

AN ENHANCEMENT OF DATA SECURITY FOR
ORACLE DATABASE

BY

HAO PANG

Bachelor of Science

Nanjing University of Science and Technology

Nanjing, China

1997

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
In partial fulfillment of
The requirements for
The Degree of
MASTER OF SCIENCE
August 2002

AN ENHANCEMENT OF DATA SECURITY FOR
ORACLE DATABASE

Thesis Approval:

H. Lu

Thesis Advisor

D. E. H...

Park...

Timothy A. Peltone
Dean of the Graduate College

ACKNOWLEDGMENTS

I wish to express my sincere appreciation to my major advisor, Dr. Huizhu Lu for her intelligent supervision, constructive guidance, inspiration and friendship. My sincere appreciation extends to my other committee members Dr. G. E. Hedrick and Dr. Nohpill Park, whose guidance, assistance, encouragement, and friendship are also invaluable.

More over, I wish to express my sincere gratitude to those who provided suggestions and assistance for this study: Dr. John P. Chandler, Ms. Danlei Li, Ms. Chun Li, and Mr. Weibo Zhang.

I would like to give my special thanks to my parents Mr. Bingqian Pang, Mrs. Fengying Xie, and my sister Ms. Ling Pang, for their support and encouragement.

Finally, I would like to thank the Department of Computer Science for supporting during these years of study.

TABLE OF CONTENTS

Chapter	Page
I. INTRODUCTION.....	1
1.1. Background.....	1
1.2. The Problem and the Purpose.....	2
1.3. Objectives.....	3
1.4. Definition of Terms.....	4
1.5. Outline of the Thesis.....	4
II. REVIEW OF THE LITERATURE.....	6
2.1. The Cryptography and the Encryption.....	6
2.2. Database Security.....	10
2.3. Oracle System and Develop Environment.....	12
2.4. PL/SQL Language.....	14
2.5. Using Java Class Methods in Oracle.....	15
III. DESIGN AND IMPLEMENTATION.....	17
3.1. Methodology.....	17
3.1.1. Field Encryption.....	18
3.1.2. Row Level Control.....	27
3.2. The Program.....	29
3.2.1. The Main Program.....	30
3.2.2. Elements in a Database.....	34
3.2.3. Subprograms in a Database.....	35
3.2.4. Test Program.....	36

Chapter	Page
IV. RESULTS AND FINDINGS.....	37
4.1. Results.....	37
4.2. Program Test.....	38
4.2.1. Field Encryption Test.....	40
4.2.2. Row Level Control Test.....	47
4.3. Performance Test.....	53
4.3.1. Space Complexity.....	54
4.3.2. Time Complexity.....	57
4.4. Security Problems.....	58
V. CONCLUSIONS.....	59
5.1. Conclusions.....	59
5.2. Future Work.....	60
REFERENCES.....	61
APPENDIXES.....	65
APPENDIX A – A TABLE OF MAJOR SUBPROGRAMS USED IN THE PROGRAM.....	66
APPENDIX B – ENCRYPTION METHOD TEST SUITES.....	68
APPENDIX C – A USER’S GUIDE.....	71

LIST OF TABLES

Table	Page
1. Supported Data Types.....	27
2. Major Subprograms in a Database.....	36
3. Test Users and Privileges.....	38
4. The Structure of the Table for Testing.....	39
5. The Content of the Table “EMP”.....	39
6. An Empty Table for Row Level Control.....	40
7. The Test List for Field Encryption.....	40
8. The Column “JOB” has been Encrypted.....	41
9. The Table “EMP” after Decoding Operation.....	42
10. The Attribute “JOB” has been Encrypted by MD5.....	43
11. The Final Result after the Field Encryption Tests.....	47
12. A List of the Tests for Row Level Control.....	47
13. Privileges for Each User.....	48
14. A List of New Row Control Levels.....	49
15. After “INSERT” Operation in Table “TEST_RC1”.....	50
16. The Result of “SELECT” Statement in RC.....	51
17. “UPDATE” Test of RC in Table “TEST_RC1”.....	52
18. The Running Time of Each Encryption Method.....	57

LIST OF FIGURES

Figure		Page
1.	A Block Diagram of the Project.....	5
2.	A Diagram of Structures with Oracle DBMS.....	18
3.	New Layer between User and DBMS.....	19
4.	User's Request Process.....	20
5.	Decode/Encode the Attribute Values.....	23
6.	Different Positions of Encrypted Attribute in Query.....	24
7.	Sample: Process the "SELECT" Statement.....	25
8.	Visual Table for Each User.....	29
9.	The Structure of the Program.....	31
10.	Interface of the Main Program.....	32
11.	Hierarchical Tree.....	33
12.	Space Cost of the Encryption Methods.....	56

NOMENCLATURE

DS	Data Security
FE	Field Encryption
RC, RLC	Row Level Control
RCL	Row Control Level, Access Level
DB	Database
App	Application
JVM	Java Virtual Machine
Cl	Length of the Cipher Text
Pl	Length of the Plain Text
N	Factor Used in RSA, Big Integer
MD2	Message-digest Algorithms 2
MD4	Message-digest Algorithms 4
MD5	Message-digest Algorithms 5
DES	The Data Encryption Standard
RSA	A Public-key Cryptosystem Algorithm
Bn	Number of the Blocks in RSA Algorithm
Bs	Size of Each Block in RSA Algorithm

CHAPTER 1

INTRODUCTION

1.1. BACKGROUND

Database security becomes more and more important with the rapid development of Internet. E-commerce and other Internet applications impel more requirements to database systems [13]. Over the past two years, there has been a huge increase in the amount of business conducted over the Internet. People now book plane tickets, make hotel reservations, rent a car, transfer money from one account to another, and buy clothes, books online. Behind all these front-end applications, a database system is the kernel part and changes the way of information that is managed and accessed to meet the demands of the Internet age. Databases often hold the most valuable electronic assets within a company, including intellectual property, transaction data, financial information, or customer data. Information, if compromised, could have a significant negative impact. In order to make our data safe, we need to continuously improve the data security methods [1].

Today's database management systems are much better than before since they can provide many security methods to the database administrators or users. The typical methods for database security are listed below [10]:

- Authentication
- Security Identification Numbers (Pin)
- Roles
- Securing Access to the Server or Database
- Permission System
- Ownership Chain

Though currently we have so many methods for data storing and transferring, none of the existing database system is perfectly secured and no one can say his or her database is unbreakable. The details of database security are described in Chapter3.

1.2. The Problem and the Purpose

Obviously, if the breaker or hacker gets the password of the database, for example, default password provided with built-in accounts, users or customers would lose all their securities. In fact, a large percentage of the security breaking incidents are made by insiders. Since right now there almost no way can protect the data in such case (Password Hacking) and most people still focus on the authority control in database and data transference, exploring some new ways to enhance the data security is the purpose of this project. This new method can protect our data and information even someone can break in the database.

In order to reach the goal of this project, the idea is we need to encrypt the data by some encryption methods before storing the data in the table. It may help us to avoid the data losing if the database has been hacked.

1.3. Objectives

The objective of this project is to build an application of Oracle relational database management system to provide enhanced security for data. It includes two major functions. The first function is to give users or database administrators a choice that they can encrypt a field of a table by a selected encryption approach among methods. The second function is to provide a grained access control or row level security control whereby a user can only access rows of data that pertain to him/her.

The main computer programs of this project contain two parts. The first part contains a management system for the two major functions. The second part is an application simulation program that used for testing the result and performance. Both programs are written with Java and PL/SQL, an enhanced SQL language from Oracle. The interface of this program is created in Oracle Forms Developer, Forms Builder6. Other parts like triggers, functions and procedures are built by PL/SQL too. Three typical algorithms, MD5, DES, and RSA, are used for encryption and are implemented by Java language and tested in Oracle JDeveloper.

In this project, Oracle Database 8i Standard Edition is used as the basic database management system. The operating system is Microsoft Windows 2000 Professional.

1.4. Definition of Terms

Field Encryption (FE): A method that can encrypt data of attributes of a database table

Row Level Control (RLC, RC): The rows in the table have been granted to different users

Data Security (DS): A security method to secure the data itself

1.5. Outline of the Thesis

The outline of this study is to add two additional functions to the Oracle database management system and create a test program. Totally, there are five parts in the project, Management System, Interface to the User, Interface to the Database, Java Classes of the Encryption Method, and Test Program as shown in Figure 1.

The first part is Management System, which is the kernel of the entire program. Management System manages both the field encryption and row level control functions. It can receive the requests from the user by User Interface and communicate with the database through the Database Interface. Test program can simulate an application environment and check the final result.

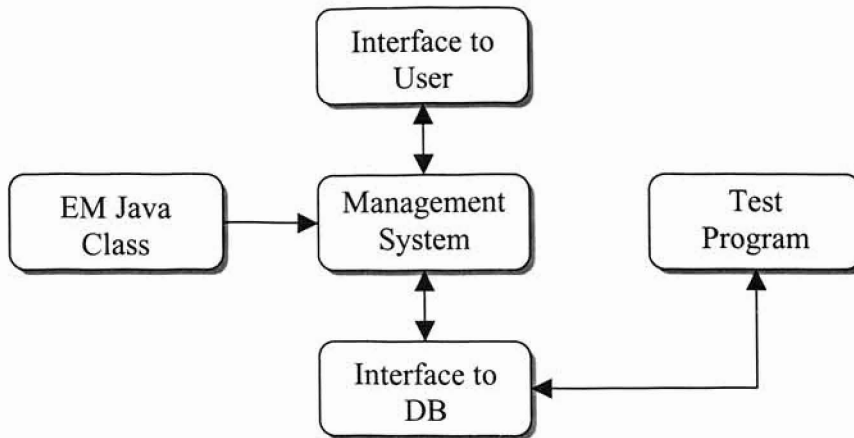


Figure1. A Block Diagram of the Project

The thesis is organized as follows.

Chapter I is the introduction to the thesis. In Chapter II, we present an overview of the subject and closely related subjects. Methodology, analysis, and the implementation of the project are given in Chapter III. We present the results and analysis of each testing for this project in Chapter IV. Finally, Chapter V contains conclusions and the future research.

CHAPTER 2

REVIEW OF THE LITERATURE

To reach the goal of this thesis, we should focus on the following areas:

1. The Cryptography and the Encryption
2. Database Security
3. Oracle Systems and Application Develop Environment
4. PL/SQL Language
5. Java Class Methods in PL/SQL

2.1. The Cryptography and the Encryption

Cryptography is an ancient mathematical science that was originally used for military communications, and designed to hide the contents of a message if it fall into the hands of the enemy. Recent developments in cryptography have added more uses, including mechanisms for authenticating users on a network, ensuring the integrity of transmitted information and preventing users from repudiating (i.e. rejecting ownership of) their transmitted messages.

Encryption is the name given to the process of applying an algorithm to a message, which secure the data and making it very difficult to get the original data. Input

to the algorithm typically involve additional secret data that called **key**, which prevents the message from being decoded even if the algorithm is publicly known [20].

There are two kinds of key systems available. Different algorithms apply to different key systems. The symmetric key system uses one key for both encryption and decryption. The public key/private key system uses one key for encryption and another key for decryption [3].

The encryption methods used in this project are MD5, DES and RSA.

1. MD5

MD2 [9], MD4 [19][17], and MD5 [18] are message-digest algorithms developed by Rivest. All three algorithms take a message of arbitrary length and produce a 128-bit message digest. MD2 was optimized for 8-bit machines, whereas MD4 and MD5 were aimed at 32-bit machines [22].

In MD2, the message is first padded so its length in bytes is divisible by 16. A 16-byte checksum is then appended to the message, and the hash value is computed on the resulting message [22]. In MD4, the message is padded to ensure that its length in bits plus 64 is divisible by 512. A 64-bit binary representation of the original length of the message is then concatenated to the message [4]. The message is processed in 512-bit blocks in the Damgård/Merkle [25] iterative structure, and each block is processed in three distinct rounds. MD5 was developed by Rivest in 1991. While it is a little bit slower than MD4, it is more secure. The algorithm consists of four distinct rounds, which has a slightly different design from that of MD4. The size of message-digest remains the same.

Den Boer and Bosselaers [5] have found pseudo-collisions for MD5. More recent work by Dobbertin has extended the techniques used so effectively in the analysis of MD4 to find collisions for the compression function of MD5 [8]. While stopping short of providing collisions for the hash function in its entirety this is clearly a significant step. [21].

2. DES

The Data Encryption Standard, or DES, was the first official U.S. government cipher intended for commercial use. It is the most widely used cryptosystem in the world. DES is an improvement of the algorithm Lucifer developed by IBM in the early 1970s. It is the best known and widely used symmetric algorithm in the world [23].

The DES has a 64-bit block size and uses a 56-bit key during execution (8 parity bits are stripped off from the full 64-bit key). The DES is a symmetric cryptosystem, specifically a 16-round Feistel cipher and was originally designed for implementation in hardware. When used for communication, both sender and receiver must know the same secret key, which can be used to encrypt and decrypt the message, or to generate and verify a message authentication code (MAC). The DES can also be used for single-user encryption, such as to store files on a hard disk in encrypted form. In a multi-user environment, secure key distribution may be difficult; public-key cryptography provides an ideal solution to this problem.

3. RSA

The RSA cryptosystem is a public-key cryptosystem that offers both encryptions and digital signatures (authentication) [24]. Ronald Rivest, Adi Shamir, and Leonard Adleman developed the RSA system in 1977 [20]; RSA stands for the first letter in each of its inventors' last names.

