International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

An Object-Oriented Database Design for Effective Classification

Ajita Satheesh*[1], Dr. Aarti Kumar[2]

Submitted: 10/09/2022 **Accepted:** 20/12/2022

Abstract*:* Data mining is a science that has been a rapidly emerging art, harnessed to uncover and exploit novel, valuable, and useful relationships in data. Given the increasing reputation of object-oriented databases, it's very critical to look at statistics mining techniques in advanced database programs. It is critical to ensure that object-oriented programming concepts are high-quality addressed through incorporating them into current databases. Object-oriented programming is used considerably to deal with user-defined data types and complicated data. Decision Theory, Machine Learning, and Classification are well-established statistics mining tasks for large-scale studies in the fields of information and literature. Classification is a well-established fact-mining assignment that has been extensively studied within the fields of data, decision concepts, and machine learning literature. This paper demonstrates the layout of object-oriented databases by incorporating object-oriented programming ideas into current relational databases. Simultaneously, an efficient approach to classification is also presented for the data mining task. Object-oriented programming concepts such as polymorphism and inheritance have been employed in the classification and design of databases. Well-equipped methodologies in self-efficient classification have been achieved with the aid of the design of object-oriented databases. The main advantage of this paper is that it employs simple Structured Query Language (SQL) queries. So that effective data classification does not have to resort to complex techniques. This method has been found to have lower implementation overhead compared to traditional databases by significantly reducing the amount of memory space used for storage.

Keywords: *Classification, Data mining, Dynamic Polymorphism, Inheritance, Object-Oriented Databases.*

1. Introduction

In today's modern-era organizations, the amount of data stored in the database is increasing at a very rapid pace. Valuable hidden knowledge is derived from these stored data for effective decision-making. The Discovery of such knowledge is very useful. Developing methods of extracting understanding from the facts saved inside the database is of utmost significance. Data mining entails analyzing information with the assistance of sample recognition technology through the use of statistical and mathematical techniques [1]. It additionally permits meaningful new correlations, promising approaches, and analysis of massive amounts of information stored in repositories. Data class is one of the essential responsibilities of records mining [2]. A Data type is a process in which the magnificence or idea of the fact enables prediction to discover a treasured set of models. Data classification plays an important role in predicting sets of classes that are self-descriptive and with an unknown class label. Data analysis is a specialized technique used to integrate data mining techniques with potential effectiveness [3]. The knowledge is stored to provide a uniform framework for data mining. Large datasets are handled efficiently with the help of this stored knowledge.

[1] *Deputy Secretary, Rajiv Gandhi Proudhyogiki Vishwavidhyalaya,State Technological University, Bhopal 462033*
ORCID ID : 0000-0002-2079-347X
[2] *Professor, Rabindranath Tagore University, Bhopal 464993*
** Corresponding Author Email: ajitasatheesh5@gmail.com*

Relational databases (RDB) play an important role in the retrieval of huge amounts of data [4]. It has been a familiar solution for efficient storage. The static additives of organizational facts evolved from RDB tables. Only easy predefined statistics sorts may be handled by the usage of RDB. Problems arising from multimedia, complicated data types and user-defined information sorts also can be encountered. RDB era fails to fulfill the problems related to complicated statistics systems. The semantics of relational databases can't be achieved without the assistance of users. The best way to overcome these problems is to use an object-oriented database (OODB)[5]. Object-oriented models capture data complexity and semantics based on the concepts of generalization and abstraction. Therefore, object-oriented databases (OODB) are being employed by many research organizations to solve problems.
The concepts of object-oriented languages are used in OODB databases. The main strength of OODB is its potential to deal with interconnected information and complex applications [6]. The present item-orientated database management structures aren't green sufficient to compete inside the market with relational opposite numbers inside the modern-day situation. Many applications are built using existing relational database management systems. Applications can be further improved by incorporating object-oriented concepts. Features can be exploited by employing Object Oriented concepts with RDBMS [7].
An innovative and new object-orientated database design is offered in this paper. OODB is designed in an efficient mode.

This has helped in accomplishing the green category on the database. The proposed approach contains the features of item-orientated programming ideas, including inheritance and polymorphism, to achieve the goals [8]. Polymorphism performs a completely critical role in object-oriented programming standards. Polymorphism is described as "one interface, many implementations". In the preceding literature, 3 distinct forms of polymorphisms had been labeled as pure, stable, and dynamic. Pure polymorphism refers to a function taking parameters of more than one statistic sort. Virtual features and dynamic polymorphism are finished by using inheritance. Runtime binding and dynamic binding can be used interchangeably at run-time to substitute polymorphic objects. Its main advantage is to simplify the definition of the client and allow it to substitute at run-time [9].

Object-oriented databases have been achieved by developing object-oriented programming concepts to extend existing relational databases. OODB is primarily structured in inheritance by employing class hierarchies. Inheritance or class relations i.e. "is-a" and "has-a" have been used to represent the class hierarchy. The object-oriented programming concept has been achieved by using polymorphism to achieve better classification. Polymorphism is designed by classifying OODB to enable the use of simple SQL queries[10]. The efficiency of the proposed approach is shown with the help of experimental results. OODBs are designed with very little implementation overhead and a lot of memory space as compared to RDBs.

The rest of the paper is organized by including various sections as follows; A brief review is shown in section 2 on how to incorporate OOP concepts into data mining and database operations. This section deals with OODB mining. A brief description of OODB is presented with the help of Section 3. The proposed approach for effective classification is presented in detail in Section 4 along with the design of the OODB. The findings are summarized in Section 6 by summarizing the results of the experiments with the help of Section 5.

2. Literature Review

Object-oriented concepts and object-oriented mining are included in the database. Successful research related to data mining tasks has inspired research in data mining. Below is a brief review of some of the research related to our work.

