《智能思维与计算社会科学方法论》

第二&三讲　数据管理方式

授课教师：郑大庆（博士、副教授）

本讲主要内容

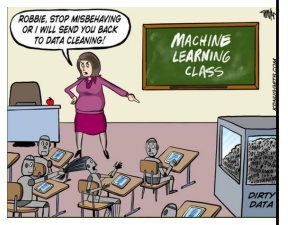
- 人工管理
- 文件系统
- 数据库系统
- 数据仓库
- 数据湖
- 湖仓一体

本讲的实验要求

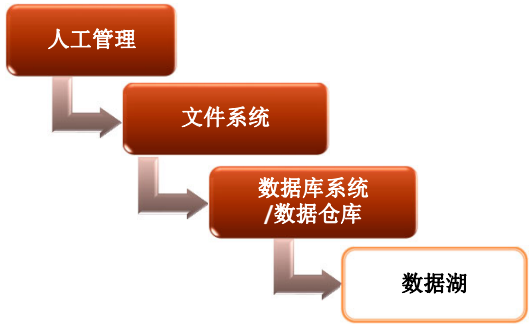
- 搭建Python运行环境
- 运行简单的Python程序
- 实现通过Python程序读取本地文件
- 实现通过Python程序操作ACCESS表

数据管理

- 数据管理是对不同类型的数据进行**收集**、**整理**、**组织**、**存储**、**加工**、**传输**、**检索**的各个过程，它是计算机一个重要的应用领域。
- 数据管理的目的之一是从大量原始的数据中抽取、推导出对人们有价值的**信息**，然后利用信息作为行动和决策的依据；数据管理的另一目的是为了借助计算机科学地保存和管理复杂、大量的数据，以便人们能够方便而充分地利用这些信息资源。



计算机数据管理的三个阶段



一、人工管理

二十世纪五十年代中期以前，计算机硬件方面，外存储器只有纸带、卡片、磁带，没有像硬盘一样可以随机访问、直接存取的外部存储设备；软件方面，没有操作系统软件和数据管理软件。

人工管理阶段的数据处理有以下特点：

数据不保存。用户把应用程序和数据一起输入内存，通过应用程序对数据进行处理，输出处理结果。任务完成后，数据随着应用程序从内存一起释放。

- **数据和程序不具有独立性**。数据由应用程序自行管理。应用程序中不仅要规定数据的逻辑结构，还要阐明数据在存储器上的存储地址。当数据改变时，应用程序也要随之改变。
- **数据不能共享**。一个应用程序中的数据无法被其他应用程序所利用。程序和程序之间不能共享数据，因而产生大量重复的数据，称为数据冗余。

一、人工管理

➢数据记录学号、姓名、三门功课的成绩

score=[["101","Mary",80,85,90],["102","Rose",80,90,95],["103","Mike",75,72,65]
,["104","Peter",65,63,58],["105","Harry",95,93,88]]

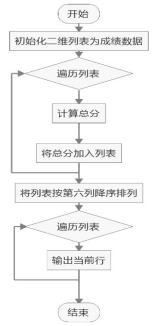
ID	Name	Chinese	Maths	English
101	Mary	80	85	90
102	Rose	80	90	95
103	Mike	75	72	65
104	Peter	65	63	58
105	Harry	95	93	88

一、人工管理

示例：计算每一位同学的总分，降序排列并输出。

```
# 展示数据和程序一体的示例

score = [["101","Mary",80,85,90],["102","Rose",80,90,95],["103","Mike",75,72,65],
         ["104","Peter",65,63,58],["105","Harry",95,93,88]]

for i in range(5):
    zf= score[i][2] + score[i][3] + score[i][4]
    # 将总分加入到列表元素中
    score[i].append(zf)

score = sorted(score, key = lambda k:k[5], reverse = True)

for j in range(5):
    print("学号：{}，姓名：{}，总分：{}".format(score[j][0],score[j][1], score[j][5]))

```

```
学号：105，姓名：Harry，总分：276
学号：102，姓名：Rose，总分：265
学号：101，姓名：Mary，总分：255
学号：103，姓名：Mike，总分：212
学号：104，姓名：Peter，总分：186
```



程序和数据混合的弊端

- **可读性差**：程序和数据混合在一起，难以区分哪些是程序代码，哪些是数据，对于代码的可读性和维护性造成影响。
- **安全性问题**：由于程序和数据混在一起，可能会导致数据被意外修改或者被恶意篡改，从而对系统的安全性产生威胁。
- **执行效率低**：程序和数据混合在一起，可能会导致程序的执行效率降低，因为在执行程序时需要先解析数据，再执行程序。
- **可扩展性差**：程序和数据混合在一起，可能会导致系统的可扩展性变差，因为在添加新功能时需要修改程序和数据，增加了开发和维护的复杂度。
- **维护困难**：程序和数据混合在一起，可能会导致维护困难，因为在修改程序时需要考虑数据的影响，而在修改数据时需要考虑程序的影响。

程序和数据混合的弊端——无法保存

```
# 程序2 数据和程序一体带来的修改困难

score = [["101","Mary",80,85,90],["102","Rose",80,90,95],["103","Mike",75,72,65],
         ["104","Peter",65,63,58],["105","Harry",95,93,88]]

for i in range(5):
    zf= score[i][2] + score[i][3] + score[i][4]
    # 将总分加入到列表元素中
    score[i].append(zf)

score = sorted(score, key = lambda k:k[5], reverse = True)

for j in range(5):
    print("学号：{}，姓名：{}，总分：{}".format(score[j][0],score[j][1], score[j][5]))

```

```
学号：105，姓名：Harry，总分：276
学号：102，姓名：Rose，总分：265
学号：101，姓名：Mary，总分：255
学号：103，姓名：Mike，总分：212
学号：104，姓名：Peter，总分：186
```


本讲主要内容

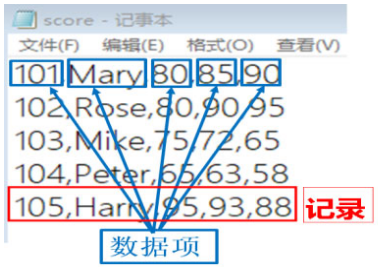
- 人工管理
- 文件系统
- 数据库系统
- 数据仓库
- 数据湖
- 湖仓一体

文件系统

文件系统

➢score.csv文件中存储五位同学的学号、姓名和三门功课的成绩。

```
1  101 Mary 80 85 90
2  102 Rose 80 90 95
3  103 Mike 75 72 65
4  104 Peter 65 63 58
5  105 Harry 95 93 88
```

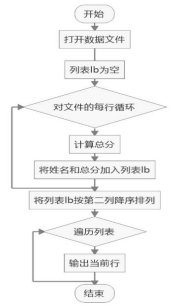


文件系统

➢例题2：计算score.csv文件中每位同学的总分，降序排列并输出

```
# 展示数据和程序分离的示例

fl=open("score.csv")
lb=[]

#将数据存入二维列表
for line in fl:
    xx =line.strip().split(" ")
    print(xx)
    zf = eval(xx[2])+eval(xx[3])+eval(xx[4]) #计算总分
    lb.append([xx[0],xx[1],zf])

lb=sorted( lb,key=lambda k:k[2] ,reverse=True)
for i in range(5):#输出列表
    print("学号{}，姓名{}，总分{}".format(lb[i][0],lb[i][1],lb[i][2]))
```

```
['101', 'Mary', '80', '85', '90']
['102', 'Rose', '80', '90', '95']
['103', 'Mike', '75', '72', '65']
['104', 'Peter', '65', '63', '58']
['105', 'Harry', '95', '93', '88']
学号105，姓名Harry，总分276
学号102，姓名Rose，总分265
学号101，姓名Mary，总分255
学号103，姓名Mike，总分212
学号104，姓名Peter，总分186
```



