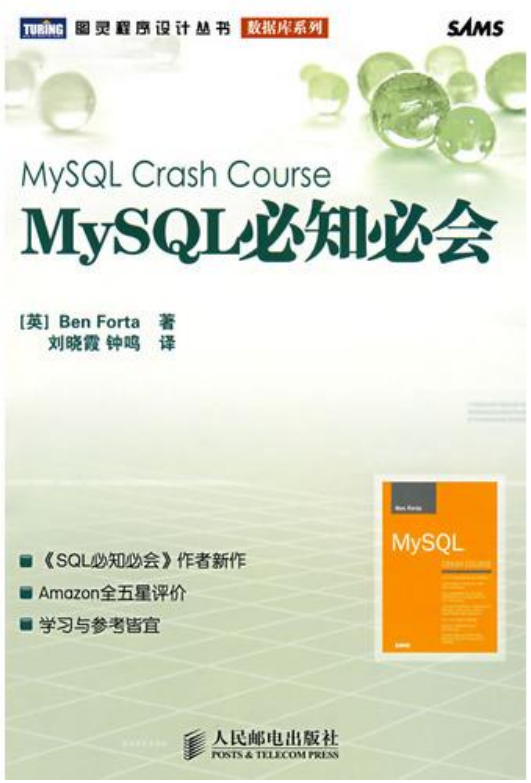
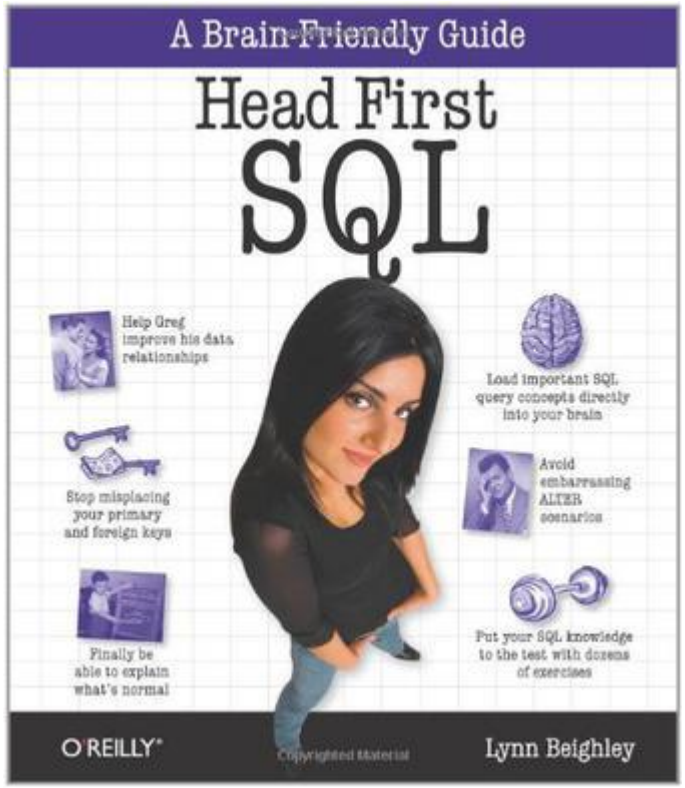
# MySQL advanced

