《交通大数据技术》

马晓磊
交通科学与工程学院
2024年春季

数据库简介

数据库

数据库Database = DB

- 长期存在的信息集合

数据库的主要特征

- 模式
  - 文件系统的设计或结构，描述系统中不同元素之间的关系和数据的层次结构。
- 查询
  - 查询是由数据库用户发出的命令，它对数据库中包含的数据执行某些操作。
- 大储存空间
- 控制访问

数据库管理系统

数据库管理系统Database Management System = DBMS

DBMS是一个强大的工具，可以有效地创建和管理大量数据，并使其能够长期保存

- 存储数据的文件的集合
- 为您访问和更新这些文件的程序

数据库管理系统

关系型数据库为用户提供了一个被组织为表（称为关系）的数据视图。

Port	Arrive	Depart	Unit	Driver
DFW	7:15AM	9:20AM	BG3388	1220
DFW	2:00PM	4:10PM	AB3391	1001
LAX	9:50PM	1:00AM	AB7782	2231

Port	City	State	Growth	Passengers
ATL	Atlanta	GA	7.1%	3,200,000
DFW	Dallas	TX	0.4%	2,000,000
LAX	Los Angeles	CA	5.3%	1,900,000

Image Credit: Antony Theobald via Flickr

关系型DBMS实例

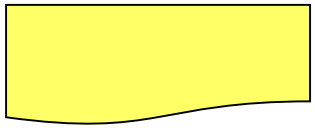
一个例子说明我们为什么需要数据库：

假设我们正在构建一个存储以下信息的系统：

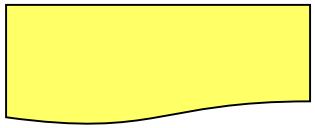
- 学生student
- 课程courses
- 教授professors
- 谁上什么课，谁教什么课

关系型DBMS实例

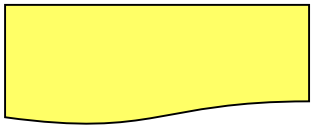
如果我们不使用 DBMS，我们该怎么做呢?



students.txt



courses.txt



professors.txt

使用文本文件

然后编写 C 语言或 Java 程序以实现特定任务。

关系型DBMS实例

程序被开发后，学生们可以使用它们来注册课程

假设学生“Johan”想要注册“CEE412”，那么程序需要进行以下操作：

- 读' student.txt '
- 读' courses.txt '
- 在' student.txt '中查找并更新“Johan”的记录
- 在' courses.txt '中查找并更新“CEE412”的记录
- 写入' student.txt '
- 写入' courses.txt '

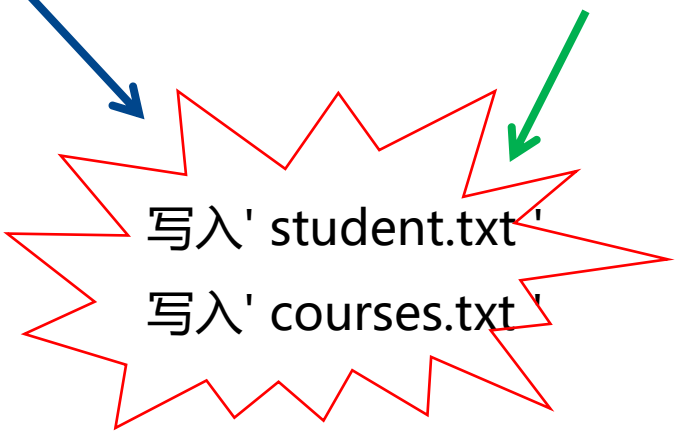
关系型DBMS实例

Johan:

- 读' student.txt ‘
- 读' courses.txt ‘
- 在' student.txt '中查找并更新“Johan”的记录
- 在' courses.txt '中查找并更新“CEE412”的记录

Jamie:

- 读' student.txt '
- 读' courses.txt '
- 在' student.txt '中查找并更新“Jamie”的记录
- 在' courses.txt '中查找并更新“CEE412”的记录



Crash

关系型DBMS实例

如果我们有大数据集（比如50GB），我们会遇到什么问题?