数据文件

➢数据文件是在计算机系统上使用的最常见类型的文件之一。

➢它可以是存储一个数据的任何文件。

➢它可以采取纯文本文件、编码后的文件（通过加密），或二进制文件格式。

➢数据文件由多种不同的应用软件建立，包含了数以千计的专有文件格式。

数据文件

➢**CSV格式**

✓CSV是一种通用的、相对简单的文件格式，被用户、商业和科学广泛应用。

✓CSV文件可以用记事本等文本编辑软件打开，也可以使用excel软件打开。

✓CSV泛指具有以下特征的任何文件：

- 纯文本，使用某个字符集，比如ASCII、Unicode、EBCDIC或GB2312；
- 由记录组成（典型的是每行一条记录）；
- 每条记录被分隔符分隔为字段（典型分隔符有逗号、分号或制表符）；
- 每条记录都有同样的字段序列。

数据文件

CSV格式

记事本打开csv文件

	A	B	C	D
1	year	rainyDays	coldDays	hotDays
2	1951	128	95	19
3	1952	151	114	10
4	1953	166	98	36
5	1954	174	108	18
6	1955	160	95	25
7	1956	139	116	40
8	1957	167	103	22

在Excel中打开csv文件

网购商品的下单转化率数据

数据文件

➢JSON格式

✓JSON是一种与开发语言无关的、轻量级的数据存储格式，全称JavaScript Object Notation。起初来源于JavaScript这门语言。由于易于阅读、编写、程序解析与生产，成为一种数据格式的标准规范，被广泛使用。

✓JSON格式文件的规定如下：

- 映射采取<键>:<值>的键值对的形式，键为要表示的列名，值为该列所对应的值。例如1951年表示为“year”:1951。
- 映射的集合用大括号（"{}"）表示。每行记录的各列存放在一个大括号中。
- 并列的数据之间用逗号（","）分隔。例如，一行记录的各列用逗号分隔，各个集合之间用逗号分隔。
- 并列数据的集合（数列）用方括号("[]")表示。

数据文件

➢JSON格式

表示天气的json格式数据

数据文件

➢XML格式

XML是一种可扩展的标记语言，脱胎于HTML文件的一种数据存储的语言。

XML格式文件的规定如下

（1）XML声明是XML文档的第一句，其格式如下：

<?xml version="1.0" encoding="utf-8"?>

（2）XML文档必须有一个根元素，就是紧接着声明后面建立的第一个元素，其他元素都是这个根元素的子元素，根元素完全包括文档中其他所有的元素。

根元素的起始标记要放在所有其他元素的起始标记之前；根元素的结束标记要放在所有其他元素的结束标记之后。

（3）在XML中，标记存放在尖括号中，所有标记必须成对出现。例如开始标记为<year>，结束标记为</year>。数据存放在开始标记和结束标记之间。<year>1951</year>表示1951年。

（4）在XML文档中，大小写是有区别的。“A”和“a”是不同的标记。注意在写元素时，前后标记的大小写要保持一致。

数据文件

➢XML文件

```
<?xml version="1.0" encoding="UTF-8"?>  声明语句
- <TQ>  根元素开始标记
  - <tqsj>  行开始标记
列开始标记 <year>1951</year>  列结束标记   数值
      <rainDays>128</rainDays>
      <coldDays>95</coldDays>
      <hotDays>19</hotDays>
  </tqsj>  行结束标记
  - <tqsj>
      <year>1952</year>
      <rainDays>151</rainDays>
      <coldDays>114</coldDays>
      <hotDays>10</hotDays>
  </tqsj>
  - <tqsj>
      <year>1953</year>
      <rainDays>151</rainDays>
      <coldDays>114</coldDays>
      <hotDays>10</hotDays>
  </tqsj>
</TQ>  根元素结束标记
```

表示天气的XML格式数据

本讲主要内容

- ➢人工管理
- ➢文件系统
- ➢数据库系统
- ➢数据仓库
- ➢数据湖
- ➢湖仓一体

数据库系统

➢ 二十世纪六十年代后期，大容量和快速存储的磁盘相继投入市场，为新型数据管理技术奠定了物质基础。此外，计算机管理的数据量急剧增长，多用户、多程序实现数据共享的要求日益增强，数据库技术应运而生。

➢ 数据库系统阶段的数据处理有以下特点：

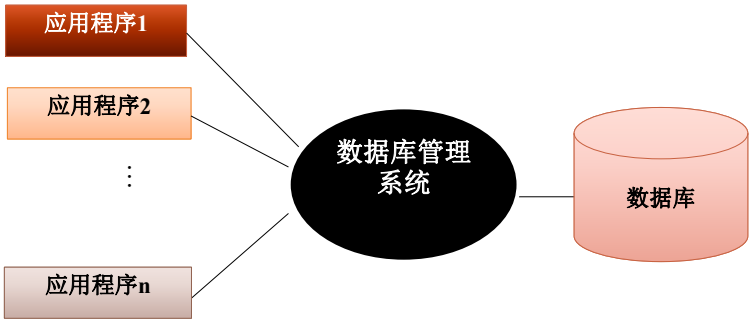
✓ **数据的共享性高，冗余度低。**建立数据库时，以面向全局的观点组织数据库中的数据。数据可被多个用户、多个应用程序共享使用，大大减少数据冗余。

✓ **采用特定的数据模型。**数据库中的数据是以一定的逻辑结构存放的，这种结构由数据库管理系统所支持的数据模型来决定。目前流行的数据库管理系统大多建立在关系模型的基础上的。

✓ **数据独立性高。数据与应用程序之间彼此独立。**当数据的存储格式、组织方法和逻辑结构或发生改变时，不需要修改应用程序。

✓ **统一的数据控制功能。**数据库由数据库管理系统来统一管理，并提供对数据的并发性、完整性、安全性等控制功能。

数据库系统



存储程序的数据表：以Access为例

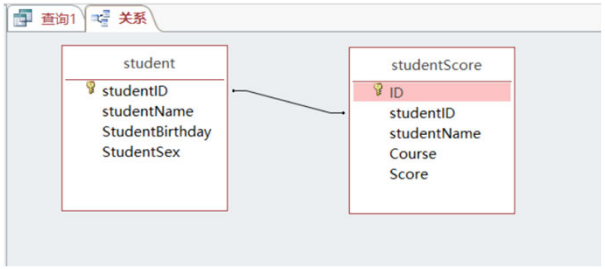
score1 | student

ID	studentid	name	course	cj
1	101	Marry	Chinese	80
2	101	Marry	Maths	85
3	101	Marry	English	90
4	102	Rose	Chinese	80
5	102	Rose	Maths	90
6	102	Rose	English	95
7	103	Mike	Chinese	75
8	103	Mike	Maths	72
9	103	Mike	English	65
10	104	Peter	Chinese	65
11	104	Peter	Maths	63
12	104	Peter	English	58
13	105	Harry	Chinese	95
14	105	Harry	Maths	93
15	105	Harry	English	88

score1 | student

ID	studentid	name	birthday	sex
5	101	Marry	2004/3/21	女
6	102	Rose	2004/8/30	女
7	103	Mike	2004/2/20	男
8	104	Peter	2003/8/19	男
9	105	Harry	2004/8/1	男
10	106	Tom	2004/1/22	男
4	107	Jack	2004/7/26	男

