



# Data Modeling

Data Boot Camp  
Lesson 9.3



# Class Objectives

---

By the end of today's class, you will:



Apply data modeling techniques to database design.



Normalize data.

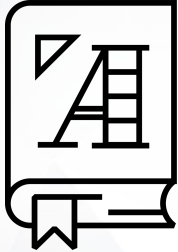


Identify data relationships.



Create visual representations of a database through entity relationship diagrams.

# Data Normalization



**Data normalization** is the process of restructuring data to a set of defined "normal forms."



The process of data normalization **eliminates data redundancy and inconsistencies.**

# Data Normalization

---



Process of restructuring data to a set of “normal forms.”



**Reduce and eliminate data redundancy** and inconsistencies



Three most common forms:



First normal form (1NF)



Second normal form (2NF)



Third normal form (3NF)



There are even more levels!

# First Normal Form (1NF)

---



Each field in a table row should contain a single value



Each row is unique

– **No repeating groups**

Student no	Student Name	Course No	Course Name	Grade
1	Chris	201, 210	Stats I, Python II	B, A
2	Abby	201, 203, 210	Stats I, Database I, Python II	A, A, B

**Raw Data**

# First Normal Form (1NF)



Each field in a table row should contain a single value



Each row is unique

– **No repeating groups**

Student no	Student Name	Course No	Course Name	Grade
1	Chris	201, 210	Stats I, Python II	B, A
2	Abby	201, 203, 210	Stats I, Database I, Python II	A, A, B

Repeating Groups

**Raw Data**



# First Normal Form (1NF)



Each field in a table row should contain a single value



Each row is unique

– No repeating groups

Student no	Student Name	Course No	Course Name	Grade
1	Chris	201, 210	Stats I, Python II	B, A
2	Abby	201, 203, 210	Stats I, Database I, Python II	A, A, B



Student no	Student Name	Course No	Course Name	Grade
1	Chris	201	Stats I	B
1	Chris	210	Python II	A
2	Abby	201	Stats I	A
2	Abby	203	Database I	A
2	Abby	210	Python II	B

Raw Data

*Normalize (Remove repeating groups)*

First Normal Form

# Second Normal Form (2NF)

Be in First Normal Form

Single Column Primary Key

- Each table has a Primary Key (PK)
- PK uniquely identifies every row
- Every non-key attribute is fully dependent on the primary key

Generally there could be a need to create a new table

Student no	Student Name	Course No	Course Name	Grade
1	Chris	201	Stats I	B
1	Chris	210	Python II	A
2	Abby	201	Stats I	A
2	Abby	203	Database I	A
2	Abby	210	Python II	B



Student no	Student Name
1	Chris
2	Abby

Student no	Course No	Course Name	Grade
1	201	Stats I	B
1	210	Python II	A
2	201	Stats I	A
2	203	Database I	A
2	210	Python II	B

# Second Normal Form (2NF)

Be in First Normal Form

Single Column Primary Key

- Each table has a Primary Key (PK)
- PK uniquely identifies every row
- **Every non-key attribute is fully dependent on the primary key**

Generally there could be a need to create a new table

Student no	Student Name	Course No	Course Name	Grade
1	Chris	201	Stats I	B
1	Chris	210	Python II	A
2	Abby	201	Stats I	A
2	Abby	203	Database I	A
2	Abby	210	Python II	B



Student no	Student Name
1	Chris
2	Abby

Student no	Course No	Course Name	Grade
1	201	Stats I	B
1	210	Python II	A
2	201	Stats I	A
2	203	Database I	A
2	210	Python II	B

*both Student No. and Course No.* (green arrow from Student no to Course No)

*Course No. only* (blue arrow from Course No to Course Name)

# Second Normal Form (2NF)

Be in First Normal Form

Single Column Primary Key

- Each table has a Primary Key (PK)
- PK uniquely identifies every row
- **Every non-key attribute is fully dependent on the primary key**

