CS542

Project Report

Backbone Talent Sharing Platform

Team 4

Jiexuan Sun, Congyang Wang, Zhaoning Su, Yiming Zhao

Table of Content

Part I. The Team

1.Team Members and Roles:

Members	Jiexuan Sun	Congyang Wang	Yiming Zhao	Zhaoning Su
Roles	DBA	Graphic Designer	Requirements Analyst	DBA
	Database Designer	Database Designer	Database Designer	Database Designer
	Web Developer	Web Developer	Web Developer	Web Developer
Works	Installs and maintains the database servers;	Be responsible for the site UX design;	Be responsible for translating the business requirements in to a specification for other technicians;	Installs and maintains the database servers;
	Be responsible for documenting the design and the decisions;	Be responsible for documenting the design and the decisions;	Be responsible for documenting the design and the decisions;	Be responsible for documenting the design and the decisions;
	Program the functionality of a website.	Program the functionality of a website.	Program the functionality of a website.	Program the functionality of a website.

Fig. 1. Team 4 Members and Roles

Part II. The Backbone Talent Sharing Platform

1.The System Introduction

1.1 The Scope

Following the success of Uber, Airbnb and other internet-based companies, more and more technicians choose to launch their career from start-ups. But, compared with those big-name companies, many start-ups are still fighting for their future. One of the remarkable feature of those internet start-ups is the high talent mobility. From the employee's perspective, this high mobility means high risk of unemployment. And because of the uncertainty, some of the first-class technicians are not willing to work in the start-ups. So, from the employer's perspective, those startups companies always face the problem of shortage of talent. On the other hand, with limited budget, start-ups usually cannot afford their own human resources department.

But, as all we know talents are the 'soul' of the companies, especially in high-tech industries. And the above two major aspects block some start-ups' development. To solve this problem, our project focus on building a talent sharing platform, named Backbone, serving for both internet start-ups and their technician workers. Besides traditional talent recording and choosing, the Backbone's aim is to solve problems of the start-ups' 'talent crisis' and help their employees' avoid the unemployment risk.

1.2 The Approach and Features

Companies Joined the Backbone platform need to provide their basic information and all their technician employee's personal information with any personal changes for recording. But this information is for companies' internal evaluation. So, it is not visible between companies to prevent 'talent stealing'. And, Backbone will require those companies to sign the NDA (non-disclosure agreement). But, once the people is in unemployment status, his/her information will be visible to all the companies in this platform for hiring. So, once the technician is dismissed or decide to quit the office, his/her performance history in the old company (personal skill score, project grade, training grade and so on) will be visible among the platform. Obviously, technicians with good performance record and high skill score will find a new position quickly. So, even though the start-ups maybe don't success at last, individual's achievements and work will be recorded. In other word, technicians will reduce their unemployment risk by working hard. This will build a virtual circle among this platform. And start-ups' technicians with genuine talent won't afraid the unemployment risk anymore. What's more, company in this platform, by sharing one third party 'human resources' department' – Backbone talent platform, will lower the human resources management cost tremendously.

Now, it's important to set a standard to evaluate employee's performance and skill level among varieties companies to Keep fair and consistent. Without the same standard, it is impossible to make comparisons between technicians and things like malicious bad review will cause serious impact on one's career life. So, each company joined the Backbone platform need to sign the contract to allow the third party to evaluate and quantify the projects/trainings difficulty and individual's performance. Individual will gain relevant skill score while finishing projects/trainings. Because of same evaluation standard, companies could accurately evaluate applicant ability and find the one they need quickly.

Also, the Backbone system will record everyone's career activities, then evaluate and 'gamify' it. People who finish projects or trainings will gain relevant skill score based on their performance. Also, once people achieve some accomplishments, like having the most SQL score this month, they will gain some experience point, which is used to rate the level. The experience point/level is a general evaluation of each employee; and it is based on skill score, work experience, accomplishments and other factors. So, based on this gamification-career record system, individuals could check his/her game 'indicators' and compared with the industry average level. In fact, these benchmarks can help spur people on, as many of us are naturally inclined to work harder when we can set our sights towards specific, attainable, quantified indicator or goals.

