Previews of TDWI course books are provided as an opportunity to see the quality of our material and help you to select the courses that best fit your needs. The previews can not be printed.

TDWI strives to provide course books that are content-rich and that serve as useful reference documents after a class has ended.

This preview shows selected pages that are representative of the entire course book. The pages shown are not consecutive. The page numbers as they appear in the actual course material are shown at the bottom of each page. All table-of-contents pages are included to illustrate all of the topics covered by a course.

TDWI Advanced Data Modeling Techniques

TABLE OF CONTENTS

THE DATA WAREHOUSING INSTITUTE
A DECADE OF EXCELLENCE

Module 1

Data Modeling Concepts

Data Models in Context

Zachman Framework Overview

THE ZACHMAN FRAMEWORK FOR ENTERPRISE ARCHITECTURE™

author: John A. Zachman

source: www.zachmaninternational.com

	DATA (What)	FUNCTION (How)	NETWORK (Where)	PEOPLE (Who)	TIME (When)	MOTIVATION (Why)
SCOPE Contextual (Planner's View)	list of things	list of processes	list of locations	list of organizations	list of events	list of goals
BUSINESS MODEL Conceptual (Owner's View)	semantic model	process model	logistics system	workflow model	master schedule	business plan
SYSTEM MODEL Logical (Designer's View)	logical data model	application architecture	distributed system architecture	human interface architecture	processing structure	business rule model
TECHNOLOGY MODEL Physical (Builder's View)	physical data model	system design	technology architecture	presentation architecture	control structure	rule design
DETAIL REPRESENTATIONS Out of Context (Implementer's View)	data definition	program logic	network architecture	security architecture	timing definition	rule specification
FUNCTIONING ENTERPRISE (Worker's View)	data	function	network	organization	schedule	strategy

Note: The items in the framework cells are *examples* of artifacts that you might produce at the intersection of each row and column. They are not intended as a prescriptive statement of things that you must produce, or as a complete list of architectural deliverables.

Data Models in Context

Zachman Framework Overview

FRAMEWORK INCEPTION

The Zachman Framework for Enterprise Architecture was developed by observing the way work was planned and performed in disciplines, such as building, that have existed for centuries. He reasoned that the same fundamental concepts exist within the information environment and that applying them can provide systems and other products with the same lasting power and reliability of buildings and quality machinery.

PERSPECTIVES

The framework recognizes people have different roles or “perspectives”, depending on their need for and use of information in the creation of a product. These perspectives need to be developed sequentially.

- PLANNER: This role is concerned with positioning the product in the context of its environment. This provides the contextual view to establish the scope.
- OWNER: This role is interested in the business deliverable and how it will be used. This provides the conceptual (business) view.
- DESIGNER: This role works with the specifications for the product to ensure that it will fulfill the owner’s expectations. This provides the logical (system) view.
- BUILDER: This role manages the assembling and fabricating of the components to create the product. This provides the physical (technology) view.
- SUBCONTRACTOR: This role builds out-of-context (reusable) components to meet the builder’s specifications without specific knowledge of where the component will be placed.

DIMENSIONS

The framework also recognizes that answers to the six interrogatories must be provided to fully describe any perspective. These answers address:

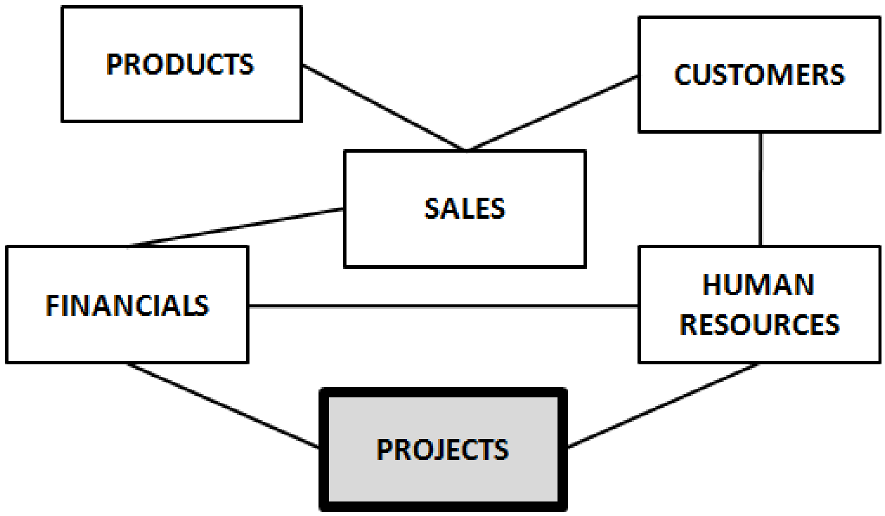
- Entities (what?)
- Activities (how?)
- Places (where?)
- People (who?)
- Timing (when?)
- Motivations (why)?

