CHAPTER 4: PROPOSED METHOD AND METHODOLOGY

4.1 NEED OF THE PROPOSED RESEARCH WORK

Cloud Service Providers have the prime duty of safeguarding our information from the cloud and know about the threats beforehand. Here focus is on reliable data sharing and so the problems need to be discussed upon.

A platform for IT services spread over the internet is cloud computing but in order to make these services useful one needs to focus how unwanted access in data should be dealt with, some sensitive data must be encrypted before outsourcing to server, this will make future relations of customer and cloud service provider strong, and data will be safe.

To enjoy the real fruits of numerous advantages of cloud computing, customers fear to have to be removed. But nowadays set of issues occurring with information, controlling of sensitive information, keeping up with confidentiality is utmost important, otherwise people will walk away because of lack of trust. If data cannot be found as and when required by people, storing information serves no value. So issues of security of cloud are the main concern to be dealt with. Now encrypted data to be searched upon and privacy preserving problems have to be dealt with. A search scheme which can work upon encrypted data must be introduced. For defining a scheme, a new methodology is the need of an hour. Security, reliability and scalability being set up in a new model and the cost of storing data in cloud server is much less. Data providers may now share information as integrity check is maintained.
4.2 CLOUD SECURITY AND PROPOSED MODELS

In the cloud, information can be stored, and servers, applications and services are present in form of resources. Customers can get services, and information can be stored at servers, and as these are the positive aspects of cloud which include outsourcing. “Pay-as-you-go” will remain as the major principle. Outsourcing is performed outside of company means the cloud work is placed outside.

Information file quality should be maintained and prevention should also be there from encrypted files. The file names which are encrypted should also be saved as it also contains data and because of encryption, the simple text manner saving of documents will be of no use [155]. Under the maintenance of Cloud Service provider (CSP), presently data from local sites are converted to outsource areas.

For the protection of information, service providers follow methods like firewalls and virtualizations which are not too reliable [156]. Nowadays all over the world, people are joining hands so as to provide data security as hackers are growing more and more. And people are dependent on CSP for information protection or maintaining of data privacy.

Several safety issues: With virtualization, information is being stored in several data centers but as data is divided into many data centers, this will suffer difference in opinion of legal systems [157]. Authorization, confidentiality and authentication serve as basic requirements along with identification and integrity [158]. Considering data lifecycle and the source of threat, answers to growing concerns are given by threat-models. Here the
issues related to security, maintaining privacy, data safety, and issues related to an infected application are a major concern.

The most important asset is information which every company or any organization unit keeps with it and many problems that do occur with security are related to it. Customer’s data which is important or may be private must be kept hidden and safe from providers. Consistency is utmost important for data as information may be replicated as per requirements but privacy must be maintained. In order to keep control over the applications so as to prevent them from getting infected, cloud server provider should monitor the servers.

Up to the levels of two – level providers and the cloud customers, information must be dealt due to privacy reasons as the data is being sent to and received from cloud customers and CSPs. From all threats, server should be set free from external issues which could harm the systems and cause data loss.

Some terms related to figure could be:

Resource Pooling: Cloud users are provided the services which are required by them from a large set of resources, this is known as resource pooling.

Multitenancy: In a shared environment, many applications work hand in hand.

Cloud Service Hijacking: An attack which manages communication, and this attack deals with control network security.
Browser Security: To keep secure data over network areas, an application which provides safety dealing regarding the internet is Browser Security.

Flooding Attack: In order to use the resources and services illegally, an attack is made as target systems send a large number of requests, this is known as Flooding Attack.

In clouds, for data utilization the factor used for searching methods are: flexibility and efficiency. Cloud customers should keep their private data aloof and should hide the real identity of customers. Challenges and concerns are coming up as to more cloud users are growing up but due to the hacks and data steal the fear factor growing up.

There may be a large number of files and customer might need only a very few of them. Sometimes people are in need of particular data but how to get it. Keywords are being used to find data for a query of the user [159]. Some of the methods of encryption help
the user extracting information of requirement and the files of use. Google is the most common search engine, as users provide set of keywords to get data needed.

![Flowchart Diagram](image)

**Figure 4.2: Multi-Keyword Ranked Search**

In figure here how the data is being searched is described as control management of data is in hands of data owner. When in need, to get data back particular files are given an index to retrieve information. After the request is made customer’s files are searched and given as per need.

**THREAT MODEL**

Because of user demands of information of cloud, the private data may get lost if data is not saved in a secured manner. This all needs to be checked and risks need to be removed, for this, the process should be made safer, so that even if the little information in the form of inputs is received by the server, customers data will not get affected [160].

In a cloud environment, there are many rising issues of threats so as to deal with them, there are threat models which are introduced, such as Microsoft’s STRIDE, which is most
popular among them. Classification of threats includes - Tampering, Information disclosure, elevation of privilege, Spoofing, Repudiation and denial of service.

To identify the threat, most commonly used model is STRIDE as it gives a general idea so as to deal with threats.

Several steps can be taken up, as rights can be given dealing with who can access, which may be set up with Cloud Service Providers (CSPs) when data is needed. Only cloud customers which are verified to access or authorized as per access rights are provided with services.

![Diagram](image)

**Figure 4.3 : Model for privacy preserving data utilization in cloud**
Here the figure above, Data Ownership is being described which is defined as the presence of legal rights and control and management of data. Data Owner implements the schemes regarding the data of the owner of information saved in clouds and how it can be distributed and use. Files reach to the data manager, and for the administration, access to cloud data and for provisioning for cloud services data manager is responsible for specifying the protocol. Giving cloud customers the services like IaaS, PaaS, SaaS, DBaaS etc is being controlled by Cloud Provider. CSPs or Cloud Service Provider is another term for Cloud Provider. Before data is outsourced to the cloud, controlling of all information is managed by Cloud Controller which is also responsible for maintaining privacy and security measures.

In Figure, important points of cloud security related paradigms are being described.

![Cloud Security Paradigms Diagram](image)

**Figure 4.4 : Cloud Security paradigms**
4.3 MULTITENANCY AND PROPOSED MODELS

Resources are distributed to customers in cloud computing which is based on virtualization techniques. Data should be dealt properly in clouds and need to be made safe from illegal people which comprise information security [161]. In the form of storage, servers,

Many tenants share cloud resources and provide services, as here with virtualization techniques users work with shared infrastructure and its services. As to provide the allowance for application configuration, hardware resources are being shared by customers by one database instance and one shared application. Single application and database are being used by tenants as application are now set up to share hardware resources and software configuration.

Many features are included in SaaS like multitenancy, on-demand access etc. SaaS and multitenancy both features are used so that users requests can be fulfilled.