2. The System Design:

2.1 Composition and Tools

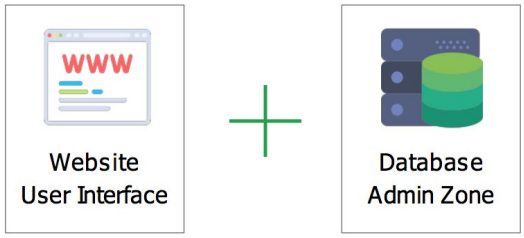

Fig. 2. Backbone Database Design Composition

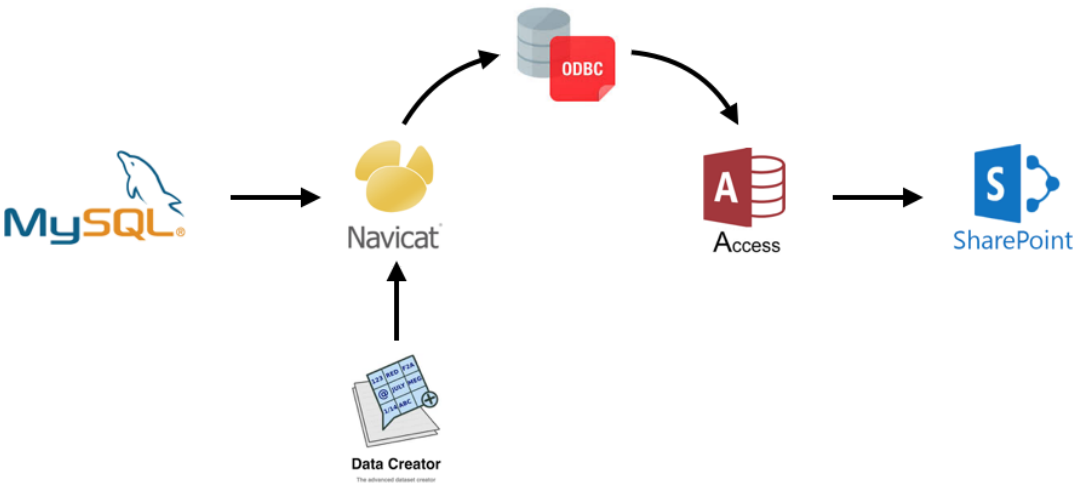

Fig. 3. Backbone Database Design Tools and Workflow

We choose MySQL as our database management system because it is popular and free of charge. Navicat is a graphical database management and development software for MySQL, MariaDB, Oracle, etc. It has a great Explorer-like graphical user interface and supports multiple database connections. What's more, it is cross-platform tool and works on both MS Windows and Mac OS. It helps team members to collaboration easily. We use these two 'tools' to build our database. And we use Data Creator to create our own dataset. ODBC (Open Database Connectivity) is a standard for accessing database management system. We use it to connect our database to MS Access/SharePoint. The latest version of MS Access/SharePoint could quickly build an 'Access' web application with decent user interface. By using these two softwares, we not only achieve our project UI requirement but also save quite a lot of time. Here, we want to state that even though we use the MS Access software to build the so-called 'Access' web application, we only

use it to design the front-end (Website). The DBMS we used is still MySQL by using ODBC, not MS Access Database.

