Chapter #5
Conceptual Modeling

Challenges of designing data marts:

→ Entity-relationship model is widely used for database design, but it is not well suited for multidimensional data, which is common in data marts (data mart is based on multidimensional view of data).

→ The ERM expressive enough to represent the majority of concepts necessary for data modeling.

→ However, the basic ERM is not able to accurately highlight the distinctive features of the multi-dimensional modeling.

→ That's why its use for data mart is not suitable.

Chalenges of designing data marts using star schema:

Designers, focus on the structure and relationships of the data within the data mart without considering the underlying

physical implementation.

→ For this, they directly define star schema structure.

Strict schema:

- Specific type of data model used in data marts.
- Characterized by a central fact table surrounded by dimension tables.
- A star schema is a relational schema.

→ Directly using star schemas for conceptual design is similar to designing a relational database without first creating an ER-schema, which can lead to problems with respect to requirements, maintenance and reuse.

→ In case of data warehouse, star schemas can be particularly problematic because:

1. They are almost completely denormalized.
2. Do not properly code the functional dependencies that define hierarchies in design and implementation of the data mart.

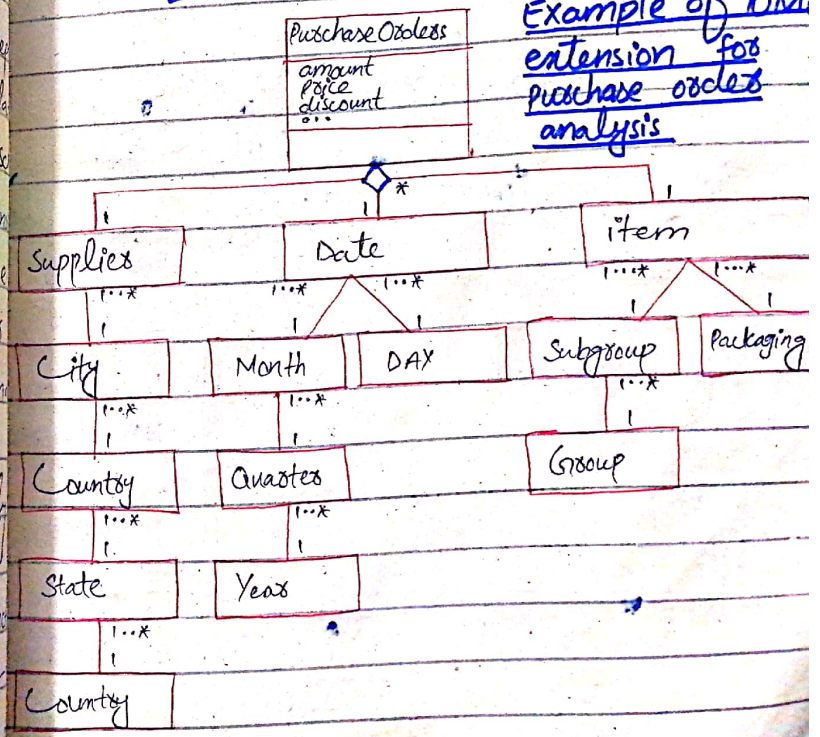
not accurately represent the relationships and dependencies between different levels of data.

→ Combining the strength of ERM and star schemas to create effective data mart design.

→ There are different approaches to multidimensional modeling some of which based on:

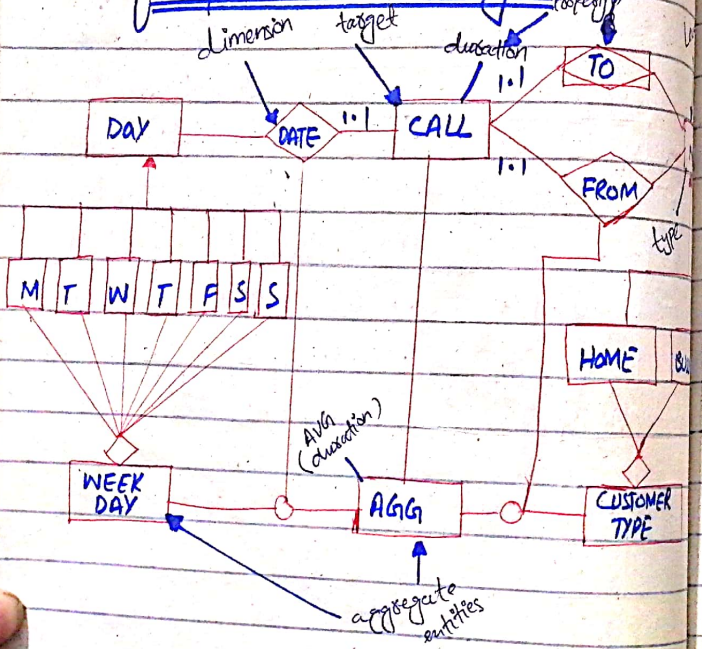
1. ERM extensions
2. UML extensions

Example of UML extension for purchase orders analysis



- Purchase orders —role→ facts
- A UML class —have→ attributes are the f... measures, models purch... orders.
- Dimensions → suppliers, date & item orders
- Aggregations → connect the facts and th... dimensions, and many-to-one associations link the different levels of aggregation.

Example of ERM extension for phone call analysis



→ The given ERM schema analyzes the calls made through a telephone company.

- Fact → target
- Measures → Properties
- Specialization hierarchies

Both model focus on fact, dimensions and measures.

Fact schema:

- Way of organizing data to make it easier to analyze.

Key elements:

Facts:
Central data points that are measured and analyzed, such as balance of checking account.

Dimensions:
Attributes that provide context for the facts like customer info, account type, time period.

Measures:
Calculations performed on the facts, such as avg balance or the total amount of deposits.

Non-usable attributes:

Attributes that are not used for aggregation but may be useful for other purposes.

