UNIVERSITI TEKNOLOGI MALAYSIA

SCHOOL OF COMPUTING

SESSION 2020/2021

SEMESTER 1

CODE & SUBJECT: SECD2523 - DATABASE

ALTERNATIVE ASSESSMENT

ENHANCEMENT OF E-TICKETING SYSTEM (KIOSK)

NAME: NURUL SYAMIRA BINTI AMAT JIFRI

MATRICS NO: A19EC0145

SECTION: 09

LECTURER'S NAME: DR. NOOR HIDAYAH ZAKARIA

INTRODUCTION

Overview of current system

The kiosk interface allows a ticket for any entertainment places such as zoo, space center, museum, and other places to be obtained easily and quickly by customers. Customers are required to log into the kiosk machine whether login as a guest or using an existing account. An email address is required if the customer chooses to login as a guest. After the customer books the ticket, the kiosk will print out a payment receipt for the booked ticket to be paid within a particular time. Customers will have a choice to pay the ticket using few methods, such as cash and credit or debit. The e-ticket will be directly sent through the customer's email address once the transaction is successful. In planning, the kiosk machine will be installed in several places with high flow of customers and ease of accessibility to be reached by everyone. For example, convenience stores (7-Eleven, Family mart, etc), shopping mall, terminal bus and airport.

This kiosk system is now enhanced with the existence of sufficent information for the customers' awareness. This is shown when customer purchase a ticket, a list of precise information must be informed to the customers. For example, the ticket must occupy the expiry date so that customer is informed with the validation of the ticket that they purchased. This will remind the customer the period of the ticket can be used at their chosen entertainment place after purchasing it.

Next, a purchase made by a member of existing account can be rewarded with collection of points. Becoming a verified member of any e-ticketing system will be beneficial if they can collect points each time they make a purchase of tickets. The members are verified by their member identification number and has a number of collected points.

Moreover, customers' details will be more precise with the addition of their name and telephone number. If there is any issue happened, they can be notified easily by their phone number in case by their email address, they did not get the information.

Mission Statement

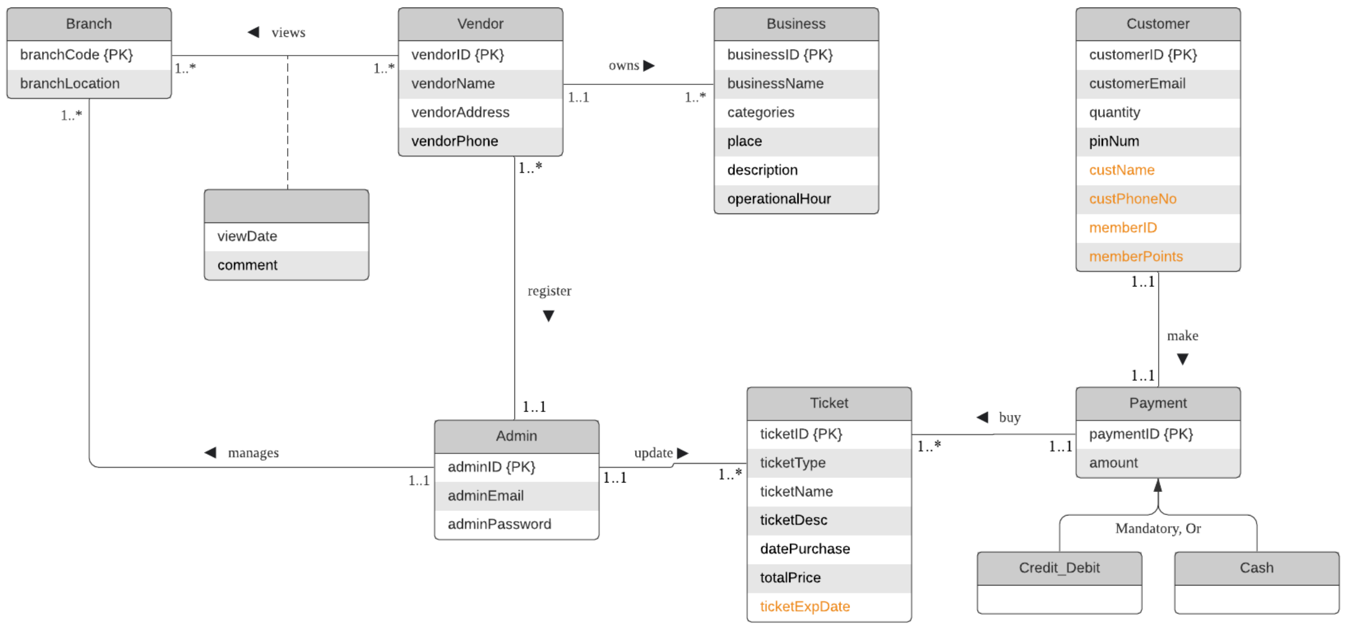
- Save customers' time. This will cut customers' waiting time so that thay will have more positive experience using the kiosk and likely, will be using the service again.
- More convenient to use. So, it will be easily accessible by cutomers to perform their purchase for specific ticket kiosk with efficient features such as ticket pricing and availability.
- Provide an easier payment method. Our target is to allow all different ranges of customers to use the e-ticketing system. If any of them is not familiar with advanced technology method, paying with cash is still available.
- Give awareness to the customer. By knowing the expiry date, the customer will not face a loss after purchasing ticket and could not use it as the ticket is not valid anymore.

Mission Objectives

- Improves customer buying experience. Kiosks can provide detailed information about the product (ticket) and services. Since kiosks are easily accessible, customers will find it convenient to visit a kiosk for inquiries. It will also instil confidence to potential customers since we are using the latest technologies.
- Simple and easy. The element of simplicity is important to developers for being able to meet the customer's needs. As no one likes to use a system which requires a lot of steps, therefore our interface design should be simple and user-friendly. We want to provide customers with an environment that makes them feel at home.
- Productivity improvements. Minimal the analyst's jobs, speed up processes, reduce human error and better meet customer expectations.
- Give customer good experience and great impression. With the feature of member get to collect points everytime purchasing, they will feel happy and satisfied. Thus, they may buy ticket from this e-ticketing system again, also may say good words to other people about it.

SUMMARY OF DATABASE PLANNING AND DESIGN: 2a

Updated Conceptual ERD



*****Attributes with coloured is the enhancement – new attributes added.**

Updated Data Dictionary

Entity name	Attributes	Description	Data Type & Length	Nulls
Vendor	vendorID	Unique id for client	5 variable char	No
	vendorName	Name of client	15 variable char	No
	vendorAddress	Office address of the client	35 variable char	Yes
	vendorPhone	Phone number	10 integers	No
Business	businessID	Unique id for each business	5 variable char/int	No
	businessName	Business name	30 variable char	No
	categories	Business categories	15 variable char	Yes
	place	Address business location	30 variable char	Yes
	description	Detail of business	Text	Yes
	operationalHour	open/close hours	Time	Yes
Admin	adminID	Unique id for admin	5 variable char/int	No
	adminEmail	Unique email for admin		No
	adminPassword	password for security		No
Ticket	ticketID	Ticket reference No	10 variable char	No
	ticketType	adult/kids or senior citizen	1 char (A, K, S)	Yes
	ticketName	Name of ticket	15 variable char	No
	ticketDesc	Ticket description/details	Text	Yes
	datePurchase	Date of purchase	Date	Yes
	totalPrice	Total ticket price	Int	No
	ticketExpDate	Expiry date of the ticket	Date	No
Customer	customerID	Unique id for customer	5 variable char/int	No
	customerEmail	Unique email	Text (unique)	No
	quantity	Quantity ticket cust buy	Number	No
	pinNum	Password	Number	Yes
	custName	Customer name	30 variable char	No

	custPhoneNo	Customer telephone number	10 integers	No
	memberID	Unique id for existing members	6 variable char/int	Yes
	memberPoints	Collected points for members	Int	Yes
Payment	paymentID	Unique payment reference	5 variable char/int	No
	amount	Total customer pay	Int	No
	paymentMethod	Cash/credit/debit	Char	Yes
Branch	branchCode	Unique id for branch	Text (unique)	No
	branchLocation	Location of branch located	Text (unique)	No

Updated system's functional requirements

Data Requirements

a. **Vendor**. Refers to a person who is a business partner of the e-ticketing system that is interested to join this platform. To become a vendor, that person should register their business to admin.