Generally there could be a need to create a new table

Student no	Student Name	Course No	Course Name	Grade
1	Chris	201	Stats I	B
1	Chris	210	Python II	A
2	Abby	201	Stats I	A
2	Abby	203	Database I	A
2	Abby	210	Python II	B

1NF Data



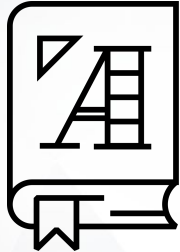
Student no	Student Name
1	Chris
2	Abby

Course No	Course Name
201	Stats I
203	Database I
210	Python II

*Normalize*

Student no	Course No	Grade
1	201	B
1	210	A
2	201	A
2	203	A
2	210	B

2NF Data



**Transitive Dependence** is the reliance or dependence of a column's value on another column through a third column.


# Transitive Dependency

---




## Transitive

- If  $X \rightarrow Y$  and  $Y \rightarrow Z$  then  $X \rightarrow Z$ .



## Dependence

- One value relies on another.
- City relies on ZIP code; age depends on birthday.



## For example:

- Say you have three columns: *CourseNo*, *InstructorNo*, *InstructorName*.
- *InstructorNo* and *InstructorName* rely on the the *CourseNo*.
- *InstructorName* also relies on the *InstructorNo*.
- So *InstructorName* relies on the *CourseNo* via the *InstructorNo*

# Third Normal Form (3NF)



**Must be in Second Normal Form**



Contain non-transitively dependent columns

Student no	Student Name	Course No	Course Name	Grade	Instructor No	Instructor Name
1	Chris	201, 210	Stats I, Python II	B, A	3	Susan
2	Abby	201, 203, 210	Stats I, Database I, Python II	A, A, B	5	Phil

# Third Normal Form (3NF) - *Convert Raw to 1NF*



**Must be in Second Normal Form**



Contain non-transitively dependent columns

Student no	Student Name	Course No	Course Name	Grade	Instructor No	Instructor Name
1	Chris	201, 210	Stats I, Python II	B, A	3, 5	Susan, Phil
2	Abby	201, 203, 210	Stats I, Database I, Python II	A, A, B	3, 7, 5	Susan, Mateo, Phil

**Raw Data**

Student no	Student Name	Course No	Course Name	Grade	Instructor No	Instructor Name
1	Chris	201	Stats I	B	3	Susan
1	Chris	210	Python II	A	5	Phil
2	Abby	201	Stats I	A	3	Susan
2	Abby	203	Database I	A	7	Mateo
2	Abby	210	Python II	B	5	Phil

**1NF**





# Third Normal Form (3NF) - *Convert 1NF to 2NF*



**Must be in Second Normal Form**



Contain non-transitively dependent columns

Student no	Student Name	Course No	Course Name	Grade	Instructor No	Instructor Name
1	Chris	201	Stats I	B	3	Susan
1	Chris	210	Python II	A	5	Phil
2	Abby	201	Stats I	A	3	Susan
2	Abby	203	Database I	A	7	Mateo
2	Abby	210	Python II	B	5	Phil

**1 NF**

Student no	Student Name
1	Chris
2	Abby

Student no	Course No	Grade
1	201	B
1	210	A
2	201	A
2	203	A
2	210	B

Course No	Course Name	Instructor No	Instructor Name
201	Stats I	3	Susan
203	Database I	7	Mateo
210	Python II	5	Phil

**2 NF**

# Third Normal Form (3NF) - *Remove Transitive Dependents*



Must be in Second Normal Form



**Contain non-transitively dependent columns**

Student no	Student Name
1	Chris
2	Abby

Student no	Course No	Grade
1	201	B
1	210	A
2	201	A
2	203	A
2	210	B

Course No	Course Name	Instructor No	Instructor Name
201	Stats I	3	Susan
203	Database I	7	Mateo
210	Python II	5	Phil