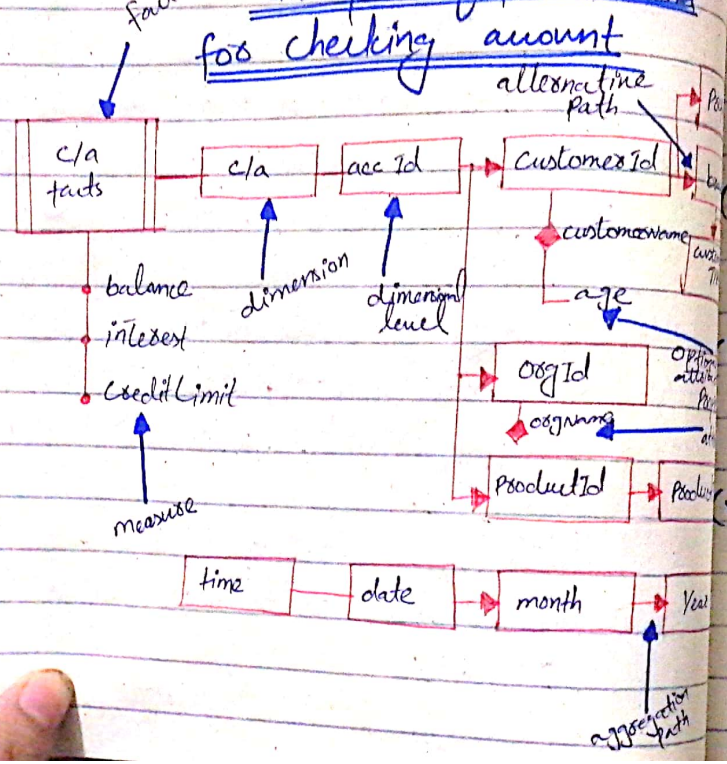
Optional attributes:

Attributes that may or may not present for given fact.

Alternate aggregation paths:

Different ways to group and analyze the data.

Example of fact schema for checking amount



Dimensional fact Model

- is a conceptual model created specifically to function as data mart design support.
- Way to picture and understand data in multi-dimensional way.

Goal of DFM:

(i) Clear design:
Give effective support to Conceptual design.

(ii) Easy queries:
Users can ask questions about the data in a simple way.

(iii) Better Communication:
Designers and users can communicate more easily about requirement Specificat

(iv) Stable platform:
Build a stable platform for logic design.

(v) Clear and expressive design documentation:
Provide clear and expressive design documentation.

- A fact schema is a diagram that shows the relationships blw dimensions and measure in a data warehouse.

Fact - schemata:

=> The conceptual representation generated by the DFM consists of a set of fact schemata.

=> Fact schemata model:
- Facts
- Measures
- Dimensions
- Hierarchies

Fact:

=> is a concept relevant to decision making processes.

=> like a story or event that can track.

=> its something that happens and can be measured.

=> They can change ouer times however Some of the concepts represented in the database are completely static.

Examples:

=> Sales:
When a product is sold.

=> Purchases:
When someone buys something.

=> Complaints:
when a customer is not happy.

example how facts are dymaic:

=> The numbers of sales can go up or down.

=> A customer complaint can be solved.

Measure:

=> are the numbers that describe the facts.

=> Numerical property of fact.

=> Describes quantitative fact aspect that is relevant to analysis.

Examples:

=> Quantity:
How many products were sold.

=> Total sales
Each sale is measured by the numbers of units sold, the unit price and the total receipts.

Note:

- Most measures are numbers, However, Sometimes Sometimes a fact has no measures in this case, only occurrence of an event is recording.

⇒ if fact has no measure, then the fact schema is said to be empty.

Dimensions:

⇒ are different ways to look at data.
⇒ are lenses though which we see data.
⇒ it is a fact property with a finite domain
⇒ Also describes an analysis coordinate of the fact.
⇒ A fact has more than one dimension.

Examples:

⇒ Time:
Years, month, day, hour, minute
⇒ Product:
Category, brand, size, color.
⇒ Sale:
Product, stores and dates.

Note:

⇒ Connection between measures and dimensions is expressed at the extensional (data level) - rather than schema level.

Primary Event:

⇒ is a single instance or particular occurrence of a fact.
⇒ Like a snapshot of the fact at a particular point in time.
⇒ Identified by one n-ple made up of a value for each dimension.
n-ple: The unique combination of values is called "n-ple."
⇒ A value for each measure is associated with each primary event.
⇒ Dimensions are normally used to identify and select primary events.

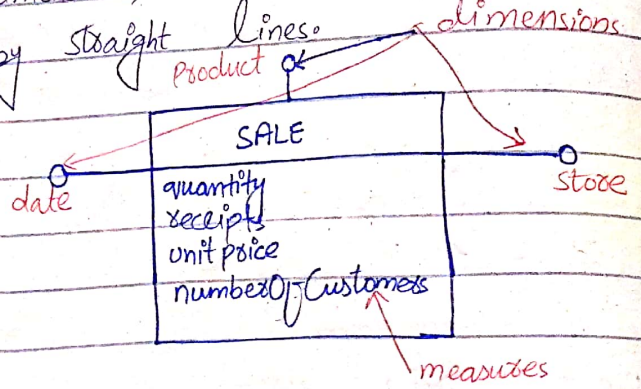
Example of primary event:

- Sale of 10 packages of detergent:
This is a specific instance of a sale. it happened on a particular day (10/10/2008) at a specific store (SmartMart) and involved a specific product (Shiny detergent).
  - The dimensions are time (10/10/2008), product (shiny detergent), store (SmartMart), and quantity (10 packages).
  - the measure is total sales ($25).

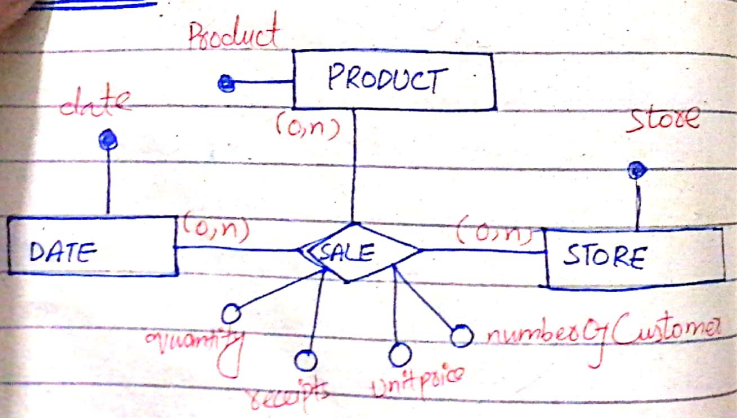
Simple fact schema for sales

⇒ A fact is represented by a box that displays the fact name along with the measure names.