The data stored on the vendor include a vendor identification number (vendorID), name (vendorName), address of the vendor's office (vendorAddress), telephone number (vendorPhone). The vendorID is unique.

b. **Business**. A vendor may run one or more businesses.

The data required on business includes business identification number (businessID), the name of the business (businessName), categories, place, business description (description), operational hours of the business (operationalHour).

c. **Admin**. Refers to a person who is responsible for user administration and maintaining the system. Admin holds and manages every data from the vendor and customer.

The data this entity hold is the identification number of admin (adminID), email address of admin (adminEmail), password that created by admin (adminPassword).

d. **Customer**. Refers to a person who buys tickets from a kiosk. There are two main types of customers: members and non-members (guests).

The data stored for each customer which is a member are member id, pin number and email while for non-members is email. The customerID is unique. As the system needs the basic information of the customer, so there are two attributes added: customer's name (custName) and telephone number (custPhoneNo). The system provides two options for the customer: as an existing member or guest. Hence, attributes that are focus on the member is added includes member identificaton number (memberID) to prove they are the verified member in the system also, the members' collected points after purchasing ticket (memberPoints). It can be a null value as some customers login as guests.

d. **Branch**. Information about a kiosk that is placed at different brach.

The data includes the identification of the branch (branchCode) and the location of the branch (branchLocation). The branchCode is unique.

e. **Ticket**. Tickets that are purchased by customers at the kiosk. The information on each ticket following the customer's choice when purchasing.

The data stored for each ticket is (ticketID), (ticketType), (ticketName), description (ticektDesc), (datePurchase) and (totalPrice). The ticketID is unique. An attribute that is added is the expiry date of the ticket that the customers purchase (ticketExpDate). By adding this, customers is informed with the period of validation for the ticket.

f. **Payment**. Information received by the customer after they have done the payment.

The data stored are identification number to each payment (paymentID), total amount that (amount), the method that customer prefers: cash or credit/debit (paymentMethod). The paymentID is unique.

Transaction Requirements

Data entry

a. Enter details of vendor and vendor business

b. Enter the details of the customer

c. Enter details of a new branch

d. Enter details of tickets (such as ticket name provided by client wax museum)

e. Enter details of payment (customers)

f. Enter details of members (customers)

Data update/deletion

a. Update/delete the detail of vendor

b. Update/delete the detail of the business

c. Update/delete the details of the branch

d. Update/delete the details of the ticket information

e. Update payment/refund details

f. Update/delete invoice

g. Update the collected points of members who has purchased ticket

Data queries

a. List details of all branch information

b. List of vendor information

c. List details of businesses that are run by a vendor

d. List details of the customer

e. Display ticket information

f. Display details of total payment by certain customer

g. Identify total ticket sales by a certain business

h. Display the details of rates made by customers

i. List the details of ticket that have not been purchased for more than three month

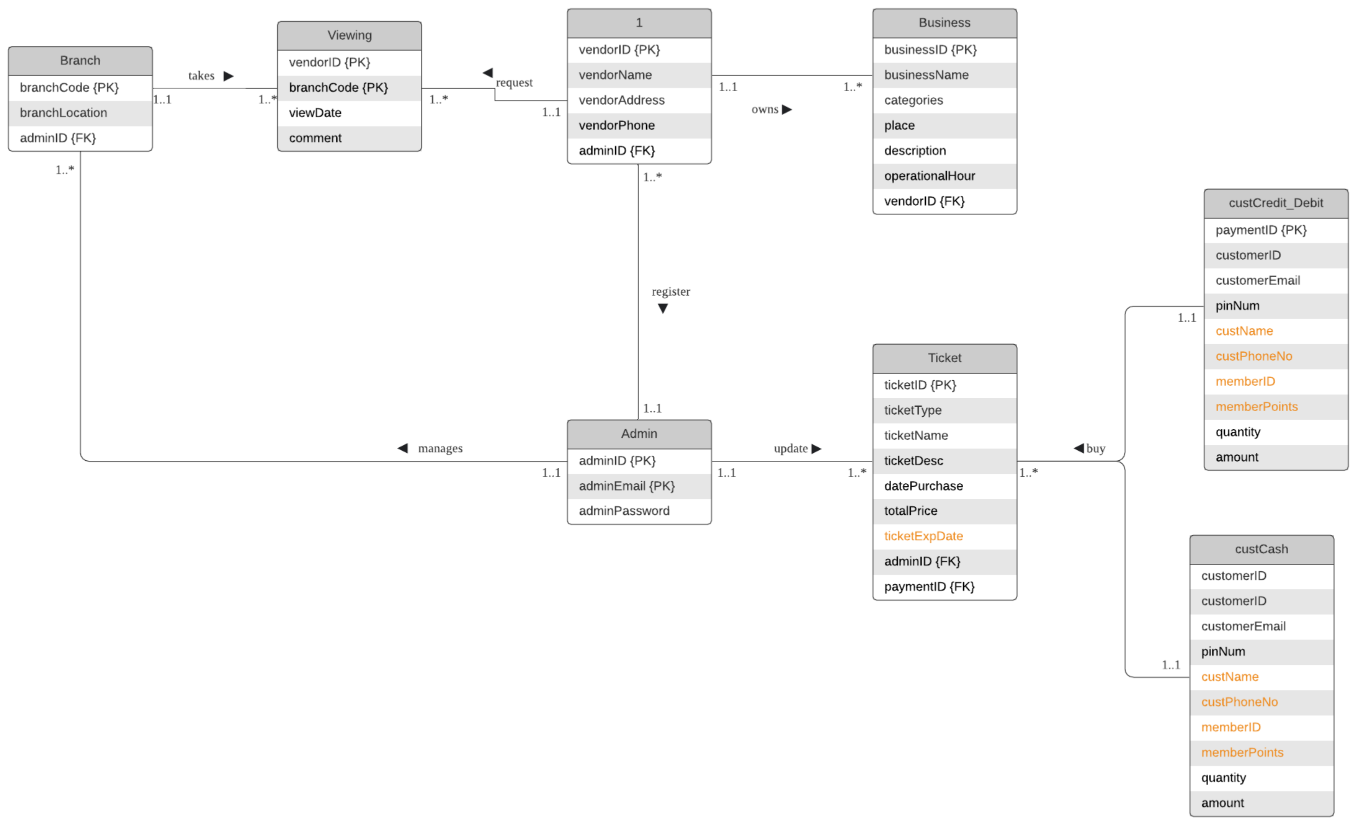
j. Produce a list on which ticket type customer most prefer

k. Present an invoice listing details of the transaction

l. Display collected points by the members

SUMMARY OF DATABASE PLANNING AND DESIGN: 2b

Updated Logical ERD

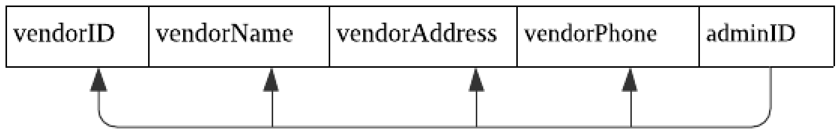


Relational Table for Logical Data Model

Business (businessID, businessName, categories, place, description, operationalHour, vendorID) **Primary Key** businessID **Alternate Key** businessName **Foreign Key** vendorID **references** Vendor(vendorID)	**Vendor** (vendorID, vendorName, vendorAddress, vendorPhone, AdminID) **Primary Key** vendorID **Foreign Key** adminID **references** Admin(adminID)
Branch (branchCode, branchLocation, AdminID) **Primary Key** branchCode **Foreign Key** adminID **references** Admin(adminID)	**Viewing** (branchCode, vendorID, viewDate, comment) **Primary Key** branchCode **references** branchCode **Primary Key** vendorID **references** Vendor(vendorID)
Admin (adminID, adminEmail, adminPassword) **Primary Key** adminID	**custCredit_Debit** (paymentID, customerID, customerEmail, pinNum, custName, custPhoneNo, memberID, memberPoints, quantity, amount) **Primary Key** paymentID **Alternate Key** customerID
Ticket (ticketID, ticketType, ticketName, ticketDesc, datePurchase, totalPrice, ticketExpDate, adminID, paymentID) **Primary Key** ticketID **Foreign Key** AdminID **references** Admin(adminID) **Foreign Key** paymentID **references** cust_Pay(paymentID)	**custCash** (paymentID, customerID, customerEmail, pinNum, custName, custPhoneNo, memberID, memberPoints, quantity, amount) **Primary Key** paymentID **Alternate Key** customerID

