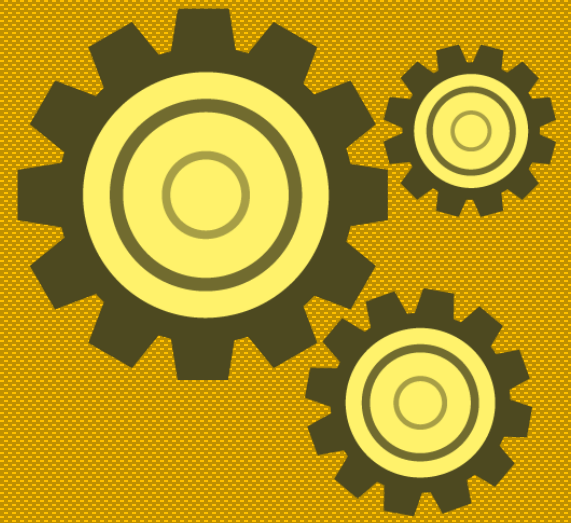
# ETL TESTING



# DWH Basics

# What is a Data warehousing?

- Data warehousing combines data from multiple, variable sources into one database which can be easily manipulated.
- This can be used for Analysis, Transformation and Reporting.
- Usually Larger companies use this data warehousing for analyzing trends over a period of time for viewing transactional data and plan accordingly.
- Data warehouse is defined as subject oriented, Integrated, Time Variant and Non-Volatile collection of data for the management for decision making processing.

# Characteristic features of a Data warehouse

- Subject Oriented: This is used to analyze particular subject area.
- Integrated: This shows that integrates data from different sources.
- Time variant: Historical data is usually maintained in a Data warehouse, i.e. retrieval can be for any period. This usually contrasts with the transactional system, in which only the most recent data is maintained. But in the Data warehouse, the transactional data is moved periodically where the recent and the previous data is also maintained.
- Non-Volatile: Once the data is placed in the data warehouse,it cannot be altered, which means we will never be able to alter the historical data.

# OLTP

- OLTP Online Transaction Processing System
- OLTP is nothing but a database which actually stores the daily transactions which is called as the current data.
- Usually OLTP is used for more of the online applications where the application needs to update very frequently in order to maintain consistency in the data.
- OLTP deals with the large number of data.
- The data in these systems are Normalized.

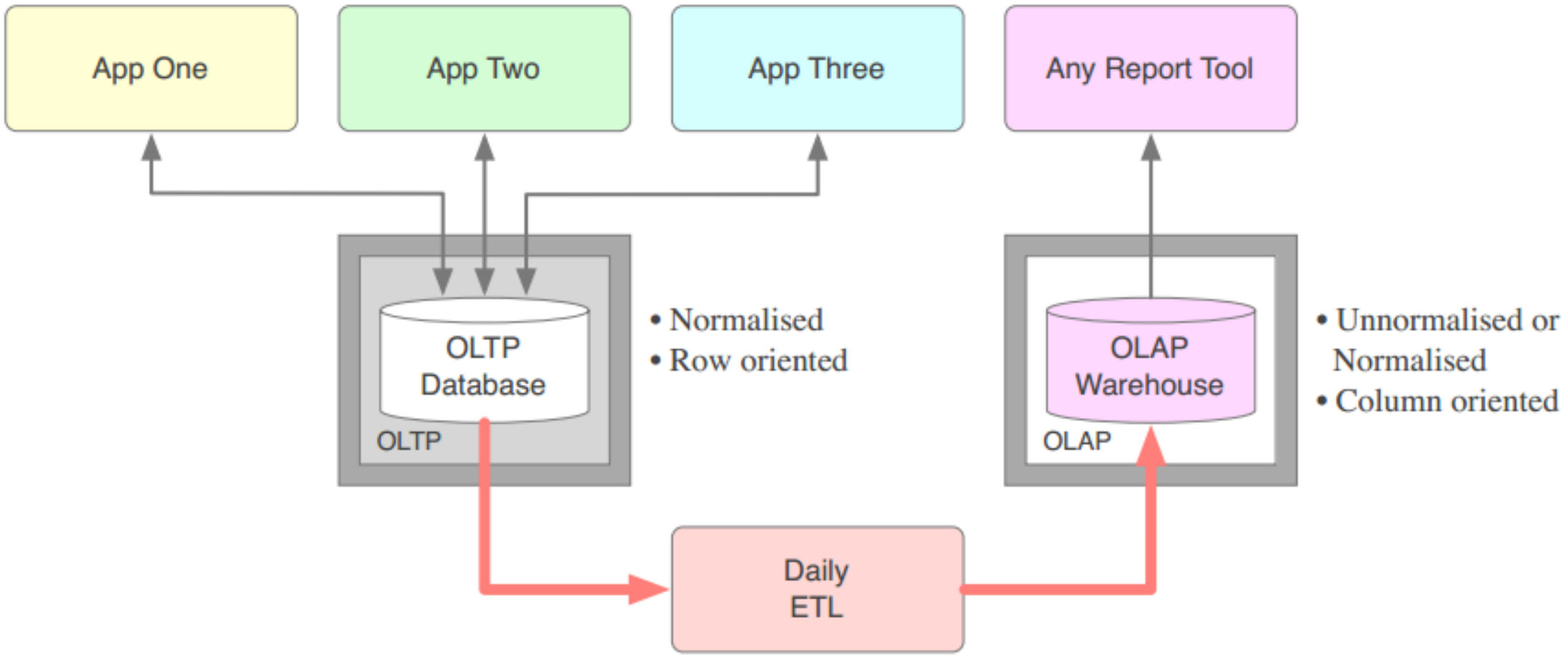
# OLAP

- OLAP Online Analytical Processing System
- OLAP deals with analyzing the data for Decision making and planning. This actually deals with the aggregations and the data in OLAP will be from different data sources
- Compared to OLTP, OLAP deals with relatively small amount of data.
- The data in these systems are De-Normalized.

# OLAP V/S OLTP

| Operational System (OLTP) | Data warehouse (OLAP) |
|---|---|
| It is designed to support business transactional processing. | It is designed to support decision making process. |
| Application oriented data | Subject oriented data |
| Current data | Historical data |
| Detailed data | Summary data |
| Volatile data | Non-volatile data |
| Less history (3-6 months) | More history (5-10 years) |
| Normalization data | De-normalization data |
| Designed for running the business | Designed for analyzing the business |
| Supports E-R modeling | Supports Dimensional modeling |
| Clerical users can access this data | Knowledge users can access this data |
| DB Size – 100MB-GB | DB Size – 100GB-TB |
| Few Indexes | Many Indexes |
| Many Joins | Some Joins |

# Typical Data Warehouse



# Difference between Database and Data warehouse

- Database: This can be treated as an OLTP system where it stores the data from the applications for use. Small applications can run with a database.
- Data warehouse: This is accumulation of the all the data related to the application, It can be from the Database where the Transaction data resides, Flat files that are used, Legacy or Mainframe sources. Large organization will need a data warehouse where there is a need for analytics for decision making.

