获准后，它便对该对象有了一定的控制，在这个事务释放它的锁之前，其他事务对该数据对象的锁请求不能获准，无法对其进行操作，从而避免了访问冲突，保证并发事务正确执行。采用封锁法后可能会出现活锁、死锁等问题，其中必须解决的问题是由于事物之间的循环等待导致的死锁问题。

**14. 简述sql和关系代数的关系和区别**

答：关系数据模型有三种等价的操作语言:关系代数、SQL、关系演算(元组关系演算和域关系演算)，它们的非过程化程度依次递增，主要应用领域也不同.SQL是关系数据库的标准语言，关系代数和关系演算是它的理论基础

联系：[关系代数](https://www.baidu.com/s?wd=%E5%85%B3%E7%B3%BB%E4%BB%A3%E6%95%B0&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d-Pjmvnvf3rj79rjfzuWcz0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHT4nWbvPj0zrH0Yrjbsnj0Y)是sql的理论基础。区别：sql是[结构化查询语言](https://www.baidu.com/s?wd=%E7%BB%93%E6%9E%84%E5%8C%96%E6%9F%A5%E8%AF%A2%E8%AF%AD%E8%A8%80&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d-Pjmvnvf3rj79rjfzuWcz0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHT4nWbvPj0zrH0Yrjbsnj0Y)，是数据库具体的技术标准和规范。[关系代数](https://www.baidu.com/s?wd=%E5%85%B3%E7%B3%BB%E4%BB%A3%E6%95%B0&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d-Pjmvnvf3rj79rjfzuWcz0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHT4nWbvPj0zrH0Yrjbsnj0Y)是数学理论。

**15.在DBMS中，通常采用多级数据模式，例如概念模式、外模式和内模式，简述数据库系统中的多级数据模式对数据独立性的影响。（8％）**

答：外模式、[概念模式](https://www.baidu.com/s?wd=%E6%A6%82%E5%BF%B5%E6%A8%A1%E5%BC%8F&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Ykn1bkujDLnHD1m1NBuAc10ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHmYnjn4PWnLPHDLrjDYP1mzn0)、内模式，有效地组织、管理数据，提高了数据库的逻辑独立性和物理独立性。用户级对应外模式，概念级对应[概念模式](https://www.baidu.com/s?wd=%E6%A6%82%E5%BF%B5%E6%A8%A1%E5%BC%8F&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Ykn1bkujDLnHD1m1NBuAc10ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHmYnjn4PWnLPHDLrjDYP1mzn0)，物理级对应内模式，使不同级别的用户对数据库形成不同的视图。

数据独立性分为逻辑独立性和物理独立性.物理独立性是指内模式改变时,概念模式保持不变,逻辑独立是指概念模式改变时,外模式不变,从而使应用程序保持不变.当内模式改变时,DBMS只要通过改变概念模式到内模式映射,即可使概念模式保持不变,从而实现了数据的物理独立性.而逻辑独立的实现正好相反.

**16.某宾馆需要建立一个住房管理系统，需求分析结构如下：**

①一个房间有多个床位，每个房间有房间号（如201、202等）、收费标准、床位数目等信息；

②客人信息包括身份证号码、姓名、性别和地址等信息；

③对每位客人的每次住宿，需记录其入住日期、退房日期和预付款信息。

根据需求分析，设计的关系数据模型如下：

房间（房间号，收费标准，床位数目）

客人（身份证号，姓名，性别，出生日期，地址）

住宿（房间号，身份证号，入住日期，退房日期，预付款额）

假设订房人即入住人，请给出上述三个关系模式存在的所有主键和外键。（外键4分，主键8分）

针对上述应用场景下的三个关系模式，写出表达下列查询要求的SQL语句（必须用单条SQL语句表达）：

1.用连接查询查找预定了编号为210号房的客人姓名； （6分）

2.查询预定过所有房间的客人的姓名； （8分）

3.查询2016年1月份只有一人预定的房间号及客人姓名； （8分）

4.对累计订房次数超过60次的客人，查询每位客人单月预订房间的最大次数及该最大次数对应的年份与月份。（8分）

注：SQL中获取Datatime型数据年份、月份的函数分别为getYear(…), getMonth(…)。

**答案：**

①主键：房间号②主键：身份证号或（姓名，出生日期，地址）

1. 主键：（房间号，身份证，入住日期）；外键：房间号，身份证号

1.SELECT 客人.姓名 FROM 客人,住宿 WHERE 客人.身份证=住宿.身份证 AND 住宿.房间号=210;

2.SELECT 姓名 FROM 客人 WHERE 身份证 IN

(SELECT 身份证 FROM 住宿 GROUP BY 身份证 HAVING COUNT(DISTINCT(房间号))=(SELECT COUNT(\*) FROM 房间));

