Design and Implementation of an Executive Search Customer Relationship Management System

Xiaohan Liu *

Department of Computer Science, University of Auckland, Auckland, New Zealand

* Corresponding author email: xliu832@aucklanduni.ac.nz

Abstract. An executive search company seeks and recruits' talents for client organizations. Therefore, it is essential for an executive search company to establish well-maintained relationships with their client and candidate talents. A Customer Relationship Management (CRM) system helps with the task of maintaining customer relationships. The underlying database is a crucial component of a CRM system because the database is responsible for storing the information of the clients and other data associated to the executive search tasks. This paper designs a database specifically for the scenario and requirements of an executive search company, according to the daily operations that the company needs to carry out. This paper designs the entities and relationships between entities based on the executive search requirements. A relational model of the database design is also included in this paper. This paper implements the database in SQL. Finally, this paper implements a user interface to manage and maintain the database of the CRM system.

Keywords: Executive Search; CRM; Database; Relational Model.

1. Introduction

For companies which establish connections with people, the demand for maintaining information of the customers is essential. With a well-maintained database of customers' information, the company can engage the relationships with their customers more efficiently by knowing the customers' information and preferences. Nowadays, companies exploit the Customers Relationship Management (CRM) system to maintain customer information using information technology.

As the name of the customer relationship management system reveals, a CRM system helps manage the relationships between a company and the company's customers. The CRM system seamlessly integrates sales, customer service, marketing, field support, and other business-related functionalities [1]. The CRM system supports these functionalities based on the information insight the company can get from the customer information database maintained by the CRM system. A CRM system aids a company in keeping their valuable customers and reduces the labour costs of manually maintaining the customer data [1]. A survey by Ližbetinová et al. [2] indicates that enterprises appreciate the usefulness of the CRM system, because the knowledge provided by the underlying customer information database helps the enterprises to carry out marketing activities more effectively.

Executive search is a recruitment service. The executive search service provider elicits and analyses the recruitment requirements from organisations and seeks the suitable highly-qualified candidates who are suitable for the role. For an executive search company, the organisations who recruit and the potential candidates are both customers of the company. The company must reach out to and establish connections with the recruiting organisation and potential candidates to accomplish an executive search service. Therefore, a CRM system is helpful for an executive search company that needs to maintain customer connections carefully. It helps the company engage with customers more efficiently by providing the functionalities described in the last paragraph.

As stated by Hualin [3], the customer information database is the engine that powers the CRM system. Only by having a robust customer information database, the CRM system can have the ability to help organisations to dig deep into the customer data to gain insights and analyse the customer values so that the organisation can carry out transactions and marketing activities more effectively. Therefore, it is important to design a well-architected database to support the CRM system. Hualin

suggests [3] that the storage structure of the customer information database should help the CRM system to categorise and segment the data into different tables with a schema that fits to the theme of the CRM system. Based on the survey by Jatana et al. [4], the relational database provides a more suitable storage structure for the purpose of the CRM system. Compared to the NoSQL database, a relational database requires a fixed database schema to restrict the data storage format so that the database can categorise the customer data into different tables more strictly. Also, the "relational" characteristic of the relational database allows the CRM system to link different tables easier to get insight from the data (e.g., link the customer table and the transaction history table to analyse the customers transact habits). Moreover, the relational database's characteristic of requiring a primary key for each table solves a data quality issue in the CRM system indicated by Petrović [5]: data duplication in the customer information database confuses the company about the correct number of customers or the number of transactions that the company has. Enforcing primary keys in a relational database solves the problem of having duplicate records in the tables. Hence, this paper will design a relational database for an executive search CRM system.

There have been research papers previously done related to CRM systems and database. Athanasoulias et al. [7] designed a CRM system for a cosmetic products company. The authors design the database of their CRM system by separating the CRM information into entities, defining each entity's primary, composite, and foreign keys, demonstrating the relationships between them, and normalising the entities to handle the potential anomalies. The authors use an Entity-Relationship diagram to represent the entities and the relationships. However, the research done by Athanasoulias et al. is specifically for a cosmetic products company. Their database design does not support the scenario of an executive search company. Li et al. [8] design and implement a database for a postal enterprise CRM system. The authors use an Entity-Relationship diagram also to indicate the entities and relationships. However, Li et al. do not indicate the foreign keys of each entity clearly in the database model diagram, and there is no implementation of a database management interface to maintain the data visually. Also, the database designed by Li et al. is for a postal enterprise, so it is unsuitable for an executive search company. Wang [9] propose a general conceptual design for CRM databases. The conceptual design is oversimplified and does not indicate each entity's primary and foreign keys. Based on Wang's general design, a more specific database design needs to develop a more detailed list of entities and the relationships according to the use cases of the database. To summarise, the above research papers 1) do not fulfil the requirements of executive search tasks, 2) some papers do not clearly state the primary keys and foreign keys of the entities in the database, and 3) some papers did not implement a database management interface.

