**DATA WAREHOUSING FINAL ASSESSMENT DOCUMENT**

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 suitable to implement this requirement? What kind of structure changes are required in a dimension table to implement SCD type 2 and type 3.

Answer:

A Slow Changing Dimension (SCD) is a dimension that stores and manages both current and historical data over time in a data warehouse.

The type of SCD suitable to implement the given requirement will be SCD Type-2 -Effective Date Range Mapping.

A Type-2 SCD retains the full history of values. When the value of a chosen attribute changes, the current record is closed. A new record is create with the changed values and this new record becomes the current record. Each record contains the effective time and expiration time to identify the time period between which the record was active.

Using SCD 2, one can easily save unlimited history with the help of surrogate key. In this structure, the table will never be effected(constant), only the no of rows will be effected(increased) and to prevent the duplication of data, primary key will be used.

Original Table:

|  |  |  |
| --- | --- | --- |
| PRODUCT\_ID | PRODUCT | CATEGORY |
| 1 | Lays | Chips |
| 2 | Amul Milk | Dairy |

SCD Type 2:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STATUS ID | PRODUCT\_ID | PRODUCT | CATEGORY | ST\_DATE | ED\_DATE |
| 100 | 1 | Lays | Chips | 01-01-2019 | 15-06-2019 |
| 101 | 2 | Amul Milk | Dairy | 01-01-2019 |  |
| 102 | 3 | Lays | Snacks | 16-06-2019 |  |

The structure changes happens in dimension table to implement SCD Type-2 and SCD Type-3 are:

ORIGINAL TABLE:

|  |  |  |
| --- | --- | --- |
| PRODUCT\_ID | PRODUCT | CATEGORY |
| 1 | Lays | Chips |
| 2 | Amul Milk | Dairy |

SCD TYPE-2:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STATUS ID | PRODUCT\_ID | PRODUCT | CATEGORY | ST\_DATE | ED\_DATE |
| 100 | 1 | Lays | Chips | 01-01-2019 | 15-06-2019 |
| 101 | 2 | Amul Milk | Dairy | 01-01-2019 |  |
| 102 | 3 | Lays | Snacks | 16-06-2019 |  |

SCD2 allows you to insert new records and changed records using two new columns (ST\_DATE and ED\_DATE) by maintaining the date range in the table to track the changes. It uses a n column primary key(STATUS\_ID) to maintain the history.

SCD TYPE-3:

|  |  |  |  |
| --- | --- | --- | --- |
| PRODUCT\_ID | PRODUCT | PREVIOUS  CATEGORY | CURRENT CATEGORY |
| 1 | Lays | Chips | Snacks |
| 2 | Amul Milk | Dairy |  |

SCD3 keeps current as well as historical data in the table. It maintains only partial historyby adding a new column PREVIOUS\_CATEGORY(previous column name). It does not maintain full history.

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

Answer:

A surrogate key is a system generated value with no business meaning that is used to uniquely identify a record in a table. The key itself could be made up of one or multiple columns.

A surrogate key like a natural key (primary key) is a column that uniquely identifies a single record in a table. But this is where the similarity stops. Surrogate keys are like surrogate mothers. They are keys that don’t have a natural relationship with rest of the table. The surrogate key is just a value that is generated and then stored with the rest of the columns in a record. The key value is typically generated at run time right before the record is inserted into a table. It is sometimes also referred to as a dumb key, because there is no meaning associated with the value. Surrogate keys are commonly a numeric number.

Surrogate Key Pros:

* N business logic in key so no changes based on business requirements.
* Less code if maintaining same key strategy across all entities.
* Better performance since key value is smaller. Less disk IO is required on, when accessing single column indexes.
* Surrogate key is guaranteed to be unique.
* If a sequence used then there is little index maintenance required since the value is ever increasing which leads to less index fragmentation.