存储程序的数据表：以Access为例



数据库系统操作：Structure Query Language

SQL语句实现两张表的连接。

SELECT score1.course, student.name, student.birthday FROM score1 INNER JOIN student ON score1.studentid = student.studentid;

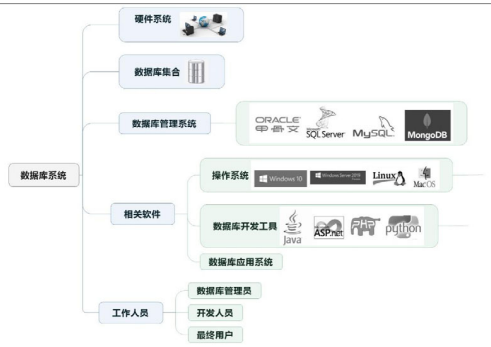
查询1 | 查询2 | score1

course	name	birthday
Chinese	Marry	2004/3/21
Maths	Marry	2004/3/21
English	Marry	2004/3/21
Chinese	Rose	2004/8/30
Maths	Rose	2004/8/30
English	Rose	2004/8/30
Chinese	Mike	2004/2/20
Maths	Mike	2004/2/20
English	Mike	2004/2/20
Chinese	Peter	2003/8/19
Maths	Peter	2003/8/19
English	Peter	2003/8/19
Chinese	Harry	2004/8/1
Maths	Harry	2004/8/1
English	Harry	2004/8/1

数据库系统

➢**数据库系统**（Database System，DBS）是指引入数据库技术的计算机系统，它实现了有组织地、动态地存储大量相关数据，提供了数据处理和信息资源共享的便利手段。

数据库系统组成



数据库系统组成：硬件系统

Ø硬件系统主要指计算机硬件设备，包括CPU、内存、外存、输入/输出设备等。

Ø由于要运行操作系统、数据库管理系统的核心程序和应用程序，要求计算机有足够大的内存；同时，由于数据库、系统软件和应用软件都保存在外存中，对计算机的外存容量的要求也很高。此外，对于网络数据库系统，还需要有网络通信设备的支持。

数据库系统组成：数据库集合（Database, DB）

➢数据库可直观地理解为数据的集合。数据库是指存储在计算机外存中、结构化的相关数据的集合。它不仅包含了描述事物本身的数据，还包含了相关数据之间的联系。

➢数据库以文件的形式存储在外存中，用户通过数据库管理系统来统一管理和控制数据。

数据库系统组成：数据库管理系统（Database Management System, DBMS）

➢数据库管理系统是对数据实行专门管理的系统软件，是数据库系统的核心。它在操作系统的基础上运行，方便用户建立、使用和维护数据库，提供数据的安全性和完整性等统一控制机制。

ORACLE 甲骨文　Microsoft SQL Server　MySQL　MongoDB

数据库系统组成

➢数据库管理系统功能

✓数据定义

◦ DBMS提供数据定义语言DDL（Data Definition Language），负责数据库对象的建立、修改和删除等。

✓数据操纵

◦ DBMS提供数据操纵语言DML（Data Manipulation Language），实现数据的基本操作，例如，对表中数据的查询、插入、删除和修改。

✓数据控制

◦ 包括安全性控制、完整性控制和并发性控制等。

✓数据库维护

◦ 包括数据库的备份和恢复，数据库的转换、数据库的性能监视和优化等。

数据库系统组成

➢相关软件

✓除了数据库管理系统，数据库系统还必须有相关软件的支持，包括操作系统、数据库开发工具、数据库应用系统等。

➢操作系统

数据库系统组成

➢相关软件

✓除了数据库管理系统，数据库系统还必须有相关软件的支持，包括操作系统、数据库开发工具、数据库应用系统等。

◦ 数据库应用系统

◦ **是指开发人员结合各领域的具体需求，利用数据库系统资源，使用开发工具所开发的给一般用户使用的应用软件，如图书管理系统、学籍管理系统、商品进销存系统等。**

数据库系统组成

➢相关软件

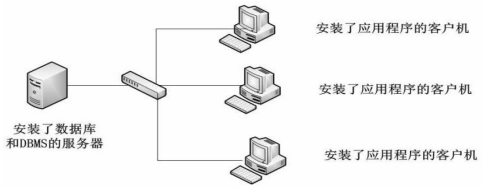
✓据库开发工具是指开发人员编写数据库应用系统所使用的软件平台，可分为两类：

◦ 一类是基于数客户机/服务器模式（C/S）的开发工具，如Visual Basic、Visual C++、Delphi等，

◦ 一类是基于浏览器/服务器模式(B/S)的开发工具，如ASP、JSP、PHP等。

数据库系统组成

➢相关软件

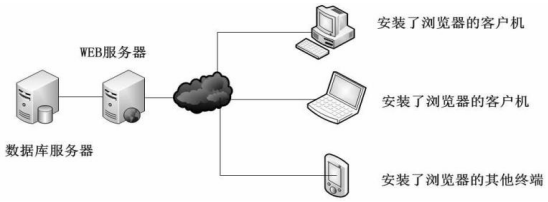


C/S模式数据库系统

在服务器结点存放数据及执行DBMS功能，客户机安装应用系统。客户端的用户请求被传送到服务器，服务器进行处理后，将处理结果返回给用户。

数据库系统组成

➢相关软件



B/S模式数据库系统

客户端仅安装浏览器软件，用户通过URL向Web服务器发出请求，Web服务器运行脚本程序，向数据库服务器发出数据请求。数据库服务器执行处理后，将结果返回给Web服务器。Web服务器根据结果产生网页文件，客户端接收到网页文件后，在浏览器中显示出来。

数据库系统组成

➢各类人员

数据库系统中还包括设计、建立、管理、使用数据库的各类人员。

✓ 数据库管理员（Database Administrator，DBA）

数据库管理员是负责全面管理和实施数据库控制和维护的技术人员，他要参与数据库的规划、设计和建立，负责数据库管理系统的安装和升级；规划和实施对数据库的备份和还原；规划和实施数据库的安全性，控制和监视用户对数据库的存取访问；监督和记录数据库的操作状况，进行性能分析，实施系统优化。