![Figure 4.5: Multitenant Architecture](image-url)
Many customers use a single instance of an application in multitenancy and also under one server multiple customers are hosted. Tenants have partitions virtually between them where data and settings for configuration are stored. Here are discussed multi Tenants Department Modes for Application Server.

**MULTI TENANTS DEPLOYMENT MODES FOR APPLICATION SERVER**

1. **Fully Isolated Application Server**

   Each tenant accesses an application server running on a dedicated server.
2. **Virtualized Application Server**

Each tenant accesses a dedicated application running on a separate virtual machine.

![Virtualized Application Server Diagram](image)

**Figure 4.7(2): Virtualized Application Server**

3. **Shared Virtual Server**

Each tenant accesses a dedicated application server running on a shared virtual machine.

![Shared Virtual Server Diagram](image)

**Figure 4.7(3): Shared Virtual Server**
4. **Shared Application Server**

The tenant shared application server and access application resources through separate session or threads.

![Shared Application Server Diagram](image)

**Figure 4.7(4): Shared Application Server**

5. **Fully isolated data center**

The tenants do not share any data center resources.

![Fully isolated data center Diagram](image)

**Figure 4.7(5): Fully isolated data center**
6. Virtualized Servers

Tenants share same host but access different databases running on separate virtual machines.

![Virtualized Servers Diagram](image)

**Figure 4.7(6): Virtualized Servers**

7. Shared Server

The tenants share same server (Hostname or IP) but access different databases.

![Shared Server Diagram](image)

**Figure 4.7(7): Shared Server**
8. **Shared Database**

The tenants share the same server and database (shared or different ports) but access different schema (tables).

![Diagram of Shared Database](image)

*Figure 4.7(8): Shared Database*

9. **Shared Schema**

The tenants share the same server, database and schema (tables). The irrespective data is segregated by key and rows.

![Diagram of Shared Schema](image)

*Figure 4.7(i): Shared Schema*
4.4 DBaaS AND PROBLEM FRAMEWORK

The main objective of DEM Model is: While maintaining security of data, at the side of server query processing can be done efficiently, keeping this objective a model for Database-as-a-Service is to be developed.

Query processing work should be handled by server. While performing work related to query, client may help the server but cannot do query processing. So at client side, there is no need to have database engine. That is why it has benefits of DbaaS model benefits.

4.5 ASSUMPTIONS

Various assumptions which have been taken in DEM Model are as follows:

1. The server is not trusted. As server can be hacked and all data will be lost. But data is considered to be the most critical asset, so data need to be secured at any cost, which remains the prime objective. So data is saved and the database is located on the server but it is kept secret because encryption is done on whole. As data is converted to a form which cannot be easily read and understood so even after maintaining security, data is leaked then it is of no use as it cannot be decrypted easily.
   Encryption and decryption keys are the clients but there is no sharing by the client to the server as having assumed that server is not trustable.

2. The client is trusted. All encryption/decryption keys reside with the client but users/operators and employees can trust client and there will be no compromise.
Means nobody can make client compromise and get the encryption/decryption keys to access to all records of the database.

3. The middle channel used to transmit between client and server is safe to communicate if security aspects are being considered.

4. Encryption measures used by DEM model are trustable. Even if in future if they are broken, many schemes may be developed having similar features of encryption and decryption and can be substituted with the old.

5. The server has big disk capacity. Practically it is seen that nowadays disk is cheap.

6. Database Engine is not disturbed by the server. All work is done in correct form. No unauthorized alterations or interferences are made by the server.

4.6 DEM MODEL ARCHITECTURE & WORKING

Here a model called DEM Model or Data Encryption Model is being introduced, see fig 4.8. The Query is given to the client by the user or operator working for the user. The query is given to client machine and processing is performed. Encryption and decryption layer and query processing framework comprises client.

According to specifications of the general user, that person gives query and this is being performed by entering the query of SQL in the form provided by the user or the operator working for the user. A new concept of a user-defined query is being introduced. For user-defined query further with the help of form, it has been explained. For SQL query processing, the first stage, there is a query processor and also query parser. Keywords and expressions are being checked for and scanning for a query is performed by a parser.
For converting the query to encrypted form, the query is being written again by query rewriter and the framework is encryption/decryption layer in the model. Encryption engine is kept aloof from the reach of the server because original data is not being disclosed with the server. For processing work, server and Round Communication Handler client work hand in hand. At the client, data about data or metadata is being monitored by Metadata Handler.

The encrypted information is being managed by Arithmetic Engine. Query Manager present at server receives changed data from Query Rewriter. The query is being worked upon by Database Engine. Information remains in encrypted form and on the server, it is stored in this form, also all processing is also in encrypted form all processing work is done. Before providing results to the user the data is decrypted back again as after processing it goes to encryption/decryption layer of the client. After decryption, the user gets the results desired.

![Data Encryption Model](image)

*Fig 4.8: Data Encryption Model [DEM]*
4.7 SOFTWARE WORKING FRAMEWORK:

Here asp.net framework is taken. DOT NET (or .NET) is a cluster of many technologies. It comprises of .NET languages – It includes C# (also known as C sharp) and Visual Basic .NET.

The .Net framework defines an environment that supports the execution of component based application. It enables differing computer languages to work together. It provides for program portability, security & common programming model for the windows platform. Also, it supports the development of many web based applications.

DOT NET framework defines many different entities:

1. Common language Runtime (CLR):

   Execution of the program is managed in this system. Mixed language programming, security and portability is being managed here.

   It can be said to be the most important part of .Net Class Library. All Dot Net code runs inside the CLR when a client requests an ASP.Net webpage, the ASP.Net service runs inside the CLR environment execute the code and creates the final page to send to client.

2. Microsoft Intermediate Language (MSIL):

   Whenever Dot Net program is compiled, using any dot net language, the source code does not get converted into executable binary code but an intermediate code is made which is known as MSIL which is interpreted by
CLR. MSIL code is converted to binary code (native code) during program execution.

This code intermediate code is independent of platform.

3. Just In Time Compilers (JITers) – When MSIL is required to be executed, CLR invokes the JIT compiler which compiles the (IL) intermediate language into the native executable code. Traditional compilers are different from JITers because compilation of IL code is here done to native code.

4. DOT NET Class Library or Framework Class Library (FCL):
   Access to runtime environment is being provided by this library. For eg., Dot net framework class library can be used for I/O operations where there is a need for displaying onto the screen.
   For common usual tasks, dot net contains a big class library. Thousands of classes are present in FCL to give access to windows API and common functions like string manipulation.

