Hani Khaled Kamal

22102516

Designing Database for Software Company Using MySQL

Dr. Essa Shahra

Computing and Digital Technology School

Birmingham City University

Birmingham, United Kingdom

May 2022

Contents

Introduction

A database is a "self-descriptive collection of integrated records" (Kroenke, 1995). This is one of the most concise and detailed definitions, although it is neither conventional nor unique, even though most separate definitions share much in common. An unofficial Google search for "database definition" yielded 23 unique results (Google, 2008). This causes fundamental issues for newcomers to the database environment. Terminology is often used interchangeably for several database items, which adds to the confusion. A database, on the other hand, is essentially a method of capturing, storing, organising, and presenting data as information, often for commercial purposes, though many other uses for databases and database technology have evolved since officially defined databases were formed.

Critical Evaluation of Database Systems

In this section I will evaluate some of the most popular and used database systems and technologies in the market today.

Database types are the patterns and structures that are used to organise data within a database management system. They are also known as database models or database families. Over the years, several distinct database types have emerged. Some are primarily historical forerunners of modern databases, although others have withstood the test of time. New types have been established in recent decades to accommodate changing requirements and various use patterns.

Your database type selection can have a significant influence on the kind of operations that your application can easily do, how you perceive your data, and the functionality that your database management system provides during development and runtime. In this article, we'll look at how database types have changed through time, as well as the benefits and drawbacks of each architecture.

Nowadays, there is different types of databases models such as:

1. Relational databases
2. Non-relational or NoSQL database
3. Object-Oriented database
4. Network database

Relational databases

The most prevalent sort of database is a relational database. It employs schema, which is a blueprint for dictating the data structure stored in the database.

For example, a corporation selling things to its clients, must have some type of stored knowledge about where these products travel, to whom, and in what quantities.

Advantages:

- Simplicity: The Relational model in DBMS is straightforward because tables with rows and columns are natural and easy to grasp.
- Query capability: It allows a high-level query language, such as SQL, to avoid sophisticated database exploration.
- Data independence: The structure of a relational database may be altered without requiring the application to be modified.
- Scalable: A database should be expanded in terms of the number of entries, or rows, and fields to improve usability.

Disadvantages:

- Few relational databases include field length limitations that cannot be exceeded.
- Relational databases may get complex as the amount of data increases and the relationships between data pieces become more sophisticated.

- Complex relational database systems may result in isolated databases where information cannot be transferred between systems.

Examples

- MySQL: is a well-organized collection of data. It might be anything from a basic grocery list to a photo gallery or a location to store massive quantities of data in a corporate network. MySQL collects and organizes data using the relational paradigm. Tables in this paradigm are made up of rows and columns, and all relationships between data items adhere to a precise logical structure. An RDBMS is basically a collection of software tools used to construct, administer, and query a database.
- Microsoft SQL Server: is a relational database management system (RDBMS) that is used in corporate IT settings to handle a wide range of transaction processing, data analytics, and analytics applications. Microsoft SQL Server, along with Oracle Database and IBM's DB2, is one of the three market-leading database technologies.

Non-relational or NoSQL database

Non-relational databases (also known as NoSQL databases) differ from standard relational databases in that their data is stored in a non-tabular format. Non-relational databases, on the other hand, may be built on data formats such as documents. A document can be quite thorough while also including a variety of different sorts of information in various forms. Non-relational databases are far more adaptable than relational databases due to their capacity to digest and arrange different types of information side by side.

Advantages:

- Large amounts of organized, semi-structured, and unstructured data can be stored in a flexible manner.
- can support rapid sprint iterations and code pushes
- may easily expand out architecture without incurring costly costs

Disadvantages:

- Non-relational databases do not allow ACID (Atomicity, Consistency, Isolation and Durability) transactions. They instead rely on "eventual constancy." The performance advantages of these databases come at the expense of consistency.
- There are several types of NoSQL databases, but there is little consistency among them.
- There is no special programming interface to the various databases.
- Not all non-relational databases are effective at automating the process of sharding or distributing the database over numerous nodes.

Examples

- MongoDB: is a free and open-source NoSQL database management system. NoSQL is a database technology that is used as an alternative to traditional relational databases. NoSQL databases are extremely handy for dealing with massive amounts of scattered data. MongoDB is a tool for managing document-oriented data, as well as storing and retrieving data.

Object-Oriented database

An object-oriented database (OOD) is a database system that can handle sophisticated data objects, such as those found in object-oriented programming languages.

Everything in object-oriented programming is an object, and many objects are highly complicated, with various characteristics and actions. To enable the storing and retrieval of object-oriented data, an object-oriented database management system collaborates with an object-oriented programming language.

Advantages:

- Complex data sets can be saved and retrieved quickly and easily.
- Object IDs are assigned automatically.
- Works well with object-oriented programming languages.

