数据库系统原理 (CS307)

讲座 9.1:关系代数

Zhong-Qiu Wang

计算机科学与工程系
南方科技大学

· 大部分内容来自 Stéphane Faroult 和《数据库系统概念（第 7版）》作者制作的幻灯片。 · 他们的原始幻灯片已修改以适应南方科技大学 CS307 的时间表。 · 幻灯片主要基于马宇欣博士提供的幻灯片

关系代数

- 一种过程语言，由一组采用一到两个关系作为输入并产生新的关系作为结果的构成广泛使用的 SQL 查询语言的基础能够以数学方式表达 SQL 语句

- 6 个基本运算符：选择：s
  - 项目：Õ
  - 重命名：r
  - 联盟：它是
  - 设置差异：–
  - 笛卡尔积：x
- 一元（对一个关系进行操作）和二元（对关系对进行操作）

选择操作

- 选择操作选择满足给定谓词的元组
  - Notation: $\sigma_{\text{页}}$（右）

p is called the selection predicate （选择谓词）

- 小写希腊字母 - σ

- 例子
  - 选择讲师关系中的那些元组,其中讲师位于“物理”系

$$\sigma_{\text{dept\_name} = \text{“物理”}}(\text{讲师})$$

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
33456	Gold	Physics	87000

选择操作

- 我们允许在选择谓词中使用 =、¹、>、³、<、£ 进行比较
- 我们可以使用
  连接词：

Ù (和), Ú (或), ¬ (不)

  - 例如，查找薪水高于 90,000 美元的物理学讲师，我们写：

S dept_name = “物理”Ù工资 > 90,000 （讲师）

- 选择谓词可能包括两个属性之间的比较
  - 例如，查找所有名称与建筑物名称相同的部门：

S dept_name=建筑物（部门）

项目运作

- 返回其参数关系的一元运算,具有一定的
  遗漏的属性

符号:Õ A1,A2,A3….Ak (r)

其中A1、 A2、 …、 Ak是属性名称， r是关系名称,Õ 是大写希腊字母 pi

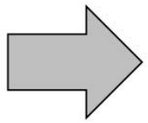
- 结果定义为通过擦除列得到的k列的关系
  未列出的
- 从结果中删除重复的行,因为关系是集合

项目运作

- 示例：消除Instructor 的dept_name属性
  - 询问：

ÕID、姓名、工资（讲师）

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000



ID	*name*	*salary*
10101	Srinivasan	65000
12121	Wu	90000
15151	Mozart	40000
22222	Einstein	95000
32343	El Said	60000
33456	Gold	87000
45565	Katz	75000
58583	Califieri	62000
76543	Singh	80000
76766	Crick	72000
83821	Brandt	92000
98345	Kim	80000

项目运作

- 示例：消除Instructor 的dept_name属性并计算每月平均工资
- 查询：

ÕID、姓名、工资/12 （讲师）

关系运算的组合

- 关系代数运算的结果是关系
  - …因此,关系代数运算可以组合成关系代数表达式

- 示例:查找物理系所有教师的姓名

$$Õname(S\ dept\_name = \text{“物理”}\ (讲师))$$

- 我们不将关系的名称作为投影运算的参数,而是给出一个表达式,该表达式求值为关系

笛卡尔积运算

- 笛卡尔积运算（用叉号 × 表示)允许我们将任意两个关系中的信息组合起来
  - 例如,关系instructor和teaches的笛卡尔积写为：

讲师 × 授课

- 我们从每对可能的元组构建一个结果元组
  - … 一个来自指导员关系,另一个来自教师关系（见下一张幻灯片）
  - 由于讲师 ID 出现在两个关系中,我们通过将属性附加到属性最初来自的关系的名称来区分这些属性
    - tutor.ID 和 teaches.ID
  - 导师×教师关系模式
    - （讲师.ID、讲师.姓名、讲师.部门名称、讲师.薪水、教学.ID、教学.课程 ID、教学.SEC ID、教学.学期、教学.年份）
    - （讲师 ID、姓名、部门名称、工资、教师 ID、课程 ID、学科 ID、学期、年份）