There is no implied sequence in answering these six questions.

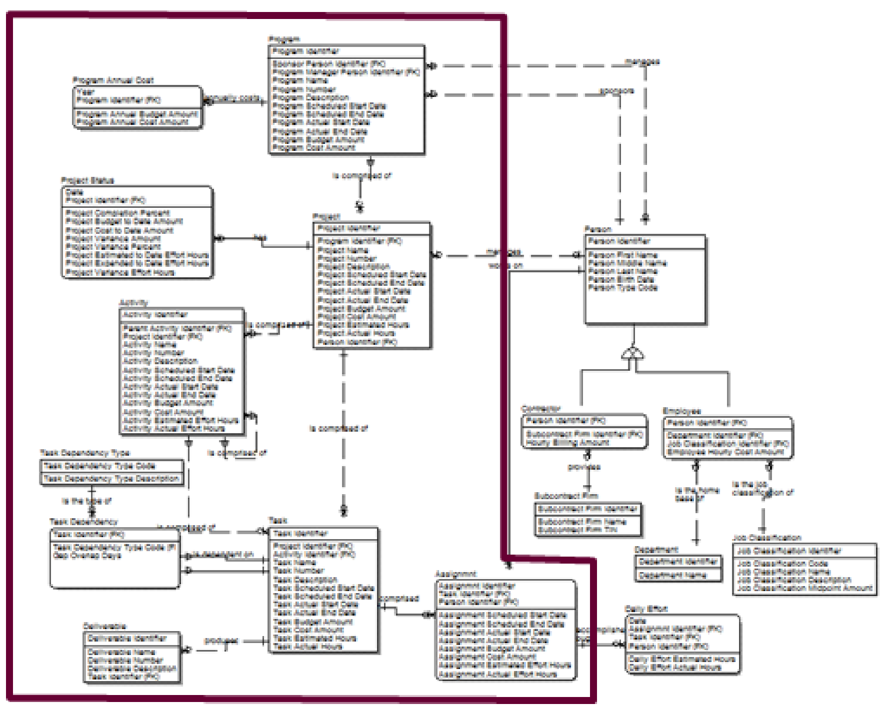
Data Models in Context

Levels of Data Models – Enterprise Perspective

PARTIAL SUBJECT AREA MODEL



PARTIAL BUSINESS DATA MODEL



Data Models in Context

Levels of Data Models – Enterprise Perspective

ENTERPRISE VS. PROJECT PERSPECTIVE

The enterprise perspective for data provides the foundation for all projects that use data. Regardless of whether or not such a perspective is physically implemented (it rarely is), its existence establishes a consistent set of business rules and definitions for all data used in the enterprise independent of any constraints imposed by the organizational structure, local processes, and system constraints.

SUBJECT AREA MODEL

Subject areas are the 10 – 20 broad groupings of data of interest to the enterprise. When these subject areas are defined so that they are mutually exclusive, the subject areas provide a framework for assigning responsibilities for data (e.g., data stewardship, modeling) and for setting the foundation for managing the business data model.

Development of the subject area model is fairly quick, and generic models are available to serve as a starting point. There are many subjects that are common across industries, including customers, products (or offerings), locations, information, facilities, and sales. In addition, specific industries have subject areas that are common across companies. For example, in the insurance industry, subject areas such as claims and incidents exist and (sold) policies may be used instead of “sales”.