Disadvantages:

- Object databases are not widely adopted. Relational databases may get complex as the amount of data increases and the relationships between data pieces become more sophisticated.
- In some situations, the high complexity can cause performance problems.

Examples

- GemStone / Smalltalk: In a single, computationally full language, GemStone Smalltalk includes data creation, data manipulation, and query capabilities. It is meant to work in a multi-user environment, with a transaction and concurrency management mechanism and a class library designed for multi-user access to objects.

Network database

A database that is organized based on record ownership, allowing records to have numerous owners, and therefore offering various access pathways to the data. CODASYL (Conference on Data Systems Languages) DBMSs are database management systems (DBMSs) that provide such features.

Advantages:

- Conceptual simplicity: The network model, like the hierarchical model, is conceptually simple and easy to create.
- Capability to manage additional connection types: The network model can handle one-to-many and many-to-many relationships, which is extremely useful in modelling real-world scenarios.
- Data access is easier and more flexible than in the hierarchical approach.
- Data integrity: The network model does not let a member to exist in the absence of an owner.
- Data independence: The network architecture outperforms the hierarchical approach in separating programmes from the complexities of physical storage.

Disadvantages:

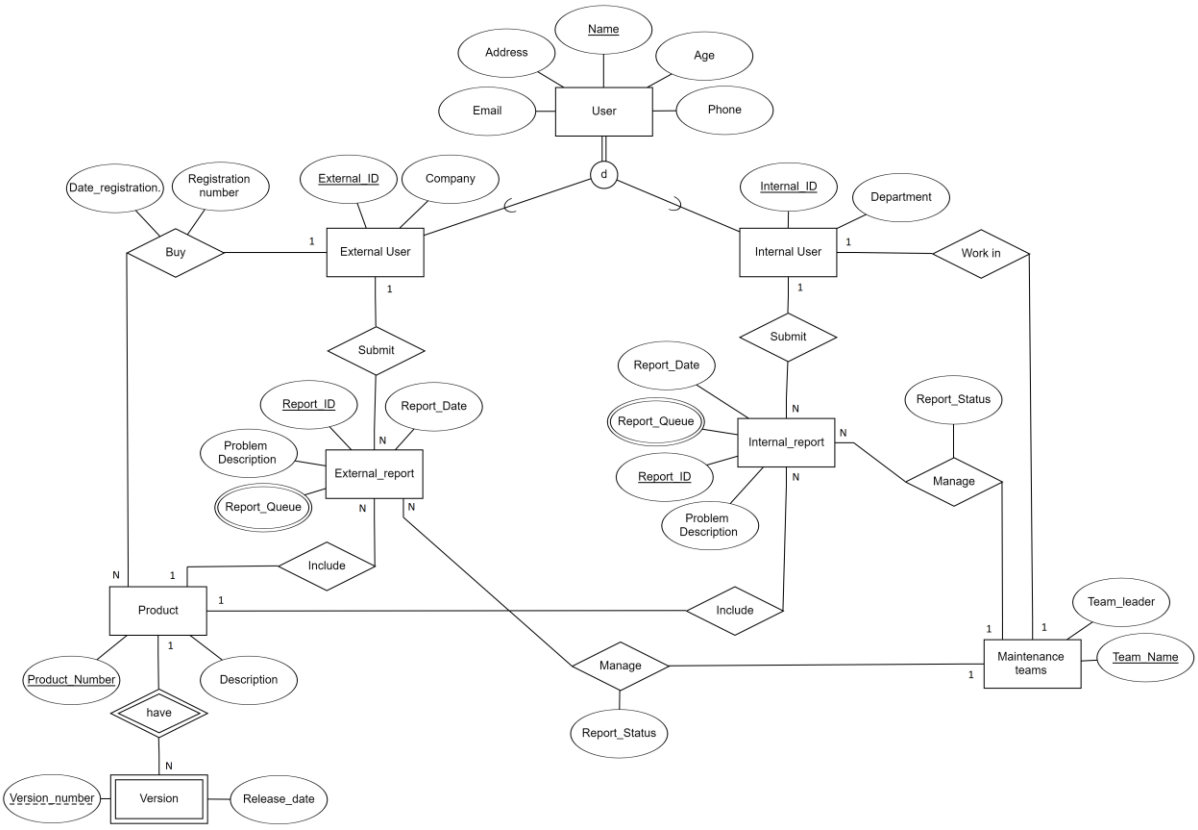
- System complexity: Because all records are kept via pointers, the entire database structure becomes extremely complicated.
- Operational Anomalies: Insertion, deletion, and updating of any record necessitate a huge number of pointer alterations.
- Lack of structural independence: it is extremely difficult to make fundamental modifications to the database. Complex relational database systems may result in isolated databases where information cannot be transferred between systems.

