ECE30030/ITP30010 – Database Systems

Relational Data

Reading: Chapter 1-2

Charmgil Hong

charmgil@handong.edu

Spring, 2025

Handong Global University

Course Overview

- Course: ECE30030/ITP30010 Database Systems
  - ITP-section#1 (English): Mon/Thu 2:30-3:45pm @OH401
  - ITP-section#2 (English): Mon/Thu 4:00-5:15pm @OH401

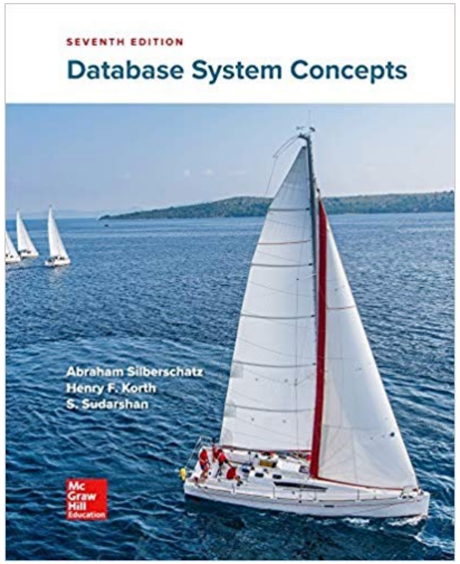
- Instructor: Charmgil Hong (홍참길)
  - Office: NTH201
  - Email: charmgil@handong.edu
  - Office hours: TBD

- Teaching Assistants: TBD

Announcements

- Homework assignment #1 will be released
  - Release: This week (Week #2)
  - Due: Two week from the release

  - You will need the textbook
    - Abraham Silberschatz, Henry F. Korth, S. Sudarshan. ***Database System Concepts, 7th edition.*** McGraw Hill. 2019.



Schedule (tentative)

	Subject		Subject
1	Admin, Introduction DBMS, Relation data model	9	Transactions
2	Relational algebra Installing a DBMS	10	Transactions Database storages
3	Structured Query Language (DML)	11	Database storages
4	Structured Query Language (DDL) Quiz	12	Indexes Quiz (tentative)
5	Entity-Rleationship (ER) diagrams	13	Indexes
6	Normalization theory	14	Keys, Functions/Procedures, Triggers
7	Advanced SQL	15	Beyond relational data
8	Advanced SQL, Constraints, Views Midterm	16	Final

Administrivia

- Grading
  - **Quiz: 5%**
  - **Midterm: 23%**
  - **Final: 25%**
  - **Assignments: 20%**
    - Make sure to submit your work before each deadline
    - Late submissions will be accepted within 24 hours after the deadline with a penalty of -20% of the assignment grade
      - Submissions made after 24 hours from the deadline will be rejected
    - For additional extensions, reasonable excuse should be submitted before the deadline
  - **Term project: 20%**
  - **Participation: 7%**

Administrivia

- Any of the followings will result in **<u>failure (F)</u>**:
    - **Conducting any form of cheating or academic dishonesty**
    - **Not appearing more than 3/4 of all meetings**
        - **Three times of tardiness will be countered as one absence**
    - **Not taking any of the midterm and final exam**

Last Lecture

- Some definitions on DB
- Motivation, Examples
- Brief history

Agenda

- R-DBMS
- Relational Data Model

Database Systems

- Database
  - Organized collection of inter-related data that models some aspect of the real-world (A. Pavlo)
    - Things related are laid together; *c.f.*, files are not like this

- Database system: *Informal definition*

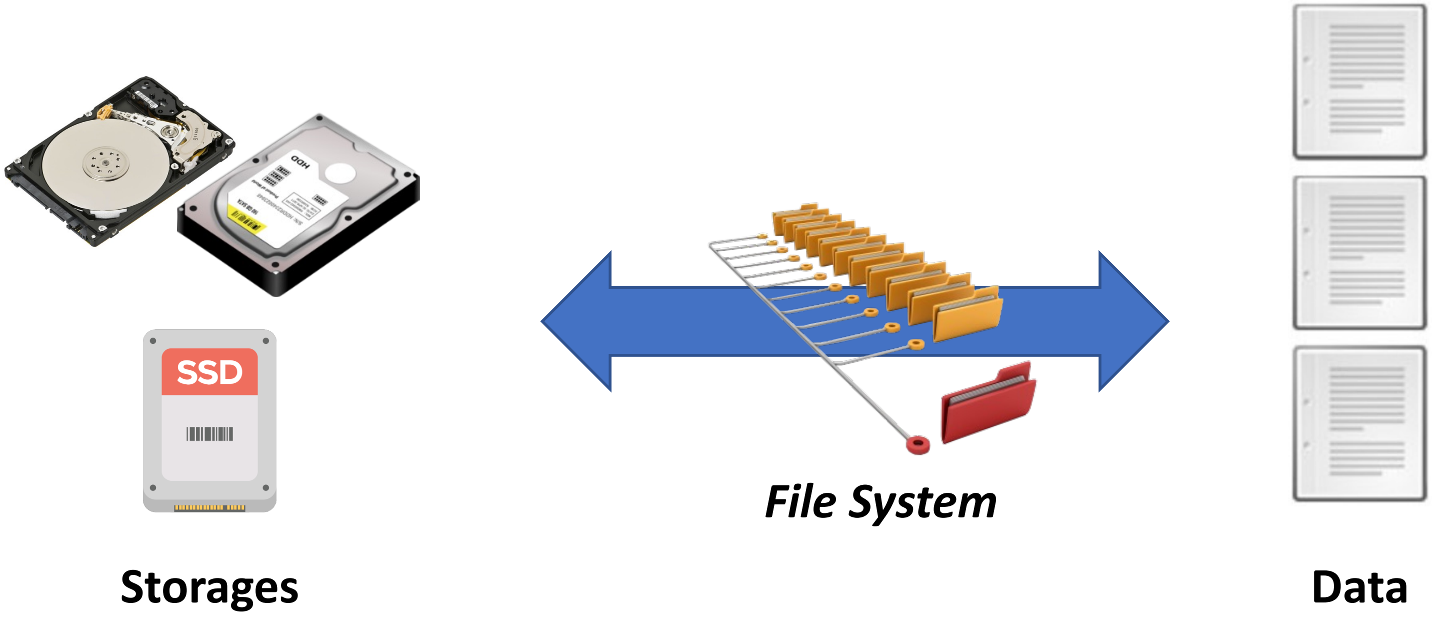
Magnetic tapes (storage)

* Image src: http://www.gettyimages.com/detail/news-photo/computer-display-with-magnetic-tape-unit-news-photo/3328014#30th-january-1970-ibm-computer-display-with-magnetic-tape-unit-picture-id3328014

Database Systems

- Database
  - Organized collection of inter-related data that models some aspect of the real-world (A. Pavlo)
    - Things related are laid together; *c.f.*, files are not like this

- Database system: *Informal definition*

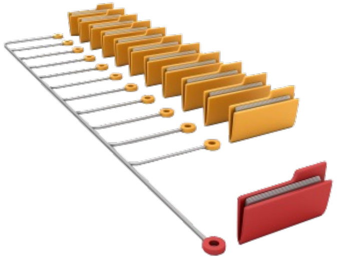


* Image src: https://www.iconfinder.com/icons/4263530/disk_drive_gray_ssd_icon; https://pixabay.com/vectors/hdd-hard-disk-drive-disk-hard-disk-154463/

Database Systems

- Database
  - Organized collection of inter-related data that models some aspect of the real-world (A. Pavlo)
    - Things related are laid together; *c.f.*, files are not like this

- Database system: *Informal definition*



File System

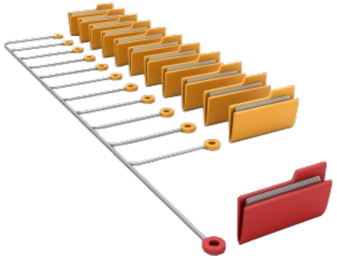
- Flat file strawman
  - Store a database as comma-separated value (CSV) files
  - Manage the CSV files using our own code
    - Use a separate file per entity
    - The applications have to parse the CSV files each time they want to read or update records

* Image src: https://www.iconfinder.com/icons/4263530/disk_drive_gray_ssd_icon; https://pixabay.com/vectors/hdd-hard-disk-drive-disk-hard-disk-154463/

