Hospital Management

Software Requirements Document

V 1.0

	Prepared By	Reviewed by	Approved By
Name	Avijit Bhuin		
Role	Intern		
Signature	Avijit Bhuin		
Date			

Contents:

1. Introduction

In today's healthcare landscape, effective management and organization of hospital operations play a crucial role in delivering quality patient care. The Hospital Management Software project aims to address this need by developing an advanced software system that streamlines various administrative and operational processes within ABC Hospital. This document serves as a comprehensive guide to understanding the requirements, scope, and key aspects of the software system.

1.1 About this Document

The purpose of this document is to systematically outline the essential requirements for the development of the "Enterprise Data Warehouse for Hospital Management." By capturing the functional and technical specifications, this document provides a clear understanding of the project's objectives and sets the foundation for scoping, design, and implementation. Additionally, it serves as a reference for stakeholders involved in the project's development, ensuring alignment and clarity throughout the development lifecycle.

-Purpose & Scope of the Document

The primary purpose of this document is to detail the functional requirements of the Hospital Management Software. These requirements include the specifications for various modules such as Patient, Physician, Appointment, Insurance, and Billing. By defining these requirements, the document aims to ensure that the developed software system meets the hospital's operational needs, enhances decision-making capabilities, and facilitates comprehensive reporting.

Furthermore, this document serves as a critical input for scoping the project, aiding in the planning and execution of development tasks. It provides project managers, developers, and other team members with a unified understanding of the system's objectives and functionalities.

1.2 Intended Audience

The target audience for this document includes members of the project team responsible for the development of the Hospital Management Software. This encompasses project leaders, developers, analysts, testers, and any other stakeholders actively engaged in the software's design and implementation. Additionally, this document can also be referenced by key hospital personnel and decision-makers to gain insights into the project's goals and functionalities.

By providing a clear understanding of the project's purpose and scope, this document ensures effective communication and alignment among all parties involved in the project's lifecycle.

2. Software System Overview

The "Enterprise Data Warehouse for Hospital Management" represents a comprehensive software system designed to revolutionize the way ABC Hospital manages its operations. By harnessing the power of data integration, this system aims to enhance decision support, streamline administrative processes, and provide valuable insights for effective hospital management.

2.1 About the Software System:

The software system encompasses several key aspects that contribute to its effectiveness in addressing the hospital's needs.

→Scope of the System

-The scope of the system revolves around five interconnected modules, each catering to a crucial aspect of hospital management:

- Patient Module: Manages patient details including personal information, addresses, and insurance coverage.

- Physician Module: Stores physician information such as specialization, experience, and department roles.

- Appointment Module: Facilitates appointment scheduling and room allocation for patients and physicians.

- Insurance Module: Records patient insurance details, including coverage amounts and policies.

- Billing Module: Handles billing information, payment details, and insurance coverage for patients.

→Exclusions:

The system's focus remains exclusively on the aforementioned modules. Additional functionalities beyond these modules are outside the scope of this project. Any enhancements or extensions will be considered in subsequent phases.

→System Perspective:

The Enterprise Data Warehouse for Hospital Management operates as an independent software system. It captures and stores historical data pertaining to five key hospital departments: patient details, physician information, appointment scheduling, billing records, and health insurance data. This comprehensive perspective enables the system to serve as a valuable repository of information for reporting and analysis.

By offering insights into patient demographics, physician performance, appointment trends, billing accuracy, and insurance utilization, the system empowers hospital administrators with the data necessary to make informed decisions.

Furthermore, the data stored within the system can be utilized across various departments and functions to generate customized reports, conduct trend analysis, and enhance operational efficiency.

In the subsequent sections, we delve deeper into the architectural aspects of the system, the role of the ETL tool (Talend), and how these components align with the project's requirements.

2.2 System Architecture

The proposed system follows a well-defined architecture to ensure scalability, maintainability, and performance. The architecture consists of two main layers:

→Physical Architecture:

The physical architecture depicts the arrangement of system elements and interfaces. It includes the presentation layer, business logic layer, and data access layer. Talend, as the ETL tool, will be primarily operating in the data access layer, facilitating data movement and transformation.

→Logical Architecture:

The logical architecture defines the processes required to provide user services. It represents the functional components of the system and how they interact to achieve the project's objectives. Talend's workflows and data integration jobs will be integral parts of the logical architecture, orchestrating the ETL processes seamlessly.