✓ 开发人员

开发人员负责应用系统的需求分析，设计应用系统的功能，使用开发工具实现应用系统功能。

✓ 最终用户

最终用户只需通过运行数据库应用系统来处理数据，不需要了解数据库的设计、维护和管理等问题。

数据库管理员  **应用系统开发人员**  **最终用户** 

关系型数据库

关系型数据库

➢概念模型

现实世界中事物及联系在人们头脑中的反映，经过人们头脑的分析、归纳、抽象，形成信息世界。

对信息世界所建立的抽象的模型，称之为概念模型。

由于概念模型是用户与数据库设计人员之间进行交流的语言，因此概念模型一方面应该能够方便、直接地表达应用中的各种语义知识，另一方面它还应该简单、清晰，易于用户理解。

目前常用**实体联系模型**表示概念模型。

关系型数据库

➢教学管理数据库设计

✓某校有多名教师，开设多门课程；一名学生可学习多门课程；一门课程可有多名学生学习；一个学生属于一个班级。设计一个学生选课系统，实现教学管理。

- 学生信息包括：学号、姓名、性别、出生日期、籍贯
- 课程信息包括：课程号、课程名称、开设学院、学分、是否必修、学时、课程简介
- 学生选课包括：学期、分数
- 教师信息包括：教师编号、姓名、性别、职称
- 班级信息包括：班级编号、班级名称、所属院系、年级

关系型数据库概念模型

➢实体

✓实体是客观存在并且可相互区别的事物。它可以是实际的事物，如读者、图书、学生、教师、课程等；也可以是抽象的事件，如借书、选课、订货等活动

➢实体属性

✓实体的特性称为属性，一个实体可以用多个属性来描述。

✓例如，学生实体可以用学号、姓名、出生日期、性别等属性来描述。

✓课程实体可以用课程编号、课程名称、开课学院、学分、是否必修课、学时、简介信息等属性来描述。

关系型数据库概念模型

➢实体型和实体集

✓用实体名及其属性集合描述的同类实体，称为实体型。

✓例如，学生（学号、姓名、出生日期、性别、籍贯）就是一个实体型。课程（课程编号、课程名称、开课学院、学分、是否必修课、学时、简介）也是一个实体型。

✓同类型实体的集合称为实体集。

- 所有的学生构成一个实体集。在学生实体集中，“2022111001 王刚 2003-07-26 男 上海”表示一位具体的学生。所有的课程也构成一个实体集。在课程实体集中，“0101 高等数学 数学院 6 是 96 所有专业数学基础课”表示一个具体的课程。

关系型数据库概念模型

➢实体间的联系：指实体集与实体集之间的联系。实体间的联系分为“一对一”、“一对多”和“多对多”3种。

✓一对一联系

- 定义：有实体集A和实体集B，若实体集A中的每个实体仅与实体集B中的一个实体联系，反之亦然，则两个实体间为一对一联系，记为1:1。
- 示例：班级和班长是两个实体集，一个班级只能有一个班长，而一个班长只能在一个班级任职，则班级和班长之间为一对一的联系。

关系型数据库概念模型

➢实体间的联系

✓一对多联系

- 定义：设有实体集A和实体集B，若对于实体集A中的每个实体，实体集B都有多个实体与之对应；反之，对于实体集B中的每个实体，实体集A中只有一个实体与之对应，则两个实体间为一对多联系，记为1:n。
- 示例：班级和学生是两个实体集，一个班级有多名学生，而一个学生只能属于一个班级，则班级和学生之间为一对多的联系。

✓多对多联系

- 定义：设有实体集A和实体集B，若对于实体集A中的每个实体，实体集B都有多个实体与之对应；反之，对于实体集B中的每个实体，实体集A中也有多个实体与之对应，则两个实体间为多对多联系，记为m:n。
- 示例：学生和课程两个实体集，一个学生可以学习多门课程，而一门课程也可以被多位学生学习，则学生和课程之间为多对多的联系。

关系型数据库概念模型

➢实体间的联系

✓多对多联系

- 设有实体集A和实体集B，若对于实体集A中的每个实体，实体集B都有多个实体与之对应；反之，对于实体集B中的每个实体，实体集A中也有多个实体与之对应，则两个实体间为多对多联系，记为m:n。
- 学生和课程两个实体集，一个学生可以学习多门课程，而一门课程也可以被多位学生学习，则学生和课程之间为多对多的联系。

关系型数据库概念模型

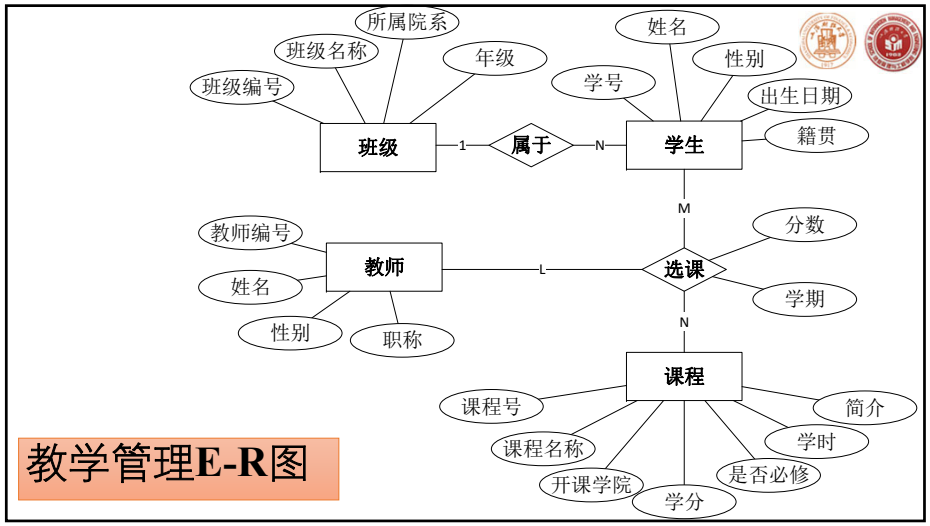
➢E-R图

✓实体-联系模型使用E-R图（Entity-Relationship Diagram）来描述概念模型。

✓在E-R图中，用矩形表示实体型，用椭圆表示实体的属性，用菱形表示实体型之间的联系，相应的实体名、属性名、联系名写明在对应的框内，用无向边将各种框连接起来，并在连接实体型的线段上标上联系的类型。

关系型数据库概念模型

➢绘制教学管理E-R图



教学管理E-R图

关系型数据库概念模型

➢数据模型

✓建立概念模型之后，为了将其转换为计算机能够管理的数据，需要按计算机系统的观点对数据建模。

✓数据模型直接面向数据库中数据的逻辑结构，有一组严格的语法和语义语言，可以用来定义、操纵数据库中的数据。它所描述的内容包括三个部分：数据结构、数据操作和数据完整性约束条件。

- **数据结构**是指储存在数据库中对象类型的集合，描述数据库组成对象以及对象之间的联系。
- **数据操作**是指对数据库中各种对象实例允许执行的操作的集合，包括操作及其相关的操作规则。
- **数据完整性约束条件**是指在给定的数据模型中，数据及其联系所遵守的一组通用的完整性规则，它能保证数据的正确性和一致性。

关系型数据库概念模型

➢数据模型

✓任何一个数据库管理系统都是基于某种数据模型的。20世纪70年代至80年代初期，广泛使用的是基于层次、网状数据模型的数据库管理系统。层次模型以树状结构表示实体与实体之间的联系，网状模型是以网状结构表示实体与实体之间的联系。