Relational Database Schemas (normalized up till BCNF)

Vendor Entity Normalization



Vendor (vendorID(key), vendorName, vendorAddress, vendorPhone, adminID)

Primary Key: vendorID

Foreign Key: adminID reference Admin

Candidate Key: vendorName

FD1: vendorID → vendorName, vendorAddress, vendorPhone, adminID (Primary key)

1NF: Meets the definition of relation (already in 1NF)

Vendor(vendorID, vendorName, vendorAddress, vendorPhone, adminID)

2NF: No partial key dependencies (already in 2NF)

Vendor(vendorID, vendorName, vendorAddress, vendorPhone, adminID)

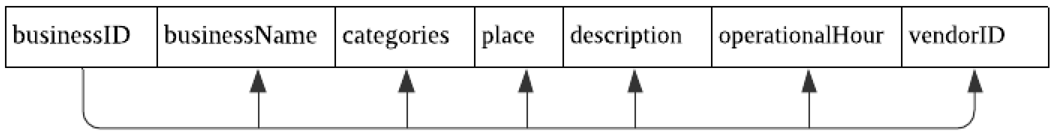
3NF: No transitive dependencies (already in 3NF)

Vendor(vendorID, vendorName, vendorAddress, vendorPhone, adminID)

BCNF: already in BCNF form

Vendor(vendorID, vendorName, vendorAddress, vendorPhone, adminID)

Business Entity Normalization



Business (businessID(key), businessName, categories, place, description, operationalHour, vendorID)

Primary Key: businessID
Foreign Key: vendorID reference Vendor
Candidate Key: businessName

FD1: businessID → businessName, categories, place, description, operationalHour, vendorID

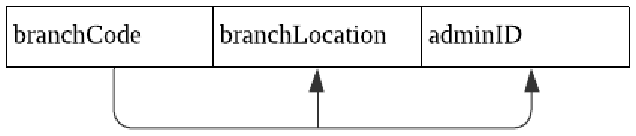
1NF: Meets the definition of relation (already in 1NF)
Business(businessID,businessName, categories, place, description, operationalHour, vendorID)

2NF: No partial key dependencies (already in 2NF)
Business(businessID,businessName, categories, place, description, operationalHour, vendorID)

3NF: No transitive dependencies (already in 3NF)
Business(businessID,businessName, categories, place, description, operationalHour, vendorID)

BCNF: already in BCNF form
Business(businessID,businessName, categories, place, description, operationalHour, vendorID)

Branch Entity Normalization



Branch (branchCode(key), branchLocation, adminID)

Primary Key: branchCode

Foreign Key: adminID reference Admin

Candidate Key: branchLocation

FD1: branchCode → branchLocation, adminID

1NF: Meets the definition of relation (already in 1NF)

Branch(branchCode, branchLocation, adminID)

2NF: No partial key dependencies (already in 2NF)

Branch(branchCode, branchLocation, adminID)

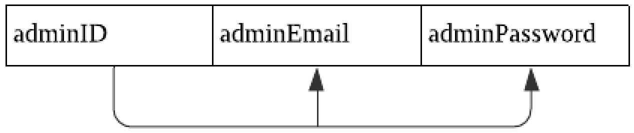
3NF: No transitive dependencies (already in 3NF)

Branch(branchCode, branchLocation, adminID)

BCNF: already in BCNF form

Branch(branchCode, branchLocation, adminID)

Admin Entity Normalization



Admin (adminID(key), adminEmail, adminPassword)

Primary Key: adminID

Candidate Key: adminEmail

FD1: adminID → adminEmail, adminPassword

1NF: Meets the definition of relation (already in 1NF)

Admin(adminID, adminEmail, adminPassword)

2NF: No partial key dependencies (already in 2NF)

Admin(adminID, adminEmail, adminPassword)

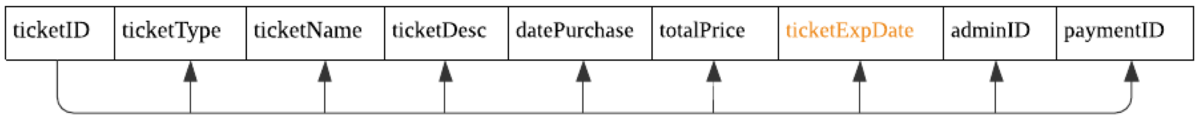
3NF: No transitive dependencies (already in 3NF)

Admin(adminID, adminEmail, adminPassword)

BCNF: already in BCNF form

Admin(adminID, adminEmail, adminPassword)

Ticket Entity Normalization



Ticket (ticketID(key), ticketType, ticketName, ticketDesc, datePurchase, totalPrice, ticketExpDate, adminID, paymentID)

Primary Key: ticketID

Foreign Key: adminID references Admin, paymentID references cust_Pay

Candidate Key: ticketType

FD1: ticketID → ticketType, ticketName, ticketDesc, datePurchase, totalPrice, ticketExpDate, adminID, paymentID

1NF: Meets the definition of relation(already in 1NF)

Ticket(ticketID, ticketType, ticketName, ticketDesc, datePurchase, totalPrice, ticketExpDate, adminID, paymentID)

2NF: was No partial key dependencies(already in 2NF)

Ticket(ticketID, ticketType, ticketName, ticketDesc, datePurchase, totalPrice, ticketExpDate, adminID, paymentID)

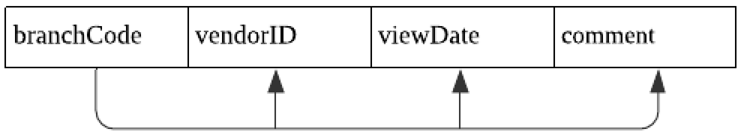
3NF: No transitive dependencies(already in 3NF)

Ticket(ticketID, ticketType, ticketName, ticketDesc, datePurchase, totalPrice, ticketExpDate, adminID, paymentID)

BCNF: already in BCNF form

Ticket(ticketID, ticketType, ticketName, ticketDesc, datePurchase, totalPrice, ticketExpDate, adminID, paymentID)

Viewing Entity Normalization



Viewing (branchCode, vendorID, viewDate, comments)

Primary Key: branchCode, vendorID

FD1: branchCode, vendorID → viewDate, comments (Primary Key)

1NF: Meets the definition of relation (already in 1NF)

Viewing (branchCode, vendorID, viewDate, comments)

2NF: No partial key dependencies (already in 2NF)

Viewing (branchCode, vendorID, viewDate, comments)

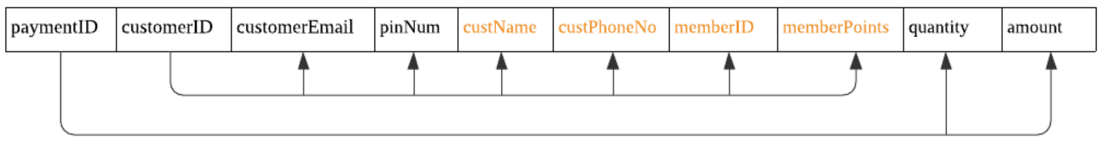
3NF: No transitive dependencies (already in 3NF)

