

The Smile Factory - Dentist Surgery

1st Given Edward Patch
Software Engineer Student (of HND Year 2)
Database and Application Development
University Trinity St. Davids (of Beverly Holland)
Swansea, Wales
Student ID: 1801492

CONTENTS

I	Introduction	1
II	Database Creation Process	1
II-A	Useful Database Tools	1
II-B	Test Evaluation Table	1
II-C	Evaluation	1
III	Performance Tuning	1
IV	Entity Relationship Diagram	2
V	Appendices	3
V-A	Test Evaluation Evidence	3
VI	References	3

Abstract—A SQL database is designed for a Dentist Surgery. The database must handle high activity, as the surgery has grown in popularity.

Index Terms—dentist, surgery, database, sql

I. INTRODUCTION

The Smile Factory is a dental and orthodontic Surgery which is located in Morriston, Swansea have requested for a database. The database will help with the increase in patients.

This report covers the design structure of the database, the creation of the database and opportunities for performance tuning of the database, for The Smile Factory dentist surgery. The ERD (Entity Relationship Design) displays The Smile Factory's database design structure.

II. DATABASE CREATION PROCESS

The database creation process will cover features, issues that arose and the fixes to the problems found during the creation of The Smile Factory's database.

A. Useful Database Tools

The following tools used to put together the The Smile Factory database with example data, to demonstrate and stress test database functionality.

- Patch, Edward [1] Insert SQL Generator - The Python application asks for the table's name, field quantity, field types (integer or string) and then asks for the user to enter

field data. The software can generate more than one rows if desired. The application prints the finalised SQL code, which enables the user to copy to the SQL files.

- [2] Postal Codes in Swansea - The application enabled a generation of postal areas. This tool saved time recording random addresses for patient and staff data input.
- Smile Dental Care [3] - The Private Fee Guide blog helped price up treatments plans to generate testing data for The Smile Factory's database.

B. Test Evaluation Table

Problem	Description	Expected Result	Actual Result	Solution
chk_telNumber	The field was too small for the constraint.	12 digits. Telephone phone number with whitespace.	Constraint error.	Increase field size by one.
Patient's Gender Field	Patient's gender field lost data on input.	Genders like 'Other' or 'Female' should be stored.	Loss of data.	Increase field size by one.
Appointments Composite Keys	Composite keys should be unique.	The composite key should only allow and generate unique data.	If the date/time-matched, with the client ID, it would count as duplicate.	Works as expected.
triStaffIDUpper Trigger	The trigger Uppercases the staff ID	's0000' should update to 'S0000'.	's0000' does update to 'S0000'.	Works as expected.
triSurgeryEventEntry Trigger	The trigger makes a new entry to Surgery Event table.	The trigger should enter a new row, based on the appointment table.	The trigger does enter new row, based on the appointment table.	Works as expected.
bankTran6 Transaction	If the customer gives incorrect card details, then it should rollback the transaction.	If the card number entered wrongly, then should rollback the transaction.	If card number entered wrongly, then it does rollback the transaction.	Works as expected.

C. Evaluation

After testing the database, the problems that cropped up have only been minor. The database has appropriate constraints and triggers, enabling accurate data entry. The SQL database works well with application sides, with appropriately named table and fields.

The SQL code will demonstrate the triggers functioning as intended, with some field data not entered correctly. The Surgery Event table is fully automated. Automation increases performance and the flexibility of data, The Smile Factory dentist surgery, to make reports.

III. PERFORMANCE TUNING

Performance tuning is crucial to making a database run smoothly, for both performance and usability. Performance tuning changes the ways, Database Engineers can reuse data, to create faster and better queries. According to Bloch, Kevin [4] "The Execution Plan tool found in SQL Server can be useful for creating indexes." After researching further, S, Vithal [5] suggests a few SQL performance methods such as using little data as possible, from available indexes. Refer to SQL code for references, of performance tuning.

