# Unit 10. Database Testing


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.

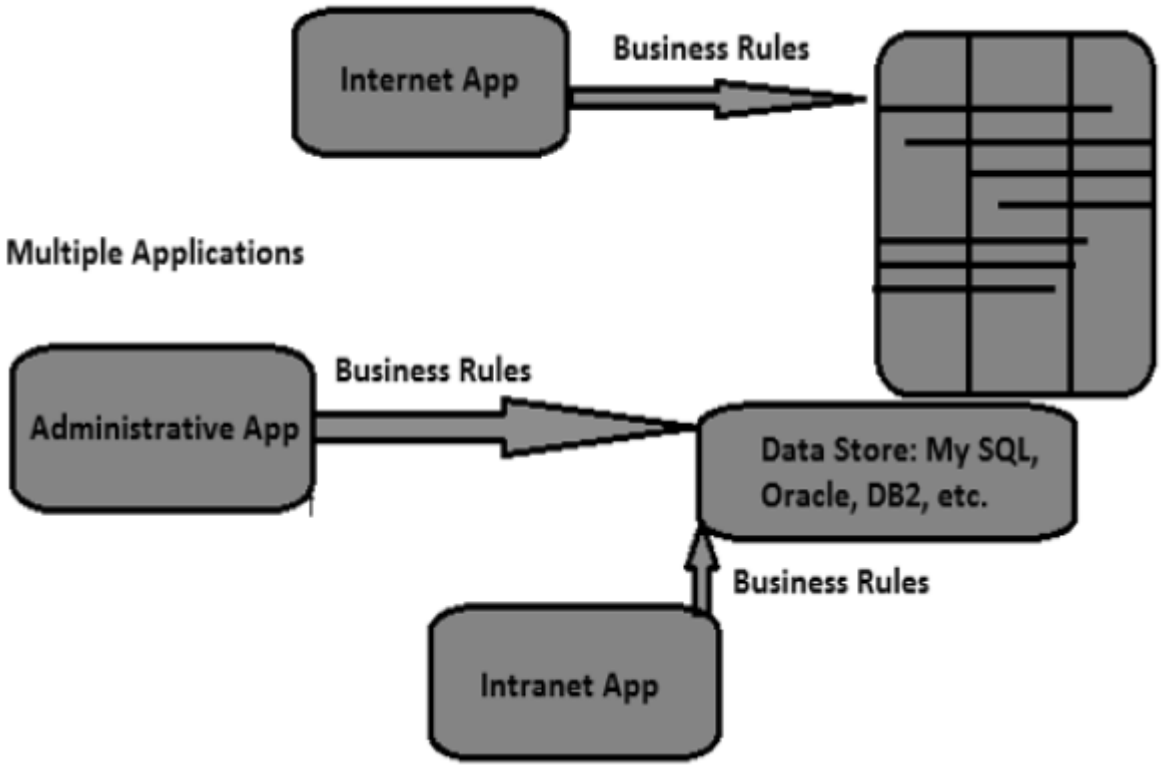
## Why You Need to Perform Database Testing?

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.<br><br>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.<br><br>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. |



# Database 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.

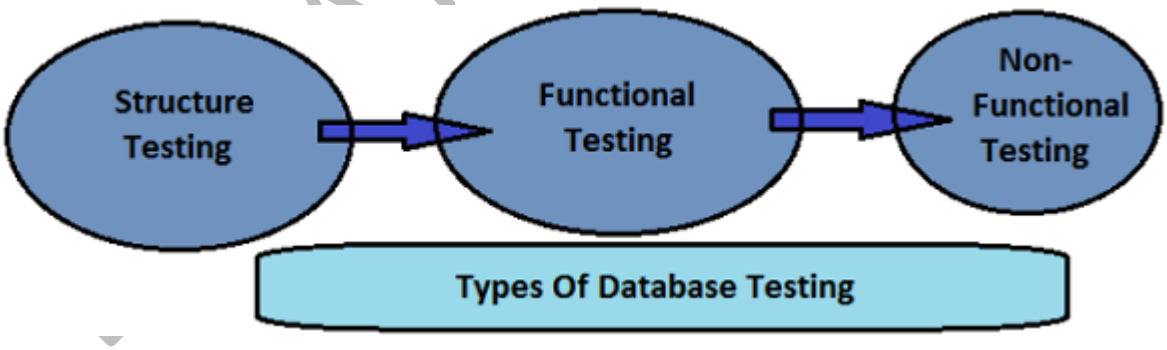
In Schema Testing −

- 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.

