Designing A Database

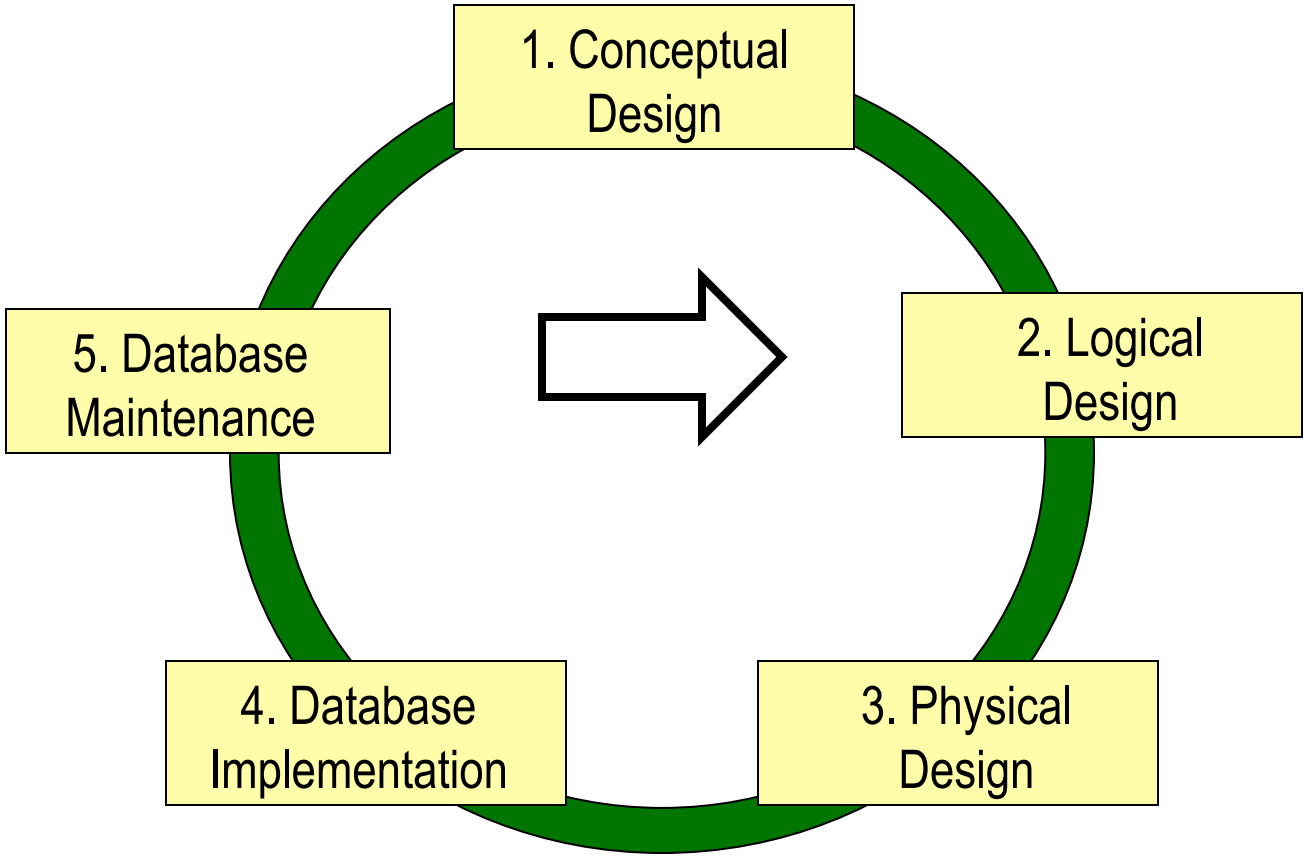
Last Lecture

- Entities
- Attributes
- Relationships
  - Cardinality
  - Modality
- Exercises

This Lecture

- Concepts
- Converting relationships
  - Binary
  - Unary
  - Ternary
- Normalization
- Exercises

Database Life Cycle



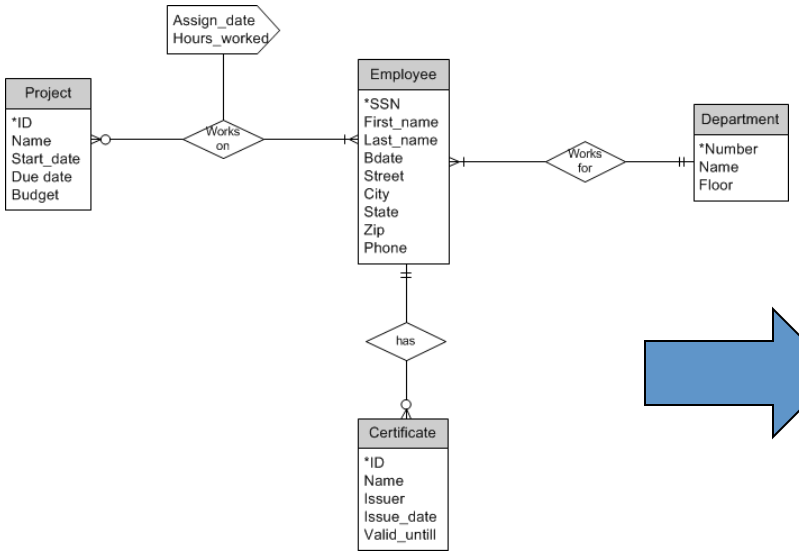
Logical Design

- Purpose:
  - To convert an organization's data model (ERD) into a set of database structures
    - Relations for a relational database
  - To ensure those relations are well-structured and free of most anomalies (normalization)

