04

From OLTP to OLAP

Notice

- **Author**
  - **João Moura Pires (jmp@di.fct.unl.pt)**

Bibliography

- **Many examples are extracted and adapted from**
  - **"From Enterprise Models to Dimensional Models: A Methodology for Data Warehouse and Data Mart Design" de Daniel L. Moody e Mark A. R. Kortink**

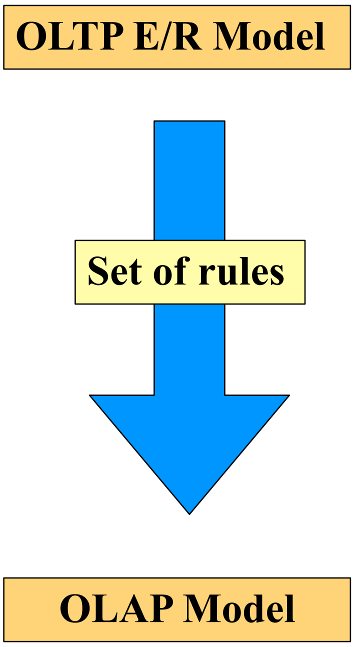
Table of Contents

- **Introduction**
- **Classification of Entities**
- **Hierarchies**
- **Production of multidimensional models**
- **Type of models**
- **Evaluation and model tuning**

Data Warehouse - Data Models

Introduction

Objective



OLTP E/R Model

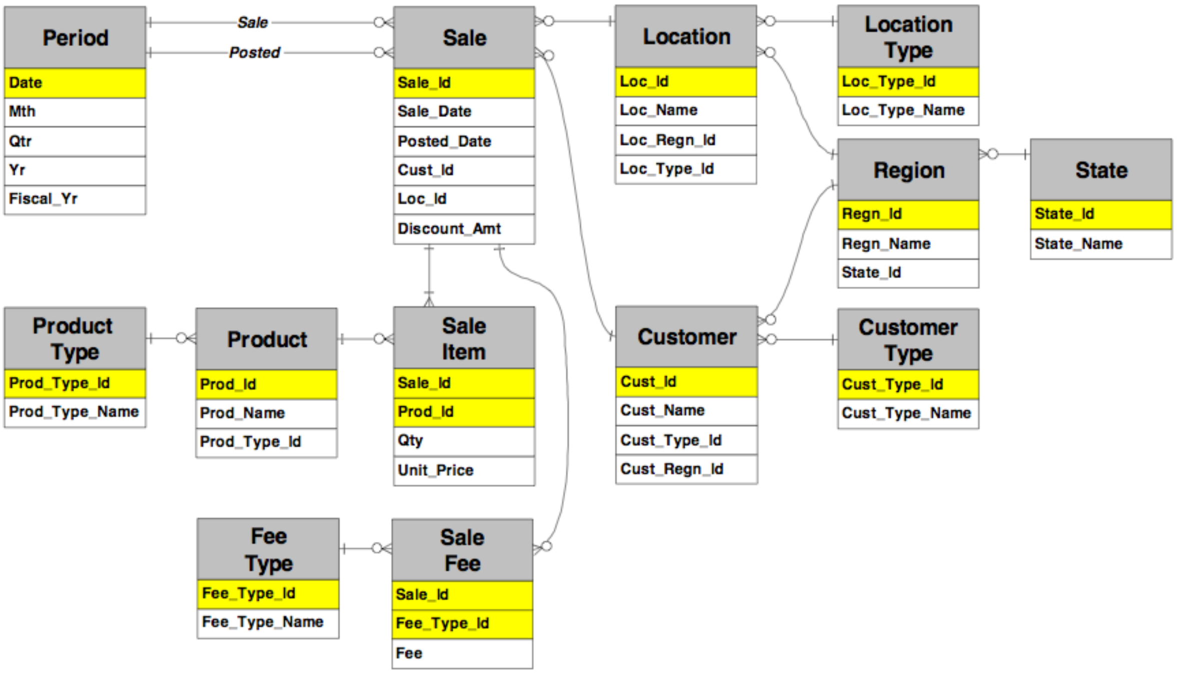
Set of rules

OLAP Model

Method in 3 steps

- **Step 1: Classify the entities of OLTP E/R model**
- **Step 2: Identify hierarchies**
- **Step 3: Produce the intended OLAP model**
  - **Flat Schema**
  - **Terraced Schema**
  - **Star Schema e Constellation Schema**
  - **Snowflake Schema**
  - **Star Cluster Schema**

Example of a “simple” OLTP



Classification of Entities

Types of Entities

- **The Entities of E/T OLTP model have to classified into one of the following categories:**
  - **Transaction Entities**
  - **Component Entities**
  - **Classification Entities**

Transaction Entities

- **Transaction Entities: the business events**
  - **Record details about particular events that occur at a point in time.**
  - **It contains measurements or quantities that may be summarized e.g. dollar amounts, weights, volumes.**
  - **They are the base for fact tables**

- **Notes and examples**
  - **Examples: Orders, insurance claims, salary payments and hotel bookings.**
  - **It is these events that decision makers want to understand and analyze. Not all transaction entities will be of interest for decision support.**

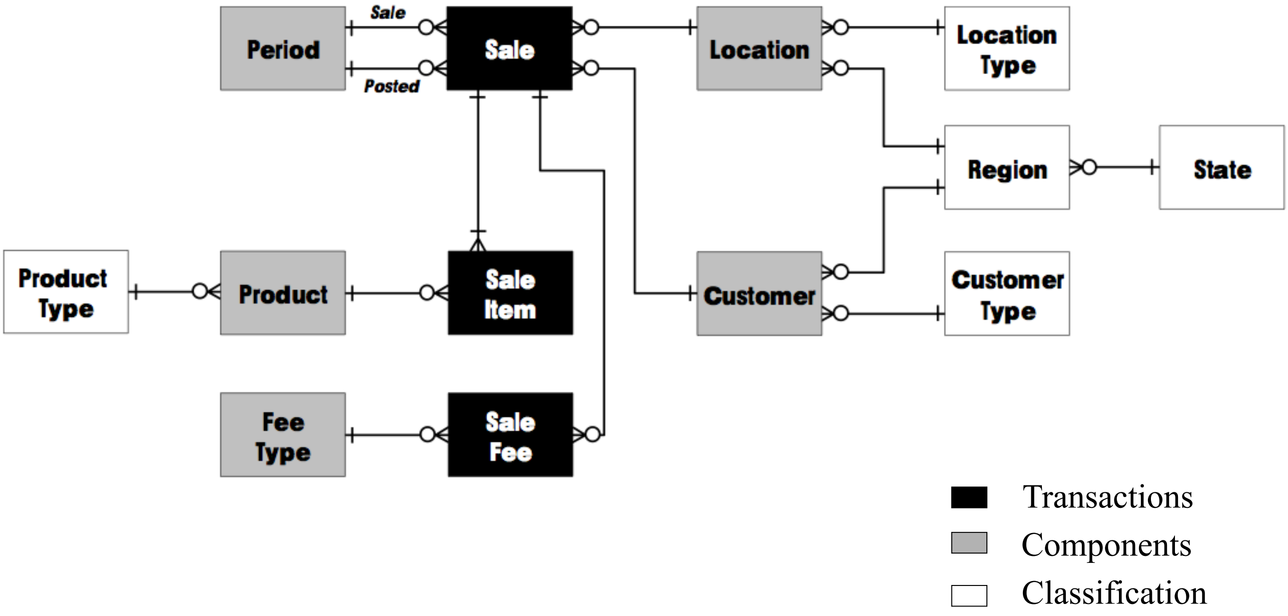
Component Entities

- **Component Entities**
  - **A component entity is one which is directly related to a transaction entity via a one-to-many relationship.**
  - **These entities define the details or components for each transaction**
    - Component entities answer the “**who**”, “**what**”, “**when**”, “**where**”, “**how**” and “**why**” of a business event
    - For example, a **sales transaction** may be defined by a number of components:
      - Customer: who made the purchase
      - Product: what was sold
      - Location: where it was sold
      - Period: when it was sold
  - **Component form the basis for constructing dimension tables in star-schemas.**

Classification Entities

- **Classification Entities**
  - **Classification entities are entities which are related to component entities by a chain of one-to-many relationships**
  - **They are functionally dependent on a component entity (directly or transitively).**
  - **Classification entities represent hierarchies embedded in the data model, which may be collapsed into component entities to form dimension tables in a star schema.**

Classification of Entitles

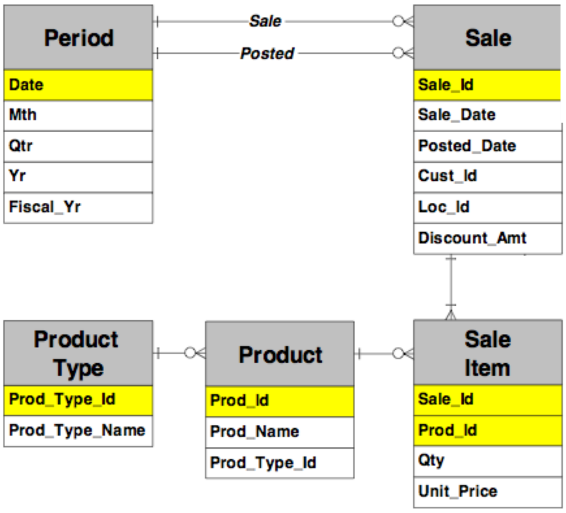


Resolving Ambiguities

- In some cases, entities may fit into multiple categories. A precedence hierarchy for resolving such ambiguities:
  - 1. Transaction entity (highest precedence)
  - 2. Classification entity
  - 3. Component entity (lowest precedence)

- In practice, some entities will not fit into any of these categories. Such entities do not fit the hierarchical structure of a dimensional model and cannot be included in the star-schema

Resolving Ambiguities



1. Sale item is considered as a Transaction entity
2. Since Sale is directly related to Sale Item by a 1:N relationship, Sale can be considered as a Component entity (of Sale Item).
3. Sale it self can be considered as a Transaction entity
4. Considering the rules for resolving ambiguities, Sale is classified as Transactional.