Database Systems

- Database
  - Organized collection of inter-related data that models some aspect of the real-world (A. Pavlo)
    - Things related are laid together; *c.f.*, files are not like this

- Database system: *Informal definition*



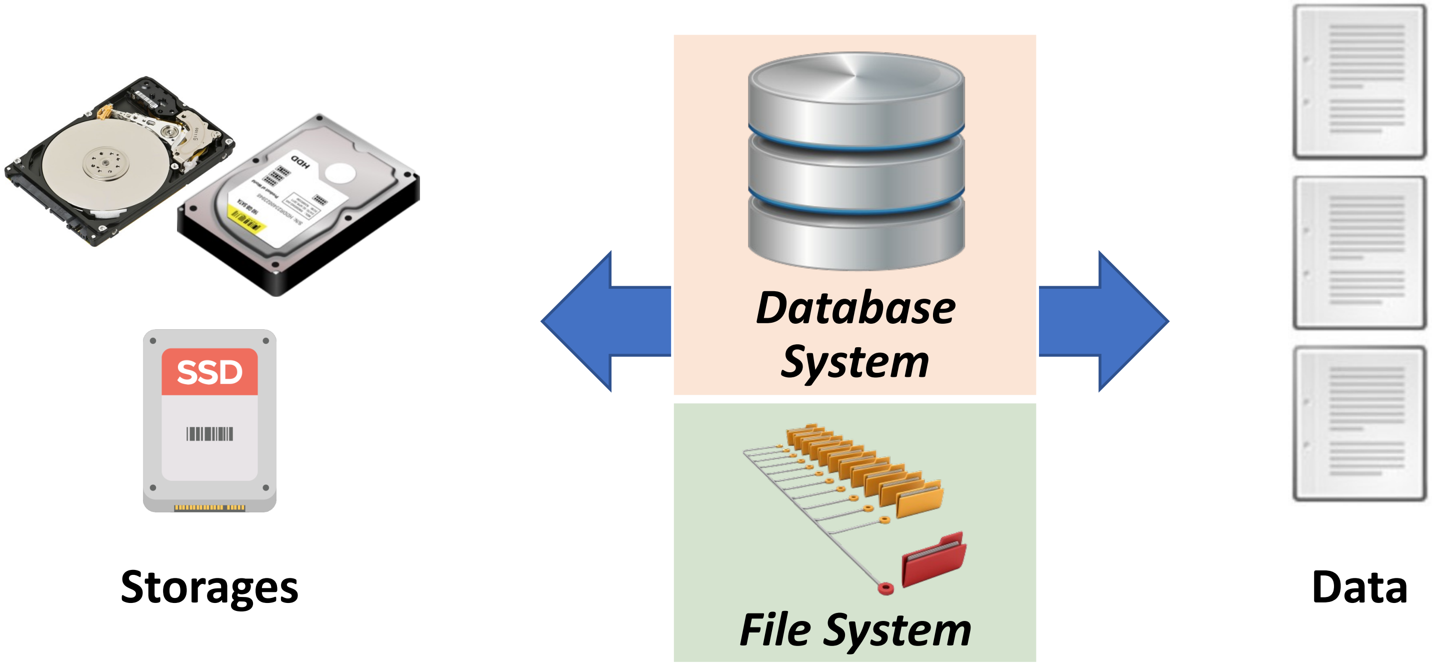
File System

- Flat file strawman
  - Issue: data integrity
    - How to examine the validity of the values?
  - Issue: implementation
    - How to find a particular record?
    - How to write a new application that uses the same data
  - Issue: durability
    - What if the machine crashes while file writing?

* Image src: https://www.iconfinder.com/icons/4263530/disk_drive_gray_ssd_icon; https://pixabay.com/vectors/hdd-hard-disk-drive-disk-hard-disk-154463/

Database Systems

- Database
  - Organized collection of inter-related data that models some aspect of the real-world (A. Pavlo)
    - Things related are laid together; *c.f.*, files are not like this
- Database system: *Informal definition*



Storages

Database System

File System

Data

* Image src: https://www.iconfinder.com/icons/4263530/disk_drive_gray_ssd_icon; https://pixabay.com/vectors/hdd-hard-disk-drive-disk-hard-disk-154463/

Database Systems

- Database management system (DBMS)
  - Software that allows applications to store and analyze information in a database
    - Access data without worrying about the file I/O-level details
  - A general-purpose DBMS is designed to allow the definition, creation, querying, update, and administration of databases

Database Systems

- DBMS as a data storage
  - Database abstraction to avoid low-level implementation and maintenance chores
    - Store database in simple data structures
    - Access data through high-level language
  - Database abstraction does not include:
    - How to implement the storage, relations, …
    - Clear separation between logical vs. physical layers

- DBMS as an interface
  - Data definition language (DDL)
  - Data manipulation language (DML)

  → Structured query language (SQL) includes both DDL and DML

Agenda

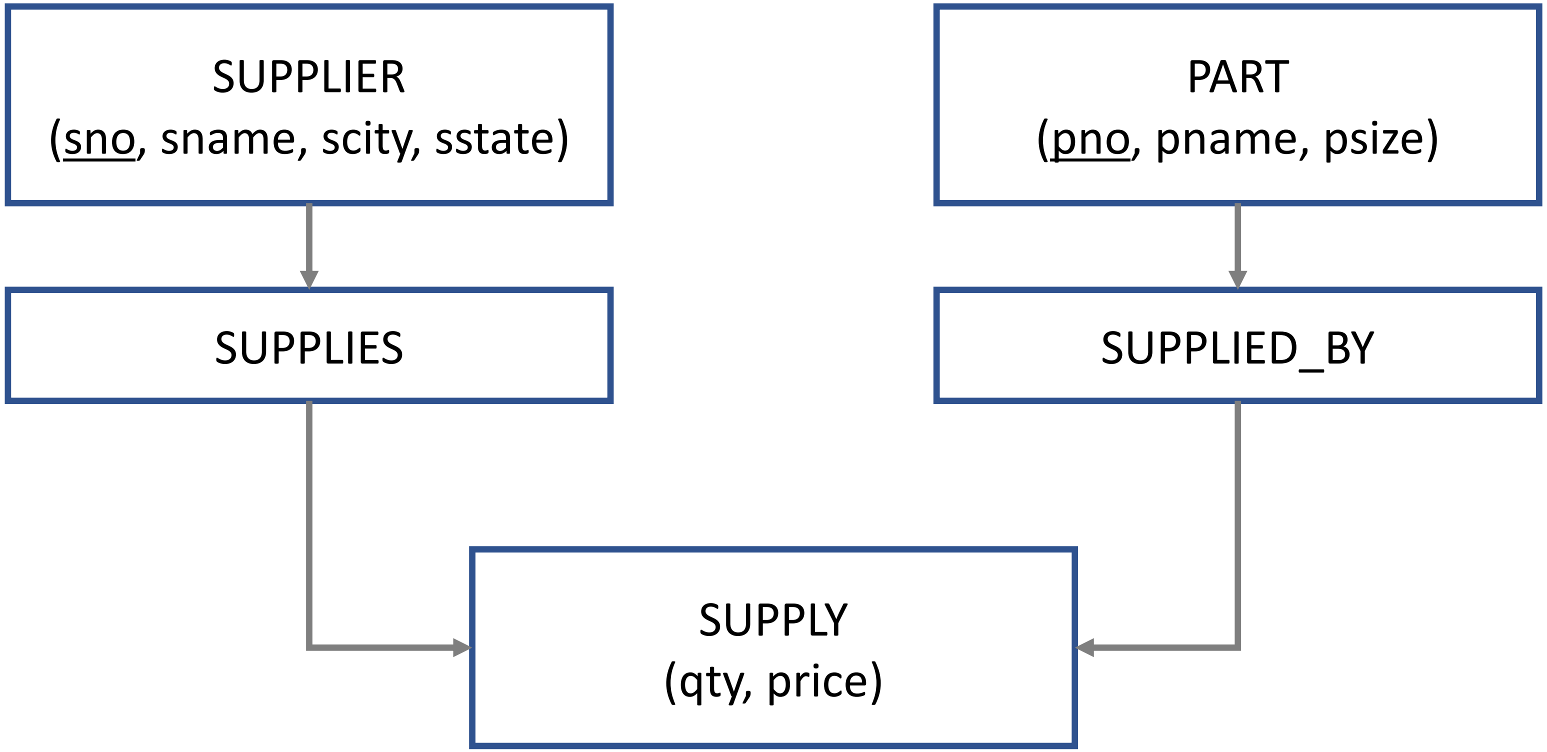
- R-DBMS
- **Relational Data Model**

Data Model

- Data model: A notion for describing data or information
  - Data model consists of three parts:
    - Structure
    - Operations
    - Constraints
  - Examples
    - Relational data model: the most conventional ← *main focus of the course*!

