Assignment_M3

1. What is RDBMS?

Ans :- The software used to store, manage, query, and retrieve data stored in a relational database is called a relational database management system RDBMS. The RDBMS provides an interface between users and applications and the database, as well as administrative functions for managing data storage, access, and performance.

2. What is SQL?

Ans: - SQL is the standard language for dealing with Relational Databases. SQL can be used to insert, search, update, and delete database records. SQL can do lots of other operations, including optimizing and maintenance of databases.

Structured query language (SQL) is a popular query language that is frequently used in all types of applications. Data analysts and developers learn and use SQL because it integrates well with different programming languages. For example, they can embed SQL queries with the Java programming language to build high-performing data processing applications with major SQL database systems such as Oracle or MS SQL Server. SQL is also fairly easy to learn as it uses common English keywords in its statements

SQL was invented in the 1970s based on the relational data model. It was initially known as the structured English query language (SEQUEL). The term was later shortened to SQL. Oracle, formerly known as Relational Software, became the first vendor to offer a commercial SQL relational database management system.

Relational database management systems use structured query language (SQL) to store and manage data. The system stores multiple database tables that relate to each other. MS SQL Server, MySQL, or MS Access are examples of relational database management systems. The following are the components of such a system.

SQL table

A SQL table is the basic element of a relational database. The SQL database table consists of rows and columns. Database engineers create relationships between multiple database tables to optimize data storage space.

SQL statements

SQL statements, or SQL queries, are valid instructions that relational database management systems understand. Software developers build SQL statements by using different SQL language elements. SQL language elements are components such as identifiers, variables, and search conditions that form a correct SQL statement.

For example, the following SQL statement uses a SQL INSERT command to store Mattress Brand A, priced $499, into a table named Mattress table, with column names brand-named and cost.

Stored procedures

Stored procedures are a collection of one or more SQL statements stored in the relational database. Software developers use stored procedures to improve efficiency and performance. For example, they can create a stored procedure for updating sales tables instead of writing the same SQL statement in different applications.

Structured query language (SQL) implementation involves a server machine that processes the database queries and returns the results. The SQL process goes through several software components, including the following.

Parser

The parser starts by tokenizing, or replacing, some of the words in the SQL statement with special symbols. It then checks the statement for the following:

Correctness

The parser verifies that the SQL statement conforms to SQL semantics, or rules, that ensure the correctness of the query statement. For example, the parser checks if the SQL command ends with a semi-colon. If the semi-colon is missing, the parser returns an error.

Authorization

The parser also validates that the user running the query has the necessary authorization to manipulate the respective data. For example, only admin users might have the right to delete data.

Relational engine

The relational engine, or query processor, creates a plan for retrieving, writing, or updating the corresponding data in the most effective manner. For example, it checks for similar queries, reuses previous data manipulation methods, or creates a new one. It writes the plan in an intermediate-level representation of the SQL statement called byte code. Relational databases use byte code to efficiently perform database searches and modifications.

Storage engine

The storage engine, or database engine, is the software component that processes the byte code and runs the intended SQL statement. It reads and stores the data in the database files on physical disk storage. Upon completion, the storage engine returns the result to the requesting application.

3. Write SQL Command?

Ans: -

Structured query language (SQL) commands are specific keywords or SQL statements that developers use to manipulate the data stored in a relational database. You can categorize SQL commands as follows.

Data definition language

Data definition language (DDL) refers to SQL commands that design the database structure. Database engineers use DDL to create and modify database objects based on the business requirements. For example, the database engineer uses the CREATE command to create database objects such as tables, views, and indexes.

Data query language

Data query language (DQL) consists of instructions for retrieving data stored in relational databases. Software applications use the SELECT command to filter and return specific results from a SQL table.

Data manipulation language

Data manipulation language (DML) statements write new information or modify existing records in a relational database. For example, an application uses the INSERT command to store a new record in the database.

Data control language

Database administrators use data control language (DCL) to manage or authorize database access for other users. For example, they can use the GRANT command to permit certain applications to manipulate one or more tables.