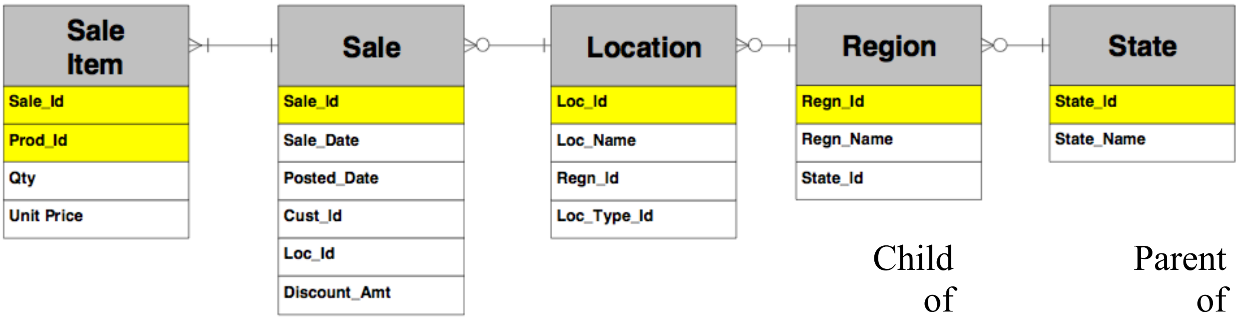
Hierarchies

Hierarchies

- A hierarchy in an Entity Relationship model is any sequence of entities joined together by one-to-many relationships, all aligned in the same direction.

BOTTOM

TOP



Sale Item: Sale_Id, Prod_Id, Qty, Unit Price

Sale: Sale_Id, Sale_Date, Posted_Date, Cust_Id, Loc_Id, Discount_Amt

Location: Loc_Id, Loc_Name, Regn_Id, Loc_Type_Id

Region: Regn_Id, Regn_Name, State_Id

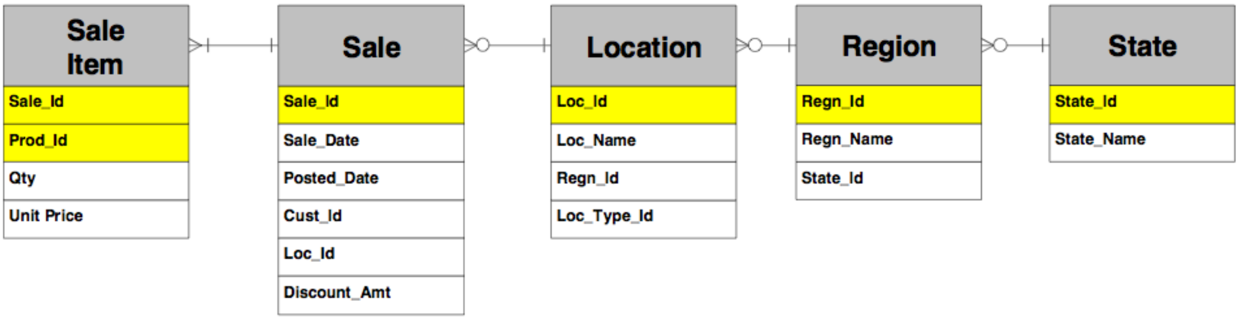
State: State_Id, State_Name

Child of State

Parent of Region

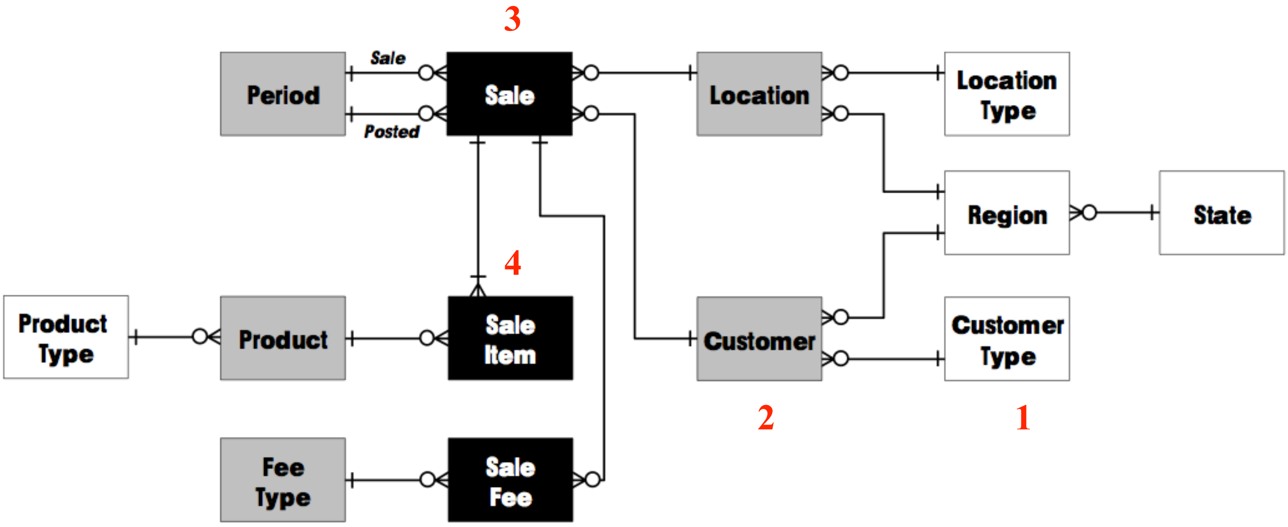
Maximal Hierarchy

- A hierarchy is called maximal if it cannot be extended upwards or downwards by including another entity.



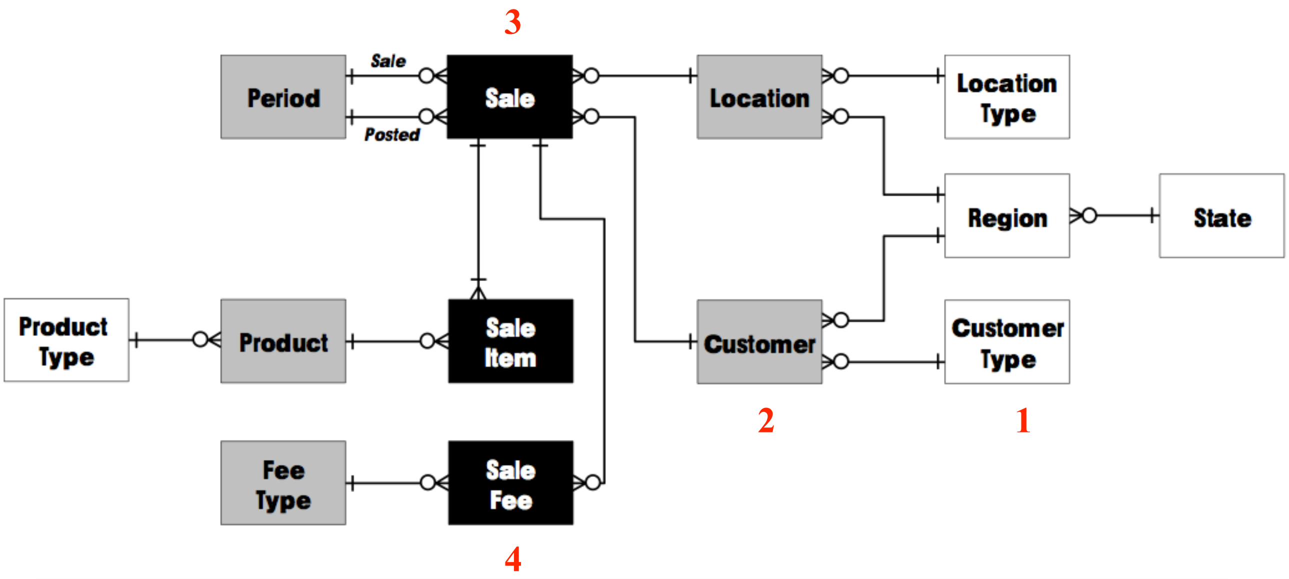
- State > Region > Location > Sale it is not maximal because it can be extend by including Sale Item

Identifying maximal hierarchies



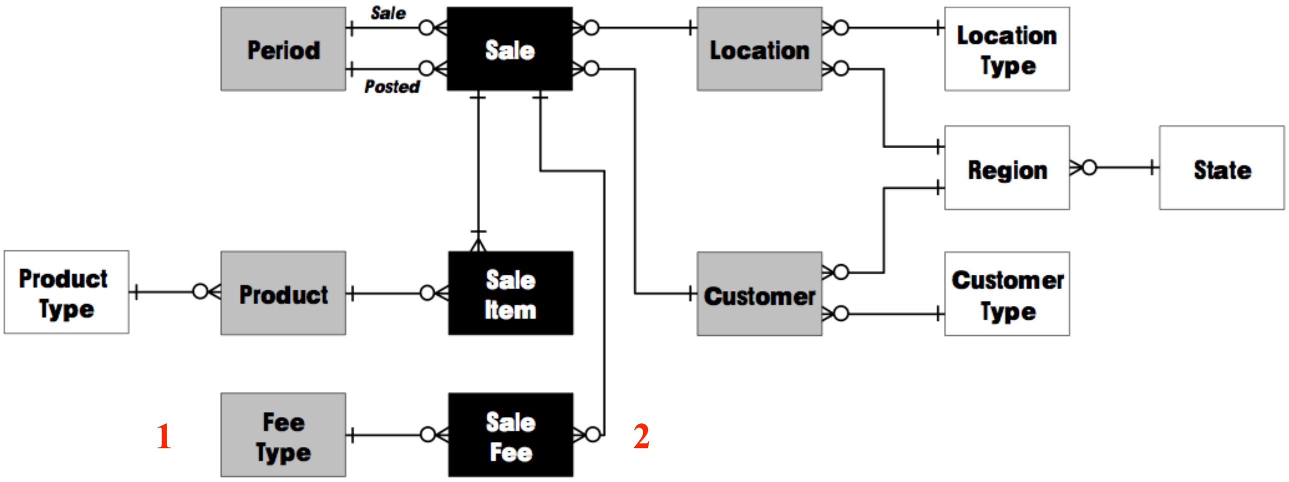
Customer Type > Customer > Sale > Sale Item >