Surrogate Keys are allowed when:

1. No property has the parameter of primary key
2. In the table, primary key is too big or complicated

For example, a table EmployeeContract may hold temporal information to keep track of contracted working hours. The business key for one contract will be identical (non-unique) in both rows however the surrogate key for each row is unique.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Surrogate key | Business key | Employee Name | Working Hours Per Week | Row Valid From | Row Valid To |
| 1 | A1019 | Bob | 50 | 01-01-2019 | 15-06-2019 |
| 57 | A4456 | John | 46 | 01-01-2019 | 23-07-2019 |
| 345 | A1019 | Bob | 35 | 16-06-2019 | 29-11-2019 |

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

Answer:

Semi Additive measures are values that you can summarize across any related dimension except time. These are those specific class of fact measures which can be aggregated across all dimension and their hierarchy except the time dimension.

For example:

Sales and costs are fully additive. If you sell 100 yesterday and 50 today then you’ve sold 150 in total. You can add them up over time.

Stock levels however are semi additive. If you had 100 in stock yesterday, and 50 in stock today, you’re total stock is 50, not 150. It doesn’t make sense to add up the measures over time, you need to find the most recent value.

1. 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.

Answer:

STORE

|  |  |  |
| --- | --- | --- |
| **STORE\_ID** | STORE\_NAME | CLUSTER\_ID |
| 101 | A | CL-01 |
| 102 | B | CL-02 |
| 103 | C | CL-02 |

CLUSTER

|  |  |
| --- | --- |
| **CLUSTER\_ID** | CLUSTER\_NAME |
| CL-01 | 1 |
| CL-02 | 2 |
| CL-03 | 3 |

STORE\_CLUSTER TABLE

|  |  |  |  |
| --- | --- | --- | --- |
| **STORE\_ID** | STORE\_NAME | **CLUSTER\_ID** | CLUSTER\_NAME |
| 101 | A | CL-01 | 1 |
| 102 | B | CL-01 | 1 |
| 102 | B | CL-02 | 2 |
| 103 | C | CL-03 | 3 |

STAR SCHEMA FOR STORE\_CLUSTER MAPPING

STORE

STORE\_ID

STORE\_NAME

CLUSTER

CLUSTER\_ID

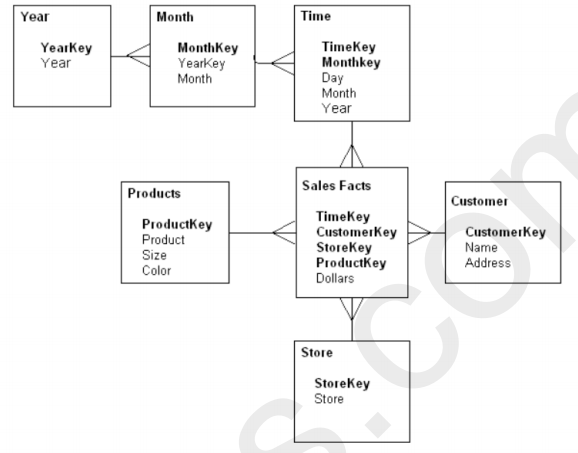
CLUSTER\_NAME

STORE\_CLUSTER

STORE\_ID

CLUSTER\_ID

1. For the given Dimensional Modelling, please identify the following:



*  How many dimensions and Facts are present?

FACT TABLES: 1; Sales Facts

DIMENSION TABLES: 6;

De-Normalised Dimension: 4; Time, Customer, Products, Store

Normalised Dimension:2; YearKey, MonthKey

* Please identify the cardinality between each table?

YEAR ----(One-to-Many)----> MONTH

MONTH ----(One-to-Many)----> TIME

TIME ----(One-to-Many)----> SALES FACTS

PRODUCT ----(One-to-Many)----> SALES FACTS

STORE ----(One-to-Many)----> SALES FACTS

CUSTOMER ----(One-to-Many)----> SALES FACTS

* How to create a Sales\_Aggr fact using the following structure (SQL Statement):



Create table Sales\_Aggr As

(Year\_ID INT(4) PRIMARY KEY,

Customer\_key INT(10) PRIMARY KEY,

Store\_Key INT(10) PRIMARY KEY,

Product\_key INT(20) PRIMARY KEY,

Dollars DOUBLE,

FOREIGN KEY (Year\_ID) REFERENCES Year(YearKey),

FOREIGN KEY (Customer\_key) REFERENCES Customer(CustomerKey),

FOREIGN KEY (Store\_Key) REFERENCES Store(Store\_Key),

FOREIGN KEY (Product\_Key) REFERENCES Product(ProductKey));

* Can you Please Modify the above snowflake schema to Star schema and draw the dimension model, showing all the cardinality?

