Technology Tarena 达内科技

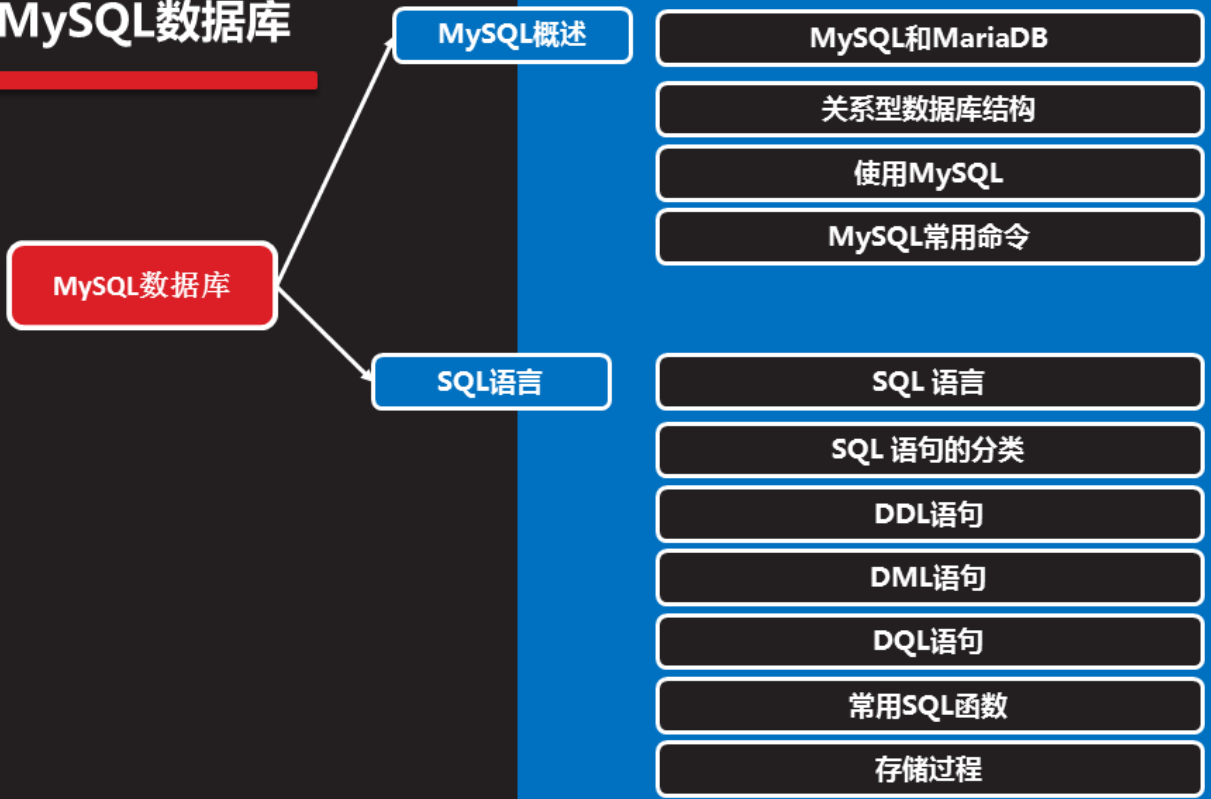
前端全栈 后台开发

Node.js Unit03

内容

上午	09:00 ~ 09:30	作业讲解和回顾
	09:30 ~ 10:20	MySQL数据库
	10:30 ~ 11:20	
	11:30 ~ 12:00	
下午	14:00 ~ 14:50	mysql模块
	15:00 ~ 15:50	
	16:00 ~ 16:50	
	17:00 ~ 17:30	总结和答疑