当许多用户同时访问我们的数据时，我们需要锁来保护它们。这不是一个简单的问题。

- 锁就像控制交通的信号灯。该信号旨在防止碰撞，

在给定的时间内只有一个用户可以“写入”。

这些问题与数据量有多大以及我们访问数据的频率有关。

关系型DBMS实例

因此，我们需要关系型DBMS

“双层数据库系统”

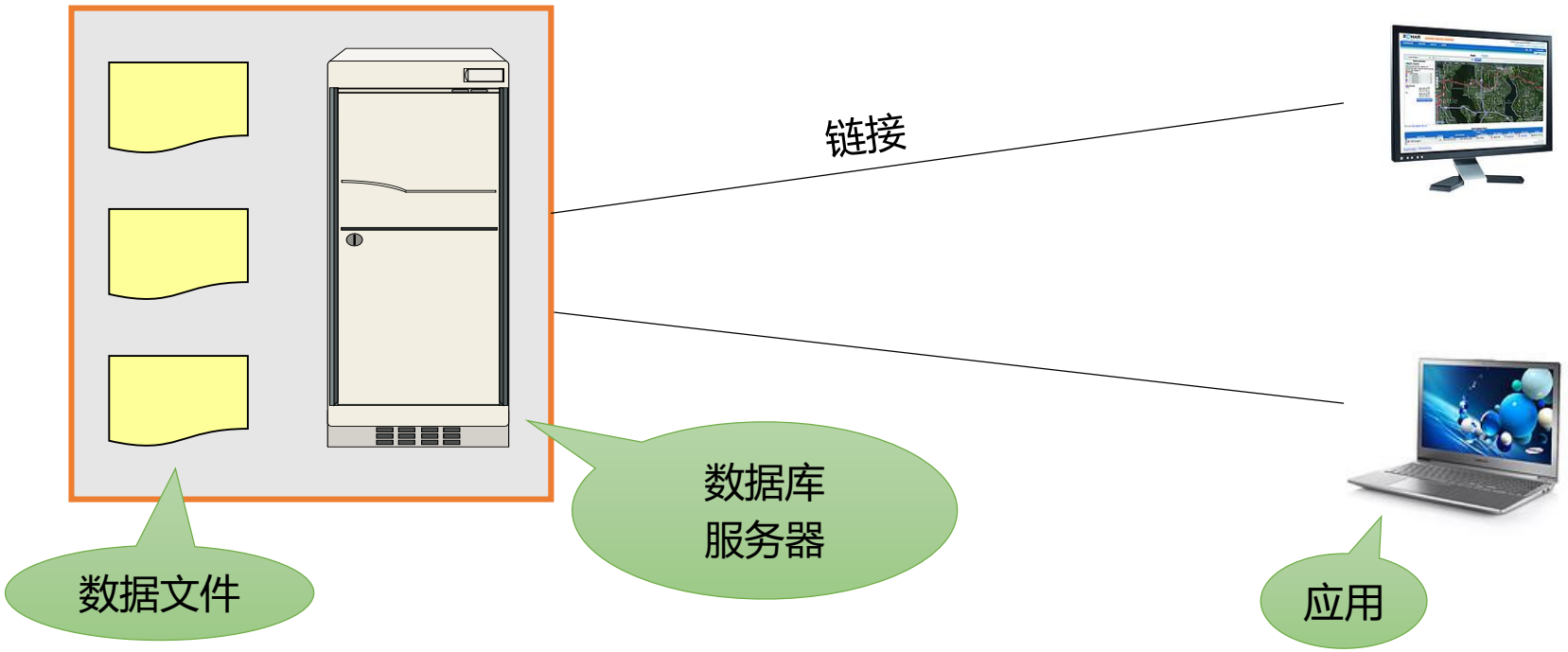


Image credit: TruckPR via Flickr, SamsungTomorrow via Flickr

关系型DBMS实例

表:

Students

SID	Name	Category
1312345	John	Undergrad
1623456	Mary	Grad
...	...	...

Takes

SID	CID
123-45-6789	CEE412
234-56-7890	CEE591
...	...

Professors

PID	Name	Category
98765	Yinhai	Professor
87654	Devyn	Asst. Professor
...	...	...

Courses

CID	Name	Enrollment	Professor
CEE412	Transportation Data Management	45	98765
CEE416	Urban Transportation Planning	35	87654
CEE591	Freight Transportation	11	76543
...	...	...	...

仍然以文件的形式实现，但在背后可能相当复杂

还有什么?

其他需要考虑的问题:

- 如果在更新过程中失去电源或通信，会发生什么情况?
- 如果用户在输入数据时出错了怎么办?
- 如何应对提高大文件或多个并发用户的性能?
- 我们如何管理访问和用户凭据?
- 其他...

