CHAPTER 1
INTRODUCTION

1.1 Database: A Formal Definition

A database is an ordered collection of related data elements intended to meet the information needs of an organization and designed to be shared by multiple users. A database is typically available to a community of users, with possibly varying requirements.

1.1.1 Key terms in the definition

1. **Ordered collection** - A database is a collection of data elements. Not just a random assembly of data structures, but a collection of data elements put together deliberately with proper order. The various data elements are linked together in the most logical manner.

2. **Related data elements** - The data elements in a database are not disjointed structures without any relationships among them. These are related among themselves and also pertinent to the particular organization.

3. **Information needs** - The collection of data elements in a database is there for a specific purpose. That purpose is to satisfy and meet the information needs of the organization. In a database for a bank, one will find data elements that are pertinent to the bank’s business. One will find customer’s bank balances and ATM transactions. Any one will not find data elements relating to a student’s major and examination grades that belong in a database for a university.

4. **Shared** - All authorized users in an organization can share the information stored in its database. Integrated information is kept in the database for the purpose of sharing so that all user groups may collaborate and accomplish the organization’s objectives.
DM (database Management) has been evolved from a Computer ‘program’ to a ‘Control component’ of a modern cyber environment. Knowledge of the database management has become an essential part of the computer science because in any of the organisation, for effective decision making, reliable and accurate data is essential. So it can be said that DM is centralized control of information and also prevents unauthorized access to database.

1.1.2 Database Management System (DBMS)
A Database Management system is a collection of interrelated data and set of programs to access this data. A collection of programs that enables one to store, modify, and extract information from a database. Data base management system is the system in which related data is stored in an “efficient” and “compact” manner. Efficient means that the data which is stored in the DBMS is accessed in very quick time and compact means that the data which is stored in DBMS covers very less space in computer’s memory. In above definition the phrase “related data” is used which means that the data which is stored in DBMS is about some particular topic. DBMS can support single user and multiuser environment. Some commercially available DBMS packages are Oracle, Sybase, and Ingress.

1.1.3 Architecture of the DBMS
DBMS is made of several different interrelated components. The way these components work together to achieve some selected goals is referred as system architecture. Database Management Systems are very complex, sophisticated software applications that provide reliable management of large amounts of data. There are two different ways to look at the architecture of a DBMS: Logical architecture and the Physical architecture. The logical architecture deals with the way data is stored and presented to users, while the physical architecture is concerned with the software
components that make up a DBMS.

1. Logical DBMS Architecture

The logical architecture describes how data in the database is perceived by users. It is not concerned with how the data is handled and processed by the DBMS, but only with how it looks. Users are shielded from the way data is stored on the underlying file system, and can manipulate the data without worrying about where it is located or how it is actually stored. This results in the database having different levels of abstraction.

The ANSI/SPARC architecture divides the system into three levels of abstraction: the internal or physical level, the conceptual level, and the external or view level. The Fig. 1.1 shows the logical architecture for a typical DBMS.

![Fig. 1.1 Architecture of Database](image)

i. **The Internal or Physical Level** - The collection of files permanently stored on secondary storage devices is known as the physical database. The physical or internal level is the one closest to physical storage, and it provides a low-level
description of the physical database, and an interface between the operating system's file system and the record structures used in higher levels of abstraction. It is at this level that record types and methods of storage are defined, as well as how stored fields are represented, what physical sequence the stored records are in, and what other physical structures exist.

ii. **The Conceptual Level** - The conceptual level presents a logical view of the entire database in its totality, which allows one to bring all the data in the database together and see it in a consistent manner. The first stage in the design of a database is to define the conceptual view, and a DBMS provides a data definition language for this purpose. It is the conceptual level that allows a DBMS to provide data independence. The data definition language used to create the conceptual level must not specify any physical storage considerations that should be handled by the physical level. It should not provide any storage or access details, but should define the information content only.

iii. **The External or View Level** - The external or view level provides a window on the conceptual view which allows the user to see only the data of interest to them. The user can be either an application program or an end user. Any number of external schemas can be defined and they can overlap each other. The System Administrator and the Database Administrator have responsibilities for the design and maintenance of the database. The external and the conceptual view are functionally equivalent for these two users.