如果不产生歧义,可以删除前缀

“讲师×教学”表

ID	name	dept_name	salary
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

×

course_id	title	dept_name	credits
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

→

instructor.ID	name	dept_name	salary	teaches.ID	course_id	sec_id	semester	year
10101	Srinivasan	Comp. Sci.	65000	10101	CS-101	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	10101	CS-315	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	10101	CS-347	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	12121	FIN-201	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	15151	MU-199	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
12121	Wu	Finance	90000	10101	CS-101	1	Fall	2017
12121	Wu	Finance	90000	10101	CS-315	1	Spring	2018
12121	Wu	Finance	90000	10101	CS-347	1	Fall	2017
12121	Wu	Finance	90000	12121	FIN-201	1	Spring	2018
12121	Wu	Finance	90000	15151	MU-199	1	Spring	2018
12121	Wu	Finance	90000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
15151	Mozart	Music	40000	10101	CS-101	1	Fall	2017
15151	Mozart	Music	40000	10101	CS-315	1	Spring	2018
15151	Mozart	Music	40000	10101	CS-347	1	Fall	2017
15151	Mozart	Music	40000	12121	FIN-201	1	Spring	2018
15151	Mozart	Music	40000	15151	MU-199	1	Spring	2018
15151	Mozart	Music	40000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
22222	Einstein	Physics	95000	10101	CS-101	1	Fall	2017
22222	Einstein	Physics	95000	10101	CS-315	1	Spring	2018
22222	Einstein	Physics	95000	10101	CS-347	1	Fall	2017
22222	Einstein	Physics	95000	12121	FIN-201	1	Spring	2018
22222	Einstein	Physics	95000	15151	MU-199	1	Spring	2018
22222	Einstein	Physics	95000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...

加盟经营

- 问题：笛卡尔积“讲师×教师”将讲师的每个元组与教师的每个元组关联起来
  - 结果行中的大部分信息都与没有教授特定课程的教师有关

instructor.ID	*name*	*dept_name*	*salary*	*teaches.ID*	*course_id*	*sec_id*	*semester*	*year*
10101	Srinivasan	Comp. Sci.	65000	10101	CS-101	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	10101	CS-315	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	10101	CS-347	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	12121	FIN-201	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	15151	MU-199	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...

加盟经营

- 为了仅获取与讲师及其教授的课程相关的“讲师×教授”的元组，我们这样写：

$$S_{\text{讲师.id = 教员.id}}(\text{讲师} \times \text{教员})$$

- 我们只获取与教师及其教授的课程相关的“教师×教师”元组
  - 即，instructor.id = teaches.id 的三元组

加盟经营

·对应表格 $\sigma_{\text{讲师.id = 教授.id}}$ (讲师×授课):

instructor.ID	*name*	*dept_name*	*salary*	*teaches.ID*	*course_id*	*sec_id*	*semester*	*year*
10101	Srinivasan	Comp. Sci.	65000	10101	CS-101	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	10101	CS-315	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	10101	CS-347	1	Fall	2017
12121	Wu	Finance	90000	12121	FIN-201	1	Spring	2018
15151	Mozart	Music	40000	15151	MU-199	1	Spring	2018
22222	Einstein	Physics	95000	22222	PHY-101	1	Fall	2017
32343	El Said	History	60000	32343	HIS-351	1	Spring	2018
45565	Katz	Comp. Sci.	75000	45565	CS-101	1	Spring	2018
45565	Katz	Comp. Sci.	75000	45565	CS-319	1	Spring	2018
76766	Crick	Biology	72000	76766	BIO-101	1	Summer	2017
76766	Crick	Biology	72000	76766	BIO-301	1	Summer	2018
83821	Brandt	Comp. Sci.	92000	83821	CS-190	1	Spring	2017
83821	Brandt	Comp. Sci.	92000	83821	CS-190	2	Spring	2017
83821	Brandt	Comp. Sci.	92000	83821	CS-319	2	Spring	2018
98345	Kim	Elec. Eng.	80000	98345	EE-181	1	Spring	2017

