Relational Database Design

Date- Feb 28

OBJECTIVE

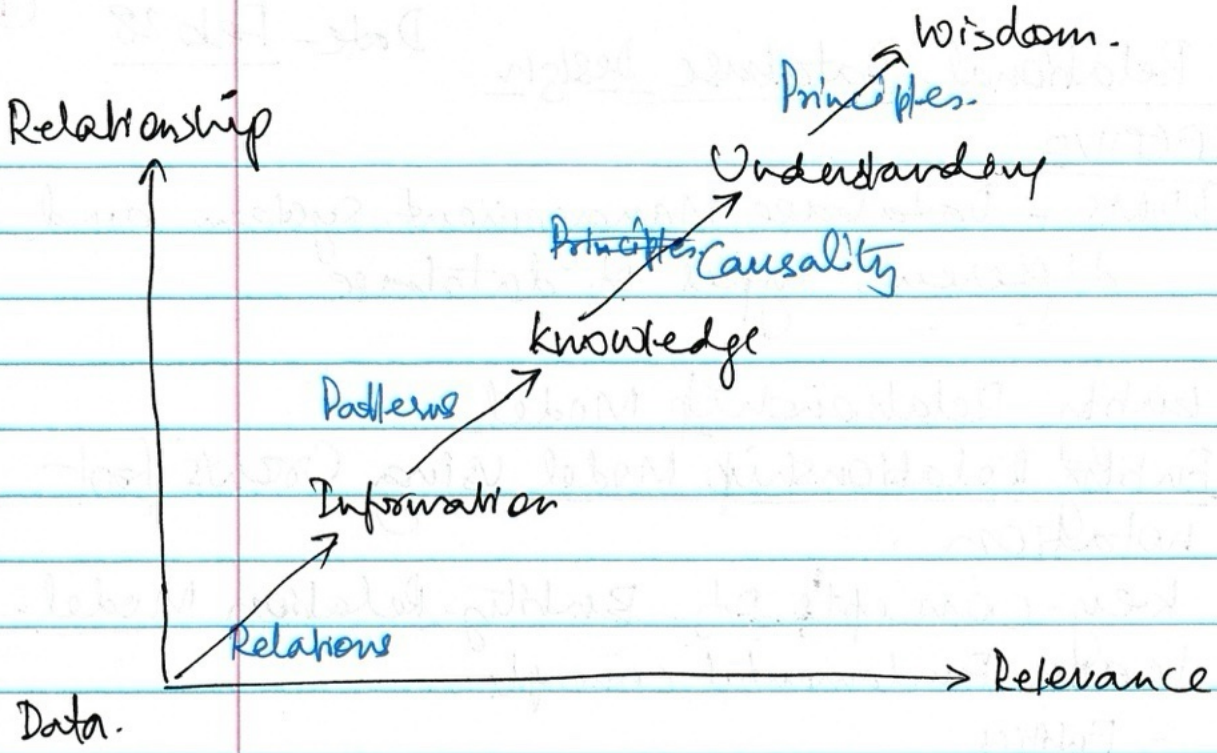
→ DBMS - Database Management system and different types of database.

→ Entity Relationship Model.
→ Entity Relationship Model using Crow's foot notation
→ key concepts of Entity Relation Model.
→ Learn Fundamental concepts
- Entities
- Relationships
- Relations
- keys
- Functional Dependencies
- Normalization.

Course Outline
- Entity Relationship Models - Conceptual model.
- Relational Models - implementable model.
- Normalization - remove data redundancy
- Case study.

Entity Relationship Model.

Data -
Information - data with meaning
knowledge - aggregate information
Intelligence - decision making analysis.



Solutions of Data Management

File System Solution. - Collection of individual files accessed by applications programs.

Limitations -

- Separated and Isolated Data.
- Data duplication
- Application Program Dependencies.
- Lack of data sharing.

Database Solution

- Structure that contains data.
- Organize data through.
  - Entities.
  - Attributes
  - Relationships.

Popular DBMS

- Microsoft Access
- MySQL
- PostgreSQL
- MongoDB (NoSQL)

Advantages of DBMS

- Sharing data
- Balancing conflicting requirement
- Controlling redundancy
- Facilitating consistancy
- Referential Integrity.
- Expanding security.
- Increasing productivity.

Contents of DBMS. Components of DBMS

- User data - stored in tables
- Metadata - data about the data
- Indexes -
- Application Metadata.

Metadata - Data that describe how user data are stored in terms of table name, column name, data type, length, primary keys. etc.
- Usually stored in system tables or system catelog

Indexes — provide an alternate means of accessing user data. sorting and searching
— allow the database to access a record without having to search through entire table.
— Updating data requires an extra step. the index must also be updated.

Applications Metadata.
— Many DBMS have storage facilities for forms reports, queries and other application components.
— Applications Metadata is accessed via the database development programs.

Questions

1. Data is raw facts and figures, and information is organised and processed data.

3. Entities are the abstract classes of subjects of interest.

Entities and Attributes.

Entity Relationship Model (ER Model)

To design a relational database, we start with an Entity-Relationship Model (ER Model)

ER Model describes
- what are the entities the database is going to record.
- What are the attributes (and identifiers) of the entities
- how the relationships among the entities

Entity - The identifiable abstract object of interest (in programming, we call class)
examples
- students -
- employees -
- products - price, manufacturer, quality etc..
- transactions -

Attributes -
- property of entity or characteristics of an entity.
Students: FirstName, Student ID, Major.
Employees: ID, SSN, Contact Info, Department, Supervisor.
Products: SKU, Category, InStockPrice, InStockQuantity.
Transactions: CustomerID, StoreID, ProductID, Product Description, Price, Quantity, Tax, Total Time

Instance - Actual Record., Rows in a table.
- record / member of an entity of interest.
Students: FirstName = "Joe", Student ID = "DB001", Major = "Data Science"
employee: ID = "E0099", SSN = "123-45-678", Contact Info = "N/A". Department = "Education", Supervisor = "Smith".

Identifiers

- a special attribute used to identify a specific instance of an entity.
- unique and maybe natural (your DBMS doesn't need to create it) SSN
  artificial as: employee ID.

Relationships

- Associations of two or more entities
- How the members of two (or more) entities are connected.

examples.

* Employees and Companies
  - How many companies a employee can work?
  - How many employees a Company can have? one to many
* Companies and products.
  - How many companies a product can belong to (one to one)
  - How many products a company can produce

Relationship - Degree

* Relationship can include one or more entities
* The degree of a relationship is the number of Entities that participate in the relationship.

Relationships of degree 1 are called Unary relationships (also called Recursive)

Relationships of degree 2 are called Binary relationships. Most relationships in databases are Binary.

Relationships - Cardinality

Cardinality refers to the number of instances of the entity involved in the relationship.
- also called max cardinality/multiplicity of a relationship.

There are three types:
- 1:N (One to Many)
- N:1 (Many to One)
- N:M (many to Many)

Relationships - Participation

- Participation of instances in a relationship may be mandatory or optional.
- Also called optionality, minimal cardinality of a relationship

There are two types:
- Mandatory
- Optional.

Example:

Case Study:- Online-education Company

Requirements:-

The company has more than 100 instructors, more than 400 courses, more than 50 online programs and around 1 million students.