The RSA algorithm works as follows: take two large primes, p and q , and compute their product $n = pq$; n is called the modulus. Choose a number, e , less than n and relatively prime to $(p-1)(q-1)$, which means e and $(p-1)(q-1)$ have no common factors except 1. Find another number d such that $(ed - 1)$ is divisible by $(p-1)(q-1)$. The values e and d are called the public and private exponents, respectively. The public key is the pair (n, e) ; the private key is (n, d) . The factors p and q may be destroyed or kept with the private key [7].

It is currently difficult to obtain the private key d from the public key (n, e) . However if one could factor n into p and q , then one could obtain the private key d . Thus the security of the RSA system is based on the assumption that factoring is difficult. The discovery of an easy method of factoring would “break” RSA.

All these cryptographies are written as different classes by Java. For example, for DES, the name of the Java class is “DES.class”. Currently, there are 3 encryption methods can be used in this project, and the name of the classes are “DES.class”, “RSA.class”, and “MD5.class”. We choose these 3 typical cryptographs because they are the most wildly used and easy to understand.

2.2. Database Security

It is evident that data security principles must be applied to sensitive information. Government, research, corporations and other organizations keep large volumes of data that are not expected to be available to non-authorized users. And if someone can access them, data should be unreadable and absolutely incomprehensible. Right security policies largely decrease risks by reducing vulnerabilities and strengthening defenses and countermeasures [12]. A database server with important information is tempting for many people and appropriate measures have to be taken. They include the physical security of the server and the number of people accessing it [26].

Data accessibility is a major goal in database security. Many companies or organizations cannot work properly if databases are down. To put the data available implies to provide the security mechanisms to ensure authentication, authorization and auditing procedures. Authentication means that user identity must be truly verified, commonly through a password only known to the user. After this first step has been completed, the system must determine the resources that the particular user id can access to. This is the authorization phase and all the tasks involved are often referred as user security administration. Finally, to detect possible intruders and ensure data integrity, auditing utilities must be activated.

While on routing and Internet from server to receiver, data passes through different devices where, if security policies have been not applied or are defective, a third-party can get access to the packets. Obviously, this is a security threat that must be

taken into strong consideration. We must ensure that data can be only seen by the same individual we sent the data to, while avoiding data corruption by a third party. To achieve the former, encryption must be applied. Data integrity is usually ensured by means of certificates and public key encryption. Another important aspect is named non-repudiation. This is extremely important in e-business and it avoids that a sender can deny to have sent the information.

There is a need to protect databases from accidental data loss. A general backup and recovery strategy must be designed depending on various factors, such as database size, volume of changes, and resources available. Attention must be paid when choosing the backup type (incremental, full) and testing the whole set of procedures to recover the system in case of disaster, and in a timely manner. Backup utilities, usually integrated into the database package (e.g., On-Tape and On-Archive in Informix, RMAN in Oracle), guarantee that backup procedures can be carried out while maintaining database security and referential integrity. Parallelism is strongly suggested, mainly for critical applications, as it ensures minimum service disruption. The latter can be also improved by using high-speed backup devices.

A schema is a logical collection of database objects (tables, views, sequences, synonyms, indexes, clusters, procedures, functions, packages, and database links). By default, each database user creates and has access to all objects in the corresponding schema. With correct privileges, user can access other resource in different schemas.

Encryption is commonly associated to information moving from one side to another. On a high risk database environment, encryption must be also enabled to the lowest level, that is, to the stored data. It adds an important layer of protection that enforces security. Any user trying to access the data not only need the right password, but the encryption key as well. One advantage of this schema is that files can be unreadable to people that having access to the database, such as an operating system administrator, but no databases privileges. Database encryption affects performance and a compromise solution must be found between performance and security, only encrypting tables, or columns with sensitive information.

It is thought that most of the attacks come from outside but, in the real world, a large percentage of the security breaking incidents are made by insiders, for example, dishonest DBA, people working in the same organization with extra knowledge about the database structure and security policy. If the previous database security recommendations are followed, damage caused by insider's activities can be limited and audited. But this still can not solve the problem, like dishonest DBA. That leads to the idea of this project—finding a new security method if the hacker can access the table.

2.3. Oracle System and Develop Environment

Oracle8i Standard Edition, from Oracle, is the first database management system designed specifically for developing, deploying, and managing departmental Internet applications. In addition to being the premiere database for traditional departmental

applications, Oracle8i Standard Edition offers unprecedented ease-of-use, power, and price-performance for workgroups or departmental applications to Web-enable users' business. [15] The Oracle data server has been designed to meet the requirements of the network-computing era. To ensure that the Oracle data server is appropriate for both small, departmental applications and enterprise-wide computing, it is offered in two configurations, Oracle8 and Oracle8 Enterprise Edition.

Both Oracle8 and the Oracle8 Enterprise Edition provide reliable and secure data management for applications ranging from small departmental applications to high-volume on-line transaction systems, or query-intensive data warehouse applications. They also provide the tools for systems management, the flexibility to distribute data efficiently, and the scalability for optimal performance from computing resources [16]. The newest version of Oracle is Oracle9.

Oracle Forms Developer is Oracle's award-winning Internet Rapid Application Development tool--part of the Internet Developer Suite. It is a highly productive end-to-end development environment for building enterprise-class, database-centric Internet applications [11]. The tool suite includes a set of integrated builders, re-entrant wizards, and property palettes that enable developers to quickly construct sophisticated, multi-lingual and highly interactive forms, charts, and business logic with minimal effort. The extensible Java client offers developers vast potential to make web applications come alive with packaged samples, such as rollover buttons, web-link types, and desktop integration to access the local file system. [14]

2.4. PL/SQL Language

PL/SQL is Oracle's procedural extension to industry-standard SQL. PL/SQL naturally, efficiently, and safely extends SQL. Its primary strength is in providing a server-side, stored procedural language that is easy-to-use, seamless with SQL, robust, portable, and secure.

Furthermore, the PL/SQL compiler and interpreter are also embedded in Oracle Developer, providing developers with a consistent and leveraged development model on both the client and the server side. In addition, PL/SQL stored procedures can be called from a number of Oracle clients, such as Pro*C™ or Oracle® Call Interface, and from Oracle® Reports and Oracle Forms Developer.

Here is an example:

```
-- available online in file 'exampl'
DECLARE
  qty_on_hand  NUMBER(5);
BEGIN
  SELECT quantity INTO qty_on_hand FROM inventory
  WHERE product = 'TENNIS RACKET'
  FOR UPDATE OF quantity;
  IF qty_on_hand > 0 THEN -- check quantity
    UPDATE inventory SET quantity = quantity - 1
    WHERE product = 'TENNIS RACKET';
    INSERT INTO purchase_record
    VALUES ('Tennis racket purchased', SYSDATE);
  ELSE
    INSERT INTO purchase_record
    VALUES ('Out of tennis rackets', SYSDATE);
  END IF;
  COMMIT;
END;
/
```

With PL/SQL, users can use SQL statements to manipulate Oracle data and flow-of-control statements to process the data. Moreover, they can declare constants and variables, define procedures and functions, and trap runtime errors. Thus, PL/SQL combines the data manipulating power of SQL with the data processing power of procedural languages.

2.5. Using Java Class Methods in Oracle

In Oracle8, user can call Java sub-function via using PL/SQL easily. There are two ways to use Java class methods in Oracle. The first is use “loadjava” command to load the java class or source code in console. The second way is define and declare the java class methods on runtime. In this project, we choose the second way to load Java class and the steps are shown below [2]:

1. Create or replace file path.

This step is used for locate the path of the Java source code file or binary file.

When Oracle Database started, DBMS will go to this location and load user’s Java code.

```
CREATE OR REPLACE DIRECTORY test_dir as
'/Oracle/Ora81/java/CryptField'; -- Path of the target file
```

2. Create or replace class file name.

Create a new class in Oracle Database from user’s files. DBMS will load this class

and add it to the database.

```
CREATE OR REPLACE JAVA CLASS USING BFILE(test_dir, 'test.class');
```

3. Create or replace a running environment.

Create functions or procedures from the Java class method.

```
CREATE OR REPLACE PROCEDURE TEST_CALL()  
  
AS LANGUAGE JAVA  
  
NAME 'test.simadd()'; -- Invoke Java method
```

4. Call the java class method program

In the program, user can call the new subprogram directly.

```
CALL TEST_CALL(); -- Call new method from Oracle
```


CHAPTER 3

DESIGN AND IMPLEMENTATION

In this chapter, the author presents the methodology of the project and describes the program built in this project.

3.1. Methodology

Obviously, in order to add new functions to an existing database management system, we create a new layer between the existing database management system and the front-end application. The integration of this new layer does not change the Oracle's DBMS. It ensures the original security abilities, and original functions would not be weakened by the new program. This layer is the kernel of the entire project. Figure 2 gives a brief view of the relations between the Oracle database and the new program. Overall, this project adds some new logic operations those can enhance the database security.

First, user's requests are sent to the New Function Management System. In this part, all queries or commands are checked carefully. If some particular conditions are met,

this management system calls stored functions and procedures to process the user's requirements. The procedure may change the original request from the user and generate some new commands. New requests are sent to the database and the system returns the results to the user.

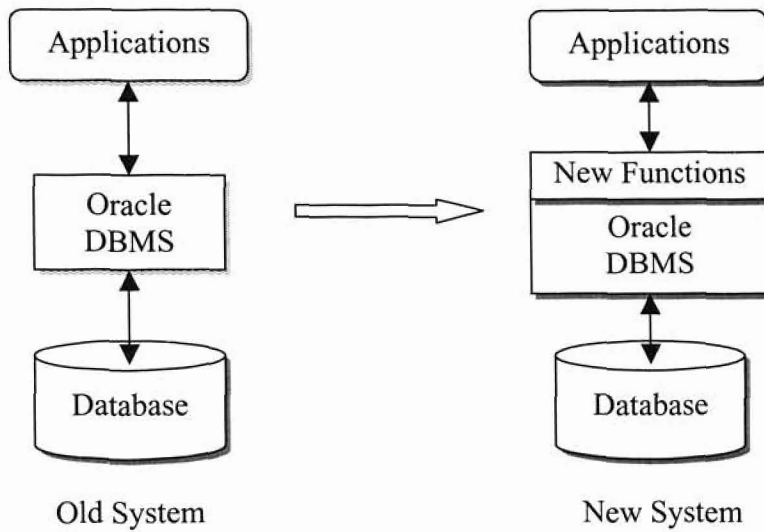


Figure 2: A Diagram of Structures with Oracle DBMS

3.1.1. Field Encryption

Field Encryption is an encryption applied to fields in a table. With some encryption methods, data is “unreadable” to the user or database administrator. The data should be encrypted before saved in the table. The user is the only one who has the key for encoding and decoding. Without the correct key, system may return an incorrect data without any warning. Normally, there are two keys in most encryption algorithms, private

key and public key. Some encryption methods, like DES, use only one key. A very special case is that some encryption methods are one-way algorithms. It is almost impossible to decode the cipher text to the original message like MD5.

As shown in Figure 3, program adds one new level between the user/application and DBMS. This new layer accepts the requests and commands from the user, determines what need to do next and sends new requests and commands to the database.

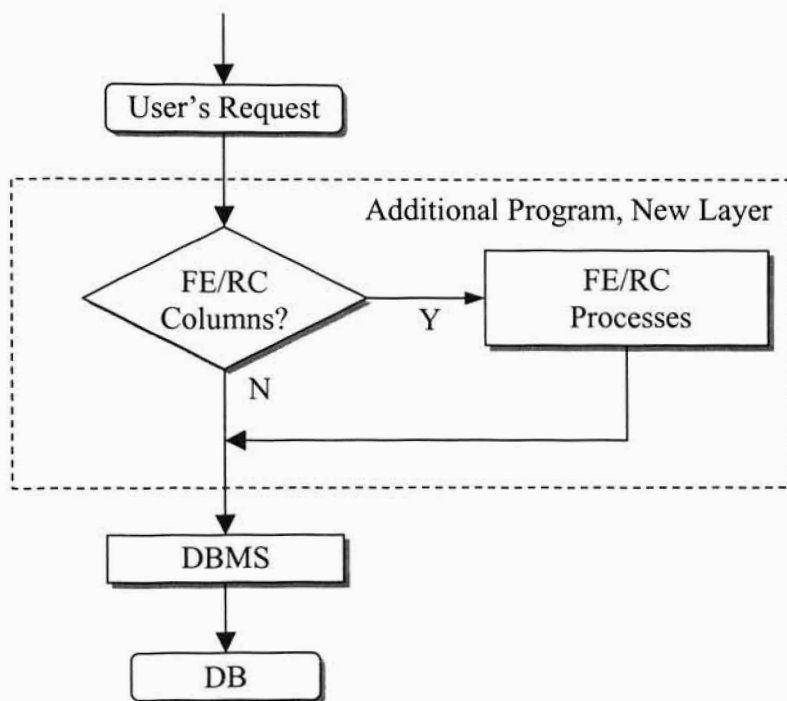


Figure 3. New Layer between User and DBMS

Generally, the methodology may be divided by these aspects:

1. User's Request Process
2. Add/Remove FE to A Table

3. SQL Query Process
4. Supported Data types
5. Limitation

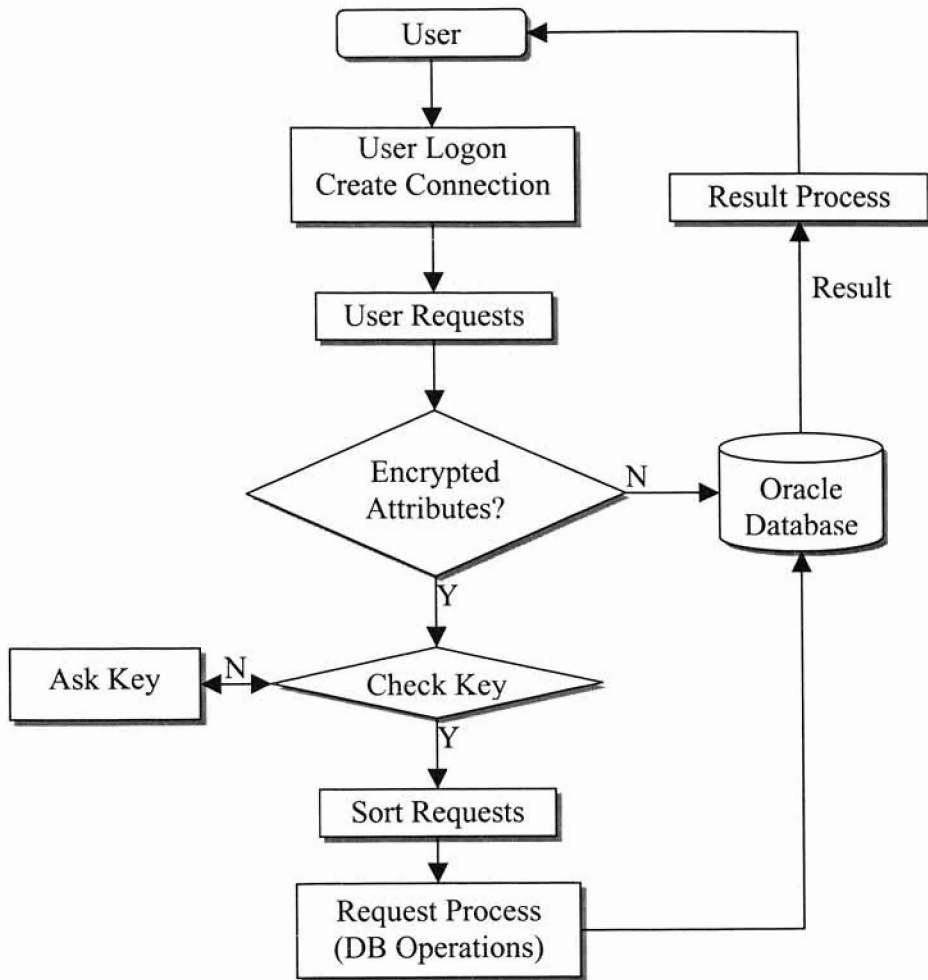


Figure 4. User's Request Process

1. User Request Process

While the user logs in and queries the database, management system may catch this event and verify the fire conditions. Specific operations would be executed if the requirements may affect the encrypted tables.

As shown in Figure 4, when user sends a SQL query to the database, for example, “SELECT NAME FROM EMP”, system checks the table “DS_FieldEncryption” and determines whether executing this query would infect some encrypted attributes.

If the query would infect some encrypted attributes, keys are required from user for decoding. User can submit the keys with their queries together by application or provide the keys when system asks in pop-up window. Program would use these keys to decode the cipher text in the table and return the plain text to the user. It should be noticed that system wouldn't tell the user whether the keys are correct. If the keys are not exactly the right keys for decoding, the program would return a different result or null value to the user. The value depends on different encryption methods. If the query does not infect an encrypted attribute, management system sends the query to Oracle's DBMS as normal and returns the results to user directly.

Some appropriate processes can be done if the requests meet the requirements. These include the query pre-process and result process. The SQL statements, like “Insert”, “Update”, “Delete” and “Select”, can be executed and monitored.

2. Add/Remove FE to a Table that Already Exist

By adding Field Encryption to a table that already exists, user can browse the attribute list and choose the one they would like to encrypt. In this step, both the attribute and the encryption method should be selected. Also, user must setup at least one key to access this field. Stored procedure encrypts all the data in this field after the user makes

the final decision. The “scene”, information used in this procedure, is stored in a system table called “DS_FieldEncryption”, which contains the name of the database, owner, table, column, the method of the encryption, etc.

The steps of this procedure are:

Step1. Choose Table, Column and Encryption Method

Step2. Setup Keys if Required

Step3. Encrypt Current Data

Step4. Update Table DS_FieldEncryption

These steps can be applied to remove operation if we change the step3 to “Decode Current Data”.

3. SQL Statement Pre-process

Depends on the size of the table, there are two ways to pre-process SQL statements. For small tables, internal procedures decode the encrypted column and then pass the request to database management system. This column should be re-encrypted after the result has been returned to the user successfully. The diagram of the procedure is shown in Figure 5. Encoding/decoding the whole column is easy to implement in both the algorithm and programming, but it is not efficient. Also, this method increases the risk of security during the processing.

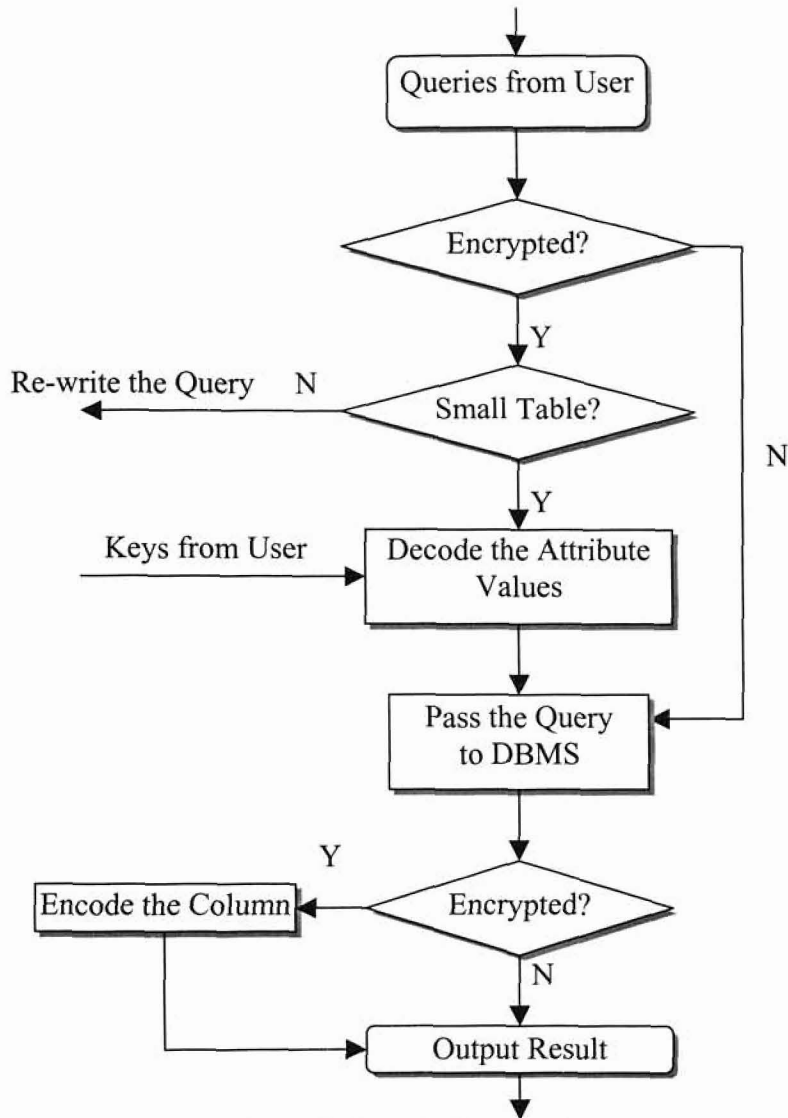


Figure 5. Decode/Encode the Attribute Values

The data may be leaked out before the column has been decoded again. In this project, we use this method for most testing program because it is the easier one.

Encoding and decoding the whole column of a big table is overhead and the running time could destroy all the advantages from this project. Then we use the second

method to perform the process in a better performance. No column needs to be decoded before the processing.

Subprograms may analyze each query and find the position of the encrypted attribute. Most of the queries can be sorted as three types. The first type: The value of the encrypted attributes should be returned to the user as show in Figure6, case a. Management system sends the query to the database and additional decoding operation would be applied after executing the query. The second type, case b: The encrypted attribute is the conditions in the query, for example, in the phase “WHERE”. The system encodes conditions into the cipher text and replaces the original message. This decoding process should be applied before executing. The last type, case c: The encrypted attributes appear in both of the two places, one should be returned to the user and another one is in the conditions that should be met. We need to combine the first and the second processes, encrypt the conditions and unencrypt the result before and after execute the query. These three cases are shown in Figure 6.

```
SELECT SAL FROM EMP WHERE ENAME="JOHN"
```

Case a. attribute SAL has been encrypted

```
SELECT SAL FROM EMP WHERE ENAME="JOHN"
```

Case b. attribute ENAME has been encrypted

```
SELECT SAL FROM EMP WHERE ENAME="JOHN"
```

Case c. attributes SAL and ENAME has been encrypted

Figure6. Different Positions of Encrypted Attribute in Query

A diagram of total processes is shown in Figure 7.

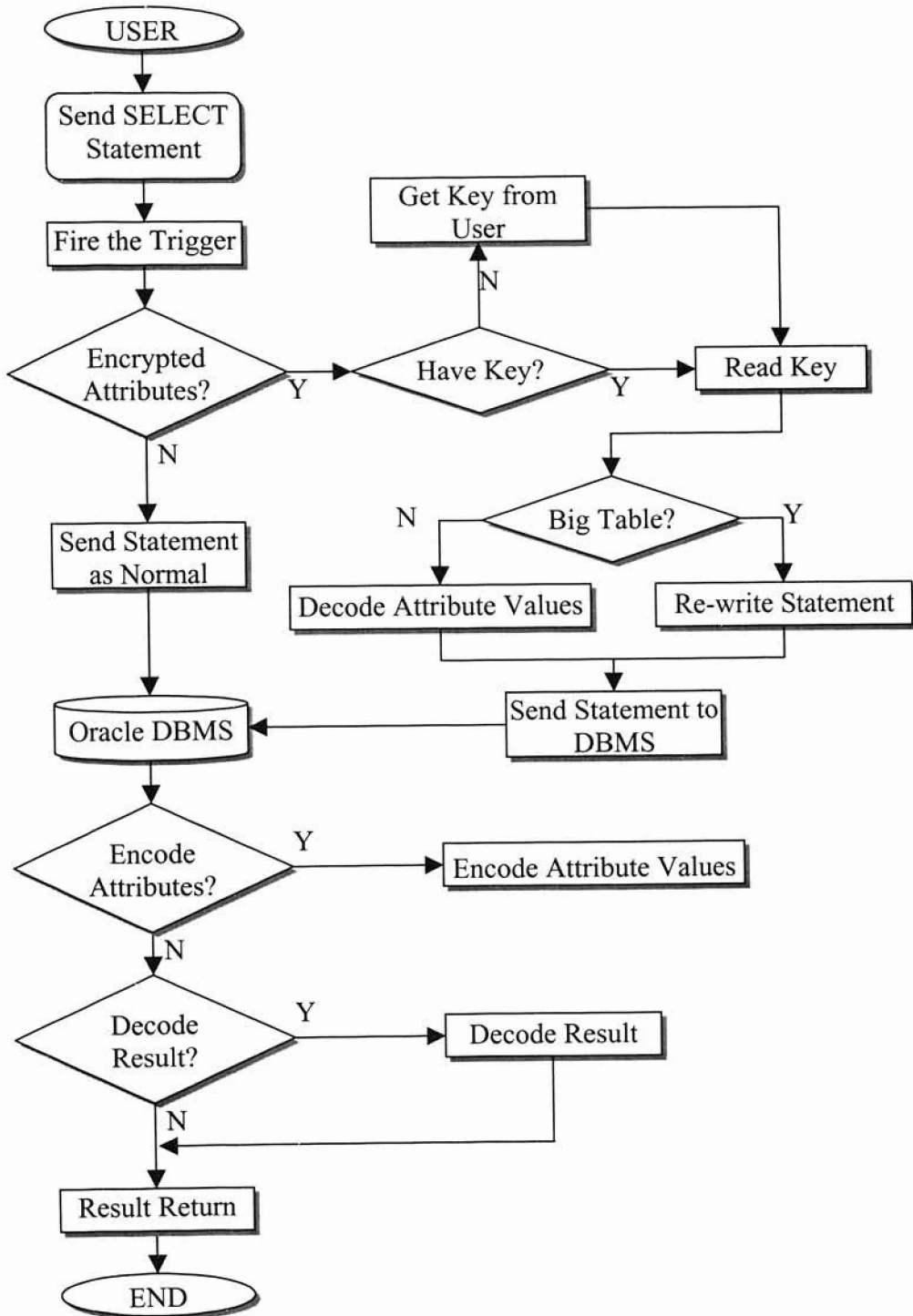


Figure 7. Sample: Process of the “SELECT” Statement

For example, when the user sends a query to database, “SELECT SAL FROM EMP WHERE ENAME=’JOHN’”, management system catches this request and checks the status from the table DS_FieldEncryption. In case a, system sends the query to the DBMA and gets the result “SAL=’a84jduai39cmgh29d7dttkj’”. Next, management system decodes the result and return “SAL=2500” to the user. In case b, system encodes the query first and sends the new query “SELECT SAL FROM EMP WHERE ENAME=’12sde3f56gfr678juygfr45’” to the DBMS, then returns the result to the user directly. In case c, system uses both of these two procedures.

