
Test Data Management
Lesson 1: Test Data Management
Capgemini

Lesson Objectives

To understand the following topics:

- What is Test Data?
- Test Data Challenges
- Challenges of Test Data sourcing
- Need for TDM
- Essential Steps for a streamlined TDM
- TDM Strategies
- Corrupted Test Data
- Preparing Ideal Test Data to ensure Maximum Test Coverage
  - Case study Example
- TDM Tools

Introduction

- With the growth of Business, Information and Technology also faced a revolutionary growth where in the testers specially experience extensive consumption of test data in the software testing life cycle.
- The testers don't only collect/maintain data from the existing sources, but also they generate huge volumes of test data to ensure quality improvement in the delivery of the product for real-world use.
- Testers need to continuously explore, learn and apply the most efficient approaches for data collection, generation, maintenance, automation and comprehensive data management for any types of functional and non-functional testing.

What is Test Data ?

- Test Data : When we start writing our test cases to verify and validate the given features and developed scenarios of the application under the test, we need information that is used as input to perform the tests for identifying and locating the defects.

- Test Data may be in any form like :
  - System test data
  - SQL test data
  - Performance test data
  - XML test data
- Test Data needs to be precise and complete for making the bugs out.

Test data information needs to be precise and complete for making the bugs out. It is what we call test data. To make it accurate, it can be names, countries, etc…, are not sensitive, where data concerning to Contact information, SSN, medical history, and credit card information are sensitive in nature.

- **Lack of Data Variety:** A lack of variety of data prevents testing of edge cases or one-off instances that may cause defects. While edge cases can be added to test data manually, this can be a labor-intensive process.
- **Lack of Production Data:** A lack of production-like data and data volumes can prevent a test team from reproducing the volume required to do sufficient Performance and Load testing. Using production environment data (masked appropriately) provides real-life examples for test teams to replicate.
- **Human Error:** Manual data creation is labor intensive and prone to human error. Manual data creation can also cause delays to testing, as data needs to be recreated for each test cycle or iteration.
- **Data Security Risk:** Leveraging a production database for testing needs without properly masking personal identifiable information (PII) and other sensitive data may add exposure risks because many non-production environments are not as properly secured as production environments
- **Data Integrity Risk:** Manually bringing over production data to a testing environment (or from another source to another target) can create potential error points if referential integrity and parent-child relationships are not preserved in the process.
- **Storage Growth:** Using a full dump of production data in a test region when it isn't needed contributes to storage growth, especially when added to the existing data already in the test region.
- **Non-Standardized TDM:** TDM team siloes across the project, partial automation of test data management processes, and ad-hoc test data requests brings in overhead to the testing organizations.

Challenges of Test Data Sourcing

- The teams may not have adequate test data generator tools knowledge and skills
- Test data coverage is often incomplete
- Less clarity in data requirements covering volume specifications during the gathering phase
- Testing teams do not have access to the data sources
- Delay in giving production data access to the testers by developers
- Production environment data may be not fully usable for testing based on the developed business scenarios

