Snowflake Data Vault 2.0

Nuwan Keshara Galappaththi
Data Engineer
LinkedIn

Contents

Project Overview

1. Project Description

- This project focused on the redesign and implementation of a robust and scalable data warehouse solution utilizing the Data Vault 2.0 methodology within the Snowflake cloud data platform. By using the publicly available 'snowflake_sample_data' database, specifically the 'tpch_sf10' schema, this project demonstrated the practical application of Data Vault 2.0 principles to real world datasets. The project covered the Staging area, Raw Data Vault, Business Data Vault and Information Delivery stages, incorporating data sources such as the Customer, Orders, Region and Nation tables, which contain 1.5 million, 15 million, 5 and 25 records respectively. Furthermore the Information Delivery stage provided a dimensional data model using a star schema to facilitate analytical querying for business intelligence and reporting.

2. Project Purpose

- Develop hands on experience with Snowflake cloud data platform and building data vault 2.0 architecture.
- Improve knowledge of Data Vault concepts and its architecture.
- Keen interest in learning new technologies and tools

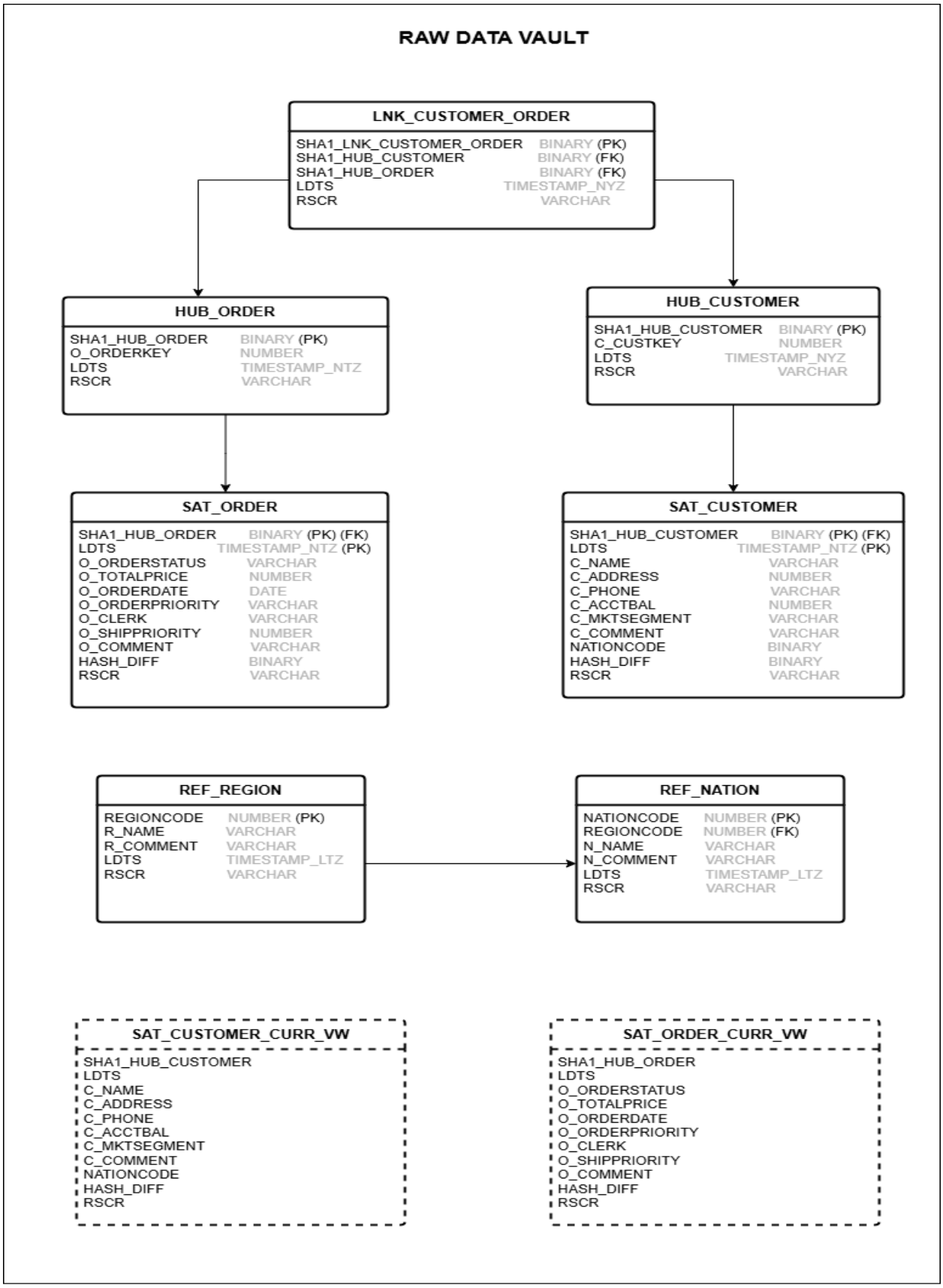
3. Reason for the project selection

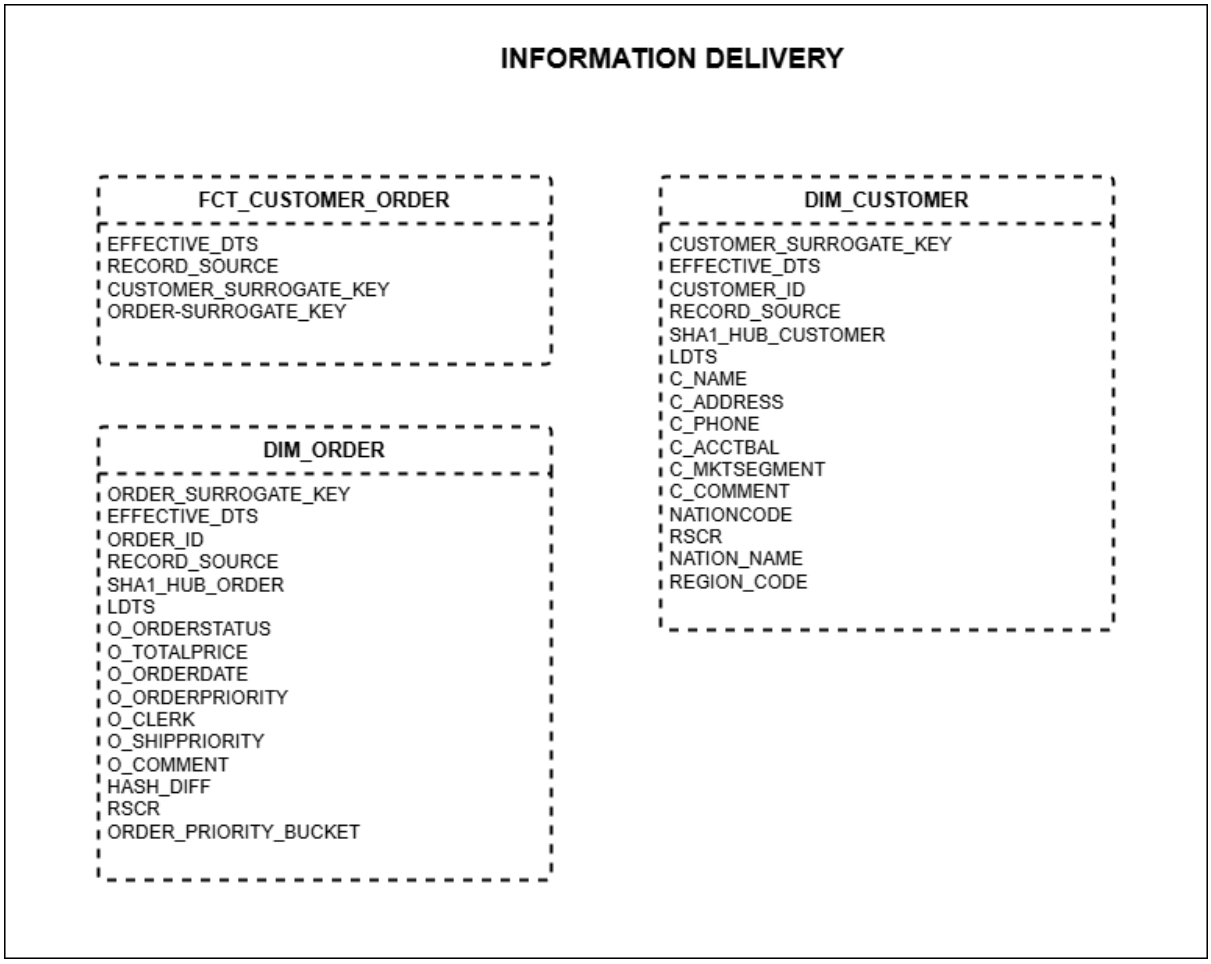
- My background research on Brightly particularly related to open Data Engineer and Data Architect positions revealed that the organization as a data project development company, focusing on Snowflake and Data Vault practices. To align with this and to gain practical experience with these technologies, I chose this specific tech stack for this project.

