Data Modelling and Design

1 Introduction

Data Modelling is the process of discovering, analysing and scoping data requirements, and then representing and communicating these data requirements in a precise form called the data model.

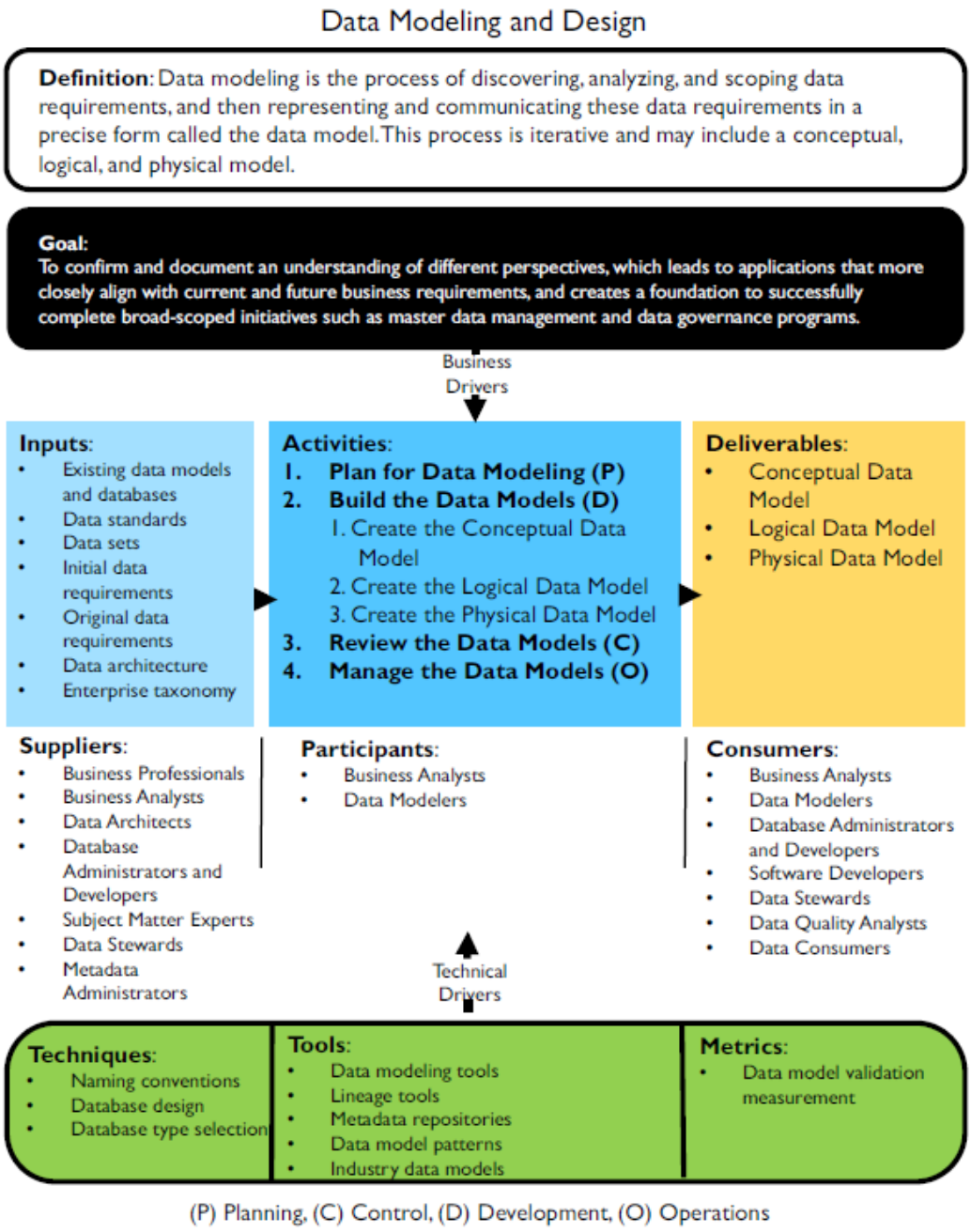
Six most commonly used schemes to represent data are:

- Relational
- Dimensional
- Object-Oriented
- Fact-Based
- Time-Based
- NoSQL

Models exist at three levels of detail:

- Conceptual
- Logical
- Physical

Each model contains a set of components such as entities, relationships attributes, facts and keys. Data Models contain Metadata uncovered during the modelling process which is essential to other data management functions.



1.1 Business Drivers

Data models:

- Provide common vocabulary around data
- Capture and document explicit knowledge about the organisation's data and systems
- Primary communications tool during projects
- Provide the starting point for customisation, integration or replacement of an application

1.2 Goals and Principles

Goal:
To confirm and document an understanding of different perspectives, which leads to applications that more closely align with current and future business requirements, and creates a foundation to successfully complete broad-scoped initiatives such as master data management and data governance programs.

Confirming and documenting understanding from different perspectives:

- **Formalisation**: A concise definition of data structures and relationships, how data is affected by implemented business rules. A disciplined structure reduces data anomalies occurring.
- **Scope definition**: Explain boundaries for data context in packages, projects or existing systems
- **Knowledge retention/documentation**: Preserves corporate memory regarding a system by capturing knowledge in an explicit form. Documentation for future projects. Understand the implications of modifications. Data models are reusable maps.

1.3 Essential Concepts

1.3.1 Data Modelling and Data Models

A model consists of diagrams of standard symbols which represent something that exists or something to be made. A data model describes the organisation's data as it is, or as how it wants it to be. Data models are the main medium to communicate data requirements from business to IT and within IT.

1.3.2 Types of data that are modelled

- **Category Information**: Data used to classify and define things (customers classified by market segment or products classified by colour)
- **Resource Information**: Master and Reference Data: objects needed to conduct operational processes such as Product, Customer, Supplier, Facility, Organization, and Account, Countries, Currencies
- **Business Event Information**: Transaction Data created while operational processes are in progress. Examples include Customer Orders, Supplier Invoices, Cash Withdrawal, and Business Meetings
- **Detail Transaction Information**: POS, Social Media, Clickstream, IoT event. Large volume and rapidly changing. Usually referred to as Big Data. Internet of Things – sensors etc.

1.3.3 Data Model Components

Basic building blocks of Data models:

- Entities
- Relationships
- Attributes
- domains

1.3.3.1 Entity

An entity is a thing about which and organisation collects information. A noun of the organisation. Questions to ask to identify entities:

Category	Definition	Examples
Who	Person or organization of interest. That is, *Who* is important to the business? Often a 'who' is associated with a party generalization, or role such as Customer or Vendor. Persons or organizations can have multiple roles or be included in multiple parties.	Employee, Patient, Player, Suspect, Customer, Vendor, Student, Passenger, Competitor, Author
What	Product or service of interest to the enterprise. It often refers to what the organization makes or what service it provides. That is, *What* is important to the business? Attributes for categories, types, etc. are very important here.	Product, Service, Raw Material, Finished Good, Course, Song, Photograph, Book
When	Calendar or time interval of interest to the enterprise. That is, *When* is the business in operation?	Time, Date, Month, Quarter, Year, Calendar, Semester, Fiscal Period, Minute, Departure Time
Where	Location of interest to the enterprise. Location can refer to actual places as well as electronic places. That is, *Where* is business conducted?	Mailing Address, Distribution Point, Website URL, IP Address
Why	Event or transaction of interest to the enterprise. These events keep the business afloat. That is, *Why* is the business in business?	Order, Return, Complaint, Withdrawal, Deposit, Compliment, Inquiry, Trade, Claim
How	Documentation of the event of interest to the enterprise. Documents provide the evidence that the events occurred, such as a Purchase Order recording an Order event. That is, *How* do we know that an event occurred?	Invoice, Contract, Agreement, Account, Purchase Order, Speeding Ticket, Packing Slip, Trade Confirmation
Measurement	Counts, sums, etc. of the other categories (what, where) at or over points in time (when).	Sales, Item Count, Payments, Balance

An **entity instance** is a particular occurrence of an **entity**.

Usage	Entity	Entity Type	Entity Instance
Common Use	Jane	Employee	
Recommended Use	Employee		Jane

Entites are represented graphically by **rectangles**

Definition of entites: (important as they are core Metadata). Definitions clarify the meaning of business vocabulary, and provide rigor to the business rules governing entity relationships.

- **Clarity:** Easy to read and grasp
- **Accuracy:** Precise and correct description of the entity
- **Completeness**: All parts of the definition are present. The scope of uniqueness is in the definition. e.g. examples of code values for a code entity.

