Business Analytics and Data Driven Decision Making

Introduction to Operational and Analytical Databases

Relational Database Management Systsem

Abid Hussain,
Associate Professor
Center for Business Data Analytics
Department of Digitalization, Copenhagen Business School, Copenhagen, Denmark.

Contact: ah.digi@cbs.dk

CBS COPENHAGEN BUSINESS SCHOOL HANDELSHØJSKOLEN

EFMD EQUIS ACCREDITED · AACSB ACCREDITED · ASSOCIATION OF MBAS ACCREDITED · CEMS · PIM

AGENDA

Agenda

- Introduction to Relational Databases (Operational and Analytical)
- Application Architectures for Databases
- Creating Relational Databases
    - Fundaments of Relational Model
- Relational Database Design
    - Conceptual Design
    - ER Model Basic Annotations
- Implementing Relational Schema using DDL

Learning Objective

- Logicaly design a simple database
- Be able to understand structure of a relational database using ER Model
- Normalization Process and Syntax are Show and Tell Examples for those who are interested in complete DB Design
- **I don’t expect that you will remember syntax !**

Literature

- This lecture will use examples and illustrations for the following books.

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

Jukic, N. (2019). Database systems: Introduction to databases and data warehouses.

DATABASE MANAGEMENT SYSTEM

Database

“Database is a structured collection of related data stored on a computer medium”

Jukic, N. (2019)

The purpose of a database is to make an easy, reliable and continuous access to data.

Jukic, N. (2019). Database systems: Introduction to databases and data warehouses.

Database Applications

- Banking: transactions
- Airlines: reservations, schedules
- Universities: registration, grades
- Sales: customers, products, purchases
- Online retailers: order tracking, customized recommendations
- Manufacturing: production, inventory, orders, supply chain
- Human resources: employee records, salaries, tax deductions
- Social Media Channels
- National registries

Database Management Systems (DBMS)

- A software to facilitate **creation**, **use** and **maintainability** of a database.
- DBMS contains information about a particular enterprise
  - Collection of interrelated data

    For example all the data relating an organization. Product, Purchases, Suppliers etc.
  - Set of programs to access the data

    MSSQL, Oracle, Pl/SQL, Data pipelines
  - An environment that is both convenient and efficient to use

    Cloud, Inhouse, A mixed setup
  - Databases can be very large.
    - Facebook Database, Twitter Database, National Population Registers

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

A Simple Application Example

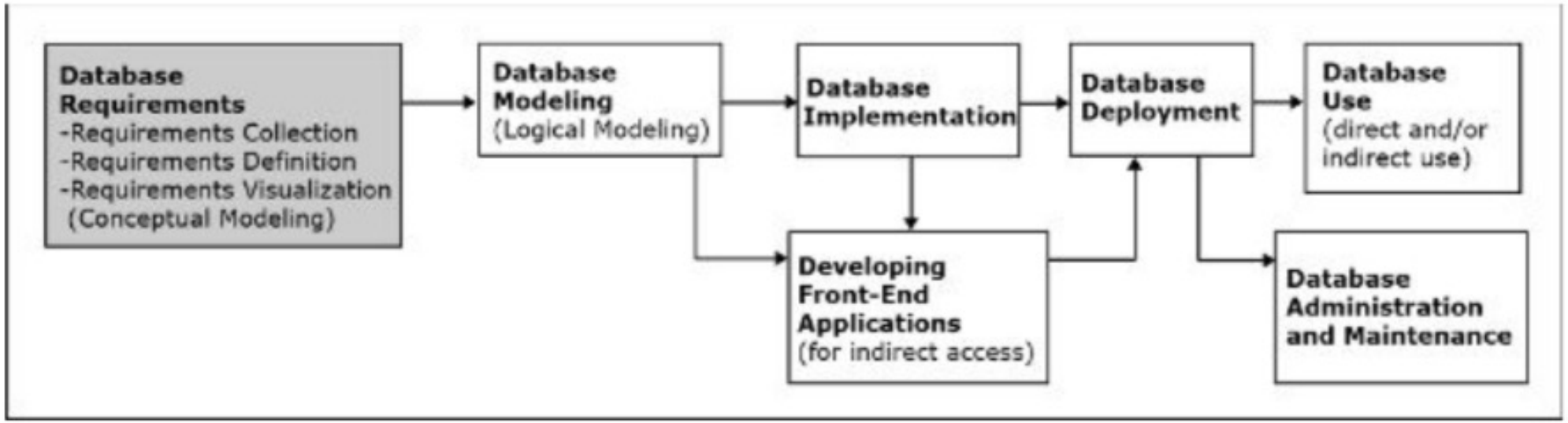
- Let us assume a University System
- What data university need to register
  - Add update delete student
  - Add update delete instructor
  - Add update delete courses
  - Assign courses to students
  - Assign instructor to courses
  - Assign instructor to students
  - Register grades
  - Calculate GPA
  - …

Types of Relational Databases for Data Science Perspective

- Operational Databases
  - Transactional databases for storing business operational data. Sales transactions, Purchase transactions, Social media interactions etc.
- Analytical Databases
  - The information stored to perform analytical tasks. For example, aggregated data, logs, meta data about procedures, transformed data.

Jukic, N. (2019). Database systems: Introduction to databases and data warehouses.

Steps for Database creation



Jukic, N. (2019). Database systems: Introduction to databases and data warehouses.

RELATIONAL DATABASE MANAGEMENT SYSTEM(RDBMS)

Relational Database Mgmt. Sys.

- Most of the world's data is stored in Relational Database Management Systems.
- According to TechRepublic News of 2021 approximately **73%** of the total data.
- The key transaction support properties ACID.
  - Atomicity
    - Entire transaction takes place or none
  - Consistency
    - Integrity constraints maintained before and after transaction
  - Isolation
    - Multiple transaction can take place simultaneously
  - Durability
    - Data is permanently persisted into disk after transaction

https://www.techrepublic.com/article/non-relationals-quiet-revolution-in-databases/

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

Relational Database – Basic Example

- Assume a retail store selling following products.
  - Product A (1000 DKK)
  - Product B (2000 DKK)
  - Product C (3000 DKK)
- We want to store customer purchase history
- Customer data to be stored is **Name, Dob, Street, Zip, City, Country, Email, Mobile**
- Sale data to be stored is **When (Timestamp), How much(Price) , What(Product), Who(Customer)**

Name	Email	Street	Zip	City	Country	Dob	Product	Price	Purchased On
James Gordon	jg@gothpol.com	Police Line	10001	Gotham	USA	12/01/1977	A	1000	12/08/2023 10.25
James Gordon	jg@gothpol.com	Police Line	10001	Gotham	USA	12/01/1977	A	2000	12/08/2023 13.10
James Gordon	jg@gothpol.com	Police Line	10001	Gotham	USA	12/01/1977	B	1000	14/08/2023 12.43
Edward Nygma	riddler@riddle.gt	Magic lane 23	10021	Gotham	USA	19/03/1980	C	3000	19/08/2023 17.25
Edward Nygma	riddler@riddle.gt	Magic lane 23	10021	Gotham	USA	19/03/1980	A	1000	21/08/2023 11.21
Oswald Cobblepot	penguin@gothm.gt	Cobblepot manor	10009	Gotham	USA	31/01/1981	C	3000	02/12/2022 14.19
Oswald Cobblepot	penguin@gothm.gt	Cobblepot manor	10009	Gotham	USA	31/01/1981	A	1000	30/08/2023 09.51
James Gordon	jg@gothpol.com	Police Line	10001	Gotham	USA	12/01/1977	A	1000	24/08/2023 16.04
Bruce Wayne	batman@waynee.goth	1007 Mountain Drive	10021	Gotham	USA	01/07/1983	C	3000	30/08/2023 09.04

Relational Database – Principles

- Concept 1: Uniquely identify each record (**Primary Key**)
  - How will you delete one record if you can't identify each record
  - How will you update ?
- Normalization: Validate Tables and Divide
  - To avoid duplication
  - Performance

Name	Email	Street	Zip	City	Country	Dob	Product	Price	Purchased On
James Gordon	jg@gothpol.com	Police Line	10001	Gotham	USA	12/01/1977	A	1000	12/08/2023 10.25
James Gordon	jg@gothpol.com	Police Line	10001	Gotham	USA	12/01/1977	A	2000	12/08/2023 13.10
James Gordon	jg@gothpol.com	Police Line	10001	Gotham	USA	12/01/1977	B	1000	14/08/2023 12.43
Edward Nygma	riddler@riddle.gt	Magic lane 23	10021	Gotham	USA	19/03/1980	C	3000	19/08/2023 17.25
Edward Nygma	riddler@riddle.gt	Magic lane 23	10021	Gotham	USA	19/03/1980	A	1000	21/08/2023 11.21
Oswald Cobblepot	penguin@gothm.gt	Cobblepot manor	10009	Gotham	USA	31/01/1981	C	3000	02/12/2022 14.19
Oswald Cobblepot	penguin@gothm.gt	Cobblepot manor	10009	Gotham	USA	31/01/1981	A	1000	30/08/2023 09.51
James Gordon	jg@gothpol.com	Police Line	10001	Gotham	USA	12/01/1977	A	1000	24/08/2023 16.04
Bruce Wayne	batman@waynee.goth	1007 Mountain Drive	10021	Gotham	USA	01/07/1983	C	3000	30/08/2023 09.04

Relational Database – Basic Example (Divide Relations according to entities)

Customer						
Name	**Email**	**Street**	**Zip**	**City**	**Country**	**Dob**
James Gordon	jg@gothpol.com	Police Line	10001	Gotham	USA	12/01/1977
Edward Nygma	riddler@riddle.gt	Magic lane 23	10021	Gotham	USA	19/03/1980
Oswald Cobblepot	penguin@gothm.gt	Cobblepot manor	10009	Gotham	USA	31/01/1981
Bruce Wayne	batman@waynee.goth	1007 Mountain Drive	10021	Gotham	USA	01/07/1983

