浙江大学管理学院
SCHOOL OF MANAGEMENT
ZHEJIANG UNIVERSITY

分布式文件系统及数据库技术
第二讲

主讲人：曹仔科 彭希羡
浙江大学管理学院
数据科学与工程管理系

AACSB ACCREDITED EFMD EQUIS ACCREDITED ASSOCIATION AMBA ACCREDITED CAMEA 中国高质量MBA教育认证 IQA CEEMAN Accredited

1、回顾：关系模型

2、关系代数

3、基础SQL语言（查询）

关系数据模型

- 关系模型是基于数学中的**关系(relation)**的概念

- 关系是由行和列组成的表
  - 列（column）即关系的属性（attribute）
  - 行（row）即关系的元组（tuple）

- 二维表（Table）、关系(Relation)——对象集
  - 行(Row)、记录(Record)——单个对象
  - 列(Col)、字段(Field)——对象的属性

关系的码(key)

浙江大学管理学院 SCHOOL OF MANAGEMENT ZHEJIANG UNIVERSITY

- 数据库通常有多张表，它们不是孤立的，而是相互联系的。这种表与表之间的联系，是“关系”概念中更关键的部分。

- **主码（主键）：我是谁**；用来唯一标识自己表中的每一行记录
  - **实体完整性**要求自己表里的每条记录都有唯一ID（且不能为空）。

- **外码（外键）：我引用了谁；**用来建立表与表之间的关联，确保引用的对象是真实存在的。
  - **引用完整性**保证表格之间的引用关系是有效的、正确的；外码的值，必须是另一个表（被引用的表）中主码存在的值，或者为空（如果允许的话）。