2.2 Entity-Relationship Diagram of the Database

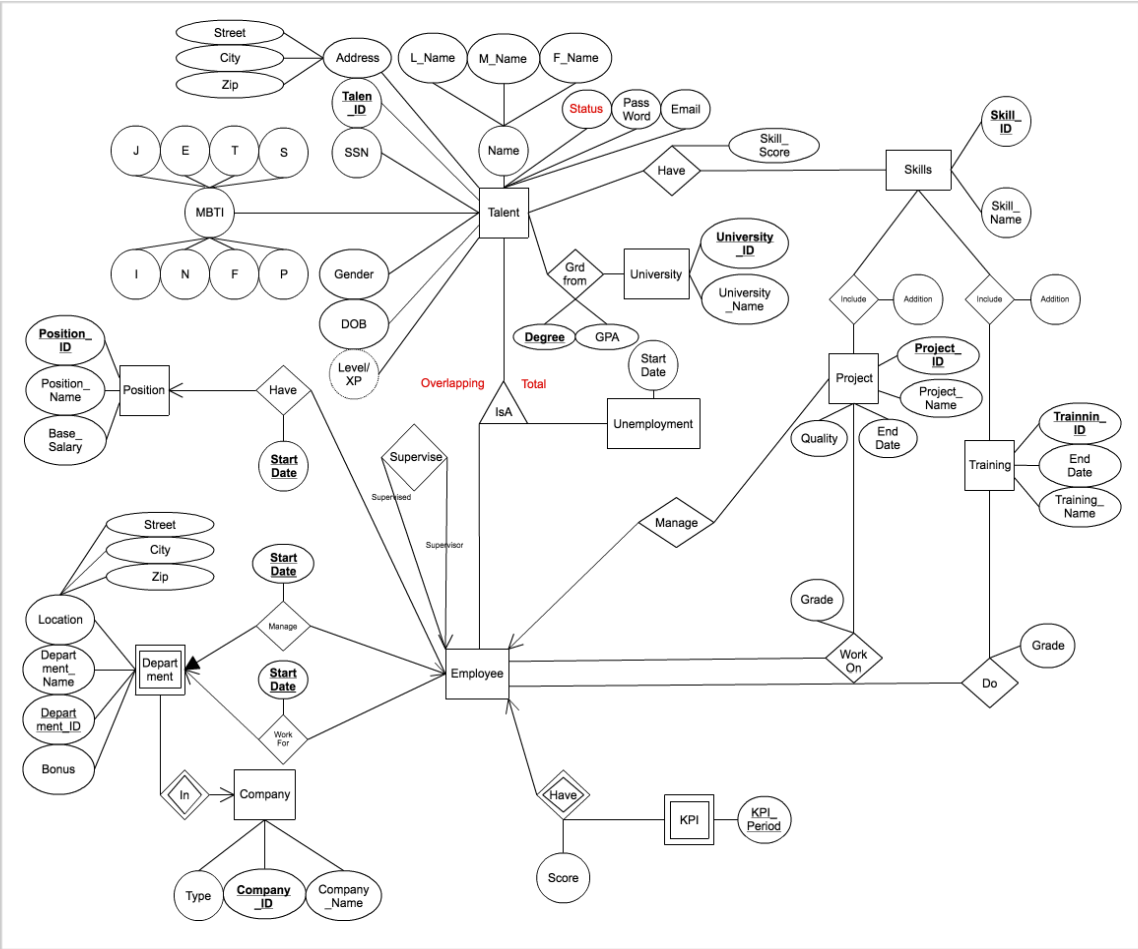

Fig. 4. Entity-Relationship Diagram of the Database

Description:

Talents joining the platform must provide all their personal informations, including education background and MBTI score. After evaluating his/her skill level and experience by Backbone technical staff, the platform will set the initial value of talent's skill score, level and xp(experience). The skill score range from 0 to 100. Once the xp reaching to 1000, the talent will level up. Then, talent's information is ready to be reviewed and selected by companies' HR. Once he/she is hired by some companies' department in some position. The EWC(Employee Working in Company)_Start_Date and EHP(Employee Have Position)_Start_Date will be recorded. And each department should be managed by at least one employee. The system will also record the start date of the employee being as a department manager. So, no matter the employee changing the position, department, company, or even being selected as the department manager, talent's career history activity will be recorded in the database. Each position have the

same base salary, which means if two employees have same position, their base salary will be the same even they work in different companies. For satisfying the requirement of salary variation among different companies' departments, each department could set their own bonus value. What's more, the employee's salary is also correlated with corresponding month KPI. So, each individual's monthly salary have the following formula:
Salary = position base salary + company's department bonus * individuals' monthly KPI score.
This part of the database play a role as the traditional human resource's system.

Also, the Backbone will host training session for all talents. Each session will focus on some skills training and each training participator will gain the corresponding skill score based on his training grade. For example, the platform host a new training session focusing on SQL language. The training's basic addition SQL skill score is 3. Mr. x attend this training and get a B grade. His SQL skill score will be added 6 point (3 * 2(B Grade)). The project is quite similar with the training. The only difference is that each project have a project manager, and each project(not the project member) will be graded as well. Here, we want to mention that for comparability, the third party - Backbone company will evaluate each training/project's difficulties and set the addition skill score and individual's grade. Once talent achieve some accomplishment, for example, attending most number of training in a year. He/she will get some XP, which could be used to level up. This part of the database is innovative, we gamify talents' career activity and combine personal development and company's performance.