3.客人.姓名, 住宿.房间号FROM `住宿` , 客人WHERE 客人.身份证 [=](http://localhost:8080/phpmyadmin/url.php?url=http%3A%2F%2Fdev.mysql.com%2Fdoc%2Frefman%2F5.5%2Fen%2Fcomparison-operators.html%23operator_equal&token=aac4d2e8ba3674251f8479d2c89f2a4b) 住宿.身份证 AND getYear(入住日期) =2016[AND](http://localhost:8080/phpmyadmin/url.php?url=http%3A%2F%2Fdev.mysql.com%2Fdoc%2Frefman%2F5.5%2Fen%2Flogical-operators.html%23operator_and&token=aac4d2e8ba3674251f8479d2c89f2a4b) get[Month](http://localhost:8080/phpmyadmin/url.php?url=http%3A%2F%2Fdev.mysql.com%2Fdoc%2Frefman%2F5.5%2Fen%2Fdate-and-time-functions.html%23function_month&token=aac4d2e8ba3674251f8479d2c89f2a4b)(入住日期) [=](http://localhost:8080/phpmyadmin/url.php?url=http%3A%2F%2Fdev.mysql.com%2Fdoc%2Frefman%2F5.5%2Fen%2Fcomparison-operators.html%23operator_equal&token=aac4d2e8ba3674251f8479d2c89f2a4b)1GROUP BY 房间号HAVING [COUNT](http://localhost:8080/phpmyadmin/url.php?url=http%3A%2F%2Fdev.mysql.com%2Fdoc%2Frefman%2F5.5%2Fen%2Fgroup-by-functions.html%23function_count&token=aac4d2e8ba3674251f8479d2c89f2a4b)(\*) [=](http://localhost:8080/phpmyadmin/url.php?url=http%3A%2F%2Fdev.mysql.com%2Fdoc%2Frefman%2F5.5%2Fen%2Fcomparison-operators.html%23operator_equal&token=aac4d2e8ba3674251f8479d2c89f2a4b)1;

4.SELECT MAX(times),y,m FROM

(SELECT COUNT(\*)times,身份证id,YEAR(入住日期) y,MONTH(入住日期) m FROM 住宿WHERE 身份证IN

(SELECT 身份证FROM 住宿GROUP BY 身份证HAVING COUNT(\*)>60) GROUP BY身份证,YEAR(入住日期),MONTH(入住日期)

) AS tGROUP BY id;

**17.、共有3个关系模式（共 40分）**

**客人（身份证号，籍贯，……）**

**房间（房间号，……）**

**住宿（身份证号，房间号，入住时间，退房时间，预定金额）**

**(1)、写出关系模式中所有的主键和外键（10分）**

房间：主键是房间号，无外键

客人：主键身份证号，无外键

住宿：主键身份证号、入住日期。外键房间号身份证号

**(2)、用单条SQL语句写出下列查询：**

**A、用连接查询所有住过301房间的南京籍客人（10分）**

Select gname

fromguest,stay

whereguest.gid=stay.gid

andstay.rid=301

and place=‘南京’;

**B、预定过所有房间的客人（10分）**

Select gname

From guest

Where gidin(select gid from stay

Group by gid

Having count(rid)=(select count(\*) from room)

);

**C、2014年入住次数最多的客人姓名和入住次数（10分）**

Select gname,c

From guest,(select gid ,count(\*) as c

From stay

Where guest.gid=a.gid;（有疑问）

Group by gid

Having getYear(indata)=2014 And c=max(c))a

**18.**

**将SUX锁的矩阵改为**

|  |  |  |  |
| --- | --- | --- | --- |
| **待加已有** | **S** | **U** | **X** |
| **S** | **Y** | **Y** | **N** |
| **U** | **Y** | **Y** | **N** |
| **X** | **N** | **Y** | **N** |

**则如下过程能否正常运行？（18分）**

**(1)、都是读操作**

能，没有事务进行写操作，所以该事务集一定可串行化，不会产生丢失更新，读脏数据等问题

**(2)、都是写操作**

能，事务只会对数据加x锁，对于加了x锁的数据其他事务不能加任和锁，所以不会产生丢失更新等问题，但易发生循环等待，产生死锁

**(3)、一部分读操作，一部分写操作**

不能，假设t1对a加s锁，t2想要对a进行写操作，对a加u锁，当a解除s锁时，t3想要对a进行写操作，对a加x锁，于是t2的操作延后或无法执行，导致丢失更新。

**数据库系统中，若事务并发执行的调度是可以串行化的，即认为该并发结果是正确的，why?**

事务并发执行的调度是可串行化的，也就是说对于该事务并发执行的调度与该事务的串行调度等价；对于串行调度，各事务的操作没有交叉，没有互相干扰，因此不会产生并发执行时的冲突问题，因此与之等价的事务并发调度也不会产生冲突，即并发结果是正确的。

**稠密索引是否一定能够提高针对索引属性查询的效率，为什么？**

稠密索引不一定能提高针对索引属性的查询效率。

① 如果是查询小文件中的全部或相当多的记录时，使用索引并不能提高查询效率，反而会因为索引增加开销；② 如果稠密索引为次索引，但不是簇集索引，也就是说一个键值对应的多条记录分散在不同的物理块中；当一个键值对应的记录较多时，取这些记录时访问物理课的I/O开反而会降低查询的效率。

student(*sid*, sname, sex, age, year, gpa) dept(*dname*, numphds) prof(*pname*, dname)

course(*cno*, cname, *dname*) major(*dname, sid*) section(*dname, cno, sectno*, pname)

enroll(*sid*, grade, *dname, cno, sectno*)