

รายงาน

การออกแบบฐานข้อมูลสำหรับการซื้อขายสินค้าและลงบัญชีของบริษัท

นำเสนอ  
รศ.ดร.วีรชัย อโณทัยไพบูลย์

โดย

นาย รเณศ ชูเผือก รหัสนักศึกษา 6210612864

รายงานฉบับนี้เป็นส่วนหนึ่งของรายวิชา วพ.230 ระบบฐานข้อมูล

คณะวิศวกรรมศาสตร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยธรรมศาสตร์

ปีการศึกษา 2563

**คำนำ**

รายงานการออกแบบฐานข้อมูลสำหรับการซื้อขายสินค้าและลงบัญชีของบริษัทนี้จัดทำขึ้นเพื่อใช้แสดงความเข้าใจต่อเนื้อหาที่ได้ทำการศึกษามาในรายวิชา วพ.230 การออกแบบฐานข้อมูล โดยในการทำรายงานครั้งนี้นั้นได้ใช้ความรู้ในเรื่องของการออกแบบฐานข้อมูล โดยทำการเขียน E/R diagram และทำการอธิบายในส่วนของ entity และ relationship ที่ปรากฎ จากนั้นทำการเขียน UML diagram ที่มีคุณสมบัติเหมือนกับ E/R diagram และยังมีการเปลี่ยน E/R diagram ให้อยู่ในรูปของ schema อีกด้วย นอกจากใช้ความรู้ในเรื่องของการออกแบบฐานข้อมูลแล้วยังทำการเขียน SQL ของตัวฐานข้อมูลที่ได้ทำการออกแบบ และนำเอาตัวอย่างของการเรียกใช้ข้อมูลต่าง ๆ มานำเสนอในรายงานฉบับนี้อีกด้วย

รเณศ ชูเผือก

**สารบัญ**

|  |  |
| --- | --- |
| เรื่อง | หน้า |
| E/R diagram |  |
| UML diagram |  |
| Schema |  |
| SQL สำหรับสร้าง Table |  |
| ตัวอย่างข้อมูล |  |
| ตัวอย่าง Query |  |