Presented an efficient way of describing the knowledge from object-oriented data mining by creating a framework ‘Escher’ for object-oriented data mining by Sun, Jing [11].

The concepts of schema and multi-object development capable of retrieving the instances of each group at a later time were proposed by Sahoo, and Santosh Kumar [12].

Osman and Abdullahi Sidow[13] proposed an approach for demonstrating technology performance with the Target Mailing App by integrating complex object properties using complex data classification techniques.

Object structure relationships and object hierarchies were integrated into the classification process. It was employed using a method called CO4.5. Advantages such as multi-target class prediction and direct complex object discovery were achieved with the help of CO4.5.

An object-oriented data mining query language was developed by de la Vega, Alfonso [14], in which various types of knowledge were mined from OODB. An object cube model was developed by Leprince, Julien [15] presented. It has been of great help in online analytical processing and class-based normalization and data mining. The integration of trends, the association rule mining algorithm, and the integration of apriori were described by Kurnia, Yusuf [16] presented with the F2 OODBMS. The vast object-oriented database was used by Merzah, Bayadaa M [17] to present a set of data mining opportunities. With the help of this, the users got knowledge of a huge object-oriented database. An object-oriented framework was proposed by Satheesh, Kumar[18] Which helps to integrate heterogeneous database schemas for data mining.

3. Object-Oriented Databases (OODB)

A significant contributor is OODB's [19] ability to represent real-world concepts in the form of a data model. OODB manages databases efficiently by providing efficient and presentable methods.

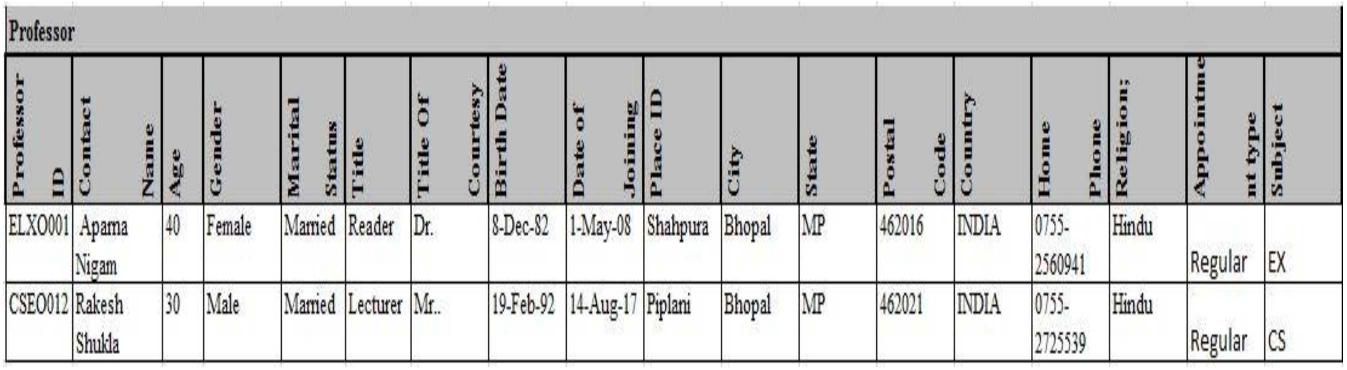
OODB has been adapted to object-oriented applications to support complex data relationships and a variety of structures, including trees and composite objects. OODB systems are great helpers for handling complex databases efficiently. It allows users to define a database to create, adjust, and drop tables and set up constraints. OODB is a group of inter-relationships between objects and objects, wherein items that are comparable in conduct and homes are organized into training. Each class is a box of shared strategies and a hard and fast of common attributes, in which instance variables and attributes of a class can be defined. The behavior of items related to method classes may be subject to the problems described in it. A class/subclass hierarchy is used to symbolize complex objects [20].

Composite object modeling and inheritance mechanisms are incorporated into the OODB model by employing its features. Class features can be divided into complex and simple methods to deal with the increased complexity. Classification and generalization are two important abstraction principles designed to structure the object-oriented approach [21]. Objects with similar properties, structure, and behavior are divided into classes, in which classification is defined as an abstract principle. With the help of generalization, all the common properties shared by multiple classes are used to create a class hierarchy by organizing them into a super class.

4. Object Oriented Concepts in Database Design

ODBMS [22] has been designed, defining three approaches. It is extended by the Object-Oriented Programming Language (OOPL) using a relational DBMS started from scratch in the third. In the first approach, OOPL with transaction support allows multiple concurrent accesses to be developed by including storage to develop an ODBMS. Creating an extended relational approach by including object-oriented features such as complex objects, polymorphism, methods and encapsulation, inheritance, and classes in which an ODBMS can use an existing relational DBMS [23]. The third approach is to create an ODBMS from the ground up using database technology such as UniSQL and OpenODB. The second approach is employed in this paper to expand relational databases. The proposed approach incorporates the concepts of inheritance and polymorphism into an OODB design to perform classification. A collection of tables is called a database. In a database, there are many tables available with common fields. In our approach, the fields are used together to form a normalized table by grouping such common sets. One of the well-known OOP concepts involves representing classes in hierarchies by which to achieve a newly created table inheritance hierarchy. An important object-oriented feature, dynamic polymorphism, is derived from having the same name and structure of methods in different classes. Which are operated differently depending on the calling object. Polymorphism is the use of object-oriented concepts to achieve classification effectively and simply. Using these object-oriented concepts allows for the design of OODB to answer even complex questions more efficiently. This enables classification to be achieved effectively.