### 4.8 QUERY FRAMEWORK AND ITS IMPLEMENTATION

#### 4.8.1 Encryption

Through our observation, it has being noticed that in maximum queries only some common SQL constructs, although there are many SQL constructs. That is why algorithm has been developed as such to support the most common operations.
DEM Model uses attribute level encryption. Here individually encryption is done. Here an encryption key is developed, used in the ENCR algorithm, to encrypt the data entered by operator/user. Here single encryption – decryption is used.

4.8.2 Decryption

Decryption is provided according to operator. Just after saving of data if user wish to check the values entered then the values of particular database will be decrypted from database on click “Show DB Value for this record”.

Along with this, decryption operation is performed in many operations

(1) When search operation is performed, a particular record need to be displayed so the result internally is in encrypted form. But after that the result is decrypted and displayed in form so that the user can understand.

(2) The “Show All” button, when clicked, displays all records from database and before displaying decrypts the records.

(3) The “Submit button” given for query execution, processes query internally on encrypted data and displays results in decrypted form.

(4) After “Updation” and “Deletion” of records, query processing is done on encrypted records, but when the results need to be displayed through Query Execution buttons, before showing results they are converted to decrypted form.
4.8.3 Insertion

Records are inserted to database. Insert query is used for this purpose. Here input is taken of records in Input Section, with the click of change button, records are encrypted and on click of Save button, Insert Query is fired.

4.8.4 Searching

Searching of records is done, so that user can see all column values of particular record. Here empid (employee id) needs to be given as input.

User can just see the records and also updation and deletion of records can be done.

4.8.5 Updation

In case User/Operator needs to modify/ change/ update a particular value or values of particular record, first with the help of searching record need to be displayed, then click in textboxes of particular column values of record and modify them and then click on “Update” button.

4.8.6 Deletion

In case User/Operator needs to delete a particular value or values of particular record, first with the help of searching record need to be displayed, then click on “Delete” button.

4.8.7 Sorting
Many a times there is need for displaying of records, in arranged form one after the other, because during insertion of records if they are displayed with the help of “Show All” button, the new record is displayed in the end. Here sorting operation is performed, column wise. Like for example there are many new records inserted and operator just wants to arrange the records according to empid, so operator just needs to click on Column Heading as empid. If the operator needs to arrange records according to empname, then user needs to click on Column Heading empname, and so on.

4.8.8 Retrieval of Records

Records can be retrieved back from database whenever required. This can be done by select query.

4.9 DEVELOPMENT OF VARIOUS PROGRAMMING CONSTRUCTS

Now the development of various programming constructs is being discussed here. Access Specifiers of C# are Public, Private, Protected, Internal[162].

Public – Method can be accessed from anywhere, including outside the class.

Protected – Method can be accessed from within the class to which it belongs, or a type belongs to that class.

Internal – Method can be accessed from within the same program

Private – Method can only be accessed from inside the class to which it belongs.
In the algorithm of ENCR or the algorithm for encryption, here input is taken in
textboxes which are mentioned as txta,txtb,txtc,txtd and txte. The text within these
textboxes can be accessed as txta.text, txtb.text, txtc.text ,txtd.text , txte.text . First of all,
a user enters text in textboxes as shown in INPUT VALUES SECTION of Figure. 4.9(5).
After entering details when user click on change button btnEncrypt_Click method gets
invoked as given in 4.10.1 Code for encryption under Algorithm Section. Here for
designing of form, a panel has been placed, a control used as a placeholder for many
controls. Now a code is given as

txta1.Text = encrypt(txta.Text);

Here the INPUT VALUES SECTION textboxes are given control id as txta, txtb and so
on as mentioned above. Textboxes of  CHANGED VALUES SECTION are mentioned
as txta1,txtb1,txtc1,txtd1,txte1 for empid,ename,deptname,empsal,empholiday etc. Here
whatever code user has entered in txta for empid, it will be accessed as txta.text then
encrypt method will be called, and the output values will be assigned to textbox txta1
using text property as txta1.text. So the output of encrypted values will be displayed in
textbox txta1, so will be in other textboxes.

Working of Encrypt Method is here as follows. First of all, in input values section as the
user/operator enters values in the five textboxes and click on change button the encrypt
function is called. The method declaration here –

\[
\text{public static String encrypt(String str)}
\]

indicates method name as encrypt and parameter type String and variable str which will
be used to give values to the method as input. So from here the values will be passed to
statement
String o = "";

The actual values given by user/operator is contained by parameter str and String type variable o contains an empty string. The next statement

for (int i = 0; i < str.Length; i++)

which is a for loop and integer type variable i serve as loop control variable whose value is first initialised with i=0 in initialization section. Then in test condition section loop will run until i< str.Length or value of the variable i is less than Length of variable str. Length is property used as String_variable_name.Length and it returns the numeric value and up to this numeric value, a loop will run. Next, in increment counter, the i variable is incremented.

With the statements

int n = 0, x = 0;

variable n and x were declared of integer and the default value is set to 0(zero). Then

n = (int)(str[i]);

indicates the textbox values that were passed to method and parameter was str that is a String type variable. Here above statement indicates that string variable str is type cast into integer type and then assigned to integer type variable n (String variable is converted index by index to integer and assigned to n). Here n indicates ASCII values(numeric) of character of String Then further statements

if (n % 2 == 0)
    x = n * 12;
else
    x = n * 13;
If \( n \) mod 2 = 0 or ASCII values are even then it is multiplied by 12 and new value is assigned to integer variable \( x \), else if \( n \) mod 2 does not result in zero then it is odd and \( n \) is multiplied by 13 and a new value is assigned to integer variable \( x \).

\[
o = o + (\text{char})x;
\]

This statement indicates conversion of \( x \) into character and then assigned to variable \( o \). Now as declared previously, \( o \) is a String variable. After processing when for loop terminates, variable \( o \) is returned.

\[
\text{if} \ (n == 48)
\]

\[
n = 126;
\]

Here the statement above indicates that if ASCII value is 48 convert it to 126. This is needed for correct encryption and decryption.

Under 4.10.2 Code for Save to database the working of Save button is given. A user/operator working with forms when on clicking on change button the encrypted values appear in Changed values section as shown in Figure. 4.9 (6) Encryption and Save Data Form. Now on click of Save button, method \( \text{btnSave\_Click} \) gets invoked. SqlConnection is made. String type variable \( ss \) is taken which contains insert query.

\[
\]

INSERT QUERY SYNTAX:
Insert into <Table name> [<Column List>]

Values (<Value List>);

As here table name is employetable and values to be inserted will appear in textboxes after encryption. Now, this insert query will be executed. The connection will be closed. Then a message on website will appear that records are inserted. Algorithm Shift_On_Server is given.

