数据库复习资料

黄新宇 著

目录

一、数据库基础知识

1. 什么是数据库？

答：是按照数据结构来组织、存储和管理数据的仓库。

2. 数据库有哪些种类？

答：数据库通常分为层次式数据库、网络式数据库和关系式数据库三种。

3. 常见的数据库？

答：大型数据库有：Oracle、Sybase、DB2、SQL server；小型数据库有：Access、MySQL、BD2 等。

4. 什么是 SQL？

答：SQL 是结构化查询语言（Structured Query Language）的缩写。

5. SQL 包含哪些部分？

答：(1) 数据定义：这一部分又称为“SQL DDL”，定义数据库的逻辑结构，包括定义数据库、基本表、视图和索引 4 部分。

(2) 数据操纵：这一部分又称为“SQL DML”，其中包括数据查询和数据更新两大类操作，其中数据更新又包括插入、删除和更新三种操作。

(3) 数据控制：对用户访问数据的控制有基本表和视图的授权、完整性规则的描述，事务控制语句等。

(4) 嵌入式 SQL 语言的使用规定：规定 SQL 语句在宿主语言的程序中使用的规则。

6. 关系数据模型的操纵与完整性的约束各有哪些？

答：关系数据模型的操纵主要包括查询、插入、删除和更新数据，这些操作必须满足关系的完整性约束条件。关系的完整性约束条件包括三大类：实体完整性、参照完整性和用户定义的完整性。

7. 关系数据模型的优缺点各有哪些？

答：优点：(1)关系模型与非关系模型不同，它是建立在严格的数据概念基础上的；(2)关系模型的概念单一；(3)关系模型的存取路径对用户透明，从而具有更高的数据独立性，更好的安全保密性，也简化了程序员的工作和数据库开发设计的工作。

缺点：由于存取路径对用户透明，查询效率往往不如非关系数据模型。因此，为了提高性能，必须对特定的查询请求进行优化，增加了开发数据库管理系统的负担。

8. 关系模型完整性规则包括哪几类？

答：关系模型的完整性规则是对关系的某种约束条件，关系模型中可以有三类完整性约束；实体完整性、参照完整性和用户定义的完整性。其中实体完整性和参照完整性是关系模型必须满足的完整性约束条件，被称作关系的两个不变性，应该由关系系统自动支持。

9. 什么是索引？

答：索引是对数据库表中一列或多列的值进行排序的一种结构，使用索引可快速访问数据库表中的特定信息。

10.SQL 中建立索引的意义？

答：建立索引是加快查询速度的有效手段。SQL 语言支持用户根据应用环境的需要，在基本表上建立一个或多个索引，以提供多种存取路径，加快查找速度。

11.什么是视图？

答：视图是一种虚拟的表，具有和物理表相同的功能。可以对视图进行增，改，查，操作，视图通常是有一个表或者多个表的行或列的子集。对视图的修改不影响基本表。它使得我们获取数据更容易，相比多表查询。

12.什么是事务？

答：数据库事务是指作为单个逻辑工作单元执行的一系列操作，要么完全地执行，要么完全地不执行。 事务处理可以确保除非事务性单元内的所有操作都成功完成，否则不会永久更新面向数据的资源。通过将一组相关操作组合为一个要么全部成功要么全部失败的单元，可以简化错误恢复并使应用程序更加可靠。一个逻辑工作单元要成为事务，必须满足所谓的 ACID（原子性、一致性、隔离性和持久性）属性。事务是数据库运行中的一个逻辑工作单位，由 DBMS 中的事务管理子系统负责事务的处理。

13.事务的属性有哪些？

答：(1) 原子性（Atomic）事务必须是原子工作单元；对于其数据修改，要么全都执行，要么全都不执行。通常，与某个事务关联的操作具有共同的目标，并且是相互依赖的。如果系统只执行这些操作的一个子集，则可能会破坏事务的总体目标。原子性消除了系统处理操作子集的可能性。

(2) 一致性（Consistent）事务在完成时，必须使所有的数据都保持一致状态。在相关数据库中，所有规则都必须应用于事务的修改，以保持所有数据的完整性。事务结束时，所有的内部数据结构（如 B 树索引或双向链表）都必须是正确的。某些维护一致性的责任由应用程序开发人员承担，他们必须确保应用程序已强制所有已知的完整性约束。例如，当开发用于转帐的应用程序时，应避免在转帐过程中任意移动小数点。

(3) 隔离性（Insulation）由并发事务所作的修改必须与任何其它并发事务所作的修改隔离。事务查看数据时数据所处的状态，要么是另一并发事务修改它之前的状态，要么是另一事务修改它之后的状态，事务不会查看中间状态的数据。这称为隔离性，因为它能够重新装载起始数据，并且重播一系列事务，以使数据结束时的状态与原始事务执行的状态相同。当事务可序列化时将获得最高的隔离级别。在此级别上，从一组可并行执行的事务获得的结果与通过连续运行每个事务所获得的结果相同。由于高度隔离会限制可并行执行的事务数，所以一些应用程序降低隔离级别以换取更大的吞吐量。