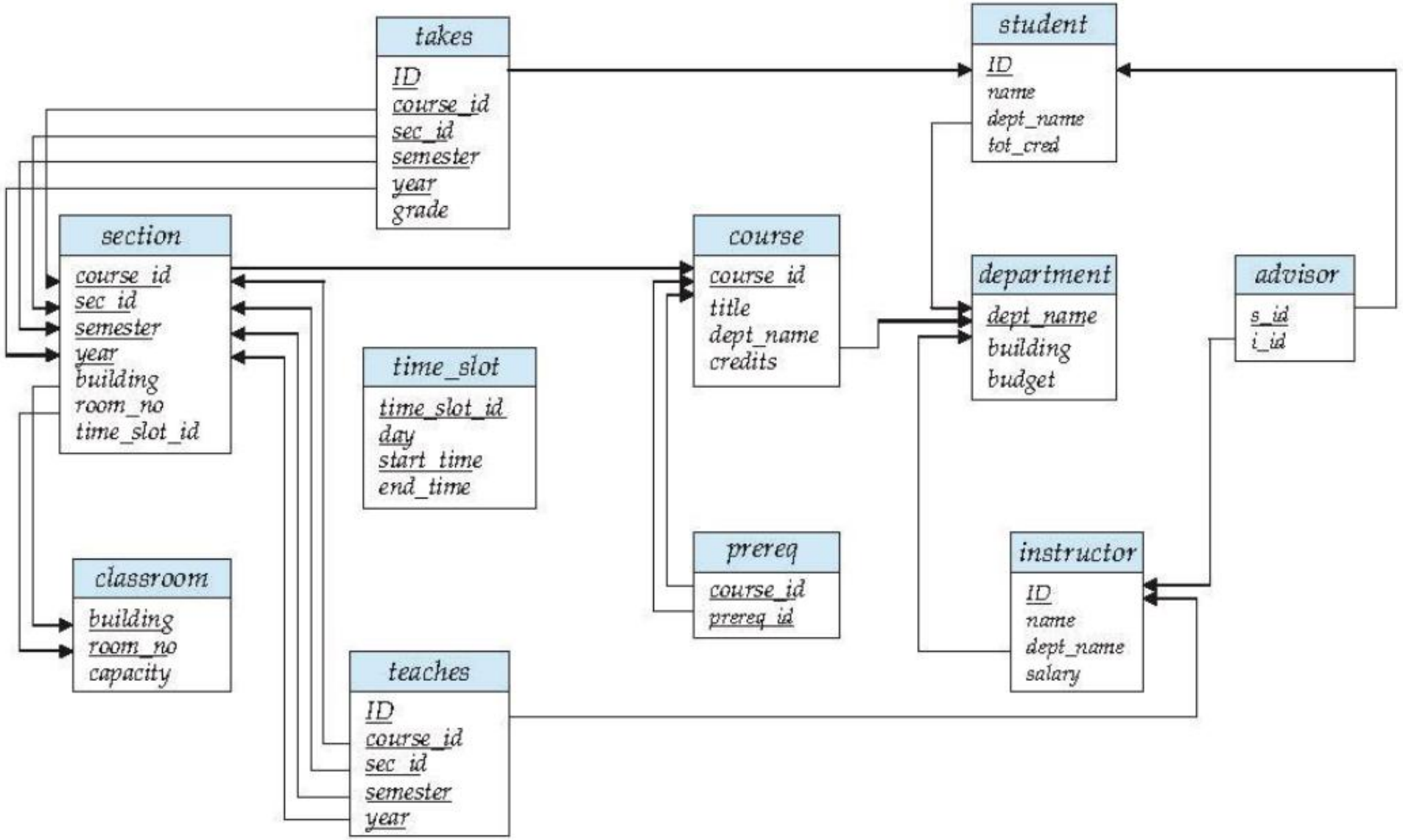
关系的通用约束

- 同一数据库中关系不能重名
- 关系中的每一单元格只含有一个单一值
- 关系的属性不能重名
- 关系的元组不能重复

关系的模式图

- 一个含有主码和外码依赖的数据库模式图可以用模式图(schema diagram)来表示
  - 每一个关系用一个矩形来表示，关系的名字显示在上方
  - 矩形内列出各个属性
  - 主码属性用下划线标注
  - 外码依赖用箭头表示（从引用表指向被引用表）

大学数据库的模式图



浙江大学 管理学院
SCHOOL OF MANAGEMENT
ZHEJIANG UNIVERSITY

关系代数

关系代数

浙江大学 管理学院
SCHOOL OF MANAGEMENT
ZHEJIANG UNIVERSITY

- 关系代数是关系数据模型的正式理论语言
- 关系代数定义的操作可以从一个或多个关系得到另一个关系，而不改变原关系
- 操作数（输入）和操作结果（输出）都是关系，因此一个操作的输出可以作为另一个操作的输入
- 像算术运算一样，关系代数的一个表达式中可以嵌套另一个表达式
  - 这样的性质被称为**封闭性（closure）**
  - 关系在关系代数下是封闭的

关系代数

- 关系代数的五个**基本**运算：
  - 选择 (Selection)
  - 投影 (Projection)
  - 笛卡尔乘积 (Cartesian product)
  - 集合并 (Set Union)
  - 集合差 (Set Difference)

- 五个基本运算可以实现大多数数据检索需求

- 从五个基本运算中还可以推出其他常用的运算：
  - 连接 (Join)
  - 集合交 (Set Intersection)
  - 除法 (Division)

选择

- $\sigma_{predicate}(R)$
  - 作用于单个关系R，得到的新关系由R中满足特定条件（由谓词predicate界定）的元组（行）组成
- 查找出表“员工”中工资大于20000的员工：
  - $\sigma_{工资 > 20000}(员工)$
- 可以运用逻辑运算与、或、非来定义更复杂的条件

投影

- $\Pi_{col1, \ldots, coln}(R)$
  - 作用于单个关系R，得到由R的一个垂直子集构成的新关系，该关系抽取R中的指定属性集的所有值并去除重复元组

- 从“员工”表创造仅显示员工号、姓名和工资的表：
  - $\Pi_{员工号, 姓名, 工资}(员工)$

集合并（Union）

- **R ∪ S**
  - 两个关系的R和S并，定义一个包含R和S中所有不同元组的新关系（即重复元组被剔除）
  - R和S必须是**并相容(union-compatible)**: 两个关系的模式完全匹配，即它们有同样多的属性，并且对应属性有相同的域（但注意属性名称并不要求相同）

集合差（Difference）

- **R – S**
  - 集合差运算定义一个新的关系，它由所有属于R但不属于S的元组构成
  - R和S必须是并相容(union-compatible)