加盟经营

·对应表格 $S_{\text{讲师.id = 教授.id}}$ (讲师×授课):

instructor.ID	*name*	*dept_name*	*salary*	*teaches.ID*	*course_id*	*sec_id*	*semester*	*year*
10101	Srinivasan	Comp. Sci.	65000	10101	CS-101	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	10101	CS-315	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	10101	CS-347	1	Fall	2017
12121	Wu	Finance	90000	12121	FIN-201	1	Spring	2018
15151	Mozart	Music	40000	15151	MU-199	1	Spring	2018
22222	Einstein	Physics	95000	22222	PHY-101	1	Fall	2017
32343	El Said	History	60000	32343	HIS-351	1	Spring	2018
45565	Katz	Comp. Sci.	75000	45565	CS-101	1	Spring	2018
45565	Katz	Comp. Sci.	75000	45565	CS-319	1	Spring	2018
76766	Crick	Biology	72000	76766	BIO-101	1	Summer	2017
76766	Crick	Biology	72000	76766	BIO-301	1	Summer	2018
83821	Brandt	Comp. Sci.	92000	83821	CS-190	1	Spring	2017
83821	Brandt	Comp. Sci.	92000	83821	CS-190	2	Spring	2017
83821	Brandt	Comp. Sci.	92000	12121	FIN-201	1	Spring	2018
98345	Kim	Elec. Eng.	80000	15151	MU-199	1	Spring	2018
				22222	PHY-101	1	Fall	2017
				...	...	...	...	...

… 将不会包含具有不同 ID 的元组（行）

回想一下:Joins 的旧写法

- 使用逗号分隔表格
  - 示例:上一张幻灯片中同一问题的解决方案

- 简单介绍一下历史:
  - SQL-1999 中引入了 join（后来而不是原来的方式）
- 问题:
  - 如果您忘记了逗号,有时它仍会起作用（解释为“重命名”）

```
select m.title, c.credited_as,
       p.first_name, p.surname
from movies m,
     credits c,
     people p
where c.movieid = m.movieid
  and p.peopleid = c.peopleid
  and m.country = 'cn'
```

SQL语法源自关系算术中的笛卡尔积形式

S movies.id = credits.id Ù people.peopleid = credits.peopleid Ù movies.country = “cn” （电影×片尾×人物）

回想一下:Joins 的旧写法

- 使用逗号分隔表格
  - 示例:上一张幻灯片中同一问题的解决方案

- 简单介绍一下历史:
  - SQL-1999 中引入了 join（后来而不是原来的方式）

- 问题:
  - 如果您忘记了逗号,有时它仍会起作用（解释为“重命名”）

```
select m.title, c.credited_as,
       p.first_name, p.surname
from movies m,
     credits c,
     people p
where c.movieid = m.movieid
  and p.peopleid = c.peopleid
  and m.country = 'cn'
```

使用逗号作为“乘号”

“select 操作”在这里写为“where”子句

SQL语法源自关系算术中的笛卡尔积形式

S movies.id = credits.id Ù people.peopleid = credits.peopleid Ù movies.country = “cn” （电影×片尾×人物）

加盟经营

- 连接操作允许我们将选择操作和
  笛卡尔积运算合并为单一运算
  - 考虑关系r(R)和s(S)：
  - 让“theta( )”成为模式R“联合”S中属性的谓词。连接s定义如下：
    操作r

! = ! （×）

- 因此， s tutor.id = teaches.id (instructor × teaches)可以等效地写成：

讲师 Instructor.id = teaches.id 教

联盟运作

- 并集运算允许我们将两个关系结合起来 · 符号： r È s

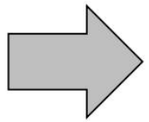
- 为了使r È s有效：
  - r, s必须具有相同的元数（相同数量的属性） · 属性域必须兼容
    - 示例： r的第二列处理的值类型与第二列相同
      s列