4. Project Objectives

- **Data Acquisition**: Extract data from source systems and make it accessible. Set up the environment to load the data.
- **Staging Area**: Load the data into the staging tables and create Snowpipe and Stream to ingest data near real-time into the Raw Data Vault and capture change data in the tables, respectively.
- **Raw Data Vault**: Implement the Data Vault 2.0 model on the dataset.
- **Business Data Vault**: Data is aggregated and transformed into tables and views for better analytics
- **Information Delivery**: Develop a dimensional model on the data to facilitate analytics and business reports

5. Project Architecture





Technology Stack:

- Snowflake cloud data platform
- Data Vault 2.0
- Dimensional Model (Star Schema)

snowflake®

Project Flow:

- Data Sources
  - snowflake_sample_data.tpch_sf10
    - Customer – 1.5 M records
    - Orders – 15 M records
    - Region – 5 records
    - Nation – 25 records

- Data Pipeline
  - Streams
  - Snowpipe
  - Tasks

- Data Vault
  - HUB_CUSTOMER
  - HUB_ORDER
  - SAT_CUSTOMER
  - SAT_ORDER
  - LNK_CUSTOMER_ORDER
  - REF_REGION
  - REF_NATION

- Information Delivery
  - DIM_CUSTOMER
  - DIM_ORDER
  - FCT_CUSTOMER_ORDER

Technical Documentation

1. Setting up the Environment

- For this project, I used the 'accountadmin' role for the purpose of the Data Vault implementation.

Ex: *USE ROLE accountadmin;*

- Create a database and use it.

Ex: *CREATE OR REPLACE DATABASE dv_lab;*
USE DATABASE dv_lab;

- Create two virtual data warehouses for generic warehouse work and to run Data Vault object pipelines

Ex:

- *CREATE OR REPLACE WAREHOUSE dv_lab_wh WITH WAREHOUSE_SIZE = 'XSMALL' MIN_CLUSTER_COUNT = 1 MAX_CLUSTER_COUNT = 1 AUTO_SUSPEND = 60 COMMENT = 'Generic WH';*

- *CREATE OR REPLACE WAREHOUSE dv_rdv_wh WITH WAREHOUSE_SIZE = 'XSMALL' MIN_CLUSTER_COUNT = 1 MAX_CLUSTER_COUNT = 1 AUTO_SUSPEND = 60 COMMENT = 'WH for Raw Data Vault object pipelines';*

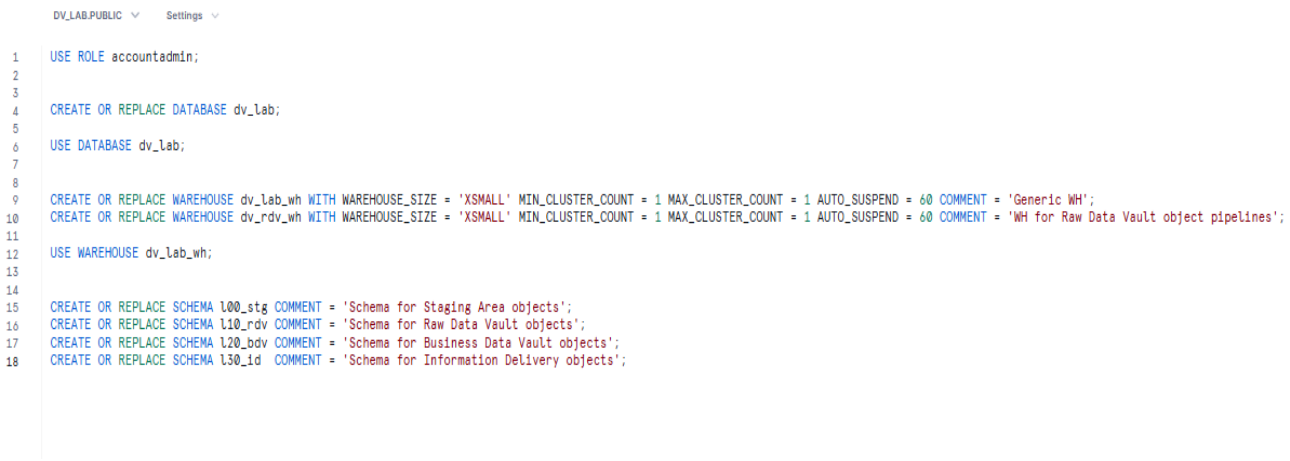
- Create schemas for the staging area, Raw Data Vault, Business Data Vault and Information Delivery.

Ex:

- *USE WAREHOUSE dv_lab_wh;*

- *CREATE OR REPLACE SCHEMA l00_stg COMMENT = 'Schema for Staging Area objects';*

- *CREATE OR REPLACE SCHEMA l10_rdv COMMENT = 'Schema for Raw Data Vault objects';*
- *CREATE OR REPLACE SCHEMA l20_bdv COMMENT = 'Schema for Business Data Vault objects';*
- *CREATE OR REPLACE SCHEMA l30_id COMMENT = 'Schema for Information Delivery objects';*



❖ Create stg_customer and stg_orders tables in the staging area.

Ex:

- *USE SCHEMA l00_stg;*

- *CREATE OR REPLACE TABLE stg_customer*
  (
  raw_json VARIANT
  , filename STRING NOT NULL
  , file_row_seq NUMBER NOT NULL
  , ldts STRING NOT NULL
  , rscr STRING NOT NULL
  );

- *CREATE OR REPLACE TABLE stg_orders*
  (
  o_orderkey NUMBER
  , o_custkey NUMBER
  , o_orderstatus STRING
  , o_totalprice NUMBER
  , o_orderdate DATE

```
, o_orderpriority       STRING
, o_clerk               STRING
, o_shippriority        NUMBER
, o_comment             STRING
, filename              STRING   NOT NULL
, file_row_seq          NUMBER   NOT NULL
, ldts                  STRING   NOT NULL
, rscr                  STRING   NOT NULL
);
```

- ❖ To demonstrate the purpose of Snowpipe, I created a stage to store customer data (JSON) and order data (CSV) files from a subset of the customer and orders source table data.

Ex:

- *CREATE OR REPLACE STAGE customer_data FILE_FORMAT = (TYPE = JSON);*
- *CREATE OR REPLACE STAGE orders_data   FILE_FORMAT = (TYPE = CSV) ;*

- ❖ Before loading customer and orders data into the stg_customer and stg_orders tables in the staging area, create a data Stream on those tables to capture change data.

Ex:

- *CREATE OR REPLACE STREAM stg_customer_strm ON TABLE stg_customer;*
- *CREATE OR REPLACE STREAM stg_orders_strm ON TABLE stg_orders;*

- ❖ Finally, before loading data into the stage, create a Snowpipe from the stage to the stg_customer and stg_orders tables in the staging area to facilitate near real-time stream data.