2. **Physical DBMS Architecture**

The physical architecture describes the software components used to enter and process data, and how these software components are related and interconnected. Although it is not possible to generalize the component
structure of a DBMS, it is possible to identify a number of key functions which are common to most database management systems. At its most basic level the physical DBMS architecture can be broken down into two parts: the back end and the front end.

The back end is responsible for managing the physical database and providing the necessary support and mappings for the internal, conceptual, and external levels. Other benefits of a DBMS, such as security, integrity, and access control, are also the responsibility of the back end. The front end is really just any application that runs on top of the DBMS. These may be applications provided by the DBMS vendor, the user, or a third party. The user interacts with the front end, and may not even be aware that the back end exists.

### 1.1.4 Relational Databases

The relational model is the basis for any relational database management system (RDBMS). A relational model has three core components: a collection of objects or relations, operators that act on the objects or relations, and data integrity methods. In other words, it has a place to store the data, a way to create and retrieve the data, and a way to make sure that the data is logically consistent. The relational model was first proposed by Dr. E. F. Codd in 1970. The world of relational database systems is a two dimensional world. Data is stored in tabular data structures where rows correspond to distinct real-world entities or relationships, and columns are attributes of those entities. The relational database model uses the mathematical concept of a relation as its data structure. Data can be depicted in these relations as rows and columns of values in a table. Tuples, or rows in the table, are the elements of the relation. Each tuple contains a value for each of the attributes of the relation. All values within a given column are elements of the domain of the attribute for that column. A domain is a set of atomic
values from which attributes draw their values, often specified in terms of data type and format. Mathematically, a relation can be defined as a Cartesian product of the form $R: A_1, A_2, \ldots, A_n$, where $R$ is a set of tuples of the form $A_1, A_2, \ldots, A_n$ and $Ai$ is an attribute which names a role played by the domain $Di$ defined for each column. A relation is a set of tuples. Consequently, the ordering of the tuples is irrelevant and typical set operations such as UNION, INTERSECTION, CARTESIAN PRODUCT, and SET DIFFERENCE can be applied to relations. The relational algebra defines the operations that can be used in the relational model to express queries, requests to retrieve data from relations in the database. This algebra contains the commonly used SELECT, PROJECT, and JOIN operators in addition to the set operators. The relational algebra has the algebraic property of closure. Any operation applied to one or more relations produces a new relation. It also has the property that the operations DIFFERENCE, UNION, PROJECT, PRODUCT, and SELECT form a complete set. All other relational operations can be defined in terms of these. Therefore, these five operators are sufficient to specify data associated with any relationship in the database design, and query languages having these operations are called relationally complete. Because the relational model has a strong mathematical foundation based on simple structures that are easy to understand and manipulate, it has become very popular and many commercial database packages based on the relational model are available.

1.2 FUZZY LOGIC

The concept of Fuzzy Logic (FL) was conceived by Lotfi Zadeh, a professor at the University of California at Berkley, and presented not as a control methodology, but as a way of processing data by allowing partial set membership rather than crisp set membership or non-membership.
In viewing the evolution of fuzzy logic, three principal phases may be discerned. The first phase, from 1965 to 1973, was concerned in the main with fuzzification, that is, with generalization of the concept of a set, with two-valued characteristic function generalized to a membership function taking values in the unit interval or, more generally, in a lattice. The basic issues and applications which were addressed were, for the most part, set-theoretic in nature and logic and reasoning were not at the center of the stage. During the second phase, 1973-1999, two key concepts were introduced [57]:

1. The concept of a linguistic variable; and
2. The concept of a fuzzy if-then rule.

Today, almost all applications of fuzzy set theory and fuzzy logic involve the use of these concepts. During the third phase (prevailed from 1999 onwards), fuzzy logic is used in two different senses:

1. In a narrow sense, in which fuzzy logic, abbreviated as FLn, is a logical system which is a generalization of multivalued logic;
2. In a wide sense, in which fuzzy logic, abbreviated as FL, is a union of FLn, fuzzy set theory, possibility theory, calculus of fuzzy if-then rules, fuzzy arithmetic, calculus of fuzzy quantifiers and related concepts and calculi.