联盟运作

- 示例:要查找 2017 年秋季学期或
  2018 年春季学期,或两者皆可

课程编号(s 期= “秋季假” Λ年=2017 年（节）) È Õcourse_id (s 学期= “春天” Λ年=2018年（节））

course_id	*sec_id*	*semester*	*year*	*building*	*room_number*	*time_slot_id*
BIO-101	1	Summer	2017	Painter	514	B
BIO-301	1	Summer	2018	Painter	514	A
CS-101	1	Fall	2017	Packard	101	H
CS-101	1	Spring	2018	Packard	101	F
CS-190	1	Spring	2017	Taylor	3128	E
CS-190	2	Spring	2017	Taylor	3128	A
CS-315	1	Spring	2018	Watson	120	D
CS-319	1	Spring	2018	Watson	100	B
CS-319	2	Spring	2018	Taylor	3128	C
CS-347	1	Fall	2017	Taylor	3128	A
EE-181	1	Spring	2017	Taylor	3128	C
FIN-201	1	Spring	2018	Packard	101	B
HIS-351	1	Spring	2018	Painter	514	C
MU-199	1	Spring	2018	Packard	101	D
PHY-101	1	Fall	2017	Watson	100	A



course_id
CS-101
CS-315
CS-319
CS-347
FIN-201
HIS-351
MU-199
PHY-101

重复项已删除

集合交运算

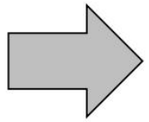
- 集合交集运算使我们能够找到同时属于输入关系
  - 符号：r Ç s
- 假设（与 Union 相同）：
  r、 s具有相同的元数 r和s的属性兼容

集合交运算

- 示例:查找 2017 年秋季和秋季教授的所有课程的集合
  2018 年春季学期

课程编号(S 期= “秋季假” Λ年=2017 年（节）) Ç Õcourse_id (S 学期= “春天” Λ年=2018年（节））

course_id	*sec_id*	*semester*	*year*	*building*	*room_number*	*time_slot_id*
BIO-101	1	Summer	2017	Painter	514	B
BIO-301	1	Summer	2018	Painter	514	A
CS-101	1	Fall	2017	Packard	101	H
CS-101	1	Spring	2018	Packard	101	F
CS-190	1	Spring	2017	Taylor	3128	E
CS-190	2	Spring	2017	Taylor	3128	A
CS-315	1	Spring	2018	Watson	120	D
CS-319	1	Spring	2018	Watson	100	B
CS-319	2	Spring	2018	Taylor	3128	C
CS-347	1	Fall	2017	Taylor	3128	A
EE-181	1	Spring	2017	Taylor	3128	C
FIN-201	1	Spring	2018	Packard	101	B
HIS-351	1	Spring	2018	Painter	514	C
MU-199	1	Spring	2018	Packard	101	D
PHY-101	1	Fall	2017	Watson	100	A



course_id
CS-101

集合差分运算

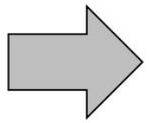
- 集合差分运算使我们能够找到在一个关系中但不在另一个关系中的元组
  - 符号： r – s
- 假设（与并集和交集相同）： r、 s具有相同的元数 r 和s的属性兼容

集合差分运算

- 示例:查找 2017 年秋季学期开设的所有课程,但不在 2018 年春季学期

课程编号(σ 期= “秋季假” ∧年=2017 年（节）) – Õcourse_id (σ 学期= “春天” ∧年=2018年（节）)