Ex:

- ```
  CREATE OR REPLACE PIPE stg_orders_pp
  AS
  COPY INTO stg_orders
  FROM
  (
  SELECT $1,$2,$3,$4,$5,$6,$7,$8,$9
  ```
  ```

, metadata$filename
, metadata$file_row_number
, CURRENT_TIMESTAMP()
, 'Orders System'
FROM @orders_data);

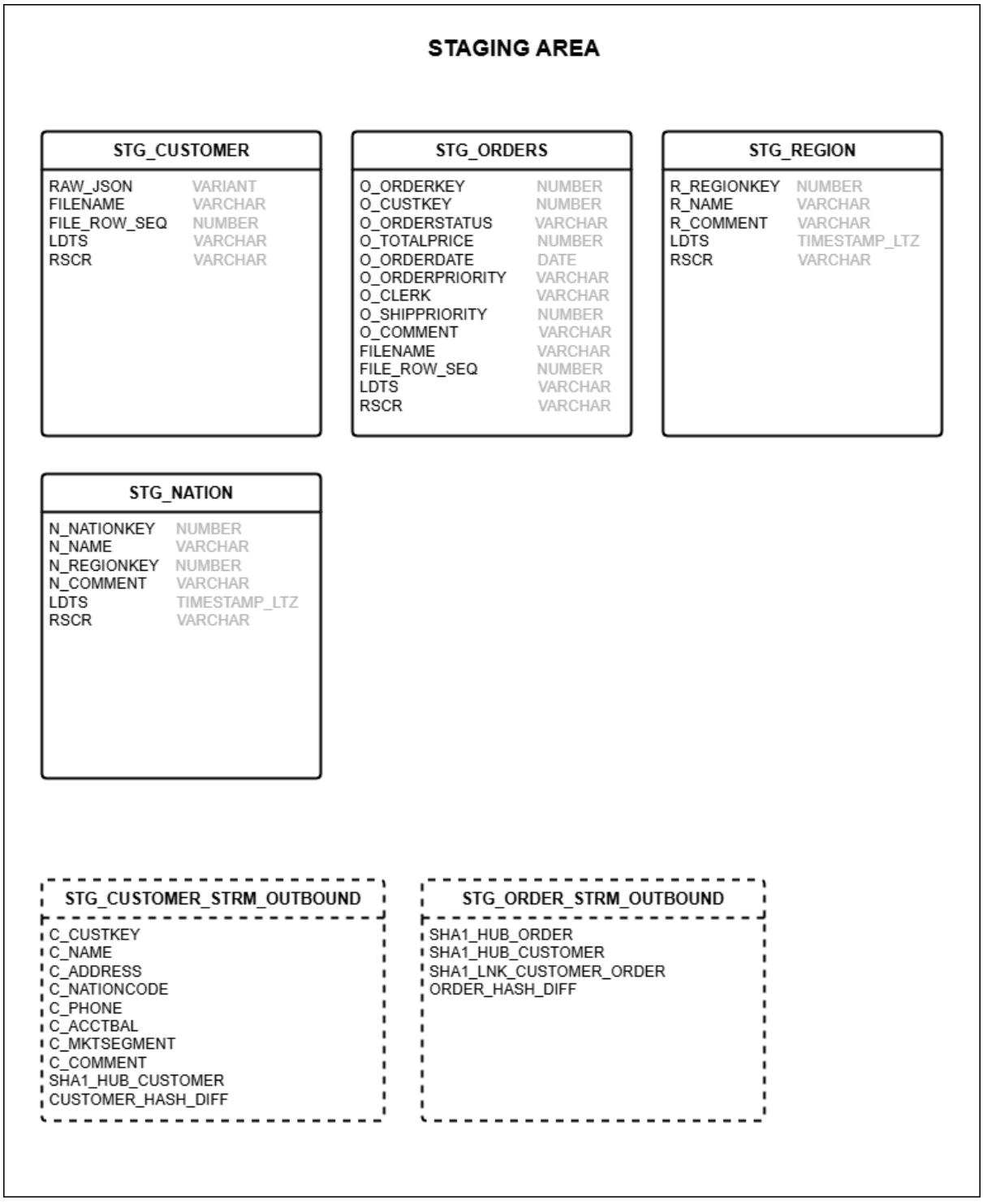
- *CREATE OR REPLACE PIPE stg_customer_pp*
  AS
  COPY INTO stg_customer
  FROM
  (
  SELECT $1
  , metadata$filename
  , metadata$file_row_number
  , CURRENT_TIMESTAMP()
  , 'Customers System'
  FROM @customer_data
  );

❖ Start Snowpipe

Ex:

- *ALTER PIPE stg_customer_pp REFRESH;*
- *ALTER PIPE stg_orders_pp REFRESH;*

2. Staging Area



- Now load the data into the staging tables

Ex: Region and Nation data are loaded into the staging area's stg_region and stg_nation tables from the source data.

- *CREATE OR REPLACE TABLE stg_nation*

AS
*SELECT src.**
, CURRENT_TIMESTAMP() ldts
, 'Static Reference Data' rscr
FROM snowflake_sample_data.tpch_sf10.nation src;

- *CREATE OR REPLACE TABLE stg_region*
  AS
  *SELECT src.**
  , CURRENT_TIMESTAMP() ldts
  , 'Static Reference Data' rscr
  FROM snowflake_sample_data.tpch_sf10.region src;

Ex: stg_nation table

	N_NATIONKEY	N_NAME	N_REGIONKEY	N_COMMENT	LDTS	RSCR
1	0	ALGERIA	0	haggle. carefully final deposits detect slyly agai	2025-03-29T09:45:46.638-07:00	Static Reference Data
2	1	ARGENTINA	1	al foxes promise slyly according to the regular accounts. bold requests alon	2025-03-29T09:45:46.638-07:00	Static Reference Data
3	2	BRAZIL	1	y alongside of the pending deposits. carefully special packages are about the iro...	2025-03-29T09:45:46.638-07:00	Static Reference Data
4	3	CANADA	1	eas hang ironic, silent packages. slyly regular packages are furiously over the tit...	2025-03-29T09:45:46.638-07:00	Static Reference Data
5	4	EGYPT	4	y above the carefully unusual theodolites. final dugouts are quickly across the fu...	2025-03-29T09:45:46.638-07:00	Static Reference Data
6	5	ETHIOPIA	0	ven packages wake quickly. regu	2025-03-29T09:45:46.638-07:00	Static Reference Data
7	6	FRANCE	3	refully final requests. regular, ironi	2025-03-29T09:45:46.638-07:00	Static Reference Data
8	7	GERMANY	3	l platelets. regular accounts x-ray: unusual, regular acco	2025-03-29T09:45:46.638-07:00	Static Reference Data
9	8	INDIA	2	ss excuses cajole slyly across the packages. deposits print aroun	2025-03-29T09:45:46.638-07:00	Static Reference Data

Ex: stg_region table

	R_REGIONKEY	R_NAME	R_COMMENT	LDTS	RSCR
1	0	AFRICA	lar deposits. blithely final packages cajole. regular waters are final requests. regu...	2025-03-29T09:46:34.667-07:00	Static Reference Data
2	1	AMERICA	hs use ironic, even requests. s	2025-03-29T09:46:34.667-07:00	Static Reference Data
3	2	ASIA	ges. thinly even pinto beans ca	2025-03-29T09:46:34.667-07:00	Static Reference Data
4	3	EUROPE	ly final courts cajole furiously final excuse	2025-03-29T09:46:34.667-07:00	Static Reference Data
5	4	MIDDLE EAST	uickly special accounts cajole carefully blithely close requests. carefully final asy...	2025-03-29T09:46:34.667-07:00	Static Reference Data

❖ Send data from customer table (10 records) and orders table(1000 records)

Ex:

- *COPY INTO @customer_data*
  FROM
  (SELECT object_construct(*)
  FROM snowflake_sample_data.tpch_sf10.customer limit 10
)
  INCLUDE_QUERY_ID=TRUE;

- *COPY INTO @orders_data*

