MySQL关系型数据库

（四）索引、事务、权限管理

作者：Daniel.Wang

主要内容

1. 索引
2. 事务
3. 权限管理

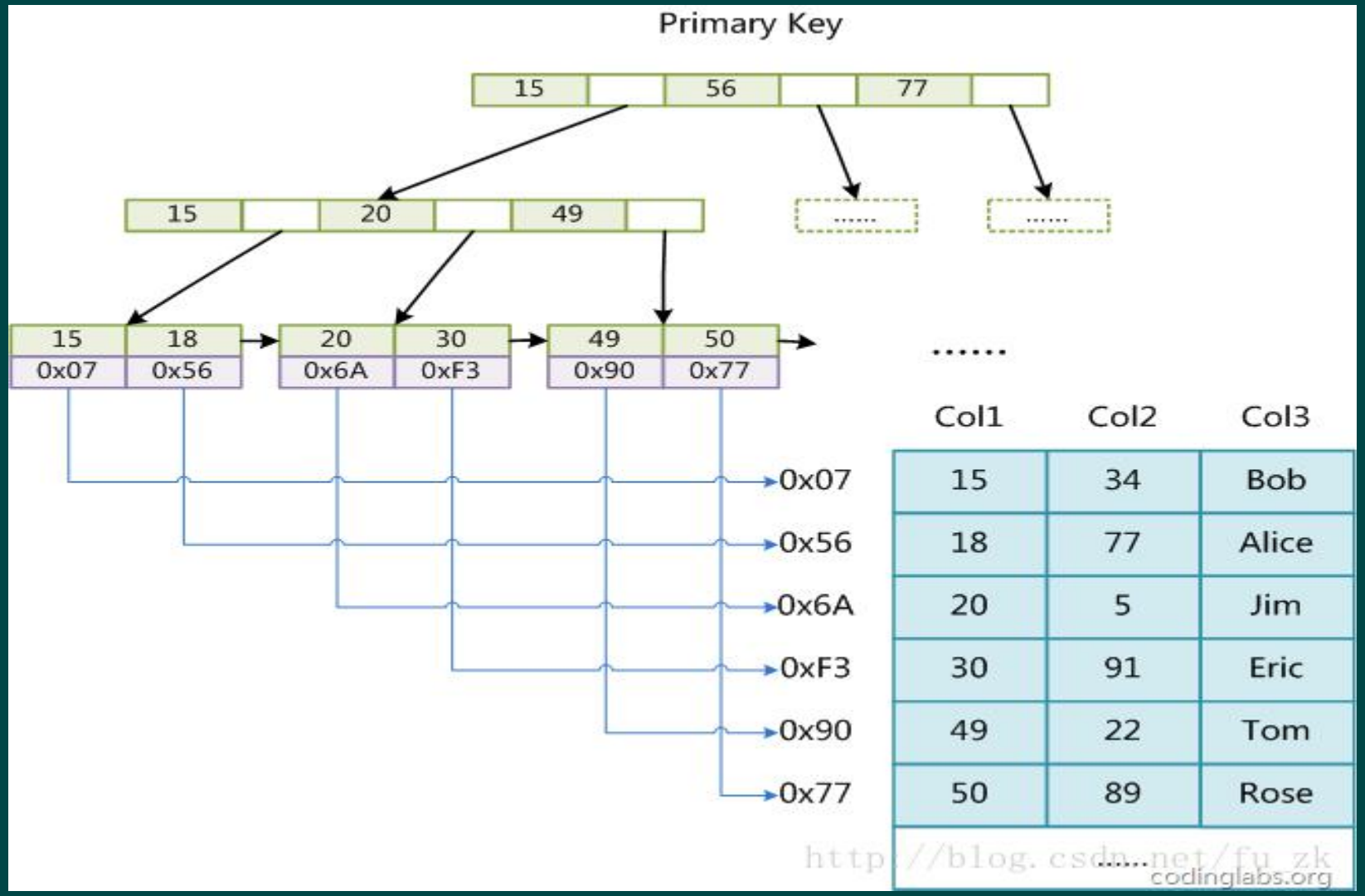
（一）索引