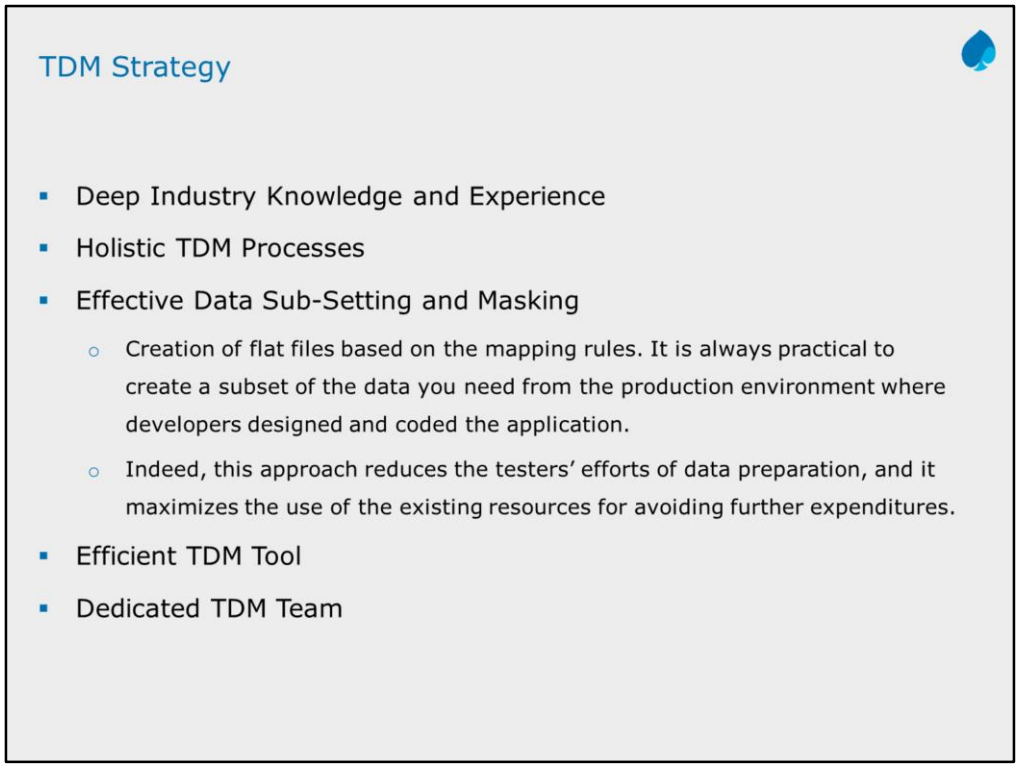
Challenges of Test Data Sourcing (Contd…)

- Large volumes of data may need in a short period of given time
- Data dependencies/combinations to test some of the business scenarios
- The testers spend more time than required for communicating with architects, database administrators and BAs for gathering data
- Mostly the data is created or prepared during the execution of the test
- Multiple applications and data versions
- Continuous release cycles across several applications
- Legislation to look after Personal Identification Information (PII)

