大数据系统

Lecture 1: Intro

Sept 6，2021

本节概要

- 课程背景
- 课程安排
- 章节组织
- 预备知识

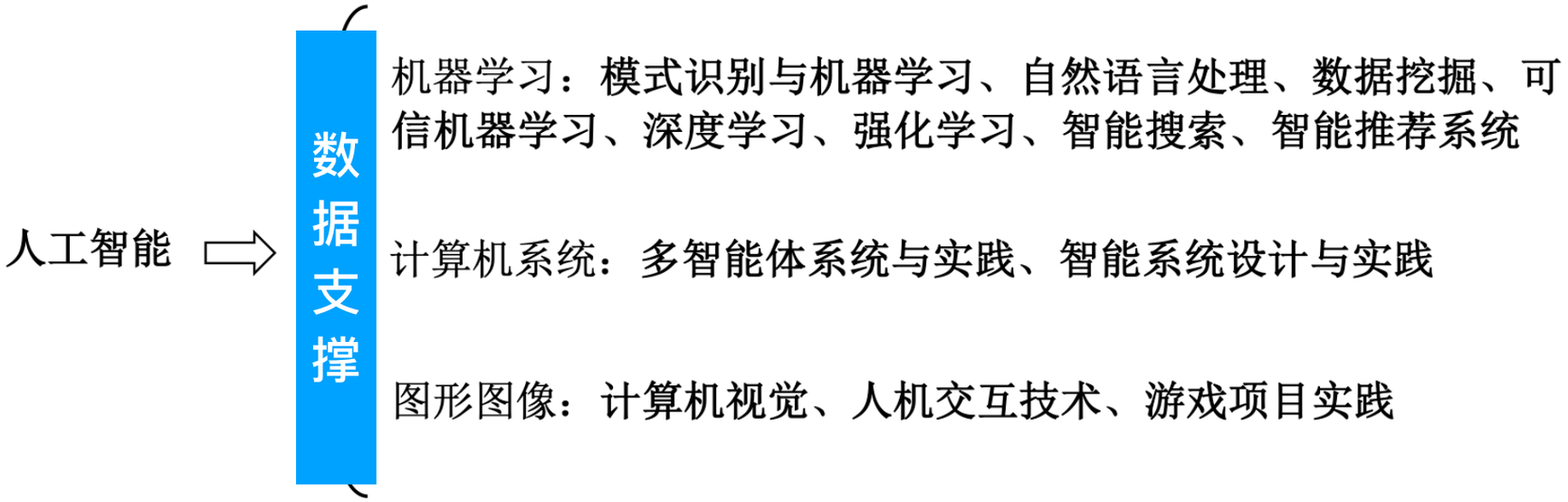
1. 课程背景

培养方案中的定位

六、专业核心课程

包括：《计算机导论》、《程序设计原理与 C 语言》、《编程思维与实践》、《数据结构》、《数字逻辑及实验》、《计算机组成与结构》、《操作系统》、《数据库系统原理》、《数据库系统实践》、《嵌入式系统原理与实践》、《计算机网络》、《编译原理》、《编译原理实践》、《人工智能》。

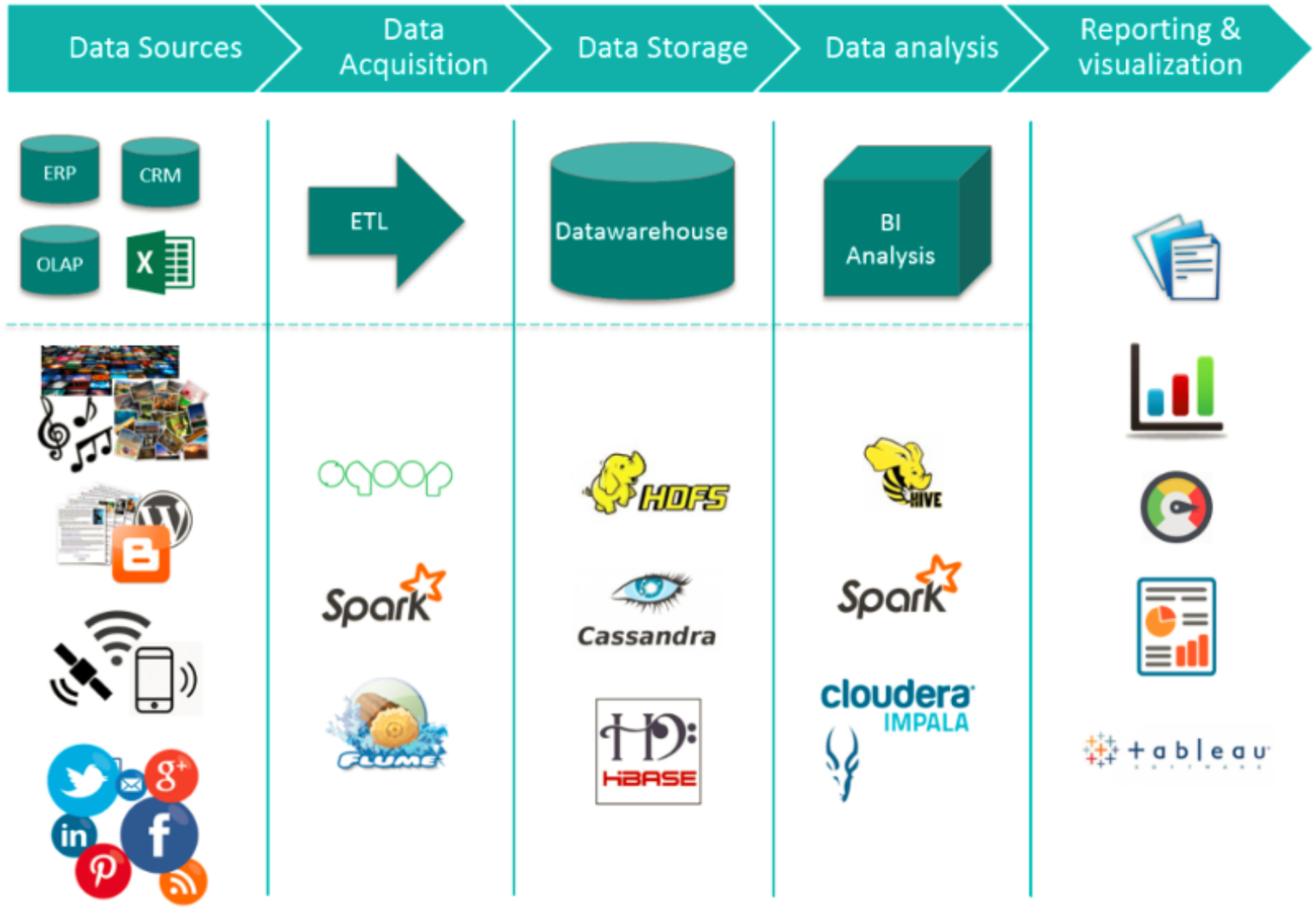
人工智能课程群如下图所示：



http://www.cs.ecnu.edu.cn/98/f3/c24558a301299/page.htm 2020级培养方案

章标题	知识点
一、数据库系统概论	1.1 数据库系统的应用和目标◆
	1.2 数据库的基本概念
	1.3 数据视图和数据模型◆
	1.4 关系数据库及其设计过程◆
	1.5 数据库体系结构及数据库历史
二、关系模型和关系代数	2.1 关系数据库的结构（表、属性、主外码）◆
	2.2 数据库模式图
	2.3 数据库模式和实例
	2.4 关系查询语言和关系代数◆
三、SQL	3.1 SQL 数据类型定义
	3.2 SQL DML 和 DDL 的基本结构◆
	3.3 SQL 附加的基本运算和空值处理◆
	3.4 聚集函数◆
	3.5 嵌套子查询◆
	3.6 数据库的修改操作
四、中级 SQL	4.1 SQL 的连接表达式◆
	4.2 数据库视图◆
	4.3 事务的基本概念和数据完整性约束◆
	4.4 SQL 的数据类型与模式
	4.5 数据库授权操作◆
五、数据库设计和 E-R 设计	5.1 数据库设计过程概览
	5.2 E-R 图的基本概念和表示方法◆
	5.3 E-R 图转换为关系模式◆
	5.4 实体-联系设计问题◆
	5.5 扩展的 E-R 特性
六、关系数据库设计	6.1 关系数据库模式设计目标
	6.2 原子域和第一范式
	6.3 第二范式和第三范式◆
	6.4 BC 范式◆
	6.5 函数依赖理论◆
	6.6 第三范式和 BC 范式的分解算法◆
七、数据存储结构和索引	7.1 数据库系统文件和记录的组织形式
	7.2 数据字典和缓冲区
	7.3 数据库索引的目的、种类和基本概念◆
	7.4 B+树索引文件◆
八、事务管理	8.1 事务的基本概念及其 ACID 特性
	8.2 事务的可串行化和可恢复性◆
	8.3 事务的并发控制协议◆
	8.4 数据库恢复系统◆
九、大数据	9.1 大数据定义及特征
	9.2 大数据存储系统◆
	9.3 GFS、MapReduce◆
	9.4 流数据和图数据库简介