- Output:
  - A set of third normal form (3NF) relations

Goal

- Create 3NF relations



PROJECT (ID, Name, … Budget)

PROJECT_EMPLOYEE (ID, SSN, …)

EMPLOYEE (SSN, First_name, …, Phone)

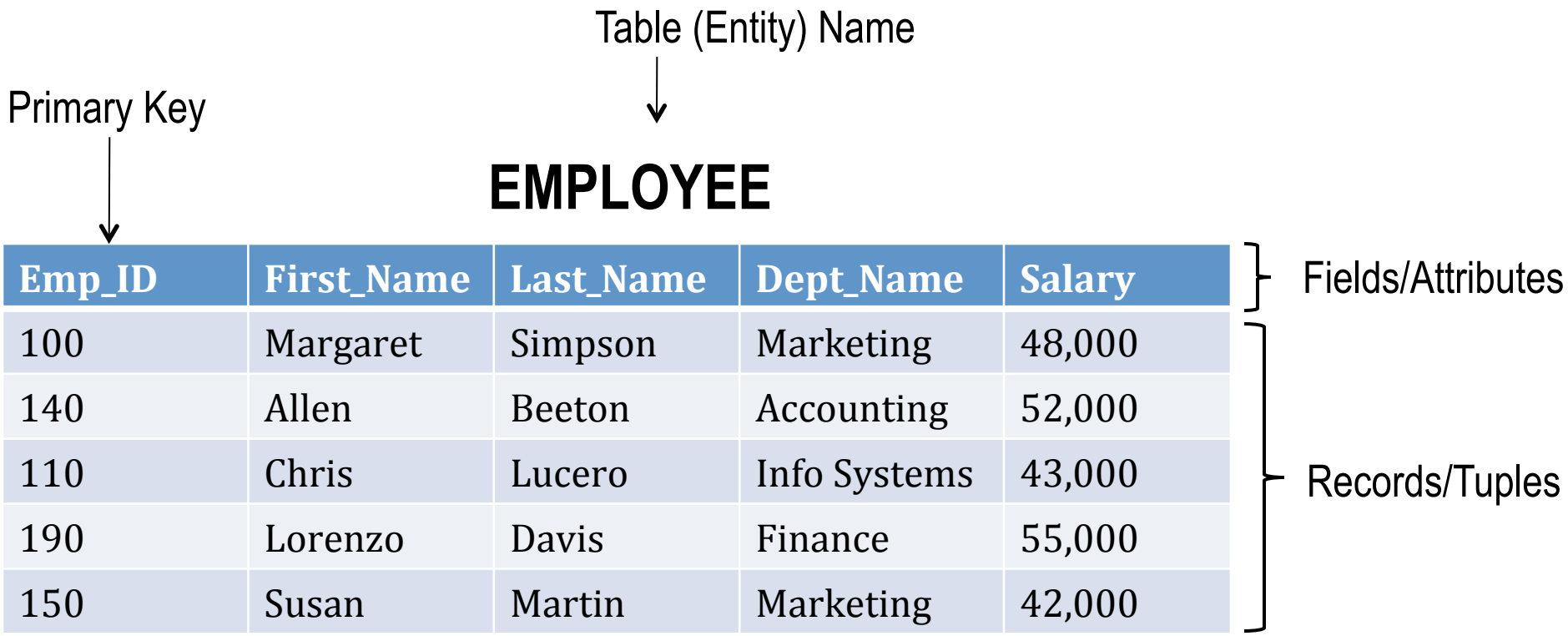
CERTIFICATE (ID, Name, …, Valid_until)

DEPARTMENT (Number, Name, Floor)

Concepts

- Candidate keys – One or more sets of attributes that can uniquely identify any record (entity instance) in a relation
- Composite key – Keys that consist of multiple attributes
- Primary key – Candidate key that is chosen as identifier
  - Decision? Practical utility (e.g., smallest key) or application needs
- Alternate key – Candidate keys that are not chosen to be the primary key

Concepts



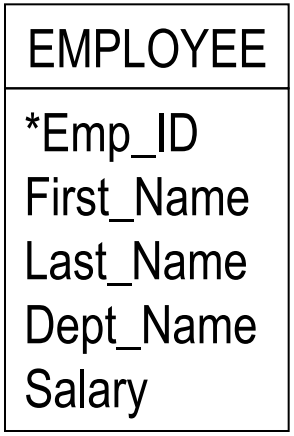
Emp_ID	First_Name	Last_Name	Dept_Name	Salary
100	Margaret	Simpson	Marketing	48,000
140	Allen	Beeton	Accounting	52,000
110	Chris	Lucero	Info Systems	43,000
190	Lorenzo	Davis	Finance	55,000
150	Susan	Martin	Marketing	42,000

Converting a Single Entity

- Entity becomes a table
- Attributes become fields of the table
- Primary key is underlined