Transaction control language

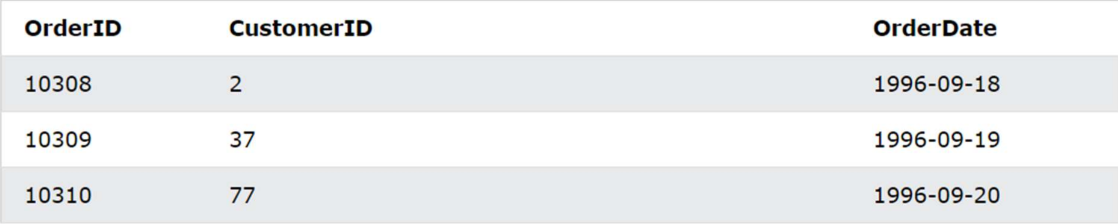
The relational engine uses transaction control language (TCL) to automatically make database changes. For example, the database uses the ROLLBACK command to undo an erroneous transaction.

4.What is join?

Ans: -

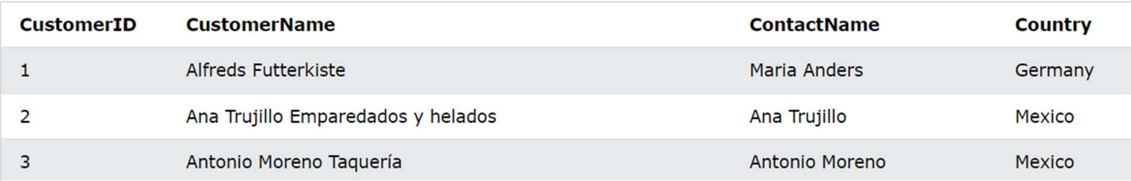
A JOIN is used to combine rows from two or more tables, based on a related column between them.

Let's look at a selection from the "Orders" table:



OrderID	CustomerID	OrderDate
10308	2	1996-09-18
10309	37	1996-09-19
10310	77	1996-09-20

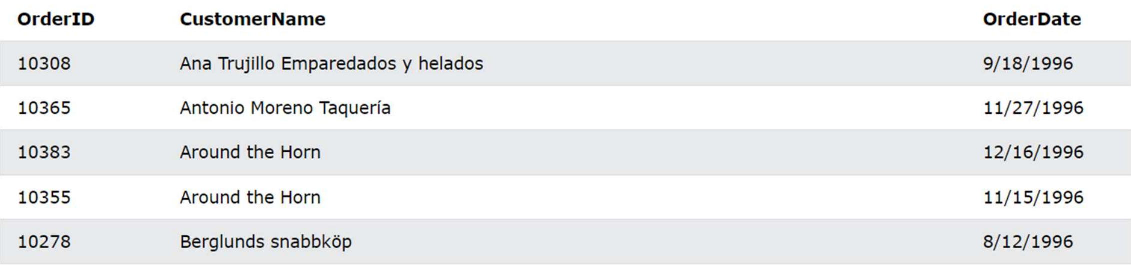
Then, look at a selection from the "Customers" table:



CustomerID	CustomerName	ContactName	Country
1	Alfreds Futterkiste	Maria Anders	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mexico

Notice that the "Customer" column in the "Orders" table refers to the "CustomerID" in the "Customers" table. The relationship between the two tables above is the "CustomerID" column.

Then, we can create the following SQL statement (that contains an INNER JOIN), that selects records that have matching values in both tables:



OrderID	CustomerName	OrderDate
10308	Ana Trujillo Emparedados y helados	9/18/1996
10365	Antonio Moreno Taquería	11/27/1996
10383	Around the Horn	12/16/1996
10355	Around the Horn	11/15/1996
10278	Berglunds snabbköp	8/12/1996

5. Write types of join?

Ans: - There are four types of join types given below

(INNER) JOIN: Returns records that have matching values in both tables

LEFT (OUTER) JOIN: Returns all records from the left table, and the matched records from the right table