```
FROM
(SELECT *
 FROM snowflake_sample_data.tpch_sf10.orders limit 1000
)
INCLUDE_QUERY_ID=TRUE;
```

CUSTOMER_DATA (9 Files)

NAME	SIZE
data_01bb5ba3-0305-f748-0003-b70a000174d6_0_0_0.json.gz	12.3MB
data_01bb5ba3-0305-f748-0003-b70a000174d6_0_2_0.json.gz	11.3MB
data_01bb5ba3-0305-f748-0003-b70a000174d6_0_4_0.json.gz	11.4MB
data_01bb5ba3-0305-f748-0003-b70a000174d6_0_1_0.json.gz	11.3MB
data_01bb5ba3-0305-f748-0003-b70a000174d6_0_7_0.json.gz	11.3MB
data_01bb5ba3-0305-f748-0003-b70a000174d6_0_6_0.json.gz	11.3MB
data_01bb5ba3-0305-f748-0003-b70a000174d6_0_5_0.json.gz	13.2MB
data_01bb5ba3-0305-f748-0003-b70a000174d6_0_3_0.json.gz	12.3MB
data_01bb5602-0305-f040-0000-0003b70a95ed_0_0_0.json.gz	1.0KB

ORDERS_DATA (2 Files)

NAME	SIZE
data_01bb5afb-0305-f749-0003-b70a000155be_0_0_0.csv.gz	33.5KB
data_01bb5602-0305-f040-0000-0003b70a95f1_0_0_0.csv.gz	33.7KB

- Now using Snowpipe, the data will automatically be sent into the stg_customer and stg_orders tables in the staging area

Ex: stg_customer table

	RAW_JSON	FILENAME	FILE_ROW_SEQ	LDTS	RSCR
1	{ "C_ACCTBAL": 528.95, "C_ADDRESS": "ACyEqDI42jQr714XXo71zKD", "C_C...	data_01bb5602-0305-f040-0000-0003b70a95ed_0_0_0.json.gz	1	2025-03-29 11:23:21.501 -0700	Customers System
2	{ "C_ACCTBAL": 12.41, "C_ADDRESS": "sw1IDlEHB7ZZ8m", "C_COMMENT": "...	data_01bb5602-0305-f040-0000-0003b70a95ed_0_0_0.json.gz	2	2025-03-29 11:23:21.501 -0700	Customers System
3	{ "C_ACCTBAL": 8475.34, "C_ADDRESS": "j33kNdjkIJ9kXIZS4dVi1GQ,NEQLC...	data_01bb5602-0305-f040-0000-0003b70a95ed_0_0_0.json.gz	3	2025-03-29 11:23:21.501 -0700	Customers System
4	{ "C_ACCTBAL": 4066.24, "C_ADDRESS": "kNHtEYOfsR 4kXOdMUVzIjCo12H0...	data_01bb5602-0305-f040-0000-0003b70a95ed_0_0_0.json.gz	4	2025-03-29 11:23:21.501 -0700	Customers System
5	{ "C_ACCTBAL": 7328.26, "C_ADDRESS": "BtWP2FDmGp,", "C_COMMENT": "ts...	data_01bb5602-0305-f040-0000-0003b70a95ed_0_0_0.json.gz	5	2025-03-29 11:23:21.501 -0700	Customers System
6	{ "C_ACCTBAL": 8114.86, "C_ADDRESS": "31cmc dzlmXgWUj", "C_COMMENT...	data_01bb5602-0305-f040-0000-0003b70a95ed_0_0_0.json.gz	6	2025-03-29 11:23:21.501 -0700	Customers System
7	{ "C_ACCTBAL": 6588.76, "C_ADDRESS": ",ITzUT2jxY3WngOpOgVgl6 IPupS5S...	data_01bb5602-0305-f040-0000-0003b70a95ed_0_0_0.json.gz	7	2025-03-29 11:23:21.501 -0700	Customers System
8	{ "C_ACCTBAL": 9368.57, "C_ADDRESS": " ItMrkW MqQPiohnCGS2j8EuHyd13...	data_01bb5602-0305-f040-0000-0003b70a95ed_0_0_0.json.gz	8	2025-03-29 11:23:21.501 -0700	Customers System
9	{ "C_ACCTBAL": 1551.87, "C_ADDRESS": "LH UGJ4xksy,qUf2nwvLqJ,FxDKN4...	data_01bb5602-0305-f040-0000-0003b70a95ed_0_0_0.json.gz	9	2025-03-29 11:23:21.501 -0700	Customers System
10	{ "C_ACCTBAL": -766.86, "C_ADDRESS": "V0CwajerAUri7gjFkQEHC", "C_COM...	data_01bb5602-0305-f040-0000-0003b70a95ed_0_0_0.json.gz	10	2025-03-29 11:23:21.501 -0700	Customers System
11	{ "C_ACCTBAL": -967.41, "C_ADDRESS": "JBoJ7i3s,AkOuvRpNNh5u7bL3ooiSv...	data_01bb5ba3-0305-f748-0003-b70a000174d6_0_3_0.json.gz	165897	2025-03-30 10:08:10.179 -0700	Customers System

Ex: stg_orders table

	O_ORDERKEY	O_CUSTKEY	O_ORDERSTATUS	O_TOTALPRICE	O_ORDERDATE	O_ORDERPRIORITY	O_CLERK	O_SHIPPRIORITY	O_COMMENT
1	35193217	649075	O	292508	1998-04-20	2-HIGH	Clerk#000004204	0	r across the pending, even foxes. even, final r
2	35193218	978914	F	94443	1992-10-07	2-HIGH	Clerk#000003648	0	ainst the furiously even ideas. blithely final p
3	35193219	600881	F	131350	1994-01-13	3-MEDIUM	Clerk#000002999	0	beans. fluffily final dependencies are carefully. ironic
4	35193220	998158	O	73130	1997-08-25	5-LOW	Clerk#000008216	0	kly unusual pinto beans cajole carefully. slyly bold gr
5	35193221	10048	O	73715	1995-07-20	2-HIGH	Clerk#000009591	0	o the carefully express dependencies. carefully ironic packages h
6	35193222	859126	O	267717	1997-01-10	2-HIGH	Clerk#000006914	0	frays. furiously bold requests against the requests boost blithely
7	35193223	317587	O	270475	1996-04-01	5-LOW	Clerk#000000826	0	e furiously regular dolphins. carefully unusual accounts boost bli
8	35193248	681700	F	294790	1993-02-26	4-NOT SPECIFIED	Clerk#000008882	0	eposits. even packages around the f
9	35193249	327913	O	225455	1996-10-22	4-NOT SPECIFIED	Clerk#000007713	0	ymptotes. ironic ideas
10	35193250	203765	O	262254	1997-03-07	2-HIGH	Clerk#000000492	0	ss accounts. blithely regular deposits nag carefully. b
11	35193251	162368	F	255357	1992-04-04	2-HIGH	Clerk#000007683	0	s wake furiously ironic
12	35193252	403021	F	62589	1994-07-03	4-NOT SPECIFIED	Clerk#000005902	0	- regular deposits wake after the ironic, express
13	35193253	1050904	F	184535	1992-04-13	5-LOW	Clerk#000005291	0	ithe instructions against the qui

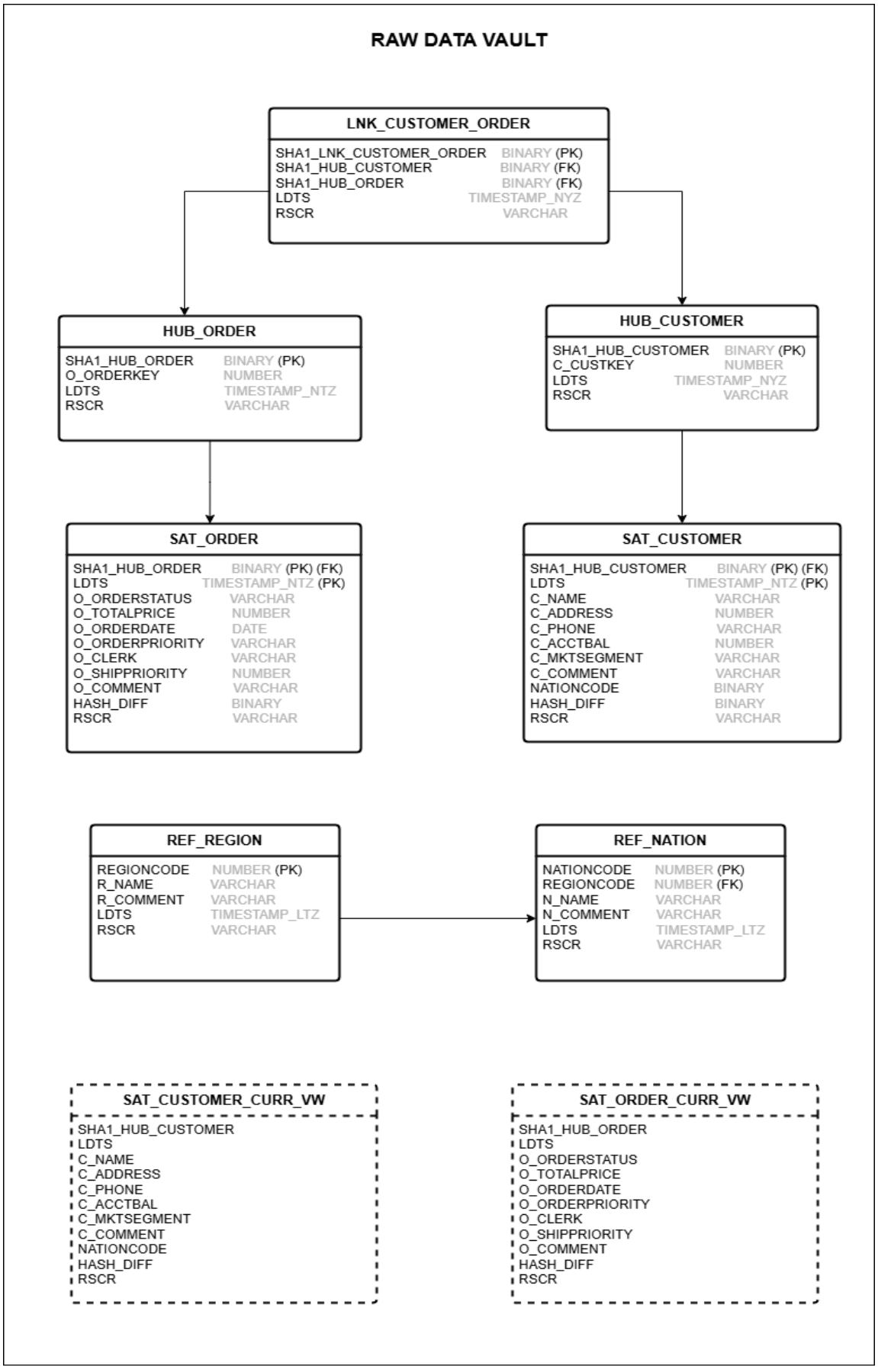
- ❖ Using the Stream data captured from change data in the stg_customer and stg_orders tables, create views for later use.

Ex:

- *CREATE OR REPLACE VIEW stg_customer_strm_outbound AS*
  *SELECT src.**
  , raw_json:C_CUSTKEY::NUMBER c_custkey
  , raw_json:C_NAME::STRING c_name
  , raw_json:C_ADDRESS::STRING c_address
  , raw_json:C_NATIONKEY::NUMBER C_nationcode
  , raw_json:C_PHONE::STRING c_phone
  , raw_json:C_ACCTBAL::NUMBER c_acctbal
  , raw_json:C_MKTSEGMENT::STRING c_mktsegment
  , raw_json:C_COMMENT::STRING c_comment
  , SHA1_BINARY(UPPER(TRIM(c_custkey))) sha1_hub_customer
  , SHA1_BINARY(UPPER(ARRAY_TO_STRING(ARRAY_CONSTRUCT(
  NVL(TRIM(c_name) ,'-1')
  , NVL(TRIM(c_address) ,'-1')
  , NVL(TRIM(c_nationcode) ,'-1')
  , NVL(TRIM(c_phone) ,'-1')
  , NVL(TRIM(c_acctbal) ,'-1')
  , NVL(TRIM(c_mktsegment) ,'-1')
  , NVL(TRIM(c_comment) ,'-1')
  ), '^'))) AS customer_hash_diff
  FROM stg_customer_strm src;

- *CREATE OR REPLACE VIEW stg_order_strm_outbound AS*
  *SELECT src.**
  , SHA1_BINARY(UPPER(TRIM(o_orderkey))) sha1_hub_order
  , SHA1_BINARY(UPPER(TRIM(o_custkey))) sha1_hub_customer
  , SHA1_BINARY(UPPER(ARRAY_TO_STRING(ARRAY_CONSTRUCT(
  NVL(TRIM(o_orderkey) ,'-1')
  , NVL(TRIM(o_custkey) ,'-1')
  ), '^'))) AS sha1_lnk_customer_order
  , SHA1_BINARY(UPPER(ARRAY_TO_STRING(ARRAY_CONSTRUCT(
  NVL(TRIM(o_orderstatus) , '-1')
  , NVL(TRIM(o_totalprice) , '-1')
  , NVL(TRIM(o_orderdate) , '-1')
  , NVL(TRIM(o_orderpriority) , '-1')
  , NVL(TRIM(o_clerk) , '-1')
  , NVL(TRIM(o_shippriority) , '-1')
  , NVL(TRIM(o_comment) , '-1')
  ), '^'))) AS order_hash_diff
  FROM stg_orders_strm src;

3. Raw Data Vault



❖ Create data vault tables

- *USE SCHEMA l10_rdv;*

Ex: hub_customer

- *CREATE OR REPLACE TABLE hub_customer*
  (
  sha1_hub_customer BINARY NOT NULL
  , c_custkey NUMBER NOT NULL
  , ldts TIMESTAMP NOT NULL
  , rscr STRING NOT NULL
  , CONSTRAINT pk_hub_customer PRIMARY KEY(sha1_hub_customer)
  );

Ex: hub_order

- *CREATE OR REPLACE TABLE hub_order*
  (
  sha1_hub_order BINARY NOT NULL
  , o_orderkey NUMBER NOT NULL
  , ldts TIMESTAMP NOT NULL
  , rscr STRING NOT NULL
  , CONSTRAINT pk_hub_order PRIMARY KEY(sha1_hub_order)
  );

Ex: sat_customer

- *CREATE OR REPLACE TABLE sat_customer*
  (
  sha1_hub_customer BINARY NOT NULL
  , ldts TIMESTAMP NOT NULL
  , c_name STRING
  , c_address STRING
  , c_phone STRING
  , c_acctbal NUMBER
  , c_mktsegment STRING
  , c_comment STRING
  , nationcode NUMBER
  , hash_diff BINARY NOT NULL
  , rscr STRING NOT NULL
  , CONSTRAINT pk_sat_customer PRIMARY KEY(sha1_hub_customer, ldts)

