PROJECT NOTES

> Initializing git hub and cloning the repo, or make a local git repo.

> Setup Draw.io (For daigrams).

Bronze Layer

Analyzing
Source System

> **Coding**
Data Ingestion

> **Validating**
Data Completness & Schema Checks

> **Docs & Versioning**
Data Documenting versioning in git.

> Business Context and Ownership

→ who owns the data ?

→ what business process does it supports.

→ System & data documentation

→ Data Model & Data catalog.

> Architecture & Technology stack.

→ How is data stored

→ what are the integration Capabilities.

> Extract & Load

→ Incremental Vs Full Load

→ Data Scope & Historical Needs

→ what is the expected size of the extracts

→ Are there any data volume limitations.

→ How to avoid impacting the source system's performance.

→ Authentication and authorizations.

1. Create TABLES (DDL scripts), for all the files, CRM, erp.

If object-id ('name', 'U') is not null
DROP name;

```
CREATE TABLE name (
    C1 DT,
    C2 DT,
    ...
    );
```

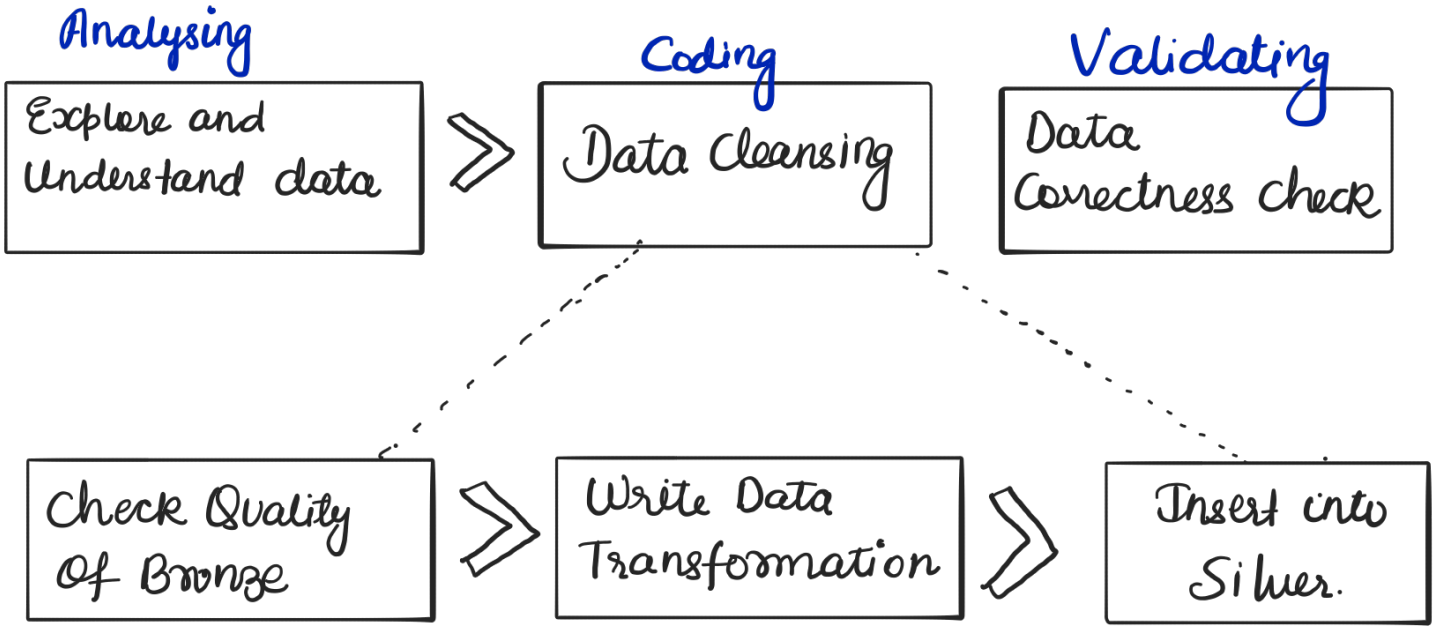
2. TRUNCATE TABLE table_name
3. BULK INSERT table_name
   FROM 'Path\filename'
   WITH (
   FIRSTROW = 2, (ROW=1, Contains info about fields not data)
   FIELDTERMINATOR = ',' → Delimeter (Separator of values).
   TABLOCK
   );

› For data quality check all the fields, and count the no of rows.
› Repeat for all the files.