### 1.2.1 General
Fuzzy logic is only a mathematical tool. It is possibly the best tool for treating uncertain, vague, or subjective information. Particularly, fuzzy logic has been applied to databases in many scientific papers and real applications. Undoubtedly, it is a modern research field and it has a long road ahead.

Fuzzy set theory is the base of fuzzy logic. In this logic, the truth-value of a sentence (or satisfaction degree) is in the real interval [0, 1]. The value 0 represents
completely false, and 1 is completely true. The truth-value of a sentence “s” will be denoted as $\mu(s)$. This logic allows giving an interpretation to linguistic terms:

1. **Predicates**: Predicates (synonym of linguistic labels) are atomic components of this logic defined by a membership function on the fuzzy set. For example, linguistic terms such as “young,” “tall,” “heavy” and “low” are predicates.

2. **Modifiers**: Modifiers linguistic terms that allow defining modified fuzzy predicates, are interpreted by means of transformations of the membership function. In this category are the natural language adverbs, for example, “very,” “relatively,” and “extremely.”

3. **Comparators**: Comparators, kinds of fuzzy predicates defined on pairs of elements, establish fuzzy comparisons; for example, “more greater than,” “approximately equal to,” and “close to.” are fuzzy comparators.

4. **Connectors**: Connectors are operators defined for combining fuzzy sentences. Fuzzy negation, conjunction, disjunction, complement (negation), intersection and union, are extensions of the classical.

5. **Quantifiers**: Quantifiers are terms describing quantities such as “most of,” “about a half,” and “around 20.” They are an extension of classical existential and universal quantifiers. Two types of fuzzy quantifiers are distinguished as Absolute and Proportional (Relative). Absolute quantifiers represent amounts that are absolute in nature such as “about 5” or “more than 20.” An absolute quantifier can be represented by a fuzzy subset $Q$, such that for any non-negative real $p \in \mathbb{R}$, the membership grade of $p$ in $Q$ (denoted by $\mu_Q(p)$) indicates the degree to which the amount $p$ is compatible with the quantifier represented by $Q$. Proportional or relative quantifiers, such as “at least half” or “most,” can be represented by fuzzy subsets defined in the unit interval $[0,1]$. 
For any proportion $p \in [0, 1]$, $\mu_Q(p)$ indicates the degree to which the proportion $p$ is compatible with the meaning of the quantifier.

Fuzzy logic (FL) is aimed at a formalization of modes of reasoning which are approximate rather than exact.

Examples:

Exact
- All men are mortal
- Shiv is a man
- Shiv is mortal.

Approximate
- Most Indians are tall
- Manu is an Indian
- It is likely that Manu is tall

Fuzzy logic has four principal facets:

1. The logical facet, FL/L, is focused on logical systems in which truth is a matter of degree – a degree which is allowed to be a fuzzy set.

2. The set-theoretic facet, FL/S, is concerned, mainly with the theory of fuzzy sets. Most of the mathematical literature on fuzzy logic relates to FL/S.

3. The relational facet, FL/R, is focused on fuzzy dependencies, granulation, linguistic variables and fuzzy rule sets. Most practical applications of fuzzy logic relate to FL/R.

4. The epistemic facet, FL/E, is concerned, mainly with knowledge representation, natural languages, semantics and expert systems. Probabilistic and possibilistic modes of reasoning are a part of this facet as well as FL/L and FL/R.
1.2 Characteristics of Fuzzy Logic

This logic is a multivalued logic [54], the main characteristics are as follows:

1. In fuzzy logic, exact reasoning is considered a specific case of approximate reasoning.

2. Any logical system can be converted into terms of fuzzy logic.

3. In fuzzy logic, knowledge is interpreted as a set of flexible or fuzzy restrictions over a set of variables (e.g., the variable Temperature is Cold).