RIGHT (OUTER) JOIN: Returns all records from the right table, and the matched records from the left table

FULL (OUTER) JOIN: Returns all records when there is a match in either left or right table

6. How Many constraints and describes it self

Ans: -

There are six main constraints that are commonly used in SQL Server that we will describe deeply with examples within this article and the next one. These constraints are:

- SQL NOT NULL
- UNIQUE
- PRIMARY KEY
- FOREIGN KEY
- CHECK
- DEFAULT

NOT NULL Constraint in SQL

By default, the columns are able to hold NULL values. A NOT NULL constraint in SQL is used to prevent inserting NULL values into the specified column, considering it as a not accepted value for that column. This means that you should provide a valid SQL NOT NULL value to that column in the INSERT or UPDATE statements, as the column will always contain data.

Assume that we have the below simple CREATE TABLE statement that is used to define the ConstraintDemo1 table. This table contains only two columns, ID and Name. In the ID column definition statement, the SQL NOT NULL column-level constraint is enforced, considering the ID column as a mandatory column that should be provided with a valid SQL NOT NULL value. The case is different for the Name column that can be ignored in the INSERT statement, with the ability to provide it with NULL value. If the null-ability is not specified while defining the column, it will accept the NULL value by default:

UNIQUE Constraints in SQL

The UNIQUE constraint in SQL is used to ensure that no duplicate values will be inserted into a specific column or combination of columns that are participating in the UNIQUE constraint and not part of the PRIMARY KEY. In other words, the index that is automatically created when you define a UNIQUE constraint will guarantee that no two rows in that table can have the same value for the columns participating in that index, with the ability to insert only one unique NULL value to these columns, if the column allows NULL.

Let us create a small table with two columns, ID and Name. The ID column cannot hold duplicate values due to the UNIQUE constraint specified with the column definition.

SQL PRIMARY KEY Constraint

The PRIMARY KEY constraint consists of one column or multiple columns with values that uniquely identify each row in the table.

The SQL PRIMARY KEY constraint combines between the UNIQUE and SQL NOT NULL constraints, where the column or set of columns that are participating in the PRIMARY KEY cannot accept a NULL value. If the PRIMARY KEY is defined in multiple columns, you can insert duplicate values on each column individually, but the combination values of all PRIMARY KEY columns must be unique. Take into consideration that you can define only one PRIMARY KEY per each table, and it is recommended to use small or INT columns in the PRIMARY KEY.

In addition to providing fast access to the table data, the index that is automatically created, when defining the SQL PRIMARY KEY, will enforce the data uniqueness. The PRIMARY KEY is used mainly to enforce the entity integrity of the table. Entity integrity ensures that each row in the table is a uniquely identifiable entity.

PRIMARY KEY constraint differs from the UNIQUE constraint in that; you can create multiple UNIQUE constraints in a table, with the ability to define only one SQL PRIMARY KEY per each table. Another difference is that the UNIQUE constraint allows for one NULL value, but the PRIMARY KEY does not allow NULL values.

Assume that we have the below simple table with two columns; the ID and Name. The ID column is defined as a PRIMARY KEY for that table, that is used to identify each row on that table by ensuring that no NULL or duplicate values will be inserted to that ID column.

FOREIGN KEY Constraint

A Foreign Key is a database key that is used to link two tables together. The FOREIGN KEY constraint identifies the relationships between the database tables by referencing a column, or set of columns, in the Child table that contains the foreign key, to the PRIMARY KEY column or set of columns, in the Parent table.

The relationship between the child and the parent tables is maintained by checking the existence of the child table FOREIGN KEY values in the referenced parent table's PRIMARY KEY before inserting these values into the child table. In this way, the FOREIGN KEY constraint, in the child table that references the PRIMARY KEY in the parent table, will enforce database referential integrity. Referential integrity ensures that the relationship between the database tables is preserved during the data insertion process. Recall that the PRIMARY KEY constraint guarantees that no NULL or duplicate values for the selected column or columns will be inserted into that table, enforcing the entity integrity for that table. The entity integrity enforced by the PRIMARY KEY and the referential integrity enforced by the FOREIGN KEY together form the key integrity.

