DATAWAREHOUSE ASSESSMENT

1. Category of a product may change over a period of time. Historical category information (current category as well as all old categories) has to be stored. Which SCD type will be sutiable to implement this requirement? What kind of structure changes are required in a dimenstion table to implement SCD type 2 and type 3.

Historical category information is stored used SCD2. ie

- Slowly Changing Dimension Type 2 also known SCD Type 2 is one of the most commonly used type of Dimension table in a Data Warehouse.
- This keeps current as well as historical data in the table.
- It allows you to insert new records and changed records using a new column.
- This method tracks historical data by creating multiple records for a given natural key in the dimensional tables with separate surrogate keys and/or different version numbers.

Example: Here I'm taking scenario where dimension is changing slowly and it require history to be maintained. If the supplier CHANGES the version numbers.

SUPPLIER_KEY	SUPPLIER_NAME	SUPPLIER	VERSIONS
101	ABC	S1	0
102	XYZ	S2	1

AFTER FEW DAYS

SUPPLIER_KEY	SUPPLIER_NAME	SUPPLIER	VERSIONS
101	ABC	S1	1
102	XYZ	S2	1

The structure change FULL LOAD FOR SCD2

SUPPLIER_KEY	SUPPLIER_NAME	SUPPLIER	VERSIONS	START_DATE	END_DATE
101	ABC	S1	0	DATE1	NULL
102	XYZ	S2	1	DATE1	NULL

INCREMENTAL LOAD FOR SCD2

The Start date/time of the changed row is equal to the End date/time of the previous row. The null End_Date in row indicates the current tuple version

SUPPLIER_KEY	SUPPLIER_NAME	SUPPLIER	VERSIONS	START_DATE	END_DATE
101	ABC	S1	0	DATE1	DATE3
102	XYZ	S2	1	DATE1	NULL
103	ABC	S1	1	DATE3	NULL

SCD3

- Slowly Changing Dimension Type 3 also known SCD Type 3 is one of the most commonly used type of Dimension table in a Data Warehouse.
- This method tracks changes using separate columns and preserves limited history.
- The Type 3 preserves limited history as it is limited to the number of columns designated for storing historical data.

SUPPLIER_KEY	SUPPLIER_NAME	SUPPLIER	VERSIONS
101	ABC	S1	0
102	XYZ	S2	1

AFTER FEW DAYS

SUPPLIER_KEY	SUPPLIER_NAME	SUPPLIER	VERSIONS
101	ABC	S1	1

102	XYZ	S2	1

FULL LOAD FOR SCD3

In the following example, an additional column has been added to the table to record the supplier's original version - only the previous version is stored.

SUPPLIER_KEY	SUPPLIER_NAME	SUPPLIER	VERSIONS	PREVIOUS VERSION
101	ABC	S1	0	NULL
102	XYZ	S2	1	NULL

INCREMENTAL LOAD FOR SCD3

SUPPLIER_KEY	SUPPLIER_NAME	SUPPLIER	VERSIONS	PREVIOUS VERSION
101	ABC	S1	1	0
102	XYZ	S2	1	NULL

2. What is surrogate key? Why it is required?

- It is sequentially generated unique number attached with each and every record in a Dimension table in any **Data Warehouse**.
- Basically it's an artificial key that is used as a substitute for a Natural Key (NK). We should have defined NK in our tables as per the business requirement and that might be able to uniquely identify any record.
- It join between the fact and dimension tables and is necessary to handle changes in dimension table attributes.
- The Primary Key identifies the unique row in the database while the Surrogate Key identifies a unique entity in the model
- It is UNIQUE since it is sequentially generated integer for each record being inserted

in the table.

- It is SEQUENTIAL since it is assigned in sequential order as and when new records are created in the table, starting with one and going up to the highest number that is needed.

ADVANTAGES OR WHY SURROGATE KEYS ARE USED

- **Performance**: Fast response times often mean efficient joins between facts and dimensions
- **Changing Source Systems**: Using natural keys tightly ties a data warehouse's integrity to the stability of the source system.
- **Slow Changing Dimensions**: It is often a requirement to track historical values of dimension records.
- **Surrogate keys are unique.** Because surrogate keys are system-generated, it is impossible for the system to create and store a duplicate value.
- **Surrogate keys apply uniform rules to all records.** The surrogate key value is the result of a program, which creates the system-generated value. Any key created as a result of a program will apply uniform rules for each record.
- **Surrogate keys stand the test of time.** Because surrogate keys lack any context or business meaning, there will be no need to change the key in the future.
- **Surrogate keys allow for unlimited values.** Sequential, timestamp, and random keys have no practical limits to unique combinations

dim_table

PATIENT_SK	PATIENT_ID	PATIENT_NAME	PATIENT_AGE
1	P001	ABC	20
2	P002	BCD	25
3	P003	CDE	19
4	P004	DEF	45

fact_table

FCT_SK	AMT	DATE
1	1000	1/1/2017
2	1500	1/2/2017
3	700	1/3/2017
4	1200	1/4/2017

4. What is a semi-additive measure? Give an example.

Semi additive facts are facts that can be summed up for some of the dimensions in the fact table,but not others.