For Load scripts :

1. Make stored Procedure.
2. Use messages
3. Use Try & Catch.
4. Use Load Time. (fore each table using variables).
5. Use Load time for each source or entire process.
6. Draw
7. Push Code. to git.

Silver Layer



1. Data exploration, try to create links between the table on draw.io

2. Create tables (DDL), just copy DDL from the bronze Layer and change the schema to Silver (ctrl+F).

   add dwh_create_date datetime2 default getdate();

> Metadata Columns:

→ Extra columns added by data engineers that do not originate from the source data.

create_date: The Records load timestamp.

update_date: The records last update timestamp.

source_system: The origin system of the record.

file_location: The file source of the record.

3. Check for duplicates and NULL values in the column you want to make a primary key. (In bronze Layer).

> Expectation: No Results.

> pick the value with the highest date (or make a standard criteria).

4. Check for unwanted spaces in string values.

Expectation: No Results.

(TRIM)

5. Data standarization & Consistency :

(CASE), eg : M → Male
F → Female
M - Married
S - Single

(UPPER).

6. Insert new data into the silver schema.

7. bronze. erp C filter out unmatched data after applying transformation, check for symbols. (-), (_), replace if inconsistent.

8. Extract data two columns from 1 column.

SUBSTRING (C, S.V, Leng(C)) .

9. Prd line (start_date, end date, ambigous), decide a Rule, then apply Rule,

Start of 2nd is end of 1st. { Lead -1 }

OR, (dateadd (-1) day).

10. Cast datetime as date.

Derived Columns :

> Create new columns based on calculations or transformation of existing ones.

> Data enrichment - adding new data to the table.

Table crm_sales_details :

1. Date quality check <=0 OR leng != 8 OR
   > Current-date OR < Start_date.

2. Can't cast int to date, so into to varchar.
   then date int > Varchar > date.

3. Sales order date cannot be greater then Ship date & vice versa, same for due dates.

4. Sales = quantity x Price

   Quantity != 0, -ve, NULLS

5. For price and quantity related issues,

> Fix the source system

> Fix the issue in data warehouse.

> Pick a Rule.

> Handle division by 0

NULLIF(Col, 0).

Table : Bronze.erp_cust_az12 :

> Remove NAS from the cid using CASE.

> Check D.O.B Range. (olddate < > getdate)

> getdate with NULL.

> Cleanup gender colum (first distinct) then

Convert it to Male, Female, N/A .

Cas when IN ('F', 'FEMALE') - Female.

Table : erp_loc_a101

→ clean cid, replace '-' with ''.

→ Normalize the country.

Table : erp_px_cat_g1v2 .

→ check cat for empty space, subcat, maintainance.

→ check the distinct cat, subcategory.

→ Just load.

Create Procedure : (silver.Load-silver).

① Print
② Truncate
③ Print inserting
④ print insertion finished.

→ Print message for each section & step.

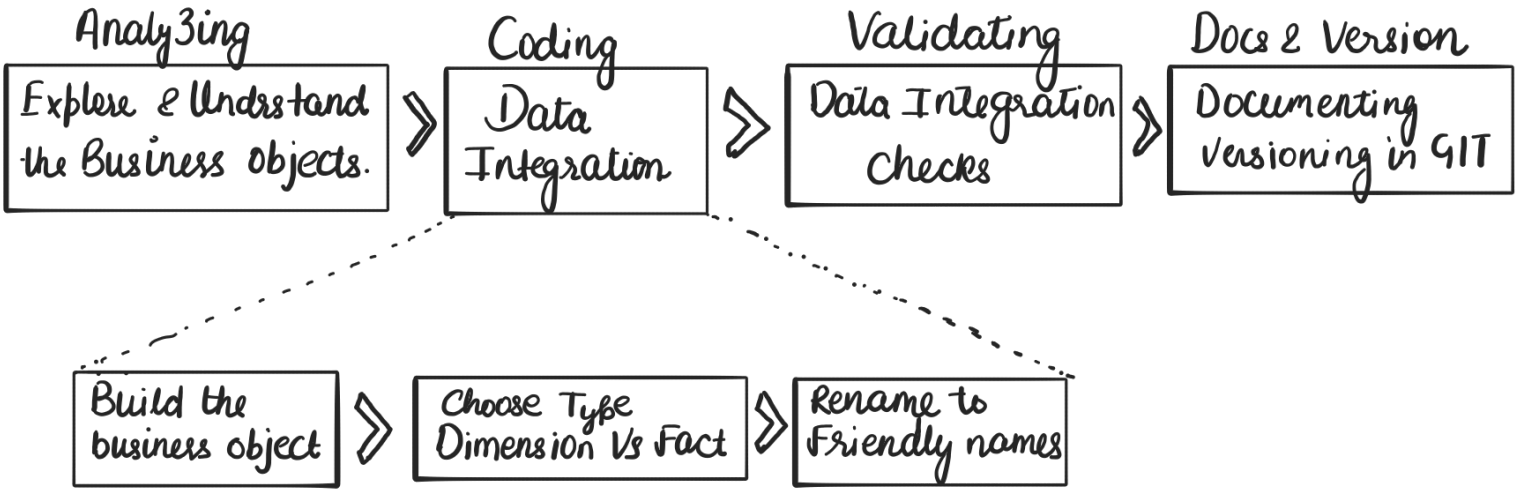
→ Implement error handling.