4. Inference is considered as a process of propagation of those restrictions.

   Inference is understood to be the process by which a result is reached, consequences are obtained, or one fact is deduced from another.

5. In fuzzy logic, everything is a matter of degree.

1.3 Fuzzy Database

A fuzzy database [25] can be defined as a database containing imperfect data, which is generally modeled as fuzzy sets. The imperfect information is found in a variety of
real-life situations. The work [8] includes an excellent discussion about the meaning of the characteristics of imperfect data.

Three types of imperfect information:-

1. **Uncertain Information** Uncertainty arises from the fact that an agent has constructed a subjective opinion about the truth of a fact, which he/she does not know for certainty. Data that are not totally trustworthy, an estimation of its reliability are available. For instance, if we ask the age of Ram to a neighbor, a co-worker, a friend and his mother, and each one gives a different reply, we can assign a reliability degree according to the strength of the relationship between Ram and the person to whom the question is asked.

2. **Imprecise information**: Imprecision arises from the existence of a value, which cannot be measured with suitable precision. When the data is not available in its maximum finest granularity, we have an approximation but not a precise value. In this case, a set or a range of values could be available, among which the actual precise value is unknown. For instance, we know that Ram is between 25 and 28 years old, but we cannot give an exact age for him.

3. **Incomplete Information**: Incompleteness arises from the absence of a value. This kind of data is defined by a gradual predicate. Usually, vague data corresponds to linguistic terms of natural language. For instance, we could know that Ram is middle-aged. The linguistic term middle-aged corresponds to a gradual predicate that is completely incompatible for age values below 30 and above 60 and completely compatible for ages between 35 and 55. The predicate is partially compatible for ages between 30 and 35, and ages between 55 and 60, where the compatibility degree gradually ascends in the former case, and gradually descends in the latter case.
In reality, a lot of information is available only in an imperfect form. This might be due to imprecision, vagueness, uncertainty, incompleteness, or ambiguities. Traditional database systems can only adequately cope with perfect data. It can handle only Binary Logic.

The first attempt to represent imprecise information on databases was the introduction of NULL values by Codd, which was further, expanded [14]. A NULL value in an attribute indicates that such a value is any value included in the domain of such an attribute. Any comparison with a NULL value originates an outcome that is neither True (T) nor False (F) called “maybe” (m) (or unknown, in the SQL of Oracle), called TRI Valued Logic. The truth tables of the classical comparators NOT, AND, and OR can be seen in Table 1.1.

<table>
<thead>
<tr>
<th>NOT</th>
<th>T</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>m</td>
<td>m</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AND</th>
<th>T</th>
<th>m</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>m</td>
<td>F</td>
</tr>
<tr>
<td>m</td>
<td>m</td>
<td>m</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OR</th>
<th>T</th>
<th>m</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>m</td>
<td>T</td>
<td>m</td>
<td>m</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>m</td>
<td>F</td>
</tr>
</tbody>
</table>

Table 1.1 Truth Tables for Trivalued Logic

Later on, another nuance was added, differentiating the NULL value in two marks: The “A-mark” representing an absent or unknown value, although it was applicable, and the “I-mark” representing the absence of the value because it is not applicable (undefined). An I-mark may be situated, for instance, in the car plate attribute of someone who does not have a car. This is a tetra valued logic where the A value, having a similar meaning to that of the m in the tri valued logic mentioned above, is generated by comparing any value containing an A-mark, and a new I value is added as a result of the comparison of any value containing an I-mark. The tetra valued logic
is shown in Table 1.2. In Galindo[25], some other approaches are summarized, like the “default values” approach by Date, similar to the DEFAULT clause in SQL, the “interval values” approach[28], which expands the relational model in order to allow that a possible value range/interval be stored in one attribute, and statistical and probabilistic databases.