1. 什么是索引（Index）

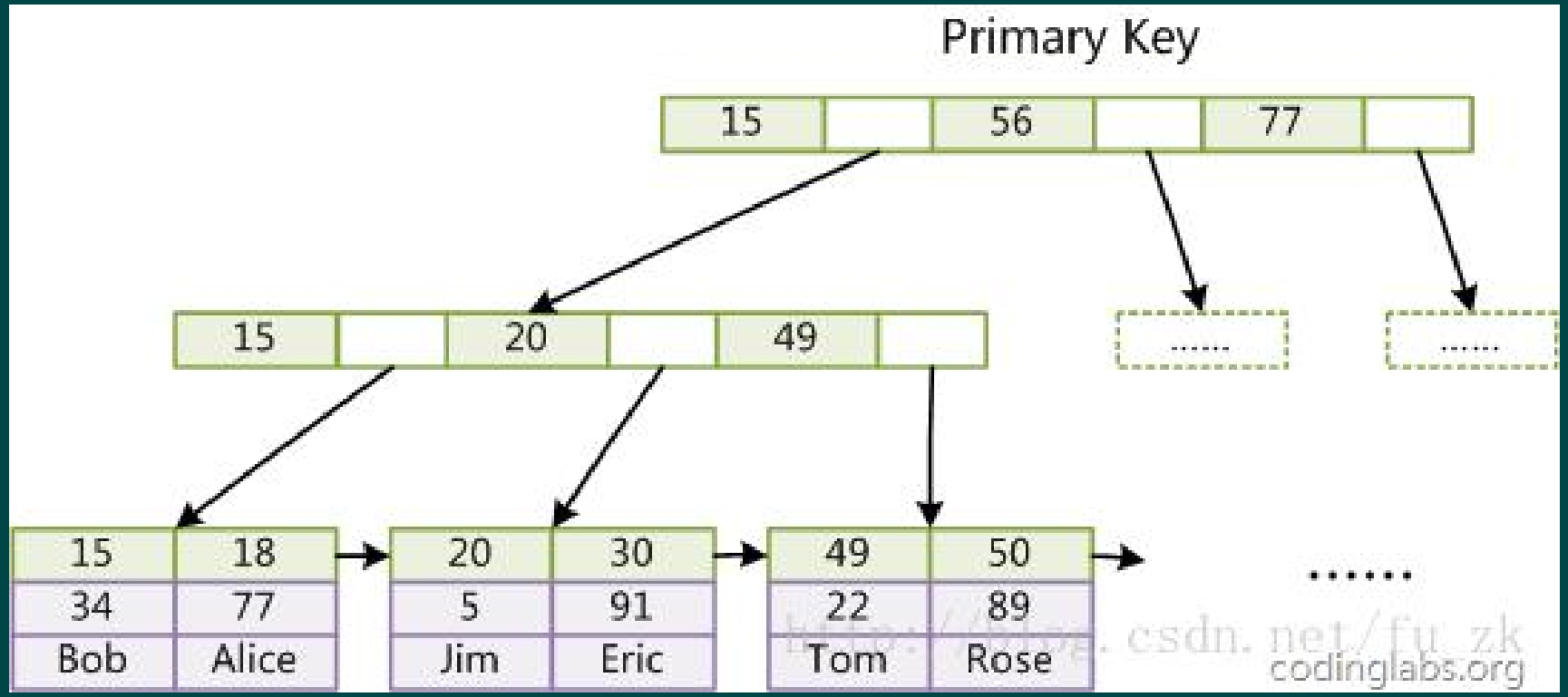
1）索引是一种提高查询的技术，**通过对数据表中一列或多列数据进行排序，在查询时避免全表扫描，从而提高查询效率。**

