Axis Bank Interview Questions

About Axis Bank

Axis Bank, India's third-largest private sector bank, offers a wide range of financial services to a wide range of clients, including small and medium-sized businesses, large and mid-sized corporations, and Agricultural and Retail businesses.

Whether graduates wishing to break into the banking industry, engineers eager to be part of Axis Bank's growth story, technology enthusiasts eager to explore all things digital, or data scientists eager to solve problems with data, Axis Bank recruits all talent and offers greater growth opportunities.

Having the opportunity to showcase your unique skills and perspectives to the world is an exciting experience. It also offers a welcoming infrastructure and room to grow, making the most of learning curves and grabbing exciting opportunities. By now, you must be wondering what you should do to prepare for the interview. Do not fret, we have covered everything for you.

In this article, we scoured the internet to pull together comprehensive lists of the most frequently asked Axis Bank Interview Questions and answers for both freshers and experienced professionals. Also included is the Axis bank recruitment process and tips on acing the interview. We will walk you through every aspect of the recruitment process that you must know to impress the recruiters at Axis Bank. Taking the time to prepare and evaluate your answers beforehand is the best way to ensure you have the best shot at the Axis Bank interview.

Let's look at the hiring process at Axis Bank.

In this article

1. Axis Bank Recruitment Process 2. Axis Bank Technical Interview Questions: Freshers and Experienced 3. Axis Bank Interview Preparation 4. Frequently Asked Questions

Axis Bank Recruitment Process

Interview Process

Axis Bank's recruitment process is straightforward and very smooth. Below, you will find details regarding Axis Bank's hiring process and how the recruitment team conducts interviews.

Companies can vary in how they interview candidates, depending on their policies and procedures. However, in general, most interviews follow a similar format. For almost all off-campus recruitment processes, you will have to register online and fill out an online application with your resume and cover letter for the job position you are interested in applying for. There are three main factors that influence the decision-making process in an interview.

What are your values?

What is your approach to your work?

What does the future hold for you?

In light of this, if you are looking to secure a technical role with Axis Bank, it is important for you to demonstrate how your expertise sets you apart from the rest. Moreover, you should also include specific details on how your efforts will contribute to the achievement of the organization's goals.

Interview Rounds

As part of the technical interview, the candidate is likely to be asked a combination of behavioural questions, situational questions, theoretical questions, puzzles, and programming challenges in order to assess their technical proficiency. As part of the Axis Bank Recruitment process, each candidate must undergo two rounds of assessment, which evaluate a candidate's analytical and technical abilities.

Round 1: Online/Written Test (Aptitude)

Round 2: Technical + HR Interview

Note: The interviewee needs to be prepared for any interview round they may encounter. In certain cases, you may have to undergo more technical rounds during the interview, depending on your previous performance in the first round, your experience, and other factors. Both the technical round and the HR round of the interviews can be conducted separately or as part of a combined round. Therefore, it would be advisable to prepare yourself beforehand. During the interview, if you are uncertain of the answer to a question, do not hesitate to tell the interviewer politely. For an applicant to be considered for a position at Axis Bank, he or she must pass both rounds.

Aptitude: In the recruitment process, aptitude tests are an effective method for ascertaining a candidate's suitability for a specific job position. Since technology and its fluctuating nature have evolved over the past few years, a lot of things have changed dramatically, and these changes are still being seen as time passes. As of today, many companies conduct aptitude tests online; however, there are some companies that conduct aptitude tests in person as well. Typically, aptitude tests are designed to measure an individual's logical and reasoning abilities, as well as their technical abilities. As well as prioritization skills, Problem-solving skills, and numerical skills are also very important. The test contains MCQs pertaining to a wide range of topics such as machine learning, statistics, probability, and even some coding questions.

Technical Interview: Following the first interview round, those who are shortlisted will be invited to the 2nd interview round, consisting of both technical and HR questions. As part of this round, there will be questions based on both technical, as well as behavioural aspects.