The most commonly used stress testing tools are **LoadRunner** and **WinRunner**.

Let us take an **example** of Stress Testing. A CRM application can take a maximum user load of 50000 concurrent users. Suppose you increase the load to 51000 and make some transactions such as updating records or adding an entry. As soon as you do the transaction, the application can sync with the database system. So the next test is to perform with a user load of 52000. Sometimes, Stress Testing is also called **Fatigue Testing**.

# Database Testing – Processes

The process to perform database testing is similar to testing of other applications. DB testing can be described with key processes given below.

- Set up the environment
- Run a test
- Check the test result
- Validate according to the expected results
- Report the findings to the respective stakeholders

Various SQL statements are used to develop the Test cases. The most common SQL statement, which is used to perform DB testing, is the **Select**statement. Apart from this, various DDL, DML, DCL statements can also be used.

**Example** – Create, Insert, Select, Update, etc.

## Database Testing Stages

DB testing is not a tedious process and includes various stages in database testing lifecycle in accordance with the test processes.

The key stages in database testing are –

- Checking the initial state
- Test run
- Outcome validation as per expected result
- Generating the results

**First stage** in DB Testing is to check the initial state of the database before starting the testing process. Then database behavior is tested for defined test cases. In accordance with the results obtained, test cases are customized.

For successful database testing, the workflow given below is executed by every single test.

- **Cleaning up the database** – If there is testable data in the database, it should be emptied.
- **Set up Fixture** – This involves entering the data into the database and check the current state of the database.

- **Perform test, verify results and generate results** − The Test is run and the output is verified. If the output is as per expected results, the next step is to generate the results as per requirement. Otherwise, testing is repeated to find the bugs in database.

# Database Testing – Techniques

This chapter explains the most common techniques that are used to perform Database Testing.

## Database Schema Testing

As mentioned earlier, it involves testing each object in the Schema.

### Verifying Databases and devices

- Verifying the name of database
- Verifying the data device, log device and dump device
- Verifying if enough space allocated for each database
- Verifying database option setting

### Tables, columns, column types rules check

Verify the items given below to find out the differences between actual and applied setting.

- Name of all the tables in database
- Column names for each table
- Column types for each table
- **NULL** value checked or not
- Whether a default is bound to correct table columns
- Rule definitions to correct table names and access privileges

### Key and Indexes

Verify the Key and indexes in each table −

- Primary key for each table
- Foreign keys for each table
- Data types between a foreign key column and a column in other table Indices, clustered or non-clustered unique or not unique

## Stored Procedure Tests

It involves checking whether a stored procedure is defined and the output results are compared. In a Stored Procedure test, the following points are checked −

- Stored procedure name
- Parameter names, parameter types, etc.
- **Output** − Whether the output contains many records. Zero rows are effected or only a few records are extracted.
- What is the function of Stored Procedure and what a stored procedure is not supposed to do?
- Passing sample input queries to check if a stored procedure extracts correct data.
- **Stored Procedure Parameters** − Call stored procedure with boundary data and with valid data. Make each parameter invalid once and run a procedure.
- **Return values** − Check the values that are returned by stored procedure. In case of a failure, nonzero must be returned.
- **Error messages check** − Make changes in such a way that the stored procedure fails and generate every error message at least once. Check any exception scenarios when there is no predefined error message.

# Trigger Tests

In a Trigger test, the tester must perform the following tasks −

- Make sure the trigger name is correct.
- Validate the trigger if it is generated for a specific table column.
- Trigger's update validation.
- Update a record with a valid data.
- Update a record with invalid data and cover every trigger error.
- Update a record when it is still referenced by a row in other table.
- Ensure rolling back transactions when a failure occurs.
- Find out any cases in which a trigger is not supposed to roll back transactions.

# Server Setup Scripts

Two types of tests should be performed −

- Setting up the database from scratch, and
- To set up an existing database.

## Integration Tests of SQL Server

Integration tests should be performed after you are through with component testing.

- Stored procedures should be called intensively to select, insert, update, and delete records in different tables to find any conflicts and incompatibility.
- Any conflicts between schema and triggers.
- Any conflicts between stored procedures and schema.
- Any conflicts between stored procedures and triggers.

