LONDON
METROPOLITAN
UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Code & Module Title

CC6001NA Advanced Database Systems Development

Assessment Weightage & Type

40% Individual coursework

Year and Semester

2018-19 Autumn

Student Name: Rajat Shrestha

London Met ID: 17030954

College ID: np01cp4a170021

Assignment Due Date: 7th January 2020

Assignment Submission Date: 7th January 2020

Word Count (Where Required): 4000

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

Abstract

This Is the first coursework of advanced database systems development year long module and details in development of an database system for an travel company.

Table of Contents

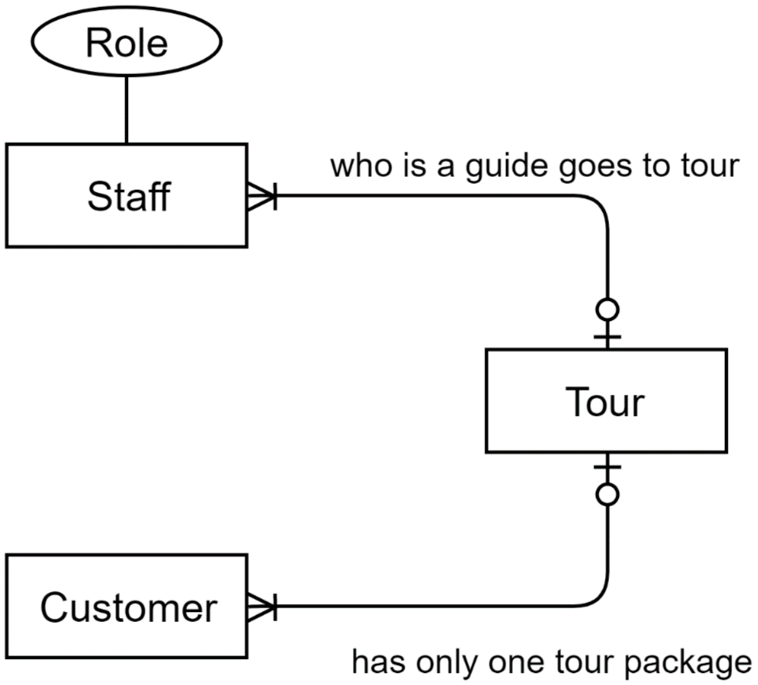
1. Introduction

The given data is analysed and then the figures are being normalized. After the normalization of the figures the tables are being generated and then again analysed on how the final ERD is created. The final ERD is created then the DDL scrips are being generated.

2. Textual Analysis

According to the scenario the tour company have the following relation between the entities:

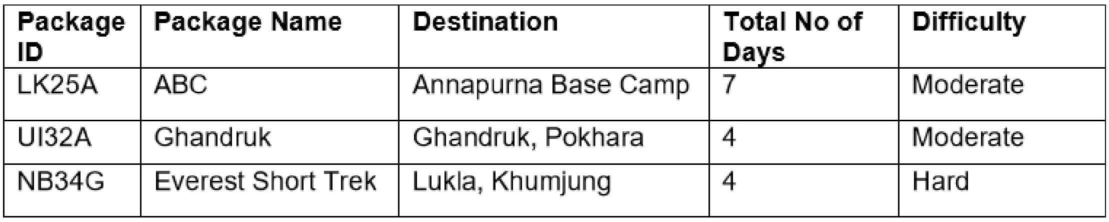
- Staffs have different roles assigned one for each
- Multiple Staffs (specifically tour guides) could be assigned for a tour
- Customer can only take one package from multiple available packages.
- The tour and package are taken as a same entity
- A registered customer might or might not have a tour assigned
- Role is taken as an attribute of staff



3. Normalization

3.1. Figure 1

The given figure gives us detail about the package record.



Package ID	Package Name	Destination	Total No of Days	Difficulty
LK25A	ABC	Annapurna Base Camp	7	Moderate
UI32A	Ghandruk	Ghandruk, Pokhara	4	Moderate
NB34G	Everest Short Trek	Lukla, Khumjung	4	Hard

As all the information is enough for a package record, storing the given records as atomic value(i.e. storing each value in unique cells) we get:

Package ID	Package Name	Destination	Days	Difficulty
LK25A	ABC	Annapurna Base Camp	7	Moderate
UI32A	Ghandruk	Ghandruk	4	Moderate
UI32A	Ghandruk	Pokhara	4	Moderate
NB34G	Everest Short Trek	Lukla	4	Hard
NB34G	Everest Short Trek	Khumjung	4	Hard

3.1.1. Un-normalized Form (UNF):

From the above table we can observe that the destination column has repeating groups in it, representing the given data in UNF:

UNF: (**<u>Package_ID</u>**, Pakage_Name, {Destination}, Total_Days, Difficulty)

Destination is the repeating group because a package might have multiple destinations for example Pokhara and Ghandruk. Package_ID is taken as the primary key as it can uniquely identify all columns.