凯盛软件

# 推荐书籍

凯盛软件

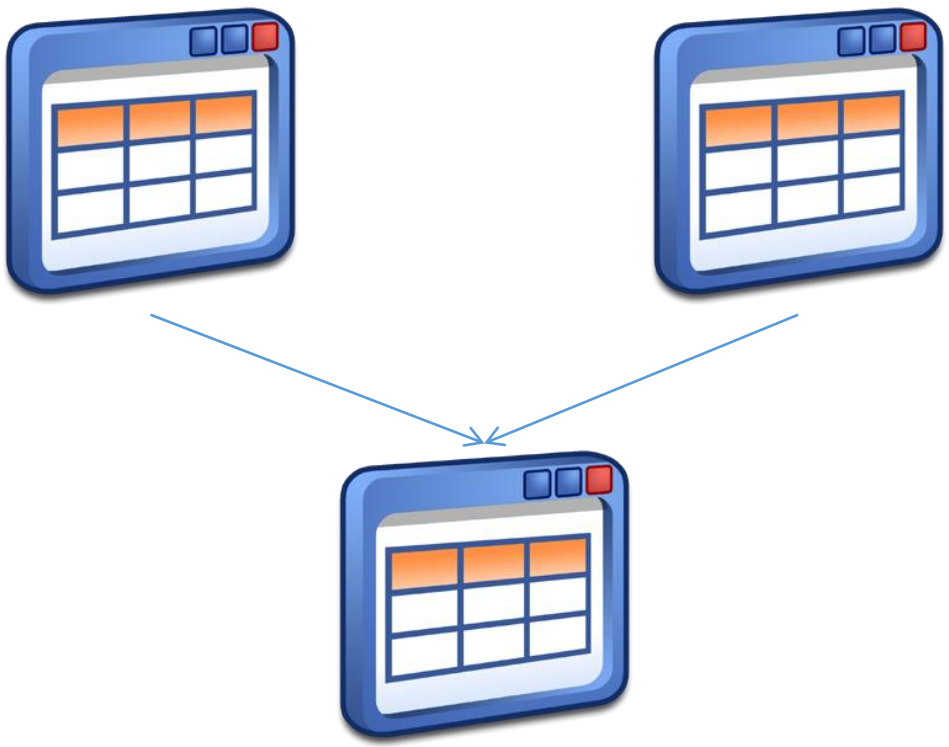

TURING 图灵程序设计丛书 数据库系列
SAMS
MySQL Crash Course
MySQL必知必会
[英] Ben Forta 著
刘晓霞 钟鸣 译
《SQL必知必会》作者新作
Amazon全五星评价
学习与参考皆宜
MySQL
人民邮电出版社
POSTS & TELECOM PRESS


A Brain-Friendly Guide
Head First
SQL
Help Greg improve his data relationships
Load important SQL query concepts directly into your brain
Stop misplacing your primary and foreign keys
Avoid embarrassing ALTER scenarios
Finally be able to explain what's normal
Put your SQL knowledge to the test with dozens of exercises
O'REILLY
Lynn Beighley

# 视图



视图是虚拟的表，与包含数据的表不一样，视图只包含使用时动态检索数据的查询，而自身不包含任何数据



# 为什么要视图

凯盛软件

- 重用SQL语句
- 简化复杂的SQL操作
- 使用表的组成部分而不是整个表
- 保护数据。可以给用户授予表的特定部分的访问权限而不是整个表的访问权限
- 更改数据格式和表示

# 视图的规则与限制

凯盛软件

- 与表一样，视图必须唯一命名
- 在一个数据库中，可以创建的视图数目没有限制
- 视图可以嵌套，即可以利用从其他视图中查询出来的数据构建新的视图
- Order by可以用在视图中，但如果从该视图检索数据的select中也含有order by,那么该视图中的Order by将会被覆盖
- 视图不能索引，也不能有关联的触发器或默认值
- 视图和表可以一起使用，例如编写一条联接表和视图的查询

# 创建视图

凯盛软件

```
CREATE VIEW v_customers AS

SELECT customers.cust_name,customers.cust_id,COUNT(orders.order_num) FROM customers

LEFT OUTER JOIN orders ON customers.cust_id = orders.cust_id

GROUP BY customers.cust_id
```

# 使用视图



```
SELECT * FROM v_customers
```

# 删除视图



DROP VIEW viewname

# 视图的更新

凯盛软件

- 对视图进行insert update delete将会影响其基表，因为视图中不包含任何数据
- 不是所有视图都可以更新
    - 含有分组（group by 和 having）
    - 联接查询
    - 子查询
    - 聚合函数
    - DISTINCT
- 不是迫不得已，不要对视图进行更新操作，因为效率低。视图主要用于查询

