Date: 14/12/2017

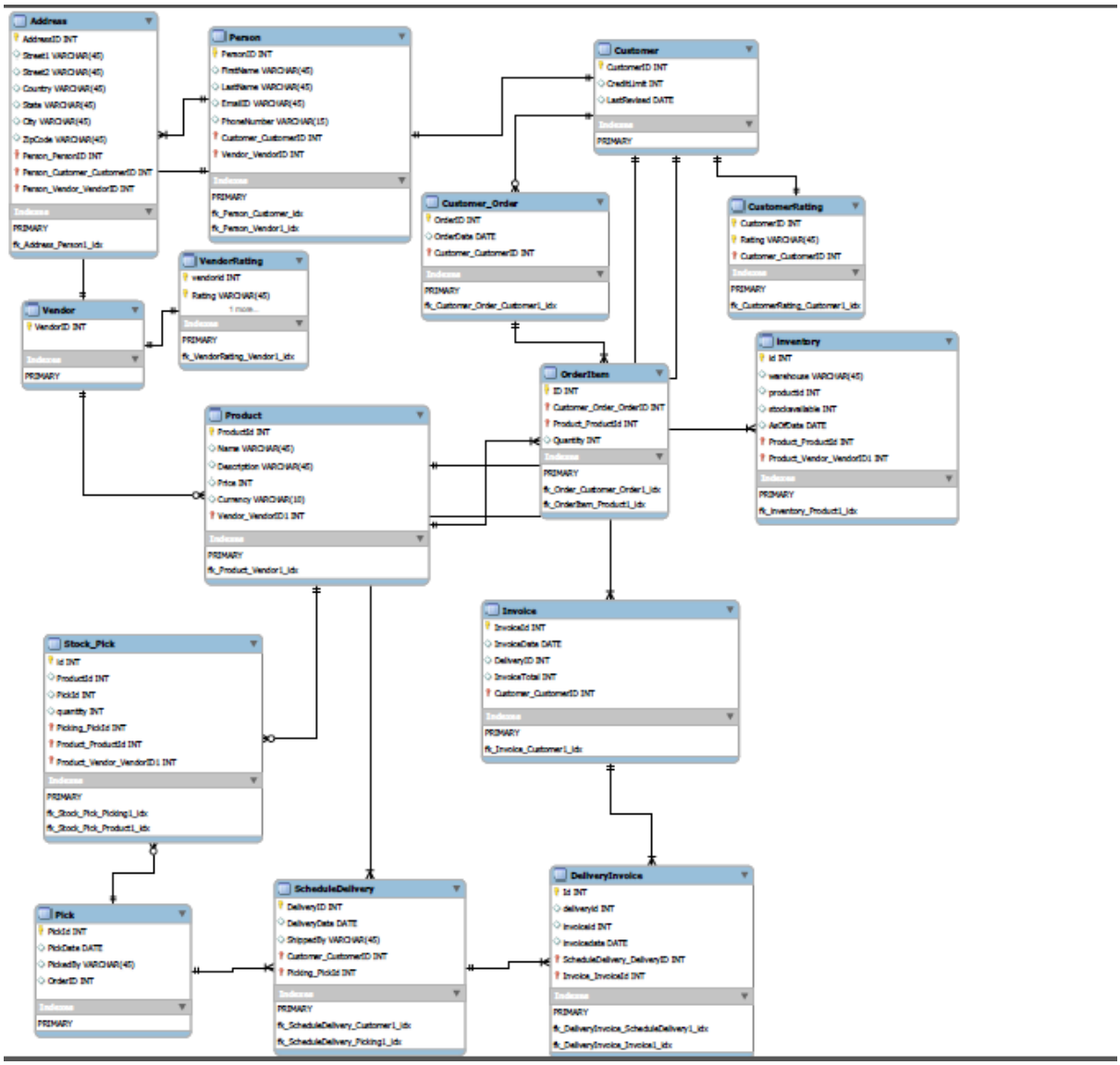
Electronic Store Management System

Problem Statement

Every business across the globe wishes to use technology to increase sales and flourish business worldwide. In this project, I am attempting to create database which manages online transactions of electronic items like laptops, desktops…etc.

The project aim is to track entire order to cash flow which involves various sub processes like receiving orders from customer, placing orders, picking, packing and scheduling delivery and generating invoice for customer.