3.1.2. First Normal Form (1NF):

As Package_Name, Total_Days, and Difficulty depends upon the Package_ID, it is assigned as a primary key. Package_Name also gives Destination

Package_ID ➔ Pakage_Name, Total_Days, Difficulty

Package_ID ➔ Destination

Removing the repeating group and creating the Destination Package and assigning package_ID and Destination_ID as composite key and creating the first normal form.

Package (**<u>Package_ID</u>**, Pakage_Name, Total_Days, Difficulty)

Destination_Package (**<u>Package_ID*, Destination</u>**)

3.1.3. Second Normal Form (2NF):

There are no Partial dependencies as all the non key elements are directly dependent on their respective primary key

Package_ID ➔ Pakage_Name, Total_Days, Difficulty

Package_ID and Destination gives the unique value for every package-destination pair:

Package (**Package_ID**, Pakage_Name, Total_Days, Difficulty)

Destination_Package (**Package_ID***, **Destination**)

3.1.4. Third Normal Form (3NF):

There are no transitive dependencies as every none key entries are dependent on the primary key assigned for each table.

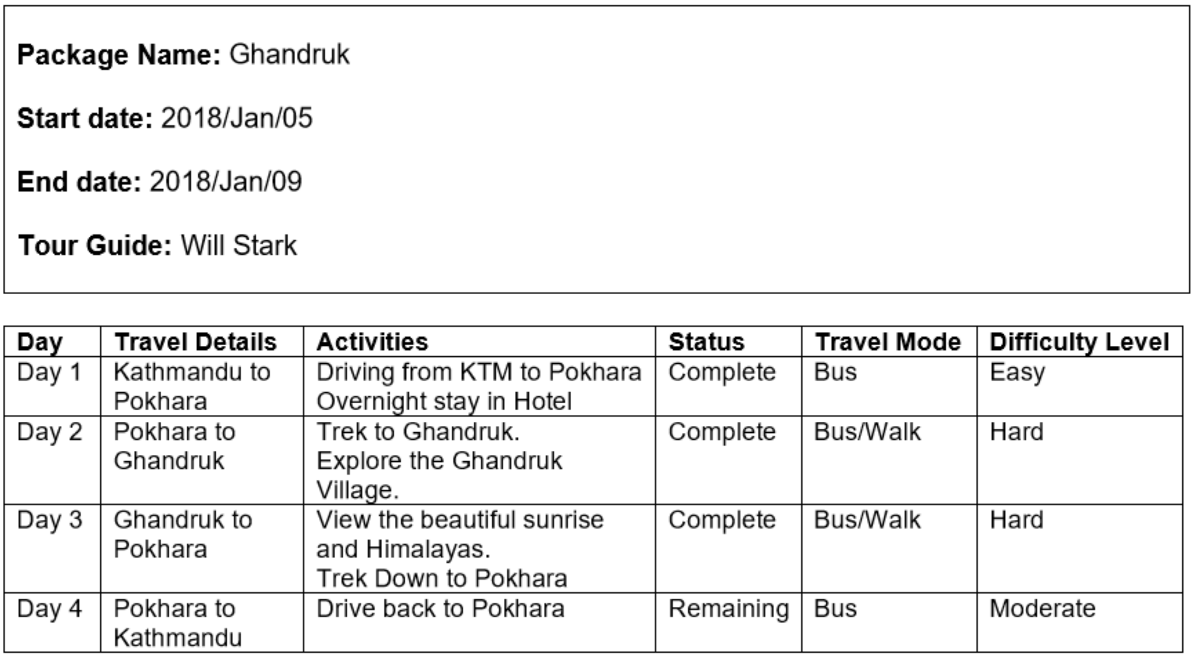
Package (**Package_ID**, Pakage_Name, Total_Days, Difficulty)

Destination_Package (**Package_ID***, **Destination**)

Since there are no transative dependencies this is the final normalized form

3.2. Figure 2

The second figure gives us detail about the tracking information.