Examples

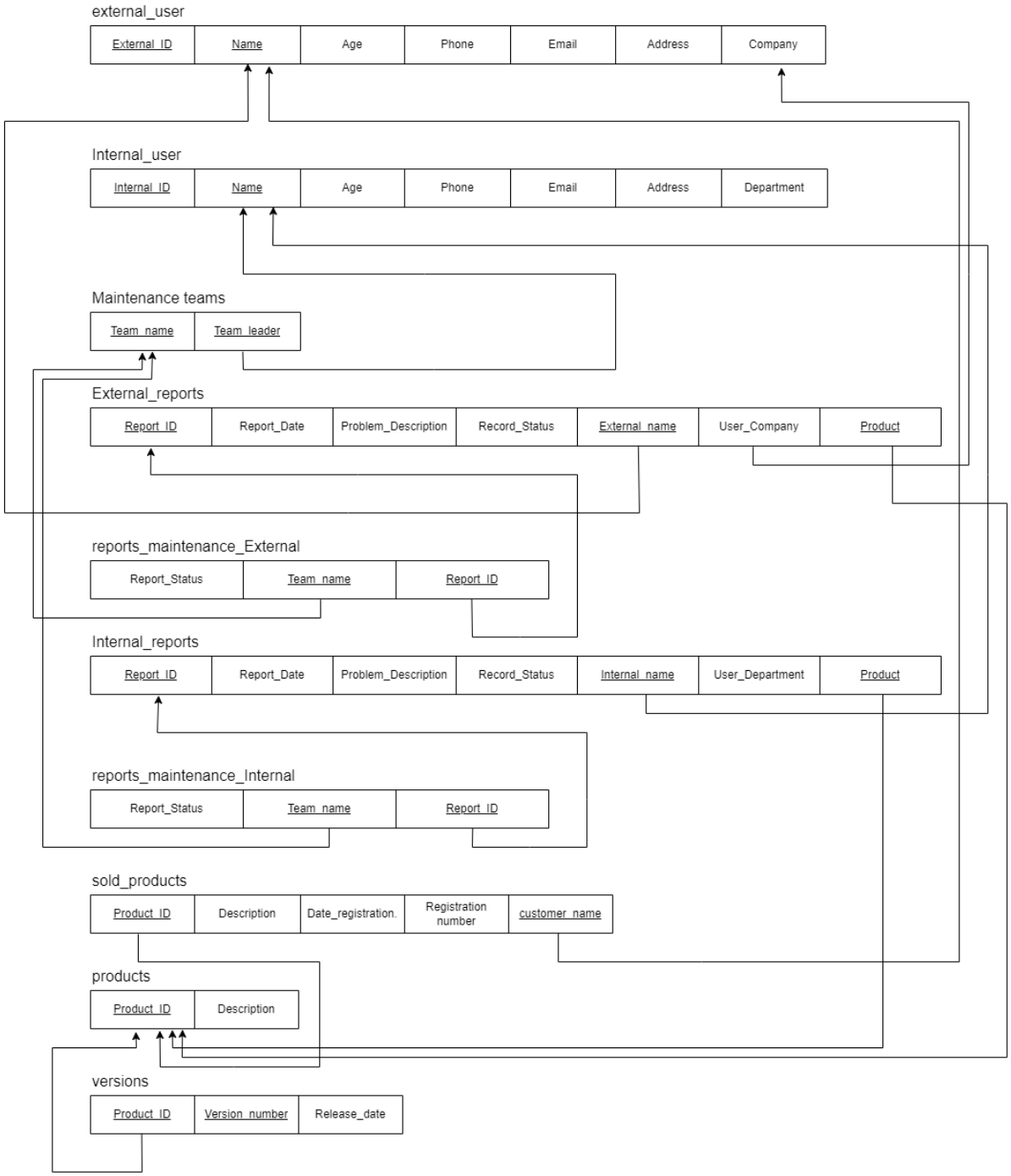
- Integrated Data Store (IDS): is a repository of databases from many administrative systems that may be stored and accessed from a single location. The IDS presently comprises data from the Mecklenburg County Register of Deeds (ROD) and the Land Use and Environmental Services Agency's Property Assessment and Land Records Management (PALRM) (LUESA).