Viewing (branchCode, vendorID, viewDate, comments)

BCNF: already in BCNF form

Viewing (branchCode, vendorID, viewDate, comments)

custCredit_Debit Entity Normalization



custCredit_Debit (<u>paymentID</u>, customerID, customerEmail, quantity, pinNum, custName, custPhoneNo, memberID, memberPoints, amount)

Primary Key : paymentID | Candidate Key: customerID

FD1: customerID → customerEmail, pinNum, custName, custPhoneNo, memberID, memberPoints **(Transitive dependency)**

FD2: paymentID → amount, quantity **(Transitive dependency)**

1NF: Meets the definition of relation (already in 1NF)

cust_Pay(paymentID{PK}, customerID, customerEmail, pinNum, custName, custPhoneNo, memberID, memberPoints, quantity, amount)

2NF: No partial key dependencies (already in 2NF)

cust_Pay(paymentID{PK}, customerID, customerEmail, pinNum, custName, custPhoneNo, memberID, memberPoints, quantity, amount)

3NF: this is not in 3N due to existence of transitive dependency

Split user relation into two new relations named cust_Pay, cust_Info.

cust_Pay(paymentID{PK}, amount, quantity, customerID{FK})

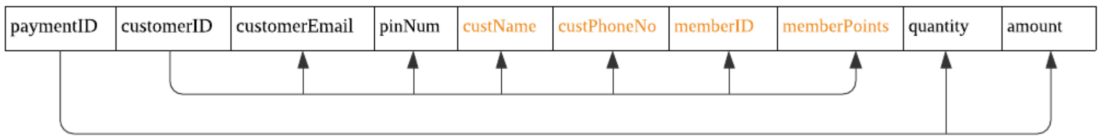
cust_Info(customerID{PK}, customerEmail, pinNum, custName, custPhoneNo, memberID, memberPoints)

BCNF: already in BCNF form

cust_Pay(paymentID{PK}, amount, quantity, customerID{FK})

cust_Info(customerID{PK}, customerEmail, pinNum, custName, custPhoneNo, memberID, memberPoints)

custCash Entity Normalization



custCash(<u>paymentID</u>, customerID, customerEmail, pinNum, custName, custPhoneNo, memberID, memberPoints, quantity, amount)

Primary Key : paymentID | Candidate Key: customerID

FD1: customerID → customerEmail, pinNum, custName, custPhoneNo, memberID, memberPoints **(Transitive dependency)**

FD2: paymentID → amount, quantity **(Transitive dependency)**

1NF: Meets the definition of relation (already in 1NF)

cust_Pay(paymentID{PK}, customerID, customerEmail, pinNum, custName, custPhoneNo, memberID, memberPoints, quantity, amount)

2NF: No partial key dependencies (already in 2NF)

cust_Pay(paymentID{PK}, customerID, customerEmail, pinNum, custName, custPhoneNo, memberID, memberPoints, quantity, amount)

3NF: this is not in 3N due to existence of transitive dependency

Split user relation into two new relations named cust_Pay, cust_Info.

cust_Pay(paymentID{PK}, amount, quantity, customerID{FK})

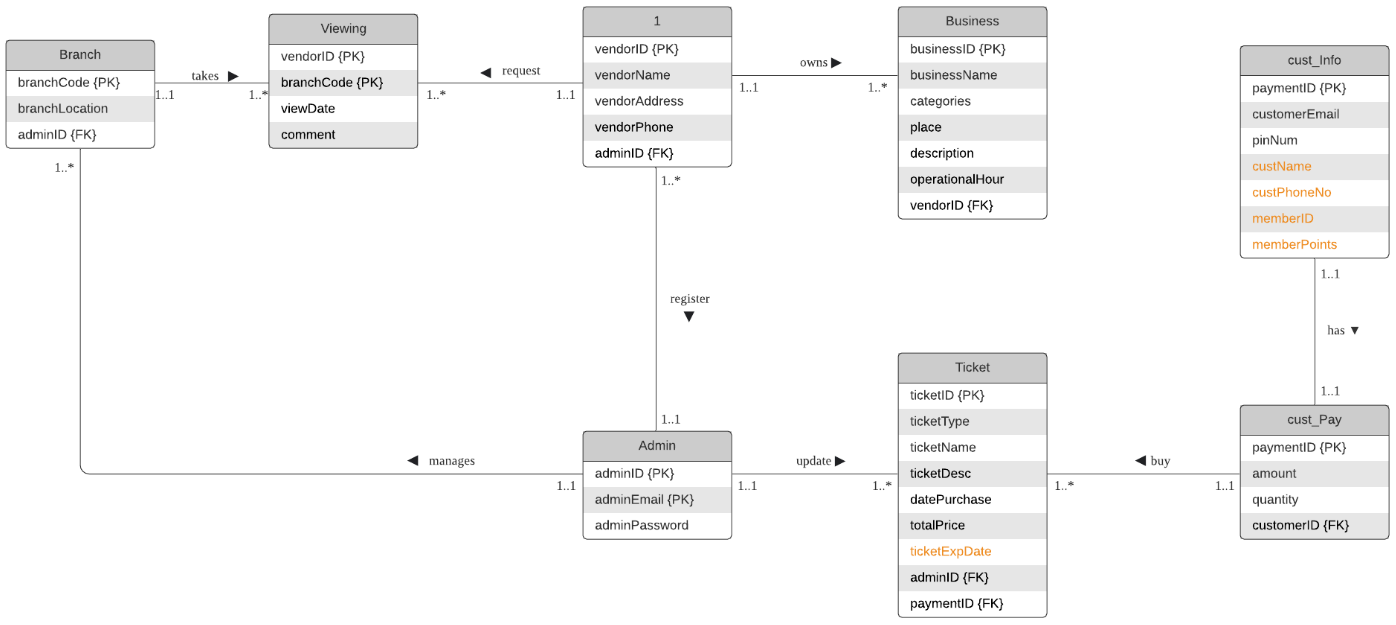
cust_Info(customerID{PK}, customerEmail, pinNum, custName, custPhoneNo, memberID, memberPoints)

BCNF: already in BCNF form

cust_Pay(paymentID{PK}, amount, quantity, customerID{FK})

cust_Info(customerID{PK}, customerEmail, pinNum, custName, custPhoneNo, memberID, memberPoints)

Final Normalized Logical ERD



Data Dictionary for the Normalized Logical Design

Entity name	Attributes	Description	Data Type & Length	Nulls
Branch	branchCode {PK}	Uniquely identifies type of branch	5 variable characters (3 letters 2 numbers)	No
	branchLocation	Location address of the branch	Characters (255)	No
	adminID {FK}	Unique id for admin references Admin(adminID)	5 variable characters (3 letters 2 numbers)	No
Viewing	vendorID {PK}	Unique id for vendor references Vendor(vendorID)	7 variable characters (2 letters 5 numbers)	No
	branchCode {PK}	Unique id for branch references Branch(branchCode)	5 variable characters (3 letters 2 numbers)	No
	viewDate	Date when retrieve infromation	Date	Yes
	comment	Insert opinion	Characters (255)	Yes
Vendor	vendorID {PK}	Unique id registration for vendor	7 variable characters (2 letters 5 numbers)	No
	vendorName	Name of vendor	15 variable characters	No
	vendorAddress	Office address of the vendor	Characters (255)	Yes
	vendorPhone	Phone number	10 integers	No
	adminID {FK}	Unique id for admin references Admin(adminID)	5 variable characters (3 letters 2 numbers)	No
Business	businessID {PK}	Unique id for each business	8 variable characters (2 letters 6 numbers)	No