Studied under a broad soft computing paradigm, fuzzy techniques have been applied to deal with imperfections of data in a more natural way and to enhance the accessibility of databases. Fuzzy techniques are the mathematical tool and most possibly the best for treating uncertain, vague, or subjective information. Particularly, fuzzy logic has been applied to databases in many scientific papers and real applications. Undoubtedly, it is a modern research field and it has a long road ahead.

<table>
<thead>
<tr>
<th>NOT</th>
<th>T</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>I</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AND</th>
<th>T</th>
<th>A</th>
<th>I</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>A</td>
<td>I</td>
<td>F</td>
</tr>
<tr>
<td>A</td>
<td>A</td>
<td>A</td>
<td>I</td>
<td>F</td>
</tr>
<tr>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OR</th>
<th>T</th>
<th>A</th>
<th>I</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>A</td>
<td>T</td>
<td>A</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>I</td>
<td>T</td>
<td>A</td>
<td>I</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>A</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

Table 1.2 Truth Tables for Tetravalued Logic

Most conventional databases in use today are based on the relational model. Values in a relation are taken from a finite set of strictly typed domain values. Each relation in the database represents a proposition and each record in a relation is a statement such that it evaluates to ‘true’ for that proposition. It could be argued, however, that this required precision actually gives an insufficient representation of the world. The model is grounded in binary black-and-white but much of reality actually exists in shades of gray. As such, the conventional relational database model has limited usefulness
1.3.1 Model of Fuzzy databases

The simplest model of fuzzy relational databases consists of adding a grade, normally in the [0, 1] interval, to each instance (or tuple). Nevertheless, the semantic assigned to this grade will determine its usefulness, and this meaning will be utilized in the query processes. This grade may have the meaning of membership degree of each tuple to the relation [39], but it may mean something different, like the dependence strength level between two attributes, thus representing the relation between them, the fulfillment degree of a condition or the importance degree of each tuple in the relation, among others. The main problem with these fuzzy models is that they do not allow the representation of imprecise information about a certain attribute of a specific entity (like the “Hot” or “cold” values for a “temperature” attribute). Besides, the fuzzy character is assigned globally to each instance making it impossible to determine the specific fuzzy contribution from each constituting attribute. These ideas are implemented in two broadly classified models as under:

1.3.1.1 Similarity relations Model

i. Buckles-Petry Model: This is the first model that utilizes similarity relations in the relational model. It was proposed by Buckles and Petry [10]. In this model, a fuzzy relation is defined as a subset of the following Cartesian product: P (D₁)× ... ×P(Dm), where P(Di) represents the parts set of a Di domain, including all the subsets that could be considered within the Di domain (having any number of elements). The data types permitted by this model are finite set of scalars (labels), finite set of numbers, and fuzzy number set. The meaning of these sets is disjunctive, that is, the real value is one belonging to the set. The equivalence types on a domain are constructed from a similarity function or relation, in which the values taken by such a relation are provided
by the user. Typically, these similarity values are standardized in the [0,1] interval, where 0 corresponds to “totally different” and 1 to “totally similar.” A similarity threshold can be established with a value between 0 and 1 in order to get the values whose similarity is greater than the threshold, or to consider those values indistinguishably.

1.3.1.2 Possibilistic Models

Models using the possibility theory to represent imprecision are also available. The most important models in this group are Prade-Testemale model, Umano-Fukami model, and GEFRED model.

i. Prade-Testemale Model: Prade and Testemale published a fuzzy relational database (FRDB) model that allows the integration of incomplete or uncertain data in the possibility theory sphere [41]. An attribute A, having a D domain, is considered. All the available knowledge about the value taken by A for an x object can be represented by a possibility distribution \( \pi(x)_A \) about \( D \cup \{e\} \), where e is a special element denoting the case in which A is not applied to x. From this formulation, all value types adopted by this model can be represented. In every possibilistic model one must take into account that, for a value \( d \in D \), if \( \pi(x)_A (d) = 1 \), then this just indicates that the d value is totally possible for A(x), and not that the d value is true for A(x), unless this is the only possible value, that is, \( \pi(x)_A (d') = 0 \).