The technical part of the interview evaluates the candidate's abilities not simply in terms of technical expertise, which is often directly related to the position they are seeking, but also in terms of their ability to handle difficult situations and conflict. There will be numerous questions regarding the programming language you are familiar with, as well as Java concepts, OOP concepts, CS fundamentals, ML concepts, etc. Also included are some coding questions. Additionally, you may be asked about past projects you worked on, and you may have to solve a few puzzles.

When it comes to the HR part of the interview, candidates are evaluated based on their communication skills, personality traits, reasoning abilities, strengths and weaknesses, suitability for the role, and so on. In addition to asking you about your previous internships, the interviewer may also ask you questions about your resume and other aspects of your career. As part of preparing for the HR interview, it is advisable that you familiarize yourself with the organization's vision, leadership principles, the technology stack they are using, and the work-life balance they encourage. As you are preparing for the interview, it can be helpful to ask yourself a series of questions.

After the interview process, the interviewers will hold a debriefing to discuss your performance. Ultimately, if the majority of interviewers recommend you for the position, the recruiters will contact you within 2-3 business days of your interview, and you will receive an offer letter within one week of your interview. Next, let's look at some of the most frequently asked technical interview questions.

Axis Bank Technical Interview Questions: Freshers and Experienced

1. Write a program that removes consecutive duplicate characters from a string.

Given string str, remove all the consecutive duplicates from a string.

For e.g:

String: 'Scaaaleer Famiillyyyy'.

Output: Scaler Family

Code:

import java.util.\*;

import java.io.\*;

class Main

{

// Function to remove adjacent duplicates characters from a string

public static String removeConsecutiveDuplicates(String str)

{

// base case

if (str == null)

{

return null;

}

char[] chars = str.toCharArray();

char prev = 0;

int k = 0;

for (char c: chars)

{

if (prev != c)

{

chars[k++] = c;

prev = c;

}

}

return new String(chars).substring(0, k);

}

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

System.out.println("Enter a string: ");

String str= sc.nextLine();

var answer = removeConsecutiveDuplicates(str);

System.out.println("\nString after removing consecutive duplicate characters: " + answer );

}

}

Output:

Enter a string:

Scaaaleer Famiillyyyy

String after removing consecutive duplicate characters: Scaler Family

**2. State difference between Function overloading and Function overriding.**

Polymorphism in Java is achieved by overriding and overloading.

Overloading Overriding

A method overloading is a situation where a class contains more than one method named the same, but with different parameters. A method override occurs when the superclass and the child class have the same names and arguments/parameters.

Different parameters are required for overloading. Overriding requires the same parameters

Overloading takes place within a class. A class can override another when the two classes share an IS-A (inheritance) relationship. To override, both the base class and its child class must exist.

Static binding is used for method overloading. Dynamic binding is used for method overriding.

Compile-time polymorphism is demonstrated here. Run-time polymorphism is demonstrated here.

If overloading breaks, a compiler error will appear, which can easily be fixed. Overriding breaks can create serious problems since the effects can be seen at runtime.

Overloading Example:

class Main

{

static int min(int x, int y)

{

return x - y;

}

static int min(int x, int y, int z)

{

return x - y - z;

}

public static void main(String args[])

{

System.out.println("x - y = " + min(10, 5));

System.out.println("x - y - z = " + min(10, 5, 1));

}

}

Output:

x - y = 5

x - y - z = 4

Overriding Example:

class Scaler

{

void show()

{ System.out.println("Scaler");

}

}

class ScalerVerse extends Scaler

{

void show()

{

super.show();

System.out.println("ScalerVerse by Scaler");

}

}

class Main

{

public static void main(String args[])

{

Scaler ob = new ScalerVerse();

ob.show();

}

}

Output:

Scaler

ScalerVerse by Scaler

**3. Explain list comprehension.**

One of the few programming languages that are easy to read is Python, which can be used to create code that is easy to write, elegant, and appears to be written in plain English. List comprehension is a compelling feature of this programming language that allows the creation of powerful functionality within a single line of code. It is an easy and compact way to create a new list from a string or another list. In this way, we can create a new list very quickly by operating on each element of the existing list. List comprehension is considerably more efficient than using a for loop to process a list.

