資料庫設計概論

課程綱要

- 關聯式資料庫邏輯結構
- 資料庫邏輯設計
  - Conceptual Design
- **Logical data model**
  - Broad data entities
  - Relationships
- **Physical data model**
  - Map to tables, records, fields

關聯式資料庫邏輯結構

- 資料以橫列直欄的方式組織於二維表格（Table）之中，各資料表（Table）存放現實世界中的實體或概念上認定存在的東西，例如：學生資料表、班級資料表、員工資料表。
- 每一直欄稱為欄位（Field）。
- 每一橫列稱為記錄（Record）。
- 每個資料表都各有其主鍵（Primary Key，PK）。
- 必要時，以某個欄位為外鍵（Foreign Key，FK）關聯到另一資料表的主鍵以獲得進一步的相關資料。

關聯式資料庫邏輯結構

- 每一直欄稱為**欄位（Field）**。

CityID	CityName
TP	台北
TC	台中
KS	高雄

EmpID	LastName	FirstName	CtryID	Extension	LastMod
integer	longstring	varchar(20)	char(2)	char(6)	longstring
101	Wang	Angle	TP	x19891	\HR\KarlD
102	Chien	Wolfgang	TC	x19433	\HR\KarlD
103	Martin	Jose	TP	x21467	\HR\AmyL

關聯式資料庫邏輯結構

- 每一橫列稱為記錄（Record）。

CityID	CityName
TP	台北
TC	台中
KS	高雄

EmpID	LastName	FirstName	CtryID	Extension	LastMod
integer	longstring	varchar(20)	char(2)	char(6)	longstring
101	Wang	Angle	TP	x19891	\HR\KarlD
102	Chien	Wolfgang	TC	x19433	\HR\KarlD
103	Martin	Jose	TP	x21467	\HR\AmyL

關聯式資料庫邏輯結構

- 每個資料表都各有其主鍵（Primary Key，PK）。
- 必要時，以某個欄位為外鍵（Foreign Key，FK）關聯到另一資料表的主鍵以獲得進一步的相關資料。