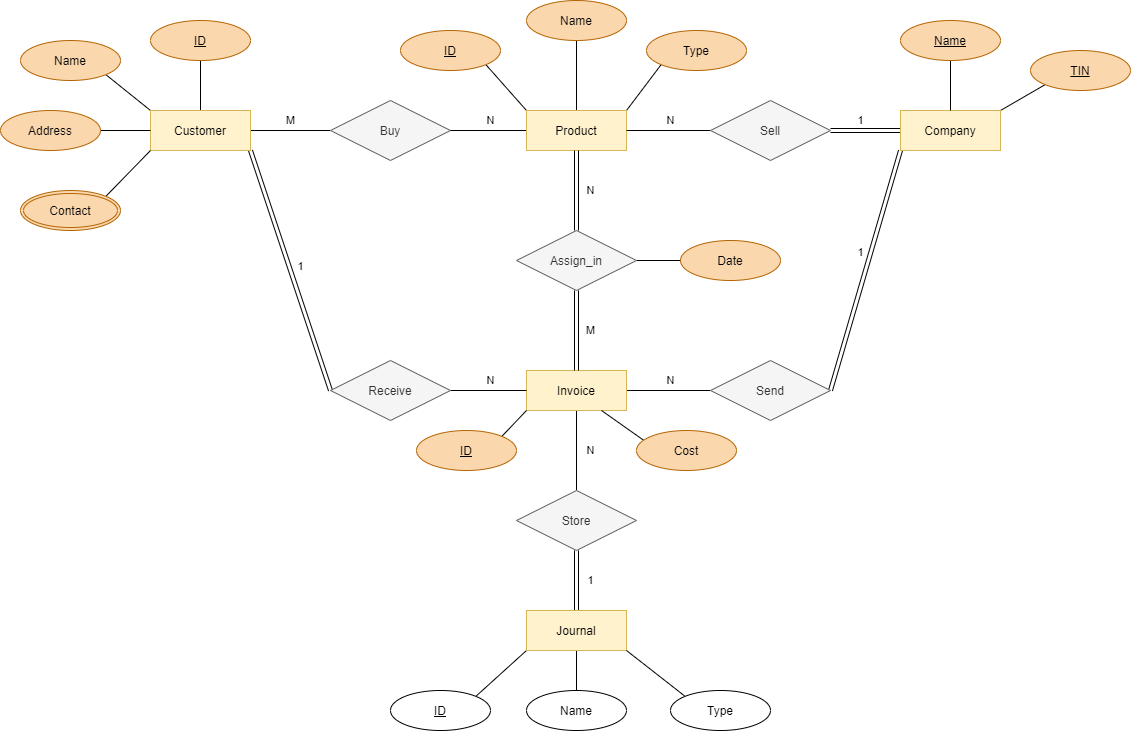
**E/R diagram**

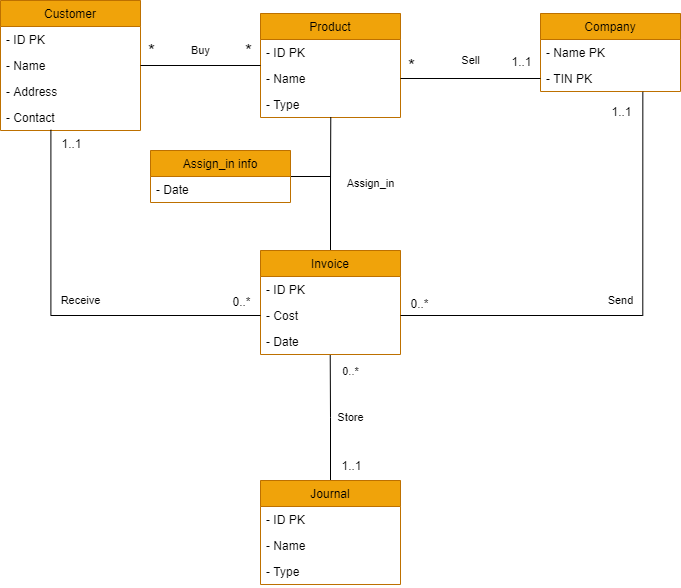
ในการออกแบบ E/R diagram ของฐานข้อมูลสำหรับการซื้อขายสินค้าและลงบัญชีของบริษัท ได้ทำการกำหนด entity อยู่ด้วยกัน 5 entity ได้แก่

* Entity “Customer” เก็บข้อมูลเกี่ยวกับลูกค้า ได้แก่ รหัสลูกค้า ชื่อ ที่อยู่ และข้อมูลติดต่อ
* Entity “Product” เก็บข้อมูลเกี่ยวกับสินค้า ได้แก่ รหัสสินค้า ชื่อสินค้า และ ประเภทสินค้า
* Entity “Company” เก็บข้อมูลเกี่ยวกับบริษัท ได้แก่ ชื่อบริษัท และ เลขที่ประจำตัวของผู้เสียภาษี
* Entity “Invoice” เก็บข้อมูลเกี่ยวกับใบเสร็จ ได้แก่ เลขที่ใบเสร็จ มูลค่า และวันที่จำหน่าย
* Entity “Journal” เก็บข้อมูลเกี่ยวกับบัญชี ได้แก่ เลขที่บัญชี ชื่อบัญชี และประเภทบัญชี

และยังมี relationships อยู่ด้วยกัน 6 คู่ ได่แก่

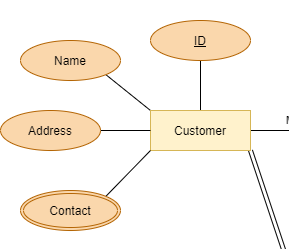
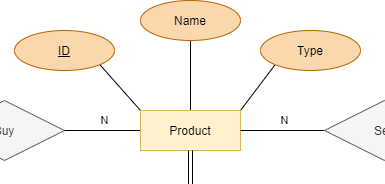
* Relationship “Buy” เป็นความสัมพันธ์แบบ many-to-many โดยที่ลูกค้าสามารถซื้อสินค้ากี่ตัวก็ได้ ส่วนสินค้าจะมีลูกค้าซื้อกี่ตัวก็ได้หรืออาจจะไม่มีลูกค้าซื้อเลยก็ได้
* Relationship “Sell” เป็นความสัมพันธ์แบบ many-to-one โดยที่บริษัทสามารถมีสินค้ากี่ตัวก็ได้หรืออาจจะไม่มีเลยก็ได้ แต่สินค้าใดสินค้าหนึ่งต้องมีผู้ผลิตเป็นบริษัทแห่งใดแห่งหนึ่งเท่านั้น
* Relationship “Assign\_in” เป็นความสัมพันธ์แบบ many-to-many โดยที่สินค้าหนึ่งตัวสามารถเขียนลงไปในใบเสร็จกี่ใบก็ได้ และสินค้าทุกตัวจำเป็นต้องเขียนลงในใบเสร็จ ส่วนของใบเสร็จนั้นก็จำเป็นต้องมีสินค้ามาเขียนลงในใบเสร็จ แต่ในใบเสร็จนั้นจะมีสินค้ากี่ตัวก็ได้
* Relationship “Receive” เป็นความสัมพันธ์แบบ many-to-one โดยที่ลูกค้าสามารถได้รับใบเสร็จกี่ใบก็ได้ แต่ใบเสร็จทุกใบจำเป็นต้องมีลูกค้าเพียงหนึ่งเท่านั้น
* Relationship “Send” เป็นความสัมพันธ์แบบ many-to-one โดยที่บริษัทสามารถออกใบเสร็จกี่ใบก็ได้ แต่ใบเสร็จทุกใบจำเป็นต้องมีผู้ออกเป็นบริษัทเพียงหนึ่งเดียว โดยบางบริษัทอาจจะไม่มีการออกใบเสร็จก็ได้
* Relationship “Store” เป็นความสัมพันธ์แบบ many-to-one โดยใบเสร็จทุกใบจำเป็นต้องลงในบัญชีใดบัญชีหนึ่งเท่านั้น แต่บัญชีใดบัญชีหนึ่งสามารถมีใบเสร็จได้หลายใบ