Instructors have info: Name, *Emp ID, SSN, DoB, Email, Salary.
Courses have info: *Title, *Course #, Time, Location, Description.
Programs: *Title, Chair, office #, Contact, Description.

Students: Name, *StuID, DoB, Email
↳ Identifiers.

What are our Entities:
- Instructors, Courses, Programs, Students.

How entities are connected.
- An Instructor may teach multiple courses, and a course might be taught by multiple instructors
- Each course must belong to one and only one online-program, and each program must have one or more courses.
- A student may take multiple courses, and a course must have one or more students.

- A student must belong to exactly one program and a program may have one or more students
  - must → mandatory
  - may → participation - optional.
  - one or more → Cardinality - Many
- A student may be friend of other students.
  - may → optional.
  - Cardinality - Many.
- A student must have exactly one instructor as advisor, and one instructor may advise one or more students.

Entity Relationship Diagram

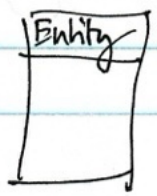
- Represent Entities (and attributes/identifier)
- Represent Relationship (and cardinality/participation)
- Provide a high level picture of what the database is organized.

Notations

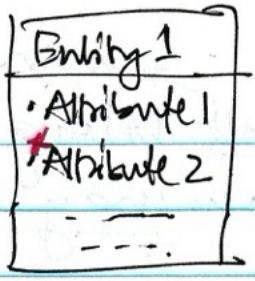
- UML - Unified Modeling Language
- Chen
- Crow's Foot — use Lucid.app to draw ERD

ERD with Crow's Foot

Entities →

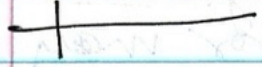


- Attributes.
- Identifiers.

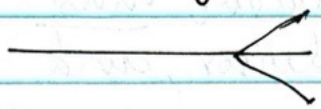


- Relationships
  - Name (normally a verb)
  - Cardinality (One, Many)

One



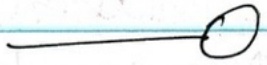
Many



- Participations

Mandatory.

optional

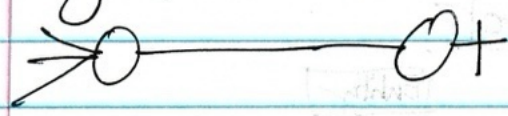


Example.

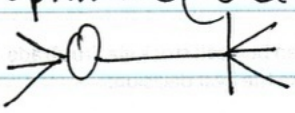
- One Mandatory (1 and only 1) to One Optional (0 or 1)

- One Mandatory (1 and 1) to Many Mandatory (1 or N)

- Many Optional (0 or N) to One Optional (0 or 1)



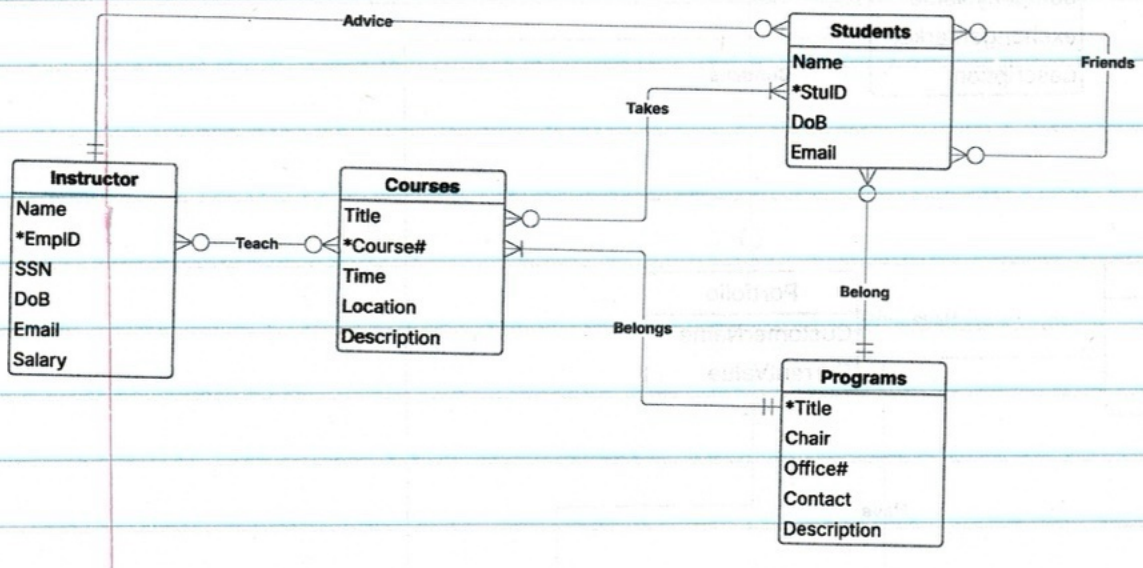
- Many optional (0&N) to Many Mandatory (1 or N)



Date – 5th Mar.

Following Entity Relationship model was drawn as part of training.

- An instructor may teach multiple courses, and a course might be taught by multiple instructors
- Each course must belong to one and only one online-program, and each program must have one ore more courses.
- A student may take muliple courses, and a course must have one or more students
- A student must belong to exactly one program, and a program may have one or more students
- A student may be friend of other students
- A student must have exactly one instructor as advisor, and one instructor may advice one or more students