DBMS的功能

DBMS向用户提供了哪些功能?

- 持久存储：持久、独立、灵活
- 程接口允许通过查询语言进行数据管理
- 事务管理： ACID测试（Atomicity原子性，Consistency一致性，Isolation隔离性，Durability持久性）

让我们依次了解这些...

持久存储

持久存储，我们为什么要关心?

断开连接或者断电：

- 数据正常。 任何操作都将被回滚的或永久的，不存在中间状态

随着时间的推移生成或收集数据的过程：

- 在生成时将其永久存储， 而不是存储在内存中，并每隔一段时间输出文本文件。

查询处理

DBMS用户使用SQL（结构化查询语言），它有两个组件：

- 数据定义语言（Data Definition Language） - DDL
- 数据操作语言（Data Manipulation Language） - DML

→ 查询语言

DBMS后台有：

- 查询优化器
- 查询引擎
- 存储管理
- 事务管理（并发、恢复）

查询处理

问题：

给定一个包含多个表和关系的数据库，找出“Johan”选的所有课程。

如何回答这个问题?

背后发生了什么?

查询处理器计算出如何高效地回答查询

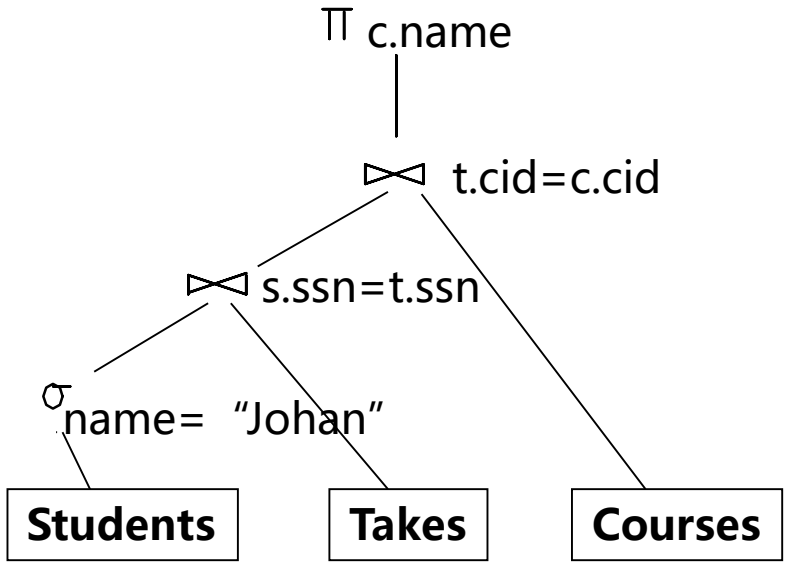
查询由查询编译器进行分析、预处理和优化

查询处理

声明式DECLARATIVE SQL查询 → 命令式IMPERATIVE查询执行计划

```
SELECT  C.name
FROM    Students S, Takes T, Courses C
WHERE   S.name =  “Johan”  and
        S.ssn = T.ssn and T.cid = C.cid
```

优化器为查询选择
最佳执行计划



事务处理

事务是一个工作单元，通常是一个或多个数据库操作，必须以原子方式执行，并且明显与其他事务隔离。

例如：

- ATM取款
- 网上交易股票
- 通过 myUW 注册一个课程

…

事务处理

事务的 ACID 特性：

恰当执行的事务通常都符合“ACID test”：

- “A”代表“atomicity”原子性 – 全有全无；
- “C”代表“consistency”一致性 – 所有规则都保持不变；
- “I” 代表“isolation”隔离性 – 所有事务独立执行；
- “D”代表“durability”持久性 – 对各种电源短缺、故障具有鲁棒性。