course_id	*sec_id*	*semester*	*year*	*building*	*room_number*	*time_slot_id*
BIO-101	1	Summer	2017	Painter	514	B
BIO-301	1	Summer	2018	Painter	514	A
CS-101	1	Fall	2017	Packard	101	H
CS-101	1	Spring	2018	Packard	101	F
CS-190	1	Spring	2017	Taylor	3128	E
CS-190	2	Spring	2017	Taylor	3128	A
CS-315	1	Spring	2018	Watson	120	D
CS-319	1	Spring	2018	Watson	100	B
CS-319	2	Spring	2018	Taylor	3128	C
CS-347	1	Fall	2017	Taylor	3128	A
EE-181	1	Spring	2017	Taylor	3128	C
FIN-201	1	Spring	2018	Packard	101	B
HIS-351	1	Spring	2018	Painter	514	C
MU-199	1	Spring	2018	Packard	101	D
PHY-101	1	Fall	2017	Watson	100	A



course_id
CS-347
PHY-101

赋值操作

- 有时通过将关系代数表达式的一部分分配给临时关系变量来编写关系代数表达式会很方便
  - 赋值运算用 ← 表示,其工作原理类似于编程语言中的赋值

- 示例:查找物理和音乐系的所有讲师

$$\text{物理学} \leftarrow \sigma_{\text{dept\_name} = \text{“物理”}}(\text{讲师})$$
$$\text{音乐} \leftarrow \sigma_{\text{dept\_name} = \text{“音乐”}}(\text{讲师})$$
$$\text{物理和音乐}$$

- 使用赋值运算,查询可以编写为一个顺序程序,该程序由一系列赋值组成,后跟一个表达式,该表达式的值显示为查询的结果

重命名操作

- 某些查询要求在查询中多次使用相同的关系
  - 重命名操作可用于为不同出现的
    相同关系
- 关系代数表达式的结果没有可用的名称
  提到他们

  - 重命名运算符，
  - 表达式x (E) r ，为此目的而提供的

返回名称为x 的表达式E的结果

- 重命名操作的另一种形式也是重命名列：

  r x(A1,A2,..An) (E)

等效查询

- 在关系代数中,有多种方法可以编写查询
  - 示例:查找有关物理学科教师讲授的课程的信息
    薪资超过 90,000 的部门
    - 查询 1

$$\sigma_{\text{dept\_name} = \text{“物理”} \wedge \text{工资} > 90{,}000}(\text{讲师})$$

    - 查询 2

$$\sigma_{\text{部门名称} = \text{“物理”}}(\sigma_{\text{工资} > 90.000}(\text{讲师}))$$

    - 两个查询不相同
      - 但它们是等效的- 它们在任何数据库上都给出相同的结果

等效查询

- 示例:查找有关物理学科教师讲授的课程的信息
  部门
  - 查询 1

    $\sigma_{dept\_name = \text{“物理”}}(\text{讲师} \bowtie_{\text{讲师.ID} = \text{教授.ID}} \text{教授})$

    – （先加入,然后选择）
  - 查询 2

    $\sigma_{dept\_name = \text{“物理”}}(\text{讲师}) \bowtie_{\text{讲师.ID} = \text{教授.ID}}$ 教

    – （先选择,然后加入）

等效查询

- 关系代数的应用:查询优化
  - 将查询转换为计算成本更低的等效查询

数据库系统原理 (CS307)

讲座 9.2:应用程序开发

Zhong-Qiu Wang

计算机科学与工程系
南方科技大学

· 大部分内容来自 Stéphane Faroult 和《数据库系统概念（第 7版）》作者制作的幻灯片。 · 他们的原始幻灯片已修改以适应南方科技大学 CS307 的时间表。 · 幻灯片主要基于马宇欣博士提供的幻灯片

