Module Title	**Database Design and Development**
Assignment Title	**Genie Delivery Service**
Examination Cycle	**Spring 2023**
Candidate Name	**SAW THET PHYOE**
Candidate No	**P00197144**
Centre Name	**KMD Computer Centre (Yangon)**
Submission Date:	**15 – January – 2023**

Important Notes:

- Please refer to the Assignment Presentation Requirements for advice on how to set out your assignment. These can be found on the NCC Education *Campus*. Scroll down the left hand side of the screen until you reach Personal Support. Click on this, and then on Policies and Advice. You will find the Assignment Presentation Requirements under the Advice section.
- You must familiarise yourself with the NCC Education Academic Dishonesty and Plagiarism Policy and ensure that you acknowledge all the sources which you use in your work. The policy is available on *Campus*. Follow the instructions above, but click on Policies rather than Advice.
- You must complete the **'Statement and Confirmation of Own Work'**. The form is available on the Policies section of *Campus*. Scroll down the left hand side until you reach Personal Support. Click on this and then click on Policies and Advice.
- Please make a note of the recommended word count. You could lose marks if you write 10% more or less than this.
- You must submit a paper copy and digital copy (on disk or similarly acceptable medium). Media containing viruses, or media which cannot be run directly, will result in a fail grade being awarded for this module.
- All electronic media will be checked for plagiarism.

Marker's comments:

Moderator's comments:

Mark:	**Moderated Mark:**	**Final Mark:**

Statement and Confirmation of Own Work

Programmed/Qualification name: Level 5 Diploma in Computing IT

All NCC Education assessed assignments submitted by students must have this statement as the cover page or it will not be accepted for marking. Please ensure that this statement is either firmly attached to the cover of the assignment or electronically inserted into the front of the assignment.

Student declaration

I have read and understood NCC Education's Policy on Academic Dishonesty and Plagiarism.
I can confirm the following details:

Student ID/Registration number	**:**	**P00197144**
Name	**:**	**SAW THET PHYOE**
Centre Name	**:**	**KMD Computer Centre (Yangon)**
Module Name	**:**	**Database Design and Development**
Module Leader	**:**	**Daw Wah Wah**
Number of words	**:**	**2280 Words**

I confirm that this is my own work and that I have not plagiarized any part of it. I have also noted the assessment criteria and pass mark for assignments.

Due Date	**:**	**15 – January – 2023**
Student Signature	**:**	**Saw Thet Phyoe**
Submitted Date	**:**	**15 – January – 2023**

Table of Contents

Task – 1

Task 1 – Business Overview and Requirements

Scenario

"Genie" is a delivery service company based in Yangon, Myanmar. They provide delivery services for various items – small to medium size, from a variety of shops in many townships. Items are delivered to customers by bike riders.

They want a database to help them manage ordering and delivering items as well as managing riders. The database will not be concerned, at least initially, with the allocation of payments to riders.

"Genie" receives tons of orders every day for items from different shops. Each customer can make many orders a day from different locations. Each order is assigned to one rider, who lives in the same township as the customer's delivery address. Customers are defined by member types – bronze, silver, gold and premium.

Payment can be made by different payment methods – cash on delivery, bank transfer and local digital money wallets such as KBZ Pay, Wave Pay or CB Pay. A single payment will have one specific payment method allocated to it.

Each rider will have several orders allocated to him and each order might include more than one item. A shop can have many items and each item is assumed to be associated with only one shop. Shops are defined by shop categories.

Examples of data are as shown in the tables below.

Sample Documents

Document 1 – Customers and Riders on Order Forms

Customer Name	Customer ID	Member Type	Order ID	Rider ID	Rider Name	Township
Kyaw Thu	GC-0012	Bronze	ORD-3111	GR-07	Soe Win	Bahan
Kyaw Thu	GC-0012	Bronze	ORD-3114	GR-08	Hla Myo Tun	Hlaing
Kyaw Thu	GC-0012	Bronze	ORD-3118	GR-09	Myint Thein	Dagon
Kyaw Thu	GC-0012	Bronze	ORD-3122	GR-07	Soe Win	Bahan
Hla Nu	GC-0028	Silver	ORD-3131	GR-12	Moe Hein	Hlaing
Hla Nu	GC-0028	Silver	ORD-3135	GR-13	Phyo Kyaw	Latha
Hla Nu	GC-0028	Silver	ORD-3138	GR-08	Hla Myo Tun	Hlaing
Htun Zaw	GC-0093	Gold	ORD-3143	GR-14	Thuta	Tamwae
Htun Zaw	GC-0093	Gold	ORD-3147	GR-12	Moe Hein	Hlaing
Htun Zaw	GC-0093	Gold	ORD-3149	GR-12	Moe Hein	Hlaing
Htun Zaw	GC-0093	Gold	ORD-3153	GR-10	Min Aung	Bahan
Maung Paing	GC-0004	Platinum	ORD-3155	GR-11	Htun Aye	Bahan
Maung Paing	GC-0004	Platinum	ORD-3158	GR-14	Thuta	Tamwae
Maung Paing	GC-0004	Platinum	ORD-3170	GR-11	Htun Aye	Bahan

Document 2 – Items and Order Details

Order ID	Order Date	Payment Type	Item Name	Item Code	Quantity
ORD-3111	16/8/2022	KBZ Pay	Cloth Face Mask	I-1090	3
ORD-3111	16/8/2022	KBZ Pay	Air-X Tablet	I-1086	1
ORD-3111	16/8/2022	KBZ Pay	Y-Si Vitamin C	I-1077	1
ORD-3114	18/8/2022	COD	Cheesecake Slice	I-2144	3
ORD-3114	18/8/2022	COD	Vanilla Ice Cream	I-2150	1
ORD-3114	18/8/2022	COD	Chicken Curry Puff	I-2138	1
ORD-3118	1/9/2022	AYA Pay	Remax TWS Ear Buds	I-2048	1
ORD-3118	1/9/2022	AYA Pay	Green Tech wireless mouse	I-2067	1
ORD-3122	2/9/2022	COD	Cosrx Acne Patch	I-3144	4
ORD-3122	2/9/2022	COD	A&C Mask Sheet	I-3178	3
ORD-3135	4/9/2022	KBZ Pay	Nestle Milo Jar	I-0126	2
ORD-3135	4/9/2022	KBZ Pay	Jordan Toothbrush	I-0130	6
ORD-3135	4/9/2022	KBZ Pay	Shark Energy Drink Can	I-0148	3

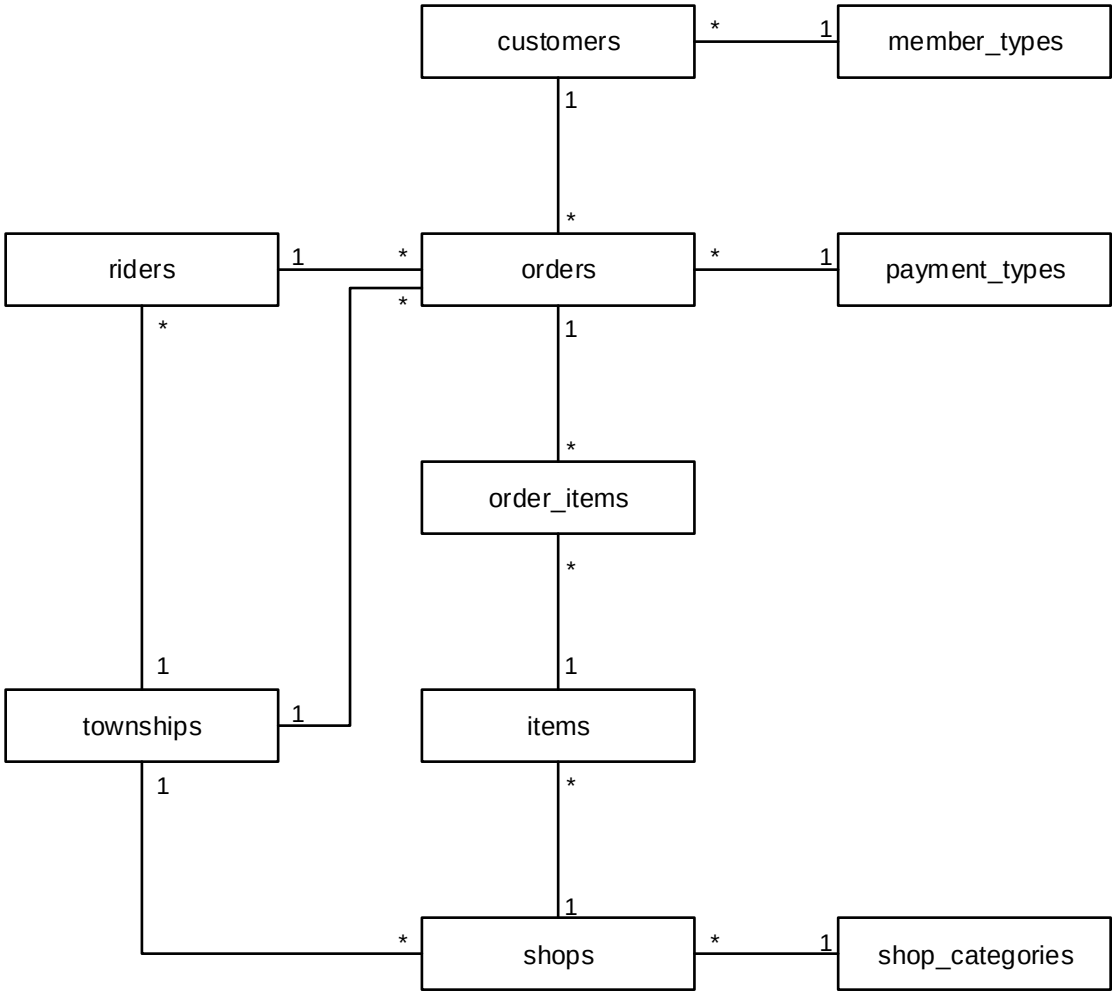
Document 3 – Shops and Items

Shop Name	Shop ID	Shop Category	Township	Item Code	Item Name	Unit Price (Kyats)
Moe's Bakery	S-097	Food	Hlaing	I-2144	Cheesecake Slice	4200
Moe's Bakery	S-097	Food	Hlaing	I-2145	Chocolate Cake	4000
Moe's Bakery	S-097	Food	Hlaing	I-2150	Vanilla Ice Cream	2700
Moe's Bakery	S-097	Food	Hlaing	I-2138	Chicken Curry Puff	5800
Shwe Oh Pharmacy	S-045	Health	Bahan	I-1077	Y-Si Vitamin-C	3200
Shwe Oh Pharmacy	S-045	Health	Bahan	I-1086	Air-X Tablet	1600
Shwe Oh Pharmacy	S-045	Health	Bahan	I-1090	Cloth Face Mask	1200
7Eleven	S-019	Convenience	Latha	I-0126	Nestle Milo Jar	6500
7Eleven	S-019	Convenience	Latha	I-0130	Jordan Toothbrush	2400
7Eleven	S-019	Convenience	Latha	I-0148	Shark Energy Drink Can	1400
Beauty Diary	S-114	Beauty	Bahan	I-3144	Cosrx Acne Patch	1800
Beauty Diary	S-114	Beauty	Bahan	I-3178	A&C Mask Sheet	2700
Tech Guru	S-012	Electronic	Dagon	I-2048	Remax TWS Ear Buds	32000
Tech Guru	S-012	Electronic	Dagon	I-2067	Green Tech wireless mouse	16500