**Determinant → Dependent**

Course No → Instructor No

Instructor No → Instructor Name

Course No → Instructor Name

# Third Normal Form (3NF) - Convert 1NF to 2NF



Must be in Second Normal Form



Contain non-transitively dependent columns

Student no	Student Name
1	Chris
2	Abby

Student no	Course No	Grade
1	201	B
1	210	A
2	201	A
2	203	A
2	210	B

Course No	Course Name	Instructor No	Instructor Name
201	Stats I	3	Susan
203	Database I	7	Mateo
210	Python II	5	Phil

**Determinant → Dependent**

Course No → Instructor No



Instructor No → Instructor Name

Course No → Instructor Name **Transitive Dependency**

# Third Normal Form (3NF) - Convert 1NF to 2NF



Must be in Second Normal Form



Contain non-transitively dependent columns

Student no	Student Name
1	Chris
2	Abby

Student no	Course No	Grade
1	201	B
1	210	A
2	201	A
2	203	A
2	210	B

Student no	Student Name
1	Chris
2	Abby

Student no	Course No	Grade
1	201	B
1	210	A
2	201	A
2	203	A
2	210	B

Course No	Course Name	Instructor No	Instructor Name
201	Stats I	3	Susan
203	Database I	7	Mateo
210	Python II	5	Phil

Course No	Course Name	Instructor No
201	Stats I	3
203	Database I	7
210	Python II	5

Instructor No	Instructor Name
3	Susan
7	Mateo
5	Phil

2 NF

3NF





## **Activity:** Pet Normalizer

In this activity, you will practice data normalization skills using the provided data.

(Instructions sent via Slack)

**Suggested Time:**  
20 Minutes





**Time's Up!** Let's Review.

# Foreign Keys

# Foreign Keys

Foreign Keys reference the primary key of another table.

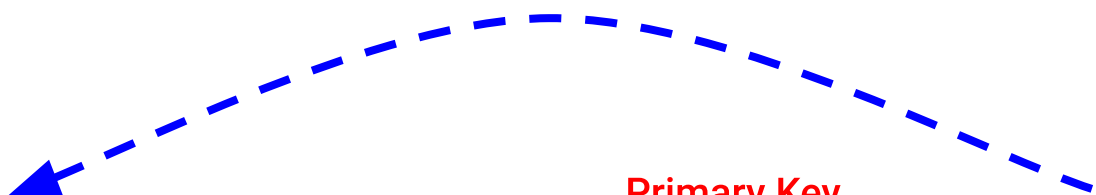


Can have a different name



Do not need to be unique

Primary Key



id	name
1	Sally
2	Charles

owner table

Primary Key

id	owner_id	name
1	1	Zeus
2	2	Kevin
8	1	Fido

Foreign Key

pet table





## **Activity:** Foreign Keys

In this activity, you will create tables with foreign keys.

(Instructions sent via Slack)

**Suggested Time:**  
15 Minutes





**Time's Up!** Let's Review.

# Data Relationships

# Data Relationships

---

01

**One-to-One**

02

**One-to-Many**

03

**Many-to-Many**

# One-to-One Relationship

ID	Name	Social Security
1	Homer	111111111
2	Marge	222222222
3	Lisa	333333333
4	Bart	444444444
5	Maggie	555555555



Each item in one column is linked to only one other item from the other column.



Here, each person in the Simpsons family can have only one social security number.



Each social security number can be assigned only to one person.

# One-to-Many Relationship

ID	Address
11	742 Evergreen Terrace
12	221B Baker Street

ID	Name	Social Security	AddressID
1	Homer	1111111111	11
2	Marge	2222222222	11
3	Lisa	3333333333	11
4	Bart	4444444444	11
5	Maggie	5555555555	11
6	Sherlock	1122334455	12
7	Watson	2233445566	12



The two tables, joined, would look like the figure on the next slide.



Each person has an address.



Each address can be associated with multiple people.

