**https://www.youtube.com/watch?v=wtwiPCs7Q0E&t=302s**

**https://www.youtube.com/watch?v=UHcfYoIvpPg**

**https://www.1keydata.com/sql/sql.html**

**http://istqbexamcertification.com/what-is-big-bang-integration-testing/**

**https://www.youtube.com/watch?v=5FUdrBq-WFo**

**Unix commands**

**Directory in unix holds other directories or files**

who am I gives the username who entered the command

users, who, w gives the users name who logged into the computer at the same time

logout logs out and breaks the connection

Ls list the files and directories

Ls-l list the files and directories with more detailed information

Ls-a list all the visible and invisible files in a directory

If the result starts with ‘d’ then it is directory else file

**Ls command example:**

drwxrwxr-x-- 2 (directory)

-rw-rw-r--1 (file)

[r--- read, w---write, x--- execute]

**Ls-l command example:**

drwxrwxr-x-- 2 4096 mar 03 10:00 (directory)

-rw-rw-r--1 4094 mar o3 10:00 (file)

w/out takes you home

$cd~ displays your home page

$cd~username displays the homepage of particular user

$cd- to go to the last directory you visited

$pwd prints the current/present working directory

$ls dirname to list the files in a directory

$mkdir dirname to create the directory in the current directory

$mkdir/temp/mydir creates the directory mydir in the temp directory

$mkdir mydir docs creates two directories mydir and docs in the current directory

$mkdir-p creates parent directory

$rmdir dirname deletes the directory

$cd dirname changes the directory

$cd/user/mydir goes to the mydirectory in the user folder

$mv olddir newdir renames the name of olddir with newdir

. (dot) refers to the current working directory

..(dot dot) refers to the parent directory

Grep searches for a particular word

Hostname prints the machine’s hostname

Finger displays information about users

|  |  |
| --- | --- |
| Command | Function |
| Ls | Lists all files |
| Ls-l | Lists files in long format |
| Ls-a | Lists all visible and invisible files |
| More filename | Shows first part of file that fits in one screen |
| Emacs filename | Editor that creates and edits the file |
| Mv filename1 filename2 | Moves a file into diff. directory or changes file name |
| Cp filename1 filename2 | Copies a file |
| Rm filename | Removes file |
| Diff filename1 filename2 | Compares files |
| Wc filename | Shows the count of lines, words and characters in a file |
| Chmod filename | Changes mode of filename |
| Gzip filename | Zips or compresses file |
| Gunzip filename | Un compresses zip file |
| Gzcat filename | Lets you look at zipped file |
| Mkdir directory name | Cretes directory |
| Cd dirname | Changes directory |
| Pwd | Present/current working directory |
| Ff filename | Finds files anywhere in the system |
| Grep string filename | Finds the particular string in the given filename |
| W | Tells who logged in and what they are doing |
| Who | Tells who logged in and where they are coming from |
| WhoamI | Returns your username |
| Finger username | Gives information about the user |
| Last-l | Tells when the user logged on and off and from where |
| Talk username | Allows to have a typed conversation with other user |
| Write username | Allows to exchange one line messages with other |
| Elm | Allows to send email messages |
| Passwd | Allows to change the password |
| Kill PID | Ends the process with the given ID |
| Du filename | Shows the disk usage of the given filename |
| Last yourusername | Lists your last logins |
| Date | Shows the current date and time |
| Cal | Shows the calender of a current month |
| Lpr | Sends file to printer |
| Df | Shows free disk space |
| Logout | Logs out |
| Gnuplot | Plots data graphically |
| Xcalc | Runs calculator |
| $cd~ | Displays home page |
| Rmdir | Removes directory |
| Mkdir-p | Creates parent directory |
| mv olddir newdir | renames the name of olddir with newdir |
| Hostname | Displays the machines hostname |
|  |  |

**Cartesian product**, also referred to as a cross-join, that returns all the rows in all the tables listed in the query. Each row in the first table is paired with all the rows in the second table. This happens when there is no relationship defined between the two tables.

**Clustered index** is a special type of **index** that re-orders the way records in the table are physically stored. Therefore, a table can have only one **clustered index**.

