**1. Tables Mapping**

|  |  |  |
| --- | --- | --- |
| **DatawarehouseTable** | **Operational DB Table** | **Aggregation/Sum** |
| RoomDim | Corp1: RoomID | No aggregation, each row is an instance in the DW |
| RoomDim | Corp2: Room | No aggregation, each row is an instance in the DW |
| RoomTypeDim | Corp1: RoomRate | No aggregation, each row is an instance in the DW |
| RoomTypeDim | Corp2: Room\_Type | No aggregation, each row is an instance in the DW |
| RoomReservationDim | Corp1: ResRoomID | No aggregation, each row is an instance in the DW |
| RoomReservationDim | Corp2: Room\_Reservation | No aggregation, each row is an instance in the DW |
| FloorDim | Corp1: Floor table needs to be defined and populated | No aggregation, each row is an instance in the DW |
| FloorDim | Corp2: Floor | No aggregation, each row is an instance in the DW |
| WingDim | Corp1: Wings | No aggregation, each row is an instance in the DW |
| WingDim | Corp2: Wing | No aggregation, each row is an instance in the DW |
| BuildingDim | Corp1: Building | No aggregation, each row is an instance in the DW |
| BuildingDim | Corp2: Building | No aggregation, each row is an instance in the DW |
| ResortDim | Corp1: Resort | No aggregation, each row is an instance in the DW |
| ResortDim | Corp2: Resort | No aggregation, each row is an instance in the DW |
| CityDim | Corp1: Resort | No aggregation, each row is an instance in the DW |
| CityDim | Corp2: City | No aggregation, each row is an instance in the DW |
| RegionDim | Corp1: Region table needs to be defined and populated | No aggregation, each row is an instance in the DW |
| RegionDim | Corp2: Region | No aggregation, each row is an instance in the DW |
| CountryDim | Corp1: Country | No aggregation, each row is an instance in the DW |
| CountryDim | Corp2: Country | No aggregation, each row is an instance in the DW |
| EventDim | Corp1: Funtivity | No aggregation, each row is an instance in the DW |
| EventDim | Corp2: Event | No aggregation, each row is an instance in the DW |
| CustomerDim | Corp1: Person but MiddleName attribute needs to be added | No aggregation, each row is an instance in the DW |
| CustomerDim | Corp2: Customer | No aggregation, each row is an instance in the DW |
| CustomerAffiliationDim | Corp1: Funtivity-Customer | No aggregation, each row is an instance in the DW |
| CustomerAffiliationDim | Corp2: Customer\_Affiliation | No aggregation, each row is an instance in the DW |
| RoomTypeReservationsFact | Corp1: Reservation | Total number and price of stays given are created |
| RoomTypeReservationsFact | Corp2: Room\_Reservation but ResortID attribute needs to be added | Total number and price of stays given are created |
| CustomerReservationsFact | Corp1: Reservation | Total number and price of stays given are created |
| CustomerReservationsFact | Corp2: Room\_Reservation | Total number and price of stays given are created |
| EventReservationsFact | Corp1: FuntivityRes | Total number and price of stays given are created |
| EventReservationsFact | Corp2: Customer\_Affiliation | Total number and price of stays given are created |

**2. Data Extract SQL Procedures**

**a. Customer Dimension**

/\* CustomerDim is a dimension table.\*/

CREATE TABLE CustomerDim

(

C\_CustomerID\_CU *INT* NOT NULL PRIMARY KEY,

C\_Corp1CustomerID\_CU *INT*,

C\_Corp2CustomerID\_CU *INT*,

T\_CustomerFirstName\_CU *VARCHAR*(50) NOT NULL,

T\_CustomerLastName\_CU *VARCHAR*(50) NOT NULL,

T\_CustomerMiddleName\_CU *VARCHAR*(50),

T\_CustomerAddress\_CU *VARCHAR*(255),

I\_CustomerPhoneNumber\_CU *VARCHAR*(50) NOT NULL

);

CREATE OR REPLACE PROCEDURE Get\_CustomerDim\_Dimension

/\* There are two resort operational DBs to merge into the DW.