→ Print duration of each step.

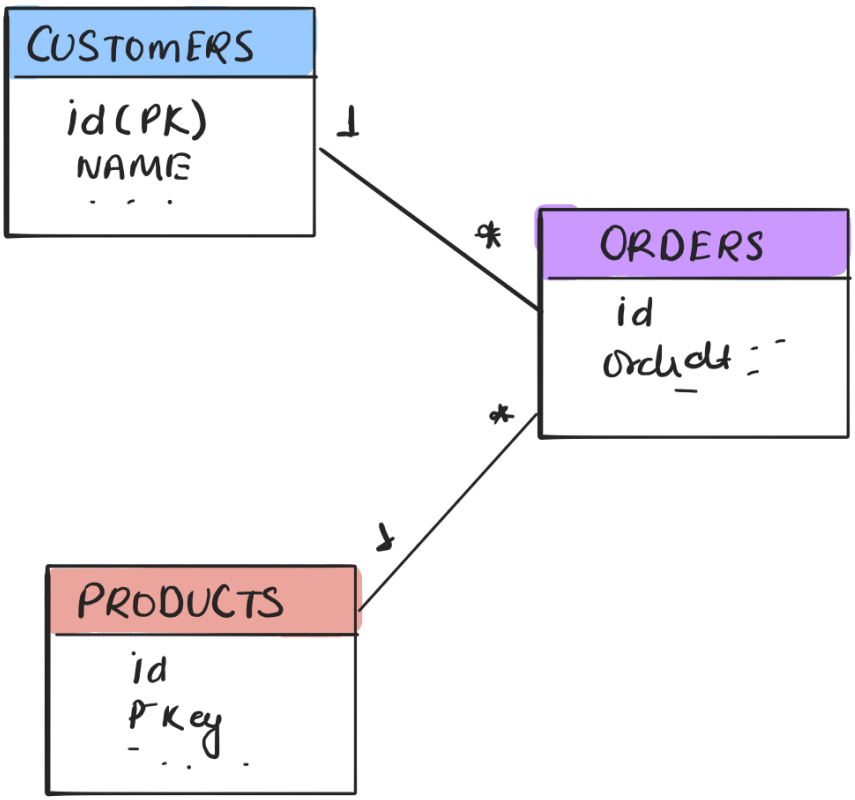
→ Print total duration.

Note: Maintain consistency across stored Procedures.

GOLD LAYER



Data Modelling:



Conceptual Model (BIG PICTURE)

> Focus is only on entity (like what tables) and Rln between them.

> No columns or stuff

Logical Data Model (BLUE PRINT)

> Specify the details (columns) in these entities

> Also establish the relationship between the entities (how they are connected).

Physical Data Model (IMPLENTATION)