2.3 Table Definitions & Mappings

The Data Warehouse will consist of several tables, each serving a specific purpose in the reporting and analysis process. These tables include:

- **Patient Module**: Contains patient related information.
- **Physician Module**: Contains physician related information.
- **Appointment Module**: Contains information about the appointments listed.
- **Insurance Module**: Connects vendors and parts with historical tracking.
- **Billing Module**: Contains customer-related data.

Talend will be responsible for extracting data from source systems, transforming it into the appropriate format, and loading it into the relevant tables in the Data Warehouse based on these mappings.

3. ETL Tool – Talend

3.1 Overview of Talend

Talend is a powerful and widely-used ETL (Extract, Transform, Load) tool that enables organizations to efficiently integrate, process, and manage data from diverse sources. With its user-friendly graphical interface and robust functionality, Talend empowers developers and data engineers to create sophisticated data integration workflows, making it an ideal choice for building data warehouses and data management solutions.

Key features of Talend include:

-Data Connectivity:

Talend supports a wide range of data sources, including databases, flat files, cloud-based applications, and web services. Its extensive library of connectors facilitates seamless data extraction from various systems.

-Data Transformation:

Talend provides a rich set of data transformation components, enabling users to cleanse, enrich, and format data as required. These transformations play a crucial role in ensuring data quality and consistency.

-Data Loading and Synchronization:

Talend simplifies the process of loading data into target systems, including data warehouses, databases, and cloud platforms. It also supports incremental data loading, ensuring that only the changed or new data is processed during subsequent runs.

-Job Orchestration:

Talend allows users to design complex ETL workflows by visually connecting different components and defining the flow of data through the system. This graphical approach enhances the readability and maintainability of ETL jobs.

-Scalability and Performance:

Talend is designed to handle large volumes of data and can be deployed in distributed environments to achieve scalability and optimize performance.

3.2 How Talend Meets Project Requirements

Talend effectively addresses several key project requirements:

Data Extraction and Transformation:

With the requirement to load data into the data warehouse, Talend's transformation capabilities come into play. The tool allows for data transformation and cleansing, ensuring that data is correctly formatted and relevant for reporting purposes. The system's modules, such as Patient, Physician, and Billing, can be effectively integrated into the data warehouse using Talend's connectors and transformation functions.

Data Loading and Integration:

With the requirement to load data into the data warehouse, Talend's transformation capabilities come into play. The tool allows for data transformation and cleansing, ensuring that data is correctly formatted and relevant for reporting purposes. The system's modules, such as Patient, Physician, and Billing, can be effectively integrated into the data warehouse using Talend's connectors and transformation functions.

Data Quality and Validation:

Talend offers data quality features that ensure the accuracy and reliability of data. This is crucial for maintaining the integrity of patient information, insurance records, and billing data. By applying data validation rules and cleansing mechanisms, Talend helps mitigate data discrepancies and inconsistencies.

Scalability and Future Expansion:

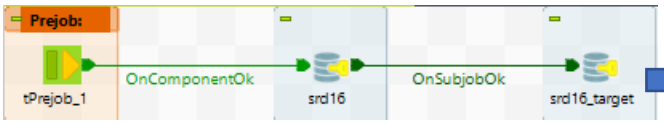
Talend's support for various data sources, including relational databases, flat files, and APIs, ensures that the system can accommodate future expansions or integrations. This scalability is crucial for accommodating potential enhancements to the system's modules or the addition of new functionalities.

In the next section, we provide visual representations of the ETL workflows using Talend, showcasing how data flows through the system to meet the project's objectives.

3.3 ETL Workflow Diagrams:

NOTE: For the input and output and the mapping file please refer to the annexure section below.