# 存储过程

凯盛软件

存储过程是一组为了完成某个特定功能而编写的并运行在数据库端的SQL程序集。

**存储过程优点**

1.通过把处理封装在简单易用的单元中，简化复杂的操作

2.提高性能。使用存储过程比使用单独的SQL语句要快

3.安全，调用者只需要调用指定的存储过程即可，而不用关心存储过程的内容

**存储过程缺点**

1.编写复杂

2.如果没有相应的权限，你将无法创建存储过程

# 创建存储过程



```
DELIMITER //
CREATE PROCEDURE productavg()
BEGIN
SELECT AVG(prod_price) AS avgprice FROM products;
END//
DELIMITER ;
```

# 调用存储过程



```
CALL productavg();
```

# 带返回值的存储过程

凯盛软件

```
DELIMITER //
CREATE PROCEDURE productpricing(
OUT pl DECIMAL(8,2), /*OUT代表是返回值参数 pl是变量名称 DECIMAL是数据类型*/
OUT ph DECIMAL(8,2),
OUT pa DECIMAL(8,2)
)
BEGIN
  SELECT MIN(prod_price) INTO pl FROM products; /*INTO将值付给变量Pl*/
  SELECT MAX(prod_price) INTO ph FROM products;
  SELECT AVG(prod_price) INTO pa FROM products;
END//
DELIMITER;
```

# 调用带返回值的存储过程

凯盛软件

```
CALL productpricing(@pricelow,@priceheight,@priceavg)

SELECT @pricelow,@priceheight,@priceavg
```

# 带传入参数的存储过程

凯盛软件

```
DELIMITER //
CREATE PROCEDURE ordertotal(
IN ordernum INT,/*IN代表传入参数*/
OUT total DECIMAL(8,2)
)
BEGIN
  SELECT SUM(item_price * quantity) INTO total FROM orderitems WHERE order_num = ordernum ;
END//
DELIMITER;
```

# 调用带传入参数的存储过程

凯盛软件

```
CALL ordertotal(20005,@total)

SELECT @total
```

# 删除存储过程



```
DROP PROCEDURE productavg；
```

# Java调用存储过程

凯盛软件

https://gist.github.com/fankay/5255473

# 触发器



触发器是MySQL响应insert、update、delete语句时自动执行的一条SQL语句，只有表支持触发器，视图不支持。

# 触发器需要的信息

凯盛软件

1.唯一的触发器名称（一个表中触发器名称唯一，而不是在一个数据库中唯一）

2.触发器关联的表

3.触发器应该响应的事件（insert？update？delete？）

4.触发器何时执行（处理之前或之后）

5.一个表的一个事件最多只能有两个触发器（处理之前和处理之后），所以一个表最多有6个触发器

6.如果响应之前的触发器执行失败，响应则不会执行；响应之前的触发器或响应执行失败，那么响应之后的触发器则不会执行

# insert触发器

凯盛软件

```
CREATE TRIGGER tr_insert_tableA
AFTER INSERT ON t_tableA
FOR EACH ROW
INSERT INTO t_tableB(val) VALUES(new.val);
```

# insert触发器

凯盛软件

1.在Insert触发器内，可引用一个名为NEW的虚拟表，访问被插入的行

2.在before insert触发器中，NEW中的值也可以被更新（运行更改被插入的值）

3.对于自动增长列，NEW在insert执行之前的值为0，在执行之后是新的自动生成的值

```
/*获取刚刚插入的自动生成的主键值*/
CREATE TRIGGER t_insert_pk_tableA
AFTER INSERT ON t_tableA
FOR EACH ROW SELECT new.id INTO @id;
/*测试触发器*/
INSERT INTO t_tableA(val) VALUES('abc');
SELECT @id;
```

# delete触发器

凯盛软件

```
DELIMITER //
CREATE TRIGGER t_delete_tableA
AFTER DELETE ON t_tableA /*DELETE后置触发器*/
FOR EACH ROW
BEGIN
  INSERT INTO t_tableB (val) VALUES(old.val);
END//
DELIMITER;

/*测试触发器*/
DELETE FROM t_tableA WHERE id = 2
```