Syntax:

newList = [ expression(element) for element in oldList if condition ]

print(newList)

OR

newList = []

for item in 'oldList':

newList.append(character)

print(newList)

Example1:

newList = []

for character in 'Scaler':

newList.append(character)

print(newList)

Output:

['S', 'c', 'a', 'l', 'e', 'r']

Example2:

newList = [character for character in 'Scaler']

print(newList)

Output:

['S', 'c', 'a', 'l', 'e', 'r']

**4. What do you mean by lambda functions and how to use them?**

As a programming technique, lambda functions are small anonymous functions that are often used when a function needs to be executed for a very short period of time. An anonymous function is a function without a name and it is declared using the lambda keyword in Python. In many cases, this can be used whenever you want to pass a function as an argument to a higher-order function, i.e., to a function that accepts other functions as arguments.

Syntax:

lambda arguments : expression

Even when there is no return statement, the function simply returns the expression result. The lambda function can accept several arguments but is restricted to only one expression.

Example1:

# Perform addition using lambda

add = lambda x: x+10

print(add(2))

Here,

lambda x: x+10' is the lambda function.

‘x’ is the argument

x + 10 is evaluated and the result of the expression is returned.

Output:

12

Example2: This example illustrates how anonymous functions can be used within another function.

def addfunc(num):

return lambda x : x + num

result = addfunc(10)

print(result(2))

Here, we have a function 'addfunc' that takes one argument 'num', where the argument is to be multiplied by a number 'x' that is unknown.

Output:

12

**5. What data type would you use if you wanted it to be immutable?**

Unlike mutable data types, immutable data types cannot be altered following their creation. Mutable data types can be changed after they have been created, but immutable data types cannot. A few examples of immutable data types include bytes, numeric data types, strings, frozen sets, and tuples.

**6. Write a program to find the minimum number of operations to make all elements of the array equal.**

Given an array 'scaler[]' of 'n' positive integers, you must determine the minimum number of operations necessary to make all elements of the array equal. Array elements can be added to, multiplied by, subtracted from, or divided by. Any operation on an element of the array will be considered a single operation.

Code:

import java.util.\*;

class Main

{

//Function to return minimum operations needed to equalize array elements

static int minOpera(int scaler[], int n)

{

HashMap p = new HashMap();

for (int i = 0; i < n; i++)

{

if (p.containsKey(scaler[i]))

{

mp.put(scaler[i], mp.get(scaler[i]) + 1);

}

else

{

mp.put(scaler[i], 1);

}

}

int maxFreq = Integer.MIN\_VALUE;

maxFreq = Collections.max(mp.entrySet(),

Comparator.comparingInt(Map.Entry::getKey)).getValue();

// Return the minimum operations required

return (n - maxFreq);

}

// Driver code

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

System.out.println("Enter total number of elements: ");

int n = sc.nextInt();

System.out.println("\nEnter elements of the array: ");

int[] scaler = new int[10];

for (int i = 0 ; i < n; i++ )

{

scaler[i]= sc.nextInt();

}

System.out.println("\nMinimum required operations = " + minOpera(scaler, n));

}

}

Output:

Enter total number of elements:

4

Enter elements of the array:

3

5

9

6

Minimum required operations = 3

7. Write a program to return the length of the longest consecutive subsequence.

Given an array scaler[] of n integers, determine the length of the longest consecutive subsequence. There is no particular order to the consecutive numbers.