Task – 2

Task 2 – ER and Data Dictionary

Entity Relationship Diagram



Data Dictionary

Table Name: townships

Primary Key: township_code

Foreign Key: totownnone

Attribute	Data Type	Size	Domain Constraint	Description
township_code	Integer	-	Unique Not Null Must be greater than 11000	Unique zip code for the township. Zip codes for Yangon region townships starts with '11'.
township_name	Varchar	50	Not Null	The name of the township

Table Name: member_types

Primary Key: member_type_id

Foreign Key: none

Attribute	Data Type	Size	Domain Constraint	Description
member_type_id	Integer	-	Unique Not Null	Unique identifier for the member type
member_type	Varchar	16	Not Null	Name of the member type

Table Name: shop_categories

Primary Key: shop_category_id

Foreign Key: none

Attribute	Data Type	Size	Domain Constraint	Description
shop_category_id	Integer	-	Unique Not Null	Unique identifier for the shop category
shop_category_name	Varchar	16	Not Null	Name of the shop category

Table Name: payment_types

Primary Key: payment_type_id

Foreign Key: none

Attribute	Data Type	Size	Domain Constraint	Description
payment_type_id	Integer	-	Unique Not Null	Unique identifier for the payment type
payment_type_name	Varchar	16	Not Null	Name of the payment type

Table Name: riders

Primary Key: rider_id

Foreign Key: township_code

Attribute	Data Type	Size	Domain Constraint	Propagation Constraint	Description
rider_id	Varchar	8	Unique Not Null Must start “GR-” following by a sequential number	-	Unique identifier for the rider
township_code	Integer	-	Not Null Must be greater than 11000	On Update Cascade On Delete No Action	Zip code of the township where the rider works
rider_name	Varchar	50	Not Null	-	Name of the rider
rider_phone	Varchar	15	Not Null	-	Contact Number of the rider
rider_address	Varchar	100	Not Null	-	Address of the rider
date_of_birth	Date	-	Not Null	-	Rider’s birth date
rider_registered_date	Date	-	Default current date	-	Registration date of the rider

Table Name: customers

Primary Key: customer_id

Foreign Key: member_type_id

Attribute	Data Type	Size	Domain Constraint	Propagation Constraint	Description
customer_id	Varchar	8	Unique Not Null Must start “GC-” following by a sequential number	-	Unique identifier for the customer
member_type_id	Integer	-	Not Null	On Update Cascade On Delete Restrict	Member type id for the customer
customer_name	Varchar	50	Not Null	-	Full name of the customer
customer_email	Varchar	50	Not Null	-	Email of the customer
customer_phone	Varchar	15	Not Null	-	Contact number of the customer
customer_registered_date	Date	-	Default current date	-	The date at when the customer registered at Genie Delivery

Table Name: shops

Primary Key: shop_id

Foreign Keys: township_code, shop_category_id

Attribute	Data Type	Size	Domain Constraint	Propagation Constraint	Description
shop_id	Varchar	8	Unique Not Null Must start with “S-” followed by a sequential number	-	Unique identifier of the shop
township_code	Integer	-	Not Null Must be greater than 11000	On Update Cascade On Delete Restrict	Zip code of the township the town is located
shop_category_id	Integer	-	Not Null	On Update Cascade On Delete Restrict	Category ID for the shop
shop_name	Varchar	50	Not Null	-	Name of the shop
shop_address	Varchar	100	Not Null	-	Address of the shop

Table Name: Items

Primary Key: item_code

Foreign Keys: shop_id

Attribute	Data Type	Size	Domain Constraint	Propagation Constraint	Description
item_code	Varchar	8	Unique Not Null Must start with “I-” followed by a sequential number	-	Unique identifier for the item
shop_id	Varchar	8	Not Null Must start with “S-” followed by a sequential number	On Update Cascade On Delete Restrict	The code of the shop from which the item is available
item_name	Varchar	50	Not Null	-	Name of the item
price	Decimal	(8,2)	Not Null	-	Unit price of the item

Table Name: orders

Primary Key: order_id

Foreign Keys: customer_id, rider_id, payment_type_id, township_code,

Attribute	Data Type	Size	Domain Constraint	Propagation Constraint	Description
order_id	Varchar	16	Unique Not Null Must start with “ORD-” followed by a sequential number	-	Unique identifier of the order
customer_id	Varchar	8	Must start “GC-” followed by a sequential number	On Update Cascade On Delete Set Null	The ID of the customer who made the order
rider_id	Varchar	8	Must start “GR-” followed by a sequential number	On Update Cascade On Delete Set Null	The ID of the rider who will be delivered the order
payment_type_id	Integer	8	Not Null	On Update Cascade On Delete No Action	The ID of the payment type for the order
township_code	Integer	-	Not Null Must be greater than 11000	On Update No Action On Delete No Action	Zip code of the township where the order is to be delivered

order_datetime	Datetime	-	Default current date	-	The exact date and time on when the customer made the order
delivered_datetime	Datetime	-	-	-	The exact date and time at when the order is delivered by the rider
delivery_address	Varchar	100	Not Null	-	The address the order is to be delivered
status	Varchar	16	Default “pending” Must be either “pending”, “delivered” or “canceled”	-	The status of the order whether it is pending, delivered or canceled

Table Name: orders_items

Primary Key: order_id + item_id

Foreign Keys: order_id, item_id

Attribute	Data Type	Size	Domain Constraint	Propagation Constraint	Description
order_id	Varchar	16	Not Null Must start with “ORD-” followed by a sequential number	On Update Cascade On Delete Cascade	Unique identifier of the order
item_code	Varchar	8	Not Null Must start with “I-” followed by a sequential number	On Update No Action On Delete No Action	Unique identifier of the item
quantity	Integer	-	Not Null	-	Total quantity of the item in the order

Task – 3

Task 3 – Normalisation

Purpose of Normalisation

Normalisation is the process of organizing data in a database. The main purpose of normalisation is to minimize redundancy (duplicate data) and to overcome potential anomalies (eliminate inconsistent dependency). Normalisation can even increase security, improve workflow and lessen costs.

Normalisation for Given Documents

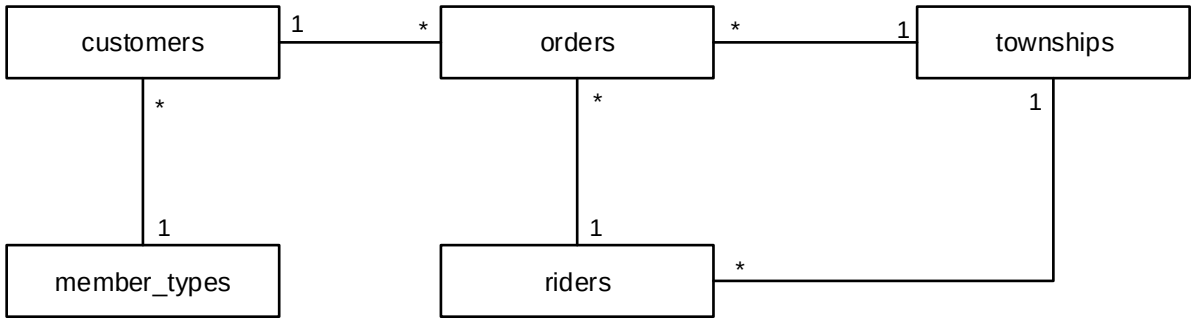
Normalisation for Document 1 – Customers and Riders on Order Forms

UNF	Level	1NF	2NF	3NF	Optimization
CustomerID	1	CustomerID(PK)	CustomerID(PK)	CustomerID(PK)	**[customers]**
CustomerName	1	CustomerName	CustomerName	CustomerName	customer_id(PK)
MemberType	1	MemberType	MemberType	MemberTypeID(FK)	member_type_id(FK)
OrderID	2				customer_name
RiderID	2	OrderID(PK)	OrderID(PK)	MemberTypeID(PK)	customer_email
RiderName	2	CustomerID(FK)	CustomerID(FK)	MemberType	customer_phone
Township	2	RiderID	RiderID(FK)		customer_address
		RiderName		OrderID(PK)	registered_date
		Township	RiderID(PK)	CustomerID(FK)	
			RiderName	RiderID(FK)	**[member_types]**
			Township	TownshipCode(FK)	member_type_id(PK)
					member_type
				RiderID(PK)	
				RiderName	**[orders]**
				TownshipCode(FK)	order_id(PK)
					customer_id(FK)
				TownshipCode(PK)	rider_id(FK)
				TownshipName	township_code(FK)
					order_datetime
					delivered_datetime

					deliverey_address **[riders]** rider_id(PK) township_code(FK) rider_name rider_phone rider_address date_of_birth registered_date **[townships]** township_code(PK) towship_name

After normalisation for the first document, five tables are obtained. Although member types can be embedded in “Customers” table, it is decided to use a separate table aiming to store more information about each member. The same concept applies for townships table. It has no dependencies, yet a separate table is used. Another reason to store township in a separate table is that it is related to more than one entity. The ER Diagram is as shown in the following figure.