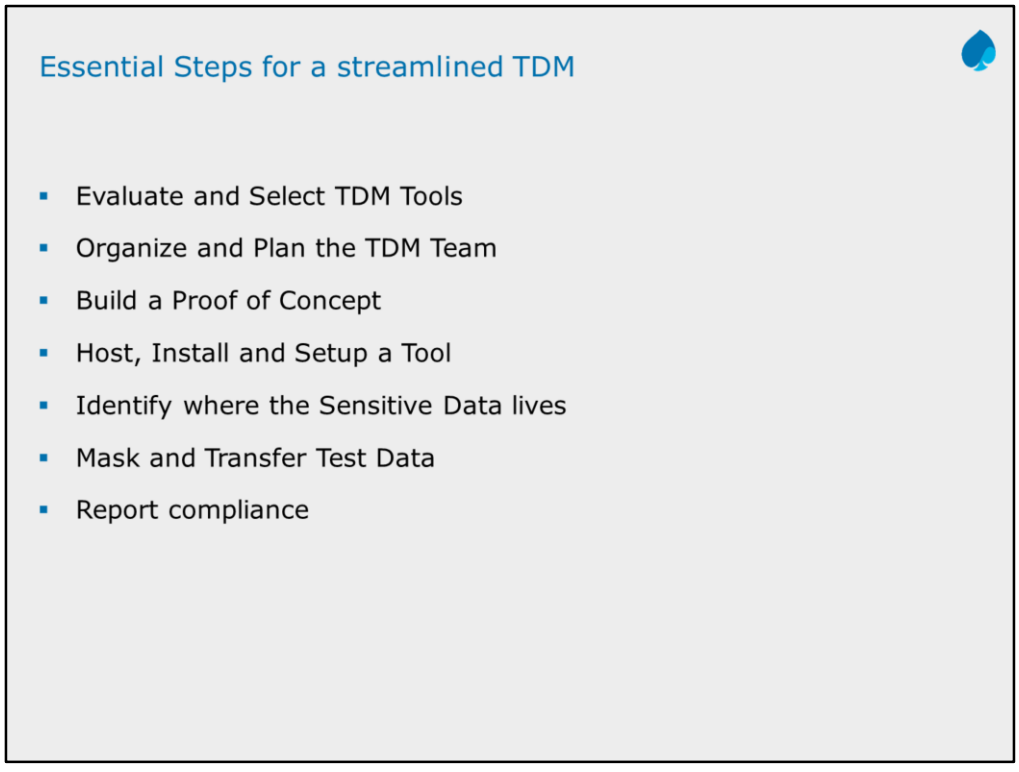
Need of TDM

Implementing a solid TDM strategy and using industry tools

- Saves time and cost by eliminating the manual work involved in identifying, transforming, deploying, and maintaining reusable sets of test data.
- Provide a balance of industry depth and low-cost test data management resources, leading to lower cost of managing and accommodating test data needs
- Provide high quality, safe test data to the needed target environments in a timely manner
- Provide a flexible delivery model for delivery of test data management services
- Reduce risk around data security, data integrity, and human error



- Deep Industry Knowledge and Experience: Harness deep industry knowledge in broad identification of test data requirements. These requirements can then be utilized to identify the source, data subset policies, and data synthesis needs
- Holistic TDM Processes: Publish clearly defined data management workflows, policies, and procedures for the creation, load, and protection of test data. Define metrics to measure data health in each environment and use reduced volumes of production data when a full load is not needed.
- Effective Data Sub-Setting and Masking: Use data subsets which contribute to effective testing as it enables usage of reallife data sets (preserves referential integrity from the source) only on a smaller scale. The use of data masking is to maintain data security by protecting PII and other sensitive data.
- An Efficient TDM Tool: Opt for automated TDM tools to reduce the manual process of creating test data and reduce the chance of errors. Reuse subset criteria and masking policies on multiple and different sources. Identify the table relationships and where the sensitive data is fast.
- Dedicated TDM Team: Dedicate resources to manage and accommodate the project's test data needs. It is recommended that this very important task not be added to the load of testers or the development team.



- Tool Evaluation and Selection –provide candid and unbiased feedback on each prospective tool's fit.
- TDM Team Organization and Planning – Having resources dedicated to managing and accommodating the project's test data needs is a full-time job.
- Proof of Concept – After a tool is selected, we can help build and demo a proof of concept against a select amount of data to demonstrate general capability and benefit.
- Tool Hosting, Installation, and Setup –host, install, and set up the chosen TDM tool to save your organization from the need of additional hardware and infrastructure.
- Identifying Where Sensitive Data Lives Completing data discovery may help confirm sensitive data for some organizations, while it may help avoid a compliance issue in future for others.
- Sub-setting, Masking, and Transferring Data reduces the size of data moving from production (or the source) to the target region(s).
- Compliance Reporting – Create reports that show proof that no PII is being kept in the target region(s) after it has been masked and moved there. It's important (and may be an audit requirement) to have proof of this fact.

Corrupted Test Data

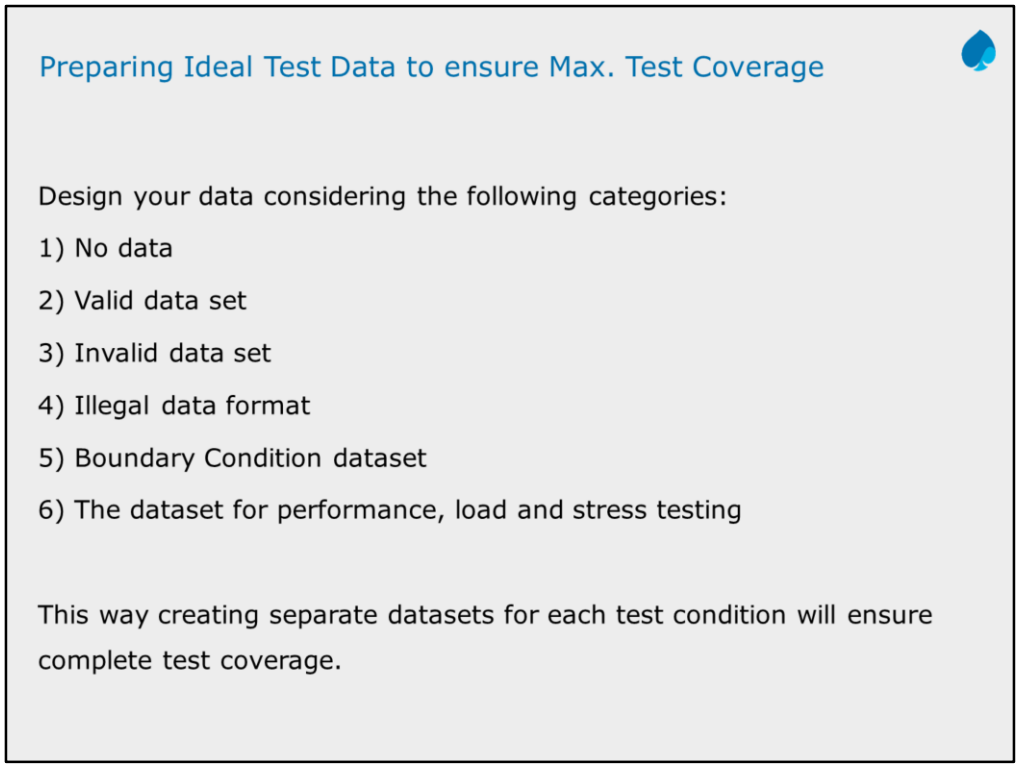
Reasons for Data Corruption :

