### MySQL进阶课程大纲

**实验 MySQL数据库的基础**

**一、实验介绍**

**1.1 知识点**

MySQL 的安装与启动

MySQL 的连接与断开

数据库的基本操作

**1.2 实验环境**

课程使用的实验环境为 Ubuntu Linux 16.04 64 位版本。实验中会用到程序：

Mysql 5.7.22、Xfce 终端

1. **实验内容**

**2.1 MySQL介绍**

**2.2 MySQL安装**

windows 下的安装

Linux 下的安装

**2.3 启动 MySQL 服务器**

**2.4 连接与断开服务器**

**2.5 创建并使用数据库**

使用 CREATE DATABASE 数据库名来创建数据库。用 SHOW DATABASES来查看当前存在的所有数据库。若数据库创建成功，会在数据库表中看到该数据库的名字。当创建成功后，我们需要 USE 数据库名 这个命令执行后，才能使用该数据库。

**2.6 删除数据库**

使用 DROP DATABASE 数据库名来删除数据库。

**实验2 数据表的使用**

1. **实验介绍**
   1. **知识点**

创建数据表

表中添加数据

查询表中数据

* 1. **实验环境**

1. **实验内容**

**2.1 创建并查看数据表**

现在已经成功地创建数据库了，我们在该数据库中创建数据表。我们使用 CREATE TABLE 数据表名 来创建数据表。

创建数据表后，使用 SHOW TABLES 来查看一下，若我们能够看到该表的名字，说明该数据表创建成功。

使用 DESCRIBE 数据表名来查看创建数据表的完整信息。

**2.2 在数据表中添加数据**

将数据加载到数据表中，有LOAD DATA和 INSERT 这两种方式。LOAD DATA 是一次可以添加多条数据，可以文本文件中的数据直接加载到数据表中。INSERET 是一次增加一条新的数据。

**2.3 查询表中所有的数据**

我们使用 SELECT \* FROM 数据表名来查看该表中的所有数据。若与刚才添加的数据相同，说明数据成功添加到数据表中。

例如：

**2.4 检索特定行的数据**

我们使用 SELECT \* FROM 数据表名 WHERE 关键字 来查询特定行的数据。

**2.5 检索特定列的数据**

我们也可以仅查看某些列的数据。我们可以使用 SELECT 列名 FROM 数据表名，这里可以写入多个列名，用逗号隔开即可。

**2.6 同时检索特定行和列的数据**

我们可以使用 SELECT 列名 FROM 数据表名 WHERE 关键字 来同时进行特定行和列的数据选择。

**2.7 排序检索数据**

我们可以的检索的数据进行排序，例如在检索学生成绩时，我们可以对学生成绩进行排序，这样使得查询结果更加清晰。我们使用 OREDER BY 数据列名 可实现对1列或者多列数据进行排序操作。该排序操作默认是升序，我们可以在其后添加关键字 DESC来变成降序。

1. **实验总结**

**实验3 MySQL查询常用知识点**

1. **实验介绍**
   1. **知识点**

NULL值操作

模式匹配

COUNT函数和MAX函数

用户变量

AUTO\_INCREMENT

* 1. **实验环境**

1. **实验内容**

**2.1 NULL 值操作**

在 MySQL 中查询数据时，当遇到查询条件的某些字段为NULL时，可能该查询语句无法正常工作。MySQL 为我们提供了操作符来解决这个问题。它们分别是：

IS NULL ：当列的值为NULL 时，返回结果为true。

IS NOT NULL：当列的值不为空时，返回结果为 true。

**2.2 模式匹配**

在学习模式匹配之前，我们首先应知道LIKE字句的用法。LIKE 相当于’=’的意思，在使用SQL模式时，不能使用’=’或者’!=’，而应使用LIKE 或者 NOT LIKE的操作符。

MySQL 也支持正则表达式的匹配，使用 % 可以匹配任意数量的字符。使用 \_ 匹配任意单个。字符。

另外，MySQL还是有一些扩展正则表达式，如下所示：

^匹配输入字符串的开始位置。

$匹配输入字符串的结束位置。

{n}匹配确定的n次，这里的n必须是一个非负整数。

**2.3 COUNT函数的使用**

当你需要知道某个类型的数据在表中出现的频数时，这时我们就需要使用计算命令。我们可以使用COUNT 函数来计算行数。

**2.4 MAX函数的使用**

查询列的最大值： SELECT MAX(数据名) as 数据名 FROM 表名

查询某列数据最大值的所在行：SELECT 数据名 FROM 表名WHERE 数据名(SELECT ,MAX(数据名) FROM 表名)