Identifying maximal hierarchies



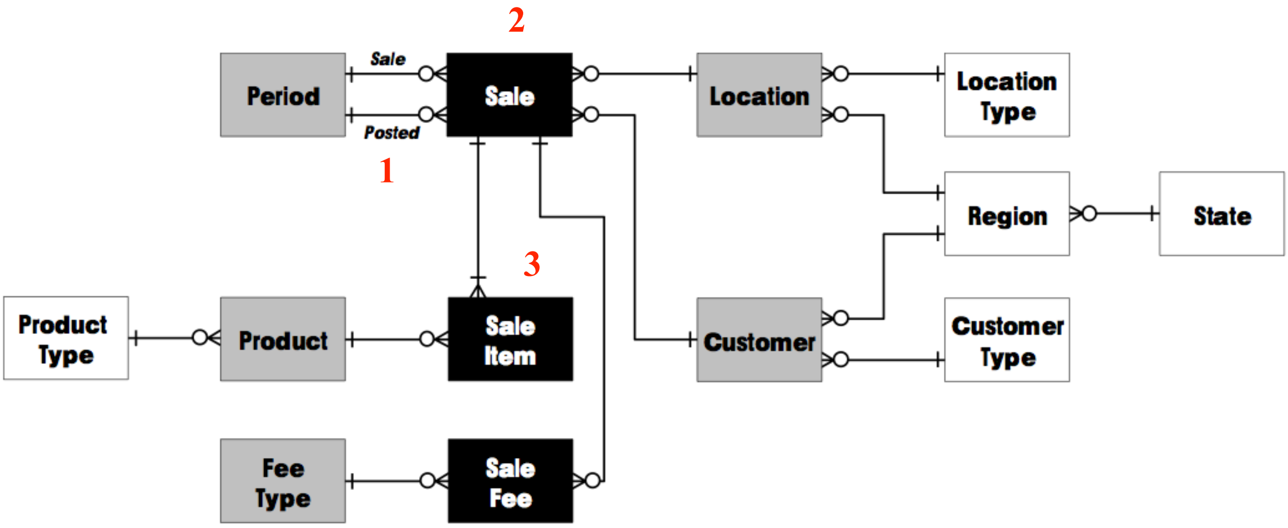
Customer Type > Customer > Sale > Sale Fee >

Identifying maximal hierarchies



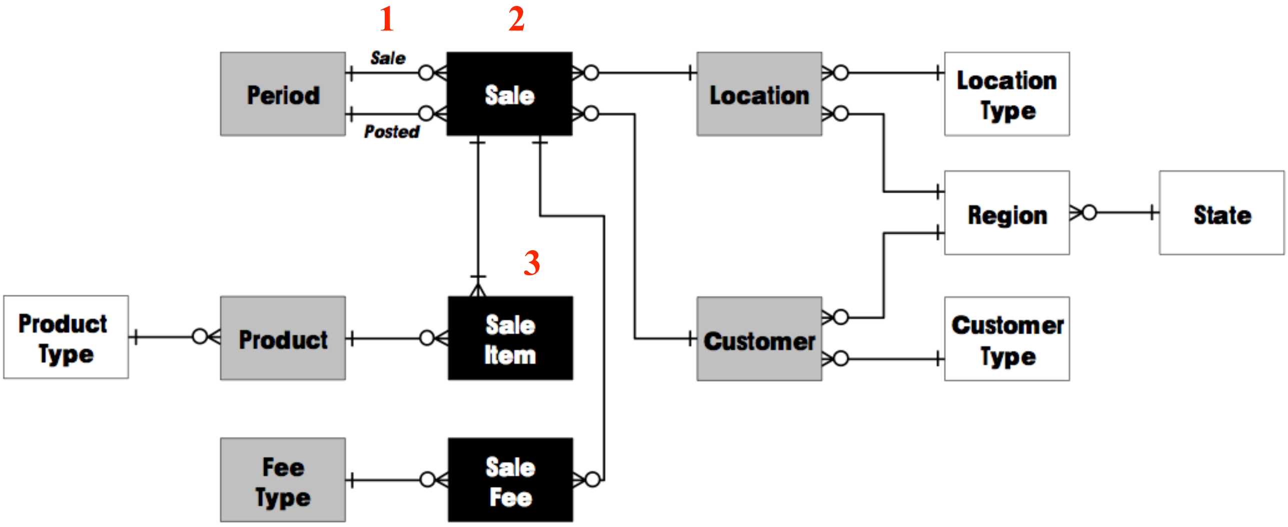
Fee Type > Sale Fee >

Identifying maximal hierarchies



Period (posted) > Sale > Sale Item >

Identifying maximal hierarchies



Period (sale) > Sale > Sale Item >

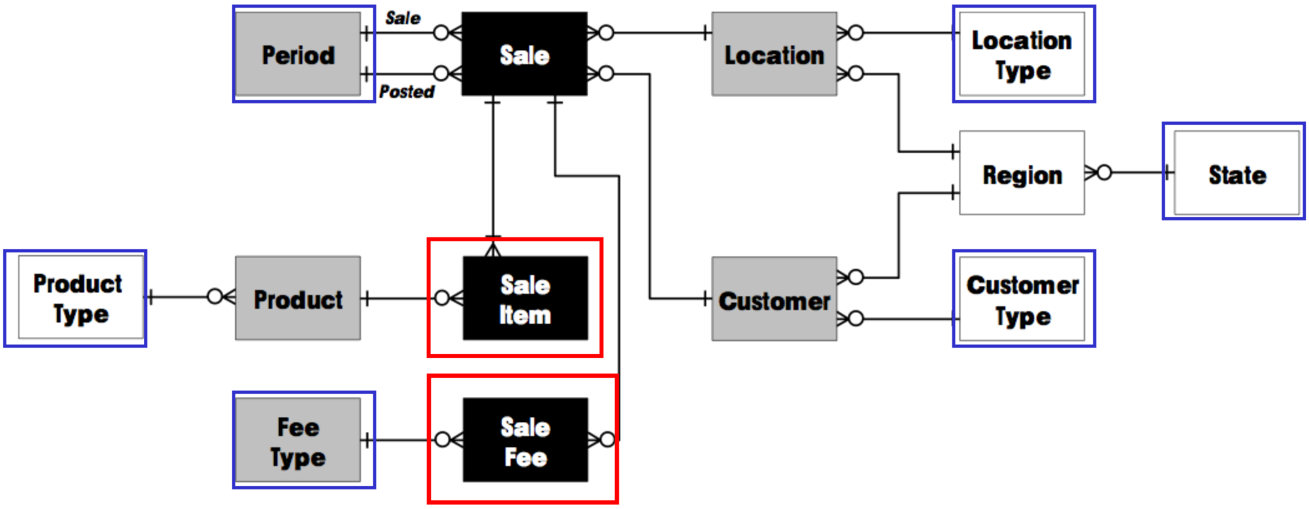
14 maximal hierarchies

- Customer type > Customer > Sale > Sale Item (ou Sale > Sale Fee) (2)
- Fee Type > Sale Fee
- Location Type > Location > Sale > Sale Item (ou Sale > Sale Fee) (2)
- Period (posted) > Sale > Sale Item (ou Sale > Sale Fee) (2)
- Period (sale) > Sale > Sale Item (ou Sale > Sale Fee) (2)
- Product Type > Product > Sale Item
- State > Region > Customer > Sale > Sale Item (ou Sale > Sale Fee) (2)
- State > Region > Location > Sale > Sale Item (ou Sale > Sale Fee) (2)

Minimal and Maximal Entities

- An entity is called minimal if it is at the bottom of a maximal hierarchy and maximal if it is at the top of one.
- Minimal entities can be easily identified as they are entities with no one-to-many relationships (or “leaf” entities in hierarchical terminology)
- Maximal entities are entities with no many to one relationships (or “root” entities).

Minimal and Maximal Entities



14 maximal hierarchies

- Customer type > Customer > Sale > Sale Item (ou Sale > Sale Fee) (2)
- Fee Type > Sale Fee
- Location Type > Location > Sale > Sale Item (ou Sale > Sale Fee) (2)
- Period (posted) > Sale > Sale Item (ou Sale > Sale Fee) (2)
- Period (sale) > Sale > Sale Item (ou Sale > Sale Fee) (2)
- Product Type > Product > Sale Item
- State > Region > Customer > Sale > Sale Item (ou Sale > Sale Fee) (2)
- State > Region > Location > Sale > Sale Item (ou Sale > Sale Fee) (2)

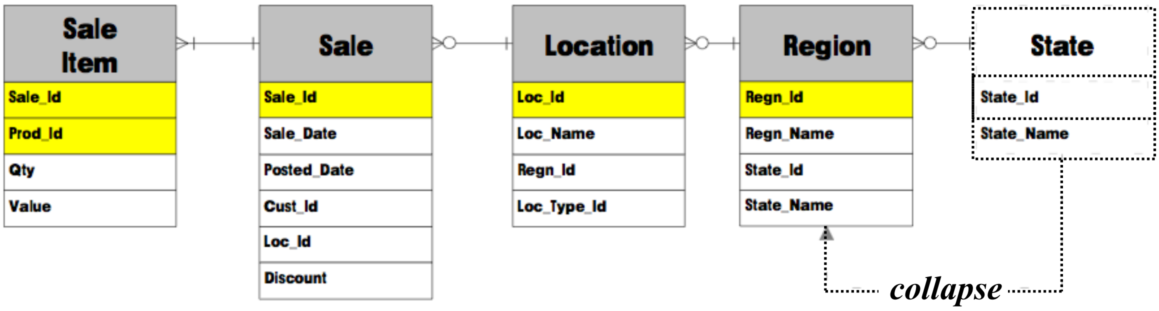
Production of multidimensional models

Production of multidimensional models

- **Two operators to produce dimensional models from Entity Relationship models:**
  - **Collapse Hierarchy**
    - Higher level entities can be “collapsed” into lower level entities within hierarchies.
  - **Aggregation**
    - The aggregation operator can be applied to a transaction entity to create a new entity containing summarized data.

Collapse Hierarchy

- Higher level entities can be “collapsed” into lower level entities within hierarchies.



Sale Item: Sale_Id, Prod_Id, Qty, Value

Sale: Sale_Id, Sale_Date, Posted_Date, Cust_Id, Loc_Id, Discount

Location: Loc_Id, Loc_Name, Regn_Id, Loc_Type_Id

Region: Regn_Id, Regn_Name, State_Id, State_Name

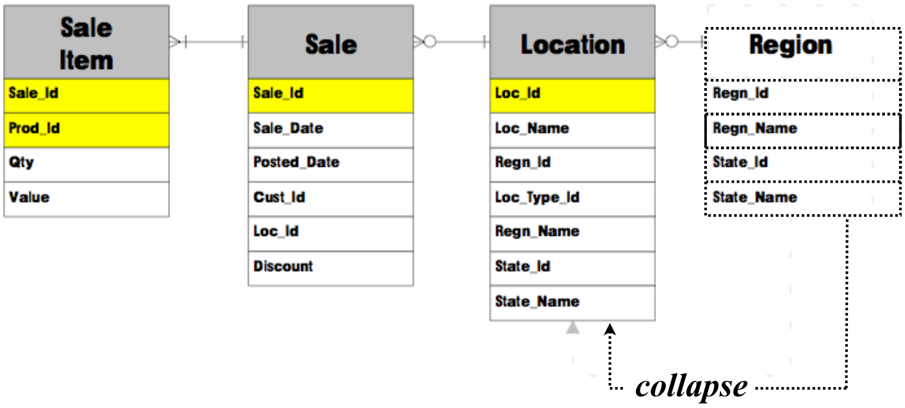
State: State_Id, State_Name

collapse

- The State entity being collapsed into the Region entity. The Region entity contains its original attributes plus the attributes of the collapsed table.
- Collapsing a hierarchy is therefore a form of denormalization.