集合交（Intersection）

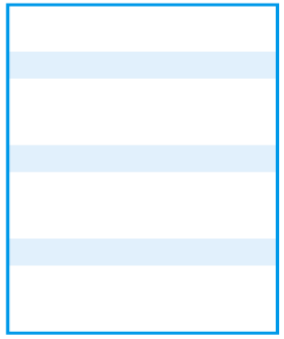
- **R ∩ S**
  - 交运算定义的新关系由既属于R又属于S的元组组成
  - R和S必须是并相容(union-compatible)

- **R ∩ S = R – (R – S)**

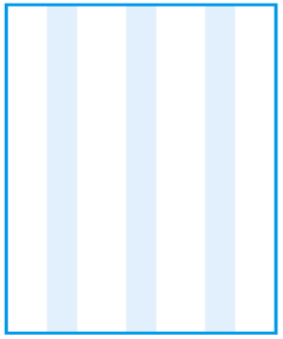
笛卡尔乘积

- **R X S**
  - 笛卡尔乘积得到一个新的关系，它是关系R中每个元组与关系S中每个元组并联的结果
  - 对两个表（比如表A和表B）进行运算，结果是一个新表。这个新表由A表的每一行依次和B表的每一行“手拉手”配对后形成的所有可能的行组合。

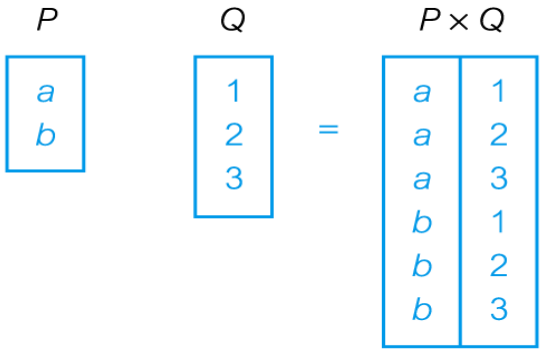
关系代数运算



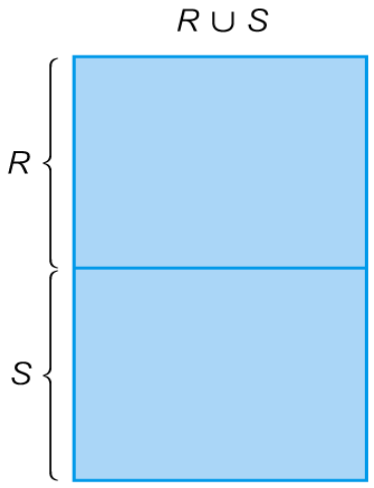
(a) Selection



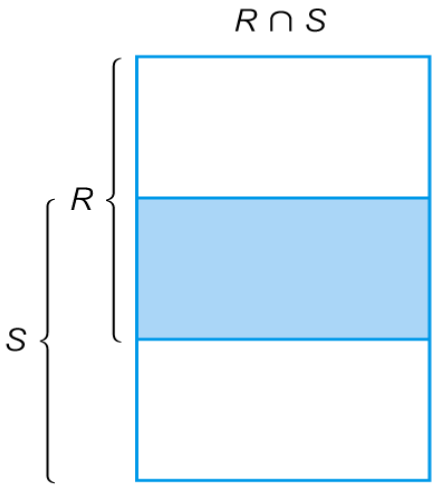
(b) Projection



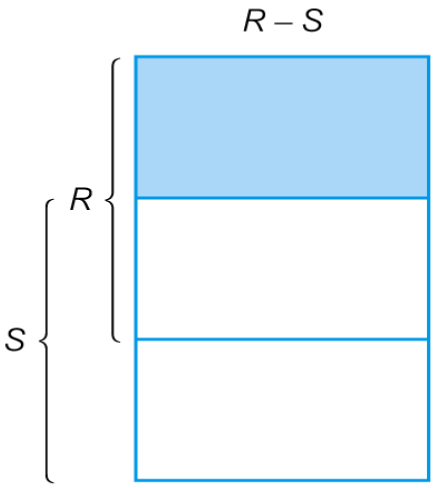
(c) Cartesian product



(d) Union



(e) Intersection



(f) Set difference

连接运算（Join）

- 连接运算
  - 由笛卡尔积运算的衍生出来的
  - 用连接谓词（predicate）对参与运算关系的笛卡尔乘积执行一次选择运算
  - 比笛卡尔积运算更常用