# Functional Testing Method

Functional testing can be performed by dividing the database into modules as per functionality. The functionalities are of the following two types –

- **Type 1** – In Type 1 testing, find out the features of the project. For each major feature, find out the schema, triggers, and stored procedures responsible to implement that function and put them into a functional group. Then test each group together.
- **Type 2** – In Type 2 testing, the border of functional groups in a back-end is not obvious. You can check the data flow and see where you can check the data. Start from the front-end.

The following process takes place –

- When a service has a request or saves data, some stored procedures will get called.
- The procedures will update some tables.
- Those stored procedures will be the place to start testing and those tables will be the place to check the test results.

# Stress Testing

Stress Testing involves getting a list of major database functions and corresponding stored procedures. Follow the steps given below for Stress Testing –

- Write test scripts to try those functions and every function must be checked at least once in a full cycle.
- Perform the test scripts again and again for a specific time period.
- Verifying the log files to check any deadlocks, failure out of memory, data corruption, etc.

# Benchmark Testing

If your database does not have any data problems or bugs, system performance can be checked. A poor system performance can be found in benchmark testing by checking the parameters given below −

- System level performance
- Identify most-likely-used functions/features
- Timing – maximum time, minimum time and average time to perform functions
- Access volume

# Testing a Database via Front-end

Back-end bugs can also be found sometimes by doing front-end testing. You can follow the simple steps given below to detect bugs by front-end testing.

- Write queries from the front-end and issue the searches.
- Pick up an existing record, change the values in some fields, and save the record. (It involves the UPDATE statement or update stored procedures and update triggers.)
- Insert a new menu item in the front-end window. Fill in the information and save the record. (It involves the INSERT statements or insertion stored procedures and deletion triggers.)
- Pick up an existing record, click on the DELETE or REMOVE button, and confirm the deletion. (It involves the DELETE statement or deletion stored procedures and deletion triggers.)
- Repeat these test-cases with invalid data and see how the database responds.

# Database Testing – Scenarios

In this chapter, we will see some common database test scenarios with respect to various testing methods.

# Structured Database Testing

Common database scenarios with respect to Structured Database Testing are given below −

- Verifying the name of database, verifying the data device, log device and dump device, verifying if enough space allocated for each database and verifying database option setting.

- Names of all the tables in database, column names for each table, column types for each table, null value check or not. Verify the Key and indexes in each table: Primary key for each table, foreign keys for each table.
- Data types between a foreign key column and a column in other table Indices, clustered or non-clustered unique or not unique.

# Functional Database Testing

Common Database Test scenarios with respect to **Functional Database Testing** are –

- Finding out the schema, triggers and stored procedures responsible to implement that function and make them into a functional group and then each group can be tested together.
- Check data flow and see where you can check the data. Start from the front-end.

# Non-Functional Database Testing

Common Database Test scenarios with respect to **Non-Functional Database Testing** are –

- Write test scripts to try major functions and every function must be checked at least once in a full cycle.
- Perform the test scripts again and again for a specific time period.
- Verifying the log files to check any deadlock, failure out of memory, data corruption, etc.
- Write queries from a front end and issue the searches. Pick up an existing record, change values in some fields and save the record. (It involves UPDATE statement or update stored procedures, update triggers.)
- Insert a new menu item in a front-end window. Fill in information and save the record. (It involves INSERT statements or insertion stored procedures, deletion triggers.)
- Pick up an existing record, click on the DELETE or REMOVE button, and confirm the deletion. (It involves DELETE statement or deletion stored procedures, deletion triggers.)
- Repeat these test-cases with invalid data and see how the database responds.

# Database Testing – Objects

**Schemas**, **tables**, **stored procedures**, and **Triggers** are key objects of a database. We have already shared DB testing types and test scenarios for these data base objects.

# Schemas

A database schema defines the structure of a database system in a format supported by the database management system. A Schema refers to how a database is structured (composed of database tables in the case of Relational Databases).

The database schema is a set of formulas called integrity constraints imposed on a database. These integrity constraints ensure compatibility between parts of the schema.

In a relational database, the schema consists of tables, fields, views, indexes, packages, procedures, functions, triggers, types, materialized views, synonyms, database links, and other elements.

Schemas are generally stored in a data dictionary. Although a schema is defined in text database language, the term is often used to refer to a graphical depiction of the database structure. In other words, schema is the structure of the database that defines the objects in the database.