ER Diagram for Document 1 – Customers and Riders on Order Forms

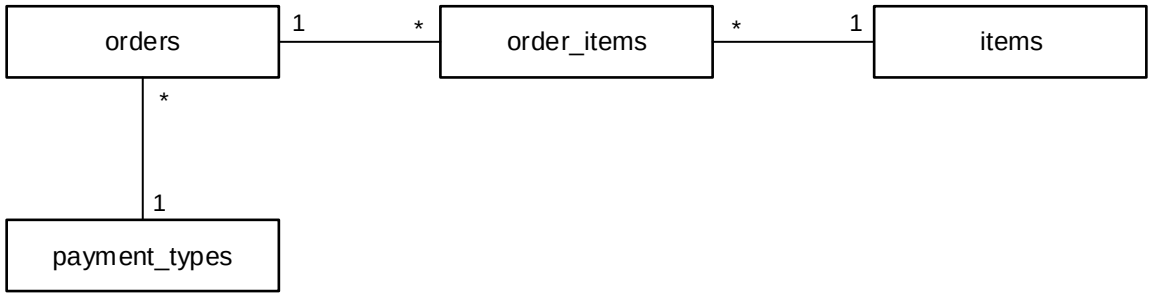


Normalisation for Document 2 – Items and Order Details

UNF	Level	1NF	2NF	3NF	Optimization
OrderID	1	OrderID(PK)	OrderID(PK)	OrderID(PK)	**[orders]**
OrderDate	1	OrderDate	OrderDate	OrderDate	order_id(PK)
PaymentType	1	PaymentType	PaymentType	PaymentTypeID(FK)	payment_type_id(FK)
ItemCode	2				order_datetime
ItemName	2	OrderID(FK) } PK	OrderID(FK,PK)	PaymentTypeID(PK)	delivered_datetime
Quantity	2	ItemCode } PK	ItemCode(FK,PK)	PaymentType	delivery_address
		ItemName	Quantity		
		Quantity		OrderID(FK,PK)	**[payment_types]**
			ItemCode(PK)	ItemCode(FK,PK)	payment_type_id(PK)
			ItemName	Quantity	payment_type
				ItemCode(PK)	**[orders_items]**
				ItemName	order_id(FK,PK)
					item_code(FK,PK)
					quantity
					[items]
					item_code(PK)
					item_name
					price

The normalisation for the second document gives four tables. Orders and Items are related to each other with many to many relationship resulting a link (dummy) table with a composite primary key. Payment types are designed to store in a separate table to make the schema more flexible and to store additional information for each payment type. The ER Diagram is as shown below.

ER Diagram for Document 2 – Items and Order Details

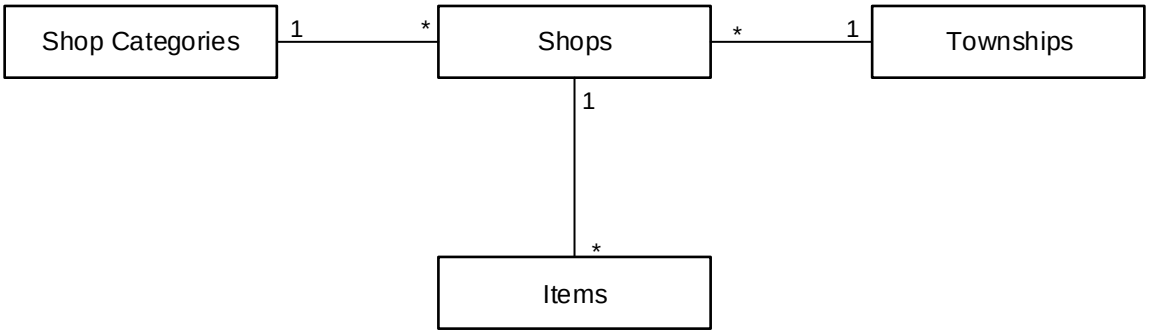


Normalisation for Document 3 – Shops and Items

UNF	Level	1NF	2NF	3NF	Optimization
ShopID	1	ShopID(PK)	ShopID(PK)	ShopID(PK)	**[shops]**
ShopName	1	ShopName	ShopName	ShopName	shop_id(PK)
ShopCategory	1	ShopCategory	ShopCategory	ShopCategoryID(FK)	shop_category_id(FK)
Township	1	Township	Township	TownshipCode(FK)	township_code(FK)
ItemCode	2				shop_name
ItemName	2	ItemCode(PK)	ItemCode(PK)	ShopCategoryID(PK)	shop_address
UnitPrice	2	ShopID(FK)	ShopID(FK)	CategoryName	
		ItemName	ItemName		**[shop_categories]**
		UnitPrice	UnitPrice	TownshipCode(PK)	shop_category_id(PK)
				TownshipName	shop_category_name
				ItemCode(PK)	**[townships]**
				ShopID(FK)	township_code(PK)
				ItemName	township_name
				UnitPrice	
					[Items]
					item_code(PK)
					shop_id(FK)
					item_name
					price

Normalisation for the third document gives four related tables. According to the given document, townships can be embedded into “shops” table. But it may be related to other entities and so it is decided to store in a separate table. The ER Diagram is as shown below.

ER Diagram for Document 3 – Shops and Items



How Normalisation Solves Possible Anomalies

Insert Anomalies

It is said that an order can be paid with different payment types. Suppose that payment types are embedded into orders. If we want to allow customers to use a new payment type and no one has not used that, it is not possible to insert into database because we need an order_id (primary key). Normalisation gives use a separate table for payment types allowing us to insert new methods and other information.

Update Anomalies

Alternative approach to designing shops and items is to embed shops in items table. But if a shop changes its name for some reason, we need to update every items from that shop. After normalisation, shops and items are split into separate tables allowing us to overcome the above problem.

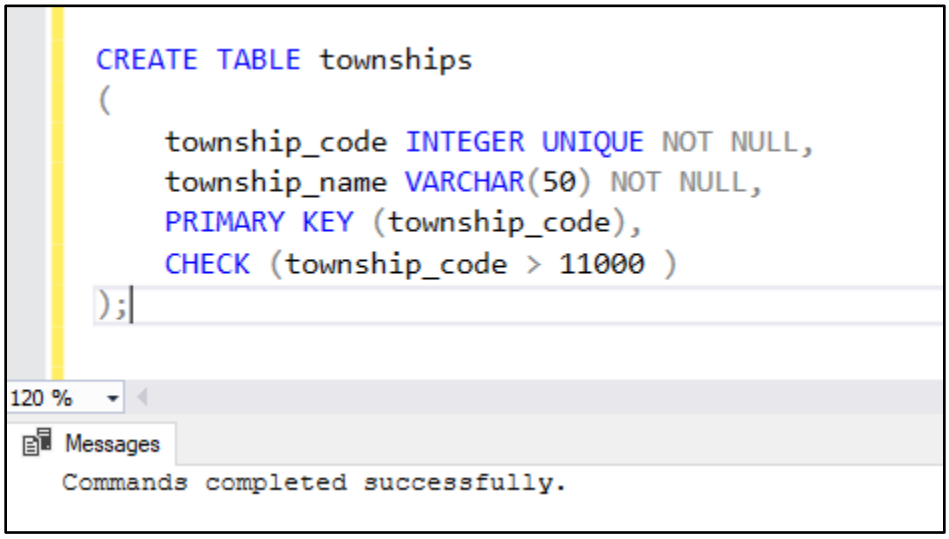
Delete Anomalies

This is a type of anomaly that cause losing information we don’t want to lose, because it is tied to data where it shouldn’t be. Normalisation helps us to overcome this kind of situation by placing every data in its related table and by splitting into separate tables if needed.