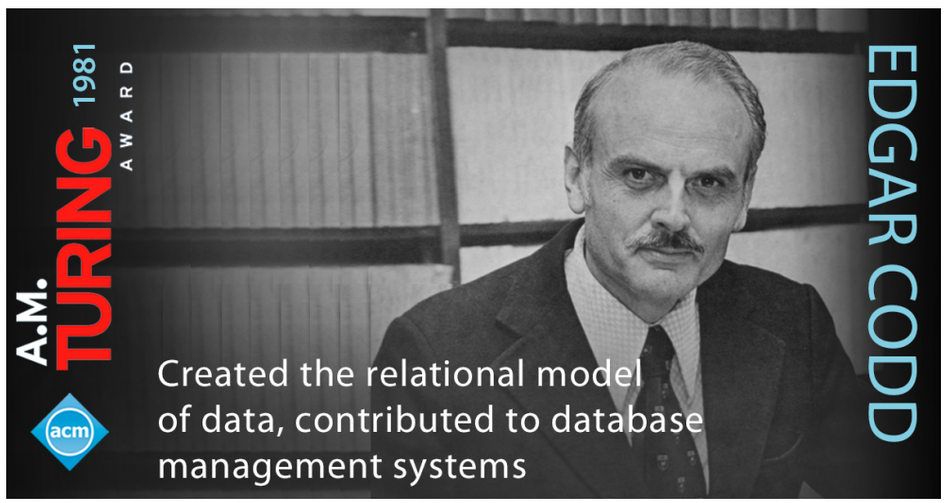
数据库必修课



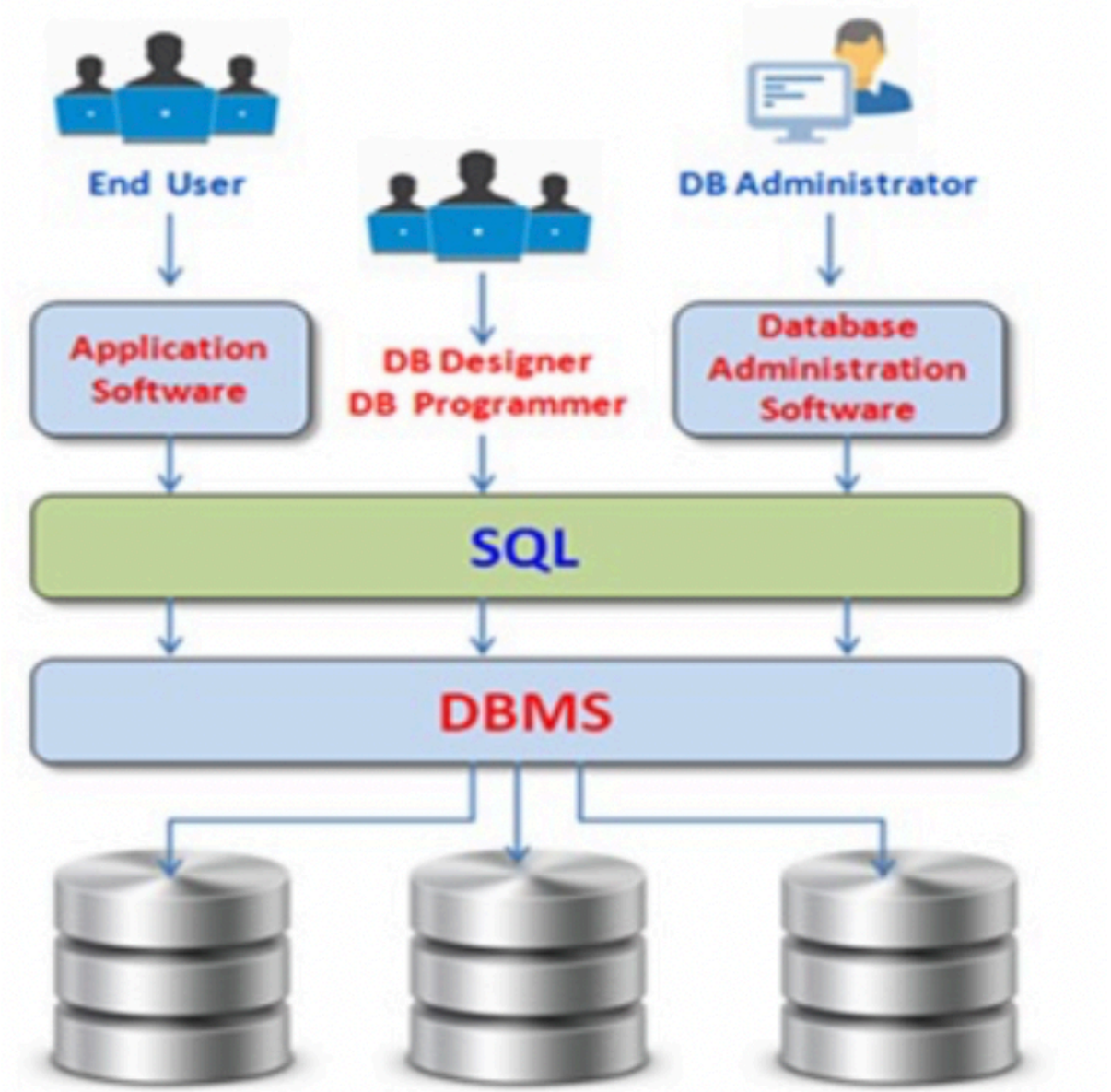
数据库领域的图灵奖


A.M.
TURING
AWARD
2014
MICHAEL STONEBRAKER
Developed and commercialized
relational database systems