To handle the above issues, the database design for an executive search CRM system needs to clearly indicate the entities and relationships according to the use cases of executive search tasks. The database design needs to clearly indicate the primary keys and foreign keys for each entity. The implementation of the database should have an interface for database management. This paper analyses the business requirements of a typical executive search company based on the possible tasks that the company can carry out. With the business requirements analyzation, the author uses an entity-relationship model to specify the entities within the database, the relationships between each entity, and the multiplicities of the relationships. This paper uses the min-max notation to represent the multiplicities of the relationships. The author indicates the primary keys and foreign keys of the tables by normalising the tables into the third normal form [6] to avoid any database anomalies. The author builds the database in Oracle SQL. Also, the author develops a web-based frontend interface for the CRM database management using JavaScript and React.js.

2. Requirements Analysis

The main customers of an executive search company are the enterprises and organizations that have the demand for recruiting and assigning talents for specific roles. Therefore, the CRM system needs to maintain the information of the enterprises or organizations in the database. To complete an

assignment task, the executive search company needs to seek potential candidates that meet the assignment requirements. Therefore, the CRM system needs to maintain the information of potential candidates in the database. Also, the database should keep information of all the assignments that the company has processed or is processing. The requirements of the executive search CRM system database are as follows:

- The database should keep information about the companies that want to recruit talents for roles within the company. Each company should have a unique identifier, the name of the company, the business telephone number, the business email address, and the physical address of the organization. A company has a category or multiple categories indicating which industry the company is in.
- The database should keep information about people that the executive search company has connections. Each person should have a unique identifier and a full name. The database keeps the education history of a person and any awards that the person received during their education, if there is any. The database also tracks the employment history of each person. The employment history contains data about the company and the department that the person has worked at or is working at, the role of the person in the specific employment history, the start and end date of the employment history, and a status indicating if the employment history is current or previous. Each person should have a referee who proves the reliability of the person. The database should also keep the information of the referee to refer to ensure the reliability of a role assignment task.
- The database should keep information on the role assignments. A role assignment is about seeking candidates for the role that a company recruits talents for. The database should maintain a unique identifier for each assignment, the company that raises the assignment, the department of the role, and the role title.
- The executive search company can publish advertisements for a specific role assignment to promote the executive search task and to increase talent-seeking efficiency. The database should keep information of each advertisement, including a unique identifier for the advertisement, the cost of the advertisement, the duration of the advertisement, and the platform that publishes the advertisement.
- The executive search company sets milestones for each assignment task to help the executive search team track the progress of the task. The database should keep information on all milestones related to the assignments, including a unique identifier for each milestone, a milestone description, the due date of the milestone, and a status indicating if the milestone is completed.

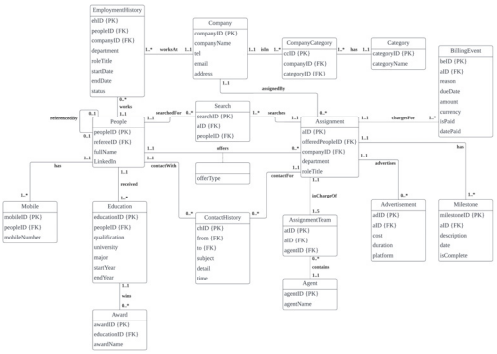

Figure 1. ER diagram of the database

- The executive search company charges fees from the recruiting companies for carrying out talent-seeking tasks for them. The database should keep information about all the billing events associated with the assignment task, including a unique identifier for each billing event, the billing reason, the due date of the bill, the amount of money that the bill charges, the currency

of the bill, a status indicating if the organization has paid for the bill, and the date that the organization pays the bill.

- A role assignment can have multiple candidates who meet the requirements of the assignment. A person can appear in multiple assignments as a candidate. Once the recruiting company has decided on a suitable person for the role and sends the offer to the person, the database should keep the information of the offer, including the person who receives the offer and the offer type.
- The database should keep information about the agents in the executive search company who seek candidates. Each assignment should have a minimum of one agent and a maximum of five agents working on it. An agent can work on none or many assignments.
- The executive search team contacts potential candidates to communicate the details of the assignments. The database should keep information about the contact history, including a unique identifier for each contact history, the sender and receiver of the contact message, the subject of the contact history, the details of the contact history, and the time that the contact happens.