DBMS的功能

要记住两点：

服务器-客户机（Client-Server）结构

- 可能很慢
- 连接繁琐
- 但对数据有好处

别人的C程序

- 程序速度可能很快
- 也可能是慢得惊人

在线慕课，什么是数据库


慕课视频片段
视频名称：Video
温馨提示：此视频框在点击“上传手机课件”时会进行转换，用手机进行观看时则会变为可点击的视频。此视频框可被拖动移位和修改大小

DBMS的功能

大型商业数据库供应商：

- Oracle
- IBM (with DB2)
- Microsoft (SQL Server)
- Sybase

一些免费和开源的数据库系统：

- MySQL
- PostgreSQL
- SQLite

数据仓库

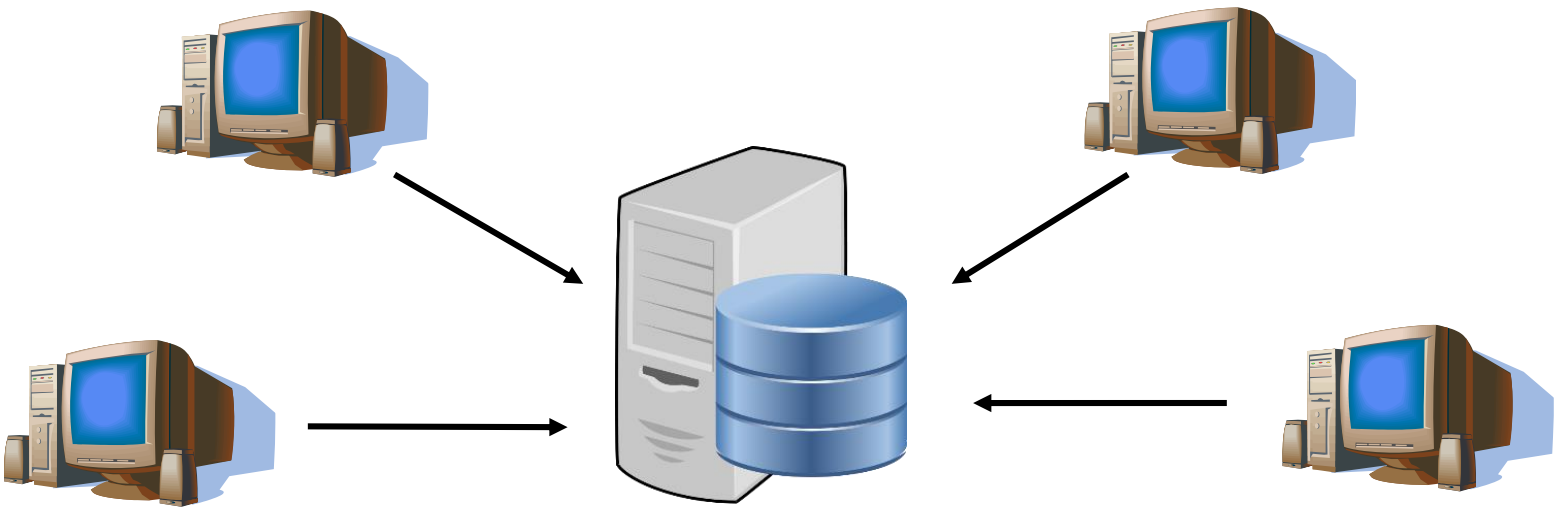
一个常见术语，什么是数据仓库?

数据仓库是专门用于查询和报告的事务数据的副本

为什么我们需要数据仓库?

- 决策支持
- 信息查询与共享
- 数据分析

数据仓库



数据仓库

每个数据库可能使用不同的术语和DBMS

所有数据都集成到集中式存储库中

数据仓库会随着旧数据库的更改而更新，但不一定是实时的

DBMS应用

在哪里使用DBMS?

- 传统“数据库”应用程序的后端
- 大型网站的后端
- Web服务后端

交通应用示例?

DBMS应用

美国环境保护局的机动车排放模拟装置（EPA’s MOVES Motor Vehicle Emission Simulator)

- 这个 Java 应用程序依赖于具有许多关系的大量数据集合，并且由 MySQL 数据库支持
- 多个组织多年来集成的数据，具有许多重要的关系

给定右边的表集：

- 我们如何（除了关系数据库之外）基于此数据构建应用程序?
- 关系模型如何使使维护这些数据集更容易?
- 除非有一个经过精心设计和记录的数据模型，否则我们怎么理解它?