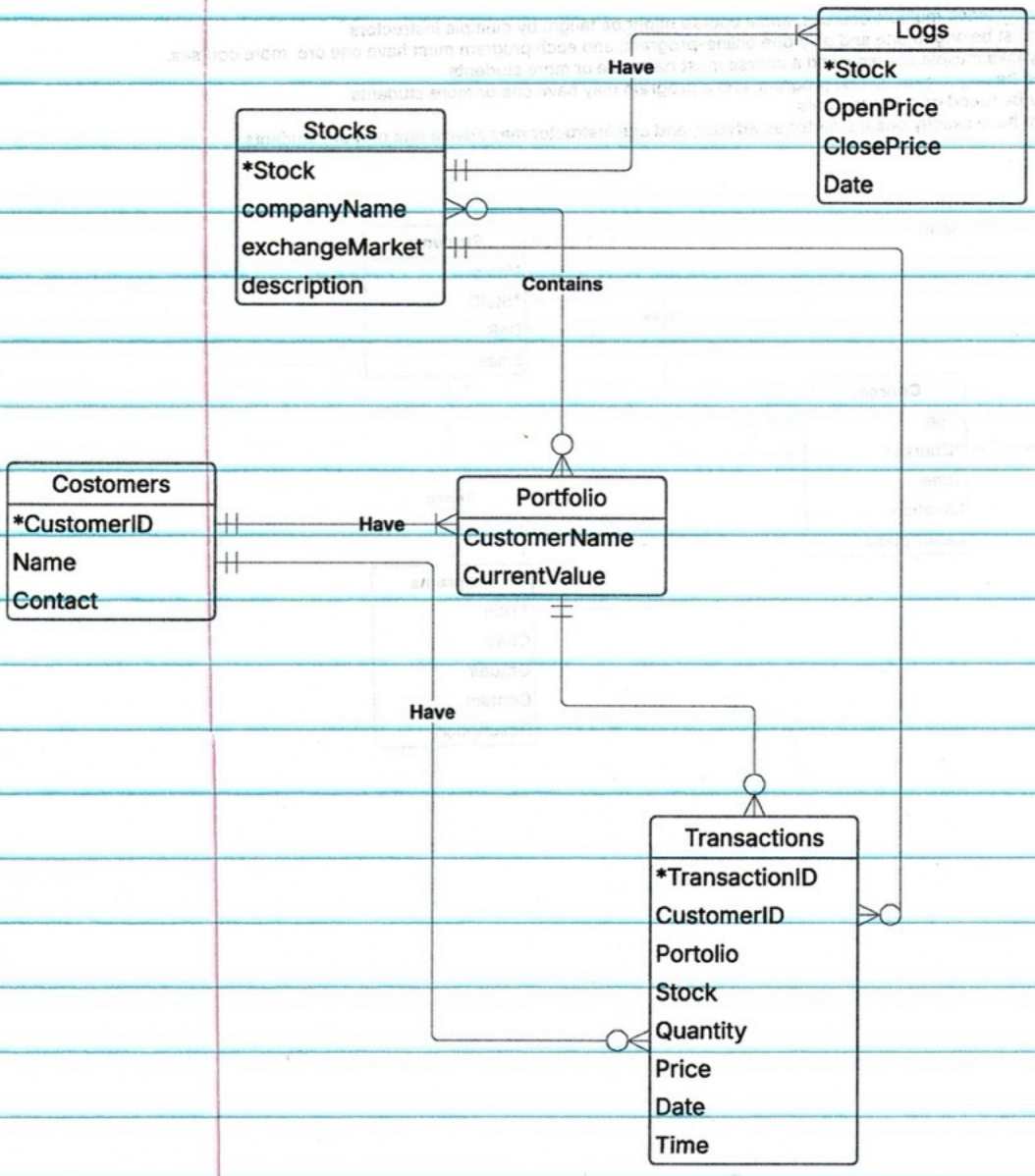


Your client is a stock-exchange company. The background story is as below:
TrueTrade provides excellent financial services to customers all over the world. Customers can buy/sell stock via TrueTrade easily and securely. TrueTrade also provides detailed information for every stock so customers can make the best decision.
The information needed for TrueTrade is:

- Stocks: including stock#, companyName, ExchangeMarket, Description
- Customers: including CustomerID, Name, Contact
- Portfolio: including CustomerName, currentValue
- Logs: including stock#, openPrice, closePrice, date
- Transactions: CustomerID, Portfolio, stock, quantity, price, date, time.

Relationships

- A customer must have one or more portfolios, and a portfolio must belong to one and only one customer.
- A portfolio may have one or more stocks, and a stock may belong to one or more portfolios.
- A stock must have one or more logs, and one log must belong to one and only one stock
- A customer may have one or more transactions, and each transaction must be done by one and only one customer
- A portfolio may be in one or more transactions, and each transaction must include one and only one portfolio
- A transaction must have a stock, and a stock may be in one or more transactions.



Date - 5th Mar Week 2

Convert ERD (Entity Relational Models) to Relational Models → implement into → Physical Database.

Relational Models.

- Organises data into two-dimensional tables: columns and rows.
- Relational Model includes:
  - Relations
  - Tuples
  - Attributes
  - keys, Foreign keys.
- Relational Models are represented by a set of Relational Schemas.
- example: stores

storeID	Street	City	Zip.
row1	- - -	- -	- -
row2	- - ..	- -	- -

→ Cardinality. # of Instances

← Degree → (No of columns)

Terminologies

- A Relation is a table with columns and rows.
- Attribute is a named column of a relation
- Domain is the set of allowable values for one or more attributes
- Tuple is a row of a relation.
- Degree is the number of attributes in a relation.
- Cardinality is the number of tuples in a relation.

- Relational Database is a collection of normalized relations with distinct relation names.

Domain → valid data of an attribute.
→ limit the value

- Name of a Relation is the name of the table.
- Tuple of a relation is a row in the table.
- The degree of relation is the number of attributes.

Relational keys

Superkey: An attribute or set of attributes that uniquely identifies a tuple within a relation.

Candidate key: Superkey (K) such that no proper subset is a superkey within the relation.

- In each tuple of R, values of K uniquely identify that tuple (uniqueness)
- No proper subset of K has the uniqueness property (irreducibility).

Primary key: Candidate key selected to identify tuples uniquely within relation.

Alternate key: Candidate keys that are not selected to be by primary key.

+ Foreign key: Connection of this relation with another. Attributes or set of attributes, within one relation that matches candidate key of some relation.

ProductDetails

Degree →

↓ Cardinality

SKU	Name	VendorID	Price	Date	Quantity	Location	Description
KB320	MouseMat	V1230	13.32	01/01/2019	3000	A3	Discontinued
YZ783	MouseMat	V3002	10.00	03/01/2019	1000	A3	Discontinued
IU990	Mouse	V3333	19.00	12/01/2019	700	A1	
IU370	Mouse	V3333	15.00	01/01/2020	500	A1	
YZ783	MouseMat	V3012	9.90	05/01/2020	1000	A3	
YZ783	MouseMat	V3012	8.90	05/01/2021	1000	A3	

What is the name of the relation: ProductDetails
What are the attributes of the relation:SKU, Name, VendorID, Price, Date, Quantity, Location, Description
What are the tuples of the relation (pick one): (IU990, Mouse, V3333, 19.00, 12/01/2019, 700, A1,,)
What is the degree of the relation: 8 No of attributes
What is the cardinality of the relation:6 No of tuples
What is the possible domain of attribute SKU: 5 chars, or 2 letters + 3 digits
What is the possible domain of attribute Name: chars
What is the possible domain of attribute VendorID: 5 chars, or 1 letter + 4 digits
What is the possible domain of attribute Price: float number or positive float number
What is the possible domain of attribute Date: Date
What is the possible domain of attribute Quantity: integers or positive integers
What is the possible domain of attribute Location: 2 chars, or 1 letter + 1 digits
What is the possible domain of attribute Description: chars
Choose some super keys for this relation:
SKU+Name+VendorID+Price+Date+Quantity+Location+Description;
SKU+Name+VendorID+Date
What might be candidate keys? SKU+VendorID+Price, SKU+VendorID+Date
What might be the primary key? SKU+VendorID+Date
What might be the alternate key? SKU+VendorID+Price

GradeBook

ID	Name	Major	Quiz1	Quiz2	Quiz3	Quiz4
001	Harry	Magic	90	90	80	100
002	Hermione	Magic	100	100	100	100
007	Ron	Magic	80	100	70	100
301	Dobby	Service	95	95	95	95
302	Severus	Education	90	90	99	100
399	Albus	Management	100	100	99	100

What is the name of the relation: GradeBook
What are the attributes of the relation: ID, Name, Major, Quiz1, Quiz2, Quiz3, Quiz4
What are the tuples of the relation (pick one): You can pick any row except the first (or the one with columns' names). For example:
What is the degree of the relation: 7
What is the cardinality of the relation: 6 (You should not answer 7, because the first row with columns' names is not a tuple)
What is the possible domain of attribute ID: 3 digits
What is the possible domain of attribute Name: some characters (you may specify less than 25 or a reasonable number of characters).
What is the possible domain of attribute Major: Most general domain will be some characters. You can also make a list of all majors, such as ["Magic", "Service", "Education", "Management"]
What is the possible domain of attribute Quiz1 - 4: some integer or float number, you may specify an interval such as [0, 100], or [0, 999] as long as it is reasonable.
Choose some super keys for this relation:
If there is no duplicate name of people:
What might be candidate keys? ID, Name
What might be the primary key? You can choose either one from above, such as Name.
What might be the alternate key? The one you didn't choose will be here, such as ID if you chose Name as the primary key.
If there is no duplicate name of people in the same major, what might be candidate keys?
What might be candidate keys? ID, Name, Major
What might be the primary key? You can choose either one from above, such as ID.
What might be the alternate key? The one you didn't choose will be here, such as Name, Major if you chose ID as the primary key.
If there are duplicate names of people in the system, what might be candidate keys?
What might be candidate keys? ID
What might be the primary key? ID
What might be the alternate key? NONE.