MySQL数据库



MySQL概述

MySQL和MariaDB

知识讲解

- MySQL 是当前最流行的开源关系型数据库；它是 LAMP（Linux Apache MySQL PHP）黄金组合中的重要一员
- 目前 MySQL 数据库有两个分支
  - mysql.com：由商业公司 Oracle 维护的版本
  - mariadb.org：由 MySQL 最初编写者组建的 MariaDB 基金会维护的版本。完全兼容前者的特性，并保证一直维持开源状态

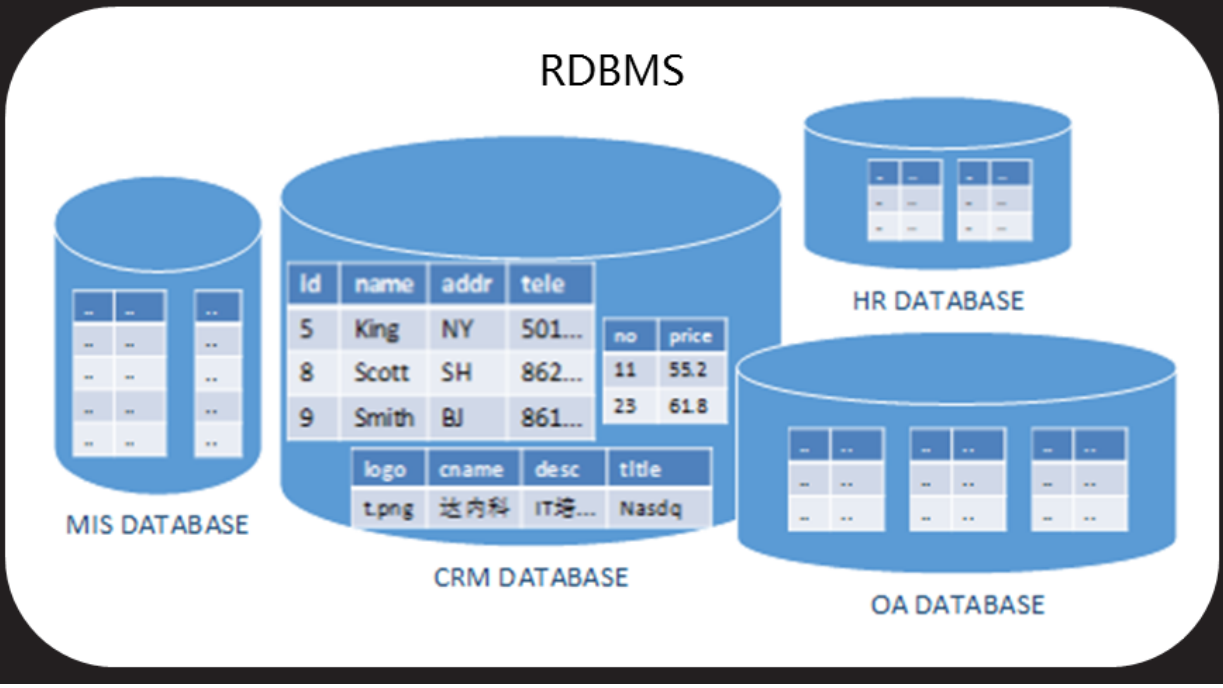
关系型数据库结构

Tarena Technology 达内科技

- RDBMS中的数据在逻辑结构上通常呈现下述层级关系：

DATABASE -> TABLE -> ROW -> COLUMN

知识讲解



使用MySQL

Tarena Technology 达内科技

- 使用MySQL服务器的基本步骤：
  - 从 mariadb.org 下载适用于当前操作系统的 MariaDB 安装文件；
  - 安装 MariaDB 服务器；
  - 启动 MariaDB 服务器端程序；
  - 使用 MariaDB 客户端工具，连接并登录 MariaDB 服务器；
  - 提交 SQL 命令给 MariaDB 服务器执行。

知识讲解

```
//交互模式执行命令
mysql  -h127.0.0.1  -uroot  -p  -P3306   dbname
```

```
//脚本模式执行命令
mysql  -h127.0.0.1  -uroot  -p  -P3306  <   d:/my.sql
```

