|  |
| --- |
| Data Modeling Preparation |
|  |

# Capture Data Modelling Requirements

* Data models have been built during the analysis and design phases of a project to ensure that the requirements for a new application are fully understood.
* After the requirement gathering is done by the data modeler by understanding the business problem , the data modeler will develop models using the ER diagram(the components that are required and how they are related) for other the client and the developers
* Find Entity
* Find their Relationship (Like Many to many , many to one , one to many ,one to one )

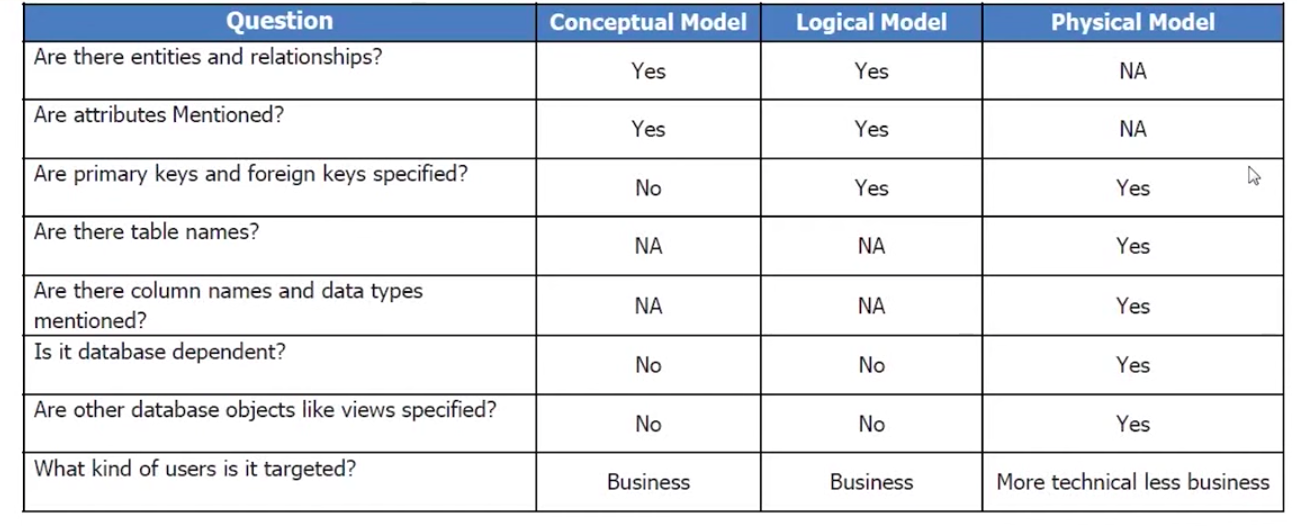
# CONCEPTUAL DATA MODEL

* In this model we first understand the business problem
* This model is something that Business users can understand easily
* Once the modeller and client agree then you move to making the logical model and this model does not have technical details

# LOGICAL DATA MODEL

* Includes relationships and entities which describes data in more details
* each entity will have attributes defined primary key for each entity will be defined and the foreign key linking the different entity will be defined.
* Tables can also be normalised n this model .
* Database independent

# PHYSICAL DATA MODEL

* Includes relationships and entities which describes data in more details
* When you move from logical to physical model Entities will become tables , attributes will become column names , data types for each columns will be specified along with precision and scale, primary key and foreign keys will be specified , column constraints will be specified . 

# Type of notation

* IDFX,INFORMATION ENGINEERING , UML

# Subject area in Erwin ?

# Type of relationship

* Identifying : The child cannot exist without parent
* non-identifying : Child may exist without parent ,
* recursive
* Sub type & Super Type : Like emp dim can have full time emp dim and part time entity dim

# How to resolve many to many relationship

* In erwin we have resolve relatioship option
* we can have 1 to many and many to one relation to resolve it
* it will have primary and fk relationship

# Roll Up :

When we do not find much difference between super type and sub type

# Roll Down :

When we do find much difference between super type and sub type

# Domain DICTIONARY:

We can create various domain type and assign to attribute to various entities.

When we can to change to a dome that would be applied automatically to the whole Model.

# Bulk editor :

when trying to edit many properties at one go . It helps. like editing table DEFINATION.

# Forward engineering

After the physical model is done , go to forward engineering .

Then select what went to create DLL for . Usually switch off the Erwin generated triggers

Then give the connection details of the DB .