The FOREIGN KEY constraint differs from the PRIMARY KEY constraint in that, you can create only one PRIMARY KEY per each table, with the ability to create multiple FOREIGN KEY constraints in each table by referencing multiple parent table. Another difference is that the FOREIGN KEY allows inserting NULL values if there is no NOT NULL constraint defined on this key, but the PRIMARY KEY does not accept NULLs.

The FOREIGN KEY constraint provides you also with the ability to control what action will be taken when the referenced value in the parent table is updated or deleted, using the ON UPDATE and ON DELETE clauses.

CHECK Constraint

A CHECK constraint is defined on a column or set of columns to limit the range of values, that can be inserted into these columns, using a predefined condition. The CHECK constraint comes into action to evaluate the inserted or modified values, where the value that satisfies the condition will be inserted into the table, otherwise, the insert operation will be discarded. It is allowed to specify multiple CHECK constraints for the same column.

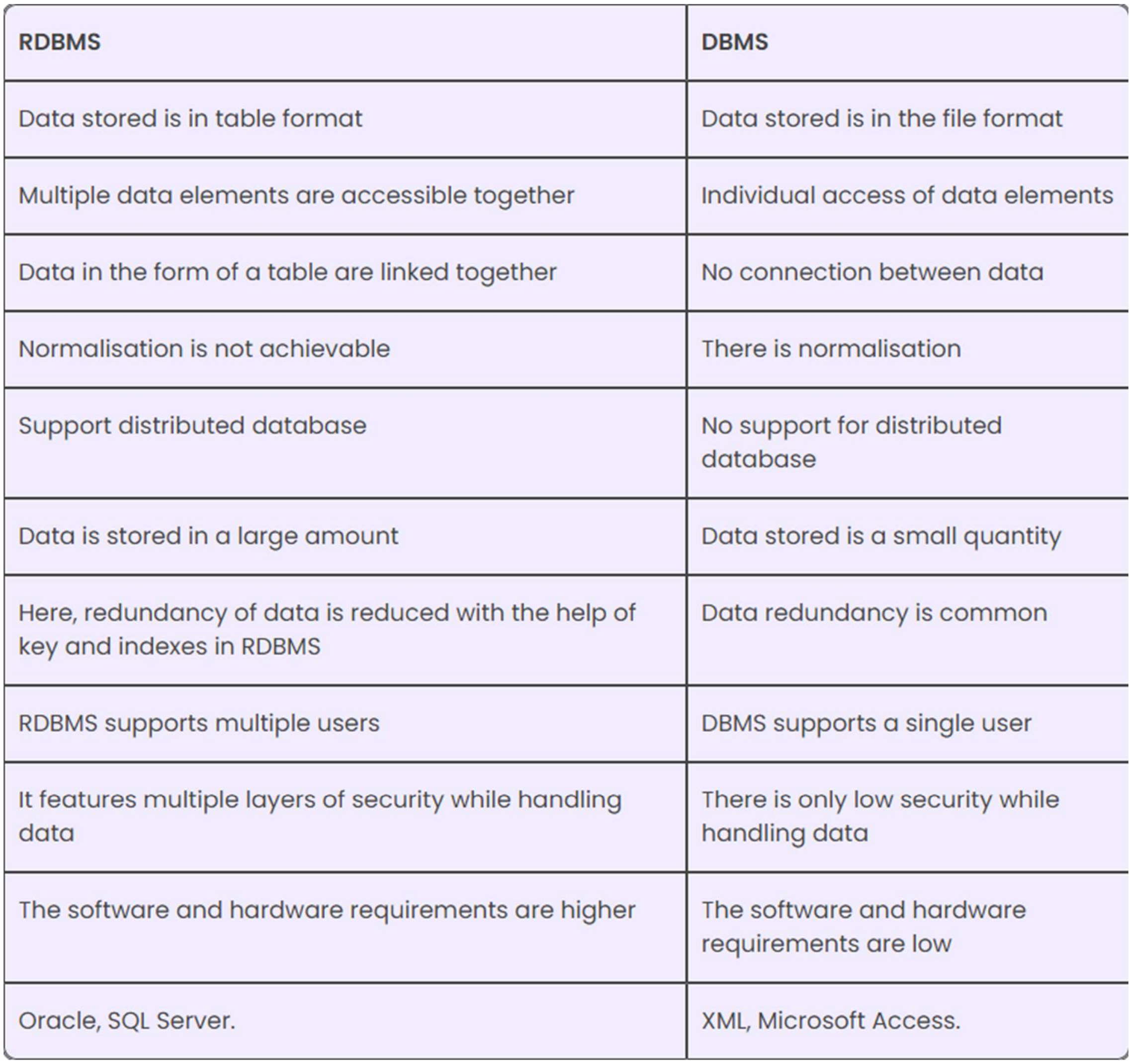
Defining the CHECK constraint condition is somehow similar to writing the WHERE clause of a query, using the different comparison operators, such as AND, OR, BETWEEN, IN, LIKE and IS NULL to write its Boolean expression that will return TRUE, FALSE or UNKNOWN. The CHECK constraint will return UNKNOWN value when a NULL value is present in the condition. The CHECK constraint is used mainly to enforce the domain integrity by limiting the inserted values to the ones that follow the defined values, range or format rules.

