MySQL 案例实战教程

案例1：MySQL的安装和基本使用

安装需要设置编码为 UTF-8， 管理用户root,密码设置 root 或 123456

案例2：MySQL的数据类型

> - MySQL支持多种类型，大致可以分为三类：数值、日期/时间和字符串(字符)类型。
> - 备注: char 和varchar 一定要指定长度，float 会自动提升为double，timestamp 是时间的混合类型，理论上可以存储 时间格式和时间戳。

类型	用途
int	整型，相当于java 的int
bigint	整型，相当于java 的 long
float	浮点型
double	浮点型
datetime	日期类型
timestamp	日期类型（可存储时间戳）
char	定长字符
varchar	不定长字符
text	大文本，用于存储很长的字符内容
blob	字节数据类型，存储图片、音频等文件

案例3：建表操作

- 语法

```
-- 删除表
DROP TABLE IF EXISTS 表名;
-- 新建表
create table 表名(
    字段名 类型 约束（主键，非空，唯一，默认值），
    字段名 类型 约束（主键，非空，唯一，默认值），
)编码，存储引擎;
```

> 在 SQL 中，我们有如下约束：

> - **NOT NULL** - 指示某列不能存储 NULL 值。
> - **UNIQUE** - 保证某列的每行必须有唯一的值。
> - **PRIMARY KEY** - NOT NULL 和 UNIQUE 的结合。确保某列（或两个列多个列的结合）有唯一标识，有助于更容易更快速地找到表中的一个特定的记录。
> - **FOREIGN KEY** - 保证一个表中的数据匹配另一个表中的值的参照完整性。
> - **CHECK** - 保证列中的值符合指定的条件。
> - **DEFAULT** - 规定没有给列赋值时的默认值。

- 实例

```
DROP TABLE IF EXISTS `websites`;
CREATE TABLE `websites` (
    id int(11) NOT NULL AUTO_INCREMENT,
    name char(20) NOT NULL DEFAULT '' COMMENT '站点名称',
    url varchar(255) NOT NULL DEFAULT '',
    alexa int(11) NOT NULL DEFAULT '0' COMMENT 'Alexa 排名',
    sal double COMMENT '广告收入',
    country char(10) NOT NULL DEFAULT '' COMMENT '国家',
    PRIMARY KEY (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

案例4：插入、删除、更新

- 插入语句

```
INSERT INTO websites(name, url,alexa,sal,country ) VALUES ('腾讯', 'https://www.qq.com',
18, 1000,'CN' ) ;
```

- 删除语句

```
delete from websites where id = 5;
```

- 更新语句

```
update websites set sal = null where id = 3
```

案例5：基本 select 查询语句

- 初始化数据

```
DROP TABLE IF EXISTS `websites`;
CREATE TABLE `websites` (
    id int(11) NOT NULL AUTO_INCREMENT,
    name char(20) NOT NULL DEFAULT '' COMMENT '站点名称',
    url varchar(255) NOT NULL DEFAULT '',
    alexa int(11) NOT NULL DEFAULT '0' COMMENT 'Alexa 排名',
    sal double COMMENT '广告收入',
```

```
    country char(10) NOT NULL DEFAULT '' COMMENT '国家',
    PRIMARY KEY (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

INSERT INTO `websites` VALUES
    (1, 'Google', 'https://www.google.cm/', '1', 2000,'USA'),
    (2, '淘宝', 'https://www.taobao.com/', '13',2050, 'CN'),
    (3, '菜鸟教程', 'http://www.runoob.com/', '4689',0.0001, 'CN'),
    (4, '微博', 'http://weibo.com/', '20',50, 'CN'),
    (5, 'Facebook', 'https://www.facebook.com/', '3', 500,'USA');

CREATE TABLE IF NOT EXISTS `access_log` (
    `aid` int(11) NOT NULL AUTO_INCREMENT,
    `site_id` int(11) NOT NULL DEFAULT '0' COMMENT '网站id',
    `count` int(11) NOT NULL DEFAULT '0' COMMENT '访问次数',
    `date` date NOT NULL,
    PRIMARY KEY (`aid`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8;
INSERT INTO `access_log` (`aid`, `site_id`, `count`, `date`) VALUES
    (1, 1, 45, '2016-05-10'),
    (2, 3, 100, '2016-05-13'),
    (3, 1, 230, '2016-05-14'),
    (4, 2, 10, '2016-05-14'),
    (5, 5, 205, '2016-05-14'),
    (6, 4, 13, '2016-05-15'),
    (7, 3, 220, '2016-05-15'),
    (8, 5, 545, '2016-05-16'),
    (9, 3, 201, '2016-05-17'),
    (10, 88, 9999, '2016-09-09');
```

- 查询语句

```
select * from websites

select id, name, url, alexa, sal, country from websites  (推荐使用的方式)
```

案例6. 分页查询

> mysql 的分页是最优雅

```
select * from websites limit 2,3 ; --从第2条(下标从0开始)开始查，查3条数据
select * from websites limit 3 ; --从第0条(下标从0开始)开始查，查3条数据
```

案例7. distinct 关键字

> DISTINCT 关键词用于返回唯一不同的值。