应用程序和用户界面

·大多数数据库用户不使用SQL 之类的查询语言 ·所有数据库的使用都是在应用程序内部进行的 ·应用程序充当用户和数据库之间的中介

数据库

·应用程序分为 ·前端

·中间层 ·后端

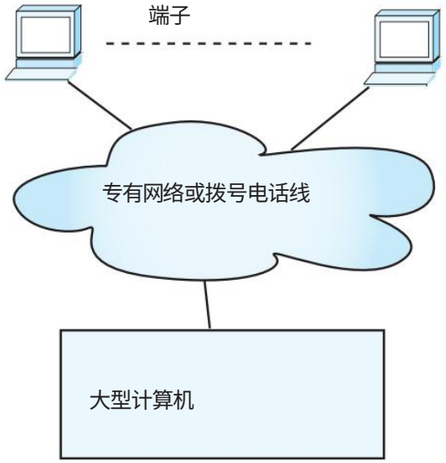
·前端:用户界面

·基于表单的界面

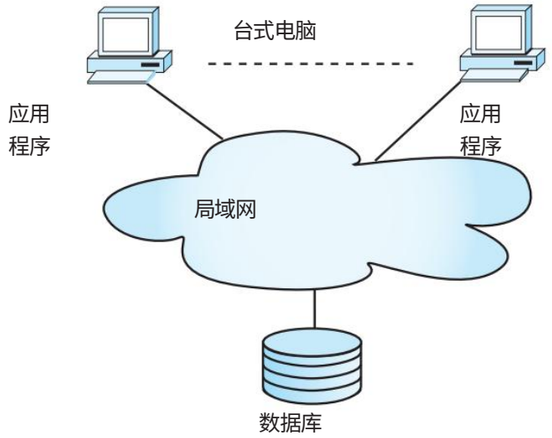
·图形用户界面 ·许多界面都是基于 Web 的

应用程序架构演进

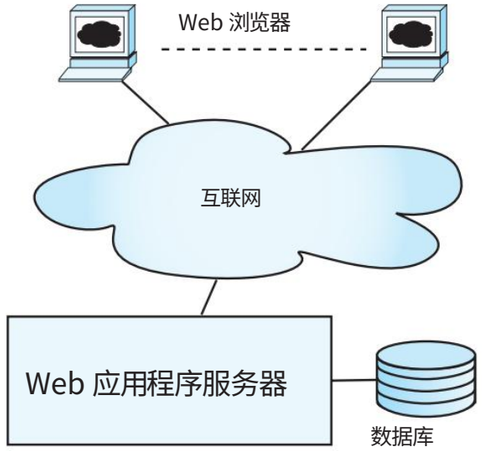
- 应用程序架构的三个不同时代
  - 大型机时代（20 世纪 60 年代和 70 年代）
  - 个人电脑时代（20 世纪 80 年代）
  - 网络时代（20 世纪 90 年代中期至今）
  - 网络和智能手机时代（2010 年至今）



(a)大型机时代



(b)个人电脑时代



(c) 网络时代

Web 界面

- 网络浏览器已成为事实上的标准用户界面
  数据库
  - 使大量用户能够从任何地方访问数据库
  - 无需下载/安装专门的代码,同时提供良好的图形用户界面
    - JavaScript、~~Flash~~ 和其他脚本语言在浏览器中运行,但需要下载透明地
  - 例如:银行、航空公司和租车预订、大学课程注册和评分等等。