Package Name: Ghandruk

Start date: 2018/Jan/05

End date: 2018/Jan/09

Tour Guide: Will Stark

Day	Travel Details	Activities	Status	Travel Mode	Difficulty Level
Day 1	Kathmandu to Pokhara	Driving from KTM to Pokhara Overnight stay in Hotel	Complete	Bus	Easy
Day 2	Pokhara to Ghandruk	Trek to Ghandruk. Explore the Ghandruk Village.	Complete	Bus/Walk	Hard
Day 3	Ghandruk to Pokhara	View the beautiful sunrise and Himalayas. Trek Down to Pokhara	Complete	Bus/Walk	Hard
Day 4	Pokhara to Kathmandu	Drive back to Pokhara	Remaining	Bus	Moderate

3.2.1. Assumptions:

- Activity and Travel mode depends on the Activity ID assigned for dividing activities
- A day in package will determine travel details, difficulty and unique sets of activities consisting of activity, travel mode and also the status of the day.
- Package ID details the tracking information of the tour.

Storing the given records in unique cells we get the following table:

Package ID	Package Name	Start Date	End Date	Guide	Day	Travel Details	Difficulty	Activity ID	Activities	Travel Mode	Status
UI32A	Ghandruk	1st Jan 2019	7th Jan 2019	Will Stark	Day 1	Kathmandu to Pokhara	Easy	A1	Driving from KTM to Pokhara.	Bus	Complete
UI32A	Ghandruk	1st Jan 2019	7th Jan 2019	Will Stark	Day 1	Kathmandu to Pokhara	Easy	A2	Overnight stay in Hotel.	Stay	Complete
UI32A	Ghandruk	1st Jan 2019	7th Jan 2019	Will Stark	Day 2	Pokhara to Ghandruk	Hard	A3	Trek to Ghandruk.	Walk	Complete
UI32A	Ghandruk	1st Jan 2019	7th Jan 2019	Will Stark	Day 2	Pokhara to Ghandruk	Hard	A4	Explore the Ghandruk Village.	Walk	Complete
UI32A	Ghandruk	1st Jan 2019	7th Jan 2019	Will Stark	Day 3	Ghandruk to Pokhara	Easy	A5	View the sunrise and Himalayas.	Walk	Complete
UI32A	Ghandruk	1st Jan 2019	7th Jan 2019	Will Stark	Day 3	Ghandruk to Pokhara	Easy	A6	Trek Down to Pokhara	Walk	Complete
UI32A	Ghandruk	1st Jan 2019	7th Jan 2019	Will Stark	Day 4	Pokhara to Kathmandu	Moderate	A7	Drive back to Kathmandu	Bus	Remaining

3.2.2. Un-normalized Form (UNF):

From the above table we can observe that the destination column has repeating groups in it, representing the given data in UNF:

UNF: (**<u>Package_ID</u>**, Package_Name, Start_Date, End_Date, Tour_Guide, {Day, Travel_Details, Difficulty_Level, {Activity_ID, Activities, , Travel_Mode }, Status})

3.2.3. First Normal Form (1NF):

Package ID is taken as the Primary key as it gives

Package_ID ➔ Package_Name, Start_Date, End_Date, Tour_Guide

Package_ID, Day ➔ Travel_Details, Status, Difficulty_Level

Package_ID, Day, Activity_ID ➔ Activities, Travel_Mode

Removing the repeating group and creating the Itenary_Tour. Assigning package_ID as Primary key and creating the first normal form.

Tour (**<u>Package_ID</u>**, Package_Name, Start_Date, End_Date, Tour_Guide)

Itenary_Tour (**<u>Package_ID</u>***, **<u>Day</u>**, Travel_Details, Difficulty_Level, Status)

Day_Activities (**<u>Package_ID</u>***, **<u>Day*</u>**, **<u>Activity_ID</u>**, Activities, Travel_Mode)

3.2.4. Second Normal Form (2NF):

Here only Activity and Travel mode are in partial dependent because it is not related to the Package ID and Package ID but directly dependent to the Activity ID.

Package_ID ➔ Package_Name, Start_Date, End_Date, Tour_Guide

Package_ID, Day ➔ Travel_Details, Status, Difficulty_Level

Package_ID, Day, Activity_ID ➔ Activities, Travel_Mode

Activity_ID ➔ Activities, Travel_Mode

Removing the partial dependency, we get:

Tour (**Package_ID**, Package_Name, Start_Date, End_Date, Tour_Guide)

Itenary_Tour (**Package_ID***, **Day**, Travel_Details, Difficulty_Level, Status)

Day_Activities (**Package_ID***, **Day***, **Activity_ID**)

Activities(**Activity_ID**, Activities, Travel_Mode)

3.2.5. Third Normal Form (3NF):

There are no transitive dependencies as every none key entries are dependent on the primary key assigned for each table.