4. Supported Data type

Not all the data types can be encrypted in this program. Because most of the encryption methods are used for ASCII text (plain text) only, the data types such as CHAR, VARCHAR2, and NUMBER can be encrypted but the binary data types like BFILE cannot be encrypted in this project. Since Oracle8i can convert the data types automatically, this program supports any data type that can be converted to VARCHAR2. The list of the build-in data types that supported by this program is shown in Table 1. This program doesn’t support user defined data types.

Data Type	Support	Data Type	Support
CHAR	Y	DATE	Y
VARCHAR2	Y	RAW	Y
NCHAR	Y	LONG RAW	N
NVARCHAR2	Y	ROWID	N
NUMBER	Y	UROWID	N
LONG	Y	CLOB	N

Data Type	Support	Data Type	Support
REAL	Y	NCLOB	N
INT	Y	BLOB	N
BIC_INT	Y	PLS_INT	Y
FLOAT	Y	BFILE	N

Table 1. Supported Data Types (Y=Yes, N=No)

5. Special Cases

If the data has been encrypted by some on-way encryption algorithms, for example, MD5, data cannot be retrieved after encryption. System must monitor this kind of encryption methods and give a warning to the user. This kind of encryption methods are only used in the attributes like “Password”, “Social Security Number”.

3.1.2. Row Level Control

Row Level Control is a grained access control that users can only access the rows that pertain to him/her. Multiple communities of users can access the same database, yet obtain only a specific set of records authorized to use. The typical application of Row Level Control is the independent suppliers can update the same table of products without getting the information that not belongs to him/her.

1. Initial (Add/Remove) a Table

An additional attribute called “DS_RC_RCL” (VARCHAR2) is added to the existing table. The initial value is “NULL”. Ordinary users who have the privileges to access this table can retrieve these tuples. The owner of the table or database

administrator can set some different access levels and assign the user to each level. All the information used in this function is stored in system table “DS_RowLevelControl”.

2. Access the Table

When the user accesses a table with Row Level Control, the system checks if the table has been controlled and the user has the privileges to access it. If this is the case, system retrieves information from the table “DS_RowLevelControl” which contains the information for this function like the user name, access level, and the table name, etc. The queries from the user must be changed by management system. It integrates the condition of access level (Access_Level) to the clause “WHERE” and submits this new query to the DBMS. This operation is not transparent to the user. He/she doesn’t know that the queries are changed and some tuples are skipped by the DBMS. Each user gets a “Visual Table” from the original table. In Figure 8, Table1 is the original table with two row control levels, RC1 and RC2. USER1 can access the rows which the value of RCL is “RC1” or “NULL. To USER1, the target table is a new virtual table, Table2, instead of Table1. This virtual table includes the rows that the RCL is “RC1” or “NULL” in Table1. Also, the virtual table for USER2 is Table3. So even though two users submit the same query, the results turned to them may be different.

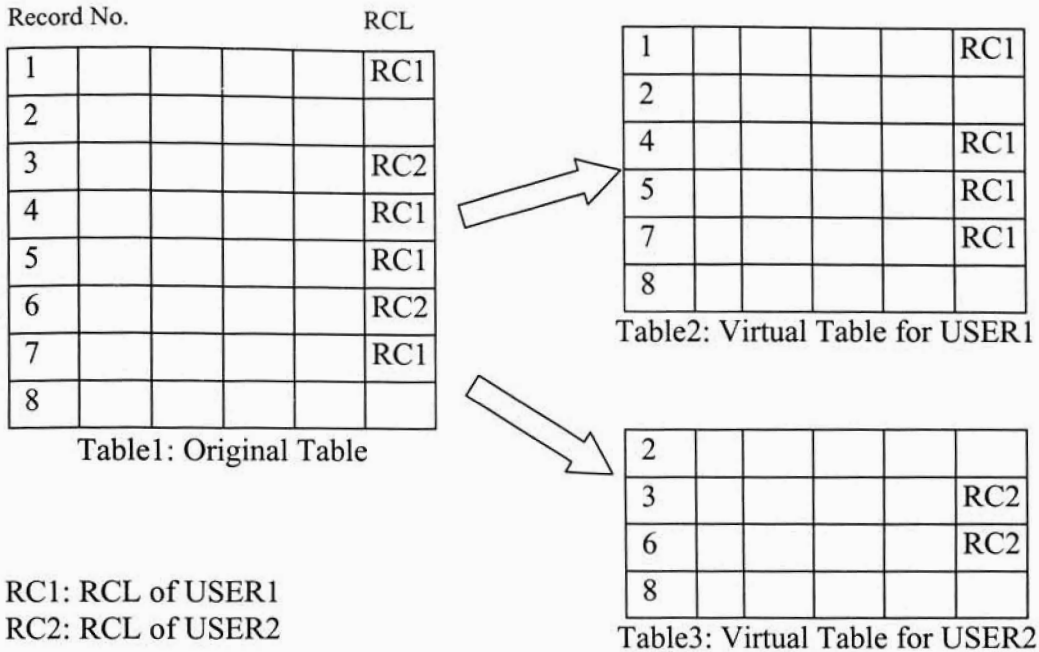


Figure 8. Visual Table for Each User

3.2. The Program

The requests for this program are listed below.

- A user-friendly interface
- Create some tables used to store the required information
- Some triggers or subprograms that can monitor the statements such as “Select”, “Insert”, “Delete”, “Update”, “Drop”, etc
- The encryption algorithm that presented by Java language
- Stored procedures and functions that can be called by both the database and the applications
- Auditing Utilizes

- Test program for the result and the performance

The program of this project includes four parts. The first part is the functions and procedures running in the database. Next, the second part is a management system independent with DBMS. Interface to the user and database is the third part in the program and the last part is the testing program that used for result and performance test.

3.2.1. The Main Program

The main program is created by Oracle Forms Developer, Forms Builder6, which has a comprehensive application framework to deploy enterprise-class applications with a rich Java interface. Main program integrates all the functions and gives the user a friendly interface. It is necessary that the main program must check the completion of each part required in this application. As a management system to enhance the security ability of database, main program can check the authorization of the user and create a connection to the Oracle.

User can use this program to maintain Field Encryption and Row Level Control. To maximize the applicable area, the program can work in two modes. One is to work in Server-Client mode or as a normal window application. The other is to run in any machine which has JVM supported via the Internet.

To perform all the functions required in this project, the program has been grouped into some modules as shown in Figure 9:

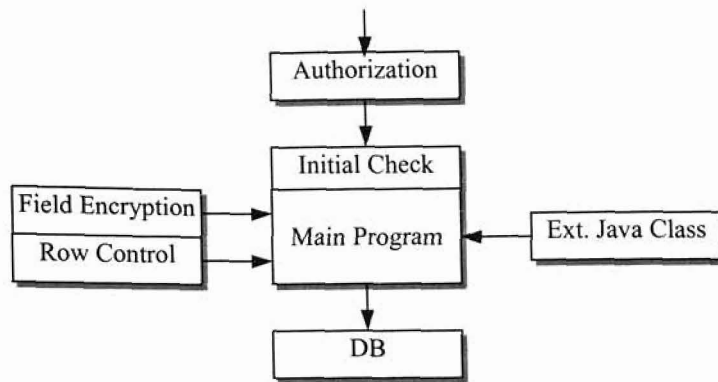


Figure 9. The Structure of the Program

The first module is “Authorization”. This part is designed to provide security protection and create a connection to the database. User must provide correct username and password based on the account of the database. This program doesn’t have its own account management system. The second module is used to check if all the parts required in this program exist in database. The checking list is tables, Java class files, subprograms, and Java methods. For example, system checks if the table “DS_FieldEncryption” is correct and the build-in function “DS_Encrypt” is valid. If not, a new corrected copy of these damaged parts will be created in the appropriate schema. User can achieve all the functions, FE and RLC, from the main program. With the help of a user-friendly interface, they can add or remove Field Encryption and Row Level Control to a table. The next module is testing program, user can verify the result of Field Encryption and Row Level Control. Also, the performance test program can calculate the space and time cost in each operation.

The interface of this main program may separate to five parts, “Menu Bar”, “Tools bar”, “Detail Tree of Database”, “Details of Table”, and “Function Area”. The functions for each part are listed below.

- Menu Bar: Display a list of common commands. Menu items include some common operations of the database.
- Tools Bar: Contains buttons with images that for basic operations
- Detail Tree of the Database: A Hierarchical Tree that shows the details of the current database.
- Table Details: The details of the current table
- Function Area: List all the options for both of the two functions, FE and RC.

The interface of the main program is shown in Figure 10. (Appendix C)

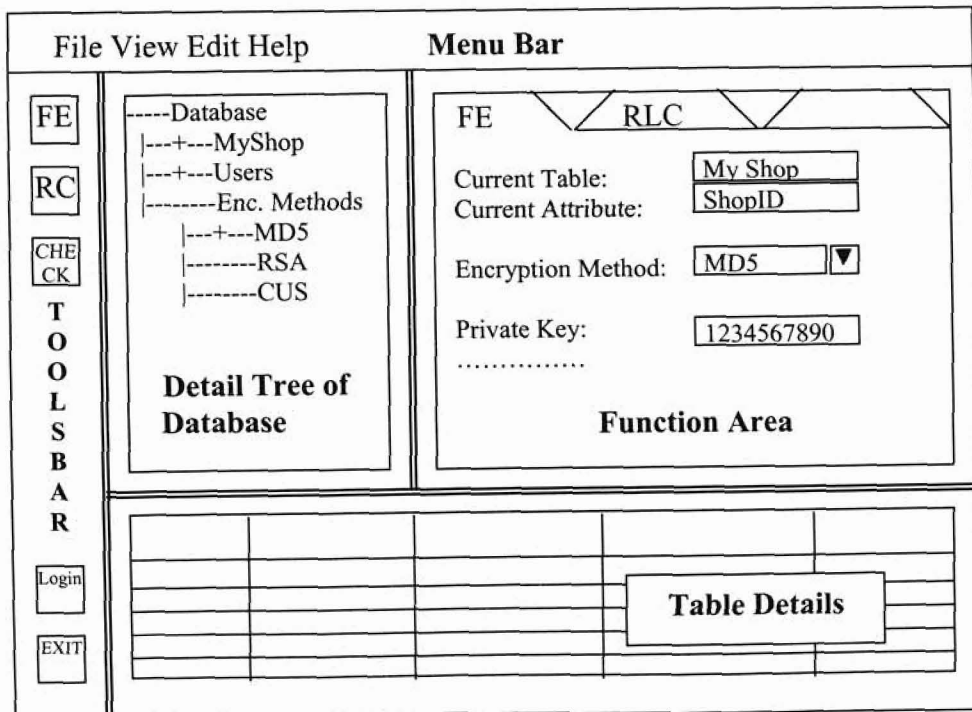






Figure 10. Interface of the Main Program


There are six buttons in the Tools Bar. User can choose each function by clicking the button.


Field Encryption: Field Encryption function, icon: 

Row Level Control: Row Level Control function, icon: 

Connect: connect to the database with a new user, icon: 

Disconnect: disconnect with database, icon: 

Test Program: call “Test Program” to test the database, icon: 

Exit: Exit the program, icon: 

The hierarchical tree in “Detail Tree of Database” can display the information of current database. User may expand and collapse the tree nodes and choose the table by clicking the name. A typical tree in running mode is shown in Figure 11.

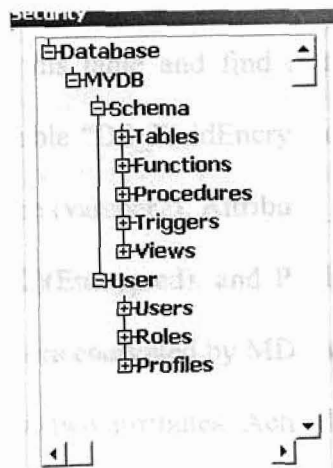


Figure 11. Hierarchical Tree

Once the user chooses a table from the tree, “Table Details” shows the details of the table, the name, columns, length, data type, and owner.

“Function Area” gives all the options for FE and RLC. In this part, user can setup Field Encryption and Row Level Control options by different tab pages. Each tab page is used for a specific function. Three of these tab pages are used to display the value of the table “DS_FieldEncryption”, “DS_RowLevelControl”, and “DS_EncMethod”.

3.2.2. Elements in a Database

Obviously we need additional space to store the information for each function. There are three major tables should be added to the database, “DS_FieldEncryption”, “DS_RowLevelControl”, and “DS_EncMethod”. All of these tables should be created in “SYS” table space.

These tables should have all the information for adding and removing functions FE and RC. System can check this table and find if the destination table has been encrypted. The attributes of the table “DS_FieldEncryption” are TB_Owner (varchar2), DB_Name (varchar2), Table_Name (varchar2), Attribute_Name (varchar2), Enc_Method (varchar2), Private_Key (varchar2)(Encrypted), and Public_Key (varchar2)(Encrypted). The Private_Key and Public_Key are encrypted by MD5 and only used for debugging. In final program, we don’t need these two attributes. Actually, the column “private_key” is only used for reference. In the final application, no keys are stored in database.