MySQL常用命令

- 下述命令并不是标准的SQL语句，属于被MySQL服务器支持的操作命令：

知识讲解

```
SOURCE e:/x.sql        #导入并提交执行指定的脚步文件

SHOW DATABASES;  #显示服务器上当前存在哪些数据库
USE dbname;

SHOW TABLES;                    #显示当前数据库中有哪些表
SHOW CREATE TABLE tblname;      #显示创建表所使用的命令
DESC tblname;                   #描述表的结构

SHOW ERRORS;                    #显示最近的错误消息
SHOW WARNINGS;                  #显示最近的警告消息
```

SQL语言

SQL 语言

知识讲解

- SQL：Structured Query Language，结构化查询语言，用于操作关系型数据库中数据的标准化语言
  - 1986年由ANSI推出；1989年由ISO推出SQL89标准；1992年由ISO推出SQL92标准；1999年由ISO推出SQL99标准；
- SQL语言被主流的关系型数据库所支持，所以相同的SQL语句在大多数数据库中都可以顺利执行；但各个厂家在实现的细节与拓展方面有所不同，比如：
  - 列类型不尽相同；
  - 分页查询实现不同，
  - 工具函数不尽相同；
  - 存储过程语法不同 ...

SQL 语句的分类

知识讲解

- 所有的SQL语句可以分为如下四类：
  - DDL：Data Define Language，定义数据的结构；
  - DML：Data Manipulate Language，操作数据行；
  - DQL：Data Query Language，查询数据；
  - DCL：Data Control Language，控制用户的权限；

DDL	DML	DQL	DCL
CREATE	INSERT	SELECT	GRANT
DROP	DELETE		REVOKE
ALTER	UPDATE		
TRUNCATE			

DDL语句

Tarena Technology 达内科技

知识讲解

- CREATE：用于创建数据库对象，如库、表、视图等

```
CREATE DATABASE erp CHARSET utf8;
CREATE TABLE emp(id INT, ename VARCHAR(32) );
```

- DROP：用于丢弃数据库对象，如库、表、视图等

```
DROP DATABASE IF EXISTS erp;
DROP TABLE emp;
```

- ALTER：用于修改数据库对象的定义

```
ALTER TABLE emp ADD weixin VARCHAR(32);
ALTER TABLE emp CHANGE phone phone VARCHAR(11);
```

- TRUCATE：用于截断表中所有记录

```
TRUNCATE TABLE emp;
```

DML语句

Tarena Technology 达内科技

知识讲解

- INSERT：向表中插入一行或多行记录

```
INSERT INTO emp(id, name) VALUES(81, 'King');
INSERT INTO emp VALUES(82, 'Tom'), (83,'Joe'), (84, 'Ted');
```

- DELETE：删除表中的记录

```
DELETE FROM emp ;
DELETE FROM emp WHERE id=81;
```

- UPDATE：修改表中的记录

```
UPDATE emp SET name='Kane';
UPDATE emp SET name='Kane' WHERE id=88;
```

创建数据库和表

- 创建数据库和表

课堂练习

DQL语句

- SELECT：查询表中的记录，是 SQL 语句中最复杂的一条语句。基本查询语法：

```
SELECT eid, ename FROM emp;
```

知识讲解

- 查询所有的列：

```
SELECT * FROM emp;
```

- 为列取别名：

```
SELECT eid AS i, ename AS n FROM emp;
```

- 只显示列上不同的值：

```
SELECT DISTINCT deptId FROM emp;
```

- 查询过程中执行运算：

```
SELECT salary*12+comm   FROM emp;
```

DQL语句（续1）

知识讲解

- 条件查询：实现根据特定的条件对结果集进行筛选。
- 进行相等或不等判定：

```
SELECT * FROM emp WHERE eid = 81;
SELECT * FROM emp WHERE salary >= 6000 ;
SELECT * FROM emp WHERE deptId <> 10;
SELECT * FROM emp WHERE age BETWEEN 20 AND 30;
SELECT * FROM emp WHERE deptId IN (10, 30, 40);
```