**TIME**

**TimeKey**

Day

Month

Year

**STORE**

**StoreKey**

Store

**CUSTOMER**

**CustomerKey**

Name

Address

**SALES FACTS**

**TimeKey**

**CustomerKey**

**StoreKey**

**ProductKey**

Dollars

**PRODUCT**

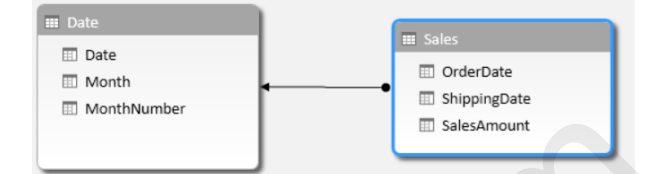
**ProductKey**

Product

Size

Color

1. For the following dimension Model can you please give an example of Circular Join and how to avoid it:



Answer:

Circular Join:

Select max(SalesAmount) from Sales, Date

Where Sales.OrderDate = Date.Date,

Sales.ShippingDate = Date.Date;

Circular Joins or loops occur when say a table A is joined to table B and in turn joined to table A. Hence the loops should be generally avoided.

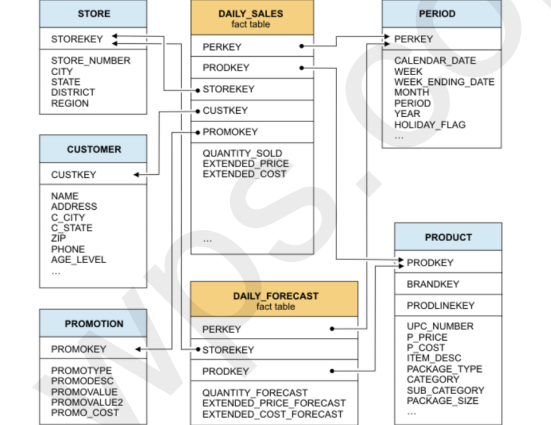
To avoid circular join, we can make use of alias name.

Select max(SalesAmount) from Sales s, Date d1, Date d2

Where s.OrderDate = d1.Date,

s.ShippingDate = d2.Date;

7. For the given Dimension Model, can you please generate a sql to get the total divergence between Quantity sold and Quantity Forecast for the current month for all the stores:



Answer:

Select ((select sum(QUANTITY\_SOLD) from DAILY\_SALES, PERIOD where PERIOD.MONTH = tochar(sysdate,’MM’))

- (select sum(QUANTITY\_FORECAST) from DAILY\_FORECAST, PERIOD

where PERIOD.MONTH = tochar(sysdate, ‘MM’));

8.For the above-mentioned dimension model, please identify the conformed and non-conformed dimensions. Additionally, identify the measure types?

Answer:

Conformed Dimensions: STORE, PERIOD, PRODUCT

Non-Conformed Dimensions: CUSTOMER, PROMOTION

Measures: Additive type: QUANTITY\_SOLD, QUANTITY\_FORECAST

Semi-Additive: EXTENDED\_PRICE, EXTENDED\_COST

Non-Additive: EXTENDED\_PRICE\_FORECAST

EXTENDED\_COST\_FORECAST

9.Make a list of differences between DW and OLTP based on Size, Usage, Processing and Data Models.

Answer:

|  |  |  |
| --- | --- | --- |
|  | DATA WAREHOUSE | OLTP |
| Size | The size of DW is more than terabytes of data | The size of OLTP ranges from few gigabytes to hundreds gigabytes |
| Usage | Type of database used for analytical processing | Collection of objects used for data retrieval, modification, and data access. |
| Processing | Analytical processing may require several minutes to run. | Databases which are OTLP require sub-second response time. |
| Data Models | Data warehouse follows star and snow flake schema model for designing the database. | The database follows the entity-relationship(ER) database, model |