Collapse Hierarchy

- We can continue doing this until we reach the bottom of the hierarchy, and end up with a single table (in this example Sale Item).



Sale Item: Sale_Id, Prod_Id, Qty, Value

Sale: Sale_Id, Sale_Date, Posted_Date, Cust_Id, Loc_Id, Discount

Location: Loc_Id, Loc_Name, Regn_Id, Loc_Type_Id, Regn_Name, State_Id, State_Name

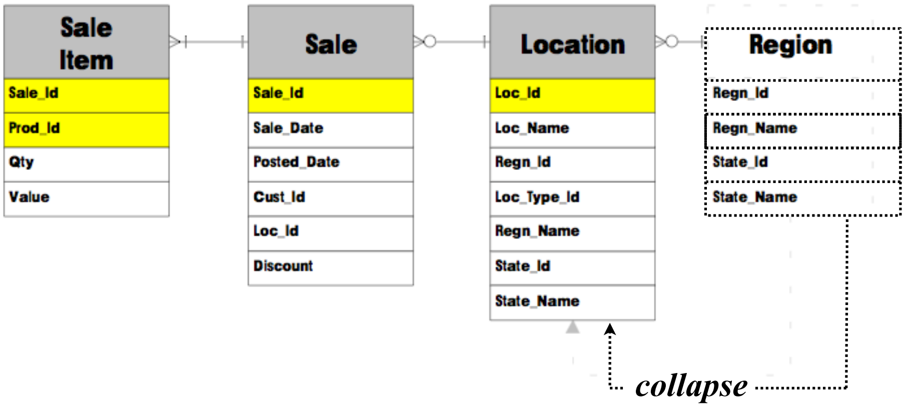
Region: Regn_Id, Regn_Name, State_Id, State_Name

collapse

- The number of rows remains the same before the operation. In this case the number of rows of table Location.

Collapse Hierarchy

- We can continue doing this until we reach the bottom of the hierarchy, and end up with a single table (in this example Sale Item).



Sale Item: Sale_Id, Prod_Id, Qty, Value

Sale: Sale_Id, Sale_Date, Posted_Date, Cust_Id, Loc_Id, Discount

Location: Loc_Id, Loc_Name, Regn_Id, Loc_Type_Id, Regn_Name, State_Id, State_Name

Region: Regn_Id, Regn_Name, State_Id, State_Name

collapse

- The number of rows remains the same before the operation. In this case the number of rows of table Location.

Aggregation

- The aggregation operator can be applied to a transaction entity to create a new entity containing summarized data.

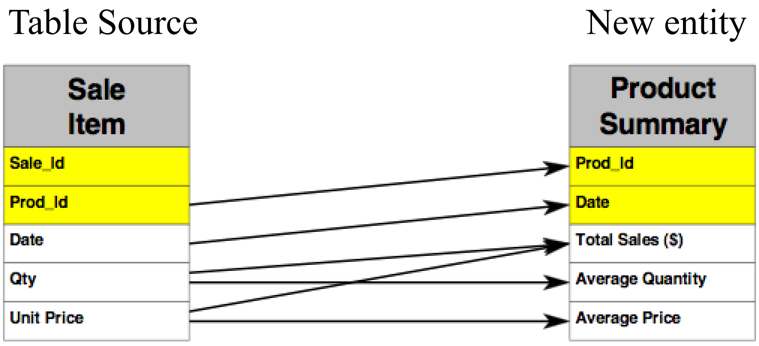
- A subset of attributes is chosen from the source entity to aggregate (the aggregation attributes). Aggregation attributes must be numerical quantities.

- Another subset of attributes chosen to aggregate by (the grouping attributes).

- Note that aggregation *loses information*: we cannot reconstruct the details of individual sale items from the product summary

Aggregation

- **This aggregated entity shows for each product the total sales amount (quantity*price), the average quantity per order and average price per item on a daily basis**



Aggregation attributes - Qty, Unit Price Grouping attributes - Prod. ID, Date

Data Warehouse - Data Models

Type of models

Dimensional Design Options

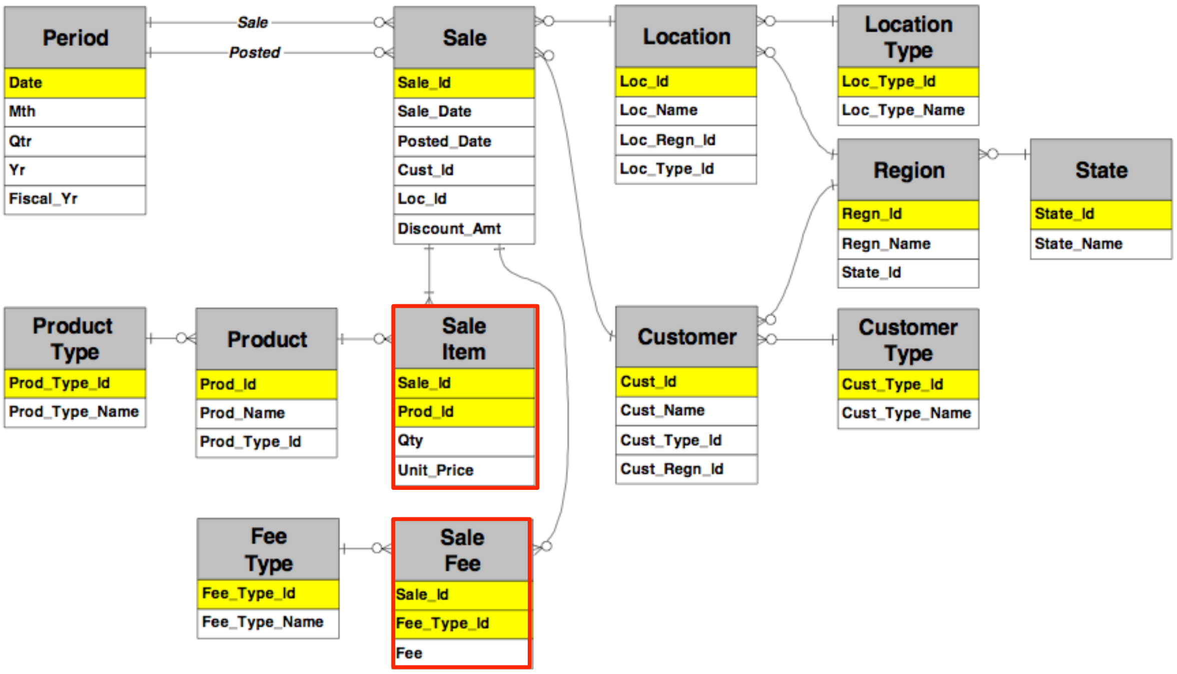
- **Threshold between complexity and redundancy**

- **Flat Schema**
- **Terraced Schema**
- **Star Schema**
- **Constellation Schema and Galaxy**
- **Snowflake Schema**
- **Star Cluster Schema**

Flat Schema

- A flat schema is the simplest schema possible without losing information. This is formed by collapsing all entities in the data model down into the minimal entities.
- **This minimizes the number of tables in the database and therefore the possibility that joins will be needed in user queries. In a flat schema we end up with one table for each minimal entity in the original data model.**
- **Such a schema is similar to the "flat files" used by analysts using statistical packages.**
- **One problem with a flat schema is that it may lead to aggregation errors when there are hierarchical relationships between transaction entities (Sales and Sales Item)**

Minimal Entities



Period
Date
Mth
Qtr
Yr
Fiscal_Yr

Sale
Posted

Sale
Sale_Id
Sale_Date
Posted_Date
Cust_Id
Loc_Id
Discount_Amt

Location
Loc_Id
Loc_Name
Loc_Regn_Id
Loc_Type_Id

Location Type
Loc_Type_Id
Loc_Type_Name

Region
Regn_Id
Regn_Name
State_Id

State
State_Id
State_Name

Product Type
Prod_Type_Id
Prod_Type_Name

Product
Prod_Id
Prod_Name
Prod_Type_Id

Sale Item
Sale_Id
Prod_Id
Qty
Unit_Price

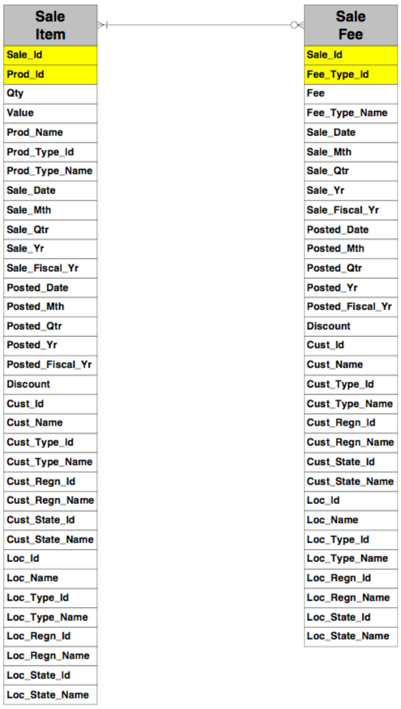
Customer
Cust_Id
Cust_Name
Cust_Type_Id
Cust_Regn_Id