2）索引是一种单独存放的数据结构，包含着数据表中所有记录的引用指针，根据该指针能快速找到数据所存储的物理位置

索引结构示意图-MyISAM



索引结构示意图-InnoDB



2. 索引的分类

1）普通索引、唯一索引

- 普通索引：MySQL中索引的基本类型，允许在定义索引的列中插入空值、重复值
- 唯一索引：索引的值必须唯一，允许有空值

2）单列索引、组合索引

- 单列索引：一个索引只包含一个列
- 组合索引：一个索引包含多个列

3. 如何创建索引

1）建表语句创建索引

- 语法：index | unique | primary key(字段名称)
- 说明：index　　普通索引

  unique　唯一索引

  primary key　主键，特殊的唯一索引

- 示例：创建表index_test，在cert_no上创建唯一索引，在name上创建普通索引

```
create table index_test (
        id int primary key,  -- 主键
        cert_no varchar(32) ,
        name varchar(32),
        unique(cert_no),  index(name)
);
```

查看索引：show index from index_test;

插入数据测试：

```
insert into index_test values(1, '0001', 'Jerry');  -- OK
insert into index_test values(2, '0001', 'Tom');  -- 违反唯一索引
insert into index_test values(NULL, '0002', 'Tom');  -- 违反主键非空索引
```

2）表创建完成后添加索引

- 语法

  create 索引类型 索引名称 on 表名(字段名)

- 示例

  drop index cert_no on index_test; -- 先删除索引

  create unique index idx_cert_no on index_test(cert_no);

3. 删除索引

1）语法：drop index 索引名称 on 表名称

2）示例：

```
drop index cert_no on index_test;
```

4. 实验：索引查询效果测试

1）实验步骤

- 第一步：创建表或找一个已存在的表，先确认该表无索引

  利用现有的orders表

- 第二步：利用脚本插入10万笔数据

  见insert_orders_many.py

- 第三步：无索引查询，记录查询时间

```
select * from orders where order_id = '2018010100000002';   -- 0.08秒
select * from orders where order_id = '2018010100055556';   -- 0.07秒
select * from orders where order_id = '2018010100999994'; -- 0.08秒
```

第四步：添加索引，执行同样查询

```
create index idx_order_id on orders(order_id);  -- 创建索引

select * from orders where order_id = '2018010100000002';   -- 0.00秒
select * from orders where order_id = '2018010100055557';   -- 0.00秒
select * from orders where order_id = '2018010100099994'; -- 0.00秒
```

2）实验结论：10万笔数据下做单表查询，在连续值的字段上执行查询，有索引比无索引查询效率高出70倍以上

5. 索引的优缺点

1）优点

- 提高查询效率
- 唯一索引能保证数据唯一性
- 在使用分组、排序等子句查询时，能减少分组、排序所需的时间

2）缺点

- 索引需要额外的存储空间
- 创建、维护索引结构需要额外的时间，这个时间会随着数据量的增加而增加
- 对表进行插入、修改、删除时，索引需要动态调整，所以会降低增、删、改的效率

6. 索引使用要点

1）总体原则

- 合适的字段上，建立合适的索引
- 索引不是越多越好，过多的索引会降低增、删、改的效率

2）适合使用索引的情况

- 在经常查询、排序、分组的列使用索引
- 数据分布相对均匀、连续的字段，适合使用索引
- 查询操作较多的表上，适合创建索引

3）不适合使用索引的情况

- 如果表的数据量很少，不建议使用索引
- 经常更新的表不宜使用过多索引
- 避免在取值范围很少的列上使用索引（例如性别，账户状态）
- 二进制字段不适合建索引

（二）数据库事务

1. 什么是事务

1）事务（Transaction）：指执行的一系列操作，要么全都执行，要么全都不执行

2）作用：保证数据的正确性、一致性。例如，A账户向B账户转账1000元，需要执行以下两个操作，并且这两个操作要么全都执行，要么全都不执行：

- 操作1：减去转出账户上的金额

```
update acct set balance = balance - 1000 where acct_no = '0001';
```