Before starting the job, we need to connect to our databases, so we're using two tdbconnection to do so:



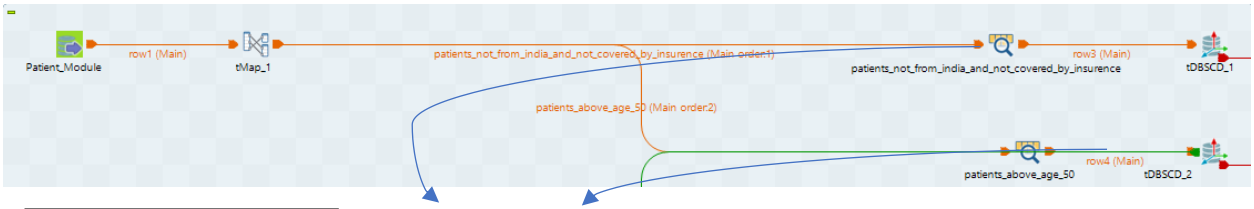
Before Moving to any table, we are establishing a secure connection to our source database and our Target database. This part is to ensure that before we start any operation the databases should be ready. Once they're connected, we can move to a table.

Let's understand each table one by one first:

Patient Module:

Functional Requirements	1) Produce report on patient details who are not from India, not covered by any insurance 2) Produce report on patient details above age 50 *3)* Dimension Patient is used to get the Patient Details of the ABC Hospital including insurance details.

IMPLEMENTATION:



INPUT FORMAT:

Patient_ID	INT
Name	VARCHAR (100)
Date_of_Birth	DATE
Gender	VARCHAR (10)
Address	VARCHAR (100)
City	VARCHAR (50)
Country	VARCHAR (50)
Phone	VARCHAR (15)
Insurance_ID	VARCHAR (20)

This display is for the sake of development only and does not serve anything towards the operation on the data.

OPERATING ON THE DATA AS PER RREQUIREMENT

patients_above_age_50:

Patient_ID	INT
Name	VARCHAR (100)
Date_of_Birth	DATE
Gender	VARCHAR (10)
Address	VARCHAR(100)
City	VARCHAR (50)
Country	VARCHAR (50)
Phone	VARCHAR (15)
Insurance_ID	VARCHAR (20)
Age	INT

patients_not_from_india_and_not_covered_by_insurance:

Patient_ID	INT
Name	VARCHAR (100)
Date_of_Birth	DATE
Gender	VARCHAR (10)
Address	VARCHAR(100)
City	VARCHAR (50)
Country	VARCHAR (50)

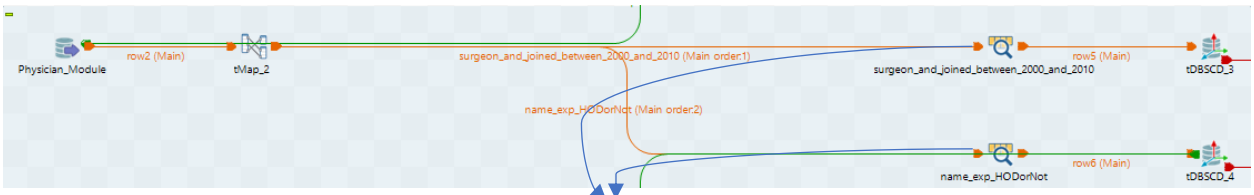
The tDBSCD components are connected to a tDBRollback for security measures.

It Proceeds to the next subjob on a "OnComponentOK" Trigger, connected to the tDBSCD_2 component.

Physician Module:

Functional Requirements	1) Report of physician details who are surgeon who joined hospital between 2000 to 2010 (year) 2) To display physician details based on i) Name ii) Experience 3) To verify whether the physicians are head of the department 4) Dimension credit is used to get the Physician information.

IMPLEMENTATION:



INPUT FORMAT:

Physician_ID	INT
Name	VARCHAR (100)
Date_of_Joining	DATE
Specialty	VARCHAR (50)
Designation	VARCHAR (50)
Experience	INT

This display is for the sake of development only and does not serve anything towards the operation on the data.

OPERATING ON THE DATA AS PER RREQUIREMENT

surgeon_and_joined_between_2000_and_2010:

Physician_ID	INT
Name	VARCHAR (100)
Date_of_Joining	DATE
Specialty	VARCHAR (50)
Designation	VARCHAR (50)
Experience	INT

The tDBSCD components are connected to a tDBRollback for security measures.

surgeon_and_joined_between_2000_and_2010:

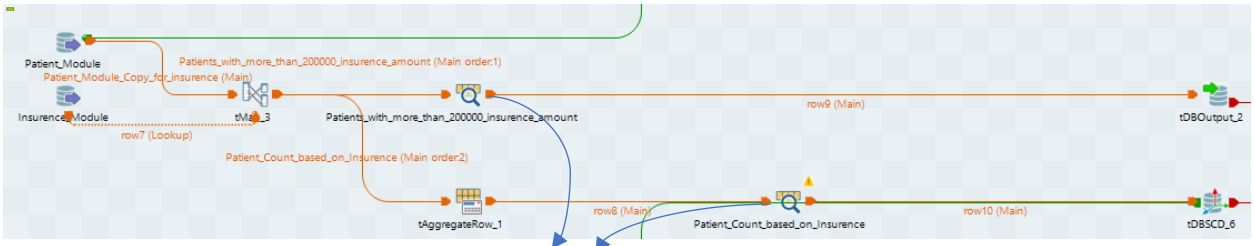
Physician_ID	INT
Name	VARCHAR (100)
Date_of_Joining	DATE
Specialty	VARCHAR (50)
Designation	VARCHAR (50)
Experience	INT
Dept_Head	VARCHAR (5)

It Proceeds to the next subjob on a "OnComponentOK" Trigger, connected to the tDBSCD_4 component.

Insurance Module:

Functional Requirements	1) To store patient info who have insurance amount greater than 2,00,000. 2) To store and fetch count of patients covered by each insurance. 3) Dimension Insurance is used to get the Patient Insurance details.

IMPLEMENTATION:



PATIENT MODULE: (MAIN)

Patient_ID	INT
Name	VARCHAR (100)
Date_of_Birth	DATE
Gender	VARCHAR (10)
Address	VARCHAR (100)
City	VARCHAR (50)
Country	VARCHAR (50)
Phone	VARCHAR (15)
Insurance_ID	VARCHAR (20)

INSURANCE MODULE: (LOOKUP)

Insurance_ID	VARCHAR (20)
Insurance_Name	VARCHAR (100)
Maximum_Coverage_Amount	DECIMAL (10,2)

This display is for the sake of development only and does not serve anything towards the operation on the data.

OPERATING ON THE DATA AS PER RREQUIREMENT

patients_with_more_than_200000_insurance_amount:

Patient_ID	INT
Insurance_Name	VARCHAR (100)
Name	VARCHAR (100)
Date_of_Birth	DATE
Gender	VARCHAR (10)
Address	VARCHAR (100)
City	VARCHAR (50)
Country	VARCHAR (50)
Phone	VARCHAR (15)
Maximum_Coverage _Amount	DECIMAL(10,2)

patient_count_based_on_insurence:

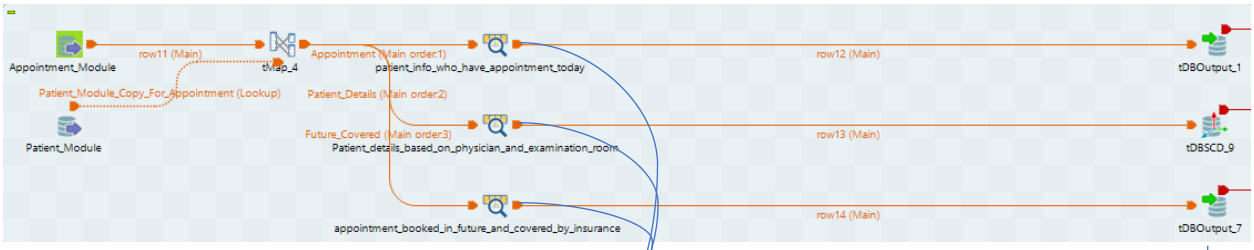
Insurance_ID	VARCHAR (20)
Insurance_Name	VARCHAR (100)
Patient_Count	INT

The tDBSCD components are connected to a tDBRollback for security measures.

It Proceeds to the next subjob on a "OnComponentOK" Trigger, connected to the tDBSCD_6 component.

Appointment Module:

Functional Requirements	1) To store patient info who have appointment today 2) To fetch Patient details based on physician and examination room. 3) To display patient who have appointment booked in future and covered by insurance. 4) Dimension Appointment is used to get the Appointment details.



appointment_module: (MAIN)

Appointment_ID	INT
Patient_ID	INT
Physician_ID	INT
Start_Date	DATE
Start_Time	TIME
Examination_Room	VARCHAR (50)

PATIENT MODULE: (LOOK UP)

Patient_ID	INT
Name	VARCHAR (100)
Date_of_Birth	DATE
Gender	VARCHAR (10)
Address	VARCHAR (100)
City	VARCHAR (50)
Country	VARCHAR (50)
Phone	VARCHAR (15)
Insurance_ID	VARCHAR (20)

This display is for the sake of development only and does not serve anything towards the operation on the data.