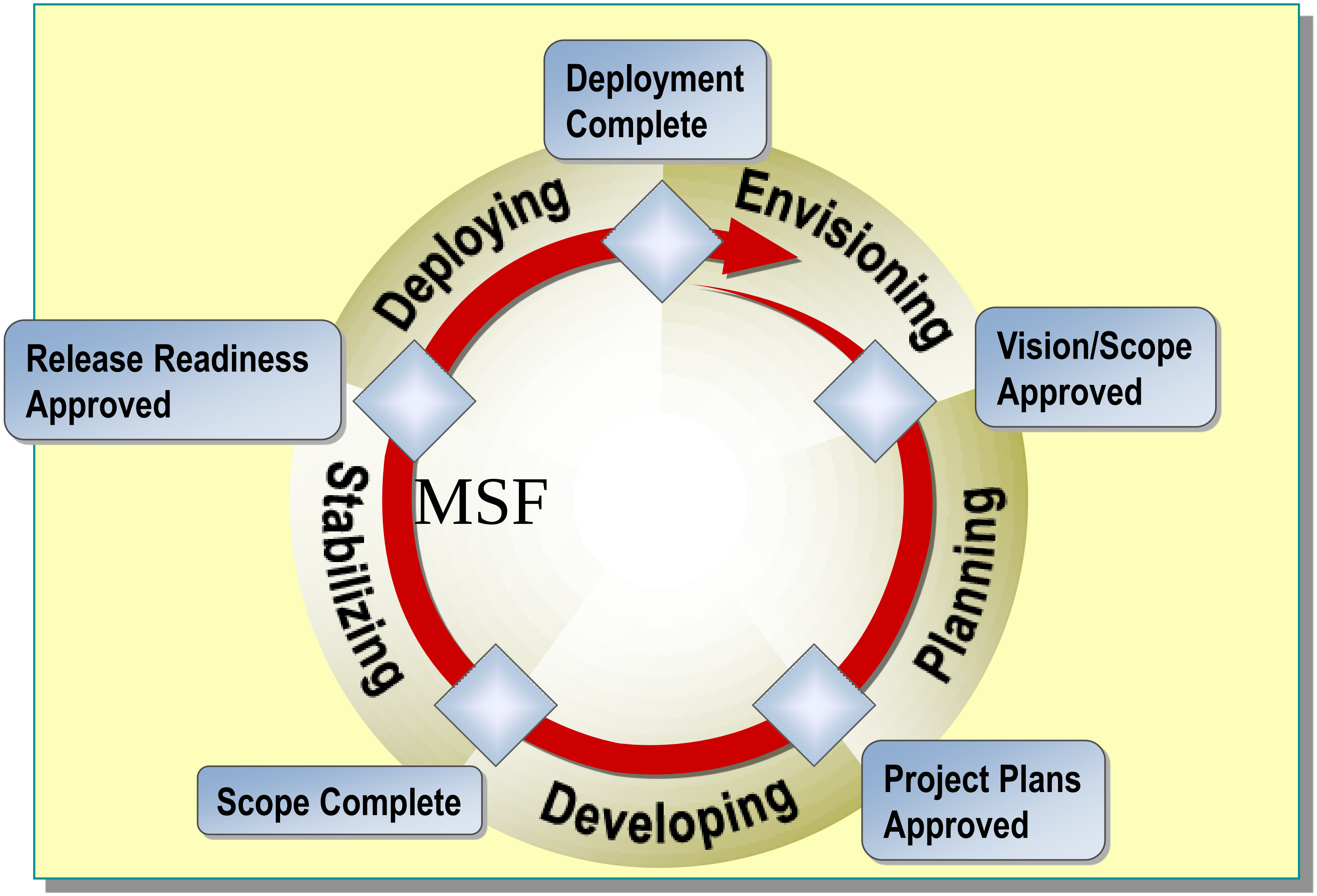
PK

CityID	CityName
TP	台北
TC	台中
KS	高雄

PK　FK

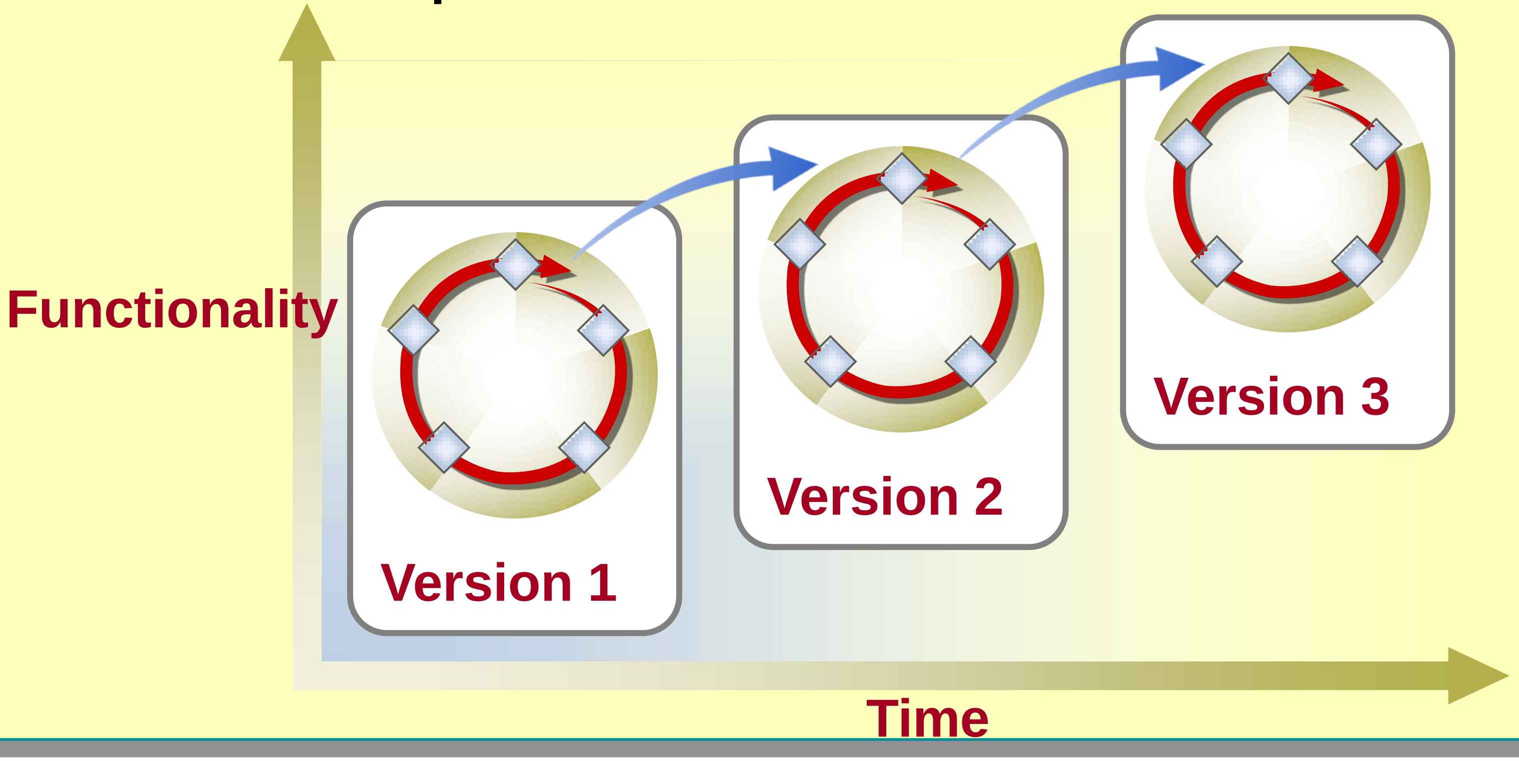
EmpID	LastName	FirstName	CtryID	Extension	LastMod
integer	longstring	varchar(20)	char(2)	char(6)	longstring
101	Wang	Angle	TP	x19891	\HR\KarlD
102	Chien	Wolfgang	TC	x19433	\HR\KarlD
103	Martin	Jose	TP	x21467	\HR\AmyL