	businessName	Business name	30 variable characters	No
	categories	Business categories	15 variable characters	No
	place	Address business location	Characters (255)	No
	description	Details of business	Characters (255)	Yes
	operationalHour	Start/End time of the business operation	Date	Yes
	vendorID {FK}	Unique id for vendor references Vendor(vendorID)	7 variable characters (2 letters 5 numbers)	No
Admin	adminID {PK}	Unique id for admin	5 variable characters (2 letters 3 numbers)	No
	adminEmail {PK}	Unique email address for each admin	Variable charcters (25)	No
	adminPassword	Password for admin to verify themselves to usie the system	Variable characters (20)	No
Ticket	ticketID {PK}	Unique id for ticket references number	10 variable characters (2 letters 8 numbers)	No
	ticketType	Adult, kids or senior citizen	1 character (A/K/S)	Yes
	ticketName	Name of the ticket	15 variable characters	No
	ticketDesc	Ticket description/details	Characters (255)	Yes
	datePurchase	Date of purchase	Date	Yes
	totalPrice	Total ticket ptice	Number	No
	ticketExpDate	Expiry date of the ticket	Date	No
	adminID {FK}	Unique id for admin	5 variable characters (3 letters 2 numbers)	No

	paymentID {FK}	Unique id for cust_Pay references cust_Pay(paymentID)	10 variable characters (2 letters 8 numbers)	No
cust_Pay	paymentID {PK}	Unique id for payment made by customers	10 variable characters (2 letters 8 numbers)	No
	amount	Total amount paid by customers	Number	No
	quantity	Total number of tickets bought by customers	Number	Yes
	customerID {FK}	Uniqueid for cust_Info references cust_Info(customerID)	Characters (10)	No
cust_Info	customerID {PK}	Unique id for customers	Number	No
	customerEmail	Unique email address for customer	Variable characters (25)	No
	pinNum	Password for customers to verify their account	Number	Yes
	custName	Customer name	30 variable characters	No
	custPhoneNo	Customer telephone number	10 integers	No
	memberID	Unique id for existing members	6 variable characters (3 letters 3 numbers)	Yes
	memberPoints	Collected points for members	Number	Yes

SUMMARY SQL IMPLEMENTATIONS: 2c

Relational Database Schemas

1. **Vendor**

```
CREATE TABLE vendor (
   vendorID          VARCHAR2 (7) NOT NULL,
   vendorName      VARCHAR2 (15) NOT NULL,
   vendorAddress   CHAR (255) NULL,
   vendorPhone     NUMBER (10) NOT NULL,
   adminID            VARCHAR2 (5) NOT NULL,
   CONSTRAINT vendor_pk PRIMARY KEY (vendorID),
   CONSTRAINT vendor_fk FOREIGN KEY (adminID) REFERENCES admin (adminID)
);
```

```
INSERT INTO vendor (vendorID, vendorName, vendorAddress, vendorPhone, adminID)
VALUES('AB24567', 'Sofia Anne', 'Lintang Sungai Keramat 11A, Kawasan Industri Klang
Utama, 42100, Penang', '6929963841', 'CG123');

INSERT INTO vendor (vendorID, vendorName, vendorAddress, vendorPhone, adminID)
VALUES('AK14567', 'Jane Ahmad', '1 56 Jln 2/3A Pusat Bandar Utama, 68100 George
Town', '1456032778', 'CG123');

INSERT INTO vendor (vendorID, vendorName, vendorAddress, vendorPhone, adminID)
VALUES('CD25678', 'Betto Khusyairi', 'Level 36 Menara citibank 165 Jalan Ampang 50450
Tanjung Bungah', '2130077630', 'DE432');

INSERT INTO vendor (vendorID, vendorName, vendorAddress, vendorPhone, adminID)
VALUES('CD12456', 'Sarah Anne', '4 Amoda Jln Imbi, 55100 Tanjung Tokong',
'2130077630', 'DE432');
```

```
INSERT INTO vendor (vendorID, vendorName, vendorAddress, vendorPhone, adminID)
VALUES('AK23567', 'Ng Mei Hui', 'Taman Serdang Perdana, Seksyen 1, Seri Kembangan.
43300 George Town', '0666740338', 'DE432');

INSERT INTO vendor (vendorID, vendorName, vendorAddress, vendorPhone, adminID)
VALUES('AB34568', 'Luqm Haikal', 'No 9179 Jalan Besar 13300, Penang', '2999195601',
'GF554');

INSERT INTO vendor (vendorID, vendorName, vendorAddress, vendorPhone, adminID)
VALUES('CD34677', 'Adira Suhaimi', 'Ground Floor, North Yu Seng Road, 98000 Balik
Pulau', '1864496526', 'GF554');

INSERT INTO vendor (vendorID, vendorName, vendorAddress, vendorPhone, adminID)
VALUES('AK77777', 'Poa Jing Yi', 'No. K-114 Jalan Sulaimani 24000 Tasek Gelugor,
Penang', '0199765432', 'GF554');

INSERT INTO vendor (vendorID, vendorName, vendorAddress, vendorPhone, adminID)
VALUES('AB55567', 'Sam Smith', '39 Psn Desa Rishah, 30100 Perai, Penang', '3882223406',
'GF445');

INSERT INTO vendor (vendorID, vendorName, vendorAddress, vendorPhone, adminID)
VALUES('AK12390', 'Dean Lewis', 'Perdana Selatan, Seri Kembangan 43300, Air Itam',
'5402396312', 'GF445');
```

2. **Business**

```
CREATE TABLE business(
   businessID               VARCHAR2(8) NOT NULL,
   businessName             VARCHAR2(30) NOT NULL,
   categories               CHAR(15) NOT NULL,
   place                    CHAR(255) NOT NULL,
   description              CHAR(255),
   start_time               DATE,
   end_time                 DATE,
   vendorID                 VARCHAR(28) NOT NULL,
   CONSTRAINT business_pk PRIMARY KEY (businessID),
   CONSTRAINT business_Vendor_fk FOREIGN KEY (vendorID) REFERENCES vendor
(vendorID)
);
```

```
INSERT INTO business (businessID, businessName, categories, place, description,
start_time, end_time, vendorID)
VALUES('AC234567', '3D Museum', 'Museum', 'Georgetown, Penang', 'UNESCO World
Heritage Site', TO_DATE('09:00:00', 'hh24:mi:ss'), TO_DATE('18:00:00', 'hh24:mi:ss'),
'AB24567');

INSERT INTO business (businessID, businessName, categories, place, description,
start_time, end_time, vendorID)
VALUES('CD456765', 'Wax Museum', 'Museum', 'M Mall 020 Penang Time Square',
'Characters of many famous actors', NULL, NULL, 'AK14567');

INSERT INTO business (businessID, businessName, categories, place, description,
start_time, end_time, vendorID)
```

```
VALUES('DR345678', 'Penang City Hall','Museum', 'Penang',  'A reminder of the colonial era
in Penang.', NULL, NULL, 'CD25678');

INSERT INTO business (businessID, businessName, categories, place, description,
start_time, end_time, vendorID)
VALUES('DR654321', 'Penang Islamic Museum', 'Museum', '128 Armenian Street,
Georgetown', 'Housed in a beautiful heritage building', TO_DATE('09:00:00', 'hh24:mi:ss'),
TO_DATE('18:00:00', 'hh24:mi:ss'), 'CD12456');

INSERT INTO business (businessID, businessName, categories, place, description,
start_time, end_time, vendorID)
VALUES('DR345123', 'Boutique Aquarium', 'Aquarium', '12, Building, Georgetown Penang',
'Get Experience with underwater Adventure', TO_DATE('10:00:00', 'hh24:mi:ss'),
TO_DATE('22:00:00', 'hh24:mi:ss'), 'AK23567');

INSERT INTO business (businessID, businessName, categories, place, description,
start_time, end_time, vendorID)
VALUES('CD234567', 'Escape Adventure Play', 'Park', '828 Jalan Teluk Bahang, 11050
penang', 'Outdoor adventure seekers on the island', TO_DATE('09:00:00', 'hh24:mi:ss'),
TO_DATE('18:00:00', 'hh24:mi:ss'), 'AB34568');

INSERT INTO business (businessID, businessName, categories, place, description,
start_time, end_time, vendorID)
VALUES('CD111111', 'Penang Botanic Gardens', 'Park', 'P828 Jalan teluk Bahang, 11050
Penang', 'Outdoor adventure seekers on the island', TO_DATE('15:00:00', 'hh24:mi:ss'),
TO_DATE('20:00:00', 'hh24:mi:ss'), 'CD34677');

INSERT INTO business (businessID, businessName, categories, place, description,
start_time, end_time, vendorID)
```

```
VALUES('AC345555', 'Penang Butterfly Farm', 'Park', 'Teluk Bahang', 'Collection of butterfly
species as well as an assortment of other insects', TO_DATE('09:00:00', 'hh24:mi:ss'),
TO_DATE('17:30:00', 'hh24:mi:ss'), 'AK77777');

INSERT INTO business (businessID, businessName, categories, place, description,
start_time, end_time, vendorID)
VALUES('WE345672', 'Hainanese Mariners' Lodge', 'Heritage', '26A Stewart Lane',
'Dedicated in preserving Penang's unique Chinese culture', NULL, NULL, 'AB55567');

INSERT INTO business (businessID, businessName, categories, place, description,
start_time, end_time, vendorID)
VALUES('AC123456', 'Penang Bridge', 'Heritage', 'Seberang Perai, Penang', 'Sight that greets
most visitors to the island', NULL, NULL, 'AK12390');
```

