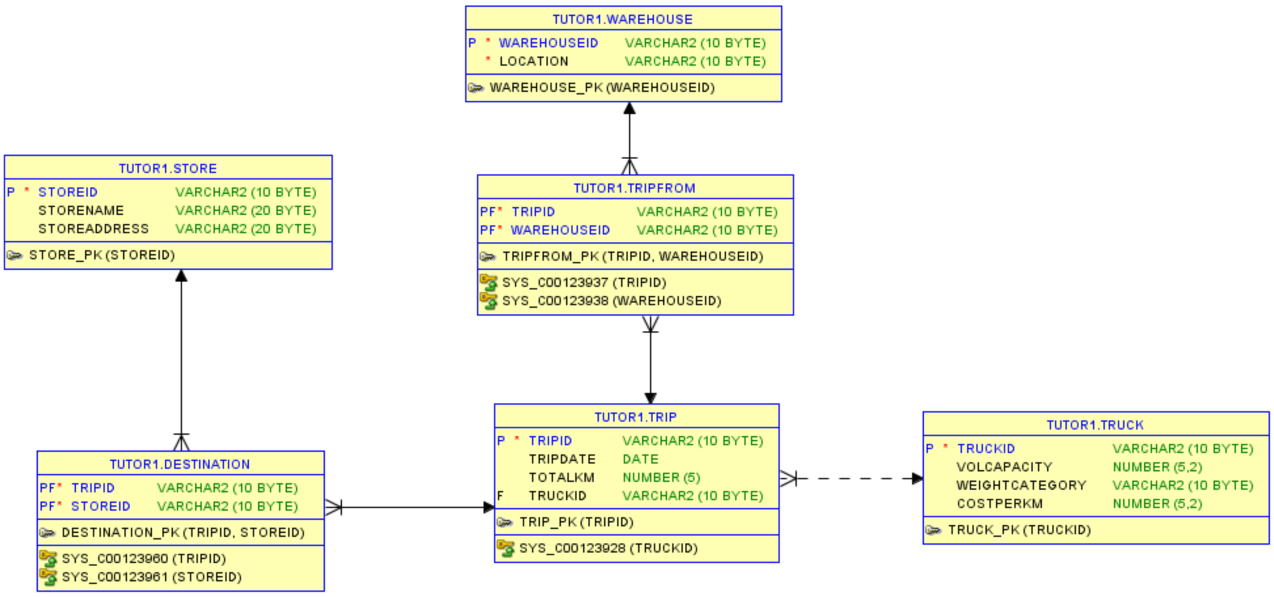
Level of Aggregation and Bridge Tables

Re-visiting the Truck Delivery Case Study

In this lesson, we are going to learn Level of Aggregation when Bridge Tables are used. Let's revisit the **Truck Delivery Case Study**, which has a bridge table connecting the Trip and the Store

The operational database of the Truck Delivery System is as follows:


TUTOR1.WAREHOUSE
P * WAREHOUSEID VARCHAR2 (10 BYTE)
* LOCATION VARCHAR2 (10 BYTE)
WAREHOUSE_PK (WAREHOUSEID)
TUTOR1.STORE
P * STOREID VARCHAR2 (10 BYTE)
STORENAME VARCHAR2 (20 BYTE)
STOREADDRESS VARCHAR2 (20 BYTE)
STORE_PK (STOREID)
TUTOR1.TRIPFROM
PF* TRIPID VARCHAR2 (10 BYTE)
PF* WAREHOUSEID VARCHAR2 (10 BYTE)
TRIPFROM_PK (TRIPID, WAREHOUSEID)
SYS_C00123937 (TRIPID)
SYS_C00123938 (WAREHOUSEID)
TUTOR1.DESTINATION
PF* TRIPID VARCHAR2 (10 BYTE)
PF* STOREID VARCHAR2 (10 BYTE)
DESTINATION_PK (TRIPID, STOREID)
SYS_C00123960 (TRIPID)
SYS_C00123961 (STOREID)
TUTOR1.TRIP
P * TRIPID VARCHAR2 (10 BYTE)
TRIPDATE DATE
TOTALKM NUMBER (5)
F TRUCKID VARCHAR2 (10 BYTE)
TRIP_PK (TRIPID)
SYS_C00123928 (TRUCKID)
TUTOR1.TRUCK
P * TRUCKID VARCHAR2 (10 BYTE)
VOLCAPACITY NUMBER (5,2)
WEIGHTCATEGORY VARCHAR2 (10 BYTE)
COSTPERKM NUMBER (5,2)
TRUCK_PK (TRUCKID)