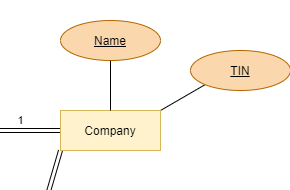
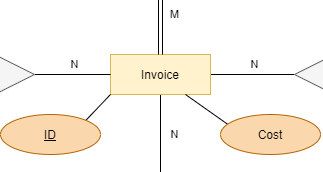
**Schema**

**Step 1: Mapping of Regular Entity Types.**

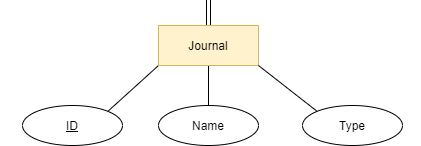
Customer Entity

Product Entity

Company Entity

Invoice Entity



Journal Entity

Customer

|  |  |  |
| --- | --- | --- |
| cID | cName | Address |

Product

|  |  |  |
| --- | --- | --- |
| pID | pName | pType |

Company

|  |  |
| --- | --- |
| companyName | TIN |

Invoice

|  |  |  |
| --- | --- | --- |
| invoiceID | Cost | Date |

Journal

|  |  |  |
| --- | --- | --- |
| jID | jName | jType |

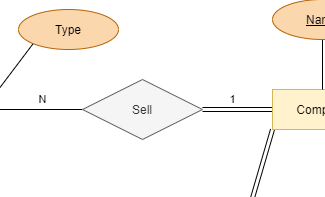
**Step 2 : Mapping of Weak Entity Types.**

None

**Step 3 : Mapping of Binary 1:1 Relation Types.**

None

**Step 4: Mapping of Binary 1:N Relationship Types.**

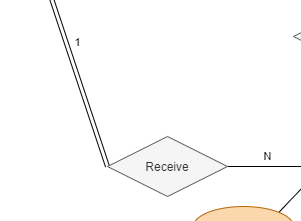


Sell Relationship

Product

|  |  |  |
| --- | --- | --- |
| pID | pName | pType |

|  |  |  |  |
| --- | --- | --- | --- |
| pID | pName | pType | TIN |

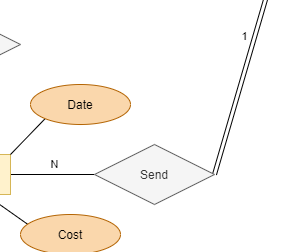


Receive Relationship

Invoice

|  |  |  |
| --- | --- | --- |
| invoiceID | Cost | Date |

|  |  |  |  |
| --- | --- | --- | --- |
| invoiceID | Cost | Date | cID |

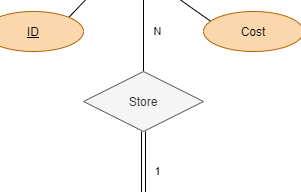


Send Relationship

Invoice

|  |  |  |  |
| --- | --- | --- | --- |
| invoiceID | Cost | Date | cID |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| invoiceID | Cost | Date | cID | TIN |



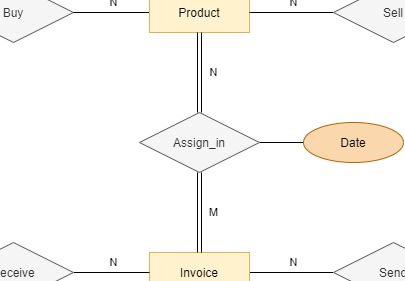
Store Relationship

Invoice

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| invoiceID | Cost | Date | cID | TIN |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| invoiceID | Cost | Date | cID | TIN | jID |