BUSINESS DATA MODEL

The business data model is an abstraction or representation of the data in an enterprise. The business data model helps manage the data asset just as the chart of accounts helps to manage the money asset and the organizational chart helps to manage the human resources. The model is typically depicted in “third normal form”. Using this modeling technique ensures that it is internally consistent, that there is no redundancy, and that the relationships among data groupings (entities) are properly represented. The model can then be used as the foundation for all systems development projects. When the business model is used to guide development, the resultant data structures should be fairly consistent and integration of data from multiple systems will be much easier.

Development of the business data model can take several months and if one does not exist, most companies don’t sanction an independent effort to create one. Instead, the model is developed incrementally as data is needed for new development efforts. As with the subject area model, generic starting points are available. *The Data Model Resource Book* by Len Silverston is an extremely useful resource.

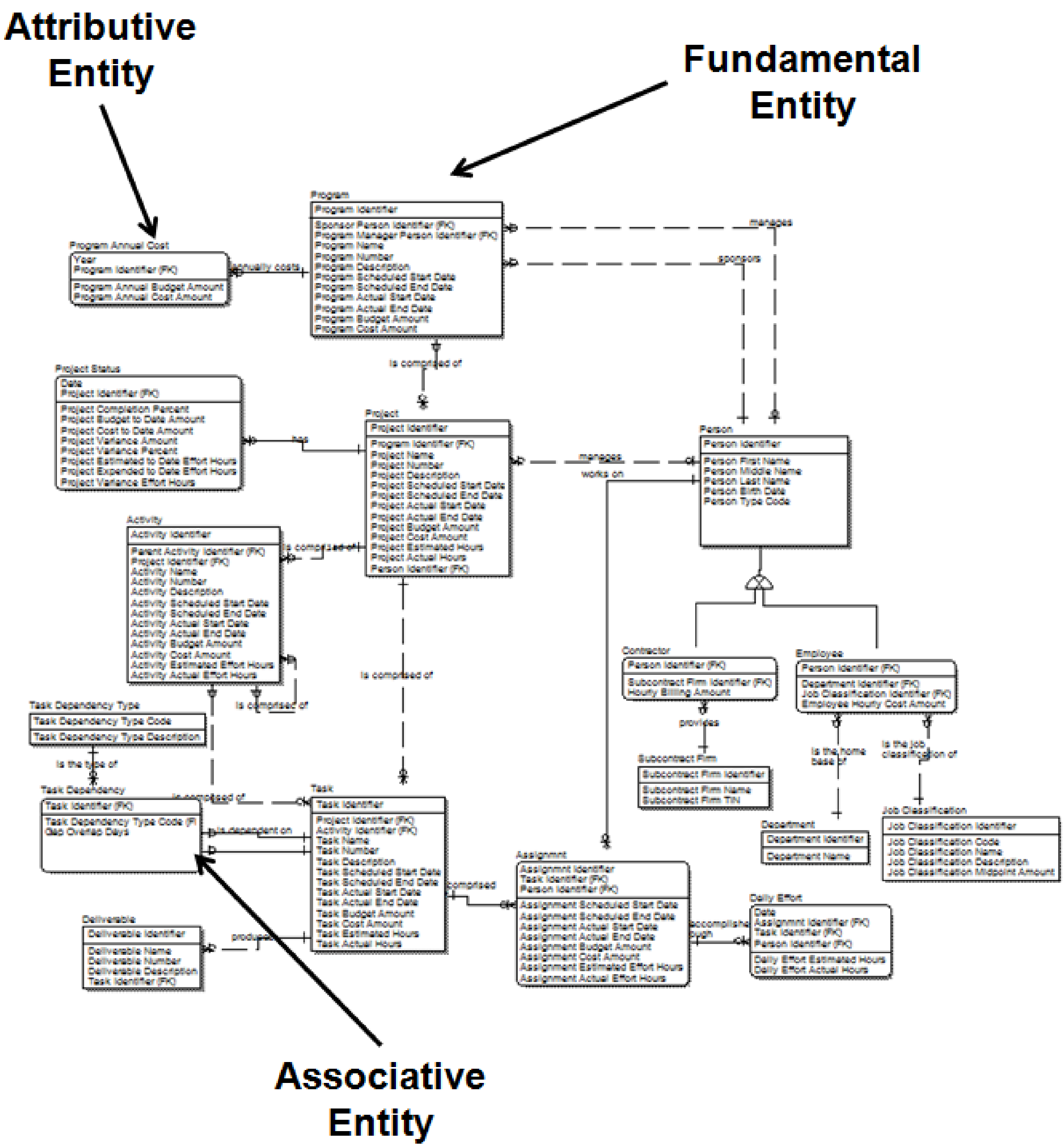
Module 2

Business Data Model Development

This page intentionally left blank.