Purchase		
Product	Purchased On	Email
A	12/08/2023 10.25	jg@gothpol.com
A	12/08/2023 13.10	jg@gothpol.com
B	14/08/2023 12.43	jg@gothpol.com
C	19/08/2023 17.25	riddler@riddle.gt
A	21/08/2023 11.21	riddler@riddle.gt
C	02/12/2022 14.19	penguin@gothm.gt
A	30/08/2023 09.51	penguin@gothm.gt
A	24/08/2023 16.04	jg@gothpol.com
C	30/08/2023 09.04	batman@waynee.goth

Product	
Product	**Price**
A	1000
B	2000
C	3000

- **Identify Primary Keys Now**
- **Think of relation cardinality (one -> many)**
- **Think of updating a Customer address**
- **Think of updating price of a Product**

Relational Database – Further refine tables

Customer

ID	Name	Email	Street	Zip	City	Country	Dob
1001	James Gordon	jg@gothpol.com	Police Line	10001	Gotham	USA	12/01/1977
1002	Edward Nygma	riddler@riddle.gt	Magic lane 23	10021	Gotham	USA	19/03/1980
1003	Oswald Cobblepot	penguin@gothm.gt	Cobblepot manor	10009	Gotham	USA	31/01/1981
1004	Bruce Wayne	batman@waynee.goth	1007 Mountain Drive	10021	Gotham	USA	01/07/1983

Product

ID	Product	Price
3001	A	1000
3002	B	2000
2003	C	3000

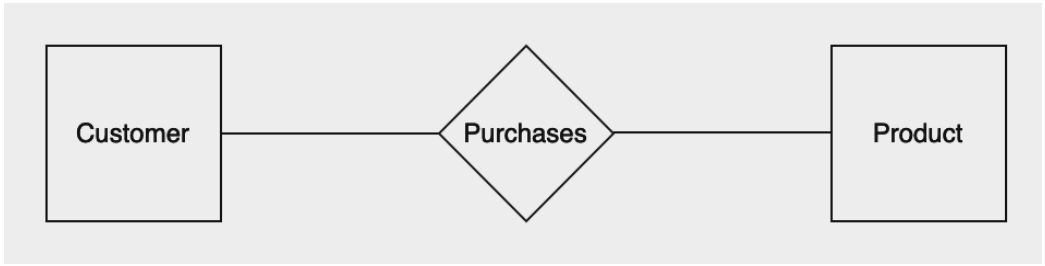
Purchase

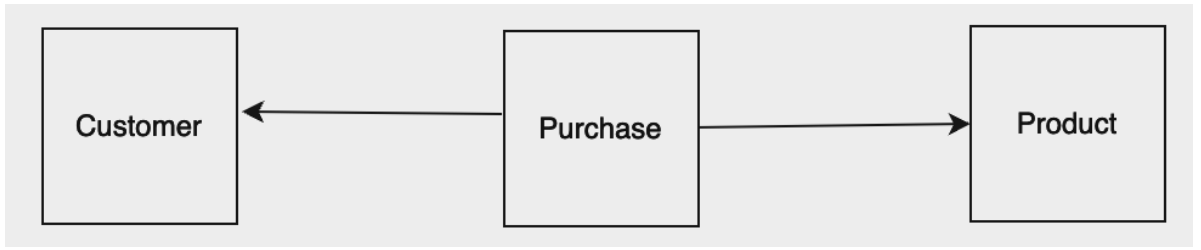
ProductID	Purchased On	CustId
3001	12/08/2023 10.25	1001
3001	12/08/2023 13.10	1001
3002	14/08/2023 12.43	1001
3003	19/08/2023 17.25	1002
3001	21/08/2023 11.21	1002
3003	02/12/2022 14.19	1003
3001	30/08/2023 09.51	1003
3001	24/08/2023 16.04	1001
3003	30/08/2023 09.04	1004

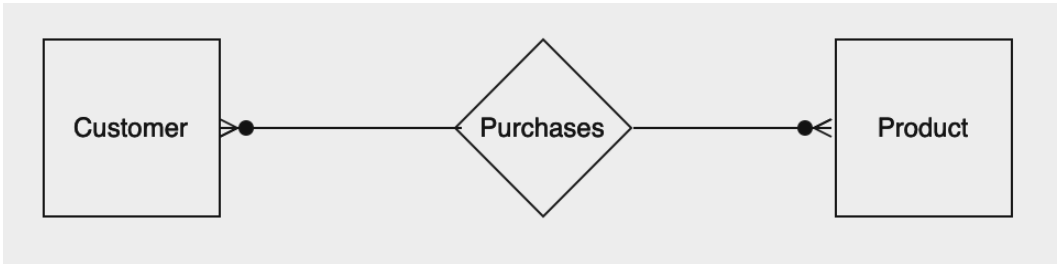
- **Creating IDs as Primary Keys**
- **Numbers are faster to search**

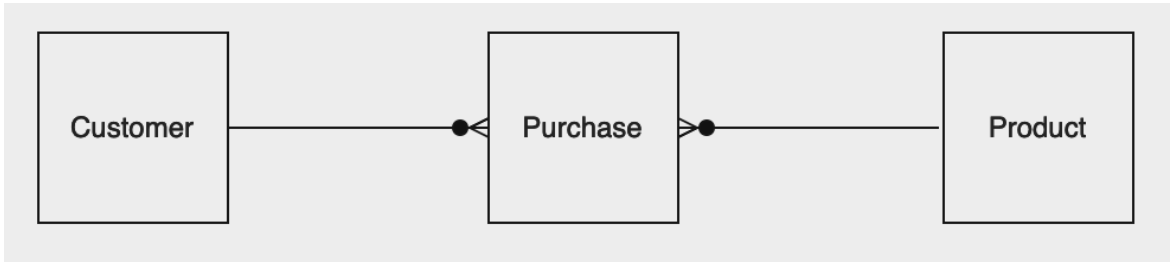
Relational Database – Entity Relation Design

- ER Model
  - **Entities**
  - **Relations**
  - **Type of relation**



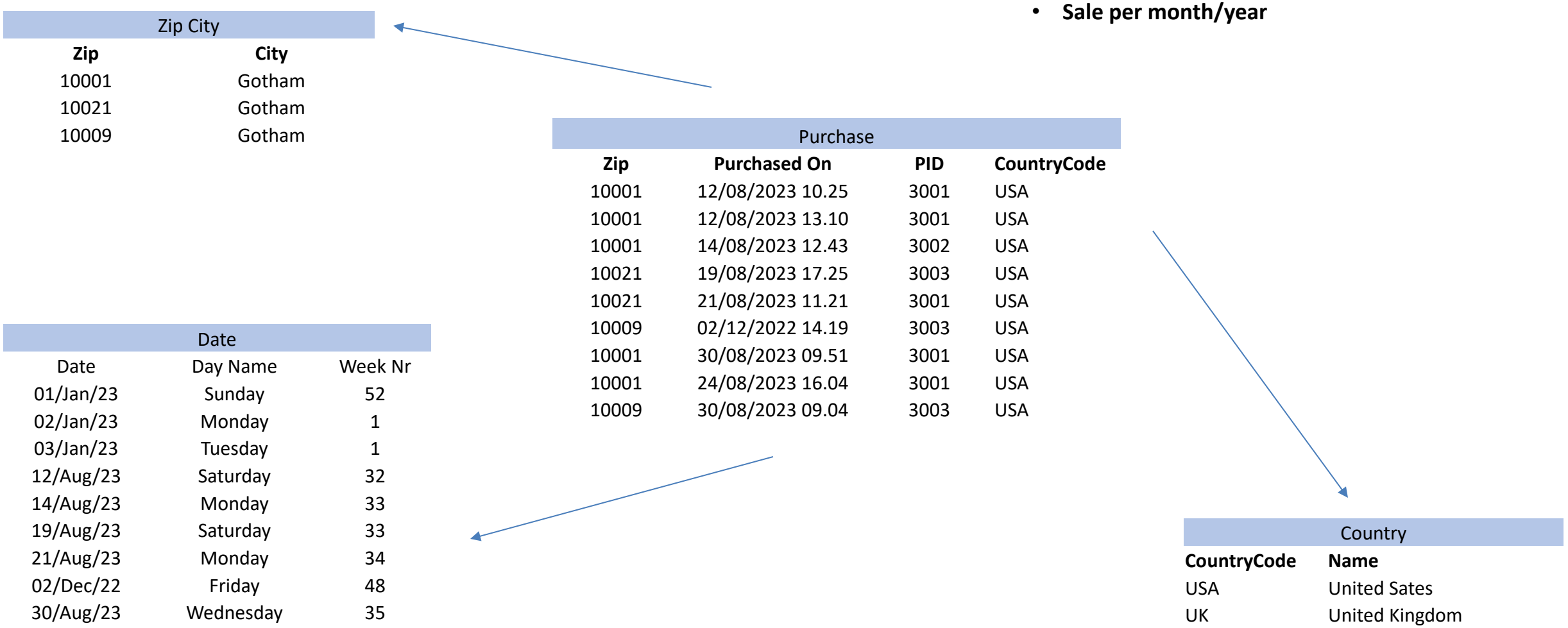






Relational Database – Convert it to Analytical Database

- Imagine following report
  - **Sale per customer country**
  - **Sale per Zip/City**
  - **Sale per month/year**



Zip City

Zip	City
10001	Gotham
10021	Gotham
10009	Gotham

Purchase

Zip	Purchased On	PID	CountryCode
10001	12/08/2023 10.25	3001	USA
10001	12/08/2023 13.10	3001	USA
10001	14/08/2023 12.43	3002	USA
10021	19/08/2023 17.25	3003	USA
10021	21/08/2023 11.21	3001	USA
10009	02/12/2022 14.19	3003	USA
10001	30/08/2023 09.51	3001	USA
10001	24/08/2023 16.04	3001	USA
10009	30/08/2023 09.04	3003	USA

