IST 659

**Bug and Lessons Learned Report**

Lost Property Report Database Management

**BY**

**Manideep & Lakshya**

horizontal line

# 

# Bugs and Fixes

1. As per the initial design, the data type of the primary key(PK) of Incident entity (IncidentId) is VARCHAR(20). Things went smooth until database design and data insertion. During the interface design, while creating a form that lets the user to create an lost/found incident, the IncidentId(PK) field has to be filled by the user which is impractical.(if this field is exposed to the user, they can use any random value because there is no constraint on IncidentId to make it follow a definite pattern and make sure it is unique so that the primary key constraint is not violated.)

Then we realised that primary key has to be auto generated to overcome the issue. So we changed the data type of primary key from VARCHAR(20) to INT IDENTITY(10001,1). Now the user doesn't have to worry about IncidentId as it is autogenerated for him by the database.



1. After fixing above problem, the next glitch was with the subtype of Incident entity i.e., lost and found entities. Since their primary key is inherited from the Incident entity, we were supposed to catch the auto-generated primary key of Incident entity, to make a successful entry in its subtype.

Then we came up with a procedure that catches this auto-generated primary key and makes an entry into subtype entity. After every insert into the Supertype entity, this procedure is executed and records are inserted into lost/found entity.





1. The Found entity had different requirements. The LocationStored attribute can take null values only if IsItemReported attribute takes the value ‘NO’. If it takes the value of ‘YES’ then StoredLoaction attribute should have a not null constraint. We understood that an if-else type of conditional statement is required in this scenario. But SQL supports if else statements in data retrieval queries or procedures but not within CREATE syntax.

After referring few online sources, we have come up with a check constraint which works as an analogous of if-else.



1. In the above scenario not null constraint is enforced on Location stored if isitemReported is ‘Yes’. But, the null constraint is not enforced if isItemReported is ‘NO’. If item not reported then its storage location must be null. But still we can go ahead and insert a not null value storedLocation attribute which is a bug that is left unfixed in this project.

## Lessons Learnt

1. Apart from business rules and physical design, the interface design must also be considered to avoid rework. In our implementation, we completely ignored the interface design and later realised data type of attributes must be changed to deal with the limitations of interface. This led to tedious and monotonous work being done repetitively.(Like dropping tables, changing data types, reinserting data with adjusted data types, making changing in associated tables so that constraints are not violated)
2. The importance of procedures, triggers and transactions in implementing a complicated scenario.
3. Lastly, the process of developing a project from scratch right from identifying the business rules and Physical design. Being from a technical background, we used to communicate only through code/comments. In this course, we learnt the importance of reports and how they keep things simple in real business environment. Reports not only help to keep track of work done, but lets a new member understand the background and progress made.