Phases in the MSF Process Model



How to Use Iteration in Projects

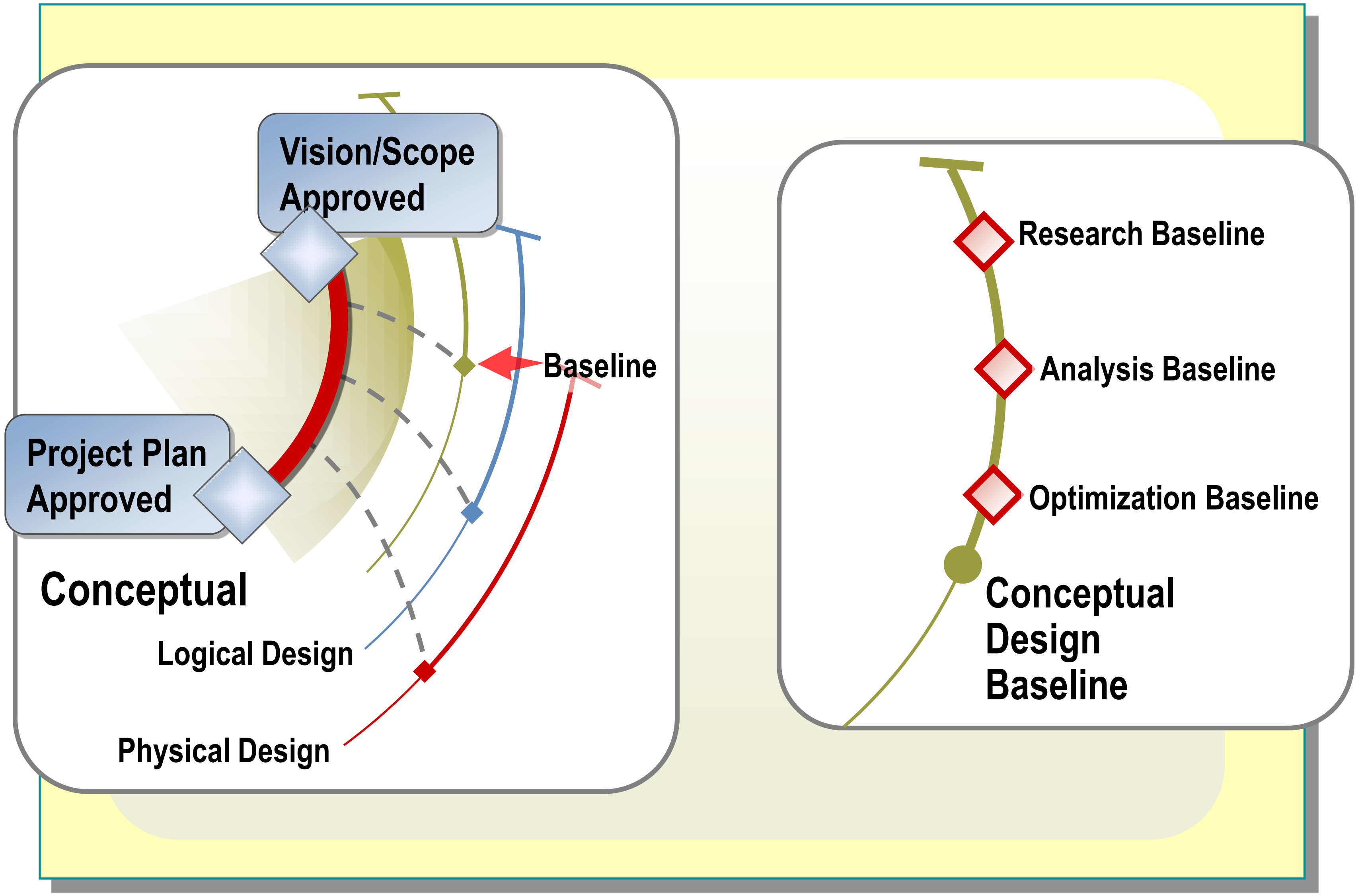
- The project develops in smaller steps
- Each iteration is identified with specific deliverables
- The team can produce versioned releases



Phases in the MSF Process Model

- **Envision** (Use Case Diagram)
- **Conceptual Design** (Use Case / Sequence / Active / State Diagram)
  - 1-1 Research → 1-2Analysis → 1-3 Optimization
- **Logical Design** (Class Diagram)
  - 2-1 Analysis → 2-2 Optimization
- **Physical Design** (Component / Deploy / Package Diagram)
  - 3-1 Research → 3-2 Analysis → 3-3 Rationalization → 3-4 Implementation

What Are the Steps in Conceptual Design?