Common type of Schemas used in a data warehouse are −

- Star Schema
- Snowflakes Schema
- Galaxy Schema

# Tables in Database

In a relational database, a table is used to organize the information into rows and columns.

**Example** − A Customer table contains information such as customer id, addresses, phone numbers, and so on as a series of columns.

Each single piece of data is a field in the table. A column consists of all the entries in a single field, such as the telephone numbers of all the customers. Fields are organized as records, which are complete sets of information (such as the set of information about a particular customer), each of which comprises a row.

# Stored Procedures

A stored procedure is a series of SQL statements stored in the database in a compiled form and multiple programs can share it. The use of stored procedures can be helpful in maintaining data integrity, data control access and improving productivity.

## Triggers

A database trigger is code that is executed in response to certain events on a particular table or view in a database. The trigger is mostly used for maintaining the integrity of the information on the database.

# Database Testing – Data Integrity

Data Integrity is important in a database. It includes data validation before insertion, updates, and deletion. Triggers must be in place to validate reference table records.

For checking Data Integrity, you need to perform the following operations −

- You need to check major columns in each table and verify if any incorrect data exists. (Characters in name field, negative percentage, etc.)
- Find out inconsistent data and insert them into relevant tables and see if any failure occurs.
- Insert a child data before inserting its parent's data. Try to delete a record that is still referenced by the data in another table.
- If a data in a table is updated, check whether the other relevant data is updated as well. You need to ensure that replicated servers or databases are in sync and contain consistent information.

# Database Testing – Data Mapping

Data mapping in a database is one of the key concept that needs to be validated by every tester. Usually the testers have to verify the user interface front end field mapping with the corresponding back end database field.

This information is given in Software requirement specification or business requirement specification SRS/BRS document. If mapping is not provided, then you need to check the coding part.

When you take any action in the front end application, there is a corresponding CRUD action get invoked, and tester have to check the every invoked action is successful or not.

## Key Aspects of Data Mapping

Given below are the key aspects of Data Mapping −

- To check the fields in the UI/Front end forms and mapped consistently with the corresponding DB table. This mapping information is defined in the requirements documents as mentioned above.
- For any action performed in the front end of an application, a corresponding CRUD 'Create, Retrieve, Update and delete' action gets initiated at the back end.
- A tester will have to check if the right action is invoked and the invoked action in itself is successful or not.

## Steps in Data Mapping Testing

Given below are the steps followed for Data Mapping Testing −

- **Step 1** − First check for syntax error in each script.
- **Step 2** − Next is to check for table mapping, column mapping, and data type mapping.
- **Step 3** − Verify lookup data mapping.
- **Step 4** − Run each script when records do not exist in destination tables.
- **Step 5** − Run each script when the records already exist in the destination tables.

# Database Testing – Performance

An application with more response time and poor performance can lead to huge problems. Database Load Testing is used to find any performance issues before you deploy your database applications for end users.

Database Load Testing helps you design database application for performance, reliability and scalability. Load Testing of Database applications involves testing the performance and scalability of your Database application with varying user load.

Database Load testing involves simulating real-life user load for the target Database application. It helps you determine how your Database application behaves when multiple users hits it simultaneously.

## Load Testing

The primary target of Load Testing is to check if most running transactions have performance impact on the database. In load testing, you need to check the following aspects −

- The response time for executing the transactions for multiple remote users should be checked.

- With normal transactions, you should include one editable transaction to check the performance of the database for these type pf transactions.
- With normal transactions, you should include one non-editing transaction to check performance of database for these type of transactions.
- Time taken by database to fetch specific records should be checked.

## Stress Testing

Stress testing is performed to identify the system **breakpoint**. Here the application is loaded in such a way that the system fails at one point. This point is called the breakpoint of the database system. Stress testing is also known as **Fatigue Testing**.

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

The most common stress testing tools are **LoadRunner** and **WinRunner**.

# Database Testing – Tools

There are various tools provided by vendors that can be used to generate Test data, to manage Test data and perform database testing like Load Testing and Regression Testing.