# Database V/S Data warehouse

| Database | Data warehouse |
|---|---|
| Database has the current data which has a chance of Updating day by day. | Data warehouse Stores the Historical data where the accuracy is maintained over period of time. |
| Contains the Day to day operations data. | Contains the Long term Operations data. |
| Database professionals, agents access this particular data. | Managers, Analysts access the data warehouse data |
| Data in the Database will always be in the form of Input. | Data in the Data warehouse will always be the output data where it is used for Analyzing trends. |
| Database is used for Transactions. | Data warehouse is used for Analytics. |
| Database has both Read/Write access. | Data warehouse has only the Read access. |
| Database contains only few number of records compared to a Data warehouse. | Data warehouse contains millions of records as the DWH gets refreshed with historic data. |
| Database is always Normalized. | Data warehouse is Denormalised |
| The data view in the database will always be Relational. | The data view in the Data warehouse id always Multidimensional |
| Database contains the detailed data. | Data warehouse contains the consolidated data. |

# ETL vs Database Testing

- Both ETL testing and database testing involve data validation, but they are not the same.
- ETL testing is normally performed on data in a data warehouse system, whereas database testing is commonly performed on transactional systems where the data comes from different applications into the transactional database.

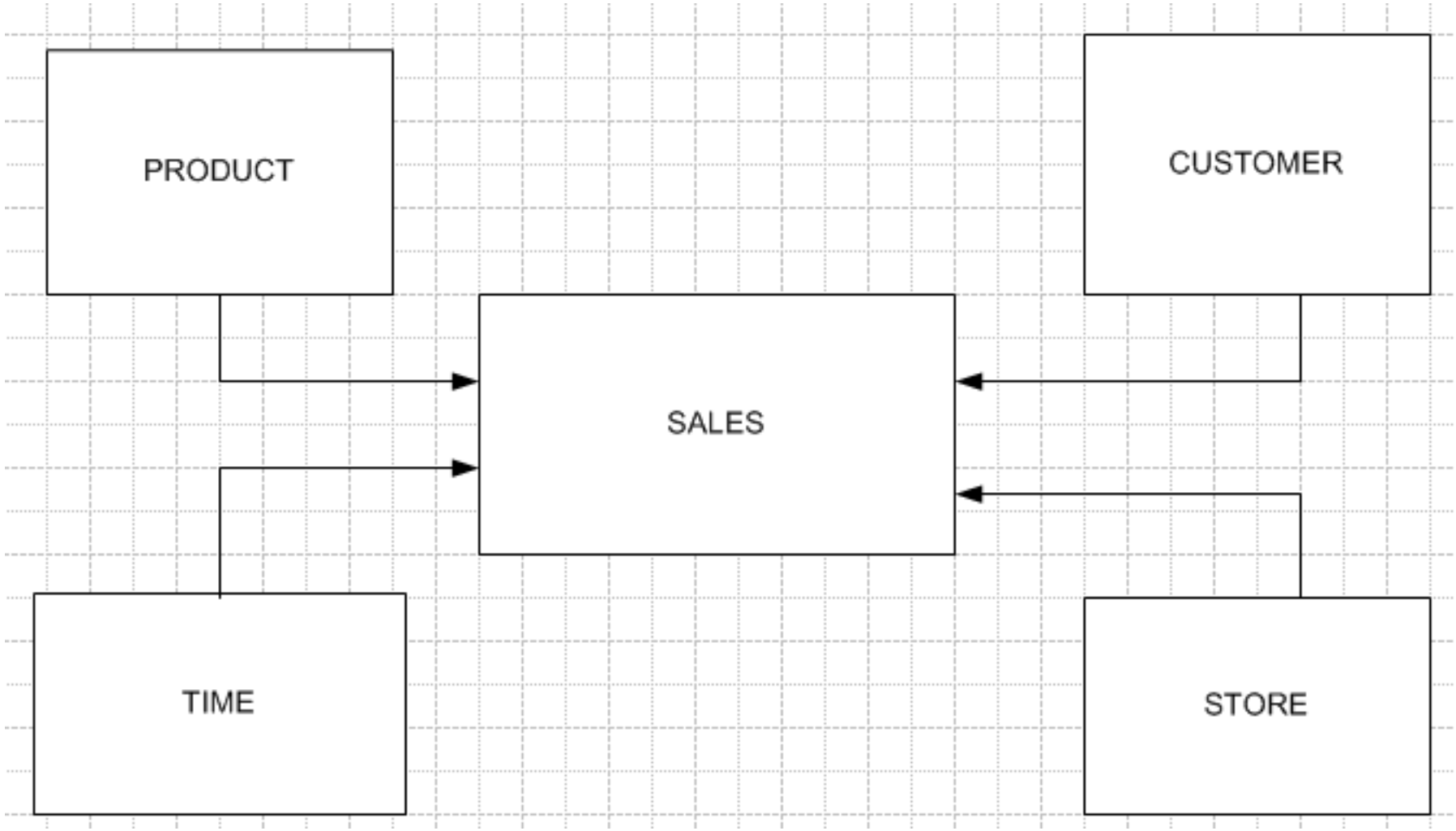
# DWH Concepts

# Data modelling concepts in data warehousing

- Data model tells how the logical structure of a database is modeled. Data Models are
- fundamental entities to introduce abstraction in DBMS. Data models define how data is connected to each other and how it will be processed and stored inside the system.
- Types of Data Models:
  - Conceptual Data Model
  - Logical Data Model
  - Physical Data Model