Customer Type
Cust_Type_Id
Cust_Type_Name

Fee Type
Fee_Type_Id
Fee_Type_Name

Sale Fee
Sale_Id
Fee_Type_Id
Fee

Flat Schema



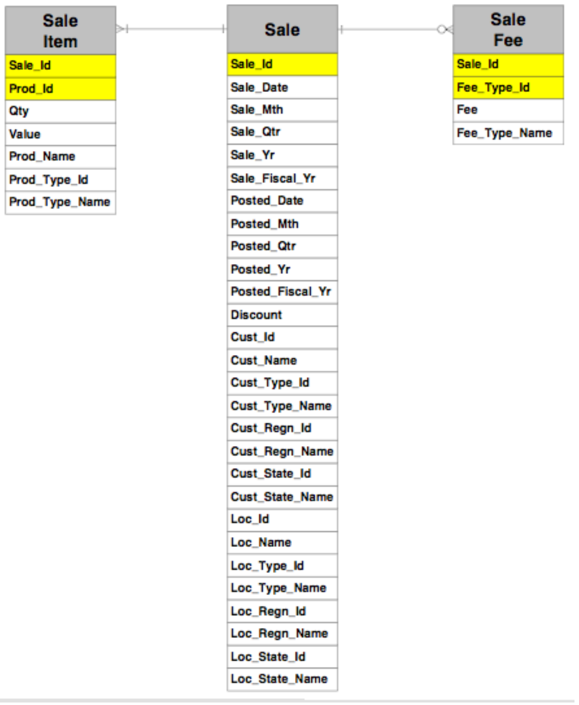
Sale Item
Sale_Id
Prod_Id
Qty
Value
Prod_Name
Prod_Type_Id
Prod_Type_Name
Sale_Date
Sale_Mth
Sale_Qtr
Sale_Yr
Sale_Fiscal_Yr
Posted_Date
Posted_Mth
Posted_Qtr
Posted_Yr
Posted_Fiscal_Yr
Discount
Cust_Id
Cust_Name
Cust_Type_Id
Cust_Type_Name
Cust_Regn_Id
Cust_Regn_Name
Cust_State_Id
Cust_State_Name
Loc_Id
Loc_Name
Loc_Type_Id
Loc_Type_Name
Loc_Regn_Id
Loc_Regn_Name
Loc_State_Id
Loc_State_Name

Sale Fee
Sale_Id
Fee_Type_Id
Fee
Fee_Type_Name
Sale_Date
Sale_Mth
Sale_Qtr
Sale_Yr
Sale_Fiscal_Yr
Posted_Date
Posted_Mth
Posted_Qtr
Posted_Yr
Posted_Fiscal_Yr
Discount
Cust_Id
Cust_Name
Cust_Type_Id
Cust_Type_Name
Cust_Regn_Id
Cust_Regn_Name
Cust_State_Id
Cust_State_Name
Loc_Id
Loc_Name
Loc_Type_Id
Loc_Type_Name
Loc_Regn_Id
Loc_Regn_Name
Loc_State_Id
Loc_State_Name

- **When we collapse numerical amounts from higher level transaction entities into another they will be repeated.**
- **In the example data model, if a Sale consists of three Sale Items, the discount amount will be stored in three different rows in the Sale Item table. Adding the discount amounts together then results in double-counting (or in this case, triple)**

Terraced Schema

- **A terraced schema is formed by collapsing entities down maximal hierarchies, but stopping when they reach a transaction entity.**
- **This results in a single table for each transaction entity in the data model, providing by the way a separation of transactional levels**

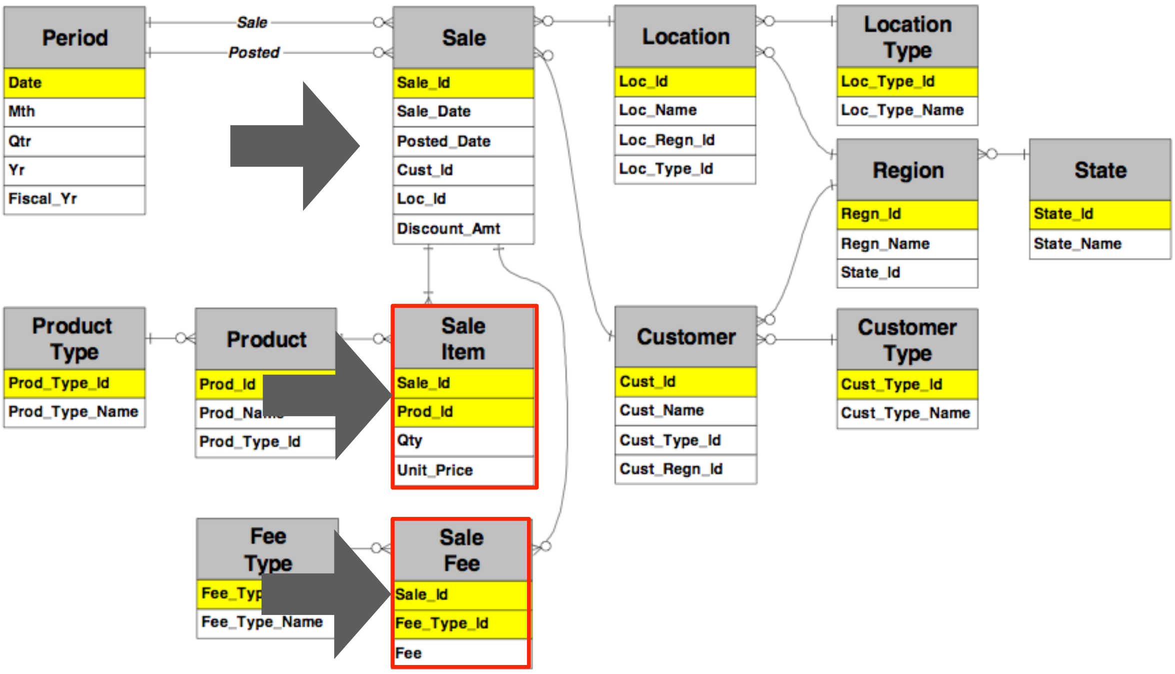


Sale Item
Sale_Id
Prod_Id
Qty
Value
Prod_Name
Prod_Type_Id
Prod_Type_Name

Sale
Sale_Id
Sale_Date
Sale_Mth
Sale_Qtr
Sale_Yr
Sale_Fiscal_Yr
Posted_Date
Posted_Mth
Posted_Qtr
Posted_Yr
Posted_Fiscal_Yr
Discount
Cust_Id
Cust_Name
Cust_Type_Id
Cust_Type_Name
Cust_Regn_Id
Cust_Regn_Name
Cust_State_Id
Cust_State_Name
Loc_Id
Loc_Name
Loc_Type_Id
Loc_Type_Name
Loc_Regn_Id
Loc_Regn_Name
Loc_State_Id
Loc_State_Name

Sale Fee
Sale_Id
Fee_Type_Id
Fee
Fee_Type_Name

Transaction entities



Period
Date
Mth
Qtr
Yr
Fiscal_Yr

Sale
Sale_Id
Sale_Date
Posted_Date
Cust_Id
Loc_Id
Discount_Amt

Location
Loc_Id
Loc_Name
Loc_Regn_Id
Loc_Type_Id

Location Type
Loc_Type_Id
Loc_Type_Name

Region
Regn_Id
Regn_Name
State_Id

State
State_Id
State_Name

Product Type
Prod_Type_Id
Prod_Type_Name

Product
Prod_Id
Prod_Na
Prod_Type_Id

Sale Item
Sale_Id
Prod_Id
Qty
Unit_Price

Customer
Cust_Id
Cust_Name
Cust_Type_Id
Cust_Regn_Id

Customer Type
Cust_Type_Id
Cust_Type_Name

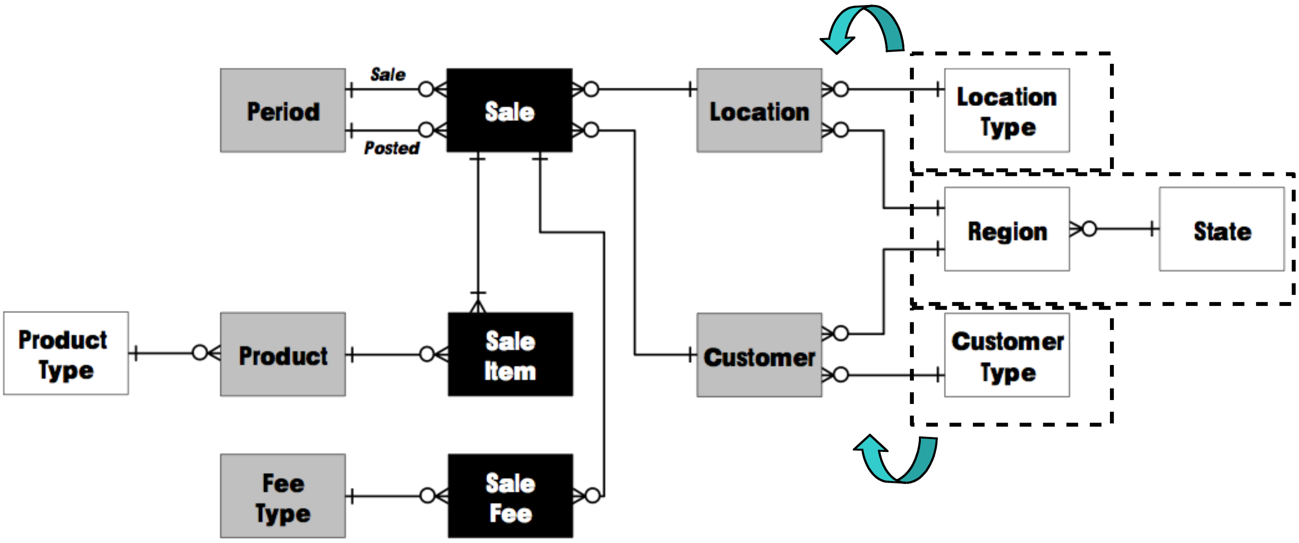
Fee Type
Fee_Typ
Fee_Type_Name

Sale Fee
Sale_Id
Fee_Type_Id
Fee

Star Schema

- Each star schema is formed in the following way:
  - A fact table is formed for each transaction entity. The key of the table is the combination of the keys of its associated component entities.
  - A dimension table is formed for each component entity, by collapsing hierarchically related classification entities into it.
  - Where hierarchical relationships exist between transaction entities, the child entity inherits all dimensions (and key attributes) from the parent entity. This provides the ability to "drill down" between transaction levels.
  - Numerical attributes within transaction entities should be aggregated by key attributes (dimensions). The aggregation attributes and functions used depend on the application.