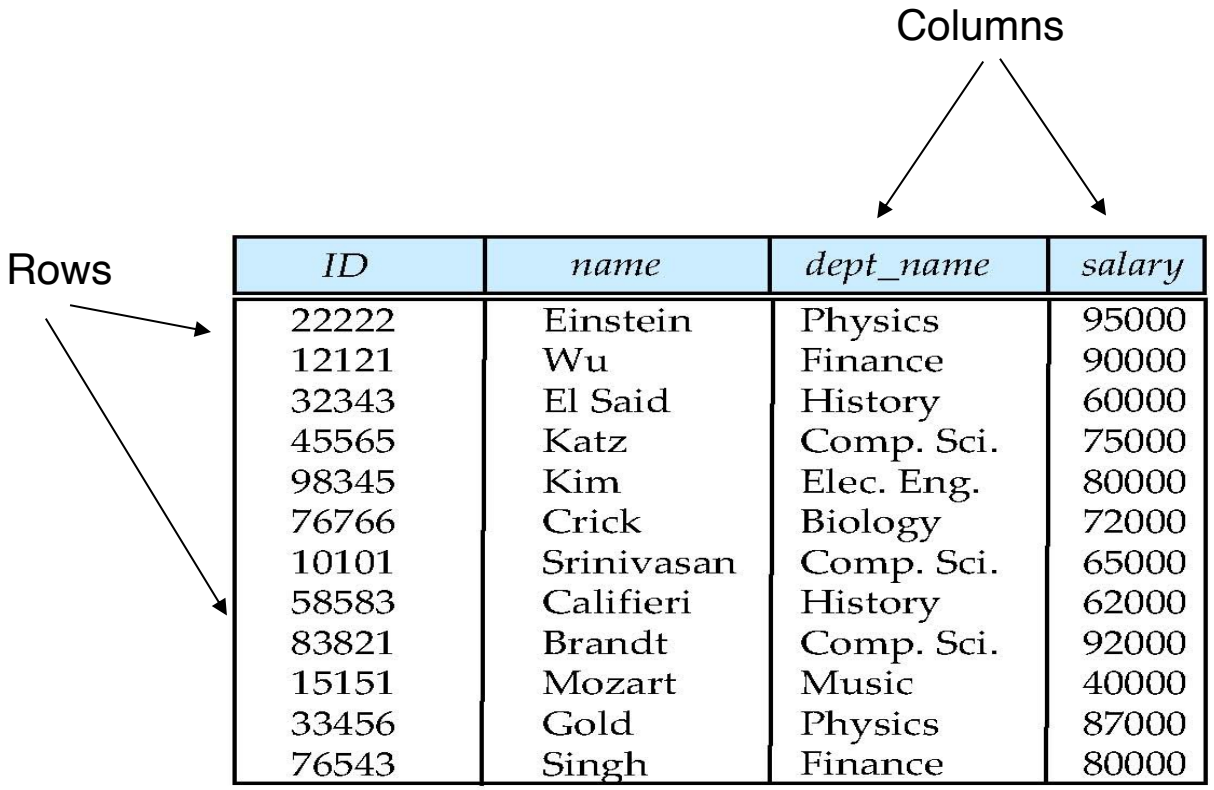
Date

Date	Day Name	Week Nr
01/Jan/23	Sunday	52
02/Jan/23	Monday	1
03/Jan/23	Tuesday	1
12/Aug/23	Saturday	32
14/Aug/23	Monday	33
19/Aug/23	Saturday	33
21/Aug/23	Monday	34
02/Dec/22	Friday	48
30/Aug/23	Wednesday	35
24/Aug/23	Thursday	34

Country

CountryCode	Name
USA	United Sates
UK	United Kingdom

Relational Database



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

(a) The *instructor* table

- A database management system where data is stored in relations which are more commonly referred as Tables
- Data is logically structured in relations.
- Data can be accessed randomly and without need of sorting
- Multiple relations are connected logically to store large set of information

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

Data Abstraction in Relational Databases

- **Physical level:** describes how a record (e.g., instructor) is stored.
- **Logical level:** describes data stored in database, and the relationships among the data.

  type *instructor* = **record**

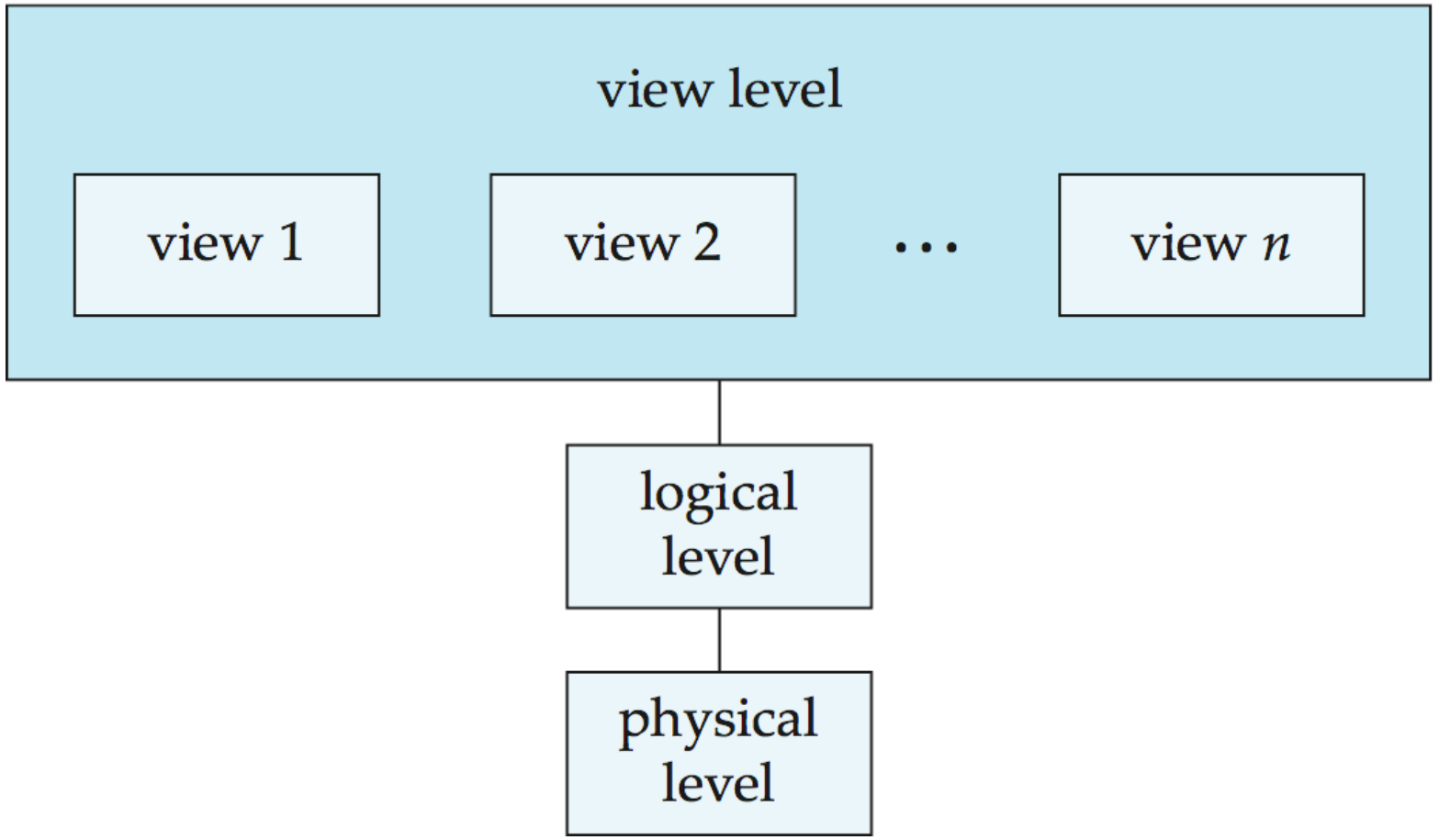
  ID : string;

  name : string;

  dept_name : string;

  salary : integer;
- **View level:** application programs hide details of data types. Views can also hide information (such as an employee's salary) for security purposes.

An Architectural View of RDBMS



Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

Interacting with RDBMS

- Structured Query Language (SQL) is the language to communicate with RDBMS to perform operations
- Two kind of operations
  - Data Definiton Language (DDL)
  - Data Manipulation Language (DML)

Structured Query Language (SQL)

- IBM Sequel language developed as part of System R project at the IBM San Jose Research Laboratory
- Renamed Structured Query Language (SQL)
- The **most widely used** commercial language
- To be able to compute complex functions SQL is usually embedded in some higher-level language
- Application programs generally access databases through one of
  - Language extensions to allow embedded SQL
  - Application program interface (e.g., ODBC/JDBC) which allow SQL queries to be sent to a database
- Commercial systems offer most standard features, plus varying special proprietary features.
- Not all examples here may work on all RDBMS as it
  - Example: A command to get first 100 records in a table in written different in MSSQL and Oracle.