Database Design

Entity Relationship Diagram (ERD) (Logical Model)

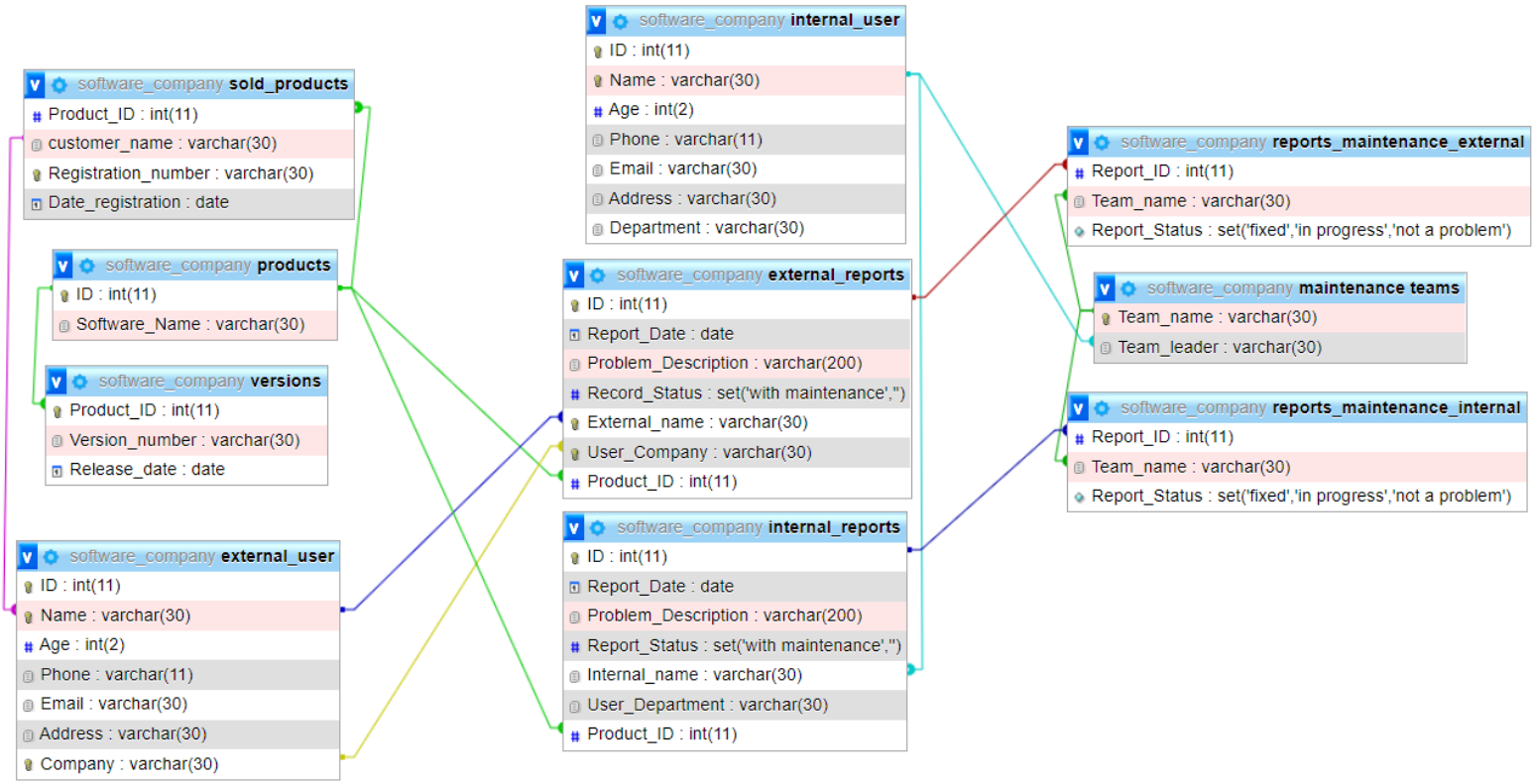


Relational Schema Mapping



Database Development

ERD for the database without the relationships between the entities and **without** optimization and normalization



Tables (Entities)

External Users Table

To store all the data for any external user with her/his corresponding company

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(11)			No	None			Change Drop More
2	Name	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
3	Age	int(2)			No	None			Change Drop More
4	Phone	varchar(11)	utf8mb4_general_ci		No	None			Change Drop More
5	Email	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
6	Address	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
7	Company	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More

Internal Users Table

To store all the data for any internal user in the company with her/his department

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(11)			No	None			Change Drop More
2	Name	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
3	Age	int(2)			No	None			Change Drop More
4	Phone	varchar(11)	utf8mb4_general_ci		No	None			Change Drop More
5	Email	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
6	Address	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
7	Department	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More

Maintenance Department Table

This is a relationship created to specify the maintenance teams and their leaders to handle the reports for the product

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	Team_name	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
2	Team_leader	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More

Products Table

To store all the data for all the company software products

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(11)			No	None			Change Drop More
2	Software_Name	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More

Versions Table (Standalone Entity)

To store all the data for all the company software products versions, so the development team can keep track with the implementation process

There is a no direct relation between the Products and versions as every product have versions and the deployment team can add the products that is released to the product table manually