SALESPERSON (SP_Number, SP_Name, Commission, Year_of_Hire)

Converting a Single Entity



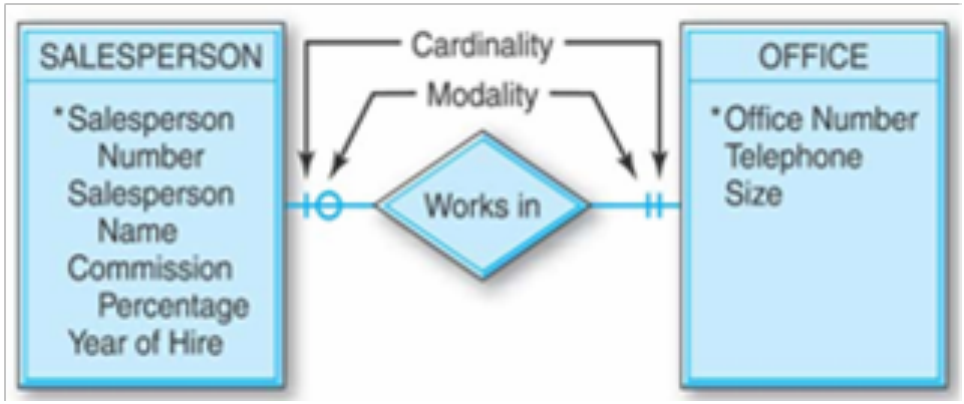
EMPLOYEE (Emp_ID, First_Name, Last_Name, Dept_Name, Salary)

EMPLOYEE

Emp_ID	First_Name	Last_Name	Dept_Name	Salary
100	Margaret	Simpson	Marketing	48,000
140	Allen	Beeton	Accounting	52,000
110	Chris	Lucero	Info Systems	43,000
190	Lorenzo	Davis	Finance	55,000
150	Susan	Martin	Marketing	42,000

Converting (1:1) Binary Relationship

- Copy primary key of "mandatory/parent" entity to "optional/child" entity
  - If relationship is symmetric (i.e. both are "mandatory"), either primary key can be copied.



Foreign Key (FK)

SALESPERSON (SP_Number, SP_Name, Commission, Year_Of_Hire, Office_Number)

OFFICE (Office_Number, Telephone, Size)

Foreign Keys

- A field (or a set of fields) in one table that is also the primary key in another table
- Mechanism for linking two tables in a relational database

EMPLOYEE

Emp_ID	First_Name	Last_Name	Salary
100	Margaret	Simpson	48,000
140	Allen	Beeton	52,000
110	Chris	Lucero	43,000
190	Lorenzo	Davis	55,000
150	Susan	Martin	42,000

COURSE

Course_Num	Course_Name	Emp_ID
4141	Java	190
6217	DB Admin	140
6136	Data Mining	140
6480	eCommerce	110

Foreign Keys

- Foreign key (FK) need not have the same name as the primary key (PK) in another relation
- Foreign keys must
  - Store data of the same type (e.g., integers) as PK
  - FK values must be a subset of PK values

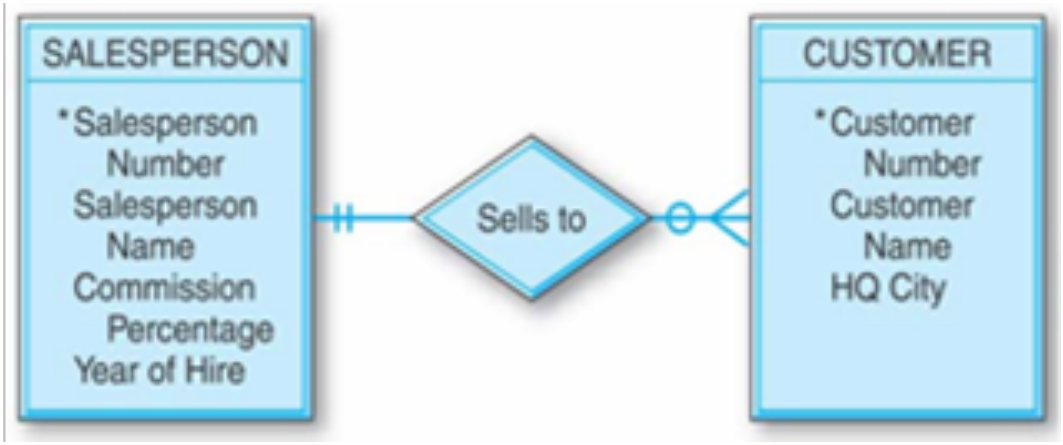
- Foreign keys are indicated by a dotted underline

SALESPERSON (SP_Number, SP_Name, Commission, Year_Of_Hire, Office_Num)

OFFICE (Office_Number, Telephone, Size)