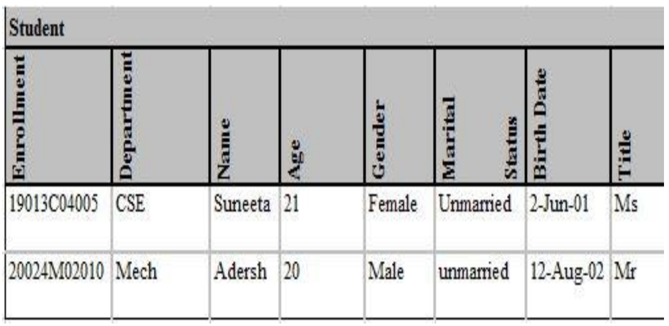
In this example University database is used, in which 't' represents a set of all tables that have some fields in common. A normalized table is constructed by adding all those common fields from the table set 't'. The efficiency of the proposed approach is illustrated by considering a traditional University system. University systems may have many tables available in the database, but this approach focuses on three tables. Professor, Student, and Adminstaff.These tables are represented in fig. 1 and alike are represented as classes by best illustrating the OOP concepts. The class structure is presented in Fig 2. Each table has a set of table-specific fields and a set of common fields (highlighted), shown in the class structure above.



Professor

Professor ID	Contact Name	Age	Gender	Marital Status	Title	Title Of Courtesy	Birth Date	Date of Joining	Place ID	City	State	Postal Code	Country	Home Phone	Religion;	Appointment type	Subject
ELXO001	Aparna Nigam	40	Female	Married	Reader	Dr.	8-Dec-82	1-May-08	Shahpura	Bhopal	MP	462016	INDIA	0755-2560941	Hindu	Regular	EX
CSEO012	Rakesh Shukla	30	Male	Married	Lecturer	Mr..	19-Feb-92	14-Aug-17	Piplani	Bhopal	MP	462021	INDIA	0755-2725539	Hindu	Regular	CS

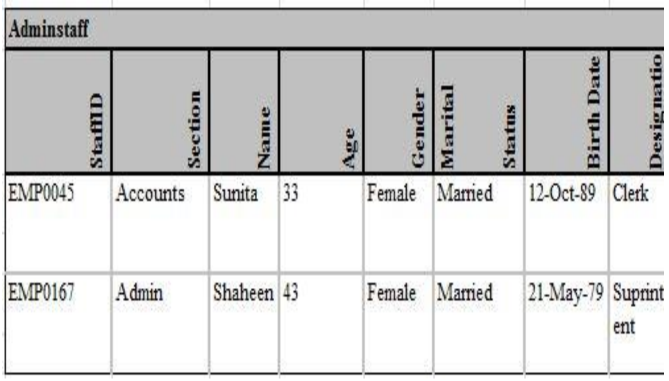
(a)



Student

Enrollment	Department	Name	Age	Gender	Marital Status	Birth Date	Title
19013C04005	CSE	Suneeta	21	Female	Unmarried	2-Jun-01	Ms
20024M02010	Mech	Adersh	20	Male	unmarried	12-Aug-02	Mr

(b)



Adminstaff

StaffID	Section	Name	Age	Gender	Marital Status	Birth Date	Designatio
EMP0045	Accounts	Sunita	33	Female	Married	12-Oct-89	Clerk
EMP0167	Admin	Shaheen	43	Female	Married	21-May-79	Suprintent

(c)

Fig. 1. Sample tables of professor, student and adminstaff

Class Professor	**Class Student**	**Class Adminstaff**
ProfessorID;	Enrollment;	StaffID;
DateofJoining;	semester;	Designation;
Place ID; City; State; Postal Code; Country; HomePhone; Religion; Department;	Name; Age; Gender; MaritalStatus; BirthDate; Religion; Title;	Title Name; Age; Gender; MaritalStatus; BirthDate; Religion;
Subject;	Department;	Section;
ContactName; Age; Gender; Marital Status; Title; Birth Date;	PlaceId; City; State; PostalCode; Country; Phone;	PlaceId; City; State; PostalCode; Country; Phone;
Appointment type;	Subject;	

Common

Fig. 2: Traditional Class Structure of Professor, Student and Adminstaff Table

Common fields such as gender, age, and name are included in the employee table. Title, fare date, etc. are included in table-specific fields. Common fields in most tables are frequent, increasing space complexity and redundancy. The tables need to be searched separately if a query is fired to retrieve a set of records. That effects data classification across the organization to meets a particular rule. OODBs achieve better classification by incorporating the legacy concept of OOP into the design.

5. Design of the OODB

To remove the problem of redundancy in the proposed approach, OODB is designed using the legacy concept of OOP. The table set 't' means all available common or common fields are searched. A single table is used to store all these common or common fields. All related tables can use it. Normalized tables have much in common with the base class of the OOP paradigm [24]. A new table named "person" has been created in the proposed approach. This table contains all those common fields. Other tables get information from the Person table without defining it. The "has-a" relationship is depicted using two important mechanisms: structure and generalization. A "has-a" relationship is represented by a structure. Both these relationships can be portrayed very well. All common fields are available in the normalized table "Person". Table "Professor, Student, and Adminstaff" has inherited table "Person." This is referred to as the "is-a" relation. a person who can be a professor, a student, or a member of the administrative staff An object reference to the table "Location" in the table "Person" is used to give an example of a creative relationship. Table Person has a "Has-A" relationship with Table Place. A person has a place, and a place has a postal code. The proposed OODB design is displayed in Figure 3 to represent the inheritance class hierarchy.