Network Data Model

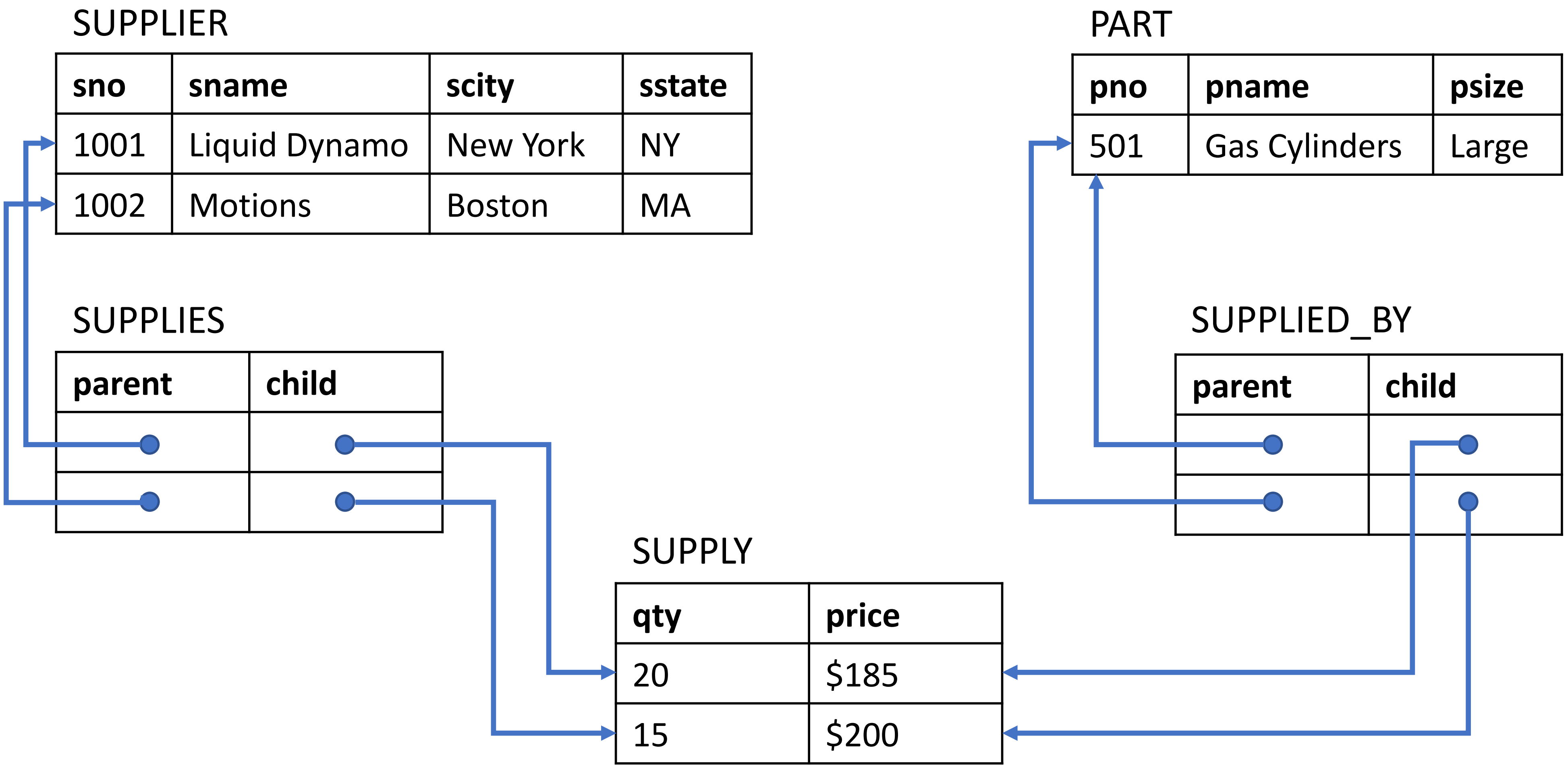
- Example: Schema



Example taken from: A. Pavlo. 15-721 Advanced Database Systems. https://15721.courses.cs.cmu.edu/spring2020/slides/01-history.pdf

Network Data Model

- Example: Instances



SUPPLIER

sno	sname	scity	sstate
1001	Liquid Dynamo	New York	NY
1002	Motions	Boston	MA

PART

pno	pname	psize
501	Gas Cylinders	Large

SUPPLIES

parent	child
●	●
●	●

SUPPLIED_BY

parent	child
●	●
●	●

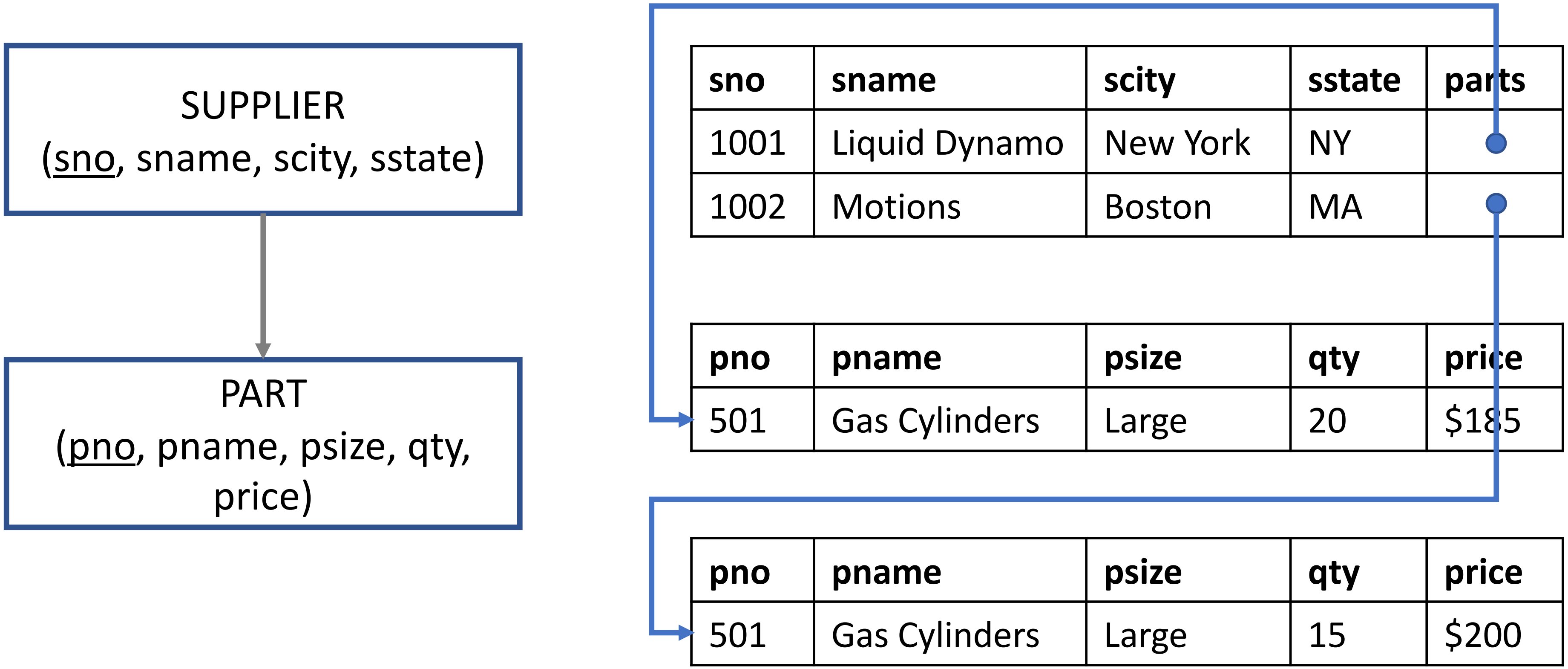
SUPPLY

qty	price
20	$185
15	$200

Example taken from: A. Pavlo. 15-721 Advanced Database Systems. https://15721.courses.cs.cmu.edu/spring2020/slides/01-history.pdf