⇒ Small circles represent the dimensions, which are linked to the fact by straight lines.



ER-schema corresponding to fact schema.



⇒ A fact expresses a many-to-many association between dimensions.

- Due to this, the ER schema corresponding to a fact schema consists of mainly n-ary relationship
- That models the fact, among entities that model dimensions
- The measures are attributes of this relationship.

Dimensional attribute:

- Stands for dimensions and other possible attributes, always with discrete values, that describe them.
- Hierarchies are used to represent relationships between different dimensional attributes. Dimensional attributes are represented by circle.

for example: A product is described by its type, by the category to which it belongs, by its brand, and by the department in which it sold. Then product, type, category, brand and department will be dimensional attributes.

Hierarchy:

- A hierarchy is a directed tree whose nodes are dimensional attributes and whose each ~~node~~ arc represent connects or model many-to-one association between dimensional attribute pairs.
- it includes a dimension, placed at the tree root and all of the dimensional attributes describe it.

for example:

A product hierarchy might have a "product category" at the top, with different "product types" branching off from it. This shows that product types belong to specific categories.

Note:

- Hierarchy term is different in ER-modeling and multidimensional modeling.
- In ER-modeling, hierarchies are used to represent inheritance (IS-A) relationships between entities.
- In multidimensional modeling, hierarchies are created in data warehouses using arcs represent relationships between attributes. These arcs can be used to express different types of relationships, such as aggregation, generalization and association.

(Some point on Next Page)

Secondary Events:

⇒ are group of primary events that share common characteristics.

⇒ Given a set of dimensional attributes (usually from different hierarchies), each n-tuple of their values identifies a secondary event.

⇒ Each secondary event has a value for each measure, which sums up the values of that measure in the corresponding primary events.

Example:

Sales can be grouped according to product category, month and store city. The n-tuple (StoreCity: Miami, product: Shiny, month: 10/2008) identifies a secondary event that aggregates all Sales of the Shiny product in Miami Stores during october 2008.

Note 2:

⇒ All attributes and measures with a fact schema must have distinct names.

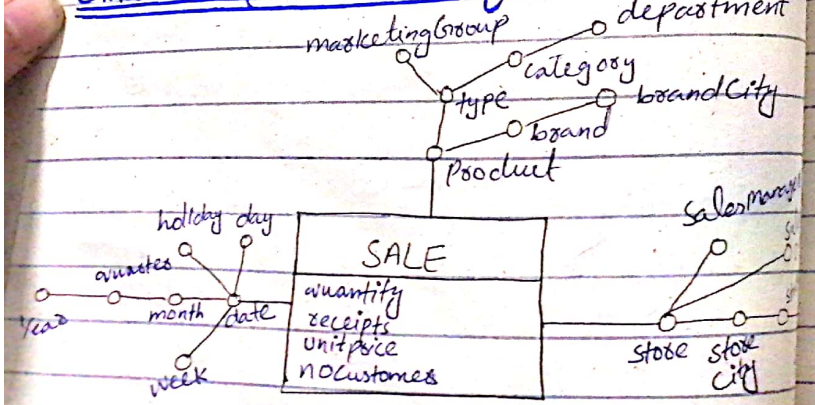
⇒ The convention proposed by Kimball et al. (1998) suggests building names from three components:

1. Object
2. Classification
3. Qualifiers

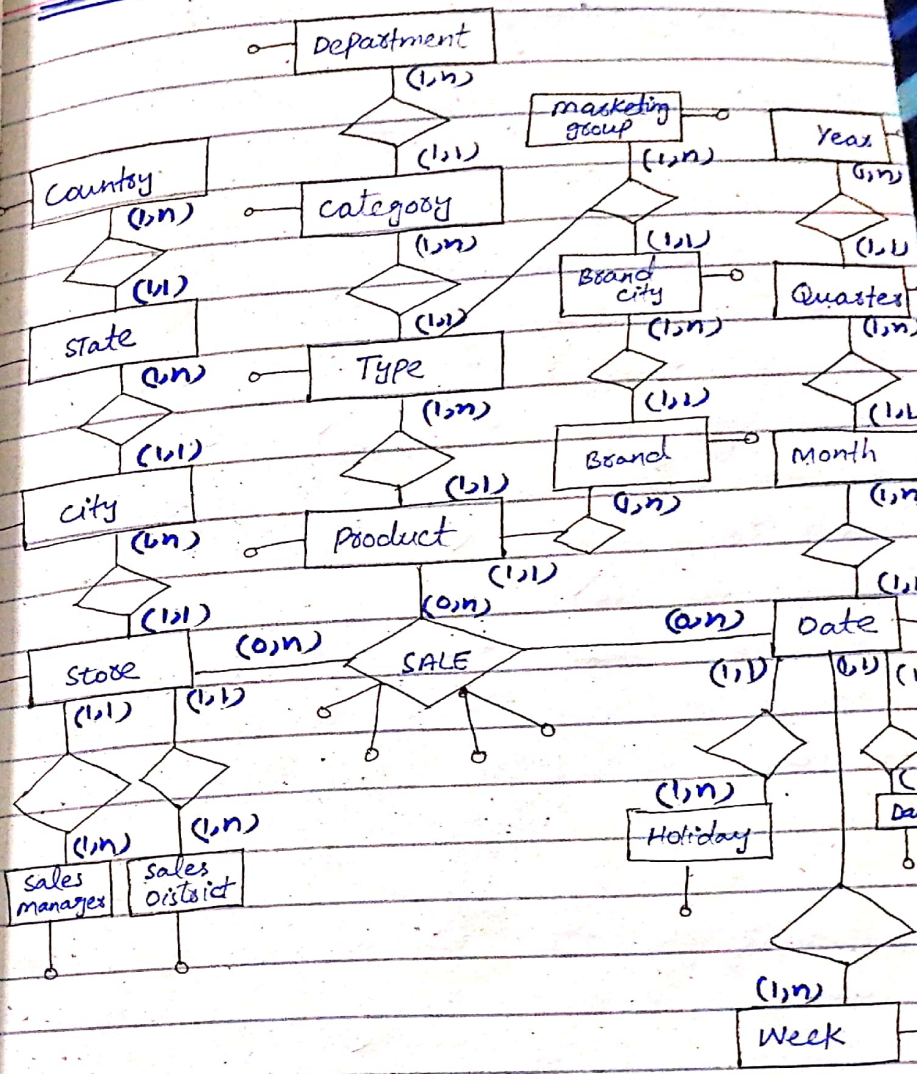
⇒ Functional dependencies establish many-to-one associations between the values of a dimension.

