ASSIGNMENT FRONT SHEET <No.1>

Qualification	**BTEC Level 5 HND Diploma in Computing and Systems Development**		
Unit number and title	Unit 04: Database Design & Development		
Assignment due		**Assignment submitted**	
Learner's name	Tran Quang Huy	**Assessor name**	
Learner's ID	GCD18457	**Submission number**	1

Learner declaration: I certify that the work submitted for this assignment is my own and research sources are fully acknowledged.			
Learner signature	Huy	**Date**	23/02/2019

Grading grid

P1	M1	D1

Assignment title	**Understand databases and data management systems**

In this assignment, you will have opportunities to provide evidence against the following criteria.
Indicate the page numbers where the evidence can be found.

Assessment criteria	Expected evidence	Task no.	Assessor's Feedback
LO1 Use an appropriate design tool to design a relational database system for a substantial problem			
P1 Design a relational database system using appropriate design tools and techniques, containing at least four interrelated tables, with clear statements of user and system requirements.	- An ERD clearly shows the complete logical design for the given scenario - Write the normalization statement for each of the entity	1	
M1 Produce a comprehensive design for a fully functional system which includes interface and output designs, data validations and data normalisation.	- Comprehensive design for a fully functional system - Data validations and data normalisation	1	
D1 Assess the effectiveness of the design in relation to user and system requirements.		1	

Summative feedback			
Assessor's Signature		Date	

Contents

LIST OF TABLE

LIST OF FIGURES

INTRODUCTION

Use an appropriate design tool to design a relational database system for substantial problem. Base on Assignment Brief that ElectroShop company sell electronic devices and they want to create a database to store system.

This system will store the store's necessary data, which are: Customer Information, Seller (Employee) Information, Product & Provider Information, Invoice Information

Designed by using Relational Database and 3NF Normalization Process, this System will provide a fully functional Database which includes: Data validations, Data normalization Data interfaces for different situations, Data modification, Calculate Seller's (Employee) bonus

In the beginning, ElectroShop has a small database system and is limited to paperwork by getting information from this Invoice:

ELECTROSHOP

PURCHASE INVOICE

Date:

InvoiceID:

CustomerID: SellerID:

Customer name:

Customer address:

Customer phone number:

Customer E-mail:

STT	Product name	Supplier name	ProductID	Calculation Unit	Sold quantity	Price	Total
A	B	C	D	E	1	2	3
1							
2							
3							
4							
5							
In total (by words):							

Seller **Customer**

Figure 1 - Purchase invoice ElectroShop

PART1: TOOLINGS

1. SQL Server

SQL (Structured Query Language) or structured query language Is a computer language type.

- Popular to create, edit and retrieve data from a relational database management system.
- The development of SQL goes far beyond the original purpose of servicing the object-relational database management systems.
- SQL is an ANSI / ISO standard

Microsoft SQL Server is a relational database management system developed by Microsoft. As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications—which may run either on the same computer or on another computer across a network (including the Internet). (Wikipedia, n.d.)

- SQL Server provides scripts for data inquiry tasks such as:
    - Insert, delete and update rows in 1 relationship
    - Journal, add, delete and modify objects in the database.
    - Control access to databases and objects of the database to ensure confidentiality, consistency and binding of databases.
    - Objects of SQL server are data tables with columns and rows. The column is called the data field and the row is the table record. The data column and the specified data type make up the structure of the table. When a table is organized into a system for a specific use of a job, it becomes a database.

2. SQL Server Management Studio:

- **SQL Server Management Studio** (SSMS) is a software application first launched with Microsoft SQL Server 2005 that is used for configuring, managing, and administering all components within Microsoft SQL Server. The tool includes both script editors and graphical tools which work with objects and features of the server. (Wikipedia, n.d.)
- Microsoft SQL Server Management Studio is an intuitive tool for managing SQL Server. With SQL Server Management Studio we can perform database interactions with commands or on the user interface. SQL Server Management Studio is designed to be simple and easy to use.
- Database in SQL: Currently, the collection of information in the real world to create a shared database (database), related to a certain object, industry, organization has become popular in life. Database used to organize and retrieve necessary information, a way to maximize support for business management, personnel, points, etc.
- A Database is a collection of lots of data that reflect the real world or part of the real world. Structured, archived follow rules based on mathematical theory. The data in Database is related to a specific field, specially organized for storing, searching and extracting data. Exploited by the Database Systems process, search, search, modify, add or remove data in Database.
- At the logical level, a DATABASE consists of multiple tables (TABLE), each defined by a name, a table containing structured data and constraints (CONSTRAINT) defined on the tables. In addition, Database also has a view (VIEW), procedures / functions ....
- At the physical level, the DATABASE of SQL Server is stored under 3 file types:
    - Data file (Data-file): consists of a main data storage file (*.mdf) containing initial data and secondary data files (*.ndf) containing data generated or not Save all in the main archive file.
    - The operation log file (*.ldf) contains transaction information, often used to restore Database if a problem occurs.
- Like accessing and opening common files, you need to create an archive file before retrieving. Similarly, you need to create a DATABASE to store data in SQL Server for future querying. SQL Server will help you manage and retrieve these data in a more structured and easy way.

3. System design method:

- Using draw.io: Application to draw the detailed Logical Design of ElectroShop's Database
- Develop the Physical Design and also functioned on Microsoft SQL Server Management Studio

PART2: DATABASE DESIGN

1. Database system overview

- The main database components of Electroshop:
    - Customers information
    - Salesperson information
    - Invoice order
    - Items information
    - Supplier information
- This relational Database system will be able to:
    - Create comparative reports.
    - Calculate:
        - Profit of Electroshop.
        - Salesperson's bonus each month
    - Manage and control data.

2. Database system architecture:

2.1. Normalization Process:

Base on ElectroShop's requirements that make Database normalization is the process of structuring a relational database in accordance with a series of so-called normal forms in order to reduce data redundancy and improve data integrity.

- Upon consideration, the UNF table will necessary fields is as followed:

UNF
customer_name
customer_mail
customer_address
customer_zipcode
customer_phone
salesperson_name
salesperson_age
salesperson_mail
salesperson_phone
salesperson_adress
order_day
order_quantity
item_kind
item_price
supplier_name
supplier_mail
supplier_adress
supplier_phone

CustomerName	...	SalespersonName	...	Orderday	Order Quanttity	KindOfItem	Price
Trần Quang Huy		Nguyễn Hà My		16/01/2018	1	Ipad	300
Trần Quang Huy		Nguyễn Hà My		16/01/2018	1	Iphone	400
Nguyễn Quang Ngọc		Trương Văn Hiếu		15/03/2018	2	Television	300
Nguyễn Quang Thắng		Nguyễn Anh Tuấn		20/06/2018	3	Laptop	500
Nguyễn Quang Ngọc		Nguyễn Công Phượng		14/10/2018	2	Mobilephone	300

Table 1 - UNF table

- The first normal form (1NF):

  Entities that can appear multiple times when considered by another entity should be separated into a new table. In this case, one customer can order many items and the same kind of items can also be ordered by different customers. So, customers and items cannot be single dependent on each other and become separated and dependent on the **order_id**.

1NF

order_id
order_day
order_quantity
customer_name
customer_mail
customer_address
customer_zipcode
customer_phone
salesperson_name
salesperson_age
salesperson_mail
salesperson_phone
salesperson_adress

Order ID	Quantity	Customer Name	Salesperson Name
1	2	Trần Quang Huy	Nguyễn Hà My
2	2	Trần Quang Huy	Nguyễn Hà My
3	3	Nguyễn Quang Ngọc	Nguyễn Công Phượng
4	4	Nguyễn Quang Thắng	Nguyễn Anh Tuấn

order_id
item_kind
item_price
supplier_name
supplier_mail
supplier_adress
supplier_phone

Order ID	Kind Of Items	Price	Supplier Name
1	Ipad	300	HCT
2	Iphone	400	Zero9
3	Television	300	ST MTP
4	Laptop	500	XV

Table 2 - The first normal form (1NF):

- The second normal form (2NF):

  After 1NF, all the entities have to be function-dependent on the keys. Base on 1NF that kind of item, number of items, items name, items price and supplier information are not dependent on the order details and belong to the item_id. So, it should make sense that items information be separated into a new set. The 2NF form will be:

2NF

order_id
order_day
order_quantity
customer_name
customer_mail
customer_address
customer_zipcode
customer_phone
salesperson_name
salesperson_age
salesperson_mail
salesperson_phone
salesperson_adress

Order ID	Quantity	Customer Name	Salesperson Name
1	2	Trần Quang Huy	Nguyễn Hà My
2	2	Trần Quang Huy	Nguyễn Hà My
3	3	Nguyễn Quang Ngọc	Nguyễn Công Phượng
4	4	Nguyễn Quang Thắng	Nguyễn Anh Tuấn

order_id
item_id
item_name
item_price

Order ID	Kind Of Items	Price	Supplier Name
1	Ipad	300	HCT
2	Iphone	400	Zero9
3	Television	300	ST MTP
4	Laptop	500	XV

item_id
item_name
item_price
supplier_name
supplier_mail
supplier_adress
supplier_phone

Item ID	Item	Price	Supplier Name
2	Ipad	300	HCT
3	Iphone	400	Zero9
4	Television	300	ST MTP
5	Laptop	500	XV

Table 3 - The second normal form (2NF):

- The third normal form (3NF):

  In this step, the Database system's requirement additional steps should be taken so that no attribute can be transitively dependent on the primary key. That means they cannot be dependent on a non-primary key attribute in the same table.

  Upon inspection, a name of supplier is actually retrievable from supplier information and similarly to item information, customer information and salesperson information are dependent on item_id, customer_id, salesperson_id.

  One customer can buy many items and one item can be bought by many customers so the Database system need to have 1 table order_detail include item_id, order_id and order_quantity to solve that problem.