OPERATING ON THE DATA AS PER RREQUIREMENT

appointment_booked_in_future_and_covered_by_insurance:

Patient_ID	INT
Name	VARCHAR (100)
Gender	VARCHAR (10)
Phone	VARCHAR (15)
Examination_Room	VARCHAR (50)
Physician_ID	INT
Insurance_ID	VARCHAR (20)
Appointment_Start_Date	DATE
Appointment_Start_Time	TIME

patient_details_based_on_physician_and_examination_room:

Appointment_ID	INT
Patient_ID	INT
Physician_ID	INT
Examination_Room	VARCHAR (50)
Name	VARCHAR (100)
Gender	VARCHAR (10)
Address	VARCHAR (100)
Phone	VARCHAR (15)
Insurance_ID	VARCHAR (20)

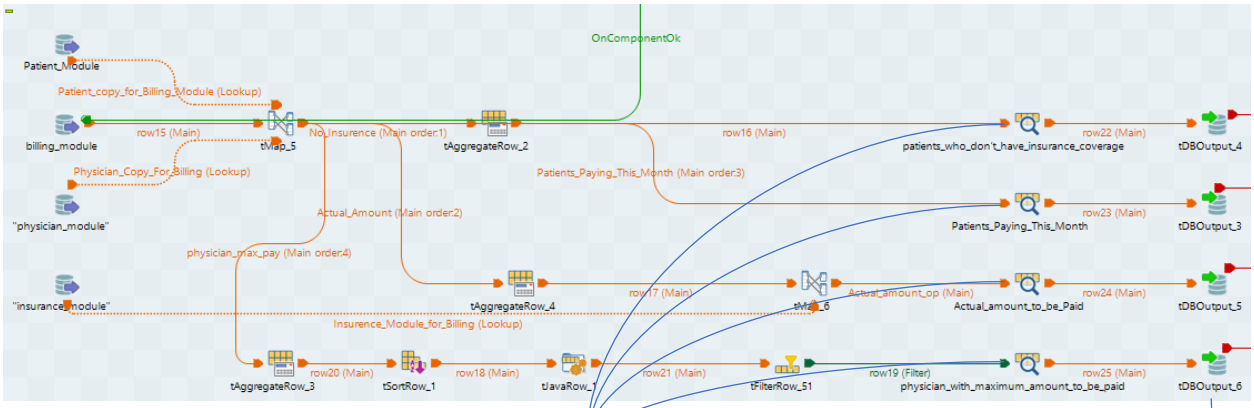
patient_info_who_have_appointment_today:

Patient_ID	INT
Name	VARCHAR (100)
Gender	VARCHAR (10)
Phone	VARCHAR (15)
Examination_Room	VARCHAR (50)
Physician_ID	INT

The tDBSCD components are connected to a tDBRollback for security measures.

It Proceeds to the next subjob on a "OnComponentOK" Trigger, connected to the tDBSCD_6 component.

Billing Module:



BILLING MODULE: (MAIN)

Payment_ID	INT
Patient_ID	INT
Physician_ID	INT
Amount	DECIMAL(10,2)
Payment_Date	DATE
Insurance_ID	VARCHAR (20)

INSURANCE MODULE: (LOOKUP)

Insurance_ID	VARCHAR (20)
Insurance_Name	VARCHAR (100)
Maximum_Coverage_Amount	DECIMAL (10,2)

PHYSICIAN MODULE: (LOOKUP)

Physician_ID	INT
Name	VARCHAR (100)
Date_of_Joining	DATE
Specialty	VARCHAR (50)
Designation	VARCHAR (50)
Experience	INT

PATIENT MODULE: (LOOK UP)

Patient_ID	INT
Name	VARCHAR (100)
Date_of_Birth	DATE
Gender	VARCHAR (10)
Address	VARCHAR (100)
City	VARCHAR (50)
Country	VARCHAR (50)
Phone	VARCHAR (15)
Insurance_ID	VARCHAR (20)

This display is for the sake of development only and does not serve anything towards the operation on the data.