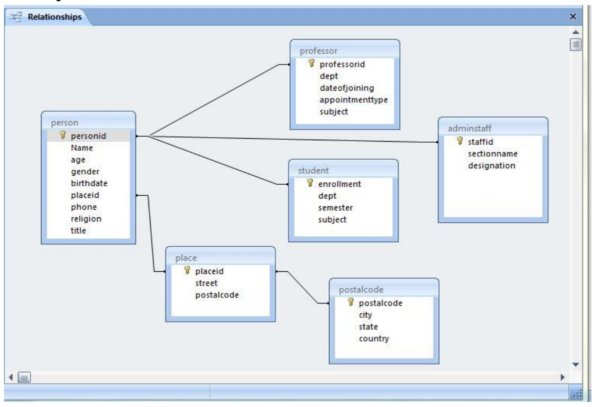
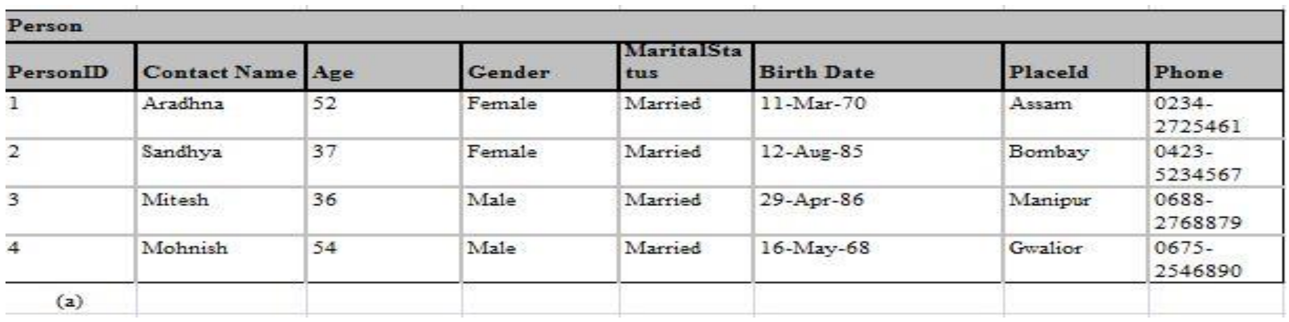


Fig. 3. Inheritance hierarchy of classes in the proposed OODB design

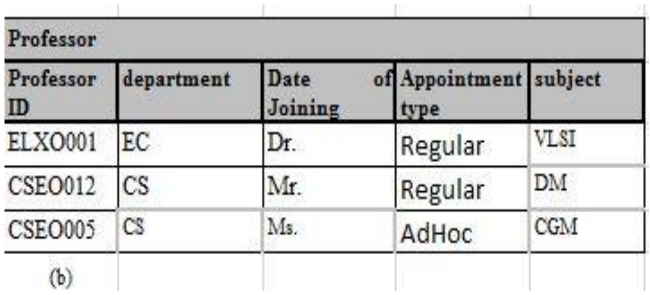
In the pictured design, the arrows (→) are used to represent the "has-a" relationship and the small triangle ( ) represents the "has-a" relationship. The fields are treated as attributes of the base class 'Person' and the normalized table 'Person' as the base class 'Person'. All common properties are present in the base class 'Person'. Special properties are inherited by other classes i.e. Professor, Student and Adminstaff. Inheritance allows for the definition of special methods in subclasses and generalized methods in the base class. E.g. by defining a method getContactNumebrs() in base class 'Person' to get contact numbers of all people associated with the organization by its subclasses can be shared.

One can use the generalized class 'Person' to represent the composition relationship to the other two classes 'Postcode' and 'Place'. The class 'Person' instance variable accesses the object reference of the classes 'PostalCode' and 'Place'. The database can be expanded by adding new tables using the inheritance concept. Common fields can be obtained from the normalized table by inheriting. The tables in the proposed OODB design are shown in Fig 4.



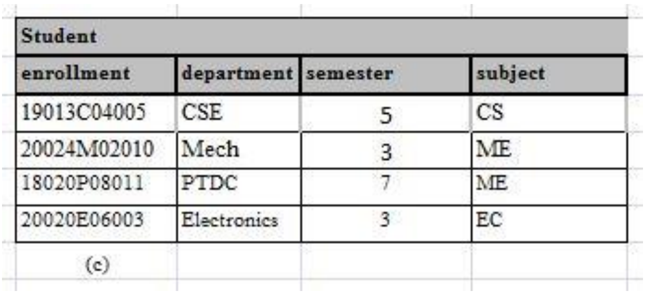
Person							
PersonID	Contact Name	Age	Gender	MaritalStatus	Birth Date	PlaceId	Phone
1	Aradhna	52	Female	Married	11-Mar-70	Assam	0234-2725461
2	Sandhya	37	Female	Married	12-Aug-85	Bombay	0423-5234567
3	Mitesh	36	Male	Married	29-Apr-86	Manipur	0688-2768879
4	Mohnish	54	Male	Married	16-May-68	Gwalior	0675-2546890

(a)



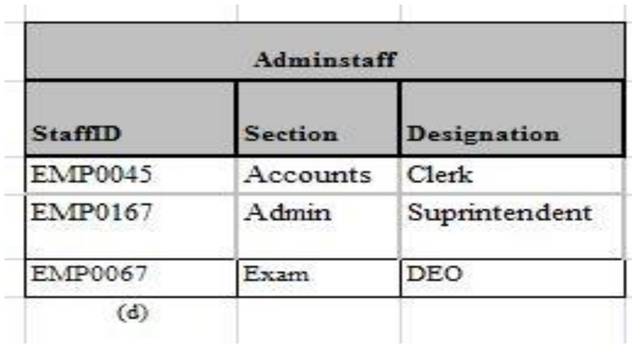
Professor				
Professor ID	department	Date of Joining	Appointment type	subject
ELXO001	EC	Dr.	Regular	VLSI
CSEO012	CS	Mr.	Regular	DM
CSEO005	CS	Ms.	AdHoc	CGM