The 3NF Form:

3NF

order_id
order_day
customer_id
salesperson_id

Order ID	Order Day	Customer ID	Salesperson ID
1	16/01/2018	1	3
2	16/01/2018	2	4
3	15/03/2018	3	5
4	20/06/2018	4	6

item_id
order_id
order_quantity

Item ID	Order ID	Order Quantity
1	2	2
2	3	2
3	4	3
4	5	4

item_id
item_kind
item_price

Item ID	Kind of Item	Price
1	Ipad	300
2	Iphone	400
3	Television	300
4	Laptop	500

supplier_id
supplier_name
supplier_mail
supplier_adress
supplier_phone

Supplier ID	Supplier Name	Supplier mail	Supplier Address	Supplier Phone
1	HCT	qwe@gmail.com	23 HCM	1234578
2	Zero9	rty@gmail.com	22 Hue	45786541
3	ST MTP	yui@gmail.com	54 HN	12458212
4	XV	iop@gmail.com	87 Dong Ha	2134578

customer_id
customer_name
customer_mail
customer_adress
customer_zipcode
customer_phone

Customer ID	Customer Name	Customer Mail	Customer Phone
1	Trần Quang Huy	qweas@gmail.com	1231234578
2	Nguyễn Quang Ngọc	rtyasd@gmail.com	45345786541
3	Nguyễn Quang Thắng	yuxzxi@gmail.com	1244258212
4	Phạm Nhật Vượng	iopdas@gmail.com	21423434578

salesperson_id
salesperson_name
salesperson_age
salesperson_mail
salesperson_phone
salesperson_adress

Salesperson ID	Salesperson Name	Salesperson Mail	Salesperson Phone
1	Nguyễn Hà My	qweas1@gmail.com	31234578
2	Nguyễn Công Phượng	rtyasd2@gmail.com	453456541
3	Trương Văn Hiếu	yuxzx3i@gmail.com	44258212
4	Nguyễn Anh Tuấn	iopda4s@gmail.com	214234578

Table 4 - The third normal form (3NF):

Conclusively, the Database system will have:

UNF	1NF	2NF	3NF
customer_name customer_mail customer_address customer_zipcode customer_phone salesperson_name salesperson_age salesperson_mail salesperson_phone salesperson_adress order_day order_quantity item_kind item_price supplier_name supplier_mail supplier_adress supplier_phone	**order_id** item_kind item_price supplier_name supplier_mail supplier_adress supplier_phone	**item_id** item_name item_price supplier_name supplier_mail supplier_adress supplier_phone	**order_id** order_day customer_id salesperson_id
	order_id order_day order_quantity customer_name customer_mail customer_address customer_zipcode customer_phone salesperson_name salesperson_age salesperson_mail salesperson_phone salesperson_adress	**order_id** **item_id** item_name item_price	**item_id** **order_id** order_quantity
		order_id order_day order_quantity customer_name customer_mail customer_address customer_zipcode customer_phone salesperson_name salesperson_age salesperson_mail salesperson_phone salesperson_adress	**item_id** item_kind item_price
			supplier_id supplier_name supplier_mail supplier_adress supplier_phone
			customer_id customer_name customer_mail customer_adress customer_zipcode customer_phone
			salesperson_id salesperson_name salesperson_age salesperson_mail salesperson_phone salesperson_adress

Table 5 - Fully Database system normal form

2.2. Entity Relationship Diagram (ER Diagram):

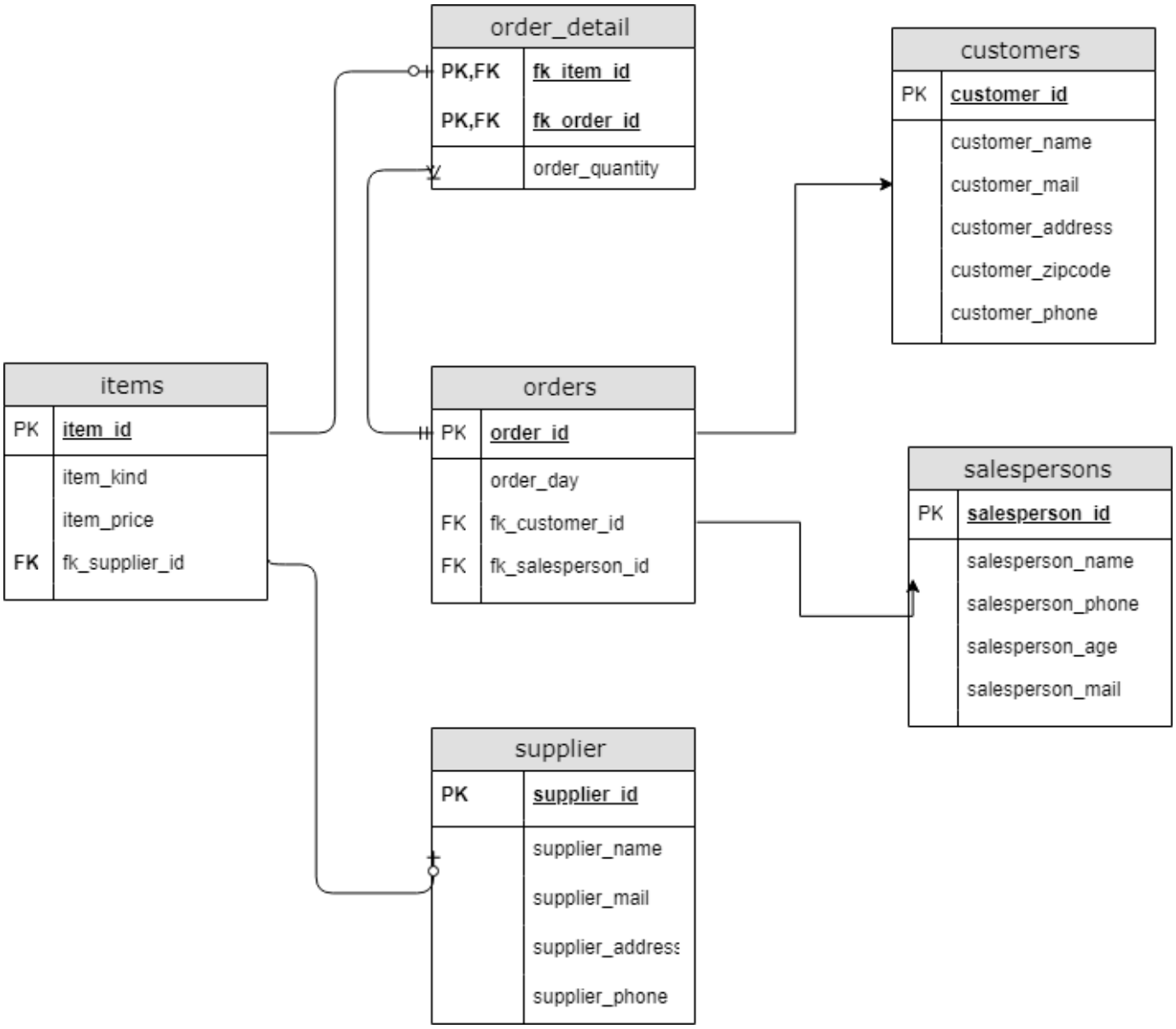


Figure 2 - EIctroShop Entity Relationship Diagram

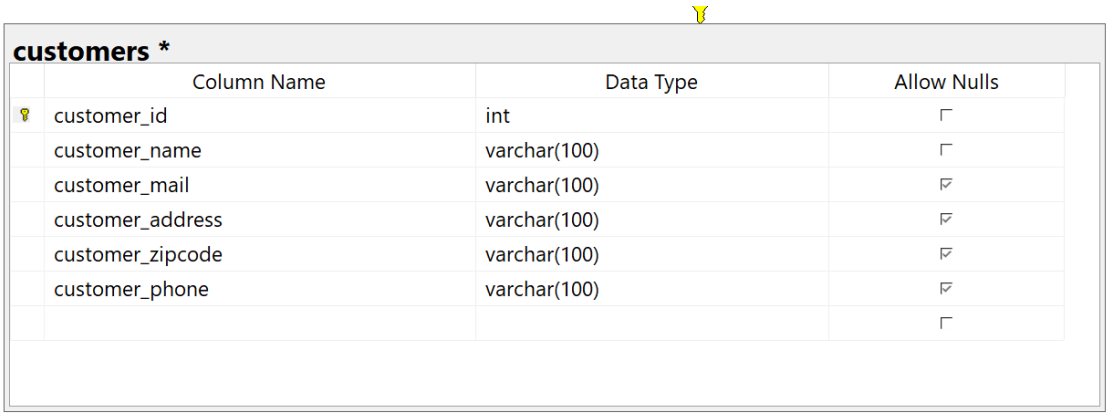
2.3. Table Descriptions:

The Database System have 6 tables, which are:

- **Customers table:**

  This table used to store information of ElectroShop's customer information, customer's email will be used to send offers, product or promotions. This table includes:

  - Customer's identity (**customer_id** - PK): This is primary key of customer table to make sure that a customer has a unique and non-duplicate ID.
  - Customer's Name (**customer_name**): Store customer name information.
  - Customer's Email (**customer_mail**): Store customer mail information and send product information as well as the completion process when customers buy products.
  - Customer's Address (**customer_address**): Store customer address information.
  - Customer's Zip code (**customer_zipcode**): Store customer zip code information.
  - Customer's Phone number (**customer_phone**): Easily contact customers when needed.



customers *

Column Name	Data Type	Allow Nulls
customer_id	int	☐
customer_name	varchar(100)	☐
customer_mail	varchar(100)	☑
customer_address	varchar(100)	☑
customer_zipcode	varchar(100)	☑
customer_phone	varchar(100)	☑
		☐

