**Data warehouse:**  
Data warehouse is a central repository of each single data

Ex-

**Database VS Data Warehouse**

In the database we store the data in a table format. Suppose there is a source and the database continuously fetching the data from there. It is basically based on transaction. Every small transaction will be written on the database.

Now the main point is if some data analyst or data scientist want to perform query on the top of this database for building report then it will effectively degrade the performance of Database.

Now the Datawarehouse came into picture.

Suppose we have a source from where a a database received multiple data. Now a person wants to perform query on the top of that database. Now Data warehouse act like a big brother of database.

Now Datawarehouse will say that give the updated data and I will store it.Now the person will be able to perform query on the top of that Datawarehouse. So, it will reduce the degradation of the performance of Database

**What id Data warehousing ?**

It is a process to fetch data from database and load it to the Data warehouse by ETL

**ETL:  
  
In Data warehouse we have two layer that is called staging and core layer.**

**In the staging layer we just duplicate the raw data from the Source.  
  
In core layer we stored the Transformed data/Curated data and all the data analyst, data scientist perform query on this core layer.**

**Datamart are basically group data taken from the core layer of data warehousing according to the business requirement. You can say it is a subset of core layer of data warehousing.**

**Suppose we have transformed the data up to core layer. Now this whole data is not needed by whole organisation. So DataMart can hold data separately for HR domain, Finance domain ETC.So If the HR domain want to perform query top of that than they can perform**

Core

Transformation

Stagging

Data warehouse

Data Mart

Data mart

Data base

Refer Picture 1

**Incremental Loading:  
  
Refer picture 1  
  
Now we have data according to this table. You can see the last date 02-05-2025.**

|  |  |  |  |
| --- | --- | --- | --- |
| **id** | **name** | **domain** | **date** |
| **1** | **aa** | **Hr** | **01-05-2025** |
| **2** | **bb** | **QA** | **02-05-2025** |
| **3** | **cc** | **QA** | **01-05-2025** |
| **4** | **dd** | **QA** | **02-05-2025** |
| **5** | **ee** | **Hr** | **01-05-2025** |

**Now we load this data up to core layer. In the next day we have suppose three more record.**

|  |  |  |  |
| --- | --- | --- | --- |
| **id** | **name** | **domain** | **data** |
| **1** | **cc** | **Hr** | **03-05-2025** |
| **2** | **ee** | **QA** | **04-05-2025** |
| **3** | **dd** | **QA** | **02-05-2025** |
| **4** | **mm** | **QA** | **03-05-2025** |

**Now if we load this data over data ware housing then we are duplicating Data, right? We don’t want that. So here the come the concept of Incremental Loading. A term is introduced that Change Data Capture (CDC).**

**We are only loading the new record. Using this CDC, we store the maximum date into a particular location and when we fetch the data next day we are simply using select \* from table where date>02-05-2025**

**So it will fetch the data up to that day we don’t need to fetch whole data.**

**One important point:  
  
We need to truncate the stagging layer it is recommended for 90 % cases but 10 % cases it is needed because we want only to pull the incremental data**

**Stagging layer also has two layers:  
Persistent: It tells hold the data**

**Transient: It tell just delete the data**

**What is Data Modeling and why:  
  
In order to store data without redundancy, storage cost we have to build data model.**

**It is a process to to structure your data .**

**Types Of Data Model:**

**Conceptual-this type data model will be holding high level business need. Like we need a product table, sales table.**

**Logical-One step beyond the conceptual model. It specify how this table will be connected, what are the constrain. All the joining key, attribute all include in this logical model. All the ER . normalization take place here**

**Physical-It is actual physical table where the data is actually residing with all the logic/constrains**

**Going back to the logical model this normalization, ER all this thing is used in database world.For Data engineering we used Data dimensional Modelling.**

**What is Data dimensional Modelling;**

**It stores your data in the form of FACT and DIM table. In data modelling we have only one fact table and multiple dimension table**

**Fact Table:**

**>>It holds the data at granular level**

**>>Its basically allow only numeric column (Fact) and foreign key which connect your multiple dimension table.**

**Dimension Table:  
>>It hold only context.**

**>>We separate our data based on the context.**

**>>You want to find revenue by month you need to go dim\_date, find revenue by product you need to go dim\_product**

Dim date

Dim date

Dim customer

Dim product

Dim date

Dim2

Fact

dim

dim

dim

dim

Star Schema (**Point to noted there is no hierarchy)**

Snowflake schema (**there is hierarchy) how can we avoid this by joining the table with hierarchical table.**

**Types Of Fact Table:**

**Granular/Transactional Fact Table:  
1 transaction = 1 row. We do not do anything**

**Periodic Fact table/Snapshot fact table:**

**1 transaction = multiple rows. They decide a period basis on daily, monthly. they will take snapshot of fact table every month and they define 1 row = monthly transaction.**

**Accumulating Fact table:**

**1 row describe the whole journey of process.  
Ex:Amazon**

**Suppose there is row which is defining order id, packed date, delivery date etc.**

**Types Of Dimensional Table;  
  
Conformed dimensions:**

Dim products

**One dimension is shared between**

**Multiple fact tables**

Fact Order

Fact Cancelled

**Role Playing Dimension:  
when we connected our dimension table wit fact table with multiple condition/relationship**

Dim table

Fact Table

**Order date**

**Cancelled order date**

Active relationship

So we have to relationship one ins active and another one is inactive. You have the leverage to use the inactive relationship .this is how this dim table playing role.

JUNK Dimension:

|  |  |
| --- | --- |
| country | region |
| A | x |
| B | x |
| B | x |
| A | x |
| A | x |
| A | x |

Now if we create dimension table on top of this it will create dimension table like below:

|  |  |
| --- | --- |
| A | X |
| B | x |
|  |  |

So this type of dimension table called junk dimension which consist of few table

Degenerate Dimension:  
Let suppose in our table there is order ID. there is context on top of that. So, if we create a dimension table then it has only one column.

This type of dimension is called Degenerate dimension

SCD:  
Type-1

Type-2

Type -3

Type-1:(Upsert= update =insert)

Before:



|  |  |  |
| --- | --- | --- |
| Product-ID | Product-name | Product\_cat |
| 1 | Honey | Food |
| 2 | Shirt | Clothing |
| 3 | Comb | Clothing |

After:



|  |  |  |
| --- | --- | --- |
| Product-ID | Product-name | Product\_cat |
| 1 | Honey | Food |
| 2 | Shirt | Clothing |
| 3 | comb | Hair |

We change the product cat for comb for comb. When we have existing key we just need to update the value and when we have new record insert that value.

Type 2(History):

The team need to check the record previous history.For that they maintain three column effective\_start\_date, effective\_end\_date, inUse  
Before

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Product-ID | Product-name | Product\_cat | Eff\_start-date | Eff\_end-date | inUse |  |  |
| 1 | Honey | Food | 01/01/2024 | 01/01/2090 | Yes |  |  |
| 2 | Shirt | Clothing | 01/01/2024 | 01/01/2090 | Yes |  |  |
| 3 | Comb | Clothing | 01/01/2024 | 01/01/2090 | Yes |  |  |

After

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Product-ID | Product-name | Product\_cat | Eff\_start-date | Eff\_end-date | inUse |  |  |
| 1 | Honey | Food | 01/01/2024 | 01/01/2090 | Yes |  |  |
| 2 | Shirt | Clothing | 01/01/2024 | 01/01/2090 | Yes |  |  |
| 3 | Comb | Clothing | 01/01/2024 | 01/01/2024 | No |  |  |
| 3 | | Comb | Hair | 01/01/2024 | 01/01/2090 | Yes |