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

Data Definition Language (DDL)

- Specification notation for defining the database schema
- Example: create table instructor (

  ID char(5),

  name varchar(20),

  dept_name varchar(20),

  salary numeric(8,2))
- DDL compiler generates a set of table templates stored in a data dictionary
- Data dictionary contains metadata (i.e., data about data)
- Database schema
- Integrity constraints
  - Primary key (ID uniquely identifies instructors)
- Authorization
  - Who can access what

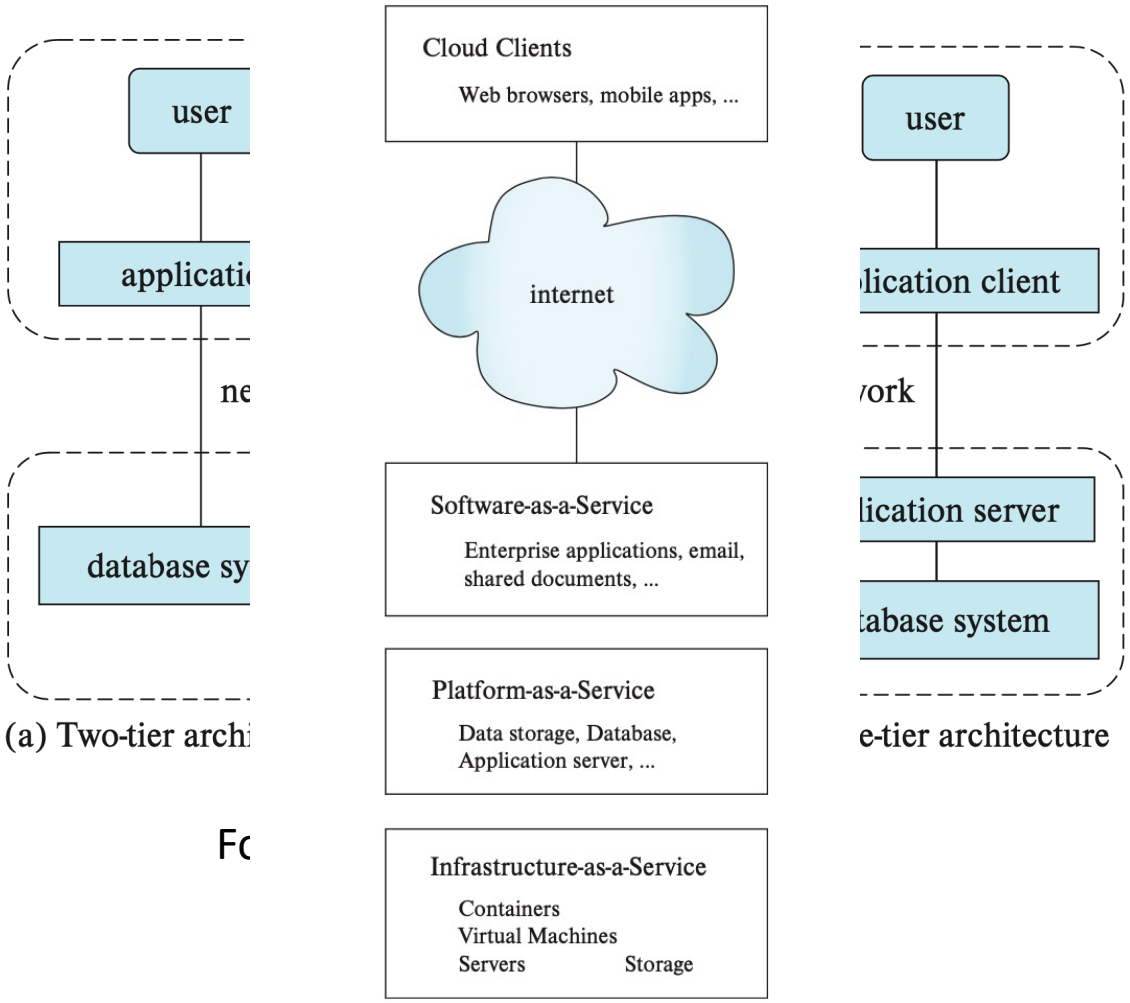
Data Manipulation Language (DML)

- Language for accessing and manipulating the data organized by the appropriate data model
- Language to gain access to all the data within Relations/Tables
- DML also known as query language
- Examples of DML
  - SELECT Queries
  - Example: Including complex aggregate and analytical functions
  - Update Queries
  - Delete Queries (Data only)

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

DATABASE ARCHITECTURES

Database Architectures

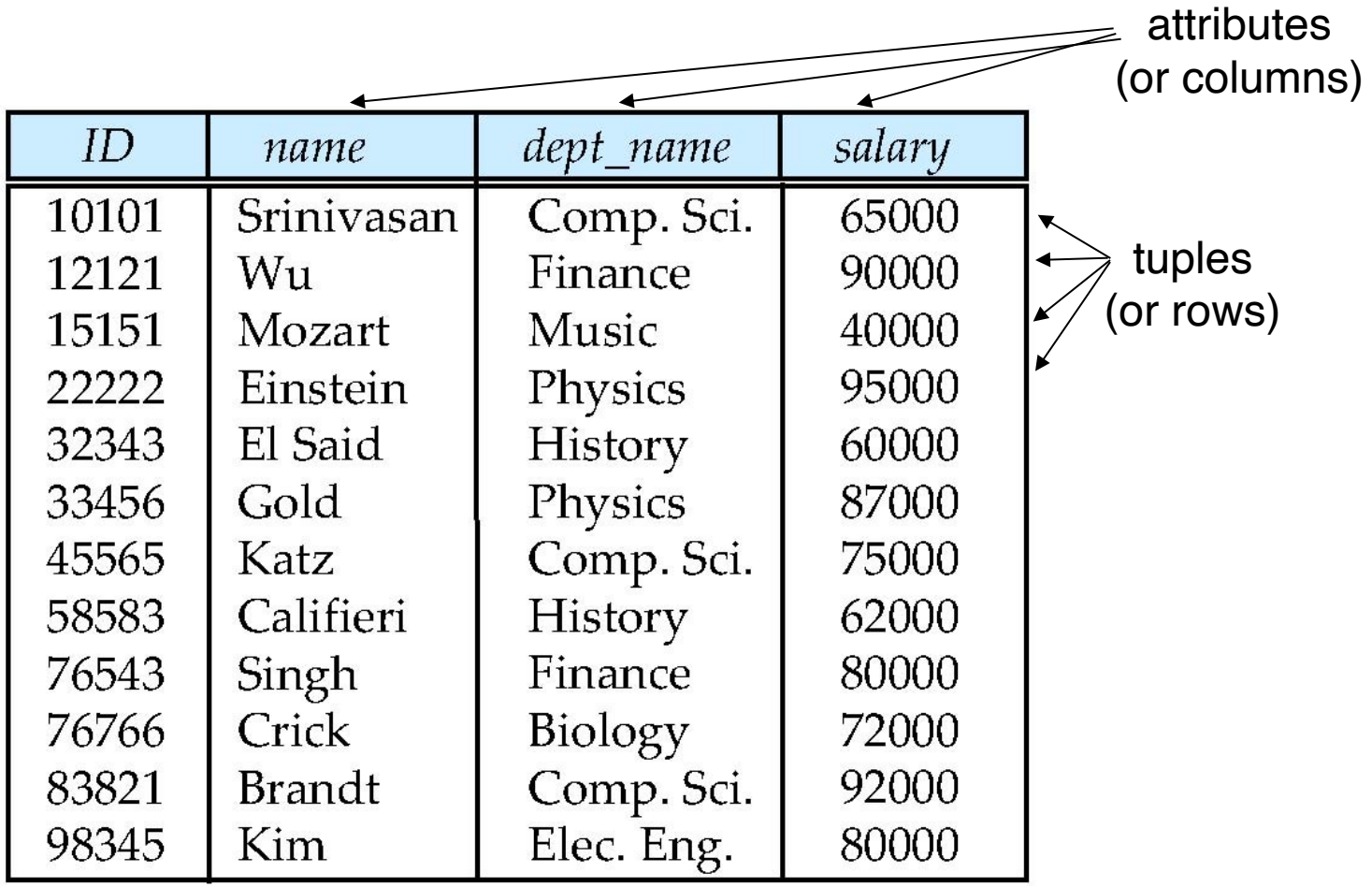


- The architecture of a database systems is greatly influenced by
  - organizational data management structures
  - the underlying application architectures to which the database is serving.
- Centralized. Classic applications within organizations. For example MS Access
- Client-server
- Cloud Systems
- Distributed

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

FUNDAMENTALS OF RELATIONAL MODEL

What is a Relation / Table



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

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

Relation Schema and Instance

- A_1, A_2, …, A_n are *attributes*

- $R = (A_1, A_2, \ldots, A_n)$ is a *relation schema*

  Example:

  instructor = (*ID, name, dept_name, salary*)

- A Tuple t is a list of values for Attributes A_1, A_2, …, A_n
- The current values (relation instance) of a relation are specified by a table
- An element t of r is a tuple, represented by a row in a table