**Step 5: Mapping of Binary M:N Relationship Types.**



Assign\_in Relationship

Assign\_in

|  |  |
| --- | --- |
| pID | invoiceID |

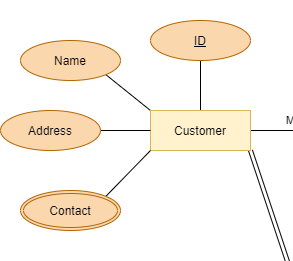


Buy Relationship

Buy

|  |  |
| --- | --- |
| cID | pID |

**Step 6: Mapping of Multivalued attributes.**



Contact Attribute

Contact

|  |  |
| --- | --- |
| cID | cContact |

**Step 7: Mapping of N-ary Relationship Types.**

None

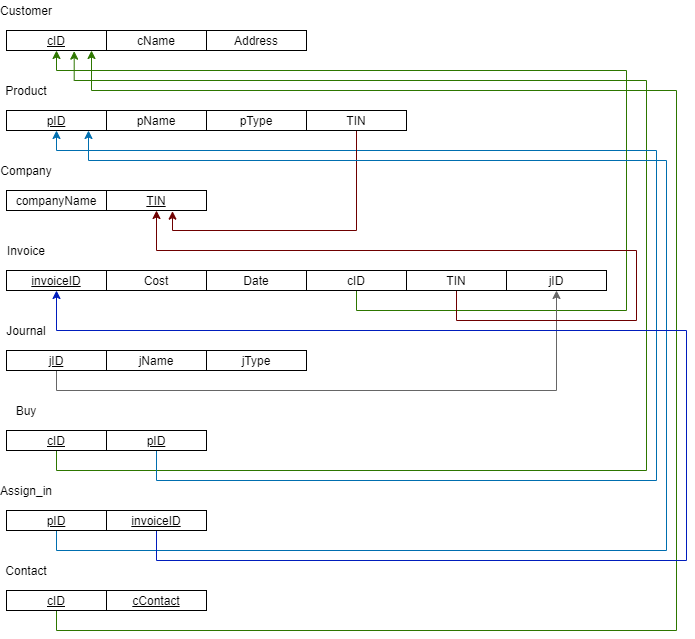
**Step 8: Options for Mapping Specialization or Generalization.**

None

**Step 9: Mapping of Union Types (Categories).**

None

**Result of mapping ER schema into a relational database**



**SQL สำหรับสร้าง Table**

drop table if exists Buy;  
drop table if exists Assign\_in;  
drop table if exists Contact;  
drop table if exists Invoice;  
drop table if exists Customer;  
drop table if exists Product;  
drop table if exists Company;  
drop table if exists Journal;  
  
create table Customer  
(  
 cID int,  
 cName varchar(100) not null,  
 Address varchar(255) not null,  
 constraint Customer\_pk  
 primary key (cID)  
);

create table Product  
(  
 pID int,  
 pName varchar(100) null,  
 pType varchar(50) null,  
 constraint Product\_pk  
 primary key (pID)  
);

create table Company  
(  
 companyName varchar(100) not null,  
 TIN int auto\_increment,  
 constraint Company\_pk  
 primary key (TIN)  
);

create table Invoice  
(  
 InvoiceID int,  
 Cost float not null,  
 Date varchar(50) null,  
 constraint Invoice\_pk  
 primary key (InvoiceID)  
);

create table Journal  
(  
 jID int,  
 jName varchar(100) not null,  
 jType varchar(100) not null,  
 constraint Journal\_pk  
 primary key (jID)  
);

alter table Product  
 add TIN int references Company (TIN);  
alter table Invoice  
 add TIN int references Company (TIN);  
alter table Invoice  
 add cID int references Customer (cID);  
alter table Invoice  
 add jID int references Journal (jID);

create table Buy  
(  
 cID int not null,  
 pID int not null,  
 constraint Buy\_Customer\_cID\_fk  
 foreign key (cID) references Customer (cID),  
 constraint Buy\_Product\_pID\_fk  
 foreign key (pID) references Product (pID)  
);  
  
create table Assign\_in  
(  
 pID int not null,  
 invoiceID int not null,  
 constraint Assign\_in\_Product\_pID\_fk  
 foreign key (pID) references Product (pID),  
 constraint Assign\_in\_Invoice\_invoiceID\_fk  
 foreign key (invoiceID) references Invoice (InvoiceID)  
);  
  
create table Contact  
(  
 cID int not null,  
 cContact varchar(255),  
 constraint Contact\_Customer\_cID\_fk  
 foreign key (cId) references Customer (cID)  
);