四种常见的连接运算（Join）

- 不同类型的连接运算：
  - 内连接（INNER JOIN）
  - 左外连接 （LEFT JOIN）
  - 右外连接（RIGHT JOIN）
  - 全外连接（FULL JOIN）

四种常见的连接运算（Join）

浙江大学 管理学院 SCHOOL OF MANAGEMENT ZHEJIANG UNIVERSITY

- 假设有两张表“员工”和“部门”：

员工ID	姓名	部门ID	薪水
1	张三	101	5000
2	李四	102	6000
3	王五	101	5500
4	赵六	NULL	4500

- **主键**: 员工ID
- **外键**: 部门ID引用于 部门表的部门ID

部门ID	部门名称	所在地
101	销售部	北京
102	技术部	上海
103	人事部	广州

- **主键**: 部门ID

四种常见的连接运算（Join）

浙江大学 管理学院 SCHOOL OF MANAGEMENT ZHEJIANG UNIVERSITY

- 内连接（INNER JOIN）：只返回两个表中匹配成功的记录。
  - 找出“有员工的部门”和“有所属部门的员工”的完整信息。

员工ID	姓名	部门ID	薪水	部门ID	部门名称	所在地
1	张三	101	5000	101	销售部	北京
2	李四	102	6000	102	技术部	上海
3	王五	101	5500	101	销售部	北京

四种常见的连接运算（Join）

- 左外连接（LEFT JOIN）：返回左表（员工表）的全部记录，以及右表（部门表）中匹配的记录。如果右表无匹配，则结果为NULL。
  - 列出所有员工，并尽可能找到他们所在的部门信息。找不到部门的，部门信息就空着。

员工ID	姓名	部门ID	薪水	部门ID	部门名称	所在地
1	张三	101	5000	101	销售部	北京
2	李四	102	6000	102	技术部	上海
3	王五	101	5500	101	销售部	北京
4	赵六	NULL	4500	NULL	NULL	NULL

四种常见的连接运算（Join）

- 右外连接（RIGHT JOIN）：返回右表（部门表）的全部记录，以及左表（员工表）中匹配的记录。如果左表无匹配，则结果为NULL。
  - 列出所有部门，并看看有哪些员工在这些部门里。找不到员工的部门，员工信息就空着。

员工ID	姓名	部门ID	薪水	部门ID	部门名称	所在地
1	张三	101	5000	101	销售部	北京
3	王五	101	5500	101	销售部	北京
2	李四	102	6000	102	技术部	上海
NULL	NULL	NULL	NULL	103	人事部	广州

四种常见的连接运算（Join）

- 全外连接（FULL JOIN）：返回左表和右表中的所有记录。当某一行在另一个表中没有匹配行时，另一个表的结果为NULL。
  - 把能关联上的信息都列出来。关联不上的，不管是员工找不到部门，还是部门找不到员工，也都全部列出来，空着就行了。

员工ID	姓名	部门ID	薪水	部门ID	部门名称	所在地
1	张三	101	5000	101	销售部	北京
2	李四	102	6000	102	技术部	上海
3	王五	101	5500	101	销售部	北京
4	赵六	NULL	4500	NULL	NULL	NULL
NULL	NULL	NULL	NULL	103	人事部	广州

聚集运算

- 当希望对数据进行汇总时（类似于报表底部的合计），需要使用聚集运算

- 主要的聚集运算：
  - COUNT：返回相关联属性值的个数
  - SUM：返回相关联属性值的总和
  - AVG：返回相关联属性值的平均值
  - MIN：返回相关联属性值的最小值
  - MAX：返回相关联属性值的最大值

分组运算

- 对数据进行分组处理也是常见和强大的功能。

- 对关系R分组运算即将R的元组根据分组属性分割成不同的组，然后对分组元组执行聚集运算来得到新的关系。

浙江大学 管理学院
SCHOOL OF MANAGEMENT
ZHEJIANG UNIVERSITY

基础SQL语言

数据库语言

- 一种数据库语言应该允许用户：
  - 建立数据库和关系结构
  - 实现数据插入、修改和删除
  - 实现简单或复杂的查询
- 功能丰富、语法简洁、易学易用
- 具有可移植性（portable）