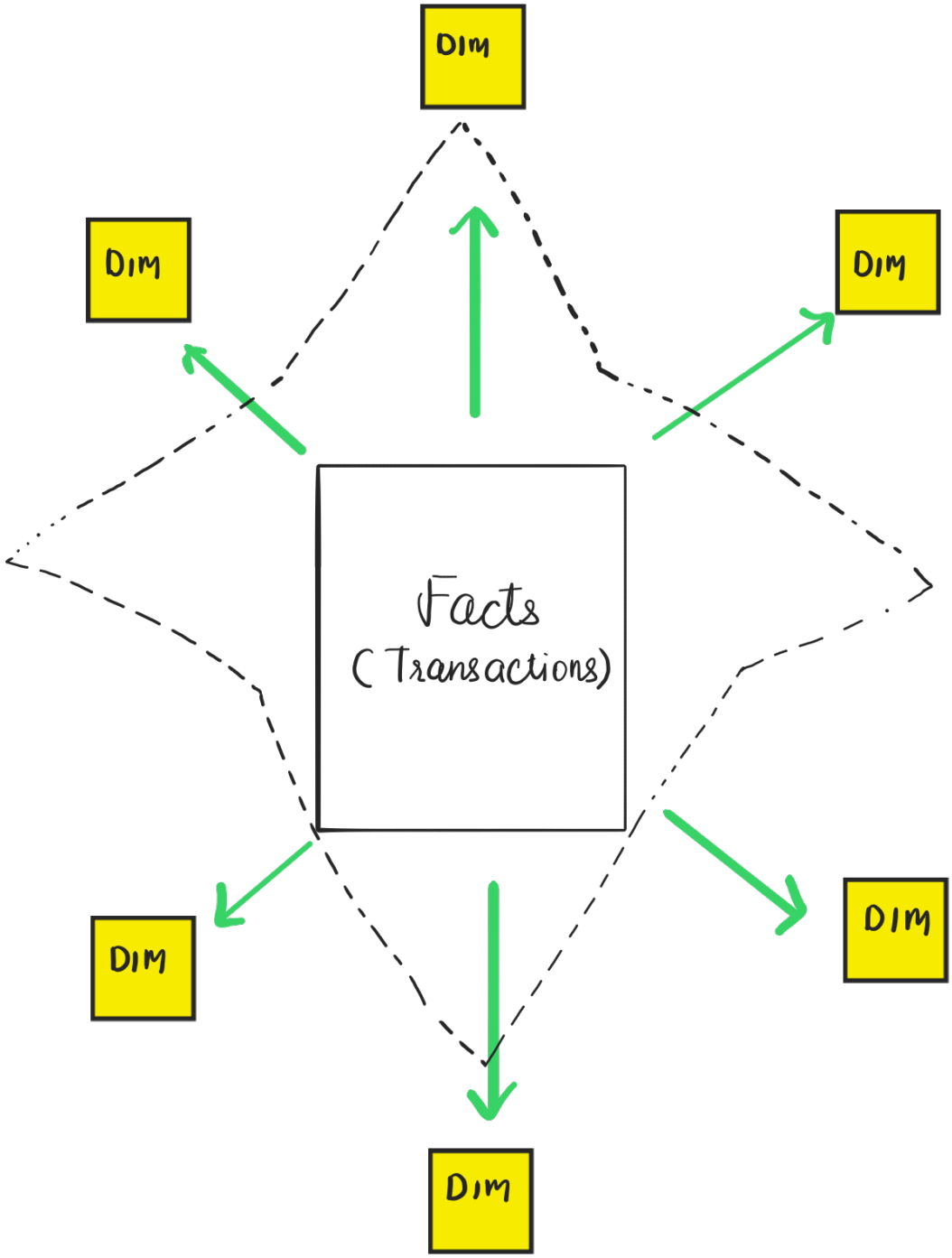
> How the data will be stored in the database

> Column Types.

NOTE: For this project we will build the logical Data Models.