3. Database Design

To fulfil the requirements of the executive search CRM system database, the database splits into multiple tables containing data that fits the purpose of the table, and the tables have relationships to each other with the correct multiplicity. The conceptual design of the database is represented in Figure1.

The database has three main entities: People, Company, and Assignment. A People instance can have a referee who is also a People instance in the database, so People has a recursive relationship referenced By to itself. A People instance can have multiple mobile numbers, so People has a relationship to the Mobile entity. A People instance must have received one or more education degrees, and the person can have been awarded zero or multiple awards for each education.

A People instance can have past or present employment history with multiple companies, and each Company instance can have one or many present or past employees. Therefore, there is a Many-to-Many relationship named Employment History between People and Company. A Company instance can have one or more industry categories, and a Category instance can have multiple companies. Hence, there is a Many-to-Many relationship between Company and Category.

A People instance can appear in multiple executive search assignments if the person matches the requirements of the assignment, and an Assignment instance could have multiple people to be the candidates. Therefore, there is a Many-to-Many relationship between People and Search named Search. The database should keep the details of the contact history between the executive search agents and the candidates for an assignment.

Each Assignment instance is associated with a Company instance indicating who is searching for executives. A Company instance can have multiple assignments going on. When the company has chosen a candidate to issue the offer for an assignment, the offer needs to specify the offer type. The executive search company assigns a team of agents for each assignment, containing one to five agents per team. An agent can be working on multiple assignments at the same time. Each Assignment instance must have one or multiple billing events that charge fees from the client company. Each Assignment instance has one or many milestones that needs to meet throughout the life cycle of the assignment. Each Assignment instance can pay for advertisements to promote the assignment.

The conceptual design of the database maps into the database's relational model to indicate the correct primary and foreign keys of the tables according to their relationships. In the E-R diagram in Figure 1, the primary key attributes of the entities are followed by the notation {PK}, the foreign key attributes of the entities are followed by the notation {FK}. The relational model is in the third normal form to avoid any possible database anomalies (e.g., insert anomaly, update anomaly, delete anomaly) that can cause problems when implementing and using the database.

4. Implementation

4.1 Database Implementation

To implement the database, the author uses Oracle SQL. The author indicates the database schema by using the CREATE operation in SQL. The system carries out all CRUD operations (create, rea, update, delete) with SQL code to maintain the database. The followings (Figure 2-6) are some code snippets as the CRUD operation examples:

Create statement: Figure 2 contains three examples of creating tables with SQL code in the database. The three tables represent the three main entities in the database design (People, Company, Assignment). The create statement specifies the data type of each attribute. For example, peopleID in People is a numeric primary key that cannot be null, and roleTitle in Assignment is a varchar type with a variable length of 50 characters. The create statement also specifies database constraints. For example, the constraint in People names referee_fk indicates that refereeID is a foreign key that references peopleID in People.

```
CREATE TABLE People (
    peopleID NUMERIC NOT NULL PRIMARY KEY,
    fullName varchar(50),
    LinkedIn varchar(200),
    refereeID NUMERIC NULL,
    CONSTRAINT referee_fk FOREIGN KEY (refereeID) REFERENCES People(peopleID)
);

CREATE TABLE Company (
    companyID NUMERIC NOT NULL PRIMARY KEY,
    companyName varchar(50),
    tel varchar(20),
    email varchar(50),
    address varchar(100)
);

CREATE TABLE Assignment (
    aID NUMERIC NOT NULL PRIMARY KEY,
    department varchar(30),
    roleTitle varchar(50),
    offeredPeopleID NUMERIC,
    offerType varchar(20),
    companyID NUMERIC NOT NULL,
    CONSTRAINT offer_people_fk FOREIGN KEY (offeredPeopleID) REFERENCES People(peopleID),
    CONSTRAINT company_fk FOREIGN KEY (companyID) REFERENCES Company(companyID)
);
```