**Description of code for decryption**

Under 4.10.2 Code for Decryption which is part of section 4.10 ALGORITHM, code for method decrypt is given. Just as method encrypt was working, same here too

```csharp
String o = "";
for (int i = 0; i < str.Length; i++)
{
    int n = 0, x = 0;
    n = (int)(str[i]);
    a string variable o is taken this string is empty. Then i is loop control variable which runs from 0(zero) to string length and incrementation of I will be done. Integer variable n and x will be initialized to zero. Variable str will be typecasted to char and assigned to n (same as encrypt method). But due to the method is decryption there is need to convert from ASCII code value to String type values, the following statements
    if (n % 2 == 0)
        x = n / 12;
    else
```
91

\[ x = \frac{n}{13}; \]

Here describes that if \( n \mod 2 \) is equal to zero or if \( n \) is a mod then \( n \) value is divided by 12 and assigned to integer \( x \).

\[ o = o + (\text{char})x; \]

Here \( x \) is converted to char and assigned to variable \( o \) as before.

When encrypted values appear in textboxes in CHANGED VALUES SECTION and user/operator clicks on Save button see Figure. 4.11(6), values got saved to the database. When the user clicks on Show DB Values for this record, database values for this record will be decrypted and displayed as shown in Figure. 4.11(7).

When user/operator clicks on Show DB Values for this record button, method `btnPanelDecrypt_Click` gets activated. Here values appearing in textboxes of changed values section are passed to the method. Here statements

\[
\text{txta2.Text} = \text{decrypt(txta1.Text)};
\]

Implies values of textboxes where encryption values appear `txta1`,`txtb1`,`txtc1`,`txtd1`,`txte1` are passed to decrypt method and the output is assigned to textboxes of decryption values `txta2`,`txtb2`,`txtc2`,`txtd2`,`txte2`. Algorithm for same is given.

**Description of code for retrieval of data or Query Execution**

As depicted in Figure.4.11(9) Query Execution form is given, where for retrieval select query is used.

Working of Select Query is as

Select Clause:
Select Column name

from tablename

where <condition>

To make above query user defined for the user or operator at some places where the user/operator needs to fill in values, at that places textboxes have been placed so as the input of that particular part of the query can be taken according to user/operator, so it becomes user defined query. Here the query framework is as follows:

Select

From tablename

Where

Equals to value

After this 2 buttons are placed one is Query Submit and the other is Show All. Suppose user gives value ‘*’ in textbox after select and then click on Query Submit button, method btnqsubmit_Click gets invoked. Refer 4.10.4 Code for submit button under 4.10 Algorithm. The statements under method,

```
String enstr = encrypt(txtqval.Text);
```

Here the value entered by the user in the txtqval, 3rd textbox that is, textbox provided for Equals to value is passed to encrypt method and result is assigned to enstr String variable. After this SqlConnection is established. Then further statements-

```
if (txtqcolval.Text == "")
```
ss = "SELECT " + txtqcols.Text + " FROM [employeetable] ";

else

ss = "SELECT " + txtqcols.Text + " FROM [employeetable] WHERE " + txtqcolval.Text + " = N'" + enstr.ToString() + "'";

Txtqcolval is textbox provided after 'Where' in Figure.4.11(9) so here any column name whose value will be specified in txtqval can be given for the search, txtqcols is textbox provided after 'Select' where all columns or few column names can be given.

If txtqcolval contains no text or the textbox is empty then Select statement will be

ss = "SELECT " + txtqcols.Text + " FROM [employeetable] "

which implies the statements can be as follows - Select * from employeetable, select empid, empname from employeetable . Here for query formation variables like txtqcols.Text have been taken so that user/operator can provide the value at run time so that here Select query also can serve as user defined query.

But if txtqcolval textbox is not empty then

ss = "SELECT " + txtqcols.Text + " FROM [employeetable] WHERE " + txtqcolval.Text + " = N'" + enstr.ToString() + "'";

becomes true. Here values for textboxes txtqcols, txtqcolval, txtqval is taken at the run time. Values which are input in txtqval are already encrypted and assigned to enstr String variable. Here also user/operator can provide the value at run time so that here Select
query also can serve as user defined query. After that showdata() method is called which will be discussed after.

When Show All button is clicked ,btnqshow_Click method is invoked. Then SqlConnection is established . Then further statements –

```
ss = "select * from [employeetable]"

showdata();
```

Then Select Query statement is assigned to String ss. Then showdata() method is called.

**Description of ShowData( ) Method**

Here SqlConnection is established. Communication between Dataset and SQL database is provided by SqlDataAdapter, which is also a part of ADO.NET Provider. Also, it resides in System.Data.SqlClient namespace. SqlDataAdapter Object is also used in combination with DataSet Object.

In ADO.Net Data set consist of a collection of zero or more tables represented by DataTable objects. In other words, for one or more DataTable objects which may contain data which is derived from the database. DataSet acts as a container. The DataTable Collection contains all DataTable objects in a DataSet. In the System.Data namespace DataTable is defined. DataTable represents a single table of memory resident data. With the help of DataAdapter object, DataTables can be filled up in DataSet. The DataAdapter method has a method Fill() which can be used for filling up data in DataSet.

In the statements rs is SqlDataAdapter, DataTable dt is populated with fill method of SqlDataAdapter. DataSet ds is declared, where new tables have been added.
DataTable dt = new DataTable();

rs.Fill(dt);

DataSet ds = new DataSet();

ds.Tables.Clear();

ds.Tables.Add(dt);

Further statements

for (int i = 0; i < dt.Rows.Count; i++)
{
    for (int j = 0; j < dt.Columns.Count; j++)
        dt.Rows[i][j] = decrypt(dt.Rows[i][j].ToString());
}

GridView1.DataSource = dt;

GridView1.DataBind();

Here in for loop, loop control variable, i will run till Data Table dt, the row count is less than i. Also, j, also a loop control variable will run till Data table dt, Column Count is less than j. After that values in data table dt are converted to String type and decrypted back, then they are assigned to Data Table. Then GridView Tool of Asp.Net is linked to it so at run time Grid View shows the output.

Description of sorting method
When the user needs to sort the data, the column's heading is clicked, then method GridView1_Sorting gets invoked. SqlConnection is established.

```csharp
SqlDataAdapter rs = new SqlDataAdapter(ss, cnn);
DataTable dt1 = new DataTable();
```

With the above statements, SqlDataAdapter rs and Data Table dt1 are used.

```csharp
rs.Fill(dt1);
DataSet ds = new DataSet();
ds.Tables.Clear();
ds.Tables.Add(dt1);
ds.Tables.Clear();
ds.Tables.Add(dt1);
```

These statements are used to make Data Set and Data Table.

```csharp
for (int i = 0; i < dt1.Rows.Count; i++)
{
    for (int j = 0; j < dt1.Columns.Count; j++)
        dt1.Rows[i][j] = decrypt(dt1.Rows[i][j].ToString());
}
```

Here in for loop, loop control variable, i will run till Data Table dt1, the row count is less than i. Also, j a loop control variable will run till Data table dt1, Column Count is less
than j. After that values in data table dt1 are converted to String type and decrypted back, then they are assigned to Data Table.