ER Diagram



ER Diagram.pdf

Date: 14/12/2017

Electronic Store Management System

Data Description

For demonstration purpose, I have used some dummy data to display End-to-End flow which involves customer, order, product, picking, packing, delivering and generating invoice.

Functionality Description

There would be 4 kinds of role on the database:

1. Administrator
2. ServiceClerk:
3. Customer
4. Vendor

The Administrator will have complete privileges on the whole database. The ServiceClerk will have access to create, modify, delete sales order, update inventory table, schedule delivery and generating invoice.

The customer calls ServiceClerk to place order and track order details. The vendor will also call ServiceClerk to inform about supplying products in a warehouse.

Depending upon the functionality each user will have its own access.

Use Cases

1. Person Table – In person table, I have included 2 persons one is customer and second is vendor and its having attributes such as firstName, lastName, EmailId, PhoneNumber

```
/*Person Table */

CREATE TABLE person(PersonId int Primary key not null,
                    firstName varchar(45),
                    lastName varchar(45));

alter table person add emailid varchar(45);
alter table person add phonenumber varchar(15);

select * from person;

Update person
set emailid = 'chitra.paryani@gmail.com'
where personid = 1;

Update person
set phonenumber = '1234567890'
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

firstName	lastName	emailid	phonenumber
chitra	parvani	chitra.parvani@gmail.com	1234567890
omega traders	terry turner	omegatraders@gmail.com	7894561230
NULL	NULL	NULL	NULL

2. Address Table – I have separately created address table and linked it with person using personId due to two reasons
   a. Person can have multiple addresses like office address, home address

Date: 14/12/2017 **Electronic Store Management System**

b. To share it with multiple entities like customer, shipper

```
/*Address Table */
CREATE TABLE Address(AddressId int primary key not null,
                     Street1 varchar(45),
                     Street2 varchar(45),
                     country varchar(45),
                     city varchar(45),
                     zipcode varchar(45),
                     personId int,
                     foreign key (personId) references person(personId));

alter table address add state varchar(45);

Insert into address values(1,'65-2','St germain street','USA','Boston','02115',1,'MA');
Select * from address;
/* Query to get address of a person */
Select * from person
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

AddressId	Street1	Street2	country	city	zipcode	personId	state
1	65-2	St germain street	USA	Boston	02115	1	MA
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Query which displays person, address details of person

```
/* Query to get address details, person details of a person */
Select * from person
inner join address
on person.personid = address.personid;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

firstName	lastName	emailid	phonenumber	AddressId	Street1	Street2	country	city
chitra	parvani	chitra.parvani@gmail.com	1234567890	1	65-2	St germain street	USA	Boston

3. Customer Table – I have created customer as subtype which inherits attributes from person supertype. Additional attributes which customer have is credit limit, lastrevised date of credit which gives information about customers credit.
   Below is the query

```
/*Customer Table */
CREATE TABLE customer( CustomerId int primary key not null,
                       creditlimit int,
                       lastrevised date,
                       foreign key (customerid) references person(personId));

Insert into customer values(1, 100000, '2017-12-13');

/* Query to get customer name, address and credit limit */
SELECT * from customer
INNER JOIN person
INNER JOIN address
on person.personid = customer.customerid and
person.personid = address.personid;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

CustomerId	creditlimit	lastrevised	firstName	lastName	emailid	phonenumber	AddressId	Street1	Street2
1	100000	2017-12-13	chitra	parvani	chitra.parvani@gmail.com	1234567890	1	65-2	St germain street

Date: 14/12/2017 **Electronic Store Management System**

4. Vendor Table – I have created vendor table as subtype which inherits attributes of person supertype. Vendor supplies products to a warehouse. Not included more details due to limited scope of this project.
   Executed query to get vendor details by joining it with person

```
/*Vendor Table */
CREATE TABLE vendor(VendorId int primary key not null,
                    foreign key (vendorId) references person(personId));

Insert into vendor values(111);

/* Query to get vendor details */
select * from vendor
INNER JOIN person
on vendor.vendorId = person.personId;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

VendorId	PersonId	firstName	lastName	emailid	phonenumber
111	111	omega traders	terry turner	omegatraders@gmail.com	7894561230

5. Customer_Order Table: This table gives information about order which is placed by customer and on which date
   Below query gives information about order placed by customer

```
/* Customer order Table */
CREATE TABLE Customer_Order(orderid int primary key not null,
                            orderdate date,
                            customerid int,
                            foreign key(customerid) references customer(customerid)
                            ON DELETE NO ACTION
                            ON UPDATE NO ACTION);