e.g A product belongs to one product type, but a product type can have many products.

Enhanced fact schema for Sales:



Eg- schema corresponding to fact schema:



Advanced Modeling

1. Descriptive Attributes:

⇒ Attributes that provide additional information about a dimensional attribute in a hierarchy.

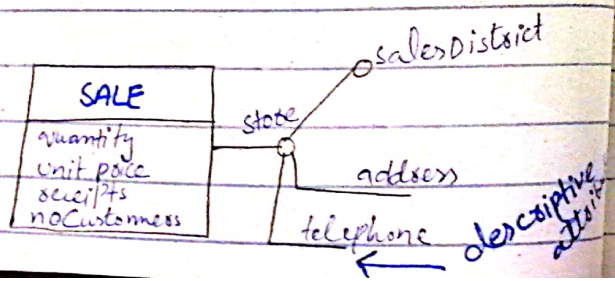
⇒ This information is not typically used for aggregation or sorting.

⇒ They are always at the "leaves" of hierarchy.

⇒ They are represented by horizontal lines in the DFM and as attributes of entities in the ERM.

example: (Dimensional attributes are stored)
- Store address
- Product weight
- Manager

⇒ Descriptive attributes often associated to dimensional attributes by 1-to-1 associ



2. Cross-Dimensional attribute:

⇒ These attributes are dimensional or descriptive attributes whose values are defined by the combination of two or dimensional attributes.

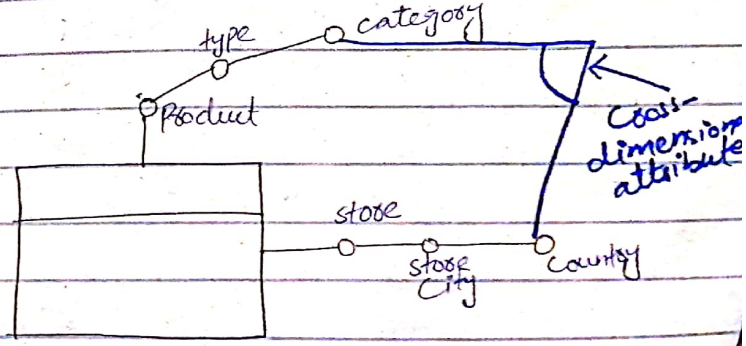
⇒ The value of a cross dimensional attribute depends on the values of other attributes from different hierarchies.

example:

Product VAT (Value added Tax), which might depend on both the product category and the country where it is sold.

Representation: They are represented by joining the arcs that define the attribute with a circular arc in the fait schema.

In ERM, they are represented by associations between etities.



3. Convergence:

⇒ Convergence deals with the shared hierarchies in a fact schema.

⇒ Hierarchies may not always be Simple trees, as two or more paths might connect the same dimensional attribute.

⇒ This happens when there are functional dependencies between attributes, even if they don't belong to the same hierarchy.

example:

- A store hierarchy where Stores are grouped into cities, states and countries, but also into sales districts.
- if store's country can be determined either through:

  Store → storecity → state → country

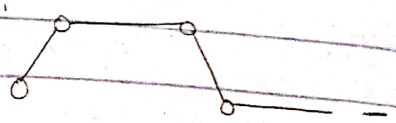
  or Store → SalesDistrict → Country

  there is a convergence.

Convergence are marked by arrows on the converging axes in the fact sch

⇒ In ERM, they are represented by redundant associations between entities.

⇒ Similar looking attribute don't always indicate a convergence. They might have different meanings and need to be represented separately.



4. Shared Hierarchies:

⇒ Shared hierarchies are portions of hierarchies that are replicated two or more times within a fact schema.

⇒ These are hierarchies that are used in multiple parts of a fact schema.

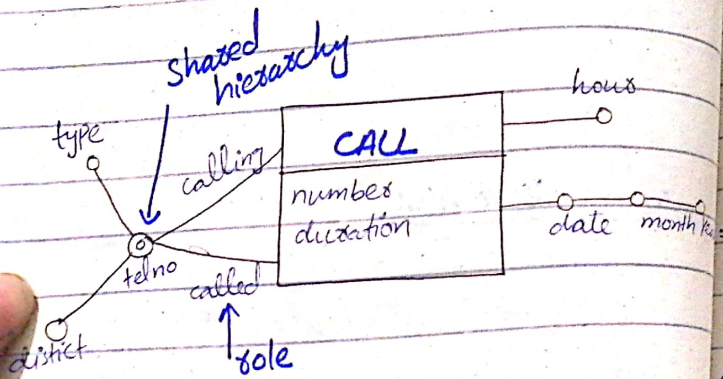
example:

- Temporal hierarchies (eg. date, month, Year)
- Geographical hierarchies (city, state, Country etc).

⇒ if have sharing hierarchies maintain distinct names for all attributes to avoid ambiguity.

⇒ Graphic notation used for this is use double circles to represent shared attributes and their descendents. This is more concise representation.

Diagram:



5. Multiple Arcs:

⇒ Used to represent many-to-many relationship between attributes.

⇒ A single value of one attribute can be associated with multiple attribute values and vice versa.

example:

⇒ Book Sales fact schema where a book can have multiple authors and an author can write multiple books.