2.3 Logic Schema of the Database

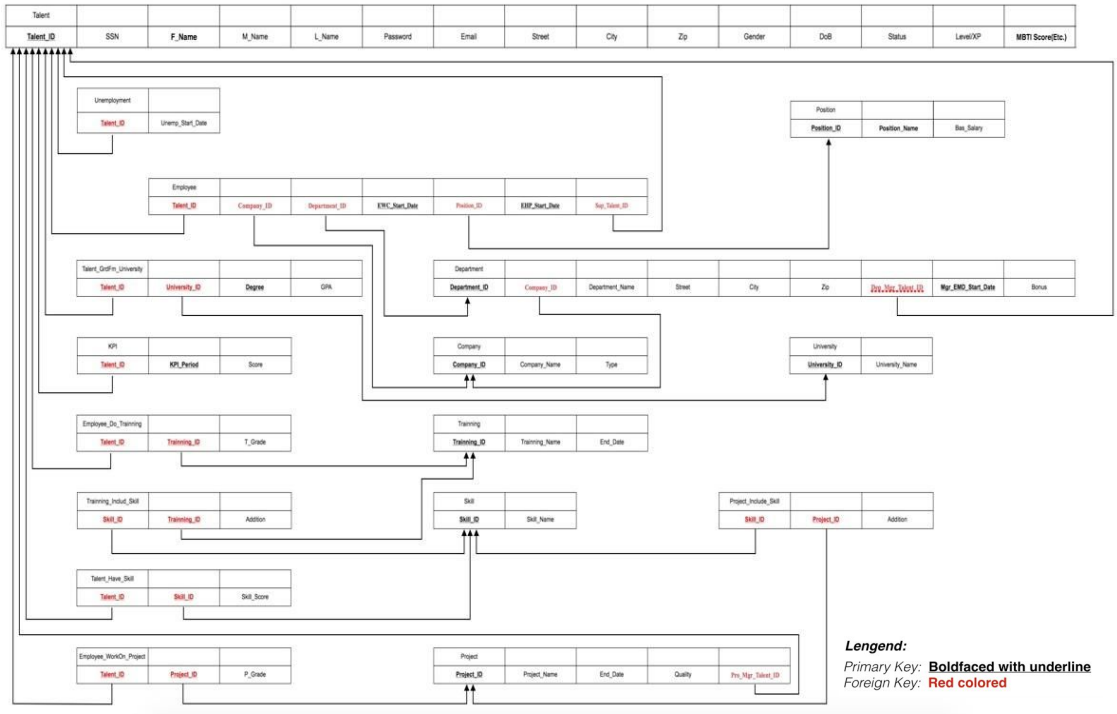

Fig. 5. Logic Schema of the Database (a)

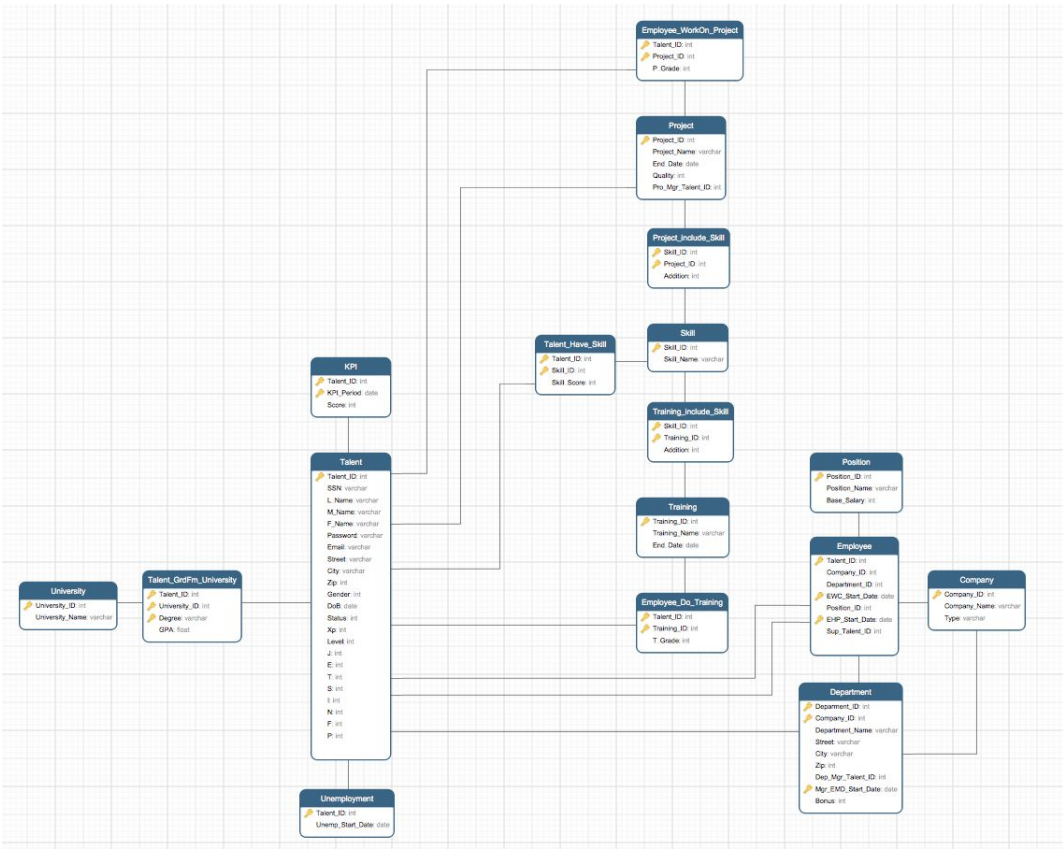

Fig. 6. Logic Schema of the Database (b)

2.4 Normalization of the Database

All the relations are in at least third normal form.