Code:

import java.io.\*;

import java.util.\*;

class Main

{

static int LongConsSubseq(int scaler[], int n)

{

Arrays.sort(scaler);

int res = 0, count = 0;

ArrList<Integer> a = new ArrList<Integer>();

a.add(10);

for (int i = 1; i < n; i++)

{

if (scaler[i] != scaler[i - 1])

a.add(scaler[i]);

}

// Traverse the array to find the maximum length

for (int i = 0; i < a.size(); i++)

{

if (i > 0 &&a.get(i) == a.get(i - 1) + 1)

count++;

else

count = 1;

res = Math.max(res, count);

}

return res;

}

// Driver code

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

System.out.println("Enter total number of elements: ");

int n = sc.nextInt();

System.out.println("\nEnter elements of the array: ");

int[] scaler = new int[10];

for (int i = 0 ; i < n; i++ )

{

scaler[i]= sc.nextInt();

}

System.out.println("\nLength of the longest contiguous subsequence is: " + LongConsSubseq(scaler, n));

}

}

Output:

Enter total number of elements:

8

Enter elements of the array:

4

6

5

7

2

9

8

1

Length of the longest contiguous subsequence is: 4

**9. What are the different data types in Python?**

Among the fundamental concepts of the Python programming language are data types. Every Python value has its own data type. Data Types are used to classify or categorize data items. In this way, we can understand what operations can be applied to a value. By default, Python contains the following data types:

1. Numeric Type: Numeric data types represent data with numeric values in Python. Floating point numbers, integers, and complex numbers fall under the numeric category.

Example:

x = 8

print("x Type: ", type(x))

y = 8.0

print("y Type: ", type(y))

z = 3 + 5j

print("z Type: ", type(z))

Output:

x Type: <class 'int'>

y Type: <class 'float'>

z Type: <class 'complex'>

2. Sequence Type: Sequence data types are ordered collections of similar or different data types. Sequence data types support efficient and organized storage of multiple values. List, String, and Tuple fall under the sequence category. List items are separated by commas and enclosed in brackets [ ] whereas Tuple items are separated by commas and enclosed in parentheses (). Strings can be represented with single quotes or double-quotes. You can denote multi-line strings using triple quotes (""") or ''').

Example: List (Mutable)

a = [4, 3.5, 'scaler']

print(a)

print("a[1] = ",a[1])

a[1] = 4.5 # changing the value at a[1]

print(a)

print(type(a))

Output:

[4, 3.5, 'scaler']

a[1] = 3.5

[4, 4.5, 'scaler']

<class 'list'>

Example2: String

str = "This is a string!"

print(s)

print(type(str))

Output:

This is a string!

<class 'str'>

Example3: Tuple (Immutable)

t = (4,'scaler', 3+4j)

print(t)

print("t[1] = ", t[1])

print(type(t))

t[1]= 57 # changing the value at t[1]

print(t)

Output:

(4, 'scaler', (3+4j))

t[1] = scaler

<class 'tuple'>

Traceback (most recent call last):

File "<string>", line 5, in <module>

TypeError: 'tuple' object does not support item assignment

3. Boolean Type: A Boolean data type contains one of two values, True or False. If a boolean object is equal to True, it is truthy (true), and if it is equal to False, it is falsy (false). Even non-boolean objects can be evaluated as true or false in a boolean context. Python only accepts True and False with capitals, otherwise it will throw an error.

Example:

print(type(True))

print(type(False))

print(type(false))

Output:

<class 'bool'>

<class 'bool'>

Traceback (most recent call last):

File "<string>", line 3, in <module>

NameError: name 'false' is not defined

4. Set Type: A Python set consists of an unordered collection of data types, with the ability to iterate, mutate, and do not contain duplicate values. They do not follow any particular order.

Example:

a = {6,8,'scaler',9,1}

print("a = ", a)

print(type(a))

Output:

a = {1, 6, 8, 9, 'scaler'}

<class 'set'>

5. Dictionary Type: Dictionary data types in Python are ordered collections consisting of key-value pairs. They are generally used for large datasets. Dictionary data types are optimized for data retrieval.

Example:

d= {1: 'Scaler', 2: 'For', 3: 'Scaler'}

print(d)

d= {'Name': 'Scaler', 1: [4, 5, 6, 7]}

print(d)

print(type(d))

Output:

{1: 'Scaler', 2: 'For', 3: 'Scaler'}

{'Name': 'Scaler', 1: [4, 5, 6, 7]}

<class 'dict'>

10. Explain data dictionaries and their components.

Similar to any other subject dictionary, a Data Dictionary contains a collection of data objects and related items contained within a database for the convenience of programmers and others who may need to reference the data. In software technology terminology, these objects are called "Metadata". Data dictionaries are often used as centralized metadata repositories. In most cases, these objects concern data ownership, relationships among data objects, and other aspects of data structure. These dictionaries present a useful guide for understanding and evaluating data.