INSERT INTO customer_order values(1, '2017-12-13',1);

/* Query to get customer order data */
SELECT * from customer_order
INNER JOIN customer
INNER JOIN Person
INNER JOIN Address
on customer_order.customerid = customer.customerid and
customer.customerid = person.personid and
person.personid = address.personid;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

CustomerId	creditlimit	lastrevised	PersonId	firstName	lastName	emailid	phonenumber	AddressId
1	100000	2017-12-13	1	chitra	parvani	chitra.parvani@gmail.com	1234567890	1

1. Product Table: product table gives information about product like ProductID, Name, description.

Date: 14/12/2017 **Electronic Store Management System**

```
/* Product Table */
CREATE TABLE product(ProductId int primary key not null,
                     Name varchar(45) Not NUll,
                     Description varchar(45) Null);

Alter table product add price int;
Alter table product add currency varchar(10);

INSERT INTO product values(1,'HP Notebook','LCD Touchscreen, AMD A8-7410 APU',111,457, 'dollar');
Select * from product;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

ProductId	Name	Description	vendorid	price	currency
1	HP Notebook	LCD Touchscreen, AMD A8-7410 APU	111	457	dollar
NULL	NULL	NULL	NULL	NULL	NULL

Below is the query which gives product vendor details

```
/*Query to get vendor details*/
SELECT * from product
Inner join vendor
Inner Join person
on vendor.vendorId = product.vendorId and
vendor.vendorId = person.personId;

select * from product;
delete from product where productId = 1;

Alter table product add vendorid int;
Alter table product add constraint fk_product foreign key (vendorid) references vendor(vendorid);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ProductId	Name	Description	ver	price	currency	Ven	Pers	firstName	lastName	emailid
1	HP Notebook	LCD Touchscreen, AMD A8-7410 APU	1...	457	dollar	111	111	omega traders	terry turner	omegatraders6

2. OrderItem Table – This table gives the information about Items ordered by customer

```
/*OrderItem Table*/
CREATE TABLE OrderItem(ID int primary key not null,
                       orderid int not null,
                       productid int not null,
                       quantity int not null,
                       foreign key (orderId) references customer_order(orderId)
                       ON DELETE NO ACTION
                       ON UPDATE NO ACTION,
                       foreign key (productId) references product(productId)
                       ON DELETE NO ACTION
                       ON UPDATE NO ACTION);

Insert into orderItem values(1,1,1,1);
Select * from orderItem;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

ID	orderid	productid	quantity
1	1	1	1
NULL	NULL	NULL	NULL

3. PICK Table – This table gives information about order pick details like pickedby, date and for which order. Location information is not included due to limitations in project and maybe enhanced in future.

Date: 14/12/2017 **Electronic Store Management System**

```sql
/* CREATE TABLE PICK */
CREATE TABLE PICK(PickId int not null primary key,
                  pickdate date,
                  pickedby varchar(45),
                  orderid int,
                  foreign key (orderid) references customer_order(orderid));

INSERT INTO PICK values(1,'2017-12-13','Suresh',1);
```

Below query gives information of order pick up details

```sql
/* SQL query to get order pick details */
SELECT * FROM customer
INNER JOIN person
INNER JOIN address
INNER JOIN customer_order
INNER JOIN product
INNER JOIN orderitem
INNER JOIN pick
on customer.customerId = person.personId and
person.personId = address.personId and
customer.customerId = customer_order.customerId and
customer_order.orderId = orderitem.orderId and
orderitem.productId = product.productId and
orderitem.orderid = pick.orderid;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

CustomerId	creditlimit	lastrevised	Pers	firstName	lastName	emailid	phonenumber	AddressId	Street1
1	100000	2017-12-13	1	chitra	parvani	chitra.parvani@gmail.com	1234567890	1	65-2

4. STOCKPICK Table – This table gives information about product pick details. This table is included to resolve many to many relationship issue between product and pick table.

```sql
/*Create Table Stock Pick */
CREATE TABLE STOCKPICK(id int not null primary key,
                       pickId int,
                       productId int,
                       quantity int not null,
                       foreign key (pickId) references pick(pickId)
                       ON DELETE NO ACTION
                       ON UPDATE NO ACTION,
                       foreign key (productId) references product(productId)
                       ON DELETE NO ACTION
                       ON UPDATE NO ACTION);

INSERT INTO stockpick values(1,1,1,1);
```