Hierarchical Data Model

- Example: Schema & Instances



SUPPLIER
(sno, sname, scity, sstate)

PART
(pno, pname, psize, qty, price)

sno	sname	scity	sstate	parts
1001	Liquid Dynamo	New York	NY	●
1002	Motions	Boston	MA	●

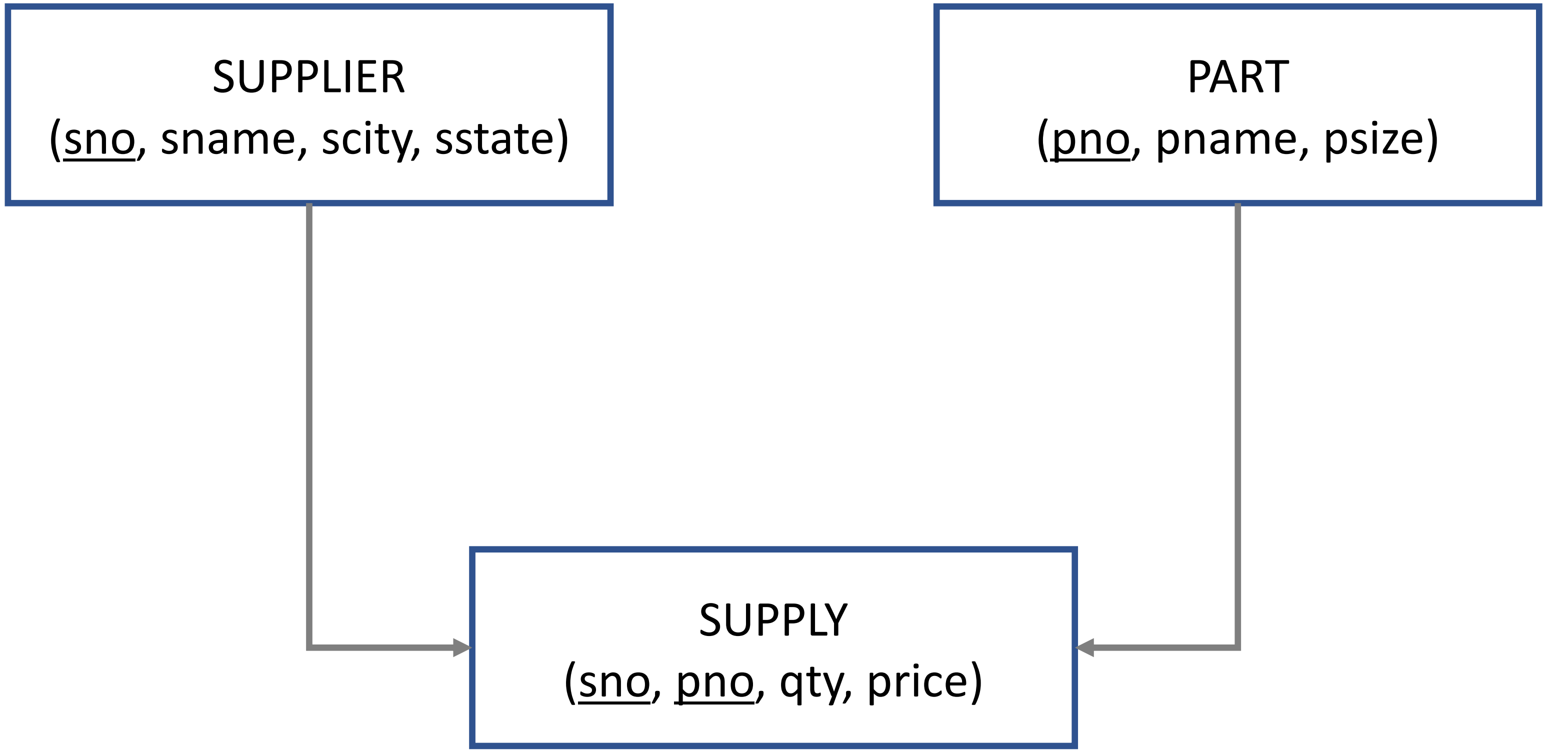
pno	pname	psize	qty	price
501	Gas Cylinders	Large	20	$185

pno	pname	psize	qty	price
501	Gas Cylinders	Large	15	$200

Example taken from: A. Pavlo. 15-721 Advanced Database Systems. https://15721.courses.cs.cmu.edu/spring2020/slides/01-history.pdf

Relational Data Model

- Example: Schema



SUPPLIER
(sno, sname, scity, sstate)

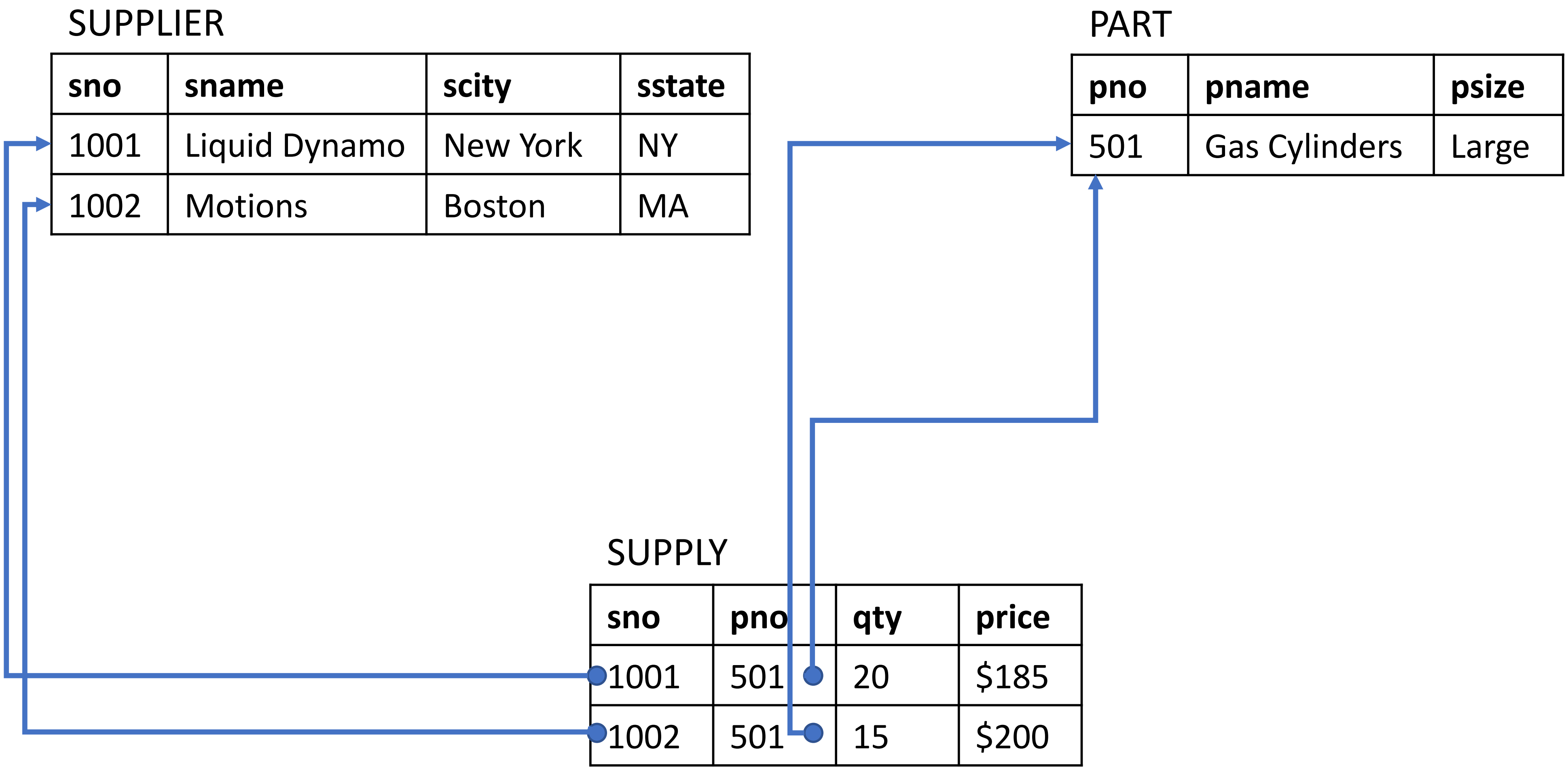
PART
(pno, pname, psize)

