Abstract—Ranking is one of the effective techniques to measure and evaluate an individual student’s performance relative to other students. The ranks are calculated based on data collected over several semesters. Ranks can provide feedback to students regarding their academic standing and may be used for nominating students for scholarships and other awards. This paper describes the design and implementation of computerized ranking assessment system which is highly flexible and can easily be integrated with any other learning assessment system. The system designed is web based and can be used to generate ranking trend chart for each student. This chart will highlight the student’s performance over various semesters and will enable students to take remedial actions to improve their performances in future. Test results show that ranking assessment data can be easily generated and delivered through web using the proposed design of the ranking assessment system.

Index Terms—Ranking assessment system, Web, Stored procedure, User-defined function, Chart, Learning assessment system

I. INTRODUCTION

A learning management system (LMS) [1] is a software application for the administration, documentation, tracking, and reporting of training programs, classroom and online events, e-learning programs, and training content. Learning assessment management information system (LAMIS) [2] is one important sub system of LMS. LAMIS provides on-line web-based assessment system to facilitate access to various learning performance related to indicators, such as students’ appraisals [3], transcripts, and skills-gap analysis data [4]. Most LMSs are web-based, developed using a variety of platforms, like Java/J2EE, Microsoft .NET or PHP. They usually employ database such as MySQL, Microsoft SQL Server [5] or Oracle as back-end.

One of the main yardsticks [2] in assessing the students’ learning in school is their rank in class. The trend of the rank in class in the latest semesters provides pointer as to the degree of effort a student puts to improve the performance to become the top student. Critical components of any good students learning assessment management information system are the design of database and its implementation.

The learning assessment system must provide simple mechanism for users to query the data base for extracting student ranking score and generate graphically the ranking trend among the latest several semesters. In this article, we have presented the design of the database using Microsoft SQL Server, to store and retrieve data [6] as requested by other software applications. Microsoft Visual Studio development environment (IDE) [7] is used to develop applications using C# programming language for the learning assessment system.

II. DATABASE DESIGN AND IMPLEMENTATION

Microsoft SQL Server 2000 [5] is used to design the database named StudMISDB, which contains the following two tables:

- TB_SCORE - stores scores for each student in each course on each semester
- TB_SUMSCORE - stores cumulative score for students on each semester.

The designs of these two tables are shown in Table I and Table II.

A. Stored Procedure Design and SQL Server Job Scheduling
Database interaction is done using one T-SQL [5] based stored procedure that aggregates the students score and inserts the aggregation result into the table table TB_SUMSCORE. The stored procedure on the SQL server, is executed within one SQL job [5] (A SQL Server job is a collection of steps executed by SQL Server Agent and it has many different options that can be configured during its creation in order to perform the duties that are needed of it). This SQL job [8] is deployed and it is scheduled for execution automatically every three months on the first Sunday of that month. The score code of the stored procedure proc_aggregate is shown as follows.

```sql
CREATE proc proc_aggregate
as
begin
delete from tb_sumscore
insert into tb_sumscore
select studid,classid,semester,sum(score) from tb_score where
convert(int,substring(semester,1,4))+2>=datepart(yy,getdate())
group by studid,classid,semester
end
```

As shown above, the old aggregated data in the table TB_SUMSCORE is emptied firstly and then the latest data in the table TB_SCORE is aggregated and inserted into the table tb_sumscore. Only the latest 2 years data is aggregated in order to do the assessment meaningfully.

Fig.1 shows the job in the database server. The scheduling of the job is shown in Fig. 2 and Fig. 3.

### B. Design the User-Defined Function and Fetch Clause

The core stored procedure [9] is proc_getrankdata designed to query the student’s rank result. In the key query clause, i.e. the fetch clause, we integrate one user-defined function [5] func_getrank, which accepts the 3 parameters, the student’s ID, the student’s class ID and the named semester, and returns the rank [10] result.