✓现在，关系模型是使用最普遍的数据模型，它以二维表的形式表示实体与实体之间的联系。关系模型以关系代数为基础，操作的对象和结果都是二维表，也就是关系。目前流行的数据库管理系统Oracle、SQL Server、MySQL等都是关系数据库管理系统。

✓在关系模型中，基本数据结构就是二维表。实体与实体之间联系用二维表来表示，数据被看成二维表中的元素。操作的对象和结果都是二维表。

关系型数据库

➢关系术语

✓关系：一个关系就是一张二维表，每个关系有个关系名。

✓对关系的描述称为关系模式，其格式为关系名（属性1，属性2，……，属性n）。

✓在mysql中，一个关系存储为一个数据表文件。关系模式对应于数据表的结构，其格式为表名（字段名1，字段名2，字段名3，……，字段名n）。

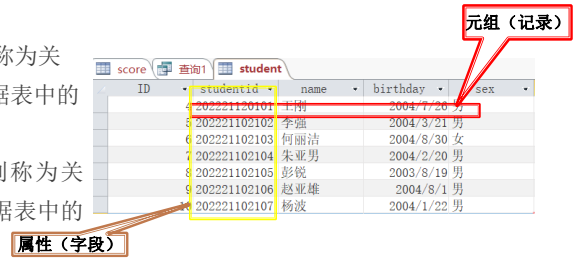
✓学生（学号，姓名，性别，出生日期，籍贯），就是“学生”关系的关系模式，即“学生”表的结构。

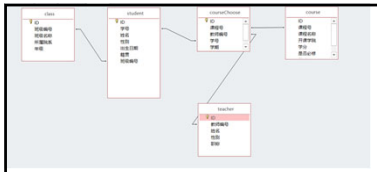
关系型数据库

➢关系术语

✓元组：二维表的一行称为关系的一个元组，即数据表中的一条记录。

✓属性：二维表的一列称为关系的一个属性，即数据表中的一个字段。





关系型数据库

Ø关系术语

ü 域

- 属性的取值范围称为域，即不同元组对同一个属性的取值所限定的范围。
- 在“student”关系中，name属性的域是文字字符，birthday属性的域是日期。在“score”关系中，score属性的域是0-100的数值。

ü 关键字

- 能唯一标识元组的属性或属性的组合称为关键字。在数据表中，能标识记录唯一性的字段或字段的组合，称为主关键字或候选关键字。
- 在“student”关系中，每一位学生的studentid是唯一的，故“studentid”可作为student表的关键字。而两位学生的姓名可能相同，所以“name”就不能作为student表的关键字。

ü 外部关键字

- 如果关系中的某个属性不是本关系的关键字，而是另一关系的关键字，这个属性就称为外部关键字。

关系型数据库

➢关系的特点

✓在关系模型中，每个关系模式必须满足一定的条件，具备以下特点。

✓关系必须规范化。最基本的要求是每个属性必须是不可分割的数据单元，即每个属性不能再细分为几个属性。

✓在一个关系中，不能出现相同的属性名。在关系模型中，同一个数据表中不能出现同名的字段。

✓在一个关系中，不能出现完全相同的元组。

✓关系中元组的次序无关紧要，即任意交换两行的位置不影响数据的实际含义。

✓关系中属性的次序无关紧要，即任意交换两列的位置不影响数据的实际含义。

关系型数据库的查询操作

➢希望查询选择001课程的所有学生信息、教师信息、课程信息

SELECT student.姓名, courseChoose.学期, course.课程名称, courseChoose.分数, teacher.姓名

FROM student, courseChoose, course,class,teacher

WHERE (((student.学号)=[courseChoose].[学号]))

And (((student.班级编号) = [class].[班级编号]))

And (((courseChoose.课程号) = [course].[课程号]))

And (((courseChoose.教师编号) = [teacher].[教师编号]))

And ((([course].[课程号]) = '001'));

关系型数据库的查询操作

student.姓	学期	课程名称	分数	teacher.姓名
李强	2022-1	Python程序设计	94	张三丰
李强	2022-2	数据科学	96	张无忌

利用Python语言操作数据库

```
import pyodbc

conn_str = (
    #r'Driver={ACCESS (*.mdb, *.accdb)};'
    r'Driver={Microsoft Access Driver (*.mdb, *.accdb)};'
    r'DBQ=D:\Access\jiaoxueguanli.accdb;'
)

conn = pyodbc.connect(conn_str)
cursor = conn.cursor()

SQL_value = 'select * from student'
cursor.execute(SQL_value)

result = cursor.fetchall()
for row in result:
    for i in range(7):
        print(row[i], "  ", end="")
    print()

#print(result)

cursor.close()
conn.close()
```

```
1   202221120101   王刚   男   2004-07-26 00:00:00   上海   101
2   202221122102   李强   男   2004-07-26 00:00:00   北京   101
3   2021110132   阿拉法提   男   2004-08-01 00:00:00   新疆   101
```

student \ class

ID	学号	姓名	性别	出生日期	籍贯	班级编号	单击以添加
1	202221120101	王刚	男	2004/7/26	上海	101	
2	202221122102	李强	男	2004/7/26	北京	101	
3	2021110132	阿拉法提	男	2004/8/1	新疆	101	
(新建)							

Python访问ACCESS的详细配置和操作：

https://blog.csdn.net/zhengdaqing/article/details/136544985?spm=1001.2014.3001.5502

59

本讲主要内容

➢人工管理

➢文件系统

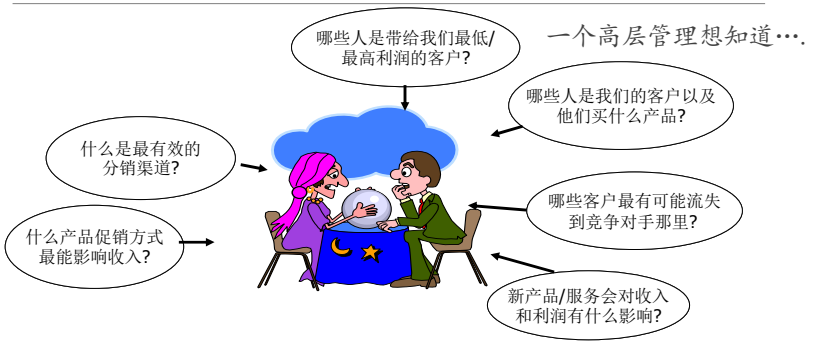
➢数据库系统

➢数据仓库

➢数据湖

➢湖仓一体

为什么需要数据仓库（Data Warehouse）



数据无处不在，然而 …

- 我找不到我所需要的数据
  - 数据分散在网络上的各个地方
  - 数据存在多个版本和格式
- 我不能获取我所需要的数据
  - 需要一个专家来获取数据
- 我无法理解所找到的数据
  - 可得到的数据，但对应的文档说明很糟糕
- 我无法使用所找到的数据
  - 数据需要从一种形式转换到另外一种形式
  - 无法分析有关数据