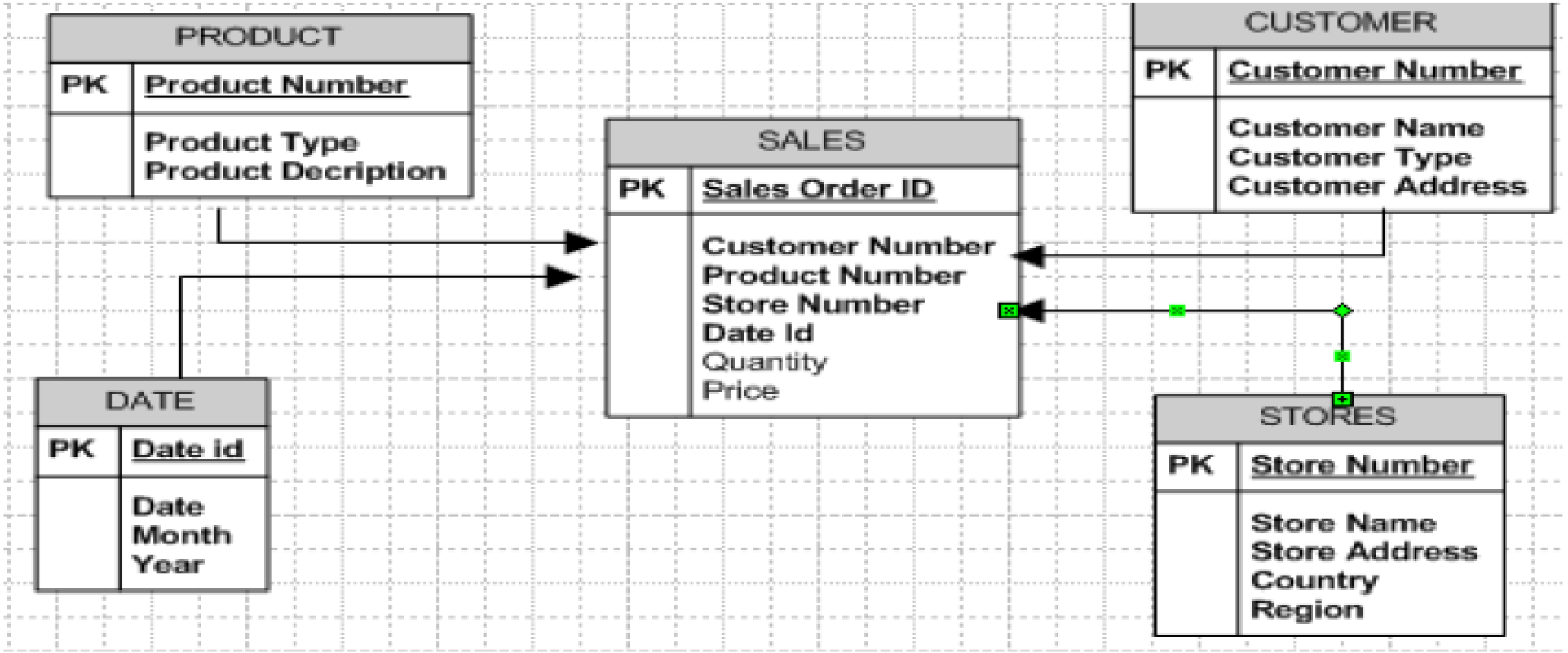
# Conceptual Data Model

- A conceptual data model identifies the highest-level relationships between the different entities. Features of conceptual data model include:
  a) Displays the important entities and the relationships among them.
  b) No attribute is specified.
  c) No primary key is specified.


PRODUCT
CUSTOMER
SALES
TIME
STORE

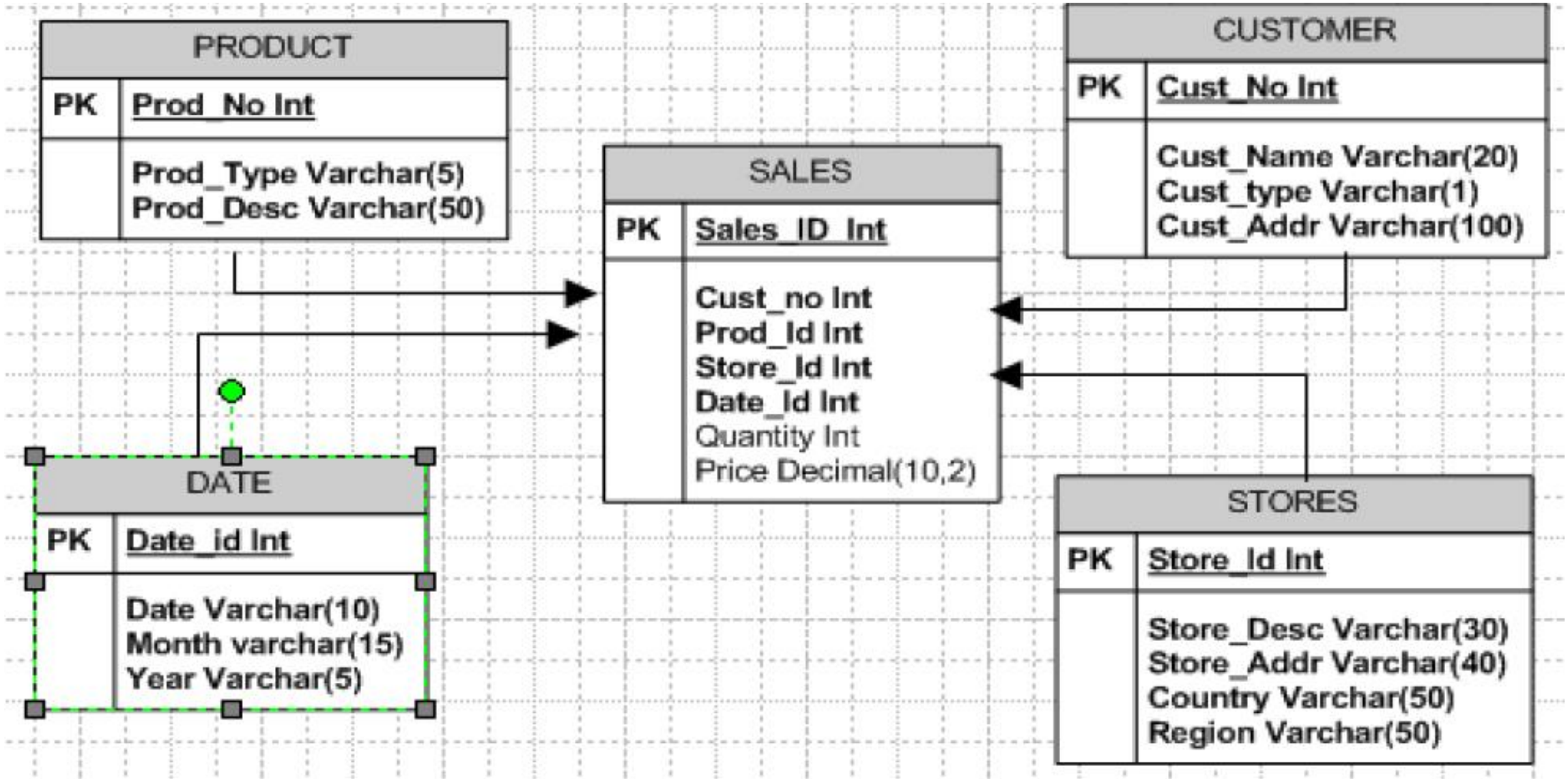
# Logical Data Model

- This defines the data as much as possible, to show how they can be physically implemented in the database.
  a) Includes all entities and relationships among them.
  b) All attributes for each entity are specified.
  c) The primary key for each entity is specified.
  d) Foreign keys (keys identifying the relationship between different entities) are specified.
  e) Normalization occurs at this level.