If your data is stored and managed in a relational database, your software can generate a data dictionary for you.

In case your data is stored and managed in spreadsheets, comma-separated values, or text files, you will have to manually create a data dictionary.

Data Dictionary components

Data dictionaries can contain a variety of contents. All of these components provide information about data and are generally classified as metadata.

Object names, aliases, and definitions

Properties of data elements (e.g., unique identifiers, size, indexes, data types, nullability, and optionality)

Business rules (such as those relating to data quality and schema objects)

System-level diagrams

ERD (Entity-relationship diagrams)

Reference data

Quality-indicator codes, etc.

11. What are different types of data dictionaries?

Data dictionaries typically fall into two categories: active and passive dictionaries. Both differ in the level of automatic synchronization they provide.

Active Data dictionary: The database management system is responsible for ensuring that any changes to the database structure reflect immediately in the data dictionary. This method of immediate update is followed by Active Data Dictionary, as it is self-updating. By doing so, there will be no discrepancies between the data dictionary and its corresponding database structure.

Passive Data dictionary: Passive dictionaries are not as useful or as easy to use as active dictionaries. Each time a database is modified, the data dictionary must be updated manually to align with the database. The dictionary is maintained independently from the database.

12. Can you explain what is static in Java?

In Java, the static keyword is primarily used to manage memory. Static keywords can be applied to blocks, methods, variables, and nested classes. If you declare a variable or a method static, it belongs to the class, instead of an individual instance of the class. This keyword is used to refer to methods or variables that are the same across every instance of a class.

Example:

import java.util.\*;

class Main

{

// x and y are static variables

static int x = 15;

static int y;

static void printStatic()

{

x = x + 2;

y = x;

System.out.println("printStatic::Value of x : "+ x + "\nValue of y : "+y);

}

public static void main(String[] args)

{

printStatic();

y = x + 5;

x++;

System.out.println("\nmain::Value of x : "+x + " \nValue of y : "+y);

}

}

Output:

printStatic::Value of x : 17

Value of y : 17

main::Value of x : 18

Value of y : 22

13. How is an abstract class different from an interface?

The interface and abstract classes are both special types of classes that contain only the declarations of methods rather than their implementations. Both serve to achieve abstraction by declaring abstract methods. Interfaces, however, are entirely different from abstract classes.

Interface Class Abstract Class

Whenever an interface class is implemented, its subclass must define all of its abstract methods and provide their implementation. Whenever an abstract class is inherited, its subclass isn't obligatory to define its abstract methods until the subclass uses the abstract methods.

Interfaces can only have abstract methods. As of Java 8, they can also have static and default methods. An abstract class may contain both abstract methods and non-abstract methods.

Multiple inheritances is supported by the interface. Multiple inheritances is not supported by abstract classes.

Interfaces can extend other Java interfaces only. Abstract classes extend other Java classes and implement multiple Java interfaces.

Java interfaces have public members by default. There can be different types of class members available in Java abstract classes, such as private and protected.

It is not possible to declare constructors or destructors in an interface. Constructors and destructors can be declared in an abstract class.

When declaring a method as abstract in an interface, the keyword "abstract" is optional. To declare a method abstract in abstract classes, the keyword "abstract" is required.

Variables must only be final and static. Static, non-static, and final, non-final variables can all be present in an abstract class.

14. Can Java applications be run without implementing the OOPs concept?

No, Java applications are unable to be run without implementing OOPs. Java applications rely on object-oriented programming concepts, and therefore cannot be implemented without them. Java is an object-oriented programming language, which means that every program must implement at least one class. Programs are usually composed of many classes and objects, which are instances of classes. C++, on the other hand, can be implemented without any OOPs, as it supports the structure programming model that is similar to the one found in C.