按组显示列的最大值：SELECT 列名1，MAX(列名2) AS 列名2 FROM 表名 GROUP BY 列名1

**2.7 用户变量的使用**

为了便于使用，我们可以定义一个特定的变量来保存该值，这样的变量就是MySQL的用户变量了。在SELECT命令中设置临时变量，需要使用@开头，然后需要使用:=来进行赋值。

**2.10 过滤重复数据**

我们可以在使用BIT\_OR 和 BIT\_COUNT这两个函数来过滤掉重复的数据。

**2.11 AUTO\_INCREMENT 的使用**

在MySQL中AUTO\_INCREMENT用于为表中的每条记录自动生成ID，这使得表中的每条记录能够被唯一标识。AUTO\_INCREMENT默认是从1开始自增的。

1. **实验总结**

**实验4 多表查询操作**

1. **实验介绍**
   1. **知识点**

联接查询

联合查询

子查询

* 1. **实验环境**

1. **实验内容**

**2.1 联接查询介绍**

联接查询是一种常见的数据库操作，即在两张表或者多张表上进行行匹配的操作。MySQL数据库支持主要三种联接查询：内连接、外联结、交叉联接。

**2.2 交叉联接查询CROSS JOIN**

CROSS JOIN是对两个表执行笛卡尔积，返回两个表中所有列的组合。若表1有m行数据，右表有n行数据，则CROSS JOIN 将返回m\*n行的表。

**2.3 内联接查询INNER JOIN**

通过INNER JOIN用户可以根据一些过滤条件来匹配表之间的数据。在逻辑查询的前三个阶段处理中，INNER JOIN 应有前两个阶段，即首先产生笛卡尔积的虚拟表，再按照过滤条件来进行数据匹配操作。

**2.4 外联接查询 OUTERJOIN**

通过OUTER JOIN用户可以根据一些过滤条件来匹配表之间的数据。它与内联接的区别在于，通过OUTER JOIN添加的保留表中存在未找到的数据。在MySQL数据库中支持 LEFT OUTER JOIN和 RIGHT OUTER JOIN。