Figure 2. Create table operations SQL code snippet

Insert statement: Figure 3 shows examples of the insert statement. The name indicates the target table that takes inserted values after “INSERT INTO”, and the values are indicated by the content in the bracket of the statement. For example, the first line in Figure 3 means a record of People with a peopleID of 001, a fullName of “John Smith”, a LinkedIn address of “https://www. linkedin. com/in/ john-smith-001/”, and an empty refereeID is inserted into the table People.

```
INSERT INTO People VALUES (001, 'John Smith', 'https://www.linkedin.com/in/john-smith-001/', '');
INSERT INTO People VALUES (002, 'John Doe', 'https://www.linkedin.com/in/john-Doe-002/', 001);
INSERT INTO People VALUES (003, 'Jane Doe', 'https://www.linkedin.com/in/jane-Doe-003/', 002);
INSERT INTO People VALUES (004, 'Jane Smith', 'https://www.linkedin.com/in/jane-smith-004/', '');
INSERT INTO Company VALUES (001, 'Apple', '100000', 'apple@apple.com',
    'One Apple Park Way; Cupertino, CA 95014, U.S.A.');
INSERT INTO Company VALUES (002, 'Lenevo', '123456', 'lenovo@lenovo.com',
    ' Building 2, No.10. Courtyard Xibeiwang East Road Haidian District 100094 Beijing,China');
INSERT INTO Company VALUES (003, 'Facebook', '234567', 'facebook@fb.com',
    'Facebook Headquarters 1 Hacker Way Menlo Park, CA 94025');
INSERT INTO Assignment VALUES (001, 'HR', 'HR Manager', '', 'permanent', 001);
INSERT INTO Assignment VALUES (002, 'IT', 'CTO', '001', 'permanent', 001);
INSERT INTO Assignment VALUES (003, 'Finance', 'CFO', '003', 'permanent', 002);
```

Figure 3. Insert operations SQL code snippet

Select statement: Figure 4 shows two examples of the select statement that carries out the read operation to the database. The first select statement in Figure 4 retrieves the field of agentName from table Agent, which works on an Assignment record with aID of 001. The second select statement in Figure 4 retrieves the field of companyName from the table Company that has any unpaid billing events associated.

```
SELECT agentName
FROM Agent
WHERE agentID IN (
    SELECT agentID
    FROM AssignmentTeam
    WHERE aID = 001
);

SELECT companyName
FROM Company
WHERE companyID IN (
    SELECT companyID
    FROM Assignment
    WHERE aID IN (
        SELECT aID
        FROM BillingEvent
        WHERE isPaid = 0
    )
);
```

Figure 4. Read operations SQL code snippet

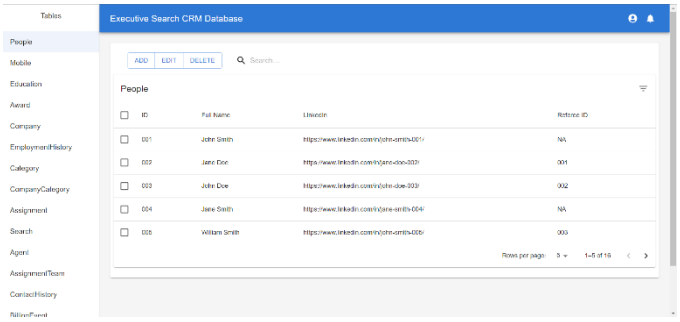

Figure 5. Database user interface of the People table

Update statement: Figure 6 shows two examples of the update statement. The update statement edits a specific record within a particular table. For example, the second update statement in Figure 6 changes the offerType field of the record with aID of 001 in the Assignment table to casual.

```
UPDATE People SET LinkedIn = 'https://www.linkedin.com/in/john-smith-0011/'
    WHERE peopleID = 001;
UPDATE Assignment SET offerType = 'casual' WHERE aID = 001;
```

Figure 6. Update operations SQL code snippet

Delete statement: Figure 7 contains two examples of the delete statement. A delete statement deletes records from the tables in the database. For example, the first delete statement in Figure 7 deletes the record with a peopleID of 004 from the People table, and the second delete statement in Figure 7 deletes the record with a companyName field of Lenovo from the Company table.

```
DELETE FROM People WHERE peopleID = 004;
DELETE FROM Company WHERE companyName = 'Lenovo';
```

Figure 7. Delete operations SQL code snippet

4.2 User Interface

To allow users to maintain the database more efficiently on a graphical user interface, the author develops a frontend interface for the CRM database. The author develops the interface with JavaScript and the React.js framework of JavaScript. Building the user interface as a web application is because a deployed web application can run on the user's browser, which saves the effort of installing an executable application on the user's local machine. The author uses JavaScript to develop the user interface because JavaScript is the industry standard [10] for developing a web application. The author chooses to use React.js as the frontend framework to develop the user interface because React.js is the most used JavaScript frontend framework in 2022 [11], so a wide range of developers can maintain the application in the case of code maintenance.