PRODUCT
PK
Product Number
Product Type
Product Decription
SALES
PK
Sales Order ID
Customer Number
Product Number
Store Number
Date Id
Quantity
Price
CUSTOMER
PK
Customer Number
Customer Name
Customer Type
Customer Address
DATE
PK
Date id
Date
Month
Year
STORES
PK
Store Number
Store Name
Store Address
Country
Region

# Physical Data Model

- This defines how the model is physically existing in the system
  a) Displays all the tables and columns.
  b) Displays foreign keys.
  c) Change the relationships into foreign keys.
  d) Entity now becomes table.
  e) Attribute now becomes column.
  f) Datatypes are also shown in this model.


PRODUCT
PK
Prod_No Int
Prod_Type Varchar(5)
Prod_Desc Varchar(50)
SALES
PK
Sales_ID_Int
Cust_no Int
Prod_Id Int
Store_Id Int
Date_Id Int
Quantity Int
Price Decimal(10,2)
CUSTOMER
PK
Cust_No Int
Cust_Name Varchar(20)
Cust_type Varchar(1)
Cust_Addr Varchar(100)
DATE
PK
Date_id Int
Date Varchar(10)
Month varchar(15)
Year Varchar(5)
STORES
PK
Store_Id Int
Store_Desc Varchar(30)
Store_Addr Varchar(40)
Country Varchar(50)
Region Varchar(50)

| Feature | Conceptual | Logical | Physical |
|---|---|---|---|
| Entity Names | ✓ | ✓ | |
| Entity Relationships | ✓ | ✓ | |
| Attributes | | ✓ | |
| Primary Keys | | ✓ | ✓ |
| Foreign Keys | | ✓ | ✓ |
| Table Names | | | ✓ |
| Column Names | | | ✓ |
| Column Data Types | | | ✓ |

# Difference between Conceptual, Logical, and Physical Data model

- Conceptual is just a high level representation of the Entities that were used as a part of DB design.
- Logical is the next level representation for a Conceptual design in which the entities and attributes comes into picture but with understandable English names. The Relationships are also defined.
- Physical is the more granular level of representing the DB design, in which the entities are now represented as Tables and the attributes are represented as columns along with the primary and foreign key relationships. By looking at the physical design, a person can develop the Database, which in other means can be called as the physical representation of the database.

# Dimensional Data Model

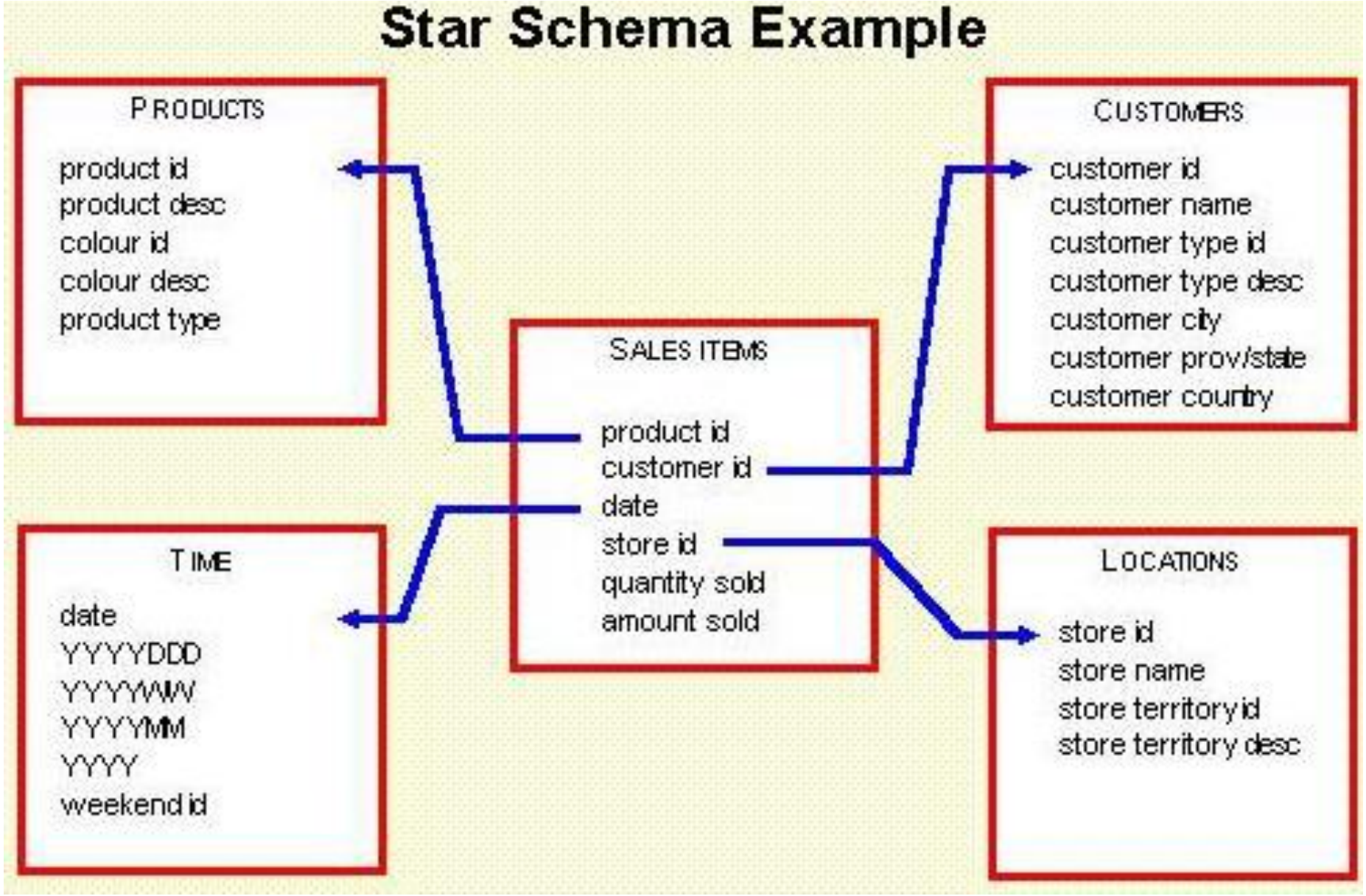
- This defines the Data modeling technique by defining the schema (Star and Snowflake schema).
- **Schema:** A database schema defines its entities and the relationship among them.
- Database schema is a descriptive detail of the database, which can be depicted by means of schema diagrams.
- All these activities are done by database designer to help programmers in order to give some ease of understanding all aspect of database.