```
, CONSTRAINT fk_sat_customer      FOREIGN KEY(sha1_hub_customer)
REFERENCES hub_customer
);
```

Ex: sat_order

- ```
  CREATE OR REPLACE TABLE sat_order
  (
   sha1_hub_order       BINARY    NOT NULL
  , ldts                TIMESTAMP NOT NULL
  , o_orderstatus       STRING
  , o_totalprice        NUMBER
  , o_orderdate         DATE
  , o_orderpriority     STRING
  , o_clerk             STRING
  , o_shippriority      NUMBER
  , o_comment           STRING
  , hash_diff           BINARY    NOT NULL
  , rscr                STRING    NOT NULL
  , CONSTRAINT pk_sat_order PRIMARY KEY(sha1_hub_order, ldts)
  , CONSTRAINT fk_sat_order FOREIGN KEY(sha1_hub_order) REFERENCES
  hub_order
  );
  ```

Ex: lnk_customer_order

- ```
  CREATE OR REPLACE TABLE lnk_customer_order
  (
   sha1_lnk_customer_order BINARY    NOT NULL
  , sha1_hub_customer      BINARY
  , sha1_hub_order         BINARY
  , ldts                   TIMESTAMP  NOT NULL
  , rscr                   STRING    NOT NULL
  , CONSTRAINT pk_lnk_customer_order  PRIMARY
  KEY(sha1_lnk_customer_order)
  , CONSTRAINT fk1_lnk_customer_order FOREIGN KEY(sha1_hub_customer)
  REFERENCES hub_customer
  , CONSTRAINT fk2_lnk_customer_order FOREIGN KEY(sha1_hub_order)
  REFERENCES hub_order
  );
  ```

❖ Create and load region and nation data into ref_region and ref_nation

Ex: ref_region

- *CREATE OR REPLACE TABLE ref_region*
  (
  regioncode NUMBER
  , ldts TIMESTAMP
  , rscr STRING NOT NULL
  , r_name STRING
  , r_comment STRING
  , CONSTRAINT PK_REF_REGION PRIMARY KEY (REGIONCODE)
)
  AS
  SELECT r_regionkey
  , ldts
  , rscr
  , r_name
  , r_comment
  FROM l00_stg.stg_region;

	REGIONCODE	LDTS	RSCR	R_NAME	R_COMMENT
1	0	2025-03-29T09:46:34.667Z	Static Reference Data	AFRICA	lar deposits. blithely final packages cajole. regular waters are final requests. regu...
2	1	2025-03-29T09:46:34.667Z	Static Reference Data	AMERICA	hs use ironic, even requests. s
3	2	2025-03-29T09:46:34.667Z	Static Reference Data	ASIA	ges. thinly even pinto beans ca
4	3	2025-03-29T09:46:34.667Z	Static Reference Data	EUROPE	ly final courts cajole furiously final excuse
5	4	2025-03-29T09:46:34.667Z	Static Reference Data	MIDDLE EAST	uickly special accounts cajole carefully blithely close requests. carefully final asy...

Ex: ref_nation

- *CREATE OR REPLACE TABLE ref_nation*
  (
  nationcode NUMBER
  , regioncode NUMBER
  , ldts TIMESTAMP
  , rscr STRING NOT NULL
  , n_name STRING
  , n_comment STRING
  , CONSTRAINT pk_ref_nation PRIMARY KEY (nationcode)

```
, CONSTRAINT fk_ref_region FOREIGN KEY (regioncode) REFERENCES
ref_region(regioncode)
)
AS
SELECT n_nationkey
    , n_regionkey
    , ldts
    , rscr
    , n_name
    , n_comment
 FROM l00_stg.stg_nation;
```

	NATIONCODE	REGIONCODE	LDTS	RSCR	N_NAME	N_COMMENT
1	0	0	2025-03-29T09:45:46.638Z	Static Reference Data	ALGERIA	haggle. carefully final deposits detect slyly agai
2	1	1	2025-03-29T09:45:46.638Z	Static Reference Data	ARGENTINA	al foxes promise slyly according to the regular accounts. bold requests alon
3	2	1	2025-03-29T09:45:46.638Z	Static Reference Data	BRAZIL	y alongside of the pending deposits. carefully special packages are about the iro...
4	3	1	2025-03-29T09:45:46.638Z	Static Reference Data	CANADA	eas hang ironic, silent packages. slyly regular packages are furiously over the tit...
5	4	4	2025-03-29T09:45:46.638Z	Static Reference Data	EGYPT	y above the carefully unusual theodolites. final dugouts are quickly across the fu...
6	5	0	2025-03-29T09:45:46.638Z	Static Reference Data	ETHIOPIA	ven packages wake quickly. regu
7	6	3	2025-03-29T09:45:46.638Z	Static Reference Data	FRANCE	refully final requests. regular, ironi
8	7	3	2025-03-29T09:45:46.638Z	Static Reference Data	GERMANY	l platelets. regular accounts x-ray: unusual, regular acco
9	8	2	2025-03-29T09:45:46.638Z	Static Reference Data	INDIA	ss excuses cajole slyly across the packages. deposits print aroun
10	9	2	2025-03-29T09:45:46.638Z	Static Reference Data	INDONESIA	slyly express asymptotes. regular deposits haggle slyly. carefully ironic hockey...
11	10	4	2025-03-29T09:45:46.638Z	Static Reference Data	IRAN	efully alongside of the slyly final dependencies.
12	11	4	2025-03-29T09:45:46.638Z	Static Reference Data	IRAQ	nic deposits boost atop the quickly final requests? quickly regula

- In the previous staging area, if new or changed data has arrived, then use a Task to execute and load the data into hub_customer and sat_customer.
- First, Task checks if there is changed data captured in stg_customer_strm. If there is, the task proceeds and inserts the new data into the Raw Data Vault customer tables. This checking occurs at 1-minute intervals.

Ex:

- ```
  CREATE OR REPLACE TASK customer_strm_tsk
   WAREHOUSE = dv_rdv_wh
   SCHEDULE = '1 minute'
  WHEN
   SYSTEM$STREAM_HAS_DATA('L00_STG.STG_CUSTOMER_STRM')
  AS
  INSERT ALL
  WHEN (SELECT COUNT(1) FROM hub_customer tgt WHERE
  tgt.sha1_hub_customer = src_sha1_hub_customer) = 0
  THEN INTO hub_customer
  ( sha1_hub_customer
  ```
  ```

```
, c_custkey
, ldts
, rscr
)
VALUES
( src_sha1_hub_customer
, src_c_custkey
, src_ldts
, src_rscr
)
WHEN (SELECT COUNT(1) FROM sat_customer tgt WHERE
tgt.sha1_hub_customer = src_sha1_hub_customer AND tgt.hash_diff =
src_customer_hash_diff) = 0
THEN INTO sat_customer
(
 sha1_hub_customer
, ldts
, c_name
, c_address
, c_phone
, c_acctbal
, c_mktsegment
, c_comment
, nationcode
, hash_diff
, rscr
)
VALUES
(
 src_sha1_hub_customer
, src_ldts
, src_c_name
, src_c_address
, src_c_phone
, src_c_acctbal
, src_c_mktsegment
, src_c_comment
, src_nationcode
, src_customer_hash_diff
, src_rscr
)
```

```
SELECT sha1_hub_customer  src_sha1_hub_customer
   , c_custkey        src_c_custkey
   , c_name           src_c_name
   , c_address        src_c_address
   , c_nationcode     src_nationcode
   , c_phone          src_c_phone
   , c_acctbal        src_c_acctbal
   , c_mktsegment     src_c_mktsegment
   , c_comment        src_c_comment
   , customer_hash_diff  src_customer_hash_diff
   , ldts             src_ldts
   , rscr             src_rscr
 FROM l00_stg.stg_customer_strm_outbound src
;
```