ii. Umano-Fukami Model: This proposal[48] also utilizes the possibility distributions in order to model information knowledge. In this model, if D is the discourse universe of A(x), \( \pi(x)_A (d) \) represents the possibility that A(x) takes the value \( d \in D \). The following kind of knowledge may be modeled:

- Unknown and applicable information,
• The non-applicable information (undefined)

• The total ignorance (do not know if it is applicable or non-applicable):

Unknown = \pi_{A(x)}(d) = 1,

Undefined = \pi_{A(x)}(d) = 0,

Null = \{1/Unknown, 1/Undefined\}

In other words, a fuzzy relation R, with m attributes, is defined as the following membership function:

\mu_R: P(U_1) \times P(U_2) \times \ldots \times P(U_m) \rightarrow P([0,1])

Where the \times symbol denotes the Cartesian product, P(U_j) with j=1, 2, ..., m is the collection of all the possibility distributions in the discourse universe U_j of the j-th R attribute. The function \mu_R associates a P([0,1]) value to every instance of the relation R, which corresponds to all the possibility distributions in the [0,1] interval; this shall be considered an R membership degree of such an instance. Finally, in the query process, expressed either in fuzzy or precise terms, the model solves the query problem by dividing the set of instances involved in the relation into three subsets, where

• The first subset contains the instances completely satisfying the query;

• The second subset groups those instances that might satisfy the query;

• The third subset consists of those instances which do not satisfy the query.

iii. The GEFRED Model: The GEFRED model dates back to 1994 and it has experienced subsequent expansions [25]. This model is an eclectic synthesis of some of the previously discussed models. One of the major advantages of this model is that it consists of a general abstraction that allows for the use of various approaches, regardless of how different they might look. As a
possibilistic model, it refers particularly to generalized fuzzy domains, thus admitting the possibility distribution in the domains, but it also includes the case where the underlying domain is not numeric but scalars of any type. It includes unknown, UNDEFINED, and NULL values as well, having the same sense as that in Umano-Fukami model. The GEFRED model is based on the definition which is called Generalized Fuzzy Domain (D) and Generalized Fuzzy Relation (R), which include classic domains and classic relations, respectively. Basically, the Generalized Fuzzy Domain is the basic domain, with possibility distributions defined for this domain and the NULL value. On the other hand, the Generalized Fuzzy Relations of GEFRED model are relations whose attributes have a Generalized Fuzzy Domain, and each attribute may be associated to a “compatibility attribute” where one can store a compatibility degree. The compatibility degree for an attribute value is obtained by manipulation processes (such as queries) performed on that relation, and it indicates the degree to which that value has satisfied or met the operation performed on it. The GEFRED model defines fuzzy comparators that are general comparators based on any existing classical comparator (>, <, =, etc.), but it does not consolidate the definition of each one. The only requirement established is that the fuzzy comparator should respect the classical comparators outcomes when comparing possibility distributions expressing no fuzzy values (crisp).

1.4 Fuzzy Querying

The term query is widely used in the database as well as information retrieval communities. Fuzzy querying is similar to the process of ordinary querying, with some complexities. A large number of tuples may be selected by fuzzy condition in
comparison to the crisp one. If a record consists of a fuzzy attribute, say height, a query such as “retrieve all tall people” will cause a considerable portion of the database being brought into the memory. Fuzzy querying allows one to express vague predicates represented by fuzzy sets. For example, even when the crisp query produces an empty response, the corresponding fuzzy query can provide an appropriate answer. This is due to the fact that crisp querying is too rigid, whereas fuzzy querying is flexible. Moreover, fuzzy querying also allows us to rank the possible replies whereas the classical crisp querying only provides a list of undifferentiated responses.

In order to manage and access an information source, an appropriate system is defined which makes it possible to store, represent and retrieve information items by means of a formal query language. Information systems that make it possible to manage information items are information retrieval systems, data base management systems and knowledge-based systems, respectively.