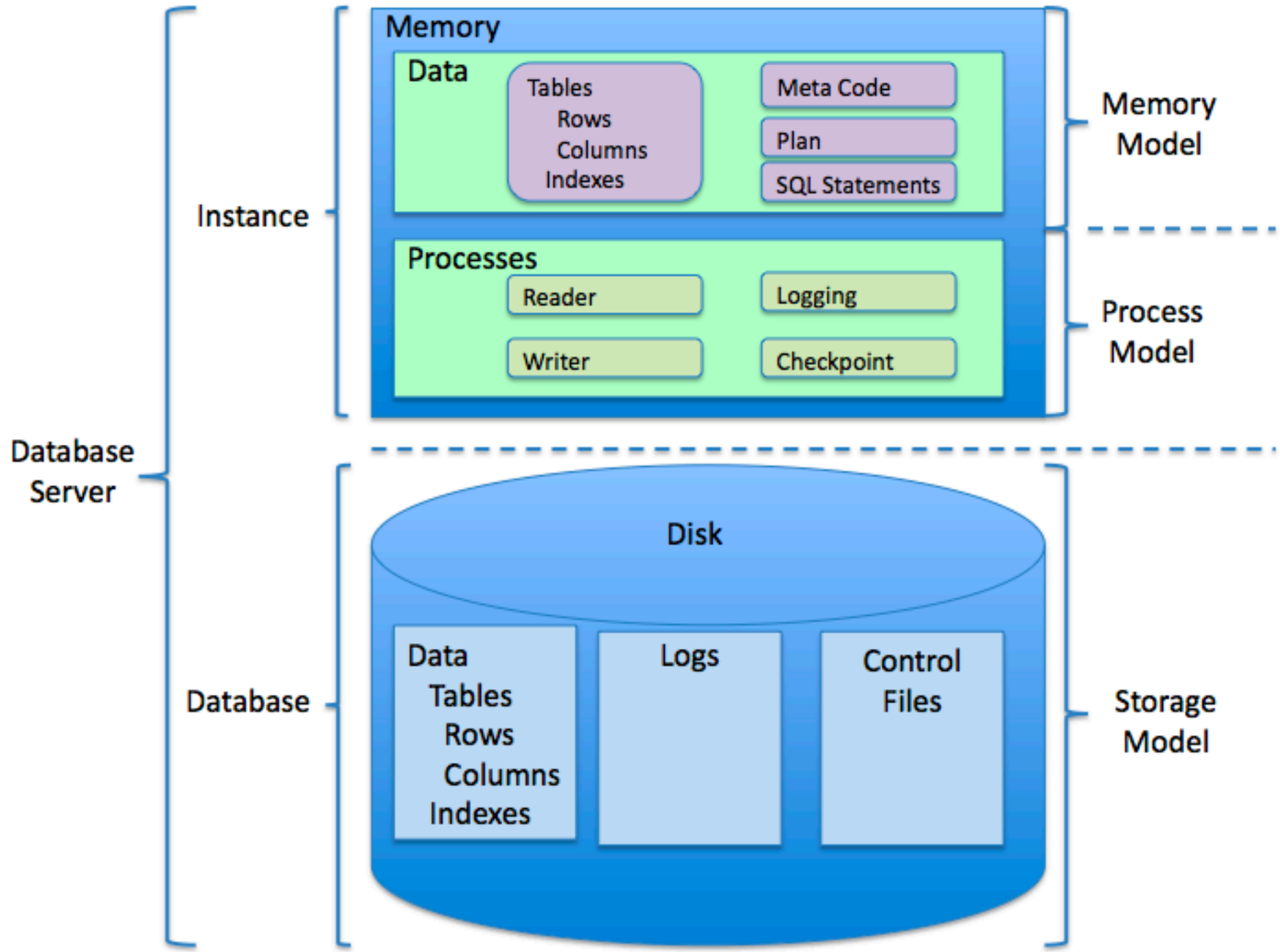

A.M.
TURING
AWARD
1998
JIM GRAY
Transaction processing


A.M.
TURING
1981
AWARD
EDGAR CODD
Created the relational model
of data, contributed to database
management systems

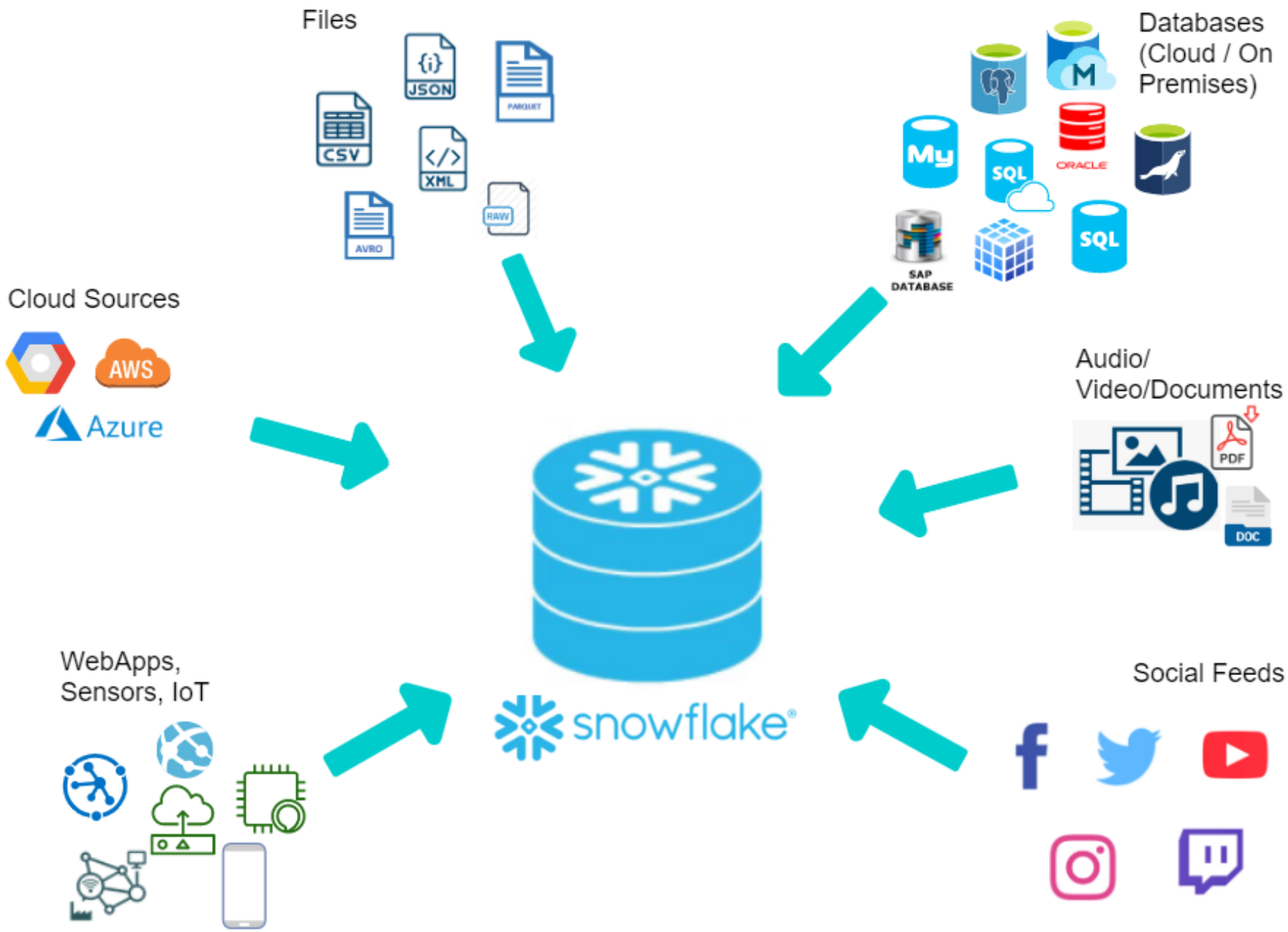
关系数据库场景


End User
DB Administrator
Application Software
DB Designer
DB Programmer
Database Administration Software
SQL
DBMS