1.3.3.2 Relationship

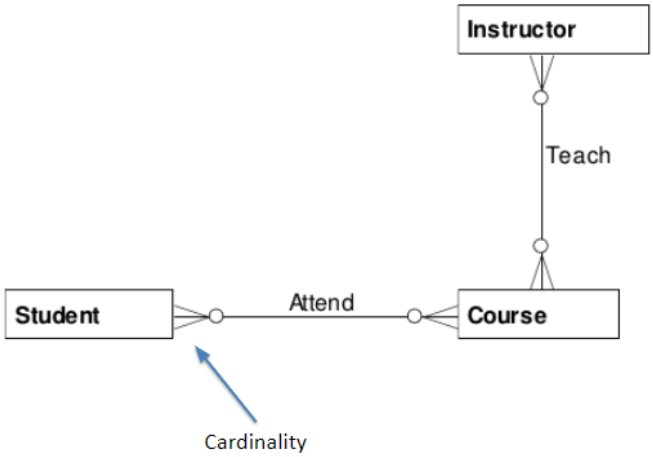
An association between entities.

Other aliases based on scheme:

- **Dimensional**: Navigation path

- **NoSQL**: Edge or Link
- **Relational on the physical level**: Constraint or Reference

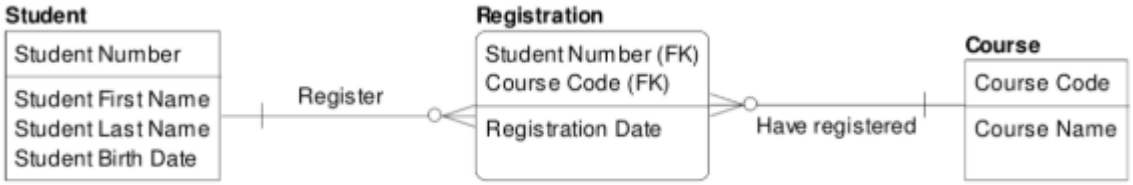
Graphically represented as lines on the diagram



Reading a Data Model: a 90 second video by Steve Hoberman
https://www.youtube.com/watch?v=adYohKb47f8

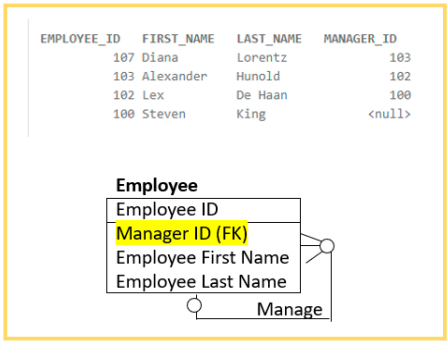
1.3.3.2.1 *Relationship cardinality*:

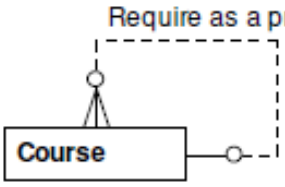
Data rules on how entities are connected are enforced. An instance of an entity may have a relationship with zero, one or many instances of another entity.



Unary (Recursive or self-referencing) relationship:

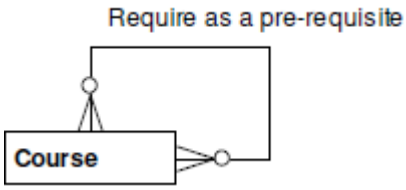
This is a nonidentifying, nonmandatory relationship in which the same entity is both the parent and the child. There is only one entity. A Foreign Key is used to distinguish between the roles or instances.





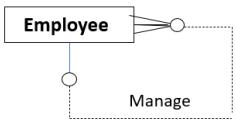
Hierarchy: One to many recursive relationship Child entity has at most 1 parent

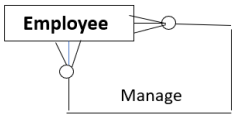
A course can be a prerequisite once

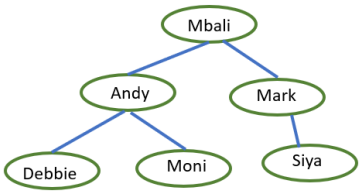


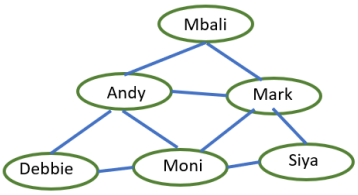
Network: Many to many recursive relationship. Entity instance can have more than one parent.

A course can be prerequisite for many courses

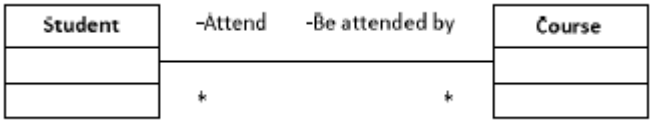




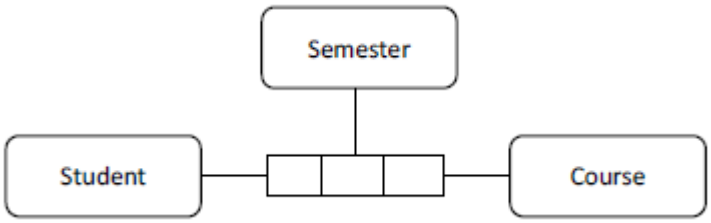




Binary Relationship: 2 Entities

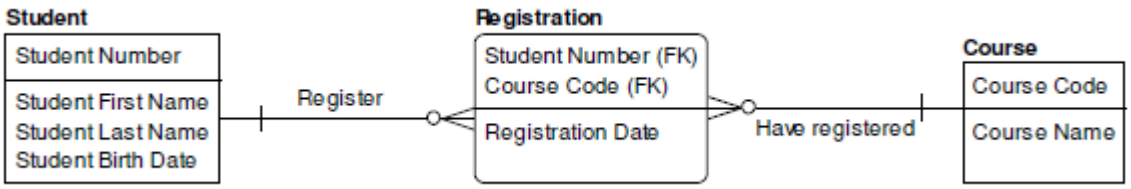


Ternary Relationship: 3 Entities.



1.3.3.2.2 Foreign Keys

Used in logical and physical data models to represent a relationship.



1.3.3.3 Attribute

A property that identifies, describes or measures an entity. A column in a table. May have domains. In a list as in Student above.

1.3.3.3.1 Identifiers (Key)

Set of one or more attributes that uniquely identifies and instance of an entity.

Types of keys:

- Construction type keys:
  - **Simple**: One attribute e.g. VIN number
  - **Surrogate:** Unique identifier, often system generated, without intelligence and not visible to end users
  - **Compound:** 2 or more attributes together e.g. *areacode+exchange+localnumber=phone number*
    IssuerID+AccountID+Check digit = Credit card number

    Each attribute is FK as for an associative entity or the PK on a Fact Table.

  - **Composite:** One compound key and at least one other simple or compound key or non-key attribute. e.g. a key on a multidimensional fact table. 2 or more attributes that identify an entity occurrence.
- Function type keys:
  - **Super Key**: any set of attributes that uniquely identify an instance
  - **Candidate key**: Minimal set of attributes that uniquely identifies an entity instance. Also called **Business** or **Natural** keys.
  - **Primary key**: The candidate key chosen to be the unique identifier. Often a surrogate key
  - **Alternate key**: Unique candidate key, but not chosen for primary. Usually a business key.

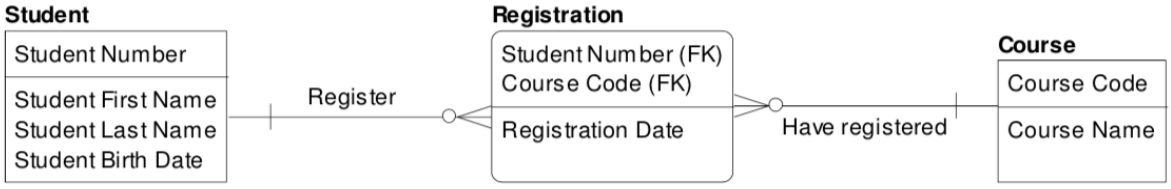
1.3.3.3.2 Identifying vs Non-Identifying Relationships

Independent Entity:

- Primary key contains only attributes belonging to that entity
- Non-identifying (weak) relationship ---------
- FK non-primary key attribute

Dependent Entity:

- The primary key contains at least one attribute from another entity
- Rectangles with rounded corners
- Identifying (strong) relationship _____



1.3.3.4 Domain

A domain is a complete set of possible values that an attribute can be assigned. It is used to standardise the characteristics of the attributes.

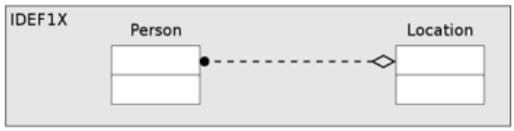
Domains can be restricted by adding **constraints** (additional rules). These can be format and/or logical.