Further statements are as follows -

```csharp
DataView dv = new DataView(dt1);
dv.Sort = e.SortExpression + " " + "ASC";
GridView1.DataSource = dv;
GridView1.DataBind();
```

Several views of data stored in Data Table are provided by Data View in ASP.NET. With the help of Data View, data in a table can be exposed with different sort orders using a data view. Also, the data can be filtered based on filter expression or by row state. So the views of data from Data Table can be customized. Here when the user/operator clicks on the column heading, sorting is performed in ascending order. Then GridView Tool of Asp.Net is linked, so at run time Grid View shows the output.

4.10 ALGORITHM

4.10.1 CODE FOR ENCRYPTION

```csharp
protected void btnEncrypt_Click(object sender, EventArgs e)
{
    if (Panel1.Visible == true)
    {
    
```
public static String encrypt(String str)
{
    String o = "";

    for (int i = 0; i < str.Length; i++)
    {
        int n = 0, x = 0;
        n = (int)(str[i]);
        if (n == 48) n = 126;
        if (n % 2 == 0) x = n * 12;
        else x = n * 13;
        o = o + (char)x;
    }
}
return o;
}

ALGORITHM

ENCR(String Str)

Initialize A to ‘ ‘;

FOR i=0 to String Length

    Initialize n to 0
    Initialize x to 0
    Set n ← Str[i]

If (n % 2==0)

    SET x ← n*12;

Else

    SET x ← n *13;

    SET A ← A+ x;

Return A

4.10.2 CODE FOR SAVE TO DATABASE

protected void btnSave_Click(object sender, EventArgs e)
{

SqlConnection cnn = new SqlConnection();

cnn.ConnectionString = "Data Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\dell\Documents\check.mdf;Integrated Security=True;Connect Timeout=30;User Instance=True";

cnn.Open();

String ss;


SqlCommand cmd = new SqlCommand(ss, cnn);

cmd.ExecuteNonQuery();
cnn.Close();

Response.Write(" record affected... ");

}

ALGORITHM

SHIFT_ON_SERVER ()
{

    Declare txta1,txtb1,txtc1,txtd1,txte1;

    String ss;

    Set ss <= “Insert into relation values (txta1,txtb1,txtc1,txtd1,txte1)”;

    Execute Query;

}
ShowData()
{
    Relation dt;
    For i=0 to Rows_Count
        For j=0 to Columns_Count
            dt.Rows[i][j] <= decrypt(dt.Rows[i][j]);
}
QShow()
{
    ss <= "select * from emptable";
    showdata();
}

4.10.3 CODE FOR DECRYPTION

public static String decrypt(String str)
{
    String o = "";
    for (int i = 0; i < str.Length; i++)
    {
        int n = 0, x = 0;
        n = (int)(str[i]);
        if (n % 2 == 0)
\[
x = \frac{n}{12};
\]

else
\[
x = \frac{n}{13};
\]

if ((char)x == '~')
\[
x = 48;
\]
\[
o = o + (\text{char})x;
\]

}

return o;

}

protected void btnPanelDecrypt_Click(object sender, EventArgs e)
{

if (Panel2.Visible == true)
{

    txta2.Text = decrypt(txta1.Text);
    txtb2.Text = decrypt(txtb1.Text);
    txtc2.Text = decrypt(txtc1.Text);
    txtd2.Text = decrypt(txtd1.Text);
    txte2.Text = decrypt(txte1.Text);

}

}
ALGORITHM

DECR (String Str)

Initialize A to "";

FOR i=0 to String Length

Initialize n to 0

Initialize x to 0

Set n ← Str[i]

If (n % 2==0)

SET x← n/12

Else

SET x← n /13

SET A← A+ x

Return A

4.10.4 CODE FOR SUBMIT BUTTON

protected void btnqsubmit_Click(object sender, EventArgs e)
{
    String enstr = encrypt(txtqval.Text);

    SqlConnection cnn = new SqlConnection();
cnn.ConnectionString = "Data Source=.;\SQLExpress;AttachDbFilename=C:\Users\dell\Documents\che.mdf;Integrated Security=True;Connect Timeout=30;User Instance=True";
cnn.Open();
if (txtqcolval.Text == "")
    ss = "SELECT " + txtqcols.Text + " FROM [employeetable] ";
else
    ss = "SELECT " + txtqcols.Text + " FROM [employeetable] WHERE " +
txtqcolval.Text + " = N" + enstr.ToString() + ";"
TextBox1.Text = ss;
    showdata();
}
public void showdata()
{
    SqlConnection cnn = new SqlConnection();

    cnn.ConnectionString = "Data Source=.;\SQLExpress;AttachDbFilename=C:\Users\dell\Documents\che.mdf;Integrated Security=True;Connect Timeout=30;User Instance=True";
cnn.Open();
SqlDataAdapter rs = new SqlDataAdapter(ss, cnn);
DataTable dt = new DataTable();
    rs.Fill(dt);
DataSet ds = new DataSet();

ds.Tables.Clear();
ds.Tables.Add(dt);

for (int i = 0; i < dt.Rows.Count; i++)
{
    for (int j = 0; j < dt.Columns.Count; j++)
        dt.Rows[i][j] = decrypt(dt.Rows[i][j].ToString());
}

GridView1.DataSource = dt;
GridView1.DataBind();

protected void btnqshow_Click(object sender, EventArgs e)
{
    SqlConnection cnn = new SqlConnection();

cnn.ConnectionString = "Data Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\dell\Documents\che.mdf;Integrated Security=True;Connect Timeout=30;User Instance=True";

cnn.Open();

    ss = "select * from [employeetable]";
    showdata();
protected void GridView1_Sorting(object sender, GridViewSortEventArgs e)
{
    SqlConnection cnn = new SqlConnection();
    cnn.ConnectionString = "Data Source=.;AttachDbFilename=C:\Users\dell\Documents\che.mdf;Integrated Security=True;Connect Timeout=30;User Instance=True";
    cnn.Open();
    ss = TextBox1.Text;
    SqlDataAdapter rs = new SqlDataAdapter(ss, cnn);
    DataTable dt1 = new DataTable();
    rs.Fill(dt1);
    DataSet ds = new DataSet();
    ds.Tables.Clear();
    ds.Tables.Add(dt1);
    ds.Tables.Clear();
    ds.Tables.Add(dt1);
    for (int i = 0; i < dt1.Rows.Count; i++)
    {
        for (int j = 0; j < dt1.Columns.Count; j++)
        {
            dt1.Rows[i][j] = decrypt(dt1.Rows[i][j].ToString());
        }
    }
    DataView dv = new DataView(dt1);
dv.Sort = e.SortExpression + " " + "ASC";