Table 6 - Customer table

	customer_id	customer_name	customer_mail	customer_address	customer_zipcode	customer_phone
1	1	Allen Ayers	posuere.cubilia.Curae@ornare.edu	294-4580 At Rd.	IC5 8NW	0819 171 8803
2	2	Dolan Lowery	sociis@nuncac.net	728 Pede, Avenue	UK7R 9NE	(0113) 072 1649
3	3	Aaron Martinez	lobortis.tellus@sempercursus.net	Ap #577-6187 Ultrices Ave	WV3 7QG	0827 814 4921
4	4	Gannon Potter	fringilla.euismod@magnaCrasconvallis.co.uk	P.O. Box 215, 4902 Nullam Av.	LD7 0JD	0845 680 8907
5	5	Stuart Mclaughlin	amet@maurissit.com	3248 Imperdiet Rd.	X6S 7JI	056 7531 6531
6	6	Elliott Cunningham	vel@arcuacorci.edu	928-9736 Semper Rd.	GW29 0YT	(014962) 24309
7	7	Clayton Compton	montes@tellus.edu	P.O. Box 208, 6020 Tortor, Rd.	A82 7XW	0845 46 48
8	8	Moses Pearson	vel.arcu.eu@augueacipsum.com	Ap #289-6765 Id St.	OO0J 3XU	07624 737901

Table 7 - Example Customer table

- Salespersons table:

This table is used to store information of ElectroShop's employee information would help to determine which salesperson is in charge of selling which order. Many orders can be sold by one salesperson so it will have many relationships. This table includes:

- Salesperson's identity (**salesperson_id** - PK): Each salesperson just only has one ID that make sure that salesperson has a unique and non-duplicate ID so this must be primary key.
- Salesperson's name (**salesperson_name**): Store salesperson name information.
- Salesperson's Age (**salesperson_age**): Store salesperson age information.
- Salesperson's phone number (**salesperson_phone**): Store salesperson phone number information.
- Salesperson's email (**salesperson_mail**): Store salesperson email information.

salespersons *

	Column Name	Data Type	Allow Nulls
🔑	salesperson_id	int	☐
	salesperson_name	varchar(100)	☐
	salesperson_phone	varchar(100)	☑
	salesperson_age	int	☑
	salesperson_mail	varchar(100)	☑
			☐

Table 9 - Salespersons table

	salesperson_id	salesperson_name	salesperson_phone	salesperson_age	salesperson_mail
1	1	Bert Washington	(016977) 5226	45	dapibus.gravida.Aliquam@nonenim.edu
2	2	Keane Rivas	07470 645211	52	velit.Aliquam.nisl@maurisid.ca
3	3	Elton Kaufman	0303 014 7073	45	non.nisi@risus.co.uk
4	4	Ferris Riggs	(01958) 800854	36	massa.Suspendisse@sedduiFusce.net
5	5	Julian Dominguez	(01089) 10436	51	ullamcorper.velit@Nulla.net
6	6	Flynn Randall	0800 1111	40	amet.ultricies@Nunccommodoauctor.org
7	7	Orlando Frank	07624 853883	33	molestie.pharetra.nibh@aceleifendvitae.co.uk
8	8	Malik Pruitt	(018750) 41060	55	nunc.Quisque@ornarelectus.ca

Table 8 - Example Salespersons table

- **Supplier table:**

  This table is used to store information of ElectroShop's supplier. This table includes:

  - Supplier's Identity (**supplier_id** - PK): This is primary key of supplier table to make sure that a supplier has a unique and non-duplicate ID.
  - Supplier's name (**supplier_name**): Store supplier name information.
  - Supplier's mail (**supplier_mail**): Store supplier mail information.
  - Supplier's address (**supplier_address**): Store supplier address information.
  - Supplier's phone number (**supplier_phone**): Store supplier phone information.

supplier

	Column Name	Data Type	Allow Nulls
🔑	supplier_id	int	☐
	supplier_name	varchar(100)	☑
	supplier_mail	varchar(100)	☑
	supplier_adress	varchar(100)	☑
	supllier_phone	varchar(100)	☑
			☐

Table 10 - Supplier table

	supplier_id	supplier_name	supplier_mail	supplier_adress	supplier_phone
1	1	Silas Brock	molestie.Sed.id@ullamcorpernislarcu.net	547-6530 Lacus, St.	0800 1111
2	2	Chadwick Daniels	ridiculus.mus.Proin@vulputate.net	P.O. Box 422, 8223 Amet Street	(017099) 57031
3	3	Aladdin Gould	ligula.Donec@Praesentinterdumligula.com	608-9434 Neque. St.	0868 860 6721
4	4	Amir Petersen	magna@iaculisquispede.org	625-6903 Aenean St.	055 5617 1458
5	5	Dominic Dale	nibh@atpedeCras.net	P.O. Box 651, 6906 A Rd.	0845 46 49
6	6	Xander Bright	adipiscing.fringilla.porttitor@Aliquam.co.uk	601-5875 Feugiat Av.	0800 1111
7	7	Mark Richard	blandit.at@odioEtiam.org	2264 Odio. St.	(01656) 945906
8	8	Nathaniel Mcdonald	a.dui@et.com	Ap #477-6616 Eu St.	(0113) 828 1200

Table 11 - Example Supplier table

- **Items table:**

  This table is used to store information of ElectroShop's orders. That also contains information about items that are being sold. The items are shown in the items table and in order details. This table includes:

  - Item identity (**item_id** – PK): This is primary key of items table to make sure that item has a unique and non-duplicate ID.
  - Kind of Items (**item_kind**): Store kind of items information.
  - Price (**item_price**): Store price of items information.
  - Supplier Identity (**fk_supplier_id**): This is foreign key to link with **supplier_id** at supplier table. This allow people can know what specific items originated.

items

	Column Name	Data Type	Allow Nulls
🔑	item_id	int	☐
	item_kind	varchar(100)	☑
	item_price	float	☑
	fk_supplier_id	int	☑
			☐

Table 12- Items table

	item_id	item_kind	item_price	fk_supplier_id
1	1	Laptop	8115	1
2	2	VGA	4202	2
3	3	Camera	7373	3
4	4	Laptop	3516	4
5	5	VGA	7119	5
6	6	CPU	2097	6
7	7	Mobliephone	1519	7
8	8	VGA	3389	8
9	9	Printer	1785	9

Table 13 - Example Items table

- **Orders table:**

  This table is used to store information of ElectroShop's orders. It is an important table to connect and reference by many others: A customer who order that items, a salesperson who sold that items, specially that connect with that Order detail table as know as invoice. This orders table includes:

  - Order's identity (**order_id** – PK): This is primary key to confirm that when customer order something this order's identity exist only and non-duplicate ID.
  - Order day (**order_day**): Store the day that customer order information.
  - Customer's identity (**fk_customer_id** - FK): This is foreign key references to **customer table.** It will allow people know information about customer is ordering.
  - Salesperson identity (**fk_salesperson_id** - FK): This is foreign key references to **supplier table**. That indicates employee who sell items to customer.

orders

	Column Name	Data Type	Allow Nulls
🔑	order_id	int	☐
	order_day	date	☑
	fk_customer_id	int	☑
	order_salesperson_id	int	☑
			☐

Table 14 - Orders table

	order_id	order_day	fk_customer_id	fk_salesperson_id
1	1	2019-09-17	1	1
2	2	2019-05-20	2	2
3	3	2019-12-19	3	3
4	4	2019-07-01	4	4
5	5	2019-01-20	5	5
6	6	2019-11-21	6	6
7	7	2018-09-17	7	7
8	8	2018-03-30	8	8

Table 15 - Example Orders table

- **Order detail table:**

  This order detail table know as invoice and this dataset stores all the invoices that are made through sale. One order will generate one invoice. Order detail table includes:

  - Item's identity (**fk_item_id** -PK, FK): This is one of two primary key with order identity that allow customer can offer more than one item. And it also foreign key references with **item_id** in **items table** to show what items customer order.
  - Order's Identity (**fk_order_id** – PK, FK): this is a last primary key with item identity that allow person can offer more than one item and foreign key references with **order_id** in **orders table** this is the main of function let customer can offer more than one item.
  - Order quantity (**order_quantity**): Store a number of each items that customer order information.

order_detail

	Column Name	Data Type	Allow Nulls
🔑	fk_item_id	int	☐
🔑	fk_order_id	int	☐
	order_quantity	int	☑
			☐

Table 16 - Order detail table

	fk_item_id	fk_order_id	order_quantity
1	1	1	8
2	2	2	4
3	3	3	8
4	4	4	9
5	5	5	10
6	6	6	8
7	7	7	10
8	8	8	6

Table 17 - Example Order detail table

2.4. Database creation and Diagram

2.4.1. Database creation code:

```
CREATE TABLE orders(
   order_id INT PRIMARY KEY,
   order_day date,
   fk_customer_id int,
   CONSTRAINT fk_customer_id
         FOREIGN KEY (fk_customer_id)
         REFERENCES customers (customer_id),
   order_salesperson_id INT,
   CONSTRAINT fk_order_salesperson_id
         FOREIGN KEY (order_salesperson_id)
         REFERENCES salespersons (salesperson_id),
);

CREATE TABLE order_detail(
   fk_item_id INT,
   CONSTRAINT fk_item_id
         FOREIGN KEY (fk_item_id)
         REFERENCES items (item_id),

   fk_order_id int,
   CONSTRAINT fk_order_id
         FOREIGN KEY (fk_order_id)
         REFERENCES orders (order_id),
   PRIMARY KEY (fk_item_id, fk_order_id),
   order_quantity INT,
);

CREATE TABLE salespersons(
   salesperson_id INT PRIMARY KEY,
   salesperson_name VARCHAR(100),
   salesperson_phone VARCHAR(100),
   salesperson_age INT,
   salesperson_mail VARCHAR(100),
);

CREATE TABLE items(
   item_id INT PRIMARY KEY,
   item_kind VARCHAR(100),
   item_price INT,
   fk_supplier_id int,
   CONSTRAINT fk_supplier_id_items
         FOREIGN KEY (fk_supplier_id)
         REFERENCES supplier (supplier_id),
);

CREATE TABLE customers(
```

```
    customer_id INT PRIMARY KEY,
    customer_name VARCHAR(100),
    customer_mail VARCHAR(100),
    customer_address VARCHAR(100),
    customer_zipcode VARCHAR(100),
    customer_phone VARCHAR(100),
);

CREATE TABLE supplier(
    supplier_id INT PRIMARY KEY,
    supplier_name VARCHAR(100),
    supplier_mail VARCHAR(100),
    supplier_adress VARCHAR(100),
    supllier_phone VARCHAR(100),
);
```