Representation:

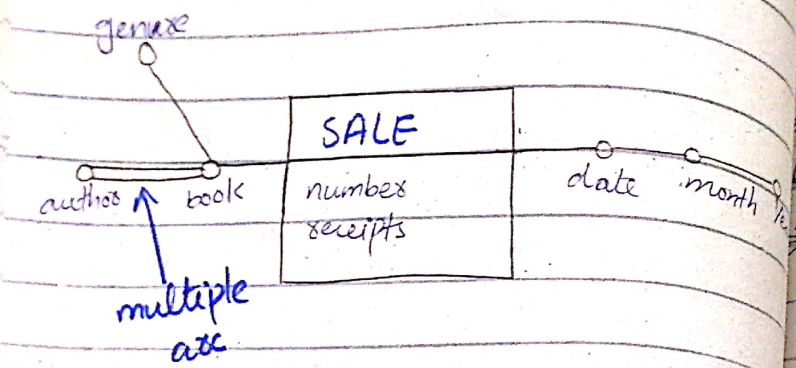
⇒ Represented by drawing two arcs between the attributes.

Coefficient:

⇒ A coefficient should be assigned to each multiple arc to define the aggregation rule for the M-2-M relationship. This ensures that aggregations are calculated correctly along hierarchies that include multiple arcs.

⇒ This is important for modeling where one entity can be associated with multiple entities of other types.

Diagram:



6. Optional Arc:

⇒ These are used to model situations where certain attributes or dimensions might not always be defined for all events.

⇒ An optional arc is marked with a dash(-) in the diagram.

Two cases to consider:

An optional arc determines a dimensional attribute.

⇒ An optional arc determines a dimension

Coverage of optional arc:

⇒ When two or more optional arcs exit from the same attribute, coverage should specify.

⇒ Coverage defines the relationship between the different options involved.

⇒ These are four types of coverage:

Total (T):

- At least one of the children is linked to each value of the parent attribute.

Disjoint (D):

- A value of the parent attribute is linked to at most one of the children.

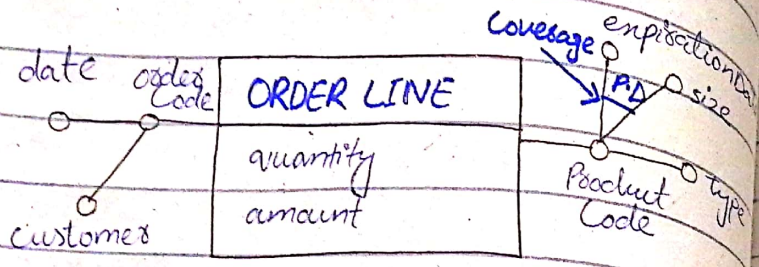
Partial (P):

- Values of the parent attribute exist for which all of the children are undefined.

Overlapping (O):

- Values of the parent attribute exist that link to values of two or more children.

Diagram:



7. Incomplete Hierarchies:

⇒ Hierarchies where one or more levels of aggregation are missing for some instances.

⇒ All entities in the hierarchy have complete information for all levels.

ex:

⇒ Geographically hierarchy where some cities might be missing their country or state information.

Representation:

⇒ By marking the missing attributes with a dash (-).

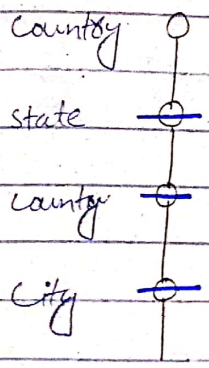
The term "en-many" is sometimes used to describe incomplete hierarchies.

Optional Arc VS incomplete hierarchy

- Entire attributes or dimensions are missing
- Some values within a hierarchy level are missing.

⇒ Modeling incomplete hierarchies can be done using either optional arc or by explicitly representing the missing levels.

Incomplete geographical hierarchy diagram:



8. Recursive Hierarchies:

⇒ Useful for modeling self-referencing relationships.

⇒ In these hierarchies parent-child relationships among levels are same, but the instances can have different length.

example:

Company organization chart where employees report to managers and managers themselves are employees.

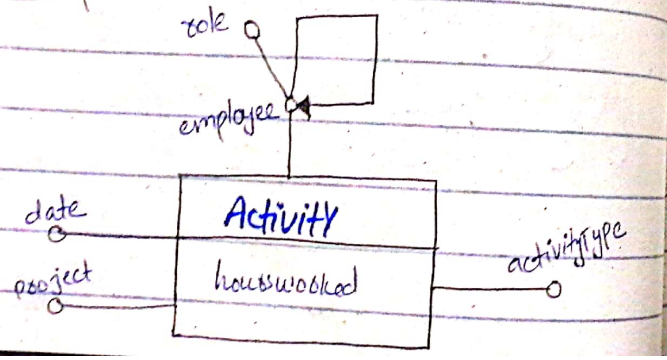
Representation:

⇒ Represented by loop.

Caution:

⇒ Not always possible to distinguish the different levels of aggregation semantically at the fact schema level.

example:



9. Additivity:

⇒ Refers to how measures can be aggregated along different dimensions.

⇒ Measure can be classified into three types:

1. Flow measures:

- These measures are additive along all dimensions.

example:

Sales quantity & no of units sold.

2. Level Measures:

- These measures are not additive along temporal dimensions but are additive along non-temporal dimensions.

example:

inventory levels and hierarchy.

3. Unit ~~price~~ Measure:

- These measures are not additive along any dimension.

example:

Unit price & Discount percentage.

Valid aggregation Operations

	Temporal	Non-temporal
flow measures	SUM, AVG, MIN, MAX	SUM, AVG, MIN, MAX
Level measures	AVG, MIN, MAX	SUM, AVG, MIN, MAX
Unit measures	AVG, MIN, MAX	AVG, MIN, MAX

Events and Aggregation

Events:

⇒ Data warehouse are designed to store and analyze large amounts of data. To efficiently manage and analyze this data, it's broken down into smaller units called events.