	SHA1_HUB_CUSTOMER	LDTS	C_NAME	C_ADDRESS	C_PHONE	C_ACCTBAL	C_MKTSEGMENT	C_COMM
1	204df9a7874b99df0f939013921ac14a2b011fac	2025-03-30T10:08:10.179Z	Customer#001477647	ZZbF0R93M6cbaOVmJYPgENj	12-775-788-9873	2065	HOUSEHOLD	wly. regi
2	300320ee5f0f345deccaf96b255f0ff55171acbf	2025-03-30T10:08:10.179Z	Customer#000590624	GtesdvvgqK wU	21-997-722-7741	6076	HOUSEHOLD	ross the
3	03ca00a383eddefc02400ffe86636fa650311bcb	2025-03-30T10:08:10.179Z	Customer#000525971	HYsfq3t8,fls25cl r4,	19-419-580-1136	2115	BUILDING	cuses ca
4	bbf912fadb09e9739628075e77903405ff9f64ab	2025-03-30T10:08:10.179Z	Customer#001256887	NggdMQo EfOvLiIlzlYomJU6R	21-639-603-3705	3660	AUTOMOBILE	pinto be
5	403af7920c6e3f1e40bd566d8401927410c76e12	2025-03-30T10:08:10.179Z	Customer#000629196	EwfVivi3,Xq1EThfPJsMjMmaLNgExK	13-937-768-9542	-81	AUTOMOBILE	lar theod
6	05f42b593c2809dd35a6cea316e3cc2b3f8df01d	2025-03-30T10:08:10.179Z	Customer#000633700	K9d,NESPmP7o,ImwYgRD1K5w29JU1UUG1t	34-568-782-9708	1190	FURNITURE	foxes are
7	17f2b41711ba25a150b5879ce5bdea6e1ff471c6	2025-03-30T10:08:10.179Z	Customer#000053659	3ICAKSx5WZ1IXDe3UdtzZ4YFdye	22-939-416-5143	430	HOUSEHOLD	ly ironic
8	a402528d0364a975e540daf5106909f18ef748ca	2025-03-30T10:08:10.179Z	Customer#000490075	8OFgy1aqntPmZlaiAqFxg0QForJm5k3IVlHs	29-169-499-2520	9842	AUTOMOBILE	the final
9	a0777594e2614e90c771414f0bfd8636323cafb1	2025-03-30T10:08:10.179Z	Customer#000182606	XlySzzFaZeLDEyCMv9 XPQPUT9	13-892-139-5559	5637	FURNITURE	: furious
10	aa4bb376e61c5bb18e371dfe79c5a1a00fbd8d22	2025-03-30T10:08:10.179Z	Customer#001386846	osaE1pvtQwjutjSLYZwnUM6SmmrmlxcHzPl2uD8	23-843-883-5349	9638	HOUSEHOLD	ajole fur
10	5bac7abd9e7b556b7ce02518c601deb6676c1bbc			1306995		2025-03-30T10:08:10.179Z	Customers System	

- Similar to the hub_customer and sat_customer tables, hub_order, sat_order, and lnk_customer_order first check the stg_order_str stream, and if there is changed data, those tables will be populated

Ex:

- *CREATE OR REPLACE TASK order_strm_tsk*

```
 WAREHOUSE = dv_rdv_wh
 SCHEDULE = '1 minute'
WHEN
 SYSTEM$STREAM_HAS_DATA('L00_STG.STG_ORDERS_STRM')
AS
INSERT ALL
WHEN (SELECT COUNT(1) FROM hub_order tgt WHERE tgt.sha1_hub_order =
src_sha1_hub_order) = 0
THEN INTO hub_order
```

```
( sha1_hub_order
, o_orderkey
, ldts
, rscr
)
VALUES
( src_sha1_hub_order
, src_o_orderkey
, src_ldts
, src_rscr
)
WHEN (SELECT COUNT(1) FROM sat_order tgt WHERE tgt.sha1_hub_order =
src_sha1_hub_order AND tgt.hash_diff = src_order_hash_diff) = 0
THEN INTO sat_order
(
 sha1_hub_order
, ldts
, o_orderstatus
, o_totalprice
, o_orderdate
, o_orderpriority
, o_clerk
, o_shippriority
, o_comment
, hash_diff
, rscr
)
VALUES
(
 src_sha1_hub_order
, src_ldts
, src_o_orderstatus
, src_o_totalprice
, src_o_orderdate
, src_o_orderpriority
, src_o_clerk
, src_o_shippriority
, src_o_comment
, src_order_hash_diff
, src_rscr
)
```

```
WHEN (SELECT COUNT(1) FROM lnk_customer_order tgt WHERE
tgt.sha1_lnk_customer_order = src_sha1_lnk_customer_order) = 0
THEN INTO lnk_customer_order
(
 sha1_lnk_customer_order
, sha1_hub_customer
, sha1_hub_order
, ldts
, rscr
)
VALUES
(
 src_sha1_lnk_customer_order
, src_sha1_hub_customer
, src_sha1_hub_order
, src_ldts
, src_rscr
)
SELECT sha1_hub_order        src_sha1_hub_order
    , sha1_lnk_customer_order src_sha1_lnk_customer_order
    , sha1_hub_customer      src_sha1_hub_customer
    , o_orderkey           src_o_orderkey
    , o_orderstatus         src_o_orderstatus
    , o_totalprice         src_o_totalprice
    , o_orderdate          src_o_orderdate
    , o_orderpriority       src_o_orderpriority
    , o_clerk             src_o_clerk
    , o_shippriority        src_o_shippriority
    , o_comment            src_o_comment
    , order_hash_diff       src_order_hash_diff
    , ldts                src_ldts
    , rscr                src_rscr
 FROM l00_stg.stg_order_strm_outbound src;
```

- ❖ Start the Tasks

Ex:

```
ALTER TASK customer_strm_tsk RESUME;
ALTER TASK order_strm_tsk    RESUME;
```

	SHA1_HUB_ORDER	O_ORDERKEY	LDTS	RSCR
1	d47a2ddcea42d59fe87f31b3110301caf038ecc4	35193217	2025-03-29T11:23:26.392Z	Orders System
2	369ad50a5977f9c98033d46126f321ddf2620a68	35193218	2025-03-29T11:23:26.392Z	Orders System
3	b7d33d48341439dc8bc814c1682a8b991a2ad515	35193219	2025-03-29T11:23:26.392Z	Orders System
4	ff100f8936029878e1be96d4d87f4ac968fa892e	35193220	2025-03-29T11:23:26.392Z	Orders System
5	590261f5c8953eca9d975153c8161073f1ff14ff	35193221	2025-03-29T11:23:26.392Z	Orders System
6	d100e6222843bb69d79c845fcca7ee1a28f3321d	35193222	2025-03-29T11:23:26.392Z	Orders System
7	b42de71fc9a00b8774b244185995777788ec04e76	35193223	2025-03-29T11:23:26.392Z	Orders System
8	dbf6bce2c3f3062cf9b041bc5617f8499143c6ed	35193248	2025-03-29T11:23:26.392Z	Orders System
9	79457443d29270b910b6eee671f228106dd4f8bf	35193249	2025-03-29T11:23:26.392Z	Orders System
10	4761a946ece3bfb9bd1f72443ed1baa72cad1471	35193250	2025-03-29T11:23:26.392Z	Orders System
11	25a305e82c871cc66970d3162ea1c73541af0f55	35193251	2025-03-29T11:23:26.392Z	Orders System

	SHA1_LNK_CUSTOMER_ORDER	SHA1_HUB_CUSTOMER	SHA1_HUB_ORDER	LDTS	RSCR
1	3db4eeaff93b3b4a7939872e4fed5ed90c3203fc	01c8306aa5afea993648ebb4abfe3b2bd97f6a18	106d928f4edbd61c4adf520c4086ed62f97a4ff9	2025-03-30T07:19:45.306Z	Orders System
2	86a2942c6c3a988cfe3959707bb25d8dd2bd3cca	6dfd2986ad4a1807a0410c6f9089fb76d5914d18	921b01d85cb66edc4ccfdfa40aaa7a9d91ee0d2b	2025-03-30T07:19:45.306Z	Orders System
3	b017737e6a176238da08df67b5eb1548a4959abe	1197514682b7d8043259b5487aade254e7892286	f3be83b7e78e8f5407e08cce800382870ed3a24d	2025-03-30T07:19:45.306Z	Orders System
4	4815b9b234fd0eabf6952118bf24956d9c5f96b5	ac03bace9be47f3fbfca2a2dfc39ab797108731d	19d611fe1b085bd87b477aa20ee92fbd13c3c35a	2025-03-30T07:19:45.306Z	Orders System
5	b67c0361c3335b2b0dc752473c2894c33ef532d4	57259a4d1fc7c1972a209efcb3e573f433bd03c5	70d1ebad91cc5be77571b41675fbc5e7aff3754c	2025-03-30T07:19:45.306Z	Orders System
6	f910e9dab51c3ba07a28d7652323e17f09c6ac9b	31348ba88ee94330d57cdd545f7f7742e617089c	57bb3cee57fb5a46d0ed987f1b58bbef21fc4b06	2025-03-30T07:19:45.306Z	Orders System
7	8e60b56ff89bb49b5fb2f22b5382e21dfb9605ba	803c7cfd4ff3f91be04018e0b51c7ee9075db72a	131b1454daf418442765118f840b598e40f18ee1	2025-03-30T07:19:45.306Z	Orders System
8	1daf531fc3ebfc762eff1477667d32693aa1a4c7	6bf7e0e03732d8eae3bb536a03e590813294182b	f04de9e65eb67ccd501118eeaa3967d20d3778c9	2025-03-30T07:19:45.306Z	Orders System
9	a084512b8aca419a39cb6df12f126ba33800d650	378fc77e7a78581856fd24a167b55a57cbd4cf27	b9db4cd3c6231569e8212ec4db6f35009ed5f73c	2025-03-30T07:19:45.306Z	Orders System
10	7a61c9d457fcb77f2915e042fc512ebbf66b7652	77d0b587c2cff89c34578f97d3a86c39f32f8631	405d953f1b3a9c1bd4627aa368ed5c00fc6fb232	2025-03-30T07:19:45.306Z	Orders System
11	ebd6dee5ddcce71cc381917c70ce96c9806f7e1e	7f917a555b60d436d1e77038b57bf70748e8c22b	ccd20d87865155cb853bea1e6a17825aa52c7ee2	2025-03-30T07:19:45.306Z	Orders System