万维网

- Web 是一个基于超文本（超文本

本）

- 超文本是一种格式化的文本，它提供到其他内容的链接
- 大多数 Web 文档都是通过以下方式格式化的超文本文档：

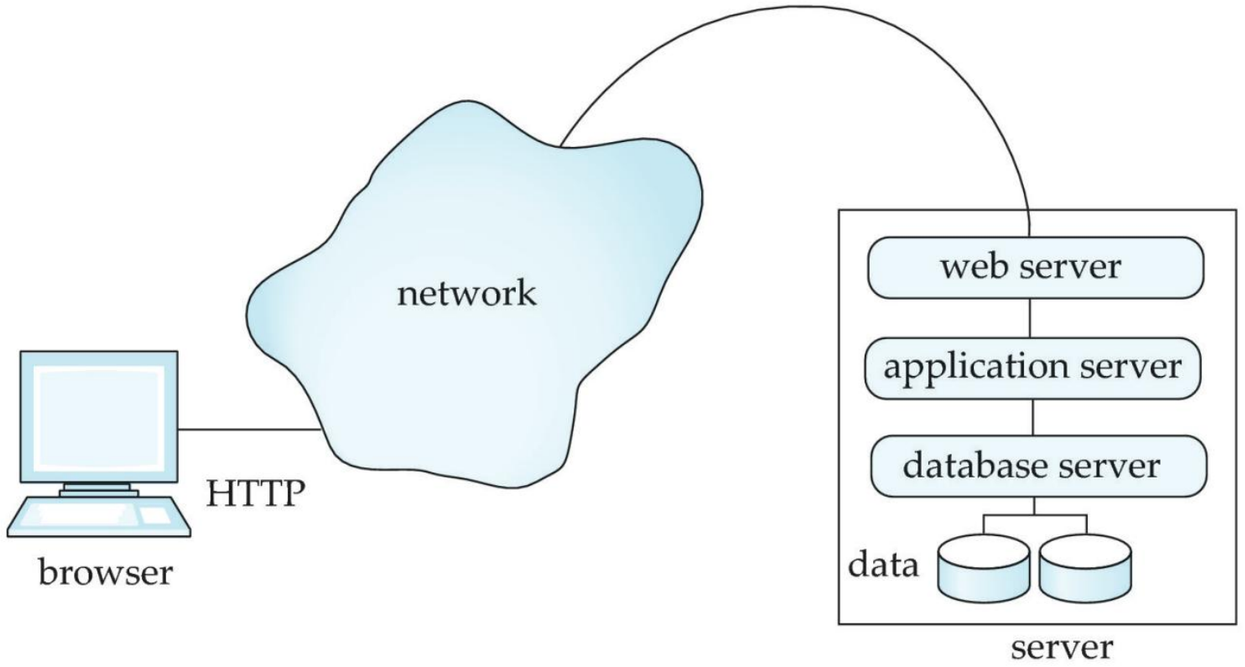
HyperText Markup Language (HTML，超文本标记语言)