Model Components

Entities



Model Components

Entities

DEFINITION

An entity is a person, place, concept, event, or thing of interest to the enterprise and for which it has the capability of capturing and retaining data. The caveat is not technically part of the definition, but it is included because of the need to approach data modeling from a pragmatic viewpoint. Data modeling is not a theoretical exercise. The data model is built to ultimately drive the design of a physical data structure. If the enterprise has no capability to capture the information, then it does not need a physical structure to store it. Hence, it is not worth spending the time to identify and define it. Each entity must be uniquely identifiable and exists only once within the business data model.

FUNDAMENTAL

A fundamental entity (a.k.a. independent entity) is one that does not depend on any other entity for its existence. Examples include:

- Employee
- Department
- Customer

ATTRIBUTIVE

An attributive entity (a.k.a. characteristic entity) is one that depends on another entity for its existence. Its primary key is a concatenated key that includes the primary key of the parent entity (entity on which it depends). It is created to resolve repeating groups within an entity. Examples include:

- Employee Address
- Dealer Objective

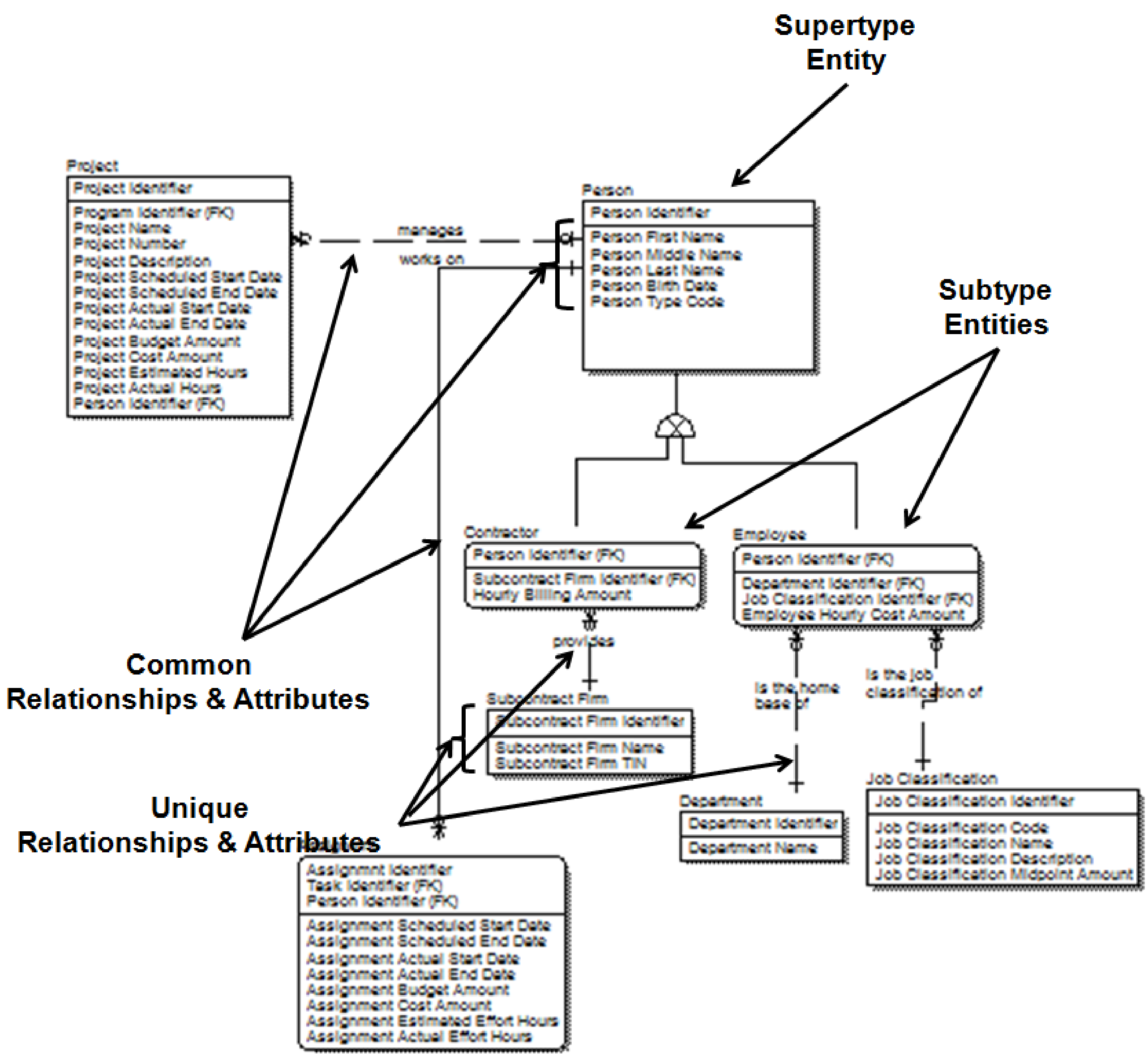
ASSOCIATIVE