- 多条件并列：

```
SELECT * FROM emp WHERE age>40 AND deptId=10;
SELECT * FROM emp WHERE salary>8000 OR salary<4000;
SELECT * FROM emp WHERE deptId NOT IN (10, 30, 40);
```

- 模糊查询：

```
SELECT * FROM emp WHERE ename LIKE '%e%';
```

DQL语句（续2）

知识讲解

- 查询排序：可以使用 ORDER BY 子句对查询结果集进行排序。
- 升序排列：

```
SELECT * FROM emp ORDER BY salary ;
```

- 降序排列：

```
SELECT * FROM emp ORDER BY salary  DESC ;
```

- 多列排序：

```
SELECT * FROM emp ORDER BY deptId, age DESC ;
```

DQL语句（续3）

知识讲解

- 分页查询：从符合条件的海量记录中仅显示一部分。不同数据库中的分页查询语法各不相同。MySQL 使用 LIMIT 关键字实现分页查询。
- 基本语法：

```
SELECT * FROM emp
[WHERE ...]
[ORDER BY ...]
LIMIT start, count;
```

- 假设每页最多显示 20 条记录，则：

```
SELECT * FROM emp LIMIT 0, 20;          #第 1 页
SELECT * FROM emp LIMIT 20, 20;         #第 2 页
SELECT * FROM emp LIMIT 40, 20;         #第 3 页
SELECT * FROM emp LIMIT (n-1)*20, 20;   #第 n 页
```

DQL语句（续4）

知识讲解

- MySQL 提供了五个聚合函数，可以对查询结果集进行特定的运算：

```
SELECT MAX(salary) FROM emp;        #查询工资最大值
SELECT MIN(salary) FROM emp;        #查询工资最小值
SELECT SUM(salary) FROM emp;        #查询工资总和
SELECT COUNT(salary) FROM emp;      #查询工资数量
SELECT AVG(salary) FROM emp;        #查询工资平均值
```

- 分组查询，指将指定列上的值相同的记录划分在一组中，在组内进行聚合运算：

```
SELECT MAX(salary) FROM emp GROUP BY deptId;
SELECT COUNT(salary) FROM emp GROUP BY deptId;
SELECT AVG(salary) FROM emp GROUP BY deptId;
```

DQL语句（续5）

知识讲解

- 子查询，是在一个查询语句中的某个或多个子句中包含其它的查询语句，是一种复合的查询语句。

```
SELECT * FROM emp
WHERE deptId = (
        SELECT did FROM dept
        WHERE dname = '研发部'
);
```

- 子查询也常常出现在 DML

```
UPDATE emp
SET salary = salary*1.1
WHERE salary < (
        SELECT AVG(salary)
        FROM emp
);
```

该语句在其它数据库中都是合法的；但MySQL却不允许执行，如何绕过MySQL中的限制的呢？

DQL语句（续6）

知识讲解

- 多表查询，也称跨表查询，指一次查询的结果集中出现来自多个表中的多个列。具体分为四种：
- (1) 内连接（INNER JOIN）：

```
SELECT ename, dname FROM emp INNER JOIN dept
ON emp.deptId = dept.did;
```

- (2) 左外连接（LEFT OUTER JOIN）：

```
SELECT ename, dname FROM emp LEFT JOIN dept
ON emp.deptId = dept.did;
```

- (3) 右外连接（RIGHT OUTER JOIN）：

```
SELECT ename, dname FROM emp RIGHT JOIN dept
ON emp.deptId = dept.did;
```