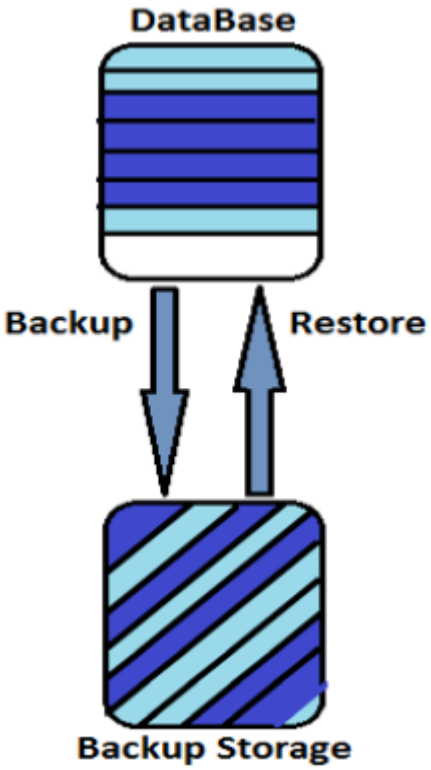
A few common tools that are used are given below.

| Sr.No | Category & Description | Examples |
|---|---|---|
| 1 | **Load Testing Tools**<br>These tools are used to put high usage loads on your database, which enables to determine whether your system's landscape will stand up to your business needs. | Web Performance<br>Rad View<br>Mercury |
| 2 | **Data Security Tools**<br>These tools are used to implement compliance and standards as per the information security regulations. | IBM Optim Data Privacy |
| 3 | **Test Data generator tools** | Data Factory |

| | | |
|---|---|---|
| | A tester uses these tools to generate the test data for a database system. These are mostly required when you have huge amount of data and you need sample to perform DB Testing. It is commonly used for Load and Stress testing. | DTM Data Generator<br>Turbo Data |
| 4 | **Test Data Management Tool**<br>These tools are used to maintain version control for test data. You have to define the expected results and then you compare it with the actual outcomes of the tests. | IBM Optim Test Data Management |
| 5 | **Tools to perform Unit Testing**<br>These tools are used to perform regression testing on your database. | SQLUnit<br>TSQLUnit<br>DBFit<br>DBUnit |

# Database Testing – Backup

The most important part of an organizational growth is its data. In case of a system failure, there is a need to restore the data. Back up is an exact copy of the database, which helps you to restore your data in case of any data loss.



Consider a finance company which has data related to its customers such as A/C number, customer names, credit and debits, duration, etc. How would such an organization deal with the pressure of losing such important information in case of a data failure?

This is the reason you back up the data so that in case of any failure of a disk, disk controller, etc. you can rely on the backup to restore it into the database.

## Types of Data Backups

There are two types of backup that can be used −

- **Physical Backups** − Physical backup includes taking back up using third-party backup tools like Veritas Net Back, IBM Tivoli Manager or user manager backups using OS utilities.
- **Logical Backups** − Logical backup of database includes taking backups of logical objects like tables, indexes, procedures, etc.

**Example** − One of common tool to take data backup is **Oracle Recovery Manager (RMAN)** that is an Oracle utility to take database backup.

**RMAN** consists of two components −

- **Target database** for which backup is required.
- **RMAN** client is used to run commands to take data backup.

**BACKUP VALIDATE** is used to test if you are able to make a valid backup of database files. It ensures −

- If backup is in place for physical or logical objects of database.
- If regular backups are set up for invaluable data.
- If the backup tool meets the backup requirements of an organization.

# Database Testing – Recovery

**Database recovery testing** is used to ensure that the database is recovered. Recovery testing allows you to find out whether the application is running properly and to check retrieving invaluable data that would have been lost if your recovery method is not properly setup.

You also check if several critical processes are running smooth to ensure that the data recovery will pass smoothly through the testing phase.

You can perform the following checks for database recovery −

- Any errors or mistakes in the backup software and you need to resolve these issues at an earlier stage.
- You need to conduct the recovery testing so that you will know what to do in case of an emergency situation.
- You need to check recovery testing needs so that you can plan for an effective recovery strategy.
- You should also know how you can recover the documents.

You need to run the recovery tests in early phase of the project. This allows you to remove and throw away every type of errors from the system. Here is a list of some of important points, which should be considered at the time of testing –

- Time span when changes or modifications occurs in database system.
- The period by which you want your recovery plan conducted.
- The sensitivity of data in database system. More critical the data is, the more regularly you will need to test the software.

## Common Steps in Database Backup and Recovery Testing

In database recovery testing, you need to run the test in the actual environment to check if the system or the data can actually be recovered in case of any disasters and any other unforeseen events in the business environment.