An associative entity (a.k.a. intersection entity) is one that depends on at least two other entities for its existence. Its primary key is a concatenated key that includes the primary keys of each of the parent entities. An additional key component may or may not exist. This entity is formed by resolving many-to-many relationships and it contains information that is at the intersection of the parent entities. Examples include:

- Inventory
- Customer Interaction

Model Components

Entities (Continued) - Supertype and Subtype



Model Components

Entities (Continued) – Supertype and Subtype

DEFINITION

Supertype and subtype entities enable the model to represent generalizations and specializations. The supertype entity represents the highest level of abstraction that is useful in the model. For example, human resource is a supertype and employee, retiree, applicant, and contractor are subtypes.

GENERALIZATION

The supertype entity represents the highest level abstraction that is useful in the model. It includes the relationships and attributes that apply to all the subtype entities. In the above example, the human resource key, first name, middle name, last name, etc. reside in the human resource entity and apply to all of the subtypes.

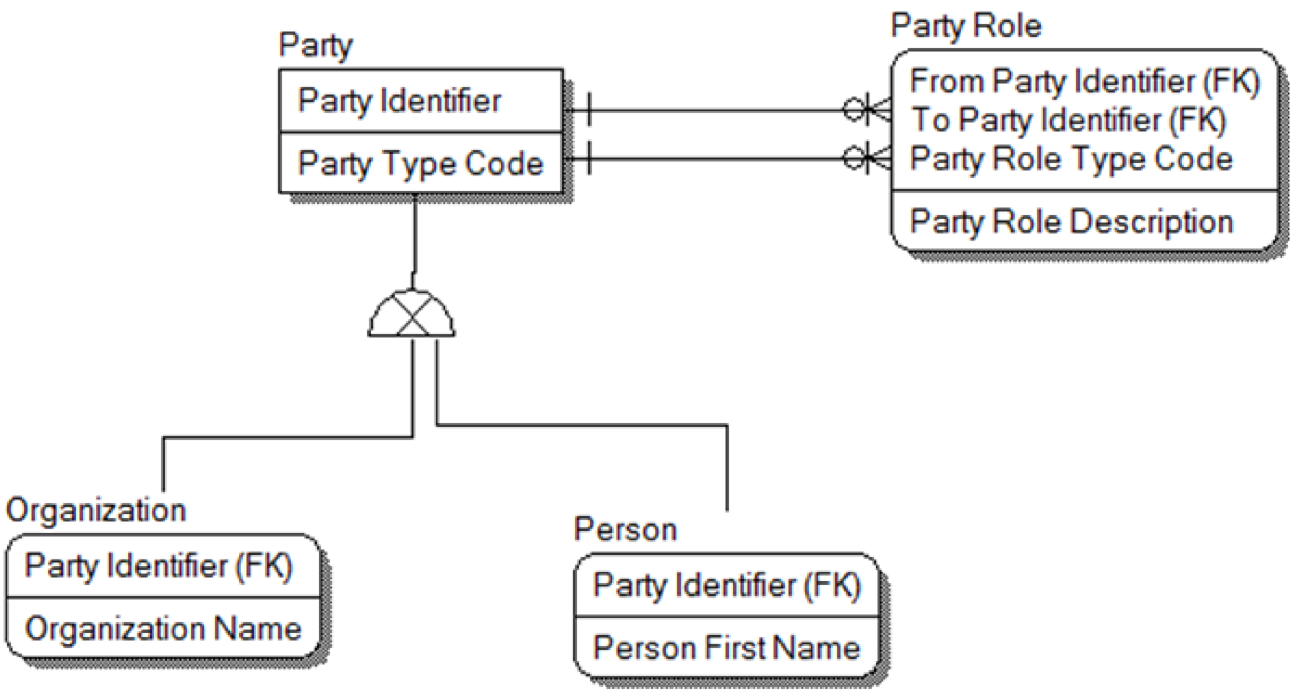
Each supertype also contains an attribute describing the types of subtype groupings that it has. For example, the human resource entity would have an attribute called human resource type.