GridView1.DataSource = dv;
GridView1.DataBind();

4.10.5 CODE FOR SEARCHING OPERATION

protected void btnSearch_Click(object sender, EventArgs e)
{
    string enstr = encrypt(txtSearch.Text);
    ss = "SELECT * FROM [employeetable] WHERE empid = N'" + enstr.ToString() + "';
SqlConnection cnn = new SqlConnection();
cnn.ConnectionString = "Data Source=.;SQLEXPRESS;AttachDbFilename=C:\Users\dell\Documents\che.mdf;Integrated Security=True;Connect Timeout=30;User Instance=True"
    cnn.Open();
SqlDataAdapter rs = new SqlDataAdapter(ss, cnn);
DataTable dt = new DataTable();
rs.Fill(dt);
DataSet ds = new DataSet();
ds.Tables.Clear();
ds.Tables.Add(dt);
if (dt.Rows.Count > 0)
{
    txtax.Text = decrypt(dt.Rows[0][0].ToString());
    txtbx.Text = decrypt(dt.Rows[0][1].ToString());
    txtcx.Text = decrypt(dt.Rows[0][2].ToString());
    txtdx.Text = decrypt(dt.Rows[0][3].ToString());
    txtex.Text = decrypt(dt.Rows[0][4].ToString());
}

ALGORITHM SEARCHING
SEARCH()
{
    String enstr,ss;
    Declare txtSearch;
    Enstr <= encrypt(txtSearch.txt);
    Set ss <= “Select all column values from employetable where empid =
enstr.toString( )”;
    Sql Connection established;
    rs <= new SqlDataAdapter
    dt <= new DataTable
    ds <= new Data Set
    ds.Tables.Add(dt);
if(dt.Rows.Count > 0)
{
    Txtax.Text <= decrypt (dt.Rows);
    Txtbx.Text <= decrypt(dt.Rows);
   Txtcx.Text <= decrypt(dt.Rows);
    Ttxtdx.Text <= decrypt(dt.Rows);
    Txtex.Text <= decrypt(dt.Rows);
}

4.10.6 CODING FOR UPDATE OPERATION

protected void btnUpdate_Click(object sender, EventArgs e)
{
    SqlConnection cnn = new SqlConnection();
    cnn.ConnectionString = "Data
Source=.;AttachDbFilename=C:\Users\dell\Documents\che.mdf;
Integrated Security=True;Connect Timeout=30;User Instance=True";
    cnn.Open();
    String ss;
    ss = "update employeetable set empID = N" + encrypt(txtax.Text).ToString() + ",
    empname=" + encrypt(txtbx.Text).ToString() + ",
depname=" + encrypt(txtcx.Text).ToString() + ",
empsal=" + encrypt(txtdx.Text).ToString() + ",
empholiday=" + encrypt(txtex.Text).ToString() + " where empid = N" + encrypt(txtSearch.Text) + ";";}
SqlCommand cmd = new SqlCommand(ss, cnn);

cmd.ExecuteNonQuery();

cnn.Close();

}
cnn.ConnectionString = "Data Source=.;Intergated Security=True;Connect Timeout=30;User Instance=True"

cnn.Open();

String ss;

ss = "delete from employeetable where empid = N" + txtax.Text + ";"

SqlCommand cmd = new SqlCommand(ss, cnn);

cmd.ExecuteNonQuery();

cnn.Close();

Response.Write(" record affected...");

}

ALGORITHM DELETE

DELETE ( )

{

    Txtax.text <= encrypt(txtSearch.Text);

    String ss;

    ss <= “ Delete from employeetable where empid = (txtax.text);

    Execute Query;

    Response.Write(“Record affected “);

}

4.10.8 CODING FOR SORTING OPERATION
protected void GridView1_Sorting(object sender, GridViewSortEventArgs e)
{
    SqlConnection cnn = new SqlConnection();
    cnn.ConnectionString = "Data Source=.;AttachDbFilename=C:\Users\dell\Documents\che.mdf;Integrated Security=True;Connect Timeout=30;User Instance=True"
    cnn.Open();
    ss = TextBox1.Text;
    SqlDataAdapter rs = new SqlDataAdapter(ss, cnn);
    DataTable dt1 = new DataTable();
    rs.Fill(dt1);
    DataSet ds = new DataSet();
    ds.Tables.Clear();
    ds.Tables.Add(dt1);
    ds.Tables.Clear();
    ds.Tables.Add(dt1);
    for (int i = 0; i < dt1.Rows.Count; i++)
    {
        for (int j = 0; j < dt1.Columns.Count; j++)
            dt1.Rows[i][j] = decrypt(dt1.Rows[i][j].ToString());
    }
    DataView dv = new DataView(dt1);
dv.Sort = e.SortExpression + " " + "ASC";
GridView1.DataSource = dv;
GridView1.DataBind();

ALGORITHM SORTING

SORTING()
{
    Sql Connection established;
    Ss <= TextBox1.Text;
    rs<= SqlDataAdapter;
    dt1 <= new Data Table;
    ds <= new Data Set ;
    ds.Tables.Add(dt1);
    For i=0 to dt1.Rows.Count;
    {
        For j=0 to dt1.Rows.Count
        {
            dt1.Rows[i][j] = decrypt(dt1.Rows[i][j]);
        }
    }
    dv<= new Data View;
    dv.Sort <= e.SortExpression"ASC" ;
    GridView1.DataSource <= dv;
GridView1.DataBind();

4.11 EXPERIMENTAL RESULTS

FORMS

Forms depict Graphical User Interface (GUI) window in programming which is component based. Users can query to the server. Forms provide an interface to the user where the user can interact, provide input and get output accordingly.

FORM DESIGN:-

Now the form design will be discussed. This is the first form, all forms are made using asp dot net framework with SQL server as backend. Language that has been taken up for code is C# (pronounced as C Sharp). With the help of forms, input has been taken from user or operator. Forms are processed with help of tables. Here employee table is taken. As the forms provide an interface to the user, an input is taken then after processing data goes to tables and from tables with the help of forms data is taken out.
Figure 4.9(1): First form to input values
Here there are two columns, one for input values and another for changed values. Input values section takes the input of five fields which are empid (employee id or unique number of employee given by organization), empname (name of employee), deptname (name of department in which employee works), empsal (salary of employee in thousands) and empholiday (holidays of employee for a month) like 0, 1, 2 etc. It is shown in Figure 4.9(1).