Tour (**Package_ID**, Package_Name, Start_Date, End_Date, Tour_Guide)

Itenary_Tour (**Package_ID***, **Day**, Travel_Details, Difficulty_Level, Status)

Day_Activities (**Package_ID***, **Day***, **Activity_ID***)

Activities(**Activity_ID**, Activities, Travel_Mode)

Since there are no transative dependencies this is the final normalized form

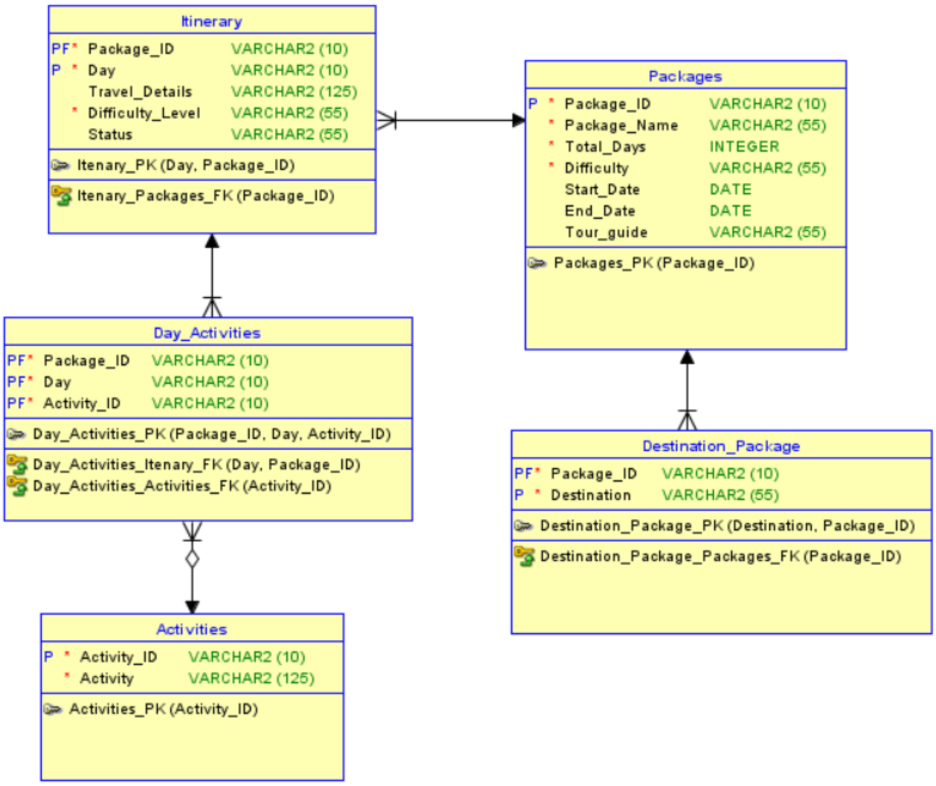
4. Integration

Now, we have all the tables from the figure 1 and 2 we can observe that the package table from figure 1 and tour table from figure 2 gives the same information. Now for integration the created tables into one and adding new tables for clarity we need to lay down some assumptions.

4.1. Assumptions:

- Package from figure(1) and tour from figure(2) represent the same table.
- Difficulty in package table gives the difficulty in the overall package
- The difficulty in Itinerary gives the difficulty level in each day of the package.

After combining the normalized tables from figure 1 and 2 we get the following relational diagram:


Itinerary
PF* Package_ID VARCHAR2 (10)
P * Day VARCHAR2 (10)
Travel_Details VARCHAR2 (125)
* Difficulty_Level VARCHAR2 (55)
Status VARCHAR2 (55)
Itenary_PK (Day, Package_ID)
Itenary_Packages_FK (Package_ID)
Packages
P * Package_ID VARCHAR2 (10)
* Package_Name VARCHAR2 (55)
* Total_Days INTEGER
* Difficulty VARCHAR2 (55)
Start_Date DATE
End_Date DATE
Tour_guide VARCHAR2 (55)
Packages_PK (Package_ID)
Day_Activities
PF* Package_ID VARCHAR2 (10)
PF* Day VARCHAR2 (10)
PF* Activity_ID VARCHAR2 (10)
Day_Activities_PK (Package_ID, Day, Activity_ID)
Day_Activities_Itenary_FK (Day, Package_ID)
Day_Activities_Activities_FK (Activity_ID)
Destination_Package
PF* Package_ID VARCHAR2 (10)
P * Destination VARCHAR2 (55)
Destination_Package_PK (Destination, Package_ID)
Destination_Package_Packages_FK (Package_ID)
Activities
P * Activity_ID VARCHAR2 (10)
* Activity VARCHAR2 (125)
Activities_PK (Activity_ID)