产生非预期的结果……

数据仓库的目的:
是将大量多种形式数据汇总，使数据创建者和数据用户能够高效灵活地了解数据概况，并且可以深入了解细节。

数据仓库（Data Warehouse）

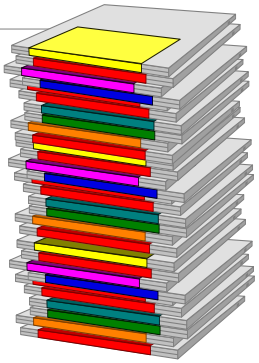
➢数据仓库（Data Warehouse）

✓**是信息的逻辑集合，这些信息来自于许多不同的业务数据库，并用于支持企业的分析活动和决策任务。**

✓**是“单一的、完整的和一致的数据存储，这些数据从多个数据源获取，经过加工成在一定程度上为最终用户可理解的形式，以用于业务管理。”**

强调：单一版本的事实

[Barry Devlin]



数据仓库（Data Warehouse）

➢数据仓库是一个**面向主题**的（Subject Oriented）、**集成**的（Integrated）、**相对稳定**的（Non-Volatile）、**反映时间变化**的（Time Variant）数据集合，用于支持管理决策和信息的全局共享（Bill Inmons, Building the Data Warehouse, 1996）。

✓主题：指用户使用数据仓库进行决策时所关心的重点方面，如：收入、客户、销售渠道等；面向主题，是指数据仓库内的信息是按主题进行组织的，而不是像业务支撑系统那样是按照业务功能进行组织的；

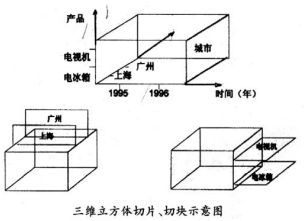
✓集成：指数据仓库中的信息不是从各个业务系统中简单抽取出来的，而是经过一系列加工、整理和汇总的过程，因此数据仓库中的信息是关于整个企业的一致的全局信息。

✓随时间变化：是指数据仓库内的信息并不只是反映企业当前的状态，而是记录了从过去某一时点到当前各个阶段的信息。

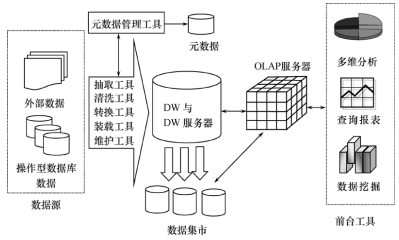
数据仓库（Data Warehouse）

➢集成各种数据

➢按照主题组织数据



三维立方体切片、切块示意图



数据仓库的目标和形式

▪数据仓库的目标，是利用组织里的数据，更加有效地辅助决策过程。

▪数据仓库是一个集成的中央数据库，包含来自于组织内的所有操作型数据源和归档系统。它包含了事务处理系统数据的拷贝、特别是用于查询分析的结构化数据。

➢数据仓库是一个关系或多维数据库，是为查询和分析、而不是事务处理设计的。它将分析工作和事务处理工作分离开来，使企业能够合并来自几个数据源的数据。

➢主要有三种形式的数据仓库：

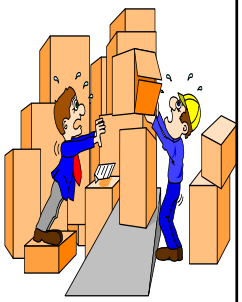
✓企业级数据仓库 - 企业级数据仓库提供的是一个供整个企业进行决策支持的中央数据库。

✓ODS(操作型数据存储) - 它涉及到企业较广范围的数据处理，但不像真正的企业级数据仓库，ODS里的数据接近实时刷新，主要用于常规业务活动。

✓数据集市（Data Mart ）--数据集市是数据仓库的子集，它支持个别区域、业务单元和业务功能。

用户要说的是...

- 应该整合整个企业内的数据
  ----可以消除数据歧义和冗余，可以只从一个数据源获取所需数据。
- 需要一个较高数据粒度的汇总数据
  ----对总体情况有一个清晰的印象，有助于制定长期和战略决策。
- 历史数据有重要作用
  ---更好理解客户行为
- 灵活的决策支持能力
  ---数据和工具的支持



Information

Data

数据仓库是一个将数据转换成信息、使其能及时供最终用户使用的过程。

数据仓库项目60%的时间，花在加载到数据仓库之前的数据预处理过程中。因此，数据仓库中的数据以转换处理后的形式存放。这样有助于快速地进行战略决策。

数据仓库示例

决策支持系统（Decision support systems, DSS）的发展

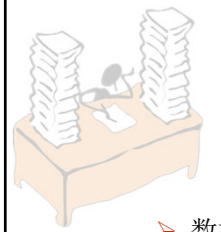
- 60年代: 批处理报表
  - ✓难于查找和分析信息
  - ✓缺乏灵活性，成本昂贵，对于每个新需求都要重新编程
- 70年代: 基于终端的DSS 和 EIS (主管信息系统)
  - ✓可以提供在线处理，只支持独立应用（基于终端的DSS），难获得整合信息，仍然缺乏灵活性，没有和桌面工具集成起来。

决策支持系统（Decision support systems, DSS）的发展

- 80年代:桌面级数据访问和分析工具
  - ✓DSS尝试包含查询工具，电子表格，图形界面
  - ✓易于使用，但是只能访问操作型数据库
- 90年代至今: 带有集成OLAP引擎和工具的数据仓库，实时数据仓库

数据挖掘如何和数据仓库共同工作?

- 数据仓库是企业的数据存储



- 数据挖掘是挖掘出企业数据中的知识

数据挖掘和数据仓库一起有效工作就变成为可能

原因如下：

1.有了相应技术的支持，使得数据仓库存有大量的数据(如：操作型数据，行为数据，以及人口统计学数据)。例如：通过条形码输入数据，通过电子商务网站等方法。

2.来自于各个数据源的数据经过清洗后，数据格式更容易满足需要。

3.现在有许多先进的统计技术，如人工神经网络等用于分析复杂的数据。

---- Gartner Group的调查

使用数据仓库的好处

- 可靠的报表
- 快速得到数据
  - ✓增强了数据的统一和整合，如名字，关键结构，粒度层次。
- 集成的数据
- 灵活的数据展示方式
- 更好的决策制定

为什么要从数据库分离出数据仓库?

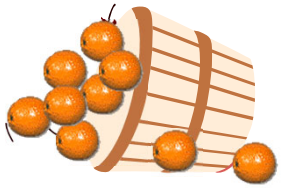
性能

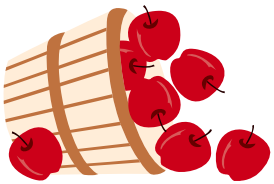
- 操作型数据库设计，应用于已知事务和工作量的场合，是为静态查询设计优化的，如每日的事务处理，而数据仓库是为大数据量的场合设计优化的。
- 对于操作型系统来说，进行复杂的OLAP查询会使其性能下降。
- 对多维视图和查询，需要特殊的数据组织方式、读取方法、实施方案。

功能

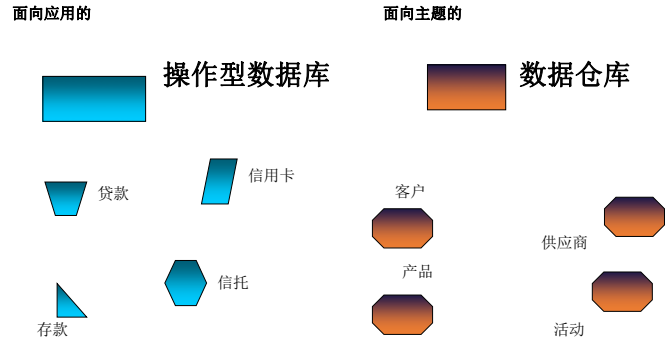
- 遗失数据：决策支持需要历史数据，而操作型数据库并不保存它。
- 数据整合：决策支持需要整合（聚合，汇总）来自多个异构数据源（操作型数据库，外部数据源）的数据。
- 数据质量：不同数据源经常使用不一致的数据表现形式、代码、格式，需要统一它们。
- 操作型数据库中，数据组织方式是满足范式要求的。而在数据仓库里存放的是经过逆范式处理的数据，可以只从一个数据点就可以获取所需数据。

数据库和数据仓库，二者的区别是什么?