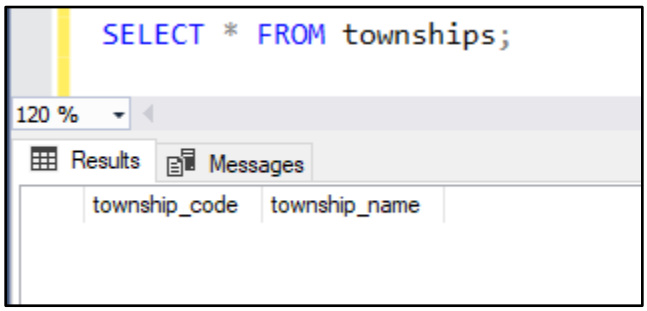
Task – 4

Task 4 – Scripts to Create Table Structures

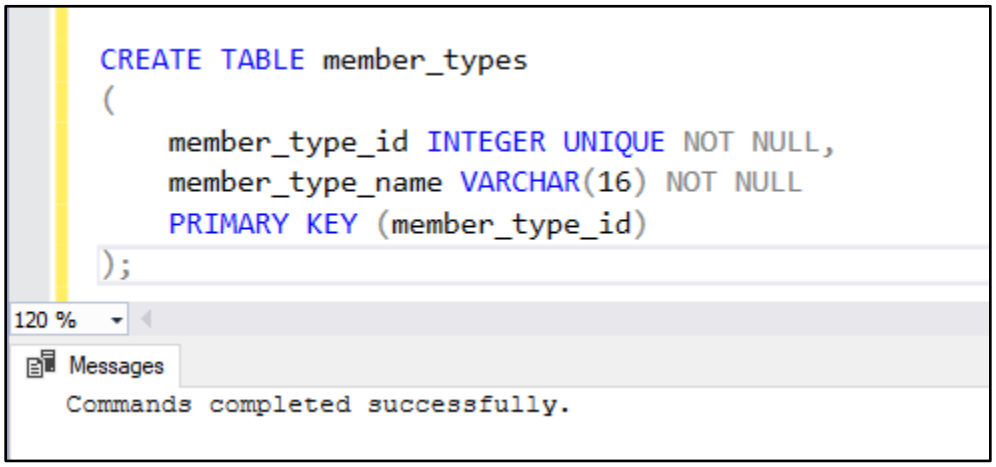
Townships Table



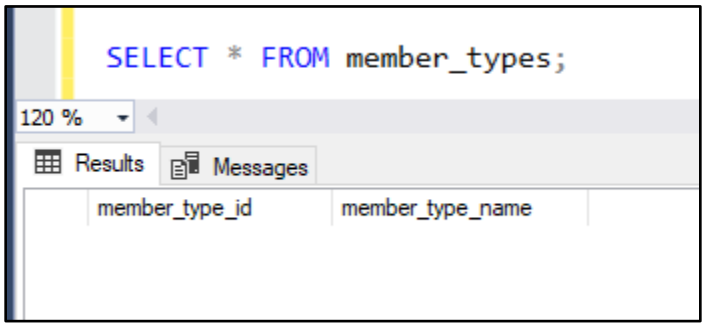
Result



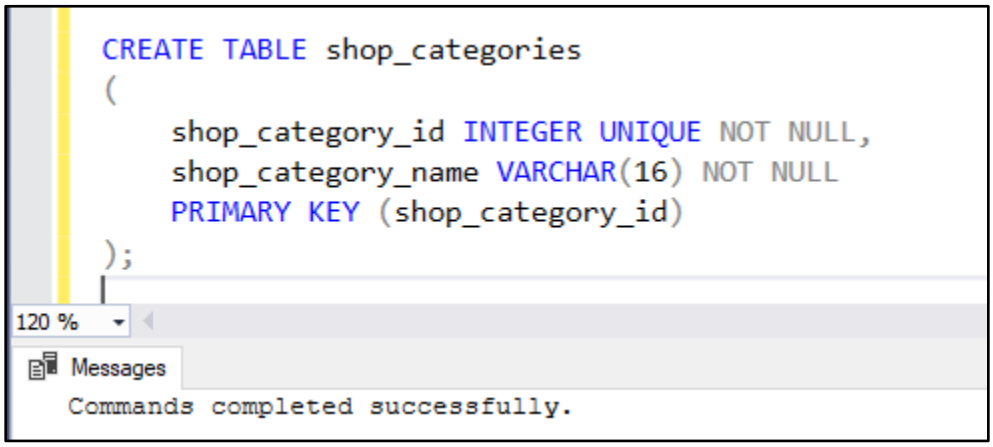
Member Types Table



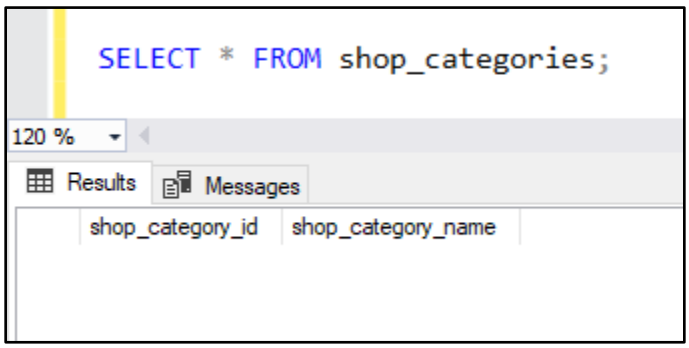
Result



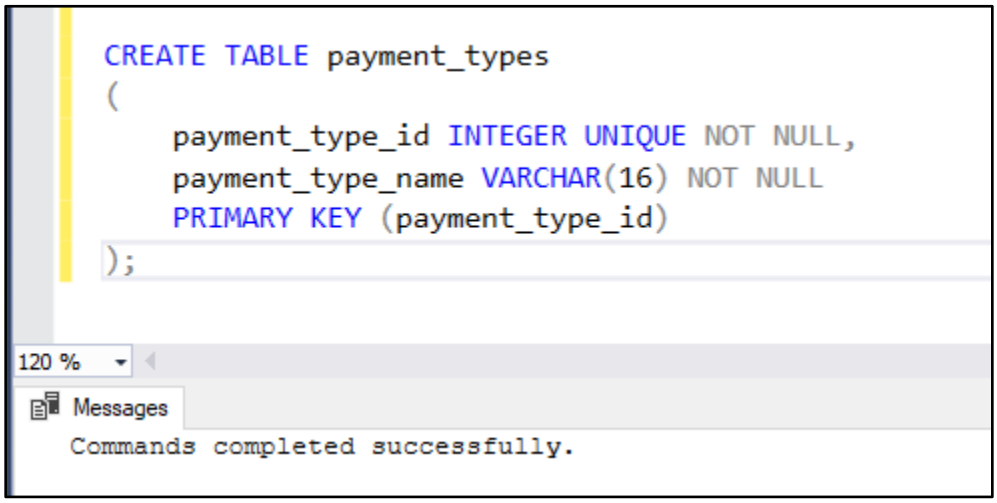
Shop Categories Table



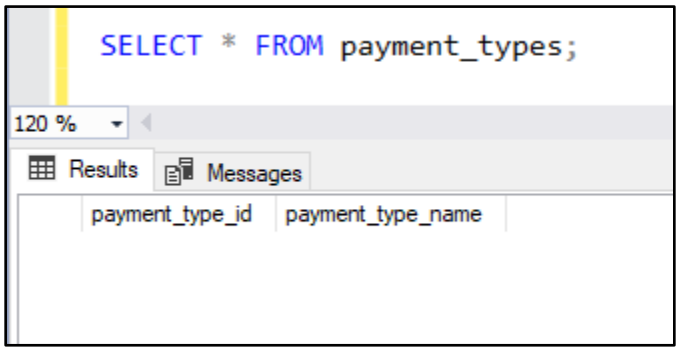
Result



Payment Type Table



Result



Riders Table

```
CREATE TABLE riders
(
    rider_id VARCHAR(8) UNIQUE NOT NULL,
    township_code INTEGER NOT NULL,
    rider_name VARCHAR(50) NOT NULL,
    rider_phone VARCHAR(15) NOT NULL,
    rider_address VARCHAR(100) NOT NULL,
    date_of_birth DATE NOT NULL,
    rider_registered_date DATE DEFAULT CURRENT_TIMESTAMP,
    CHECK (rider_id LIKE('GR-%')),
    CHECK (township_code > 11000),
    PRIMARY KEY (rider_id),
    FOREIGN KEY (township_code) REFERENCES townships(township_code)
    ON UPDATE CASCADE
    ON DELETE NO ACTION
);
```

120 %

Messages

```
Commands completed successfully.
```

Result

```
SELECT * FROM riders;
```

120 %

Results | Messages

rider_id	township_code	rider_name	rider_phone	rider_address	date_of_birth	rider_registered_date

Customers Table

```
CREATE TABLE customers
(
    customer_id VARCHAR(8) UNIQUE NOT NULL,
    member_type_id INTEGER NOT NULL,
    customer_name VARCHAR(50) NOT NULL,
    customer_email VARCHAR(50) NOT NULL,
    customer_phone VARCHAR(15) NOT NULL,
    customer_registered_date DATE DEFAULT CURRENT_TIMESTAMP,
    CHECK (customer_id LIKE('GC-%')),
    PRIMARY KEY (customer_id),
    FOREIGN KEY (member_type_id) REFERENCES member_types(member_type_id)
    ON UPDATE CASCADE
    ON DELETE NO ACTION
);
```

120 %

Messages

```
Commands completed successfully.
```

Result

```
SELECT * FROM customers;
```

120 %

Results | Messages

customer_id	member_type_id	customer_name	customer_email	customer_phone	customer_registered_date

Shops Table

```
CREATE TABLE shops
(
     shop_id VARCHAR(8) UNIQUE NOT NULL,
     township_code INTEGER NOT NULL,
     shop_category_id INTEGER NOT NULL,
     shop_name VARCHAR(50) NOT NULL,
     shop_address VARCHAR(100) NOT NULL,
     CHECK (shop_id LIKE('S-%')),
     CHECK (township_code > 11000),
     PRIMARY KEY (shop_id),
     FOREIGN KEY (township_code) REFERENCES townships(township_code)
     ON UPDATE CASCADE ON DELETE NO ACTION,
     FOREIGN KEY (shop_category_id) REFERENCES shop_categories(shop_category_id)
     ON UPDATE CASCADE ON DELETE NO ACTION
);
```

120 %

Messages

```
Commands completed successfully.
```

Result

```
SELECT * FROM shops;
```

120 %

Results | Messages

shop_id	township_code	shop_category_id	shop_name	shop_address

Items Table

```
CREATE TABLE items
(
    item_code VARCHAR(8) UNIQUE NOT NULL,
    shop_id VARCHAR(8) NOT NULL,
    item_name VARCHAR(50) NOT NULL,
    price DECIMAL(8,2) NOT NULL,
    CHECK (item_code LIKE ('I-%')),
    CHECK (shop_id LIKE ('S-%')),
    PRIMARY KEY (item_code),
    FOREIGN KEY (shop_id) REFERENCES shops(shop_id)
    ON UPDATE CASCADE
    ON DELETE NO ACTION
);
```

120 %

Messages

```
Commands completed successfully.
```

Result

```
SELECT * FROM items;
```

120 %

Results | Messages

item_code	shop_id	item_name	price	in_stock

Orders Table

```
CREATE TABLE orders
(
    order_id VARCHAR(16) UNIQUE NOT NULL,
    customer_id VARCHAR(8),
    rider_id VARCHAR(8),
    payment_type_id INTEGER NOT NULL,
    township_code INTEGER NOT NULL,
    order_date DATE DEFAULT CURRENT_TIMESTAMP,
    order_time TIME DEFAULT CURRENT_TIMESTAMP,
    delivered_time TIME,
    delivery_address VARCHAR(100) NOT NULL,
    order_status VARCHAR(50) DEFAULT 'pending',
    CHECK (order_id LIKE('ORD-%')),
    CHECK (customer_id LIKE('GC-%')),
    CHECK (rider_id LIKE('GR-%')),
    CHECK (township_code > 11000),
    CHECK (order_status IN('pending','delivered','canceled')),
    PRIMARY KEY (order_id),
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id)
    ON UPDATE CASCADE ON DELETE SET NULL,
    FOREIGN KEY (rider_id) REFERENCES riders(rider_id)
    ON UPDATE CASCADE ON DELETE SET NULL,
    FOREIGN KEY (payment_type_id) REFERENCES payment_types(payment_type_id)
    ON UPDATE CASCADE ON DELETE NO ACTION,
    FOREIGN KEY (township_code) REFERENCES townships(township_code)
    ON UPDATE NO ACTION ON DELETE NO ACTION
);
```

120 %

Messages

```
Commands completed successfully.
```

Result

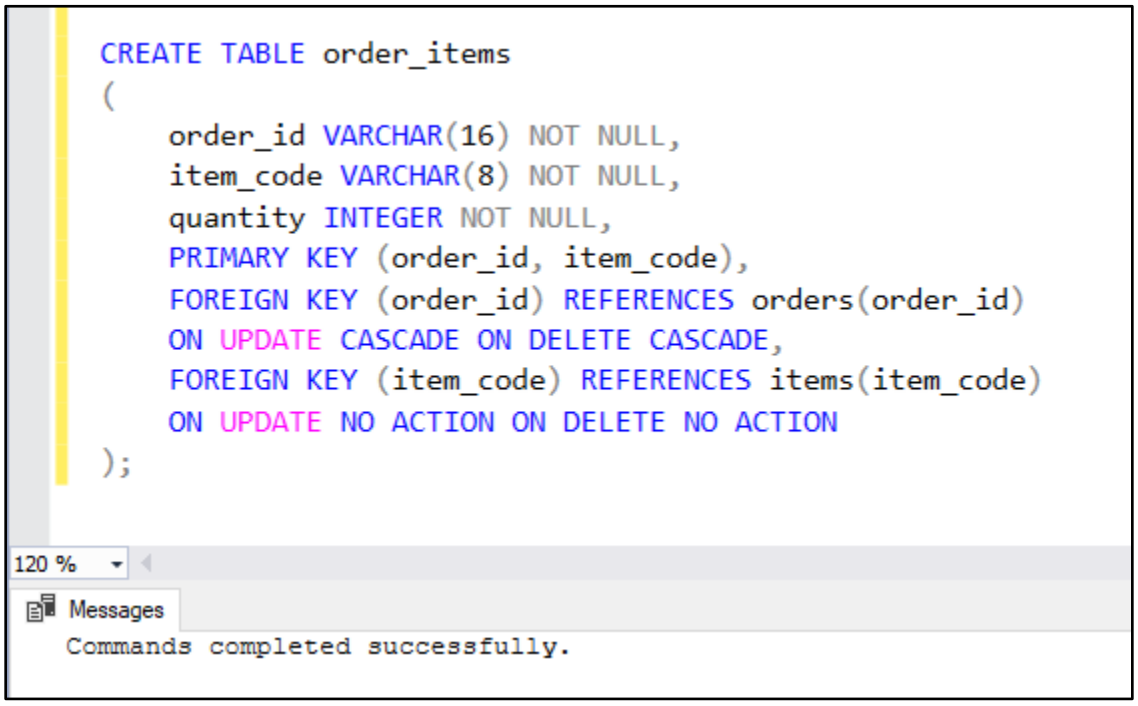
```
SELECT * FROM orders;
```

120 %

Results | Messages

order_id	customer_id	rider_id	payment_type_id	township_code	order_date	order_time	delivered_time	delivery_address	order_status

Order Items Table



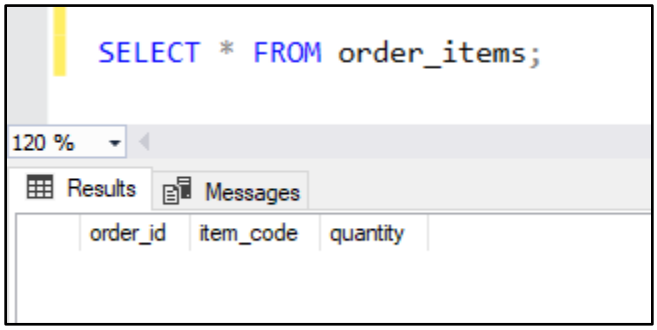
```
CREATE TABLE order_items
(
    order_id VARCHAR(16) NOT NULL,
    item_code VARCHAR(8) NOT NULL,
    quantity INTEGER NOT NULL,
    PRIMARY KEY (order_id, item_code),
    FOREIGN KEY (order_id) REFERENCES orders(order_id)
    ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (item_code) REFERENCES items(item_code)
    ON UPDATE NO ACTION ON DELETE NO ACTION
);
```

120 %

Messages

Commands completed successfully.