The DDL generated is as follows:

```
CREATE TABLE Activity
  (
    Activity_ID   VARCHAR2 (10) NOT NULL ,
    Activity_Name VARCHAR2 (125)
  ) ;
ALTER TABLE Activity ADD CONSTRAINT Activity_PK PRIMARY KEY ( Activity_ID ) ;
CREATE TABLE Day_Activities
  (
    Package_ID  VARCHAR2 (10) NOT NULL ,
    DAY         VARCHAR2 (10) NOT NULL ,
    Activity_ID VARCHAR2 (10) NOT NULL
  ) ;
ALTER TABLE Day_Activities ADD CONSTRAINT Day_Activities_PK PRIMARY KEY ( Activity_ID, P
ackage_ID, DAY ) ;
CREATE TABLE Destination_Package
  (
    Package_ID  VARCHAR2 (10) NOT NULL ,
    Destination VARCHAR2 (55) NOT NULL
  ) ;
ALTER TABLE Destination_Package ADD CONSTRAINT Destination_Package_PK PRIMARY KEY ( Dest
ination, Package_ID ) ;
CREATE TABLE Itinerary
  (
    Package_ID       VARCHAR2 (10) NOT NULL ,
    DAY              VARCHAR2 (10) NOT NULL ,
    Travel_Details   VARCHAR2 (125) ,
    Difficulty_Level VARCHAR2 (55) NOT NULL ,
    Status           VARCHAR2 (55)
  ) ;
ALTER TABLE Itinerary ADD CONSTRAINT Itenary_PK PRIMARY KEY ( DAY, Package_ID ) ;
CREATE TABLE Packages
  (
    Package_ID   VARCHAR2 (10) NOT NULL ,
    Package_Name VARCHAR2 (55) NOT NULL ,
    Total_Days   INTEGER NOT NULL ,
    Difficulty   VARCHAR2 (55) NOT NULL ,
    Start_Date   DATE ,
    End_Date     DATE ,
    Tour_guide   VARCHAR2 (55)
  ) ;
ALTER TABLE Packages ADD CONSTRAINT Packages_PK PRIMARY KEY ( Package_ID ) ;
ALTER TABLE Day_Activities ADD CONSTRAINT Day_Activities_Activity_FK FOREIGN KEY ( Activ
ity_ID ) REFERENCES Activity ( Activity_ID ) ;
ALTER TABLE Day_Activities ADD CONSTRAINT Day_Activities_Itinerary_FK FOREIGN KEY ( DAY,
 Package_ID ) REFERENCES Itinerary ( DAY, Package_ID ) ;
ALTER TABLE Destination_Package ADD CONSTRAINT Destination_Package_Packages_FK FOREIGN K
EY ( Package_ID ) REFERENCES Packages ( Package_ID ) ;
ALTER TABLE Itinerary ADD CONSTRAINT Itenary_Packages_FK FOREIGN KEY ( Package_ID ) REFE
RENCES Packages ( Package_ID ) ;
```

4.2. Addition of entities

According to the initial ER diagram, there are other entities than just tour or package. This database still lacks the presence of the staff and customer entities for being sensible. So, for the addition of these entities, the following assumptions were made:

4.2.1. Addition of Staffs

The following assumptions were made while adding the staffs:

- Staffs have Name (first name and last name), address and phone number.
- Staffs are identified by Staff ID.
- Guides are also staff and have guide ID which is referenced to the staff ID in a tour.
- Each staff have only one role assigned to them by the company
- The guide from the package is removed as a package might have multiple guides.

UNF = (Staff_ID, {Package_ID*}, First_name, Last_Name, Address, Phone_No, Role, Description)

Since Package ID is the repeating group the 1NF will be

Staff (**<u>Staff_ID</u>**, First_name, Last_Name, Address, Phone_No, Role, Description)

Guide_Package (**<u>Staff_ID*</u>**, **<u>Package_ID*</u>**)

Since there are no partial functional dependencies but there is one transitive dependencies so the 3nf will be:

Staff (**<u>Staff_ID</u>**, First_name, Last_Name, Address, Phone_No, Role*)

Role (**<u>Role</u>**, Description)

Guide_Package (**<u>Staff_ID*</u>**, **<u>Package_ID*</u>**)

4.2.2. Addition of Customers

The following assumptions were made while adding the customers:

- Customers have Name (first name and last name), address and phone number.
- Staffs are identified by Staff ID
- Customers can only book one package only once at a time

Now from the following assumptions normalizing the customer tables

UNF = (Customer_ID, {Package_ID*}, First_name, Last_Name, Address, Phone_No)