	SHA1_HUB_ORDER	LDTS	O_ORDERSTATUS	O_TOTALPRICE	O_ORDERDATE	O_ORDERPRIORITY	O_CLERK	O_SHIPPRIORITY	O_COMMENT
1	106d928f4edbd61c4adf520c4086ed62f97a4ff9	2025-03-30T07:19:45.306Z	F	87957	1994-02-28	2-HIGH	Clerk#000004403	0	slyly silent ideas print abov
2	921b01d85cb66edc4ccfdfa40aaa7a9d91ee0d2b	2025-03-30T07:19:45.306Z	O	218508	1995-06-29	1-URGENT	Clerk#000004708	0	al theodolites according to
3	f3be83b7e78e8f5407e08cce800382870ed3a24d	2025-03-30T07:19:45.306Z	O	53903	1996-10-07	2-HIGH	Clerk#000001097	0	nts sleep according to the
4	19d611fe1b085bd87b477aa20ee92fbd13c3c35a	2025-03-30T07:19:45.306Z	O	97358	1997-05-02	1-URGENT	Clerk#000001460	0	ic deposits. pinto beans af
5	70d1ebad91cc5be77571b41675fbc5e7aff3754c	2025-03-30T07:19:45.306Z	O	69581	1997-07-03	3-MEDIUM	Clerk#000006020	0	ven accounts sleep
6	57bb3cee57fb5a46d0ed987f1b58bbef21fc4b06	2025-03-30T07:19:45.306Z	O	131072	1997-05-19	1-URGENT	Clerk#000000086	0	he bold packages use requ
7	131b1454daf418442765118f840b598e40f18ee1	2025-03-30T07:19:45.306Z	F	42287	1993-12-25	3-MEDIUM	Clerk#000006899	0	ar deposits cajole furiously
8	f04de9e65eb67ccd501118eeaa3967d20d3778c9	2025-03-30T07:19:45.306Z	F	141021	1992-09-20	2-HIGH	Clerk#000001395	0	maintain quickly even pack
9	b9db4cd3c6231569e8212ec4db6f35009ed5f73c	2025-03-30T07:19:45.306Z	F	218805	1992-01-04	3-MEDIUM	Clerk#000004700	0	s above the regular reques
10	405d953f1b3a9c1bd4627aa368ed5c00fc6fb232	2025-03-30T07:19:45.306Z	O	93871	1997-08-13	2-HIGH	Clerk#000007024	0	its. blithely regular instruct
11	ccd20d87865155cb853bea1e6a17825aa52c7ee2	2025-03-30T07:19:45.306Z	F	354423	1994-06-25	1-URGENT	Clerk#000005153	0	usual decoys haggle. silen
12	d9b1817b08e493f0d3ea37eb54687e4ca01b5e60	2025-03-30T07:19:45.306Z	F	273310	1992-01-23	2-HIGH	Clerk#000001697	0	haggle fluffily fur

- ❖ Create two views for later use in the Information Delivery stage.

Ex: sat_customer_curr_vw

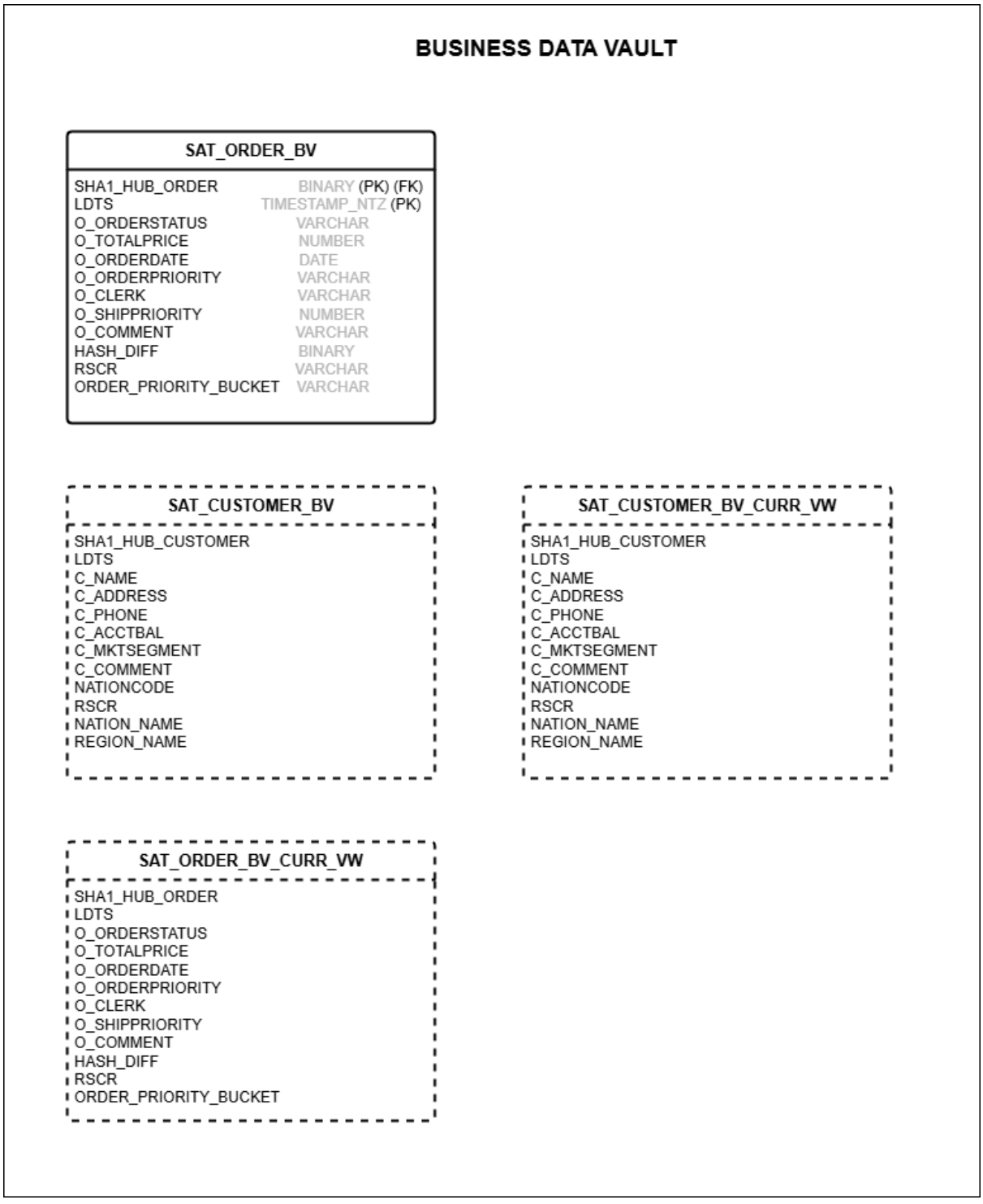
- *CREATE VIEW sat_customer_curr_vw*
  AS
  *SELECT **
  FROM sat_customer
  QUALIFY LEAD(ldts) OVER (PARTITION BY sha1_hub_customer ORDER BY ldts) IS NULL;

Ex: sat_order_curr_vw

- *CREATE OR REPLACE VIEW sat_order_curr_vw*
  AS
  *SELECT **

```
FROM    sat_order
QUALIFY LEAD(ldts) OVER (PARTITION BY sha1_hub_order ORDER BY ldts) IS
NULL;
```

4. Business Data Vault



- Create a view

Ex: sat_customer_bv

- *CREATE OR REPLACE VIEW sat_customer_bv*
  AS

```
SELECT rsc.sha1_hub_customer
    , rsc.ldts
    , rsc.c_name
    , rsc.c_address
    , rsc.c_phone
    , rsc.c_acctbal
    , rsc.c_mktsegment
    , rsc.c_comment
    , rsc.nationcode
    , rsc.rscr
    , rrn.n_name              nation_name
    , rrr.r_name              region_name
 FROM l10_rdv.sat_customer       rsc
 LEFT OUTER JOIN l10_rdv.ref_nation rrn
   ON (rsc.nationcode = rrn.nationcode)
 LEFT OUTER JOIN l10_rdv.ref_region rrr
   ON (rrn.regioncode = rrr.regioncode)
;
```

- Create a table to populate data

Ex: sat_order_bv

```
CREATE OR REPLACE TABLE sat_order_bv
(
 sha1_hub_order       BINARY   NOT NULL
, ldts                TIMESTAMP NOT NULL
, o_orderstatus       STRING
, o_totalprice        NUMBER
, o_orderdate         DATE
, o_orderpriority     STRING
, o_clerk             STRING
, o_shippriority      NUMBER
, o_comment           STRING
, hash_diff           BINARY   NOT NULL
, rscr                STRING   NOT NULL
-- additional attributes
, order_priority_bucket  STRING
, CONSTRAINT pk_sat_order PRIMARY KEY(sha1_hub_order, ldts)
```

```
, CONSTRAINT fk_sat_order FOREIGN KEY(sha1_hub_order) REFERENCES
l10_rdv.hub_order
)
AS
SELECT sha1_hub_order
    , ldts
    , o_orderstatus
    , o_totalprice
    , o_orderdate
    , o_orderpriority
    , o_clerk
    , o_shippriority
    , o_comment
    , hash_diff
    , rscr
    -- derived additional attributes
    , CASE WHEN o_orderpriority IN ('2-HIGH', '1-URGENT')          AND o_totalprice
>= 200000 THEN 'Tier-1'
          WHEN o_orderpriority IN ('3-MEDIUM', '2-HIGH', '1-URGENT') AND
o_totalprice BETWEEN 150000 AND 200000 THEN 'Tier-2'
          ELSE 'Tier-3'
      END order_priority_bucket
  FROM l10_rdv.sat_order;
```

- Create a new task to populate the sat_order_bv table after the order_strm_task (the task to populate hub_order, sat_order, and lnk_customer_order from stream data in the staging area) is executed.