SUPPLY
(sno, pno, qty, price)

Example taken from: A. Pavlo. 15-721 Advanced Database Systems. https://15721.courses.cs.cmu.edu/spring2020/slides/01-history.pdf

Relational Data Model

- Example: Instances



SUPPLIER

sno	sname	scity	sstate
1001	Liquid Dynamo	New York	NY
1002	Motions	Boston	MA

PART

pno	pname	psize
501	Gas Cylinders	Large

SUPPLY

sno	pno	qty	price
1001	501	20	$185
1002	501	15	$200

Example taken from: A. Pavlo. 15-721 Advanced Database Systems. https://15721.courses.cs.cmu.edu/spring2020/slides/01-history.pdf

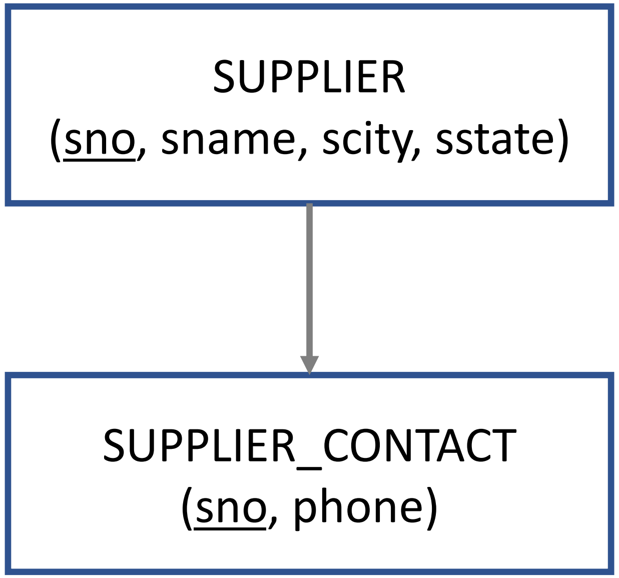
Object-Oriented Data Model

- Example: Application code & Schema

```
Class Supplier {
    int sno;
    String sname;
    String scity;
    String sstate;
    String phone[];
}
```

sno	sname	scity	sstate
1001	Liquid Dynamo	New York	NY

sno	phone
1001	212-111-2222
1001	917-333-4444


SUPPLIER
(sno, sname, scity, sstate)

SUPPLIER_CONTACT
(sno, phone)

Example taken from: A. Pavlo. 15-721 Advanced Database Systems. https://15721.courses.cs.cmu.edu/spring2020/slides/01-history.pdf

Object-Oriented Data Model

- Example: Application code & Object

```
Class Supplier {
    int sno;
    String sname;
    String scity;
    String sstate;
    String phone[];
}
```

Supplier

```
{
    “sno”: 1001,
    “sname”: “Liquid Dynamo”,
    “scity”: “New York”,
    “sstate”: “NY”,
    “phone”: [
        “212-111-2222”,
        “917-333-4444”
    ]
}
```

Example taken from: A. Pavlo. 15-721 Advanced Database Systems. https://15721.courses.cs.cmu.edu/spring2020/slides/01-history.pdf

Data Model

- Data model: A notion for describing data or information
  - Data model consists of three parts:
    - Structure
    - Operations
    - Constraints
  - Examples
    - Relational data model: the most conventional ← *main focus of the course*!
    - NoSQL
      - Key/value
      - Graph
      - Document
      - Column-family
    - Machine learning
      - Array/matrix
    - Misc.: hierarchical, network

Relational Data Model

- Relational data model: A data model describes data in terms of relations

- Relation
  - An unordered set that contains the relationship of attributes that represent entities

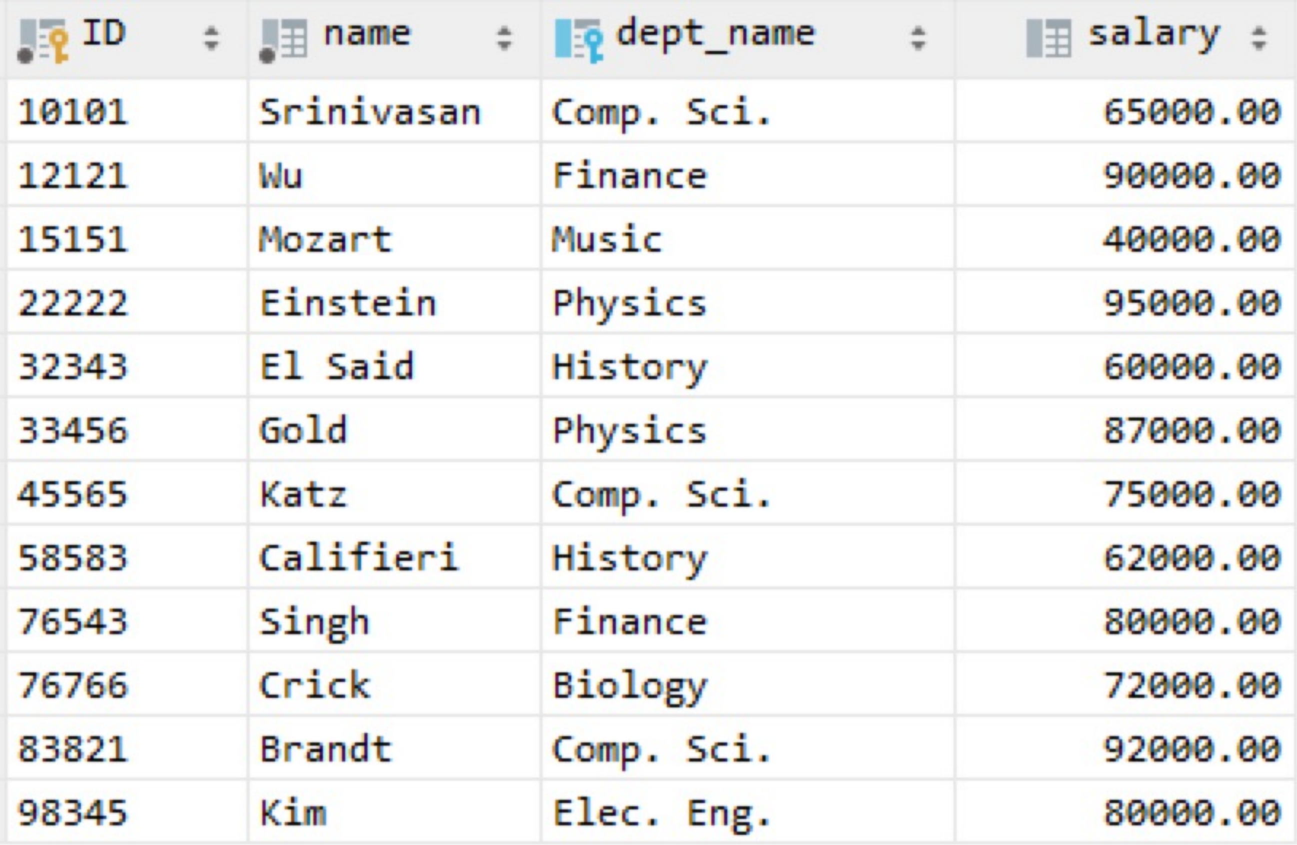
Relation (Table)

- Attribute (column)
  - Attribute values are required to be atomic (indivisible data type)
    - String is an atomic data type in most database systems
  - The set of allowed values for each attribute is called the domain of the attribute
  - NULL is a member of every domain, indicating that the value is “unknown”
    - The NULL values cause complications in many operations

- Tuple (row)
  - A tuple is a set of attribute values (also known as its domain) in the relation
  - Each tuple has one value for each attribute of the relation
  - Values are (normally) atomic/scalar