CVL	CVL (Continued)	INF	INF (Continued)
AgeCatagory	SCCVType	AtBaseEmissions	SourceTypeModelYeargrou
AgeGroup	Sector	ATRatio	SourceTypePolProcess
BaseFuel	SourceTypeModelYear	ATRatioNGas	SourceTypeTechAdjustmen
ComplexModelParameterN	SourceUseType	ComplexModelParmeter	SourceTypeYear
ComplexModelParmeters	State	County	SourceUseType
County	SulfurBase	CrankCaseEmissionRatio	StartTeamAdjustment
DayOfAnyWeek	SulfurModelName	DataSource	SulferEmissionRate
DriveSchedule	TankTemperatureGroup	DriveSchedule	SulferModelCoeff
EmissionProcess	WeightClass	DriveScheduleSecond	SulfurCapAmount
EngineSize	Year	EmissionRate	TankTemperatureRise
EngineTech	Zone	EmissionRateByAge	TankVaporGenCoeffs
FuelFormulation	OffNetworkLink	FuelAdjustment	TemperatureAdjustment
FuelModelYearGroup	TemperatureProfileID	FuelFormulation	Year
FuelParameterName		FuelModelWTFactor	ZoneMonthHour
FuelSubType		FuelSubType	GeneralFuelRatio
FuelSupplyYear	**ASSOC**	FuelType	DriveScheduleSecondLink
FuelType	CountyYear	FullACAdjustment	CriteriaRatio
GeneralFuelRatioExpressio	DriveScheduleAssoc	GreetManfAndDisposal	
Grid	FuelEngineTechAssoc	GreetWellToPump	
HourDay	GridZoneAssoc	HCPermiationCoeff	**DIST**
HourOfAnyDay	HourDay	HCSpeciation	AverageSPeedDistribution
HPMSVType	OperatingModePolutantProcessA	HPMSVTypeYear	CumTVVCoeffs
IMInspectfreq	PollutantProcessAssoc	IMCoverage	DayVMTFraction
IMModelYearGroup	PollutantProcessModelYear	IMFactor	FuelEngineFraction
IMTestStandards	SourceTypeModelYeargroup	LinkAverageSpeed	FuelSupply
IMTestType	SourceTypePolProcess	LinkHourVMTFraction	HourVMTFraction
Link		M6SulfurCoeff	OperatingModeDistributior
ModelYear		MeanFuelParameters	RoadTypeDistribution
ModelYearGroup	**CMIT**	MethaneTHCRatio	SCCRoadTypeDistribution
MonthGroupOfAnyYear	AvergeTankGasoline	MonthGroupHour	SCCVTDistribution
MonthofAnyYear	AveragеTankTemperature	MonthVMTFraction	SizeWeightFraction
OMDGPolProcessRepresen	ColdSoakInitialFraction	NONO2Ratio	SoakActivityFraction
OperatingMode	ColdSoakTankTemperature	OperatingMode	SourceBin
OxyThreshName	ExtendedIdleHours	PM10EmissionRatio	SourceBinDistributon
Pollutant	OperatingModeDistribition	Pollutant	SourceTypeAgeDistributior
PollutantDisplayGroup	SHO	PollutantDisplayGroup	Zone
RegulatoryClass	SHP	RefuelingFactors	ZoneRoadType
RetroInputAssociations	SoakActivityFraction	SampleVehicleDay	AVGSpeedBin
SampleVehicleDay	SourceBin	SampleVehiclePopulatio	RegClassFraction
SCC	SourceBinDistribution	SampleVehicleTrip	RoadOpModeDistribution
SCCProcess	SourceHours	SourceTypeAge	LinkSourceTypeHour
SCCRoadType	Starts	SourceTypeHour	
SCCRoadType	StartsPerVehicle	SourceTypeModelYear	

数据库设计

步骤：

- 需求建模（概念、图片）
  - 决定哪些实体应该是应用的一部分，以及它们应该如何链接。.
- 模式设计和实现（逻辑、物理）
  - 决定一组表、属性
  - 定义数据库系统中的表
  - 填充数据库（插入元组）
- 用DBMS编写应用程序
  - 轻松地应对数据管理。

例：冲突引起的延误

用于量化由冲突引起的延误

至少需要:

- 冲突/事故数据: 时间，地点，类型等
- 交通量和速度：地点，时间，流量，车速等

需要不同交通数据和事故之间的关系

在这种情况下，关系是由位置和时间定义的

例：冲突引起的延误

这个设计看起来怎么样?

Incidents			
Date	Time	Route	Milepost
12/22/2012	8:22:35.00	005	171.30
11/31/2012	8:12:22.00	005	156.90
10/11/2012	11:58:21.00	167	19.05

Detectors				
ID_Number	Route	Direction	Lane	Milepost
10031	005	Increasing	1	145.1
10032	005	Increasing	2	145.1
10033	005	Increasing	3	145.1

Loop_Data					
Date	Time	ID_Number	Occupancy	Volume	Speed
12/1/2012	13:22:35.00	10031	3.3	3	68.18
12/1/2012	4:12:22.55	10031	3.0	2	50.68
12/1/2012	22:58:21.21	10031	8.0	6	56.25

考虑数据的形式以及它与应用需求的关系...

