**Chapter 2 Practical Exercise**

**Table that would be in form 1 but not 2:**

**RESERVATION:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **RESERVATION\_ID** | **TRIP\_ID** | **TRIP\_DATE** | **CUSTOMER\_NUM** | **LAST\_NAME** | **FIRST\_NAME** |
| 1600001 | 40 | 3/26/2016 | 101 | Northfold | Liam |
| 1600002 | 21 | 6/8/2016 | 101 | Northfold | Liam |
| 1600003 | 28 | 9/12/2016 | 103 | Kasuma | Sujata |
| 1600004 | 26 | 10/16/2016 | 104 | Goff | Ryan |
| 1600005 | 39 | 6/25/2016 | 105 | McLean | Kyle |
| 1600006 | 32 | 6/18/2016 | 106 | Morontoia | Joseph |
| 1600007 | 22 | 7/9/2016 | 107 | Marchand | Quinn |

If we had a table that was a combination of the current RESERVATION & CUSTOMER tables, it could look something like this. The issue is that if a reservation is cancelled and say, deleted, its possible for rows that hold customer information to be deleted as well. Breaking these two tables into separate tables with their own primary keys would resolve the dependency issue. A solution could look like this:

**RESERVATION:**

|  |  |  |  |
| --- | --- | --- | --- |
| **RESERVATION\_ID** | **TRIP\_ID** | **TRIP\_DATE** | **CUSTOMER\_NUM** |
| 1600001 | 40 | 3/26/2016 | 101 |
| 1600002 | 21 | 6/8/2016 | 101 |
| 1600003 | 28 | 9/12/2016 | 103 |
| 1600004 | 26 | 10/16/2016 | 104 |
| 1600005 | 39 | 6/25/2016 | 105 |
| 1600006 | 32 | 6/18/2016 | 106 |
| 1600007 | 22 | 7/9/2016 | 107 |

**CUSTOMER:**

|  |  |  |
| --- | --- | --- |
| **CUSTOMER\_NUM** | **LAST\_NAME** | **FIRST\_NAME** |
| 101 | Northfold | Liam |
| 103 | Kasuma | Sujata |
| 104 | Goff | Ryan |
| 105 | McLean | Kyle |
| 106 | Morontoia | Joseph |
| 107 | Marchand | Quinn |

The dataset we have is already addressing this problem, but in the scenario I presented this would work as a fix.

**Table that would be in form 2 but not 3:**

**TRIP:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **TRIP\_ID** | **TRIP\_NAME** | **TYPE** | **GUIDE\_NUM** | **LAST\_NAME** | **FIRST\_NAME** |
| 2 | Mt Ascutney - North Peak | Hiking | AM01 | Abrams | Miles |
| 21 | Long Pond | Hiking | AM01 | Abrams | Miles |
| 30 | McLennan Reservation Hike | Hiking | AM01 | Abrams | Miles |
| 4 | Bradbury Mountain Ride | Biking | BR01 | Boyers | Rita |
| 8 | Black Pond | Hiking | BR01 | Boyers | Rita |
| 9 | Big Rock Cave | Hiking | BR01 | Boyers | Rita |
| 12 | Cadillac Mountain Ride | Biking | BR01 | Boyers | Rita |
| 24 | Mt Adams | Hiking | BR01 | Boyers | Rita |

Assuming we had a table like the one above, we could quickly convert this to form 3 by separating the guide information into its own table, to remove the guide dependencies. In this case names are dependant.

**TRIP:**

|  |  |  |  |
| --- | --- | --- | --- |
| **TRIP\_ID** | **TRIP\_NAME** | **TYPE** | **GUIDE\_NUM** |
| 2 | Mt Ascutney - North Peak | Hiking | AM01 |
| 21 | Long Pond | Hiking | AM01 |
| 30 | McLennan Reservation Hike | Hiking | AM01 |
| 4 | Bradbury Mountain Ride | Biking | BR01 |
| 8 | Black Pond | Hiking | BR01 |
| 9 | Big Rock Cave | Hiking | BR01 |
| 12 | Cadillac Mountain Ride | Biking | BR01 |
| 24 | Mt Adams | Hiking | BR01 |

**GUIDE:**

|  |  |  |
| --- | --- | --- |
| **GUIDE\_NUM** | **LAST\_NAME** | **FIRST\_NAME** |
| AM01 | Abrams | Miles |
| BR01 | Boyers | Rita |

1. The functional dependency in the example is the guide information. To remove the dependencies, you could separate the guide names into a separate table. By chance this is similar to my example above:

TRIP (TRIP\_ID, TRIP\_NAME, STATE\_ABBREVIATION, STATE\_NAME, GUIDE\_NUM)

GUIDE (GUIDE\_NUM, GUIDE\_LAST, GUIDE\_FIRST)

Also, even though it’s unlikely that a state name will change, you could go as far as separating the state info too:

TRIP (TRIP\_ID, TRIP\_NAME, STATE\_ABBREVIATION, GUIDE\_NUM)

GUIDE (GUIDE\_NUM, GUIDE\_LAST, GUIDE\_FIRST)

STATE (STATE\_ABBREVIATION, STATE\_NAME)

1. 1. CREATE PARTICIPANT (PARTICIPANT\_NUM, LAST\_NAME, FIRST\_NAME, ADDRESS, CITY, STATE, POSTAL\_CODE, PHONE, BIRTHDATE)
   2. CREATE CLASS (CLASS\_NUM, CLASS\_DESC, MAX\_GRP\_SIZE, CLASS\_FEE)
   3. CLASS\_RESERVATION\_LIST (PARTICIPANT\_NUM, LAST\_NAME, FIRST\_NAME, CLASS\_NUM, CLASS\_DESC, CLASS\_DATE)
   4. (CLASS\_DATE, CLASS\_NUM, CLASS\_DESC, PARTICIPANT\_NUM, LAST\_NAME, FIRST\_NAME)

Graphical user interface, application

Description automatically generated

I could see issues with normalization with this request, as there is a lot of redundancy between parts c & d of their request. Those tables are nearly identical, so perhaps the best solution would be to hold the class and participant data in their own tables as requested, and then have a third table with a class\_reservation\_id, to identify the times when a participant has requested a class spot. The participant list could simply be a report that is generated daily and does not need to be it own table. If this was a table, it would be holding identical data as the reservation table.