- HTML 文档包含
  - 文本以及字体规范和其他格式说明
  - 指向其他文档的超文本链接，可与区域相关联

文本。

  - 表单，使用户能够输入数据，然后这些数据可以发送回 Web 服务器

三层 Web 架构

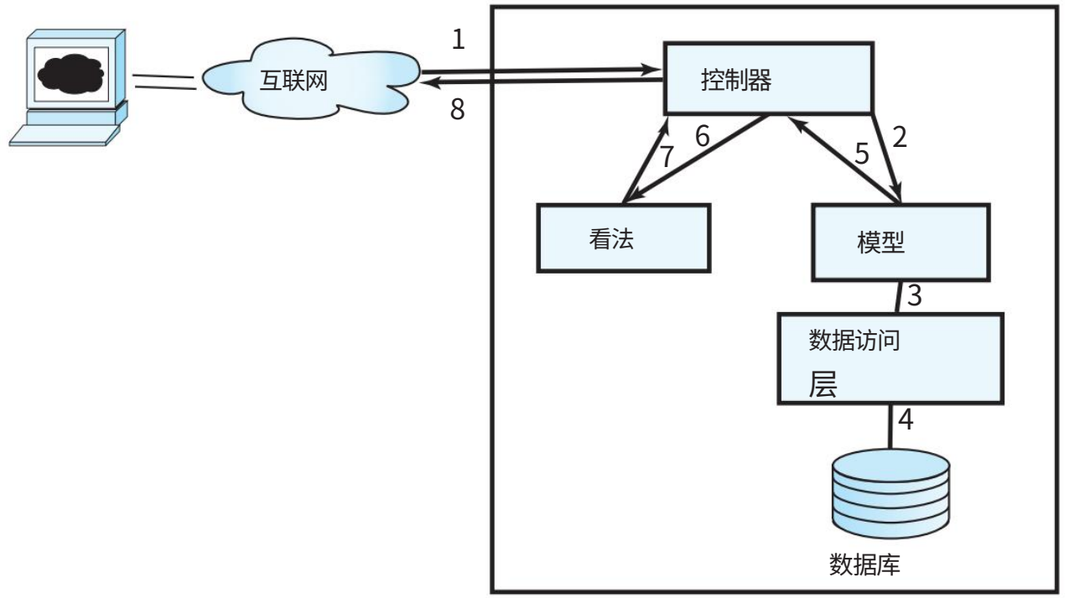


HTML 和 HTTP

- HTML 提供格式、超文本链接和图像显示功能
  - 包括表格、样式表（用于改变默认格式）等。
- HTML 还提供输入功能：从一组选项中选择：弹出菜单、单选按钮、检查列表：输入值
  - 文本框
  - 将填写的输入发送回服务器，由可执行程序执行服务器
- 超文本传输协议 (HTTP)，用于与 Web 服务器

JavaScript

- JavaScript 的应用非常广泛
  - 构成新一代 Web 应用程序（称为 Web 2.0 应用程序)的基础
    提供丰富的用户界面
- JavaScript 函数可以
  - 检查输入的有效性
  - 通过改变底层文档对象模型来修改显示的网页
    （DOM)显示的 HTML 文本的树表示
    - 与 Web 服务器通信以获取数据并使用以下方式修改当前页面
      获取数据,无需重新加载/刷新页面
    - 构成 Web 2.0 应用程序中广泛使用的 AJAX 技术的基础
    - 例如,在下拉菜单中选择一个国家/地区时,该国家/地区的州列表
      自动填充到链接的下拉菜单中



Web/应用服务器

应用程序架构Web 浏览器

- 应用层
  - 演示或用户界面
    - 模型-视图-控制器 (MVC) 架构– 模型:业务逻辑 – 视图:数据呈现,取决于显示设备 – 控制器:接收事件、执行操作并向用户返回视图
  - 业务逻辑层
    - 提供数据的高级视图和对数据的操作 通常使用对象数据模型 隐藏数据存储模式的细节
  - 数据访问层
    - 业务逻辑层和底层数据库之间的接口 提供从业务层对象模型到数据库关系模型的映射

业务逻辑层

- 提供实体的抽象
  - 例如,学生、教师、课程等
- 执行执行操作的业务规则
  - 例如,学生只有完成先决条件并支付了学费后才能报名参加课程
    学费
- 支持工作流,该工作流定义了涉及多个参与者的任务如何
  进行
  - 例如,如何处理学生申请大学的申请
  - 执行任务的步骤顺序
  - 错误处理
    - 例如,如果没有及时收到推荐信该怎么办

对象关系映射

- 允许在面向对象的数据模型上编写应用程序代码,同时将数据存储在传统的关系数据库中
  - 替代方案:实现面向对象或对象关系数据库来存储对象模型
    - 尚未取得商业上的成功
- 模式设计者必须提供对象数据和关系模式之间的映射 - 例如,Java 类 Student 映射到关系 student,并带有相应的属性映射 - 一个对象可以映射到多个关系中的多个元组
- 应用程序打开一个会话,连接到数据库 - 可以使用 session.save(object) 创建对象并将其保存到数据库
  - 用于在数据库中创建适当元组的映射
- 可以运行查询来检索满足指定谓词的对象

Web 服务

- 允许使用远程过程调用机制访问 Web 上的数据
- 两种方法被广泛使用
  - 表述状态转移 (REST):允许使用标准 HTTP 请求通过 URL 进行执行请求并返回数据
    - 返回的数据以 XML 或 JavaScript 对象表示法 (JSON) 编码
  - 大型 Web 服务:
    - 使用 XML 表示发送请求数据以及返回结果
    - 建立在 HTTP 之上的标准协议层

自学

- 要点：
  - 连接数据库（例如 PostgreSQL)并在程序中运行 SQL 查询的技术 / 库 / API
    - ODBC、JDBC?
    - SQLAlchemy?Spring Boot?
    - Hibernate?
    - (Web) 后端框架
      - Spring Boot、Flask、
      - Django
      - NodeJS
  - 前端框架
    - React、Vue