2.4.2. Diagram

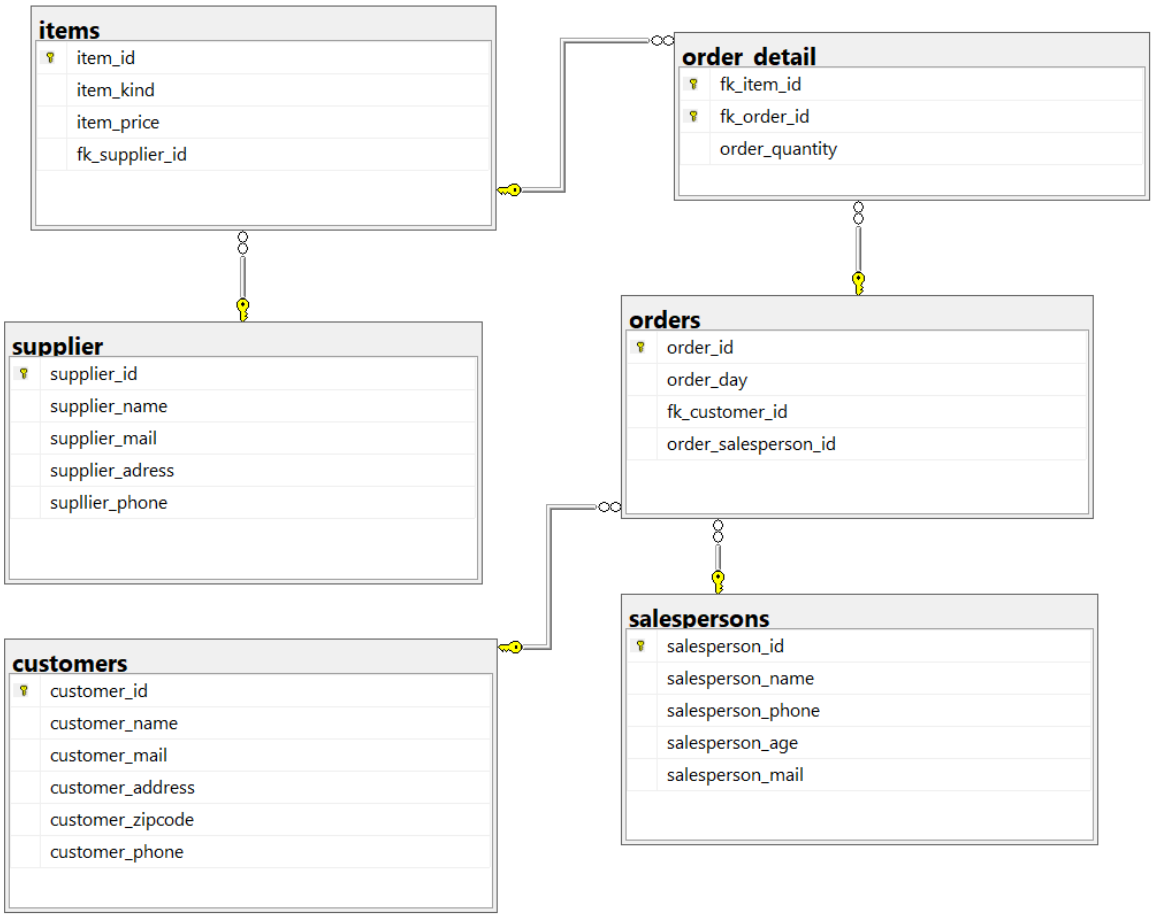


Figure 3 - Database System Diagram

PART3: QUERIES

1. Database System can sent mail to all customer:

ElectroShop would like to add all the customers to the database so that they can send a mail-shot to them with any offers that are available, as well as the catalogue which is produced annually:

```
 SELECT customer_id AS CustomerID,
      customer_name AS CustomerName,
      customer_mail AS CustomerMail
FROM customers;
```

	CustomerID	CustomerName	CustomerMail
1	1	Allen Ayers	posuere.cubilia.Curae@ornare.edu
2	2	Dolan Lowery	sociis@nuncac.net
3	3	Aaron Martinez	lobortis.tellus@sempercursus.net
4	4	Gannon Potter	fringilla.euismod@magnaCrasconvallis.co.uk
5	5	Stuart Mclaughlin	amet@maurissit.com
6	6	Elliott Cunningham	vel@arcuacorci.edu
7	7	Clayton Compton	montes@tellus.edu
8	8	Moses Pearson	vel.arcu.eu@augueacipsum.com
9	9	Reese Wheeler	sapien@Nullam.org

Table 18 - Database system - mail customer

2. List items, and min/max of price

The Database system lists the number of price in each items, sorted high to low. And show the min or max of price's items.

```
SELECT items.item_kind AS KindOfItem,
   items.item_price AS Price
FROM items
ORDER BY items.item_price DESC;
```

	KindOfItem	Price
1	Laptop	9982
2	Iphone	9926
3	Television	9879
4	Camera	9852
5	Mobliephone	9850
6	Printer	9833
7	Printer	9822
8	VGA	9576
9	Printer	9490
10	Mobliephone	9465
11	VGA	9340

Table 19 - List price each items

```sql
SELECT * FROM items
WHERE item_price = (SELECT MIN(item_price) FROM items);

SELECT * FROM items
WHERE item_price = (SELECT MAX(item_price) FROM items);
```

	item_id	item_kind	item_price	fk_supplier_id
1	73	Mobliephone	217	73

	item_id	item_kind	item_price	fk_supplier_id
1	45	Laptop	9982	45

Table 20 - Max and Min price of item

3. Report which customers buy the most goods every quarter.

ElectroShop will synthesize customers who buy the largest number of products each quarter then there will be gifts for that customer.

```sql
SELECT customers.customer_id AS CustomerID,
       customers.customer_name AS CustomerName,
       order_detail.order_quantity AS Quantity
FROM customers, order_detail, orders
WHERE (order_detail.order_quantity = (SELECT MAX(order_quantity) FROM order_detail))
       AND (order_detail.fk_order_id = orders.order_id)
       AND (orders.fk_customer_id = customers.customer_id)
       AND (orders.order_day BETWEEN '4/1/2019' AND '7/1/2019');
```

	CustomerID	CustomerName	Quantity
1	59	Lester Dodson	10
2	76	Chancellor Fleming	10

Table 21 - Which customers buy the most good ever quarter

4. Quarterly sales report

The system will show each type of product, the amount sold in a quarter and the revenue achieved by each product.

```
SELECT items.item_kind,
       SUM(order_detail.order_quantity) AS TotalQuantity,
       SUM(order_detail.order_quantity*items.item_price) AS Profit
FROM items, order_detail, orders
WHERE (orders.order_day BETWEEN '4/1/2018' AND '7/1/2018')
       AND (items.item_id = order_detail.fk_item_id)
       AND (orders.order_id = order_detail.fk_order_id)
       GROUP BY items.item_kind;
```

	item_kind	TotalQuantity	Profit
1	Camera	1	9852
2	Iphone	4	39704
3	Laptop	3	20932
4	Mobliephone	10	44991
5	Refrigerator	14	49856
6	Television	10	72675
7	VGA	2	1068

Table 22- Quarterly sales report

5. Summarize the results of sales staff each quarter

Based on the quarterly sales results of Electroshop managers will easily assess the quality of work of each employee, thereby making appropriate plans for the next quarter

```
SELECT salespersons.salesperson_name AS SalespersonName,
    items.item_kind AS KindOfItem,
    SUM(order_detail.order_quantity) AS Quantity,
    SUM(order_detail.order_quantity*items.item_price) AS Profit
FROM salespersons,items,order_detail,orders
WHERE (salespersons.salesperson_id = orders.fk_salesperson_id)
    AND (orders.order_id = order_detail.fk_order_id)
    AND (items.item_id = order_detail.fk_item_id)
    AND (orders.order_day BETWEEN '4/1/2018' AND '7/1/2018')
GROUP BY salespersons.salesperson_name, items.item_kind;
```

	SalespersonName	KindOfItem	Quantity	Profit
1	Josiah Myers	Camera	1	9852
2	Clayton Horn	Iphone	4	39704
3	Benedict Richards	Laptop	2	17318
4	Dustin Frye	Laptop	1	3614
5	Evan Ferguson	Mobliephone	3	651
6	Ian Abbott	Mobliephone	1	4104
7	Neil Rios	Mobliephone	6	40236
8	Benedict Austin	Refrigerator	8	6696
9	Chancellor Johnston	Refrigerator	1	2075

Table 23 - Summarize the results of sales staff each quarter

6. Bonuses for quarterly employees have reached the target

Each quarter ElectroShop will synthesize the number of items employees have sold, based on which employees with sales of more than $ 15,000 each quarter will receive additional bonuses.

```
SELECT salespersons.salesperson_name AS SalespersonName,
       SUM(order_detail.order_quantity*items.item_price) AS Profit
FROM salespersons,items,order_detail,orders
WHERE (salespersons.salesperson_id = orders.fk_salesperson_id)
       AND (orders.order_id = order_detail.fk_order_id)
       AND (items.item_id = order_detail.fk_item_id)
       AND (orders.order_day BETWEEN '1/1/2018' AND '4/1/2018')
       AND ((order_detail.order_quantity*items.item_price) >= 15000)
GROUP BY salespersons.salesperson_name;
```

	SalespersonName	Profit
1	Denton Eaton	49704
2	Malik Pruitt	20334

Table 24 - Bonuses for employee

7. Loyal customers every quarter

The system will indicate loyal customers each quarter corresponding to the number of purchase times at ElectroShop more than 2 times, from which ElectroShop will have better marketing plans for the future.

```
SELECT customers.customer_name AS CustomerName,
       COUNT(orders.fk_customer_id) AS TimesPurchases
FROM customers, orders
WHERE (orders.order_day BETWEEN '4/1/2019' AND '7/1/2019')
       AND (customers.customer_id = orders.fk_customer_id)
GROUP BY customers.customer_name
HAVING COUNT(orders.fk_customer_id) >=2;
```

CustomerName	TimesPurchases
Abbot Mccall	3
Brett Montoya	3
Chancellor Fleming	2
Demetrius Barber	2

Table 25- Loyal customers every quarter

PART4: CONCLUSION

1. Overview Database System

This Database was built by SQL Server management based on ElectroShop's requirement. The main function of Database system is store information about Customer, salesperson, item, product and supplier. Through 6 tables in third normal form (3NF) format that make database has each table cell should contain a single value, the record be unique and has no transitive functional dependencies with single column Primary Key.