2.5 Constraint and Trigger of the Database

For lack of space, we just showing some of the constraint. For more details, please check the database and the SQL file.

a. Zip code must be five-digit number.
b. Email must be as the following format: xxxxxx@xxx.com
c. Employee 'EHP_Start_Data' (The date the employee having the position) cannot be earlier than the 'EWC_Start_Data' (The data the employee working in the company).
d. etc.

For lack of space, we just showing three representative triggers. For more details, please check the database and the SQL file.

a. When insert a new talent data, we not only insert the data in table 'talent', but also in table 'talent_grdfm_university' and 'talent_have_skill'. Otherwise before we do the insert we will find whether there's a same Talent_Id exists before.

```
DELIMITER $
create trigger insert_talent after insert on `talent` for each row begin
    IF (new.`Talent_ID` not IN (SELECT Talent_ID FROM `talent_grdfm_university`))
    THEN
        insert into `talent_grdfm_university`
        VALUES(new.`Talent_ID`, 0, 9, 0);
    end if;
    IF (new.`Talent_ID` not IN (SELECT Talent_ID FROM `talent_have_skill`))
    THEN
        insert into `talent_have_skill` VALUES(new.`Talent_ID`,0,0);
    end if;
END$
DELIMITER ;
```

b. When a talent exit Backbone, we will delete all of his data in every tables.

```
SET FOREIGN_KEY_CHECKS=0; # to cancel the foreign key constraint
SET SQL_SAFE_UPDATES = 0; # to avoid the error 1175
DELIMITER $
CREATE TRIGGER `delete_talent` AFTER DELETE ON `talent` FOR EACH ROW begin
    DELETE FROM talent_grdfm_university WHERE
    talent_grdFm_university.`Talent_ID` NOT IN
    (SELECT talent.`Talent_ID` FROM talent);
    DELETE FROM talent_have_skill WHERE
    talent_have_skill.`Talent_ID` NOT IN
    (SELECT talent.`Talent_ID` FROM talent);
END$
DELIMITER ;
```

c. When one finish a training, he will get the extra-score

```
DELIMITER $
create trigger train_score after insert on `employee_do_training`
for each row begin
    update `talent_have_skill` SET
    talent_have_skill.Skill_Score=talent_have_skill.Skill_Score+new.T_Grade
    WHERE talent_have_skill.Talent_ID=new.Talent_ID AND
    talent_have_skill.Skill_ID in
    (select training_include_skill.Skill_ID from `training_include_skill`
    JOIN `employee_do_training` ON
    new.Training_ID=training_include_skill.Training_ID);
END$
DELIMITER ;
```

2.6 Index Setting of the Database

By assuming the real-life usage scenario, we test different kinds of index combinations. Based on the tradeoff of time consumption between searching and creating. We set the following index.

Index
Talent: F_Name, L_Name, Status
Talent_GrdFm_University: GPA
Talent_Have_Skill: Skill_Score
Skill: Skill_Name
University: University_Name
Unemployment: None
Employee: None
Company: Company_Name
Department: Department_Name
Position: Position_Name
KPI: None
Training: Training_Name
Project: Project_Name
Training_Include_Skill: None
Project_Include_Skill: None
Employee_Do_Training: T_Grade
Employee_Workon_Project: P_Grade

Fig. 7. The Index Setting of the Database

2.7 Data Create/Transfer

As all we know the company's' human resources data is not open to the public, and the Backbone talent sharing platform is still on the early stage of thinking. Also, we think that the most important part of this platform is the structure of the database, so the authenticity of the data won't affect the building of the Backbone talent sharing platform. Thus, we decide to create our own data. Specifically, we use the Data Creator to create our data. The data is created in CSV (Comma-Separated Values) format. And we use 'import wizard' to transfer our CSV file into the database.

Specifically, we simulate 5000 talents with completed information including 5 skills score and 8 MBTI score. Each talent might graduate from 5 universities with 4 different kinds of degree. For those having jobs, they might work as 5 different roles in 2 companies. Each company has 2 departments. Also, there are 5 trainings and 5 projects. Talents randomly attend those trainings/projects and get their corresponding skill score addition/grade. All the career activities, like finishing project/training, changing the department, changing position, job-hopping, quitting the platform, achieving accomplishment, could achieve by setting different start/ending date among one's career timeline. All the numerical value is uniformly distributed in its corresponding range.