The sample data is as follows (we focus only on the tables that will be used in the star schema; that is the Trip Table, Truck Table, and Destination Table):

Trip Table

TripID	Date	TotalKm	TruckID
Trip1	14-Apr-2013	370	Truck1
Trip2	14-Apr-2013	570	Truck2
Trip3	14-Apr-2013	250	Truck3
Trip4	15-Jul-2013	450	Truck1
Trip5	15-Jul-2013	175	Truck2
…	…	…	…

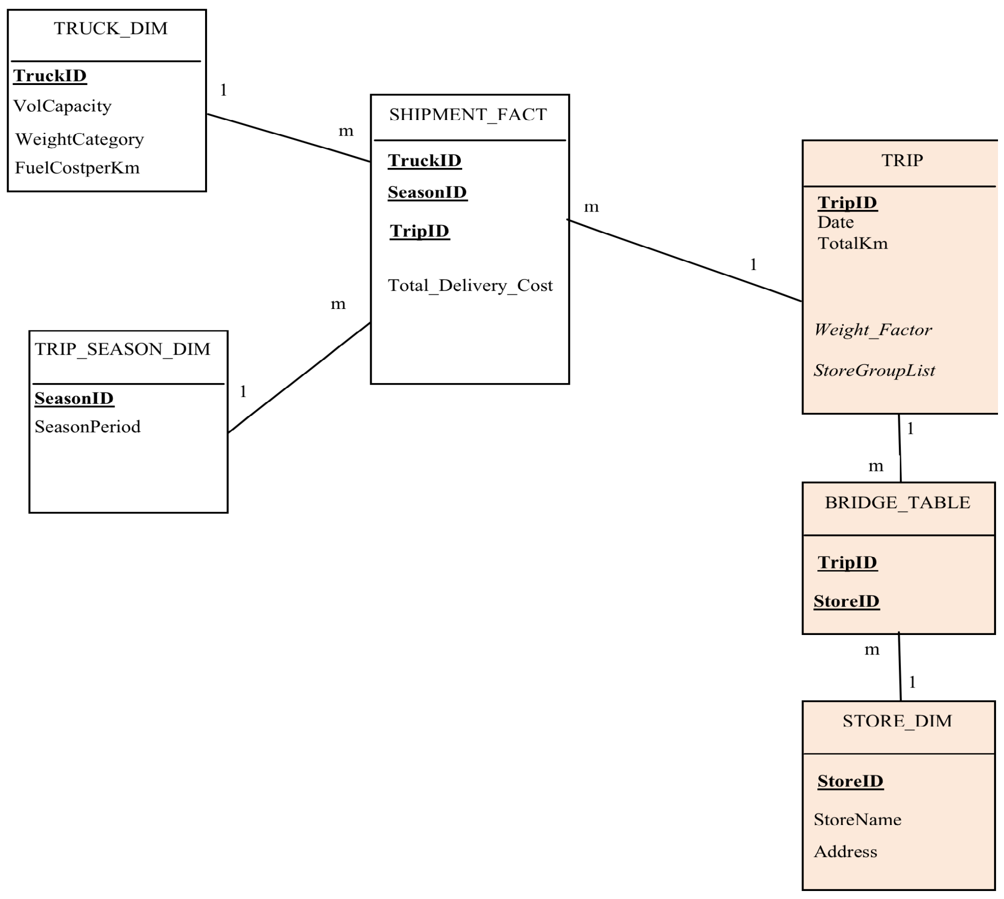
Truck Table

TruckID	VolCapacity	WeightCategory	CostPerKm
Truck1	250	Medium	$1.20
Truck2	300	Medium	$1.50
Truck3	100	Small	$0.80
Truck4	550	Large	$2.30
Truck5	650	Large	$2.50
…	…	…	…

Destination Table

TripID	**StoreID**
Trip1	M1
Trip1	M2
Trip1	M4
Trip1	M3
Trip1	M8
Trip2	M4
Trip2	M1
Trip2	M2
…	…

The star schema is shown as follows:



Note that the Trip Dimension has a Weight Factor and a Store Group List (ListAgg) attributes. Also, the Store Dimension is connected to the Trip Dimension through a Bridge Table. The three dimension tables (Trip, Bridge Table, and Store) are as follows:

Trip Dimension

TripID	Date	TotalKM	WeightFactor	StoreGroupList
Trip1	14-Apr-2013	370	0.2	M1_M2_M3_M4_M8
Trip2	14-Apr-2013	570	0.33	M1_M2_M4
Trip3	14-Apr-2013	250		
…	…	…	…	…

Bridge_Table

TripID	**StoreID**
Trip1	M1
Trip1	M2
Trip1	M4
Trip1	M3
Trip1	M8
Trip2	M4
Trip2	M1
Trip2	M2
…	…