/ \*------------------------------------------------------------\*/

/\* This procedure creates the Customer dimension table for the

/\* data warehouse by extracting data from both operational databases.

/\*------------------------------------------------------------\*/

/\* Customer Dimension process.

/\* Get customers for the CustomerDim dimension table of the data warehouse from operational database one.

/\* Declare CustomerDim Table Population variables and cursors for DB1 and DB2 \*/

declare @Corp1CustomerID *int*

declare @Corp2CustomerID *int*

declare @existingCustomerID *int*

declare @FirstName *nvarchar*(50)

declare @MiddleName *nvarchar*(50)

declare @LastName *nvarchar*(50)

declare @Address *nvarchar*(255)

declare @PhoneNumber *nvarchar*(50)

declare @getCustomerDimDB1 CURSOR

declare @getCustomerDimDB2 CURSOR

/\* Define the SQL statement to query the first operational DB \*/

set @getCustomerDimDB1 = CURSOR for

Select C\_PersonID\_PR, T\_PersonFName\_PR, T\_PersonMName\_PR, T\_PersonLName\_PR, T\_StreetAddress\_PR, I\_PhoneNumber\_PR from Person

/\* Opening the cursor returns a result set based on the query \*/

open @getCustomerDimDB1;

/\* get the first record\*/

fetch next

from @getCustomerDimDB1 into @Corp1CustomerID, @FirstName, @MiddleName, @LastName, @Address, @PhoneNumber

while @@FETCH\_STATUS = 0

begin

/\* check to see if the customer is already in the table \*/

@existingCustomerID = 0

select C\_CustomerID\_CU into @existingCustomerID from CustomerDim where C\_Corp1CustomerID\_CU = @Corp1CustomerID;

If @existingCustomerID == 0

/\* customer does not exist in dimension table so insert new customer into CustomerDim table \*/

begin try

insert into CustomerDim (C\_CustomerID\_CU, C\_Corp1CustomerID\_CU, T\_CustomerFirstName\_CU,

T\_CustomerMiddleName\_CU, T\_CustomerLastName\_CU, T\_CustomerAddress\_CU, I\_CustomerPhoneNumber\_CU)

values (CustomerDimSequence, @Corp1CustomerID, @FirstName, @MiddleName, @LastName, @Address, @PhoneNumber);

end try

begin catch

print 'Error in insert for CustomerDim DB1';

end catch

else

print 'Customer customer from DB1 exists in dimension table CustomerDim, id = ' + @existingCustomerID;

end

fetch next

from @getCustomerDimDB1 into @Corp1CustomerID, @FirstName, @MiddleName, @LastName, @Address, @PhoneNumber

end

close @getCustomerDimDB1

deallocate @getCustomerDimDB1

/\* Get customers from the second operational DB\*/

/\* Define the SQL statement to query the second operational DB \*/

set @getCustomerDimDB2 = CURSOR for

Select C\_CustomerID\_Cu, T\_CustomerFirstName\_Cu, T\_CustomerMiddleName\_Cu, T\_CustomerLastName\_Cu, T\_CustomerAddress\_Cu, I\_CustomerPhoneNumber\_Cu from Customer

/\* Opening the cursor returns a result set based on the query \*/

open @getCustomerDimDB2;

/\* get the first record\*/

fetch next

from @getCustomerDimDB2 into @Corp2CustomerID, @FirstName, @MiddleName, @LastName, @Address, @PhoneNumber

while @@FETCH\_STATUS = 0

begin

/\* check to see if the customer is already in the table \*/

@existingCustomerID = 0

select C\_CustomerID\_CU into @existingCustomerID from CustomerDim where C\_Corp2CustomerID\_CU = @Corp2CustomerID;

If @existingCustomerID == 0

/\* customer does not exist in dimension table so insert new customer into CustomerDim table \*/

begin try

insert into CustomerDim (C\_CustomerID\_CU, C\_Corp2CustomerID\_CU, T\_CustomerFirstName\_CU,