- When more than one tester working on different modules of an AUT in the testing environment at the same time, the chances of data getting corrupted is very high.
- In the same environment, the testers modify the existing data as per their need/requirements of the test cases.
- As soon as the next tester picks up the modified data, and he/she perform another execution of the test, there is a possibility of that particular test failure which is not the code error or defect.

Corrupted Test Data

Minimizing the chances of Data Corruption

- Having the backup of your data
- Return your modified data to its original state
- Data division among the testers
- Keep the data warehouse administrator updated for any data change/modification



1) No data: Run your test cases on blank or default data. See if proper error messages are generated.

2) Valid data set: Create it to check if the application is functioning as per requirements and valid input data is properly saved in database or files.

3) Invalid data set: Prepare invalid data set to check application behavior for negative values, alphanumeric string inputs.

4) Illegal data format: Make one data set of illegal data format. The system should not accept data in an invalid or illegal format. Also, check proper error messages are generated.

5) Boundary Condition dataset: Dataset containing out of range data. Identify application boundary cases and prepare data set that will cover lower as well as upper boundary conditions.

6) The dataset for performance, load and stress testing: This data set should be large in volume.

Preparing Ideal Test Data to ensure Max. Test Coverage

Case Study

- Consider a Login Page with below given requirements
  - Username & password can be min. 4 alphabets and max. 15 alphabets
  - Sample correct Username: admin
  - Sample correct Password: pass
- Create Test Data as below for the given Data set categories

No.	Test Case	No Data	Valid Data	Invalid/Illegal Data
1	Username		admin	server
2	Password		pass	@@#$123

Preparing Ideal Test Data to ensure Max. Test Coverage

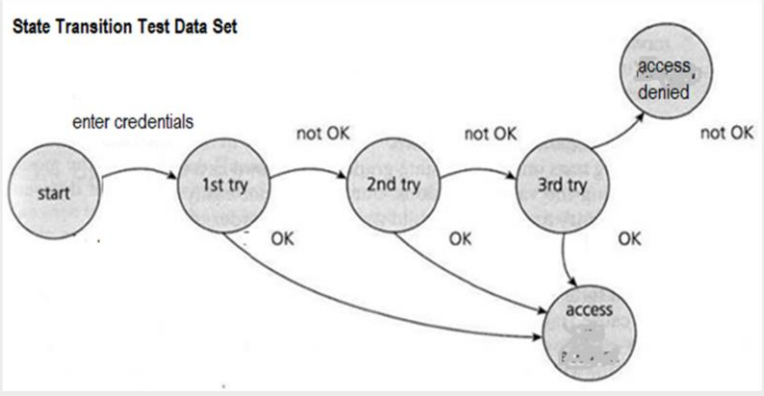
Case Study (Contd…)