3. Branch

```
CREATE TABLE branch(
   branchCode        VARCHAR2(5) NOT NULL,
   branchLocation    VARCHAR2(255) NOT NULL,
   adminID           VARCHAR2(5) NOT NULL,
   CONSTRAINT branch_pk PRIMARY KEY (branchCode),
   CONSTRAINT branch_fk FOREIGN KEY (adminID) REFERENCES admin (adminID)
);
```

```
INSERT INTO branch (branchCode, branchLocation, adminID)
VALUES('EQ12M', 'Gurney Plaza', 'CG123');

INSERT INTO branch (branchCode, branchLocation, adminID)
VALUES('EQ22M', 'Penang Times Square', 'CG123');
```

```
INSERT INTO branch (branchCode, branchLocation, adminID)
VALUES('EQ21M', 'Gurney Paragon Mall', 'KE222');

INSERT INTO branch (branchCode, branchLocation, adminID)
VALUES('MQ12M', 'Penang Plaza', 'DE432');

INSERT INTO branch (branchCode, branchLocation, adminID)
values('MQ45M', 'Campbell st Mall', 'DE432');

INSERT INTO branch (branchCode, branchLocation, adminID)
VALUES('WE32M', 'Street Art Georgetown', 'KE123');

INSERT INTO branch (branchCode, branchLocation, adminID)
VALUES('WE45M', 'Penang Road', 'GF554');

INSERT INTO branch (branchCode, branchLocation, adminID)
VALUES('WE12M', 'Komtar Bus Terminal', 'GF554');

INSERT INTO branch (branchCode, branchLocation, adminID)
VALUES('QW12M', 'Penang Sentral Bus', 'KE554');

INSERT INTO branch (branchCode, branchLocation, adminID)
VALUES('QW56M', 'Penang International Airport', 'GF445');
```

4. Admin

```
CREATE TABLE admin(
   adminID           VARCHAR2 (5) NOT NULL,
   adminEmail        VARCHAR2 (25) NOT NULL,
   adminPassword  VARCHAR2 (20) NOT NULL,
   CONSTRAINT admin_pk PRIMARY KEY (adminID)
);
```

```
INSERT INTO admin (adminID, adminEmail, adminPassword)
VALUES ('CG123', 'admin1@yahoo.com', 'Ke4jar');

INSERT INTO admin (adminID, adminEmail, adminPassword)
VALUES ('KE222', 'admin2@yahoo.com', 'UVxy65');

INSERT INTO admin (adminID, adminEmail, adminPassword)
VALUES ('DE432', 'admin3@yahoo.com', '1jaTqYG');

INSERT INTO admin (adminID, adminEmail, adminPassword)
VALUES ('KE123', 'admin4@yahoo.com', '56FuqT');

INSERT INTO admin (adminID, adminEmail, adminPassword)
VALUES ('GF554', 'admin5@yahoo.com', 'TRq76Vch');

INSERT INTO admin (adminID, adminEmail, adminPassword)
VALUES ('KE554', 'admin6@yahoo.com', 'TRsx78Cv');

INSERT INTO admin (adminID, adminEmail, adminPassword)
VALUES ('GF445', 'admin7@yahoo.com', 'MN98xty');
```

```
INSERT INTO admin (adminID, adminEmail, adminPassword)
VALUES ('KE234', 'admin8@yahoo.com', 'EG78xctY');

INSERT INTO admin (adminID, adminEmail, adminPassword)
VALUES ('KE345', 'admin9@yahoo.com', 'K$Ncl1');

INSERT INTO admin (adminID, adminEmail, adminPassword)
VALUES ('DE433', 'admin10@yahoo.com', 'Vfe56tf');
```

5. Ticket

```
CREATE TABLE ticket(
   ticketID              VARCHAR2 (10) NOT NULL,
   ticketType            CHAR (1),
   ticketName            VARCHAR2 (25) NOT NULL,
   ticketDesc            CHAR (255),
   datePurchase          VARCHAR2 (9),
   totalPrice            NUMBER (5) NOT NULL,
   ticketExpDate         DATE,
   adminID               VARCHAR2 (5) NOT NULL,
   paymentID             VARCHAR2 (10) NOT NULL,
   CONSTRAINT ticketID_pk PRIMARY KEY (ticketID),
   CONSTRAINT ticket_Admin_fk FOREIGN KEY (adminID) REFERENCES
admin(adminID),
   CONSTRAINT Ticket_cust_Pay_fk FOREIGN KEY (paymentID) REFERENCES
cust_Pay(paymentID)
);
```

```
INSERT INTO ticket (ticketID, ticketType, ticketName, ticketDesc, datePurchase, totalPrice,
ticketExpDate, adminID, paymentID)
VALUES('MO98765432', 'A', '3D Museum',  NULL, TO_DATE('12-01-2021', 'DD-MM-
YYYY'), '20.00', TO_DATE('12-01-2022', 'DD-MM-YYYY'), 'CG123', 'EK13590123');

INSERT INTO ticket (ticketID, ticketType, ticketName, ticketDesc, datePurchase, totalPrice,
ticketExpDate, adminID, paymentID)
VALUES('MO34456783', 'S', '3D Museum', NULL, TO_DATE('12-01-2021', 'DD-MM-
YYYY'), '15.00', TO_DATE('12-01-2022', 'DD-MM-YYYY'), 'CG123', 'EK13590123');

INSERT INTO ticket (ticketID, ticketType, ticketName, ticketDesc, datePurchase, totalPrice,
ticketExpDate, adminID, paymentID)
VALUES('MO45676654', 'K', '3D Museum', NULL, TO_DATE('12-01-2021', 'DD-MM-
YYYY'), '10.00', TO_DATE('12-01-2022', 'DD-MM-YYYY'), 'CG123', 'EK13590123');

INSERT INTO ticket (ticketID, ticketType, ticketName, ticketDesc, datePurchase, totalPrice,
ticketExpDate, adminID, paymentID)
VALUES('MD44568975', 'A', 'Boutique Aquarium', NULL, TO_DATE('12-01-2021', 'DD-
MM-YYYY'), '25.00', TO_DATE('12-03-2022', 'DD-MM-YYYY'), 'DE432', 'EK17630204');

INSERT INTO ticket (ticketID, ticketType, ticketName, ticketDesc, datePurchase, totalPrice,
ticketExpDate, adminID, paymentID)
VALUES('MD33245644', 'A', 'Wax Museum', NULL, TO_DATE('23-01-2021', 'DD-MM-
YYYY'), '25.00', TO_DATE('23-12-2021', 'DD-MM-YYYY'), 'DE432', 'EK56370215');

INSERT INTO ticket (ticketID, ticketType, ticketName, ticketDesc, datePurchase, totalPrice,
ticketExpDate, adminID, paymentID)
VALUES ('MD77546777', 'A', 'Escape AdventurePlay', NULL, TO_DATE('17-01-2021', 'DD-
MM-YYYY'), '30.00', TO_DATE('17-01-2022', 'DD-MM-YYYY'), 'CG123', 'EK34780308');
```

```
INSERT INTO icket (ticketID, ticketType, ticketName, ticketDesc, datePurchase, totalPrice,
ticketExpDate, adminID,t paymentID)
VALUES ('MO98665455','A','Escape AdventurePlay',NULL,TO_DATE('17-01-2021', 'DD-
MM-YYYY'), '30.00', TO_DATE('17-01-2022', 'DD-MM-YYYY'), 'CG123', 'EK34780308');

INSERT INTO ticket (ticketID, ticketType, ticketName, ticketDesc, datePurchase, totalPrice,
ticketExpDate, adminID, paymentID)
VALUES ('MD22415633', 'K', 'Escape Adventure Play', NULL, TO_DATE('17-01-2021',
'DD-MM-YYYY'), '15.00', TO_DATE('17-01-2022', 'DD-MM-YYYY'), 'CG123',
'EK34780308');

INSERT INTO ticket (ticketID, ticketType, ticketName, ticketDesc, datePurchase, totalPrice,
ticketExpDate, adminID, paymentID)
VALUES('MD34255123', 'K', 'Boutique Aquarium', NULL, TO_DATE('14-01-2021', 'DD-
MM-YYYY'), '20.00', TO_DATE('14-03-2022', 'DD-MM-YYYY'), 'KE123', 'EK17630204');

INSERT INTO ticket (ticketID, ticketType, ticketName, ticketDesc,  datePurchase, totalPrice,
ticketExpDate, adminID, paymentID)
VALUES('MO97557785', 'S', 'Penang Botanic Gardens', NULL, TO_DATE('30-01-2021',
'DD-MM-YYYY'), '12.00', TO_DATE('31-12-2021', 'DD-MM-YYYY'), 'KE554',
'EK41230236');
```