Sources of Information

- **Artifacts**
  - Physical items in the business environment: training manuals, job aids
- **Systems**
  - Information systems and other processes that accomplish something: inventory tracking systems, intranets
- **People**
  - Stakeholders in the business who are sources of valuable insight and information

Techniques for Gathering Information

Technique	Description
Shadowing	Directly observe individuals doing their job to discover current practices and problems
Interviews	Gather specific information from individuals
Focus groups	Query a group to discover attitudes and shared perceptions
Surveys	Collect detailed and statistically significant data
User instruction	Ask end users to teach you how they work with a system
Prototyping	Simulate a system that would be impractical to test directly
Instrumented versions	Use an instrumented application to record how users perform tasks

Categories of Information

- **Business**

  Interaction between goals and objectives, products and services, financial structures, and major organizational structures

- **Applications**

  Automated and non-automated services that support the business processes

- **Operations**

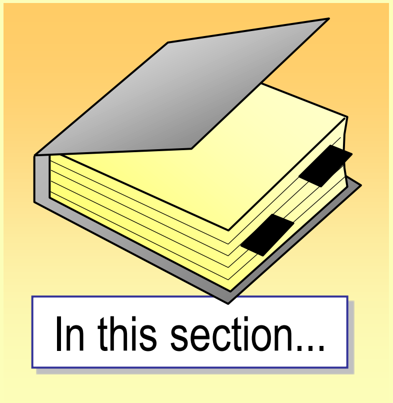
  Information needed to run business processes

- **Technology**

  Technical services needed to perform and support the business mission

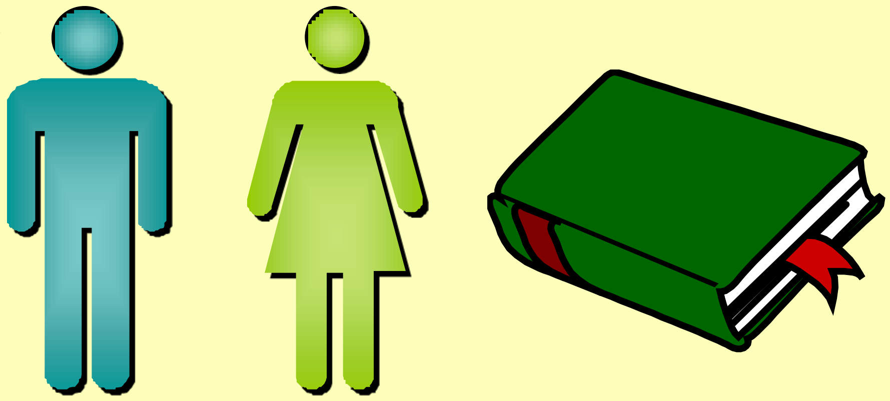
Logical Data Design

- Entity
- Attribute
- Relationship



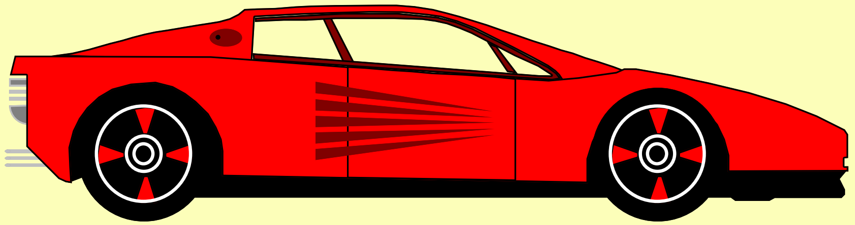
Deriving Entities

- Entities represent real-world objects about which information will be stored
- When deriving entities, look for nouns or noun phrases during analysis
- Rows in database tables
- Common examples
  - People
  - Books