After entering the values, there is CHANGE button, as records are not to be saved directly as user input is given. After clicking on change button the input values get encrypted or get converted to another form and these changed values get shown on changed values section.

Changed values section consists of five textboxes of empid, ename, deptname, empsal, and empholiday, these are same as Input Section but here encrypted values are shown after a click of change button. Values are encrypted internally. After that below the changed section, there is Save button. On click of Save button, the values get saved to the database.

In usual working just as input is taken there is a tendency to save to database but here data is encrypted just because the cloud data on which work is done, will be stored on the server is not safe so here as from server data can be leaked so original data is not provided to the server. If user wishes to see data just inserted, back from database that is also possible as shown in Figure 4.9(2).
Form to show decryption value of record which is just saved to database

Figure 4.9(2) : Form to show decrypt values
Figure 4.9 (3): Database Design
Figure 4.9(4): Encrypted Database Values
DATABASE DESIGN: The actual database design and encrypted database is shown here in the Figure 4.9(3) and Figure 4.9(4).

WORKING THROUGH THE FOLLOWING FIGURE:

In the next following figure 4.9(5), it is seen that in Input values section entered values are empid: 102, empname: Shiv, Deptname: sales, Empsal: 12000, Empholiday: 0. This is a very basic step but as these values are directly not saved to the database. On click of Change button, they are converted to an encrypted form in Changed Values section as shown in next figure 4.9(6). With the click of Save Button, encrypted values which are shown in textboxes in form as shown get saved.

After that, there is a choice given to user or operator that if the user wishes the database values just saved user can view in decrypted form. Here as the user clicks the "Show DB Value for this record " button and so database values in decrypted form are shown, first blank form is shown Figure.4.9(2).

In Figure.4.9 (6), in encryption and save data form, the values entered are encrypted, after that in Figure. 4.9(7) that entered values which are saved to the database are decrypted back and shown here. In Figure. 4.9(6), just as values are encrypted and saved to the database, likewise several values can be entered as shown in Figure. 4.9(8).

Till here input of data, encryption and insertion of records to the database is shown here.
Figure 4.9(5): Entering of user details
On click of Save button, values are saved to database

Message displayed when record got saved

Figure 4.9(6): Encryption and Save Data Form
Values saved to database just now can be retrieved back.
Eg. See here the record values just saved in previous figure and check from here too.

Figure. 4.9(7): Database values decrypted
Here in Input Values section, a record which is to be inserted is given

Encrypted Values given as an example

Figure 4.9 (8): Encryption & Save Data Form(2)
In figure 4.9(9), in query execution section, working of Select Query is as

Select Clause:
Select Column name
from tablename
where <condition>

To make above query user defined for the user or operator at some places where the user/operator needs to fill in values, at that places textboxes have been placed so as an input of that particular part of the query can be taken according to user/operator, so it becomes user defined query. Here the query framework is as follows:

Select
From tablename
Where
Equals to value

Take a simple example to form query:

(a) Select empname, empid,
From emp
Where empid = 10;

This can be run using the forms.Here the query can be placed as-

Select empname, empid
From tablename
Where empid
Equals to value 10
As for the above running of the query, the tablename has been specified beforehand in code so here it has not been mentioned. It has been taken as default.

Another example can be taken,

(b) Select *  
From emp

This query statement means that all column names of table emp should be displayed. Here * means all columns. At this place in the last example, few column names from the table were mentioned.

The same can be specified as -

```
Select *  
From tablename  
Where  
Equals to value
```

Result and working through forms have been displayed in Figure 4.9(9). In this form, it can be noticed that when the command is given as written above the output showing tables all columns empid, empname, deptname, empsal, empholiday is mentioned in figure 4.9(9).
Figure 4.9 (9): Result

User Defined Query Execution

Show All Values decrypts full table and displays in front of user

Query Execution section where data can be retrieved according to requirement.
(c) Select *
From emp
Where deptname = 'sales';

Here in this query where it is asked to display all records of emp table where dept number is equal to sales. This is also shown in Figure 4.9(10), where input for query is given

Select *
From tablename
Where deptname
Equals to value sales

Likewise in Figure 4.9(11), input for the query is given as the user/operator wants to display all records from table employee where deptname is purchase. For the query-

(d) Select *
From emp
Where deptname = 'purchase';

This query can be shown in Figure. Also, the results are displayed along with input to query.

Select *
From tablename
Where deptname
Equals to value purchase

Next, for the query given below the input and output are displayed in Figure 4.9(11) –
Query Submit Button to fire user defined queries. It displays particular answer according to query.

Display values from employee table whose department name is sales.

Figure 4.9(10): Result
Query to display only those records whose department name is purchase

Results displayed

Figure 4.9(11): Result
e) Select * 
From emp 
Where empsal = '10000';

For the query given below the input and output are displayed in Figure 4.9(12) -

Here the query framework is as follows:

Select 
From tablename 
Where 
Equals to value 

f) For the query given below the input and output are displayed in Figure 4.9(13) -

Select * 
From emp 
Where empid = 107;

g) For the query given below the input and output are displayed in Figure 4.9(14) -

Select * 
From emp 
Where empholiday = 0;

h) For the query given below the input and output are displayed in Figure 4.9(15) -

Select empid 
From emp 
Where empholiday = 0;
Select only those records whose employee salary is 10000

Figure 4.9 (12): Result
Figure 4.9(13): Result
Figure 4.9(14): Result
Figure 4.9(15): Result

Query to display only those employee id whose employee holiday is zero

Result displayed
i) For the query given below the input and output are displayed in Figure 4.9(16) -

```sql
Select empid,empname,deptname
From emp
Where empholiday=0;
```

j) For the query given below the input and output are displayed in Figure 4.9(17) -

```sql
Select *
From emp
Where empholiday=3;
```

k) For the query given below the input and output are displayed in Figure 4.9(18) -

```sql
Select empid,empname,deptname
From emp
Where empholiday=2;
```

l) For the query given below the input and output are displayed in Figure 4.9(19) -

```sql
Select *
From emp
Where empid=115;
```

m) For the query given below the input and output are displayed in Figure 4.9(20) -

```sql
Select *
From emp
Where empid=115;
```
Figure 4.9(16): Result

Query to display empid, ename and deptname of employees where empholiday is zero.
Result displayed

Query to display all column values of employees where `emp_holiday` is 3.

Figure 4.9(17): Result
Query to display empid, empname, deptname of employees where empholiday is 2.