Example: a *Relation*

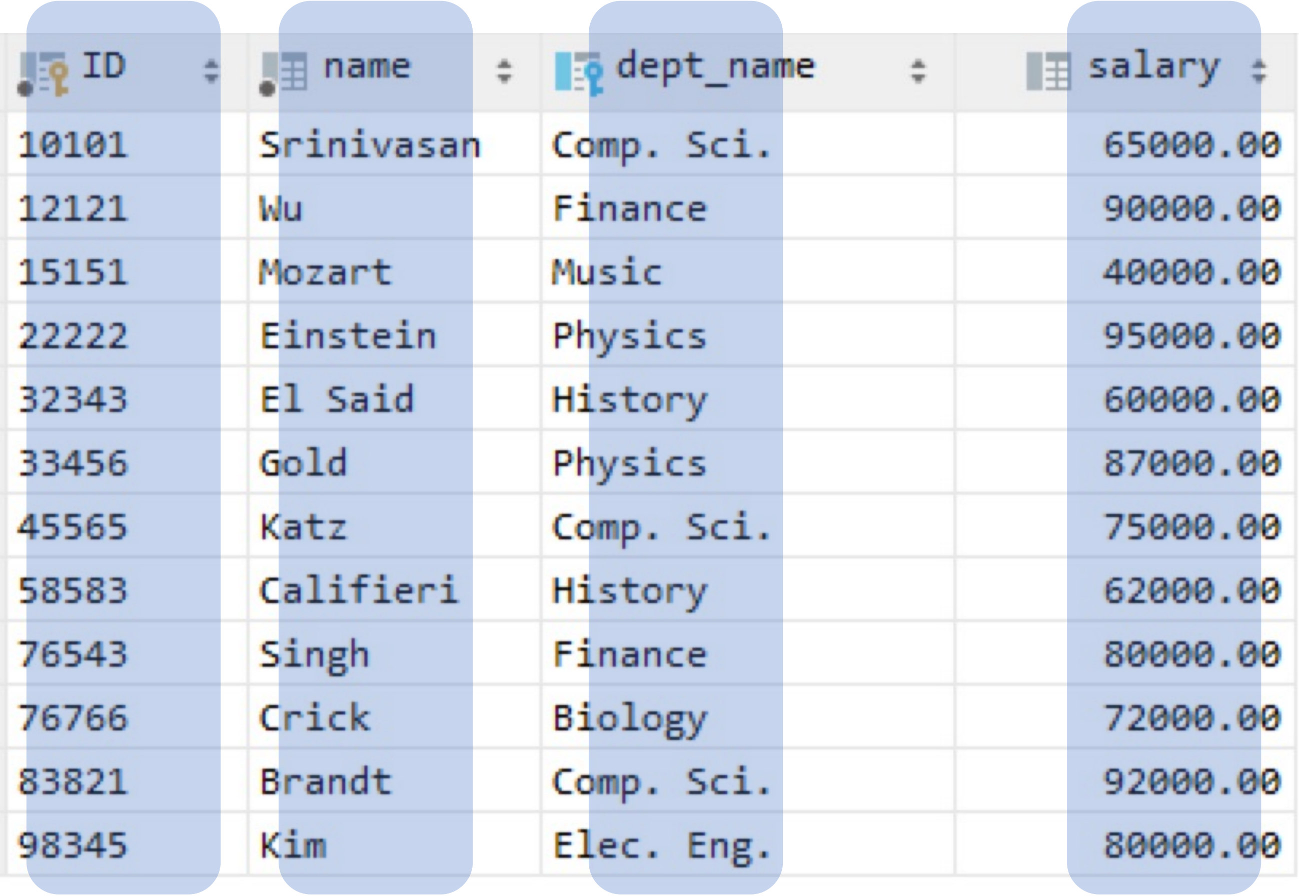
- *n*-ary relation = table with *n* columns



ID	name	dept_name	salary
10101	Srinivasan	Comp. Sci.	65000.00
12121	Wu	Finance	90000.00
15151	Mozart	Music	40000.00
22222	Einstein	Physics	95000.00
32343	El Said	History	60000.00
33456	Gold	Physics	87000.00
45565	Katz	Comp. Sci.	75000.00
58583	Califieri	History	62000.00
76543	Singh	Finance	80000.00
76766	Crick	Biology	72000.00
83821	Brandt	Comp. Sci.	92000.00
98345	Kim	Elec. Eng.	80000.00

Example: a *Relation*

- *n*-ary relation = table with *n* columns

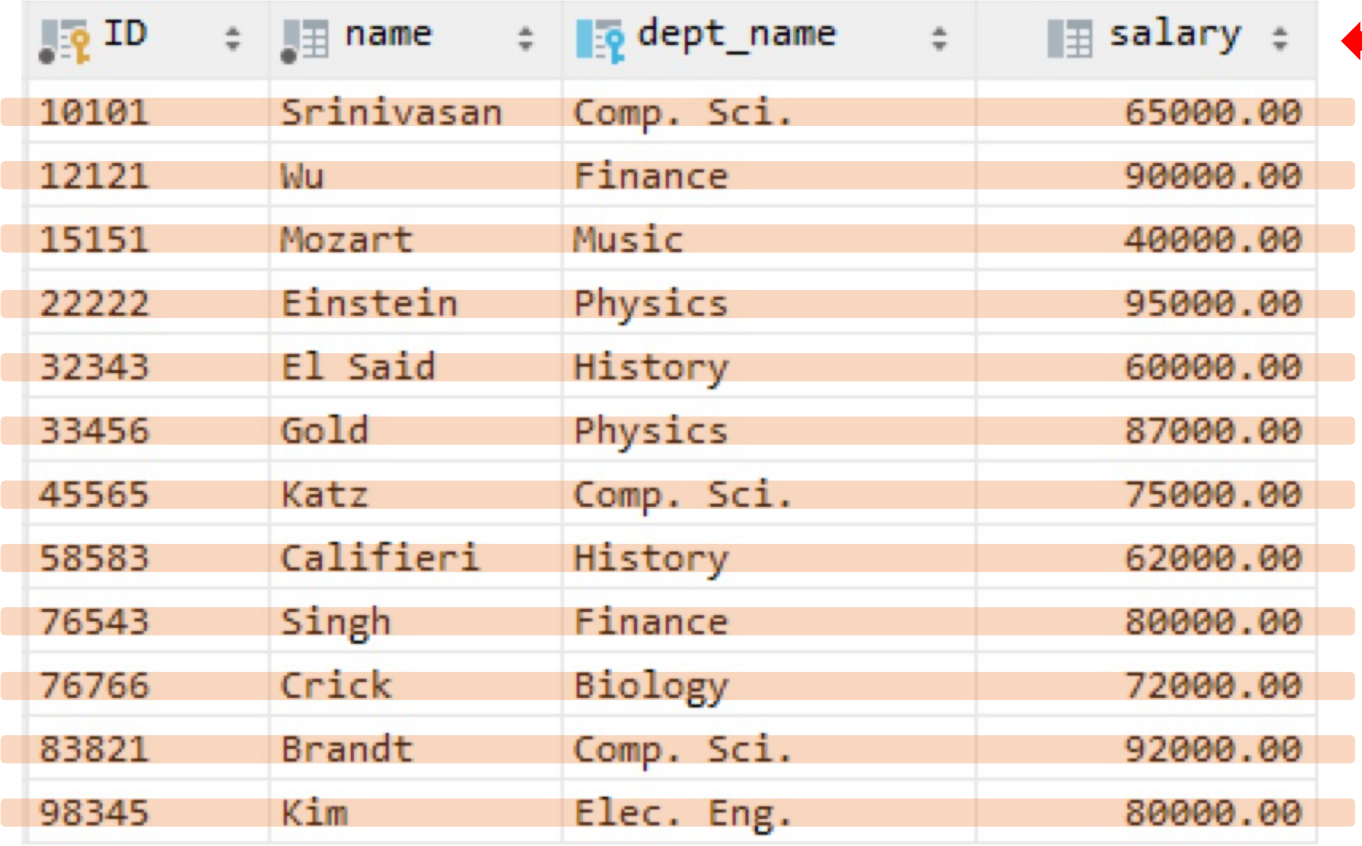


ID	name	dept_name	salary
10101	Srinivasan	Comp. Sci.	65000.00
12121	Wu	Finance	90000.00
15151	Mozart	Music	40000.00
22222	Einstein	Physics	95000.00
32343	El Said	History	60000.00
33456	Gold	Physics	87000.00
45565	Katz	Comp. Sci.	75000.00
58583	Califieri	History	62000.00
76543	Singh	Finance	80000.00
76766	Crick	Biology	72000.00
83821	Brandt	Comp. Sci.	92000.00
98345	Kim	Elec. Eng.	80000.00