- **结构化查询语言 (Structured Query Language, SQL)** 是满足这些要求的最常用的数据库语言

SQL

- SQL是一种**面向转换**的语言
  - 它只关心将输入关系转换成所需要的输出关系
  - 非过程化：用户只需告诉需要，不需关心**如何**得到所需结果的过程
  - 由标准的英语单词组成（如SELECT, INSERT, CREATE等）

- SQL分为两大部分
  - 数据定义语言（DDL）：用于定义数据库结构和数据的访问控制
  - 数据操作语言（DML）：用于查询和更改数据

SQL标准

- SQL已经被ISO制定标准
- SQL也成为名义上和实际使用中关系数据库的标准语言
- 各种用户都可以使用SQL：数据库系统管理员，程序开发员，以及普通（熟练）用户
- 可以读成‘see-quel’，但官方读法其实是‘S-Q-L’.

SQL书写规则

- SQL语句包括**保留字**和**用户自定义字**
  - 保留字是SQL的固定部分，必须准确拼写，并且不能跨行拼写
  - 用户自定义字由用户根据一定的语法规则自己定义，用来表示关系名、列名、视图名等

SQL书写规则

- SQL语句大多数情况下是**不区分大小写（case-insensitive)**
  - 唯一的例外：字符数据常量是区分大小写的（如，某人的姓以‘ZHANG’存储，如以‘Zhang’查询是不能匹配的）

- 为了使SQL语句更具可读性，一般也采用如下一些规则（非必须）：
  - 每一个子句另起一行书写
  - 每一子句与其他子句的开始字符处于同一列
  - 如果一个子句由几个部分组成，分写在不同行，并使用缩进表明这种关系

常量（Literals）

- 常量是指SQL语言使用的**不变量**
- 不同的数据类型有不同形式的常量形式（以后会详细讲）
- 所有非数值型的常量**必须用**单引号引起来
  - 如，’ 张三’ ； ‘李四’ ； ‘Hangzhou’
- 所有数值型的常量**一定不能用**单引号引起来
  - 如， 身高（170） ； 体重（65）

浙江大学 管理学院
SCHOOL OF MANAGEMENT
ZHEJIANG UNIVERSITY

SQL数据定义

- **CREATE DATABASE**
  - 创建数据库
- **CREATE TABLE**
  - 创建表（关系）

浙江大学 管理学院
SCHOOL OF MANAGEMENT
ZHEJIANG UNIVERSITY

SQL数据操作

- **INSERT**
  - 插入新数据
- **SELECT**
  - 查询数据
- **UPDATE**
  - 更新数据
- **DELETE**
  - 删除数据

创建数据库

- **CREATE DATABASE school_db;**
- **USE school_db;**

- 创建数据库：school_db
- 使用该数据库，以便在该数据库上进行后续操作

创建表

```
CREATE TABLE Students (
        student_id INT IDENTITY(1,1) PRIMARY KEY,
        name NVARCHAR(50) NOT NULL,
        age INT,
        gender NCHAR(1),
        enrollment_date DATE
);
```

- 创建记录学生信息表Students
- IDENTITY(seed, increment)：主键从起始数字开始自动增加一个步长

创建表

```
CREATE TABLE Courses (
    course_id INT IDENTITY(1,1)  PRIMARY KEY,
    course_name NVARCHAR(100) NOT NULL,
    teacher NVARCHAR(50),
    credit INT
);
```

- 创建记录课程信息表Courses

创建表及设置外键

```
CREATE TABLE Takes (
    student_id INT,
    course_id INT,
    grade INT,
    CONSTRAINT FK_takes_student
        FOREIGN KEY (student_id) REFERENCES Students(student_id),
    CONSTRAINT FK_takes_course
        FOREIGN KEY (course_id) REFERENCES Courses(course_id)
);
```

- 创建记录选课信息表Takes
- 设置适当的外键，即引用约束

数据插入 – 简单INSERT语句

INSERT INTO TableName [(columnList)]
VALUES (dataValueList)

- 列名columnList可以忽略。如果忽略，SQL会严格按照表格创建时的顺序
- columnList和dataValueList数目必须相同，各项必须直接对应
- 数据类型必须一致

INSERT语句

```
INSERT INTO Students (name, age, gender, enrollment_date)
VALUES
('张三', 18, 'M', '20230901'),
('李四', 17, 'F', '2023-09-01'),
('王五', 19, 'M', '2023-09-02');
```