Attributes Data Types

- Sufficient close to Data
- Avoid text/string/char for every data attribute
- Date is a challenging attribute

bigint	**int8**	signed eight-byte integer	json		textual JSON data
bigserial	Serial8	autoincrementing eight-byte integer	numeric [(***p***, ***s***)]	decimal [(***p***, ***s***)]	exact numeric of selectable precision
boolean	bool	logical Boolean (true/false)	text		variable-length character string
character	char	fixed-length character string	timestamp	timestamptz	date and time, including time zone
character varying	varchar	variable-length character string	real	float4	single precision floating-point number (4 bytes)
integer	int, int4	signed four-byte integer	Date		calendar date (year, month, day)

https://www.postgresql.org/docs/current/datatype.html

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

Relations Order

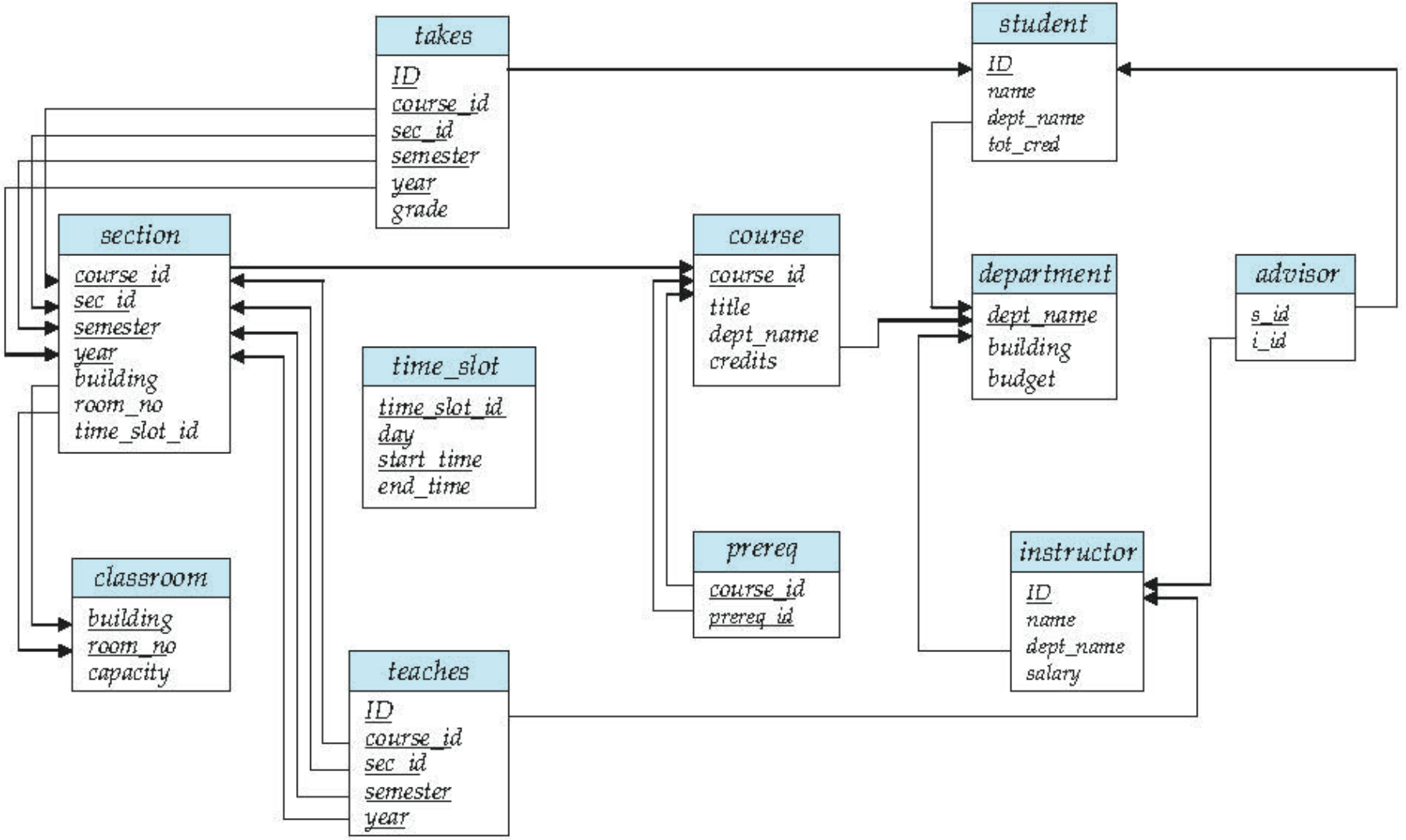
- *Order of tuples is irrelevant (tuples may be stored in an arbitrary order)*
- *Example: instructor relation with unordered tuples*

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

Keys – Super Key, Candidate Key, Primary Key, Foreign Key

- *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: {ID} and {ID,name} are both superkeys of instructor.*
- *Superkey K is a **candidate** key if K is minimal. Does not have repeating attribute.*
  - *Example: {ID} is a candidate key for Instructor*
- *One of the candidate keys is selected to be the **primary key**.*
- **Foreign key** constraint: Value in one relation must appear in another
  - Referencing relation
  - Referenced relation
  - Example – dept_name in instructor is a foreign key from instructor referencing department

Let us examine this Entity Relation Diagram



Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

RELATIONAL DATABASE DESIGN

Design Phases

- There are three phases of a Relational Database design
- Conceptual Design
  - Requirements gathering
  - Identify purpose and entities/relations
  - Define business rules
  - Typical outcome is a rough Entity Relationship sketch
- Logical Design
  - Define initial design further with annotations. i.e. ER model
  - Define relations between entities using Foreign Keys
  - Apply Normalization
  - Independent of the underlying platform
- Physical Design
  - Implement Database in chosen Platform using DDL statement
  - Datatypes, constraints and index decisions

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

Conceptual Design

- Analyze the given organization or scenario
- Work close to stake holders
- Identify Entities and their relations
- Discuss Business rules and future perspectives
- Discuss underlying technologies
- Further refine each entity to create final design using well proven methods.

SkemaLabel	**StudentCourse**							
Attributes								
studentid	name	tot_cred	course_id	coutse_name	section_id	semester	year	building

SkemaLabel	**CoursePreReq**			
Attributes				
course_name	course_id	prerequisite_course_id	course_id	prerequisite_course_name

ENTITY RELATIONSHIP MODEL
(ER MODEL)

Entity Relationship Diagram

- Entity Relationship Diagram is a conceptual schema
- It is used to represent the logical structure of database graphically
- It provides an overview of all the entities in one system and their relation to each other
- Three elements of ER Diagram
  - Entity Sets
  - Attributes
  - Relationship Sets

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

Entity Sets (Instructor and Student)

instructor_ID	instructor_name
76766	Crick
45565	Katz
10101	Srinivasan
98345	Kim
76543	Singh
22222	Einstein

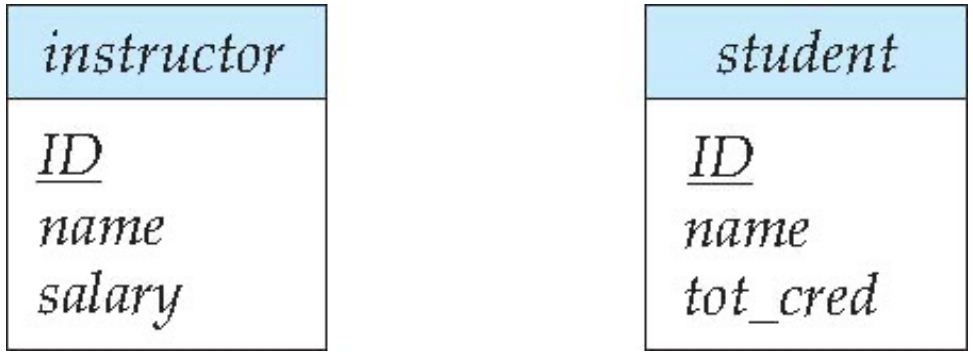
instructor

student-ID	student_name
98988	Tanaka
12345	Shankar
00128	Zhang
76543	Brown
76653	Aoi
23121	Chavez
44553	Peltier

student

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

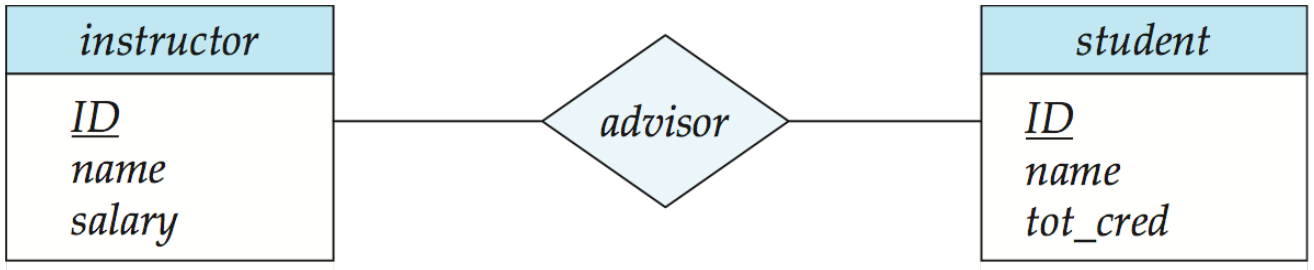
Entity Sets Representation



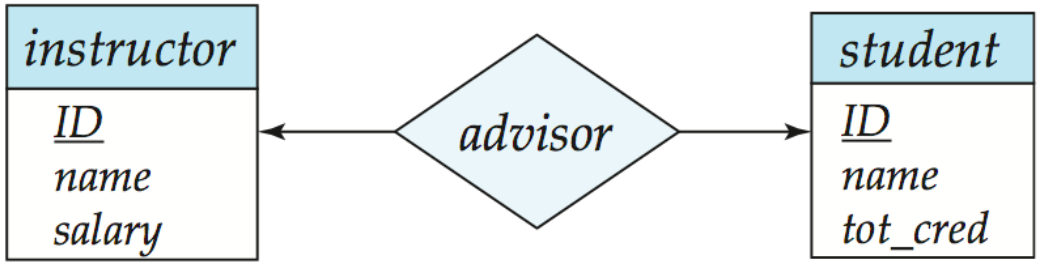
- Entities can be represented graphically as follows:
  - Rectangles represent entity sets.
  - Attributes listed inside entity rectangle
  - Underline indicates primary key attributes

Relationship Sets

- Diamond represents Relationships



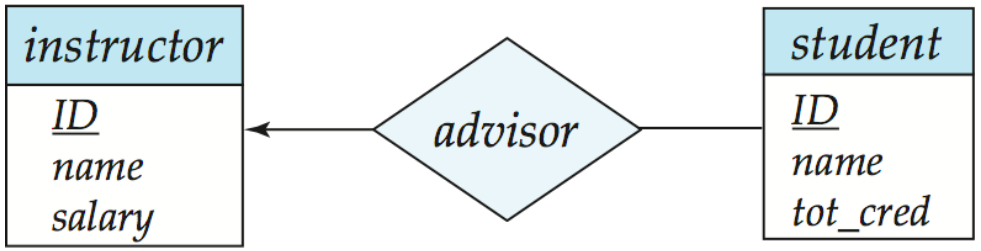
Cardinality Sets



- This presentatiton express cardinality constraints by drawing either a directed line (→), signifying “one,” or an undirected line (—), signifying “many,” between the relationship set and the entity set.