Converting (1:N) Binary Relationship

- Primary key on the "one" side becomes foreign key on the "many" side

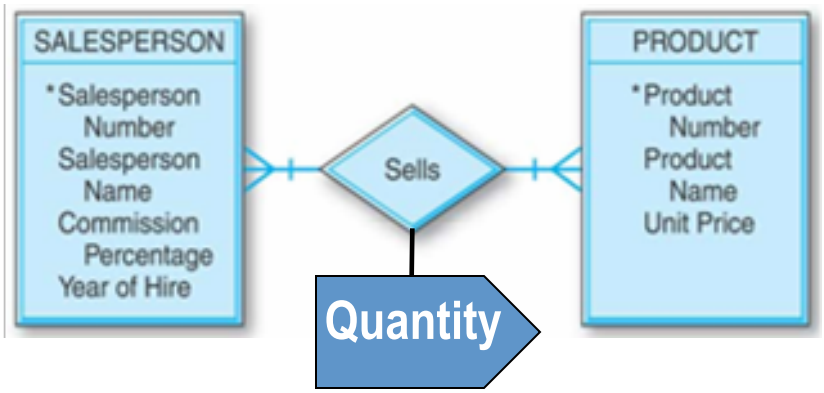


SALESPERSON (SP_Number, SP_Name, Commission, Year_Of_Hire)

CUSTOMER (Cust_Number, Cust_Name, HQ_City, SP_Number)

Converting (M:N) Binary Relationship

- Create intersection table (associative entity) with a composite primary key consisting of PKs of both parent entities



SALESPERSON (SP_Number, SP_Name, Commission, Year_Of_Hire)

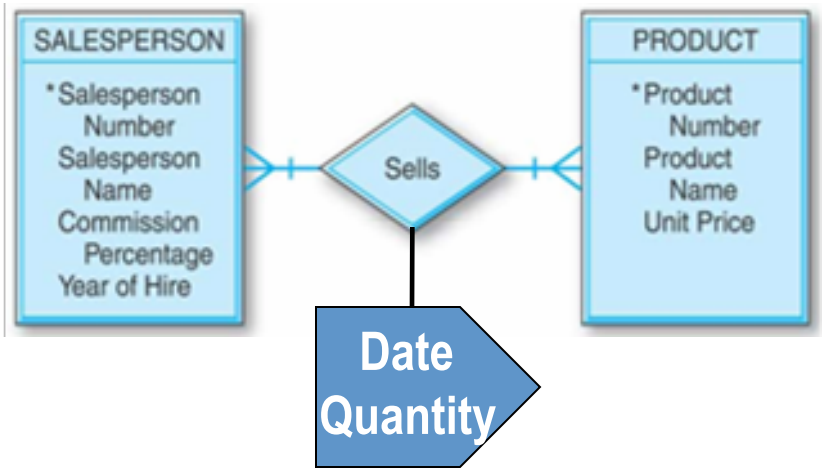
PRODUCT (Product_Number, Product_Name, Unit_Price)

SALE (SP_Number, Product_Number, Quantity) ← **Intersection Table**

Composite Primary Key

Converting (M:N) Binary Relationship

- Composite PK may also include additional fields



SALESPERSON (SP_Number, SP_Name, Commission, Year_Of_Hire)

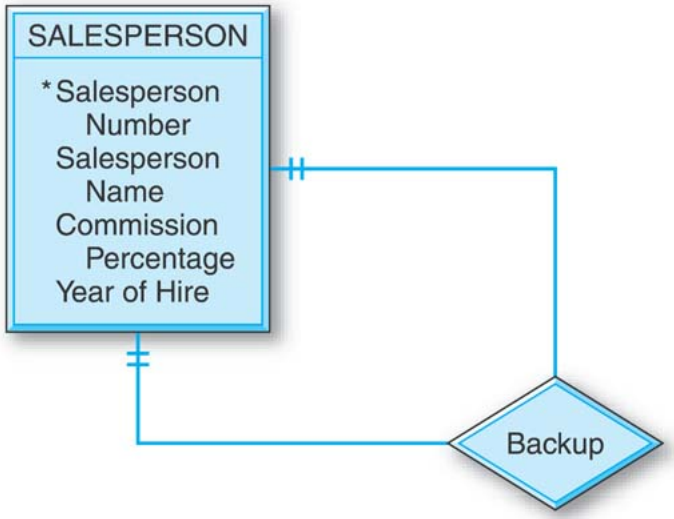
PRODUCT (Product_Number, Product_Name, Unit_Price)

SALE (SP_Number, Product_Number, Date, Quantity)

Converting (1:1) Unary Relationship

- Same as binary – Add primary key as a foreign key in the SAME table.
- Create a unique name for FK
  - Field names must be unique

Business rule: "Each salesperson has one backup salesperson"

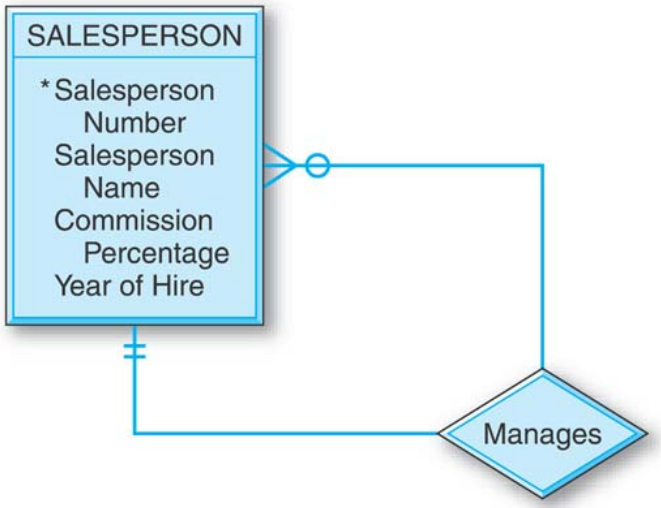


SALESPERSON (SP_Number, SP_Name, Commission, Year_Of_Hire, Backup_Number)