DEFAULT Constraint

A DEFAULT constraint is used to provide a default column value for the inserted rows if no value is specified for that column in the INSERT statement. The Default constraint helps in maintaining the domain integrity by providing proper values for the column, in case the user does not provide a value for it. The default value can be a constant value, a system function value or NULL.

7. Difference between RDBMS vs DBMS?

Ans: - DBMS stands for Database Management System, and RDBMS is the acronym for the Relational Database Management system. In DBMS, the data is stored as a file, whereas in RDBMS, data is stored in the form of tables. To know what is the difference between RDBMS and DBMS, check out the table below.



RDBMS	DBMS
Data stored is in table format	Data stored is in the file format
Multiple data elements are accessible together	Individual access of data elements
Data in the form of a table are linked together	No connection between data
Normalisation is not achievable	There is normalisation
Support distributed database	No support for distributed database
Data is stored in a large amount	Data stored is a small quantity
Here, redundancy of data is reduced with the help of key and indexes in RDBMS	Data redundancy is common
RDBMS supports multiple users	DBMS supports a single user
It features multiple layers of security while handling data	There is only low security while handling data
The software and hardware requirements are higher	The software and hardware requirements are low
Oracle, SQL Server.	XML, Microsoft Access.

8. What is API Testing?

Ans: -

API testing is a type of software testing that analyses an application program interface (API) to verify that it fulfils its expected functionality, security, performance and reliability. The tests are performed either directly on the API or as part of integration testing.

An API is code that enables the communication exchange of data between two software programs. An application typically consists of multiple layers, including an API layer. API layers focus on the business logic in applications, defining requests such as how to make them and the data formats used.

As opposed to user interface (UI) testing, which focuses on validating the application's look and feel, API testing focuses on analysing the application's business logic as well as security and data responses. An API test is generally performed by making requests to one or more API endpoints and comparing the responses with expected results.

9. Types of API Testing?

Ans: -

API testing involves verifying the API's functionality, reliability, performance, and security. This type of testing typically involves sending requests to the API and checking the responses to ensure that they match the expected results. API testing can be automated using specialized software tools, which can help to save time and reduce the risk of human error.

Some of the key benefits of API testing include improved quality, reduced development costs, and faster time-to-market. By detecting issues at the earliest stages of the development process, API testing can help to prevent costly errors and delays down the line. Additionally, API testing can help to ensure that the API is secure and meets regulatory compliance requirements.

There is total 9 types of API testing:

1. Validation Testing

This type of testing ensures that the API is returning the expected results and in the correct format. Validation testing involves checking that the input parameters, output format, response code, and data type are correct.