- 操作2：转入账户上加上相同金额

```
update acct set balance = balance + 1000 where acct_no = '0002';
```

2. 事务的特征

事务有以下4个特征，习惯上简称为ACID特性：

1）原子性（Atomicity）

一个事务是不可分割的整体，要么全都执行，要么全都不执行

2）一致性（Consistency）

事务执行完成后，数据库从一个一致性状态变成另一个一致性状态

3）隔离性（Isolation）

不同的事务不相互影响、干扰

4）持久性（Durability）

一旦事务提交，对数据库的修改就必须永久保留下来

3. 使用事务情况及先决条件

1）以下情况下，需要启用事务：

- 一个交易涉及到多个增、删、改操作
- 需要保证数据一致性、正确性

2）MySQL下使用事务的先决条件

- 表的存储引擎类型必须为InnoDB

4. MySQL如何操作事务

1）启动事务

- 显式启动：start transaction
- 隐式启动：执行insert, update, delete操作时

2）提交事务：commit

3）回滚事务：rollback

4）数据库事务示例

- 第一步：创建测试账户表并插入测试数据

```
create table acct(
        acct_no varchar(32) primary key,
        acct_name varchar(64) not null,
        balance decimal(16,2) default 0
);
insert into acct values('0001', 'Jerry', 1000);
insert into acct values('0002', 'Tom', 2000);
```

- 第二步：通过事务，保证两次操作全部执行

```
start transaction;
update acct set balance = balance - 200 where acct_no = '0001';
update acct set balance = balance + 200 where acct_no = '0002';
commit;
```

- 第三步：将第二步语句中的commit改为rollback，测试事务回滚
- 第四步：演示事务隔离性，在第二步的语句中，在执行commit之前，新开启一个数据库连接查询两个账户余额（未发生变化）

5. 事务对哪些语句起作用

SQL语句按照功能，可分为以下四类，事务只能用于数据操作语言

- 数据查询语言(DQL)：用于用户从数据库请求获得数据
- 数据定义语言(DDL)：定义数据结构，如创建、修改、删除数据库对象（表、字段、索引）
- **数据操纵语言(DML)：对数据库进行追加、修改、删除（可用于事务）**
- 数据控制语言(DCL)：授予或回收权限，控制及操纵事务，对数据库进行监视

6. 事务对性能的影响

事务会对数据进行过加锁，一个事务正在操作数据时候，另一个事务要操作相同的数据，则必须等待，等待上一个事务提交。所以，事务会降低数据库的增删改效率，换来的好处是保证了数据一致性

（三）权限管理

1. 权限概述

1）什么是权限：用户可以进行哪些操作

2）权限分类

- 用户类：创建/删除用户、给用户授权
- 库/表操作：创建/删除库、创建/删除表
- 数据操作：增、删、改、查

3）权限表：MySQL将权限记录到表中，通过查询权限表来决定用户可以进行哪些操作。主要有以下几个权限表：

- user表：最重要的权限表，记录允许连接到服务器的账号信息和权限
- db表：记录库的授权信息
- tables_priv表：记录表的授权信息
- columns_priv表：记录授权的字段信息

2. 权限操作

1）授予权限

- 语法：

  grant 权限列表 on 数据库名.表名

  to '用户名'@'客户端地址'

  [identified by '密码']

  [with grant option]