# delete触发器

凯盛软件

1.在DELETE触发器代码中，可以引用一个OLD的虚拟表，访问被删除的行

2.OLD表中的值全部是只读的，不能更新

# update触发器

凯盛软件

```
/*将a表中修改后的名字都改为大写*/
DELIMITER //
CREATE TRIGGER t_update_tableA
BEFORE UPDATE ON t_tableA
FOR EACH ROW
BEGIN
  SET new.val = UPPER(new.val);
END//
DELIMITER;
/*测试触发器*/
UPDATE t_tableA SET val = 'xyz' WHERE id = 1;
SELECT * FROM t_tableA;
```

# update触发器

凯盛软件

1.在UPDATE触发器代码中，可以引用一个名为OLD的虚拟表访问以前的值，引用一个名为NEW的表访问新更新的值

2.在befor update触发器中，new表中的值允许被更新（允许更改将要用于update语句中的值）

3.OLD表中的值都是只读的，不能更行

# 删除触发器



```
DROP TRIGGER tr_insert_tableA
```

# 事务

凯盛软件

- 在MySQL中只有使用了Innodb数据库引擎的数据库或表才支持事务
- 事务处理可以用来维护数据库的完整性，保证成批的MySQL操作要么完全执行，要么完全不执行
- 事务用来管理insert、update、delete语句

# 关于事务的术语

凯盛软件

- 事务(transaction)：指一组SQL语句
- 回滚(rollback)：指撤销指定SQL语句的过程
- 提交(commit)：指将未存储的SQL语句结果写入到数据库中
- 保留点(savepoint)：指事务处理中设置的临时占位符，可以对它进行回滚

- 事务处理的关键在于将SQL语句组分解为逻辑块，并明确规定数据何时应归回滚，何时提交

# 回滚事务

凯盛软件

```
START TRANSACTION;/*开始事务*/
DELETE FROM t_tableA WHERE id = 1;
ROLLBACK;/*事务回滚*/
```

# 提交事务



```
START TRANSACTION;
DELETE FROM t_tableA WHERE id = 1;
COMMIT; /*事务提交*/
```

# 保留点



```
START TRANSACTION;
DELETE FROM t_tableA WHERE id = 4;
SAVEPOINT s1; /*声明一个保留点*/
DELETE FROM t_tableA WHERE id = 5;
ROLLBACK TO s1;  /*回滚到s1保留点*/
```

# 事务的四个特征（ACID属性）

凯盛软件

- 原子性（Atomic）：组成事务的处理语句组成了一个逻辑单元，这是最小的执行单位
- 一致性（Consistent）：在事务处理执行之前和之后，数据是一致的
- 隔离性（Isolated）：一个事务的处理对另一个事务没有影响
- 持续性（Durable）：当事务处理成功后，其结果在数据库中被永久记录下来

# JDBC中的事务

# 索引



索引是优化数据库查询速度重要途径

# 索引类型

凯盛软件

- 普通索引：最基本的索引类型，没有唯一性之类的限制
  - CREATE INDEX valindex ON t_tableA(val(20))
  - CREATE INDEX products_index ON products(prod_name(25),prod_price)
- 唯一索引：和普通所以基本相同，但所有的索引列只能出现一次，保持唯一性
  - CREATE UNIQUE INDEX valindex2 ON t_tableB(val(20))
- 主键索引：主键索引是一种特殊的唯一索引，在建立主键时自动创建
- 全文索引：全文索引可以在varchar或text类型上创建

# 索引的缺点



1.虽然索引大大提高了查询速度，但会降低更新表的速度，比如对表的insert、update、delete操作，因为更新表时，MySQL不仅仅要保存数据，还要保存索引文件

2.建立索引会占用磁盘空间。如果在一个大表上创建了多种索引组合，索引文件会膨胀的很快。