Primary Events	Secondary Events
are the most detailed level of data, representing the smallest unit of activity.	Derived from primary events through aggregation.
ex: a single sale of a product.	ex: Represent higher level of detail, like total sales for a specific product category.

- Events are building blocks of data warehouse.
- Cube represent all possible combinations of primary events, but some cubes are sparse.

Missing Events:

They have different interp... depending on the type of fact ...

1. Lossy-grained fact schema:

In this type of schema, miss... events are assumed to have ... elevant values, typically zero.

ex:

if a product was not sold in a store on a certain day, it assumed that the sales quantity was zero.

2. Lossless-grained fact schema:

In this schema, missing events truly represent the absence of an event.

Representing missing events:

Two approaches to represent mi... events in lossy-grained schemas:

1. Coverage schema:

⇒ A new schema is cre... explicitly represent the abse... of an event.

3. Null values in the original schema:

⇒ use null values in the original schema to represent missing events.

- Lead to wasted space if the number of missing events is high.

Aggregation:

⇒ The process of combining data from multiple rows into a single row, typically by summing up values.

Group-by set:

⇒ A subset of a dimensional attributes in a fact schema that does not contain two attributes related by a functional dependency.

⇒ Group-by sets define different levels of aggregation.

Primary group-by sets	Secondary group-by sets
• The set of all dimensions in a fact schema.	• Subsets of the primary group-by set.

◦ Represents the most detailed level of aggregation.	◦ Represent different levels of aggregation.

ex:

G_{11} = {Product, state, quarter}

G_{12} = {Country, date}

G_{13} = {Year}

G_{14} = { }

Roll-up order:

⇒ Defines the hierarchy of group-by sets.

⇒ Specifies how secondary group by sets can be rolled up to the primary group-by set.

Relationship:

A group-by set G_{11} is said to roll-up to another group-by G_{12} ($G_{11} < G_{12}$) if for each attribute in G_{11}, there is a corresponding attribute in G_{12} belonging to the same hierarchy.

Ex:

The group-by set (type, store, month) rolls up to the group-by set (type, store, quarter) because the date hierarchy is rolled up to the same hierarchy.

Lattic Structure:

⇒ The set of all possible group-by sets in a fact schema form a lattic structure.

⇒ The primary group-by set is the maximum element in the lattic.

⇒ The empty group-by set is the minimum element in the lattic.

⇒ For each pair of group-by sets there exists a superior group-by set (roll-up) and an inferior group-by set (drill-down).

Aggregating additive measures

⇒ Additive measures are those that can be summed up to calculate the total value for higher level of aggregation.

ex: The total sales for a year can be calculated by summing up the sales for each month.

Associative property:

⇒ The associative property of the sum operator allows for efficient calculation of aggregated values at different levels.

Aggregating Non-additive Measures:-

⇒ Cannot simply summed up. They require different techniques:

1. Distributive
2. Algebraic
3. Holistic

1. Distributive Measures:

⇒ These measures can be calculated by applying the same aggregation function to partial aggregates.

ex:

- MIN & MAX functions are distributive.
- Minimum value of a group can be find by finding the value of each subgroup and then take minimum of those minimums.

2. Algebraic Measures:

⇒ These measures require additional information beyond the partial aggregates to calculate the correct values.

ex:

AVG function is algebraic. Cannot simply the average the averages of subgroups to get the overall average. Find Count and sum of each subgroup to Calculate the correct average.

3. Holistic Measures:

⇒ These measures cannot be calculated from partial aggregates at all. They require access to the underlying data.

Aggregating With Convergence & Cross-Dimensional Attributes

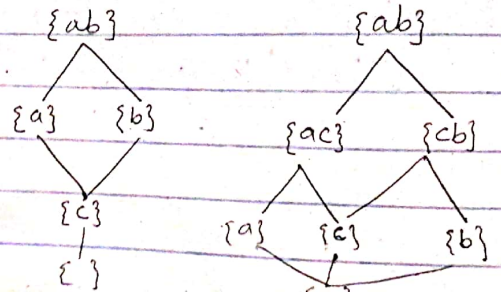
Convergence:

⇒ Convergence refers to a situation where a lower-level dimension

is related to higher level dimen
through a clear hierarchy.
⇒ Aggregation of convergence
straightforward.
⇒ Roll-up values from lower
to higher levels without ambiguity.

Cross-dimensional attributes:

⇒ Cross-dimensional attributes are
attributes that are related to mu
dimensions.
ex: a product might be
associated with both a category an
a brand.
⇒ Aggregation with cross-dimensiona
attributes can be more complex.
⇒ Ensure that the aggregation is
done correctly based on the relation
between the dimensions and the cro
dimensional attribute.



Aggregating with optional & multiple Arcs

Optional Arcs:

⇒ An optional arc exists between two dimensions when there is a one-to-many relationship. This means that one value in the higher level dimension can be associated with multiple values in the lower level dimensions.
⇒ To handle optional arc during aggregation, introduce a fictitious "No value" to represent missing values.
⇒ This allows to roll-up values from lower levels to higher levels, even these are missing values.

Multiple Arcs:

⇒ multiple Arcs occur when a lower-level dimension is related to multiple higher level dimensions.
⇒ Aggregation of these is very complex. Assigning weights to arc can help in allocating values to different dimensions.

Aggregation in empty Fact Schemas:

⇒ Schemas that don't have any measures.

⇒ In these schemas, the primary events only record the occurrence of events in an application domain.

⇒ In empty fact schemas, the only aggregation possible is counting the number of primary events. This is achieved using the COUNT aggregate operator.

Aggregating with functional Dependencies among dimensions:

⇒ A functional dependency exists when one dimension determines the value of another dimension.

Aggregating With incomplete or Recursive Hierarchies:

Incomplete Hierarchies:

⇒ Incomplete hierarchies occurs when there are missing values for one or more attributes in the hierarchy.

⇒ Lead to difficulties in aggregation as it becomes challenging to roll up values to higher levels of the hierarchy.

Recursive hierarchies:

⇒ A recursive hierarchy occurs when a dimension has a self-referential relationship. Means a dimension can be both a parent and a child to itself.

