DATABASE TESTING

tutorialspoint

SIMPLY EASY LEARNING

About the Tutorial

Database testing includes performing data validity, data integrity testing, performance check related to database and testing of procedures, triggers and functions in the database. This is an introductory tutorial that explains all the fundamentals of Database testing.

Audience

This tutorial has been designed for all those readers who want to learn the basics of Database testing. It is especially going to be useful for all those software testing professionals who are required to test the applications in order to find out how they affect the database performance.

Prerequisites

We assume the readers of this tutorial have hands-on experience of handling a database using SQL queries. In addition, it is going to help if the readers have an elementary knowledge of basic database concepts.

Copyright & Disclaimer

Table of Contents

About the Tutorial i
Audience i
Prerequisites i
Copyright & Disclaimer i
Table of Contents ii

1. DB Testing – Overview 1
Why You Need to Perform Database Testing? 1
Database Testing Vs Front-End Testing 2

2. DB Testing – Types 4
Structural Database Testing 4
Functional Testing 6
Nonfunctional Testing 7

3. DB Testing – Processes 9
Database Testing Stages 9

4. DB Testing – Techniques 10
Database Schema Testing 10
Stored Procedure Tests 10
Trigger Tests 11
Server Setup Scripts 11
Functional Testing Method 12
Stress Testing 12
Benchmark Testing 12
Testing a Database via Front-end 13

5. DB Testing – Scenarios 14
Structured Database Testing 14
Functional Database Testing 14
Non-Functional Database Testing 14

6. DB Testing – Objects 16
Schemas 16
Tables in Database 16
Stored Procedures 16
Triggers 17

7. DB Testing – Data Integrity 18

8. DB Testing – Data Mapping 19
Key Aspects of Data Mapping 19
Steps in Data Mapping Testing 19

9. DB Testing – Performance (Load, Stress) 20
Load Testing 20
Stress Testing 20

10. DB Testing – Tools 21

11. DB Testing – Database Backup 22
Types of Data Backups 22

12. DB Testing – Recovery 24

13. DB Testing – Security 25
Types of Threats on a Database System 25
Database Security Testing Techniques 26
Database Security Testing Tools 27

14. DB Testing – Challenges 29

15. DB Testing – Interview Questions 31

1. DB TESTING – OVERVIEW

Database testing includes performing data validity, data integrity testing, performance check related to database and testing of procedures, triggers and functions in the database.

Example

Consider an application that captures the day-to-day transaction details for users and stores the details in the database. From database testing point of view, the following checks should be performed:

- The transactional information from the application should be stored in the database and it should provide correct information to the user.
- Information should not be lost when it is loaded to database.
- Only completed transactions should be stored and all incomplete operations should be aborted by the application.
- Access authorization to database should be maintained. No unapproved or unauthorized access to user information should be provided.

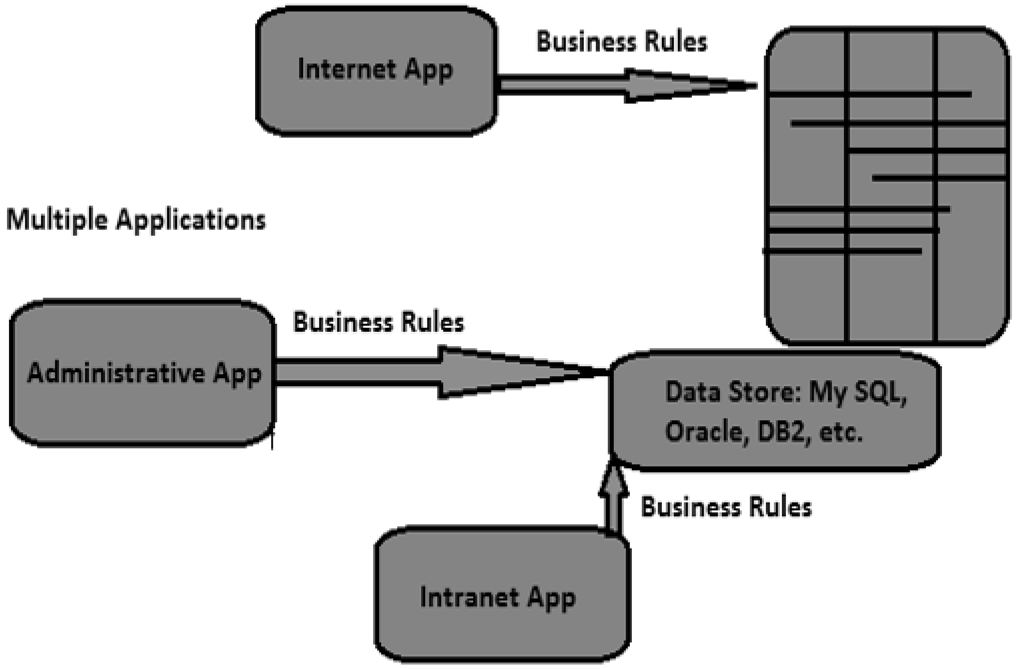
Why You Need to Perform Database Testing?