Derivation of Sale Star Schema



Period

Sale

Posted

Sale

Location

Location Type

Region

State

Product Type

Product

Sale Item

Customer

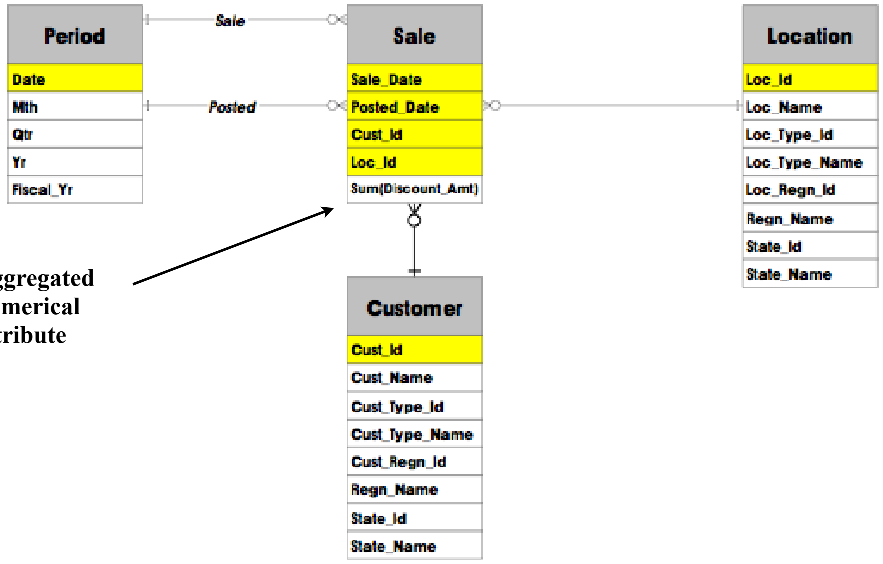
Customer Type

Fee Type

Sale Fee

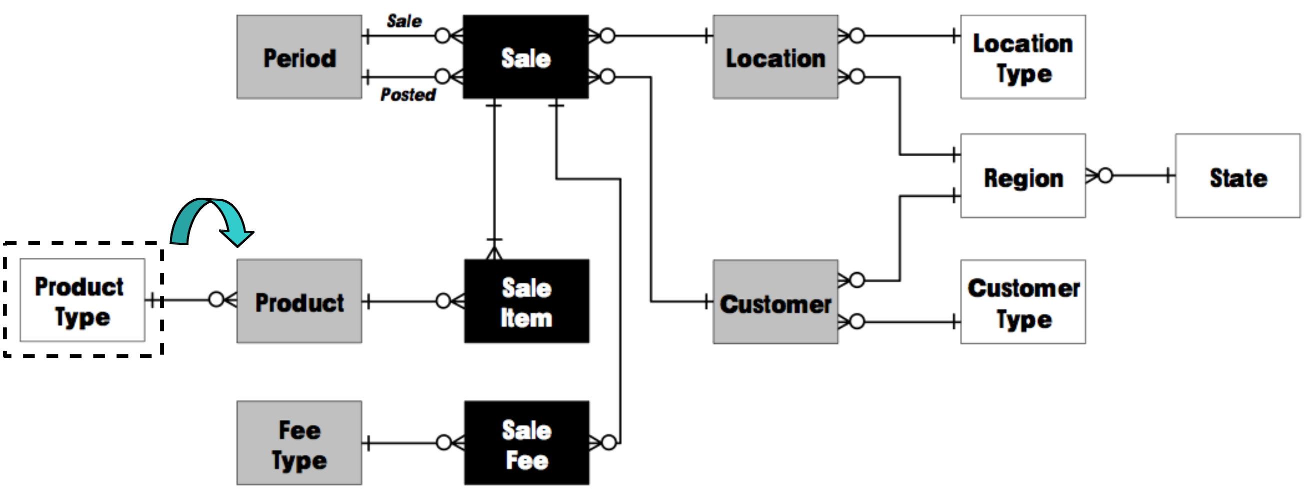
Derivation of Sale Star Schema

- This star schema has four dimensions, each of which contains embedded hierarchies



Aggregated numerical attribute

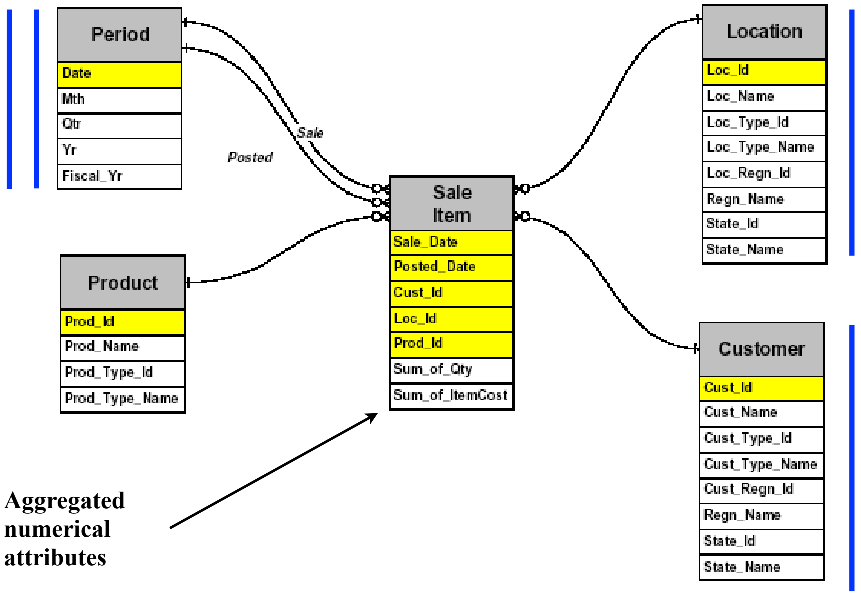
Derivation of Sale Item Star Schema



- Where hierarchical relationships exist between transaction entities, the child entity inherits all dimensions (and key attributes) from the parent entity

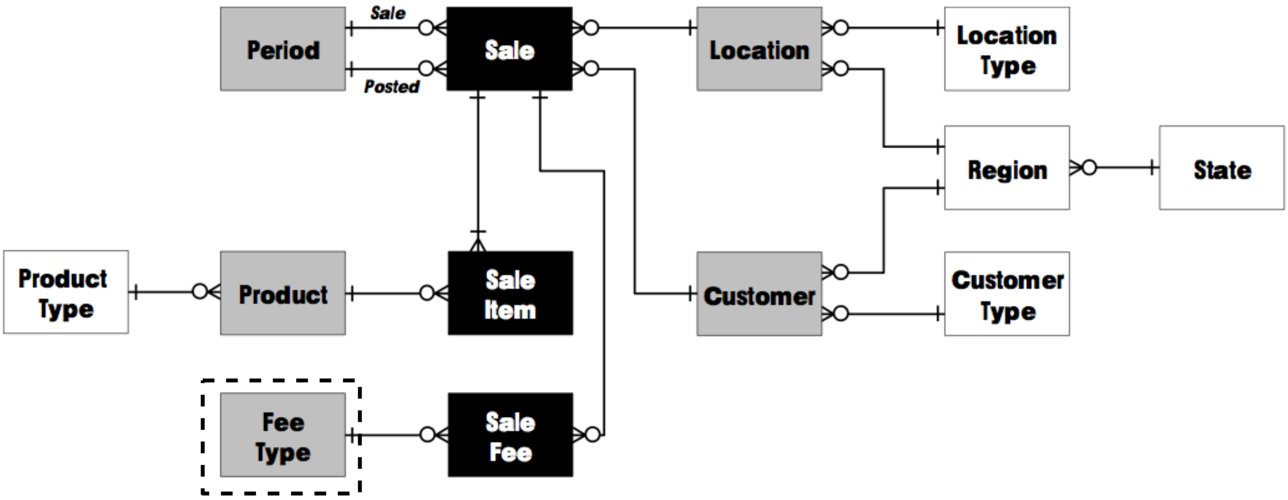
Derivation of Sale Item Star Schema

- This star schema has five dimensions, including four dimensions from its "parent" transaction entity (Sale)



Aggregated numerical attributes

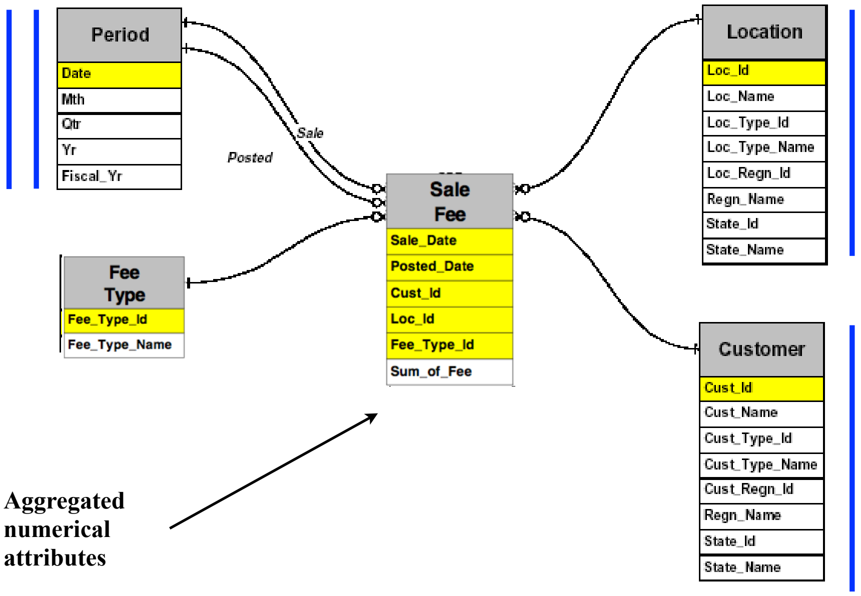
Derivation of Sale Fee Star Schema



- Where hierarchical relationships exist between transaction entities, the child entity inherits all dimensions (and key attributes) from the parent entity

Derivation of Sale Fee Star Schema

- **This star schema has five dimensions, including four dimensions from its “parent” transaction entity (Sale)**