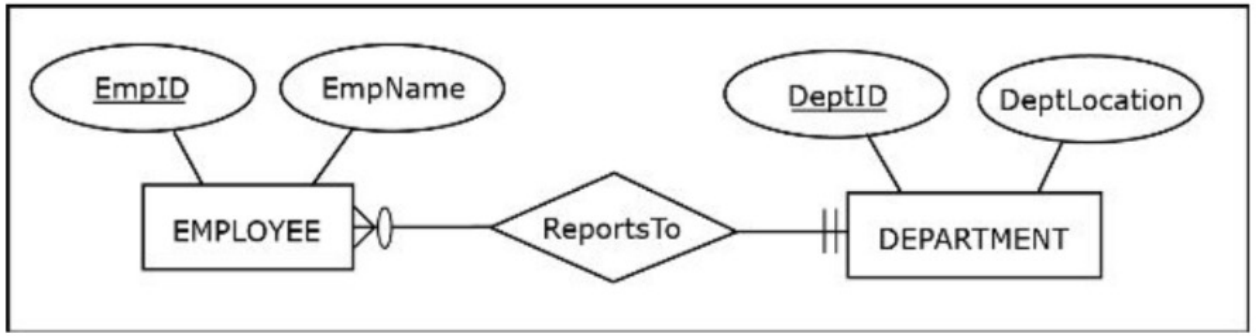
- One-to-one relationship between an instructor and a student :
  - A student is associated with at most one instructor via the relationship advisor
  - A student is associated with at most one department via stud_dept

One-Many Relationship



- one-to-many relationship between an instructor and a student
  - an instructor is associated with several (including 0) students via advisor
  - a student is associated with at most one instructor via advisor

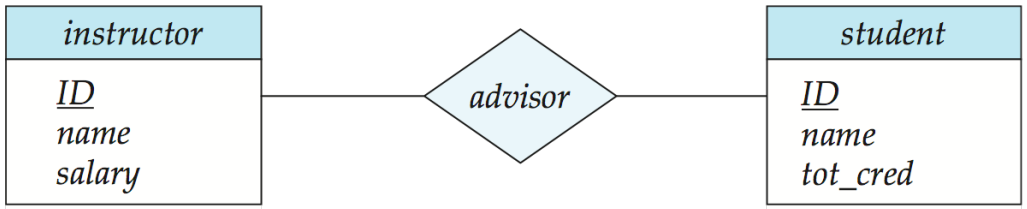
One-Many Relationship – Another annotation

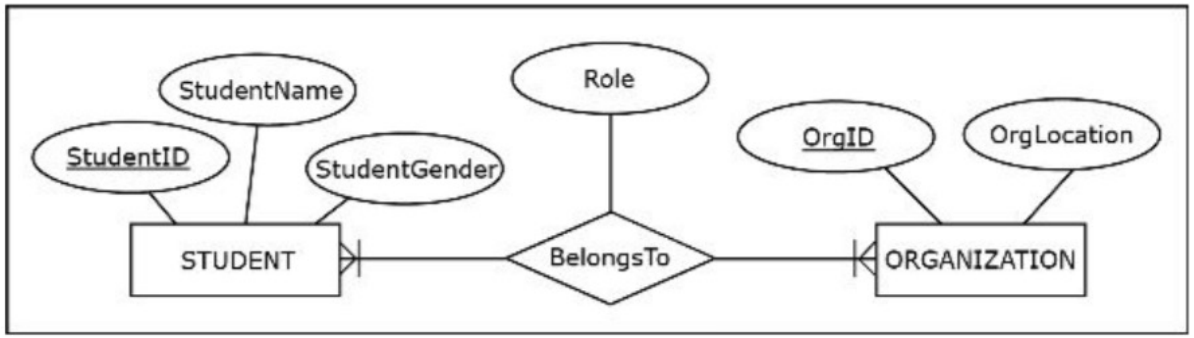


- one-to-many relationship between an employee and a department
  - an employee reports to one and only one department
  - a department can have 0 or many employees

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

Many-Many Relationship

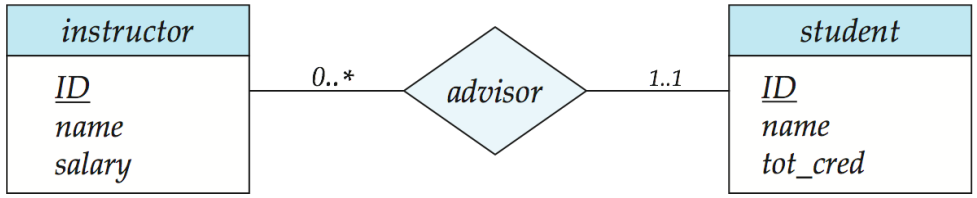




- An instructor is associated with several (possibly 0) students via advisor
- A student is associated with several (possibly 0) instructors via advisor
- A student can have different role in many organizations.
- One organization can have many students

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

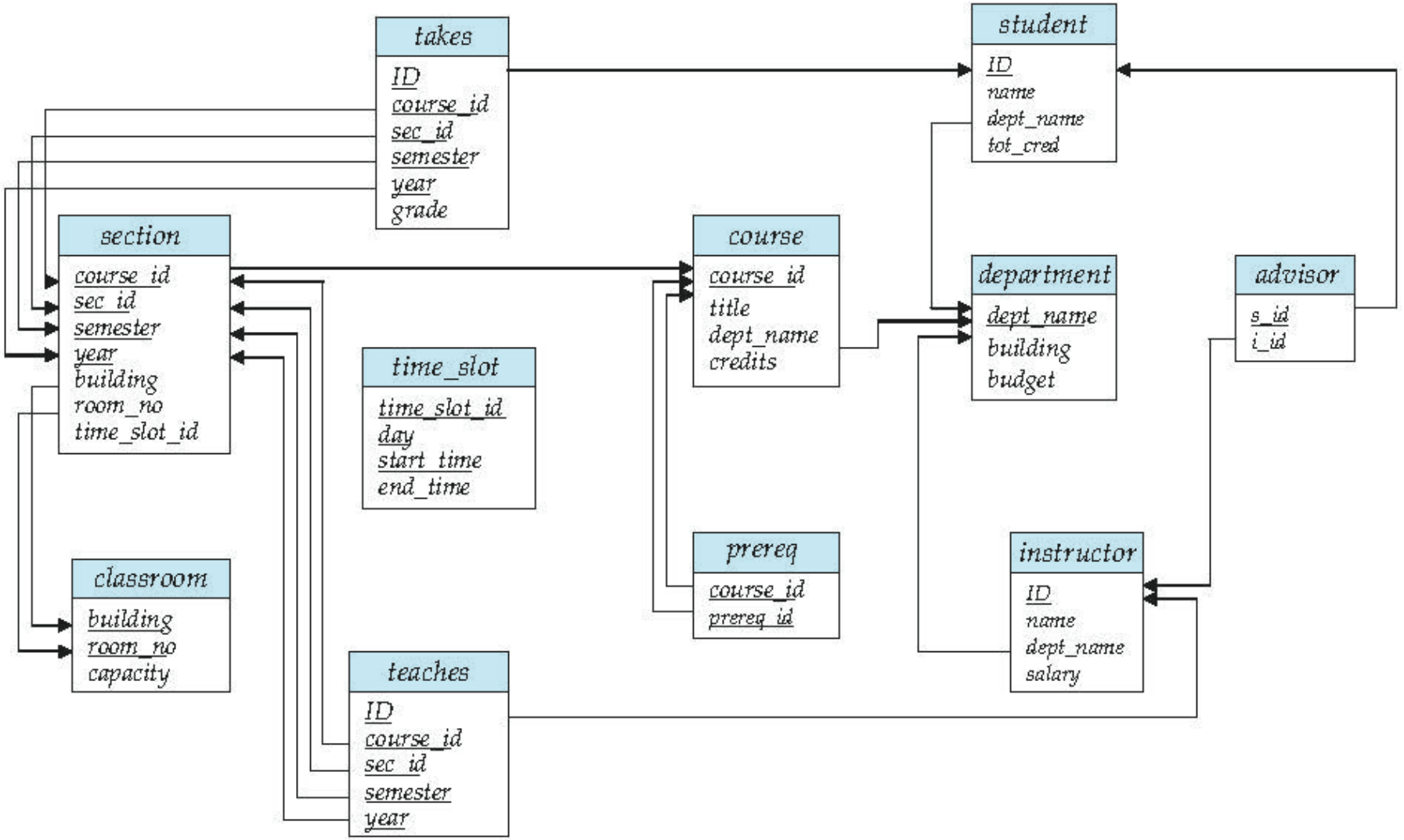
Notation for Complex Constraints



Instructor can advise 0 or more students. A student must have 1 advisor; cannot have multiple advisors

- A line may have an associated minimum and maximum cardinality, shown in the form l..h, where l is the minimum and h the maximum cardinality
- A minimum value of 1 indicates total participation.
- A maximum value of 1 indicates that the entity participates in at most one relationship
- A maximum value of * indicates no limit.

Entity Relation Diagram Implemented (Simplified)



Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

NORMALIZATION DATABASE PROCESS

Normalization

- Normalization is a well known method for relational database design.
- The goal is to derive a set of schemas each of which are in “good form”
  - Allow storing information without un-necessary redundancy
  - Allow information retrieval easily and completely
- The approach works in the following manner:
  - Find Good Form for tables
  - Decompose if needed to further tables
  - Evaluate again
  - The decomposition should not result in Loss of Information.