数据库设计

数据库设计注意事项:

- 在决定特定的实现之前，就数据库的结构达成一致
- 以最佳方式存储数据

考虑以下问题:

- 要建模的实体
- 实体之间的关系
- 域中存在哪些约束
- 如何实现好的设计

数据库设计的方法

实体/关系模型 (E/R):

- 本质上更有关联性

对象定义语言（ODL）：

- 更接近于面向对象的模型
- 本课程未涵盖

两者都可以（半自动）转换为关系模式

数据库术语

实体（Entity）：

- 实体是可以存储数据的单个对象。它是表的“主题”。实体及其相互关系通过使用实体关系图进行建模。

属性（Attribute）：

- 与数据库对象有关的单个数据项。数据库模式将一个或多个属性与每个数据库实体相关联。属性也称为字段或列。

数据库术语

集（Set）：

- 一种对象的集合，称为集合的元素，以这样一种方式指定：我们原则上可以判断给定的对象是否属于它。集合中元素的顺序和重复无关紧要，例如，{1，2，3}={3，2，1}={1，3，1，2，2}。

列表（List）：

- 一种数据结构，它包含许多可能是不同类型的值，通常从头到尾进行顺序访问，即“有序列表”。

list {1, 3, 5} 和 {3, 5, 1} 相同么?

数据库术语

元数据（Metadata）：

- 元数据实际上是“关于数据的数据”。该术语是指有关数据本身的信息。

模式（Schema）：

- 描述数据库中关系的元数据集合。它可以简单地描述为数据库的“布局”或概述将数据组织到表中的方式的蓝图。

元组（Tuple）：

- 一条记录，表或关系中的一行（实现时）

数据库术语

关系（Relation）：

- 某个域中元组的无序集合，实现时为数据库表

关系（Relationship）：

- 与关系Relation不同描述实体集或关系之间的关系

数据库术语

属性
Attributes

元组Tuple

	CabName	UnitType	ID	Lat	Lon	Route	Milepost	direction	UnitName	isHOV	isMetered	isDuplicate	i	
16	002es00504	main	7	47.951900000	-122.101400000	002	5.04000	E	002es00504:_ME___2	0	0	0	0	0
17	002es00504	speed	8	47.951900000	-122.101400000	002	5.04000	E	002es00504:_ME__S1	0	0	0	0	0
18	002es00504	speed	9	47.951900000	-122.101400000	002	5.04000	E	002es00504:_ME__S2	0	0	0		
19	002es00504	trap	8535	47.951900000	-122.101400000	002	5.04000	E	002es00504:_ME__T1	0	0	0		
20	002es00504	trap	8655	47.951900000	-122.101400000	002	5.04000	E	002es00504:_ME__T2	0	0	0		
21	002es00504	other	10	47.951900000	-122.101400000	002	5.04000	E	002es00504:_ME_O_1	0	0	0	0	0
22	002es00504	station	7815	47.951900000	-122.101400000	002	5.04000	E	002es00504:_ME_Stn	0	0	0	0	0
23	002es00504	other	11	47.951900000	-122.101400000	002	5.04000	E	002es00504:_ME_X_1	0	0	0	0	0
24	002es00504	main	12	47.951900000	-122.101400000	002	5.04000	W	002es00504:_MW___1	0	0	0	0	0
25	002es00504	main	13	47.951900000	-122.101400000	002	5.04000	W	002es00504:_MW___2	0	0	0	0	0
26	002es00504	speed	14	47.951900000	-122.101400000	002	5.04000	W	002es00504:_MW__S1	0	0	0	0	0
27	002es00504	speed	15	47.951900000	-122.101400000	002	5.04000	W	002es00504:_MW__S2	0	0	0	0	0
28	002es00504	trap	8279	47.951900000	-122.101400000	002	5.04000	W	002es00504:_MW__T1	0	0	0	0	0
29	002es00504	trap	8798	47.951900000	-122.101400000	002	5.04000	W	002es00504:_MW__T2	0	0	0	0	0
30	002es00504	other	16	47.951900000	-122.101400000	002	5.04000	W	002es00504:_MW_O_1	0	0	0	0	0
31	002es00504	station	7962	47.951900000	-122.101400000	002	5.04000	W	002es00504:_MW_Stn	0	0	0	0	0
32	002es00504	other	17	47.951900000	-122.101400000	002	5.04000	W	002es00504:_MW_X_1	0	0	0	0	0
33	002es01429	main	18	NULL	NULL	002	14.29...	E	002es01429:_ME___1	0	0	0	0	0