- function func_getrank - returns the value, which is the calculation of the total numbers of the classmates whose total score is greater than the named students, and the stored procedure
- proc_getrankdata - this stored procedure is called to display the rank data in his class in the latest four semesters

The source code for func_getrank is shown as follows.

```sql
CREATE function [dbo].[func_getrank] (@studID varchar(20),@classID varchar(20),@semester varchar(12)) returns int
as
begin
declare @cnt int
declare @sum float
select @sum=sumscore from tb_sumscore where studID=@studID and semester=@semester
if exists(select sumscore from tb_sumscore where semester=@semester and classID=@classID and sumscore>@sum)
begin
select @cnt=count(sumscore) +1 from tb_sumscore where semester=@semester and classid=@classID and sumscore>@sum
end
else
set @cnt=1
return @cnt
end
```

### TABLE I. DESIGN OF THE DATA TABLE :TB_SCORE

<table>
<thead>
<tr>
<th>ColumnName</th>
<th>Data Type</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>scoreID</td>
<td>int</td>
<td>Automatic Increase by 1, ID for table data</td>
</tr>
<tr>
<td>studID</td>
<td>Varchar(30)</td>
<td>Students identity number</td>
</tr>
<tr>
<td>studName</td>
<td>Varchar(30)</td>
<td>Students’ name</td>
</tr>
<tr>
<td>classID</td>
<td>Varchar(30)</td>
<td>Class Identity number</td>
</tr>
<tr>
<td>courseID</td>
<td>Varchar(30)</td>
<td>Courses’ Identity number</td>
</tr>
<tr>
<td>courseName</td>
<td>Varchar(30)</td>
<td>Courses’ name</td>
</tr>
<tr>
<td>Semester</td>
<td>Varchar(30)</td>
<td>Semester</td>
</tr>
<tr>
<td>Score</td>
<td>Float</td>
<td>Score of the course</td>
</tr>
</tbody>
</table>

### TABLE II. DESIGN OF THE DATA TABLE : TB_SUMSCORE

<table>
<thead>
<tr>
<th>ColumnName</th>
<th>Data Type</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>studID</td>
<td>Varchar(30)</td>
<td>Students identity number</td>
</tr>
<tr>
<td>classID</td>
<td>Varchar(30)</td>
<td>Class Identity number</td>
</tr>
<tr>
<td>Semester</td>
<td>Varchar(30)</td>
<td>Semester</td>
</tr>
<tr>
<td>SumScore</td>
<td>Float</td>
<td>Total Score of the student in one semester</td>
</tr>
</tbody>
</table>
With the completion of the user defined function [11], we can call this function in the stored procedure, which provides the fetch service [12] from the database, and the parameters of this function are sent by the table’s fields, which are studID, ClassID and semester, and the return value are correlated to each row of the table. The following is the source code for proc_getrankdata.

```sql
CREATE proc proc_getrankdata
@studID varchar(30)
As
select top 4 semester,dbo.func_getrank(studID,classID,semester) as 'rank' from tb_sumscore where studID=@studID order by semester desc
```

C. Testing the Stored Procedure and User-Defined Function

The system [13] is tested by populating table TB_SCORE with the sample student data [14] shown in the appendix. Both the user defined function [11] and the stored procedure are tested using the Query Analyzer and the test result is shown in Fig.4 and Fig.5.

As can be seen from Fig.5, the student whose student ID is 200910501 ranks third in the third semester of 2009, ranks first in the second semester of 2009 and ranks third in the first semester of 2009; and then we can analyze that this student’s rank in his class is neither in the up mode nor in the down mode. It is in an unsteady wave mode [15].

III. SOFTWARE ARCHITECTURE OF STUDENTS RANKING ASSESSMENT SYSTEM

Dynamic applications are applications that build content “on the fly” in response to request made. They can reflect one named student’s latest trend of rank in class from the database.

Internet Information Services (IIS) [16] – formerly called Internet Information Server – is a web server application [17] and set of feature extension modules created by Microsoft for use with Microsoft Windows. IIS 7.5 supports HTTP, HTTPS, FTP, FTPS, SMTP and NNTP. It is an integral part of Windows Server family of products, as well as certain editions of Windows XP [18], Windows Vista and Windows 7. IIS [16] is not turned on by default when Windows is installed. The configuration of the IIS is shown in Fig.6, which shows one virtual directory--studinfo, whose web files include queryRank.aspx, queryRank.aspx.cs and web.configure and it is started when the operation system is running.