Current_balance is a semi_additive fact, as it makes sense to add them up for all accounts

CUSTOMER_DIM

Date
Account
Current_balance(SA)
Profit
daily_balance

3. Stores are grouped in to multiple clusters. A store can be part of one or more clusters. Design tables to store this store-cluster mapping information.

A **table cluster** is a group of tables that share common columns and store related data in the same blocks.

The **cluster key** is the column or columns that the clustered tables have in common.

Consider clustering tables when they are primarily queried (but not modified) and records from the tables are frequently queried together or joined. Because table clusters store related rows of different tables in the same data blocks, properly used table clusters offer the Less storage is required to store related table and index data because the cluster key value is not stored repeatedly for each row.

Cluster mapping is joining two or more tables.

Store_cluster

STOREID

20	Store_name	Location_id
	Ganesha_juice	560010

Owner id	ownername
201	harish
202	sunny

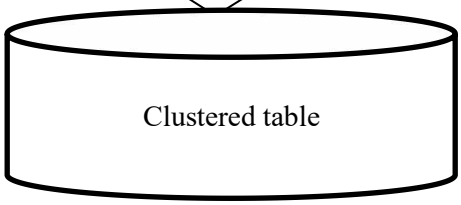
Cluster key is **store id**

department_cluster

STOREID

110	department_name	Location_id
	hotel	560010

id	ownername
201	abhi
202	adi



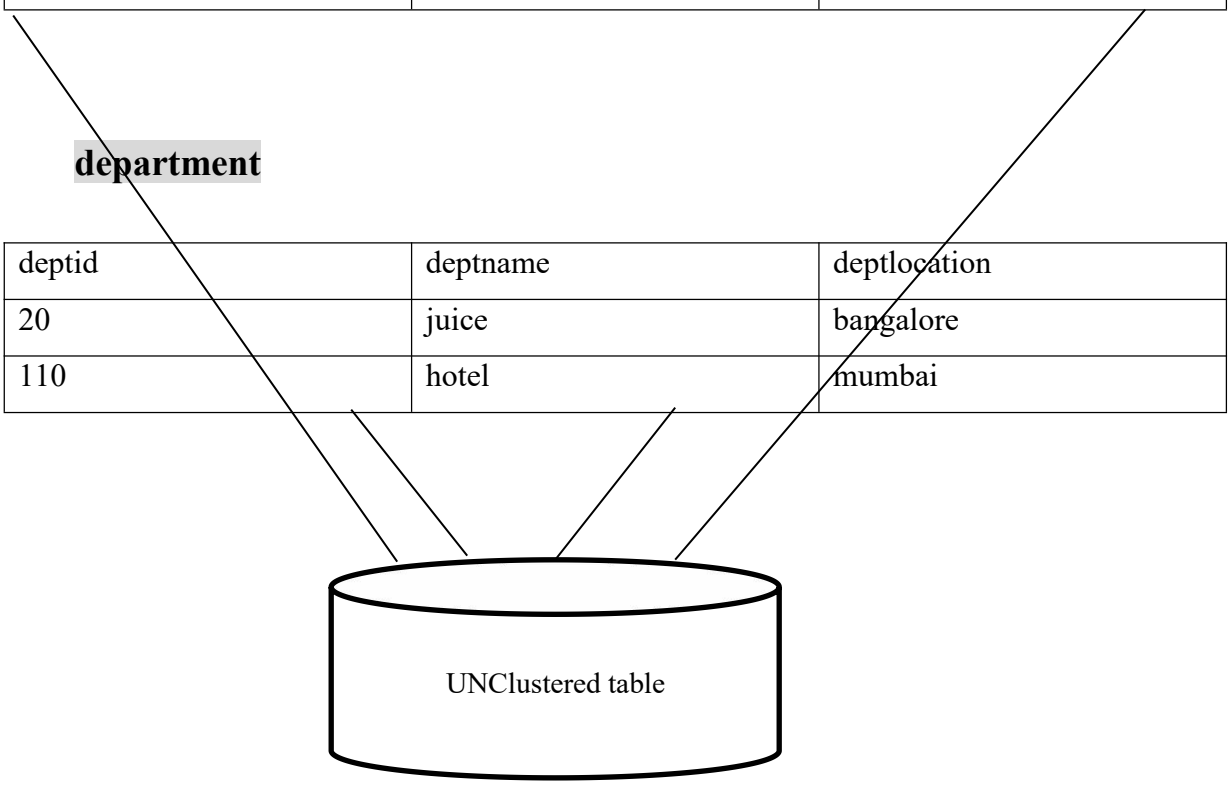
Ie joining two tables .

Stores

Store_id	owner_name	Department id
201	abhi	20
202	adi	20
203	sunny	110
204	harish	110

department

deptid	deptname	deptlocation
20	juice	bangalore
110	hotel	mumbai



Create cluster store_cluster

(store_id number(4));

Create index idx_store_dept_cluster

On cluster store_cluster;

The database stores the rows and locates them with the index shows the store_cluster , which contains employees and departments. The database stores rows for store in department 20 together, department 110 together. If the tables are not clustered, then the database does not ensure that the related rows are stored together.