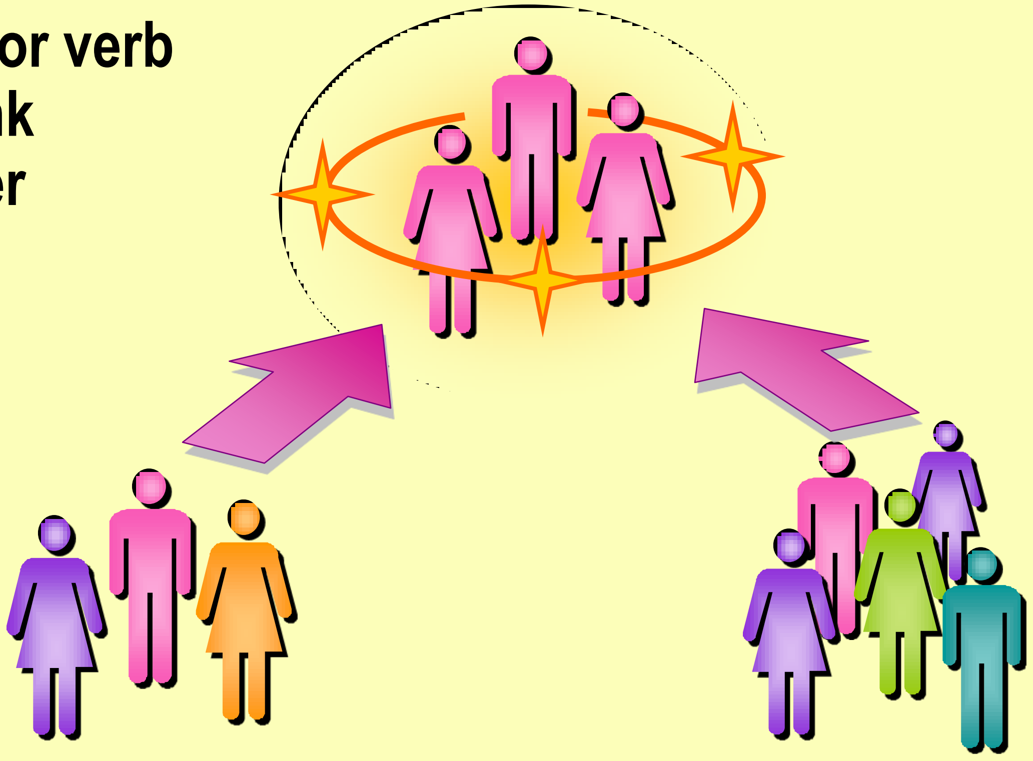
Deriving Attributes

- Descriptive information about an entity
- Attached to entity that they most closely describe
- Columns in database tables
- Example: Attributes of a car
  - Color
  - Make
  - Model
  - Year



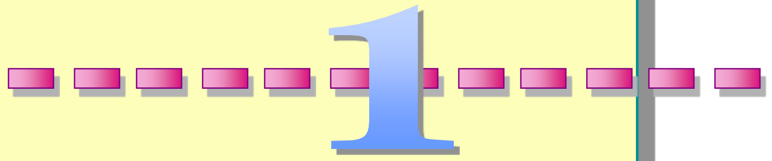
Identifying Relationships Between Entities

- Relationships represent associations among entities
- Look for verbs or verb phrases that link entities together



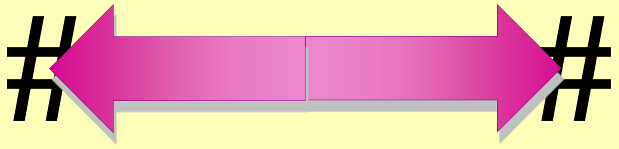
Overview of Cardinality and Existence

- Cardinality determines the number of instances of an entity that are allowed in a relationship
- Existence determines what entities must exist for the relationship to have meaning, given a specified cardinality



Determining Cardinality

- **Cardinality further defines a relationship by assigning it to one of three major categories:**
  - One-to-one
  - One-to-many
  - Many-to-many

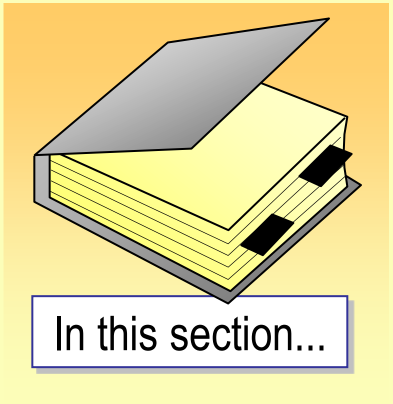


Determining Existence

- Depicts the conditions under which a relationship between two entities can exist
- Necessary when one entity requires an instance of another entity
- Can be one of two categories
  - Mandatory
  - Optional

Entity/Relationship Modeling

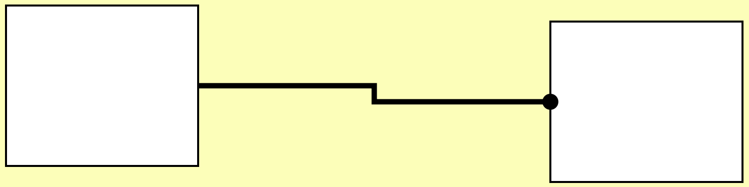
- Syntax
- Creating the Logical Data Model
- Activity 4.2: Creating a Logical Data Model



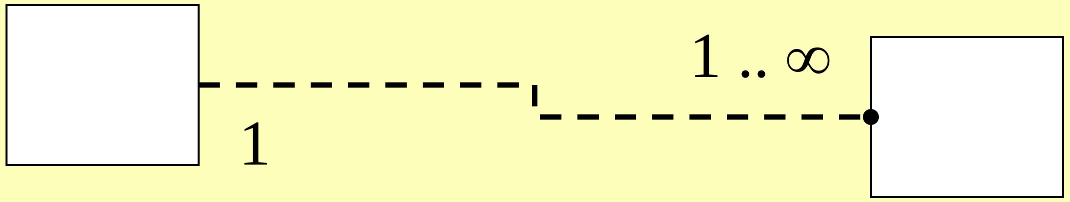
Syntax

- Represented by the entity/relationship diagram
- Entities and attributes are represented as rectangles