6. cust_Pay

```
CREATE TABLE cust_Pay(
  paymentID           VARCHAR2(10) NOT NULL,
  quantity             NUMBER(4) NOT NULL,
  amount               NUMBER(7,2) NOT NULL,
  customerID           NUMBER(5) NOT NULL,
  CONSTRAINT cust_Pay_pk PRIMARY KEY (paymentID),
   CONSTRAINT cust_Pay_fk FOREIGN KEY (customerID) REFERENCES cust_Info
(customerID)
);
```

```
INSERT INTO cust_Pay (paymentID, quantity, amount, customerID)
VALUES ('EK13590123', 3, 45.00, 20541);

INSERT INTO cust_Pay (paymentID, quantity, amount, customerID)
VALUES ('EK37530145', 4, 54.00, 24178);

INSERT INTO cust_Pay (paymentID, quantity, amount, customerID)
VALUES ('EK65490086', 3, 37.50, 29871);

INSERT INTO cust_Pay (paymentID, quantity, amount, customerID)
VALUES ('EK17630204', 2, 45.00, 45723);

INSERT INTO cust_Pay (paymentID, quantity, amount, customerID)
VALUES ('EK56370215', 1, 15.00, 32568);

INSERT INTO cust_Pay (paymentID, quantity, amount, customerID)
VALUES ('EK34780308', 3, 75.00, 14782);

INSERT INTO cust_Pay (paymentID, quantity, amount, customerID)
VALUES ('EK26320168', 3, 50.00, 19874);

INSERT INTO cust_Pay (paymentID, quantity, amount, customerID)
VALUES ('EK15110136', 4, 90.00, 38269);
```

```
INSERT INTO cust_Pay (paymentID, quantity, amount, customerID)
VALUES ('EK19980232', 2, 30.00, 53214);

INSERT INTO cust_Pay (paymentID, quantity, amount, customerID)
VALUES ('EK41230236', 1, 12.00, 41102);
```

7. cust_Info

```
CREATE TABLE cust_Info (
    customerID        NUMBER(5) NOT NULL,
    customerEmail     VARCHAR2(25) NOT NULL,
    pinNum            NUMBER(6),
    custName          VARCHAR2(30) NOT NULL,
    custPhoneNo       NUMBER(10) NOT NULL,
    memberID          VARCHAR2(6),
    memberPoints      NUMBER(5),
    CONSTRAINT cust_Info_pk PRIMARY KEY (customerID)
);
```

```
INSERT INTO cust_Info (customerID, customerEmail, pinNum, custName, custPhoneNo,
memberID, memberPoints)
VALUES(20541, 'aliayaa32@gmail.com', 123456, 'Alia Amira', '1456987532', NULL,
NULL);

INSERT INTO cust_Info (customerID, customerEmail, pinNum, custName, custPhoneNo,
memberID, memberPoints)
VALUES(24178, 'ahmad_azmi@gmail.com’, 789101, 'Ahmad Azmi', '6523145789',
'MEM102', 00187);
```

```
INSERT INTO cust_Info (customerID, customerEmail, pinNum, custName, custPhoneNo,
memberID, memberPoints)
VALUES(29871, 'nursofea@yahoo.com', 086429, 'Nur Sofea', '4213574165', 'MEM154',
00095);

INSERT INTO cust_Info (customerID, customerEmail, pinNum, custName, custPhoneNo,
memberID, memberPoints)
VALUES(45723, 'siti_aishah@gmail.com', 246810, 'Siti Aishah', '4321785426', 'MEM203',
00326);

INSERT INTO cust_Info (customerID, customerEmail, pinNum, custName, custPhoneNo,
memberID, memberPoints)
VALUES(32568, 'hamzahhh567@yahoo.com', 050100, 'Abdul Hamzah', '3547852136',
NULL, NULL);

INSERT INTO cust_Info (customerID, customerEmail, pinNum, custName, custPhoneNo,
memberID, memberPoints)
VALUES(14782, 'ammarhasif@gmail.com', 050200, 'Muhd Ammar Hasif', '4752341687',
NULL, NULL);

INSERT INTO cust_Info (customerID, customerEmail, pinNum, custName, custPhoneNo,
memberID, memberPoints)
VALUES(19874, 'muhd_amier@gmail.com',240200, 'Muhd Amier', '1739605207', 'MEM141',
00165);

INSERT INTO cust_Info (customerID, customerEmail, pinNum, custName, custPhoneNo,
memberID, memberPoints)
VALUES(38269, 'shukriyahya@gmail.com', 190397, 'Shukri Yahya', '1266787254',
'MEM174', 00214);
```

```
INSERT INTO cust_Info (customerID, customerEmail, pinNum, custName, custPhoneNo,
memberID, memberPoints)
VALUES(53214, 'nurulainnur18@gmail.com', 090600, 'Nurul Ainnur', '1729076248',
'MEM092', 00113);

INSERT INTO cust_Info (customerID, customerEmail, pinNum, custName, custPhoneNo,
memberID, memberPoints)
VALUES(41102, 'anasofiya@gmail.com', 130600, 'Ana Sofiya', '1762880966', 'MEM133',
00205);
```