Converting (1:N) Unary Relationship

- Add primary key as a foreign key in the SAME table

Business rule: "Each salesperson manages zero or more salespersons"

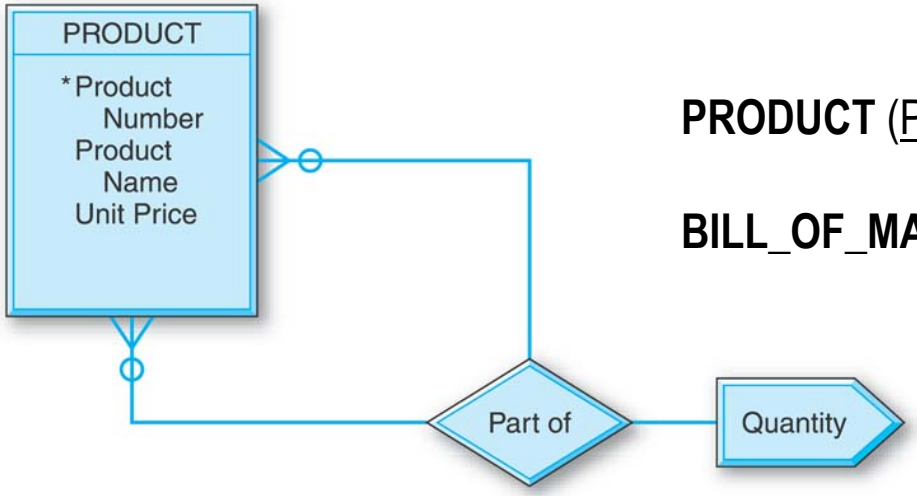


SALESPERSON (SP_Number, SP_Name, Commission, Year_Of_Hire, Manager)

Converting (M:N) Unary Relationship

- Create an intersection table with a composite primary key
  - PK of original relation used for each part of composite PK

Business rule: "A product can be a set of other products"

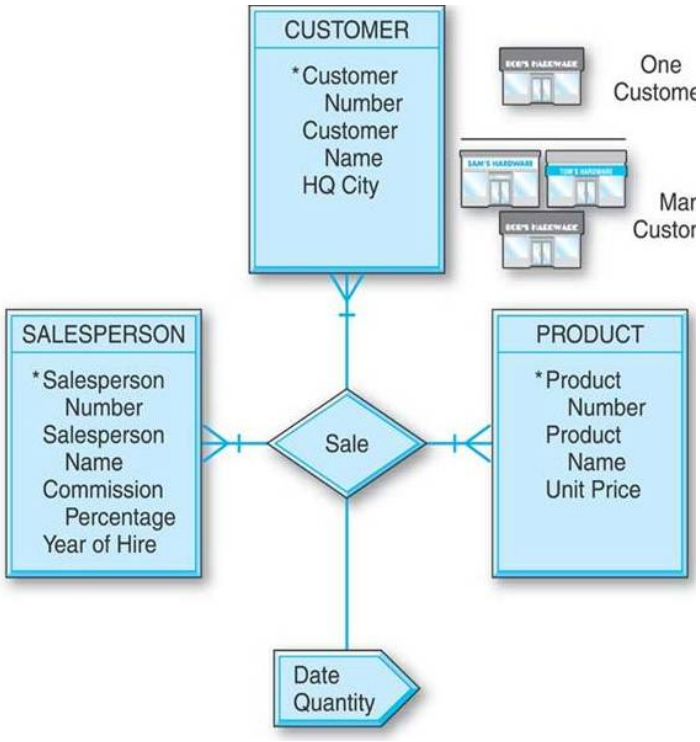


PRODUCT (Product_Num, Product_Name, Unit_Price)

BILL_OF_MATERIAL (Product_Num, Part_Num, Quantity)

Converting Ternary Relationship

- Ternary (n-ary) relationships map to FOUR (n+1) tables:
  - One table for each original entity
  - One table for the associative entity



CUSTOMER (Cust_Num, Cust_Name, HQ_City)

SALESPERSON (SP_Num, SP_Name, Commission, Year_of_Hire)

PRODUCT (Prod_Num, Prod_Name, Unit_Price)

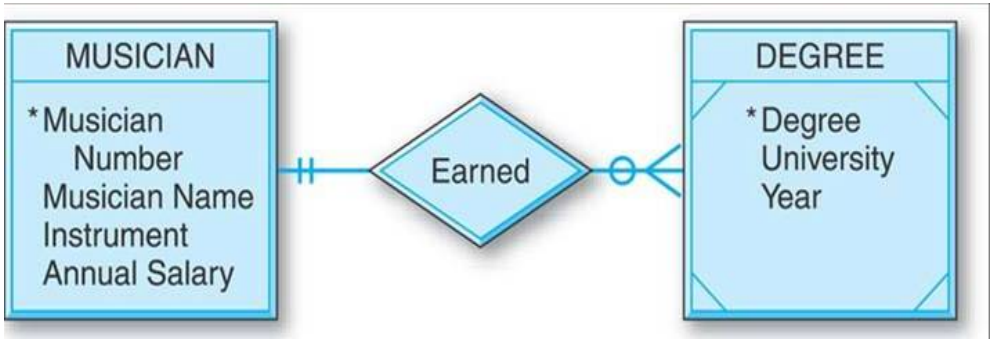
TRANSACTION (Cust_Num, SP_Num, Prod_Num, Sale_Date, Quantity)