SPECIFICATION

Specification is achieved through the subtypes. Each subtype contains relationships and attributes that are unique to it. For example there may be a relationship between employee and department, while that relationship would not apply to the other subtypes. Similarly, a retiree would have a retirement date, but the other subtypes would not.

Model Components

Entities (Continued) – Supertype and Subtype



Model Components

Entities (Continued) – Supertype and Subtype

OVERVIEW

Within business intelligence, supertypes and subtypes often get transformed into dimensions within the star schema. Care must be taken to ensure that the relationships are structured to provide correct results. This requires distinguishing between two types of subtypes – classifications and roles.

CLASSIFICATIONS

Classifications are mutually exclusive subtypes. For example, a Customer may have subtypes of Residential Customer and Commercial Customer. If the business rule is that the same customer cannot be both a residential and commercial customer, these subtypes are classifications.

For classifications, the sum of the number of occurrences of each of the subtypes is the same as the number of occurrences of the supertype.

ROLES

Roles are subtypes that are not mutually exclusive. In the human resource example, if a retiree can simultaneously also be a contractor, the subtypes are roles.
For roles, the sum of the number of occurrences of each of the subtypes is greater than or equal to the number of occurrences of the supertype. In building data marts, one must be very careful with dimensions that are roles to avoid duplicate counts.

PARTY

A modeling generalization approach is to use the concept of “Party”, which is defined as a person or organization. With this approach, the role of the party (e.g., supplier, customer, employee) is a separate (associative) entity which relates two parties. The advantage of this approach is that information about the party is shown only once regardless of the role. The major disadvantage is that the level of generalization is sometimes too high to effectively communicate the rules indicated by the model.

A compromise that is sometimes made is to represent parties explicitly within the business model and to transform these into the party concept for the system or technology models.

Module 3

System and Physical Data Model Development

This page intentionally left blank.

Data Modeling Roles

Data Analyst / Data Modeler

WHO:	IT person responsible for developing and maintaining data models
ROLE:	Lead data model development Understand data in scope and business access expectations Transform business data model into system model
TRAITS:	Recognition of data as an asset Data modeling expertise (including tools) Analysis capabilities
CHALLENGES:	Determining the type of model and degree of denormalization Knowing when to stop

Data Modeling Roles

Data Analyst / Data Modeler

WHO ARE THESE

The data analyst / data modeler is the person who transforms the business data model into the systems model and supports the database administrator in creating the technology model.

MODEL DEVELOPMENT ROLE

The data analyst works with the development team to understand the scope of the effort. For business intelligence projects, the emphasis is on understanding the data within scope and how the business user expects to access or use the data. Armed with this information, the data analyst can create the starting point for the system data model and then proceed to fully develop it for the scope of the initiative.