T\_CustomerMiddleName\_CU, T\_CustomerLastName\_CU, T\_CustomerAddress\_CU, I\_CustomerPhoneNumber\_CU)

values (CustomerDimSequence, @Corp2CustomerID, @FirstName, @MiddleName, @LastName, @Address, @PhoneNumber);

end try

begin catch

print 'Error in insert for CustomerDim DB2';

end catch

else

print 'Customer customer from DB2 exists in dimension table CustomerDim, id = ' + @existingCustomerID;

end

fetch next

from @getCustomerDimDB2 into @Corp2CustomerID, @FirstName, @MiddleName, @LastName, @Address, @PhoneNumber

end

close @getCustomerDimDB2

deallocate @getCustomerDimDB2

DBMS\_OUTPUT.PUT\_LINE ("CustomerDim Dimension Table Loaded at: " || *sysdate*

END;

/

**b. Resort Dimension**

/\* ResortDim is a dimension table.\*/

CREATE TABLE ResortDim

(

C\_ResortID\_RS INT PRIMARY KEY,

C\_Corp1ResortID\_CU INT,

C\_Corp2ResortID\_CU INT,

T\_ResortName\_RS VARCHAR(50) NOT NULL,

T\_StreetAddress\_RS VARCHAR(255) NOT NULL,

C\_CityName\_RS VARCHAR(50) NOT NULL,

FOREIGN KEY (C\_CityName\_RS) REFERENCES CityDim(C\_CityName\_CI)

);

CREATE OR REPLACE PROCEDURE Get\_ResortDim\_Dimension

/\* There are two resort operational DBs to merge into the DW.

/ \*------------------------------------------------------------\*/

/\* This procedure creates the Resort dimension table for the

/\* data warehouse by extracting data from both operational databases.

/\*------------------------------------------------------------\*/

/\* Resort Dimension process.

/\* Get resorts for the ResortDim dimension table of the data warehouse from operational database one.

/\* Declare ResortDim Table Population variables and cursors for DB1 and DB2 \*/

declare @Corp1ResortID int

declare @Corp2ResortID int

declare @existingResortID int

declare @ResortName nvarchar(50)

declare @StreetAddress nvarchar(255)

declare @CityName nvarchar(50)

declare @getResortDimDB1 CURSOR

declare @getResortDimDB2 CURSOR

/\* Define the SQL statement to query the first operational DB \*/

set @getResortDimDB1 = CURSOR for

Select C\_ResortID\_RS, T\_ResortName\_RS, T\_StreetAddress\_CR, T\_City\_CR from Resort

/\* Opening the cursor returns a result set based on the query \*/

open @getResortDimDB1;

/\* get the first record\*/

fetch next

from @getResortDimDB1 into @Corp1ResortID, @ResortName, @StreetAddress, @CityName

while @@FETCH\_STATUS = 0

begin

/\* check to see if the resort is already in the table \*/

@existingResortID = 0

select C\_ResortID\_CU into @existingResortID from ResortDim where C\_Corp1ResortID\_CU = @Corp1ResortID;

If @existingResortID == 0

/\* resort does not exist in dimension table so insert new resort into ResortDim table \*/

begin try

insert into ResortDim (C\_ResortID\_CU, C\_Corp1ResortID\_CU, T\_ResortName\_RS, T\_StreetAddress\_RS, C\_CityName\_RS)

values (ResortDimSequence, @Corp1ResortID, @ResortName, @StreetAddress, @CityName);

end try

begin catch

print 'Error in insert for ResortDim DB1';

end catch

else

print 'Resort resort from DB1 exists in dimension table ResortDim, id = ' + @existingResortID;

end

fetch next

from @getResortDimDB1 into @Corp1ResortID, @ResortName, @StreetAddress, @CityName

end

close @getResortDimDB1

deallocate @getResortDimDB1

/\* Get resorts from the second operational DB\*/

/\* Define the SQL statement to query the second operational DB \*/

set @getResortDimDB2 = CURSOR for

Select C\_ResortCode\_Rs, T\_ResortName\_Rs, T\_StreetAddress\_Rs, C\_CityName\_Rs from Resort

/\* Opening the cursor returns a result set based on the query \*/

open @getResortDimDB2;