# Dimension, Attribute & Hierarchy

- Terms commonly used in Dimensional data model.
- **Dimension**: A category of information.
  - For example, the Date dimension.
- **Attribute**: A unique level within a dimension.
  - For example, Month is an attribute in the Date Dimension.
- **Hierarchy**: The specification of levels that represents relationship between different attributes within a dimension.
  - For example, one possible hierarchy in the Date dimension is Year → Quarter → Month → Day.

# Star Schema

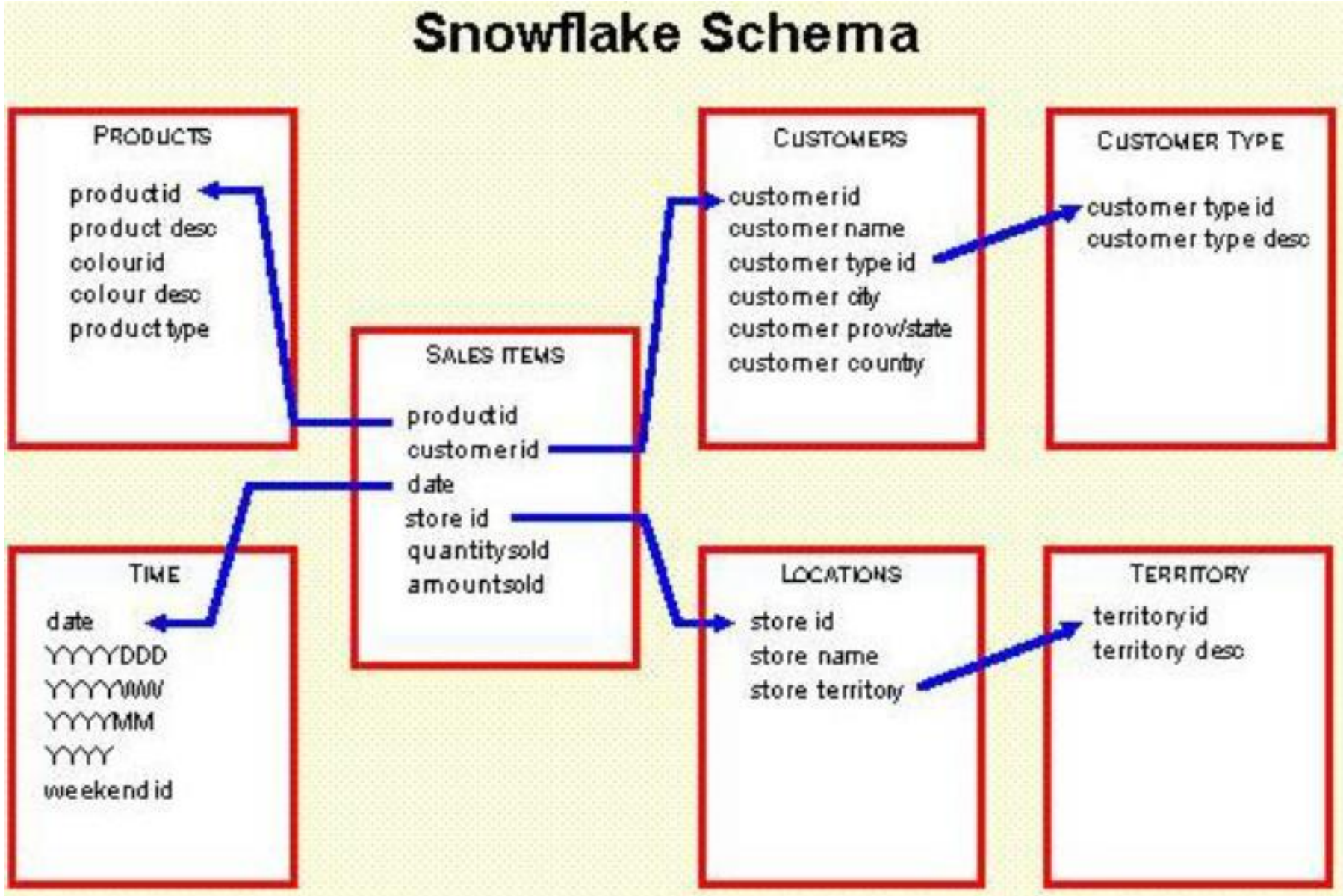
- This consists of the fact table in the middle and the dimensional table around it.
- Fact table is usually a sum of all the dimensions.
- Dimension table is a single entity.
- A primary key in a dimension table is represented as a foreign key in a fact table.
- Note that different dimensions are not related to one another.


Star Schema Example
PRODUCTS
product id
product desc
colour id
colour desc
product type
CUSTOMERS
customer id
customer name
customer type id
customer type desc
customer city
customer prov/state
customer country
SALES ITEMS
product id
customer id
date
store id
quantity sold
amount sold
TIME
date
YYYYDDD
YYYYWW
YYYYMM
YYYY
weekend id
LOCATIONS
store id
store name
store territoryid
store territory desc

# Snowflake Schema

- This is an extension of the Star Schema, where each point of a start is divided into more granular level.
- Each Dimension table is still normalized into multiple lookup tables.
- The main advantage of the snowflake schema is to improve the query performance due to minimized disk storage and joining can also happen on smaller lookup tables.
- Disadvantage is maintenance efforts to manage the number of lookup tables.

# Snowflake Schema



# SCD (Slowly Changing Dimensions)

- Slowly changing dimensions (SCD) determine how the historical changes in the dimension tables are handled. Implementing the SCD mechanism enables users to know to which category an item belonged to in any given date.
- Slowly Changing Dimensions are often categorized into three types namely Type1, Type2 and Type3.
- Ex: Consider a customer with name Rahul who is living in London from 2003.This can be depicted in a table as below:

| Customer Number | Customer Name | Year | Location |
|---|---|---|---|
| 1001 | Rahul | 2003 | London |

# Type 1 SCD: Replaces the old entry with the new value

- The customer Rahul has moved from London to Paris in the year 2005.
- In this Type 1 SCD, the table will be changed as below:

| Customer Number | Customer Name | Year | Location |
|---|---|---|---|
| 1001 | Rahul | 2005 | Paris |

- In this case, the previous entry which is treated as history is lost.