2. UI Testing

UI testing validates that the API works correctly within the application's user interface. This type of testing ensures that the UI is accurately reflecting the API's results and that the API is handling the UI's inputs correctly.

3. Functional Testing

Functional testing verifies that the API functions correctly and meets the required specifications. This type of testing can include testing the API's business logic, input validation, output validation, and error handling.

4. Load Testing

Load testing involves testing the API's performance and stability under stressful conditions. This type of testing simulates high traffic and heavy usage scenarios to ensure that the API can handle a large number of concurrent users and requests.

5. Runtime and Error Detection

This type of testing ensures that the API can handle runtime errors and exceptions. This includes testing for network timeouts, memory leaks, incorrect input parameters, and other errors that can occur during runtime.

6. Penetration Testing

Penetration testing is a type of security testing that involves simulating attacks from hackers to detect vulnerabilities and weaknesses in the API. This type of testing can include network scanning, vulnerability scanning, and manual penetration testing.

7. API Hacking

API hacking is security testing techniques that exploits vulnerabilities in an API. Attackers (and testers) can target API endpoints to gain access to data, disrupt services, or hijack the entire system. Ethical hackers can train by attacking intentionally vulnerable APIs, which can be downloaded from the Internet. Then, they can turn to the organization's own APIs to test their resilience and find weaknesses.

8. Security Testing

Security testing aims to identify security-related vulnerabilities and flaws in the API and ensure that the API meets the required security standards. This type of testing includes testing for vulnerabilities such as SQL injection, cross-site scripting (XSS), cross-site request forgery (CSRF), and others.

9. Fuzz Testing

Fuzz testing involves feeding unexpected and invalid inputs into the API to test its ability to handle unexpected input and recover from errors. This type of testing can uncover security vulnerabilities or unexpected behaviour in the API.

10. What is Responsive Testing?

Ans: -

Responsive website testing ensures that users have the best experience with your site, regardless of their device. The goal of testing responsive websites is to ensure a seamless experience across different digital devices. In this day and age, we live in a world where technology has enabled convenience, and we are now dependent on our devices to function.

Because of the growing market for mobile devices, businesses are developing strategies to create user-friendly websites. They use mobile-first design, progressive web apps, single-page applications, and more. However, for a unified user experience across devices and platforms, we need to consider screen resolutions and device capabilities.

Responsive website testing is a process that ensures your website works well on multiple devices by using CSS media queries based on the user's device where the website is accessed.

In simpler terms, responsive testing is a process that enables you to check how well a website works on various types of devices, including desktops and smartphones. A website that responds well to all screen sizes and resolutions gives your business a competitive edge over other companies.

Responsive website testing is a process that ensures your website works well on multiple devices by using CSS media queries based on the user's device where the website is accessed.

In simpler terms, responsive testing is a process that enables you to check how well a website works on various types of devices, including desktops and smartphones. A website that responds well to all screen sizes and resolutions gives your business a competitive edge over other companies.

Responsive testing of web apps is crucial at every stage of development to ensure that the end-user requirements are met. Here are the following reasons highlighting the importance to test responsive websites:

Plethora of Devices, OS & Browsers: To ensure that your site's content is available to all visitors, verification of the content needs to be done for different screen-sized mobiles, operating systems, and browsers. While a site designed in one browser may appear as intended in another browser, it should not be assumed that this is necessarily the case.

Need for Robustness: It is crucial to ensure that the website is loading at the same speed on different devices and browsers so that users do not become frustrated by lagging or timed-out content. If a website loads slowly or doesn't display correctly on mobile devices, users will have a poor experience. Therefore, testing a website's performance is essential for ensuring users have a positive experience on mobile-responsive websites.

Website Navigation: When testing a mobile website, one of the most common defects found is that pages don't load as expected when navigated among the site's links. It also happens that links are missing, images are not loaded, or timed out while playing with navigation.