  Ex:

```
CREATE OR REPLACE TASK l10_rdv.hub_order_strm_sat_order_bv_tsk
  WAREHOUSE = dv_rdv_wh
  AFTER l10_rdv.order_strm_tsk
AS
INSERT INTO l20_bdv.sat_order_bv
SELECT
  sha1_hub_order
, ldts
, o_orderstatus
, o_totalprice
```

```
, o_orderdate
, o_orderpriority
, o_clerk
, o_shippriority
, o_comment
, hash_diff
, rscr
-- derived additional attributes
, CASE WHEN o_orderpriority IN ('2-HIGH', '1-URGENT')          AND o_totalprice
>= 200000 THEN 'Tier-1'
     WHEN o_orderpriority IN ('3-MEDIUM', '2-HIGH', '1-URGENT') AND
o_totalprice BETWEEN 150000 AND 200000 THEN 'Tier-2'
     ELSE 'Tier-3'
  END order_priority_bucket
FROM sat_order_strm;

ALTER TASK l10_rdv.hub_order_strm_sat_order_bv_tsk RESUME;
ALTER TASK l10_rdv.order_strm_tsk RESUME;
```

- ❖ Create 2 views for later use in Information Delivery stage

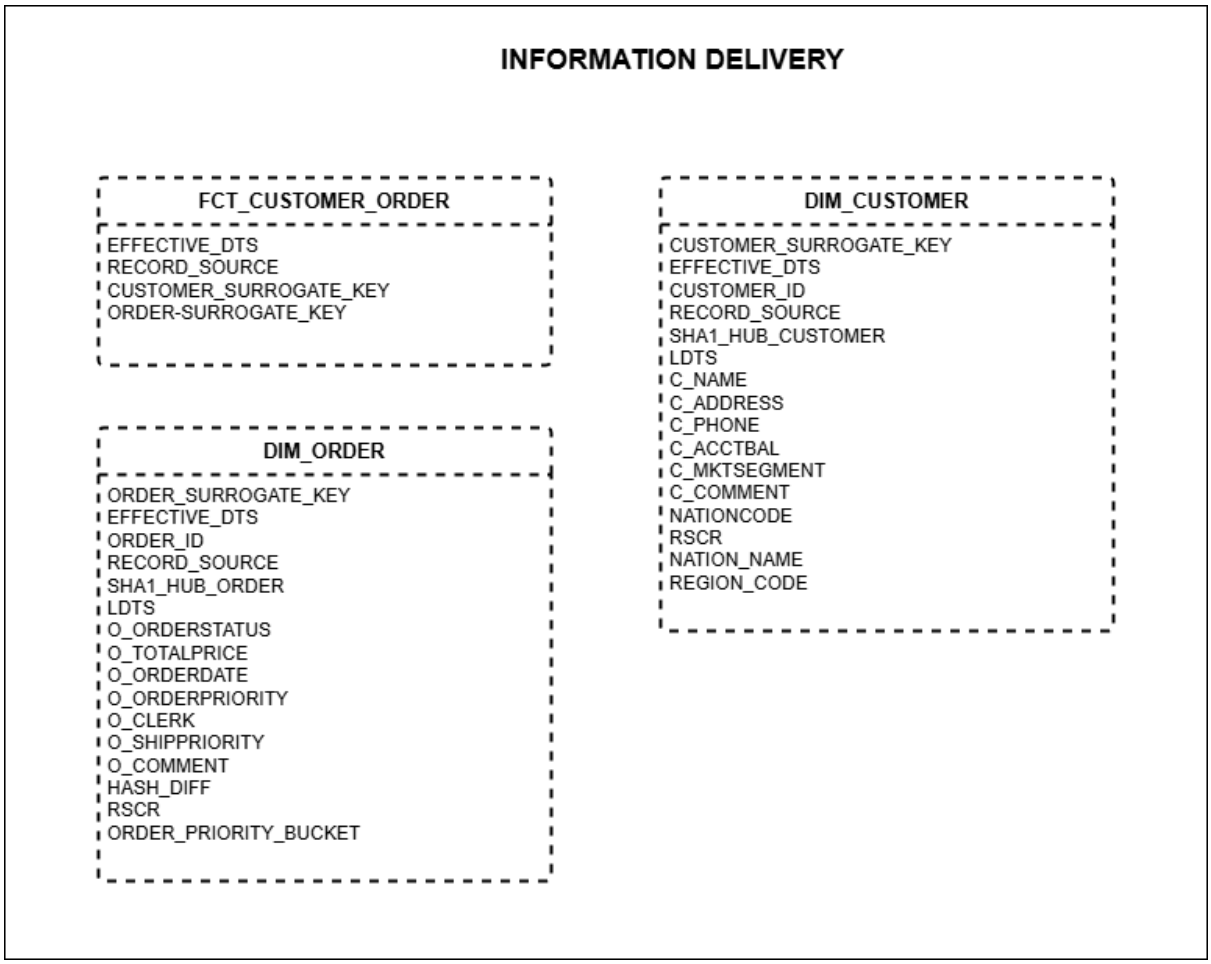
Ex: sat_order_bv_curr_vw

- ```
  CREATE VIEW sat_order_bv_curr_vw
  AS
  SELECT  *
  FROM    sat_order_bv
  QUALIFY LEAD(ldts) OVER (PARTITION BY sha1_hub_order ORDER BY ldts) IS
  NULL;
  ```

Ex: sat_customer_bv_curr_vw

- ```
  CREATE VIEW sat_customer_bv_curr_vw
  AS
  SELECT  *
  FROM    sat_customer_bv
  QUALIFY LEAD(ldts) OVER (PARTITION BY sha1_hub_customer ORDER BY ldts) IS
  NULL;
  ```

5. Information Delivery



- Implement dimensional modelling views for fast analytics queries and reporting.

Ex: dim_customer

- ```
  CREATE OR REPLACE VIEW dim1_customer
  AS
  SELECT hub.sha1_hub_customer            AS dim_customer_key
      , sat.ldts                  AS effective_dts
      , hub.c_custkey                 AS customer_id
      , sat.rscr                  AS record_source
      , sat.*
   FROM l10_rdv.hub_customer              hub
      , l20_bdv.sat_customer_bv_curr_vw      sat
   WHERE hub.sha1_hub_customer             = sat.sha1_hub_customer;
  ```
  ```

Ex: dim_order

- *CREATE OR REPLACE VIEW dim1_order*
  AS
  SELECT hub.sha1_hub_order AS dim_order_key
  , sat.ldts AS effective_dts
  , hub.o_orderkey AS order_id
  , sat.rscr AS record_source
  *, sat.**
  FROM l10_rdv.hub_order hub
  , l20_bdv.sat_order_bv_curr_vw sat
  WHERE hub.sha1_hub_order = sat.sha1_hub_order;

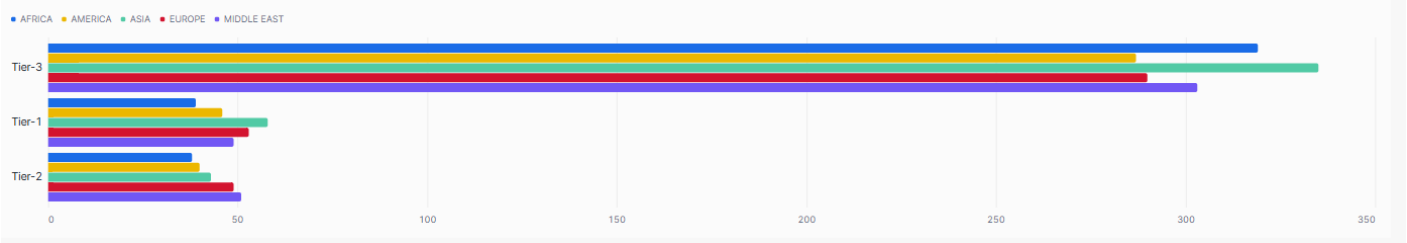
Ex: fct_customer_order

- *CREATE OR REPLACE VIEW fct_customer_order*
  AS
  SELECT lnk.ldts AS effective_dts
  , lnk.rscr AS record_source
  , lnk.sha1_hub_customer AS dim_customer_key
  , lnk.sha1_hub_order AS dim_order_key
  -- this is a factless fact, but here you can add any measures, calculated or derived
  FROM l10_rdv.lnk_customer_order lnk;

❖ Create charts from Snowsight

Ex:

- *SELECT dc.nation_name*
  , dc.region_name
  , do.order_priority_bucket
  , COUNT(1) cnt_orders
  FROM fct_customer_order fct
  , dim1_customer dc
  , dim1_order do
  WHERE fct.dim_customer_key = dc.dim_customer_key
  AND fct.dim_order_key = do.dim_order_key
  GROUP BY 1,2,3;


AFRICA
AMERICA
ASIA
EUROPE
MIDDLE EAST
Tier-3
Tier-1
Tier-2
0
50
100
150
200
250
300
350

References

- [https://quickstarts.snowflake.com/guide/vhol_data_vault/index.html?index=..%2F..index#0] – Snowflake
- [https://www.snowflake.com/en/emea/] - Snowflake