# Type 2 SCD: Creates an New entry in the table

- The New record is inserted into the same table, In this Type 2 SCD, the table will be changed as below:

| Customer Number | Customer Name | Year | Location |
|---|---|---|---|
| 1001 | Rahul | 2003 | London |
| 1001 | Rahul | 2005 | Paris |

- In this case, each record will be treated differently, which makes the table to grow fast and complicates the ETL process. This is mainly for tracking the historical changes.

# Type 3 SCD: Original Entry get modified to reflect the new entry

- New fields are added to the table to main the history
- In this Type 3 SCD, the table will be changed as below:

| Customer Number | Customer Name | Year | Location | Old Year | Old Location |
|---|---|---|---|---|---|
| 1001 | Rahul | 2005 | Paris | 2003 | London |

- The previous record itself is modified such that neither the history is lost and even the new record is also displayed. But this can accommodate only one change. This Type3 SCD is rarely used, when the changes are prone to change only once.

# Normalization

- Normalization is a database design technique which organizes tables in a manner that reduces redundancy and dependency of data.
- It divides larger tables to smaller tables and link them using relationships.

# Business Intelligence

- Business intelligence is the set of techniques and tools for the transformation of raw data into meaningful and useful information for business analysis purposes.
- BI technologies are capable of handling large amounts of unstructured data to help identify, develop and otherwise create new strategic business opportunities.
- The goal of BI is to allow for the easy interpretation of these large volumes of data.

# Difference between Data warehouse and Business Intelligence

- Data warehouse is a way of storing data and creating information through leveraging data marts. Data marts are segments or categories of information and/or data that are grouped together to provide 'information' into that segment or category.
- Data warehouse does not require Business Intelligence to work.
- Reporting tools can generate reports from the DW.
- Business Intelligence is the leveraging of DW to help make business decisions and recommendations.
- Information and data rules engines are leveraged here to help make these decisions along with statistical analysis tools and data mining tools.

# Difference between ETL and BI Tools

- An ETL tool is used to extract data from different data sources, transform the data, and load it into a DW system.
- BI tool is used to generate interactive and ad-hoc reports for end-users, dashboard for senior management, data visualizations for monthly, quarterly, and annual board meetings.
- The most common ETL tools include – IBM Data Stage, Informatica – Power Center, Microsoft – SSIS, Oracle Data Integrator ODI etc.
- Some popular BI tools include – SAP Business Objects, IBM Cognos, Jasper Soft, Microsoft BI Platform, Tableau, Oracle Business Intelligence Enterprise Edition, etc.

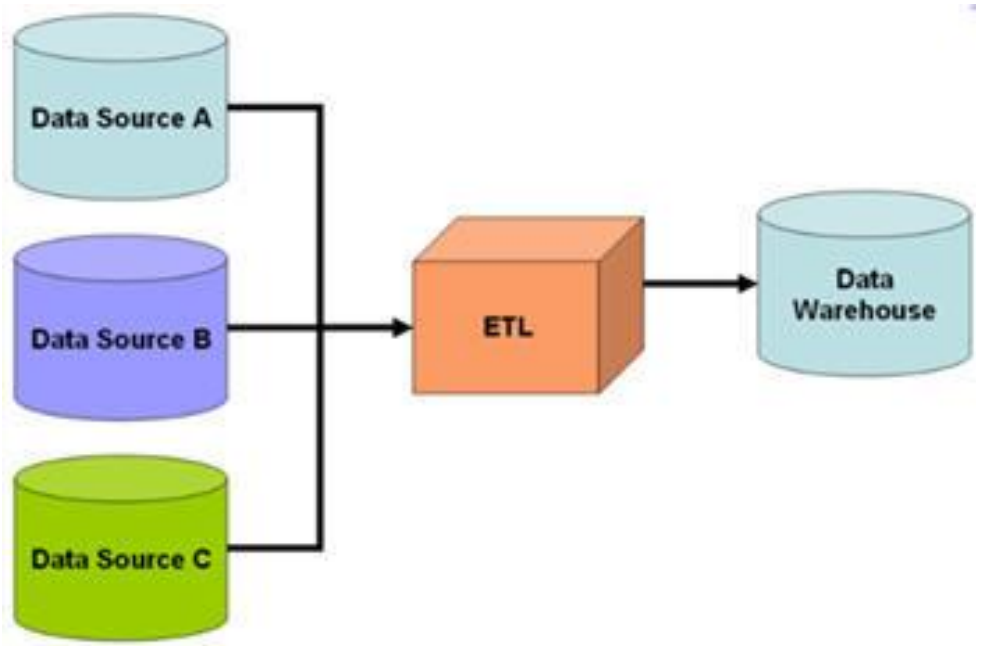
# SQL


Adobe Acrobat Document

# ETL Testing

# What is ETL?

- ETL stands for Extract-Transform-Load and it is a process of how data is loaded from the source system to the data warehouse.




Extracting from source
Transformation
Loading into Target
Bill_id Date Amount
1 10-Oct-15 100
2 05-Nov-15 150
3 17-Nov-15 50
Bill_id Date Amount
1 10-Nov-15 100
2 11-Nov-15 150
3 12-Nov-15 50
Bill_id Date Amount
1 10-10-15 100
2 05-11-15 150
3 17-11-15 50
4 10-11-15 100
5 11-11-15 150
6 12-11-15 50
Bill_id Date Amount
1 10-10-15 100
2 05-11-15 150
3 17-11-15 50
4 10-11-15 100
5 11-11-15 150
6 12-11-15 50

# Types of data load in ETL

- There are two major types of data load available based on the load process.
- **1.Full Load (Bulk Load)**
- The data loading process when we do it at very first time. It can be referred as Bulk load or Fresh load.
- **2. Incremental load (Refresh load)**
- The modified data alone will be updated in target followed by full load. The changes will be captured by comparing created or modified date against last run date of the job.