- 向学生表中插入三行记录
- 注意省略了student_id列的值，因为创建表是设置了主键自增：identity(1,1)

INSERT语句

```
INSERT INTO Courses (course_name, teacher, credit)
VALUES
('数据库原理', '张教授', 3),
('Web开发', '李老师', 4),
('数据结构', '王教授', 3);
```

- 向课程表中插入三行记录
- 注意省略了course_id列的值，因为创建表是设置了主键自增：identity(1,1)

数据查询 – SELECT语句

```
SELECT [DISTINCT | ALL]
  {* | [columnExpression [AS newName]] [,...] }
FROM       TableName [alias] [, ...]
[WHERE     condition]
[GROUP BY  columnList]  [HAVING     condition]
[ORDER BY  columnList]
```

扩展的巴克斯范式（BNF notation）

- 定义SQL语句时采用扩展的BNF范式：
  - 大写字母用于表示保留字，必须准确拼写
  - 小写字母用户表示用户自定义字
  - 竖线（|）表示从选项中选择，如 a|b|c 代表从a，b，c中选择一个
  - 大括号 {} 代表**必须元素**
  - 中括号 [] 代表**可选元素**（非必须元素）
  - 省略号 … 代表某一个元素**可选择重复**零到多次

SELECT语句

查询出表中所有的记录：

```
SELECT student_id, name, age, gender, enrollment_date
FROM Students;
```

可以使用通配符“*”来代替所有列

```
SELECT *
FROM Students;
```

SELECT语句：使用DISTINCT去重

- 查询结果默认保留所有记录（ALL）：

```
SELECT credit
FROM Courses;
```

- 可以使用DISTINCT来去除重复元组：

```
SELECT DISTINCT credit
FROM Courses;
```

SELECT语句：返回结果进行字段计算

```
SELECT  student_id,  name,  age+4,  gender,
  enrollment_date

FROM Students;
```

- 返回结果对age的值都加4

SELECT语句：字段计算

- 加减乘除运算都可以出现在表达式中
- 可以用括号来建立更复杂的表达式
- 计算可以涉及字段
- 算数表达式所引用的列必须是数字类型

SELECT语句：使用列别名

- SELECT 列名 AS 别名 FROM ...;

- **作用**：给列或计算结果起一个更易读的名字。

```
SELECT student_id AS 学号, name AS 名字, age AS 年龄,
  gender AS 性别, enrollment_date AS 注册日期
FROM Students;
```

- AS也可以省略。

上机操作

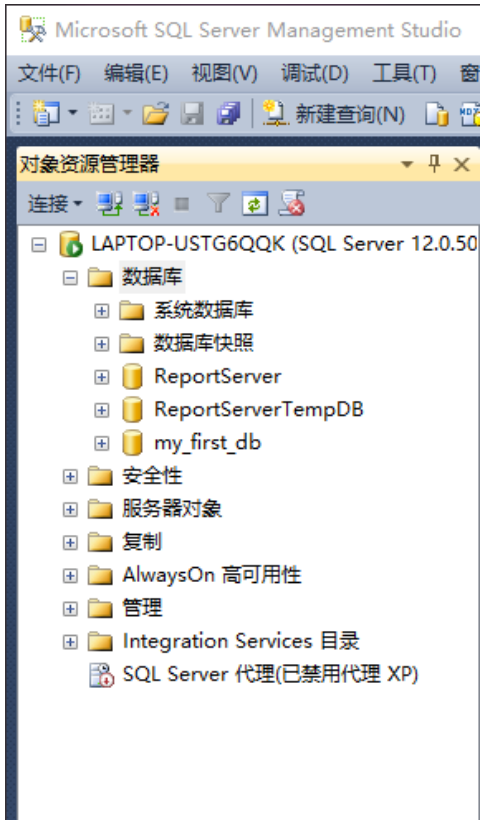
- Microsoft SQL Server
  - 创建数据库
  - 创建关系
  - 简单查询