LEFT OUTER JOIN 是以左表为主表，RIGHT OUTER JOIN 是以右表为主表。

需注意的是 OUTER JOINI需要制定ON字句，不然MySQL 数据库会抛出异常。

**2.5 联合查询**

联合查询是将多个SELECT语句的查询结果合并到一块的查询，可以使用UNION 或者 UNION ALL 来进行操作。

**2.6 子查询**

子查询是将一个查询语句嵌套在另一个查询语句中。子查询中可以包含：IN、NOT IN、ANY、ALL、EXISTS 和 NOT EXISTS等关键字。

1. **实验总结**

**实验5 MySQL中的约束和索引**

1. **实验介绍**
   1. **知识点**

约束

索引

* 1. **实验环境**

1. **实验内容**

外键和索引是两种重要的数据库对象。常用的数据库对象还有表、数据字典、视图、函数、存储过程、触发器，详细内容请看文档。

**2.1 约束介绍**

约束是插入和处理数据库数据的规则。

**主键约束**

PRIMARY KEY 用于定义表的主键，是唯一确定表中每一条记录的标识符。

**外键约束**

FOREIGN KEY是表中的一列，其值必须列在另一表的主键中。外键约束有三种：严格模式、级联模式、置空模式。

**唯一约束**

UNIQUE是用来设置某列数据不能重复，但是可以有空值。

**检查约束**

CHECK是保证一列中的数据满足指定的条件。

**默认值约束**

DEFAULT 是用于在插入操作时，某一列没有值，系统会自动把之前设置的默认值赋值到该列。

**2.2 索引类型介绍**

目前常用的索引类型有以下四个：

**FULLTEXT**：全文索引是一种特殊类型的索引，它查找的是文本中的关键词。

**HASH**：哈希索引是基于哈希实现的，只有精确匹配索引的所有列的查询才有效。

**BTREE**：B-Tree索引是索引的值按照顺序存储。

**RTREE**：R-Tree空间数据索引是MyISAM表支持空间数据索引，可以用作地理数据存储。

**2.3 索引种类介绍**

索引是存储引擎用于快速找到记录的一种数据结构，在MySQL中也叫做键。

**普通索引**

INDEX是最基本的索引，没有任何限制。

**唯一索引**

UNIQUE是索引列的值必须唯一，但允许有空值。

**组合索引**

多列值组成一个索引，专门用于组合搜索，其效率大于索引合并。

**全文索引**

FULLTEXT是对文本的内容进行分词，进行搜索。

**主键索引**

PRIMARY KEY是一种特殊的唯一索引，它不允许有空值。

**删除索引**

使用 DROP INDEX 可以删除索引。

**2.4约束和索引的区别**

约束是为了保证数据的完整性，而索引是为了提高查询数据的效率。

1. **实验总结**

**实验6 MySQL权限和用户账户的管理**

1. **实验介绍**

**1.1知识点**

MySQL 权限

用户账户管理

**1.2实验环境**

1. **实验内容**

**2.1 MySQL 权限的介绍**

账户权限信息被存储在 mysql的数据库中的user，db，host,tables\_priv，columns\_priv,procs\_priv表中。

MySQL的权限有三大级别：

**全局性的管理权限**：作用于整个MySQL实例级别

**数据库级别的权限**：作用于某个指定的数据库上或者所有的数据库上

**数据库对象级别的权限**：作用于指定的数据库对象上（表、视图等）或者所有的数据库对象上

**2.2 MySQL 用户账户的管理**

增加新用户账户

删除用户账户

限制账户资源

设置账户密码

1. **实验总结**

**实验7 数据库的事务**

1. **实验介绍**
   1. **知识点**

事务

事务处理

* 1. **实验环境**

1. **实验内容**

**2.1 事务的介绍**

事务是数据库区别于文件系统的重要特性之一。它用于处理操作量大和复杂度高的数据。事务必须同时满足ACID四个条件：原子性（Atomicity）、一致性（Consistency）、隔离性（Isolation，）、持久性（Durability）。

**2.2 事务的分类**

平扁式事务，带有保存点的平扁式事务，链事务，嵌套事务，分布式事务

**2.3 事务控制语句**

事务都是自动提交的，即执行SQL语句后，马上执行COMMIT操作。

用户可以使用的事务控制语句有：

**BEGIN 或 START TRANSACTION**：显式地开启一个事务

**COMMIT 或COMMIT WORK**：不过二者是等价的。COMMIT 会提交事务，并使已对数据库进行的所有修改成为永久性的

**ROLLBACK 或 ROLLBACK WORK**：不过二者是等价的。回滚会结束用户的事务，并撤销正在进行的所有未提交的修改

**SAVEPOINT identifier**：SAVEPOINT 允许在事务中创建一个保存点，一个事务中可以有多个 SAVEPOINT

**RELEASE SAVEPOINT identifier**：删除一个事务的保存点，当没有指定的保存点时，执行该语句会抛出一个异常

**ROLLBACK TO identifier**：把事务回滚到标记点

**SET TRANSACTION**：用来设置事务的隔离级别。InnoDB 存储引擎提供事务的隔离级别有READ UNCOMMITTED、READ COMMITTED、REPEATABLE READ 和 SERIALIZABLE。

**2.4 事务处理**

**用 BEGIN, ROLLBACK, COMMIT来实现**

BEGIN开始一个事务

POLLBACK 事务回滚

COMMIT 事务确认

**用 SET 来改变 MySQL 的自动提交模式**

SET AUTOCOMMIT=0禁止自动提交

SET AUTOCOMMIT=1开启自动提交

1. **实验总结**

**实验8 MySQL的存储引擎**

1. **实验介绍**
   1. **知识点**

MySQL的存储引擎

InnoDB存储引擎中的锁

* 1. **实验环境**

1. **实验内容**

**2.1 存储引擎介绍**

存储引擎是指表的类型以及表在计算机上的存储方式。

**2.2 查看存储引擎**

使用SHOW ENGINES 可以查看MySQL的存储引擎信息。

**2.3 MySQL常用的存储引擎**

**InnoDB存储引擎[重点讲解]**

InnoDB存储引擎是MySQL默认事务型引擎，它被设计用来处理大量的短期事务。InnoDB的数据存储在表空间中，表空间是由InnoDB管理的一个黑盒子，由一系列的数据文件组成。