8. Viewing

```
CREATE TABLE viewing(
   vendorID          VARCHAR2(7) NOT NULL,
   branchCode        VARCHAR2(5) NOT NULL,
   viewDate          DATE,
   comments          CHAR(255),
   CONSTRAINT viewing_pk PRIMARY KEY (vendorID, branchCode)
);
```

```
INSERT INTO viewing (branchCode, vendorID, viewDate, comments)
VALUES('EQ12M', 'AB24567',TO_DATE('15-01-2021', 'DD-MM-YYYY'), 'This is great' );

INSERT INTO viewing (branchCode, vendorID, viewDate, comments)
VALUES('EQ22M', 'AK14567', TO_DATE('15-01-2021', 'DD-MM-YYYY'), 'This is great');
```

```
INSERT INTO viewing (branchCode, vendorID, viewDate, comments)
VALUES('EQ21M', 'AC22341',TO_DATE('16-01-2021', 'DD-MM-YYYY'), 'The date doesnt
really coincide with my time');

INSERT INTO viewing (branchCode, vendorID, viewDate, comments)
VALUES('MQ12M', 'CD25678',TO_DATE('17-01-2021', 'DD-MM-YYYY'), 'There a lot of
good thing');

INSERT INTO viewing (branchCode, vendorID, viewDate, comments)
VALUES('MQ45M', 'CD12456', TO_DATE('17-01-2021', 'DD-MM-YYYY'), 'There a lot of
good thing');

INSERT INTO viewing (branchCode, vendorID, viewDate, comments)
VALUES('MQ46M', 'AK23567', TO_DATE('17-01-2021', 'DD-MM-YYYY'), 'There a lot of
good thing');

INSERT INTO viewing (branchCode, vendorID, viewDate, comments)
VALUES('WE32M','BC22345', TO_DATE('18-01-2021', 'DD-MM-YYYY'), 'The branch is
good');

INSERT INTO viewing (branchCode, vendorID, viewDate, comments)
VALUES('WE45M','AB34568', TO_DATE('19-01-2021', 'DD-MM-YYYY'), 'This branch is
entertaining');

INSERT INTO viewing (branchCode, vendorID, viewDate, comments)
VALUES('WE12M', 'CD34677', TO_DATE('19-01-2021', 'DD-MM-YYYY'), 'This branch is
entertaining');
```

```
INSERT INTO viewing (branchCode, vendorID, viewDate, comments)
VALUES('WE56M', 'AK77777', TO_DATE('19-01-2021', 'DD-MM-YYYY'), 'This branch is
entertaining');

INSERT INTO viewing (branchCode, vendorID, viewDate, comments)
VALUES('QW12M', 'AB55661', TO_DATE('20-01-2021', 'DD-MM-YYYY'), 'The date
match on my time');

INSERT INTO viewing (branchCode, vendorID, viewDate, comments)
VALUES('QW56M', 'AB55567', TO_DATE('21-01-2021', 'DD-MM-YYYY'), 'This is
awesome');

INSERT INTO viewing (branchCode, vendorID, viewDate, comments)
VALUES('QW77M', 'AK12390', TO_DATE('21-01-2021', 'DD-MM-YYYY'), 'This is
awesome');
```

Set of Queries

Customer

1. Join cust_Pay and cust_Info table of customer basic information and their payment

```
SELECT p.paymentID, p.quantity, p.amount, i.custName, i.custPhoneNo

FROM cust_Pay p JOIN cust_Info i

USING (customerID)
```

2. Fetch result where quantity of ticket >= 3

```
SELECT paymentID, quantity

FROM cust_Pay

WHERE quantity >= 3
```

3. Fetch result of customer info in ascending order of customer email

```
SELECT customerID, customerEmail, pinNum

FROM cust_Info

ORDER BY customerEmail
```

4. Fetch the result where add column which is discount

```
ALTER TABLE cust_Pay
ADD (discount NUMBER (7,2) DEFAULT (0.15) NOT NULL)
```

5. Fetch the result where collected points by members >= 200

```
SELECT custName, memberID, memberPoints

FROM cust_Info

WHERE memberPoints >= 200
```

6. Fetch the result in descending order of collected points by customers who are members

```
SELECT custName, custPhoneNo, memberID, memberPoints
FROM cust_Info
ORDER BY memberPoints DESC
```

7. Fetch result of expiry date of customers' purchased ticket

```
SELECT ticketID, ticketName, datePurchase, ticketExpDate, paymentID

FROM ticket
```

Admin

1. Join Admin and Branch table

```
SELECT b.branchCode, b.branchLocation, ad.adminEmail, ad.adminPassword

FROM branch b JOIN admin ad

USING (adminID)
```

2. Fetch the result where add column admin salary with not null

```
ALTER TABLE admin

ADD adminSalary NUMBER(7,2) DEFAULT (10000) NOT NULL
```

3. Fetch distinct value from the adminID column in Admin table

```
SELECT DISTINCT adminID FROM admin
```

4. Fetch the result where admin password contains character – T

```
SELECT * FROM admin
WHERE adminPassword LIKE '%T%'
```

5. Fetch result where add column Address for customer

```
ALTER TABLE cust_Info
ADD (custAddress CHAR(255) NULL)
```

6. Fetch result in ascending order of ticket expiry date

```
SELECT ticketID, ticketName, datePurchase, ticketExpDate

FROM ticket

ORDER BY ticketExpDate
```

7. Fetch the result where admin change the expiry date of ticket

```
UPDATE ticket

SET ticketExpDate = TO_DATE('31-12-2021', 'DD-MM-YYYY')

WHERE datePurchase = TO_DATE('12-01-2021', 'DD-MM-YYYY')
```

Vendor

1. Join Vendor and Business table (different names columns)

```
SELECT vendor.vendorName name, business.businessName business
FROM vendor JOIN business
ON (vendor.vendorID=business.vendorID)
```

2. Fetch result for business with categories of park

```
SELECT businessName, categories

FROM business

WHERE categories = ‘Park’
```

3. Fetch result for vendor under Admin = GF554

```
SELECT vendorID, vendorName, vendorAddress, vendorPhone, adminID

FROM vendor

WHERE adminID = ‘GF554’
```

4. Change one of the vendor phone number

```
UPDATE vendor

SET vendorPhone = ‘0166740338’

WHERE vendorID = ‘AK23567’
```

5. Using quote(q) operator for Ticket table

```
SELECT ticketID || q'[ is refer to: ]' || ticketName AS "Ticket references"
FROM ticket
```

6. Using column aliases to find comments with specific word ‘criticism’ in Viewing table

```
SELECT comments AS "Criticism"
FROM viewing
```

7. Using quote(q) operator for viewing date in Ticket table

```
SELECT datePurchase || q'[ will be valid until: ]' || ticketExpDate AS "Ticket Validation"
FROM ticket
```

Conclusion

By performing an enhancement on a completed database system, we can find some part that we can add for a better system. For me, I have added five new attributes to two entities. This is because, I think by storing the added attributes, the system can be more meaningful for the customers to use. The limitation to do the enhancement is I think of the conseuqunces while doing the SQL statements. It may affect all the other entities, so a proper and deeply thinking to make a new process to it. My contribution to the group project and this alternative assesment is I make myself to understand and study the flow of this database system well. For example, I understand the concept of this kiosk system, what is the requirements, the flow in the system also, functions of many data that will be stored in the system. In conclusion, this project is a very good and useful task for the students.

REFLECTION

It was meaningful to complete the phases of project with the members. Individually, to complete this alternative assesment of the enhancement from the group project. This group project was successfully be completed within time with great cooperation between the members. The limitation in this group project is maybe by different group members’ time management. Due to current pandemic, it is hard to keep up with university courses and things in home. Anyways, each of us managed to do the assigned part very well and able to complete each phase on time before deadlines.

I was feeling motivated to learn more about some topics when I have understood basic concept at the first part. During first few classes, I was a bit confused with ERD and data flow diagram (DFD) from System and Analysis Design class. I have mixed up drawing diagram but by times, I am able to differentiate both diagrams.

During completing the group project, I am able to know about database system. There are phases in creating a well functioned and complete database system. From having an agreement on a system that we are going to propose and complete the system to start discussing on how the flow in the system will work. I can learn how to draw and develop entity relationship diagram (ERD), identify relationship between entities, learn about constraints in database management system and many more. Basically, I have learned about database management system that includes all about database that a student or a person that will works in this industry should know. With my group members, I had a quality time discussing concepts in database and drawing ERD together. It made us understand more about the topic.

Doing group project in phases really help in this project as the work will be done in time and sequentially. Hence, all members can understand and know where the progress is. My biggest lesson from doing the project is to get explanation and assurance on concepts that you are confused on. This is because when you are stuck on certain topic or concept and take a long time to understand, it may affect the progress of group project. I must assure that I understand on topics well so that I can keep up with my group’s pace on completing the project.