/\* get the first record\*/

fetch next

from @getResortDimDB2 into @Corp2ResortID, @ResortName, @StreetAddress, @CityName

while @@FETCH\_STATUS = 0

begin

/\* check to see if the resort is already in the table \*/

@existingResortID = 0

select C\_ResortID\_CU into @existingResortID from ResortDim where C\_Corp2ResortID\_CU = @Corp2ResortID;

If @existingResortID == 0

/\* resort does not exist in dimension table so insert new resort into ResortDim table \*/

begin try

insert into ResortDim (C\_ResortID\_CU, C\_Corp2ResortID\_CU, T\_ResortName\_RS, T\_StreetAddress\_RS, C\_CityName\_RS)

values (ResortDimSequence, @Corp2ResortID, @ResortName, @StreetAddress, @CityName);

end try

begin catch

print 'Error in insert for ResortDim DB2';

end catch

else

print 'Resort resort from DB2 exists in dimension table ResortDim, id = ' + @existingResortID;

end

fetch next

from @getResortDimDB2 into @Corp2ResortID, @ResortName, @StreetAddress, @CityName

end

close @getResortDimDB2

deallocate @getResortDimDB2

DBMS\_OUTPUT.PUT\_LINE ("ResortDim Dimension Table Loaded at: " || sysdate

END;

/

**c. Room Type Reservations Fact**

CREATE TABLE RoomTypeReservationsFact

(

C\_TimeID\_TR INT NOT NULL,

T\_RoomType\_TR VARCHAR(50) NOT NULL,

C\_ResortID\_TR INT NOT NULL,

N\_TotalNumOfStays\_TR DECIMAL,

N\_TotalPrice\_TR DECIMAL,

FOREIGN KEY (C\_TimeID\_TR) REFERENCES TimeDim(C\_TimeID\_TI),

FOREIGN KEY (T\_RoomType\_TR) REFERENCES RoomTypeDim(C\_RoomType\_RT),

FOREIGN KEY (C\_ResortID\_TR) REFERENCES ResortDim(C\_ResortID\_RS),

PRIMARY KEY(C\_TimeID\_TR, T\_RoomType\_TR, C\_ResortID\_TR)

);

CREATE TABLE TimeDim

(

C\_TimeID\_TI INT PRIMARY KEY,

N\_TimeYear\_TI INT NOT NULL,

N\_TimeMonth\_TI INT NOT NULL,

T\_TimeDay\_TI VARCHAR(50) NOT NULL,

);

/\* Populate RoomTypeReservationsFact Fact table from the first operational DB \*/

CREATE OR REPLACE PROCEDURE Get\_RoomTypeReservationsFact\_Fact

declare @TimeId int

declare @Month int

declare @Day int

declare @Year int

declare @OriginReservationId

declare @ResortId

declare @RoomType

declare @ReservationDate

declare @TotalPrice

declare @TotalStays

/\* Define the SQL statement to query the first operational DB \*/

set @getRoomTypeReservationsFactDB1 = CURSOR for

Select C\_ReservationID\_RN, C\_ResortID\_RN, C\_RoomRateID\_RN, T\_ReservationDate\_RN FROM Reservation

ORDER BY C\_ReservationID\_RN group by C\_ReservationID\_RN, C\_ResortID\_RN, C\_RoomRateID\_RN, T\_ReservationDate\_RN;

/\* Opening the cursor returns a result set based on the query \*/

open @getRoomTypeReservationsFactDB1;

fetch next

from @getRoomTypeReservationsFactDB1 into @OriginReservationId, @ResortId, @RoomType, @ReservationDate

while @@FETCH\_STATUS = 0

begin

/\* Extract date fields from timestamp \*/

@Month = Month(@ReservationDate)

@Day = Day(@ReservationDate)

@Year = Year(@ReservationDate)

/\* Find the time dimension row for this fact row (should have \*/

Select TimeID into @TimeID from TimeInfo where time.day = @day and time.month = @month and time.year = @year;

/\* Find the RoomReservationDim dimension joined with RoomTypeDim for this fact row to calculate TotalStays and TotalPrice\*/

