International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 7, July - 2013

Fraud Detection And Integrity Of Database Without Using Pl-SQL Logic

Rohit Miri
Head, Dept of CSE

Pushpa Sharma
M.Tech. C.S.E. Scholar
Bilaspur, India

S. R.Tandan
Asst. Professor
Department of Computer Science & Engineering
Dr. C.V. Raman Institute of Science & Technology
Bilaspur, India

Abstract

Database Administrators *are the personnel responsible for the overseeing, management and physical design of the database in an organization. Their duties further include the evaluation, selection and implementation of a Database Management System. Choosing the most suitable Database Administrator for the respective company is of vital importance, in order to maximize control opportunities and minimize control problems for the business. We can also check the credibility of employee with the help of virtual table.*

Introduction

The Sensible companies use internal controls as a means of regulating their own information. Whether for security reasons or to ensure legal, accurate and reliable accounting data and records, these internal control encompasses the overall policies and processes of the business. Therefore, the role of a Database Administrator can cause a company serious security control problems. As the Database Administrator generally has top-level access to the database and the system, a malicious Database Administrator would be able to steal or sabotage important files or records.

Here we can also check the credibility of employers with the help of virtual table. Employee is not aware of database that how many tables are there in, and how the operations are going on.

Methodology

Suppose we have maintained a database of our software firm or medical store or any other small shop. We have our software for maintaining the product or item of our company or shop. We hire many employees for maintain the software. How we can check the credibility of our employers without using the PL-SQL logic. In Pl-SQL we can perform many security issue related programming.

See Figure no 1.1 indicates the schema diagram of software company. Where tables have created for maintaining the company of employers and products. i.e Employee details – personnal information of employees are there, project details- project information of employees are there, Employee_Skill_Details – academic skill of employees are there etc.

Figure 1.1 shows the schema diagram Employees.

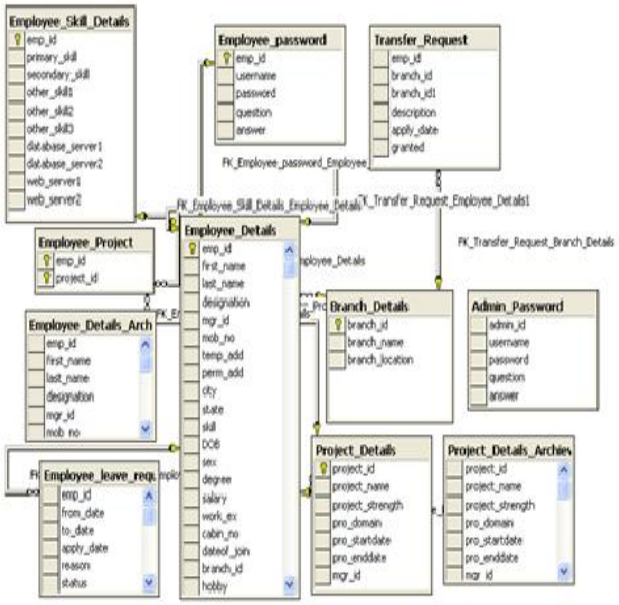

Figure 1.1 ER Diagram of Employee

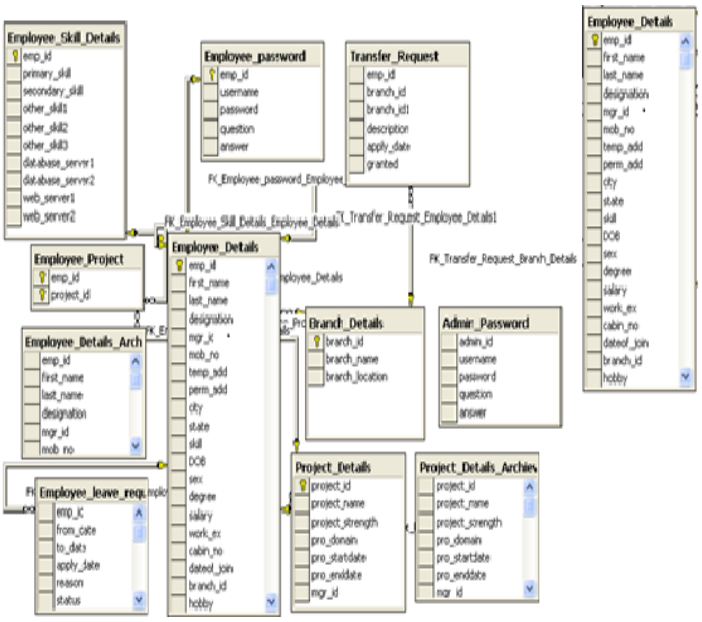

Figure 1.2 ER Diagram of Employee

Now make some modification on Figure 1.1.make duplicate table of employee Details to check the chance of fraud data in this table. We can make duplicate of any table or all tables.