2.8 SQL Query

For lack of space, we just showing some of the SQL Queries.

a. Search talents' basic personal information, such as Name, University, etc.

```
SELECT L_Name, M_Name, F_Name, SSN, Gender, DoB, University_Name, Degree, GPA
FROM Talent, Talent_GrdFm_University, University
WHERE Talent.Talent_ID=Talent_GrdFm_University.Talent_ID
            and Talent.Talent_ID='UserID'
            and University.University_ID=Talent_GrdFm_University.University_ID;
```

b. Search talents' working information.

```
SELECT Company_Name, Department_Name,Street, City, Zip
FROM ((Employee JOIN Department on
                Employee.Department_ID=Department.Deparment_ID)
               JOIN Company on Company.Company_ID=Employee.Company_ID)
WHERE Talent_ID=UserID;
```

c. Search talents' skills and rank in their own company.

```
SELECT A.Skill_Name,count(*)/5000 as Percentage
FROM
(SELECT Skill_Score,Skill_Name
FROM Skill JOIN Talent_Have_Skill on Skill.Skill_ID=Talent_Have_Skill.Skill_ID
) A
JOIN
(
SELECT Skill_Name,Skill_Score
FROM Skill JOIN Talent_Have_Skill on Skill.Skill_ID=Talent_Have_Skill.Skill_ID
WHERE Talent_ID=UserID
) B
on A.Skill_Name=B.Skill_Name
WHERE A.Skill_Score>B.Skill_Score
GROUP BY A.Skill_Name;
```

d. Search talents' skills and rank in their own company.

```
SELECT C.Skill_Name, small/total as Percentage
FROM
(
(
SELECT A.Skill_Name,count(*) as small
FROM
(
SELECT Skill_Score,Skill_Name
FROM
Skill JOIN Talent_Have_Skill on Skill.Skill_ID=Talent_Have_Skill.Skill_ID
WHERE Talent_ID in (
SELECT Talent_ID
FROM Employee
WHERE Company_ID=1
)
) A
JOIN
(
SELECT Skill_Name,Skill_Score
FROM Skill JOIN Talent_Have_Skill on Skill.Skill_ID=Talent_Have_Skill.Skill_ID
WHERE Talent_ID=541964
) B
on A.Skill_Name=B.Skill_Name
WHERE A.Skill_Score>B.Skill_Score
GROUP BY A.Skill_Name
) C
JOIN
(
SELECT count(*) AS total,Skill_Name
FROM
Skill JOIN Talent_Have_Skill on Skill.Skill_ID=Talent_Have_Skill.Skill_ID
WHERE Talent_ID in (
SELECT Talent_ID
FROM Employee
WHERE Company_ID=1
)
GROUP BY Skill_Name
) D
on C.Skill_Name=D.Skill_Name
);
```

e. Search talents' Training history.

```
SELECT End_Date, Training_Name,Skill_Name,T_Grade
FROM
((
SELECT Training_ID,End_Date, Training_Name
FROM Training
WHERE Training_ID in (SELECT Training_ID FROM Employee_Do_Trainning WHERE Talent_ID=UserID)
) A_1
JOIN
(
SELECT Training_ID, T_Grade
FROM Employee_Do_Trainning
WHERE Talent_ID=UserID
) A_2
on A_1.Training_ID=A_2.Training_ID
)
JOIN
(
SELECT Training_ID,Skill_Name
FROM
(SELECT *
FROM Skill) A_3
JOIN
(SELECT *
FROM Trainning_includ_Skill
WHERE Training_ID in (SELECT Training_ID FROM Employee_Do_Trainning WHERE Talent_ID=UserID)
) A_4
on A_3.Skill_ID=A_4.Skill_ID
) A_5
on A_1.Training_ID=A_5.Training_ID;
```

f. Search talents' project history.

```
SELECT End_Date, Project_Name,Skill_Name,P_Grade,Quality
FROM
(((
SELECT Project_ID,P_Grade
FROM Employee_WorkOn_Project
WHERE Talent_ID=UserID
) A
JOIN
(
SELECT Project_ID, Project_Name, End_Date, Quality
FROM Project
WHERE Project_ID in (SELECT Project_ID FROM Employee_WorkOn_Project WHERE Talent_ID=UserID)
) B
on A.Project_ID=B.Project_ID
)
JOIN
Project_Include_Skill on Project_Include_Skill.Project_ID=A.Project_ID
JOIN
Skill
on Skill.Skill_ID=Project_Include_Skill.Skill_ID
);
```