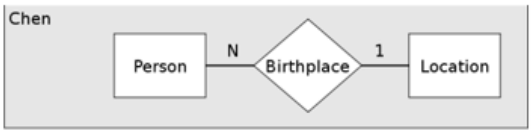
Defining domains:

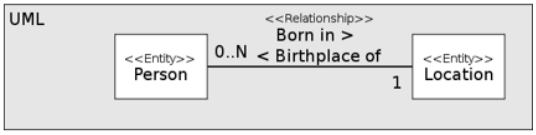
- **Data type:** standard type of data in an attribute assigned to that domain
- **Data Format:** domains using templates and character limitations
- **List:** Finite set of values
- **Range:** values of same data type between max and min values or can be open ended
- **Rule-based**: e.g. compare values to a calculated value

1.3.4 Data Modelling Schemes and Notations

Scheme	Sample Notations
Relational	Information Engineering (IE) Integration Definition for Information Modeling (IDEF1X) Barker Notation Chen
Dimensional	Dimensional
Object-Oriented	Unified Modeling Language (UML)
Fact-Based	Object Role Modeling (ORM or ORM2) Fully Communication Oriented Modeling (FCO-IM)
Time-Based	Data Vault Anchor Modeling
NoSQL	Document Column Graph Key-Value







Scheme	Relational Database Management System (RDBMS)	Multidimensional Database Management System (MDBMS)	Object Databases	Document	Column	Graph	Key-Value
Relational	CDM LDM PDM	CDM LDM	CDM LDM	CDM LDM	CDM LDM	CDM LDM	CDM LDM
Dimensional	CDM LDM PDM	CDM LDM PDM					
Object-Oriented	CDM LDM PDM		CDM LDM PDM				
Fact-Based	CDM LDM PDM	CDM LDM	CDM LDM	CDM LDM	CDM LDM	CDM LDM	CDM LDM
Time-Based	PDM						
NoSQL			PDM	PDM	PDM	PDM	PDM

The models which can be built for each scheme, based on the technology.

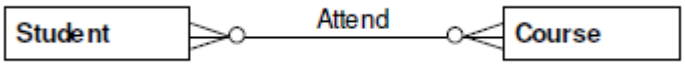
1.3.4.1 Relational

Dr Edward Codd (1970) proposed that data could be manages most effectively in terms of two dimensional relations. reducing redundancy and data storage. This approach was based on the mathematics of set theory.

Relationships capture business rules.

Exact expression of business data, and have one fact in one place (removal of redundancy). Ideal for operational systems requiring quick data entry, individual transaction processing and accurate storage.

Information Engineering (IE) syntax:

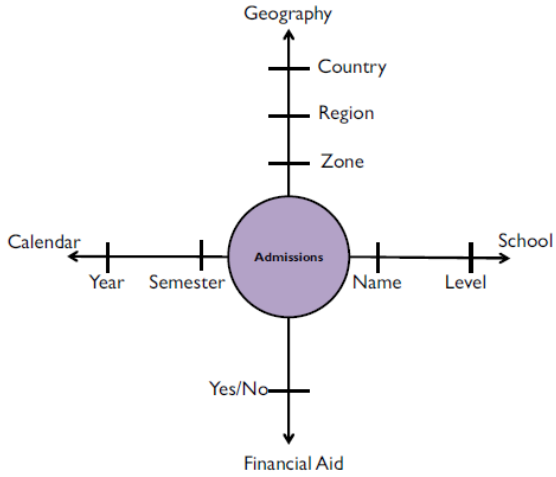


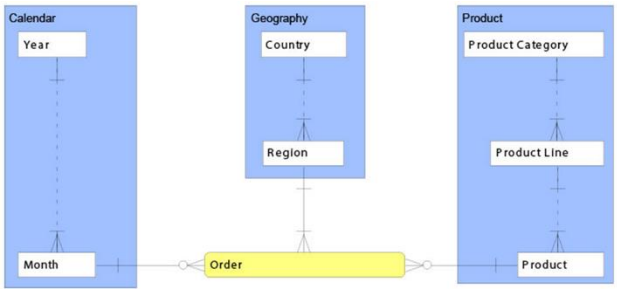
1.3.4.2 Dimensional

General Mills and Dartmouth College in the 1960s. Data is structured to optimise query and analysis of large amounts of data. Batch processing.

Dimensional models capture business questions focussed on a particular business process. Relationships capture navigation paths to answer the business question.

Axis (Peacock) notation diagram: Admissions

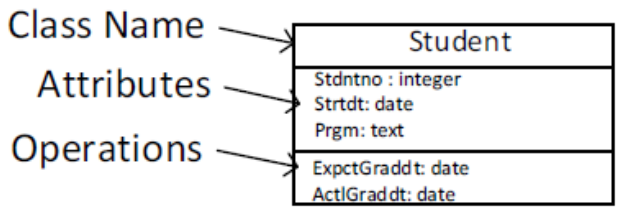


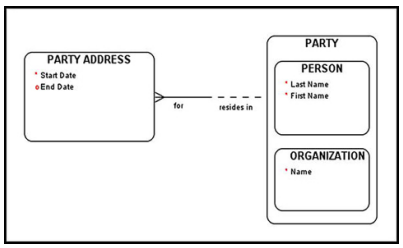


- **Fact Tables:** The measurement, result of calculations or algorithms. Is numeric. Metadata is important for understanding. Fact tables consist of a large number of rows. 90% of the data.
- **Dimension Tables:** Mostly textual. Used for querying. Highly denormalised. Must have a unique identifier for each row. Dimensions have attributes that change. Slowly Changing Dimensions (SCDs) manage changes based on the rate and type of change.
- **Snowflaking:** Normalising the star schema
- **Grain:** The most detail a row in the fact table can have
- **Conformed Dimensions:** Built with the entire organisation in mind and shared across dimensional models
- **Conformed Facts:** Use standardised definitions of terms across marts.

1.3.4.3 Object Oriented (UML)

Unified Modelling Language (UML) class model is used for databases. Specifies classes (entity types) and their relationship types.

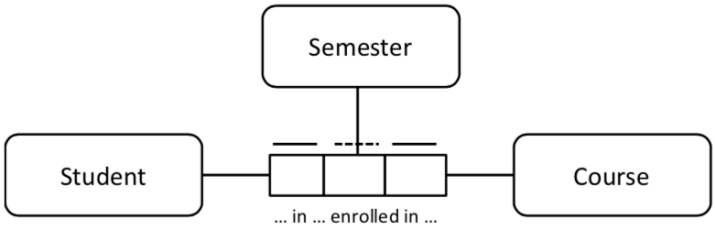




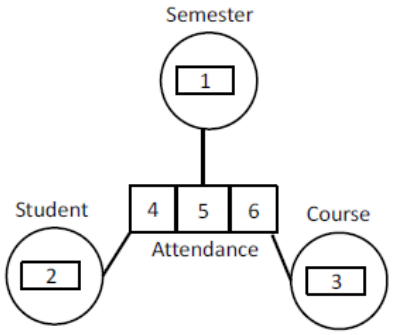
1.3.4.4 Fact based modelling

Based on natural verbalisation in the business domain:

- **Object role modelling (ORM):** Model driven engineering approach which verbalises required information at a conceptual level in a controlled natural language.



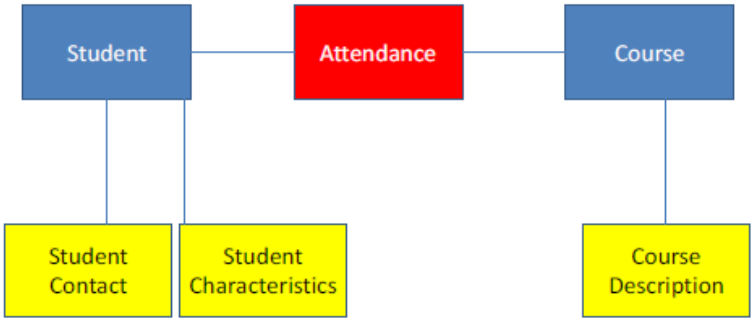
- **Fully communication oriented modelling (FCO-IM):**



1.3.4.5 Time-Based

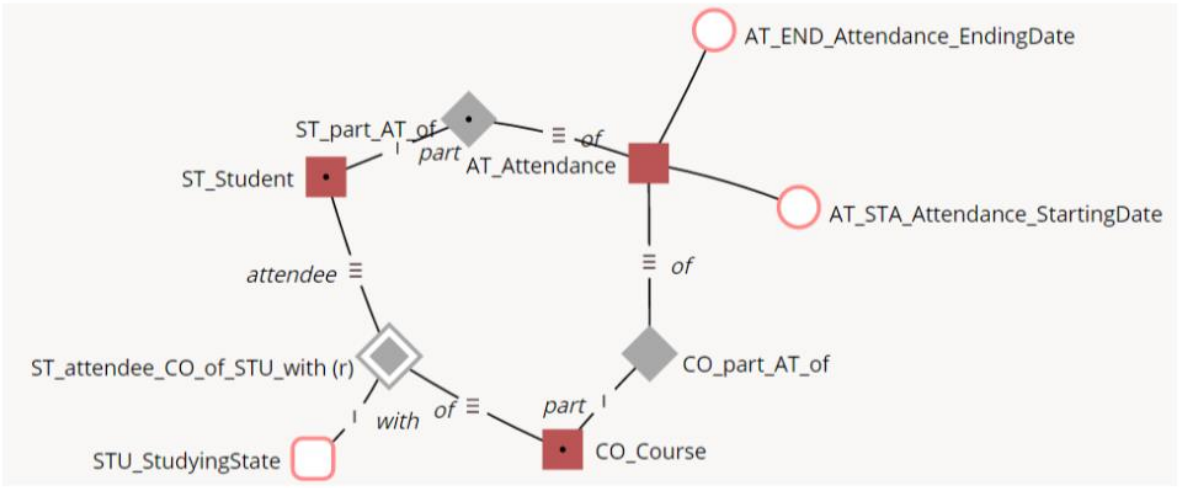
Time-based patterns are used when data values must be in chronological order and with time values. Used for data warehouses in a RDBMS environment.