A. Conceptual Software Architecture

The conceptual software architecture [19] is shown in Fig.7 and the architecture contains the following components:

- Database server, which serves the data storage [20], accepts the requests from the web server, sends the responses to the web server, and starts the SQL Server
B. Design of User Interface

The user interface [22] is shown in Fig.8. It contains the following controls:
- One input textbox for inputting Student ID
- One data source control [22] SqlDataSource1 for connecting to database and execute one command of the stored procedure, which accepts one parameter of student ID from the textbox control
- One chart control [22] for displaying rank information graphically with x-axis data is set to semester and y-axis data is set to rank in class.
- One button control [22] SHOW that executes the related stored procedure [9] proc_getrankdata to display the results graphically.

B. Design of User Interface

The user interface [22] is shown in Fig.8. It contains the following controls:
- One input textbox for inputting Student ID
- One data source control [22] SqlDataSource1 for connecting to database and execute one command of the stored procedure, which accepts one parameter of student ID from the textbox control
- One chart control [22] for displaying rank information graphically with x-axis data is set to semester and y-axis data is set to rank in class.
- One button control [22] SHOW that executes the related stored procedure [9] proc_getrankdata to display the results graphically.

IV. TESTING OF THE PROGRAMS

In order to test programs, web server [24] is configured and test results depicting the individual ranking trend charts are shown for two students in Fig.9 and Fig.10. The chart displays the semester on x-axis and student rank on y-axis.

V. CONCLUSION

In this paper, we have discussed the design and implementation of the student ranking assessment [25] management information system. We have proposed a web based solution interacting with Microsoft SQL server database to produce charts depicting students’ rank in their class every semester. A benefit to our approach is that with the help of the chart control, the trend of the students learning results can be shown clearly, and it provides a good feedback to the students, who will take measures to improve their ranks in future.
conceptual design framework can be easily integrated in any other teaching or learning management information systems [26].

Appendix A SAMPLE DATA FOR THE PROGRAM TESTING

Sample data of the table TB_SCORE is shown in the following table III.

<table>
<thead>
<tr>
<th>studID</th>
<th>stud Name</th>
<th>classID</th>
<th>Semester</th>
<th>Course</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>2009105 01</td>
<td>Zhang</td>
<td>cs200901</td>
<td>2009-1</td>
<td>Database</td>
<td>89</td>
</tr>
<tr>
<td>2009105 01</td>
<td>Zhang</td>
<td>cs200901</td>
<td>2009-2</td>
<td>C++</td>
<td>85</td>
</tr>
<tr>
<td>2009105 01</td>
<td>Zhang</td>
<td>cs200901</td>
<td>2009-1</td>
<td>Maths</td>
<td>78</td>
</tr>
<tr>
<td>2009105 01</td>
<td>English</td>
<td>cs200901</td>
<td>2009-1</td>
<td>English</td>
<td>93</td>
</tr>
<tr>
<td>2009105 01</td>
<td>Zhang</td>
<td>cs200901</td>
<td>2009-2</td>
<td>ASP.Net</td>
<td>77</td>
</tr>
<tr>
<td>2009105 01</td>
<td>Zhang</td>
<td>cs200901</td>
<td>2009-2</td>
<td>Algebra</td>
<td>99</td>
</tr>
<tr>
<td>2009105 01</td>
<td>Zhang</td>
<td>cs200901</td>
<td>2009-3</td>
<td>Computer</td>
<td>80</td>
</tr>
<tr>
<td>2009105 02</td>
<td>Li</td>
<td>cs200901</td>
<td>2009-1</td>
<td>Database</td>
<td>82</td>
</tr>
<tr>
<td>2009105 02</td>
<td>Li</td>
<td>cs200901</td>
<td>2009-2</td>
<td>C++</td>
<td>89</td>
</tr>
<tr>
<td>2009105 02</td>
<td>Li</td>
<td>cs200901</td>
<td>2009-1</td>
<td>Maths</td>
<td>97</td>
</tr>
<tr>
<td>2009105 02</td>
<td>Li</td>
<td>cs200901</td>
<td>2009-1</td>
<td>English</td>
<td>90</td>
</tr>
<tr>
<td>2009105 02</td>
<td>Li</td>
<td>cs200901</td>
<td>2009-2</td>
<td>ASP.Net</td>
<td>85</td>
</tr>
<tr>
<td>2009105 02</td>
<td>Li</td>
<td>cs200901</td>
<td>2009-2</td>
<td>Algebra</td>
<td>70</td>
</tr>
<tr>
<td>2009105 02</td>
<td>Li</td>
<td>cs200901</td>
<td>2009-3</td>
<td>Compile</td>
<td>78</td>
</tr>
<tr>
<td>2009105 02</td>
<td>Li</td>
<td>cs200901</td>
<td>2009-3</td>
<td>Computer</td>
<td>99</td>
</tr>
<tr>
<td>2009105 03</td>
<td>Zhao</td>
<td>cs200901</td>
<td>2009-3</td>
<td>Computer</td>
<td>90</td>
</tr>
<tr>
<td>2009105 04</td>
<td>Wang</td>
<td>cs200901</td>
<td>2009-1</td>
<td>Database</td>
<td>99</td>
</tr>
<tr>
<td>2009105 04</td>
<td>Wang</td>
<td>cs200901</td>
<td>2009-2</td>
<td>C++</td>
<td>69</td>
</tr>
<tr>
<td>2009105 04</td>
<td>Wang</td>
<td>cs200901</td>
<td>2009-1</td>
<td>Math</td>
<td>87</td>
</tr>
<tr>
<td>2009105 04</td>
<td>Wang</td>
<td>cs200901</td>
<td>2009-1</td>
<td>English</td>
<td>95</td>
</tr>
<tr>
<td>2009105 04</td>
<td>Wang</td>
<td>cs200901</td>
<td>2009-2</td>
<td>ASP.Net</td>
<td>79</td>
</tr>
<tr>
<td>2009105 04</td>
<td>Wang</td>
<td>cs200901</td>
<td>2009-2</td>
<td>Algebra</td>
<td>97</td>
</tr>
<tr>
<td>2009105 04</td>
<td>Wang</td>
<td>cs200901</td>
<td>2009-3</td>
<td>Computer</td>
<td>90</td>
</tr>
</tbody>
</table>