面向应用的 vs 面向主题的



数据库	数据仓库 (DSS)
➢用来运行事务处理	➢用来分析业务
➢详细的数据	➢汇总和精炼的数据
➢当前最新数据	➢快照数据
➢孤立的数据	➢集成的数据
➢重复访问	➢即席查询
➢操作人员	➢知识用户 (管理者)

78

数据库	数据仓库
➢对性能敏感	➢对性能不敏感
➢同时访问少量的记录(数十条)	➢同时访问大量的数据 (数百万条)
➢可以读和更新	➢主要是读(批量更新)
➢没有数据冗余	➢存在数据冗余
➢DB大小:100MB -100 GB	➢DB大小:100 GB - TB
➢用户：数百个- 数千个	➢用户：数个- 数百个

79

比较项目	数据库	数据仓库
特征	操作处理	信息处理
面向	事务	分析
用户	DBA,数据库专业人员	业务分析员
功能	日常操作	长期信息需求，决策支持
DB结构	基于E-R，面向应用	星型/雪花，面向主题
数据	当前的	历史的
汇总	原始的，高度详细	汇总的，统一的
视图	一般关系	多维的
查询	简单事务	复杂查询
存取	读/写（修改）	读
操作	主关键字上索引或散列	大量扫描
访问数据量	数笔	数百万笔
用户数	数千	数百
DB2规模	MB/GB	GB/TB
优先级	高性能，高可用性	高灵活性，端点用户自治
度量	事务吞吐量	查询吞吐量，响应时间

总的来说...

- 数据库 系统用来“运行”事务处理

- 数据仓库用来帮助“优化”业务

数据管理解决问题的思路：解耦

人工管理 → 文件系统 → 数据库系统 → 数据仓库系统

82

本讲主要内容

- 人工管理
- 文件系统
- 数据库系统
- 数据仓库
- 数据湖
- 湖仓一体

数据湖

Ø湖、海、河的差异

- ü “河”强调流动性
- ü “湖”天然分层，需要治理，否则“数据沼泽”
- ü “海”：无边无界

Ø数据湖的特点

- ü 湖是沉淀的
- ü 湖是分层的
- ü 湖是需要治理的

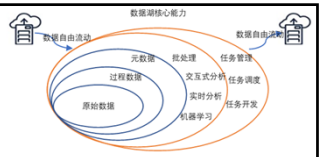
数据湖的概念

➢数据湖的多种定义

✓Wikipedia：数据湖是一类存储数据自然/原始格式的系统或存储，通常是对象块或者文件。数据湖通常是企业中全量数据的单一存储。全量数据包括原始系统所产生的原始数据拷贝以及为了各类任务而产生的转换数据，各类任务包括报表、可视化、高级分析和机器学习。数据湖中包括来自于关系型数据库中的结构化数据（行和列）、半结构化数据（如CSV、日志、XML、JSON）、非结构化数据（如email、文档、PDF等）和二进制数据（如图像、音频、视频）。Hadoop是最常用的部署数据湖的技术。

✓AWS：数据湖是一个集中式存储库，允许您以任意规模存储所有结构化和非结构化数据。用户可以按原样存储数据（无需先对数据进行结构化处理），并运行不同类型的分析-从控制面板和可视化到大数据处理、实时分析和机器学习，以指导做出更好的决策。

数据湖的特性



➢数据湖提供足够用的数据存储，可以存储一个企业/组织中的所有数据。

➢数据湖可以存储海量的任意类型的数据，包括结构化、半结构化和非结构化数据。

➢数据湖中的数据是原始数据，是业务数据的完整副本。数据湖中的数据保持了他们在业务系统中原来的样子。

➢数据湖需要具备完善的数据管理能力（完善的元数据），可以管理各类数据相关的要素，包括数据源、数据格式、连接信息、数据schema、权限管理等。

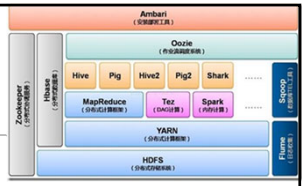
➢数据湖需要具备多样化的分析能力，包括但不限于批处理、流式计算、交互式分析以及机器学习；同时，还需要提供一定的任务调度和管理能力。

➢数据湖需要具备完善的数据生命周期管理能力。不光需要存储原始数据，还需要能够保存各类分析处理的中间结果，并完整的记录数据的分析处理过程，能帮助用户完整详细追溯任意一条数据的产生过程。

➢数据湖需要具备完善的数据获取和数据发布能力。数据湖需要能支撑各种各样的数据源，并能从相关的数据源中获取全量/增量数据；然后规范存储。数据湖能将数据分析处理的结果推送到合适的存储引擎中，满足不同的应用访问需求。

➢数据湖支持大数据，包括超大规模存储以及可扩展的大规模数据处理能力。

数据湖基本架构的演化（一）



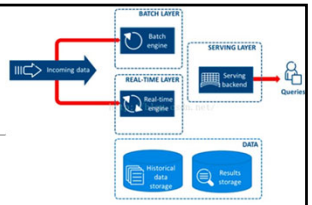
➢以Hadoop为代表的离线数据处理基础设施

✓以HDFS为核心存储，以MapReduce为基本计算模型的批量数据处理架构;

✓围绕HDFS和MR，不断完善大数据平台的数据处理能力

- 面向在线KV操作的HBase；
- 面向SOL的HIVE；
- 面向工作流的PIG；

数据湖基本架构的演化（二）

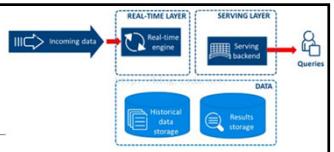


➢Lambda架构

✓Lambda架构采用流式计算引擎，其核心理念是“流批一体”；

✓数据流入平台后一分为二，一部分走批处理模式，另一部分走流式计算模式；

数据湖基本架构的演化（三）



➢Kappa架构

✓Kappa架构是基于流计算的理念；

✓加大流计算的并发性，统一处理批处理和流式处理两种计算模式；

➢Kappa架构的优缺点

✓逐渐囊括了应用所需的各类数据处理能力，大数据平台成为了一个全量数据处理平台；

✓除了关系型数据库，其余的数据几乎都被纳入到大数据平台进行统一的处理。

✓数据湖针对解决存储和计算，而忽略的数据资产化管理。

数据湖组件参考架构

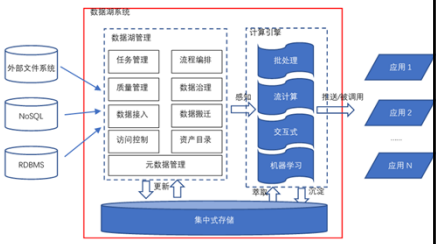
➢数据湖与大数据平台相同的地方是具备处理超大规模数据所需的存储和计算能力；

➢数据湖具备更为完善的数据管理能力

✓更强大的数据接入能力

✓更强大的数据管理能力

✓可共享的元数据



AWS数据湖解决方案

➢基于AWS Lake Formation构建，AWS Lake Formation本质上是一个管理性质的组件，它与其他AWS服务互相配合，来完成整个企业级数据湖构建功能。