Period
Date
Mth
Qtr
Yr
Fiscal_Yr

Fee Type
Fee_Type_Id
Fee_Type_Name

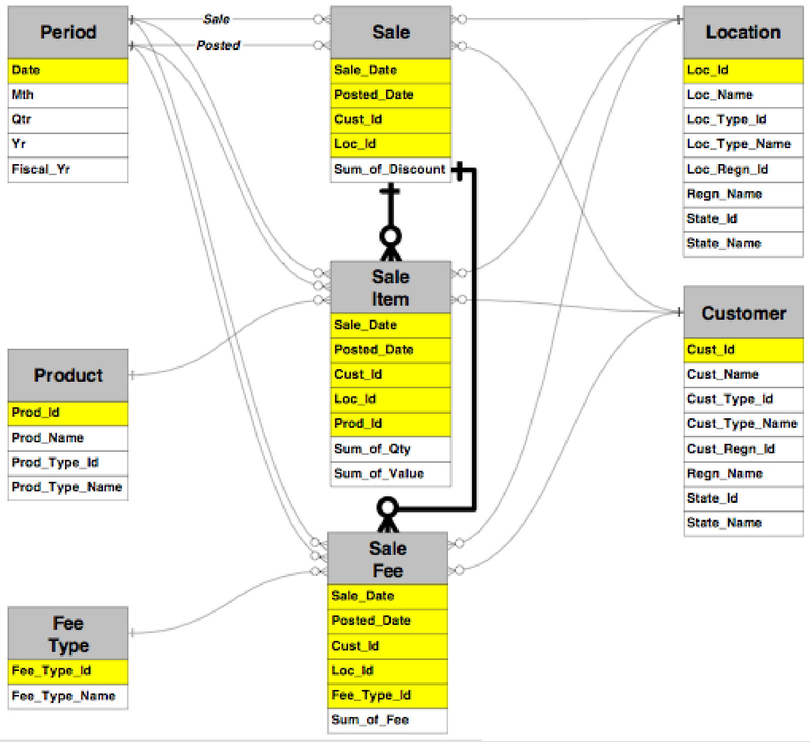
Sale Fee
Sale_Date
Posted_Date
Cust_Id
Loc_Id
Fee_Type_Id
Sum_of_Fee

Location
Loc_Id
Loc_Name
Loc_Type_Id
Loc_Type_Name
Loc_Regn_Id
Regn_Name
State_Id
State_Name

Customer
Cust_Id
Cust_Name
Cust_Type_Id
Cust_Type_Name
Cust_Regn_Id
Regn_Name
State_Id
State_Name

Aggregated numerical attributes

Constellation Schema



Period
Date
Mth
Qtr
Yr
Fiscal_Yr

Sale
Sale_Date
Posted_Date
Cust_Id
Loc_Id
Sum_of_Discount

Location
Loc_Id
Loc_Name
Loc_Type_Id
Loc_Type_Name
Loc_Regn_Id
Regn_Name
State_Id
State_Name

Sale Item
Sale_Date
Posted_Date
Cust_Id
Loc_Id
Prod_Id
Sum_of_Qty
Sum_of_Value

Product
Prod_Id
Prod_Name
Prod_Type_Id
Prod_Type_Name

Customer
Cust_Id
Cust_Name
Cust_Type_Id
Cust_Type_Name
Cust_Regn_Id
Regn_Name
State_Id
State_Name

Sale Fee
Sale_Date
Posted_Date
Cust_Id
Loc_Id
Fee_Type_Id
Sum_of_Fee

Fee Type
Fee_Type_Id
Fee_Type_Name

Galaxy

- A set of star schemas or constellations can be combined together to form a galaxy.
- A galaxy is of a collection of star schemas with shared dimensions. Unlike a constellation schema, the fact tables in a galaxy do not need to be directly related.

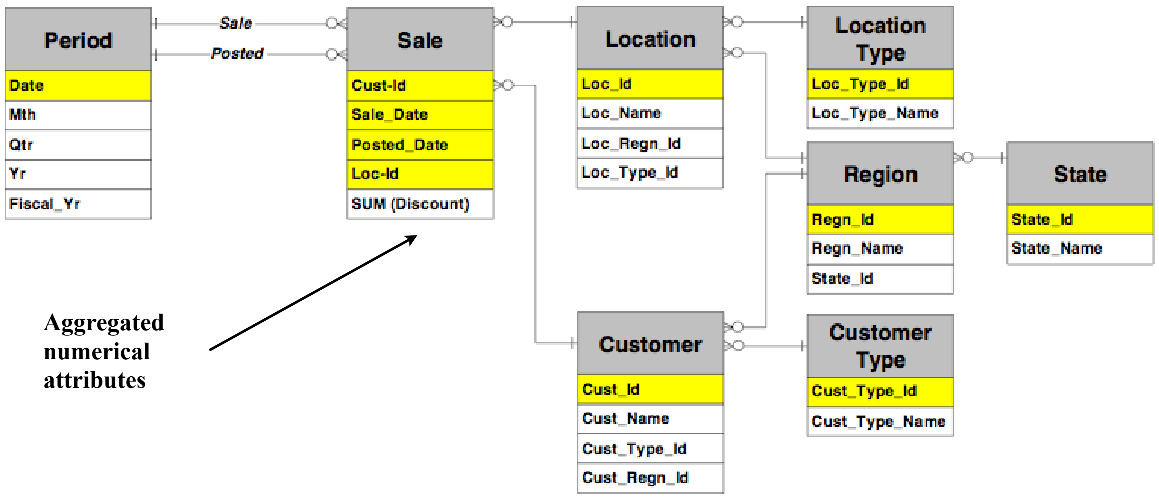
Snowflake

- In a star schema, hierarchies in the original data model are collapsed or denormalized to form dimension tables. Each dimension table may contain multiple independent hierarchies.
  - A snowflake schema is a star schema with all hierarchies explicitly shown.
  - A snowflake schema can be formed from a star schema by expanding out (normalising) the hierarchies in each dimension.
  - Alternatively, a snowflake schema can be produced directly from an Entity Relationship model.

Snowflake

- Alternatively, a snowflake schema can be produced directly from an Entity Relationship model:
  - A fact table is formed for each transaction entity. The key of the table is the combination of the keys of the associated component entities
  - Each component entity becomes a dimension table.
  - Where hierarchical relationships exist between transaction entities, the child entity inherits all relationships to component entities (and key attributes) from the parent entity.
  - Numerical attributes within transaction entities should be aggregated by the key attributes. The attributes and functions used depend on the application.

Snowflake - from the Sale transaction entity.



Period: Date, Mth, Qtr, Yr, Fiscal_Yr

Sale: Cust-Id, Sale_Date, Posted_Date, Loc-Id, SUM (Discount)

Location: Loc_Id, Loc_Name, Loc_Regn_Id, Loc_Type_Id

Location Type: Loc_Type_Id, Loc_Type_Name

Region: Regn_Id, Regn_Name, State_Id

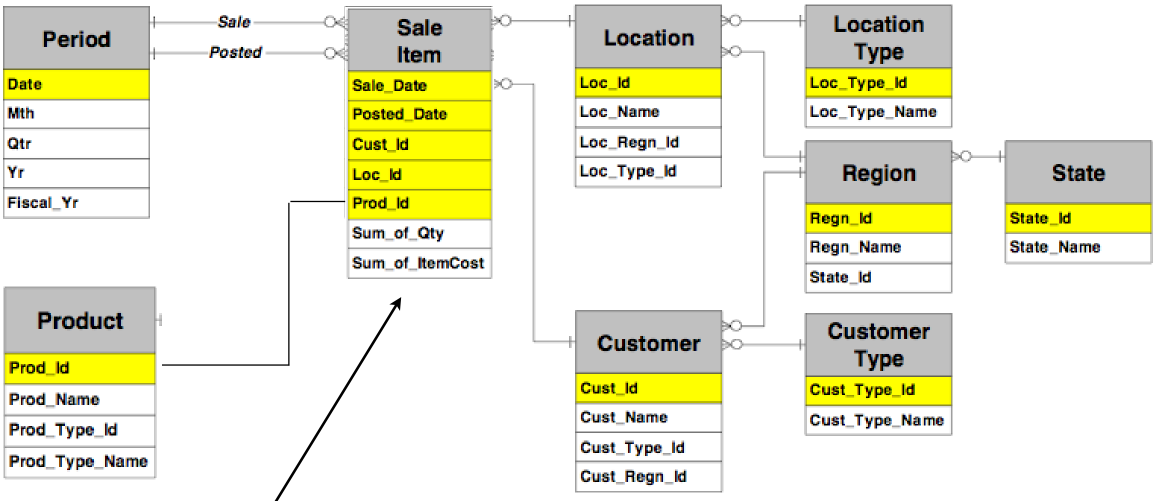
State: State_Id, State_Name

Customer: Cust_Id, Cust_Name, Cust_Type_Id, Cust_Regn_Id

Customer Type: Cust_Type_Id, Cust_Type_Name

Aggregated numerical attributes

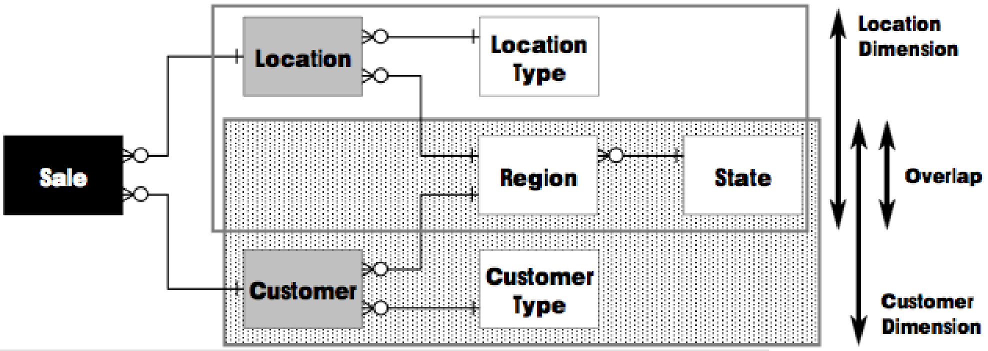
Snowflake - from the Sale Item transaction entity.