**Desc table name:** Gives the information about the table columns, their data types, primarykey information etc.

**Three tables inner join query:**

select od.\*, os.\*, pr.\*

from orderdetails od

inner join orders os

on od.orderid=os.orderid

inner join products pr

on pr.productid=od.productid

**Testing**

**Testing** is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. Testing can also be defined as A process of analyzing a software item to detect the differences between existing and required conditions and to evaluate the features of the software item.

Testing can be started from the Requirements Gathering phase and continued till the deployment of the software.

**Testing** involves identifying bug/error/defect in a software without correcting it. Normally professionals with a quality assurance background are involved in bugs identification. Testing is performed in the testing phase.

**Debugging** involves identifying, isolating, and fixing the problems/bugs. Developers who code the software conduct debugging upon encountering an error in the code. Debugging is a part of White Box Testing or Unit Testing. Debugging can be performed in the development phase while conducting Unit Testing or in phases while fixing the reported bugs.

**Manual testing** includes testing a software manually, i.e., without using any automated tool or any script. In this type, the tester takes over the role of an end-user and tests the software to identify any unexpected behavior or bug. There are different stages for manual testing such as

Unit testing,

Integration testing,

System testing, and

User acceptance testing.

Testers use test plans, test cases, or test scenarios to test a software to ensure the completeness of testing. Manual testing also includes exploratory testing, as testers explore the software to identify errors in it.

**Testing Methods**

**Black box testing:** Tester has no knowledge of internal working of applications and no access of source code.

**Grey box testing:** Tester has limited knowledge of internal working of applications and limited access of source code.

**White box testing:** Tester has full knowledge of internal working of applications and full access of source code.

**Testing Levels**

**Functional testing:** It is a type of black-box testing that is based on the specifications of the software that is to be tested. The application is tested by providing input and then the results are examined that need to conform to the functionality it was intended for. Functional testing of a software is conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements.

**Types of functional testing:**

**Unit testing:** This type of testing is performed by developers before the setup is handed over to the testing team to formally execute the test cases. Unit testing is performed by the respective developers on the individual units of source code assigned areas.

**Integration testing:** It is defined as the testing of combined parts of an application to determine if they function correctly. Integration testing can be done in two ways: Bottom-up integration testing and Top-down integration testing. First bottom-up and then top-down testing are performed. Can be performed by either developers or testers.

**System testing:** It tests the system as a whole. Once all the components are integrated, the application as a whole is tested rigorously to see that it meets the specified Quality Standards. This testing is performed by a specialized testing team.

**Regression testing:** Whenever a change in a software application is made, it is quite possible that other areas within the application have been affected by this change. Regression testing is performed to verify that a fixed bug hasn't resulted in another functionality or business rule violation. The intent of regression testing is to ensure that a change, such as a bug fix should not result in another fault being uncovered in the application.

**Acceptance testing:** This is the most important type of testing, as it is conducted by the Quality Assurance Team who will gauge whether the application meets the intended specifications and satisfies the client’s requirement. The QA team will have a set of pre-written scenarios and test cases that will be used to test the application.

**Alpha testing**: First stage of testing and will be performed amongst the teams (developer and QA teams). Unit testing, integration testing and system testing when combined is known as alpha testing.

**Non-Functional Testing:** It is based upon testing an application from its non-functional attributes. Nonfunctional testing involves testing a software from the requirements which are nonfunctional in nature but important such as performance, security, user interface, etc.

**Performance testing:** Identifies the performance issues like speed, capacity, stability etc.

**Load testing:** Is a process of testing the behavior of a software by applying maximum load in terms of software accessing and manipulating large input data. It can be done at both normal and peak load conditions. It identifies the maximum capacity of software and its behavior at peak time.

**Stress testing:** Testing the behavior of a software under abnormal conditions. For example,

Shutdown or restart of network ports randomly, Turning the database on or off, and Running different processes that consume resources such as CPU, memory, server, etc.

**Usability testing**: It is a black-box technique and is used to identify any error(s) and improvements in the software by observing the users through their usage and operation.

**Security testing:** Security testing involves testing a software to identify any flaws and gaps from security and vulnerability point of view. Ex: confidentiality, authentication, and availability etc.

**Portability testing:** It includes testing a software with the aim to ensure its reusability and that it can be moved from another software as well like: Transferring an installed software from one computer to another, and Building executable (.exe) to run the software on different platforms.