- (4) 全连接（FULL JOIN）—— MySQL 不支持

DQL语句（续7）

知识讲解

- UNION 操作符用于将两个查询结果集合并为一个大的结果集。
- 合并结果集，重复数据仅显示一遍：

```
SELECT ename FROM emp_cn
UNION
SELECT ename FROM emp_us;
```

- 合并结果集，允许出现重复数据：

```
SELECT ename FROM emp_cn
UNION ALL
SELECT ename FROM emp_us;
```

常用SQL函数

知识讲解

- MySQL 中还提供了丰富的工具函数，可以实现日期时间、字符串等数据的处理，以及常用数学函数、加密、流控制函数等。

函数名	含义	函数名	含义
CEILING( )	数字上取整	CURDATE( )	当前系统日前
FLOOR( )	数字下取整	NOW( )	当前系统日期时间
ROUND( )	数字四舍五入	DATE_ADD( )	日期相加
UPPER( )	字符串转换为大写	YEAR( )	获取年份
LOWER( )	字符串转换为小写	PASSWORD( )	字符串加密
TRIM( )	裁剪空白字符	IF...THEN...END IF	条件判定
LAST_INSERT_ID( )	最近生成的自增编号	LOOP...END LOOP	循环执行

存储过程

知识讲解

- 在关系型数据库管理系统中，为了完成特定功能，可能需要连续的执行若干条 SQL 语句，如：
  - 用户登录：需要执行多条查询语句和更新语句；
  - 删除员工：需要删除员工信息、考勤信息，修改部门信息、资产信息；
  - 添加购物车：需要查询用户、查询/创建购物车、添加购物车内容；

- “存储过程（Stored Procedure）”就是将某一功能所需的一组 SQL 语句存储在数据库服务器中，经过第一次编译后再次调用无需再次编译，用户通过指定存储过程的名字并给出参数来执行它。

存储过程（续1）

知识讲解

- 声明存储过程：

```
DELIMITER   //
CREATE PROCEDURE insertUser(
        IN uname VARCHAR(32),
        IN upwd VARCHAR(32),
        OUT result INT )
BEGIN
        INSERT INTO users VALUES( NULL, uname, upwd);
        SELECT last_insert_id() INTO result;
END //
DELIMITER   ;
```

- 调用存储过程：

```
CALL insertUser( 'tom', '123', @uid );
SELECT @uid;
```

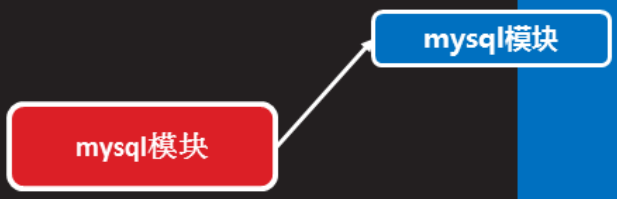
Technology Tarena 达内科技

实现数据库操作

- 使用 DQL 语句
- 使用函数
- 使用存储过程

课堂练习

mysql模块



- mysql模块概述
- 创建数据库连接
- 使用数据库连接池
- 提交SQL语句
- ? 占位符
- 获取自增 ID
- 获取影响的行数
- 执行多条语句
- 调用存储过程

mysql模块

mysql模块概述

- Node.js 官方目前没有提供任何数据库的访问模块，必须使用 NPM 下载第三方模块来实现 MySQL/MariaDB 数据库的访问
  - 可以在 npmjs.org 上搜索 mysql或mariadb，可以看到数千条结果，使用 npm 工具下载需要的即可

知识讲解

```
npm  install  mysql
//mysql 目录模块会保存在当前目录下的 node_modules 子目录下
```

创建数据库连接

Technology Tarena 达内科技

- 使用 mysql 模块的 createConnection 方法可以创建到 MySQL 服务器的连接

```
const mysql= require( 'mysql' );
var conn = mysql.createConnection( options );
```

知识讲解

- 其中的 options 参数是一个对象，可以声明如下属性：
  - host：服务器域名或 IP 地址；
  - port：端口号
  - user：用户名
  - password：密码
  - database：数据库名
  - charset：连接所用字符集