- Relationships are represented as lines (with dots) between entities

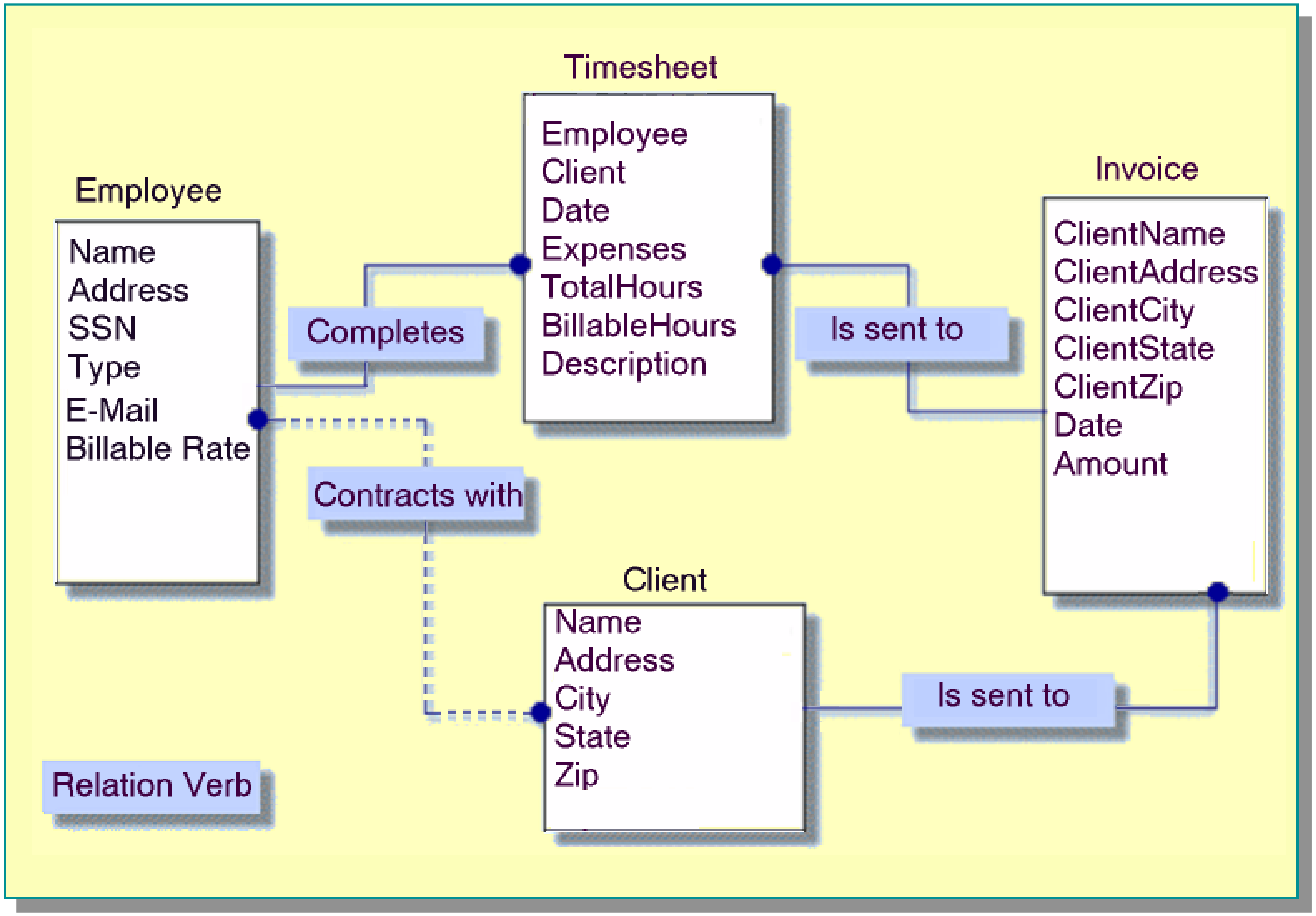


- Cardinality is denoted by a number at each end of the relationship



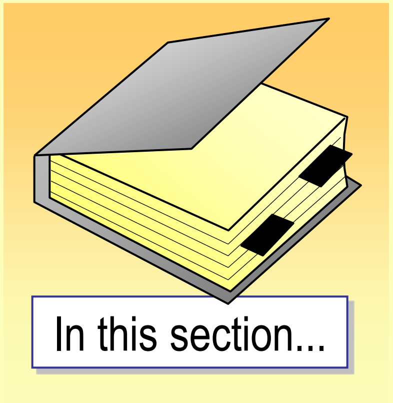
- Existence is denoted as a solid or dashed line

Creating the Logical Data Model



Normalization Basics

- Normalizing Logical Models
- Creating a First Normal Form Data Model
- First Normal Form Example
- Moving to a Second Normal Form Data Model
- Creating a Third Normal Form Data Model
- Third Normal Form Example
- Benefits of Normalization