Result



```
SELECT * FROM order_items;
```

120 %

Results | Messages

order_id	item_code	quantity

Above scripts are executed on SQL Server 2019 Express using SQL Server Management Studio for development environment. Tables without foreign key references are created first. And then, tables which have one-to-many relationships are created and finally the table with a composite primary key (dummy table) is created. I have encountered with MSSQLSERVER_1785 error a lot which is caused by implementing propagation constraints that can cause multiple cascade paths.

SQL makes the database well-structured and makes it fit with the business needs by providing strong relationships between tables. The use of constraints ensures the accuracy and reliability of the data. Using build-in date functions such as CURRENT_TIMESTAMP for default value makes date and time data in the database more precise.

Task – 5

Task 5 – Data Population

Insert Data into Townships Table

```
INSERT INTO townships VALUES
    (11201, 'Bahan'),
    (11211, 'Tamwae'),
    (11081, 'Yankin'),
    (11052, 'Hlaing'),
    (11121, 'Alone'),
    (11141, 'Latha'),
    (11191, 'Dagon'),
    (11143, 'Pabedan'),
    (11161, 'Botahtaung'),
    (11041, 'Kamayut');
```

120 %

Messages

```
(10 rows affected)
```

Result

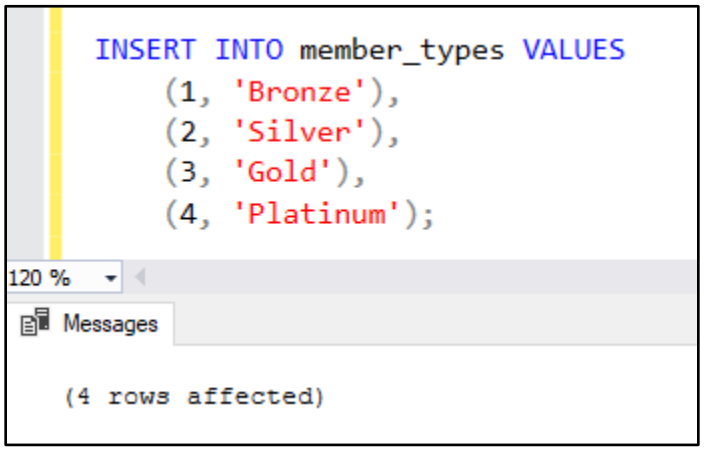
```
SELECT * FROM townships;
```

120 %

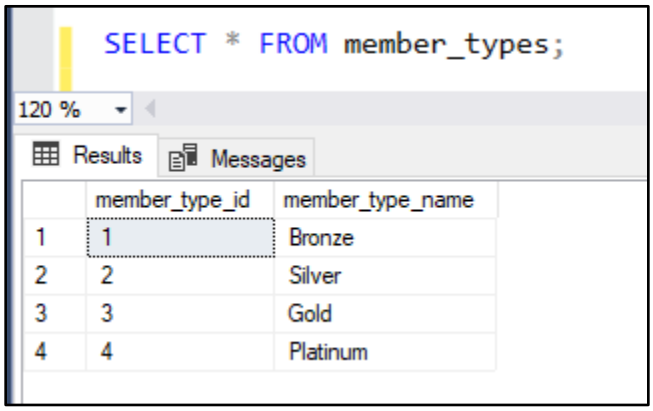
Results | Messages

	township_code	township_name
1	11041	Kamayut
2	11052	Hlaing
3	11081	Yankin
4	11121	Alone
5	11141	Latha
6	11143	Pabedan
7	11161	Botahtaung
8	11191	Dagon
9	11201	Bahan
10	11211	Tamwae

Insert Data into Member Types Table



Result



Insert Data into Shop Categories Table

```
INSERT INTO shop_categories VALUES
    (1, 'Food'),
    (2, 'Health'),
    (3, 'Beauty'),
    (4, 'Grocery'),
    (5, 'Convenience'),
    (6, 'Household'),
    (7, 'Fashion'),
    (8, 'Electronic');
```

120 %

Messages

```
(8 rows affected)
```

Result

```
SELECT * FROM shop_categories;
```

120 %

Results | Messages

	shop_category_id	shop_category_name
1	1	Food
2	2	Health
3	3	Beauty
4	4	Grocery
5	5	Convenience
6	6	Household
7	7	Fashion
8	8	Electronic

Insert Data into Payment Types Table

```
INSERT INTO payment_types VALUES
    (1, 'Cash on Delivery'),
    (2, 'KBZ Pay'),
    (3, 'AYA PAY'),
    (4, 'Wave Pay'),
    (5, 'Bank Transfer');
```

120 %

Messages

```
(5 rows affected)
```

Result

```
SELECT * FROM payment_types;
```

120 %

Results | Messages

	payment_type_id	payment_type_name
1	1	Cash on Delivery
2	2	KBZ Pay
3	3	AYA PAY
4	4	Wave Pay
5	5	Bank Transfer

Insert Data into Riders Table

```
INSERT INTO riders
    (rider_id, township_code, rider_name, rider_phone, rider_address, date_of_birth)
VALUES
    ('GR-07', 11201, 'Soe Win', '09123456789', 'No.3 Mya Street, Sayar San Ward', '1994/02/20'),
    ('GR-08', 11052, 'Hla Myo Tun', '09123456789', 'No.4 Thamine 1 Street, Kyo Kone Ward', '1995-08-02'),
    ('GR-09', 11191, 'Myint Thein', '09123456789', 'No.5 Kyaung Street, Ba Htoo Ward', '1992-03-11'),
    ('GR-10', 11201, 'Min Aung', '09123456789', 'No.82 U Chit Maung Road, Bo Sein Hman Ward', '2002-09-29'),
    ('GR-11', 11201, 'Htun Aye', '09123456789', 'No.92 Pearl Street, Kyeik San Ward', '1998-04-01'),
    ('GR-12', 11052, 'Moe Hein', '09123456789', 'No.25 Parami Road, Thamine 2 Ward', '1998-10-23'),
    ('GR-13', 11141, 'Phyo Kyaw', '09123456789', 'No.101 Shwe Bone Thar Road, Latha 3 Ward', '1997-11-03'),
    ('GR-14', 11211, 'Thuta', '09123456789', 'No.78 Thida Road, Kyauk Myaung Ward', '1996-07-09'),
    ('GR-03', 11081, 'Nyan Lin', '09123456789', 'No.56 Yan Shin Road, Yankin 7 Ward', '2003-09-12'),
    ('GR-22', 11121, 'Min Min', '09123456789', 'No.5 16th Street, Pyi Htaung Su Ward', '2004-01-01');
```

120 %

Messages

```
(10 rows affected)
```

Result

```
SELECT * FROM riders;
```

120 %

Results | Messages

	rider_id	township_code	rider_name	rider_phone	rider_address	date_of_birth	rider_registered_date
1	GR-03	11081	Nyan Lin	09123456789	No.56 Yan Shin Road, Yankin 7 Ward	2003-09-12	2023-01-11
2	GR-07	11201	Soe Win	09123456789	No.3 Mya Street, Sayar San Ward	1994-02-20	2023-01-11
3	GR-08	11052	Hla Myo Tun	09123456789	No.4 Thamine 1 Street, Kyo Kone Ward	1995-08-02	2023-01-11
4	GR-09	11191	Myint Thein	09123456789	No.5 Kyaung Street, Ba Htoo Ward	1992-03-11	2023-01-11
5	GR-10	11201	Min Aung	09123456789	No.82 U Chit Maung Road, Bo Sein Hman Ward	2002-09-29	2023-01-11
6	GR-11	11201	Htun Aye	09123456789	No.92 Pearl Street, Kyeik San Ward	1998-04-01	2023-01-11
7	GR-12	11052	Moe Hein	09123456789	No.25 Parami Road, Thamine 2 Ward	1998-10-23	2023-01-11
8	GR-13	11141	Phyo Kyaw	09123456789	No.101 Shwe Bone Thar Road, Latha 3 Ward	1997-11-03	2023-01-11
9	GR-14	11211	Thuta	09123456789	No.78 Thida Road, Kyauk Myaung Ward	1996-07-09	2023-01-11
10	GR-22	11121	Min Min	09123456789	No.5 16th Street, Pyi Htaung Su Ward	2004-01-01	2023-01-11

Insert Data into Customers Table

```
INSERT INTO customers
    (customer_id, member_type_id, customer_name, customer_email, customer_phone)
VALUES
    ('GC-0028', 2, 'Hla Nu', 'hlanu1@gmail.com', '09123456789'),
    ('GC-0012', 1, 'Kyaw Thu', 'kyawthu1@gmail.com', '09123456789'),
    ('GC-0093', 3, 'Htun Zaw', 'htunzaw1@gmail.com', '09123456789'),
    ('GC-0004', 4, 'Maung Paing', 'mgpaing1@gmail.com', '09123456789'),
    ('GC-0023', 2, 'Zaw Win', 'zawwin1@gmail.com', '09123456789'),
    ('GC-0092', 3, 'Kaung Htet', 'kaunghtet1@gmail.com', '09123456789'),
    ('GC-0099', 1, 'Soe Min', 'soemin1@gmail.com', '09123456789'),
    ('GC-0033', 1, 'Win Naing', 'winnaing1@gmail.com', '09123456789'),
    ('GC-0021', 1, 'Soe Htet', 'soehtet1@gmail.com', '09123456789'),
    ('GC-0067', 2, 'Sithu', 'sithu1@gmail.com', '09123456789');
```