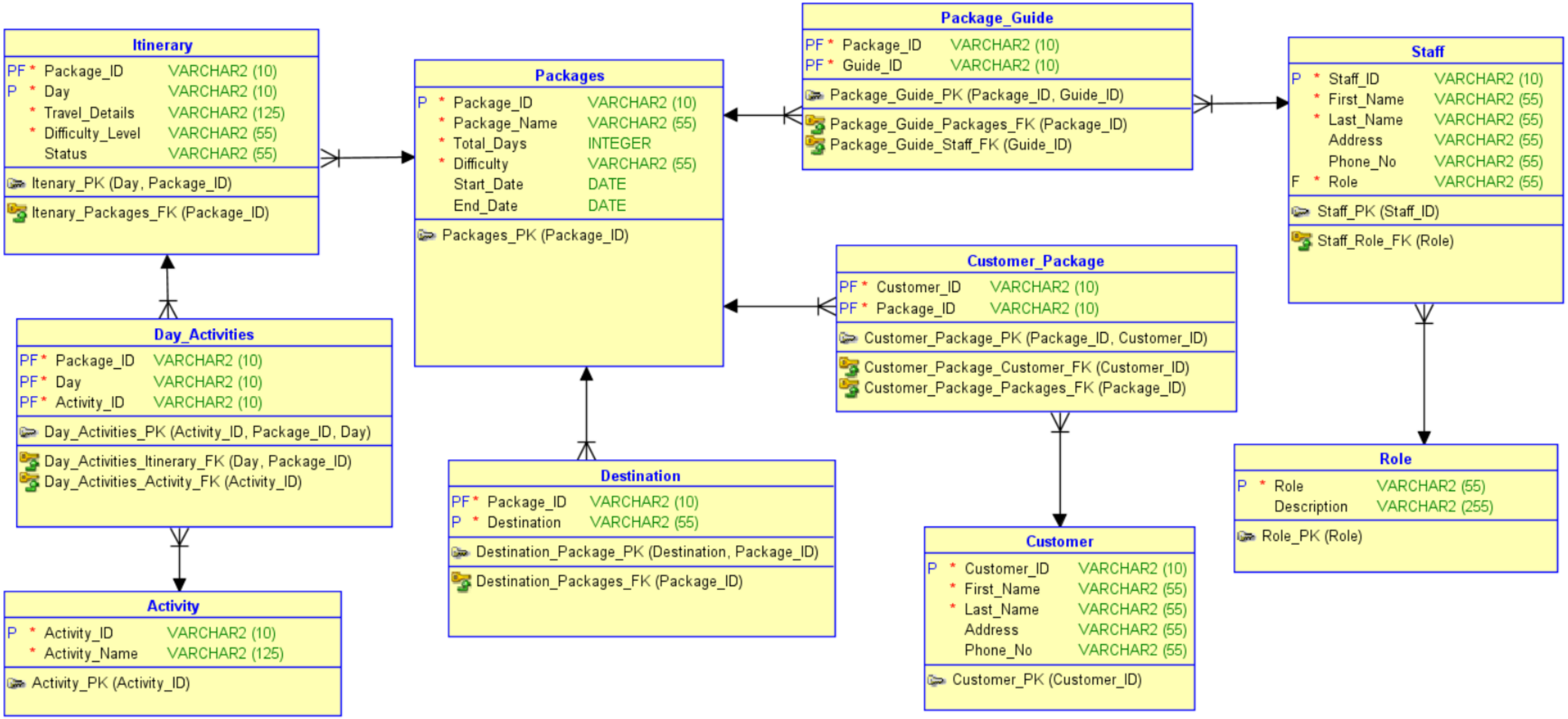
Since Package ID is the repeating group the 1NF will be

Customer (**Customer_ID**, First_name, Last_Name, Address, Phone_No)

Customer_package (**Customer_ID***, **Package_ID***)

Since there are no partial functional dependencies or transitive dependencies this is the final Normalized form

4.3.2. Final ER-diagram:



Itinerary

PF	*	Package_ID	VARCHAR2 (10)
P	*	Day	VARCHAR2 (10)
	*	Travel_Details	VARCHAR2 (125)
	*	Difficulty_Level	VARCHAR2 (55)
		Status	VARCHAR2 (55)

Itenary_PK (Day, Package_ID)

Itenary_Packages_FK (Package_ID)

Packages

P	*	Package_ID	VARCHAR2 (10)
	*	Package_Name	VARCHAR2 (55)
	*	Total_Days	INTEGER
	*	Difficulty	VARCHAR2 (55)
		Start_Date	DATE
		End_Date	DATE

Packages_PK (Package_ID)

Package_Guide

PF	*	Package_ID	VARCHAR2 (10)
PF	*	Guide_ID	VARCHAR2 (10)

Package_Guide_PK (Package_ID, Guide_ID)

Package_Guide_Packages_FK (Package_ID)
Package_Guide_Staff_FK (Guide_ID)