IV. ENTITY RELATIONSHIP DIAGRAM

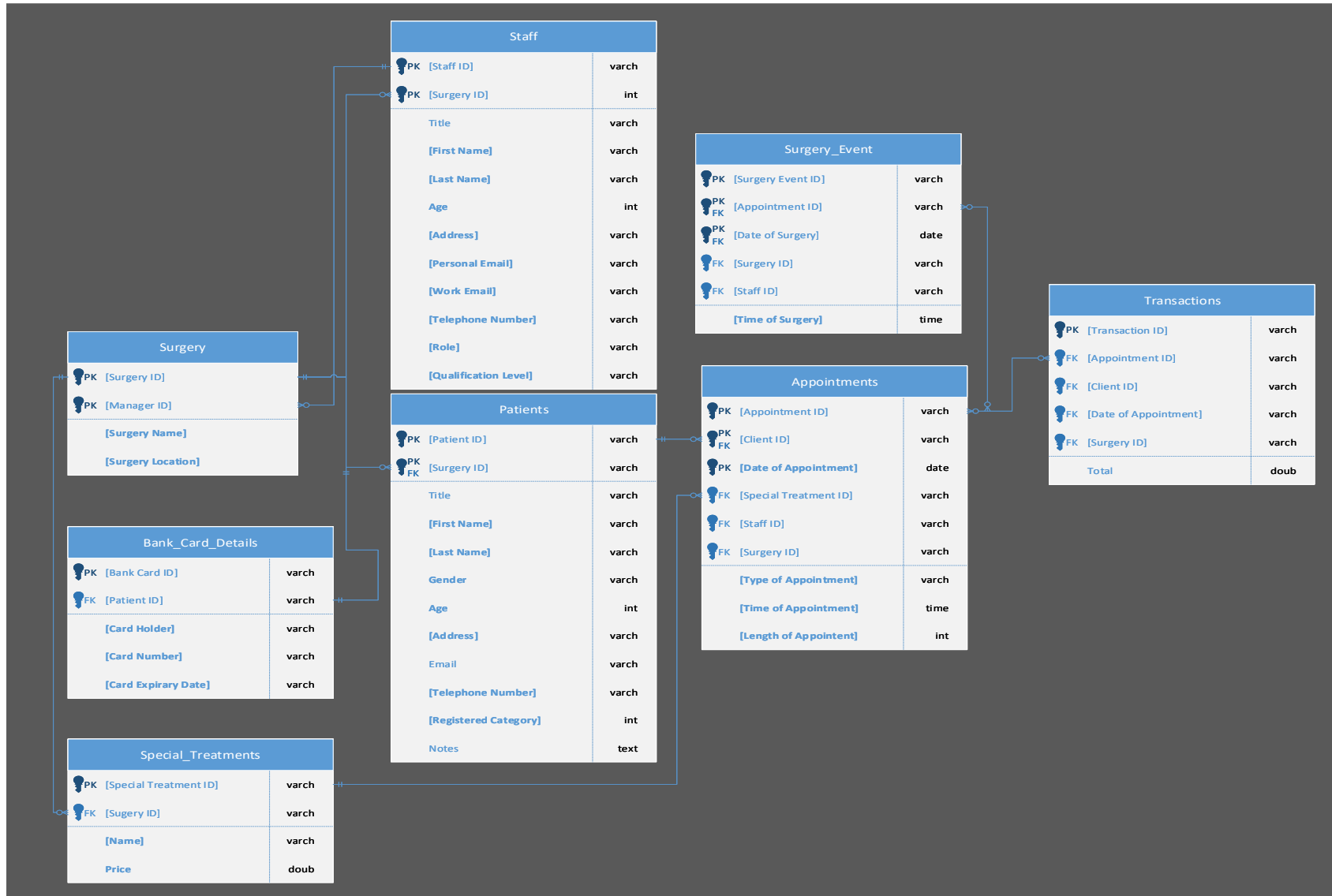


Fig. 1. The Smile Factory - Entity Relationship Diagram

V. APPENDICES

A. Test Evaluation Evidence

Patients Table's Telephone Number Field:

Checks for 12 characters but only accepts 11.

```
[Telephone Number] char(11) NOT NULL CONSTRAINT  
chk_telNumber CHECK([Telephone Number] LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'),
```

Fix:

```
[Telephone Number] char(12) NOT NULL CONSTRAINT  
chk_telNumber CHECK([Telephone Number] LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'),
```

Patients Table's Gender Field:

Female is a six-character word. Gender field only accepts five characters.

```
Gender char(5) NOT NULL,
```

Fix:

```
Gender char(6) NOT NULL,
```

VI. REFERENCES

REFERENCES

- [1] E. Patch, "SQL Insert Data - Jupyter Notebook," Mar. 2021. [Online]. Available: <https://github.com/TheLifeOfComputing/SQL-Generators>
- [2] "Postcode Swansea | postal-code.co.uk." [Online]. Available: <https://postal-code.co.uk/postcode/Swansea>
- [3] Smile Dental Care, "Private Dental Fee Guide." [Online]. Available: <https://www.smile-dentalcare.co.uk/private-dental-fee-guide/>
- [4] K. Bloch, "SQL Database Performance Tuning for Developers." [Online]. Available: <https://www.toptal.com/sql-server/sql-database-tuning-for-developers>
- [5] V. S, "Steps to Optimize SQL Query Performance – Best Practices," Jan. 2019. [Online]. Available: <https://dwgeek.com/steps-to-optimize-sql-query-performance-best-practices.html/>