# One-to-Many Relationship

address s.ID	Address	pers on.ID	Name	Social Security	AddressID
11	742 Evergreen Terrace	1	Homer	111111111	11
11	742 Evergreen Terrace	2	Marge	222222222	11
11	742 Evergreen Terrace	3	Lisa	333333333	11
11	742 Evergreen Terrace	4	Bart	444444444	11
11	742 Evergreen Terrace	5	Maggie	555555555	11
12	221B Baker Street	6	Sherlock	112233445	12
12	221B Baker Street	7	Watson	223344556	12



The two tables in the previous slide, if joined, would look like this.



Each person has an address.



Each address can be associated with multiple people.

# Many-to-Many Relationship

---

ID	Child
1	Bart
2	Lisa
3	Maggie

ID	Parent
11	Homer
12	Marge



Each child can have more than one parent.



Each parent can have more than one child.



# Many-to-Many Relationship

---

ChildID	Child	ParentID	Parent
1	Bart	11	Homer
1	Bart	12	Marge
2	Lisa	11	Homer
2	Lisa	12	Marge
3	Maggie	11	Homer
3	Maggie	12	Marge



Each child can have more than one parent.

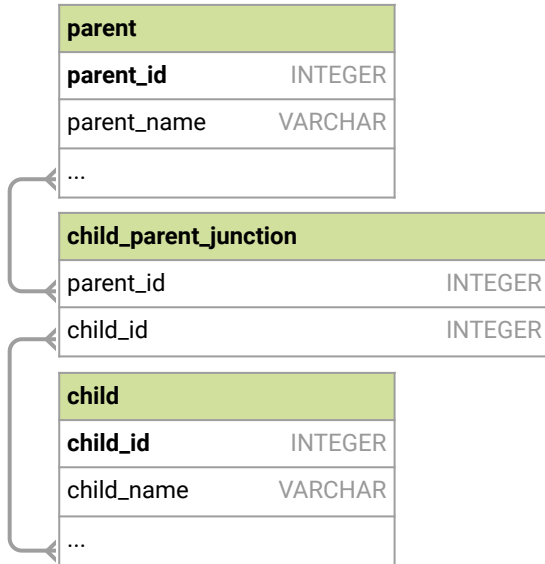


Each parent can have more than one child.



The two tables are joined using a ***junction table*** (next slide)

# Junction Table is used to store many-to-many relationships



	parent_id integer	child_id integer
1	11	1
2	11	2
3	11	3
4	12	1
5	12	2
6	12	3

*You can join the junction table with the child and parent tables to view the relationship*

	parent_name character varying (255)	child_name character varying (255)
1	Homer	Bart
2	Homer	Lisa
3	Homer	Maggie
4	Marge	Bart
5	Marge	Lisa
6	Marge	Maggie

The Junction table contains many parent\_ids and many child\_ids



## **Activity:** Data Relationships

In this activity, you will create table schemata for students and available courses, and then create a junction table to display all courses taken by students.

(Instructions sent via Slack)