Multiple Images and Videos: When creating a responsive website, it is essential to test whether all types of images and videos are displayed as expected on different phones, browsers, etc. Sometimes some videos play well on Android, but they don’t even load on iOS, or some images appear broken on some versions of a mobile operating system while they are perfect on others. Such issues will give a terrible impression if testing is not done correctly.

11. Which types of tools are available for Responsive Testing?

Ans: -

Testing responsive design for websites is important to ensure that your site will look and behave the way you want on different devices. Every modern web designer will know the principles of responsive web design, which aims to make sites render perfectly on whatever device or screen size they appear on. (see our pro’s guide to responsive web design for a refresher).

But theory is one thing, and practice is another. To ensure your site is fully responsive, you need to test it on different device. Most of us don’t have the budget to acquire the hundreds of physical devices needed to carry out a real-world test, but, happily, there are other solutions. The website testing tools below allow you to test responsive design in a virtual environment.

01. Google Chrome Inspect

A screengrab from Google Chrome Inspect, one of the best website responsiveness testing tools

One of the quickest and easiest website responsiveness tests is available directly in your browser if you're using Chrome. As well as allowing you to inspect code, the Chrome Inspect tool allows you to virtually view a site on different screen sizes. Just right-click a website and click ‘Inspect’. When the window opens, look for the device’s icon to the left of the 'Elements' button. You can input a specific size or drag the corner of the window to manually choose the breakpoint.

02. **Responsinator**

See what your site looks like in different viewports with Responsinator
The beauty of Responsinator lies in its simplicity. Just type in your web page's URL and this free, browser-based tool shows you how your web page renders in the most popular screen shapes and sizes. Brilliantly, you can then interact with your page, clicking on links, typing into search fields and so on. Note that these are generic devices, though, not specific ones.

03. Google DevTools Device Mode

DevTools Device Mode emulates different devices within Chrome
DevTools' Device Mode offers an easy way for developers to simulate mobile devices within the Chrome browser. You can use it to learn how your site appears across different screen sizes and resolutions, including Retina screens. You can even simulate device inputs for touch, geolocation and device orientation within the emulator.

04. Browser Stack

Browser Stack is a paid-for testing tool aimed at enterprise

Browser Stack is one of the most advanced, full-featured testing tools around. The paid-for app offers access to more than 1,000 mobile and desktop browsers for testing purposes, a list which is continually being updated, based on market trends and usage statistics based on Browser Stack's 36,000 customers. With users including Twitter, Microsoft, AirBnB and Mastercard, it's obviously doing something right.

05. Cross Browser Testing

CrossBrowserTesting offers a huge array of real-world devices and testing features

Browser Stack's biggest rival in the testing space is CrossBrowserTesting, which offers more than 1,500 browsers and devices to test your responsive website on. Its all-in-one platform allows you to run parallel automated tests, compare screenshots visually, swipe and interact with your website on real-world devices, and remotely debug your code as you go. It offers a free trial.

06. Test sigma

Finally, Test sigma is another paid-for browser-based option. This cross browser and responsiveness testing tool provides automated responsiveness testing for more than 1,000 browser OS and 2,000 iOS and Android devices. It boasts features like script less automation testing, parallel testing for large test suites and drill-down test reports and integration with popular CI/CD and bug reporting tools. Customer support is also reported to be good. Again, there is a free trial.

Note that there are aspects of mobile devices that virtual tools cannot simulate, such as the architecture of mobile CPUs. If in doubt, it's best to actually run your page on a mobile device if possible. Remote Debugging can be used to view, change, debug, and profile a page's code from your laptop or desktop while it runs on a mobile. For more tools, see our pick of the best web design software.

12. What is the full form of .ipa, .apk?

Ans:- APK: Android Application Package

IPA: iOS App Store Package

13. How to create step for to open the developer option mode ON?

Ans:- 1.Go to Settings > About phone.

2.Scroll down to Build number.

3.Tap Build number seven times. After the first few taps, you should see the steps counting down until you unlock the developer options. You may also have to tap in your PIN for verification.