使用数据库连接池

Technology Tarena 达内科技

- 为了提高数据库访问效率，可以在应用启动时一次性的创建多个数据库连接，保存在一个集合中，随用随取——这样的数据库连接对象的集合称为“连接池”。

知识讲解

```
var pool  = mysql.createPool({
        host            : '127.0.0.1',
        user            : 'root',
        password        : 'root_pwd',
        database        : 'db_name',
        connectionLimit : 10
});
pool.getConnection(function(err, conn) {
        connection.query('...', function(err, result) {
                connection.release();          //释放连接
        });
});
```

提交SQL语句

Tarena Technology 达内科技

- 创建连接后，调用 query() 方法执行数据访问

知识讲解

```
var mysql      = require('mysql');
var connection = mysql.createConnection({
          host     : '127.0.0.1',
          user     : 'root',
          password : 'root_pwd',
          database : 'db_name'
});
connection.connect();
connection.query('SELECT 1 + 1 AS solution',
          (err, rows, fields)=> {
                    if (err) throw err;
                    console.log('The solution is: ', rows[0].solution);
});
connection.end();
```

? 占位符

Tarena Technology 达内科技

- 为了防止 SQL 注入漏洞，可以使用 ? 占位符，填充 SQL 语句中的变量：

知识讲解

```
connection.query('UPDATE users SET foo = ?, bar = ?,
baz = ? WHERE id = ?',
['a', 'b', 'c', userId],
function(err, results) {
          // ...
});
```

```
var article  = {id: 1, title: 'Hello MySQL'};
var query = connection.query('INSERT INTO posts SET ?',
article, function(err, result) {
          // ...
});
```

获取自增ID

Technology Tarena 达内科技

- 若连接上刚刚执行的是一条 INSERT 语句，且产生了自增主键值，可以使用如下方法获得该值：

知识讲解

```
connection.query('INSERT INTO posts SET ?',
        {title: 'test'}, function(err, result) {
                if (err) throw err;
                console.log(result.insertId);
});
```

获取影响的行数

Technology Tarena 达内科技

- 若执行的是一条 DML（INERT、DELETE、UPDATE）语句，可以使用如下方法获取该语句影响的行数

知识讲解

```
connection.query(
        'DELETE FROM posts WHERE title = "wrong"',
        function (err, result) {
                if (err) throw err;
                console.log('deleted ' +
                        result.affectedRows + ' rows');
})
```

```
connection.query('UPDATE posts SET ...', function (err, result) {
        if (err) throw err;
        console.log('changed ' +
                        result.changedRows + ' rows');
})
```

执行多条语句

Technology Tarena 达内科技

- 创建连接时，声明 multipleStatements 属性为 true，可以同时执行多条SQL 语句

知识讲解

```
var conn  = mysql.createConnection({
        host              : '127.0.0.1',
        user              : 'root',
        password          : 'root_pwd',
        database          : 'db_name',
        multipleStatements              : true
});
conn.connect(()=>{
        conn.query( 'INSERT ... ; INSERT ... ', ()=>{
                conn.query( 'UPDATE... ', ()=>{})
        })
});
```

调用存储过程

Technology Tarena 达内科技

- 创建连接时，声明 multipleStatements 属性为 true，可以执行存储过程

知识讲解

```
var conn  = mysql.createConnection({
        host              : '127.0.0.1',
        user              : 'root',
        password          : 'root_pwd',
        database          : 'db_name',
        multipleStatements              : true
});
conn.connect(()=>{
        var sql = "CALL insertUser( 'tom', '123', @result );
                        SELECT @result";
        conn.query( sql, ()=>{})
});
```

Technology Tarena 达内科技

使用 mysql 模块

- 使用 mysql 模块

课堂练习

Technology Tarena 达内科技

总结和答疑