What is the main purpose of the relational model
- To convert entity relationship models into an intermediate step for implementation.
- To describe the structure of relations and how they are connected.

Relational Schema

- Defines a relation by a set of attributes and their domain.
- Relational Database schema.
- General format:
  - Name (Attribute$_1$, Attribute$_2$ .. Attribute(FK) .. AttributeN)
    - Attribute$_1$: key
    - Attribute(FK): foreign key.
- example: Stores (storeID, Street, City, Zip)
  Employees (EmpID, FirstName, LastName, DoB, Position, Departm storeID(FK))

Properties of Relations.

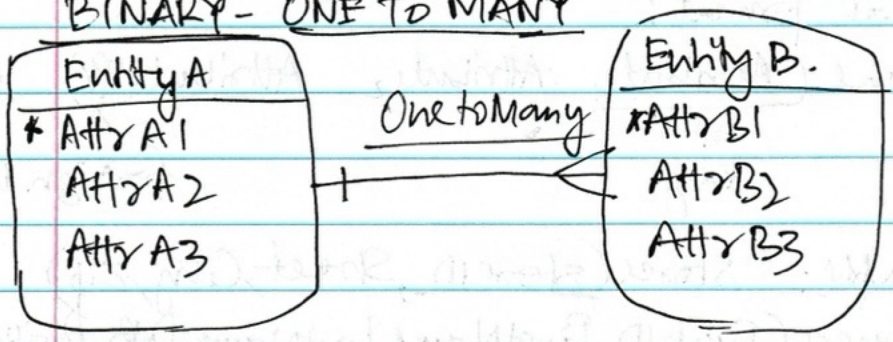
- Each tuple is distinct; there are no duplicate tuples
- Order of attributes (columns) have no significance.
- Order of tuples (rows) has no significance, in theory.
- Each cell of relation contains exactly one value.
- Each attribute has a distinct name.
- Value of an attribute are all from same domain.
- Relation name is distinct from all other relation in a Relational Model.

Converting ERADs to RMs

General Steps:

- Each entity will be converted directly into a relation
- The attributes of the entity become the attributes of the relation.
- The identifiers of the Entity becomes a key of the relation.
- Relationships will be mapped as foreign keys.

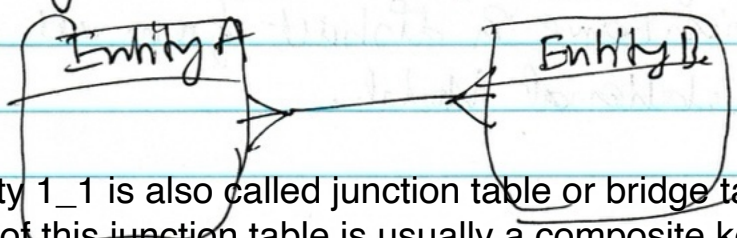
BINARY - ONE TO MANY



- Entity A (AttrA1, AttrA2, AttrA3)
- Entity B (AttrB1, AttrB2, AttrB3, AttrA1 (FK))

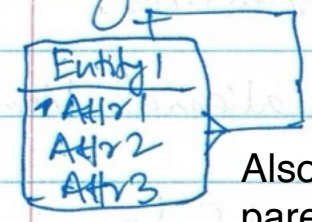
BINARY - MANY TO MANY

- Entity A (AttrA1, AttrA2, AttrA3)
- Entity B (AttrB1, AttrB2, AttrB3)
- Entity 1_2 (AttrA1 (fk), AttrB1 (fk))



Entity 1_1 is also called junction table or bridge table. The primary key of this junction table is usually a composite key consisting of both foreign key. alternately you can add auto-incrementing surrogate key (ID) to the junction table

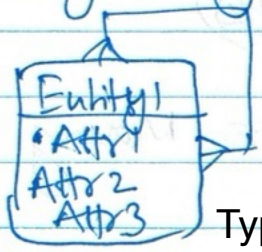
Unary, One to Many



Entity1(Attr1, Attr2, Attr3, Attr1B(fk))

Also called recursive relationship, normally hierarchical or parent-child relationship. Used to represent org structure, file directories, or family tree

Unary, Many to Many



Entity1(Attr1, Attr2, Attr3)
Entity1_1(Attr1A(fk), Attr1B(fk))

Typically happens in scenarios like representing relationships between people or hierarchical system

One to One

- Merge: we can merge two entities as one, and find a primary key for the new entity.

- Not Merge:
  - One to One can be treated as a special case of One to many.
  - If both sides are mandatory, we can choose either side as one, and other side as many.
  - If one one side is mandatory, we choose this side as one, and the other as many.
  - If both sides are optional, we have to rethink about primary key.

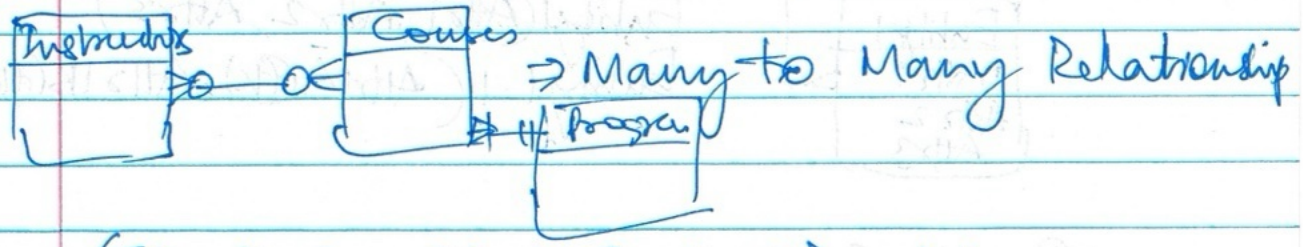
More than Binary

- If the relationship has more than 2 entities involved then we need to find a way to separate them to

Some binary/unary relationships.
- we cannot handle these relationships.

PRACTICE - Convert ERAD to a Relational Model.