4 attributes (columns)

Example: a *Relation*

- *n*-ary relation = table with *n* columns



ID	name	dept_name	salary
10101	Srinivasan	Comp. Sci.	65000.00
12121	Wu	Finance	90000.00
15151	Mozart	Music	40000.00
22222	Einstein	Physics	95000.00
32343	El Said	History	60000.00
33456	Gold	Physics	87000.00
45565	Katz	Comp. Sci.	75000.00
58583	Califieri	History	62000.00
76543	Singh	Finance	80000.00
76766	Crick	Biology	72000.00
83821	Brandt	Comp. Sci.	92000.00
98345	Kim	Elec. Eng.	80000.00

Header

12 tuples (rows, or records)

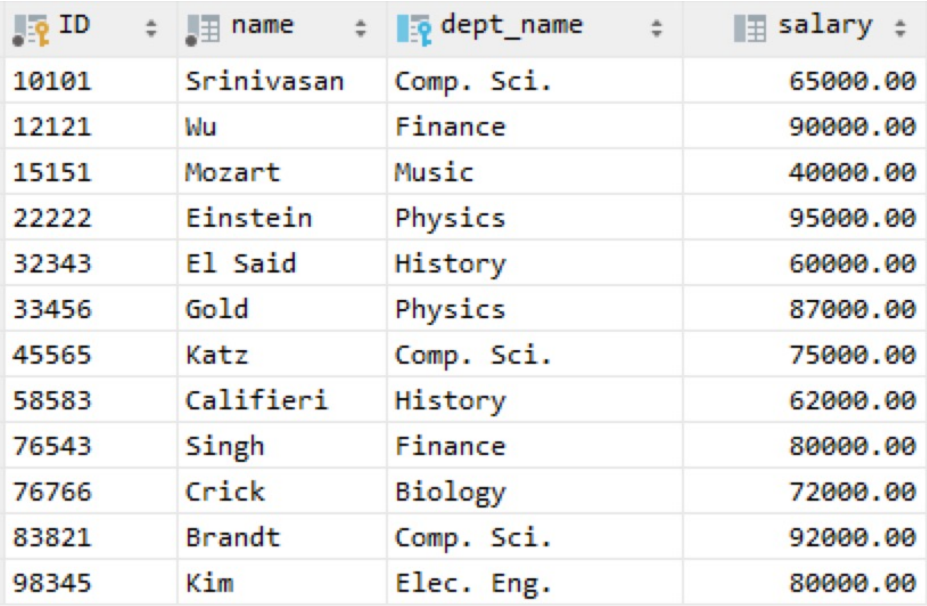
Relation (Table)

- Attribute (column)
  - Attribute values are required to be atomic (indivisible data type)
    - String is an atomic data type in most database systems
  - The set of allowed values for each attribute is called the domain of the attribute
  - NULL is a member of every domain, indicating that the value is “unknown”
    - The NULL values cause complications in many operations

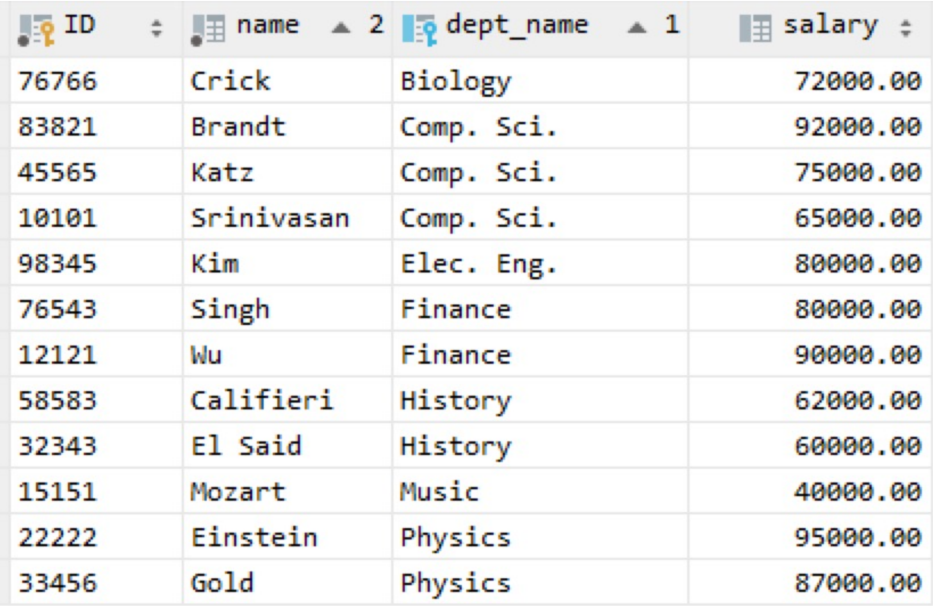
- Tuple (row)
  - A tuple is a set of attribute values (also known as its domain) in the relation
  - Each tuple has one value for each attribute of the relation
  - Values are (normally) atomic/scalar

Relation (Table)

- Relations are unordered: Order of tuples is irrelevant (tuples may be stored in an arbitrary order)
    - Example



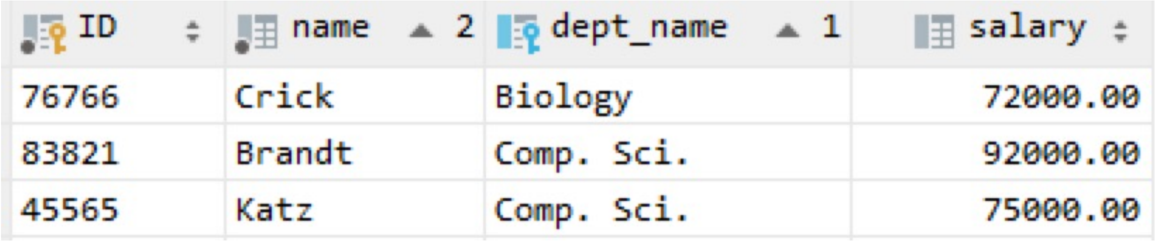
ID	name	dept_name	salary
10101	Srinivasan	Comp. Sci.	65000.00
12121	Wu	Finance	90000.00
15151	Mozart	Music	40000.00
22222	Einstein	Physics	95000.00
32343	El Said	History	60000.00
33456	Gold	Physics	87000.00
45565	Katz	Comp. Sci.	75000.00
58583	Califieri	History	62000.00
76543	Singh	Finance	80000.00
76766	Crick	Biology	72000.00
83821	Brandt	Comp. Sci.	92000.00
98345	Kim	Elec. Eng.	80000.00



ID	name ▲ 2	dept_name ▲ 1	salary
76766	Crick	Biology	72000.00
83821	Brandt	Comp. Sci.	92000.00
45565	Katz	Comp. Sci.	75000.00
10101	Srinivasan	Comp. Sci.	65000.00
98345	Kim	Elec. Eng.	80000.00
76543	Singh	Finance	80000.00
12121	Wu	Finance	90000.00
58583	Califieri	History	62000.00
32343	El Said	History	60000.00
15151	Mozart	Music	40000.00
22222	Einstein	Physics	95000.00
33456	Gold	Physics	87000.00

Notations

- Using a table



ID	name ▲ 2	dept_name ▲ 1	salary
76766	Crick	Biology	72000.00
83821	Brandt	Comp. Sci.	92000.00
45565	Katz	Comp. Sci.	75000.00

- Using a set notation

  Structure: *instructor*(*ID, name, dept_name, salary*),

  Tuples: (*76766, Crick, Biology, 72000.00*),
  (*83821, Brandt, Comp. Sci., 92000.00*),
  (*45565, Katz, Comp. Sci., 75000.00*)

  - Mathematically, sets do not have orders nor duplicates
  - However, we implicitly treat each tuple as an ordered set
    - (*76766, Crick, Biology, 72000.00*) != (*72000.00, Biology, Crick, 76766*)

Keys

- Key
  - One type of constraints
  - One or more attributes form a key
  - A key for a relation → do NOT allow duplicates of the same values of the key attributes