Select DATEDIFF(day, D\_ArrivalDate\_RR, D\_DepartureDate\_RR) into @TotalStays,

N\_RoomPrice\_RT \* DATEDIFF(day, D\_ArrivalDate\_RR, D\_DepartureDate\_RR) into @TotalPrice

from RoomReservationDim rr JOIN RoomTypeDim rt on rr.C\_RoomType\_RR = rt.C\_RoomType\_RT

where rr.C\_RoomReservationNum\_RR = @OriginReservationId;

/\* Insert the new fact row into the RoomTypeReservationsFact table \*/

begin try

insert into RoomTypeReservationsFact (C\_TimeID\_TR, T\_RoomType\_TR, C\_ResortID\_TR, N\_TotalNumOfStays\_TR, N\_TotalPrice\_TR)

values (@TimeId, @RoomType, @ResortId, @TotalStays, @TotalPrice);

end try

begin catch

print 'Error in insert for RoomTypeReservationsFactDB1';

end catch

fetch next

from @getRoomTypeReservationsFactDB1 into @OriginReservationId, @ResortId, @RoomType, @ReservationDate

end

close @getRoomTypeReservationsFactDB1

deallocate @getRoomTypeReservationsFactDB1

/\* Define the SQL statement to query the second operational DB \*/

/\* Assumption: ResortID column/attribute has been added to Room\_Reservation table in Corp2 \*/

set @getRoomTypeReservationsFactDB2 = CURSOR for

Select C\_RoomReservationNumber\_Rr, C\_ResortID\_Rr, C\_RoomType\_Rr, D\_ArrivalDate\_Rr FROM Room\_Reservation

ORDER BY C\_RoomReservationNumber\_Rr group by C\_RoomReservationNumber\_Rr, C\_ResortID\_Rr, C\_RoomType\_Rr, D\_ArrivalDate\_Rr;

/\* Opening the cursor returns a result set based on the query \*/

open @getRoomTypeReservationsFactDB2;

fetch next

from @getRoomTypeReservationsFactDB2 into @OriginReservationId, @ResortId, @RoomType, @ReservationDate

while @@FETCH\_STATUS = 0

begin

/\* Extract date fields from timestamp \*/

@Month = Month(@ReservationDate)

@Day = Day(@ReservationDate)

@Year = Year(@ReservationDate)

/\* Find the time dimension row for this fact row (should have \*/

Select TimeID into @TimeID from TimeInfo where time.day = @day and time.month = @month and time.year = @year;

/\* Find the RoomReservationDim dimension joined with RoomTypeDim for this fact row to calculate TotalStays and TotalPrice\*/

Select DATEDIFF(day, D\_ArrivalDate\_RR, D\_DepartureDate\_RR) into @TotalStays,

N\_RoomPrice\_RT \* DATEDIFF(day, D\_ArrivalDate\_RR, D\_DepartureDate\_RR) into @TotalPrice,

from RoomReservationDim rr JOIN RoomTypeDim rt on rr.C\_RoomType\_RR = rt.C\_RoomType\_RT

where rr.C\_RoomReservationNum\_RR = @OriginReservationId;

/\* Insert the new fact row into the RoomTypeReservationsFact table \*/

begin try

insert into RoomTypeReservationsFact (C\_TimeID\_TR, T\_RoomType\_TR, C\_ResortID\_TR, N\_TotalNumOfStays\_TR, N\_TotalPrice\_TR)

values (@TimeId, @RoomType, @ResortId, @TotalStays, @TotalPrice);

end try

begin catch

print 'Error in insert for RoomTypeReservationsFactDB2';

end catch

fetch next

from @getRoomTypeReservationsFactDB2 into @OriginReservationId, @ResortId, @RoomType, @ReservationDate

end

close @getRoomTypeReservationsFactDB2

deallocate @getRoomTypeReservationsFactDB2

DBMS\_OUTPUT.PUT\_LINE ("RoomTypeReservations Fact Table Loaded at: " || sysdate

END;

/

**3. Data Load SQL Statements**