Store Table

StoreID	StoreName	Address
M1	Myer City	Melb City
M2	Myer Chaddy	Chadstone
M3	Myer HiPoint	HighPoint
M4	Myer Donc	Doncaster
M5	Myer North	Northland
M6	Myer South	Southland
M7	Myer East	Eastland
M8	Myer Knox	Knox
…	…	…

The granularity of the Fact Table is determined by the Trip, which is at a "Day" level. Each trip will have an entry in the Fact Table. The Fact Table looks as follows:

Fact Table

TruckID	**SeasonID**	**TripID**	Total_Delivery_Cost
Truck1	Autumn	Trip1	444
Truck3	Autumn	Trip3	200
Truck2	Winter	Trip5	262.5
Truck2	Autumn	Trip2	855
Truck1	Winter	Trip4	540

In order to understand this Fact Table, let's look at the Trip Table in the operational database, which is shown as follows: There are five trips (namely Trip1 to Trip5). Trip1 to Trip3 were done in April (Autumn season), whereas Trip 4 and Trip 5 were done in July (Winter season). Because the granularity of the Fact Table is based on the Trip, the Fact Table has 5 records; the same number of records in the Trip Table.

Trip Table

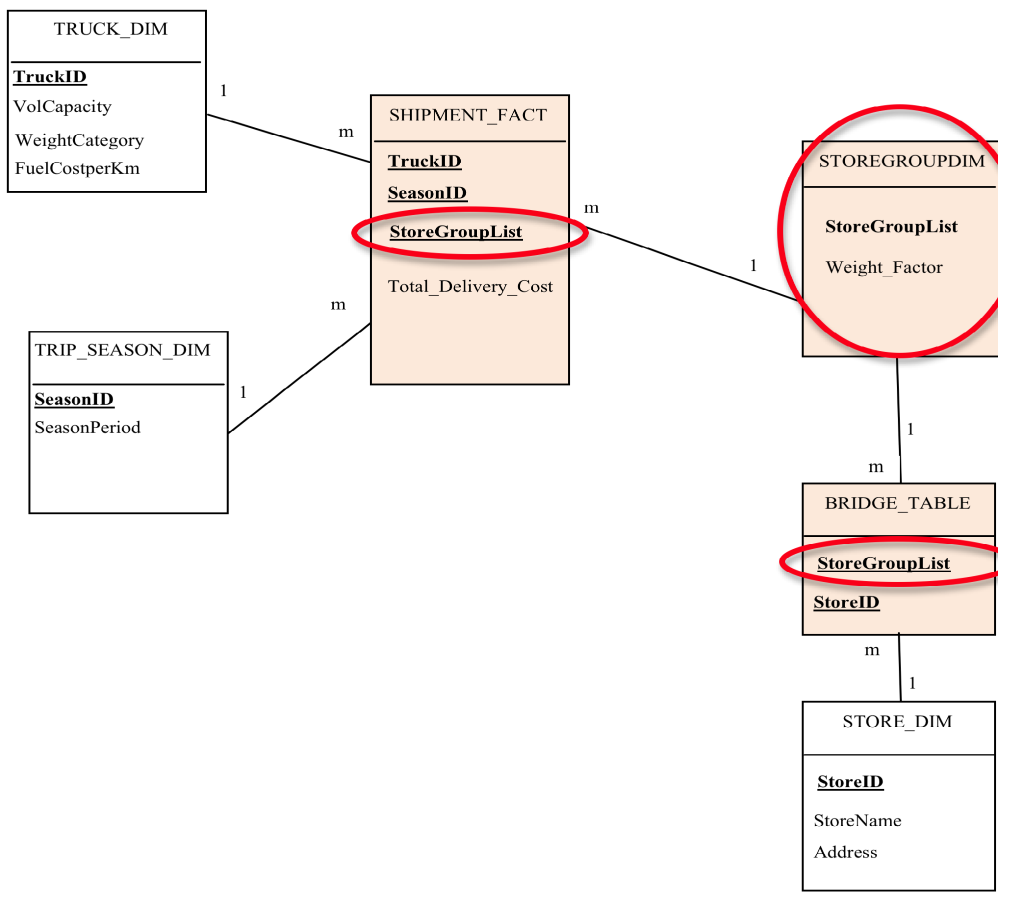
TripID	Date	TotalKm	TruckID
Trip1	14-Apr-2013	370	Truck1
Trip2	14-Apr-2013	570	Truck2
Trip3	14-Apr-2013	250	Truck3
Trip4	15-Jul-2013	450	Truck1
Trip5	15-Jul-2013	175	Truck2
…	…	…	…

If we want to increase the level of aggregation of the Truck Delivery star schema, we should not use the Trip as the level of granularity, because Trip level granularity is rather high (it means that the level of aggregation is low, because it is done at a Trip level). Increasing the level of aggregation means that some trips need to be combined, so the level of aggregation is based on "multiple trips", not based on "single trips". The question is how to combine several trips into one record?