There are multiple reasons why database testing is performed. There is a need to perform data integrity, validation and data consistency check on database as the backend system is responsible to store the data and is accessed for multiple purpose.

Given below are some common reasons for Database testing:

- To ease the complexity of calls to database backend, developers increase the use of **View** and **Stored** Procedures.
- These **Stored** procedures and **Views** contain critical tasks such as inserting customer details (name, contact information, etc.) and sales data. These tasks need to be tested at several levels.
- **Black-box testing** performed on front-end is important, but makes it difficult to isolate the problem. Testing at the backend system increases the robustness of the data. That is why database testing is performed on back end system.
- In a database, data comes from multiple applications and there is a possibility that harmful or incorrect data is stored in the database. Therefore, there is a need to check

database components regularly. In addition, data integrity and consistency should be checked regularly.



Database Testing Vs Front-End Testing

Database testing is different from front-end UI testing. The following table highlights the key differences:

Database Testing	UI Testing
Database testing is known as data validation and integrity testing or back-end testing.	UI testing or front-end testing is also called Application testing or GUI testing.
Database testing involves testing of back-end components, which are not visible to users. This includes database components and DBMS systems such as My SQL, Oracle.	UI testing involves checking functionalities of an application and its components like forms, graphs, menus, reports, etc. These components are created using front-end development tools like VB.net, C#, Delphi, etc.

Database testing involves checking stored procedures, views, schemas in database, tables, indexes, keys, triggers, data validations and data consistence check.	UI testing involves checking the functionality of application, buttons, forms and fields, calendar and images, navigation from one page to other, and the overall functionality of the application.
To perform DB testing, a tester needs a thorough knowledge of database concept- like procedures and functions, views, indexes, keys and good hands-on SQL.	To perform UI testing, a tester needs a good understanding of business requirements, application functional knowledge, coding, etc.
Data comes from multiple heterogeneous data sources over web applications, Intranet applications and various other applications.	Data is entered manually into applications. It involves functional testing of front-end applications.

2. DB TESTING – TYPES

Based on the function and structure of a database, DB testing can be categorized into three categories:

- **Structural Database Testing** – It deals with table and column testing, schema testing, stored procedures and views testing, checking triggers, etc.

- **Functional Testing –** It involves checking functionality of database from user point of view. Most common type of Functional testing are White box and black box testing.

- **Nonfunctional Testing –** It involves load-testing, risk testing in database, stress testing, minimum system requirements, and deals with the performance of the database.

Structural Database Testing

Structural database testing involves verifying those components of database, which are not exposed to end users. It involves all the components of repository, which are used to store the data and are not changed by the end users. Database administrators with good command over SQL stored procedures and other concepts normally perform this testing.

Discussed are the common components tested with respect to Structural Testing:

Schema / Mapping Testing

It involves validating the objects of front-end application with database object mapping.

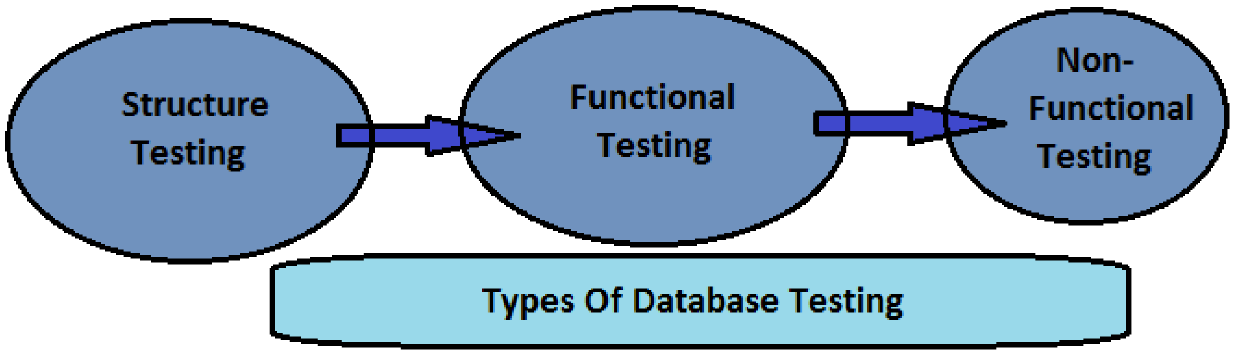
In Schema Testing:

- Sometimes it happens that the end user application objects are not correctly mapped or compatible with database objects. Therefore, checking the validation of the various schema formats associated with the databases is required.
-
- It is required to find the unmapped objects in database, like tables, views, columns etc. is required.