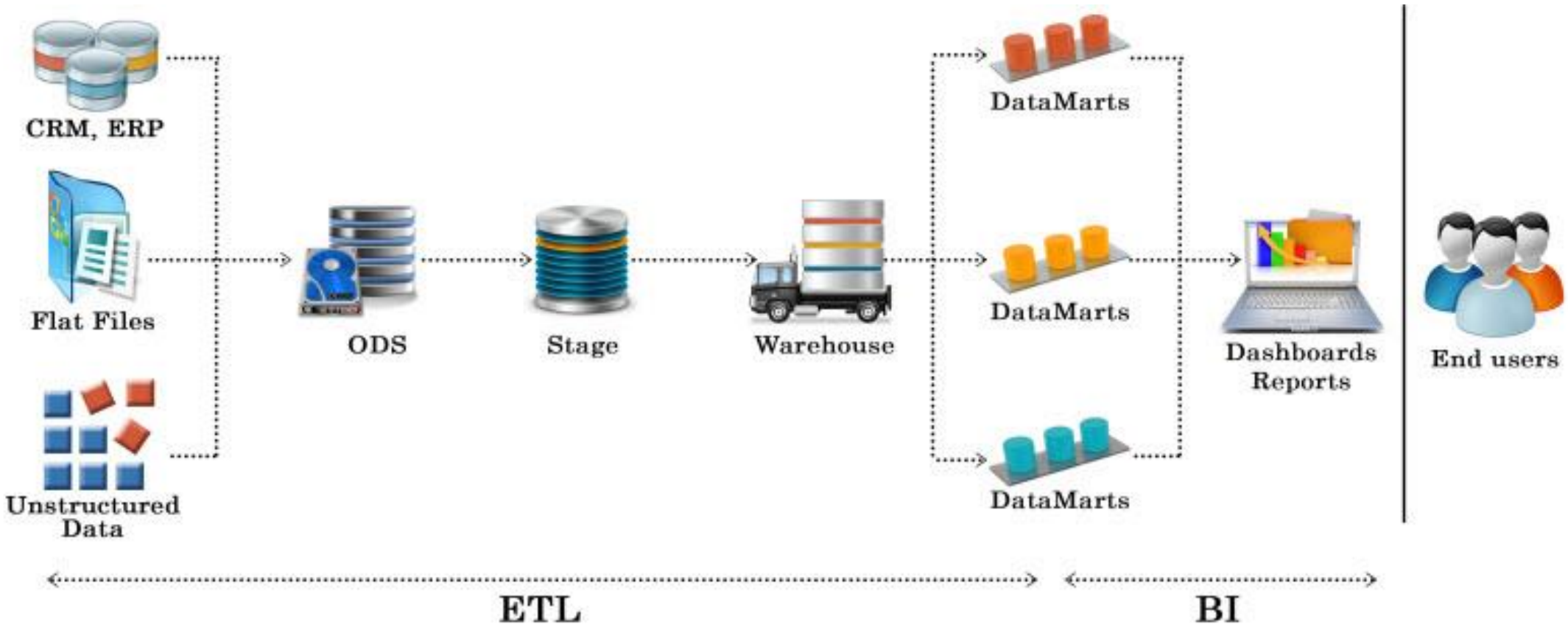
# Data Cleansing (data scrubbing)

- Data cleansing is a process of removing irrelevant and redundant data, and correcting the incorrect and incomplete data. It is also called as **data cleaning** or **data scrubbing**.
- **Irrelevant** – deleting data which are not required for business or not needed anymore
- **Redundant** – deleting the duplicate data
- **Incorrect** – updating incorrect values with correct value
- **Incomplete** – updating incomplete values with full information

# Data masking

- Organizations never want to disclose highly confidential information into all users. The process of masking/hiding/encrypting sensitive data is called data masking.
- **Data masking Techniques:**
  - Below are few usual ways or techniques of doing data masking,
  - Replacing subset of data with dummy characters (ex: aaa)
  - Replacing subset of data with dummy numbers (ex: 1111)
  - Replacing subset of data with dummy random numbers (ex: 1576)
  - Replacing subset of data with dummy special characters (ex: xxx)
  - Shuffling the characters or numbers

# ETL Process


CRM, ERP
Flat Files
Unstructured Data
ODS
Stage
Warehouse
DataMarts
DataMarts
DataMarts
Dashboards Reports
End users
ETL
BI

# ODS ( Operational Data Store)

- It is a database which has integrated data from different business or sources with different rules.
- It gets data from the transactional database directly.
- It will have a limited period of history data, hardly 30 to 90 days of data.

# Staging area

- A staging area, or landing zone, is an intermediate storage area used for data processing during the extract, transform and load (**ETL**) process.

# Warehouse

- Data warehouse (DW or DWH), also known as an enterprise data warehouse (EDW), is a system used for reporting and data analysis, and is considered a core component of business intelligence.
- DWs are central repositories of integrated data from one or more disparate sources.

# Data mart

- The **data mart** is a subset of the data warehouse and is usually oriented to a specific business line or team.

# Data Retention – Archiving or Purging

- **Data retention** means defining how long the data need to be available in the database.
- **Why data retention policy is required?**
- **Performance** will be impacted due to increase of data.
- The **cost** will be high if database accumulates data.
- The old record **may not be useful** for end user over a certain period of time.

- **What is purging?**
- It means deleting data from a database which crosses the defined retention time.
- **What is archiving?**
- Archiving means moving the data which crosses the defined retention time to another database (archival database).
- **What to select Archiving or Purging?**
- If the old records won’t be required at any point of time then purging will be applied, else the data will be archived.

# Example

- A retail shop maintains a data warehouse where all the sales data will be loaded at the month level, as business is growing day by day still more data will be getting loaded.
- The shop has been running for the **past 10 years** now the data warehouse database size has got increased tremendously.
- Also, the shop management says they do not want to view the report at month level for 10-year-old data. Hence they are planning to remove the data older than 10 years.
- At the same time, they want to keep the data at year level instead of month level.
- So the requirement would be, roll up all month data's into year level for data older than 10 years and delete the month level data.

# ETL Testing Life Cycle/Process

- Similar to other Testing Process, ETL also go through different phases. The different phases of ETL testing process is as follows.
  1. Requirement analysis
  2. Test planning
  3. Test design
  4. Test execution
  5. Defect retesting
  6. Test Closure/Sign off

- **Requirement analysis**
- The major inputs for the testing team would be data model and mapping document. When we start our analysis itself we need to make sure that the source table or files are correct.
- **Test planning**
- There is not much difference between functional test plans except few items, here we need to mention the data flow in both scope and out-scope sections.

- **Test design**
- Test cases will be prepared along with mapping document. In this stage itself, we need to find requirement related defects by doing analysis on source data and mapping documents such as **data type, data length, and relationships.**
- **Test execution**
- Once all entry criteria’s are all set, initial execution can be performed with bulk load jobs. All the stages from Source to target will be tested one by one.

- **Defect retesting**
- Fixed defects will be retested and validated in the case of any rejection. Based on Impact analysis, the test cases need to be executed as part of defect fix.
- **Sign off**
- Based on exit criteria of test e execution, the sign off mail to be sent to stakeholders in order to be proceeded push the code to next level.