⇒ Aggregating data in recursive hierarchies require multiple levels of recursion.

Time

1. Transactional vs Snapshot Schemata

⇒ These are two main ways of representing data in database hous...

Feature	Transactional schemata	Snapshot schemata
Focus	Records individual events or transactions	Captures pe... snapshots ... data.
Data Granularity	Fine-grained, detailed data	Coasse-grai... Summarize... data.
Data Structure	Typically a fact table with detailed measures and dimensions	Fact table... measures an... dimensions ... rerenting a sp... point in time
Data Update	Frequent updates as new transactions occur.	Periodic upd... at specific intervals.
Ex:	Bank deposite/withdrawls Sales, Transactions,	Monthly sale... figures, end-... day inventor...
Data update	Frequent updates as new transactions occur.	Periodic updates at specific intervals.
Data analysis	Suitable for detailed analysis of individual transactions and trends over time.	Best for analyzing trends and comparisons between different Periods.
Key Considerations	Data volume and storage requirements, query performance, and real-time processing needs.	Data accuracy and consistency, Data retention Policies, And data freshness.

⇒ Transactional schemata is ideal for capturing a detailed history of events and analyzing individual transactions.

⇒ Snapshot schemata are better suited for summarizing data at specific points in time and analyzing trends over longer Periods.

2. Late Update:

⇒ When change something after it's already been recorded.

⇒ Refers to the situation where the values of certain metrices or dimensions associated with a specific event are updated after the initial record of that event.

⇒ This often occurs due to delays in data processing, errors in initial data, or the need for more accurate information.

for example:

- imagine a data warehouse tracking sales data. Initially, a sale might be recorded with an estimated revenue figure.
- Later, when the final invoice is processed, the actual revenue figure might be different and the data warehouse needs to be updated to reflect this change.

Key challenges of late updates:

Data Consistency:

Ensuring data consistency across different tables and reports can be challenging when late update occurs.

Data Quality:

- Can impact the the overall quality of the data in the data-warehouse.

Data Analysis and Reporting:

- Late updates can lead to inaccurate or incomplete analysis and reporting.

Addressing Late updates:

To effectively handle late updates data warehouse designers and administrators can implement strategies such as:

Incremental load processes:

- Regularly updating the data warehouse with the latest information.

Change Data Capture:

- Identifying and tracking changes

in source systems to ensure tim updates.

Data Quality Checks:

- Implementing robust data quality checks to identify and correct errors.

Versioning:

- Keeping track of different versions of data to allow for historical analysis.

Alerting Mechanisms:

- Setting up alerts to notify relevant stakeholders about late updates and potential data quality issues.

3. Dynamic Hierarchies:

⇒ Dynamic hierarchies are hierarch that change over time.

⇒ This is important because the organizations often restructure, and these changes need to be reflected in th data warehouse.

Example:

- a company with a hierarchica structure for its sales organization.
  - Region
  - District
  - Store.

Overtime, the company might restructure adding new districts or moving stores between districts. A dynamic hierarchy allows the data warehouse to adapt to these changes.

Four ways to analyze data with dynamic hierarchies:

1. Today-for-today
2. Yesterday-for-today
3. Today-for-Yesterday
4. Today-and-Yesterday

How dynamic Hierarchies are implemented?

⇒ Dynamic hierarchies are typically implemented in data warehouse tools like Power BI or Tableau.

With dynamic hierarchy we can do operations:

- Drill Down
- Drill Up

Overlapping Fact Schema

⇒ Means how different fact tables in a data warehouse can overlap if they share common dimensions.

⇒ Different fact tables in a data warehouse represent different sets of facts or measurements.

⇒ Sometimes need to compare data from two different fact tables.

Drill-across queries:

⇒ Query that compares measures across different fact tables is called a "drill-across" query.

Comparable fact schemata

⇒ Defining how to determine if two fact-tables can be combined for drill-across.

⇒ Two fact tables can be combined if they have at least one dimension in common.

Example:

⇒ In real-world scenarios, finding common dimensions can be challenging.

⇒ There are no perfect way to find common dimensions, but they provide some guidelines:

1. Dimensions should have overlapping values to be considered the same.
2. it is hard to determine if dimensions are the same if they come from different systems with different data types.
3. if the dimensions have the same name in different fact tables, that's a good sign they might be the same.

⇒ In well designed data warehouse, dimensions should be consistent across different fact tables.

⇒ If dimensions are consistent, it is easy to identify common dimensions.

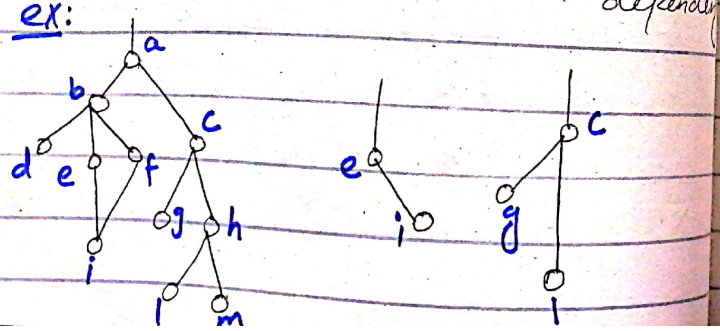
Reducing Hierarchies

⇒ How simplify dimensions when combining fact tables.

Def:

⇒ Taking original tree H and breaking it down into smaller trees. Choose a specific group of relatives (X) (attributes) and the new smaller trees will only include those relatives (attributes) and anyone that directly relate to them. (Transitive or functional dependency)

ex:



Overlapping Compatible Fact Schemata:

⇒ Two fact tables can be combined if they have at least one dimension in common. This is called "overlapping compatible fact schemata".

⇒ Two comparable fact schemata are called compatible when reduction of their hierarchies to common attributes are equal.

Compatibility:

Fact tables are consistent if they have consistent functional dependencies.