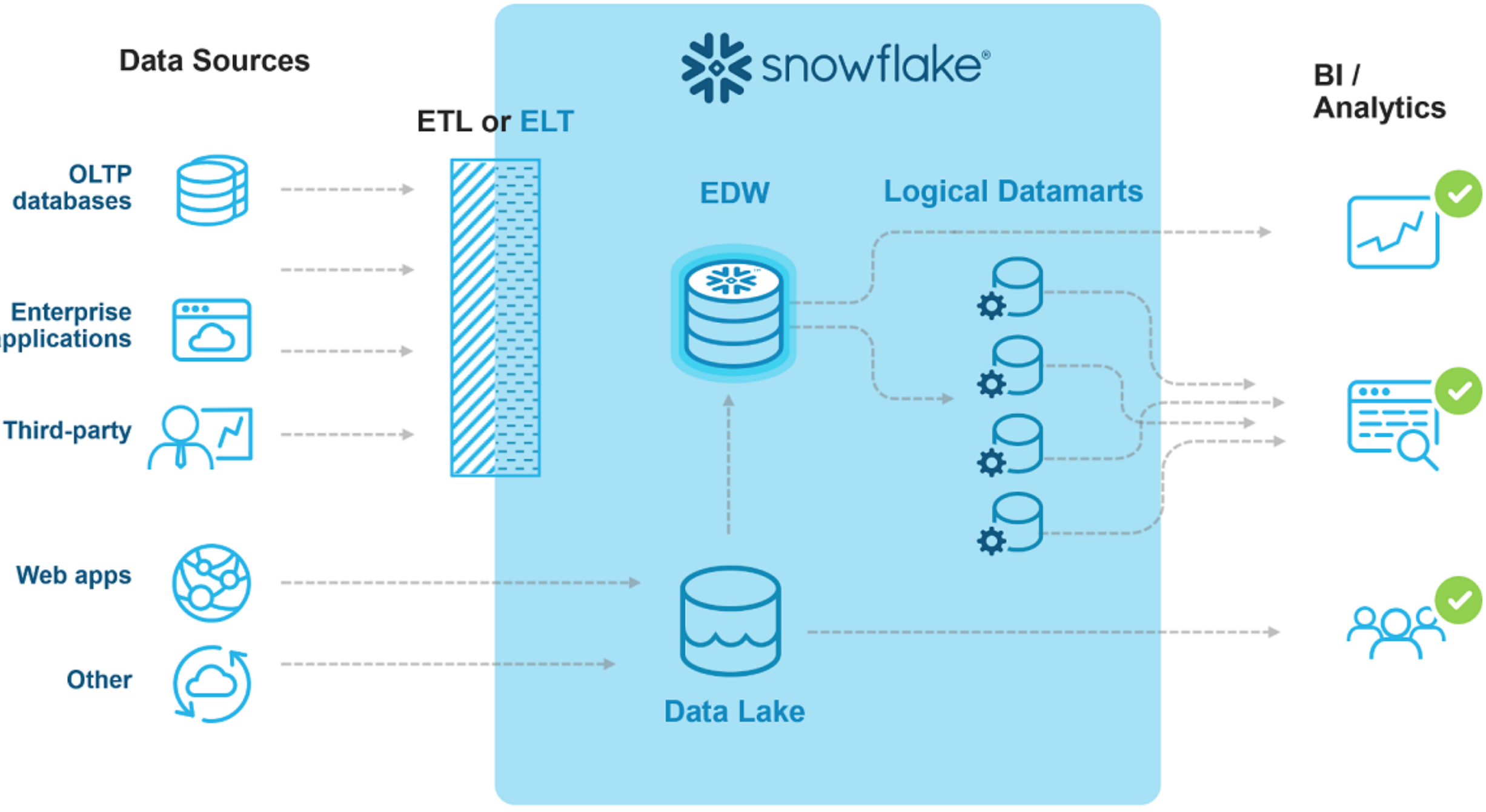
关系数据库架构


Memory
Data
Tables
Rows
Columns
Indexes
Meta Code
Plan
SQL Statements
Processes
Reader
Writer
Logging
Checkpoint
Memory Model
Process Model
Instance
Database Server
Database
Disk
Data
Tables
Rows
Columns
Indexes
Logs
Control Files
Storage Model

snowflake


Files
JSON
PARQUET
CSV
XML
RAW
AVRO
Databases
(Cloud / On
Premises)
My
SQL
ORACLE
SQL
SAP
DATABASE
Cloud Sources
AWS
Azure
Audio/
Video/Documents
PDF
DOC
snowflake
WebApps,
Sensors, IoT
Social Feeds

Snowflake


Data Sources
snowflake
ETL or ELT
BI / Analytics
OLTP databases
EDW
Logical Datamarts
Enterprise applications
Third-party
Web apps
Other
Data Lake

数据是基石

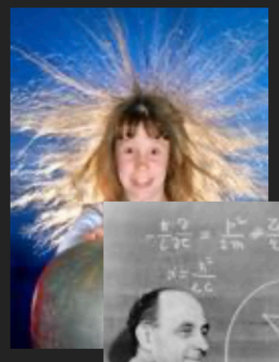
The Unreasonable Effectiveness of Data

Alon Halevy, Peter Norvig, and Fernando Pereira, Google

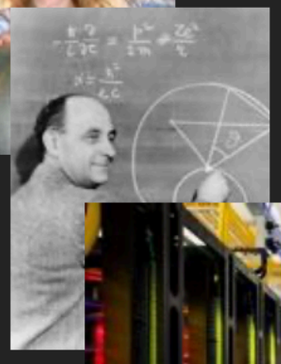
https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/35179.pdf