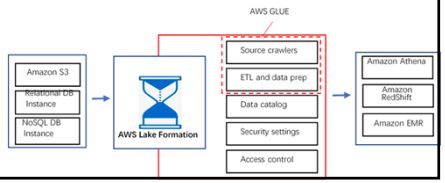
✓数据流入：元数据流入（数据资源目录“安全设置、访问控制策略”）、业务数据流入

✓数据应用：通过各类外部计算引擎，提供丰富的计算模式支持；

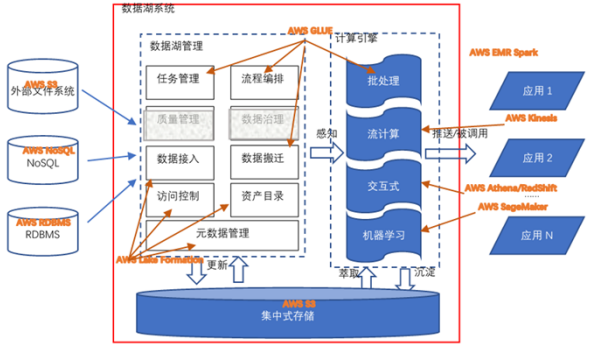
✓数据计算：AWS GLUE进行基本的数据处理；

✓数据沉淀：Amazon S3；

✓权限管理



数据湖组件VS AWS数据湖解决方案

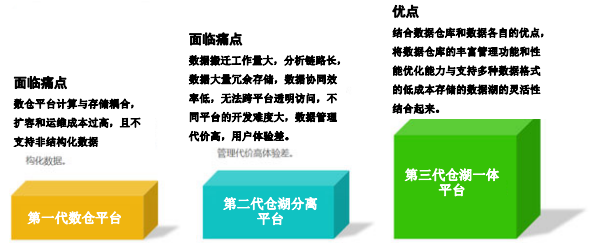


本讲主要内容

- 人工管理
- 文件系统
- 数据库系统
- 数据仓库
- 数据湖
- 湖仓一体

下一代演进方向：湖仓一体

- 湖仓一体早期方案
  - ✓Facebook：拥有一个300PB的Hadoop数据湖，仍然需要使用一个50TB数据仓库支持决策分析；
  - ✓LinkedIn：有一个规模较大的Hadoop数据湖，末端使用Aster Data数仓做运营分析

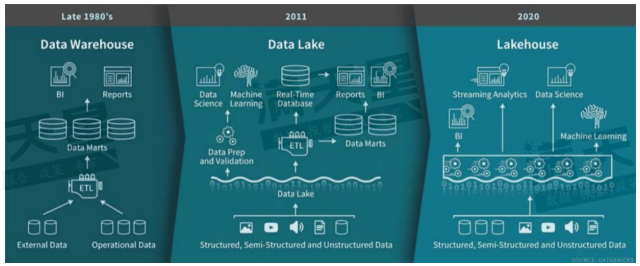


湖仓一体（Data Lakehouse）

- 湖仓一体：一种新出现的数据架构，同时吸收了数据仓库技术和数据湖存储技术的优势，为企业提供一个统一、可共享的数据底座（DataBricks）。
- 湖仓一体试图去融合数仓和数据湖这两者之间的差异，通过将数仓构建在数据湖上，使得存储变得更为廉价和弹性，同时湖仓一体能够有效地提升数据质量，减小数据冗余。

湖仓一体

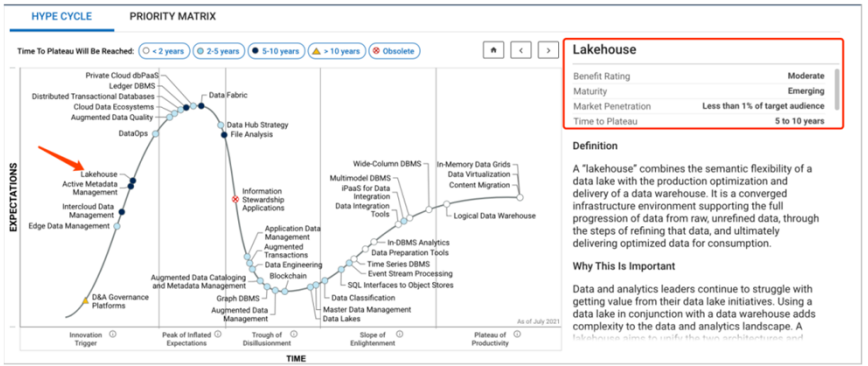
- 湖仓一体技术发展历程



数据仓库、数据湖、湖仓一体关键技术对比

	数据仓库	数据湖	湖仓一体
数据类型	来自事务系统、运营数据库和业务线应用程序的关系数据	所有类型的数据，结构化、半结构化和非结构化	所有类型的数据，结构化、半结构化和非结构化
产品形态	标准化产品	架构	架构
使用成本	起步成本高，但数据量增长带来的成本增加呈线性	起步成本低，但数据量增长带来的成本增加较快	起步成本低，与数据湖有同样的数据增长性，综合成本有效下降
数据质量	入库前需要高度清洗及提炼，可作为重要事实依据的高度监管数据	包含原始数据在内的任何数据，使用前需要清洗和标准化处理	包含原始数据在内的任何数据，使用过程中流式清洗提炼，并可按需保存为标准化的事实数据
功能	通过历史的结构化数据进行数据分析	机器学习、探索性分析、数据发现、流处理	实时查询、实时分析、探索性分析、数据发现，可利用机器学习通过流处理持续动态地建立数据模型
优势	1. 完全面向分析构建，分析结果更准确； 2. 统一的数据存储解决方案可增强决策制定能力； 3. 可做出更具成本效益的决策	1. 可保留原始数据的全部信息； 2. 支持实时数据源； 3. 更好的可扩展性和敏捷性； 4. 更快的准备数据	1. 具备完善的数据管理能力； 2. 数据可以实现即时追溯； 3. 丰富的计算引擎； 4. 更高的数据实时性
劣势	1. 成本较高； 2. 无法存储非结构化数据； 3. 造成原始数据的部分信息丢失	1. 数据过于原始，使用起来复杂； 2. 数据缺乏一致性、隔离性，无法保证数据质量	1. 实现湖仓间的数据无缝打通和顺畅流动较难； 2. 市场未形成行业共识标准

湖仓一体的数据成熟度曲线



谢 谢!

Questions & Comments