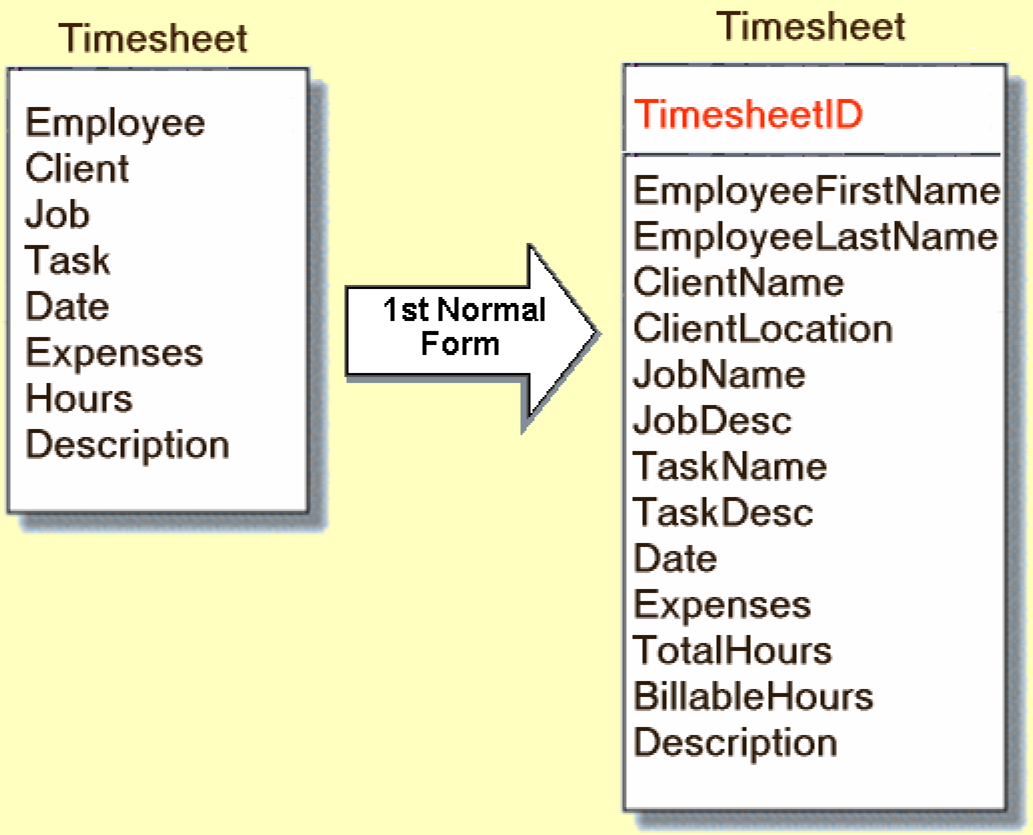
Normalizing Logical Models

- **Process of eliminating duplicate data, and usually, defining relationships among tables**
- **Normal forms**
  - First normal form
  - Second normal form
  - Third normal form
- **Normalized databases typically include more tables with fewer columns**

Creating a First Normal Form Data Model

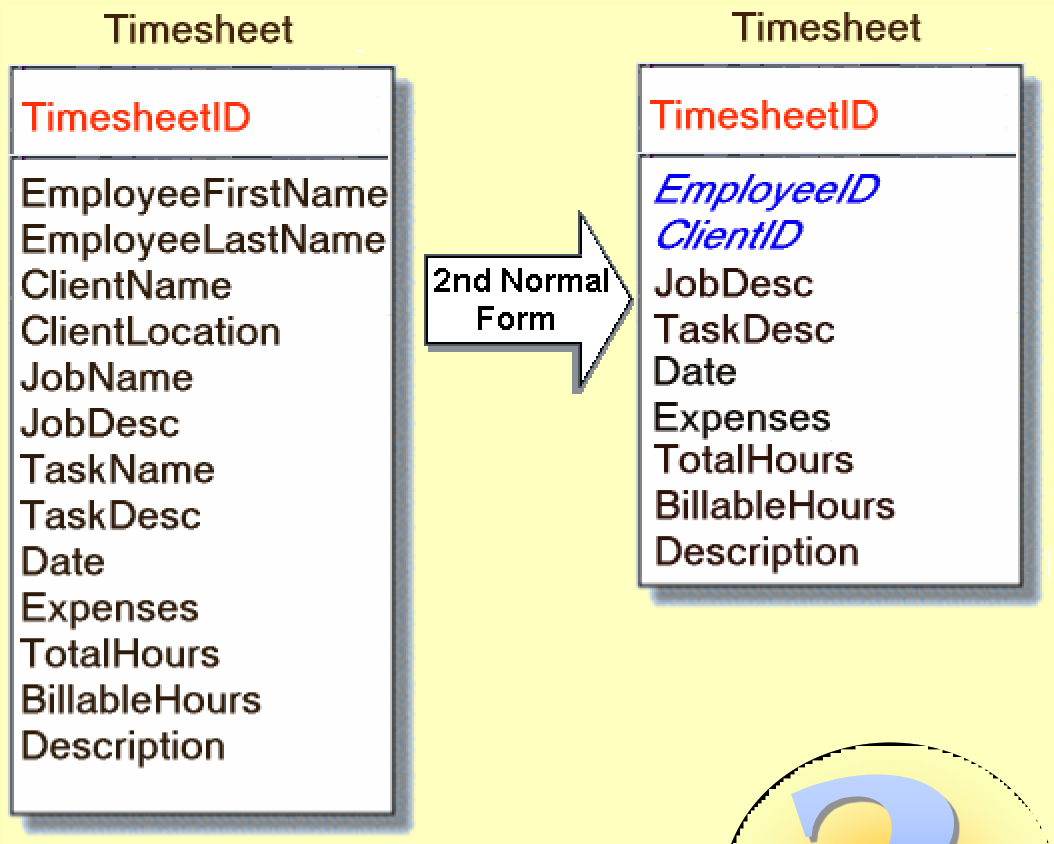
- Create two-dimensional tables
- Assign only one value to each cell
- Assign a single meaning to each column