- 习题+上机作业见“学在浙大” （习题+上机作业2）

SQL Server创建数据库

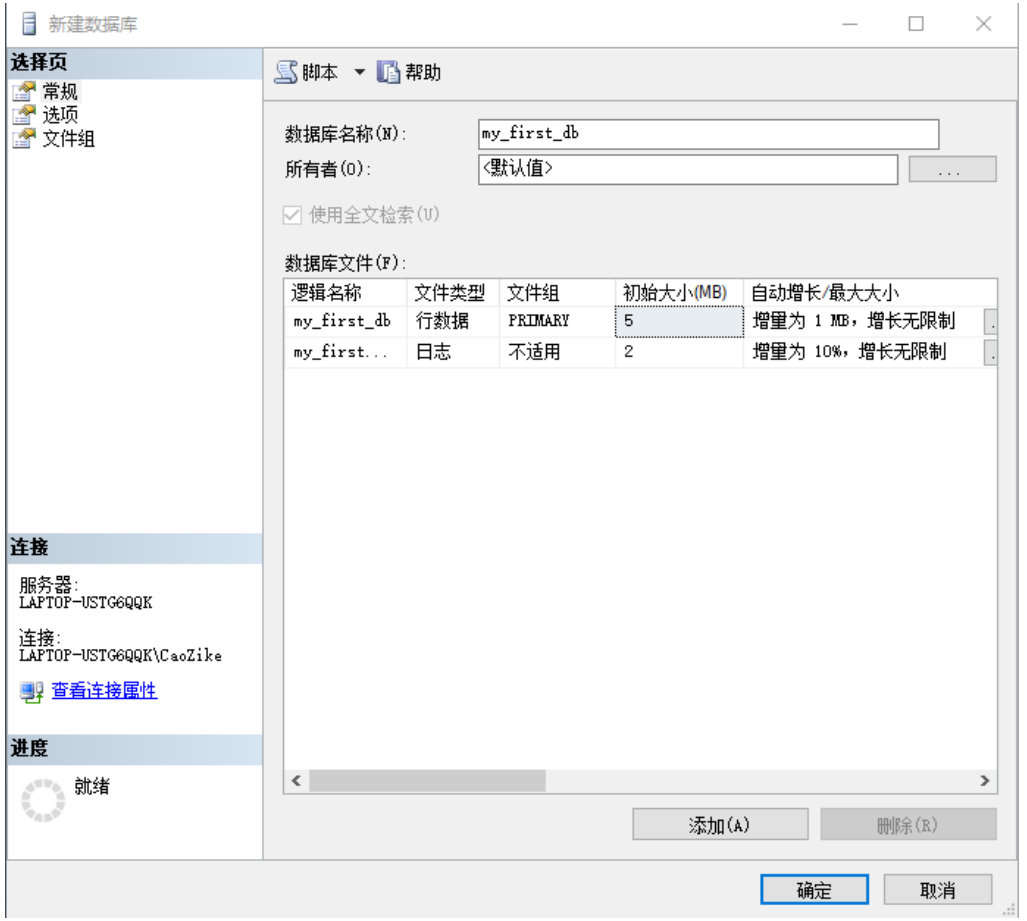
- 数据库连接成功后，找到左侧【对象资源管理器】中的【数据库】节点

- 右击【数据库】节点，选择【新建数据库】



SQL Server创建数据库

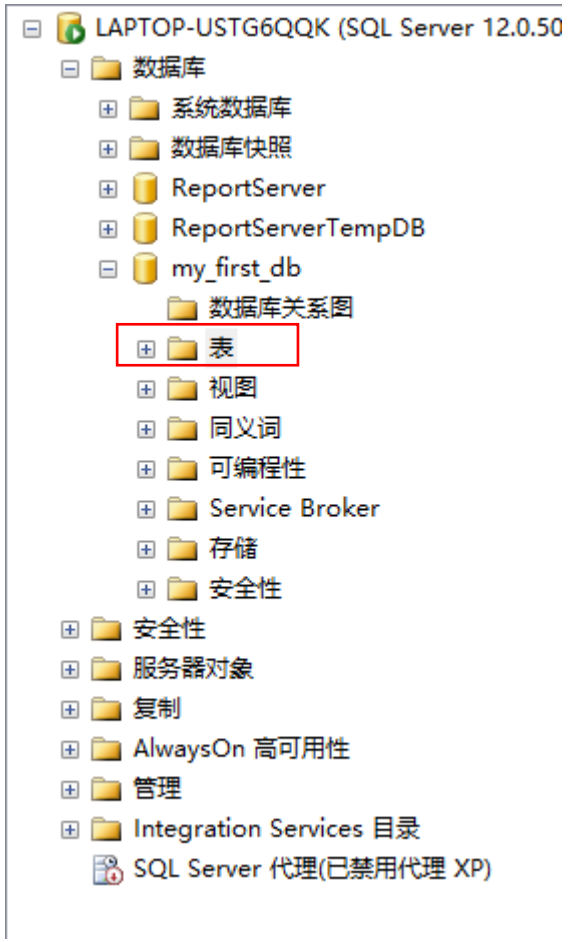
- 给数据库取名
- （暂时）接受所有默认设置
- 点击【确定】，创建新的数据库