**TESTING DOCUMENTATION**

Testing documentation involves the documentation of artifacts that should be developed before or during the testing of Software. Documentation for software testing helps in estimating the testing effort required, test coverage, requirement tracking/tracing, etc. Some of the commonly used documented artifacts related to software testing are: Test Plan, Test Scenario, Test Case, and Traceability Matrix.

A **test case** is a set of conditions or variables under which a tester will determine whether a system under test satisfies requirements or works correctly.

The process of developing test cases can also help find problems in the requirements or design of an application.

**A test case template will have the following conditions:**

Test suite Id, Test case Id, test case summary, related requirement, prerequisites, test procedure, test data, expected results, actual result, status, remarks, created by, date of creation, executed by, date of execution, test environment.

Example:

Test suite id TS01

Test case id TC01

Test case summary To verify whether the student is pass or fail in the exam

Related requirement R01

Pre-requisites 1. Student is authorized

2. student appears the exam

3. student scores are available

4. maximum score to pass the exam should be known

Test-procedure select the score attained by student

Enter the score attained to pass the exam

Compare score to pass the exam with the score attained by student

Test-data valid student id, invalid student id, student score: 0, 5, 30, 50, 100

Expected result no data should be obtained if invalid student id is entered

Fail should be shown in the records if the score less than pass score is entered

Pass should be shown in the records if the score more than or equal to pass score is entered

Actual result if student is valid and score less than pass score is entered then fail shown in records

if student is valid and score more than pass score is entered then pass shown in records

Status Pass

Remarks test case for students score evaluation

Created by Vijji

Date of creation 03.08.2017

Executed by vijji

Date of execution 03.08.2017

Test environment OS: Windows

MSoffice: Word

Browser: Google Chrome

**Test scenario**

Test scenario is any functionality that is to be tested. It is also called as test condition.

It is a one line information that tells us what we need to test. Test scenarios means thinking of test requirements in detail.

**Test Condition**: An item or event of a component or system that could be verified by one or more **test** cases. e.g., a function, transaction, feature, quality attribute or structural element.

**Software Testing Life Cycle (STLC)**

STLC is roughly divided into three parts:

1. Test plan
2. Test design
3. Test execution

**Test plan**: A document describing the scope, approach, resources and schedule of intended test activities. It identifies amongst others test items, the features to be tested, the testing tasks, who will do each task, degree of tester independence, the test environment, the test design techniques and entry and exit criteria to be used, and the rationale for their choice, and any risks requiring contingency planning. It is a record of the test planning process.

**master test plan**: A test plan that typically addresses multiple test levels.

**phase test plan**: A test plan that typically addresses one test phase.

**Regression testing**

**Regression testing** is a type of software testing that intends to ensure that changes (enhancements or defect fixes) to the software have not adversely affected it.

The likelihood of any code change impacting functionalities that are not directly associated with the code is always there and it is essential that regression testing is conducted to make sure that fixing one thing has not broken another thing. During regression testing, new test cases are not created but previously created test cases are re-executed.

**LEVELS APPLICABLE TO:** Regression testing can be performed during any level of testing (Unit, Integration, System, or Acceptance) but it is mostly relevant during System Testing.

**EXTENT:** In an ideal case, a full regression test is desirable but oftentimes there are time/resource constraints. In such cases, it is essential to do an impact analysis of the changes to identify areas of the software that have the highest probability of being affected by the change and that have the highest impact to users in case of malfunction and focus testing around those areas.

**Integration Testing** is a [level of software testing](http://softwaretestingfundamentals.com/software-testing-levels/) where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. **Test drivers** and **test stubs** are used to assist in Integration Testing.

* **Integration testing:**Testing performed to expose defects in the interfaces and in the  
  interactions between integrated components or systems. See also c*omponent integration  
  testing, system integration testing*.
* **Component integration testing:**Testing performed to expose defects in the interfaces and  
  interaction between integrated components.
* **System integration testing:**Testing the integration of systems and packages; testing  
  interfaces to external organizations (e.g. Electronic Data Interchange, Internet).

During the process of manufacturing a ballpoint pen, the cap, the body, the tail and clip, the ink cartridge and the ballpoint are produced separately and unit tested separately. When two or more units are ready, they are assembled and Integration Testing is performed. For example, whether the cap fits into the body or not.