数据对科学的推动

Introduced the idea of the
Fourth Paradigm of Science



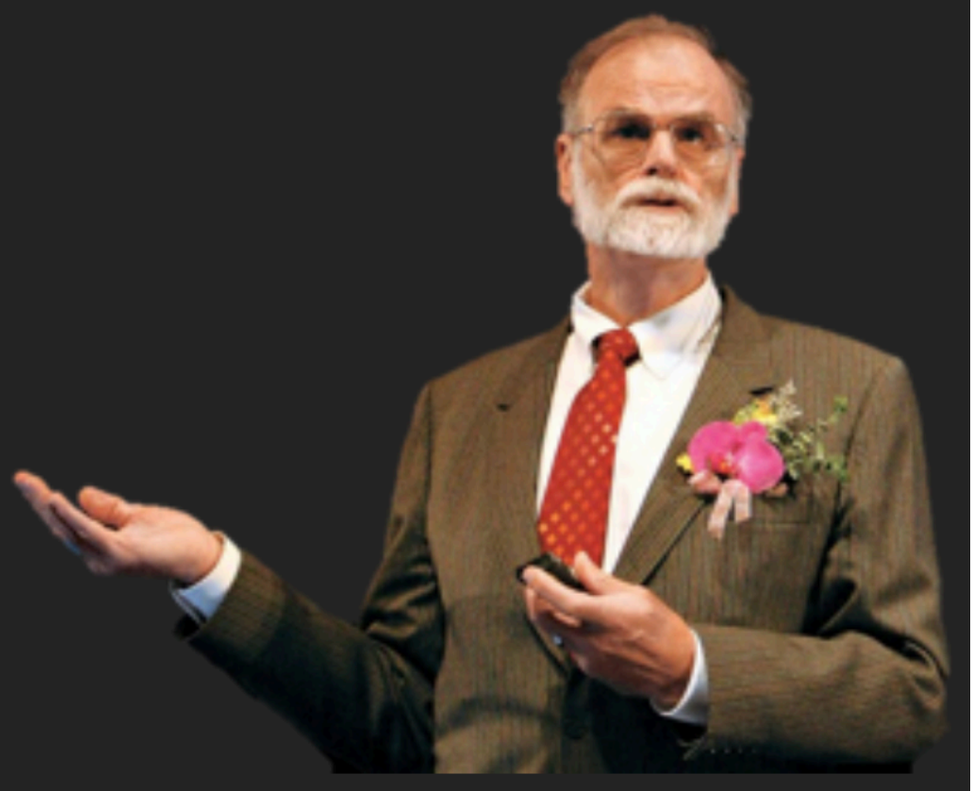
Experimental



Theoretical

Simulation

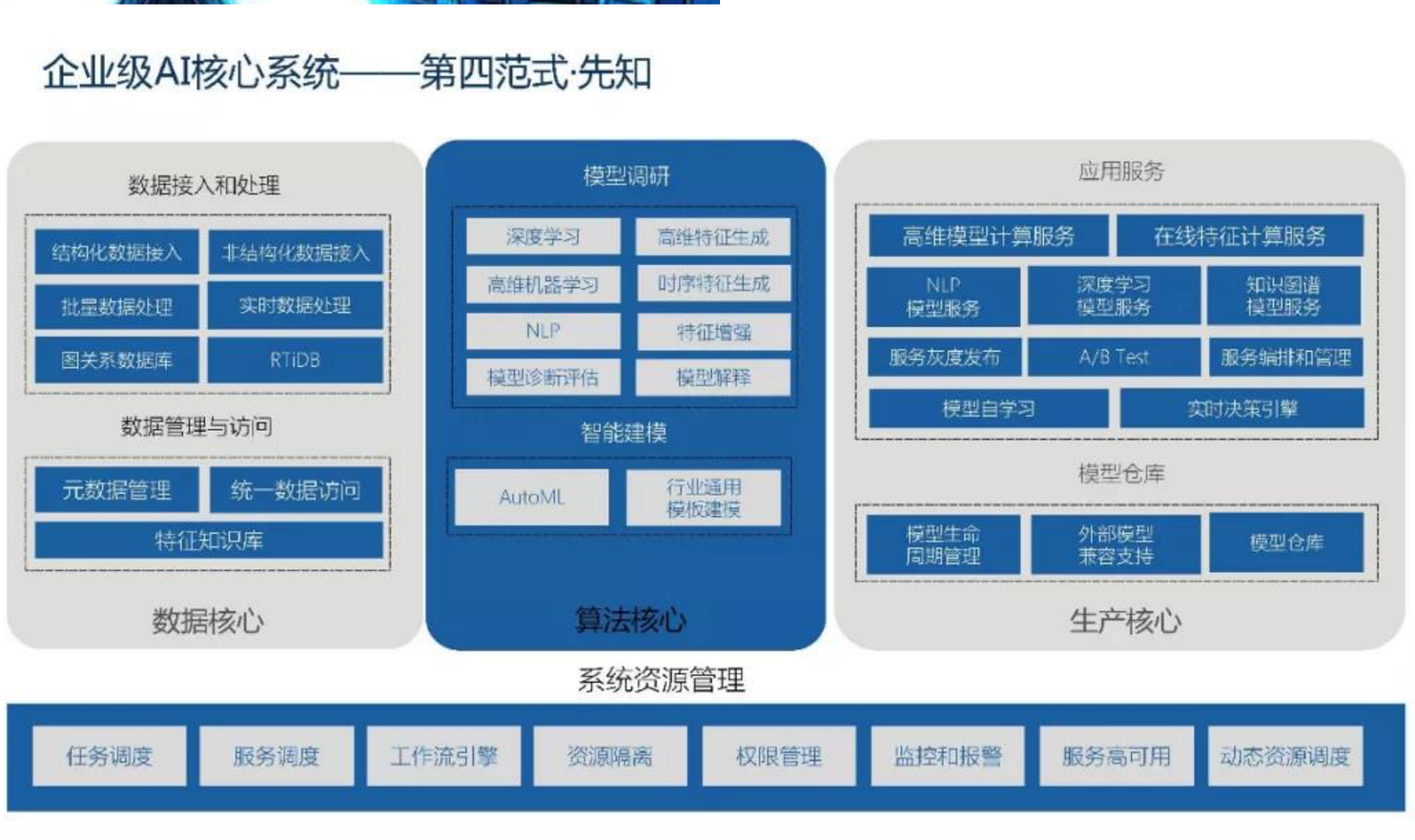
Data
Intensive



Jim Gray

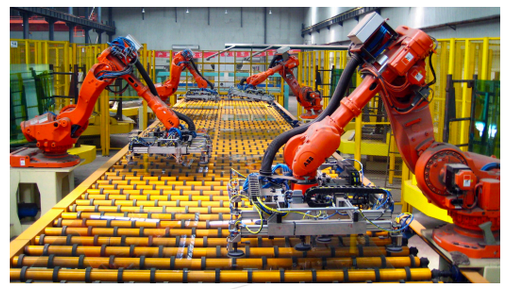
‘第四范式’


艾问观察：第四范式IPO——数据的极端与浪漫
IASK艾问
向世界讲好中国创新故事

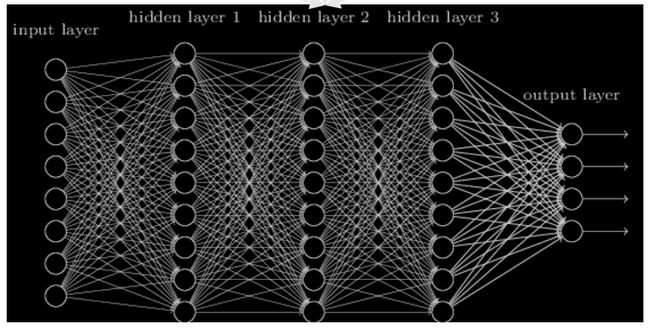

企业级AI核心系统——第四范式·先知
数据接入和处理
结构化数据接入
非结构化数据接入
批量数据处理
实时数据处理
图关系数据库
RTiDB
数据管理与访问
元数据管理
统一数据访问
特征知识库
数据核心
模型调研
深度学习
高维特征生成
高维机器学习
时序特征生成
NLP
特征增强
模型诊断评估
模型解释
智能建模
AutoML
行业通用
模板建模
算法核心
应用服务
高维模型计算服务
在线特征计算服务
NLP
模型服务
深度学习
模型服务
知识图谱
模型服务
服务灰度发布
A/B Test
服务编排和管理
模型自学习
实时决策引擎
模型仓库
模型生命
周期管理
外部模型
兼容支持
模型仓库
生产核心
系统资源管理
任务调度
服务调度
工作流引擎
资源隔离
权限管理
监控和报警
服务高可用
动态资源调度

数据是新的架构基石

Software 2.0 is eating Software 1.0



1000x Productivity: Google shrinks language translation code from 500k LoC to 500 lines of dataflow.