- **Data Vault**: Detail oriented, time-based, normalised tables that support one or more functional areas of the business. Between 3NF and Star schema to meet needs of enterprise data warehouses.
  - **3 types of entities**:
    - **Hubs**: the primary key
    - **Links**: provide transaction integration between hubs
    - **Satellites**: provide the context of the hub primary key



- **Anchor Modelling**: Graphical notation for information that changes over time in both structure and content. Similar to traditional data modelling but with extensions for temporal data.
  - **4 basic concepts**:
    - **Anchors**: model entities and events
    - **Attributes**: model properties of anchors
    - **Ties**: model the relationship between anchors
    - **Knots**: model shared properties such as states

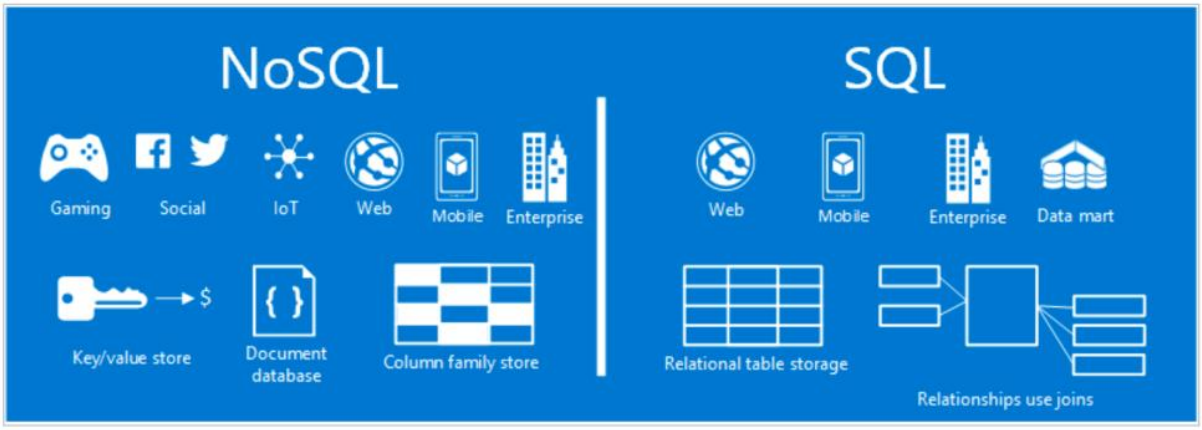
On the anchor model in Figure 45, **Student**, **Course**, and **Attendance** are anchors, the gray diamonds represent ties, and the circles represent attributes.



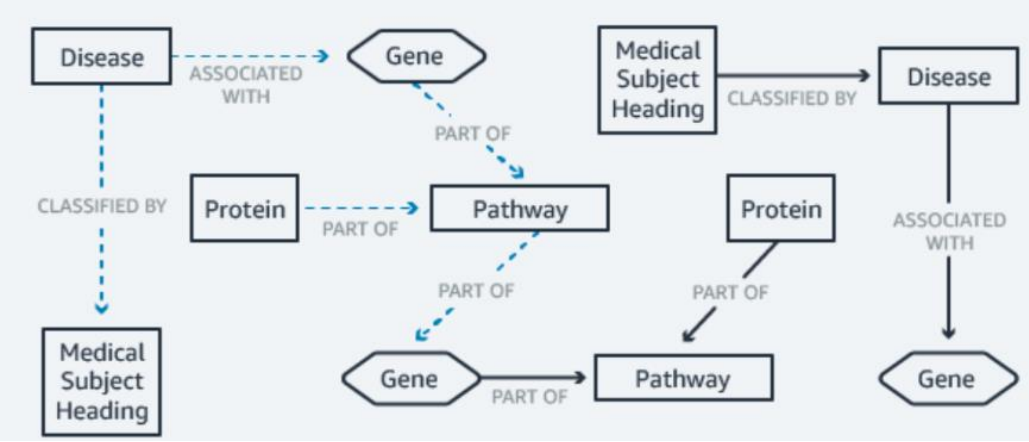
1.3.4.6 NoSQL

NoSQL is the name for databases built on non-relational technology. There are four types of NoSQL databases:

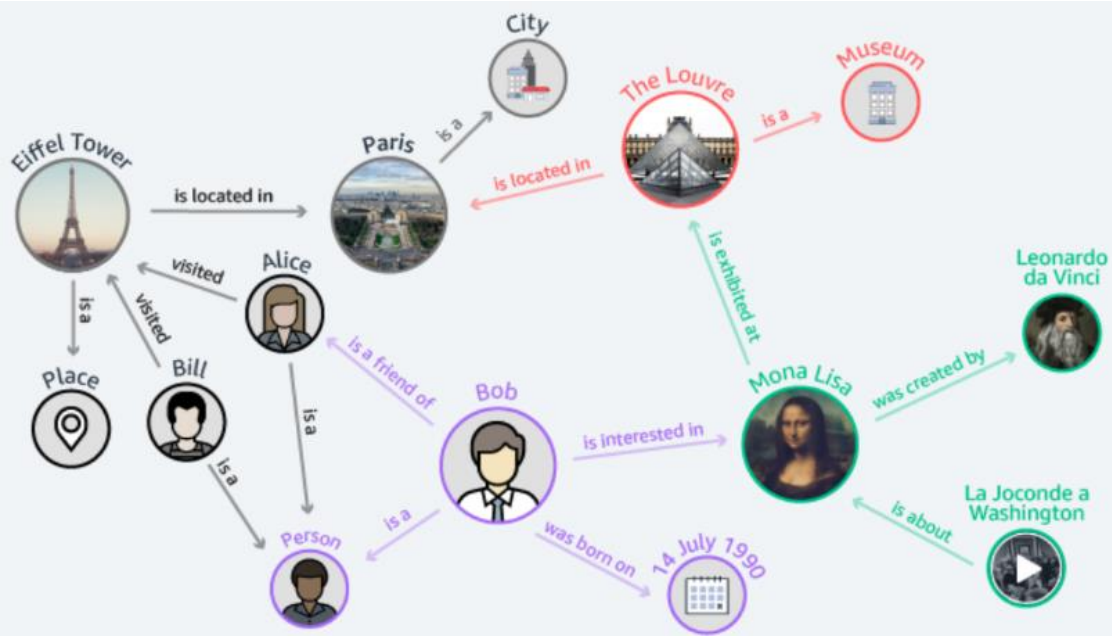
- **Document:**
  - Store the business subject in one structure called a document. For example, instead of storing Student, Course, and Registration information in three distinct relational structures, properties from all three will exist in a single document called Registration.
- **Key-value:**
  - Key-value databases allow an application to store its data in only two columns ('key' and 'value').
  - Value column can store anything (i.e. text OR video**)**
- **Column-oriented:**
  - RDBMSs work with a predefined structure and simple data types, such as amounts and dates, whereas column-oriented databases, such as Cassandra, can work with more complex data types including unformatted text and imagery
  - Store each column in its own structure
- **Graph**:
  - A graph database is designed for data whose relations are well represented as a set of nodes with an undetermined number of connections between these nodes



Knowledge Graph (Model)



Knowledge Graph (Data)



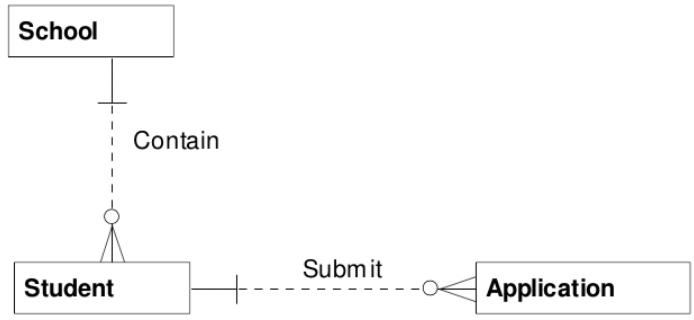
1.3.5 Data Model Levels of Detail

- **Conceptual** – Real world view of the enterprise modelled in the database

- **Logical** – Subsets of the enterprise model which represent the data requirements in a specific usage context, independent of technology.
- **Physical** – Internal or machine view. Describes the stored representation of the enterprise's data