“Good Form” or Normal Form

- Each Normal Form of schema is defined on the basis of certain globally defined requirements that must be met to establish that given schema is in a given normal form.
- Normal Forms are as follows
  - 1NF (First Normal Form)
  - 2NF (Second Normal Form)
  - 3NF (Third Normal)
- Generally 3NF form is considered a good form for schema however following two forms are used as well
  - BCNF (Boyce–Codd Normal Form)
  - 4NF (Fourth Normal Form)

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

1NF – First Normal Form

- Domain is atomic if its elements are considered to be indivisible units
  - Examples of non-atomic domains:

    Set of names, composite attributes

    Identification numbers like CS101 that can be broken up into parts
- A relational schema R is in **first normal form** if the domains of all attributes of R are atomic
  - Non-atomic values complicate storage and encourage redundant (repeated) storage of data
  - Example: Set of accounts stored with each customer,

    and set of owners stored with each account
- Values stored in a column should be of the same domain
- All the columns in a table should have unique names
- And the order in which data is stored, does not matter

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

1NF – First Normal Form (Automicity)

- Atomicity is actually a property of how the elements of the domain are used.
  - Example: Strings would normally be considered indivisible

    Suppose that students are given roll numbers which are strings of the form **CS0012** or **EE1127**
  - If the first two characters are extracted to find the department, the domain of roll numbers is not atomic.
  - Doing so is a bad idea: leads to encoding of information in application program rather than in the database

1NF – First Normal Form (Examples)

ID	FirstName	LastName	MobilePhone	Courses
3453	Abid	Hussain	12345678;9876543	Image Processing;Differential Equations
3456	Adam	Duke	02020202;08080808	Mathematics;Databases

ID	FirstName	LastName	MobilePhone	Course
3453	Abid	Hussain	12345678	Image Processing
3456	Adam	Duke	2020202	Mathematics
3453	Abid	Hussain	9876543	Differential Equations
3456	Adam	Duke	8080808	Databases

- This schema does not represent atomic values
  - Mobile phones contains multiple values
  - Courses contain multiple values
- This schema contains only Atomic values
  - **MobilePhone** is repeated but is atomic
  - Simmilarly with **Course**
  - This results in repetition however this will be solved in further NF forms

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

1NF – First Normal Form (Examples)

This Table is in 1NF

InstructorId	coursename	dept_name	credits	building	budget	instructor_name	salary
48507	Diffusion and Phase Transformation	Mech. Eng.	3	Rauch	520350.65	Lent	107978.47
97302	Diffusion and Phase Transformation	Mech. Eng.	3	Rauch	520350.65	Bertolino	51647.57
43779	Image Processing	Astronomy	3	Taylor	617253.94	Romero	79070.08
48507	Differential Equations	Mech. Eng.	3	Rauch	520350.65	Lent	107978.47
97302	Differential Equations	Mech. Eng.	3	Rauch	520350.65	Bertolino	51647.57
77346	Thermodynamics	Geology	3	Palmer	406557.93	Mahmoud	99382.59
50330	Differential Geometry	Physics	3	Wrigley	942162.76	Shuming	108011.81
74420	Differential Geometry	Physics	3	Wrigley	942162.76	Voronina	121141.99
80759	Antidisestablishmentarianism in Modern America	Biology	4	Candlestick	647610.55	Queiroz	45538.32
81991	Antidisestablishmentarianism in Modern America	Biology	4	Candlestick	647610.55	Valtchev	77036.18
6569	Manufacturing	Finance	3	Candlestick	866831.75	Mingoz	105311.38
28097	Number Theory	English	4	Palmer	611042.66	Kean	35023.18
4233	Number Theory	English	4	Palmer	611042.66	Luo	88791.45
72553	Number Theory	English	4	Palmer	611042.66	Yin	46397.59
95709	Number Theory	English	4	Palmer	611042.66	Sakurai	118143.98
63395	Elastic Structures	Cybernetics	3	Mercer	794541.46	McKinnon	94333.99
65931	Elastic Structures	Cybernetics	3	Mercer	794541.46	Pimenta	79866.95
90376	Elastic Structures	Cybernetics	3	Mercer	794541.46	Bietzk	117836.50
99052	Elastic Structures	Cybernetics	3	Mercer	794541.46	Dale	93348.83
3199	Marine Mammals	Elec. Eng.	3	Main	276527.61	Gustafsson	82534.37
42782	Marine Mammals	Elec. Eng.	3	Main	276527.61	Vicentino	34272.67
73623	Marine Mammals	Elec. Eng.	3	Main	276527.61	Sullivan	90038.09
79653	Marine Mammals	Elec. Eng.	3	Main	276527.61	Levine	89805.83

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

1NF – First Normal Form (Issues with the Schema)

- Data Anomaly
  - The data is still heavily duplicated.
- Insertion Anomaly
  - If we need to add an instructor in Database without assigning a course. We have to keep null values in course and building fields
  - A new building information can’t be added without adding course information.
- Updation Anomaly
  - Just changing instructor for a course will require updating all records.
- Deletion Anaomaly
  - Let say a building has to be deleted? It will result in deletion of instructors and many more.

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

2NF – Second Normal Form

- Let us recall Primary Keys and Candidate Keys. Keys that uniquely identify each record.
- Identify Keys.
  - Now from attributes **InstructorId, coursename, dept_name, credits, building, budget, instructor_name, salary** which is unique key.
  - We need to refer to business rules to identify that
    - Department has only one building
    - Instructor has only one department
  - Hence we can say ***InstructorId*** and ***dept_name*** together can be a **candidate key** and also the **primary key**.

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

2NF – Second Normal Form

- A relation is in second normal form 2NF if
  - The relation is in 1NF
  - There is not Partial Dependency
  - Partial Dependency exists, when for a composite primary key, any attribute in the table depends only on a part of the primary key and not on the complete primary key.
- Example: For our current schema attributes

  InstructorId, coursename, dept_name, credits, building, budget, instructor_name, salary

  Primary key = (**InstructorId, dept_name**) which is a composite key.

  Attributes: ***coursename, credits, instructorname*** can be identified by part of primary key (***InstructorId)***

  Attributes: ***building, budget, salary*** can be identified by part of primary key (***dept_name***)

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

2NF – Second Normal Form (Continued..)

Notice duplicates

InstructorInfo

InstructorId	coursename	credits	instructor_name	salary
48507	Diffusion and Phase Transformation	3	Lent	107978.47
97302	Diffusion and Phase Transformation	3	Bertolino	51647.57
43779	Image Processing	3	Romero	79070.08
48507	Differential Equations	3	Lent	107978.47
97302	Differential Equations	3	Bertolino	51647.57
77346	Thermodynamics	3	Mahmoud	99382.59
50330	Differential Geometry	3	Shuming	108011.81
74420	Differential Geometry	3	Voronina	121141.99
80759	Antidisestablishmentarianism in Modern America	4	Queiroz	45538.32

department

dept_name	building	budget
Accounting	Saucon	441840.92
Astronomy	Taylor	617253.94
Athletics	Bronfman	734550.70
Biology	Candlestick	647610.55
Civil Eng.	Chandler	255041.46
Comp. Sci.	Lamberton	106378.69
Cybernetics	Mercer	794541.46
Elec. Eng.	Main	276527.61
English	Palmer	611042.66

- To remove the partial dependency, we remove the attributes causing that and create new relations.
- Resulting Relations of 2NF
  - InstuctorInfo (key: InstructorId)
  - Department (key: dept_name)

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

2NF – Second Normal Form (Continued..)

- We have following two table
  - InstructorInfo (InstructorId, coursename, credits, instructor_name, salary)
  - Department (dept_name, building, budget)
- Next we check each relation for Normal Forms.

3NF – Third Normal Form

- A relation is in third normal form 3NF if
  - The relation is in 2NF
  - There is no Transitive Dependency
  - Transitive Dependency exists, when a non key attribute depends on another non key attribute.
  - Example: For our current schema attributes

    InstructorInfo (InstructorId, coursename, credits, instructor_name, salary)