ACKNOWLEDGMENT

The author, Zhiqiang Yao wishes to thank all the reviewers for their valuable comments and suggestions that improved the quality of this paper; what’s more, he wishes to thank Abhijit Sen for his good advice on updating this paper and for his necessary support in teaching and scientific research. Thanks a lot for his great cooperation in this paper.

This research is supported by the project (No.2011011019) of Langfang Scientific Technology Bureau, Hebei province.
REFERENCES


[22] Geoffrey T Leblond,Michael A Kittel, ASP.Net 2.0 Cookbook, O'Reilly Media , 2005-12-01


Zhiqiang Yao received the 4-year bachelor degree of Mechanical Engineering from Xi’an JiaoTong University in 1994 and learned the 3-year master of Hydraulic Engineering from Tsinghua University in 2000. His major field of study is on computer programming and computer application.He was an MCSD (Microsoft Certified Solution Developer).

He has 4 years experiences of working in IT companies and during that time, he was responsible for developing the e-learning systems. In 2003, he worked at the Research Institute of the Armed Police Force Academy and did several scientific research projects. In 2009, he began to work at Computer Science Department of North China Institute of Aerospace Engineering, and he taught the First Course of Database, ASP.Net Programming, and XML and so on.

Associate Professor Yao published more than 10 papers, 4 of which were indexed by EI.

Abhijit Sen holds a PhD. from McMaster University, Hamilton, Ontario, Canada and Master of Science degree from University of California, Berkeley, USA, B.Tech in Electrical Engineering from Indian Institute of Technology, Kharagpur, India. He has over 25 years of academic and administrative experience at Kwantlen Polytechnic University. He served as a chair of the department for over 14 years. He is currently the Professor of the Computer Science and Information System at Kwantlen Polytechnic University, BC, Canada. Abhijit also was a visiting professor at Waikato University, Hamilton, New Zealand, and Munich University of Applied Sciences, Germany, Centre for Development of Advanced Computing, India, and North China Institute of Aerospace Engineering, China.

His current research interests are in the areas of Wireless Security, Radio Frequency Identification (RFID), Computing Education and Teaching Methodologies, Distributed Systems and Databases.