```
select distinct country from websites
```

案例8. where 语句

> 作为条件筛选, 运算符: > < >= <= <> != =
>
> is null is not null (因为在sql 语句中null 和任何东西比较都是假，包括它本身)
>
> like in

```
select * from websites where sal >500
```

案例9. 逻辑条件: and 、 or

```
select * from websites where sal >= 0 and sal <=2000 ; -- 收入在 0 到 2000 之间

select * from websites where sal between 0 and 2000;  -- 和上面一样的，没事找事

select * from websites where sal < 5 or sal is null ; -- 收入小于5 或者没收入
```

> 注意： null 的条件判断用 is null 或 is not null

案例10. order by

排序: 默认情况下是升序，asc 可以省略 。

```
select * from websites order by sal asc,alexa desc ;--先根据sal 升序排序，再根据 alexa 降序
```

案例11. like 和 通配符

> - like 模糊查询
> - 通配符
>   - `%` ： 0个或多个字符
>   - `_`: 1 个字符

案例12. in

- 匹配多个条件

  ```
  select * from websites where country in ('USA','鸟国','CN');
  ```

- 等价于

  ```
  select * from websites where country = 'USA' or country = '鸟国' or country = 'CN'
  ```

案例13. 别名

```
select tt.name '网站名字' from websites tt
```

案例14. Group by 分组查询

> 注意：分组时候的筛选用 having
>
> 常见的几个组函数： max() min() avg() count() sum()

```
select avg(sal) aa from websites where sal is not null group by country having aa > 1500
```

案例15. 子查询

> 把查询的结果当作一个表来使用

案例16. 连接查询

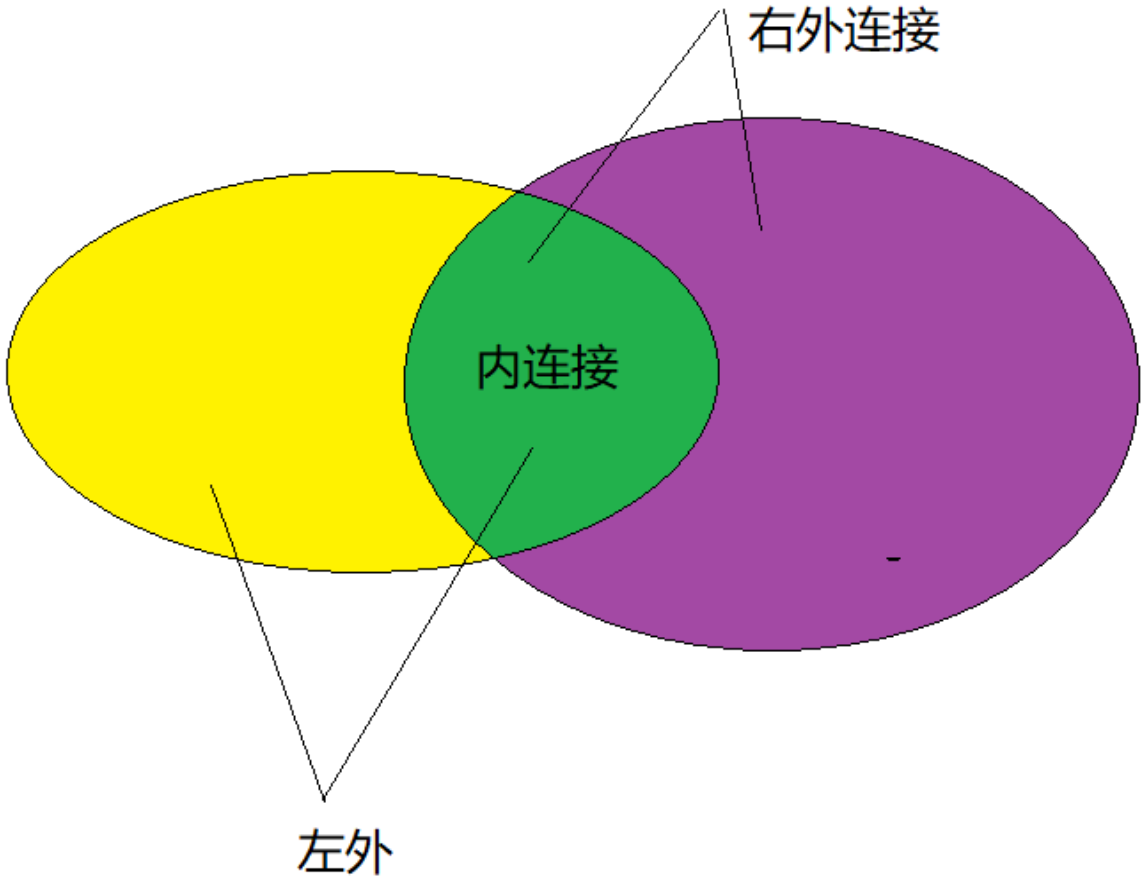
```
select name,count,date from websites w , access_log a ; --著名的笛卡尔积，没什么意义的

select name,count,date from websites w , access_log a where w.id = a.site_id; -- 这是 1992 的语法

select name,count,date from websites w inner join access_log a on w.id = a.site_id; -- 这是 1999 年的语法，推荐使用

select name,count,date from websites w left outer join access_log a on w.id = a.site_id; -- 把没有访问的网站也显示出来
-- 注意：inner 和 outer 是可以默认省略的。
```

- 如图：



- 查询结果

案例17. Null 处理I 函数