The attributes of the table “DS_RowLevelControl” are Owner (varchar2), Table_Name (varchar2), Access_Level (varchar2), Access_User (varchar2), and Access_Group (varchar2). User defines the value of Access_Level and it can be any text like “Supplier1” and “Supplier2”. The name of the user who is granted with this access level is stored in column Access_User. If the access level is assigned to a user group, the information will be stored in attribute Access_Group.

The table “DS_EncMethod” is used to save the list of the encryption methods. The columns are Enc_Name (varchar2), Enc_Class (varchar2), Input_Datatype (varchar2), Input_Datalength (varchar2), class_path (varchar2)(For future works), key# (Number).

3.2.3. Subprograms in a Database

To insure the program can run as our expectation, we should create the procedures and functions not only in the management system, but also in the database. Both of the application and management system can call these subprograms from the internal or external of the database.

The major functions for these subprograms are shown in Table 2. A full list is shown in APPENDIX A.

Name	Location	Function
DS_Encrypt	Functions-SYS	Encrypt the field, call by other program from internal or external
DS_UnEncrypt	Functions-SYS	Un-encrypt the column
DS_RowControl	Functions-SYS	Add Row Level Control to the table
DS_UnControl	Functions-SYS	Delete the Row Level Control in the table

Table 2. Major Subprograms in a Database

3.2.4. Test Program

The test program is used to test the results of the project. It should simulate an application and interact with the database. Based on the functions we need to achieve, this program should perform these tests:

1. Add/Remove FE to a table
2. Add/Remove RC to a table
3. Basic inquire operation for both FE and RC
4. Basic DML test for both FE and RC

Additionally, a simple performance test in this program is built in test program. It can calculate the time and space required in each function.

CHAPTER 4

RESULTS AND FINDINGS


In this chapter, the author presents results of functionality tests and also gives a simple performance test for time and space complexity analysis.

4.1. Results

The result of this project gives an enhanced security to the Oracle database management system. It provides an alternative way for the user and the database administrator to protect their data. Using Field Encryption can keep the data safe even though the password has been divulged. With the protection of Row Level Control, different users can only access part of the tuples that are granted to them.

Finally, we can enhance the security of the database. This enhanced security method is a “shell” around the database. First, queries from users and applications should be passed to this “shell”, and then submitted to the DBMS. Logically, this procedure can be applied to any applications.

4.2. Program Test

The tests include basic functions test and the basic operation tests. In order to test the result of this project, a program was built to test each function. This program can simulate the front-end applications, accept queries from the user or forms and return the results to the user. It can be activated by clicking the button  in the “Tools Bar”.

A special environment should be created for testing. This environment contains some database users and tables. These test users should have different roles and privileges. The tables should have some attributes with different data types. A list of these testers and privileges is shown in Table 3.

Username	Password	Role	System Privilege
HAO	*****	DBA	SYSDBA
TESTUSER	*****		
TESTUSER1	*****		
TESTUSER2	*****	CONNECT	BECOME USER
TESTUSER3	*****	CONNECT	SYSOPER
TESTUSER4	*****	DBA	SYSDBA

Table 3. Test Users and Privileges

The tablespace for these users is “USER”. Each user has at least three test tables, “EMP”, “EMP1”, and “EMP2”, respectively. User “HAO” has some additional tables like “TEST” and “TEST2”. The structure of the table is shown in Table 4. It has eight columns and three data types.

Table Name	Name	Display Name	Width	Is Visible	columnDataType
"HAO"."EMP"	"EMPNO"	EMPNO	100	<input checked="" type="checkbox"/>	NUMBER
"HAO"."EMP"	"ENAME"	ENAME	100	<input checked="" type="checkbox"/>	VARCHAR2
"HAO"."EMP"	"JOB"	JOB	100	<input checked="" type="checkbox"/>	VARCHAR2
"HAO"."EMP"	"MGR"	MGR	100	<input checked="" type="checkbox"/>	NUMBER
"HAO"."EMP"	"HIREDATE"	HIREDATE	100	<input checked="" type="checkbox"/>	DATE
"HAO"."EMP"	"SAL"	SAL	100	<input checked="" type="checkbox"/>	NUMBER
"HAO"."EMP"	"COMM"	COMM	100	<input checked="" type="checkbox"/>	NUMBER
"HAO"."EMP"	"DEPTNO"	DEPTNO	100	<input checked="" type="checkbox"/>	NUMBER

Table 4. The Structure of the Table for Testing

To perform the testing, each table should have some records as shown in Table 5.

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-Dec-1980 12:00:00 AM	800		20
7499	ALLEN	SALESMAN	7698	20-Feb-1981 12:00:00 AM	1600	300	30
7521	WARD	SALESMAN	7698	22-Feb-1981 12:00:00 AM	1250	500	30
7566	JONES	MANAGER	7839	02-Apr-1981 12:00:00 AM	2975		20
7654	MARTIN	SALESMAN	7698	28-Sep-1981 12:00:00 AM	1250	1400	30
7698	BLAKE	MANAGER	7839	01-May-1981 12:00:00 AM	2850		30
7782	CLARK	MANAGER	7839	09-Jun-1981 12:00:00 AM	2450		10
7788	SCOTT	ANALYST	7566	19-Apr-1987 12:00:00 AM	3000		20
7839	KING	PRESIDENT		17-Nov-1981 12:00:00 AM	5000		10
7844	TURNER	SALESMAN	7698	08-Sep-1981 12:00:00 AM	1500	0	30
7876	ADAMS	CLERK	7788	23-May-1987 12:00:00 AM	1100		20
7900	JAMES	CLERK	7698	03-Dec-1981 12:00:00 AM	950		30
7902	FORD	ANALYST	7566	03-Dec-1981 12:00:00 AM	3000		20
7934	MILLER	CLERK	7782	23-Jan-1982 12:00:00 AM	1300		10

Table 5. The Content of the Table "EMP"

For Row Level Control, additional table called "TEST_RC" and "TEST_RC1" were created for user "HAO". The structure of the tables are as same as the table EMP. Table "TEST_RC3" is empty as shown in Table 6.

EMPNO ‡ ENAME ‡ JOB ‡ MGR ‡ HIREDATE ‡ SAL ‡ COMM ‡ DEPTNO ‡

Table 6. An Empty Table for Row Level Control

In the test of Field Encryption and Row Level Control, we set some special conditions to obtain an environment as close as the real world. We will explain each condition below during the test. Testing should be well planned and organized. All of the test procedures are numbered so that it is easy to track and analyze the results. Each test has been well designed and we are trying to cover the cases as much as possible.

4.2.1. Field Encryption Test

Table 7 is a list of the tests for Field Encryption.

Test No.	Test Type	Method	Description
TEST FE-1	Add/Remove	DES	Add/Remove FE to a Table
TEST FE-2	Add	MD5	Add MD5 to a Table
TEST FE-3	SQL	SELECT	Test “Select” Query
TEST FE-4	SQL	INSERT	Test “Insert” Query
TEST FE-5	SQL	UPDATE	Test “Update” Query

Table 7. The Test List for Field Encryption

Test FE-1: Basic Function Test – Add and Remove Field Encryption (DES).

This is a test that verifies whether the management system can add and remove FE with DES encryption method to a table.

Test Conditions:

Table: HAO.EMP

Encryption Method: DES

Key: 12345678

After encoding, one or more fields (columns) were encrypted. The data of encrypted attributes is invisible to the user and database administrator. In Table 8, the field “JOB” was encrypted by DES encryption method. The data in column “JOB” is unmeaning. No one can get the real information from such data without the correct key.

EMPNO	ENAME	MGR	HIREDATE	SAL	COMM	DEPTNO	JOB
7369	SMITH	7902	17-Dec-1980 12:00:00 AM	800		20	a0fa72026bd7209f
7499	ALLEN	7698	20-Feb-1981 12:00:00 AM	1600	300	30	7e5aa5f5707c3193deff9...
7521	WARD	7698	22-Feb-1981 12:00:00 AM	1250	500	30	7e5aa5f5707c3193deff9...
7566	JONES	7839	02-Apr-1981 12:00:00 AM	2975		20	68c5deb91fd37064
7654	MARTIN	7698	28-Sep-1981 12:00:00 AM	1250	1400	30	7e5aa5f5707c3193deff9...
7698	BLAKE	7839	01-May-1981 12:00:00 AM	2850		30	68c5deb91fd37064
7782	CLARK	7839	09-Jun-1981 12:00:00 AM	2450		10	68c5deb91fd37064
7788	SCOTT	7566	19-Apr-1987 12:00:00 AM	3000		20	017abb04a603f8f3
7839	KING		17-Nov-1981 12:00:00 AM	5000		10	b2fac0c7111a28fabb85...
7844	TURNER	7698	08-Sep-1981 12:00:00 AM	1500	0	30	7e5aa5f5707c3193deff9...
7876	ADAMS	7788	23-May-1987 12:00:00 AM	1100		20	a0fa72026bd7209f
7900	JAMES	7698	03-Dec-1981 12:00:00 AM	950		30	a0fa72026bd7209f
7902	FORD	7566	03-Dec-1981 12:00:00 AM	3000		20	017abb04a603f8f3
7934	MILLER	7782	23-Jan-1982 12:00:00 AM	1300		10	a0fa72026bd7209f

Table 8. The Column “JOB” has been Encrypted by DES

The de-encrypt operation is based on the same attribute, “JOB”. After decoding, the column “JOB” was “recovered” and “converted” to the original data. This result of table “EMP” is shown in Table 9. The contents of the table are as same as the table before encryption. (See. Table 4)

EMPNO	ENAME	MGR	HIREDATE	SAL	COMM	DEPTNO	JOB
7369	SMITH	7902	17-Dec-1980 12:00:00	800		20	CLERK
7499	ALLEN	7698	20-Feb-1981 12:00:00	1600	300	30	SALESMAN
7521	WARD	7698	22-Feb-1981 12:00:00	1250	500	30	SALESMAN
7566	JONES	7839	02-Apr-1981 12:00:00	2975		20	MANAGER
7654	MARTIN	7698	28-Sep-1981 12:00:00	1250	1400	30	SALESMAN
7698	BLAKE	7839	01-May-1981 12:00:00	2850		30	MANAGER
7782	CLARK	7839	09-Jun-1981 12:00:00	2450		10	MANAGER
7788	SCOTT	7566	19-Apr-1987 12:00:00	3000		20	ANALYST
7839	KING		17-Nov-1981 12:00:00	5000		10	PRESIDENT
7844	TURNER	7698	08-Sep-1981 12:00:00	1500	0	30	SALESMAN
7876	ADAMS	7788	23-May-1987 12:00:00	1100		20	CLERK
7900	JAMES	7698	03-Dec-1981 12:00:00	950		30	CLERK
7902	FORD	7566	03-Dec-1981 12:00:00	3000		20	ANALYST
7934	MILLER	7782	23-Jan-1982 12:00:00	1300		10	CLERK

Table 9. The Table “EMP” after Decoding Operation

Conclusion: This program can add and remove FE to a table by DES encryption method.

Test FE-2: Basic Function Test – Add and Remove Field Encryption (MD5).

This test is as same as Test FE-1, but no decoding operation required because MD5 is on-way encryption method. The test is designed to check the function FE and we don't recommend encrypting the attributes by MD5 until the user very sure about that.

Test Conditions:

Table: EMP

Encryption Method: MD5

MD5 is one-way encryption method. The data that been encrypted can not be decode again. To test this function, we encode the attribute “JOB” and the result is shown in Table 10 below. For the correctness of the MD5, please check the APPENDIX-B.

EMPNO	ENAME	MGR	HIREDATE	SAL	COMM	DEPTNO	JOB
7369	SMITH	7902	17-Dec-1980 1...	800		20	9a9a36d86bfa94fcbf00bdcaa37a26b
7499	ALLEN	7698	20-Feb-1981 1...	1600	300	30	f18eefcc83a5a935719f4425e4a3ccaa
7521	WARD	7698	22-Feb-1981 1...	1250	500	30	f18eefcc83a5a935719f4425e4a3ccaa
7566	JONES	7839	02-Apr-1981 12...	2975		20	929a43a844650a2ba236b9a50d1bbd2c
7654	MARTIN	7698	28-Sep-1981 1...	1250	1400	30	f18eefcc83a5a935719f4425e4a3ccaa
7698	BLAKE	7839	01-May-1981 1...	2850		30	929a43a844650a2ba236b9a50d1bbd2c
7782	CLARK	7839	09-Jun-1981 1...	2450		10	929a43a844650a2ba236b9a50d1bbd2c
7788	SCOTT	7566	19-Apr-1987 12...	3000		20	ac6a408e6279ea17541e0da336295a6e
7839	KING		17-Nov-1981 1...	5000		10	d0b04d0db4037ea35a7f72dad8c9c6f1
7844	TURNER	7698	08-Sep-1981 1...	1500	0	30	f18eefcc83a5a935719f4425e4a3ccaa
7876	ADAMS	7788	23-May-1987 1...	1100		20	9a9a36d86bfa94fcbf00bdcaa37a26b
7900	JAMES	7698	03-Dec-1981 1...	950		30	9a9a36d86bfa94fcbf00bdcaa37a26b
7902	FORD	7566	03-Dec-1981 1...	3000		20	ac6a408e6279ea17541e0da336295a6e
7934	MILLER	7782	23-Jan-1982 1...	1300		10	9a9a36d86bfa94fcbf00bdcaa37a26b

Table 10. The Attribute “JOB” has been Encrypted by MD5

Conclusion: This program can add Field Encryption by MD5 to a table.