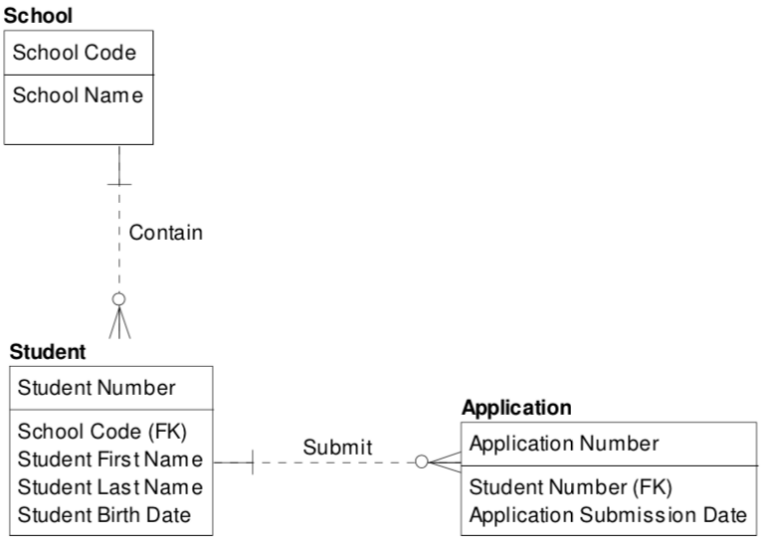
1.3.5.1 Conceptual (CDM)

The conceptual data model captures the high-level data requirements as a collection of related concepts. Basic business entities and the relationships between them (business rules) are described.



1.3.5.2 Logical (LDM)

A detailed, technology independent representation in a specific usage context. An extension of conceptual model. Attributes are assigned by applying normalisation.



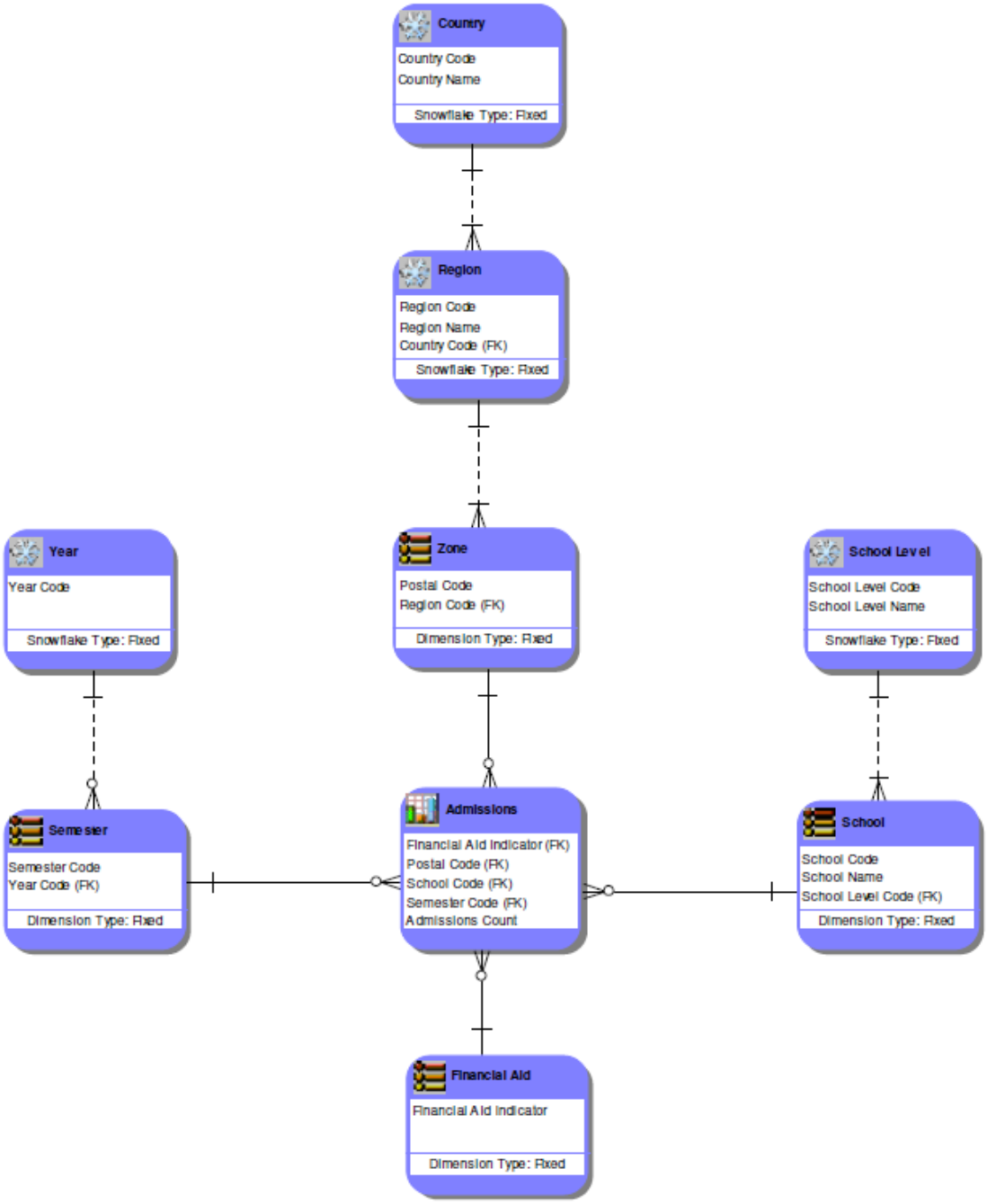
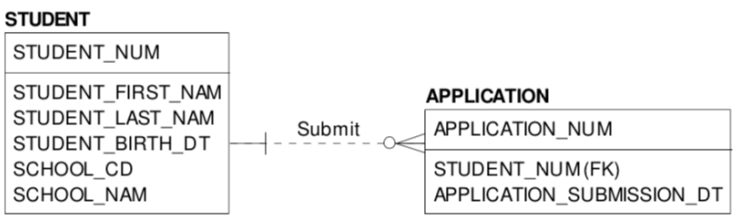


Figure 49 Dimensional Logical Data Model

1.3.5.3 Physical (PDM)

Detailed technical solution using the logical data model. Built for a particular technology of DBMS.



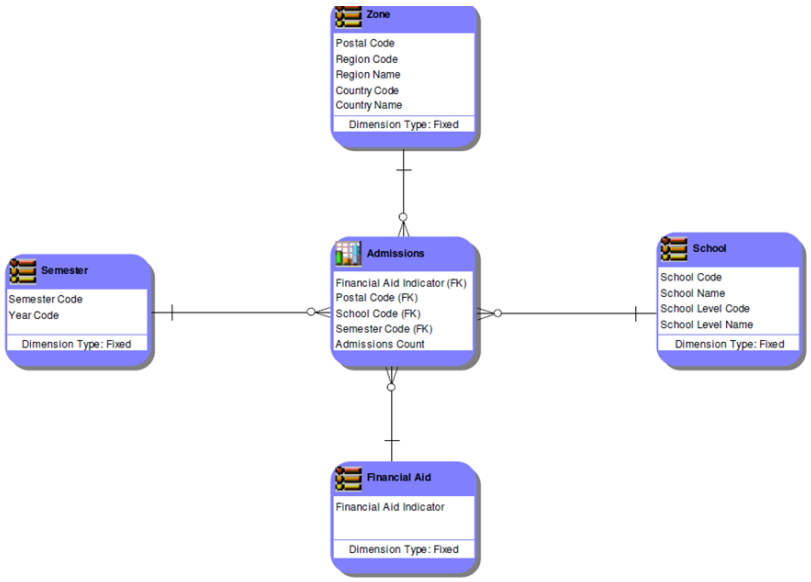


Figure 51 Dimensional Physical Data Model

A **Canonical Model** is a type of physical data model which describes the structure of data moving between systems. Structures should be generic and reusable. Used in the Enterprise Service Bus or Enterprise Application Integration (EAI).

Views are virtual tables used to simplify data access, control data access and rename columns without redundancy and loss of referential integrity due to denormalisation.

Partitioning: Process of splitting a table to improve performance

- **Vertically split**: subset tables contain subsets of columns
- **Horizontally split**: a value in a column is a delimiter to create subset tables

Denormalisation: Deliberate transformation of normalised logical model entities into physical tables with redundant data structures. Done to improve performance

1.3.6 Normalisation

Normalisation is the process of applying rules to organise business complexity into stable data structures. The basic goal is to eliminate redundancy by keeping each attribute in only one place. Requires understanding of attribute's relationship to its primary key.