How we can check the how much integrity is maintain in the table employee details.

Primary table (First Table) is a table that contains the first entries . its contents are not changeable when we perform the modification, deletion later on. It's a actual content.

Secondary table (Second Table) is a table that contains also the first entries but it contents are changeable when we perform the modification or deletion operations.

In this below diagram Figure no 1.2 the First table (Employee_ Details_Archeive) is not make any relationship with any table. It is a alone. But the second table (Employee_Details) that makes the relationship (Primary and foreign key relationship) with many of the tables. Field and structure of both the tables are identical.

Now see how can check the chance of fraud data with the help of both the tables.

1. If we are inserting the values in Employee_Details_archive table (First Table) , these values of first table is also inserted to the second table at same time. It means the table 1 is a back up table which contains the first inserted values. Later on some values may be changes with the updation and deletion operations on table Employee_Details. We can find the chance of fraud data after comparing the data values between the first and second tables.

2. After comparing both of the tables data, we can retrieve that rows that are not matching.

3. If the number of rows are not matching in both of the tables. It means there is a chance of some product of our company may be stolen or deleted without our knowledge. i.e see the Figure no 1.3 and 1.4 where the no of rows are not matching.

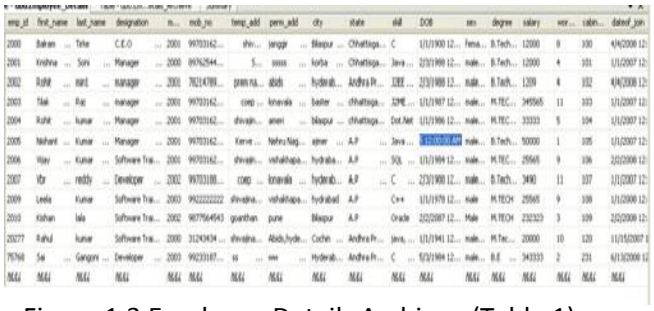

Figure 1.3 Employee Details Archieve (Table 1)

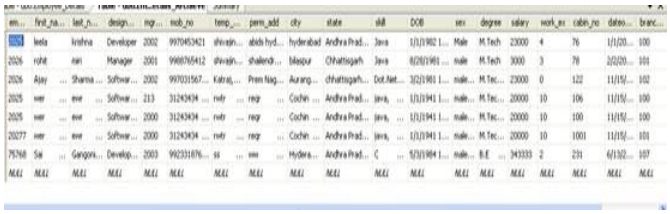

Figure 1.4 Employee Details (Table 2)

4. We can also conform by comparing the each field of both the tables. If some field of both the table is not matching .i.e see the Figure no 1.3 and Figure 1.4 . if we compare field values of both the tables there is chance of illegal modification of data values. Total quantity of item or product may be less in table 2(Employee details) as compared to table 1. It means our item is stolen or missing.