(4) 持久性（Duration）事务完成之后，它对于系统的影响是永久性的。该修改即使出现致命的系统故障也将一直保持。

14.什么是主键？

答：主键是数据表中的一个或多个字段，它的值用于唯一地标识表中的某一条记录。

15.什么是外键？

答：如果数据表中存在公共关键字，而这个公共关键字在一个数据表中是主键，那么这个公共关键字被称为另一个数据表的外键。外键表示了两个数据表之间的相关联系。

16.MySQL 的数据库引擎的类型

答：在默认情况下，MYSQL 支持三个引擎：ISAM、MYISAM 和 HEAP。另外两种类型 INNODB 和 BERKLEY（BDB），也常常可以使用。

(1) ISAM

ISAM 是一个定义明确且历经时间考验的数据表格管理方法，它在设计之时就考虑到数据库被查询的次数要远大于更新的次数。因此，ISAM 执行读取操作的速度很快，而且不占用大量的内存和存储资源。ISAM 的两个主要不足之处在于，它不支持事务处理，也不能够容错：如果硬盘崩，那么数据文件无法恢复。如果把 ISAM 用在

关键任务应用程序里，那就必须经常备份你所有的实时数据，通过其复制特性，MYSQL 能够支持这样的备份应用程序。

(2) MYISAM

MYISAM 是 MYSQL 的 ISAM 扩展格式和缺省的数据库引擎。除了提供 ISAM 里所没有的索引和字段管理的大量功能，MYISAM 还使用一种表格锁定的机制，来优化多个并发的读写操作。其代价是你需要经常运行 OPTIMIZE TABLE 命令，来恢复被更新机制所浪费的空间。MYISAM 还有一些有用的扩展，例如用来修复数据库文件的 MYISAMCHK 工具和用来恢复浪费空间的 MYISAMPACK 工具。

MYISAM 强调了快速读取操作，这可能就是为什么 MYSQL 受到了 WEB 开发如此青睐的主要原因：在 WEB 开发中你所进行的大量数据操作都是读取操作。所以，大多数虚拟主机提供商和 INTERNET 平台提供商只允许使用 MYISAM 格式。

(3) HEAP

HEAP 允许只驻留在内存里的临时表格。驻留在内存里让 HEAP 要比 ISAM 和 MYISAM 都快，但是它所管理的数据是不稳定的，而且如果在关机之前没有进行保存，那么所有的数据都会丢失。在数据行被删除的时候，HEAP 也不会浪费大量的空间。HEAP 表格在你需要使用 SELECT 表达式来选择和操控数据的时候非常有用。要记住，在用完表格之后就删除表格。

(4) INNODB 和 BERKLEYDB

INNODB 和 BERKLEYDB（BDB）数据库引擎都是造就 MYSQL 灵活性的技术的直接产品，这项技术就是 MYSQL++ API。在使用 MYSQL 的时候，你所面对的每一个挑战几乎都源于 ISAM 和 MYISAM 数据库引擎不支持事务处理也不支持外来键。尽管要比 ISAM 和 MYISAM 引擎慢很多，但是 INNODB 和 BDB 包括了对事务处理和外来键的支持，这两点都是前两个引擎所没有的。如前所述，如果你的设计需要这些特性中的一者或者两者，那你就要被迫使用后两个引擎中的一个了。

17.数据库中的锁是什么？

答：业务逻辑的实现过程中，往往需要保证数据访问的排他性。如在金融系统的日终结算处理中，我们希望针对某个 cut-off 时间点的数据进行处理，而不希望在结算进行过程中

（可能是几秒种，也可能是几个小时），数据再发生变化。此时，我们就需要通过一些机

制来保证这些数据在某个操作过程中不会被外界修改，这样的机制，给我们选定的目标数据上锁，使其无法被其他程序修改。

18.乐观锁和悲观锁是什么？

答：(1) 悲观锁

悲观锁，正如其名，它指的是对数据被外界（包括本系统当前的其他事务，以及来自

外部系统的事务处理）修改持保守态度，因此，在整个数据处理过程中，将数据处于锁定

状态。悲观锁的实现，往往依靠数据库提供的锁机制（也只有数据库层提供的锁机制才能

真正保证数据访问的排他性，否则，即使在本系统中实现了加锁机制，也无法保证外部系

统不会修改数据）。

(2) 乐观锁

相对悲观锁而言，乐观锁机制采取了更加宽松的加锁机制。悲观锁大多数情况下依靠数据库的锁机制实现，以保证操作最大程度的独占性。但随之而来的就是数据库性能的大量开销，特别是对长事务而言，这样的开销往往无法承受。如一个金融系统，当某个操作员读取用户的数据，并在读出的用户数据的基础上进行修改时（如更改用户帐户余额），如果采用悲观锁机制，也就意味着整个操作过程中（从操作员读出数据、开始修改直至提交修改结果的全过程，甚至还包括操作员中途去煮咖啡的时间），数据库记录始终处于加锁状态，可以想见，如果面对几百上千个并发，这样的情况将导致怎样的后果。