BOUNDARY VALUE ANALYSIS			EQUIVALENCE CLASS PARTITIONING	
Min/Max	**Size**	**Action**	**Valid**	**Invalid**
Min	4	Pass	A-Z A-z	0-9 Special characters Blank spaces
Max	15	Pass		
Min-1	3	Fail		
Min+1	5	Pass		
Max-1	14	Pass		
Max+1	16	Fail		

DECISION TABLE TEST					
Conditions	Username	T	T	F	F
	Password	T	F	T	F
Actions	Expected Result	Logged In	Error: Please enter correct Password	Error: Please enter correct Username	Error: Please enter correct Username

Preparing Ideal Test Data to ensure Max. Test Coverage

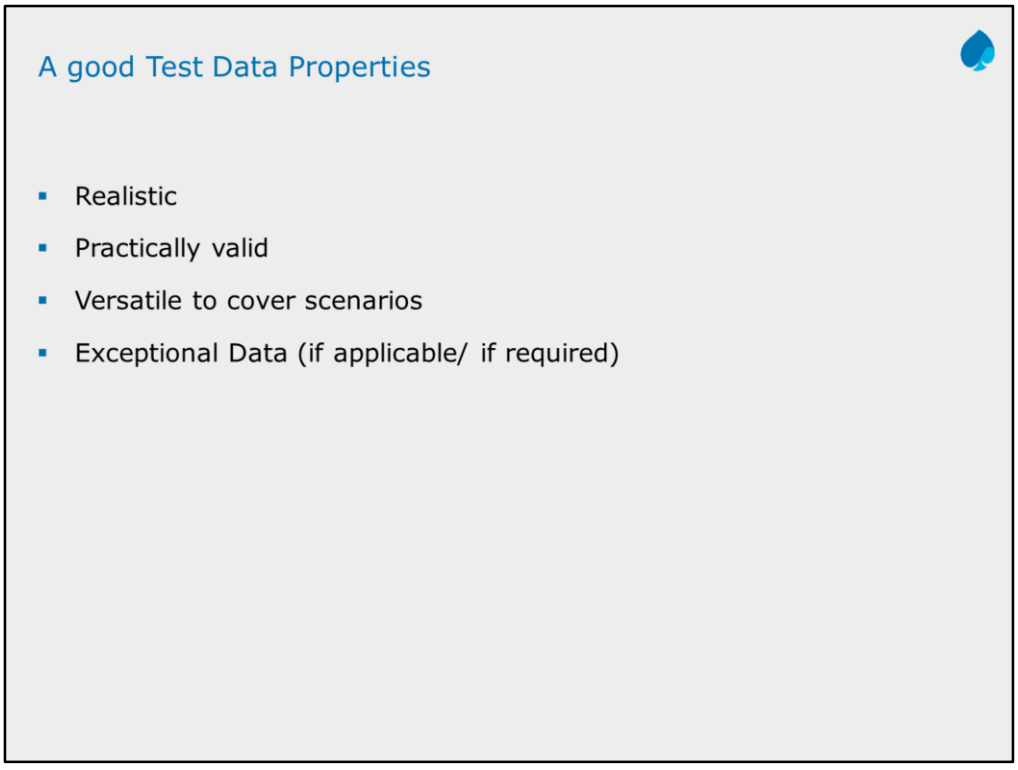
Case Study (Contd…)



Preparing Ideal Test Data to ensure Max. Test Coverage

Case Study(Contd..)

USE CASE TEST DATA		
Success Scenario	**Step**	**Description**
User	1	Enters correct username and password
System	2	Validates credentials
System	3	Access open to the website
Failure Scenario	**Step**	**Description**
User	1	Enters invalid username and password
System	2	Validates credentials and Access Denial message
User	3&5	Re-enters invalid username
System	4&6	Access Denial message – same for third try
System	7	Application closed