Aggregated
numerical
attributes

Star Cluster Schema

- **Star Schema (full denormalized) versus Snowflake (full normalize)**
- **Star cluster schema as one which has the minimal number of tables while avoiding overlap between dimensions. It is a star schema which is selectively “snowflaked” to separate out hierarchical segments or sub-dimensions which are shared between different dimensions.**



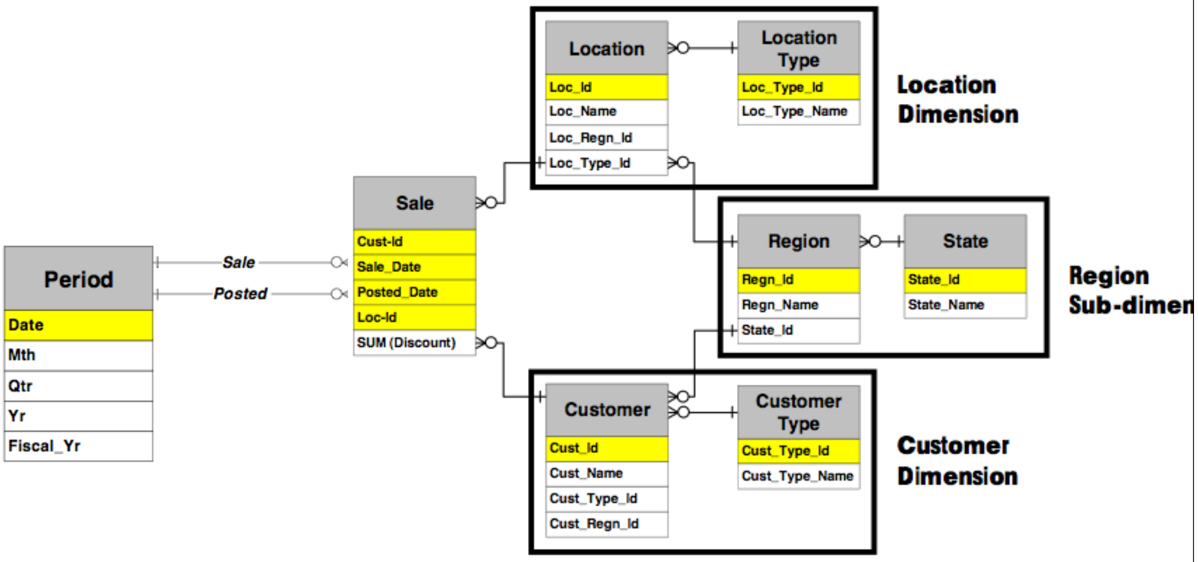
Star Cluster Schema - Deriving from E/R Model

- A star cluster schema may be produced from an Entity Relationship model using the following procedure. Each star cluster is formed by:
  - A fact table is formed for each transaction entity. The key of the table is the combination of the keys of the associated component entities.
  - Classification entities should be collapsed down their hierarchies until they reach either a fork entity or a component entity.
    - If a fork is reached, a sub-dimension table should be formed. The sub-dimension table will consist of the fork entity plus all its ancestors. Collapsing should begin again after the fork entity. When a component entity is reached, a dimension table should be formed.

Star Cluster Schema - Deriving from E/R Model

- Each star cluster is formed by (cont):
    - Where hierarchical relationships exist between transaction entities, the child entity should inherit all dimensions (and key attributes) from the parent entity.
    - Numerical attributes within transaction entities should be aggregated by the key attributes (dimensions). The attributes and functions used depend on the application.

Star Cluster Schema - Deriving from E/R Model



Location
Loc_Id
Loc_Name
Loc_Regn_Id
Loc_Type_Id

Location Type
Loc_Type_Id
Loc_Type_Name

Location Dimension

Sale
Cust-Id
Sale_Date
Posted_Date
Loc-Id
SUM (Discount)

Period
Date
Mth
Qtr
Yr
Fiscal_Yr

Sale

Posted

Region
Regn_Id
Regn_Name
State_Id

State
State_Id
State_Name

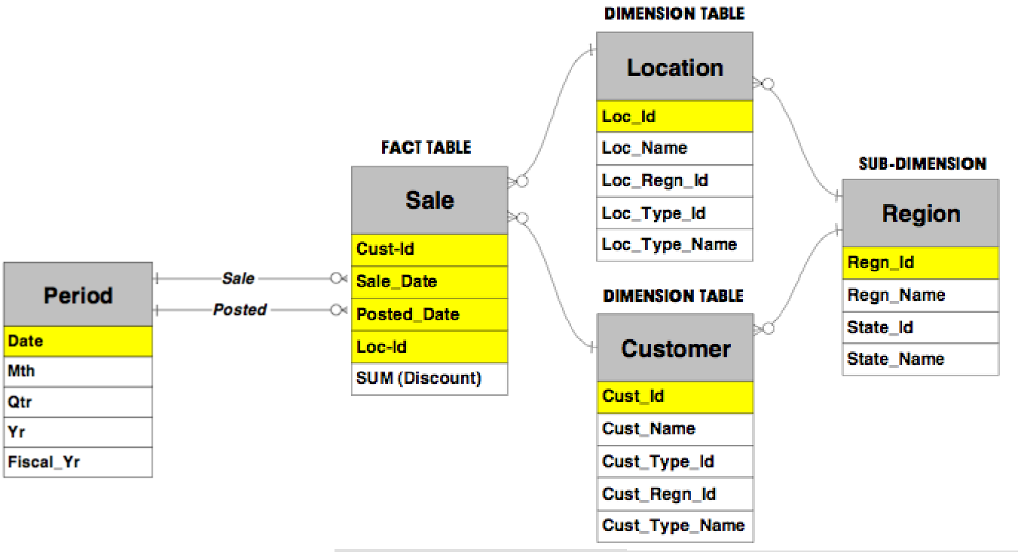
Region Sub-dimen

Customer
Cust_Id
Cust_Name
Cust_Type_Id
Cust_Regn_Id

Customer Type
Cust_Type_Id
Cust_Type_Name

Customer Dimension

Star Cluster Schema - Deriving from E/R Model



DIMENSION TABLE

Location
Loc_Id
Loc_Name
Loc_Regn_Id
Loc_Type_Id
Loc_Type_Name

FACT TABLE

Sale
Cust-Id
Sale_Date
Posted_Date
Loc-Id
SUM (Discount)

SUB-DIMENSION

Region
Regn_Id
Regn_Name
State_Id
State_Name

Period
Date
Mth
Qtr
Yr
Fiscal_Yr

Sale

Posted

DIMENSION TABLE

Customer
Cust_Id
Cust_Name
Cust_Type_Id
Cust_Regn_Id
Cust_Type_Name

Star Cluster Schema - Deriving from E/R Model

- **If required, views may be used to reconstruct a star schema from a star cluster schema. This gives the best of both worlds: the simplicity of a star schema while preserving consistency between dimensions.**
- **As with star schemas, star clusters may be combined together to form constellations or galaxies.**

Evaluation and model tuning

The need for evaluation and model tuning

- In practice, dimensional modeling is an iterative process. These procedures are useful for producing a first cut design, but this will need to be refined to produce the final data mart design.
- Most of these modifications have to do with further simplifying the model and dealing with non hierarchical patterns in the data.
  - Combining Fact Tables
  - Combining Dimension Tables
  - Produce pre-aggregated stars

The need for evaluation and model tuning

- Combining Fact Tables
  - Fact tables with the same primary keys (i.e. the same dimensions) should be combined. This reduces the number of star schemas and facilitates comparison between related facts (e.g. budget and actual figures).
- Combining Dimension Tables
  - Creating dimension tables for each component entity often results in a large number of dimension tables. To simplify the data mart structure, related dimensions should be consolidated together into a single dimension table.

The need for evaluation and model tuning

Many to Many Relationships

- **Many-to-many relationships cause problems in dimensional modeling because they represent a “break” in the hierarchical chain, and cannot be collapsed.**
- **Options for dealing with many-to-many relationships:**

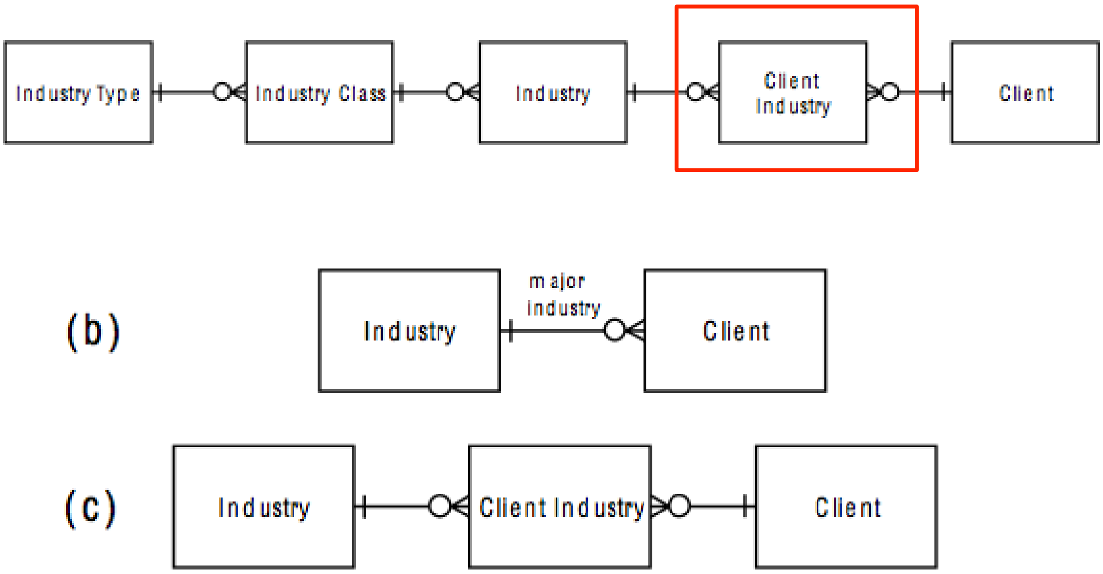
  a. Ignore the intersection entity (eliminate it from the data mart).

  b. Convert the many-to-many relationship to a one-to-many relationship, by defining a “primary” relationship. eventually consider also a “secondary” one-to-many relationship.

  c. Include it as a many-to-many relationship in the data mart such entities may be useful to expert analysts but will not be amenable to analysis using an OLAP tool.

The need for evaluation and model tuning

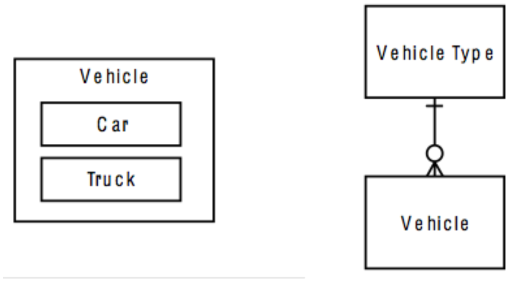
Many to Many Relationships (cont) - Example



The need for evaluation and model tuning

Handling Subtypes

- Supertype/subtype relationships can be converted to a hierarchical structure by removing the subtypes and creating a classification entity to distinguish between subtypes. This can then be converted to a dimensional model in a straightforward manner.


Vehicle
Car
Truck
Vehicle Type
Vehicle