There are various tools in the market that can be used to perform object mapping in schemas.

Example: In Microsoft SQL Server, a tester can write simple queries to check and validate schemas in the database.

If the tester wants to make changes to a table structure, he/she should ensure that all the **stored** procedures having that table are compatible with this change.



Stored Procedures and Views Testing

In this testing, a tester ensures that the manual execution of stored procedures and views generate the required result.

The tester ensures:

- If it enables the required triggers to be executed as expected.
- If the development team has covered all the loops and conditions by passing input to applications in the procedures.
- If there are any unused stored procedures in the database.
- TRIM operations are applied properly when the data is fetched from required tables in database.
- Validation of the overall integration of the stored procedure modules as per as the requirements of the application under test.
- Exception and error handling mechanisms are followed.

The most common tools that are used to perform stored procedures testing are **LINQ**, **SP Test tool**, etc.

Trigger Testing

In trigger testing, a tester needs to ensure the following:

- Whether the coding conventions are followed during the coding phase of the triggers.
- See the triggers executed meets the required conditions.
- Whether the trigger updates the data correctly, once they have been executed.

- Validation of Update/Insert/Delete triggers functionality w.r.t application under test.

Tables and Column testing

The key areas covered in this testing are:

- Validating the data types in the database to field values in front-end application.
- Validating the length of data field in database to length of data types in the application.
- Checking if there are any unmapped tables or columns in the database from application field objects.
- Naming conventions of database tables and columns are verified, if they are in accordance with business requirement or not.
- Validating the Keys and Indexes in the database, i.e., primary and foreign keys in tables are defined as per requirement.
- Check if the primary keys and their corresponding foreign keys are same in two tables.
- Check Unique and NOT NULL characteristics of keys are maintained.
- Length and data type of keys and indexes are maintained as per requirement.

Database Server Check

Database Server check involves verifying:

- If the database server can handle the expected number of transactions as per the business requirement.
- If the configuration details of database servers meets the business requirement.
- If the user authorization is maintained as per requirement.

Functional Testing

Functional testing is performed keeping in mind an end-user point of view; whether the required transactions and operations run by the end-users meet the business specifications.

Black Box Testing

Black Box Testing involves verifying the integration of database to check the functionality. The test cases are simple and are used to verify incoming data and outgoing data from the function.

Various techniques such as cause-effect graphing technique, equivalence partitioning and boundary-value analysis are used to test the functionality of the database.

Its **advantages** are as follows:

- It is fairly simple and is performed in the early stages of development.
- Cost of developing test-cases is less as compared to white-box testing.

Its disadvantages are as follows:

- A few errors cannot be detected.

- It is unknown how much program needs to be tested.

White Box Testing

White Box Testing deals with the internal structure of the database and the specification details are hidden from the users. It involves the testing of database triggers and logical views, which are going to support database refactoring.

It performs module testing of database functions, triggers, views, SQL queries etc. This type of testing validates database tables, data models, database schema etc. It checks rules of Referential integrity. It selects default table values to check on database consistency.

The most common techniques used to perform white box testing are condition coverage, decision coverage, statement coverage, etc.

Coding errors can be detected in white-box testing, so internal bugs in the database can be eliminated. The limitation of white-box testing is that SQL statements are not covered.

Nonfunctional Testing

Nonfunctional testing involves performing load testing, stress testing, checking minimum system requirements to meet business specification, risk finding and performance optimization of database.

Load Testing

The primary target of load testing is to check if most running transactions have performance impact on the database.

In Load testing, the tester checks:

- The response time for executing the transactions for multiple remote users.
- Time taken by the database to fetch specific records.

Examples of load testing in different testing types:

- Running most used transaction repeatedly to see performance of database system.
- Downloading a series of large files from the internet.
- Running multiple applications on a computer or server simultaneously.

Stress Testing

Stress testing is performed to identify the system breakpoint. In this testing, application is loaded in such a way that the system fails at one point. This point is called the **breakpoint** of database system.

Determining the state of database transactions involves a significant amount of effort. Proper planning is required to avoid any time and cost-based issues.