15. What will be the output of the following code?

Code:

import java.util.\*;

public class Main{

public static void main(String[] arr)

{

System.out.println("ScalerEdge");

}

public static void main(String arr)

{

System.out.println("ScalerVerse");

}

}

Output:

ScalerEdge

Reason:

In this case, the main() method is overloaded. The JVM (Java Virtual Machine), however, only understands methods that have a String[] argument in their definition. ScalerEdge, as a result, is printed in the output and the overloaded main method is ignored.

16. Is class and structure the same? If not, what are the differences between a class and a structure?

In C++, a structure behaves similarly to a class, with a few small exceptions/differences. The most significant exception relates to the hiding of implementation details. By default, structures do not hide their implementation details from users, while classes hide all implementation details by default and thus do not allow programmers to access them.

Class Structure

A class is a user-defined data structure or type in C++ that is declared by using the keyword class. In essence, a class is a collection of methods and fields. A structure is a set of variables of different data types (such as strings, integers, bools, etc.) grouped together under a single name. A variable within the structure is known as a data member.

A class provides support for inheritance. Usually, a class can create a subclass that inherits the methods and properties of its parent class. Inheritance is not supported in structure.

A class is essentially a reference type and its objects are created in heap memory. A structure is essentially a value type and its objects are created in stack memory.

All members of the class are automatically set to be private if none of the access specifiers are provided. All members of the structure are automatically set to be public if none of the access specifiers are provided.

By default, the base classes and structures of a class are private. Base classes/structures of a structure are public by default.

Data abstraction and inheritance are implemented using it. It is used for the grouping of data.

Any type of constructor and destructor could be present in it. It may have only a parameterized constructor.

Null values may be assigned to class variables. Structure members, however, cannot have null values.

Syntax:

class class-name

{

// data

// functions

};

Syntax:

struct structure\_name

{

// member declarations.

}

17. How does C++ support Polymorphism?

Polymorphism is one of the most important concepts in object-oriented programming. Basically, it means that a single entity can take on more than one form, i.e., a single function or operator can behave differently in different ways. C++ is an Object-Oriented programming language, which means that it supports Polymorphism in its programming constructs.

Example:

In C++, the + operator performs two specific functions. In the case of numbers (like floating-point numbers or integers), it performs addition.

int a = 10;

int b = 5;

int res= a + b; //res = 15

In the case of strings, it performs string concatenation.

string first\_Name = "Scaler ";

string last\_Name = "Academy";

string full\_Name= first\_Name + last\_Name; //full\_Name = "Scaler Academy"

Types of Polymorphism

Compile Time Polymorphism: Also called early binding or static binding, this type of polymorphism occurs during compilation. By making use of features like templates, function overloading, operator overloading, and default arguments, C++ features compile-time polymorphism.

Runtime Polymorphism: Also known as late binding or dynamic binding, this type of polymorphism occurs when a function is invoked during runtime. In C++, runtime polymorphism is supported by features such as virtual functions and function overriding. Functions defined in a base class with the virtual keyword constitute virtual functions. They are intended to ensure that the function is overridden. We may not be able to override the base class function if we do not include the virtual keywords.

18. What do you mean by DELETE and TRUNCATE statements in SQL?

Both TRUNCATE and DELETE statements are part of the SQL delete query. Both are used to get rid of unwanted records or rows in a table.

DELETE TRUNCATE

You can delete all records from a table or only certain records matching your criteria with the SQL Delete command. The Truncate command in SQL deletes all rows or tuples from a table.

It is a DML (Data Manipulation Language) command It is a DDL (Data Definition Language) command.

A WHERE clause can be used with the DELETE command. WHERE clauses are used with DELETE commands to remove or delete only the rows that satisfy a condition, otherwise all rows are removed by default. TRUNCATE commands that do not contain a WHERE clause, unlike DELETE commands.

Tuples are locked before being removed with the DELETE command. A data page must be locked prior to removing table data using TRUNCATE.