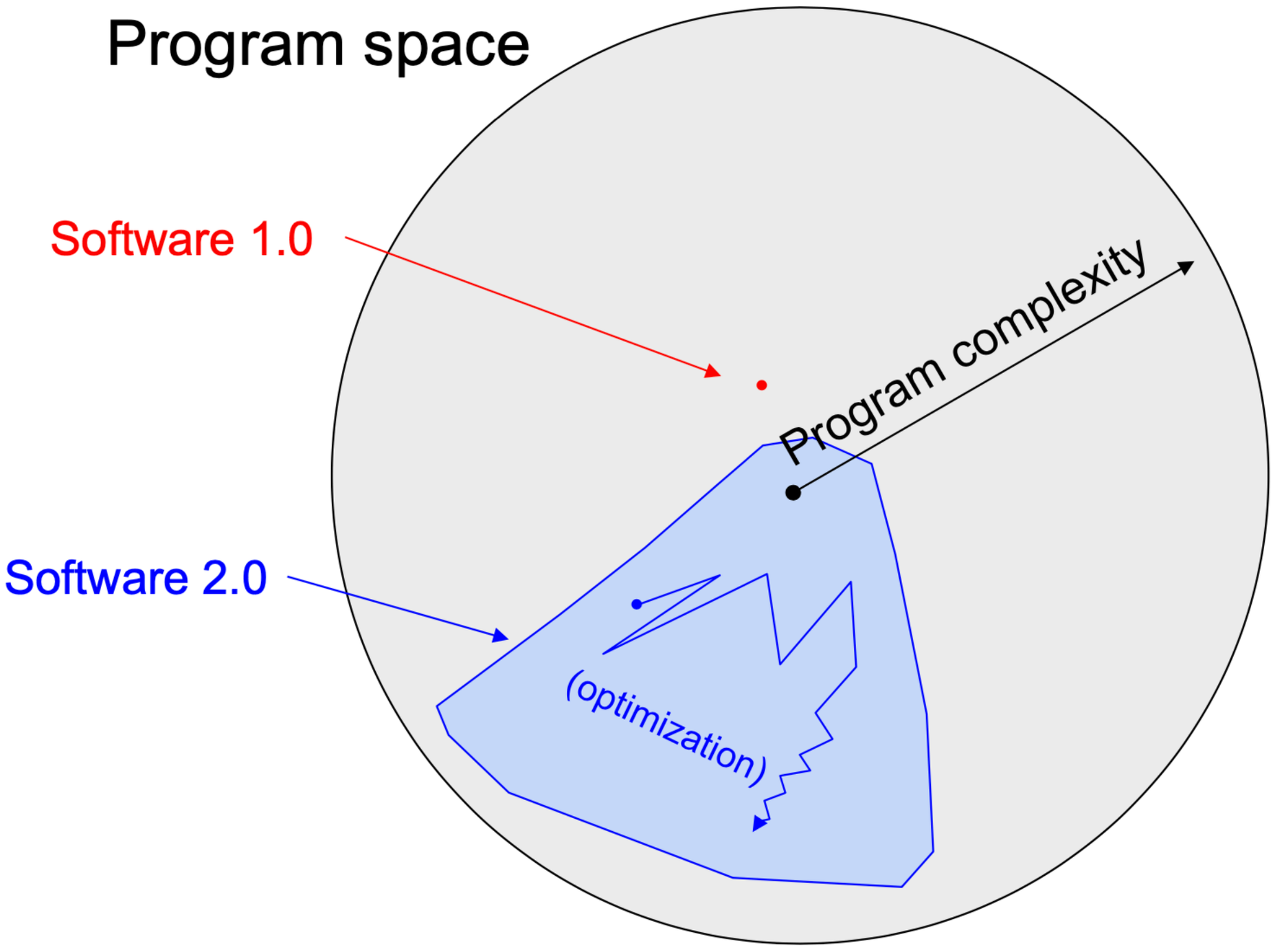
Classical problems ML 1st

- ETL & Cleaning (Holoclean.io)
- DB Tuning Peloton (CMU)
- Networks Pensieve (MIT) .
- NeuroCore (Stanford)

ıttps://jack-clark.net/2017/10/09/import-ai-63-google-shrinks-language-translation-code-rom-500000-to-500-lines-with-ai-only-25-of-surveyed-people-believe-automationbetter-jobs

"Software 2.0", Andrej Karpathy, https://medium.com/@karpathy/software-2-0-a64152b37c35

工程化需求


Program space
Software 1.0
Program complexity
Software 2.0
(optimization)

工程化需求

If optimization is doing most of the coding, what are the humans doing?

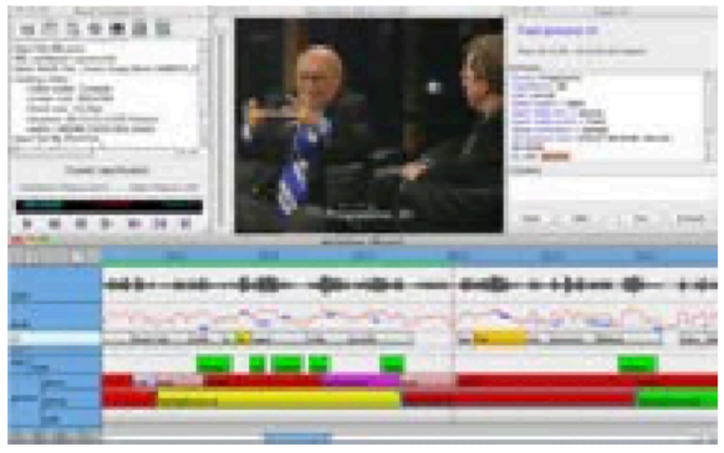
- Design and develop cool algorithms
- Analyze running times

Algorithm 1 GreedyPlus Algorithm

```
alignment ← null
EAWeight ← user input
while ∃u, v s.t. [(u ∈ D1 and v ∈ D2) or (u ∈ B1 and v ∈ B2)] and u, v ∉ alignment do
    bestPair ← null
    for all u, v do
        if Score(u, v) + EA(u, v) > Score(bestPair) + EA(bestPair) then
            bestPair ← (u, v)
        end if
        alignment ← alignment + bestPair
    end for
end while

EA(u,v) {
EAScore ← 0
for all (w, x) ∈ alignment do
    if (u, w) ∈ E1 and (v, x) ∈ E2 then
        EAScore ← EAScore + EAWeight
    end if
end for
return EAScore
}
```

1. Label



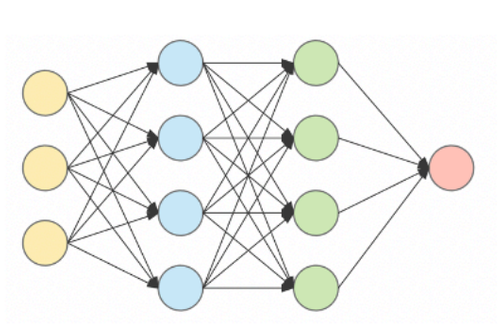
2. Maintain surrounding “dataset infrastructure”

- Flag labeler disagreements, keep stats on labelers, “escalation” features
- Identify “interesting” data to label
- Clean existing data
- Visualize datasets

算法对训练数据的需求

ML Application =

Model + Data + Hardware



?

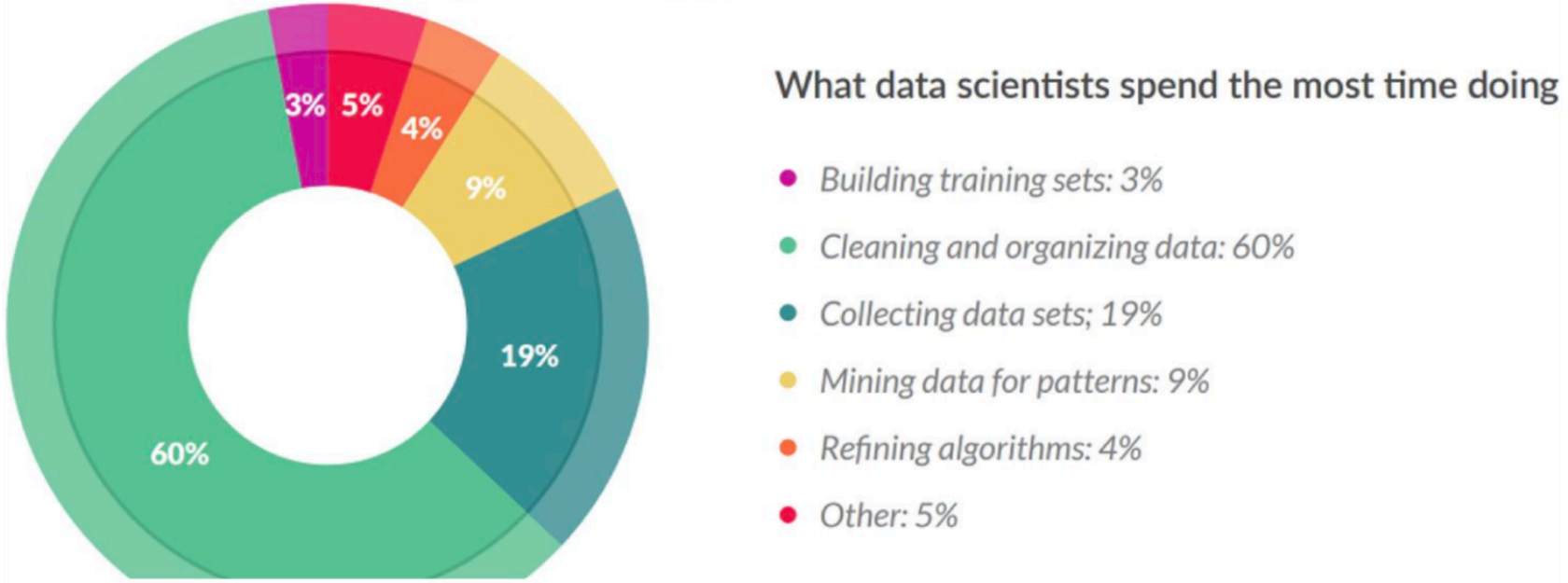
State-of-the-art models and hardware are available.

Training data is not

训练数据的高成本

The Achilles' Heel of Modern Analytics

is low quality, erroneous data



Cleaning and organizing the data comprises 60% of the time spent on an analytics or AI project.