**METHOD:** Any of [Black Box Testing](http://softwaretestingfundamentals.com/black-box-testing/), [White Box Testing](http://softwaretestingfundamentals.com/white-box-testing/), and [Gray Box Testing](http://softwaretestingfundamentals.com/gray-box-testing/) methods can be used. Normally, the method depends on your definition of ‘unit’.

**TASKS**

Integration Test Plan: Prepare, Review, Rework, Baseline

Integration Test Cases/Scripts: Prepare, Review, Rework, Baseline, Integration Test, Perform

**When is Integration Testing performed:** Integration Testing is performed after [Unit Testing](http://softwaretestingfundamentals.com/unit-testing/) and before [System Testing](http://softwaretestingfundamentals.com/system-testing/).

**Who performs Integration Testing:** Either Developers themselves or independent Testers perform Integration Testing.

**APPROACHES**

* ***Big Bang*** is an approach to Integration Testing where all or most of the units are combined together and tested at one go. This approach is taken when the testing team receives the entire software in a bundle. Big Bang Integration Testing tests only the interactions between the units and System Testing tests the entire system.
* ***Top Down*** is an approach to Integration Testing where top level units are tested first and lower level units are tested step by step after that. This approach is taken when top down development approach is followed. **Test Stubs** are needed to simulate lower level units which may not be available during the initial phases.
* ***Bottom Up*** is an approach to Integration Testing where bottom level units are tested first and upper level units step by step after that. This approach is taken when bottom up development approach is followed. **Test Drivers** are needed to simulate higher level units which may not be available during the initial phases.
* ***Sandwich/Hybrid*** is an approach to Integration Testing which is a combination of Top Down and Bottom Up approaches.
* Tips: Ensure that you have a proper Detail Design document where interactions between each unit are clearly defined. In fact, you will not be able to perform Integration Testing without this information.
* Ensure that you have a robust Software Configuration Management system in place. Or else, you will have a tough time tracking the right version of each unit, especially if the number of units to be integrated is huge.
* Make sure that each unit is first unit tested before you start Integration Testing.
* As far as possible, automate your tests, especially when you use the Top Down or Bottom Up approach, since regression testing is important each time you integrate a unit, and manual regression testing can be inefficient.

**Test approach** is the test strategy implementation of a project, defines how testing would be carried out. **Test approach has two techniques:**

* **Proactive -**An approach in which the test design process is initiated as early as possible in order to find and fix the defects before the build is created.
* **Reactive -**An approach in which the testing is not started until after design and coding are completed.

**Types of test approach:**

* Dynamic and heuristic approaches
* Consultative approaches
* Model-based approach that uses statistical information about failure rates.
* Approaches based on risk-based testing where the entire development takes place based on the risk
* Methodical approach, which is based on failures.
* Standard-compliant approach specified by industry-specific standards.

**Factors to be considered**

Risks of product or risk of failure or the environment and the company.

* Expertise and experience of the people in the proposed tools and techniques.
* Regulatory and legal aspects, such as external and internal regulations of the development process.
* The nature of the product and the domain.

D**efect is an error or a bug**, in the application which is created. A programmer while designing and building the software can make mistakes or errors. These mistakes or errors means that there are flaws in the software. These are called defects.

When actual result deviates from the expected result while testing a software application or product then it results into a defect. Hence, any deviation from the specification mentioned in the product functional specification document is a defect. In different organizations, it’s called differently like **bug, issue, incidents or problem.** These defects or bugs occur because of an error in logic or in coding which results into the [**failure**](http://istqbexamcertification.com/what-is-a-failure-in-software-testing/)or unpredicted or unanticipated results.

While testing a software application or product if large number of defects are found then it’s called **Buggy.** When a tester finds a bug, or defect it’s required to convey the same to the developers. Thus, they report bugs with the detail steps and are called as Bug Reports, issue report, problem report, etc.