HOW TO ENCOURAGE INVOLVEMENT

The data analyst's involvement typically does not need to be encouraged. This is the person's job.

HOW TO OVERCOME ABSENCE

Sometimes other priorities preclude involvement by the data analyst. When this happens, the project leader should try to emphasize the importance of the data analyst's involvement. There may be opportunities to shift project activities to accommodate the data analyst's priorities and still complete the project on time.

Failing that, the project leader should try to enlist someone else with the requisite skills. Often database administrators can also create the system data model.

Denormalization

Overview

DENORMALIZATION ACTIVITIES

- Inclusion of reference data in transactions
  - Customer name within sales transaction
- Inclusion of same element in multiple tables
  - Product name in tables that refer to product
- Creation of aggregations and summaries
  - To help migrate data to data marts
- Compression of hierarchies into a single table
  - To create star schema
- Creation of tables with data commonly used together
  - To minimize joins upon retrieval

DENORMALIZATION TRADE-OFFS

- Development Time
  - Data structure design
  - Design, coding, and testing to reliably replicate data
- Processing
  - Extra processing needed to place same data in multiple places
  - Reduction in joins and full table scans
- Storage
  - Duplicate storage consumes disk space
- Consistency
  - Vigilance needed to ensure synchronization

Denormalization

Overview

DESCRIPTION

Denormalization is the process of including redundant data within a database to improve performance and usability. Within traditional systems, data is denormalized to improve transaction processing performance. Within the business intelligence environment, denormalization is performed to improve query performance and data navigation. Examples of denormalization include:

- Inclusion of reference data (e.g., customer name) within transaction tables
- Inclusion of the same data element in multiple tables
- Creation of aggregations and summaries
- Compression of hierarchies into a single table
- Creation of tables with data that is commonly used together

COST

Denoromalization improves query performance by reducing the data joins and full-table scans needed to retrieve the data. This improvement requires costs during both development and operation of a system. Typical costs include:

- Development – the denormalization steps need to be designed and the code to reliably replicate the data needs to be built and tested
- Processing – it requires extra processing to place the same data element in multiple places
- Storage – duplicate storage consumes disk space
- Consistency – duplicating data requires careful vigilance to ensure that all copies of the data are updated simultaneously

DESIGN CONSIDERATIONS

In the business intelligence environment, the primary consideration for denormalization is the data access requirement and the design of the information delivery layer. That layer typically includes dimensional models, and these structures are inherently denormalized. Each dimension contains a hierarchy, with the hierarchy compressed into a single table. In addition, the fact table is sometimes at an aggregated level.