Normalisation levels:

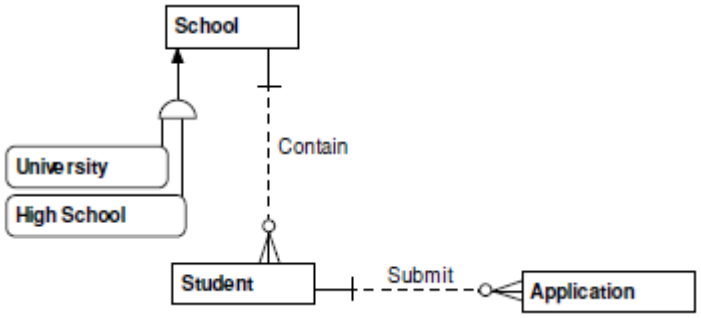
- **First normal form (1NF)**: Valid primary key. Each attribute depends on key. Eliminate repeating groups and ensure each attribute is atomic and not multi valued.
- **Second normal form (2NF)**: Each entity has the minimal primary key and each attribute depends on the complete primary key
- **Third normal form (3NF)**: no hidden primary keys. Each attribute depends on no attributes outside the key ("the key, the whole key and nothing but the key")
- **Boyce / Codd normal form (BCNF)**: Resolves overlapping composite candidate keys (hidden business rules)
- **Fourth normal form (4NF)**: Resolves all many to many relationships in pairs until they can't be broken down into smaller pieces.
- **Fifth normal form (5NF)**: Resolves inter-entity dependencies into basic pairs, and all join dependencies use parts of primary keys.

1.3.7 Abstraction

Abstraction is the removal of details in such a way as to broaden applicability to a wide class of situations while preserving the important properties, e.g. Party/Role structure

Includes:

- **Generalisation**: Groups common attributes and relationships into **supertype** entities
- **Specialisation**: Separates distinguishing attributes within an entity into **subtype** entities.
    - Subtypes can also be created using **roles** or **classification** e.g. Party with subtypes Individual and Organisation



2 Activities

2.1 Plan for Data Modelling

Tasks to plan for:

- Evaluating organisational requirement
- Creating standards
- determining data model storage

Deliverables of the modelling process:

- **Diagram:** The visual that captures the requirements in precise form. Depicts:
    - **Level of detail:** Conceptual, logical or physical
    - **Scheme:** Relational, Dimensional, Object-oriented, Fact-based, Time-based or NoSQL
    - **Notation:** information engineering, unified modelling language, object role modelling
- **Definitions**: for entities, attributes and relationships are essential for precision
- **Issues and outstanding questions**:
- **Lineage**: Where the data comes from.
    - A source/target mapping: Can see source system attributes and how they populate target system attributes
    - Trace components from conceptual to logical to physical
    - Data modeller obtains a good understanding of data requirements and can determine the source attributes
    - Source attributes can be used to check accuracy of the model and mapping

2.2 Build the data model

Modelling involves studying previous analysis, existing data models and databases etc.

Modelling is an iterative process: draft the model then go back to business analysts for clarification, update the model then ask more questions. This increases the precision of the model.

2.2.1 Forward Engineering

Build a new application from the requirements

- CDM: Understand the scope and key terminology
- LDM: Documents the business solution
- PDM: Documents the technical solution

2.2.1.1 Conceptual Data Modelling

Steps to create a CDM:

- **Select scheme:** Relational, dimensional, fact-based or NoSQL
- **Select notation:** Appropriate notation for selected scheme
- **Complete initial CDM:** Capture the viewpoint of the user group
    - Collect the highest-level concepts (nouns)
    - Collect the activities (verbs) that connect these concepts. Relationships can go both ways or involve more than two concepts.
- **Incorporate enterprise terminology:** ensure consistency with enterprise terminology and rules
- **Obtain signoff:** reviewed for data modelling best practices and that it meets requirements

2.2.1.2 Logical Data Modelling

LDM captures the data requirements within the scope of the CDM

- **Analyse information requirements:** elicitation, organisation, documentation, review, refinement approval and change control of business requirements.
- **Analyse existing documentation:** pre-existing artefacts provide a jump start for a new model
- **Add associative entities:** Used to describe the many-to-many relationships
- **Add attributes:** should be atomic
- **Assign domains:** allow for consistency of value sets and format
- **Assign keys:** Identify primary and alternate keys

2.2.1.3 Physical Data Modelling

LDM must be modified to perform well within storage applications

- **Resolve logical abstractions:** subtypes and supertypes become separate entities
- **Add attribute details:** Technical names, physical domain, physical data types and length of fields as well as constraints such as NOT NULL
- **Add reference data objects:** Small Reference Data value sets
- **Assign surrogate keys:** Unique key values not visible to business
- **Denormalise for performance:** dimensional structures
- **index for performance:** optimise query performance. Prevents every row being read (table scan)
- **Partition for performance:** Partition on a date key is usually recommended
- **Create views:** Requirements driven. Control access to certain data elements or embed joins or filters.

2.3 Review the Data Model

Apply a data quality verifier such as Steve Hoberman's Data Model Scorecard ® for quality control.

2.4 Maintain the Data Models

The data models need to be kept current. Update when business requirements or processes change. Compare physical with logical regularly.

3 Tools

- **Data Modelling Tools:** may support forward engineering from conceptual to logical to physical including DDL generation. Also reverse engineer. Some support naming standards and metadata storage.
- **Lineage Tools:** Capture and maintenance of source structures for each attribute in the data model. Excel is most commonly used.
- **Data Profiling Tools:** explores data content and validates it according to metadata and identifies data quality gaps/deficiencies
- **Metadata Repositories:** stores descriptive data about the model including diagram and definitions. Enables sharing.
- **Data Model Patterns:** Reusable modelling structures
- **Industry Data Models:** Pre-built for an entire industry. Needs to be customised.

4 Best Practices

4.1 Best Practices in Naming Conventions

ISO 11179 Metadata Registry is the international standard for representing metadata. Data architects, data analysts and database administrators jointly develop standards for an organisation, to complement related IT standards. Names should be unique and descriptive. Logical names must be meaningful to business users, whereas physical names must conform to DBMS restrictions.

4.2 Best Practices in Database Design

Design principles for the DBA (PRISM):

- **Performance and ease of use:** Quick and easy access
- **Reusability:** Multiple applications can use the data
- **Integrity:** Data should always have valid business meaning and value
- **Security:** Only available to authorised users
- **Maintainability:** ensure the cost of creating, storing, maintaining, using and disposing of data does not exceed its value to the organisation.

5 Data Model Governance

5.1 Data Model and Design Quality Management

Data models and database designs should be a reasonable balance between the short term needs and the long term needs of the enterprise.

5.1.1 Develop Data Modelling and Design Standards

Data modelling and database design standards help meet business data requirements, conform to Enterprise and Data Architecture and ensure data quality. Standards should include the following:

- List and description of standards data modelling and database design deliverables
- List of standard names, abbreviations and abbreviation rules
- List of standard naming formats for all data model objects
- List and description of standard methods of creating and maintaining these objects

- List and description of data modelling and database design roles and responsibilities.
- List and description of all Metadata properties captured in data modelling and database design
- Metadata quality expectations and requirements
- Too use guidelines
- guidelines for preparing and leading design reviews
- Guidelines for versioning models
- Practices that are discouraged

5.1.2 Review Data Model and Database Design Quality

- **Requirements Reviews:** Project team
  - Starting model
  - Changes made to the model
  - Rejected options
  - How well new model conforms to modelling and architectural standards
- **Design reviews:** Group of diverse subject matter experts
  - Chair meeting with an agenda to maintain order and move forward
  - Participants are given required documentation and chair solicits input
  - Summarise group's consensus finding
  - If there is no approval:
    - Modeller reworks
    - Final say should be given by the owner of the system

5.1.3 Manage Data Model Versioning and Integration

Preserve the lineage of changes:

- **Why** the project or situation required the change
- **What** and **how** the object changed
- **When** the change was approved and made to the model
- **Who** made the change
- **Where** the change was made (which model)

Modelling tool may have a repository which provides versioning and integration, else preserve models on DDL exports or XML files in a standard source code management system.

5.2 Data Modelling Metrics

Steve Hoberman's Data Model Scorecard® provides a way to measure the quality of a data model.