乐观锁机制在一定程度上解决了这个问题。乐观锁，大多是基于数据版本(Version)记录机制实现。何谓数据版本？即为数据增加一个版本标识，在基于数据库表的版本解决方案中，一般是通过为数据库表增加一个“version” 字段来实现。

19.什么是第一范式、第二范式、第三范式？

答：第一范式：数据库表的每一列都是不可分割的原子数据项，而不能是集合，数组，记录等非原子数据项。如果实体中的某个属性有多个值时，必须拆分为不同的属性（1NF）无重复的列

第二范式：满足第一范式前提，当存在多个主键的时候，才会发生不符合第二范式的情况。（2NF）属性完全依赖于主键

第三范式：满足第二范式前提，如果某一属性依赖于其他非主键属性，而其他非主键属性又依赖于主键，那么这个属性就是间接依赖于主键，这被称作传递依赖于主属性。（3NF）属性不依赖于其它非主属性

二、常用 SQL 语句

选取 MySQL 数据库为例介绍。

1. 连接数据库

mysql –h 127.0.0.1 –u root –p

然后按提示输入数据库密码。

2. 创建数据库

CREATE DATABASE database_name

以 UTF-8 编码：

CREATE DATABASE database_name DEFAULT CHARSET utf8 COLLATE utf8_general_ci;

以 GBK 编码：

create database database_name DEFAULT CHARACTER SET gbk COLLATE gbk_chinese_ci;

3. 删除数据库

drop database database_name

4. 修改数据库名称

sp_renamedb 'old_name', 'new_name'

5. 创建数据表

create table table_name(column1 type1 [not null] [primary key],column2 type2 [not null],..)

使用旧表创建新表：

create table table_new like table_old

create table table_new as select column1,column2… from table_old definition only

6. 删除数据表

drop table table_name

7. 常用简单的 SQL 语句

选择：select * from table1 where 范围

插入：insert into table1(column1,column2) values(value1,value2)

删除：delete from table1 where 范围

更新：update table1 set column1=value1 where 范围

查找：select * from table1 where column1 like ‘%value1%’

排序：select * from table1 order by column1,column2 [desc]

总数：select count as totalcount from table1

求和：select sum(column1) as sumvalue from table1

平均：select avg(column1) as avgvalue from table1

最大：select max(column1) as maxvalue from table1

最小：select min(column1) as minvalue from table1

8. 添加删除主键

添加主键： Alter table tablename add primary key(column)

说明：删除主键： Alter table tablename drop primary key(column)

9. 创建删除索引

创建索引：create [unique] index idxname on tabname(col….)

删除索引：drop index idxname

注意：索引是不可更改的，想更改必须删除重新建。

10.创建、删除视图

创建视图：create view viewname as select ...

删除视图：drop view viewname

11.常用高级查询

(1) UNION 运算符

UNION 运算符通过组合其他两个结果表（例如 TABLE1 和 TABLE2）并消去表中任何重复行而派生出一个结果表。当 ALL 随 UNION 一起使用时（即 UNION ALL），不消除重复行。两种情况下，派生表的每一行不是来自 TABLE1 就是来自 TABLE2。

(2) EXCEPT 运算符

EXCEPT 运算符通过包括所有在 TABLE1 中但不在 TABLE2 中的行并消除所有重复行而派生出一个结果表。当 ALL 随 EXCEPT 一起使用时 (EXCEPT ALL)，不消除重复行。

(3) INTERSECT 运算符

INTERSECT 运算符通过只包括 TABLE1 和 TABLE2 中都有的行并消除所有重复行而派生出一个结果表。当 ALL 随 INTERSECT 一起使用时 (INTERSECT ALL)，不消除重复行。

注：使用运算词的几个查询结果行必须是一致的。

12.使用外连接

(1) 左外连接（左连接）：结果集几包括连接表的匹配行，也包括左连接表的所有行。

select table1.column1, table1. column 2, table1.column3, table2.column3, table2.column4, table2.column5 from table1 LEFT JOIN table2 ON table1.column3 = table3.column3;

(2) 右外连接(右连接)：结果集既包括连接表的匹配连接行，也包括右连接表的所有行。

select table1.column1, table1. column 2, table1.column3, table2.column3, table2.column4, table2.column5 from table1 RIGHT JOIN table2 ON table1.column3 = table3.column3;

(3) 全外连接：不仅包括符号连接表的匹配行，还包括两个连接表中的所有记录。

select table1.column1, table1. column 2, table1.column3, table2.column3, table2.column4, table2.column5 from table1 FULL JOIN table2 ON table1.column3 = table3.column3;

(4) 内连接：

select table1.column1, table1. column 2, table1.column3, table2.column3, table2.column4, table2.column5 from table1 INNER JOIN table2 ON table1.column3 = table3.column3;

13.分组 Group by

“Group By”从字面意义上理解就是根据“By”指定的规则对数据进行分组，所谓的分组就是将一个“数据集”划分成若干个“小区域”，然后针对若干个“小区域”进行数据处理。

```
select column1, sum(column2) as column3 from table_name group by column1;
```