Given below are the common actions performed in Database Recovery Testing –

- Testing of database system
- Testing of the SQL files
- Testing of partial files
- Testing of data backup
- Testing of Backup tool
- Testing log backups

# Database Testing – Security

Database security testing is done to find the loopholes in security mechanisms and also about finding the vulnerabilities or weaknesses of database system.

The main target of database security testing is to find out vulnerabilities in a system and to determine whether its data and resources are protected from potential

intruders. Security testing defines a way to identify potential vulnerabilities effectively, when performed regularly.

Given below are the primary objectives of performing database security testing −

- Authentication
- Authorization
- Confidentiality
- Availability
- Integrity
- Resilience

# Types of Threats on a Database System

## SQL Injection

This is most common type of attack in a database system where malicious SQL statements are inserted in the database system and are executed to get critical information from the database system. This attack takes advantage of loopholes in implementation of user applications. To prevent this, user inputs fields should be carefully handled.

## Privilege Elevation in Database

In this attack, a user already has some access in the database system and he only tries to elevate this access higher level so that he/she can perform some unauthorized activities in database system.

## Denial of Service

In this type of attack, an attacker makes a database system or application resource unavailable to its legitimate users. Applications can also be attacked in ways that render the application, and sometimes the entire machine, unusable.

## Unauthorized Access to data

Another type of attack is gaining unauthorized access to data within an application or database system. Unauthorized access includes −

- Unauthorized access to data via user based applications
- Unauthorized access to by monitoring the access of others
- Unauthorized access to reusable client authentication information

### Identity Spoofing

In Identity Spoofing, a hacker uses the credentials of a user or device to launch attacks against network hosts, steal data or bypass access controls to database system. Preventing this attack requires IT-infrastructure and network-level mitigations.

### Data Manipulation

In a data manipulation attack, a hacker changes data to gain some advantage or to damage the image of database owners.

## Database Security Testing Techniques

### Penetration Testing

A penetration test is an attack on a computer system with the intention of finding security loopholes, potentially gaining access to it, its functionality and data.

### Risk Finding

Risk Finding is a process of assessing and deciding on the risk involved with the type of loss and the possibility of vulnerability occurrence. This is determined within the organization by various interviews, discussions and analysis.

### SQL Injection Test

It involves checking the user inputs in application fields. For example, entering a special character like ',' or ';' in any text box in a user application should not be allowed. When a database error occurs, it means that the user input is inserted in some query, which is then executed by the application. In such a case, the application is vulnerable to SQL injection.

These attacks are a big threat to data as the attackers can get access to important information from the server database. To check SQL injection entry points into your web application, find out code from your code base where direct MySQL queries are executed on the database by accepting some user inputs.

SQL Injection Testing can be performed for Brackets, Commas, and Quotation marks.

### Password Cracking

This is the most important check while performing database system testing. To access critical information, hackers can use a password-cracking tool or can guess a common username/password. These common passwords are easily available on internet and also password cracking tools exist freely.

Therefore, it is necessary to check at the time of testing if the password policy is maintained in the system. In case of any banking and finance applications, there is a need to set a strict password policy on all the critical information database systems.

### Security Audit of Database System

A security audit is a process of evaluating company's security policies at a regular time interval to determine whether necessary standards are followed or not. Various security standards can be followed as per business requirement to define the security policy and then assessment of set policies against those standards can be done.

Example of most common security standards are ISO 27001, BS15999, etc.

## Database Security Testing Tools

There are various system testing tools available in market, which can be used to test OS and application check. Some of the most common tools are discussed below.

### Zed Attack Proxy

It is a penetration-testing tool for finding vulnerabilities in web applications. It is designed to be used by people with a wide range of security experience and as such is ideal for developers and functional testers who are new to penetration testing. It is commonly used for Windows, Linux, Mac OS.

### Paros

All HTTP and HTTPS data between server and client, including cookies and form fields, can be intercepted and modified using these scanners. It is used for Cross-platform, Java JRE/JDK 1.4.2 or above.

### Social Engineer Toolkit

It is an open source tool and human elements are attacked rather than the system element. It enables you to send emails, java applets etc. containing the attack code. It is preferred for Linux, Apple Mac OS X and Microsoft Windows.

### Skipfish

This tool is used to scan their sites for vulnerabilities. Reports generated by the tool are meant to serve as a foundation for professional web application security assessments. It is preferred for Linux, FreeBSD, MacOS X, and Windows.