Instructors(Name, EmpID, SSN, DoB, Email, Salary, Supervs.(fk))
Course(Title, Course#, Time, Location, Description, Title(fk))
Instructor_Course(EmpID(fk), Course#(fk))



⇒ Many-to-Many Relationship

Programs(Title, Chair, Office#, Contact, Description
(Add Title to Course as foreign key)

Students(Name, StuID, DoB, Email, Title(fk), EmpID(fk))

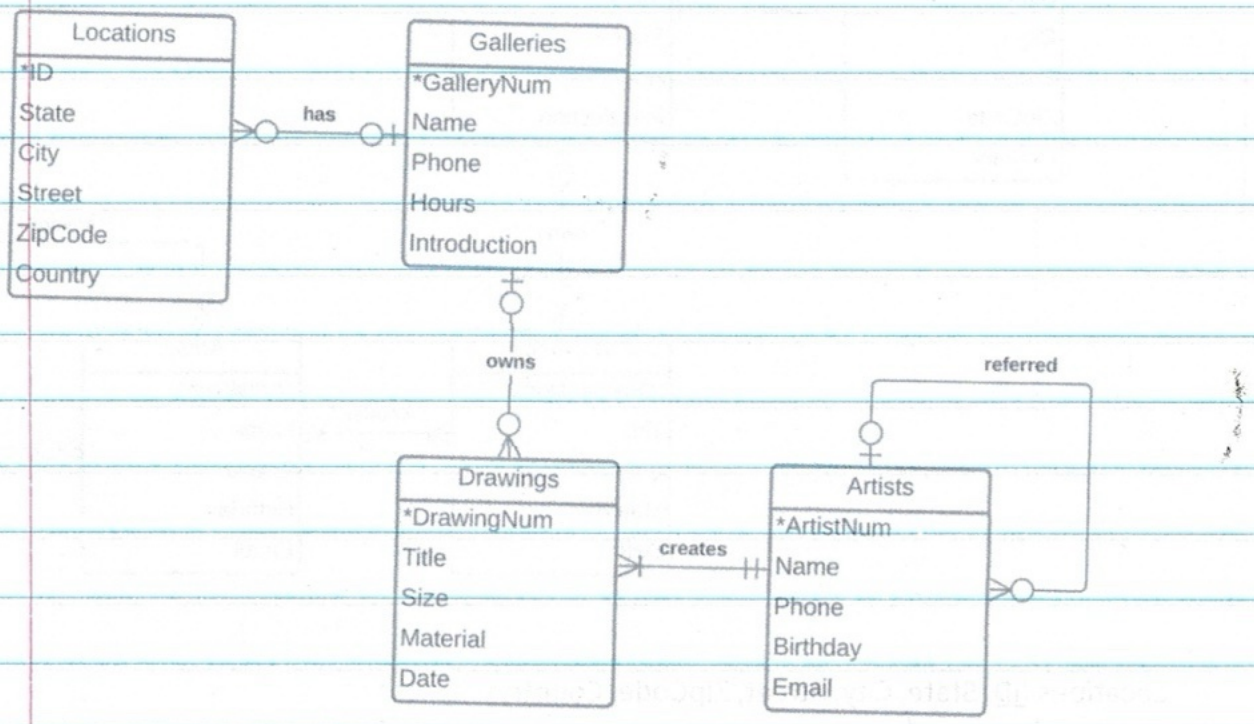
Students_Courses(StuID(fk), Course#(fk))

Students_students(StuID(fk), FriendID(fk))

ERD 2

Given the ERD below, convert it to a Relational Model (a set of Relational Schemas). Make sure you underline the primary key and add (fk) after the foreign key.

(It is a similar ERD than previous one - pay attention to the differences)



Locations (ID, State, City, Street, ZipCode, Country, GallaryNum(fk))

Galleries (GalleryNum, Name, Phone, Hours, Introduction)

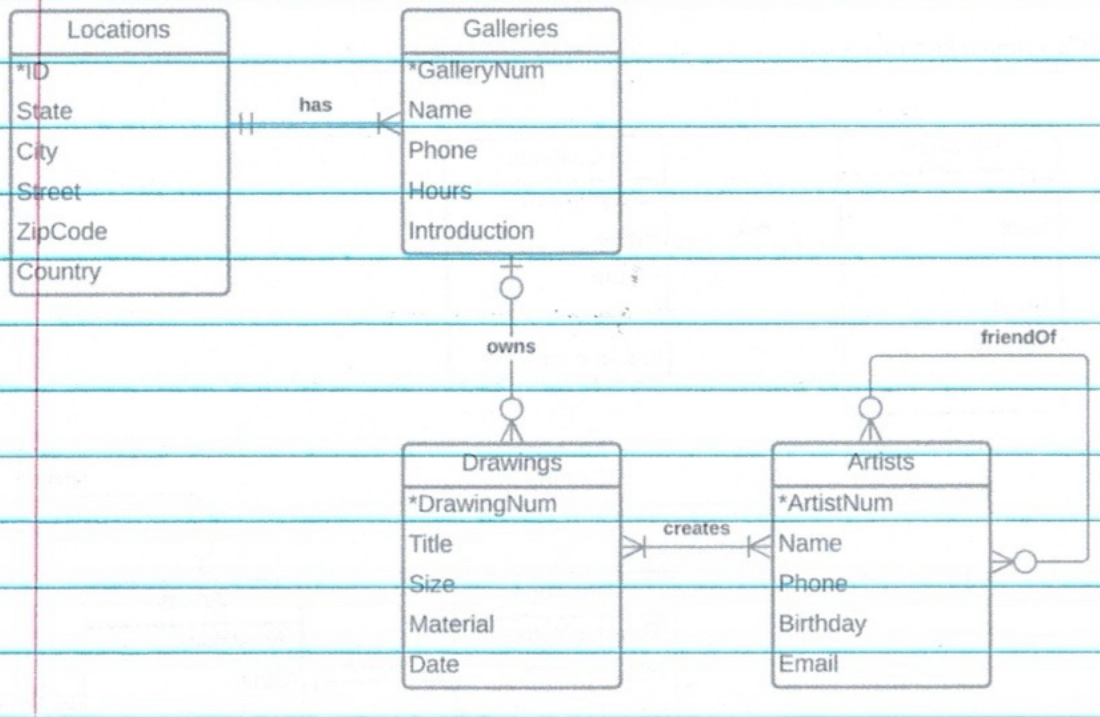
Drawings (DrawingNum, Title, Size, Material, Date, GalleryNum(fk), ArtistNum(fk))

Artists (ArtistNum, Name, Phone, Birthday, Email, ReferrerNum(fk))

22

ERD 1

Given the ERD below, convert it to a Relational Model (a set of Relational Schemas). Make sure you underline the primary key and add (fk) after the foreign key.



Locations (ID, State, City, Street, ZipCode, Country)

Galleries (GalleryNum, Name, Phone, Hours, Introduction, LocationID(fk))

Drawings (DrawingNum, Title, Size, Material, Date, GalleryNum(fk))

Artists (ArtistNum, Name, Phone, Birthday, Email)

Drawings_Artists(DrawingNum(fk), ArtistNum(fk))

Artists_Artists(Artist1Num(fk), Artist2Num(fk))

DATA REDUNDANCY AND NORMALIZATION

Normalization — Data Redundancy.

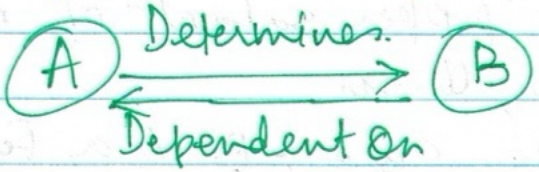
- Major aim of Relational Database Design is to group attributes into relations to minimize data redundancy.

Normalization Process

Normal Forms

Functional Dependency - describe the relationship among attributes in the same relation.