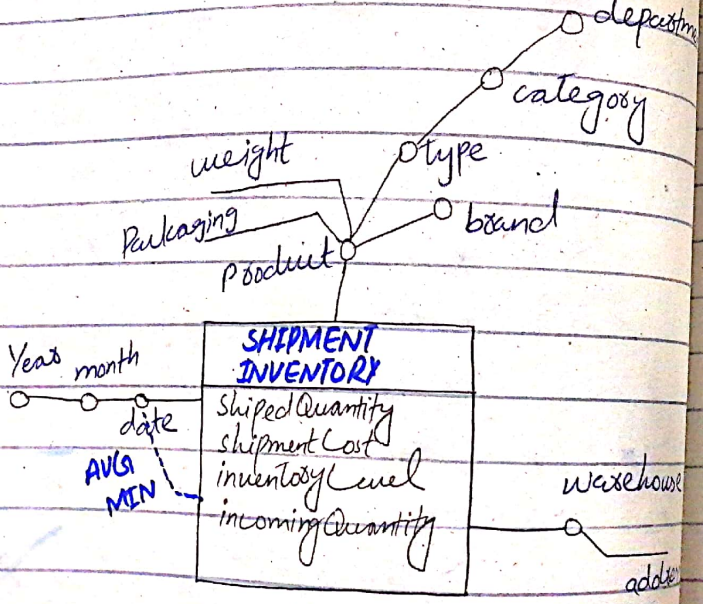
Overlapping:

Combining compatible fact tables creates a new fact table with shared dimensions and combined measures.

⇒ Overlapping two compatible fact schemata results in an overlapped fact schema in which:

1. Hierarchies are the common reduction of the hierarchies in source fact schemata.
2. Measures are the union of the measures in source fact schemata.
3. Each attribute domain is an intersection of the corresponding attribute domains in source fact schemata.

Ex: overlapping INVENTORY and SHIPMENT schemata.



Formalizing the Dimensional Fact Model:

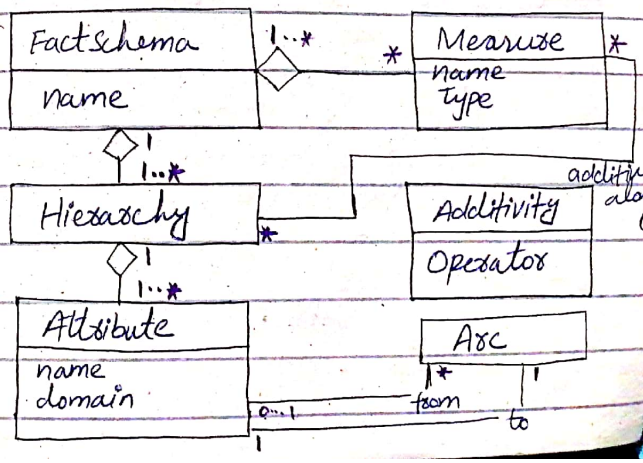
⇒ it defines how to make the Dimensional Fact model more organized and understandable.

⇒ The DFM is a way to arrange data in a multi-dimensional structure, often used in data warehouses.

(1) MetaModel:

- Simplified diagram of DFM. It is a blueprint that shows the different parts of the DFM and how they connect to each other.

ex: UML metamodel of the DFM



(2) intensional Properties:

⇒ Properties that define the mea... and relationships between the elem... of the DFM.

Hierarchy:

⇒ A hierarchy h is a pair $(A, \leq)$ where

- A is a finite set of dimensio... attributes
- $\leq_n$ is a partial order based on the attributes of A

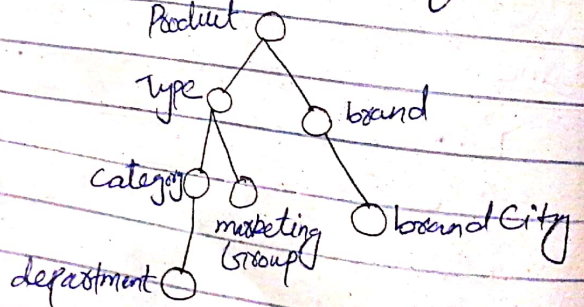
ex:

Product hierarchy

A = { Product, Type, category, depart..., marketingGroup, brand, brandCity }

Hasse diagram of hierarchy:

↳ visual representation of hierarchy



Fact Schema:

⇒ Let F be a fact schema. And it is define as a pair (H,M) in which:

- H is a finite set of hierarchies.
- M is a finite set of measures.

Group-by set:

⇒ Let F be a fact schema. a Group by set of F, is a subset of attributes, $G \subseteq Atts(F)$ such that for each pair of attributes a_i and a_i' contained in G from the same hierarchy h. neither $a_i \leq_n a_i'$ nor $a_i' \leq_n a_i$ holds.

Roll-up Order:

⇒ Given a fact schema F, a partial roll-up order $\leq$ on the set of all possible group by sets of F as follows:

- $G_i \leq G_i'$ if and only if $G_i \rightarrow G_i'$

(3) Extensional Properties:

⇒ Focuses on the actual data that fills the dimensional Fact model.

Hierarchy instance:

⇒ Given a hierarchy $h = (A, \leq_n)$, an

instance of h is a pair (D, L) where
- D is the set of domains Dom(ai) of the attributes in A.
- L is a family of roll-up functions that map each value of Dom(ai) for each pair of ai and aj attributes such that $a_i \leq_h a_j$

$$up_{a_j}^{a_i} : Dom(a_i) \longmapsto Dom(a_i)$$

ex:

Dom(city) = {'Abbyville', 'Auburn', ...}
Dom(state) = {'kanas', 'oklahoma', 'tarsonadi'}

$$up_{state}^{city}('Abbyville') = 'kanas'$$

Coordinate:

- $Dom(G) = Dom(a_i) \times \dots \times Dom(a_j)$

Cube & Primary Event:

- $C : Dom(Path(F)) \longmapsto Dom(m_1) \times \dots Dom(m_n)$

Aggregation and secondary events:

- $C^F : Dom(G) \longmapsto Dom(m_1) \times \dots \times Dom(m_n)$
- $c'(\alpha').m_i = \bigsqcup_{\substack{\alpha \in (G \sqcup F) \\ \alpha : up}} c(\alpha).m_i$