#	Category	Total score	Model score	%	Comments
1	How well does the model capture the requirements?	15			
2	How complete is the model?	15			
3	How well does the model match its scheme?	10			
4	How structurally sound is the model?	15			
5	How well does the model leverage generic structures?	10			
6	How well does the model follow naming standards?	5			
7	How well has the model been arranged for readability?	5			
8	How good are the definitions?	10			
9	How consistent is the model with the enterprise?	5			
10	How well does the metadata match the data?	10			
	TOTAL SCORE	100			

Description of each category:

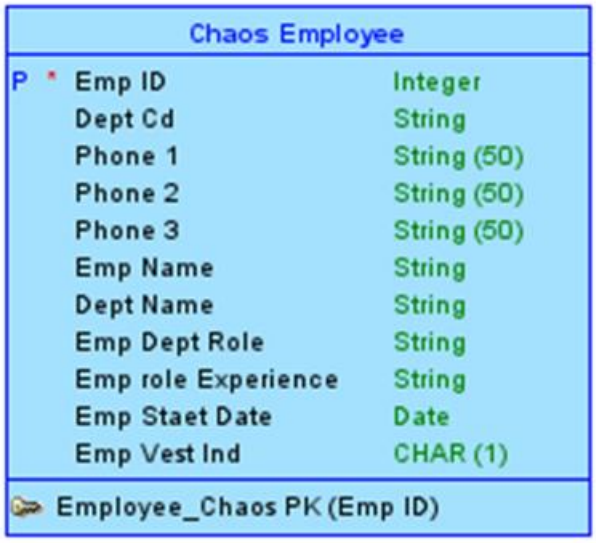
1. **How well does the model capture the requirements?** The model supports all required queries
2. **How complete is the model?** Completeness of requirements and completeness of Metadata, nothing extra
3. **How well does the model match its scheme?** Level of detail (conceptual, logical or physical) and the scheme (Relational, dimensional, NoSQL etc.) matches the definition of the type of model being reviewed.
4. **How structurally sound is the model?** Validate the design practices to ensure a database can be built
5. **How well does the model leverage generic structures?** Appropriate level of abstraction
6. **How well does the model follow naming standards?** Ensure correct and consistent naming standards have been applied to the model
7. **How well has the model been arranged for readability?** Parent entities above child, groupings and shorter relationship lines improve readability
8. **How good are the definitions?** Clear, complete and accurate
9. **How consistent is the model with the enterprise?** If one exists
10. **How well does the Metadata match the data?** Confirm the actual data to be stored fits in the model.

6 Normalisation Example (Steve Hoberman's Mastering Data Modelling Masterclass)

Normalisation

- **First normal form (1NF)**: Valid primary key. Each attribute depends on key. Eliminate repeating groups and ensure each attribute is atomic and not multi valued.
- **Second normal form (2NF)**: Each entity has the minimal primary key and each attribute depends on the complete primary key
- **Third normal form (3NF)**: no hidden primary keys. Each attribute depends on no attributes outside the key ("the key, the whole key and nothing but the key")
- **Boyce / Codd normal form (BCNF)** Resolves overlapping composite candidate keys (hidden business rules)
- **Fourth normal form (4NF)** Resolves all many to many relationships in pairs until they can't be broken down into smaller pieces.
- **Fifth normal form (5NF)**: Resolves inter-entity dependencies into basic pairs, and all join dependencies use parts of primary keys.

Chaos Logical Model



Chaos Employee Data

Emp ID	Dept Cd	Phone 1	Phone 2	Phone 3	Emp Name	Dept Name	Emp Dept Role	Emp Role Experience	Emp Start Date	Emp Vest Ind
1	DG, DM	083-676-9948	083-111-1112, 083-111-1113	083-111-1111	Howard Diesel	Data Governance, Data Modelling	EIM Manager, Data Modeller	5, 10	01-Jan-18	N
2	DM	082-875-5674	083-111-1113	083-111-1111	Veronica Diesel	Data Modelling	Data Modeller	6	01-Jan-18	N
3	MD	083-408-2593	083-111-1114	083-111-1111	Paul Grobler	Master Data	Master Data Manager	8	01-Jan-17	Y

1. Attribute Names are UNCLEAR & NOT single-valued
    1. Phone 1, Phone 2, Phone 3
2. Data IS NOT single-valued
    1. Dept Cd
    2. Phone 2
    3. Dept-Name
    4. Emp Dept Role
    5. Emp Role Experience

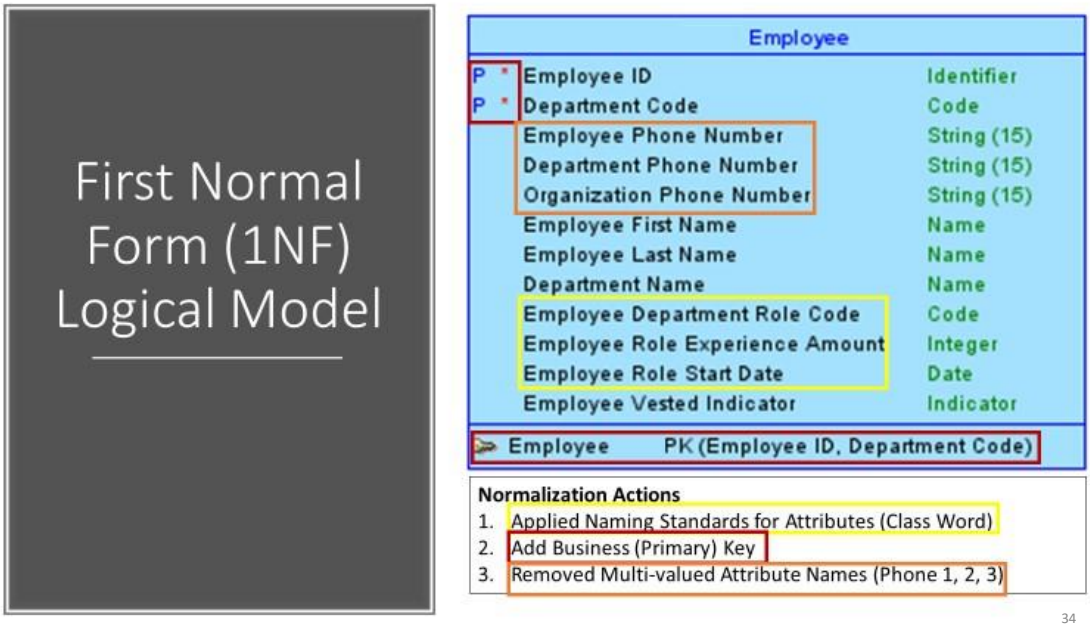
Normalization in a nutshell

Every attribute is single-valued (1NF) and depends completely (2NF) and only on its primary key (3NF).

Steve Hoberman

It is ALL about the Key, the WHOLE Key and NOTHING but the Key

So Help me Codd



1NF Data Observations

	Employee ID	Department Code	Employee Phone Number	Department Phone Number	Organization Phone Number	Employee First Name	Employee Last Name	Department Name	Employee Role Experience Amount	Employee Start Date	Employee Vested Indicator
1	1	DG	083-676-9948	083-111-1112	083-111-1111	Howard	Diesel	Data Governance	5	2018-01-01	N
2	1	DM	083-676-9948	083-111-1113	083-111-1111	Howard	Diesel	Data Modelling	10	2018-01-01	N
3	2	DM	082-875-5674	083-111-1113	083-111-1111	Veronica	Diesel	Data Modelling	6	2018-01-01	N
4	3	MD	083-408-2593	083-111-1114	083-111-1111	Paul	Grobler	Master Data	8	2017-01-01	Y

1. Attribute Names are CLEAR
    1. Department Code
    2. Role Experience Amount (Class Word)
2. Attribute Names are SINGLE-VALUED
    1. Phone 1 – Employee Phone Number
    2. Phone 2 - Department Phone Number
    3. Phone 3 – Organization Phone Number
3. Data IS SINGLE-VALUED
    1. No Data Separators

Second Normal Form (2NF)

Ensure minimal set of attributes that uniquely identify each entity instance

- Are all of the attributes in the primary key needed to retrieve a single instance of [[insert attribute name here]]?