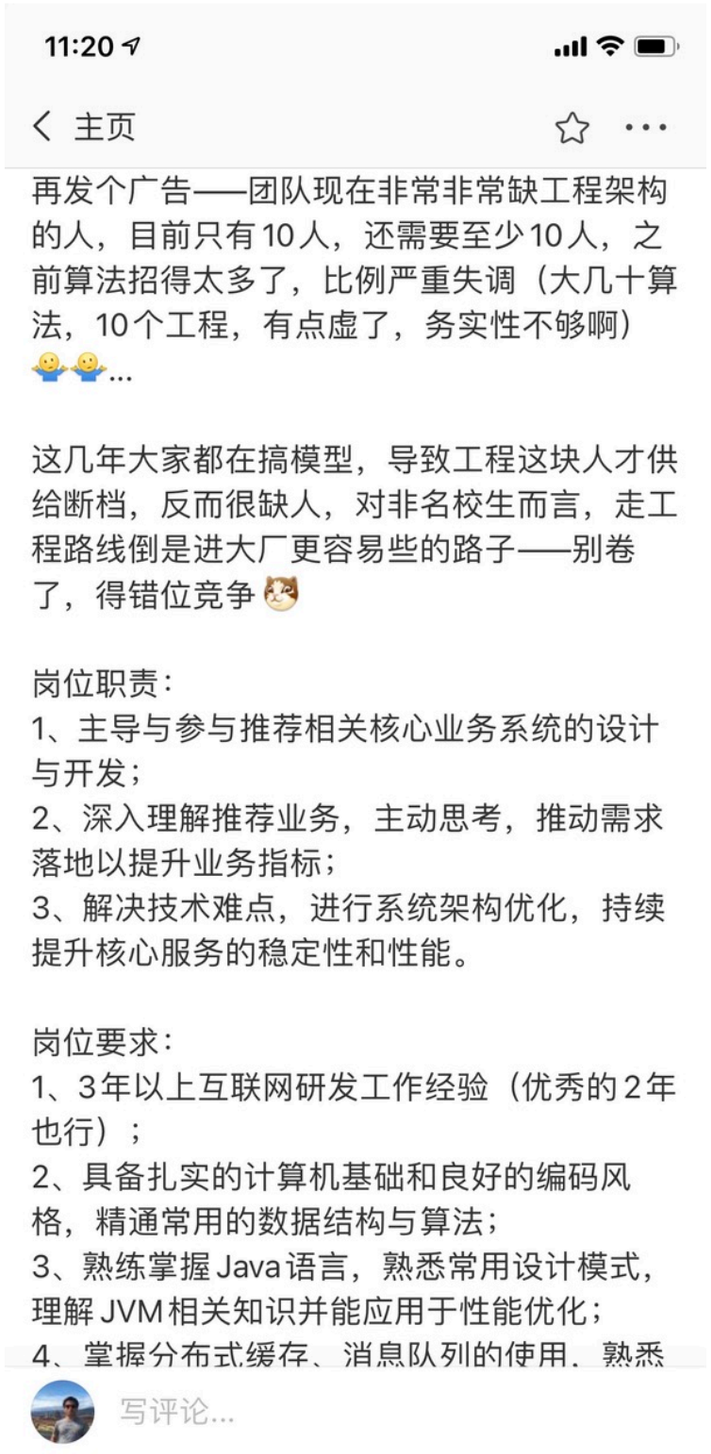
训练数据的高成本

那些给人工智能打工的人

发布时间：2018-10-16 11:09 作者：刘敏 GQ报道 来源：刘敏 GQ报道

摘要：这里是中国特色的数据车间，他们遍布在河南、山东、河北等地的四五线小城里，日以继夜地为世界领先的AI产品服务。

工程化需求


11:20
主页
再发个广告——团队现在非常非常缺工程架构的人，目前只有10人，还需要至少10人，之前算法招得太多了，比例严重失调（大几十算法，10个工程，有点虚了，务实性不够啊）
...
这几年大家都在搞模型，导致工程这块人才供给断档，反而很缺人，对非名校生而言，走工程路线倒是进大厂更容易些的路子——别卷了，得错位竞争
岗位职责：
1、主导与参与推荐相关核心业务系统的设计与开发；
2、深入理解推荐业务，主动思考，推动需求落地以提升业务指标；
3、解决技术难点，进行系统架构优化，持续提升核心服务的稳定性和性能。
岗位要求：
1、3年以上互联网研发工作经验（优秀的2年也行）；
2、具备扎实的计算机基础和良好的编码风格，精通常用的数据结构与算法；
3、熟练掌握Java语言，熟悉常用设计模式，理解JVM相关知识并能应用于性能优化；
写评论...


算法
自然语言处理
计算机视觉
算法工程师
秋招
如何看待2021年秋招算法岗灰飞烟灭?
18年是否值得进入，19年供大于求，20年一片红海诸神黄昏。 去年都诸神黄昏了，今年换个词——“灰飞烟灭”。 21届找算法相关工作的同学不妨进来谈谈感想。...显示全部
已关注
查看回答
邀请回答
好问题 608
22 条评论
分享

2. 课程安排

What Are Data-Intensive Systems?

Relational databases: most popular type of data-intensive system (MySQL, Oracle, etc)

Many systems facing similar concerns: message queues, key-value stores, streaming systems, ML frameworks, your custom app?

Goal: learn the main issues and principles that span all data-intensive systems

Typical System Challenges

Reliability in the face of hardware crashes, bugs, bad user input, etc

Concurrency: access by multiple users

Performance: throughput, latency, etc

Access interface from many, changing apps

Security and data privacy

Practical Benefits of Studying These Systems

Learn how to select & tune data systems

Learn how to build them

Learn how to build apps that have to tackle some of these same challenges

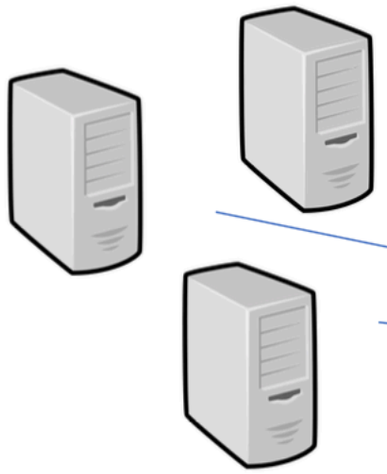
- » E.g. cross-geographic-region billing app, custom search engine, etc

应用场景

System	Logical Data Model	Physical Storage	API	Other Features
Relational databases	Relations (i.e. tables)	B-trees, column stores, indexes, …	SQL, ODBC	Durability, transactions, query planning, migrations, …
TensorFlow	Tensors	NCHW, NHWC, sparse arrays, …	Python DAG construction	query planning, distribution, specialized HW
Apache Kafka	Streams of opaque records	Partitions, compaction	Publish, subscribe	Durability, rescaling
Apache Spark RDDs	Collections of Java objects	Read external systems, cache	Functional API, SQL	Distribution, query planning, transactions*

使用愿景

Clients / users



Queries

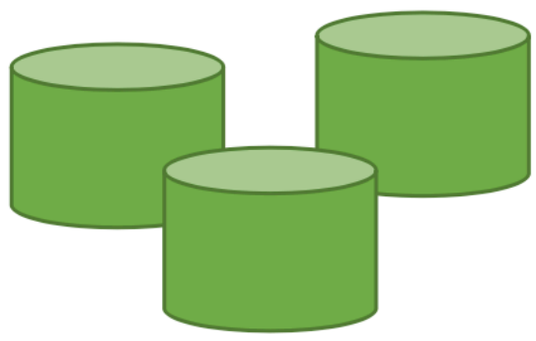
Data mgmt. system

Logical dataset (e.g. table, graph)

First Name	Last Name	Address	City	Age
Mickey	Mouse	123 Fantasy Way	Anaheim	73
Bat	Man	321 Cavern Ave	Gotham	54
Wonder	Woman	987 Truth Way	Paradise	39
Donald	Duck	555 Quack Street	Mallard	65
Bugs	Bunny	567 Carrot Street	Rascal	58
Wiley	Coyote	999 Acme Way	Canyon	61
Cat	Woman	234 Purrfect Street	Hairball	32
Tweety	Bird	543	Itotltaw	28

Administrator

Physical storage (data structures)



Two Big Ideas

Declarative interfaces

» Apps specify *what* they want, not *how* to do it

» Example: “store a table with 2 integer columns”, but not how to encode it on disk

» Example: “count records where column1 = 5”