**Suggested Time:**  
15 Minutes





**Time's Up!** Let's Review.

# Take a Break!

---



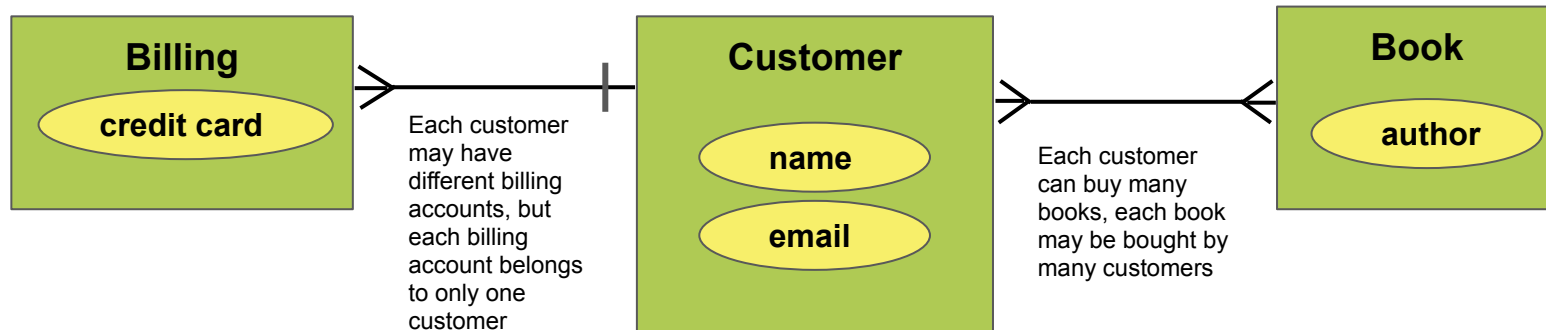
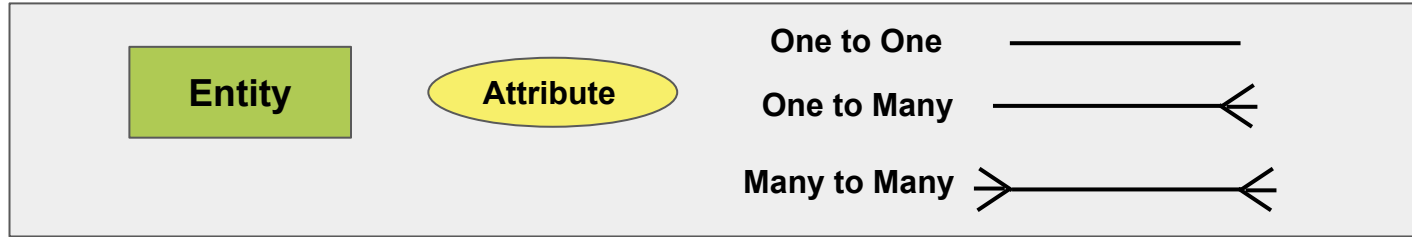
# Entity Relationship Diagrams



An **entity relationship diagram**, or **ERD**, is a visual representation of entity relationships within a database.

# ERDs

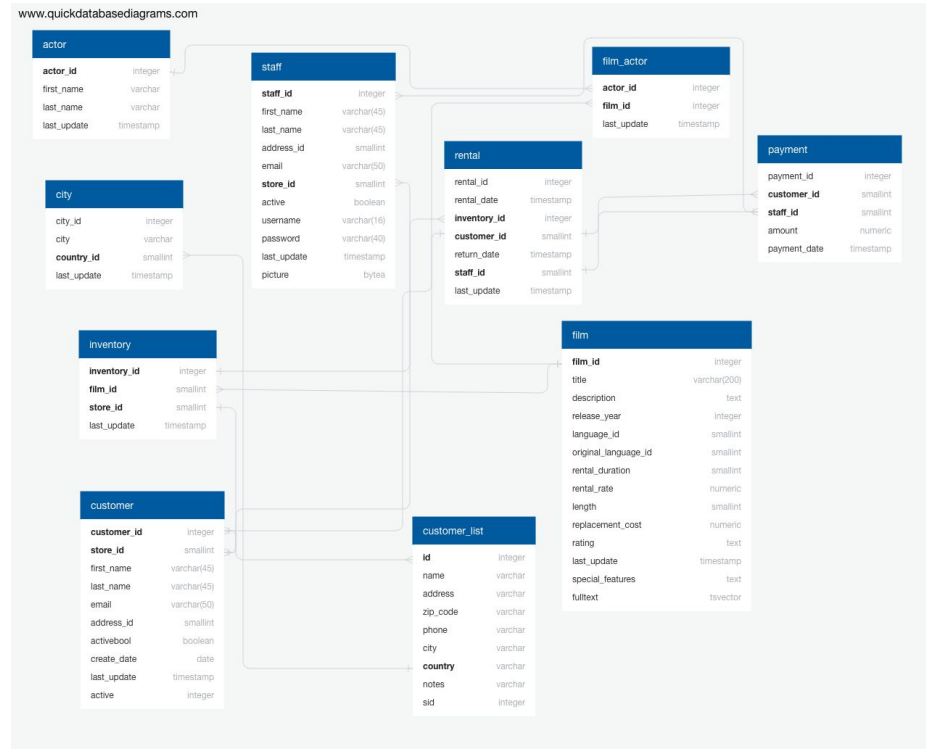
ERD use the following notation to create the models.