OR

TRANSACTION (Trans_Num, Cust_Num, SP_Num, Prod_Num, Sale_Date, Quantity)

Weak / Dependent Entities

- Create separate tables for strong and weak entities
- PK of the weak table is a composite key:
  - PK of strong entity
  - PK of the weak entity



MUSICIAN (Musician_Num, Musician_Name, Instrument, Annual_Salary)

DEGREE (Musician_Num, Degree, University, Year)

Identifier of Strong Entity

Well-Structured Tables

- Definition:
  - A table that contains minimal data redundancy and allows users to insert, modify, and delete rows in the table without errors or inconsistencies

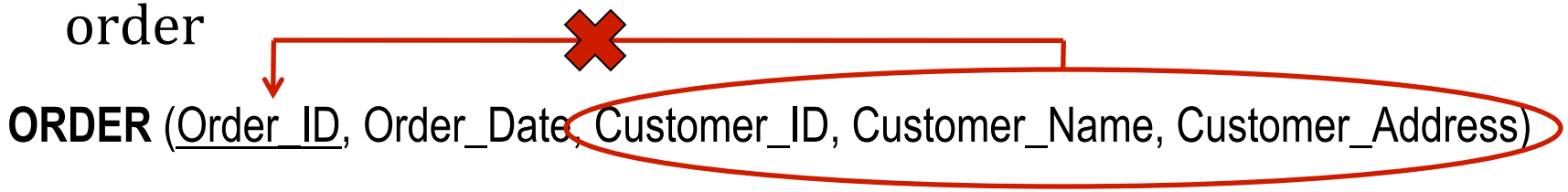
- Three anomalies to avoid in a well-structured table

Anomalies

- Insertion anomalies – occurs when we cannot store a value for one field because the value of another field is unknown.
- Update anomalies – occurs when we have to change multiple records in a table to update a single value of a field
- Deletion anomalies – occurs when deleting the value for one field unexpectedly also removes the value for another field

Insertion Anomaly Example

- Can't add data about a customer until he/she places an order



ORDER

Order_ID	Order_Date	Customer_ID	Customer_Name	Customer_ Address
1283	04/23/2008	132	John Doe	123 Main…
1746	05/05/2010	132	John Doe	123 Main…
2219	07/03/2010	289	Sarah Anderson	431 32nd…
2892	01/15/2011	132	John Doe	123 Main…
3284	02/04/2011	110	Rebecca Lowe	772 State…
???	**???**	**445**	**Laura Fisher**	**111 S 3rd…**

Update Anomaly Example

- If a customer changes address, multiple order records have to be updated

ORDER (Order_ID, Order_Date, Customer_ID, Customer_Name, Customer_Address)

ORDER

Order_ID	Order_Date	Customer_ID	Customer_Name	Customer_ Address
1283	04/23/2008	132	John Doe	123 Main…
1746	05/05/2010	132	John Doe	123 Main…
2219	07/03/2010	289	Sarah Anderson	431 32nd…
2892	01/15/2011	132	John Doe	456 S 3rd…
3284	02/04/2011	110	Rebecca Lowe	772 State…

Delete Anomaly Example

- Dropping an order can drop all information about that customer

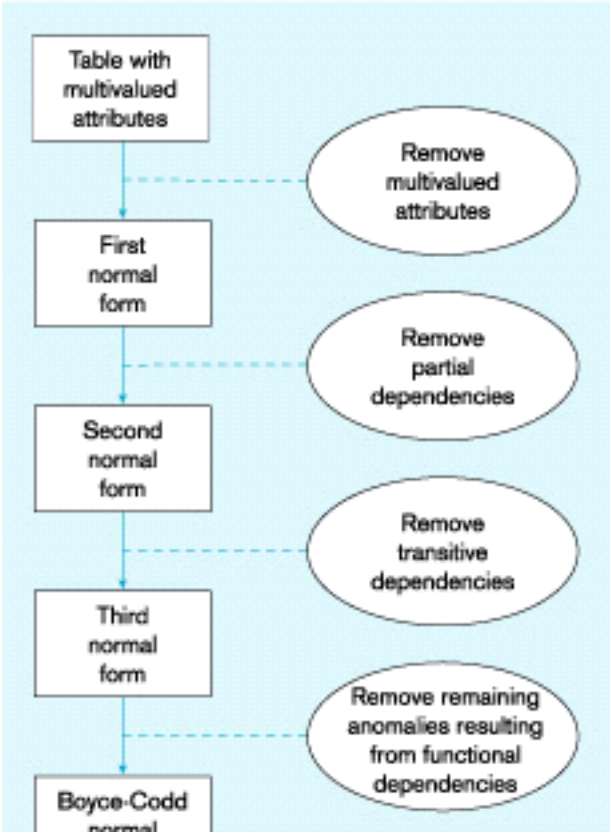
ORDER (Order_ID, Order_Date, Customer_ID, Customer_Name, Customer_Address)