It will create them in DB

# Reverse engineering

Go to reverse engineer ing option

Choose the db type (Oracle )

Select all the objects types to reverse engineer

Then give connection details

Password

We reverser engineer everything

# Name standard and Name Setting

Tools in erwin is name configuration

glossory

# Compare two model :

* Using complete compare wizard
* Compare the model
* Find the changes
* Generate the DDL for the changes

# Source to Target mapping :

* In the model , enable data movement inside model properties
* Create Data Movement sources
* Add new sources
* Provide the details of sources ( host , port ..etc )
* Add movement table to it
* This can be added manually to come from a local csv file or ERWIN model or a database via reverse engineering or a data mart
* Now columns will come automatically from those source tables
* After that go to a table in the model which we wan to map
* Find data source tab ( It comes if data movement is enabled )
* Add the add source
* Click on the table column and map to respective source
* Save the report

# DetaiL ETL mapping report and Communication

* Go to report designer
* Create a new report
* Select Table
* Select the properties for target table which needs to shows in the report. Like table name ,Its columns
* Choose the DS column property under column where we can note the transformation rule or ETL rule
* Choose the data source movement column for source table
* Save the report in CVS to share with Team

# Create ER mapping report and Communication

* Go to report designer
* Instead of table choose Subject area
* Choose ER diagram
* Under that choose the Graphical object
* Select Target Table and column
* Select data movement source
* DS Comment properties under target column properties to keep ETL rule mentioned
* We can export it to PDF or HTML. That will include diagram and All the mapping

# What is data profiling?

* Data profiling is the process of reviewing source data, understanding structure, content and interrelationships, and identifying potential for data projects.

# Types of data profiling

* Structure discovery

Structure discovery helps understand how well data is structured—for example, what percentage of phone numbers do not have the correct number of digits.

* Content discovery

Looking into individual data records to discover errors. Content discovery identifies which specific rows in a table contain problems, and which systemic issues occur in the data (for example, phone numbers with no area code).

* Relationship discovery

Tool Used ex: Informatica

# Best Practice of Data modeling

* Load detailed atomic data into dimensional structures.
* Structure dimensional models around business processes
* Ensure that every fact table has an associated date dimension table.
* Ensure that all facts in a single fact table are at the same grain or level of detail.
* Resolve many-to-many relationships in fact tables
* Resolve many-to-one relationships in dimension tables.
* Store report labels and filter domain values in dimension tables
* Make certain that dimension tables use a surrogate key.

# What are surrogate keys? Why would you use them in a data warehouse instead of using the business keys from the operational systems?

* There are various reasons why we cannot simply reuse our existing natural or business keys.  First, business keys usually have a business meaning in the OLTP systems, such as social security number for Employee and VAT number for Company.  Hence, they are tied to the business setting and requirements and if these would change (e.g., due to a merger or acquisition, a new legislation) then all tables using those keys need to be updated, which may be a resource intensive operation in a data warehouse environment, because not only the current state is stored, but also historical data.  Surrogate keys essentially buffer the data warehouse from the operational environment by making it immune to any operational changes.
* when compared to surrogate keys, business keys are usually bigger in size which will result in big indexes and slow down index traversal and, consequently, query execution time. using surrogate keys will save space and improve performance
* surrogate keys can also be successfully used to deal with slowly changing dimensions.

# Surrogate Key Cons

* Extra column(s)/index for surrogate key will require extra disk space
* Extra column(s)/index for surrogate key will require extra IO when insert/update data
* The surrogate key value can't be used as a search key
* Extra ETL

# Degenerated DimenSion

* To avoid an expensive join between this fact dimension and the fact table, the attribute of this dimension are degenerate i.e. moved from the dimension to the fact table
* <https://datacadamia.com/data/type/cube/modeling/degenerate>

# What are conformed dimensions?

* Conformed dimensions are the dimensions which can be used across multiple data marts in combination with multiple fact tables accordingly.
* A conformed dimension is a dimension that has exactly the same meaning and content when being referred from different fact tables. It can refer to multiple tables in multiple data marts within the same organization.

# Inmon and Kimball

* Kimball > First Data Marts > Combined Ways > Data Warehouse  
  Inmon > First Data Warehouse > Data marts

# Data Vault

<https://aptitive.com/blog/3-reasons-to-implement-a-data-vault-model-and-2-reasons-not-to/>

# How to Validate a data model once ready ?

* Model Validation Check

The following generic check model validation routines are implemented for most databases (some are database dependent):

* Tables without any columns, without any indexes, without any primary key columns, or has more than 128 indexes
* Indexes without any columns, having more than 15 columns, redundent
* Columns which refer to undefined datatypes, whose FK datatypes differ from the migrating parent PK datatype
* Keyword violations
* Use of invalid characters
* Invalid length of table name, index name, column name, constraint name, validation rule name, or default value name (based on database or user-defined restrictions)

# Check Your Model/Validate SQL

You can create a Model Validation report before you forward engineer your model. The validation process reports errors associated with correct forward engineering. You can also run the Model Validation report against your open model when you click the Forward Engineer - Check Model icon Forward Engineering Check Model and validate SQL on the Database Toolbar.

**To check your model/validate SQL**

1. Open the model that you want to forward engineer. On the Actions menu, click Forward Engineer, Check Model.

**Note:** For Logical/Physical models, switch to the Physical view.

The report opens in a new window. The report includes a header with the name of the model, the date of the model, and a list of relevant corrupt objects or errors.

1. Click Save As.

The report is saved in RTF format to an external file.