Test FE-3: SQL Statement Test – SELECT

This test is based on a table called “HAO.EMP”. The attribute “JOB” is encrypted by DES. Program simulates an application and sends query “SELECT” to the management system. Since the attribute “JOB” can appear in two positions in each statement, we test “SELECT” statement by at least two different queries.

Test Conditions:

1. Table: HAO.EMP2
2. Encryption Method: DES
3. Encrypted Attribute: JOB
4. Key: 12345678

1) SQL Statement 1: The encrypted attribute is after SELECT and should be returned to the user.

SELECT JOB WHERE DEPT=10;

Result without the key: *NULL*

Result with the key: *MANAGER*

PRESIDENT

CLERK

Result with wrong key "123456": *NULL*

2) SQL Statement 2: The encrypted attribute is in condition WHERE

SELECT EMPNO WHERE JOB="CLERK";

Result without the key: *NULL*

Result with the key: *7369*

7876

7900

7934

Result with wrong key "123456": *NULL*

Conclusion: User can access the encrypted table by SQL statement “SELECT”.

System returns the right answer with the correct key.

Test FE-4: SQL Statement Test – INSERT

This is used to test whether the SQL statement “INSERT” can work in an encrypted table.

Test Conditions:

1. Table: HAO.EMP
2. Encryption Method: DES
3. Encrypted Attribute: JOB
4. Key: 12345678

SQL Statement: INSERT INTO HAO.TEMP2 (EMPNO, ENAME, JOB, DEPTNO) VALUE (8000, 'HAO', 'TESTJOB', 10)

Result: Use the Oracle to check the contents of the table HAO.EMP2. The record had been inserted to the table. The “Insert” statement can work on an encrypted table.

Conclusion: User can insert a record to an encrypted table if the user has the correct key.

Test FE-5: SQL Statement Test – Update

Test FE-5 tests if the SQL statement “Update” can work in an encrypted table. Like “Select” command, we should test at least two “Update” statements. One is set the encrypted attribute in the phrase “WHERE” as one of the update conditions. Another one is the encrypted attribute is in the inquiry phrase and should be returned to the user.

Test Conditions:

1. Table: HAO.TEMP
2. Encryption Method: DES
3. Encrypted Attribute: JOB
4. Key: 12345678

SQL Statement 1: UPDATE HAO.TEMP SET JOB='TESTCLERK' WHERE EMPNO=8000;

Result: Use the statement “SELECT JOB FROM HAO.TEMP WHERE EMPNO=8000;” to test the result. Then the system returns “TESTCLERK”.

SQL Statement 2: UPDATE HAO.TEMP SET SAL=2222, DEPTNO=30 WHERE JOB='TESTCLERK';

Result: Use the statement “SELECT SAL, DEPTNO FROM HAO.TEMP WHERE EMPNO=8000;” to test the result. Then the system returns “2222” and “30”.

The final result after all these operations above is shown in Table 11.

EMPNO	ENAME	MGR	HIREDATE	SAL	COMM	DEPTNO	JOB
7369	SMITH	7902	17-Dec-1980 ...	800		20	a0fa72026bd7209f
7499	ALLEN	7698	20-Feb-1981 ...	1600	300	30	7e5aa5f5707c3193...
7521	WARD	7698	22-Feb-1981 ...	1250	500	30	7e5aa5f5707c3193...
7566	JONES	7839	02-Apr-1981 ...	2975		20	68c5deb91fd37064
7654	MARTIN	7698	28-Sep-1981 ...	1250	1400	30	7e5aa5f5707c3193...
7698	BLAKE	7839	01-May-1981 ...	2850		30	68c5deb91fd37064
7782	CLARK	7839	09-Jun-1981 ...	2450		10	68c5deb91fd37064
7788	SCOTT	7566	19-Apr-1987 ...	3000		20	017abb04a603f8f3
7839	KING		17-Nov-1981 ...	5000		10	b2fac0c7111a28fab...
7844	TURNER	7698	08-Sep-1981 ...	1500	0	30	7e5aa5f5707c3193...
7876	ADAMS	7788	23-May-1987 ...	1100		20	a0fa72026bd7209f
7900	JAMES	7698	03-Dec-1981 ...	950		30	a0fa72026bd7209f
7902	FORD	7566	03-Dec-1981 ...	3000		20	017abb04a603f8f3
7934	MILLER	7782	23-Jan-1982 ...	1300		10	a0fa72026bd7209f
8000	HAO			2222		30	3027d7d9605fefe5...

Table 11. The Final Result after the Field Encryption Tests

Conclusion: Users can update the records in an encrypted table.

4.2.2. Row Level Control Test

Like the testing in Field Encryption, first we should test whether the user can add and remove Row Level Control in a table. Second, we test basic query operation. The list of the tests is shown in Table 12.

Test No.	Test Type	Method	Description
TEST RC-1	Add/Remove		Add/Remove RC to a Table
TEST RC-2	SQL	SELECT	Test "Select" Query
TEST FE-3	SQL	UPDATE	Test "Update" Query

Table 12. A List of the Tests for Row Level Control

Test RC-1: Basic Function Test – Add and Remove Row Level Control.

To test the result of this function, we set two test methods. One is to create an empty table, “TEST_RC1”, and insert new records to it. The second is to use an existing table called “TEST_RC2”.

Test Conditions:

1. Table: HAO.TEST_RC1 (Empty Table), TEST_RC2
2. Owner: HAO

Users who can access this table are HAO, TESTUSER1, TESTUSER2, TESTUSER3, and TESTUSER4. The sample privileges for each user are shown in Table 13.

Username	SELECT	UPDATE	INSERT	DELETE
TESTUSER1	✓			
TESTUSER2	✓	✓		
TESTUSER3	✓	✓	✓	✓
TESTUSER4	✓	✓	✓	✓
HAO	✓	✓	✓	✓

Table 13. Privileges for Each User

The symbol “✓” means this user has the privilege of the operation. We set the sample row control level as shown in Table 14. These levels are identifications to distinguish the different users. User can set them to any words.

Username	New Level
TESTUSER1	RC1
TESTUSER2	RC2
TESTUSER3	RC3
TESTUSER4	RC4

Table 14. A List of New Row Control Levels

The next step is inserting some records to this table by different users. For example, login with the user “TESTUSER3” and insert one record to the table by the query “INSERT INTO HAO.TEST_RC1 (EMPNO, ENAME, JOB, SAL, DEPTNO) VALUES (3000, ‘N3000’, ‘SALESMAN’, 3000, 20)”. Because only the user “TESTUSER3”, “TESTUSER4”, and “HAO” have the privileges to insert a new record to the table “HAO.TEST_RC1”, the value of the field “DS_RC_RCL” is “RC3”.

In the beginning of the test, we change the current user to the owner “HAO” and insert some new records to make the table a little bit complex. The value of attribute “DS_RC_RCL” in these new records is “NULL”. The result of the table “HAO.TEST_RC1” is shown in Table 15.

EMPNO	ENAME	JOB	MGR	HIR...	SAL	CO...	DEPT...	DS_RC_RCL
3000	N3000	SALESMAN			3000		20	RC3
3010	N3010	MANAGER			3500		30	RC3
3020	N3020	CLERK			3020		10	RC3
3030	N3030	CLERK			3030		10	RC3
4010	N4010	CLERK			4000		20	RC4
4020	N4020	MANAGER			5000		10	RC4
4030	N4030	SALESMAN			4030		10	RC4
5000	N5000	CLERK			3000		20	RC4
9999	HAO	SALESMAN			9999		20	
9800	H1	CLERK			9000		30	
9000	H2	MANAGER			8000		10	

Table 15: After “INSERT” Operation in Table “TEST_RC1”

In the last column, “DS_RC_RCL”, the new level “RC3” and “RC4” mean that those tuples are added by user “TESTUSER3” and “TESTUSER4”. Value “NULL” implies that those records are added by the owner of this table, “HAO”, if the owner does not set any row control level during the inserting.

Test RC-2: SQL Statement – SELECT

This test is a little bit complex because we should access the table as different users. To test the result of the basic SQL operation, some additional tuples are added to the table. These new records are granted to the user “TESTUSER1” and “TESTUSER2”. Then we have the chance to test if this program works.

We test this function by at least two SQL statements and we must execute each statement by different users.

Test Conditions:

Table: The Table used in last test “HAO.TEST_RC1”.

User: Variable

SQL Statements:

① SELECT ENAME FROM HAO.TEST_RC1 WHERE JOB='CLERK';

② SELECT JOB FROM HAO.TEST_RC1 WHERE DEPTNO=10;

Result:

Here we only list the number of the records that have been returned.

No.	Username	SQL Statement#	Number of Result (No RLC)	Number of Result (RLC)
1	TESTUSER1	①	10	2
2	TESTUSER1	②	8	2
3	TESTUSER2	①	10	3
4	TESTUSER2	②	8	2
5	TESTUSER3	①	10	4
6	TESTUSER3	②	8	3
7	TESTUSER4	①	10	4
8	TESTUSER4	②	8	4

Table 16. The Result of “SELECT” Statement in RC

In Table 16, the first column is the name of the user. We test the program by login with different users. The second column is the number of the statement. We designed two statement for the testing, statement ① and ②. The values in the last 2 columns are the number of the record that the system returned to the user. “NO RLC” means the normal operation without Row Level Control. “RLC” means the result with Row Level Control.

Conclusion: For each user, the results with and without RLC are different because with RLC each user can only access the tuples belong to him/her.

Test RC-3: SQL Statement - Update

This test is based on the table that created in the last test. In the test statement below, the ‘X’ and ‘Y’ are variables. We change the value ‘Y’ to return different records and ‘X’ can be any significance value. The null value in column “RCL” means that any users who have the privileges to access this table can access the record. From the result shown in Table 17, the records can be access by the user if the row control level is “NULL”. Obviously User cannot change the value of the tuple if the tuple already be granted to the other user.

SQL Statement: Update HAO.TEST_RC1 Set SAL='X' Where EMPNO='Y';

No.	Current User	X	Y	Result	RCL	Granted User
1	TESTUSER1	1234	9999	✘		ALL
2	TESTUSER2	1234	9999	✓		ALL
3	TESTUSER2	1234	8100	✘	RC1	TESTUSER1
4	TESTUSER2	1234	7100	✓	RC2	TESTUSER2
5	TESTUSER3	2222	3333	✓	RC3	TESTUSER3
6	TESTUSER3	2222	7100	✘	RC4	TESTUSER4
7	TESTUSER3	2222	9000	✓		ALL
8	TESTUSER4	1111	4010	✓	RC4	TESTUSER4
9	TESTUSER4	1234	8000	✘	RC1	TESTUSER1

No.	Current User	X	Y	Result	RCL	Granted User
10	TESTUSER4	2345	7000	✘	RC2	TESTUSER2
11	TESTUSER4	3456	9800	✓		ALL
12	HAO	9999	3000	✘	RC3	TESTUSER3

✓: Operation succeed ✘: Operation failed

Table 17. “UPDATE” Test of RC in Table “TEST_RC1”

From the result listed in Table4-15, we can get these conclusions:

1. User who has the “Update” privilege can update the data if the RCL is “NULL”
2. User who has the “Update” privilege can update the data if the RCL belong to this user
3. Other “Update” operations are not allowed

4.3. Performance Test

Performance tests focus on both the space complexity and time complexity. It must be remembered that encryption has a high cost in over-head due to the processing power needed to execute the complex encryption/decryption algorithms [6]. Normally, the cipher text is longer than the original message. Then replacing the original message by cipher text occupies more space for storing. Time complexity is as same as the space complexity. Encoding and decoding take long time during the query operation. Depending on different encryption methods, this extra cost can be calculated and compared each other.

Although Field Encryption and Row Level Control imply more space and time needed, both of these functions are simply used for some specific tables. For example, there are 50 tables in user's database and maybe only 3 or 4 tables need to be encrypted or row level controlled. Users don't need to encrypt all the attributes such as "USER_NAME", "PRODUCT_ID", and "CAR_LENGTH".

The testing hardware environment is in an IBM computer with Pentium II Celeron 650MHz processor, and 192MB RAM. The operating system is Microsoft Windows 2000 Professional + SP2.

4.3.1. Space Complexity Analysis

Field Encryption:

In Field Encryption, the size of cipher text is equal to or greater than the plain text. Different encryption methods may have different results.

MD5:

The algorithm takes as input a message of arbitrary length and produces as output a 128-bit "fingerprint" or "message digest" of the input. It is conjectured that it is computationally infeasible to produce two messages having the same message digest, or to produce any message having a given pre-specified target message digest. The length of the cipher text of MD5 encryption is 32 words.

DES:

From the algorithm of DES, each round takes as input the 64-bit output of the previous round, and the 48-bit per-round key, and produces a 64-bit output. After the 16th round, the 64-bit output is subjected to another permutation, which happens to be the inverse of the initial permutation. Suppose the length of the plain text is Pl . From the algorithm of the DES, the space complexity is:

$$Cl(Pl) = (Pl \bmod 8) \times 16 + 16$$

Cl : length of cipher text

Pl : length of plain text

That's mean if the plain text increase by 8; the cipher text should increase 16.