First Normal Form Example



Moving a to Second Normal Form Data Model

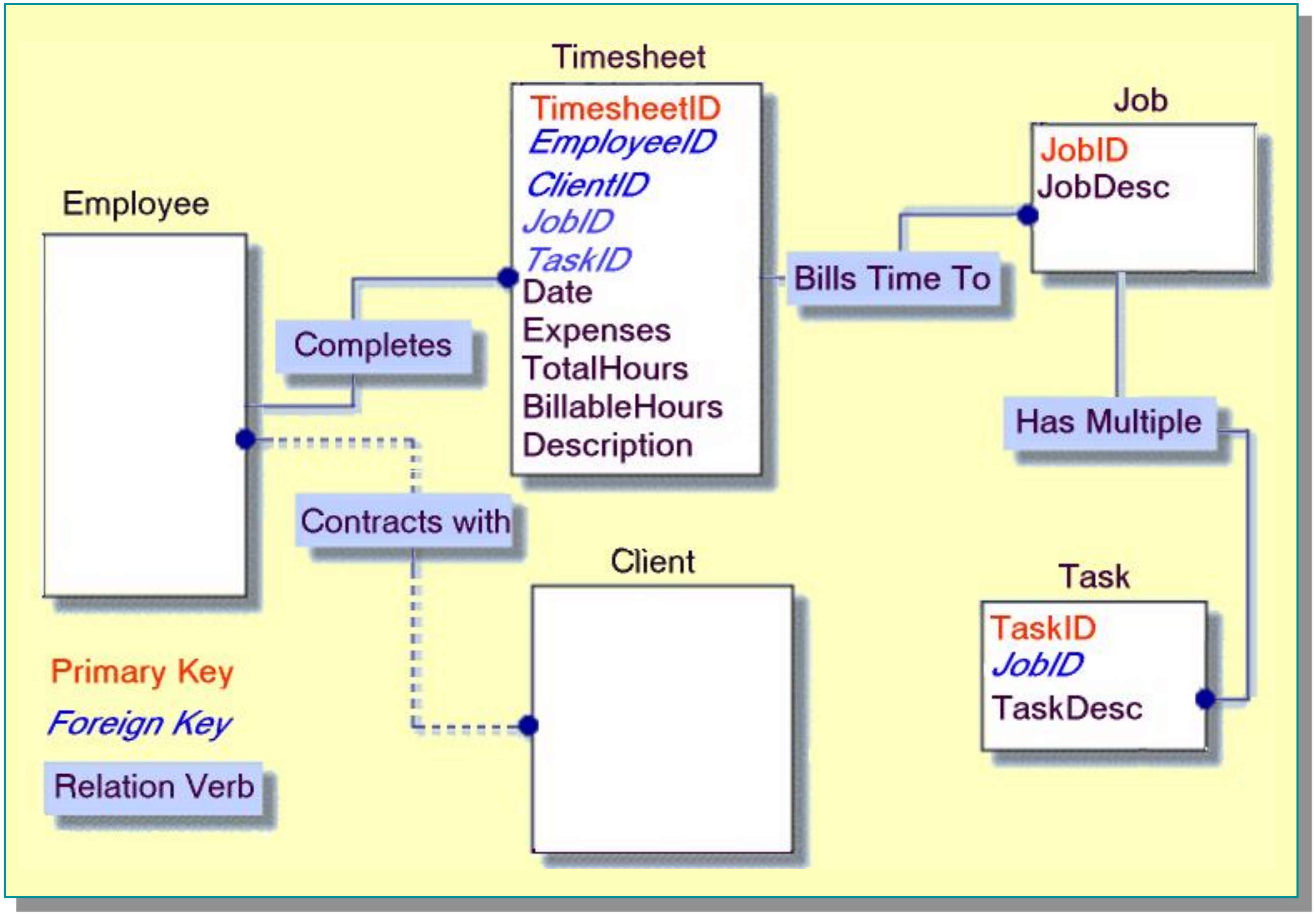
- Eliminate redundant data within an entity
- Move attribute that depends on only part of a multivalue key to a separate table
- Consolidate information when possible



Creating a Third Normal Form Data Model

- Eliminate any columns that do not depend on a key value for their existence
- Generally, move any data not directly related to entity to another table
- Reduce or eliminate update and deletion anomalies
- Verify that no redundant data remains

Third Normal Form Example


Timesheet
TimesheetID
EmployeeID
ClientID
JobID
TaskID
Date
Expenses
TotalHours
BillableHours
Description
Job
JobID
JobDesc
Employee
Completes
Bills Time To
Has Multiple
Contracts with
Client
Task
TaskID
JobID
TaskDesc
Primary Key
Foreign Key
Relation Verb