This Defect report or Bug report consists of the following information:

* **Defect ID** – Every bug or defect has its unique identification number
* **Defect Description** – This includes the abstract of the issue.
* **Product Version** – This includes the product version of the application in which the defect is found.
* **Detail Steps** – This includes the detailed steps of the issue with the screenshots attached so that developers can recreate it.
* **Date Raised** – This includes the Date when the bug is reported
* **Reported By** – This includes the details of the tester who reported the bug like Name and ID
* **Status** – This field includes the Status of the defect like New, Assigned, Open, [**Retest**](http://istqbexamcertification.com/what-is-retesting/), [**Verification**](http://istqbexamcertification.com/what-is-verification-in-software-testing-or-what-is-software-verification/), Closed, Failed, Deferred, etc.
* **Fixed by** – This field includes the details of the developer who fixed it like Name and ID
* **Date Closed** – This includes the Date when the bug is closed
* **Severity –**Based on the severity (Critical, Major or Minor) it tells us about impact of the defect or bug in the software application
* **Priority –**Based on the Priority set (High/Medium/Low) the order of fixing the defect can be made. (Know more about [**Severity and Priority**](http://istqbexamcertification.com/what-is-the-difference-between-severity-and-priority/))

**Differences between delete and truncate**

Delete is a DML command and Truncate is a DDL command. Delete is slower than truncate. When we use delete command to delete the data from the database, it stores the whole data in rollback gap where we can get the data back after deleting the data. Whereas using truncate will not store the data in rollback gap and will rub it.

While using delete statement we can use where clause.

If a table is referenced by foreign key then truncate wont work.

**Difference between the primary key and unique key**

* A primary key cannot have a null value. Whereas a unique key can have one null value.
* A table has only one primary key. But a table can have ‘n’ number of unique keys.
* A primary key creates clustered index automatically whereas a unique creates a non-clustered index.
* Both the primary and unique keys contain distinct values only.

**Foreign key**: It identifies uniquely each row of another table and this key it points out to the primary key of other table.

**View**: views are like virtual tables. They contain the rows and columns same as tables. A view can be created from a table. Views are not virtually present and it takes less space to store.

Join: join statement is used to combine the data or rows from two or more tables in a database with a common field between them.

**Index**: it is a data structure used to improve the speed of data retrieval from the database. Data can only be stored in one order. These indexes occupy extra space in disk but allow faster search.

**Table**: A table is a set of organized data. A table consists of rows and columns. Columns are vertical, and rows are horizontal. A table has a specified number of columns called **fields** and ‘n’ number of rows called **records**.

**Self join:** Used to join a table to itself as like it is a second table.

**SET OPERATORS:** union, intersect and minus operators are called set operators.

**BETWEEN AND IN DIFFERENCE:**

Between is used to display the rows in the range of values and in is used to display the data in specific set of values.

**ACID PROPERTIES:**

**Atomicity**: if one part of the transaction fails then, the entire transaction fails by leaving the database state unchanged.

**Consistency**: it ensures that the transaction meets all the validation rules. Transaction never leaves the database without completing its state.

**Isolation**: concurrent property of isolation should not be met. Main goal is to provide concurrency control.

**Durability**: if once the transaction is committed then it will remain so.

**TRIGGER**:

Trigger is used to execute the batch of sql code when an insert, update or delete statements are executed against a specific table.

Usually triggers are a special type of stored procedures that are defined to execute automatically in place or after data modifications.

**Null value** is unassigned or unavailable. **Zero** is a number and **blank** space is a character.

**Even records selection from table:**

select \* from

(select country\_id, country\_name, rownum rn

from hr.countries

order by country\_id)

where mod (rn, 2) <> 0;

**Odd records selection from table:**

select \* from

(select country\_id, country\_name, rownum rn

from hr.countries

order by country\_id)

where mod (rn, 2) <> 1;

**To select the columns according to the row number**

**Between two values**

select region\_id, country\_name from

(select rownum rn, region\_id, country\_name from hr.countries)

where rn >5 and rn <10

**Greater than particular value:**

select region\_id, country\_name from

(select rownum r, region\_id, country\_name from hr.countries)

where r > 15

**Less than a particular value:**

Select \* from table\_name where rownum < 10;

**To find the nth highest salary in a table:**

select \* from (select e.\*, rank () over (order by salary desc) as rank from Employee e ) where rank = 2;

select \* from (select e.\*, rank () over (order by salary desc) as rank from Employee e ) where rank = 4;

select \* from (select e.\*, rank () over (order by salary desc) as rank from Employee e ) where rank in (1, 4, 8);

select \* from (select e.\*, rank () over (order by salary desc) as rank from Employee e ) where rank between 2 and 10;