- 说明：

  - 权限列表：表示有哪些权限

    all : 表示所有权限

    select,update,insert,... : 分别制定权限

- 权限列表：表示有哪些权限

  all：表示所有权限

  select,update,insert,...：分别制定权限

- 数据库名.表名

  .　　表示为所有库下所有表

  bank.acct　表示bank库下acct表

  bank.*　　表示bank库下所有表

- 客户端地址

  %　　表示所有客户端

  localhost　表示本机本机

  192.168.0.5　表示192.168.0.5这台机器

- with grant option：表示是否有授权权限

授予权限示例

- 示例1：给用户Daniel授予所有库、所有表下所有权限，并将密码设置为'123456'，并且允许给其他用户授权
  - 第一步：授权

    ```
    grant all privileges on *.*
    to 'Daniel'@'%' identified by '123456'
    with grant option;
    ```

  - 第二步：重新加载权限：FLUSH PRIVILEGES;
  - 第三步：然后查看user表核实

    ```
    select * from mysql.user where user = 'Daniel'\G
    ```

  - 第四步：通过增、删、改操作进行验证

- 示例2：给用户Tom授权，能对所有库、所有表进行查询，限定只能从本机进行登录，并将密码设置为'123456'
  - 第一步：授权

    grant select on *.*

    to 'Tom'@'localhost' identified by '123456';
  - 第二步：重新加载权限：FLUSH PRIVILEGES;
  - 第三步：然后查看user表核实

    select * from mysql.user where user = 'Tom'\G
  - 第四步：验证

    使用Tom登录本机，执行select（成功），insert操作（失败）

    使用Tom和非本机IP登录（不能登录）

课堂练习

- 给用户Jerry授权，能对eshop库下所有表进行增、删、改、查，可以从任意客户端地址登录

```
grant select,insert,update,delete
on eshop.* to 'Jerry'@'%' identified by '123456';
```

2）吊销权限

- 语法：

  revoke 权限列表 on 库名.表名

  from '用户名'@'客户端地址'

- 示例：吊销Jerry用户eshop库下所有表的删除权限，root用户执行以下命令

  revoke delete on eshop.* from 'Jerry'@'%';

3）查看权限

- 查看自己的权限：show grants;
- 查看其他人的权限：show grants for 'Tom'@'localhost'

（四）总结与回顾

1. 索引

- 索引是一种提高查询的技术，**通过对数据表中一列或多列数据进行排序，在查询时避免全表扫描，从而提高查询效率**
- 如何创建索引：

```
create table 表名 (
        ......
        unique(字段1),index(字段2)
);
```

- 索引优缺点：

  优点：提升查询效率，保证唯一性

  缺点：额外时间、空间开销

- 索引使用要点
  - 原则：建立合适的索引，索引不宜过多
  - 适合使用索引的情况
    - 在经常查询、排序、分组的列使用索引
    - 数据分布相对均匀、连续的字段，适合使用索引
    - 查询操作较多的表上，适合创建索引
  - 不适合使用索引的情况
    - 如果表的数据量很少，不建议使用索引
    - 经常更新的表不宜使用过多索引
    - 避免在取值范围很少的列上使用索引（例如性别，账户状态）
    - 二进制字段不适合建索引

2. 数据库事务

- 作用：保证一组操作，要么全执行，要么全不执行
- 特性：ACID

  原子性：一组操作，要么全执行，要么全不执行

  一致性：事务执行完成后，数据库从一个一致性状态变成另一个一致性状态

  隔离性：不同的事务不相互影响、干扰

  持久性：一旦事务提交，对数据库的修改就必须永久保留下来
- 使用事务前提：InnoDB存储引擎
- 操作：

  开启：start transaction，执行增、删、该语句时

  提交：commit

  回滚：rollback

3. 权限

- 什么是权限：用户可以执行哪些操作
- 授权

  grant 权限列表 on 数据库名.表名

  to '用户名'@'客户端地址'

  [identified by '密码']

  [with grant option]
- 吊销：revoke delete on bank.* from 'Jerry'@'%';
- 查看：
  - ✓ 查看自己的权限：show grants;
  - ✓ 查看其他人的权限：show grants for 'Tom'@'localhost'