120 %

Messages

```
(10 rows affected)
```

Result

```
SELECT * FROM customers;
```

120 %

Results | Messages

	customer_id	member_type_id	customer_name	customer_email	customer_phone	customer_registered_date
1	GC-0004	4	Maung Paing	mgpaing1@gmail.com	09123456789	2023-01-11
2	GC-0012	1	Kyaw Thu	kyawthu1@gmail.com	09123456789	2023-01-11
3	GC-0021	1	Soe Htet	soehtet1@gmail.com	09123456789	2023-01-11
4	GC-0023	2	Zaw Win	zawwin1@gmail.com	09123456789	2023-01-11
5	GC-0028	2	Hla Nu	hlanu1@gmail.com	09123456789	2023-01-11
6	GC-0033	1	Win Naing	winnaing1@gmail.com	09123456789	2023-01-11
7	GC-0067	2	Sithu	sithu1@gmail.com	09123456789	2023-01-11
8	GC-0092	3	Kaung Htet	kaunghtet1@gmail.com	09123456789	2023-01-11
9	GC-0093	3	Htun Zaw	htunzaw1@gmail.com	09123456789	2023-01-11
10	GC-0099	1	Soe Min	soemin1@gmail.com	09123456789	2023-01-11

Insert Data into Shops Table

```
INSERT INTO shops VALUES
    ('S-097', 11052, 1, 'Moe''s Bakery', 'No.101, Insein Road, Oak Kyin Ward'),
    ('S-045', 11201, 2, 'Shwe Oh Pharmacy', 'No.24, Nat Mouk Road, Bo Gyote Ward'),
    ('S-019', 11141, 5, '7Eleven', 'No.88, Bone Gyi Road, Aung Mingalar Ward'),
    ('S-114', 11201, 3, 'Beauty Diary', 'No.18, U Chit Maung Road, R Zar Ni Ward'),
    ('S-012', 11191, 8, 'Tech Guru', 'No.73, Koe Htet Kyi Road, Ward 33'),
    ('S-023', 11201, 1, 'Bonchon Chicken', 'No.81, Yadanar Street, Kyeik Ka San Ward'),
    ('S-035', 11141, 8, 'Technoland', 'No.126, Sint Oh Tan Street, Latha 3 Ward'),
    ('S-028', 11201, 1, 'Tea Leaf', 'Ground Floor, Ocean Super Center(Tamwae)'),
    ('S-101', 11143, 5, 'CoCo', 'No.97, Anawyahtar Road, 18th Ward'),
    ('S-119', 11081, 4, 'ABC Minimart', 'No.7, Mingalar Road, Yan Kin Ward'),
    ('S-020', 11081, 1, 'Yankin Kyay Oh', 'No.83, Kanbe Road, Yan Kin Ward');
```

120 %

Messages

```
(11 rows affected)
```

Result

```
SELECT * FROM shops;
```

120 %

Results | Messages

	shop_id	township_code	shop_category_id	shop_name	shop_address
1	S-012	11191	8	Tech Guru	No.73, Koe Htet Kyi Road, Ward 33
2	S-019	11141	5	7Eleven	No.88, Bone Gyi Road, Aung Mingalar Ward
3	S-020	11081	1	Yankin Kyay Oh	No.83, Kanbe Road, Yan Kin Ward
4	S-023	11201	1	Bonchon Chicken	No.81, Yadanar Street, Kyeik Ka San Ward
5	S-028	11201	1	Tea Leaf	Ground Floor, Ocean Super Center(Tamwae)
6	S-035	11141	8	Technoland	No.126, Sint Oh Tan Street, Latha 3 Ward
7	S-045	11201	2	Shwe Oh Pharmacy	No.24, Nat Mouk Road, Bo Gyote Ward
8	S-097	11052	1	Moe's Bakery	No.101, Insein Road, Oak Kyin Ward
9	S-101	11143	5	CoCo	No.97, Anawyahtar Road, 18th Ward
10	S-114	11201	3	Beauty Diary	No.18, U Chit Maung Road, R Zar Ni Ward
11	S-119	11081	4	ABC Minimart	No.7, Mingalar Road, Yan Kin Ward

Insert Data into Items Table

```
INSERT INTO items VALUES
    ('I-2144', 'S-097', 'Cheesecake Slice', 4200),
    ('I-2145', 'S-097', 'Chocolate Cake', 4000),
    ('I-2150', 'S-097', 'Vanilla Ice Cream', 2700),
    ('I-2138', 'S-097', 'Chicken Curry Puff', 5800),
    ('I-1077', 'S-045', 'Y-Si Vitamin-C', 3200),
    ('I-1086', 'S-045', 'Air-X Tablet', 1600),
    ('I-1090', 'S-045', 'Cloth Face Mask', 1200),
    ('I-0126', 'S-019', 'Nestle Milo Jar', 6500),
    ('I-0130', 'S-019', 'Jordan Toothbrush', 2400),
    ('I-0148', 'S-019', 'Shark Energy Drink Can', 1400),
    ('I-3144', 'S-114', 'Cosrx Acne Patch', 1800),
    ('I-3178', 'S-114', 'A&C Mask Sheet', 2700),
    ('I-2048', 'S-012', 'Remax TWS Ear Buds', 32000),
    ('I-2067', 'S-012', 'Green Tech Wireless Mouse', 16500),
    ('I-2712', 'S-023', 'Spicy Fried Chicken', 2200),
    ('I-2716', 'S-023', 'Chicken Burger', 5200),
    ('I-1910', 'S-035', 'Apple Original Earphone', 45000),
    ('I-1914', 'S-035', 'Type C to Lightening Adaptor', 11000),
    ('I-0010', 'S-028', 'Brown Sugar Coffee', 3300),
    ('I-0015', 'S-028', 'Green Tea', 3300),
    ('I-0288', 'S-101', 'Mevius Cigarette Pack', 2600),
    ('I-0275', 'S-101', 'Pilot Pen Blue', 5400),
    ('I-3286', 'S-119', 'ABC Iced Coffee', 700),
    ('I-3289', 'S-119', 'Vaseline Body Lotion', 9700),
    ('I-0006', 'S-020', 'Kyay Oh Si Chat Pork', 5500),
    ('I-0009', 'S-020', 'Prawn Tanpura', 7000);
```

120 %

Messages

```
(26 rows affected)
```

Result

```
SELECT * FROM items;
```

120 %

Results | Messages

	item_code	shop_id	item_name	price
1	I-0006	S-020	Kyay Oh Si Chat Pork	5500.00
2	I-0009	S-020	Prawn Tanpura	7000.00
3	I-0010	S-028	Brown Sugar Coffee	3300.00
4	I-0015	S-028	Green Tea	3300.00
5	I-0126	S-019	Nestle Milo Jar	6500.00
6	I-0130	S-019	Jordan Toothbrush	2400.00
7	I-0148	S-019	Shark Energy Drink Can	1400.00
8	I-0275	S-101	Pilot Pen Blue	5400.00
9	I-0288	S-101	Mevius Cigarette Pack	2600.00
10	I-1077	S-045	Y-Si Vitamin-C	3200.00
11	I-1086	S-045	Air-X Tablet	1600.00
12	I-1090	S-045	Cloth Face Mask	1200.00
13	I-1910	S-035	Apple Original Earphone	45000.00
14	I-1914	S-035	Type C to Lightening Adaptor	11000.00
15	I-2048	S-012	Remax TWS Ear Buds	32000.00
16	I-2067	S-012	Green Tech Wireless Mouse	16500.00
17	I-2138	S-097	Chicken Curry Puff	5800.00
18	I-2144	S-097	Cheesecake Slice	4200.00
19	I-2145	S-097	Chocolate Cake	4000.00
20	I-2150	S-097	Vanilla Ice Cream	2700.00
21	I-2712	S-023	Spicy Fried Chicken	2200.00
22	I-2716	S-023	Chicken Burger	5200.00
23	I-3144	S-114	Cosrx Acne Patch	1800.00
24	I-3178	S-114	A&C Mask Sheet	2700.00
25	I-3286	S-119	ABC Iced Coffee	700.00
26	I-3289	S-119	Vaseline Body Lotion	9700.00

Insert Data into Orders Table

```
INSERT INTO orders
    (order_id, customer_id, rider_id, payment_type_id, township_code, delivery_address)
VALUES
    ('ORD-4512', 'GC-0067', 'GR-22', 2, 11121, 'Yangon International Hotel, Pyi Htaung Su Road'),
    ('ORD-4513', 'GC-0033', 'GR-03', 2, 11081, 'Yankin Tower, Sayar San Road'),
    ('ORD-4514', 'GC-0099', 'GR-11', 2, 11201, 'No.43, Bahan 3th Street');

INSERT INTO orders VALUES
    ('ORD-3111', 'GC-0012', 'GR-07', 2, 11201, '2022-08-16 17:45:00', '2022-08-16 18:10:00',
        'Phoe Sein Hotel, Bo Myat Htun Street', 'delivered'),
    ('ORD-3114', 'GC-0012', 'GR-08', 1, 11052, '2022-08-18 7:15:00', '2022-08-18 7:35:00',
        'No.5, Thu Kha Street', 'delivered'),
    ('ORD-3131', 'GC-0028', 'GR-12', 3, 11052, '2022-09-19 13:30:00', '2022-09-19 13:50:00',
        'No.43, Thu Kha Street', 'delivered'),
    ('ORD-3143', 'GC-0093', 'GR-14', 2, 11211, '2022-09-23 12:45:00', '2022-09-23 13:15:00',
        'No.88, Kyar Kwat Thit Street', 'delivered'),
    ('ORD-3155', 'GC-0004', 'GR-11', 1, 11201, '2022-09-24 15:05:00', '2022-09-24 15:38:00',
        'No.3, U Chit Maung Housing', 'delivered'),
    ('ORD-3202', 'GC-0021', 'GR-13', 4, 11141, '2022-09-25 8:15:00', '2022-09-25 8:40:00',
        'No.66, 19th Street', 'delivered'),
    ('ORD-3205', 'GC-0092', 'GR-22', 2, 11121, '2022-09-29 11:45:00', NULL,
        'No.121, Aung Myin Street', 'canceled'),
    ('ORD-3206', 'GC-0092', 'GR-22', 2, 11121, '2022-09-29 11:52:00', '2022-09-29 12:16:00',
        'No.121, Aung Myin Street', 'delivered');
```