OPERATING ON THE DATA AS PER RREQUIREMENT

patients_who_don't_have_insurance_coverage:

Payment_ID	INT
Patient_ID	INT
Name	VARCHAR (100)
Gender	VARCHAR (10)
Address	VARCHAR (100)
Phone	VARCHAR (15)
Amount	DECIMAL(10,2)

patients_paying_this_month:

Payment_ID	INT
Payment_Date	DATE
Patient_ID	INT
Name	VARCHAR (100)
Gender	VARCHAR (10)
Address	VARCHAR (100)
Country	VARCHAR (50)
Phone	VARCHAR (15)
Insurance_ID	VARCHAR (20)
Amount	DECIMAL (10,2)

actual_amount_to_be_paid:

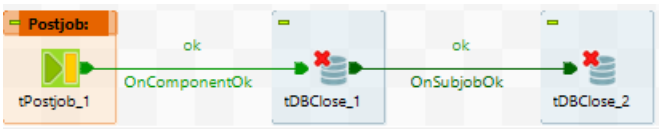
Patient_ID	INT
Name	VARCHAR (100)
Address	VARCHAR (100)
Phone	VARCHAR (15)
Actual_Amount_to_be_Paid	DECIMAL(10,2)

physician_with_maximum_amount_to_be_paid:

Physician_ID	INT
Name	VARCHAR (100)
Specialty	VARCHAR (50)
Designation	VARCHAR (50)
Amount	DECIMAL(10,2)

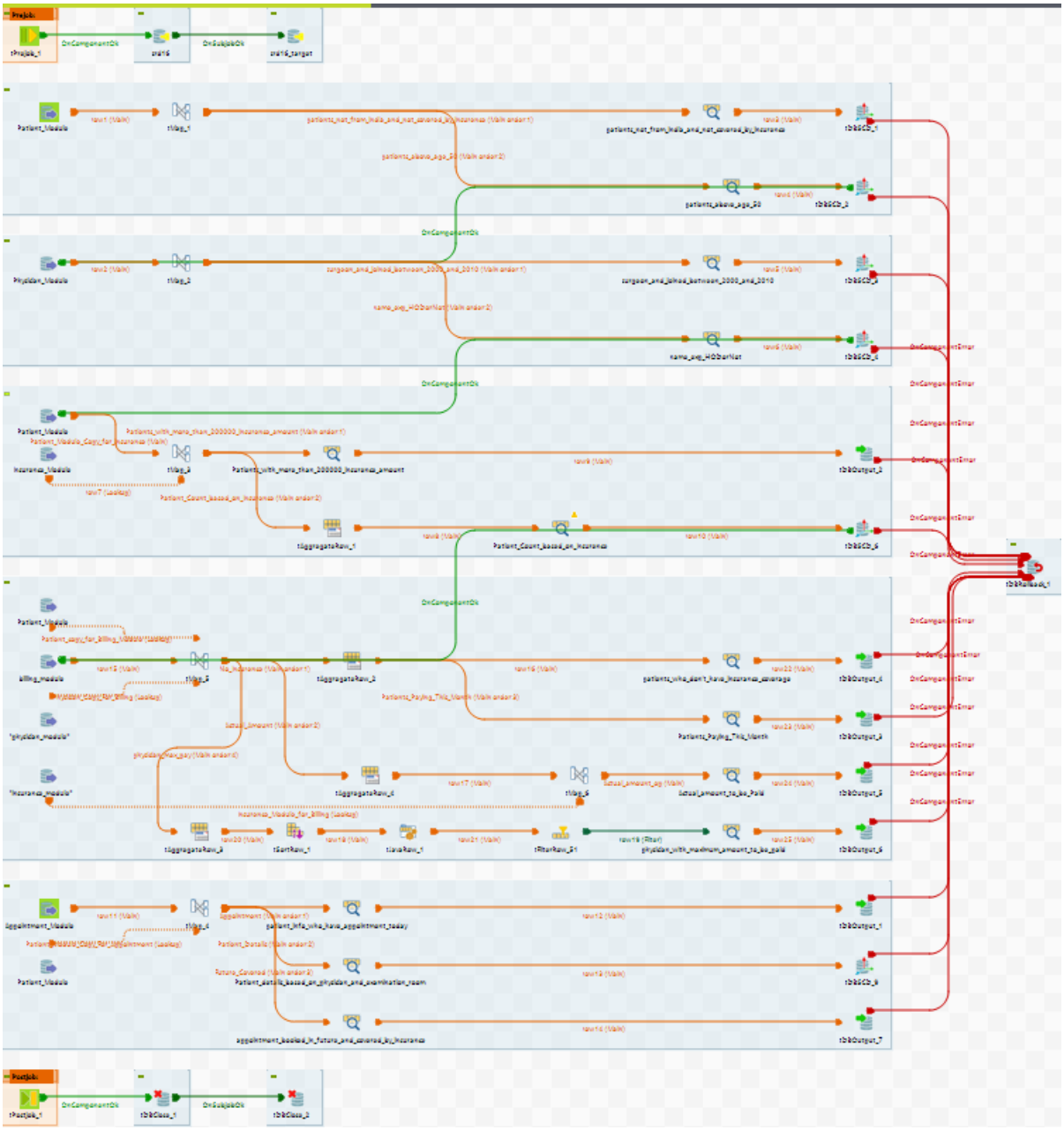
The tDBSCD components are connected to a tDBRollback for security measures.