ORDER

Order_ID	Order_Date	Customer_ID	Customer_Name	Customer_ Address
1283	04/23/2008	132	John Doe	123 Main…
1746	05/05/2010	132	John Doe	123 Main…
2219	07/03/2010	289	Sarah Anderson	431 32nd…
2892	01/15/2011	132	John Doe	456 S 3rd…
3284	02/04/2011	110	Rebecca Lowe	772 State…

Normalization

- The process of creating well-structured relations that:
  - Are relatively free from insertion, update, and delete anomalies
  - Have minimum data redundancy
- Done in layers:
  - First normal form (1NF)
  - Second normal form (2NF)
  - Third normal form (3NF)
    - Industry standard
  - Boyce-Codd normal form (BCNF)
  - Fourth Normal Form (4NF)
  - Fifth Normal Form (5NF)
  - Domain-Key Normal Form (DKNF)

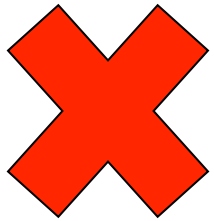


First Normal Form (1NF)

- Criterion 1: Each field value is atomic (no field is multi-valued)
  - Address is another example

EMPLOYEE

Emp_ID	Course	Name
0001	ISM4141	Jack Smith



EMPLOYEE

Emp_ID	Course	First_ Name	Last _ Name
0001	ISM4141	Jack	Smith

First Normal Form (1NF)

- Criterion 2: There are no repeating groups
- Solution: Modify PK to eliminate multi-valued fields in any cell

EMPLOYEE

Repeating Group

Emp_ID	Course	First_ Name	Last _ Name	Dept	Salary	Date _Comp
0001	ISM4141 ISM4222 ISM6332	Jack	Smith	ISDS	$1,000	12/10 12/10 05/10
0002	MAN3141	Maria	Anderson	MAN	$2,000	08/09
0003	FIN1231	Beth	Doe	FIN	$3,000	12/10

0NF: **EMPLOYEE** (Emp_ID, Course, First_Name, Last_Name, Dept, Salary, Date_Comp)

1NF: **EMPLOYEE** (Emp_ID, Course, First_Name, Last_Name, Dept, Salary, Date_Comp)

Functional Dependencies

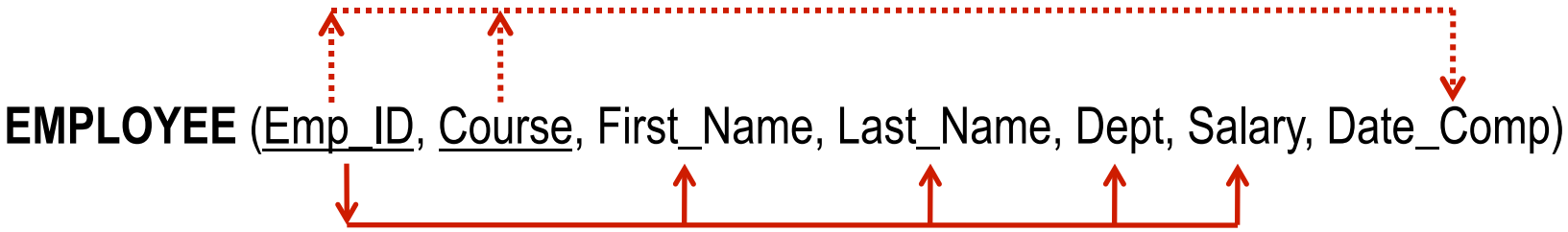
- Value of one particular field is associated with a single, specific value of another field
  - SSN → can determine a person's name, address, birth date, phone, etc.
  - ISBN → can determine a book's title, author, subject

- Good FD:
  - **EMPLOYEE** (SSN, Name, Address, Phone)

- Bad (partial) FD:
  - **PRODUCT** (Part_ID, Factory_ID, Part_Name, Part_Size, Factory_Name)

Second Normal Form (2NF)

- Criteria: 1NF + no partial dependency
  - Partial dependency – some non-key field(s) is determined by part of the PK and not the entire PK
  - Full functional dependency – every non-key field is determined by the entire PK (directly or indirectly)



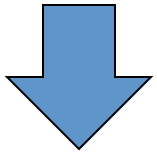
Partial dependency:

EmpID → (First_Name, Last_Name, Dept, Salary)

Second Normal Form (2NF)

- Solution: Decompose 1NF table into two tables

EMPLOYEE (Emp_ID, Course, First_Name, Last_Name, Dept, Salary, Date_Comp)



EMPLOYEE (Emp_ID, First_Name, Last_Name, Dept, Salary)

EMPLOYEE_COURSE (Emp_ID, Course, Date_Comp)

Third Normal Form (3NF)

- Criteria: 2NF + no transitive dependencies
  - Transitive dependency – functional dependencies between non-key attributes

SALES

Cust_ID	Name	Salesperson	Region
8023	Anderson	Smith	South
9167	Bancroft	Hicks	West
7924	Hobbs	Smith	South
6837	Tucker	Hernandez	East
8596	Eckersley	Hicks	West
7018	Arnold	Faulb	North