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	Product_ID	int(11)			No	None			Change Drop More
2	Version_number	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
3	Status	set('released', 'under development')	utf8mb4_general_ci		No	None			Change Drop More
4	Date_Released	date			No	None			Change Drop More

Submissions (standalone Entity)

This table is for the use of the users (external/internal) to submit the reports so the quality team will check them and then move them to maintenance

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	Name	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
2	Problem_Description	varchar(200)	utf8mb4_general_ci		No	None			Change Drop More
3	Product_ID	int(11)			No	None			Change Drop More
4	Version	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
5	Submit_Date	date			No	None			Change Drop More
6	Company_Department	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More

Sold products Table

In this table the sales team record every purchase from any external user

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	Product_ID	int(11)			No	None			Change Drop More
2	customer_name	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
3	Registration_number	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
4	Date_registration	date			No	None			Change Drop More

External Reports Table

Table to store all the data for reporting external users that submitted a problem with the purchase products

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(11)			No	None			Change Drop More
2	Report_Date	date			No	None			Change Drop More
3	Problem_Description	varchar(200)	utf8mb4_general_ci		No	None			Change Drop More
4	Record_Status	set('with maintenance', '')	utf8mb4_general_ci		No	None			Change Drop More
5	External_name	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
6	User_Company	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
7	Product_ID	int(11)			No	None			Change Drop More

External Reports under maintenance Table

This table is for maintenance team to deal with the problems and fix them and put status for every report

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	Report_ID	int(11)			No	None			Change Drop More
2	Team_name	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
3	Report_Status	set('fixed', 'in progress', 'not a problem')	utf8mb4_general_ci		No	None			Change Drop More

Internal Reports Table

Table to store all the data for reporting internal users that submitted a problem with any product that they are using or testing

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID	int(11)			No	None			Change Drop More
2	Report_Date	date			No	None			Change Drop More
3	Problem_Description	varchar(200)	utf8mb4_general_ci		No	None			Change Drop More
4	Record_Status	set('with maintenance', '')	utf8mb4_general_ci		No	None			Change Drop More
5	External_name	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
6	User_Company	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
7	Product_ID	int(11)			No	None			Change Drop More

Internal Reports under maintenance Table

This table is for maintenance team to deal with the problems and fix them and put status for every report but for internal reports

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	Report_ID	int(11)			No	None			Change Drop More
2	Team_name	varchar(30)	utf8mb4_general_ci		No	None			Change Drop More
3	Report_Status	set('fixed', 'in progress', 'not a problem')	utf8mb4_general_ci		No	None			Change Drop More

Database Normalization

1NF (First Normal Form)

Rules:

- Each table cell should contain a single value.
- Each record needs to be unique.

Implementation:

Every cell has a single value and there is no duplication in the table's values, examples:

`external_reports`

ID	Report_Date	Problem_Description	Record_Status	External_name	User_Company	Product_ID
8010	2022-01-11	Windows Displaying Blue Screen	with maintenance	Michell	Weiss Spirt & Guyer	9916
8011	2022-02-20	Slow Downloading and Uploading	with maintenance	Dewitt	In Communications Inc	9910

`internal_user`

ID	Name	Age	Phone	Email	Address	Department
20972	Pok	64	01866248660	Pok@localhost.com	Birmingham	Maintenance
20608	Karma	26	01857864722	Karma@localhost.com	West Midlands	Maintenance
20530	Hani	41	01400269033	Hani@localhost.com	Birmingham	Maintenance

2NF (Second Normal Form)

Rule:

Single Column Primary Key that does not functionally dependant on any subset of candidate key relation

Implementation:

All the tables achieve this rule

3NF (Third Normal Form)

If a relation is in 2NF and no non key attribute is transitively reliant on the primary key, it is in third normal form.

`external_reports`

ID	Report_Date	Product_ID	Problem_Description	Record_Status	External_name	User_Company
8010	2022-01-11	9916	Windows Displaying Blue Screen	with maintenance	Michell	Weiss Spirt & Guyer
8011	2022-02-20	9910	Slow Downloading and Uploading	with maintenance	Dewitt	In Communications Inc
8012	2022-05-13	9910	Lack of a plan		Evan	K & R Associates Inc
8013	2022-05-02	9911	Incomplete documentation		Corrinne	In Communications Inc
8014	2022-03-15	9997	Malware Attack	with maintenance	Lura	Industrial Engineering Assocs
8015	2022-04-30	9941	Incorrect calculations		Charlesetta	Cain

ID	Product_ID	Report_Date	Problem_Description	Report_Status	Internal_name	User_Department
7010	9941	2022-02-11	Corrupt Drivers	with maintenance	Peter	Management
7011	9947	2022-02-17	Inability to Access Email	with maintenance	Octavio	Management
7012	9919	2022-04-02	New Applications Don't Install		Martha	Management
7013	9947	2022-03-02	Outdated systems	with maintenance	mee	Management
7014	9919	2022-05-15	Full understanding of requirements		Peter	Management