1. Click Close to close your report.

# Validation Rules

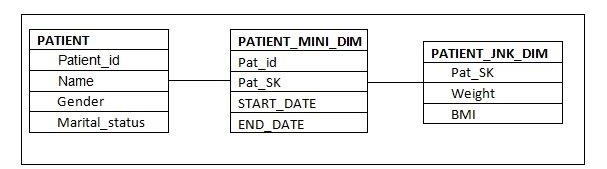
You can create validation rules to enforce the business rules of your organization. The rules that you create can establish a value, a list of values, or a range of acceptable values to store in a table, column, entity, or attribute. Use the Validation Rule Editor to create rules.

For example, you can create a validation rule that lists: **Comedy, Drama, Mystery, Horror, New Releases, Foreign, Science Fiction** as valid values for a column named**movie\_genre**. You can then attach this rule to one or more columns using the Table Column Editor. Then you can check that only these valid values are entered in these columns.

To assign a validation rule to a table, you can use the Validation tab in the Table Editor dialog.

# Rapidly changing Dimension and what is the use case and how to handle it

* A dimension is a fast changing or rapidly changing dimension if one or more of its attributes in the table changes very fast and in many rows. Handling rapidly changing dimension in data warehouse is very difficult because of many performance implications.
* As you know slowly changing dimension type 2 is used to preserve the history for the changes. But the problem with type 2 is, with each and every change in the dimension attribute, it adds new row to the table. If in case there are dimensions that are changing a lot, table become larger and may cause serious performance issues. Hence, use of the type 2 may not be the wise decision to implement the rapidly changing dimensions
* Separate Rapidly Changing Attribute by Implementing Junk Dimension
* The idea here is to separate the rapidly changing attribute from the slowly changing ones and move those attribute to another table called junk dimension and maintain the slowly changing attribute in same table. In this way, we can handle situation of increasing table size
* The attribute like patient\_id, Name, Gender, Marital\_status will not change or changes very rarely. And attribute like weight and BMI (body mass index) changes every month based on the patient visit to hospital. So, we need to separate the weight column out of the patient table otherwise we end up filling the table if we use SCD type 2 on PATIENT dimension. We can put the weight column which is rapidly changing into junk dimension table. Below is the structure of Junk dimension table
* https://dwgeek.com/rapidly-changing-dimension-data-warehouse.html/



* How to maintain Data Lineage

# sparsity and density

* Sparsity and density are terms used to describe the percentage of cells in a database table that are not populated and populated, respectively. The sum of the sparsity and density should equal 100%.
* A table that is 10% dense has 10% of its cells populated with non-zero values. It is therefore 90% sparse – meaning that 90% of its cells are either not filled with data or are zeros.
* Data explosion is a key issue in most of the OLAP server environments. It is closely related with the sparsity phenomenon in multidimensional model a relatively high percentage of the data points of the multidimensional space contain no value.
* There are two types of OLAP engines: Relational OLAP (ROLAP) and Multidimensional OLAP (MOLAP). In ROLAP, data is itself stored in a relational database whereas with MOLAP, a large multidimensional array is built with the data (Kaser & Lemire 2003). One difficulty with MOLAP is that the array is often sparse. These typically include provisions for handling sparse arrays, and they apply advanced indexing and hashing to locate the data when performing queries. ROLAP systems employing also specialized index structures, such as bit-mapped indices, to achieve good query performance (Pedersen et al. 2001).
* <https://studylib.net/doc/7392187/sparsity-handling-and-data-explosion-in-olap-systems>
* With huge number of dimensions there are chances of increasing sparsity in an OLAP as most of the intersections will not have data.
* Ex: In an insurance company of 100 emp .
* 4 Dimensions ( time, services, offices and sales Channels ) and one measure,
* We have 10 services, 10 offices and 10 channels . So Our input space consists of 10\*10\*10 = 1000 points for a specific moment of time
* It is possible some customers to use many services (offered in different offices), but most of the customers use no more than 2 services
* Means 100\*2=200 instances available for 1000 points
* But if we add a new dimension 10 more attribute then it becomes 10000 points but the number of emp remains same . Like Client Gender , education status .
* So the they have significant impact on the size of the multidimensional cube and the sparsity of the data.

Solutions :

* **Online or preliminary calculation**: precalcuated aggregated table or materialized view
* OLAP indexing: Bit Map or B tree index
* OLAP Compression Algo
* OLAP Partitioning:

# Type of Dimensions

* Role Playing Dimension
* Confirmed Dimension
* Degenerated dimension
* Junk Dimensions
* Slowly Changing Dimensions
* Rapidly Changing Dimensions
* Stacked dimensions
* Inferred Dimensions
* Shrunken Dimensions

# Example of Recursive relation ship from Your PROJECT?

# Data Redaction and Data Masking

* Oracle Data Masking Pack is mainly used for non-production environments to mask data sets in bulk. However, Data Redaction is used in production environments for redacting data in applications, without having to make changes to the application itself.