# ETL Test Scenarios

- **Table structure verification**
- The column name, data type and data length of target table will be verified against the requirement.
- **Constraints check**
- Ensure that all required constraints are available.
- **Index check**
- Ensure that index created with required columns.
- **Source data validation**
- Record the source table count and ensure that there won’t be any junk or bad data exists.
- **Data count check**
- Comparing the target data count against source data count along with major filter or join condition.

- **Data comparison check**
- Ensure that source data was moved correctly to the target table by comparing data.
- **Duplicate data validation**
- Inject duplicate entries in source table based on unique identifiers and ensure that the duplicate record will be rejected.
- **Data with primary key and foreign key check**
- Test the primary key and foreign key relationship with different test data for parent and child table.
- **NULL check**
- Inject a data with NULL for an NOTNULL column and verify that data will be rejected.

- **Data precision check**
- Create test data in source table with different precisions and ensure the loaded data has precision as per requirement.
- **Date format check**
- All date columns are loaded in defined date format or not.
- **All business transformation rules validation**
- Every transformation mentioned in TSD needs to be tested, keep separate test cases for every transformation.

# Active & Passive Transformations

- **Active Transformation**
- The output record count of the transformation **may or may not equal** to input record count.
- For example, when we apply filter transformation for age column with the condition of age between 25 and 40. In this case, the data will come out which satisfies this condition, hence the outcome count cannot be predicted.
- **Passive Transformation**
- The output record count of the transformation is **equal** to input record count.
- For example, when we apply expression transformation to concatenate first name and last name columns, in this case, the data will come out even though the columns do not have values.

# Types of Transformations

- Filter Transformation
- Expression Transformation
- Aggregator Transformation
- Joiner Transformation
- Union Transformation
- Sorter Transformation
- Rank Transformation

- **Filter**
- It’s an active transformation. A column can be selected as a filter with a condition. The data will be returned if it satisfies the filter condition else data will be rejected.
  - Ex: select * From employees where **department_id=10**;
- **Expression**
- It’s a passive transformation. An expression can be mentioned like concatenation or replacement for NULL values. The expression will be applied to a specific column and returned.
  - Ex: select first_name,**salary+1000  totsal** from employees;

- **Aggregator**
- It’s an active transformation. An aggregate function can be applied to a measure such as Max, Avg, Max, count and Min etc.
  - Ex: Select department_id ,**min(salary),max(salary)** From employees **group by** department_id;
- **Joiner**
- It’s an active transformation. It joins 2 or more sources along with join condition. The data will be returned if it satisfies the join condition else data will be rejected.
  - Ex: select First_name,departments.DEPARTMENT_NAME From employees,departments where **employees.department_id=departments.department_id**;
- **Union**
- It’s an active transformation. The two or more sources can be merged with this transformation.
  - Ex: SELECT employee_id , job_id, department_id FROM employees **UNION** SELECT employee_id , job_id, department_id From job_history;

- **Sorter**
- It’s an active transformation. The sorting column can be selected along with the order to be sorted either ascending or descending. Based on the column and order the rows will be sorted.
  - Ex: select * From employees **orderby** salary desc;
- **Rank**
- The rank number will be generated based on given Grouping column and order.
- Ex:
  - select first_name, salary, **rank() over( order by salary)**rank from employees;
  - select first_name, salary, **dense_rank() over( order by salary)**rank from employees;

# Difference between rank(), dense_rank(),row_number()

- select first_name, salary, rank() over( order by salary)rank from employees;
- select first_name, salary, dense_rank() over( order by salary)rank from employees;
- select first_name, salary, row_number() over( order by salary)rank from employees;
- select first_name,salary,rank() over( order by salary)rank, dense_rank() over( order by salary) dense_rank, row_number() over( order by salary) rownumber from employees;

| id | name | dept | salary | Rank | Densrank | Rownum |
|---|---|---|---|---|---|---|
| 1009 | I | TECH | 900000 | 1 | 1 | 1 |
| 1006 | F | TECH | 800000 | 2 | 2 | 2 |
| 1005 | E | QA | 700000 | 3 | 3 | 3 |
| 1008 | H | QA | 700000 | 3 | 3 | 4 |
| 1010 | J | QA | 600000 | 5 | 4 | 5 |
| 1007 | G | TECH | 400000 | 6 | 5 | 6 |
| 1003 | C | HR | 300000 | 7 | 6 | 7 |
| 1002 | B | HR | 200000 | 8 | 7 | 8 |
| 1004 | D | QA | 200000 | 8 | 7 | 9 |
| 1001 | A | HR | 100000 | 10 | 8 | 10 |

# Data Model & Mapping Document

- **ETL mapping Document :**
- Columns mapping between source and target
- Data type and length for all columns of source and target
- Transformation logic for each column
- ETL jobs
- **DB Schema of Source, Target:**
- It should be kept handy to verify any detail in mapping sheets.

# DWH Sources, Targets & ETL Mapping sheet.

Microsoft Excel
Worksheet

# ETL Test Cases

Microsoft Excel
Worksheet

# ETL Test Case Characteristics

- **Simple:**
- The test case description and expected result need to be a very simple language where other can easily understandable.
- **Small:**
- Do not keep lengthy test cases, break into small ones which will make our job easy for doing impact analysis and for regression test suite identification and execution.
- **Must have SQL Query:**
- I have come across  situation that some of the test steps do not have SQL query itself, because of this tester need to form the SQL query during test execution time
- **Valid SQL Query:**
- Always execute the SQL query before uploading into test case repository since there are chances that the query could throw errors. Also, use right built-in function and join type and condition appropriately.
- **Up to date:**
- There are chances happen like on job name, log filename, parameter file names, paths could be changed. The ETL test case needs to be updated based on all modifications.

# ETL Test Execution

ETL Scripts.sql

# ETL BUGS

# ETL Bugs

- Table structure issue
- Index unable to drop issue
- Index is not created after job run
- Data issue in source table
- Data count mismatch between source and target
- Data not matching between source and target
- Duplicate data loaded issue
- Trim and NULL issue

# ETL Bugs…

- Data precision issue
- Date format issue
- Business transformation rules issue
- Performance issue

# ETL testing Best Practices in Defect Logging

- Detail Summary
- Provide Test data
- Relevant SQL query
- Description at Depth
- Don’t miss to upload proof

# key responsibilities  of an ETL tester

- Understanding requirements.
- Raising clarifications.
- Test plan preparations.
- Test case preparation with SQL queries based on Technical design or mapping document.
- Reviewing test cases and SQL queries.
- Executing test cases.
- Raising defects
- Active participant in defect triage call.
- Tracking defects.
- Updating and sharing regular status.
- Giving signoff.

# FAQ’S