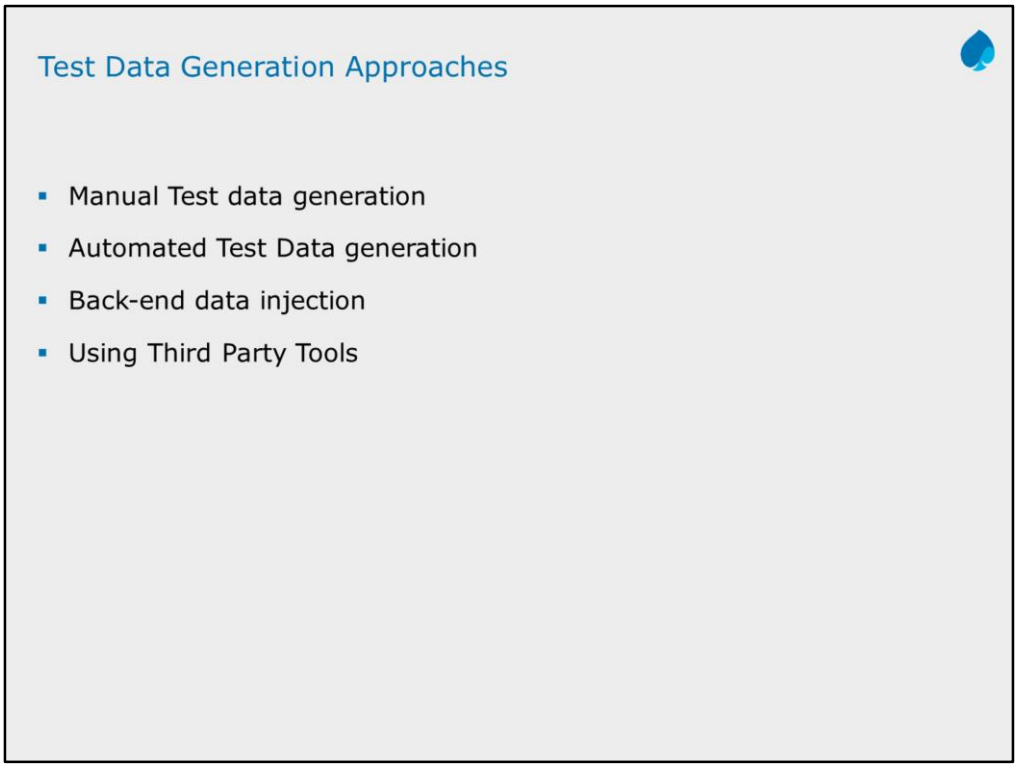


1) Realistic:
By realistic, it means the data should be accurate in the context of real-life scenarios. For example, in order to test the 'Age' field, all the values should be positive and 18 or above. It is quite obvious that the candidates for admission in the university are usually 18 years old

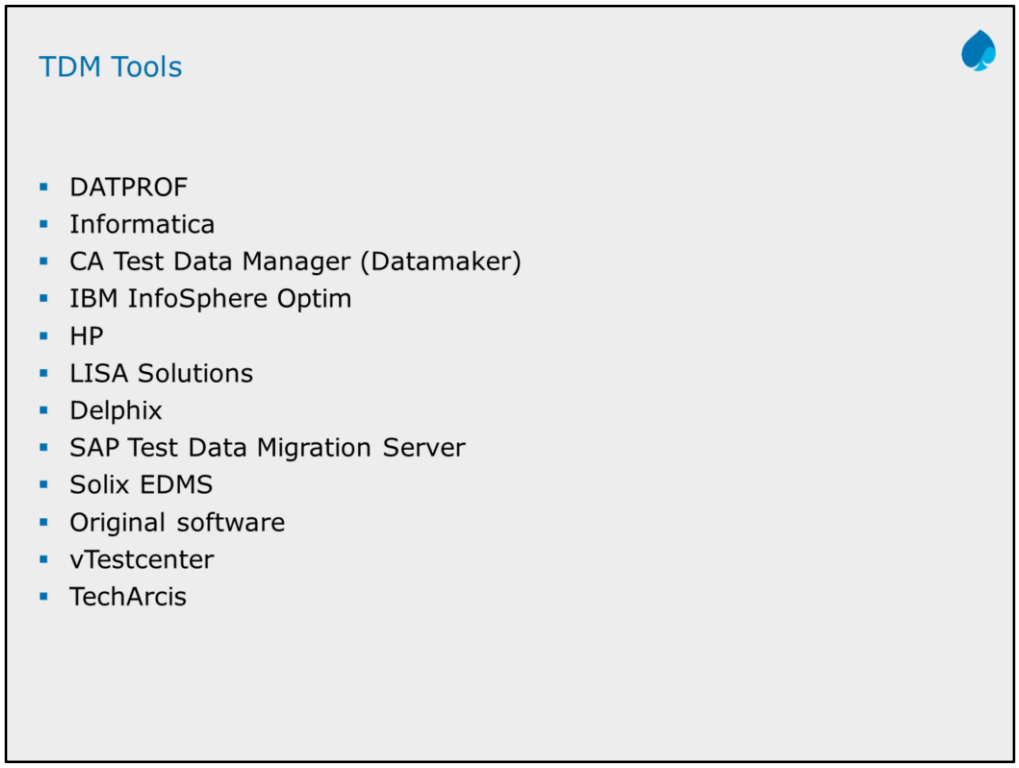
2. Practically valid:
This is similar to realistic but not the same. This property is more related to the business logic of AUT e.g. value 60 is realistic in the age field but practically invalid for a candidate of Graduation or even Masters Programs. In this case, a valid range would be 18-25 years (this might be defined in requirements).