Staff

P	*	Staff_ID	VARCHAR2 (10)
	*	First_Name	VARCHAR2 (55)
	*	Last_Name	VARCHAR2 (55)
		Address	VARCHAR2 (55)
		Phone_No	VARCHAR2 (55)
F	*	Role	VARCHAR2 (55)

Staff_PK (Staff_ID)

Staff_Role_FK (Role)

Customer_Package

PF	*	Customer_ID	VARCHAR2 (10)
PF	*	Package_ID	VARCHAR2 (10)

Customer_Package_PK (Package_ID, Customer_ID)

Customer_Package_Customer_FK (Customer_ID)
Customer_Package_Packages_FK (Package_ID)

Day_Activities

PF	*	Package_ID	VARCHAR2 (10)
PF	*	Day	VARCHAR2 (10)
PF	*	Activity_ID	VARCHAR2 (10)

Day_Activities_PK (Activity_ID, Package_ID, Day)

Day_Activities_Itinerary_FK (Day, Package_ID)
Day_Activities_Activity_FK (Activity_ID)

Destination

PF	*	Package_ID	VARCHAR2 (10)
P	*	Destination	VARCHAR2 (55)

Destination_Package_PK (Destination, Package_ID)

Destination_Packages_FK (Package_ID)

Role

P	*	Role	VARCHAR2 (55)
		Description	VARCHAR2 (255)

Role_PK (Role)

Customer

P	*	Customer_ID	VARCHAR2 (10)
	*	First_Name	VARCHAR2 (55)
	*	Last_Name	VARCHAR2 (55)
		Address	VARCHAR2 (55)
		Phone_No	VARCHAR2 (55)

Customer_PK (Customer_ID)

Activity

P	*	Activity_ID	VARCHAR2 (10)
	*	Activity_Name	VARCHAR2 (125)

Activity_PK (Activity_ID)