Note:

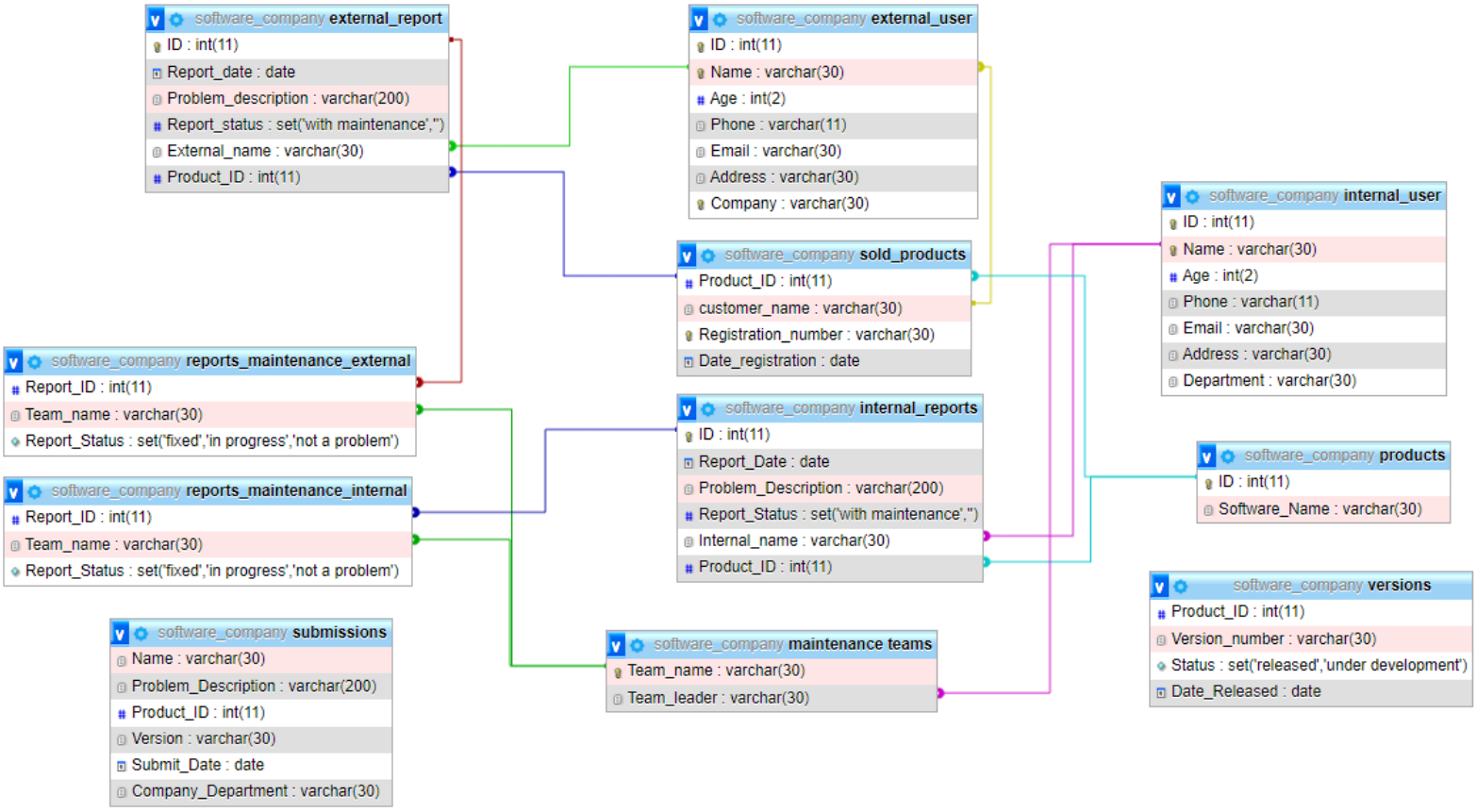
User_company is fully dependent on External user name so the normalization was that the column will drop because its already in the user details inside table (External_User) and the same as Department in internal reports

Solutions:

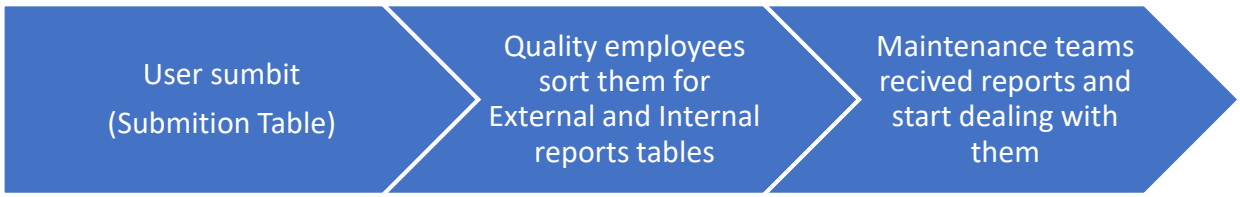
ID	Report_date	Problem_description	Report_status	External_name	Product_ID
8010	2022-01-11	Windows Displaying Blue Screen	with maintenance	Michell	9916
8011	2022-02-20	Slow Downloading and Uploading	with maintenance	Dewitt	9910
8012	2022-05-13	Lack of a plan		Evan	9910
8013	2022-05-02	Incomplete documentation		Corrinne	9911
8014	2022-03-15	Malware Attack	with maintenance	Lura	9997
8015	2022-04-30	Incorrect calculations		Charlesetta	9941

ID	Report_Date	Problem_Description	Report_Status	Internal_name	Product_ID
7010	2022-02-11	Corrupt Drivers	with maintenance	Peter	9941
7011	2022-02-17	Inability to Access Email	with maintenance	Octavio	9947
7012	2022-04-02	New Applications Don't Install		Martha	9919
7013	2022-03-02	Outdated systems	with maintenance	mee	9947
7014	2022-05-15	Full understanding of requirements		Peter	9919

Final ERD Design and Relations



Workflow



Queries scenarios

- Find the total number of software purchases

```
SELECT COUNT(Product_ID) AS Total_orders FROM sold_products;
```

Total_orders
39

- Find the best seller product

```
SELECT sold_products.Product_ID, COUNT(Product_ID) AS `Best_seller` FROM sold_produ
cts GROUP BY Product_ID ORDER BY `Best_seller` DESC LIMIT 1;
```

Product_ID	Best_seller
9941	12

- Find the number of employees that are above 60 for retirement options

```
SELECT internal_user.Name, internal_user.Age FROM internal_user WHERE

internal_user.Age >= 60;
```

Name	Age
Luis	65
Reed	63
Vicente	60
Pok	64

- Find the last release date for the latest product

```
SELECT versions.Product_ID, versions.Version_number, max(versions.Date_Released)
FROM versions

LIMIT 1;
```

Product_ID	Version_number	max(versions.Date_Released)
9919	7.8	2022-03-22

- Find the city that has the most users

```
SELECT external_user.Address, COUNT(address) FROM external_user GROUP BY external_u
ser.Address DESC LIMIT 1;
```

Address	COUNT(address)
West Midlands	3

- Find the latest external report date and the username with the company

```
SELECT external_report.External_name,external_user.Company,
max(external_report.Report_date) FROM external_user,external_report;
```

External_name	Company	max(external_report.Report_date)
Michell	Alan D Rosenburg Cpa Pc	2022-05-13

- Find out how many reports in total

```
CREATE VIEW All_Reports AS
    select external_report.ID from external_report union all
       select internal_reports.ID from internal_reports;

SELECT COUNT(all_reports.ID) FROM all_reports
```

COUNT(all_reports.ID)
11

- Find the number of employees in quality ac

```
Select COUNT(internal_user.ID) FROM internal_user WHERE internal_user.Department =
'Quality AC';
```

COUNT(internal_user.ID)
5

- Find the orders that are in 2022

```
SELECT customer_name,Product_ID , Date_registration FROM sold_products

WHERE Date_registration >= '2022-01-01';
```

customer_name ▲ 1	Product_ID	Date_registration
Charisse	9941	2022-04-11
Charlesetta	9941	2022-05-05
Dewitt	9910	2022-05-15
Dewitt	9960	2022-03-13
Dewitt	9919	2022-05-06
Dewitt	9910	2022-03-09
Edgar	9960	2022-02-24
Eric	9947	2022-01-24
Fernanda	9941	2022-02-04
France	9911	2022-03-07
France	9941	2022-02-24
Michell	9982	2022-04-04
Niesha	9960	2022-02-19
Tyisha	9910	2022-02-02
Ulysses	9941	2022-02-08
Yuette	9960	2022-01-05

- Find the name of the users that purchased the software “9982” and age below 50

```
SELECT DISTINCT sold_products.customer_name, external_user.Age FROM external_user,
sold_products

WHERE sold_products.Product_ID = 9982 AND external_user.Age < 50 and
external_user.Name = sold_products.customer_name;
```

customer_name	Age
Yuette	43

- Find the name of the users that purchased the software” Zamit” in 2022

```
CREATE VIEW User_orders AS SELECT sold_products.customer_name,
sold_products.Date_registration, products.Software_Name FROM  sold_products, products
WHERE sold_products.Product_ID = products.ID;

SELECT user_orders.customer_name , user_orders.Date_registration
,user_orders.Software_Name  FROM user_orders

WHERE user_orders.Date_registration >= '2022-01-01' and

user_orders.Software_Name = 'Zamit';
```

customer_name	Date_registration	Software_Name
Dewitt	2022-05-15	Zamit
Dewitt	2022-03-09	Zamit
Tyisha	2022-02-02	Zamit