Employee

Employee Identifier Department Code
Employee Phone Number Department Phone Number Organization Phone Number Employee First Name Employee Last Name Department Name Employee Start Date Employee Vested Indicator

Employee		
P *	Employee ID	Identifier
P *	Department Code	Code
	Employee Phone Number	String (15)
	Department Phone Number	String (15)
	Organization Phone Number	String (15)
	Employee First Name	Name
	Employee Last Name	Name
	Department Name	Name
	Employee Department Role Code	Code
	Employee Role Experience Amount	Integer
	Employee Role Start Date	Date
	Employee Vested Indicator	Indicator
Employee	PK (Employee ID, Department Code)	

Questions and Answers for 2NF

- Are all the attributes dependent on the WHOLE key (Employee ID and Department Code)
  - No
    - Employee Name is not dependent on Department Code
    - Department Code is not dependent on Employee Identifier
    - Organization Phone is not dependent on the Employee Identifier
    - Employee Role Experience not dependent on Department Code
    - Employee Role Start Date not dependent on Department Code
  - Yes (the WHOLE key: Employee ID & Department Code)
    - Employee Department Role Code

Second Normal Form (2NF) Logical Model – Step 1

Normalization Actions

1. Separated Employee & Department
2. Created a JOIN table to resolve MANY-TO-MANY

2NF Data

	Employee_ID	Employee_Phone_Number	Employee_First_Name	Employee_Last_Name	Employee_Start_Date	Employee_Vested_Indicator
1	1	083-676-9948	Howard	Diesel	2018-01-01	N
2	2	082-875-5674	Veronica	Diesel	2018-01-01	N
3	3	083-408-2593	Paul	Grobler	2017-01-01	Y

	Employee ID	Department Code	Employee Department Role Name	Employee Role Experience Amount	Employee Start Date
1	1	DG	EIM Manager	5	2018-01-01
2	1	DM	Data Modeller	10	2018-01-01
3	2	DM	Data Modeller	6	2018-01-01
4	3	MD	Master Data Manager	8	2017-01-01

	Department_Code	Department_Name	Department_Phone_Number	Organization_Phone_Number
1	DG	Data Governance	083-111-1112	083-111-1111
2	DM	Data Modelling	083-111-1113	083-111-1111
3	MD	Master Data	083-111-1114	083-111-1111

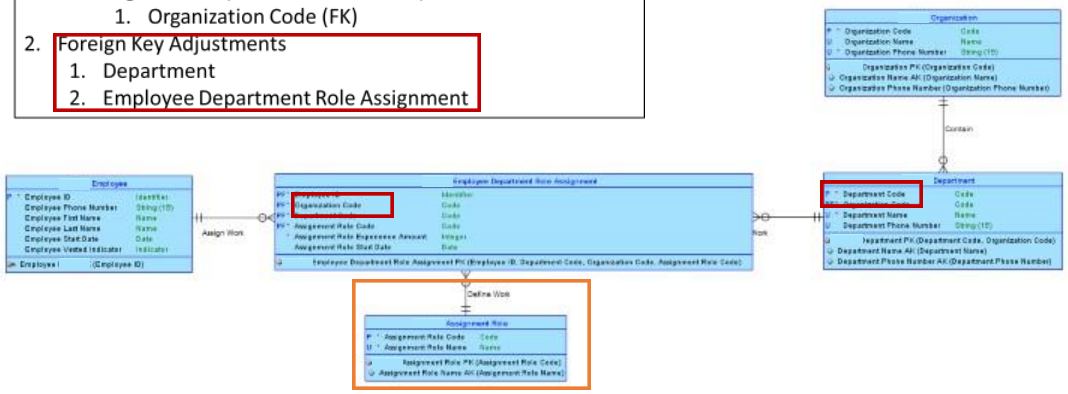
Questions and Answers for 2NF – Step 2

- Are all the attributes dependent on the WHOLE key (Employee ID and Department Code)
  - No (Employee Department Assignment)
    - Employee Department Role Name is not dependent on the Department Code
  - No (Department)
    - Organization Phone is not dependent on the Department Code
  - Ask Business:
    - Is "Role Experience" related the the Employee and Role or the Assignment Experience of the Employee to the Department

Second Normal Form- Step 2

Normalization Actions

1. WHOLE KEY
   1. Assignment Role
   2. Organization (No MANY-TO-MANY)
      1. Organization Code (FK)
2. Foreign Key Adjustments
   1. Department
   2. Employee Department Role Assignment



2NF Data: Step 2

	Employee_ID	Employee_Phone_Number	Employee_First_Name	Employee_Last_Name	Employee_Start_Date	Employee_Vested_Indicator
1	1	083-676-9948	Howard	Diesel	2018-01-01	N
2	2	082-875-5674	Veronica	Diesel	2018-01-01	N
3	3	083-408-2593	Paul	Grobler	2017-01-01	Y

	Employee ID	Organization Code	Department Code	Assignemnt Role Experience Amount	Assignment Role Start Date
1	1	MDS	DG	5	2019-05-09
2	1	MDS	DM	10	2019-05-09
3	2	MDS	DM	6	2019-05-09
4	3	MDS	MD	8	2019-05-09

	Department_Code	Organization_Code	Department_Name	Department_Phone_Number
1	DG	MDS	Data Governance	083-111-1112
2	DM	MDS	Data Modelling	083-111-1113
3	MD	MDS	Master Data	083-111-1114

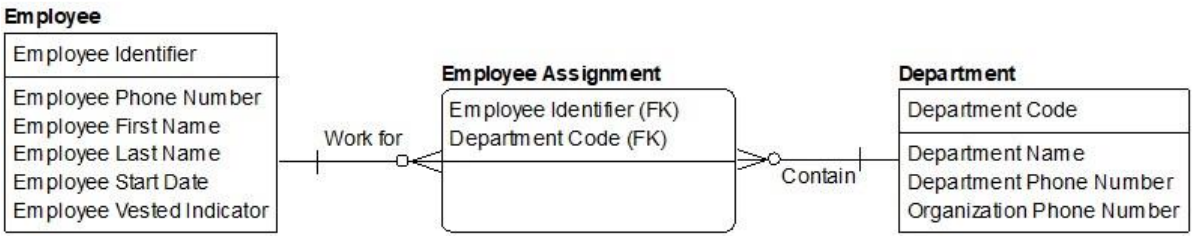
	Organization_Code	Organization_Name	Organization_Phone_Number
1	MDS	Modelware Systems	083-111-1111

	Assignment_Role_Code	Assignment_Role_Name
1	DM	Data Modeller
2	DG	EIM Manager
3	MD	Master Data Manager

Third Normal Form (3NF)

Remove hidden dependencies

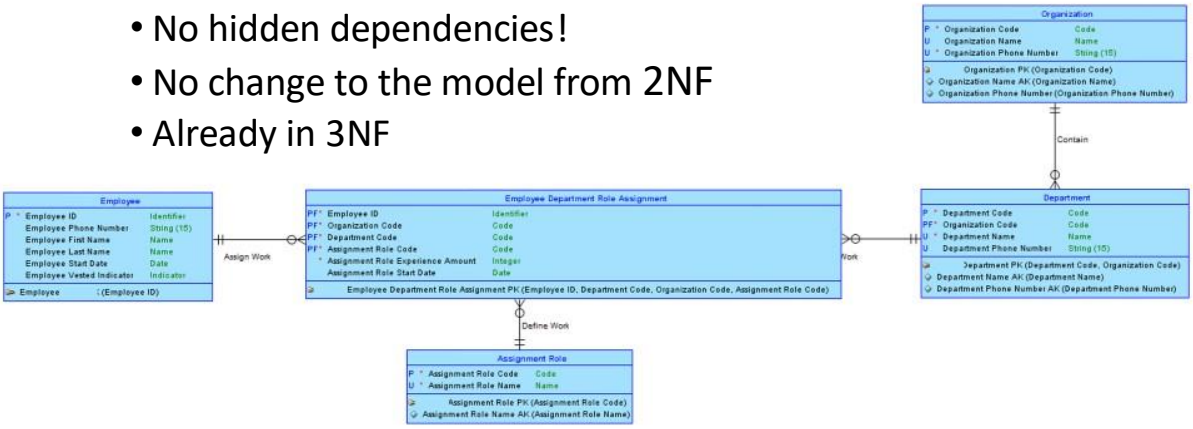
- Is [[insert attribute name here]] a fact about any other attribute in this same entity?



Third Normal Form (3NF)

Remove hidden dependencies

- No hidden dependencies!
- No change to the model from 2NF
- Already in 3NF



Third Normal Form Data

	Employee_ID	Employee_Phone_Number	Employee_First_Name	Employee_Last_Name	Employee_Start_Date	Employee_Vested_Indicator
1	1	083-676-9948	Howard	Diesel	2018-01-01	N
2	2	082-875-5674	Veronica	Diesel	2018-01-01	N
3	3	083-408-2593	Paul	Grobler	2017-01-01	Y

	Employee ID	Organization Code	Department Code	Assignment Role Name	Assignemnt Role Experience Amount	Assignment Role Start Date
1	1	MDS	DG	EIM Manager	5	2019-05-09
2	1	MDS	DM	Data Modeller	10	2019-05-09
3	2	MDS	DM	Data Modeller	6	2019-05-09
4	3	MDS	MD	Master Data Manager	8	2019-05-09

	Department_Code	Organization_Code	Department_Name	Department_Phone_Number
1	DG	MDS	Data Governance	083-111-1112
2	DM	MDS	Data Modelling	083-111-1113
3	MD	MDS	Master Data	083-111-1114

	Organization_Code	Organization_Name	Organization_Phone_Number
1	MDS	Modelware Systems	083-111-1111

	Assignment_Role_Code	Assignment_Role_Name
1	DM	Data Modeller
2	DG	EIM Manager
3	MD	Master Data Manager