3. Versatile to cover scenarios:
There may be several subsequent conditions in a single scenario, so choose the data shrewdly to cover maximum aspects of a single scenario with the minimum set of data, e.g. while creating test data for result module, do not only consider the case of regular students who are smoothly completing their program. Give attention to the students who are repeating the same course and belong to different semesters or even different programs.

4. Exceptional data (if applicable/required):
There may be certain exceptional scenarios that occur less frequently but demand high attention when occurred, e.g. disabled students related issues.



Manual Test data generation: In this approach, the test data is manually entered by testers as per the test case requirements. It is a time taking the process and also prone to errors.
Automated Test Data generation: This is done with the help of data generation tools. The main advantage of this approach is its speed and accuracy. However, it comes at a higher cost than manual test data generation.
Back-end data injection: This is done through SQL queries. This approach can also update the existing data in the database. It is speedy & efficient but should be implemented very carefully so that the existing database does not get corrupted.
Using Third Party Tools: There are tools available in the market that first understand your test scenarios and then generate or inject data accordingly to provide wide test coverage. These tools are accurate as they are customized as per the business needs. But, they are quite costly.



DATPROF is a top tool that provides, data masking, synthetic test data generation, Test Data Subsetting technologies, and a test data provisioning platform.
Informatica Test Data management tool is a top tool that provides automated data subsetting, data masking, data connectivity, and test data-generation capabilities. It automatically finds out sensitive data locations. It is fulfilling the increasing demand for the test data.
CA Test Data Manager is another top tool that provides high synthetic data generation solutions. The design of this tool is very flexible that simplify the functionality of the testing. It is the product of CA technologies. It acquires the DataMaker of Grid-tools. It is also called an Agile designer, DataFinder, Fast DataMaker, DataMaker.
IBM InfoSphere Optim tool has built-in workflow and on-demand service facilities. This feature helps in continuous testing and agile software development.
LISA Solutions are an automatic testing tool that creates a virtual dataset that gives a high level of functional accuracy. The tool can import test data from a different type of data sources such as excel sheets, XML, log files, etc. Tester or developers can easily manipulate the test data and integrate them into a single place.
Delphix Test data tool provides high quality and faster testing. During development, testing, training or reporting, redundant data is shared across all this process. This sharing of data is called data virtualization or virtual data. The virtual data of the tool provide complete, full size and real data sets in the few minutes that takes very few spaces.
SAP test data management server creates a small test data subset and provides a non-production environment for development, testing, and training. It increases the data extraction that reduces the infrastructure expenditures and storage space in the testing environment.

Summary

- In this lesson, you have learnt:
  - What is Test Data?
  - Test Data Challenges
  - Challenges of Test Data sourcing
  - Need for TDM
  - Essential Steps for a streamlined TDM
  - TDM Strategies
  - Corrupted Test Data
  - Preparing Ideal Test Data to ensure Maximum Test Coverage
  - TDM Tools