Since the DELETE command maintains the transaction log for each deleted record, it is slower than TRUNCATE. TRUNCATE is faster since it does not scan each record before removing it; rather, it removes entire data blocks at once.

Transaction logs are maintained for each deleted record, allowing us to roll back the change. Using the ROLLBACK or COMMIT statements, we can restore deleted data. There are no transaction logs for each deleted row, so we cannot roll back the changes. Once the command has been executed, we cannot revert the changes.

Because it keeps a log for each deleted row, it consumes more transaction space than truncate. Since a transaction log is maintained for the entire page instead of for each row, it occupies less space than the DELETE command.

Syntax:

DELETE FROM TableName WHERE condition;

Syntax:

TRUNCATE TABLE TableName;

19. What are the SQL Constraints?

In SQL, constraints specify the rules for table data. To limit the type of data that can be entered into a table, constraints are used. This ensures that the table's data is accurate and reliable. Data actions are aborted when a constraint is violated. Constraints may be defined at the column or table level. As the name implies, a column-level constraint is applied to a particular column, whereas a table-level constraint is applied to the entire table. Constraints can be specified when creating the table, or afterward using the ALTER TABLE statement.

Syntax:

CREATE TABLE table\_name

(

col1 datatype(size) constraint\_name,

col2 datatype(size) constraint\_name,

col3 datatype(size) constraint\_name,

....

);

Constraints available in SQL

NOT NULL: This constraint prevents us from storing null values in a column. If a column is set to NOT NULL, then the null value can no longer be stored in the given column.

UNIQUE: If a column is set to UNIQUE, then every value in the column must be unique. This means that no value should be repeated in the column.

PRIMARY KEY: A primary key of a table is a field that uniquely identifies each row within the table. This constraint specifies a field as a primary key in a table.

FOREIGN KEY: A foreign key of a table is a field that uniquely identifies each row of another table. This constraint specifies a field as a foreign key in a table.

CHECK: Using this constraint, the values of a column can be validated to meet a specific condition. This ensures that the value stored within a column meets a particular condition.

DEFAULT: This constraint sets a default value for the column when no user value is specified.

20. Is it always necessary to create objects of a class?

No, it is not always necessary to create instances (objects) of a particular class. It is necessary to create an object if a class contains non-static methods, but if the class contains static methods, objects do not have to be created. A static method can be invoked/called directly using the class name, which means you don't need to create a new object of the class. A non-static method can only be called on an object, which means you must create a new instance (object) of the class.

21. What are the most common clauses in SELECT statements?

SQL SELECT statements retrieve data from the database according to specified conditions, if any. SELECT statement may include the following clauses:

WHERE: Used to filter out records, i.e., it fetches and returns only relevant data from a table that meets certain criteria.

ORDER BY: Sort records in ascending order or descending order in SQL.

GROUP BY: Groups rows that have the same values in a result set.

FROM: Specifies the name of the table from which data is to be retrieved.

LIMIT: Sets the number of rows to return.

TOP: Specifies how many rows of tables should be shown in the result.

22. What pointer type should you use if you are implementing the heterogeneous linked list in C?

Since the heterogeneous linked list consists of different types of data, it is not feasible for ordinary pointers to be used for this purpose. Hence, you will need to use a generic pointer type such as a void pointer since the void pointer can hold a pointer to any type.

Useful Resources

Software Engineering Interview Questions

Technical Interview Questions

Coding Interview Questions

Interview Resources

DSA- Programming

Mock Interview

Axis Bank Interview Preparation

Interview Preparation Tips

Since tech jobs have become more competitive in recent years, the recruiting process has evolved into a very complex and extensive process that evaluates not only your programming and problem-solving skills but also your individual traits and experience. Researching frequently asked questions isn't the only step in preparing for an interview. You will need to convince several prospective employers that you are well suited for the position that you are applying for. To assist you, we have compiled a list of our most valuable interview tips that may provide you with useful information. Listed below are some tips both for novices and experts for preparing for an Axis Bank interview.