The user can be easy:

- Retrieve data, statistics profits from sales every month, quarter or year.
- Checking information of employees, customers, suppliers, easily communicate with employees and customers when necessary, send email notifications about orders and product marketing.
- Calculation: total sales, sales, profits, salaries, bonuses for employees, etc.
- Based on this database system, it is possible to calculate the future development direction for ElectroShop such as: what kind of products should be strengthened, marketing, given programs, products to attract potential customers and increase loyal customers with ElectroShop.

2. Evaluate the responsiveness of Database with user requirements

- Business:
  ElectroShop database system meets business requirements such as:
  - Calculate total revenue, cost, profit

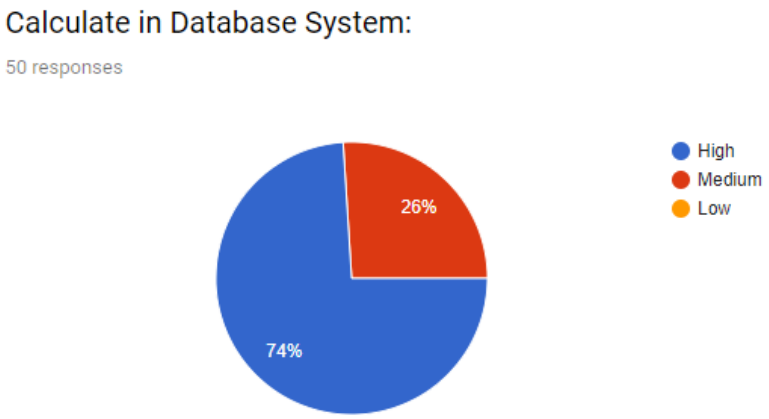


Figure 4 - Calculate in Database System

- Orienting short-term and long-term business plans:

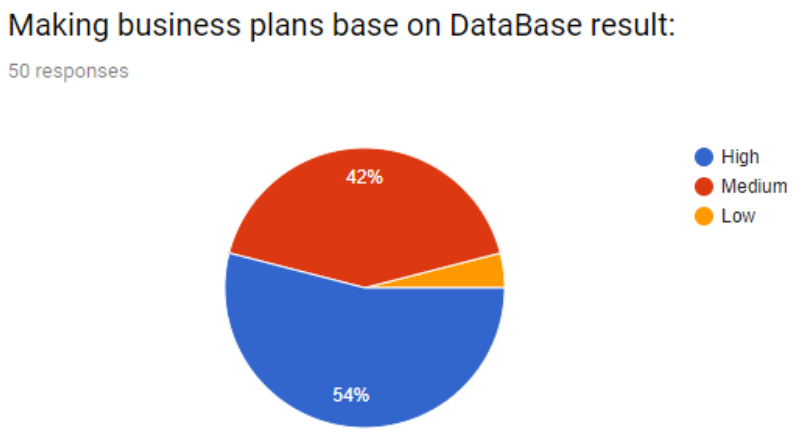


Figure 5 - Orienting short-term and long-term business plans

- Making Marketing Plans such as sent mail, phone call, sent advertisement, etc.

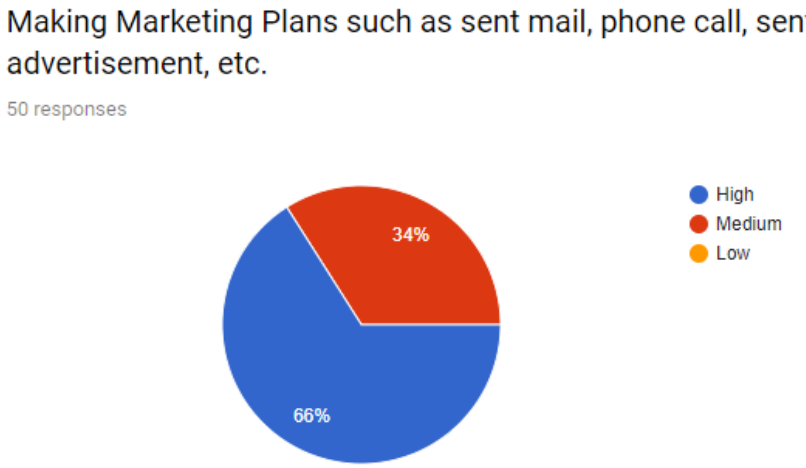


Figure 6- Making Marketing Plans

- Query Performance

  The most obvious place to look for poor query performance is in the query itself. Problems can result from queries that take too long to identify the required data or bring the data back. Look for these issues in queries:

  - Selecting more data than needed: It is not enough to write queries that return the appropriate rows; queries that return too many columns can cause slowness both in selecting the rows and retrieving the data. It is better to list the required columns rather than writing SELECT*. When the query is based on selecting specific fields, the plan may identify a covering index, which can speed up the results. A covering index includes all the fields used in the query. This means that the database can generate the results just from the index. It does not need to go to the underlying table to build the result. Additionally, listing the columns required in the result reduces the data that's transmitted, which also benefits performance.

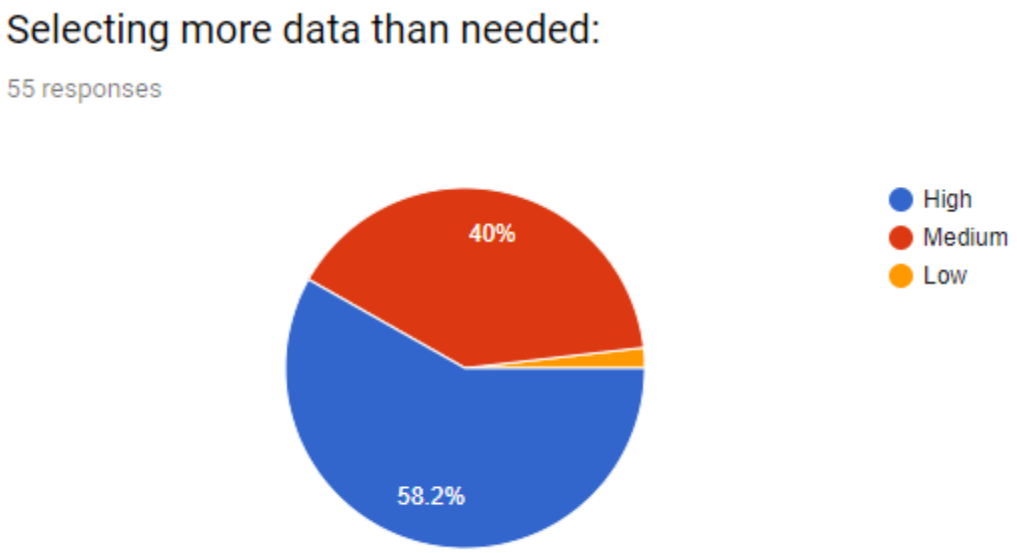


Figure 7- Selecting more data than needed

  - Inefficient joins between tables: Joins cause the database to bring multiple sets of data into memory and compare values, which can generate many database reads and significant CPU. Depending on how the tables are indexed, the join may require scanning all the rows of both tables. A poorly written join on two large tables that requires a complete scan of each one is very computationally expensive. Other factors that slow down joins include joining on columns that are different data types, requiring conversions, or a join condition that includes LIKE, which prevents the use of

indexes. Avoid defaulting to using a full outer join; use inner joins when appropriate to bring back only the desired data.

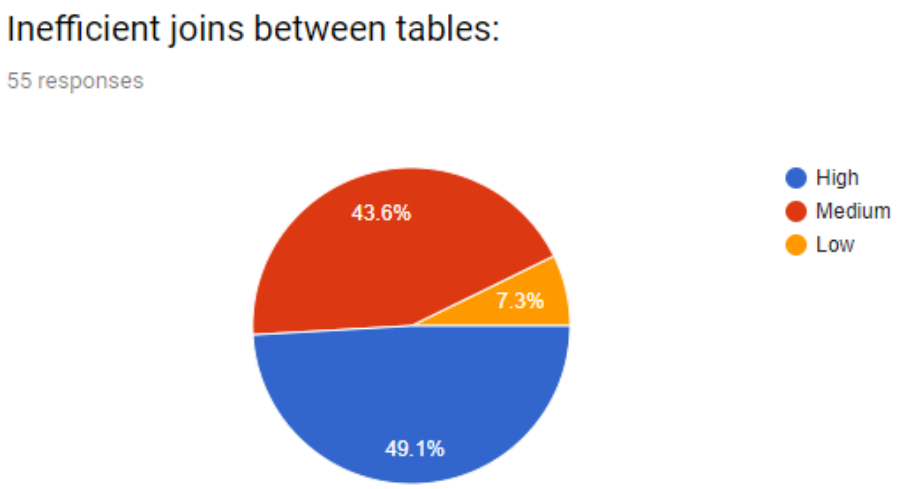


Figure 8 - Inefficient joins between tables

- Too few or too many indexes: When there aren't any indexes that the query optimizer can use, the database needs to resort to table scans to produce query results, which generates a large amount of disk input/output (I/O). Proper indexes also reduce the need for sorting results. Indexes on non-unique values do not provide as much help as unique indexes in generating results. If the keys are large, the indexes become large as well, and using them creates more disk I/O. Most indexes are intended to help the performance of data retrieval, but it is important to realize that indexes also impact the performance of data inserts and updates, as all associated indexes must be updated.