120 %

Messages

```
(3 rows affected)

(8 rows affected)
```

Result

```
SELECT * FROM orders;
```

120 %

Results | Messages

	order_id	customer_id	rider_id	payment_type_id	township_code	order_datetime	delivered_datetime	delivery_address	order_status
1	ORD-3111	GC-0012	GR-07	2	11201	2022-08-16 17:45:00.000	2022-08-16 18:10:00.000	Phoe Sein Hotel, Bo Myat Htun Street	delivered
2	ORD-3114	GC-0012	GR-08	1	11052	2022-08-18 07:15:00.000	2022-08-18 07:35:00.000	No.5, Thu Kha Street	delivered
3	ORD-3131	GC-0028	GR-12	3	11052	2022-09-19 13:30:00.000	2022-09-19 13:50:00.000	No.43, Thu Kha Street	delivered
4	ORD-3143	GC-0093	GR-14	2	11211	2022-09-23 12:45:00.000	2022-09-23 13:15:00.000	No.88, Kyar Kwat Thit Street	delivered
5	ORD-3155	GC-0004	GR-11	1	11201	2022-09-24 15:05:00.000	2022-09-24 15:38:00.000	No.3, U Chit Maung Housing	delivered
6	ORD-3202	GC-0021	GR-13	4	11141	2022-09-25 08:15:00.000	2022-09-25 08:40:00.000	No.66, 19th Street	delivered
7	ORD-3205	GC-0092	GR-22	2	11121	2022-09-29 11:45:00.000	NULL	No.121, Aung Myin Street	canceled
8	ORD-3206	GC-0092	GR-22	2	11121	2022-09-29 11:52:00.000	2022-09-29 12:16:00.000	No.121, Aung Myin Street	delivered
9	ORD-4512	GC-0067	GR-22	2	11121	2023-01-12 06:57:27.043	NULL	Yangon International Hotel, Pyi Htaung Su Road	pending
10	ORD-4513	GC-0033	GR-03	2	11081	2023-01-12 06:57:27.043	NULL	Yankin Tower, Sayar San Road	pending
11	ORD-4514	GC-0099	GR-11	2	11201	2023-01-12 06:57:27.043	NULL	No.43, Bahan 3th Street	pending

Insert Data into Orders Items Table

```
INSERT INTO order_items VALUES
    ('ORD-3111', 'I-1090', 3),
    ('ORD-3111', 'I-1086', 1),
    ('ORD-3111', 'I-1077', 1),
    ('ORD-3114', 'I-2144', 3),
    ('ORD-3114', 'I-2150', 1),
    ('ORD-3202', 'I-1910', 1),
    ('ORD-3202', 'I-1914', 1),
    ('ORD-3205', 'I-0006', 1),
    ('ORD-4513', 'I-0006', 2),
    ('ORD-4513', 'I-0009', 1);
```

120 %

Messages

```
(10 rows affected)
```

Result

```
SELECT * FROM order_items;
```

120 %

Results | Messages

	order_id	item_code	quantity
1	ORD-3111	I-1077	1
2	ORD-3111	I-1086	1
3	ORD-3111	I-1090	3
4	ORD-3114	I-2144	3
5	ORD-3114	I-2150	1
6	ORD-3202	I-1910	1
7	ORD-3202	I-1914	1
8	ORD-3205	I-0006	1
9	ORD-4513	I-0006	2
10	ORD-4513	I-0009	1

Suitable data, including data provided in sample documents, are inserted into each table in the same order of tables are created. For some tables which have a column with a default value, there will be two insert statements – one for inserting all fields and one for inserting only mandatory data. Referencing errors are frequently encountered because some referential data are not inserted. For example, shops are inserted only for six townships although there are ten townships in the database. And riders are inserted only for five of those six townships. So, for inserting order items, shops are limited to five.

Task – 6

Task 6 – SQL reports

Query 1

Retrieving shop names and how many orders each shop get

```
SELECT
    s.shop_name AS 'Shop',
    COUNT (DISTINCT o.order_id) AS 'Number of Orders'
    FROM orders o
    JOIN order_items oi
        ON oi.order_id = o.order_id
    JOIN items i
        ON i.item_code = oi.item_code
    JOIN shops s
        ON s.shop_id = i.shop_id
    GROUP BY s.shop_name;
```

120 %

Results | Messages

	Shop	Number of Orders
1	Moe's Bakery	1
2	Shwe Oh Pharmacy	1
3	Technoland	1
4	Yankin Kyay Oh	2

Query 2

Retrieving shop categories and how many shops in each category has in descending order

```
SELECT sc.shop_category_name AS 'Shop Category',
    COUNT(s.shop_id) AS 'Number of Shops'
    FROM shop_categories sc
    JOIN shops s
        ON s.shop_category_id = sc.shop_category_id
    GROUP BY sc.shop_category_name
    ORDER BY 'Number of Shops' DESC;
```

120 %

Results | Messages

	Shop Category	Number of Shops
1	Food	4
2	Convenience	2
3	Electronic	2
4	Grocery	1
5	Health	1
6	Beauty	1

Query 3

Retrieving three most used payment type and how many times they were used on previous orders

```
SELECT pt.payment_type_name AS 'Payment Method',
    COUNT(o.payment_type_id) AS 'Number of Orders'
    FROM orders o
    JOIN payment_types pt
        ON pt.payment_type_id = o.payment_type_id
    WHERE o.order_status = 'delivered'
    GROUP BY pt.payment_type_name
    ORDER BY 'Number of Orders' DESC
        OFFSET 0 ROW
        FETCH NEXT 3 ROW ONLY;
```

120 %

Results | Messages

	Payment Method	Number of Orders
1	KBZ Pay	3
2	Cash on Delivery	2
3	Wave Pay	1

Query 4

Retrieving the customer who ordered the most in 2022

```
SELECT c.customer_name AS 'Most Ordered Customer',
    COUNT(o.order_id) AS 'Number of Orders'
    FROM customers c
    JOIN orders o
        ON o.customer_id = c.customer_id
    WHERE o.order_status != 'canceled' AND YEAR(o.order_datetime) = 2022
    GROUP BY c.customer_name
    HAVING COUNT(o.order_id) = (SELECT MAX(tbl.num_of_orders)
                                FROM (
                                    SELECT customer_id,
                                        COUNT(order_id) AS num_of_orders
                                    FROM orders
                                    GROUP BY customer_id ) tbl
                                );
```

120 %

Results | Messages

	Most Ordered Customer	Number of Orders
1	Kyaw Thu	2

Query 5

Retrieving total number of orders for each month in 2022

```
SELECT
    DATENAME(m,order_datetime) AS 'Month',
    COUNT(MONTH(order_datetime)) AS 'Total Orders'
    FROM orders
    WHERE YEAR(order_datetime) = 2022
    GROUP BY DATENAME(m, order_datetime);
```

120 %

Results | Messages

	Month	Total Orders
1	August	2
2	September	6

Query 6

Retrieving member types and how many customers each member type has in descending order

```
SELECT mt.member_type_name AS 'Member Type',
    COUNT(mt.member_type_name) AS 'Number of Customers'
    FROM member_types mt
    JOIN customers c
        ON c.member_type_id = mt.member_type_id
    GROUP BY mt.member_type_name
    ORDER BY COUNT(mt.member_type_name) DESC;
```

120 %

Results | Messages

	Member Type	Number of Customers
1	Bronze	4
2	Silver	3
3	Gold	2
4	Platinum	1

Query 7

Retrieving available riders in Bahan Township

```
SELECT
    r.rider_name AS 'Rider Name',
    r.rider_id AS 'Rider ID',
    t.township_name AS 'Township'
    FROM townships t
    JOIN riders r
        ON r.township_code = t.township_code
    JOIN orders o
        ON o.rider_id = r.rider_id
    WHERE  o.order_status != 'pending' AND  t.township_name = 'Bahan';
```

120 %

Results | Messages

	Rider Name	Rider ID	Township
1	Soe Win	GR-07	Bahan
2	Htun Aye	GR-11	Bahan

Query 8

Retrieving riders who live in ‘Bahan’ or ‘Alone’ twonship and how many pending or completed orders they have

```
SELECT r.rider_name AS 'Rider Name',
    t.township_name AS 'Township',
    COUNT(o.order_id) AS 'Completed Orders'
    FROM orders o
    JOIN riders r
        ON r.rider_id = o.rider_id
    JOIN townships t
        ON t.township_code = r.township_code
    WHERE t.township_name IN ('Bahan','Alone') AND o.order_status != 'canceled'
    GROUP BY r.rider_name, t.township_name;
```

120 %

Results | Messages

	Rider Name	Township	Completed Orders
1	Htun Aye	Bahan	2
2	Min Min	Alone	2
3	Soe Win	Bahan	1

Query 9

Retrieving order details for order ID – ‘ORD-3111’ including shop name, price, buy quantity and subtotal for each item

```
SELECT oi.order_id AS 'Order ID',
    i.item_name AS 'Item Name',
    s.shop_name AS 'Shop Name',
    i.price AS 'Unit Price',
    oi.quantity AS 'Quantity',
    i.price*oi.quantity AS 'Sub Total'
    FROM order_items oi
    JOIN items i
        ON i.item_code = oi.item_code
    JOIN shops s
        ON s.shop_id = i.shop_id
    WHERE oi.order_id = 'ORD-3111';
```

120 %

Results | Messages

	Order ID	Item Name	Shop Name	Unit Price	Quantity	Sub Total
1	ORD-3111	Y-Si Vitamin-C	Shwe Oh Pharmacy	3200.00	1	3200.00
2	ORD-3111	Air-X Tablet	Shwe Oh Pharmacy	1600.00	1	1600.00
3	ORD-3111	Cloth Face Mask	Shwe Oh Pharmacy	1200.00	3	3600.00

Query 10

Retrieving order summary for order ID ‘ORD-3111’

```sql
SELECT o.order_id AS 'Order ID',
    c.customer_name AS 'Customer Name',
    r.rider_name AS 'Rider Name',
    pt.payment_type_name AS 'Payment Type',
    t.township_name AS 'Township',
    total_tbl.shop_name AS 'Shop Name',
    qty_tbl.qty AS 'Total Items',
    total_tbl.total AS 'Total Amount',
    FORMAT(o.order_datetime, 'MMM dd yyyy') AS 'Order Date',
    FORMAT(o.order_datetime, 'hh:mm tt') AS 'Order Time',
    FORMAT(o.delivered_datetime, 'hh:mm tt') AS 'Delivered Time',
    o.delivery_address AS 'Delivery Address'
    FROM orders o
    JOIN customers c
        ON c.customer_id = o.customer_id
    JOIN riders r
        ON r.rider_id = o.rider_id
    JOIN payment_types pt
        ON pt.payment_type_id = o.payment_type_id
    JOIN townships t
        ON t.township_code = o.township_code
    JOIN (SELECT order_id, SUM(quantity) qty
            FROM order_items
            GROUP BY order_id) qty_tbl
        ON qty_tbl.order_id = o.order_id
    JOIN (SELECT oi.order_id AS order_id, s.shop_name AS shop_name, SUM(i.price*oi.quantity) AS total
            FROM order_items oi
            JOIN items i ON i.item_code = oi.item_code
            JOIN shops s ON s.shop_id = i.shop_id
            GROUP BY oi.order_id, s.shop_name) total_tbl
        ON total_tbl.order_id = o.order_id
    WHERE o.order_id = 'ORD-3111';
```

120 %

Results | Messages

	Order ID	Customer Name	Rider Name	Payment Type	Township	Shop Name	Total Items	Total Amount	Order Date	Order Time	Delivered Time	Delivery Address
1	ORD-3111	Kyaw Thu	Soe Win	KBZ Pay	Bahan	Shwe Oh Pharmacy	5	8400.00	Aug 16 2022	05:45 PM	06:10 PM	Phoe Sein Hotel, Bo Myat Htun Street

Query executed successfully.