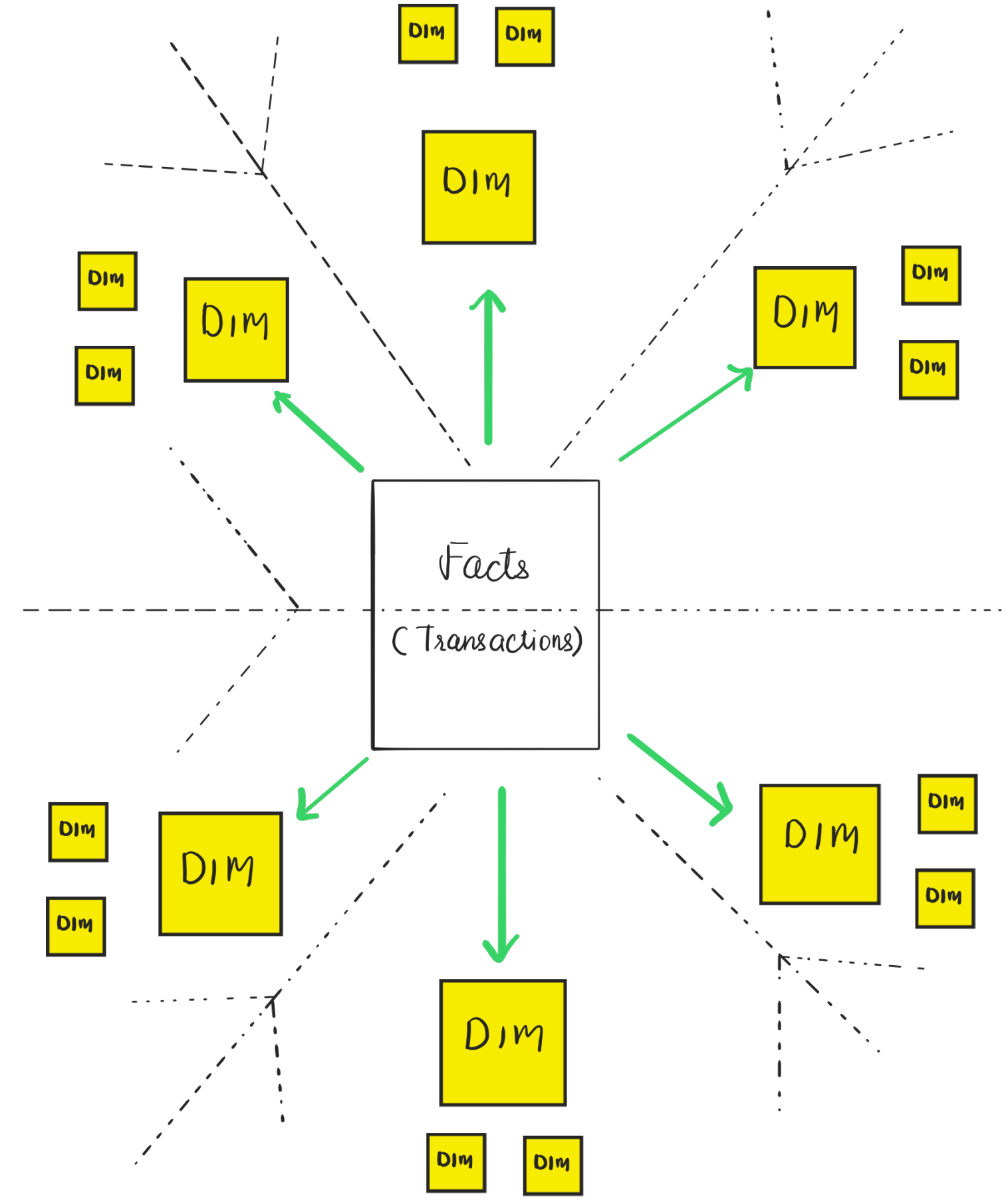
Schemas

Star Schema :



Dimensions, are description about Transactions

Snow Flake Schema



→ Dimensions are split into smaller dimension.

Dimensions

> Descriptive information that gives context to your data. (who, what were).

Facts

> Quantative information that represents events (How much, how many).

> Explore Business objects:

> label Tables, (like sales, products, customers).

> Create dimension Customer

> Join all tables that have cust inform. and then create customer view that has all the neccesary details about customer available

> If same data comes from mutiple Columns. then we have to do data integration.
(ASK system expert). only for matching data.
(USE CASE handle NULLS) { Take any one source as master }.

> Rename columns (meaningful name)

> Assign a primary key.

Surrogate Keys: System generated unique identifier assigned to each record in table.

> DDL Based generations.

> Query- based using window function (Row- Number).

→ In this project we have used query-based window function.

Create Dimension Product :-

> Join all tables with the product info.

> Filter out historical data, only current products.
(Connection is done using prd-key not the prd-id)

> Check for duplicacy.

> Arrange the columns.

> Rename Columns.

> Create surrogate key

> Create view.

Create Fact Sales :

> Facts connects multiple dimensions.
> Use the dimension's surrogate key instead of IDS to easily connect facts with dimensions.
> Join crm_sales_details with our views.
> Rename column, arange them.
> Create view.

Steps after Creating Facts & Dimensions :

> Check Foriegn key integrity (Right table p.k is NULL).
> Draw data models.(gold).
> Create Data Catalog.
> Extend Data Flow diagram

Gotchas :

→ If DDL was created earlier, and we have derived columns then update DDL for that table.

→ Check for data types in DDL after performing operations in the silver Layer.

→ After every transformation Run it through the quality check scripts.