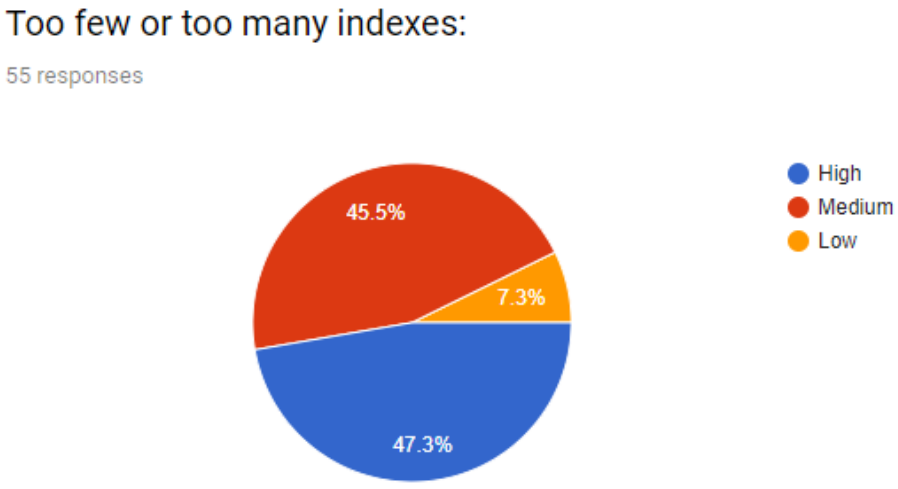


Figure 9 - Too few or too many indexes

- Too much literal SQL causing parse contention: Before any SQL query can be executed, it must be parsed, which checks syntax and permissions before generating the execution plan. Because parsing is expensive, databases save the SQL they've parsed to reuse it and eliminate the parsing time. Queries that use literal values cannot be shared, as the WHERE clauses differ. This results in each query being parsed and added to the shared pool. Because the pool has limited space, some saved queries are discarded to make room. If those queries recur, they need to be parsed again.

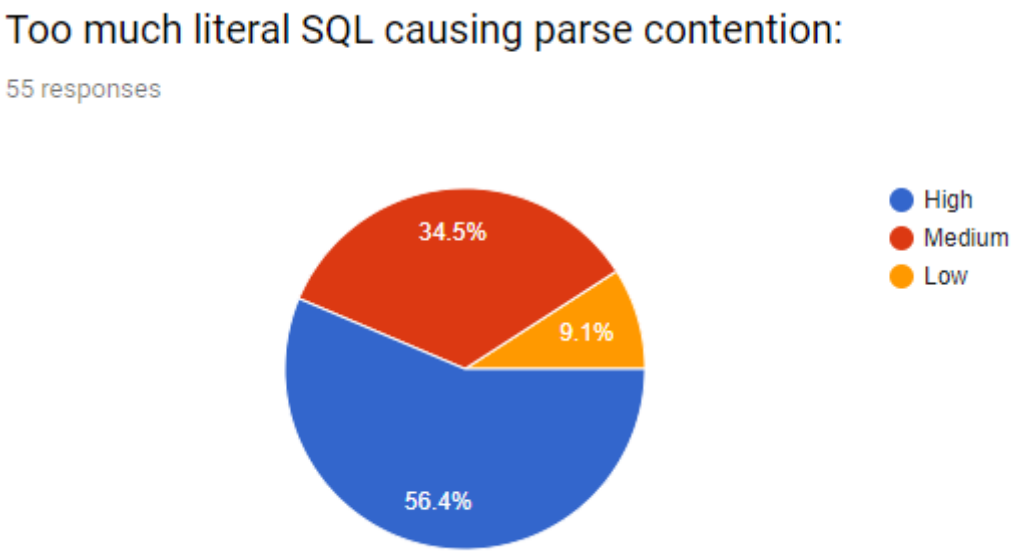


Figure 10 - Too much literal SQL causing parse contention

- User and Query Conflicts: Databases are designed to be multi-user, but the activities of multiple users can cause conflicts. (Habib, 2015)

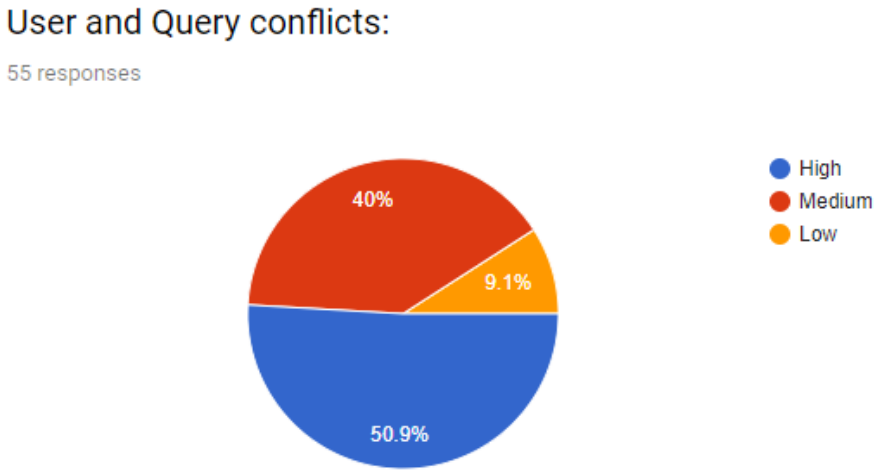


Figure 11- User and Query Conflicts

- Data:
  - Easy to import data into the system

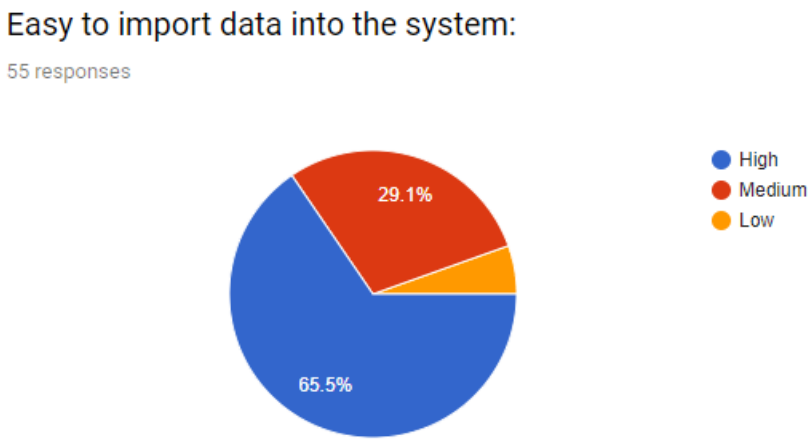


Figure 12- Easy to import data into the system

- Easily manipulate adding, editing, deleting data

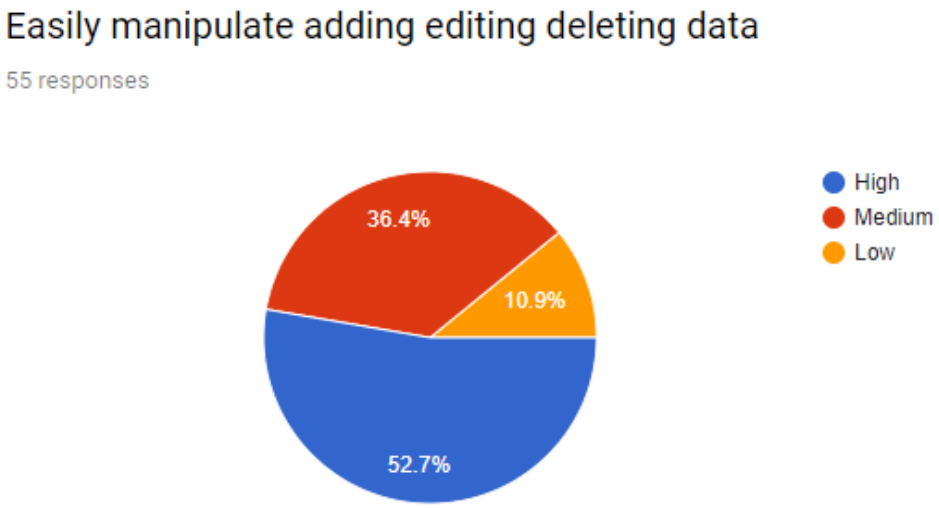


Figure 13- Easily manipulate adding, editing, deleting data

- Encryption

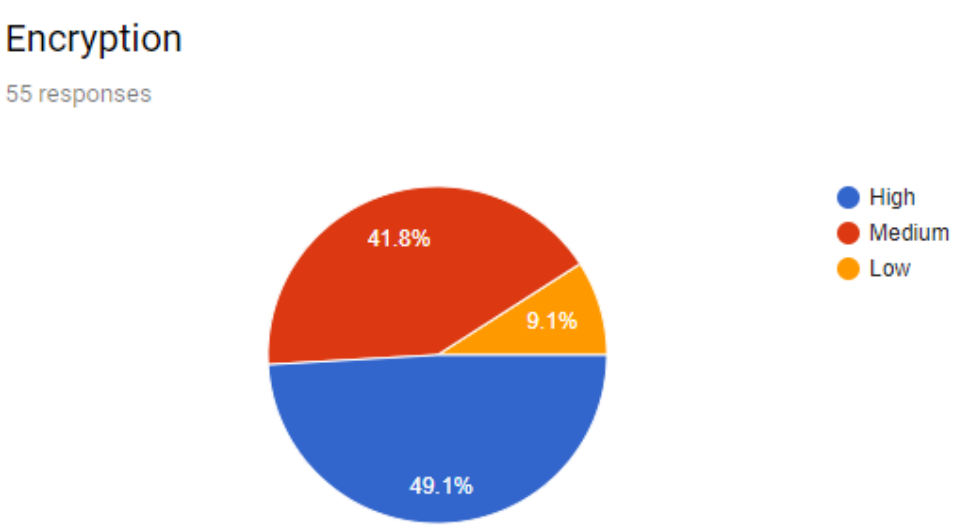


Figure 14 - Encryption

- Data saved in standard format:

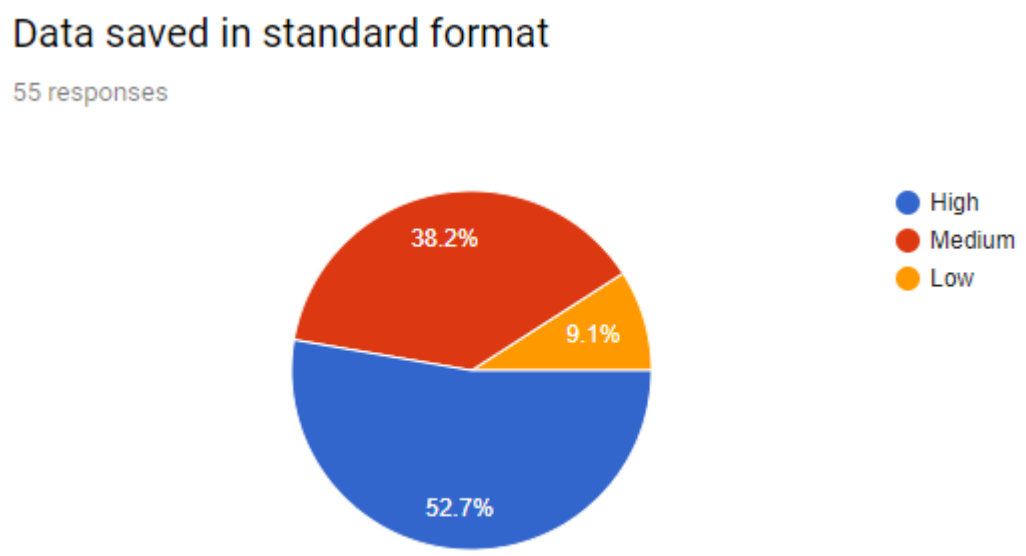


Figure 15- Data saved in standard format

- Other Issues
  - Security of Database system

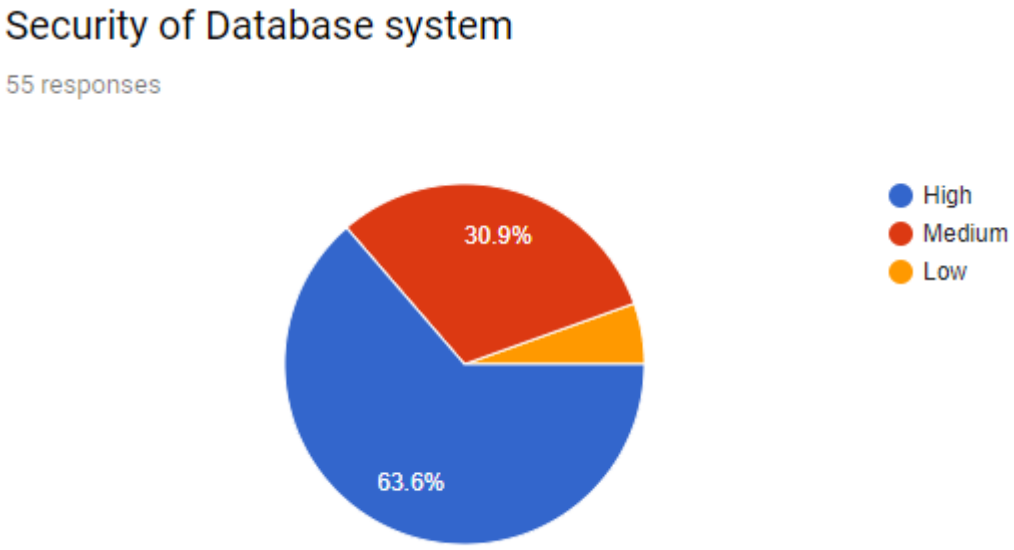


Figure 16 - Security of Database system

- Easy maintenance

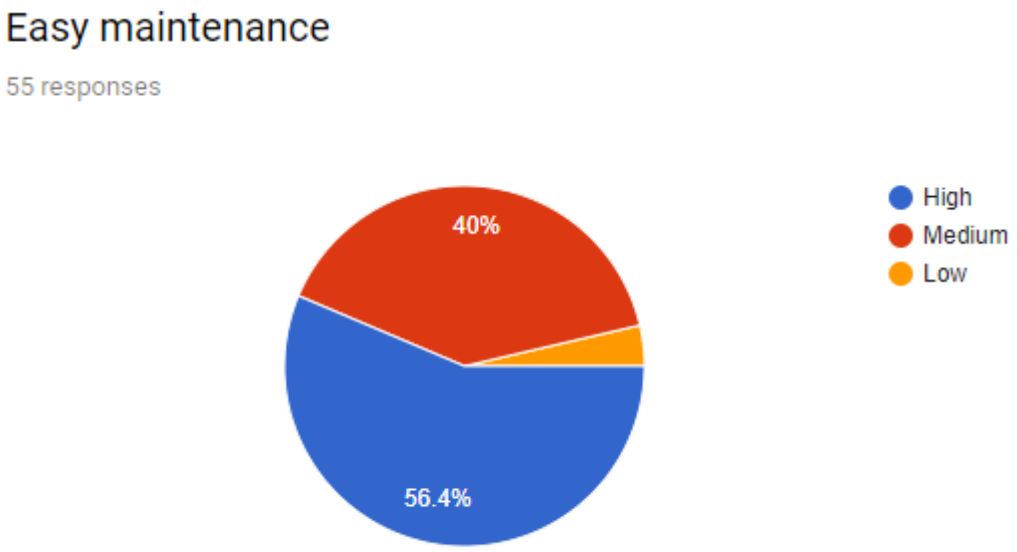


Figure 17- Easy maintenance

- Easy to upgrade the system:

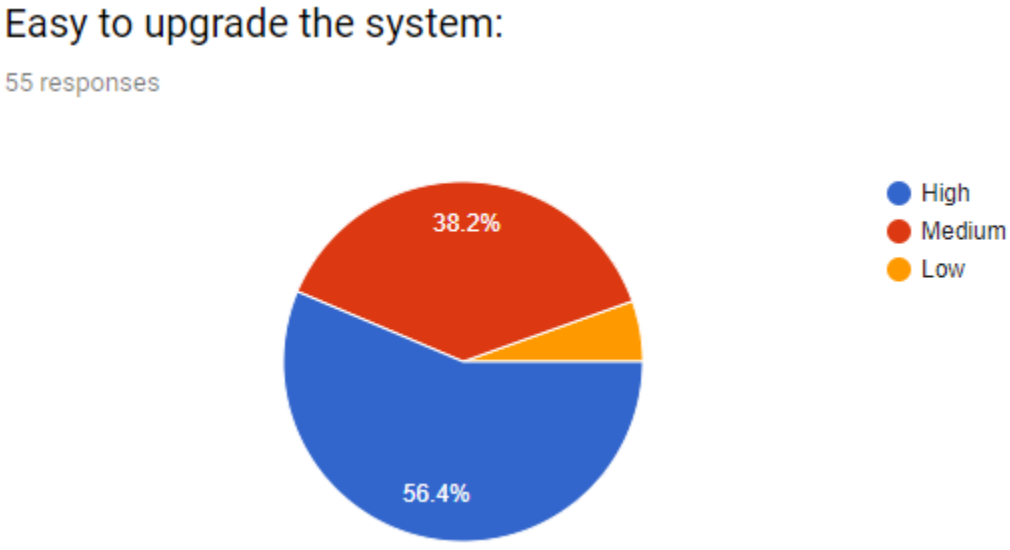


Figure 18 - Easy to upgrade the system

- General assessment:
    - The Database system is useful:

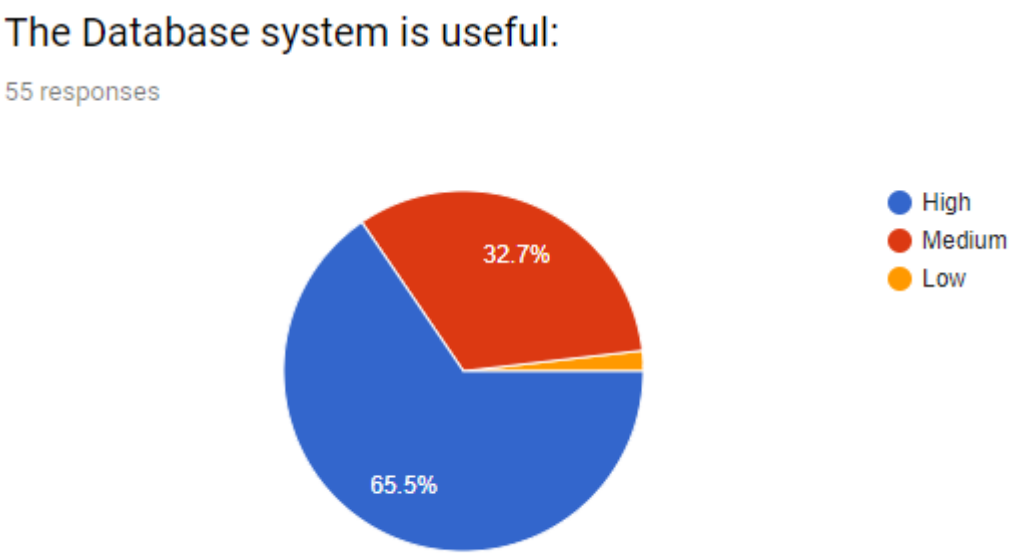


Figure 19 - The Database system is useful

    - User satisfaction:

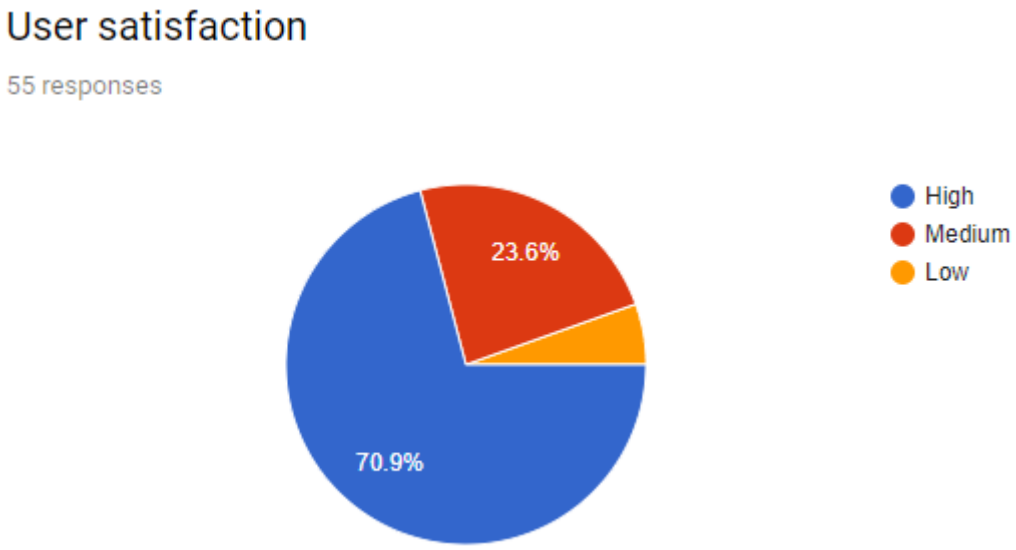


Figure 20 - User satisfaction

Evaluation Criteria	Reviews from users	Evaluate
Business:		
- Calculate	- User almost satisfied with database calculate	High
- Business plans	- The Database cannot work well in making business plans	Medium
- Making Marketing Plans such as sent mail, phone call, sent advertisement, etc.	- Easily to contact customers and marketing	High
Query Performance:		
- Selecting more data than needed	- This system still acceptable to select more data	High
- Inefficient joins between tables	- Tables in this database inefficient to join between	Medium
- Too few or too many indexes	- There are a few excess information	Medium
- Too much literal SQL causing parse contention	- Database cause parse contention	High
- User and Query Conflicts	- Query is not clean enough and make user confusing	High
Data:		
- Easy to import data into the system	- User easy import any kind of data into database	High
- Easily manipulate adding, editing, deleting data	- User easy add, edit, delete data in database system	High
- Encryption	- The data still not encryption in the database	Low
- Data saved in standard format	- Most data is saved in standard format	High
Other Issues:		
- Security of Database system	- Security of Database system is acceptable	Medium
- Easy maintenance	- The maintenance problem is relatively good	Medium
- Easy to upgrade the system	- Update the database system accordingly	High
General assessment:		
- The Database system is useful:	- The user feels the database is useful for the job	High
- User satisfaction	- Most users feel satisfied when using this database system	High

Table 26 - Evaluate the responsiveness of Database with user requirements

3. Points to note and improve

3.1. Data type in SQL Server

If storing the same data type, you can't classify what is the date, where the string is? where the number is? Therefore, the implementation of operators and search also becomes very difficult during data query.

On the other hand, posing a real problem: When you design a commercial database, installing Database storage is very important. We have a small example as follows, assuming:

- In a table, every 1 record corresponds to 1 byte of memory.
- One day you save 1,000,000 records that will take up 1,000,000 bytes.
- If saved a year will be 365 million bytes.

So if you store more than 1 byte per day, you will cause loss of capacity up to 365 million bytes / year. In contrast, the lack of capacity causes system stagnation such as memory shortage, insufficient storage capacity. It was just a small illustrative problem, and in reality it was often many times like that.

Today, computer capacity is often quite large, so the storage capacity is lost so it can be accepted temporarily. But what about mobile programming? According to the trend of using modern equipment, you clearly see that saving capacity is very important. Small devices cannot store too large capacity, so if data loss occurs, it will cause stagnation affecting the equipment system.

Therefore, it is necessary to determine the appropriate DATA TYPE for each data attribute to ensure optimal memory during use.

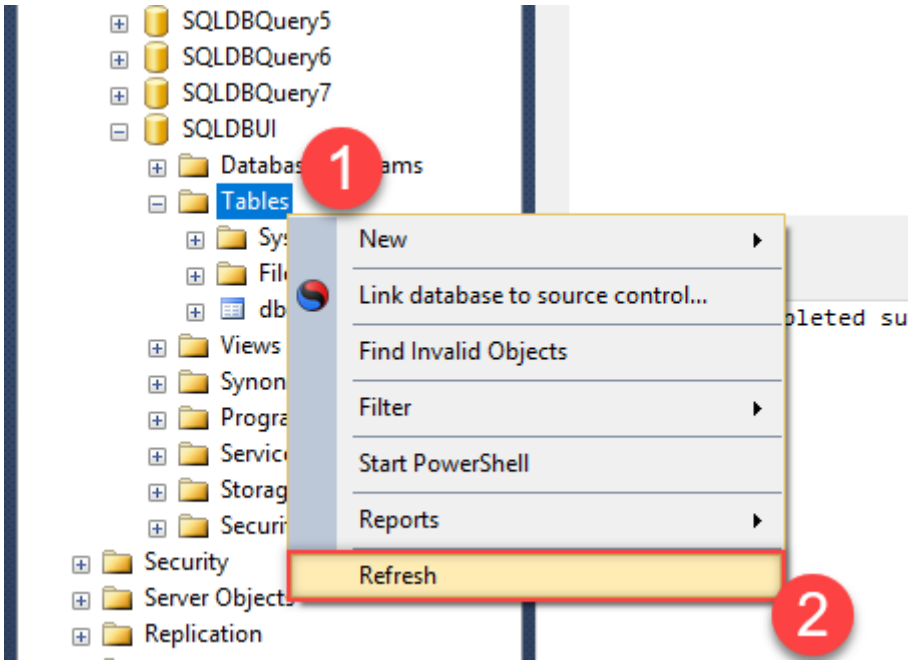
In this Database system of ElectroShop used most data types “varchar(100)” so it can not include Unicode types so in the future, when someone input the value with Unicode types will make Database system have trouble. Finally, we should change type of data from “varchar(100)” to “Nvarchar(100)” this is better solution.

3.2. Primary Key & Foreign Key and Initialize, delete, edit Table in SQL Server

A database consists of many tables, between the tables have a relationship with each other through the Primary Key & Foreign Key.

- Primary Key, Foreign Key: In this ElectroShop, the Primary Key and Foreign Key is used for define Only unique, non-duplicate data exists but not contains non-empty values (NULL) and Foreign Key must have the same data type, the same number of fields that have the corresponding Primary Key sorted. This will cause the Database system have a lot of trouble, so contains non-empty values (NULL) should be add to Primary Key and Foreign Key in each tables of Database system.
- Some notes in table initialization:
    - Create a new Table in the current Database: Before create a new table, make sure that we are using right database, if not use syntax: “use <database>”.

- The Table has not been found: In the process of creating a Table with code, in some cases we cannot see the Table that we just created, so right-click Table and Refresh to update the Table list.



- Each Table and Column has only one name in the Table.

- In addition, one of the important things is to know relations (Relationship) between the tables:
  - One-to-One Relationships or 1-1 relations: in this relationship, a row on table A cannot be linked to more than 1 row on Table B and vice versa.
  - One-to-Many Relationships or 1-n relationships: in this relationship, a row on the table A can be linked to multiple rows on Table B.
  - Many-to-Many Relationships or relationships: in this relationship, a row on Table A can be linked to multiple rows on Table B and a row on Table B can also link to multiple rows on Table A. As we can see in the example above a book can be written by many authors and an author can also write many books. Therefore the relationship between Books and Authors is Many to Many relations. In this case, an intermediate table is often used to solve the problem (AuthorBook table).
- Do not use multiple temporary tables when querying

APPENDIX: Survey Form

Code:……

Survey users on the database, the feasibility of the system on a scale: High - Medium – Low.

Evaluation Criteria	Evaluate		
	High	Medium	Low
Business:			
Calculate			
Business plans			
Making Marketing Plans such as sent mail, phone call, sent advertisement, etc.			
Query Performance:			
Selecting more data than needed			
Inefficient joins between tables			
Too few or too many indexes			
Too much literal SQL causing parse contention			
User and Query Conflicts			
Data:			
Easy to import data into the system			
Easily manipulate adding, editing, deleting data			
Encryption			
Data saved in standard format			
Other Issues:			
Security of Database system			
Easy maintenance			
Easy to upgrade the system			
General assessment:			
The Database system is useful:			
User satisfaction			

- Google Form: https://goo.gl/forms/kS3JUKCDD5jcs1e62

References

Habib, O. (2015, 12 23). *Cisco*. Retrieved from Cisco: https://blog.appdynamics.com/engineering/top-6-database-performance-metrics-to-monitor-in-enterprise-applications/

Wikipedia. (n.d.). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Microsoft_SQL_Server

Wikipedia. (n.d.). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/SQL_Server_Management_Studio