Task – 7

Task 7 – Assessment of Design and Implementation

Logical and Physical Design

A well-functioning database is completed designed for Genie Delivery Service through the following steps. First, sample documents are collected to determine what data is to be stored in the database. Entities and relationships between them are identified from the given scenario and a conceptual design (an entity relationship diagram) is drawn for the database. The logical design of the database is developed by normalizing all sample documents and drawing an ER diagram for each normalization. And then the relationships between entities are checked again and the final ER diagram for the organization is obtained. And an accompanying data dictionary is produced by identifying what attributes are to be held by a particular entity as well as their data types, sizes and constraints – primary key, foreign key, propagation and domain.

For physical design of the database, SQL server is used as a DBMS. Using SQL server management studio as a managing tool, a database is created as well as tables for each entity in the same order as they are described in the data dictionary using CREATE statement which is a DDL language – subset of SQL. All constraints are also defined right away in the CREATE statement without using ALTER statement to modify each table. Then suitable data are inserted into all tables using the INSERT statement.

Data Retrieving

For retrieving data, the SELECT statement, most commonly used in DML language is used joining multiple tables with JOIN keyword. For a more user-friendly result of the response, alias, temporary names to define each column, are used in every query. Most of the data retrieved from the database in task 6 are more related to business insights rather than day-to-day operations of the business, yet they are provided here by writing complex queries to demonstrate how the database implementation met specific requirements of the organization.

Derived Data

Some information required for the business should not need to be provided by users when it can be easily derived from related columns of the same table or other related tables in the database. For example, when a customer orders a number of items from a shop, subtotals of each item, total, VATs and grand total would need to be provided in the order form. A part of a customer order form is as below.

Sample Order Form

Order ID: ORD-3111

Customer Name: Kyaw Thu

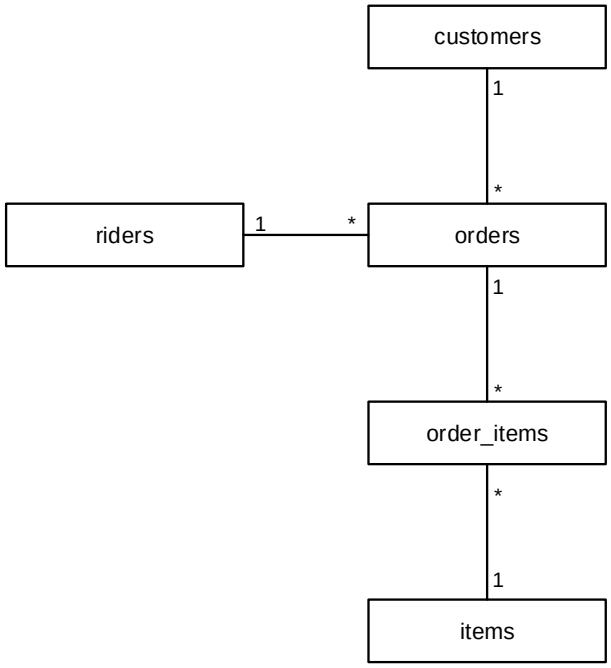
Rider Name: Soe Win

Date: August 16, 2022

No	Item Name	Unit Price	Quantity	Sub Total
1	Y-Si Vitamin C	3200	1	3200
2	Air X Tablet	1600	1	1600
3	Cloth Face Mask	1200	3	3600
			Total:	8400
			VAT (5%):	420
			Grand Total:	8820

For the above order form, users should not need to provide information in the highlighted column since they can be derived from other data. The derived data can be stored in the database by specifying the calculation in the related table schema or it can be easily calculated at the runtime. For the database for Genie Delivery Service, it is decided not to store derived data to reduce data redundancy. So, to get derived data, it needs to specify in the query by using manual calculation or aggregate functions. A pseudo code to retrieve information in the above order form is shown in the following figure along with the ER diagram.

Entity Relationship Diagram and Pseudo Code for Deriving Data



Information for above order form can be obtained by querying this

orders.order_id
riders.rider_name
customers.customer_name
orders.order_date
items.item_name
items.unit_price
order_items.quantity
/Subtotal (order_items.quantity * items.quantity where order_items.item_code = items.item_code)
/Total (sum(Subtotal))
/VAT (Total * 0.05)
/Grand Total (Total + VAT)

Task – 8

Task 8 – Future Development of a Data Warehouse

Data Warehouse

A data warehouse is a central repository of information that is specifically designed for data analytics, which involves reading large amounts of data to understand relationships and trends across the data. It is typically populated with data from a variety of sources, including transactional systems, external data feeds, and other databases.

A data warehouse is used by a broad group of people in an organization including business analysts, data scientists and business intelligence professionals. They use the data and insights from the data warehouse to make strategic decisions, track performance, build predictive models, perform advanced analytics, and create reports and visualizations. Data warehouses power these features by storing data efficiently to minimize the input and output of data and delivering quick query results to hundreds and thousands of users concurrently. (Anon., n.d.)

Why the organization would need a data warehouse

As the amount of data collected by Genie Delivery increases, it will become difficult to manage, analyze, and create reports out of that data. And storing large amounts of data that are not used to perform day-to-day operation may slow down the performance of transactions. Such data can be used to analyze the business and so they should not be permanently deleted. For such a scenario, building a data warehouse can be beneficial. A data warehouse will allow the organization to consolidate and analyze large amounts of data from various sources, such as townships, riders, customer orders, and financial data. This will enable the organization to perform advanced analytics and reporting, gain insights on customer behavior and preferences, and optimize the operations to improve customer satisfaction and retention. A data warehouse can also be used to identify fraudulent activities, such as duplicate orders, refunds, or chargebacks. (Anon., n.d.)

Components of a data warehouse

To build a data warehouse for a delivery service organization, the following components will be needed.

- Data sources – All the databases and systems where the company's data is stored, such as orders, payment systems, riders and customers
- ETL (Extract, Transform, Load) tools – Tools to extract data from several sources, transform it into a single format that can be loaded into the data warehouse
- Meta data – Information about the structure, content, and context of the data and data integration processes
- Reporting and analysis tools – Tools for business users and analysts to query the data warehouse and create reports and visualizations
- Data governance and security – A plan to ensure that data is accurate, complete, and secure, and that data access is controlled and audited
- Data modeling – A process of arranging pieces of data in a data mart using schemas such as star, snowflake and starflake
- Technical expertise – Expertise in database design, ETL, and data modeling

(Anon., n.d.)

Task – 9

Task 9 – Distributed Database Option

Distributed Database

A distributed database is a group of databases spread out across several sites connected by a network. It appears as a single logical database located at one site since all databases are connected. (Moore, 2022) (Anon., 2020)

Distributed Database Architectures

Distributed databases can be homogenous or heterogeneous depending on the architecture. In a homogenous system, hardware, softwares and database management systems are uniform across all the sites and the data structure at each site must be identical or compatible while heterogeneous architecture allows different sites to have different physical resources and attributes. If Genie Delivery wants to expend business to other cities, a distributed database should be designed in homogenous architecture to make the system consistent and easy to manage and deploy for all sites. (Moore, 2022) (Anon., 2020)

Distributed Database Implementation Methods

A distributed database can be implemented either by fragmentation or replication depending on specific requirements of the organization. In fragmentation, data is split into smaller chunks and distributed those chunks across the different relative sites. Fragmentation or partitioning can be done by horizontally or vertically. In horizontal partitioning, data is split by rows by defining a partitioning key to decide which site the rows belong to. Horizontal partitioning can reduce the response time by skipping non-relevant partitions for a query. But if the organization often needs to analyze all data in the database, horizontal partitioning cannot beneficial and vertical partitioning can be used instead. Vertical partitioning split data by columns and it can significantly reduce costs associated with accessing data. Although vertical partitioning is helpful, manipulating data from the whole row such as a delete statement can be an issue since the operation needs to be run on every partition.

For replication, instances of data can reside at more than one site at any given time. One of the site is designated as the primary site and data is synced with other sites periodically – synchronously or asynchronously. If one of the site needs to upgrade or experiences a downtime, applications can switch to other sites to keep transactions running without waiting time. Replication offers speed and consistency but for large volumes of data, more disk space is needed across different sites, bumping up costs. (Anon., 2020) (Subramanian, 2020)

Factors that need to be considered in designing a distributed database

To design an efficient and effective distributed database that meets the specific requirements of the organization, there are several factors that needs to be considered including:

- Data Volume and Nature
- Expected Workload
- Availability and Performance
- Network infrastructure
- Hardware Resources
- Backup and Recovery
- Security
- Cost and Complexity
- Data Partitioning and Replication Strategy
- Scalability
- Monitoring and Management

(Subramanian, 2020) (Moore, 2022)

References

Anon., 2020. *Fauna.* [Online]
Available at: https://fauna.com/blog/the-why-and-how-of-distributed-databases
[Accessed 9 1 2023].

Anon., n.d. *AWS.* [Online]
Available at: https://aws.amazon.com/data-warehouse/
[Accessed 11 1 2023].

Anon., n.d. *SAP.* [Online]
Available at: https://www.sap.com/insights/what-is-a-data-warehouse.html
[Accessed 10 1 2023].

Moore, L., 2022. *Tech Target.* [Online]
Available at: https://www.techtarget.com/searchoracle/definition/distributed-database
[Accessed 9 1 2023].

Subramanian, M., 2020. *Designing Factors of Distributed Database System : A Review,* s.l.: International Journal of Knowledge Engineering and Data Mining. 12 7-10.

Candidate Checklist

Please use the following checklist to ensure that your work is ready for submission.

Have you read the NCC Education documents 'What is Academic Misconduct? Guidance for Candidates' and 'Avoiding Plagiarism and Collusion: Guidance for Candidates' and ensured that you have acknowledge all the sources that you have used in your work?	✔
Have you completed the 'Statement and Confirmation of Own Work' form and attached it to your assignment? You must do this.	✔
Have you ensured that your work has not gone over or under the recommended word count by more than 10%?	✔
Have you ensured that your work does not contain viruses and can be run directly?	✔