### Vega

It is an open source, multiplatform web security tool that is used to find instances of SQL injection, cross-site scripting (XSS), and other vulnerabilities in web applications. It is preferred for Java, Linux, and Windows.

### Wapiti

Wapiti is an open source and web-based tool that scans the web pages of the web application and check for scripts and forms where it can inject data. It is built with Python and can detect File handling errors, Database, XSS, LDAP and CRLF injections, Command execution detection.

### Web Scarab

It is written in Java and is used for analyzing the applications that communicate through HTTP/HTTPS protocols. This tool is primarily designed for developers who can write code themselves. This tool is not OS dependent.

# Database Testing – Challenges

To perform database testing successfully, a tester should collect the requirements from all the sources, like technical and functional requirements. There is a possibility that a few requirements are at a high level, so there is a need to breakdown those requirements into the small parts. Testing database is a complex task and the testers face many challenges while performing this testing. Most common database testing challenges are –

## Testing scope is too large

A tester needs to identify the test items in database testing otherwise he may not have a clear understanding of what he would test and what he would not test. Therefore, if you are clear on the requirement, you may waste a lot of time testing uncritical objects in the database.

When you have a list of objects to test, next is to estimate the effort required to design the tests and execute the tests for each test item. Depending on their design and data size, some database tests may take a long time to execute.

As the database size is too large, it becomes a big challenge to find out the objects that have to be tested and those which are to be left out.

## Scaled-down test database

Normally testers are provided with a copy of the development database to test. That database only have little data, which is sufficient to run the application. So there

is a need to test the development, staging and as well as production database system.

## Changes in database structure

This is one of the common challenges in DB testing. Sometimes, it happens that you design or execute a test, and the database structure has been changed at that time. This is necessary that you should be aware of the changes made to the database during testing.

Once the database structure changes, you should analyze the impact of the changes and modify the tests. In addition, if multiple users use the test database, you would not be sure about the test results so you should ensure that the test database is used for testing purpose only.

Another challenge in DB testing is that you run multiple tests at the same time. You should run one test at a time at least for the performance tests. You do not want your database performing multiple tasks and under-reporting performance.

## Complex test plans

The database structure is normally complex and it has huge data, so there is a possibility that you are executing incomplete or same tests repeatedly. So there is a need to create a test plan and proceed accordingly and checking the progress regularly.

## Good understanding of SQL

To test a database, you should have a good knowledge of SQL queries and the required database management tools.

# Database Testing – Interview Questions

What do you understand by DB testing?
Why do we need to perform database testing?
What are the different steps involved in Database Testing?
What are the different categories of DB testing? Explain.
Name a few tools that can be used to test Stored Procedures in a database.
What is a join in SQL?
What are the different types of joins?What is a self-join in SQL?
How can you test an SQL Query in WinRunner?
Explain the steps to test a Stored Procedures in database.
What are the different types of SQL statements?
What are DDL statements in SQL?What is an Operator in SQL?
How many types of operators are there in SQL?
What is the function of Union operator?
What type of operator is this?

What is the difference between Union and Union All?
What is a trigger?
How to check if a trigger is fired or not?
How to invoke a trigger on demand?
How do you write test-cases for Database testing?
How DB testing is different from Front-end testing?
Explain the process of database testing.
Which SQL statements are commonly used to develop test-cases for database testing?
What is a View in database?
How is it related to data independence?
What is VDL (View Definition Language)?
What is normalization?
What is indexing and what are the different kinds of indexing?
Define SQL and how is it different from other conventional programming Languages?
What are stored procedures?
What are the advantages of using them?
What are cursors in PL/SQL?
In Oracle, what is cold backup and hot backup?
What is an SQL subquery?

You have been provided with a set of tables and asked to create a new database to store them.

While checking the data values in the tables, what points to be considered for this?
How do you test if your database is updated when data is entered in front-end application?
What is data-driven testing?What is retesting and how it is different from data driven testing?
What are the types of data driven testing?
What is performance testing?
What are the key points that should be considered while performing database recovery testing?
Name a few tools that are used by a tester to generate test data for a database system.
What are the common types of data backups?
Name the common actions performed in Database recovery testing?
What do you understand by database security testing?
Name a few objectives of Database security testing.What is SQL Injection threat?
Name a few tools that can be used to perform database security testing.
What are the common challenges that you face while performing database testing?