IMPLEMENTATION

How we can perform this logic in an advanced java programming Language. We have written the insert code for both the tables for this logic. I have not taken all the figure of mentioned table. For sake of simplicity I have taken only 5 fields from both the tables.

```
import java.io.*;

import java.sql.*;

import javax.servlet.*;

import javax.servlet.http.*;

public class AddEmployee_Details extends HttpServlet{

 public void init(ServletConFigure conFigure) throws ServletException{

  super.init(conFigure);

 }

 /**Process the HTTP Get request*/

 private static final long serialVersionUID = 1L;

 public void doGet(HttpServletRequest req, HttpServletResponse res)
throws ServletException, IOException {

/**Employee_Details=(emp_id, first_name,last_name, designation, mgr_id
)*/

        Connection connection = null;

        RequestDispatcher dispatch = null;

        res.setContentType("text/html");

        PrintWriter out = res.getWriter();

        //get the variables entered in the form

        String emp_id1 = req.getParameter("emp_id");

        String first_name1 = req.getParameter("first_name");

        String last_name1 = req.getParameter("last_name");

        String designation1 = req.getParameter("designation");

        String mgr_id1 = req.getParameter("mgr_id");

        int emp_id2 = Integer.parseInt(emp_id1);

        int mgr_id2 = Integer.parseInt(mgr_id1);

    try {

// Load the database driver
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
                // Get a Connection to the database connection =
DriverManager.getConnection("jdbc:odbc:rohits");          //Add
the data into the database

String sql = "insert into Employee_Details values(?,?,?,?,?)";

PreparedStatement pst = connection.prepareStatement(sql);



        pst.setInt(1, emp_id2);

        System.out.println(emp_id2);

        pst.setString(2, first_name1);

        pst.setString(3, last_name1);

         pst.setString(4, designation1);

        pst.setInt(5, mgr_id2);

        pst.executeUpdate();

String sql1 = "insert into Employee_Details_Archieve values(?,?,?,?,?)";

PreparedStatement pst1 = connection.prepareStatement(sql1);
```

```
            pst1.setInt(1, emp_id2);
            pst1.setString(2, first_name1);
            pst1.setString(3, last_name1);
            pst1.setString(4, designation1);
            pst1.setInt(5, mgr_id2);
            pst1.executeUpdate();
            pst1.close();
            pst.close();
            out.println(" Hello : ");
      req.setAttribute("message", "Employee is added successfully.");
dispatch = getServletConFigure().getServletContext().
getRequestDispatcher ("/jsp/admin_main_page.jsp");
                        } catch (ClassNotFoundException e) {
                                    out.println("Couldn't load database
driver: " + e.getMessage());
                        } catch (SQLException e) {
                                    if (e.getErrorCode()==547){
req.setAttribute("message", "Manager Id  doesnot exists.");
 } else if (e.getErrorCode()==2627){
req.setAttribute("message", "Employee Id  is already exists.");
} else {
req.setAttribute("message", "Employee Id is already exists.");
        }
dispatch = getServletConFigure().getServletContext().
getRequestDispatcher("/jsp/addemployee.jsp");
out.println("SQLException caught: " + e.getMessage());
} catch (Exception e) {
out.println(e);
} finally {
 // Always close the database connection.
            try {
                        if (connection != null)
                                    connection.close();
            } catch (SQLException ignored) {
                        out.println(ignored);
                        }
                        }
                        dispatch.forward(req, res);
                  }
}
```

We know that the operator is unaware of coding of software. The above coding shows the how we insert the first values in both the tables.i.e

String sql1 = "insert into Employee_Details_Archieve values (?,?,?,?,?)";

String sql = "insert into Employee_Details values(?,?,?,?,?)";

First insert Query inserts the values in first table. Whose values is not changeable. we treat this table as a primary table or back up table.

Second insert Query inserts the values in second table. whose values may be changed on performing updation and deletion operations.

The schema diagram may be of any small shop. Where a single person can maintain the software. So there is chance of deleting the record of some item without our knowledge or they can illegal modified some item details. For example he can delete some product record; he can modified the total quantity of item for their illegal income. The logic behind this paper is very useful for handing these type of fraud. This concept may be very useful in many areas. Computer Operator is unaware of coding of software.

Conclusion

We can perform database security with the help of virtual (backup) table.

This paper also performs the security issue of database without using PL-SQL.

This paper is very useful in stock market to check the credibility of our employee that is performing some operation on our company software.

In this paper we can also check the integrity of our database

REFERENCES

1. Walton, G.N. AIRNET – A Computer Program for Building Airflow Network Modeling. National Institute of Standards and Technology. 1989; NISTIR 89-4072.
2. Axley, J. Progress Toward a General Analytical Method for Predicting Indoor Air Pollution in Buildings – Indoor Air Quality Modeling Phase III Report, National Institute of Standards and Technology. 1988; NBSIR 88-3814.
3. Dols, W.S.; Walton, G.N. CONTAMW 2.0 User Manual. National Institute of Standards and Technology. 2002. NISTIR 6921.
4. Persily, A.K.; Ivy, E.M. Input Data for Mulitzone Airflow and IAQ Analysis. National Institute of Standards and Technology. 2001; NISTIR 6585.
5. ASTM. Standard Practice for Full-Scale Chamber Determination of Volatile Organic Emissions from Indoor Materials/Products. D 6670-01. American Society for Testing and Materials. 2001.
6. ASTM. Standard Guide for Small-Scale Environmental Chamber Determinations of Organic Emissions from Indoor Materials/Products. D 5116-97. American Society for Testing and Materials. 1997.
7. European Communities. European Concerted Action Indoor Air Quality & Its Impact on Man (EUR 13593), Guideline for the Characterization of Volatile Organic Compounds Emitted fro
m Indoor Materials and Products Using Small Test Chambers. Report No. 8. COST Project 613. Luxembourg: Office for Publications of the European Communities, 1991.
8. Matthews, T.G. Atmospheric Environment. **1987**, 21, 321 – 329.
9. Zhang, J.S.; Shaw, C.Y.; Sander, D.; Zhu, J.P.; Huang, Y. MEDB-IAQ: A Material Emission Database and Single-Zone IAQ Simulation Program – A Tool for Building Designers, Engineers and Managers. National Research Council Canada. 1999.