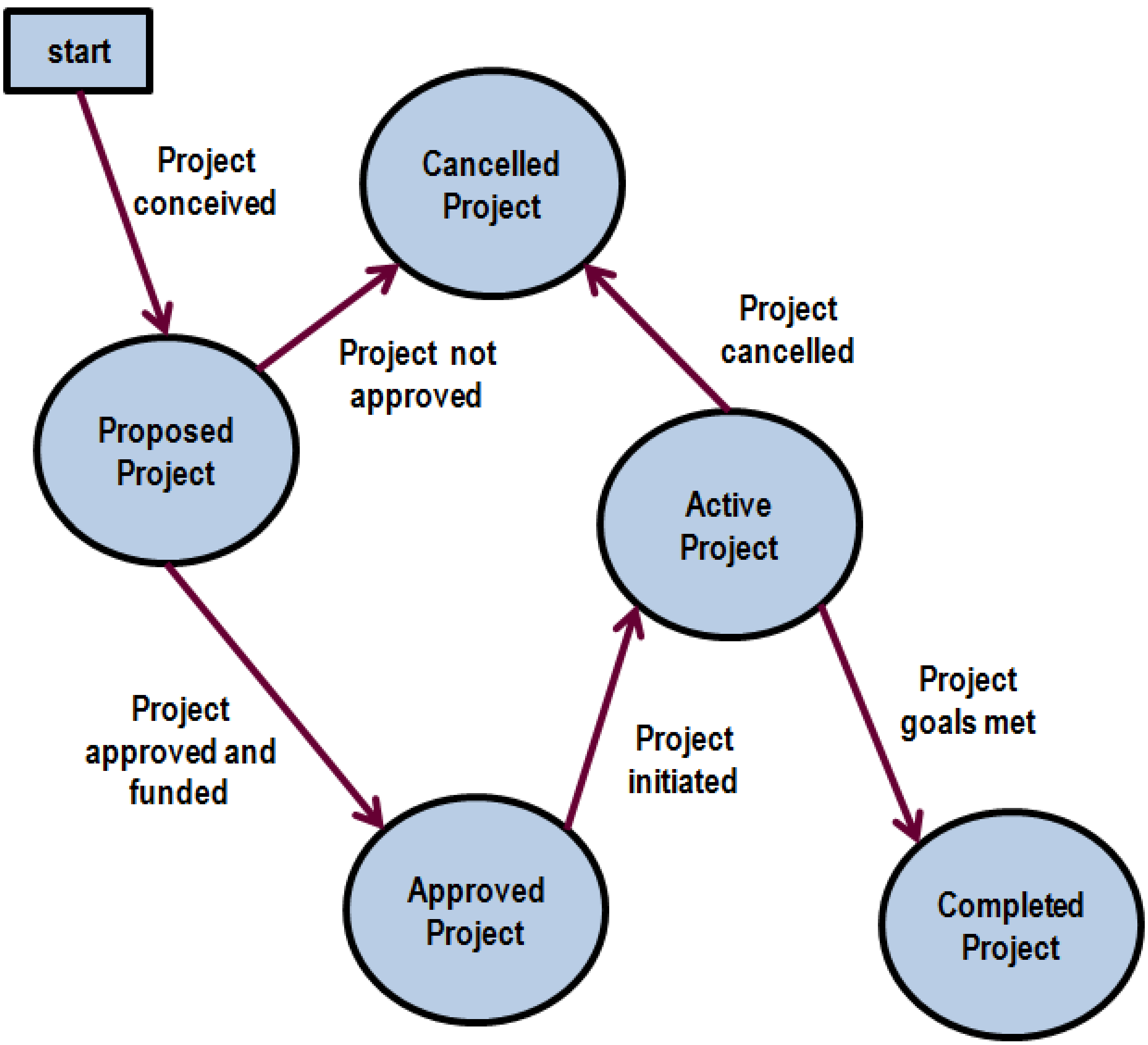
In a hub and spoke environment, aggregation tables, within the data warehouse, are often built to facilitate building the data marts.

Module 4

Additional Concepts

Complementary Models

State Transition Model



Complementary Models

State Transition Model

DESCRIPTION

The state transition model is a way of depicting a single entity throughout its lifecycle. It provides information about the entity at each state and also describes the actions needed for it to change from one state to another. For example, a sale has several states, including proposal, order, fulfillment, delivery, and payment. The state transition model would depict each of these states and indicate what is necessary for the sale to move from one state to the next, recognizing that there may be options. For example in the lifecycle of a loan, after the application state, the loan may proceed to an approved state or a rejected state.

COMPONENTS

The model consists of the following major components:

- States – the unique set of business circumstances in which an occurrence of an entity may exist
- Actions – the events that must take place for the entity to move from one state to the next

In addition, terminators to indicate the start and end points of the lifecycle and descriptions of each state (including any preconditions and postconditions) should be included to complete the model.

PERSPECTIVE

The state transition model is focused on a single entity and should represent its states and the business rules governing the transitions from an enterprise perspective. Restricting the view to a single business area defeats the purpose of this model since the transitions would not be fully represented.

PROCESS

Development of this model requires the following major activities:

- Determine the entity to be modeled.
- Identify the states of interest (both linear and tangential). Name and describe each of these.
- Identify the actions that cause changes in the state.
- Identify preconditions and postconditions for each state.
- Verify correctness (and adjust if needed).

DATA MODEL SIGNIFICANCE

This model is extremely useful for traditional application development, since the actions from migrating from state to state are often within the scope of the systems and this provides a comprehensive view of what needs to be accomplished. If a state transition exists, it can be used to help validate the business data model and the way in which the supertypes and subtypes for the entity are represented