g. Search company's department information for corresponding HR.

```
SELECT Department_Name, TOP_KPI, BOTTOM_KPI, AVG_KPI
FROM
(
SELECT Department_ID, max(Score) as TOP_KPI, min(Score) as BOTTOM_KPI, avg(Score) as AVG_KPI
FROM
(
SELECT Department_ID, Talent_ID
FROM Employee
WHERE Company_ID=GivenComID and Department_ID in(
                                    SELECT Department_ID
                                    FROM Department
                                WHERE Company_ID=GivenComID)
) A
JOIN
(
SELECT F.Talent_ID,Score
FROM
((
SELECT KPI.Talent_ID, max(KPI_Period) as KPI_Period
FROM KPI
GROUP BY KPI.Talent_ID) F
JOIN
KPI
on F.Talent_ID=KPI.Talent_ID and F.KPI_Period=KPI.KPI_Period)
)B
on A.Talent_ID=B.Talent_ID
GROUP BY Department_ID
) C
JOIN
(
SELECT Department_ID,Department_Name
FROM Department
WHERE Company_ID=GivenComID
)D
on C.Department_ID=D.Department_ID
```

h. Search company's project information for HR.

```
SELECT Project_Name, End_Date, Skill_Name, Quality, L_Name as Pro_Name
FROM
(Project
JOIN
Project_include_Skill
on Project.Project_ID=Project_include_Skill.Project_ID)
JOIN Skill
on Project_include_Skill.Skill_ID=Skill.Skill_ID
JOIN  Talent
on Talent.Talent_ID=Project.Pro_Mgr_Talent_ID
JOIN Employee
on Employee.Talent_ID=Talent.Talent_ID
WHERE Company_ID=GivenComID;
```

i. Search company's training information for HR.

```
SELECT L_Name as Name,Training_Name, End_Date, T_Grade
FROM
Talent
JOIN Employee_Do_Training
on Employee_Do_Training.Talent_ID=Talent.Talent_ID
JOIN Training
on Training.Training_ID=Employee_Do_Training.Training_ID
JOIN Employee
on Employee.Talent_ID=Talent.Talent_ID
WHERE Company_ID=GivenComID
```

2.9 User-Interface Design

For lack of space, we just showing one page of the website. The following graph is the screenshot of the talent basic information searching page (searching any talent named Jason).