(b)



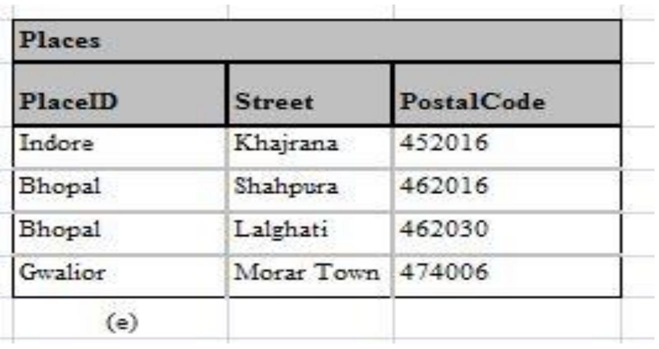
Student			
enrollment	department	semester	subject
19013C04005	CSE	5	CS
20024M02010	Mech	3	ME
18020P08011	PTDC	7	ME
20020E06003	Electronics	3	EC

(c)



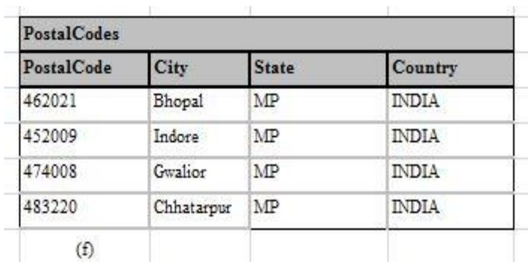
Adminstaff		
StaffID	Section	Designation
EMP0045	Accounts	Clerk
EMP0167	Admin	Suprintendent
EMP0067	Exam	DEO

(d)



Places		
PlaceID	Street	PostalCode
Indore	Khajrana	452016
Bhopal	Shahpura	462016
Bhopal	Lalghati	462030
Gwalior	Morar Town	474006

(e)



PostalCodes			
PostalCode	City	State	Country
462021	Bhopal	MP	INDIA
452009	Indore	MP	INDIA
474008	Gwalior	MP	INDIA
483220	Chhatarpur	MP	INDIA

(f)

Fig. 4. Tables in the proposed OODB design-Professor, Student, Adminstaff, places and postalcode.

5.1 Classification in the designed OODB

Classifications in OODB are explained in this section with sub-sections designed to demonstrate simple SQL queries and the OOP concept of polymorphism. Dynamic polymorphism or overdue binding facilitates definition techniques with identical names in different instructions[25]. The technique referred to as runtime is determined based on the calling object. The concept of polymorphism using the classify() method for all tables produces sample code.

5.1.1 Classification in the table 'Person'

The SQL query in the below code classifies the person based on their city, state and country.

```
Class Person
{	classify(String Option)
{if(option.equals("city")||option.equals("country")||option.equals("State"))
query = new String("SELECT Person.Personid,name,age, "+option+" FROM person,PostalCode, Place "+" WHERE person.PlaceId=[Place].placeid
And"+"[PostalCode].postalcode= [Place].postalcode order by "+option);
          }};
```

5.1.2 Classification in the table 'Professor'

The below SQL query performs city wise, state wise and country wise classification of Professor.

```
Class Professor extends Person
{	classify(String Option)
          {if(option.equals("city")||option.equals("country")||
option.equals("state"))
query = new String ("SELECT personId, name,age, "+ option +        " FROM person, Professor, PostalCode, Place "+        "WHERE (person.personid=Professor.Professorid And " +        "person.PlaceId=[Place].placeid And "+ "[PostalCode].postalcode=[Place].postalcode)orderby        "+ option);
          }};
```

5.1.3 Classification in the table 'Student'

The below SQL query classifies the Student based on city, state and country.

```
Class Student extends Person
{classify(String Option)
          {if(option.equals("city")||option.equals("Country")||option.equals("state"))
query = new String( "SELECT Student.Enrollment, name,age,        "+option+ " FROM person, Student, PostalCode, Place "+
"WHERE person.personid=Student.Enrollment And "+ "person.PlaceId= [Place].placeid And "+ "[PostalCode].
postalcode=[Place].postalcode order by "+option);
          }};
```

5.1.4 Classification in the table 'Adminstaff'

The Adminstaff are classified based on city, state and country using the following SQL query.

```
Class Adminstaff extends Person
```

```
   {  classify (String Option)
   {if(option.equals("city")||option.equals("country")||option.equals ("state"))

   query = new String( "SELECT Adminstaff.Staffid, name,age,
    "+option+ " FROM person, Adminstaff, PostalCode, Place "+ "WHERE person.personid=Adminstaff.Staffid And "+
    "person.PlaceId=[Place].placeid And "+ "[PostalCode].
     postalcode=[Place].postalcode order by "+option);
       }};
```