**InnoDB关键特性：插入缓冲、两次写、自适应哈希索引、异步IO、刷新邻接页。**

**MyISAM存储引擎**

MYISAM存储引擎提供了大量的特性，包括全文索引、空间函数、压缩等，但它不支持事务和行级锁。

**MEMORY存储引擎**

MEMORY存储引擎将表中的数据存储到内存中，未查询和引用其他表数据提供快速访问。

**ARCHIVE存储引擎**

ARCHIVE存储引擎仅仅支持插入和查询这两种功能。

**MERGE存储引擎**

MERGE存储引擎是MyISAM存储引擎的一个变种，对Merge类型的表可以进行查询、更新、删除操作。

更多存储引擎请查看文档学习。

**2.4 InnoDB存储引擎中的锁**

InnoDB实现了两种标准的行级锁：

**共享锁S Lock**：允许事务读一行数据

**排他锁X Lock**：允许事务删除或更新一行数据

意向锁是InnoDB额外的锁方式，也有两种：

**意向共享锁IS Lock**：事务想要获得表中某几行的共享锁

**意向排他锁IX Lock**：事务想要获得表中某几行的排他锁

1. **实验总结**

**实验9 MySQL的优化处理**

1. **实验介绍**

**1.1知识点**

数据库设计优化

表创建优化

索引优化

查询性能优化

**1.2实验环境**

1. **实验内容**

**2.1数据库设计的优化策略**

创建索引：在查询时，索引是能够很好提高查询性能。

复合索引：当一条查询语句中有多个匹配字段时，我们可以在这几列上创建复合索引来提高查询效率。

无NULL值：在创建数据库时，尽量避免有NULL值，若列中有空值，将不会包含在索引中，我们可以使用0去代替NULL。

使用短索引：节省查询时间，节省磁盘空间和I/O操作。

尽量不要包含多个列的排序，尽量不使用LIKE语句，不要在列上进行运算

不要使用NOT IN：可以用NOT EXISTS来代替 NOT IN

**2.2表创建的优化策略**

在设计表时，我们尽量避免字段有NULL出现，我们可以使用0去代替NULL。尽量使用INT而不是BIGINT。使用枚举或整数代替字符串类型。尽量使用TIMESTAMP而非DATETIME。用整型来存IP。

**2.3 索引优化**

索引并不是越多越好，要根据查询有针对性的创建，考虑在WHERE和ORDER BY命令上涉及的列建立索引，可根据EXPLAIN来查看是否用了索引还是全表扫描。

尽量避免在WHERE子句中对字段进行NULL值判断，否则将导致引擎放弃使用索引而进行全表扫描。

值分布很少的字段不适合做索引。字符字段只做前缀索引。字符字段最好不要做主键。尽量不用UNIQUE，由程序保证约束。使用多列索引时主意顺序和查询条件保持一致，同时删除不必要的单列索引。

**2.4查询性能优化**

在查询结果后面加上LIMIT来避免向数据库请求不必要的数据。

避免重复查询相同的数据，初次查询时，将这个数据缓存起来，需要的时候再从缓存中取出来。

在MySQL中衡量开销的三个指标：响应时间、扫描行数、返回行数，这三个指标都会记录到慢日志中，检查慢日志是找出扫描过多行数查询的好办法。

在设计查询时，我们需要考虑，是否需要将一个复杂的查询分成几个简单的查询，比如需要定期清理大量数据时，我们可以将DELETE语句分成多个较小的查询，这样可以尽可能小的影响MySQL的性能。

很多高性能的应用都会将关联查询进行分解，就是可以对每一张表进行单表查询，然后将结果关联起来。

1. **实验总结**

**实验10 MySQL中常见的日志文件**

1. **实验介绍**
   1. **知识点**

错误日志

通用查询日志

二进制日志

慢速查询日志

* 1. **实验环境**

1. **实验内容**

**2.1错误日志**

MySQL错误日志是用来记录MySQL运行过程中比较严重的警告和错误信息。

以及MySQL每次启动和关闭的信息。

查看错误日志详细信息：SHOW VARIABLES LIKE ‘LOG\_ERR’

**2.2 通用查询日志**

记录建立的客户端连接和执行的语句

查看数据库的版本号和存储引擎等信息：SHOW VARIABLE LIKE’%VEREISION%’

查看当前的通用日志是否开启：SHOW VARIABLE LIKE’%GENERAL%’

开启通用日志查询：SET GLOBAL GENERAL\_LOG=ON

关闭通用日志查询：SET GLOBAL GENERAL\_LOG=OFF

查看当前慢文件的格式：SHOW VARIABLES LIKE’%LOG\_OUTPUT%’

**2.3 二进制日志**

查看二进制日志：

show binary logs是用于查看当前使用了哪些二进制文件。

show binlog events是用于查看日志中进行了哪些操作。

show master status 是用于显示主服务器中的二进制日志信息。

删除二进制日志：

reset master 是用于删除所有日志

purge master logs to "binlog\_name.00000X"是用于清空00000X之前的所有日志。

--expire\_logs\_days=N是用于过了N天的日志将会被清空。

**2.4 慢速查询日志**

记录所有执行时间超过long\_query\_time秒的所有查询或者不适用索引的查询

查看当前慢日志是否开启：SHOW VARIABLES LIKE ‘%QUERY%’