# ERDs

Entities, their data types, and relationships are all illustrated in the diagram.



# ERDs

---

There are three models used when creating diagrams:



**Conceptual:** basic information containing table and column names.



**Logical:** slightly more complex than conceptual models with IDs and attribute..

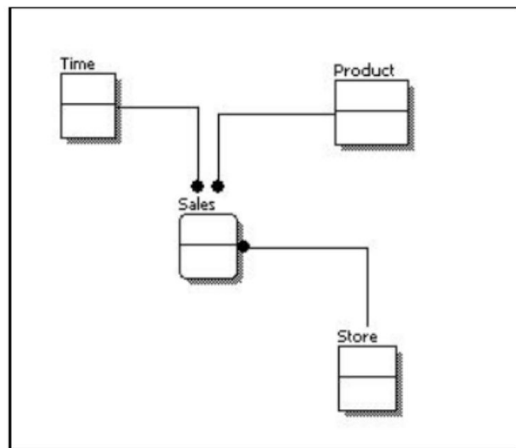


**Physical:** the blueprint of the database, with data types and physical relationships between entities.

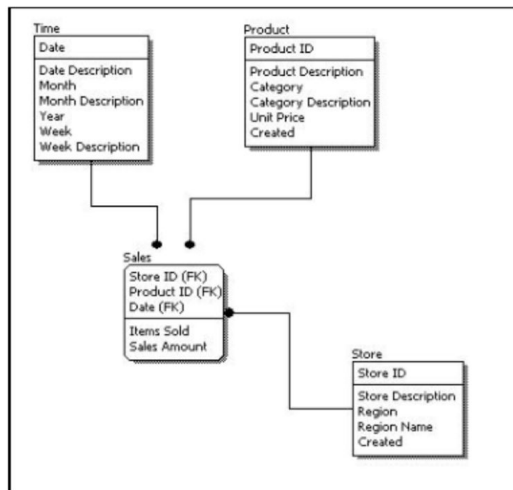
# ERDs

There are three models used when creating diagrams:

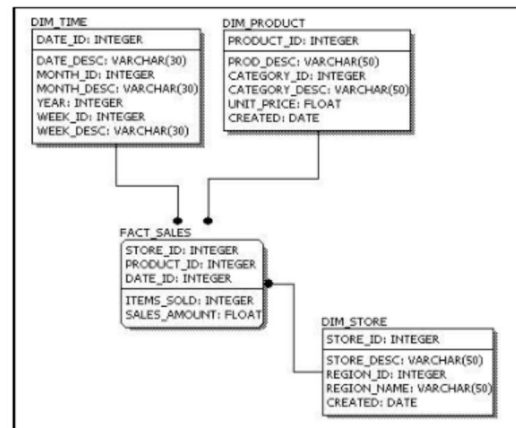
## Conceptual Model Design

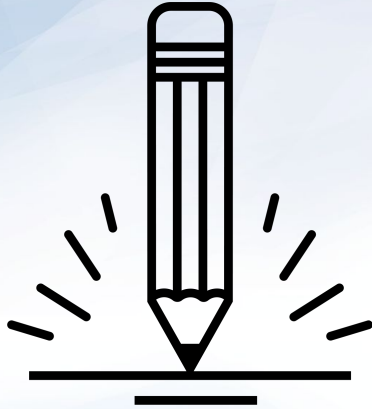


## Logical Model Design



## Physical Model Design





## **Activity:** Designing an ERD, Part II

In this activity, you will further improve on the ERD by creating a physical ERD.

(Instructions sent via Slack)

**Suggested Time:**  
15 Minutes





**Time's Up!** Let's Review.