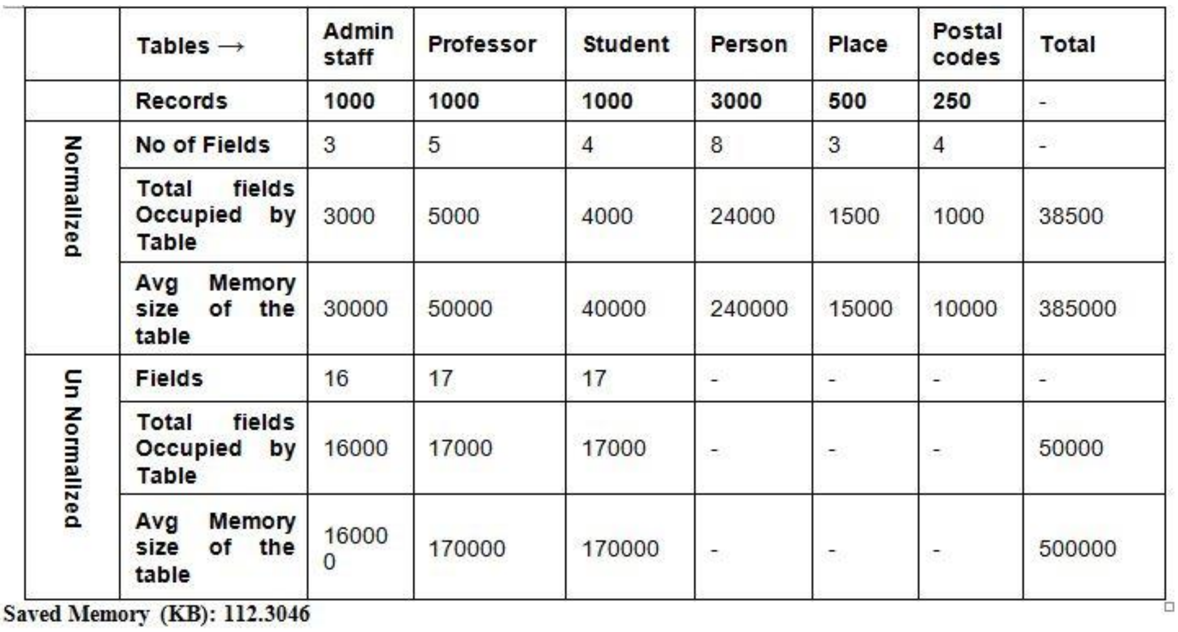
The main objective of the proposed approach is to complete the classification process using a simple SQL query. Existing classifications have been used for OODB employing complex techniques such as nearest neighbor methods, neural networks, and decision trees. The classify() method is used for individual entities such as Adminstaff, Student and Professor. The classify() method can be reached with the help of the following simple code.

```
   switch(option)
    {   case 1:{
        person = new Professor();
        person. classify(field); break;
              }
        case 2:{
        person = new Adminstaff();
        person. classify(field); break;
              }
        case 3:{
        person = new Student();
        person. classify(field); break;
              }
        case 4:{
        person = new Person();
        person. classify(field); break;
              }
         }
```

By defining the method 'classify' in all the classes the method to be called based on the object is fixed. The above practice of OOP is presented in the form of dynamic polymorphism. Writing and managing code becomes very easy by integrating the concept of polymorphism. In designed OODB, simple SQL queries are used to perform the classification task effectively. By incorporating OOP concepts in the proposed approach, the maximum benefits of OOP can be exploited. The task of classification is also done more effectively for designing OODBs.

6. Result and Outcome

With the help of this section, the experimental results of the proposed approach are depicted. In general, an organization can have any number of tables. In this database, we have considered only three tables to show, covering object-oriented programming concepts. There are too many records in each table. Object-oriented programming concepts have made a successful attempt to substantially reduce implementation overhead. Along with this, the memory space has been reduced to a great extent. With the help of this proposed approach, the memory space saved is calculated by comparative analysis of the space used after normalization of tables and before normalization. For this purpose, different numbers of records are used, such as 5000 (561.5234 KB of memory), 4000 (with 449.2187 KB), 3000 (with 336.91406 KB of memory), 2000 (with 224.6093 KB of memory) and 1000 (with 112.3046 KB of memory saved). Results of comparative analysis are displayed in tables of fig. 5.



	Tables →	Admin staff	Professor	Student	Person	Place	Postal codes	Total
	Records	1000	1000	1000	3000	500	250	-
Normalized	No of Fields	3	5	4	8	3	4	-
	Total fields Occupied by Table	3000	5000	4000	24000	1500	1000	38500
	Avg Memory size of the table	30000	50000	40000	240000	15000	10000	385000
Un Normalized	Fields	16	17	17	-	-	-	-
	Total fields Occupied by Table	16000	17000	17000	-	-	-	50000
	Avg Memory size of the table	160000	170000	170000	-	-	-	500000

Saved Memory (KB): 112.3046

	Tables →	Admin staff	Professor	Student	Person	Place	Postal codes	Total
	Records	**2000**	**2000**	**2000**	**6000**	**1000**	**500**	-
Normalized	No of Fields	3	5	4	8	3	4	-
	Total fields Occupied by Table	6000	10000	8000	48000	3000	2000	77000
	Avg Memory size of the table	60000	100000	80000	480000	30000	20000	770000
Un Normalized	Fields	16	17	17	-	-	-	-
	Total fields Occupied by Table	32000	34000	34000	-	-	-	100000
	Avg Memory size of the table	320000	340000	340000	-	-	-	1000000