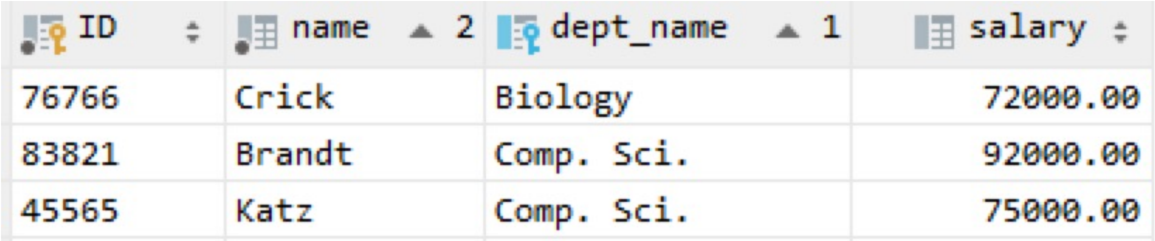
Keys

- Definitions
  - Let $K \subseteq R$
  - K is a superkey of R if values for K are sufficient to identify a unique tuple of each possible relation $r(R)$
    - Example:
  - Superkey K is a candidate key if K is minimal
    - Example:
  - One of the candidate keys is selected to be the primary key
    - Which one to choose?

Primary Keys

- A relation's primary key uniquely identifies a single tuple
- Some DBMSs automatically create an internal primary key if you do not define one
  - *E.g.*, SQL:2003 (SEQUENCE), MySQL (AUTO_INCREMENT)

- Example
  - instructor(ID, name, dept_name, salary)



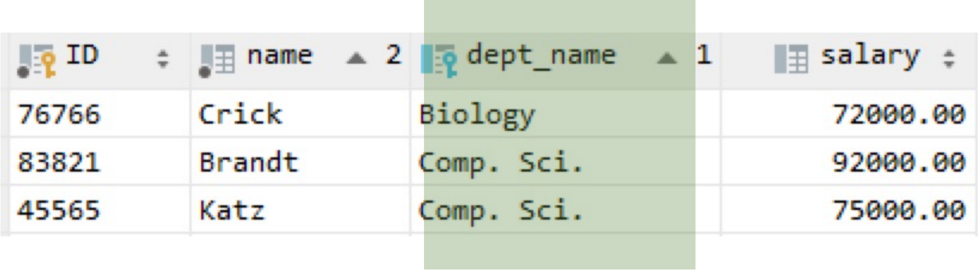
ID	name ▲ 2	dept_name ▲ 1	salary
76766	Crick	Biology	72000.00
83821	Brandt	Comp. Sci.	92000.00
45565	Katz	Comp. Sci.	75000.00

Foreign Keys

- A foreign key specifies that an attribute from one relation has to map to a tuple in another relation
  - Value in one relation must appear in another relation
    - Referencing relation → Referenced relation

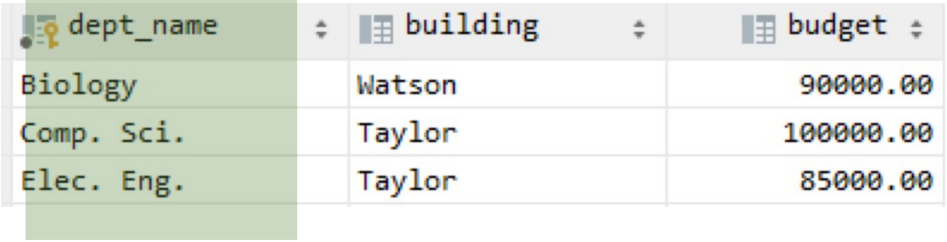
- Example

Relation: instructor



ID	name	dept_name	salary
76766	Crick	Biology	72000.00
83821	Brandt	Comp. Sci.	92000.00
45565	Katz	Comp. Sci.	75000.00

Relation: department



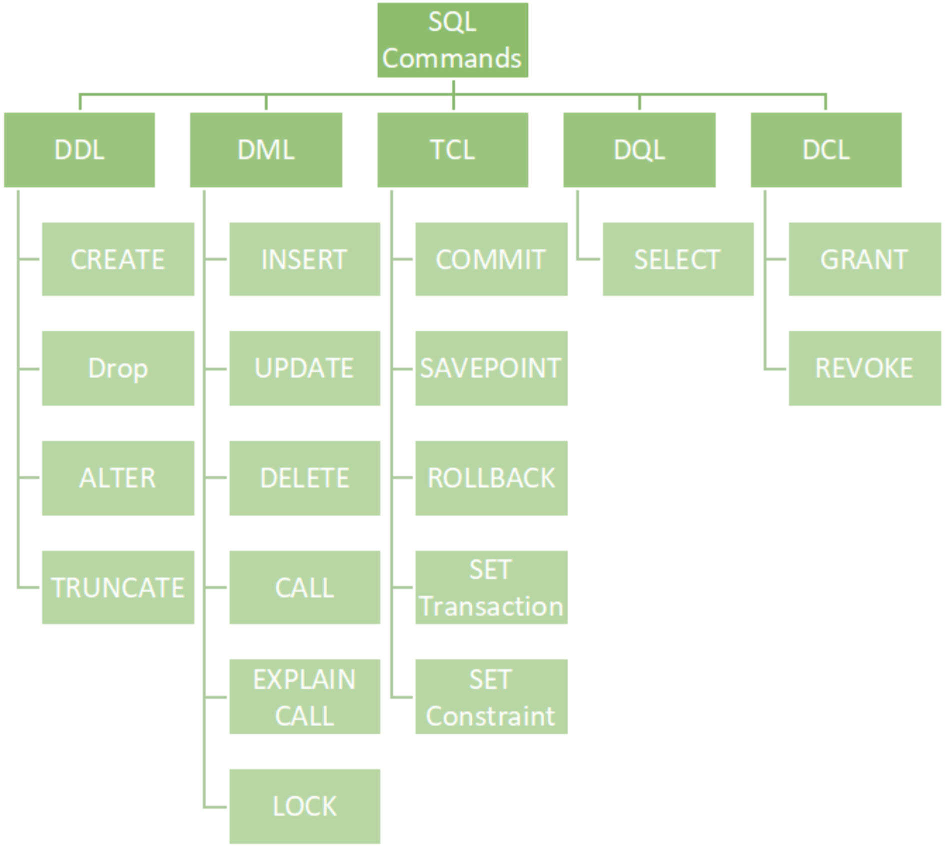
dept_name	building	budget
Biology	Watson	90000.00
Comp. Sci.	Taylor	100000.00
Elec. Eng.	Taylor	85000.00

Data Language

- Data definition language (DDL)
  - How to represent relations and information in a database
    - Defines database schemas
- Data manipulation language (DML)
  - How to store and retrieve information from a database
  - Procedural
    - The query specifies the (high-level) strategy the DBMS should use to find the desired results
    - Based on relational algebra
  - *C.f.*, there are non-procedural DML
    - The query specifies only what data is wanted and not how to find it
    - Based on relational calculus – *this is related to query optimization*

Data Language

- A bit more specific …
  - DDL
  - DML
  - TCL: Transaction Control Lang.
  - DQL: Data Query Lang.
  - DCL: Data Control Lang.



Database Schema

- Database: a collection of relations (tables)
- Database schema: the logical structure of the database
- Database instance: a snapshot of the data in the database at a given instant in time
    - Relation instance: a snapshot of a relation (attributes and tuples) at a given instant in time

ECE30030/ITP30010 Database Systems

Relational Algebra

Reading: Chapter 2

Charmgil Hong

charmgil@handong.edu

Spring, 2024

Handong Global University

Agenda

- Relational algebra
    - Select
    - Project
    - Cartesian product
    - Join
    - Rename
    - Union
    - Set-intersection
    - Set-difference

Algebra

- Mathematical system consisting of
  - Operands: variables or values from which new values can be constructed
  - Operators: symbols denoting procedures that construct new values from given operands

Relational Algebra

- A procedural language consisting of a set of operations that take one or two relations as input and
produce a new relation as their output

- Basic operators
  - Select: σ
  - Project: ∏
  - Cartesian product: ×
  - Join: ⋈
  - Rename: ρ

  - Union: ∪
  - Set-intersection: ∩
  - Set-difference: –