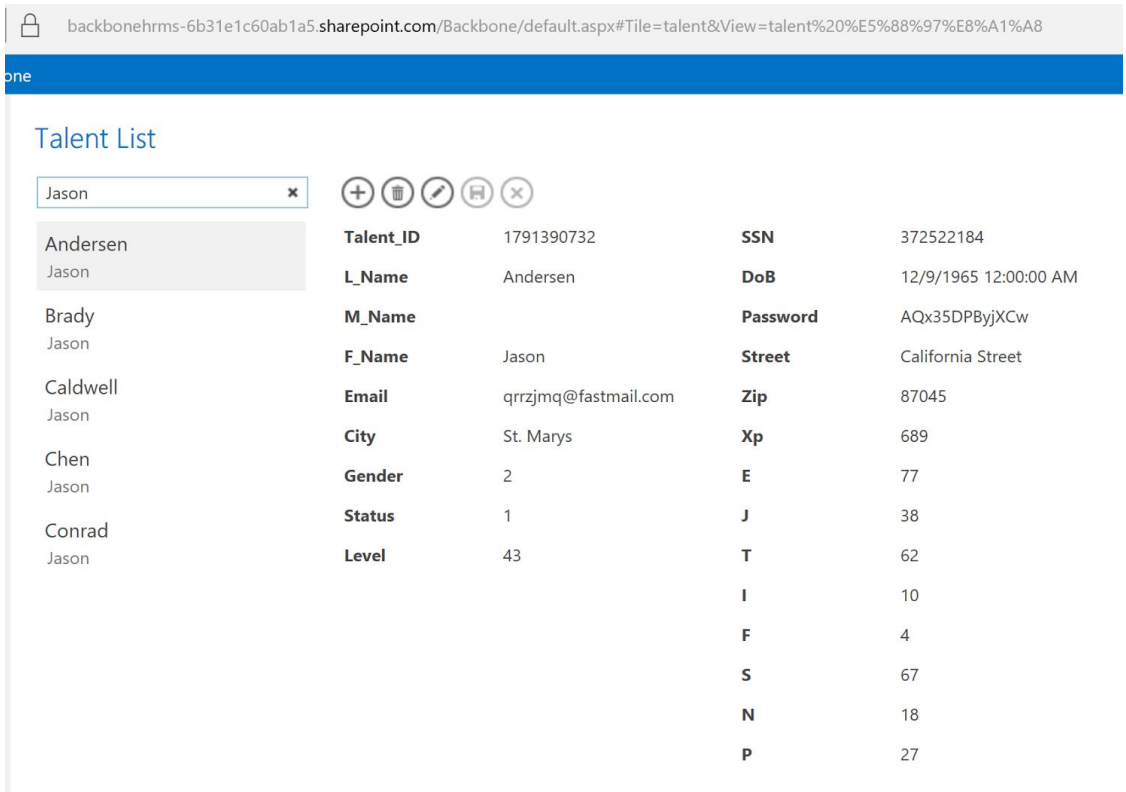

Fig. 8. Backbone User-Interface Design of Talent List

Part III. The Summary

1. The Challenge

In our program, we need a huge amount of data to simulate all the possible scenario. And, obviously, the human resources data we want is definitely not open to public for every company. So it's impossible for us to get the real data. However, in order to consider comprehensively and meet every possible real-life situation. We decide to create our own data and test it with query. Here comes the most challenge part. We have to think carefully about every possible situation. Because no one want to change the structure of the database after building it. And then, we need to create the scenario by creating fake data to simulate it. Compared with real-life data, we must think reversely. We cannot build our database based on real-life data (happened scenarios). Here, we have no data to reference. So, we have to build our own data for each possible scenarios. We rebuild and test over and over again to make sure that the database covers different kinds of situations, even some extreme small possible events. What's more, in order to be statistically meaningful and be as close as to real-life. We create our data randomly.

2. What we Learned

Through building the database, we realize the importance of index setting in a database. A good index setting could improve the performance tremendously. But, a bad index setting would slow down the 'speed' without any doubt. So, how to set the index is a kind of 'art'. We must consider with the real-life usage of the database. However, as we mentioned above, we create the data by ourselves. Based on our experience, we make our own assumption of the database usage statistics. Then, with these assumptions, we try many different index setting and test them one by one. And we analyze each query's index usage and running time. The following graph shows one query's statistics. And the final index tuning result is shown above in part 2.6. Even Though, we 'invest' a plenty of time on index tuning part, we think it is worthful. We learn a lot from index tuning and these knowledge is beyond the course.

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Employee_WorkOn	[Null]	ref	PRIMARY,Project_I	PRIMARY	4	const	3	100.00	Using index
1	SIMPLE	Employee_WorkOn	[Null]	eq_ref	PRIMARY,Project_I	PRIMARY	8	const,backbone.E	1	100.00	[Null]
1	SIMPLE	Project	[Null]	eq_ref	PRIMARY	PRIMARY	4	backbone.Employe	1	100.00	[Null]
1	SIMPLE	Project_Include_Sk	[Null]	ref	PRIMARY,Project_I	Project_ID	4	backbone.Employe	2	100.00	Using index
1	SIMPLE	Skill	[Null]	ALL	PRIMARY	[Null]	[Null]	[Null]	4	25.00	Using where; Usin

Fig. 9. The Index Design for the Backbone System

No one could deny that tech skills are important, however we also learn lots of 'soft' skills by doing this project. On the one side, we learn how to collaborate with other teammates and

how to work in a team environment. And since we all have the different background, taking each member's advantages is quite essential. Jiaxuan know some basic knowledge about database before attending this course, so he controls the program's progress and build the structure of the database. Congyang masters many kinds of softwares and he is a good user interface designer. He play an important role in web designing part. Yiming and Zhaoning both focus on the programing part. On the other side, we learn the skill called 'project management'. We separate the task into different parts, and we set a timeline for helping us to achieve our final goal step by step.

Reality is always more complex than the book. When facing problems or challenges we have never seen in the textbook or learned before, we change ourselves to 'self-study mode'. We search those problems through google engine; we post our questions on Stackoverflow; we ask specialists who might know how to solve the problem. In order to achieve our goal, we utilize different resources we could access. This process do train our problem solving skills.

3. The Conclusion

The project and the course are coming to the end. Through this project, we not only learn technical skills about database but also learn soft skills like how to collaborate. And we believe that the experience from this project will be a cornerstone that leads us to a successful career in the future.

Part IV. The Resources and Reference

[1] Garcia-Molina, H. (2008). Database systems: the complete book. Pearson Education India.

[2] Singh, Jitendra. "CS 542." CS 542. Prof. J Singh, 31 Aug. 2016. Web. 19 Sept. 2016.

[3] Human resource management system. In Wikipedia. Retrieved September 19, 2016, from https://en.wikipedia.org/wiki/Human_resource_management_system

[4] Human resources. In Wikipedia. Retrieved September 19, 2016, from https://en.wikipedia.org/wiki/Human_resources

[5] Dessler, G. (2009). A framework for human resource management. Pearson Education India.

[6] Legge, K. (1995). What is human resource management?. In Human Resource Management (pp. 62-95). Macmillan Education UK.

[7] Steve Sims, Vice President of Badgeville (2016) How Gamification Can Improve HR Management https://www.thebalance.com/gamification-hr-management-improvement-1917995