4.3.3. Final DDL Script:

```
-- Generated by Oracle SQL Developer Data Modeler 4.1.5.907
--   at:        2020-01-04 21:17:27 NPT
--   site:      Oracle Database 11g
--   type:      Oracle Database 11g




CREATE TABLE Activity
  (
    Activity_ID   VARCHAR2 (10) NOT NULL ,
    Activity_Name VARCHAR2 (125) NOT NULL
  ) ;
ALTER TABLE Activity ADD CONSTRAINT Activity_PK PRIMARY KEY ( Activity_ID ) ;


CREATE TABLE Customer
  (
    Customer_ID VARCHAR2 (10) NOT NULL ,
    First_Name  VARCHAR2 (55) NOT NULL ,
    Last_Name   VARCHAR2 (55) NOT NULL ,
    Address     VARCHAR2 (55) ,
    Phone_No    VARCHAR2 (55)
  ) ;
ALTER TABLE Customer ADD CONSTRAINT Customer_PK PRIMARY KEY ( Customer_ID ) ;


CREATE TABLE Customer_Package
  (
    Customer_ID VARCHAR2 (10) NOT NULL ,
    Package_ID  VARCHAR2 (10) NOT NULL
  ) ;
ALTER TABLE Customer_Package ADD CONSTRAINT Customer_Package_PK PRIMARY KEY ( Package_ID, Customer_ID ) ;


CREATE TABLE Day_Activities
  (
    Package_ID  VARCHAR2 (10) NOT NULL ,
    DAY         VARCHAR2 (10) NOT NULL ,
    Activity_ID VARCHAR2 (10) NOT NULL
  ) ;
ALTER TABLE Day_Activities ADD CONSTRAINT Day_Activities_PK PRIMARY KEY ( Activity_ID, Package_ID, DAY ) ;
```

```
CREATE TABLE Destination
  (
    Package_ID  VARCHAR2 (10) NOT NULL ,
    Destination VARCHAR2 (55) NOT NULL
  ) ;
ALTER TABLE Destination ADD CONSTRAINT Destination_Package_PK PRIMARY KEY ( Destination, Package_ID ) ;


CREATE TABLE Itinerary
  (
    Package_ID       VARCHAR2 (10) NOT NULL ,
    DAY              VARCHAR2 (10) NOT NULL ,
    Travel_Details   VARCHAR2 (125) NOT NULL ,
    Difficulty_Level VARCHAR2 (55) NOT NULL ,
    Status           VARCHAR2 (55)
  ) ;
ALTER TABLE Itinerary ADD CONSTRAINT Itenary_PK PRIMARY KEY ( DAY, Package_ID ) ;


CREATE TABLE Package_Guide
  (
    Package_ID VARCHAR2 (10) NOT NULL ,
    Guide_ID   VARCHAR2 (10) NOT NULL
  ) ;
ALTER TABLE Package_Guide ADD CONSTRAINT Package_Guide_PK PRIMARY KEY ( Package_ID, Guide_ID ) ;


CREATE TABLE Packages
  (
    Package_ID   VARCHAR2 (10) NOT NULL ,
    Package_Name VARCHAR2 (55) NOT NULL ,
    Total_Days   INTEGER NOT NULL ,
    Difficulty   VARCHAR2 (55) NOT NULL ,
    Start_Date   DATE ,
    End_Date     DATE
  ) ;
ALTER TABLE Packages ADD CONSTRAINT Packages_PK PRIMARY KEY ( Package_ID ) ;


CREATE TABLE Role
  (
    Role        VARCHAR2 (55) NOT NULL ,
    Description VARCHAR2 (255)
  ) ;
ALTER TABLE Role ADD CONSTRAINT Role_PK PRIMARY KEY ( Role ) ;
```

```
CREATE TABLE Staff
  (
    Staff_ID   VARCHAR2 (10) NOT NULL ,
    First_Name VARCHAR2 (55) NOT NULL ,
    Last_Name  VARCHAR2 (55) NOT NULL ,
    Address    VARCHAR2 (55) ,
    Phone_No   VARCHAR2 (55) ,
    Role       VARCHAR2 (55) NOT NULL
  ) ;
ALTER TABLE Staff ADD CONSTRAINT Staff_PK PRIMARY KEY ( Staff_ID ) ;


ALTER TABLE Customer_Package ADD CONSTRAINT Customer_Package_Customer_FK FOREIGN KEY
 ( Customer_ID ) REFERENCES Customer ( Customer_ID ) ;

ALTER TABLE Customer_Package ADD CONSTRAINT Customer_Package_Packages_FK FOREIGN KEY
 ( Package_ID ) REFERENCES Packages ( Package_ID ) ;

ALTER TABLE Day_Activities ADD CONSTRAINT Day_Activities_Activity_FK FOREIGN KEY ( A
ctivity_ID ) REFERENCES Activity ( Activity_ID ) ;

ALTER TABLE Day_Activities ADD CONSTRAINT Day_Activities_Itinerary_FK FOREIGN KEY (
DAY, Package_ID ) REFERENCES Itinerary ( DAY, Package_ID ) ;

ALTER TABLE Destination ADD CONSTRAINT Destination_Packages_FK FOREIGN KEY ( Package
_ID ) REFERENCES Packages ( Package_ID ) ;

ALTER TABLE Itinerary ADD CONSTRAINT Itenary_Packages_FK FOREIGN KEY ( Package_ID )
REFERENCES Packages ( Package_ID ) ;

ALTER TABLE Package_Guide ADD CONSTRAINT Package_Guide_Packages_FK FOREIGN KEY ( Pac
kage_ID ) REFERENCES Packages ( Package_ID ) ;

ALTER TABLE Package_Guide ADD CONSTRAINT Package_Guide_Staff_FK FOREIGN KEY ( Guide_
ID ) REFERENCES Staff ( Staff_ID ) ;

ALTER TABLE Staff ADD CONSTRAINT Staff_Role_FK FOREIGN KEY ( Role ) REFERENCES Role
( Role ) ;


-- Oracle SQL Developer Data Modeler Summary Report:
--
-- CREATE TABLE                            10
-- CREATE INDEX                             0
-- ALTER TABLE                             19
-- CREATE VIEW                              0
-- ALTER VIEW                               0
-- CREATE PACKAGE                           0
-- CREATE PACKAGE BODY                      0
-- CREATE PROCEDURE                         0
-- CREATE FUNCTION                          0
-- CREATE TRIGGER                           0
-- ALTER TRIGGER                            0
-- CREATE COLLECTION TYPE                   0
-- CREATE STRUCTURED TYPE                   0
-- CREATE STRUCTURED TYPE BODY              0
-- CREATE CLUSTER                           0
-- CREATE CONTEXT                           0
-- CREATE DATABASE                          0
-- CREATE DIMENSION                         0
-- CREATE DIRECTORY                         0
-- CREATE DISK GROUP                        0
-- CREATE ROLE                              0
-- CREATE ROLLBACK SEGMENT                  0
-- CREATE SEQUENCE                          0
-- CREATE MATERIALIZED VIEW                 0
-- CREATE SYNONYM                           0
-- CREATE TABLESPACE                        0
-- CREATE USER                              0
--
-- DROP TABLESPACE                          0
-- DROP DATABASE                            0
--
-- REDACTION POLICY                         0
--
-- ORDS DROP SCHEMA                         0
-- ORDS ENABLE SCHEMA                       0
-- ORDS ENABLE OBJECT                       0
--
-- ERRORS                                   0
-- WARNINGS                                 0
```