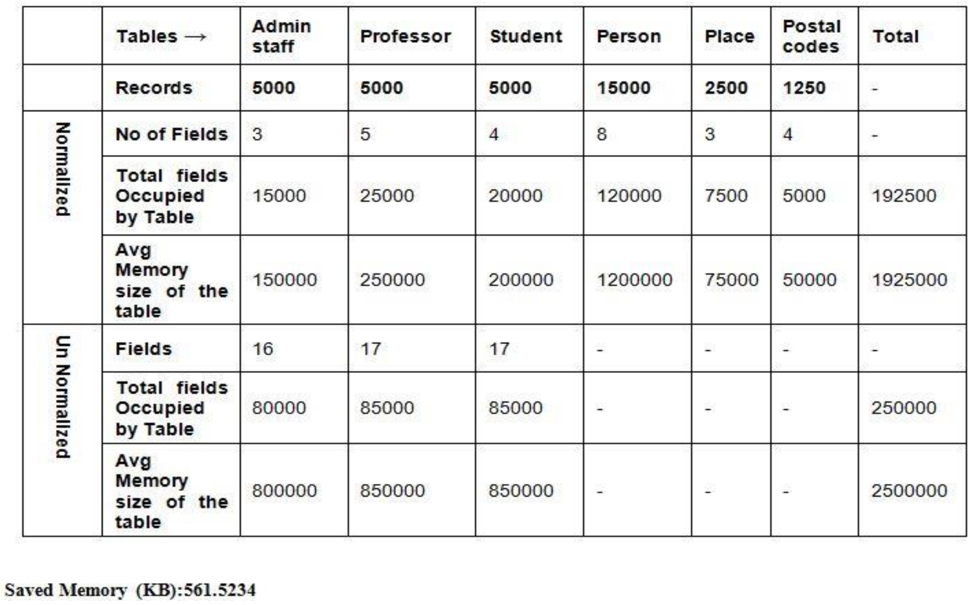
Saved Memory (KB): 224.6093

	Tables →	Admin staff	Professor	Student	Person	Place	Postal codes	Total
	Records	**3000**	**3000**	**3000**	**9000**	**1500**	**750**	-
Normalized	No of Fields	3	5	4	8	3	4	-
	Total fields Occupied by Table	9000	15000	12000	72000	4500	3000	115500
	Avg Memory size of the table	90000	150000	120000	720000	45000	30000	1155000
Un Normalized	Fields	16	17	17	-	-	-	-
	Total fields Occupied by Table	48000	51000	51000	-	-	-	150000
	Avg Memory size of the table	480000	510000	510000	-	-	-	1500000

Saved Memory (KB): 336.91406

	Tables →	Admin staff	Professor	Student	Person	Place	Postal codes	Total
	Records	**4000**	**4000**	**4000**	**12000**	**2000**	**1000**	-
Normalized	No of Fields	3	5	4	8	3	4	-
	Total fields Occupied by Table	12000	20000	16000	96000	6000	4000	154000
	Avg Memory size of the table	120000	200000	160000	960000	60000	40000	1540000
Un Normalized	Fields	16	17	17	-	-	-	-
	Total fields Occupied by Table	64000	68000	68000	-	-	-	200000
	Avg Memory size of the table	640000	680000	680000	-	-	-	2000000

Saved Memory (KB): 449.2187



	Tables →	Admin staff	Professor	Student	Person	Place	Postal codes	Total
	Records	5000	5000	5000	15000	2500	1250	-
Normalized	No of Fields	3	5	4	8	3	4	-
	Total fields Occupied by Table	15000	25000	20000	120000	7500	5000	192500
	Avg Memory size of the table	150000	250000	200000	1200000	75000	50000	1925000
Un Normalized	Fields	16	17	17	-	-	-	-
	Total fields Occupied by Table	80000	85000	85000	-	-	-	250000
	Avg Memory size of the table	800000	850000	850000	-	-	-	2500000

Saved Memory (KB):561.5234

Fig. 5. The results of comparative analysis in tabular form with varying number of instances.

A graphical representation of the results is presented with the help of Fig 6. The memory space saved has also been found to increase as the number of records in each table increases.

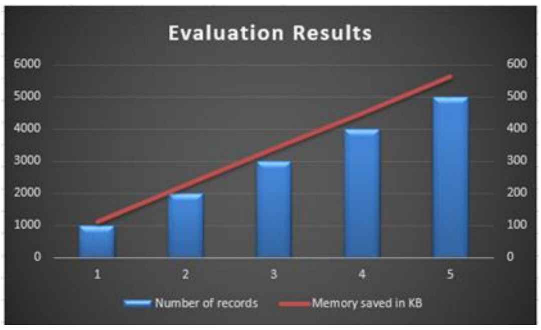


Fig. 6. Graph demonstrating the above evaluation results

In addition, a lot of memory space is saved by including entity-specific methods in common subclasses and methods in generalized classes. In the proposed approach, normalizing classes avoid redefining the methods for the respective classes. If there are 'm' classes, placing the common methods in the base class can save memory space of

$$(m-1)\sum_{j=1}^{n} memory\ size\ of\ \ j^{th}\ \ method$$

Where ‘n’ is the number of common methods in the super class.

7. Conclusion

Research in this field has increased rapidly in the last few decades due to the increasing interest in data mining. Management of complex information and user data types can be easily achieved by incorporating object-orientation concepts into relational database management systems. Through this paper, one such approach is presented in which the design and classification of object-oriented databases have been effectively used in a very simple way. Object-oriented programming concepts like polymorphism and inheritance are covered in the presented approach. This design for simple SQL queries is introduced by an efficient classification function from OODB. The effectiveness of the presented approach is easily demonstrated in the experimental results. Through the proposed technique implementation overhead has reduced by using OODB's design. It has also helped in reducing the amount of memory space required as well.

References

[1] Dogan and D. Birant, "Machine learning and data mining in manufacturing," Expert Systems with Applications, vol. 166, p. 114060, 2021.

[2] H. Das, B. Naik, and H. Behera, "Classification of diabetes mellitus disease (DMD): a data mining (DM) approach," in Progress in computing, analytics and networking: Springer, 2018, pp. 539-549.

[3] J. Yang et al., "Brief introduction of medical database and data mining technology in big data era," Journal of Evidence-Based Medicine, vol. 13, no. 1, pp. 57-69, 2020.

[4] H. M. Faisal, M. A. Tariq, A. Alghamdi, and N. Alowain, "A query matching approach for object relational databases over semantic cache," in Application of Decision Science in Business and Management: IntechOpen, 2019.

[5] C. Gutiérrez and J. F. Sequeda, "Knowledge graphs," Communications of the ACM, vol. 64, no. 3, pp. 96-104, 2021.