```
select name,ifnull(count,0),ifnull(date,'') from websites w left outer join access_log a on
w.id = a.site_id
```

经典练习(Oracle 的自带练习)

- 员工信息表

```
-- 员工信息表
CREATE TABLE emp(
    empno INT,
    ename VARCHAR(50),
    job VARCHAR(50),
    mgr INT,
```

```
    hiredate DATE,
    sal DECIMAL(7,2),
    comm DECIMAL(7,2),
    deptno INT
) ;

INSERT INTO emp values(7369,'SMITH','CLERK',7902,'1980-12-17',800,NULL,20);
INSERT INTO emp values(7499,'ALLEN','SALESMAN',7698,'1981-02-20',1600,300,30);
INSERT INTO emp values(7521,'WARD','SALESMAN',7698,'1981-02-22',1250,500,30);
INSERT INTO emp values(7566,'JONES','MANAGER',7839,'1981-04-02',2975,NULL,20);
INSERT INTO emp values(7654,'MARTIN','SALESMAN',7698,'1981-09-28',1250,1400,30);
INSERT INTO emp values(7698,'BLAKE','MANAGER',7839,'1981-05-01',2850,NULL,30);
INSERT INTO emp values(7782,'CLARK','MANAGER',7839,'1981-06-09',2450,NULL,10);
INSERT INTO emp values(7788,'SCOTT','ANALYST',7566,'1987-04-19',3000,NULL,20);
INSERT INTO emp values(7839,'KING','PRESIDENT',NULL,'1981-11-17',5000,NULL,10);
INSERT INTO emp values(7844,'TURNER','SALESMAN',7698,'1981-09-08',1500,0,30);
INSERT INTO emp values(7876,'ADAMS','CLERK',7788,'1987-05-23',1100,NULL,20);
INSERT INTO emp values(7900,'JAMES','CLERK',7698,'1981-12-03',950,NULL,30);
INSERT INTO emp values(7902,'FORD','ANALYST',7566,'1981-12-03',3000,NULL,20);
INSERT INTO emp values(7934,'MILLER','CLERK',7782,'1982-01-23',1300,NULL,10);
INSERT INTO emp values(7981,'MILLER','CLERK',7788,'1992-01-23',2600,500,20);
```

- 部门信息表

```
CREATE TABLE dept(
    deptno      INT,
    dname       varchar(14),
    loc     varchar(13)
);

INSERT INTO dept values(10, 'ACCOUNTING', 'NEW YORK');
INSERT INTO dept values(20, 'RESEARCH', 'DALLAS');
INSERT INTO dept values(30, 'SALES', 'CHICAGO');
INSERT INTO dept values(40, 'OPERATIONS', 'BOSTON');
```

- 基本查询

```
--所有员工的信息
--薪资大于等于1000并且小于等于2000的员工信息
--从员工表中查询出所有的部门编号
--查询出名字以A开头的员工的信息
--查询出名字第二个字母是L的员工信息
--查询出没有奖金的员工信息
--所有员工的平均工资
--所有员工的工资总和
--所有员工的数量
--最高工资
--最少工资
--最高工资的员工信息
--最低工资的员工信息
```

- 分组查询

```
--每个部门的平均工资
```

- 子查询

```
-- 单行子查询(> < >= <= = <>)
    -- 查询出高于10号部门的平均工资的员工信息

-- 多行子查询(in  not in any all)    >any  >all
    -- 查询出比10号部门任何员工薪资高的员工信息

-- 多列子查询(实际使用较少)   in
    -- 和10号部门同名同工作的员工信息
-- Select接子查询
    -- 获取员工的名字和部门的名字
-- from后面接子查询
    -- 查询emp表中经理信息
-- where 接子查询
    -- 薪资高于10号部门平均工资的所有员工信息
-- having后面接子查询
    -- 有哪些部门的平均工资高于30号部门的平均工资

-- 工资>JONES工资
-- 查询与SCOTT同一个部门的员工
-- 工资高于30号部门所有人的员工信息
-- 查询工作和工资与MARTIN完全相同的员工信息
-- 有两个以上直接下属的员工信息
-- 查询员工编号为7788的员工名称,员工工资,部门名称,部门地址
```

- SQL查询的综合案例

1. 查询出高于本部门平均工资的员工信息
2. 列出达拉斯加工作的人中,比纽约平均工资高的人
3. 查询7369员工编号,姓名,经理编号和经理姓名
4. 查询出各个部门薪水最高的员工所有信息