Result displayed

Figure 4.9(18): Result
Figure 4.9(19): Result

Query to display all column values of employees where empid is 115

Result Displayed
Figure 4.9(20): Result

Result Displayed

Query to display all column values of employees where deptname is management
Select * 

From emp 

Where deptname=management;

o) For the query given below the input and output are displayed in Figure 4.9(21) - 

Select empid,empname 

From emp;

Searching – As previously stated, records can be searched according to employee id. In Figure 4.9(22), we can see that textbox is provided to enter employee id. There are textboxes provided, as the record corresponding to the emp id will be fetched and displayed.

Updation – With the help of updation operation, any modification to the values can be done. First search the record, then do the corrections and click update as shown in Figure. 4.9(22-25).

Deletion – Records will be searched, then unwanted records can be deleted by clicking on delete button as shown in Figure. 4.9(26-27).

Sorting – Display all records by Show All button. Records can be sorted according to any column value, just click on column heading and records will be sorted according to that as shown in Figure.4.9(28-30).
Query to display only empname and empid

Figure 4.9(21): Result
Figure 4.9(22): Searching, Updation and Deletion Form
According to empid 113, searching and displaying record

Figure 4.9(23): Example of searching of record
Figure 4.9(24): Searching and Updating of record value through an example
Figure 4.9(25) : Record Updation through an example
All records displayed through table till the end

Figure. 4.9 (26): Records displayed
Figure 4.9(27): Delete Operation
Record added to illustrate sorting according to empid

Figure. 4.9(28): Addition of record
<table>
<thead>
<tr>
<th>Employee</th>
<th>Department</th>
<th>Salary</th>
<th>Empid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beena</td>
<td>Management</td>
<td>25000</td>
<td>1</td>
</tr>
<tr>
<td>Hari</td>
<td>Management</td>
<td>25000</td>
<td>1</td>
</tr>
<tr>
<td>Om</td>
<td>Sales</td>
<td>15000</td>
<td>1</td>
</tr>
<tr>
<td>Manoj</td>
<td>Purchase</td>
<td>10000</td>
<td>2</td>
</tr>
<tr>
<td>Khushi</td>
<td>Purchase</td>
<td>10000</td>
<td>1</td>
</tr>
<tr>
<td>Seema</td>
<td>Sales</td>
<td>10000</td>
<td>3</td>
</tr>
<tr>
<td>Amit</td>
<td>Purchase</td>
<td>10000</td>
<td>1</td>
</tr>
<tr>
<td>Sharmishtha</td>
<td>Sales</td>
<td>10000</td>
<td>0</td>
</tr>
<tr>
<td>Hina</td>
<td>Manufacture</td>
<td>10000</td>
<td>0</td>
</tr>
<tr>
<td>Lakshya</td>
<td>Sales</td>
<td>12000</td>
<td>1</td>
</tr>
<tr>
<td>Rishabh</td>
<td>Marketing</td>
<td>15000</td>
<td>1</td>
</tr>
<tr>
<td>Vinita</td>
<td>Marketing</td>
<td>20000</td>
<td>0</td>
</tr>
<tr>
<td>Aastha</td>
<td>Marketing</td>
<td>20000</td>
<td>0</td>
</tr>
<tr>
<td>Chetan</td>
<td>Marketing</td>
<td>15000</td>
<td>1</td>
</tr>
<tr>
<td>Deepika</td>
<td>Marketing</td>
<td>15000</td>
<td>0</td>
</tr>
<tr>
<td>Girdhar</td>
<td>Management</td>
<td>25000</td>
<td>0</td>
</tr>
<tr>
<td>Beena</td>
<td>Management</td>
<td>25000</td>
<td>0</td>
</tr>
<tr>
<td>Gopal</td>
<td>Sales</td>
<td>15000</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure. 4.9(29): showing records before sorting and after sorting to empid
<table>
<thead>
<tr>
<th>empid</th>
<th>empname</th>
<th>deptname</th>
<th>empsal</th>
<th>emphpday</th>
</tr>
</thead>
<tbody>
<tr>
<td>108</td>
<td>Aashish</td>
<td>purchase</td>
<td>10000</td>
<td>1</td>
</tr>
<tr>
<td>126</td>
<td>Aastha</td>
<td>marketing</td>
<td>20000</td>
<td>0</td>
</tr>
<tr>
<td>120</td>
<td>Amit</td>
<td>purchase</td>
<td>10000</td>
<td>1</td>
</tr>
<tr>
<td>113</td>
<td>Ankit</td>
<td>management</td>
<td>25000</td>
<td>1</td>
</tr>
<tr>
<td>107</td>
<td>Ankita</td>
<td>management</td>
<td>10000</td>
<td>0</td>
</tr>
<tr>
<td>114</td>
<td>Beena</td>
<td>management</td>
<td>25000</td>
<td>1</td>
</tr>
<tr>
<td>130</td>
<td>Beena</td>
<td>management</td>
<td>25000</td>
<td>0</td>
</tr>
<tr>
<td>127</td>
<td>Chetan</td>
<td>marketing</td>
<td>15000</td>
<td>1</td>
</tr>
<tr>
<td>111</td>
<td>Chetna</td>
<td>purchase</td>
<td>17000</td>
<td>1</td>
</tr>
<tr>
<td>128</td>
<td>Deepika</td>
<td>marketing</td>
<td>15000</td>
<td>0</td>
</tr>
<tr>
<td>129</td>
<td>Girdhar</td>
<td>management</td>
<td>25000</td>
<td>0</td>
</tr>
<tr>
<td>131</td>
<td>Gopal</td>
<td>sales</td>
<td>15000</td>
<td>0</td>
</tr>
<tr>
<td>115</td>
<td>Hari</td>
<td>management</td>
<td>25000</td>
<td>1</td>
</tr>
<tr>
<td>122</td>
<td>Hina</td>
<td>manufacture</td>
<td>10000</td>
<td>0</td>
</tr>
<tr>
<td>110</td>
<td>Kavita</td>
<td>manufacture</td>
<td>20000</td>
<td>2</td>
</tr>
<tr>
<td>118</td>
<td>Khushi</td>
<td>purchase</td>
<td>10000</td>
<td>1</td>
</tr>
<tr>
<td>106</td>
<td>Kovid</td>
<td>sales</td>
<td>15000</td>
<td>1</td>
</tr>
<tr>
<td>101</td>
<td>Lakshmi</td>
<td>sales</td>
<td>10000</td>
<td>0</td>
</tr>
<tr>
<td>123</td>
<td>Lakshya</td>
<td>sales</td>
<td>12000</td>
<td>1</td>
</tr>
<tr>
<td>117</td>
<td>Manoj</td>
<td>purchase</td>
<td>10000</td>
<td>2</td>
</tr>
</tbody>
</table>

**Figure. 4.9(30):** Showing records before and after sorting according to empname