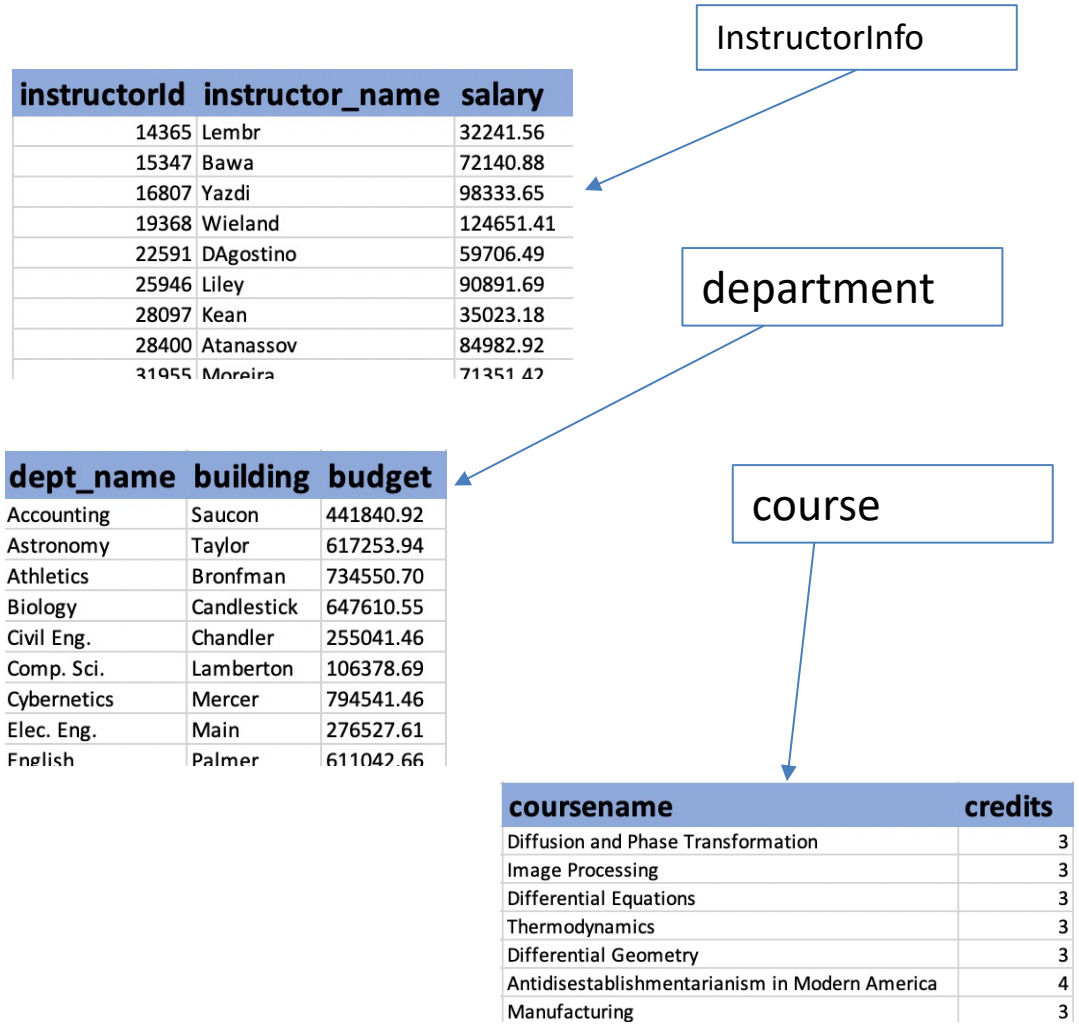
    credits can be independently depending on **coursename** without have any influence of key.

    Department (dept_name, building, budget)

  Same applies here that budget can depend on building only however becasue assumption that one department has only one building. We can accept this as good enough Normal Form

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

3NF – Third Normal Form (Continued..)



instructorId	instructor_name	salary
14365	Lembr	32241.56
15347	Bawa	72140.88
16807	Yazdi	98333.65
19368	Wieland	124651.41
22591	DAgostino	59706.49
25946	Liley	90891.69
28097	Kean	35023.18
28400	Atanassov	84982.92
31955	Moreira	71351.42

dept_name	building	budget
Accounting	Saucon	441840.92
Astronomy	Taylor	617253.94
Athletics	Bronfman	734550.70
Biology	Candlestick	647610.55
Civil Eng.	Chandler	255041.46
Comp. Sci.	Lamberton	106378.69
Cybernetics	Mercer	794541.46
Elec. Eng.	Main	276527.61
English	Palmer	611042.66

coursename	credits
Diffusion and Phase Transformation	3
Image Processing	3
Differential Equations	3
Thermodynamics	3
Differential Geometry	3
Antidisestablishmentarianism in Modern America	4
Manufacturing	3

- To remove the transitive dependency, we remove the attributes causing that and create new relations.
- Resulting Relations of 3NF
  - InstuctorInfo (key: InstructorId)
  - Course (key: coursename)
  - Department (key: dept_name)

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

Finishing your design

- After Testing and Normalizing all the relations to third Normal form. There are steps that you need to perform.
  - Assign Unique Ids. For example, StudentId, InstructorId
  - Define foreign key relations.
    - The attributes with a primary key role in the main table are added to the referenced table to connect.
    - This primary key of primary table is referred as Foreign Key in the detailed/referenced table
  - Foreign key relations are drawn to connect different relations. This can result in creating new relations. For example connecting a student and course taken relation requires.
    - a new relation named takes
    - studentid and courseid is the data in that table.
  - Draw revised ER Model to keep an overview of entity structure.

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

Final Set of Relational Model

classroom(<u>*building*</u>, <u>*room_number*</u>, *capacity*)
department(<u>*dept_name*</u>, *building*, *budget*)
course(<u>*course_id*</u>, *title*, *dept_name*, *credits*)
instructor(<u>*ID*</u>, *name*, *dept_name*, *salary*)
section(<u>*course_id*</u>, <u>*sec_id*</u>, <u>*semester*</u>, <u>*year*</u>, *building*, *room_number*, *time_slot_id*)
teaches(<u>*ID*</u>, <u>*course_id*</u>, <u>*sec_id*</u>, <u>*semester*</u>, <u>*year*</u>)
student(<u>*ID*</u>, *name*, *dept_name*, *tot_cred*)
takes(<u>*ID*</u>, <u>*course_id*</u>, <u>*sec_id*</u>, <u>*semester*</u>, <u>*year*</u>, *grade*)
advisor(<u>*s_ID*</u>, *i_ID*)
time_slot(<u>*time_slot_id*</u>, <u>*day*</u>, <u>*start_time*</u>, *end_time*)
prereq(<u>*course_id*</u>, <u>*prereq_id*</u>)

IMPLENTATION OF DATABASE DESIGN

Data Definition Language (DDL)

- The SQL data-definition language (DDL) allows the specification of information about relations, including:
  - The schema for each relation.
  - The domain of values associated with each attribute. For example: Datatypes of values
  - Integrity constraints

Domain Types in SQL (Data Types)

- char(n). Fixed length character string, with user-specified length n.
- varchar(n). Variable length character strings, with user-specified maximum length n.
- int. Integer (a finite subset of the integers that is machine-dependent).
- smallint. Small integer (a machine-dependent subset of the integer domain type).
- numeric(p,d). Fixed point number, with user-specified precision of p digits, with d digits to the right of decimal point. (ex., numeric(3,1), allows 44.5 to be stores exactly, but not 444.5 or 0.32)
- real, double precision. Floating point and double-precision floating point numbers, with machine-dependent precision.
- float(n). Floating point number, with user-specified precision of at least n digits.
- **Other complex types to store Images, Json, Huge Text Documents …**

Silberschatz, A., Korth, H., Sudarshan, S. Database System Concepts. 7th Edition. McGraw-Hill Education. ISBN13: 9780078022159

Create Table Construct

- An SQL relation is defined using the create table command:

  create table r (A1 D1, A2 D2, ..., An Dn,

  (integrity-constraint1),

  ...,

  (integrity-constraint$_k$))

  - r is the name of the relation
  - each Ai is an attribute name in the schema of relation r
  - Di is the data type of values in the domain of attribute Ai

Create Table Construct

Example:

```
create table instructor (
        ID              char(5),
        name            varchar(20),
        dept_name       varchar(20),
        salary          numeric(8,2))
```

Integrity Constraints in Create Table

- not null
- primary key (A1, ..., An)
- foreign key (Am, ..., An) references r

Example:

```
create table instructor (
        ID              char(5),
        name            varchar(20) not null,
        dept_name       varchar(20),
        salary          numeric(8,2),
        primary key (ID),
        foreign key (dept_name) references
department);
```

primary key declaration on an attribute automatically ensures **not null**

Create Table Examples

```
create table student (
        ID                  varchar(5),
        name                varchar(20) not null,
        dept_name           varchar(20),
        tot_cred            numeric(3,0),
        primary key (ID),
        foreign key (dept_name) references department);

create table takes (
        ID                  varchar(5),
        course_id           varchar(8),
        sec_id              varchar(8),
        semester            varchar(6),
        year                numeric(4,0),
        grade               varchar(2),
        primary key (ID, course_id, sec_id, semester, year) ,
        foreign key (ID) references  student,
        foreign key (course_id, sec_id, semester, year)
        references section);
```

```
create table course (
        course_id           varchar(8),
        title               varchar(50),
        dept_name           varchar(20),
        credits             numeric(2,0),
        primary key (course_id),
        foreign key (dept_name)
        references department);
```

Updates on Tables

- **Insert**
  - *insert into instructor values (‘10211’, ’Smith’, ’Biology’, 66000);*
  - Insert into instructor(A1,A2,A3) Values(V1,V2,V3);
- **Delete**
  - Remove all tuples from the student relation

    Example : *delete from student;*
- **Drop Table**
  - *drop table r*
- **Alter**
  - *alter table r add A D*
  - where A is the name of the attribute to be added to relation r and D is the domain of A.
  - All exiting tuples in the relation are assigned null as the value for the new attribute.

Insertion

- Add a new tuple to *course*

  insert into *course*

  values (’CS-437’, ’Database Systems’, ’Comp. Sci.’, 4);

- or equivalently

  insert into *course* (*course_id*, *title*, *dept_name*, *credits*)

  values (’CS-437’, ’Database Systems’, ’Comp. Sci.’, 4);

- Add a new tuple to *student* with *tot_creds* set to null

  insert into *student*

  values (’3003’, ’Green’, ’Finance’, *null*);

Insertion

- Add all instructors to the *student* relation with tot_creds set to 0

  insert into *student*

  select *ID, name, dept_name, 0*

  from *instructor*

- The **select from where** statement is evaluated fully before any of its results are inserted into the relation.

  Otherwise queries like

  insert into *table*1 **select** * **from** *table*1

  would cause problem

Deletion

- Delete all instructors

  delete from *instructor*

- Delete all instructors from the Finance department

  delete from *instructor*

  where *dept_name*= ’Finance’;

- Delete all tuples in the *instructor* relation for those instructors associated with a department located in the Watson building.

  delete from *instructor*

  where *dept name* **in** (**select** *dept name*

  from *department* **where** *building* = ’Watson

Updates

- Increase salaries of instructors whose salary is over $100,000 by 3%, and all others by a 5%
  - Write two **update** statement

```
update instructor
    set salary = salary * 1.03
    where salary > 100000;
update instructor
    set salary = salary * 1.05
    where salary <= 100000;
```

  - The order is important

THANK YOU