The database interface displays all tables that exist in the database. The list of tables in the database is on the left-hand side of the interface. In the screenshot in Figure 5, the interface shows the People table. The interface shows the rows of record of each table on the right in the main area of the page. The interface retrieves all records from a table in the database, and the user can view the records in the list view, as shown in Figure 5. At the bottom of the list view, the user can select the number of rows to display on one page of the table. In the screenshot in Figure 5, the table shows five rows per page. On top of the table is a toolbar containing an ADD button, an EDIT button, a DELETE button, and a search bar. The users can carry out the insert operation by clicking the ADD button, edit a row of records by selecting a row and clicking the EDIT button, and delete a row of records by selecting a row and clicking the DELETE button. Also, the users can carry out a blurred search by entering a keyword in the search bar above each table. The user can sort the table ascending or descending by clicking on the column header in the table.

Figure 8 is the screenshot of the modal in the interface that allows users to add a new record to the table. The content of the modal depends on the attributes of the specific table. For example, Figure 8

shows that the user needs to enter the full name (a compulsory text box), the LinkedIn address (an optional text box), and the referee ID (an optional text box) to insert a new record into the table People. The peopleID field of People is automatically generated by the database when inserting the record into the table.

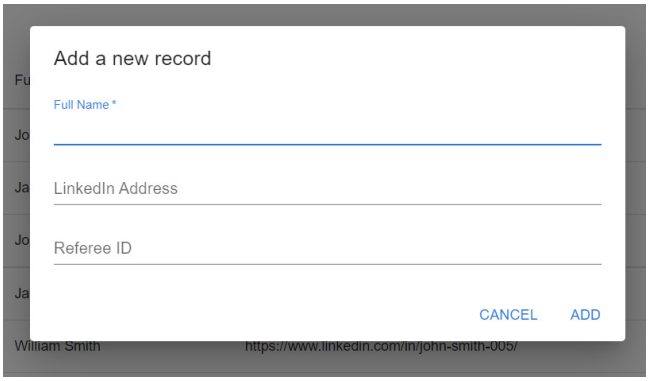

Figure 8. The modal of adding a new record into the People table

5. Conclusion

This paper designs and implements a CRM database for the executive search purpose. As discussed in the above sections, the design of the database can well meet the requirements of an executive search CRM system. The database is designed to carry out use cases of the daily operations of an executive search company. The relational modelling of the database can avoid database anomalies. The implementation of the database management user interface is a web-based application so that it can be convenient for usage. Also, a cross-platform web-based application saves the effort for subsequent multi-platform development. The selection of the language and the framework are for extensibility, that popular language and framework save the learning cost for developers who continue to work on the database management interface.

References

[1] Y. Xu, D. C. Yen, B. Lin, and D. C. Chou, “Adopting customer relationship management technology,” Industrial Management & Data Systems, vol. 102, no. 8, pp. 442–452, 2002.

[2] L. Ližbetinová, P. Štarchoň, D. Weberová, E. Nedeliaková, and M. Juříková, “The approach of smes to using the customer databases and CRM: Empirical Study in the Slovak Republic,” Sustainability, vol. 12, no. 1, p. 227, 2019.

[3] W. Hualin, “The exploitation and utilization of customer database in CRM,” 2010 International Conference on E-Business and E-Government, 2010.

[4] N. Jatana, S. Puri, M. Ahuja, I. Kathuria, and D. Gosain, “A survey and comparison of relational and non-relational database,” International Journal of Engineering Research & Technology, vol. 1, August 2012.

[5] M. Petrović, “Data quality in Customer Relationship Management (CRM): Literature review,” Strategic Management, vol. 25, no. 2, pp. 40–47, 2020.

[6] P. A. Bernstein, “Synthesizing third normal form relations from functional dependencies,” ACM Transactions on Database Systems, vol. 1, no. 4, pp. 277–298, 1976.

[7] G. Athanasoulias, and P. Chountalas, “Increasing business intelligence through a CRM approach: An implementation scheme and application framework,” International Journal of Information, Business and Management, pp. 146-178, 2019.

[8] P. Li and C. Lu, "Database design and implementation based on postal enterprise CRM," 2010 2nd International Conference on Computer Engineering and Technology, pp. V1-720-V1-723, 2010.

[9] Y. Wang, “Database design on Customer Relationship Management System,” Proceedings of the 2016 International Conference on Economics, Social Science, Arts, Education and Management Engineering, 2016.

[10] "State of the Developer Nation Report," SlashData, 2021. Accessed: Sep. 20, 2022. [online]. Available at: https://slashdata-website-cms.s3.amazonaws.com/sample_reports/_TPqMJKJpsfPe7ph.pdf.

[11] L. S. Vailshery, "Most used web frameworks among developers 2022," Statista, 09-Aug-2022. [Online]. Available: https:// www.statista.com/statistics/ 1124699/ worldwide-developer-survey-most-used-frameworks-web/. [Accessed: 30-Sep-2022].