1. **实验总结**

**实验11 数据库的恢复与备份**

1. **实验介绍**
   1. **知识点**

数据备份

自动恢复

基于binlog的恢复

* 1. **实验环境**

1. **实验内容**

**2.1 数据库备份的介绍**

数据库备份是指通过导出数据或者复制表文件的方式来制作数据库的副本。当数据库出现故障或遭到破坏时，将备份的数据库加载到系统，从而使数据库从错误状态恢复到备份时的正确状态。

**2.2 数据库备份的实现**

我们可以使用 SELECT INTO ... OUTFILE或者BACKUP TABLE来对数据库进行备份。

备份策略例子

**2.3 数据库恢复的介绍**

数据库恢复是以数据库的备份为基础的，与备份相对应的系统维护和管理操作。系统进行恢复操作时，先执行一些系统安全性的检查，包括检查所要恢复的数据库是否存在、数据库是否变化及数据库文件是否兼容等，然后根据所采用的数据库备份类型采取相应的恢复措施。

**2.4 自动恢复**

指定恢复时间

指定恢复位置

**2.5 利用binlog 进行数据恢复**

**binlong的基本配置**

binlong需要在MySQL中的配置文件的的mysqld节点中进行配置：

server-id

log\_bin=日志路径

expire\_logs\_days=保存几天的日志

max\_binlog\_size=每个binlog的大小

binlog\_format=binlog的模式设置

[binlog有三种格式：ROW是针对行记录日志，每行修改产生一条记录。STATEMENT是针对sql语句的，每条语句产生一条记录。MIX是结合了ROW和STATEMENT这两种格式的优点]

**查看binlog的状态**

查看binlog文件：SHOW BINARY LOGS

查看日志状态：SHOW VARIABLES LIKE’%log\_bin%’

查看日志文件位置：SHO MASTER STATUS

**基于binlog数据恢复的具体操作**

先根据操作时间来定位对应的binlog文件

筛选具体的数据库

压缩回文件分析

利用binlog2sql生成回滚语句

1. **实验总结**

**实验12 MySQL的高可用性**

1. **实验介绍**

**1.1知识点**

高可用性

**1.2实验环境**

1. **实验内容**

**2.1高可用性的介绍**

高可用性通常以百分比表示，它不是绝对的，是一种相对更高的可用性。每个应用对可用性的需求不同。可用性每提高一点，所花费的成本远会超过之前，所有需要保证多少可用时间，取决于你能够承担多少可用成本。

**2.2高可用性的实现**

我们可以同时进行以下两步来获得高可用性：

提升平均失效时间MTBF

降低平均恢复时间MTTR

**2.3避免单点失效**

共享存储或者磁盘复制

MySQL同步复制：MySQL Cluster、Percona XtraDB Cluster

基于复制的冗余

**2.4故障转移和故障恢复**

提升备库或者切换角色

虚拟IP地址或者IP接管

中间件解决方案

在应用中处理故障转移

**2.5高可用的方案介绍**

公司常采用三种高可用方案：MHA、MGR、PXC

MHA（Master High Availability）由MHA Manager（管理节点）和MHA Node（数据节点）两部分组成。

MGR（MySQL Group Replication）是基于Paxos协议的状态机复制，彻底解决了基于传统的异步复制和半同步复制中数据一致性问题无法保证的情况。

POX （Percona XtraDB Cluster）是MySQL高可用性和可扩展性的解决方案。

1. **实验总结**

**实验13 MySQL的集群**

1. **实验介绍**
   1. **知识点**

集群

* 1. **实验环境**

1. **实验内容**

**2.1 集群的介绍**

MySQL集群是两项技术的结合，NDB数据库，以及作为SQL前端的MySQL数据引擎。NDN是一个分布式、具有容错性、非共享的数据库，它提供同步复制以及节点间数据的自动分片。集群的目的在于提供容错性和高性能。

通过多个MySQL服务器分配负载，从而最大程序地达到高性能，通过在不同位置存储数据保证高可用性和冗余。

MySQL集群分为三种节点：管理节点、数据节点、SQL节点。

**2.2 搭建一个分布式集群**

准备集群搭建环境

准备安装包

集群搭建流程

集群配置与启动

集群测试

关闭集群

1. **实验总结**