Practicing leads to perfection: As companies are always seeking talented candidates, it is vital to prepare for an interview. You can reference past interviews to get a better idea of what to expect on the interview day. No matter what the interview involves, you can prepare excellent answers to common questions and anecdotes that showcase your skills. In this way, you can determine where your weaknesses are and work on improving them. Make sure you prepare well for the aptitude section by taking aptitude tests on GFG, Testpot, and Indiabix.

Do your homework: Focus on your technical skills before you go on an interview, as skilled candidates are always in demand. As such, it is advisable to review the job requirements prior to applying, and then prepare accordingly. Additionally, you should review the fundamentals. An ideal candidate should be familiar with programming skills, OOP (Object Oriented Programming), DSA (Data Structures and Algorithms), design patterns, DBMS concepts, networking, Machine learning, and so on.

Take it a step further: In the case of a technical discussion, please share your personal opinions and interests as well. A room full of like-minded people is more likely to discuss technical topics that connect. Keep an optimistic and enthusiastic attitude throughout the interview. This will enable the interviewer to better assess your abilities and personality. However, be careful with your answers, as the interviewer might attempt to trip you up to determine whether or not you are bluffing.

Feel free to ask questions: It's best not to dive headfirst into a coding challenge even if you are familiar with it. At the end of an interview, feel free to ask any questions you might have. This way, you can determine whether the role is a good fit for you and demonstrate to the interviewer that you are interested in the position. A list of questions for an interview shows interest, enthusiasm, and engagement - all qualities employers seek. Asking thoughtful questions in an interview shows you are interested in the position. Please refrain from asking questions that tend to focus too much on what the organization can do for you, such as concerning salary and holiday policies.

Showcasing your expertise: Come up with answers that reflect an innovative approach to the problem. To be competitive in today's business environment, companies must seek out people who bring disruptive ideas to the table. Providing examples is the best way to demonstrate your expertise since you are demonstrating how your experience relates to the position you are seeking. Instead of boasting about your skill set, describe how you have applied it.

Build a strong portfolio: As a prospective software engineer, you should develop an impressive portfolio that reflects your experience. A portfolio is often the first thing an employer sees about an applicant, so you want to make sure yours stands out in a good way and helps you land an interview. Be sure to include some impressive projects that you have worked on in your resume. Be precise in crafting your resume and do not oversell or undersell yourself, since the emphasis is on the information you include and your level of authenticity.

Frequently Asked Questions

1. Why do you want to join Axis?

Irrespective of your background, whether you are an engineer or a graduate of commerce, you can mention that you are passionate about finance and especially banking. Answering this question illustrates your commitment to that particular employer and will indicate whether you researched the company before applying. Use this opportunity to demonstrate your genuine interest in the organization. Mention the following points, if applicable:

If you are good with numbers and have interpersonal skills, you can say that these skills, which you have acquired over time, will enable you to make better decisions as a banker.

Additionally, you can state that your graduation helped you develop leadership skills that may prove to be of greater value in a banking environment.

It might be useful to cite something from your past experience, such as your time as a student secretary or a social club president, or what you have done to figure out a solution to a problem with limited information.

Moreover, you can mention that banks follow a structured approach to promotions and as such, there is a clear progression in your career besides having a stable job.

As a banker, you are likely to feel that this position offers incredible opportunities and challenges, and this incites you to work harder and contribute more. Also, you will have exposure to a variety of disciplines including finance, accounting, audit, and legal.

2. How long is the Axis Bank Interview Process?

Axis Bank's hiring process is dynamic and fast-paced, usually consisting of two interview rounds that include a variety of questions. Axis Bank interviews may take between three and four weeks to complete, depending on the job's difficulty.

3. What is the Eligibility Criteria at Axis Bank?

Axis Bank Eligibility Criteria for Software Engineers:

Eligibility Criteria Details

Graduation Eligible: BE/B.Tech, ME/M.Tech, MCA, BCA, M.Com, B.Com, B.Sc, M.Sc

Academic and Graduation Criteria 50% throughout 10th, 12th, and UG/PG

Backlogs No active Backlogs