[6] S. Palanisamy and P. SuvithaVani, "A survey on RDBMS and NoSQL Databases MySQL vs MongoDB," in 2020 International Conference on Computer Communication and Informatics (ICCCI), 2020, pp. 1-7: IEEE.

[7] Y. M. Win, "COMPARATIVE ANALYSIS OF RELATIONAL AND OBJECT ORIENTED APPROACHES FOR GIS DATABASE."

[8] S. Kaur and P. Singh, "How does object-oriented code refactoring influence software quality? Research landscape and challenges," Journal of Systems and Software, vol. 157, p. 110394, 2019.

[9] A. Dingle, Object-oriented Design Choices. Chapman and Hall/CRC, 2021.

[10] K. Domdouzis, P. Lake, and P. Crowther, "Object and Object Relational Databases," in Concise Guide to Databases: Springer, 2021, pp. 165-187.

[11] J. Sun and Y. Sun, "Analysis of Data Mining Based on Object Oriented Analysis Method," in Journal of Physics: Conference Series, 2018, vol. 1069, no. 1, p. 012041: IOP Publishing.

[12] S. K. Sahoo, "Artificial Intelligence Based Multi-object Inspection System," in Computational Intelligence in Data Mining: Springer, 2020, pp. 205-211.

[13] A. S. Osman, "Data mining techniques," 2019.

[14] A. de la Vega, D. García-Saiz, M. Zorrilla, and P. Sánchez, "Lavoisier: A DSL for increasing the level of abstraction of data selection and formatting in data mining," Journal of Computer Languages, vol. 60, p. 100987, 2020.

[15] J. Leprince, C. Miller, and W. Zeiler, "Data mining cubes for buildings, a generic framework for multidimensional analytics of building performance data," Energy and Buildings, vol. 248, p. 111195, 2021.

[16] Y. Kurnia, Y. Isharianto, Y. C. Giap, and A. Hermawan, "Study of application of data mining market basket analysis for knowing sales pattern (association of items) at the O! Fish restaurant using apriori algorithm," in Journal of Physics: Conference Series, 2019, vol. 1175, no. 1, p. 012047: IOP Publishing.

[17] Krishna, P. R. ., and P. . Rajarajeswari. “EapGAFS: Microarray Dataset for Ensemble Classification for Diseases Prediction”. International Journal on Recent and Innovation Trends in Computing and Communication, vol. 10, no. 8, Aug. 2022, pp. 01-15, doi:10.17762/ijritcc.v10i8.5664.

[18] B. M. Merzah, "Software quality prediction using data mining techniques," in 2019 International Conference on Information and Communications Technology (ICOIACT), 2019, pp. 394-397: IEEE.

[19] A. Satheesh and A. Kumar, "An Effective Classification Rule Mining Algorithm for Object Oriented Databases."

[20] N. Patil, P. Kiran, N. Kiran, and N. P. KM, "A survey on graph database management techniques for huge unstructured data," International Journal of Electrical and Computer Engineering, vol. 8, no. 2, p. 1140, 2018.

[21] Ahmed Cherif Megri, Sameer Hamoush, Ismail Zayd Megri, Yao Yu. (2021). Advanced Manufacturing Online STEM Education Pipeline for Early-College and High School Students. Journal of Online Engineering Education, 12(2), 01–06. Retrieved from http://onlineengineeringeducation.com/index.php/joee/article/view/47

[22] I. TROFIMOV, L. TROFIMOV, and S. PODKOVALNIKOV, "DATA STORAGE AND REPRESENTATION FOR STUDYING PROBLEMS OF PROSPECTIVE ELECTRIC POWER SYSTEMS," in XIII Balkan Conference on Operational Research Proceedings, 2018, p. 90: FON.

[23] A. A. Mohammed, "Design and implementation of a prototype data mining agent system," Altınbaş Üniversitesi, 2019.

[24] D. Corrales-Garay, M. Ortiz-de-Urbina-Criado, and E.-M. Mora-Valentín, "Understanding open data business models from innovation and knowledge management perspectives," Business Process Management Journal, 2022.

[25] K. Chaitanya, R. Venkatesh, and T. Bikku, "An Efficient Model for Medical Data Classification using Gene Features," International Journal of Advanced Computer Science and Applications, vol. 10, no. 11, 2019.

[26] Agarwal, D. A. . (2022). Advancing Privacy and Security of Internet of Things to Find Integrated Solutions. International Journal on Future Revolution in Computer Science &Amp; Communication Engineering, 8(2), 05–08. https://doi.org/10.17762/ijfrcsce.v8i2.2067

[27] E. Lotfi and B. Mohammed, "Teaching object oriented programming concepts through a mobile serious game," in Proceedings of the 3rd International Conference on Smart City Applications, 2018, pp. 1-6.

[28] P. Kanwar and M. Rathore, "Cognitive Study of Data Mining Techniques in Educational Data Mining for Higher Education," in Information and Communication Technology for Competitive Strategies (ICTCS 2020): Springer, 2021, pp. 247-258.

[29] Joy, P., Thanka, R., & Edwin, B. (2022). Smart Self-Pollination for Future Agricultural-A Computational Structure for Micro Air Vehicles with Man-Made and Artificial Intelligence. International Journal of Intelligent Systems and Applications in Engineering, 10(2), 170–174. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/1743

[30] Kitsana Waiyamai, Chidchanok Songsiri, Thanawin Rakthanmanon, “Object-oriented database mining: Use of object oriented concepts for improving data classification technique” , International Conference on Computational Sc, Springer, Berlin, Heidelberg 2004

Author contributions

Ajita Satheesh:, Methodology, Software, Field study, Writing-Original draft preparation, Visualization, Investigation, **Aarti Kumar:** Conceptualization,Validation., Writing - Reviewing and Editing.