Transactions

» Encapsulate multiple app actions into one *atomic* request (fails or succeeds as a whole)

» Concurrency models for multiple users

» Clear interactions with failure recovery

Declarative Interface Examples

SQL

- » Abstract “table” data model, many physical implementations
- » Specify queries in a restricted language that the database can optimize

TensorFlow

- » Operator graph gets mapped & optimized to different hardware devices

Functional programming (e.g. MapReduce)

- » Says what to run but not how to do scheduling

Transaction Examples

SQL databases

» Commands to start, abort or end transactions based on multiple SQL statements

Apache Spark, MapReduce

» Make the multi-part output of a job appear atomically when all partitions are done

Stream processing systems

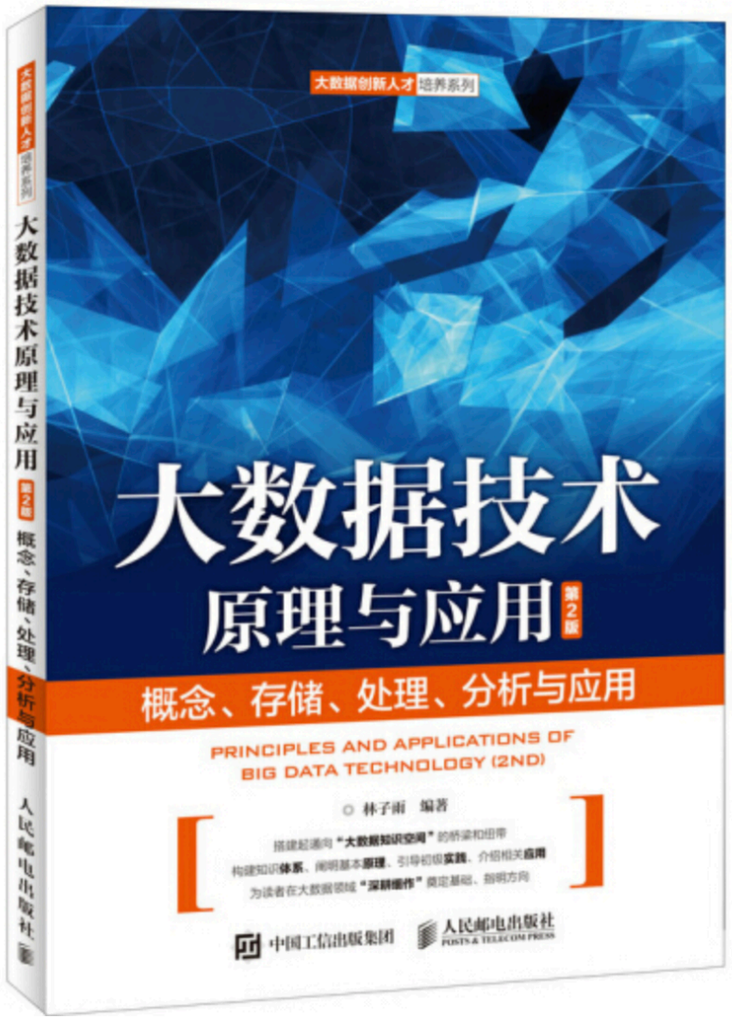
» Count each input record exactly once despite crashes, network failures, etc

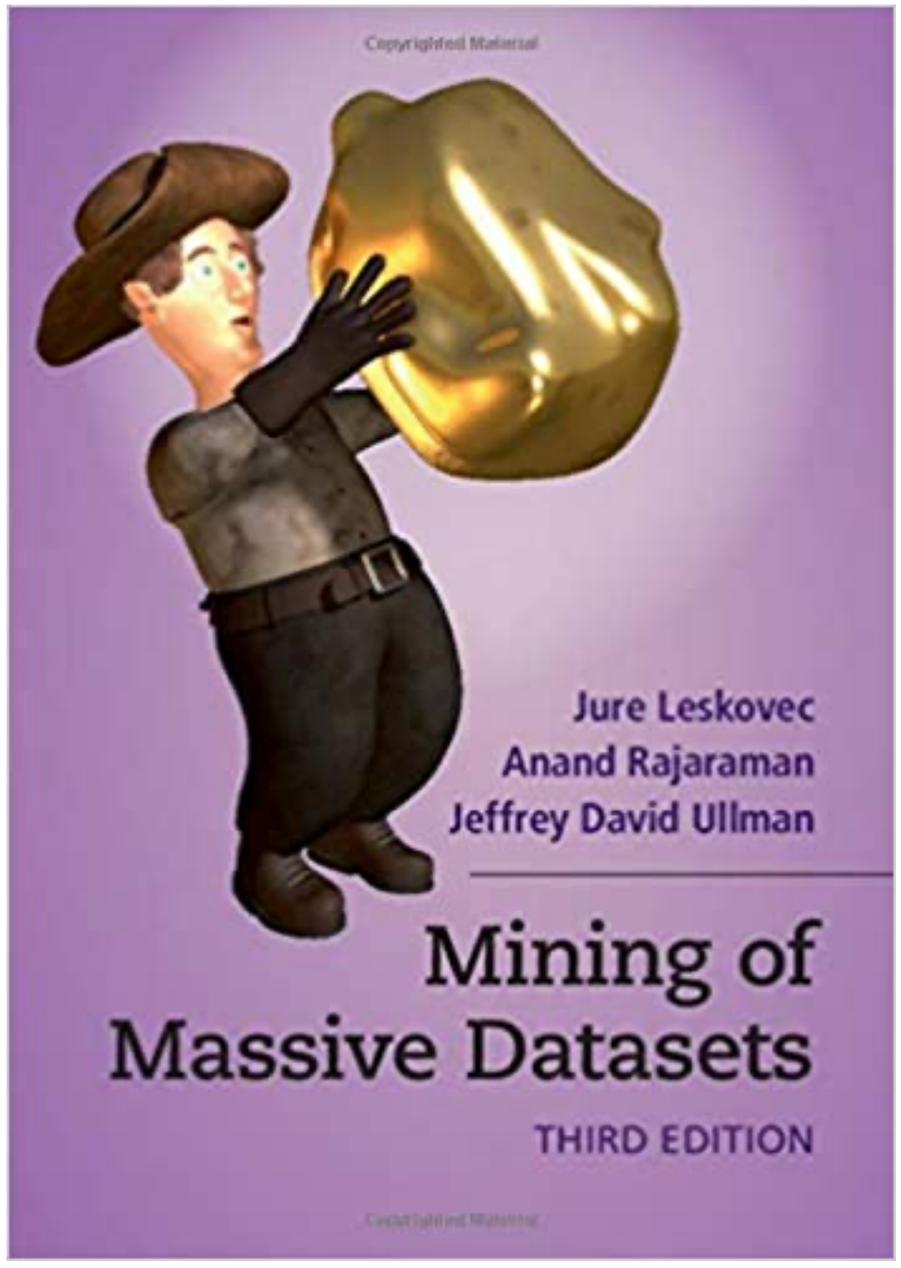
预期目标

- 基本理论
  - 一致性、并发、协调、CAP等
- 相关模型
  - 拜占庭、quorum、chubby等
- 动手能力
  - hbase，tensorflow，flink等

参考教材

- 分布式
- 数据库
- 框架





参考课程

- [MIT] Software Systems for Data Science, http://dsg.csail.mit.edu/6.S080/
- [Berkeley] Principles and Techniques of Data Science, http://www.ds100.org/sp20/
- [厦门大学] http://dblab.xmu.edu.cn/post/bigdata/
- [Stanford] Mining of Massive Dataset. https://web.stanford.edu/class/cs246/

沟通方式

- 微信群
- 邮件
- 大夏学堂

考核方式

- 36学时，2学分
- 40% 书面作业
- 50% 项目作业
- 10% activities

3. 章节组织

课程安排

- 基本的存储管理
  - 概念，模型
  - 框架
-

课程安排

- 面向ML的新框架方式
  - 概念，模型
  - 框架，interface
-

课程安排

- 面向新应用的内容
  - declarative，end-to-end等