Security Design and implementation

In this software company system Security levels will consist of three categories in this software company system

Roles to be created:

1- Admins (management department)
2- Employees (internal users exclude management)
3- Users (external users)

Creating roles

```
CREATE ROLE admin, employees, users;
```

Creating users

External users to submit reports after purchasing products

```
CREATE USER 'Aleshia'@'localhost', 'Evan'@'localhost', 'France'@'localhost', 'Ulyss
es'@'localhost', 'Tyisha'@'localhost', 'Eric'@'localhost', 'Marg'@'localhost', 'Laq
uita'@'localhost', 'Lura'@'localhost', 'Yuette'@'localhost', 'Fernanda'@'localhost'
, 'Charlesetta'@'localhost', 'Corrinne'@'localhost', 'Niesha'@'localhost', 'Rueben'
@'localhost', 'Michell'@'localhost', 'Edga’@'localhost';
```

Internal users to manipulate data inside the tables

```
CREATE USER
'Tamesha'@'localhost','Tess'@'localhost','Leonard'@'localhost','Svetlana'@'
localhost','Pok'@'localhost','Augustine'@'localhost','Karma'@'localhost','R
eed'@'localhost','Hani'@'localhost','Milly'@'localhost','Luis'@'localhost',
'Ciara'@'localhost','Alethea'@'localhost','Margurite'@'localhost','Vernice'
@'localhost','Vicente'@'localhost';
```

Managment users to manipulate data inside the tables

```
CREATE USER 'Mee'@'localhost', 'Peter'@'localhost', 'Octavio'@'localhost', 'Martha'
@'localhost';
```

Assigning roles privileges

- `GRANT ALL on software_company.* TO admin;`
- `GRANT CREATE, ALTER, DROP, INSERT, UPDATE, DELETE, SELECT, REFERENCES on sof tware_company.* TO employees;`
- `GRANT INSERT on software_company.submissions TO users;`
-

Assigning users to roles

- ```
  GRANT users to 'Aleshia'@'localhost', 'Evan'@'localhost',
  'France'@'localhost', 'Ulysses'@'localhost', 'Tyisha'@'localhost',
  'Eric'@'localhost', 'Marg'@'localhost', 'Laquita'@'localhost',
  'Lura'@'localhost', 'Yuette'@'localhost', 'Fernanda'@'localhost',
  'Charlesetta'@'localhost', 'Corrinne'@'localhost', 'Niesha'@'localhost',
  'Rueben'@'localhost', 'Michell'@'localhost', 'Edga'@'localhost';
  ```

- ```
  GRANT employees TO 'Tamesha'@'localhost','Tess'@'localhost','Leonard'@'local
  host','Svetlana'@'localhost','Pok'@'localhost','Augustine'@'localhost','Karm
  a'@'localhost','Reed'@'localhost','Hani'@'localhost','Milly'@'localhost','Lu
  is'@'localhost','Ciara'@'localhost','Alethea'@'localhost','Margurite'@'local
  host','Vernice'@'localhost','Vicente'@'localhost';
  ```

- ```
  GRANT admin to 'Mee'@'localhost', 'Peter'@'localhost',
  'Octavio'@'localhost', 'Martha'@'localhost';
  ```

Conclusion

The database is stable and optimise and fully functional in terms of fixing and managing reports and keeping the data for products and orders with the ability to store all the details for external users that purchase and employees in different departments
  ```

References

- Murach's MySQL : training & reference. Author: Joel Murach. Publisher: Fresno, CA : Mike Murach & Associates, Inc., [2015].

- *Hernandez, M., 1997. Database design for mere mortals. Reading, Mass.: Addison-Wesley.*

- *C.J. Date., 2012. Database Design and Relational Theory: Normal Forms and All That Jazz (Theory in Practice). O'Reilly Media, Inc.*

- *What is a Relational Database? (n.d.). Retrieved May 20, 2022, from https://www.techtarget.com/searchdatamanagement/definition/relational-database*

- *Vicknair, C., Nan, X., Macias, M., Chen, Y., Zhao, Z., & Wilkins, D. (2010). A comparison of a graph database and a relational database: A data provenance perspective. Proceedings of the Annual Southeast Conference. https://doi.org/10.1145/1900008.1900067*

- *DBMS | Types of Databases - javatpoint. (n.d.). Retrieved May 20, 2022, from https://www.javatpoint.com/types-of-databases*