RSA:

In RSA, the size of the key is variable. Because this is a test program, we already set the bit length to 64 so the length of the keys and N are almost constant. The length of the key1 is around 20. The length of key2 (factor N) is 39 and the length of key3 is 38. In real application, the length of the key should be more then 1000 to make sure that it is secured enough. The block size of the cipher text is the size of the key and the block size of the plain text must be smaller then the key length. From the algorithm of RSA the space complexity in this program is:

$$Cl(Pl) = ((Pl + 7 + 3) / 4) \times \text{Length}(N)$$

Which Pl is the length of plain text and $(Pl+7+3)/4$ is the number of the blocks. $\text{Length}(N)$ is the length of the factor N . In our project, the size of factor N is 39. Since the length of some blocks is 38, then the final result is not exactly the times of 39. Let B_n is

the number of the blocks and B_s is the size of each block. The range of the length of cipher text C_l is

$$(B_n \times B_s - B_n) < C_l \leq (B_n \times B_s)$$

Figure 12 shows the space cost for each encryption methods. The category of X-axis is the length of the message. The category of Y-axis is the length of the cipher text. From this chart, the RSA is not good for the purpose of this project because it takes much more space then other encryption methods.

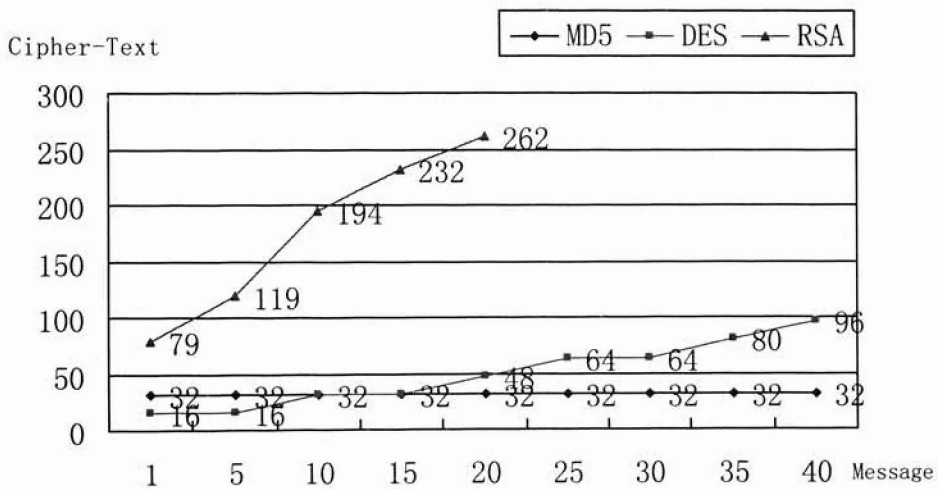


Figure 12. Space Cost of the Encryption Methods

Row Level Control:

The space cost for the Row Level Control is the size of new column added to the table. The data type of the column is VARCHAR2(22). Then the space complexity is $O(N)$ where N is the number of rows in a table. Symbol n is the number of the rows in the table.

4.3.2. Time Complexity

Each time the database management system encoding or decoding the columns in Field Encryption requires additional time to process. Also, in Row Level Control, system need time to check the RLC status and process the user's queries. Depending on different query procedures, the cost time may be variable.

Field Encryption:

The running time for this function is the time to analyze the user input plus the time of the encoding and decoding. The time of the analysis can be considered as a constant and this operation only need to do once. Then the most of the time is used for encoding and decoding.

If we retrieve the data by decoding the entire column, it will take a long time. The time cost for encoding/decoding a whole column is shown in Table 18. The length of the test string is 31 and the number of the test rows is 100, 200, 500, 1000, and 1500. In the result of DES and RSA, the first value is for encoding only and the second value is the result for both of the encoding and decoding operations. The unit of the time is second.

Rows	100		200		500		1000		1500	
Time	E	A	E	A	E	A	E	A	E	A
MD5	0		1		3		6		9	
DES	2	3	3	6	7	13	14	27	20	38
RSA	16	31	31	60	78	155	157	314	236	473

E: Encoding A: Encoding and Decoding Unit: second

Table 18. The Running Time of Each Encryption Method

Obviously, encoding/decoding entire column is not good for large table because encoding and decoding are overhead. It is can only be used for small tables. System can analyze the queries and change them. The time complexity for these operations is constant ($O(1)$).

Row Level Control:

The additional time cost for searching with Row Level Control is $O(N)$ where N is the number of rows in a table. The time required for each response to the user is the time to analyze and rewrite the quires.

4.4. Security Problems

In Field Encryption, the keys used for decoding are not stored in the database and this implementation. This design insures that no one can obtain the keys from the database. Adding Field Encryption to the database does not increase the risk of security. Actually, it reduces the risk to lose the data.

As same as Field Encryption, because Row Level Control does not change the logic structure of DBMS, adding this function to the database will not give the breaker more chances to break in the system. This new function is not designed to against the people who want to break in the database. It protects the data for end-users only.

CHAPTER 5

CONCLUSIONS

5.1. Conclusions

From results of tests, the author provides users with an enhanced data security for Oracle database by Field Encryption and Row Level Control. Test results show that the technology is feasible and correct.

For Field Encryption, users who want to get the data should provide not only the key for identification, but also the key for encryption. They would not lose their privacy even if the database is hacked. No one, including the database administrator, can read the data directly. For the row control, different users can access the same database, yet obtain only a specific set of records authorized to use.

Some disadvantages are obvious. Additional space is required for field encryption and row level control. Users' queries have to be pre-processed before sending to the database. It requires extra execution time.

5.2. Future Work

Making this program compatible with other databases is the suggested future work. Because PL/SQL is supported by Oracle only, the best way to apply these functions to other databases is modulating the program as Java classes and methods. Currently, most of the database management systems can support Java language and import Java class.

REFERENCES

1. Anstey, A. David. High Performance Oracle8 Object-Oriented Design, The Coriolis Group, Inc, 1998.
2. comp.lang.java. FAQ From: elharo@sunsite.unc.edu, From Newsgroups: "comp.lang.java.programmer", "alt.www.hotjava", Cafe Au Lait, 1997.
3. CyPost Corporation, The New Breed of Privacy and Security Applications for Document Storage & Electronic Communications, An Overview of Digital Cryptography, 1999
4. Den Boer, B. Bosselaers, A. An Attack on the Last Two Rounds of MD4, Advances in Cryptology - Crypto '91, Springer-Verlag, 194-203, 1992.
5. Den Boer, B. Bosselaers, A. Collisions for the Compression Function of MD5, Advances in Cryptology - Eurocrypt '93, Springer-Verlag, 293-304, 1994.
6. Detailed JCE-Provider, Information and Performance Measurement Results, From WWW at "http://www.nue.et-inf.uni-siegen.de/SignStreams/csp/detailed/detailed_JCEInfos.html", 2000.
7. Dobbertin, H. Alf Swindles Ann. RSA Laboratories CryptoBytes, vol.1, pp.4-5, 1995.

8. Dobbertin, H. The Status of MD5 After a Recent Attack, RSA Laboratories
CryptoBytes, vol.2, pp.1-6, 1996.
9. Kaliski, B.S. RFC 1319: The MD2 Message-Digest Algorithm, RSA Laboratories,
1992.
10. Lewis, Morris. Creating a Manageable Security Plan, From the WWW at URL:
“<http://www.sqlmag.com/Articles/Index.cfm?ArticleID=15446>”, 1999.
11. Lulushi, Albert. Oracle[®] Developer/2000[™] Forms – The Practitioner’s Guide,
Prentice Hall PTR, 1999.
12. Microsoft Corporation. Microsoft Security Bulletin (MS00-035): Frequently Asked
Questions, From the WWW at URL: “[http://www.microsoft.com/technet
/security/bulletin/fq00-035.asp](http://www.microsoft.com/technet/security/bulletin/fq00-035.asp)”, 2000.
13. Northcutt, Stephen. Information Assurance Foundations. Core Issues and
Challenges, SANS GIAC LevelOne Seminar PDF, Version 1.3, 2000.
14. Oracle Corporation. Oracle Forms Release6i Technical Information, From WWW
at “<http://otn.oracle.com/products/forms/content.html>”, 1999.
15. Oracle Corporation. (1997) Oracle8i Standard Edition, From the WWW at
“<http://www.oracle.com/ip/dep/otn/database/8i/index.html?std.html>”, 1997.
16. Oracle Tech Net. (1998) Oracle8i Standard Edition, From the WWW at:
“http://technet.oracle.com/docs/products/oracle8/doc_index.htm”, 1998.

17. Rivest, R.L. RFC 1320: The MD4 Message-Digest Algorithm, Network Working Group, 1992.
18. Rivest, R.L. RFC 1321: The MD5 Message-Digest Algorithm, Internet Activities Board, 1992.
19. Rivest, R.L. The MD4 Message Digest Algorithm, Advances in Cryptology - Crypto '90, Springer-Verlag , 303-311, 1991.
20. Rivest, R.L. Shamir, A. Adleman, L.M. A Method for Obtaining Digital Signatures and Public-key Cryptosystems, Communications of the ACM 21, 120-126, 1978.
21. Robshaw, M.J.B. On Recent Results for MD2, MD4 and MD5, RSA Laboratories Bulletin 4, 1996.
22. Rogier, N. Chauvaud, P. The Compression Function of MD2 is not Collision Free, Selected Areas in Cryptography '95, Ottawa, Canada, 1995.
23. RSA Laboratories. FAQ about Today's Cryptgraph 4.1, DES Section: What is DES, From the WWW at "<http://www.rsasecurity.com/rsalabs/faq/3-2-1.html>", RSA, 2002.
24. RSA Laboratories FAQ about Today's Cryptgraph 4.1, RSA Section: What is RSA, From the WWW at "<http://www.rsasecurity.com/rsalabs/faq/3-1-1.html>", RSA, 2002.

25. RSA Laboratories. What is a hash function?, RSA Laboratories FAQ, From WWW at "<http://www.rsasecurity.com/rsalabs/faq/2-1-6.html>", RSA, 1999.
26. Trinanes, Joaquin A. Database Security in High Risk Environments. From WWW at "http://rr.sans.org/appsec/db_sec.php", 2001.

APPENDIXES

APPENDIX-A

A TABLE OF MAJOR SUBPROGRAMS USED IN THE PROGRAM

Here is the list of the major subprograms used in this application. This list only contains the subprograms or elements that relate to the goal of this project directly. Others like the modules for interface and outputs are not included in this list.

Category	Name	Location	Comments
Tables	DS_FieldEncryption	DB	Used to save all the encryption information like the method, private key, etc.
	DS_RowLevelControl	DB	Used to store the information for Row Control
	DS_EncMethod	DB	The list of encryption methods
Procedures	CheckUsers	App	Authorization and connection check
	CheckTables	App	Check all the tables will be used in the program. If not exist, create it.
	Check_Files	App	Check all the files for encryption methods
	Restore_SubPrograms	App	Restore the procedures, triggers, functions that need to be store in the database
	DS_AlterTables	DB	Alter the table for Row Level Control
	DS_RequireKey	DB	Require the private key from the user

Category	Name	Location	Comments
	DS_RedoSelectFE	DB	New SELECT if the user access some fields that had been encrypted
	DS_RedoSelectRC	DB	New SELECT if the user access some rows not granted to him/her
Functions	DS_Encrypt	DB	Encrypt the field, return the status of this operation
	DS_UnEncrypt	DB	Un-encrypt the field, return the status of this operation
	DS_RC_ADD	DB	Add control level
	DS_RC_DEL	DB	Delete the control level
Sequences	DS_AutoID	DB	Provide an auto ID number for the attribute "rec_id" of the tables

*: "DS" is the shorthand of "Data Security"

** : "APP" means the application

***: "DB" means the database

APPENDIX B

ENCRYPTION METHOD TEST SUITES

To test the correctness of the encryption algorithm, we set test suites to check it and the result is shown below.

MD5 test suite:

Message	Cipher Text
	d41d8cd98f00b204e9800998ecf8427e
a	0cc175b9c0f1b6a831c399e269772661
abc	900150983cd24fb0d6963f7d28e17f72
message digest	f96b697d7cb7938d525a2f31aaf161d0
abcdefghijklmnopqrstuvwxyz	c3fed3d76192e4007dfb496cca67e13b
ABCDEFGHIJKLMNOPQRSTUVWXYZabcd efghijklmnopqrstuvwxyz0123456789	d174ab98d277d9f5a5611c2c9f419d9f
12345678901234567890123456789012345678 90123456789012345678901234567890123456 7890	57edf4a22be3c955ac49da2e2107b67a

DES test suite (key=12345678):

Message	Cipher Text
a	15d69826f6027d8b
abc	1a78fb115b4f0b94
message digest	541eb60650bb63ffec5ed7ce36b24e8
abcdefghijklmnopqrstuvwxy	15139e949228b146c4e5462b7bfd27e9b2fb5db6 6974ff6c8c8be40b06595f10
ABCDEFGHIJKLMN OPQRSTUVWXYZ WXYZabcdefghijklmnop rstuvwxy 0123456789	6b1370aa34e3076a5b1e6fc508fd9ede179f9919e 4ae31fce3a81bab1b8c5ec267c86cf097b0888212 e6e2a383eeb845dbbda1543ddfd7982726de8e35 bd1844
123456789012345678901234567890 123456789012345678901234567890 12345678901234567890	77db4f3383300c7b488232b72665b60f57894fe0 a8da21d380ddf7b61db8c1f4689c07dd379ef8a17 7db4f3383300c7b488232b72665b60f57894fe0a 8da21d380ddf7b61db8c1f4689c07dd379ef8a1de ff9285a09c383b

RSA test suite:

Key1= 15178972973989135559

Key2(N)= 236894149634191864863633893835838147399

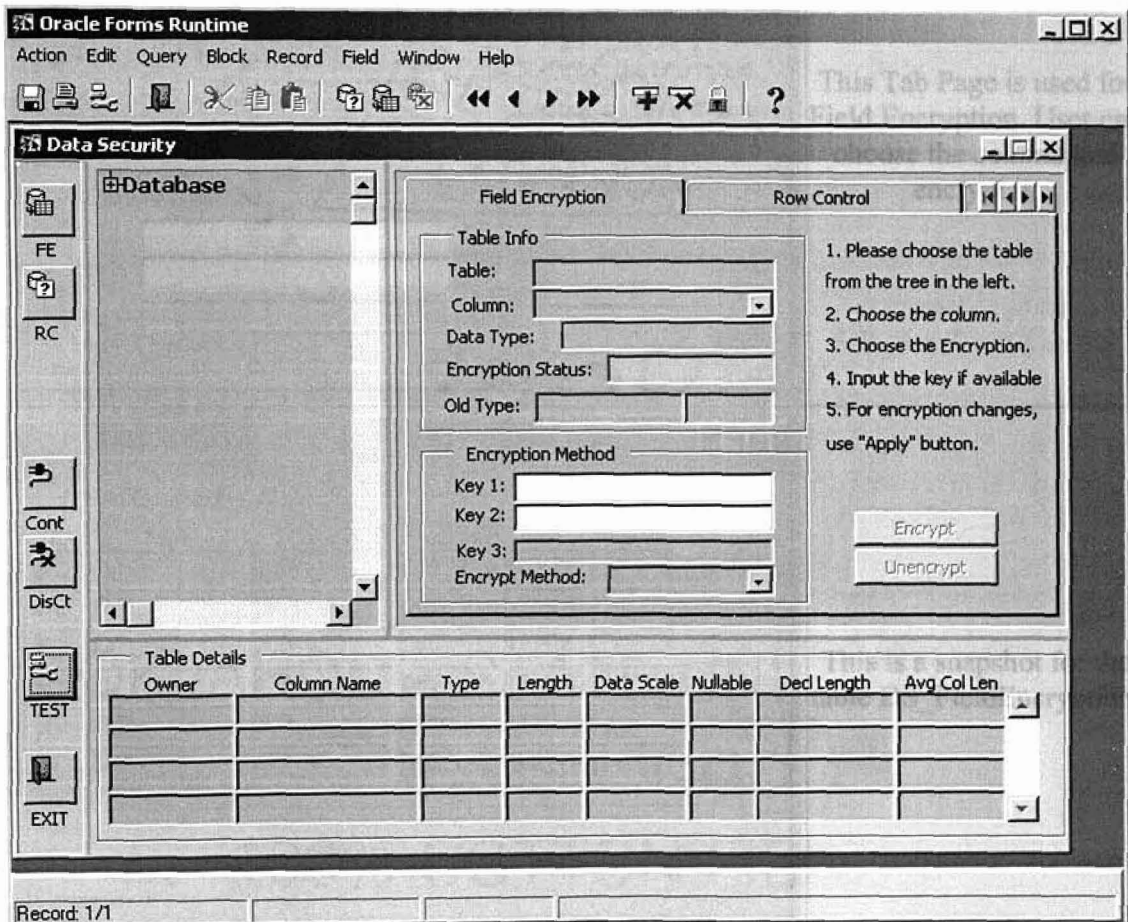
Message	Cipher Text
a	10365217734591472119427179898581744717316833 8408671897204617555919518308721281
abc	10365217734591472119427179898581744717317427 98025660524900607818684034580774792186537779 15502999198057919569472977904
message digest	10365217734591472119427179898581744717361641 02942847064105938591537109081369512572446605 11374918672840698055801549271369420986329495 29850254242103597471649187508308045636852489 86592273494022349417726350027186973769803184 1441659355892
abcdefghijklmnopqrstuvwxy	10365217734591472119427179898581744717344433 58084533067622664800872531552914519391030281 80483654554636854051660605761573756559179691 02755989294304527863434228110363823396190729 18441080929665175514244446458588561857199715 07089308931822873640636311792842729881075488 85099382358739797661971512798553227710182992

Message	Cipher Text
	76138255599804880847585915218633178024061

APPENDIX C

A USER'S GUIDE

Main Program:



Main Program

User can choose the functions from the tools bar in the left by clicking the button.

The details for each part are described in Chapter-3.

All the function operations, for example, add and remove FE to a table, are grouped by different tab pages and listed in function area.

The tab pages for each function are listed below.

Field Encryption

Field Encryption Table

Table Info

Table: EMP3

Column: DEPTNO

Data Type: NUMBER

Encryption Status: Not Encrypted

Old Type:

Encryption Method

Key 1:

Key 2:

Key 3:

Encrypt Method:

1. Please choose the table from the tree in the left.

2. Choose the column.

3. Choose the Encryption.

4. Input the key if available

5. For encryption changes, use "Apply" button.

Encrypt

Unencrypt

This Tab Page is used for Field Encryption. User can choose the column and encrypt it.

Field Encryption

Field Encryption Table

Field Encryption Table

ID	Table	Column	Encryption	Key1	Key2
1	TEMP	JOB	DES	1234567	
4	EMP	EMPNO	DES	1234567	
6	EMP_HAO_EN	JOB	DES	1234567	
3	TEMP_HAO	JOB	DES	1234567	
2	EMP	JOB	MDS	1234567	
5	EMP	DEPTNO	DES	1234567	

Refresh

This is a snapshot for the table DS_FieldEncryption.

Row Control Row Control Table Details

Table: Available Levels: ▲
 Owner: ▼ RLC Status:

Available Control Level

Available Users:	RC1	RC2	RC3	RC4	<NULL>
<input type="checkbox"/> TESTUSER1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> TESTUSER2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> TESTUSER3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> TESTUSER4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> TESTUSER4_2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

User:
 New Level:

This is used for function Row Level Control.

Row Control Row Control Table Details

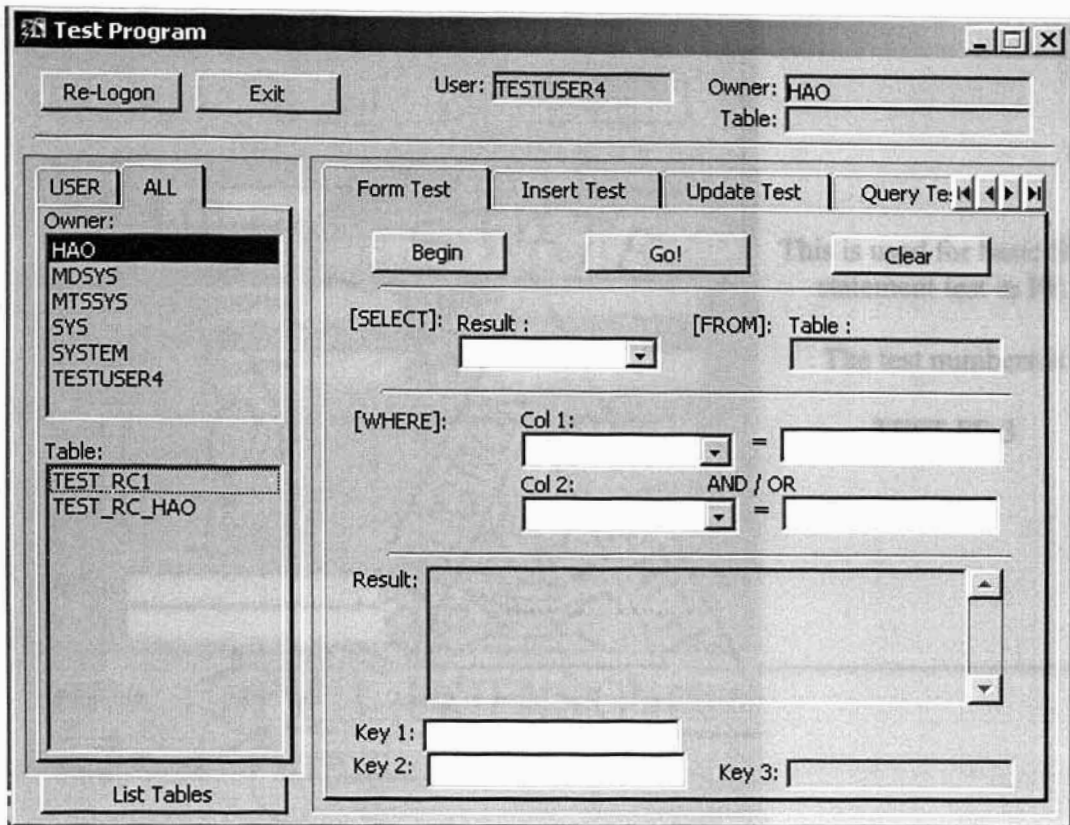
Row Level Control

Rec Id	Table Name	Owner	Access Level	Access User	Access Group
5	TEST_RC_F	HAO	u1	TESTUSER1	<input type="text"/>
7	TEST_RC_F	HAO	uuu2	TESTUSER2	<input type="text"/>
1	TEST_RC1	HAO	RC1	TESTUSER1	<input type="text"/>
2	TEST_RC1	HAO	RC2	TESTUSER2	<input type="text"/>
3	TEST_RC1	HAO	RC3	TESTUSER3	<input type="text"/>
1	TEST_RC1	HAO	RC4	TESTUSER4	<input type="text"/>

A snapshot of table DS_RowLevelControl

These 4 tab pages can do all the operations for FE and RC. It gives users a friendly interface. Next image is the snapshot of the test program.

Test Program:



Test Program

Tester can login the database by different username. The tables are listed in the left part. Tester can choose the tab pages for different test purpose.

The tab pages in test program are list below.

Form Test | Insert Test | Update Test | Query Test: << >> >>>

Begin | Go! | Clear

[SELECT]: Result : [FROM]: Table :

COMM [DEPTNO] TEST_RC1

[WHERE]: Col 1: DEPTNO = 10

Col 2: AND / OR

ENAME =

Result:

Key 1: Key 2: Key 3:

This is used for basic SQL statement test in FE.

The test numbers is
TEST FE-3

Form Test | Insert Test | Update Test | Query Test: << >> >>>

Begin | INSERT | Clear

JOB	VARCHAR2	<NULL>
MGR	NUMBER	<NULL>
HIREDATE	DATE	<NULL>
SAL	NUMBER	<NULL>
COMM	NUMBER	<NULL>
DEPTNO	NUMBER	<NULL>
EMPNO	NUMBER	<NULL>
ENAME	VARCHAR2	<NULL>

New Value: ADD TO LIST

Key1: Key2: Key3:

This is used for basic SQL statement test in FE.

The test numbers is
TEST FE-4

Form Test | Insert Test | Update Test | Query Test

Begin UPDATE Clear

[FROM]: Table: EMP3

[WHERE]: Column1: DEPTNO =

[VALUE]:

JOB	VARCHAR2	<NULL>
MGR	NUMBER	<NULL>
HIREDATE	DATE	<NULL>
SAL	NUMBER	<NULL>
COMM	NUMBER	<NULL>
DEPTNO	NUMBER	<NULL>
EMPNO	NUMBER	<NULL>
ENAME	VARCHAR2	<NULL>

New Value: Add To List

KEY1: KEY2: KEY3:

This is used for basic SQL statement test in FE.

The test numbers is

TEST FE-5

Update Test | Query Test | RC Select Test | RC Maint Test

Begin RLC Result No RLC Result Clear

[SELECT]: DEPTNO [FROM]: TEST_RC1

[WHERE]:

Column 1: COMM =

Column 2: ENAME =

Total Record: Record Found:

This is used for basic SQL statement test in RC.

The test numbers is

TEST RC-2

Query Test | RC Select Test | RC Maint Test | RC Maint

BEGIN Insert CLEAR

RLC Status: RLC RCL: /

DS_RC_RCL	VARCHAR2	<NULL>
EMPNO	NUMBER	<NULL>
ENAME	VARCHAR2	<NULL>
JOB	VARCHAR2	<NULL>
MGR	NUMBER	<NULL>
HIREDATE	DATE	<NULL>
SAL	NUMBER	<NULL>
COMM	NUMBER	<NULL>
DEPTNO	NUMBER	<NULL>

ADD/UPDATE =>

This is used for basic SQL statement test in RC.

The test numbers is

TEST RC-3

RC Select Test | RC Maint Test | RC Maint 2 | Performan

Begin Go Clear

RLC: RLC Level:

[White Input Area]

[Grey Output Area]

This is used for basic SQL statement test in RC. User input SQL statement in the white area (input) and system returns the result in grey area (output).

RC Maint Test | RC Maint 2 | Performance

Input:

Key 1: Records#: 1000

Key 2: Encryption: MD5

Key 3: ENCRYPT ONLY

Cipher Text:

Result:

Time (s):

Length:

Input:

Output:

This is the performance test page. User can test the space and time cost here.

VITA 2

Hao Pang

Candidate for the Degree of

Master of Science

Thesis: AN ENHANCEMENT OF DATA SECURITY FOR ORACLE DATABASE

Major Field: Computer Science

Biographical:

Personal Data: Born in Tianjin, China on March 29, 1975, the son of Bingqian Pang and Fengying Xie.

Education: Graduated from Nanjing No.7 High School, Nanjing, China in July 1993; received Bachelor of Science degree in Computer Science from Nanjing University of Science and Technology, Nanjing, China in July 1993. Completed the requirements for the Master of Science degree with a major in Computer Science at Oklahoma State University in August, 2002.

Experience: Employed by JiangSu SinKou High Tech. Inc. as a network manager; employed by JiangSu YiYou Inc. as a C programmer.