The industry standard for the querying of relational databases is the Structured Query Language (SQL).

SQL has commands to deal with all aspects concerning the creation, maintenance and use of a database such as the creation of tables, insertion of rows, querying the database, security issues, etc. The syntax of a basic query in SQL [43] is:

```
SELECT attribute_list
FROM relation_list
WHERE conditions
```

In terms of relational algebra the SELECT, FROM and WHERE clauses correspond to the projection, join and selection operations, respectively. From the fuzzy querying perspective the most interesting is the WHERE clause. The requirements of the information sought are specified and, these conditions very often are prone to
imprecision, vagueness etc. The concept of fuzzy querying may be considered either in
the context of a crisp, traditional database or a fuzzy database.

One of the distinguishing features of fuzzy querying is the concept of a matching
degree belonging to the [0, 1] interval. The fuzzy query evaluation against a crisp
database may be considered as a special case of a more general and complex case of
fuzzy, possibility based databases.

Often, a user faces the problem of how to express her or his information requirements
in a formal query language supported by a given information system interface. These
formal languages usually require a crisp (precise, unambiguous) specification of a
query, while, for human beings, a query is best expressed in terms of a natural
language, a very powerful, but ambiguous and imprecise medium. Thus, adding some
flexibility to traditional querying systems seems to be a critical issue for enhancing
their effectiveness and efficiency.

1.4.1 Fuzziness in Database Querying

Database management systems are meant for the maintenance and processing of highly
structured data. This research is limited to the case of the relational data model. In
the relational database model, the main structure used for the representation of the
class of real world entities is the table. Representation of imperfect information within
the framework of the relational data model can be categorized in three main groups of
fuzzy logic based approach

The first approach is a natural extension to the main notion laying ground for the
relational data model, i.e., the notion of the relation. Namely, instead of the crisp
relation it is postulated to employ the fuzzy relation that allows expressing some
imperfect information about the items represented by particular tuples of such a
relation. The querying of such a fuzzy database relies on the algebra of fuzzy relations.
This approach is mainly undermined by the lack of a clear, unambiguous interpretation of the grade of membership of a tuple in a relation.

Two other approaches preserve the crispness of the relations and represent the imperfect information directly at the level of values of the attributes. The value of an attribute is still assumed to be atomic, single element belonging to the attribute’s domain, but information about this value may be imperfect. In the second approach, proposed by Buckles and Petry, this value is represented as a crisp subset of the attribute’s domain. In the third approach, proposed by Prade and Testemale, this value is represented as a possibility distribution on the attribute’s domain. In the approach of Buckley and Petry the domain of an attribute is additionally equipped with a fuzzy similarity relation indicating how similar to each other all pairs of the values of a given attribute are.

1.5 Organisation of Thesis

The present thesis entitled “Handling Imprecision through Relational Databases Using Fuzzy Techniques” is an attempt to introducing a new concept of handling imprecision using fuzzy techniques. The thesis comprises six chapters which are summarised as follows:

Chapter 1 is introductory in nature which includes origin, history and definition of database and fuzzy application to database and also explains various terms related to the work discussed in subsequent chapter of the thesis. Chapter 2 analyses the imprecision without fuzzy logic and fuzzy database models. It also put the light on the fuzzy approach to database querying and make comparisons on available fuzzy querying. It also discusses the idea of using iris as biometric recognition. Chapter 3 describes the database design for storage of fuzzy information in traditional database. It also discusses GEFRED model and limits of existing work. These limits are
overcome by introducing the Fuzzy Metadata. Chapter 4 discusses Fuzzy database querying. It describes the limitation of SQL. New concept of queries based on fuzzy logic along with principal of functioning is also described. Here a new model is proposed and implemented to handle the weaknesses of existing work. Chapter 5 extends the applications of fuzzy database in the field of Iris Recognition. The purpose of taking the idea of fuzzy database with iris in the chapter is to avoid the possibility of mismatching and overhead occurred during the recognition process. Chapter 6 gives the conclusion of the thesis and a step towards the future work.