Remember that each trip may deliver goods to many stores. For example, Trip1 delivers to 5 stores, namely M1, M2, M3, M4, and M8, as shown by the StoreGroupList attribute in the Trip Dimension table. If there are other trips which deliver to these 5 stores (by the same truck and on the same season), then we can combine these trips into one record; this record indicates the collection of stores to where these trips deliver. Therefore, instead of storing Trip1 and Tripx separately, we combine them into one record in the dimension table, and make the StoreGroupList as the identifier of the new dimension.

So, if originally we have, for example, 100 trips, after combining several trips based on their destinations, we potentially reduce the number of records to much less than 100. Consequently, the level of aggregation of the star schema increases.

The new star schema that uses StoreGroupList as the base on the level of aggregation is shown as follows:



The StoreGroupDim replaces the Trip Dimension table. This new dimension consists of only two attributes, which are StoreGroupList, and the Weight Factor. These two attributes are actually from the previous Trip Dimension table. Because the ID of the new dimension changed, the Fact table has StoreGroupList attribute (instead of the TripID attribute). Also the Bridge Table has changed its attribute.

The SQL commands to implement the above star schema are as follows:

```
--StoreGroupDim
create table StoreGroupDim as
select distinct StoreGroupList, WeightFactor
from TripDim2;

--StoreGroupBridge
create table StoreGroupBridge as
select distinct StoreGroupList, StoreID
from   TripDim2 T, Bridge_Table_Dim1 B
where  T.TripId = B.TripId;

      Drop Table TruckTempFact2;

      -- Truck Temp Fact
      Create Table TruckTempFact2 As
      Select
         tk.TruckID,
         tp.TripID,
         tp.TripDate,
         (tk.CostPerKM * tp.TotalKm) As ShipmentCost
      From Truck tk, Trip tp
      Where tk.TruckID = tp.TruckID;

      Alter Table TruckTempFact2
      Add (SeasonID Varchar2(10));

      Update TruckTempFact2
      Set SeasonID = 'Summer'
      Where to_char(TripDate, 'MM') in ('12', '01', '02');

      Update TruckTempFact2
      Set SeasonID = 'Autumn'
      Where to_char(TripDate, 'MM') in ('03', '04', '05');

      Update TruckTempFact2
      Set SeasonID = 'Winter'
      Where to_char(TripDate, 'MM') in ('06', '07', '08');

      Update TruckTempFact2
      Set SeasonID = 'Spring'
      Where to_char(TripDate, 'MM') in ('09', '10', '11');


--TempFact3
create table TruckTempFact3
as select *
from TruckTempFact2;

alter table TruckTempFact3
add (StoreGroupList varchar(100));

update TruckTempFact3 T
set StoreGroupList = (
  select listagg(D.StoreId, '_') within group
         (order by D.StoreId) as StoreGroupList
  from Destination D
  where T.TripId = D.TripId
);

--TruckFact3
create table TruckFact3
as select
  TruckId, SeasonId, StoreGroupList,
  sum(ShipmentCost) as TotalCost
from TruckTempFact3
where StoreGroupList is not null
group by TruckId, SeasonId, StoreGroupList;
```

There are several things to note from the above SQL:

1. The new dimension called **StoreGroupDim** actually extracts the two attributes from the original Trip Dimension table.
2. The new **Bridge Table** also gets its data from the Trip Dimension table and the original Bridge Table.
3. For the **TempFact3** table, it is actually identical to the previous TempFact table (TempFact2), so we can either copy from TempFact2, or rebuild a new TempFact, called TempFact3.
4. TempFact3 needs to alter to include the StoreGroupList, which is a **ListAgg attribute**.
5. Finally, the **Fact table** is basically a group by calculation from the TempFact3 table.

Summary

In this lesson, we have learned how to increase the level of aggregation of a star schema, when the original star schema has a Bridge Table.

To increase the level of aggregation, several trips sharing the same destinations (as well as the Truck and Season) are combined into one record in the new dimension. This record basically already exists in the previous Trip dimension, called the StoreGroupList, which is a ListAgg attribute. So, instead of using TripID, we use StoreGroupList as the identifier of the new dimension.

Because we combine several trips into one dimension record, we increase the level of aggregation of the star schema.