Transitive dependency: Salesperson → Region

Third Normal Form (3NF)

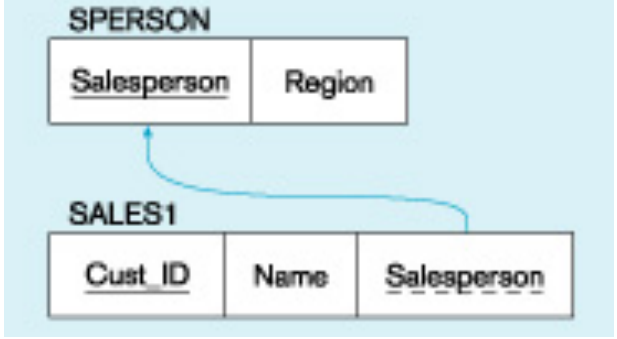
- Solution: Decompose 2NF table into multiple tables

SALES1

Cust_ID	Name	Salesperson
8023	Anderson	Smith
9167	Bancroft	Hicks
7924	Hobbs	Smith
6837	Tucker	Hernandez
8596	Eckersley	Hicks
7018	Arnold	Faulb

SPERSON

Salesperson	Region
Smith	South
Hicks	West
Hernandez	East
Faulb	North



2NF: **SALES** (Cust_ID, Name, Salesperson, Region)

3NF: **SALES1** (Cust_ID, Name, Salesperson)
SPERSON (Salesperson, Region)

Normal Forms

- Unnormalized – multi-valued fields and/or repeating groups
- First normal form (1NF) – No multi-valued fields or repeating groups
- Second normal form (2NF) – 1NF + no partial dependencies
- Third normal form (3NF) – 2NF + no transitive dependencies

Normalization Example

- Identify multi-valued fields, partial dependencies, transitive dependencies, and current normal form

BOOK (<u>isbn</u>, title, publisher, address)

Table	MV?	PD?	TD?	NF?
BOOK				

Normalization Example

- Identify multi-valued fields, partial dependencies, transitive dependencies, and current normal form

BOOK (<u>isbn</u>, title, publisher, address)

Table	MV?	PD?	TD?	NF?
BOOK	address			none

Normalization Example

- Identify multi-valued fields, partial dependencies, transitive dependencies, and current normal form

BOOK (<u>isbn</u>, title, publisher, street1, street2, city, state, zip)

Table	MV?	PD?	TD?	NF?
BOOK				

Normalization Example

- Identify multi-valued fields, partial dependencies, transitive dependencies, and current normal form

BOOK (<u>isbn</u>, title, publisher, street1, street2, city, state, zip)

Table	MV?	PD?	TD?	NF?
BOOK	No	No	publisher	2NF

What anomaly exists?

Normalization Example

- Identify multi-valued fields, partial dependencies, transitive dependencies, and current normal form

BOOK (isbn, title, publisher)

PUBLISHER (publisher, street1, street2, city, state, zip)

Table	MV?	PD?	TD?	NF?
BOOK				
PUBLISHER				

Normalization Example

- Identify multi-valued fields, partial dependencies, transitive dependencies, and current normal form

BOOK (isbn, title, publisher)

PUBLISHER (publisher, street1, street2, city, state, zip)

Table	MV?	PD?	TD?	NF?
BOOK	No	No	No	3NF
PUBLISHER	No	No	No	3NF

Normalization Example

- Identify multi-valued fields, partial dependencies, transitive dependencies, and current normal form

ORDER (<u>order_num</u>, <u>product_id</u>, description)

Table	MV?	PD?	TD?	NF?
ORDER				

Normalization Example

- Identify multi-valued fields, partial dependencies, transitive dependencies, and current normal form

ORDER (<u>order_num</u>, <u>product_id</u>, description)

Table	MV?	PD?	TD?	NF?
ORDER	No	Prod ID	No	1NF

Normalization Example

- Identify multi-valued fields, partial dependencies, transitive dependencies, and current normal form

ORDER (<u>order_num</u>, <u>product_id</u>)

PRODUCT (<u>product_id</u>, description)

Table	MV?	PD?	TD?	NF?
ORDER				
PRODUCT				

Normalization Example

- Identify multi-valued fields, partial dependencies, transitive dependencies, and current normal form

ORDER (<u>order_num</u>, <u>product_id</u>)

PRODUCT (<u>product_id</u>, description)

Table	MV?	PD?	TD?	NF?
ORDER	No	No	No	3NF
PRODUCT	No	No	No	3NF

Denormalization

- Why Denormalize?
  - Normalization may lead to too many tables → costly to join
- Maintain 3NF (industry standard):
  - Don't go further than 3NF

- Use caution when denormalizing:
  - Synonyms: Different name, same meaning
    - Student_ID and Student_No
  - Homonyms: Same name, different meanings
    - Phone ------ Home or Work?
  - Transitive dependencies may emerge when merging two tables together

This Lecture

- Concepts
  - Keys (PK, FK, Candidate, Composite, Alternate)
- Converting relationships (1..1, 1..M, M..N)
  - Binary
  - Unary
  - Ternary
- Normalization (1NF, 2NF, 3NF)
  - Anomalies