- If A determines B, then B is functional dependent on A.
- A → B is a deterministic relationship, and is a functional dependency.



Practice -

Gradebook.

ID	Name	Major	Quiz1	Quiz2	Quiz3	Quiz4	Total.
001	Harry	Magic	90	80	100	100	370
002	Ron	Swize	95	95	95	95	380

① ID → Name, Major, Quiz1, Quiz2. ... Grade.

② Quiz1, 2, 3, 4 → Total.

③ Total → Grade.

Grade
A-
A

ProductDetails

SKU	Name	VendorID	Price	Date	Quantity	Location	Descriptn on Discounts
KB320	Mat	V1234	13.95	01/01/2009	350	A3	

(1) SKU → Name

(2) SKU, VendorID, Date → Price, Quantity, Location, Descriptn

Special Types of FDs

- Full/Partial FD (Functional Dependencies)
- Transitive FDs.

Full/Partial FD.

- In one relation, if a set of attributes $A: (a_1, a_2 \ldots a_n)$ determines attribute B:

  $A: (a_1, a_2 \ldots a_n) \longrightarrow B$.

- If there is no proper subset of A also determines B, then

  $A: (a_1, a_2 \ldots a_n) \longrightarrow B$ is a Full FD. It is full because we need all attributes in A to determine B.

- If there is a proper subset of A determines B, then $A: (a_1, a_2 \ldots a_n) \longrightarrow B$ is a partial functional dependency.
- To achieve full FD, we need to minimize A by removing unnecessary attributes.

Transitive FDs

- In one relation, if an attribute A determines attribute B, and B determines attribute C
  - $A \rightarrow B$
    $B \rightarrow C$

  we say $A \rightarrow C$ is Transitive Functional Dependency

Examples: (1)

Is there a partial functional dependency? Is there a transitive functional dependency?

Assignments (A,B,C,D,E,F,G,H,I,J,K,L,N,Q)

fd1: A,B,C → D,E,F,G,H,I,J,K,L,N,Q

No partial or transitive fd

Example (2)

Assignments (A B C D E F G H I J K L N Q)

fd1: A,B,C → D E F G H I J K L N Q

fd2: D → L,N,Q.

fd3: L → N.

No partial

transitive – A,B,C → D

D → L, L → N

Example (3)

Assignments (A,B,C,D E F G H I J K L N Q)

fd1: A,B,C → D E F G H I J K L N Q.

fd2: A → D,E,F,L,N,Q.

fd3: B,C → G,H.

fd4: D → L,N,Q.

fd5: L → N.

partial.

transitive

A,B,C → D, D → L, L → N

Example 4:

CustomerContact (FirstName, Lastname, DoB, Email, Zodiac, Street, Country, City, State, Zip, Company, CompanyInfo)

fd1: FirstName, LastName, DoB → Email, Zodiac, Street, Country, City, State, Zip, Company, CompanyInfo)

fd2: Lastname, DoB → Email. → partial.

fd3: DoB → Zodiac — partial

fd4: Zip → City, State — transitive

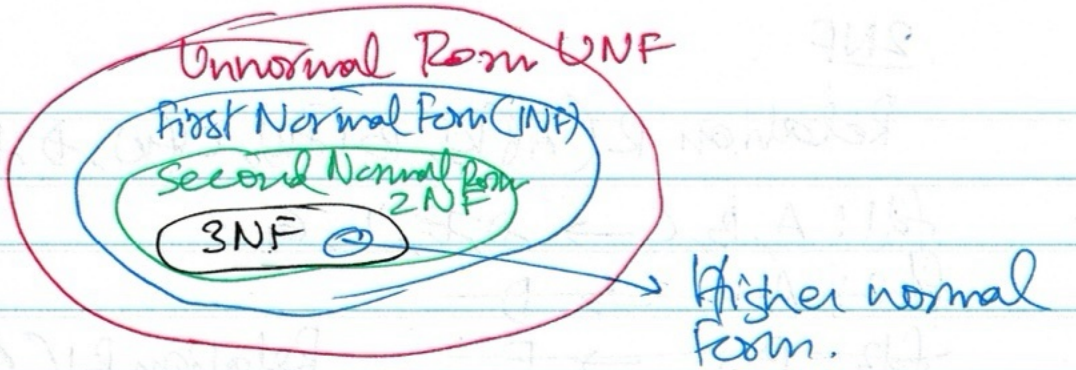
fd5: Company → CompanyInfo.

transitive.

partial →

Normalization

- Normalization Process is converting a relation from less restricted form to more restricted form.
- The level of restriction is called Normal Form.



UNF → table contains repeated rows/values in a cell.

1NF → (1) No repeating value in a cell. One and only one value.

2NF → (1) It must be in 1NF

(2) Every non-primary key attribute is fully functionally dependent on the primary key.

OR

If there is any non-primary key attribute is partially functionally dependent on the primary key, a relation is not in 2NF.

- How to convert 1NF to 2NF
  - eliminate Partial Dependency.
  - Determinants will be primary keys in new relations; will be foreign keys in original relations.

1NF to 2NF

- Relation R (A, B, C, D, E, F, G) 1NF

fd1: A, B, C → D, E, F, G

fd2: A → D

fd3: B, C → E

fd4: F → G.

2NF

Relation R(A(pk), B(pk), C(pk), ~~D, E~~, F, G)

fd1: A, B, C → ~~D, E~~, F, G

~~fd2: A → D~~

~~fd3: B, C → E~~

fd4: F → G

Relation R1(A, D)

fd1: A → D

Relation R2 (B, C, E)

fd: B, C → E

3NF Third Normal Form

(1) Relation is in 2NF

(2) NO non-primary key attribute is transitively dependent on the primary key.

Relation R(A, B, C, F, G)

fd1: A, B, C → F, G

fd2: F → G.

Not in 3NF

Why? Because of (fd2)

2NF ⇓ 3NF

Relation R(A, B, C, F(fk))

fd1: A, B, C → ~~F G~~

Relation R1 (F, G)

fd1: F → G.

Relation 1 – 3NF *(no partial FD or no transitive FD)*

Which normal form is the relational model below?

Assignments(A, B, C, D, E, F, G, H, I, J, K, L, N, Q)
fd1: A, B, C -> D, E, F, G, H, I, J, K, L, N, Q

Relation 2 – 2NF *(no partial FD but transitive FD)*

Which normal form is the relational model below?

Assignments(A, B, C, D, E, F, G, H, I, J, K, L, N, Q)
fd1: A, B, C -> D, E, F, G, H, I, J, K, L, N, Q
fd2: D -> L, N, Q
fd3: L -> N

Relation 3 – 1NF *(partial FD)*

Which normal form is the relational model below?

Assignments(A, B, C, D, E, F, G, H, I, J, K, L, N, Q)
fd1: A, B, C -> D, E, F, G, H, I, J, K, L, N, Q
fd2: A -> D, E, F, L, N, Q
fd3: B, C -> G, H
fd4: D -> L, N, Q
fd5: L -> N

Relation 4 – 1NF *(partial FD)*

Which normal form is the relational model below?