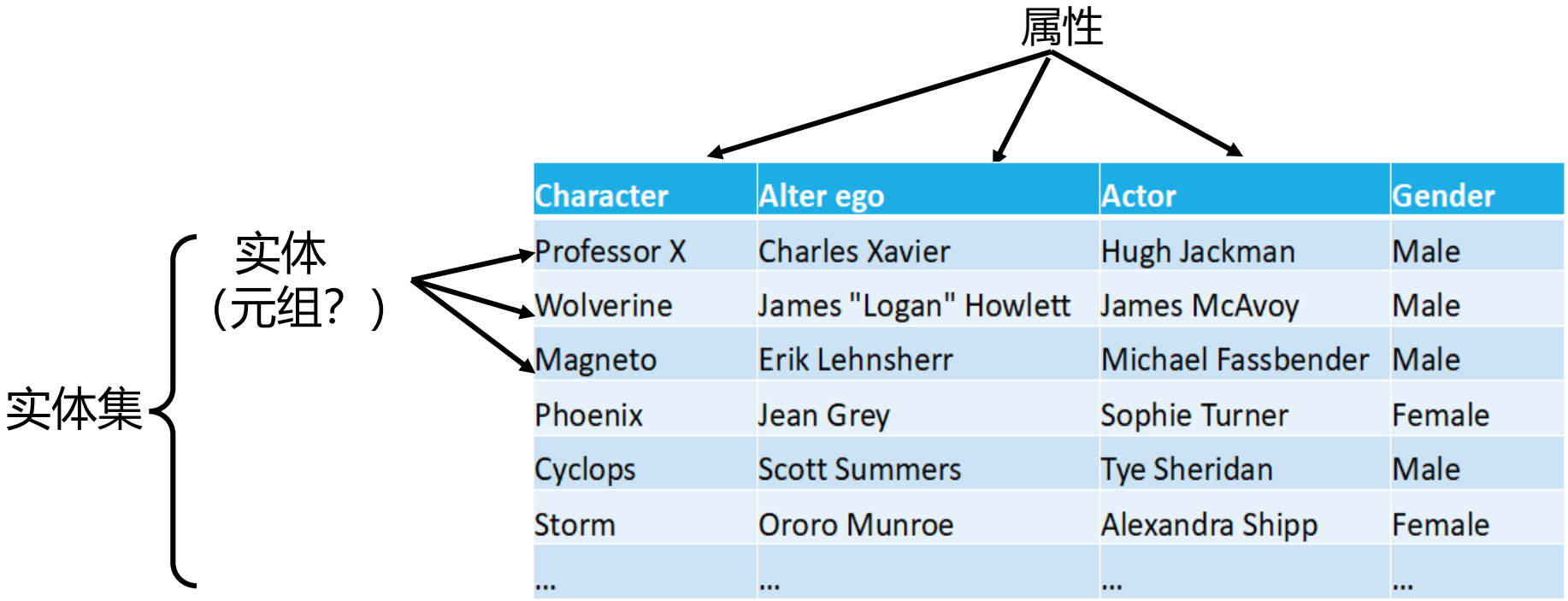
数据库术语

实体Entities：
我们要存储信息的
唯一可识别的“物体”

属性Attribute：
我们要存储的关于实体的信息，
实体的属性

Name
Actor
Gender
Birthdate
Etc.

数据库术语



Character	Alter ego	Actor	Gender
Professor X	Charles Xavier	Hugh Jackman	Male
Wolverine	James "Logan" Howlett	James McAvoy	Male
Magneto	Erik Lehnsherr	Michael Fassbender	Male
Phoenix	Jean Grey	Sophie Turner	Female
Cyclops	Scott Summers	Tye Sheridan	Male
Storm	Ororo Munroe	Alexandra Shipp	Female
...	...	...	...

“元组”和“实体”不是可互换的术语。为什么？

元组=记录或表行，实体=概念性事物或事件

扩展学习：数据库的前世今生

https://developer.aliyun.com/learning/course/62