SQL Server创建关系（表）

- 在左侧【对象资源管理器】中的【数据库】节点找到新创建的数据库并展开

- 右击【表】节点，选择【新建表】



SQL Server创建关系（表）

- 新建一个表，如右图所示

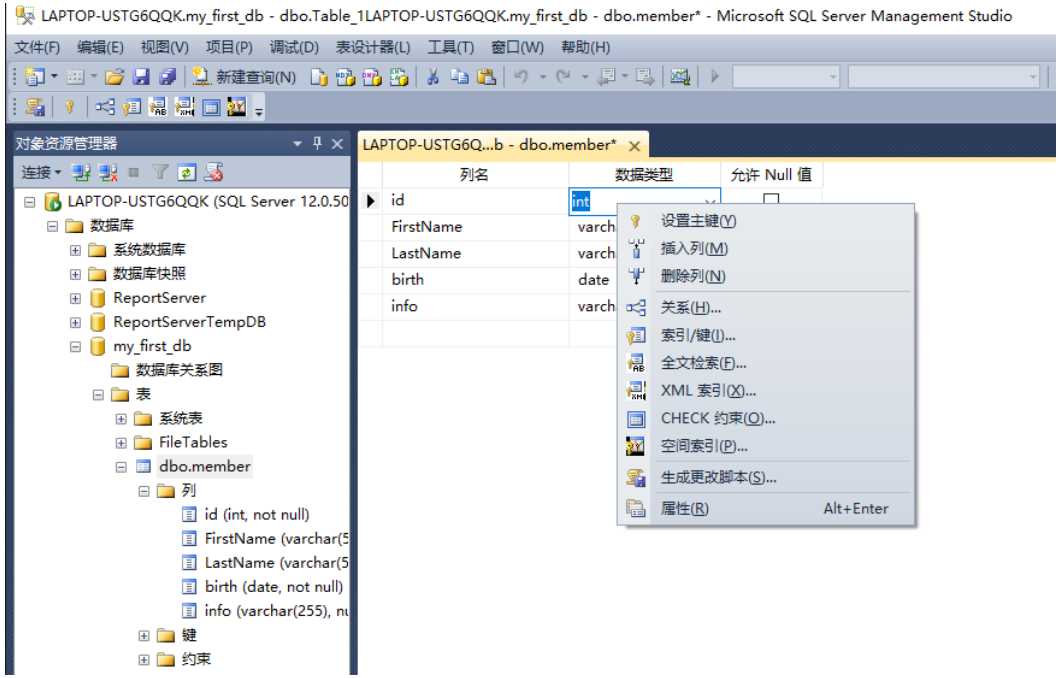
	列名	数据类型	允许 Null 值
	id	int	☐
	FirstName	varchar(50)	☐
	LastName	varchar(50)	☐
▶	birth	date	☐
	info	varchar(255)	☑
			☐

- 选择数据类型和定义是否允许空值
  - 数据类型问题以后会详细讨论
- 设计完成后，单击【保存】，输入表名(member)，单击【确定】

SQL Server添加主码约束

- 在【对象资源管理器】中选择member表节点右击，然后选择【设计】

- 在表设计窗口选择【id】字段对应的行，右击并选择【设置主键】



浙江大学管理学院
SCHOOL OF MANAGEMENT
ZHEJIANG UNIVERSITY

谢谢！

下次见！

AACSB ACCREDITED EFMD EQUIS ACCREDITED ASSOCIATION AMBA ACCREDITED CAMEA 中国高质量MBA教育认证 IQA CEEMAN Accredited