CustomerContact (FirstName, LastName, DoB, Email, Zodiac, Street, County, City, State, Zip, Company, CompanyInfo)
fd1: FirstName, LastName, DoB -> Email, Zodiac, Street, County, City, State, Zip, Company, CompanyInfo
fd2: LastName, DoB → Email
fd3: DoB → Zodiac
fd4: Zip → City, State
fd5: Company → CompanyInfo

Normalization Process:

Relational Model 1

Given the relational model below, normalize it to 3NF. Make sure you list the Functional Dependencies along with each relation and include all relations (each relation with its functional dependencies) in the summary.

Customers (CID, FirstName, LastName, Email, Address, Company, CompanyInfo)
FD1: CID → FirstName, LastName, Email, Address, Company, CompanyInfo
FD2: Company → CompanyInfo

Answer:

Customers (CID, FirstName, LastName, Email, Address, Company (fk), ~~CompanyInfo~~)
FD1: CID → FirstName, LastName, Email, Address, Company, ~~CompanyInfo~~

CompanyInfo (Company, CompanyInfo)
FD1: Company → CompanyInfo

Relational Model 2

Given the relational model below, normalize it to 3NF. Make sure you list the Functional Dependencies along with each relation, and include all relations (each relation with its functional dependencies) in the summary.

CustomerContact (FirstName, LastName, DoB, Email, Zodiac, Street, County, City, State, Zip, Company, CompanyInfo)
fd1: FirstName, LastName, DoB → Email, Zodiac, Street, County, City, State, Zip, Company, CompanyInfo
fd2: LastName, DoB → Email
fd3: DoB → Zodiac
fd4: Zip → City, State
fd5: Company → CompanyInfo

Answer:

CustomerContact (FirstName, LastName (fk), DoB (fk), ~~Email~~, ~~Zodiac~~, Street, County, ~~City, State~~, Zip (fk), Company (fk), ~~CompanyInfo~~)

fd1: FirstName, LastName, DoB → Email, Zodiac, Street, County, City, State, Zip, Company, CompanyInfo

~~fd2: LastName, DoB → Email~~

~~fd3: DoB → Zodiac~~

~~fd4: Zip → City, State~~

~~fd5: Company → CompanyInfo~~

LastNameDob (LastName, DoB, Email)
fd1: LastName, DoB → Email

zipCityState (zip, City, State)
fd1: zip → City, State

DobZodiac (DoB, Zodiac)
fd1: DoB → Zodiac

CompanyInfo (Company, CompanyInfo)
FD1: Company → CompanyInfo

Question 1: Normal Forms

1A: What normal form is the following relation in (key is underlined), and Why? Make sure you provide both Answer and Reasoning:

- STORE_ITEM (SKU, EventID, Vendor, Style, Price, Warranty)
    - FD1: SKU, EventID → Vendor, Style, Price
    - FD2: SKU → Vendor, Style, Warranty
    - FD3: Vendor → Warranty

Answer: 1NF, Since it is partially functionally dependent (fd2) and transitively dependent (fd2, fd3)

1B: What normal form is the following relation in (key is underlined), and Why? Make sure you provide both Answer and Reasoning:

- SKU (SKU, Vendor, Style, Warranty)
- FD1: SKU → Vendor, Style, Warranty
- FD2: Vendor → Warranty

Answer: 2NF - Since it is fully functionally dependent (fd1) and not 3NF as it is transitively dependent (fd1, fd2)

1C: What normal form is the following relation in (key is underlined), and Why? Make sure you provide both Answer and Reasoning:

- SKU (SKU, Vendor, Style, Warranty)
- FD1: SKU → Vendor, Style, Warranty
- FD2: Vendor, Style → Warranty

Answer: 2NF - Since it is fully functionally dependent (fd1) and not 3NF as it is transitively dependent (fd1, fd2)

Question 2: Normalization Process

We have a relational model represented as a relational schema and its functional dependencies given as below:

- TRANSCRIPT (ID, fName, lName, major, majorDescription, courseID, courseDescription, courseGrade)
- FD1: ID, courseID -> fName, lName, major, majorDescription, courseDescription, courseGrade
- FD2: ID -> fName, lName, major, majorDescription
- FD3: courseID - > curseDescription
- FD4: major -> majorDescription

2A: Normalize it to 2NF

Answer:

- TRANSCRIPT (ID (fk), ~~fName~~, ~~lName~~, ~~major~~, ~~majorDescription~~, courseID (fk), ~~courseDescription~~, courseGrade)
    - FD1: ID, courseID -> fName, lName, major, majorDescription, courseDescription, courseGrade
    - ~~FD2: ID -> fName, lName, major, majorDescription~~
    - ~~FD3: courseID - > curseDescription~~
    - ~~FD4: major -> majorDescription~~
- Course (CourseID, courseDescription)
    - FD3: courseID - > curseDescription
- Major (Major, MajorDescription)
    - FD4: major -> majorDescription
- Student (ID, fName, lName, major (fk))
    - FD2: ID -> fName, lName, major

Week 4

Saturday, March 1, 2025 2:13 PM

In database design, "cardinality" and "participation" are important concepts related to relationships between entities in a database schema.

Cardinality: Cardinality describes the number of instances of one entity that can correspond to instances of another entity in a relationship. In simpler terms, it shows how many instances of one table can be linked to instances in another table. There are three main types of cardinality:

One-to-One (1:1): Each row in the first table corresponds to at most one row in the second table, and vice versa. For example, a "NationalID" table and a "DriverLicense" table might have a 1:1 relationship, where each national ID number corresponds to exactly one driver license record, and each driver license record corresponds to exactly one national ID number.

One-to-Many (1:N): Each row in the first table can correspond to one or more rows in the second table, but each row in the second table corresponds to only one row in the first table. For example, a "Department" table and an "Employee" table might have a 1:N relationship, where each department can have multiple employees, but each employee belongs to only one department.

Many-to-Many (M:N): Each row in either of the tables can correspond to one or more rows in the other table. This type of relationship requires a junction or linking table to resolve it. For example, a "Student" table and a "Course" table might have a M:N relationship, as each student can enroll in multiple courses, and each course can have multiple students. A "Enrollment" table would be used to resolve this relationship, with foreign keys to both the Student and Course tables.

Participation: Participation in a relationship describes whether a particular entity must participate (i.e., have a corresponding record) in the relationship. There are two types of participation:

Total Participation: Every instance of an entity must participate in the relationship. In our examples, the "Department" table has total participation in its relationship with the "Employee" table, because every department must have at least one employee.

Partial Participation: An instance of an entity is not required to participate in the relationship. In our examples, the "NationalID" table has partial participation in its potential relationship with the "DriverLicense" table, because not every national ID number necessarily has a corresponding driver license entry.

Understanding these concepts is essential for designing well-structured and efficient database schemas, as they help ensure data integrity and facilitate data retrieval and manipulation.

Week4 - Assignment

PetsCare - is a local clinic that takes care of pets. PetsCare has served the community for more than 20 years, and has been successful in the business.
- including pet wellness and vaccination,
- medical services,
- surgery including spay and neuter,
- dental cleaning and treatment,
- grooming, etc.

Recently, the current owner, Claire, who is the daughter of the founder of PetsCare, realized that the old file system solution is very inefficient. She and her team have to spend lots of time taking records, querying for information, and maintaining the data. She decided to turn to you, for a Relational Database Management System.

PetsCare wants to record information about
- customers,
- pets,
- staff,
- visits.