Before exiting the job, we need to close our connection to our database, so we're using two "tdbclose" to do so:



Before exiting the job, we are closing the connection to the database.

After this the job will be closed.



As you can see every table is divided into different sub jobs and every sub job is interconnected with a “On Component OK” component and it’s also connected to a “tdbrollback” component that rolls back the database to the previous save point in case there is some kind of error in any of the sub jobs.

After the job is done we need to build it and then use it along with tack scheduler in order to automate this process every day or every week or every month or according to customer demands.

4. Annexure:

4.1 Terms & Conditions:

-Licensing Information:

The ETL project using Talend is subject to specific licensing terms and conditions. These terms govern the usage, distribution, and modification of the project deliverables. The licensing information is outlined as follows:

1. The ETL project's source code and documentation are the intellectual property of Cognizant Technology Solutions Ltd. and are protected under applicable copyright laws.

2. The project's deliverables, including Talend job designs, ETL workflows, and documentation, are solely for internal use by Cognizant Technology Solutions Ltd. and may not be distributed or shared with external parties without prior written consent.

3. Any modifications or enhancements to the ETL project must comply with Cognizant Technology Solutions Ltd. 's change management process and be approved by the designated project authorities.

4. Cognizant Technology Solutions Ltd. shall not hold Talend responsible for any issues or challenges arising from the usage of the Talend software, as per the terms and conditions set forth by Talend's licensing agreement.

-Data Privacy and Security Policies:

As part of the ETL project's implementation, data privacy and security policies are paramount to safeguard sensitive information. The following policies are to be adhered to:

1. Data Encryption: All sensitive data transmitted between systems and during ETL processes must be encrypted to prevent unauthorized access.

2. Access Controls: Access to the Data Warehouse and related systems shall be granted based on the principle of least privilege, ensuring that only authorized personnel can access sensitive data.

3. Data Anonymization: Personally identifiable information (PII) and other sensitive data must be anonymized or pseudonymized when not required for specific reporting or analysis.

4. Data Retention: Data retention policies shall be defined to manage the storage and archival of data in compliance with legal and regulatory requirements.

5. Audit Logging: Comprehensive audit logs shall be maintained to track data access, changes, and user activities for accountability and troubleshooting purposes.

6. Disaster Recovery: Robust data backup and disaster recovery mechanisms shall be established to ensure data availability and continuity in case of unforeseen events.

4.2 Appendix:

-Source File:

SOURCE_FILES.xlsx

All the data used as source to test this project has been provided in this file, separated in different sheets for each table. (Please use your Cognizant account to view)

-Target File:

TARGET_FILES.xlsx

All the generated output according to the customer demand is here. (Please use your Cognizant account to view)

-Project:

Hospital_Management.zip

Here is a copy of the project for further reference.

-Source file Script:

srd16_script.sql

With this you can create the tables for the source files in MysqlWorkbench 8.0.

-Target file Script:

srd16_target_script.sql

With this one can create the target files generated through the job in the database directly. For further reference.

THANK YOU