Claire will be available to meet via Zoom tomorrow to discuss details. You should prepare questions for building the DBMS.

Step1: Questions
- What information do you record about customers
  - Do customers bring multiple pets
- What information do you record about pets
- What information do you record about staff
- What information do you record about visits
- Do you have any
  - Vendors for equipment, services, etc
  - Suppliers

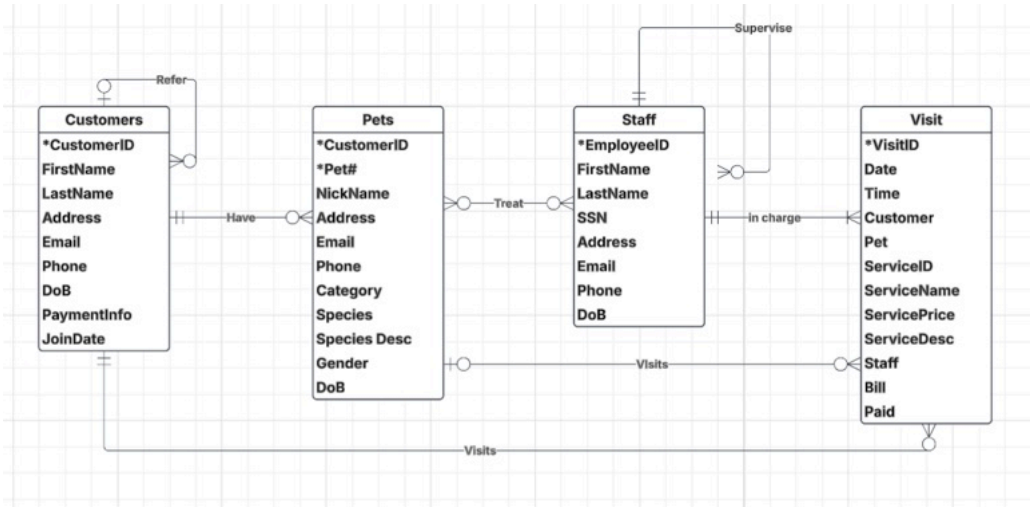
Details about Entities / Relations:
1. Customers: CustomerID, FirstName, LastName, Address, Email, Phone, DoB, PaymentInfo, JoinDate. CustomerID is the identifier.
2. Pets: CustomerID, Pet#, NickName, Address, Email, Phone, Category, Species/Breed, Species/Breed Description, Gender, DoB, Notes. CustomerID and Pet# together as the identifier.
3. Staff: EmployeeID, FirstName, LastName, SSN, Address, Email, Phone, DoB. EmployeeID is the identifier.
4. Visit: VisitID, Date, Time, Customer, Pet, ServiceID, ServiceName, ServicePrice, ServiceDescription, Staff, Bill, Paid. VisitID is the identifier

identifier.

Their relationships are:

1. A customer may have one or more (zero or more) pets; A pet must belong to one and only one (exactly one) customer.
2. A staff may treat one or more pets (zero or more); and A pet may be familiar with one or more (zero or more) staff.
3. A customer may have one or more (zero or more) visits; and each visit must be done by one and only one (exactly one) customer.
4. A pet may have one or more (zero or more) visits; and each visit may involve one (zero or one) pet.
5. A staff must be in charge of one or more (one or more) visits; and a visit must be charged by one and only one (exactly one) staff.
6. A customer may be referred by one (zero or one) customer; and a customer may refer one or more (one or more) customers.
7. A staff must have one and only one (exactly one) supervisor; A staff may supervise one or more (zero or more) staff.

ERD:



Convert the ERD into the Relational Model

Customers (CustomerID, FirstName, LastName, Address, Email, Phone, DoB, PaymentInfo, JoinDate, ReferredByCustomerID (fk))

Pets (CustomerID(fk), Pet#, NickName, Address, Email, Phone, Category, Species, SpeciesDesc, Gender, DoB)

Staff (EmployeeID, FirstName, LastName, SSN, Address, Email, Phone, DoB, SupervisorID(fk))

Visit (VisitID, Date, Time, CustomerID(fk), Pet#(fk), ServiceID, ServiceName, ServicePrice, ServiceDesc, StaffID(fk), Bill, Paid)

Pets_Staff (CustomerID(fk), Pet#(fk), EmployeeID(fk)) Why this one at this stage?

Normalize the Relational Model to 3NF

Customers (CustomerID, FirstName, LastName, Address, Email, Phone, DoB, PaymentInfo, JoinDate, ReferredByCustomerID(fk)).

- **FD1: CustomerID → FirstName, LastName, Address, Email, Phone, DoB, PaymentInfo, JoinDate, ReferredByCustomerID**

This will remain as same for now

- 1NF - yes as no cell has multiple values
- 2NF - yes as no partial functional dependencies
- 3NF - yes as no transitive dependencies

Pets (CustomerID(fk), Pet#, NickName, Address, Email, Phone, Category, Species/Breed, Species/Breed Description, Gender, DoB, Notes).

FD1: CustomerID, Pet# → NickName, Address, Email, Phone, Category, Species/Breed, Species/Breed Description, Gender, DoB, Notes.
FD2: CustomerID → Address, Email, Phone
FD3: Species/Breed → Species/Breed Description

Pets (CustomerID(fk), Pet#, NickName, ~~Address, Email, Phone~~, Category, Species/BreedID (fk), ~~Species/Breed Description~~, Gender, DoB, Notes).

FD1: CustomerID, Pet# → NickName, Address, Email, Phone, Category, Species/Breed, Species/Breed Description, Gender, DoB, Notes.

~~**FD2: CustomerID → Address, Email, Phone**~~

~~**FD3: Species/Breed → Species/Breed Description**~~

Customer_1 (CustomerID, Address, Email, Phone)
FD2: CustomerID → Address, Email, Phone

Species/Breed_Desc (Species/BreedID, Species_Desc)
FD3: Species/Breed → Species/Breed Description

Staff (EmployeeID, FirstName, LastName, SSN, Address, Email, Phone, DoB, SupervisorID(fk)).
FD1: EmployeeID → FirstName, LastName, SSN, Address, Email, Phone, DoB, SupervisorID

Visit (VisitID, Date, Time, CustomerID(fk), Pet(fk), ServiceID, ServiceName, ServicePrice, ServiceDescription, EmployeeID(fk), Bill, Paid).
FD1: VisitID → Date, Time, CustomerID, Pet, ServiceID, ServiceName, ServicePrice, ServiceDescription, EmployeeID, Bill, Paid
FD2: ServiceID → ServiceName, ServicePrice, ServiceDescription.

Visit (VisitID, Date, Time, CustomerID(fk), Pet(fk), ServiceID (fk), ~~ServiceName, ServicePrice, ServiceDescription~~, EmployeeID(fk), Bill, Paid).
FD1: VisitID → Date, Time, CustomerID, Pet, ServiceID, ~~ServiceName, ServicePrice, ServiceDescription~~, EmployeeID, Bill, Paid
~~FD2: ServiceID → ServiceName, ServicePrice, ServiceDescription.~~

Service_Details (ServiceID, ServiceName, ServicePrice, ServiceDescription)

Pets_Staff(CustomerID(fk), Pet#(fk),EmployeeID(fk))
There is no non-primary-key attribute.