5. ScheduleDelivery – This table gives information about delivery schedule like delivery date, shipped by, pick up by whom

Date: 14/12/2017 **Electronic Store Management System**

```
/*Create Table ScheduleDelivery */
CREATE TABLE ScheduleDelivery(DeliveryId int primary key not null,
                              Deliverydate date,
                              shippedby varchar(45),
                              pickId int,
                              customerid int,
                              invoiceid int,
                              foreign key (pickId) references pick(pickId),
                              foreign key (customerId) references customer(customerid)
                              );
```

6. Invoice Table – Invoice is generated once delivery is done.

```
/*Create Invoice Table */
CREATE TABLE invoice(invoiceId int primary key not null,
                     invoicedate date,
                     deliveryid int,
                     invoicetotal int,
                     customerid int,
                     foreign key (customerid) references customer(customerid));

Insert into invoice values(1, '2017-12-15',1,30000,1);
```

7. DeliveryInvoice Table– It is a bridge table between ScheduleDelivery and invoice table

```
/*Create Delivery Invoice Table */
CREATE TABLE DeliveryInvoice(Id int primary key not null,
                             deliveryid int,
                             invoiceid int,
                             invoicedate date,
                             foreign key (deliveryid) references scheduledelivery(deliveryid)
                             ON DELETE NO ACTION
                             ON UPDATE NO ACTION,
                             foreign key (invoiceId) references invoice(invoiceId)
                             ON DELETE NO ACTION
                             ON UPDATE NO ACTION);

INSERT INTO DeliveryInvoice values(1, 1, 1, '2017-12-15');
```

Queries

Query which give complete order to cash flow like customer, product, shipper, delivery and invoice details.

```
/*Query complete O2C flow - order delivered to customer*/
SELECT person.firstName, person.lastName, product.name AS productName, DeliveryInvoice.invoicedate ,
product.price,product.currency,ScheduleDelivery.shippedby from customer
INNER JOIN scheduledelivery
INNER JOIN stockpick
INNER JOIN product
INNER JOIN person
INNER JOIN DeliveryInvoice
on customer.customerid = scheduledelivery.customerid and
customer.customerid = person.personid and
scheduledelivery.pickid = stockpick.pickid and
stockpick.productid = product.productid and
deliveryinvoice.deliveryId = scheduledelivery.deliveryId;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

firstName	lastName	productName	invoicedate	price	currency	shippedby
chitra	parvani	HP Notebook	2017-12-15	457	dollar	Best Buy

Date: 14/12/2017 **Electronic Store Management System**

Added details for another customer

```
/*Query complete O2C flow - order delivered to customer*/
SELECT person.firstName, person.lastName, product.name AS productName, DeliveryInvoice.invoicedate ,
product.price,product.currency,ScheduleDelivery.shippedby from customer
INNER JOIN scheduledelivery
INNER JOIN stockpick
INNER JOIN product
INNER JOIN person
INNER JOIN DeliveryInvoice
on customer.customerid = scheduledelivery.customerid and
customer.customerid = person.personid and
scheduledelivery.pickid = stockpick.pickid and
stockpick.productid = product.productid and
deliveryinvoice.deliveryId = scheduledelivery.deliveryId;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

firstName	lastName	productName	invoicedate	price	currency	shippedby
chitra	parvani	HP Notebook	2017-12-15	457	dollar	Best Buy
priya	parvani	HP ENVY	2017-12-16	1...	dollar	Amazon

Analytics

A great deal of analytics can be done in order to cash flow. Below are the few queries which I have included

1. **Finding top customer based on purchases he/she has done**

```
/*Top Customer*/
SELECT person.firstName, person.lastName, product.name AS productName, DeliveryInvoice.invoicedate ,
product.price,product.currency,ScheduleDelivery.shippedby from customer
INNER JOIN scheduledelivery
INNER JOIN stockpick
INNER JOIN product
INNER JOIN person
INNER JOIN DeliveryInvoice
on customer.customerid = scheduledelivery.customerid and
customer.customerid = person.personid and
scheduledelivery.pickid = stockpick.pickid and
stockpick.productid = product.productid and
deliveryinvoice.deliveryId = scheduledelivery.deliveryId
ORDER BY product.price DESC
LIMIT 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

firstName	lastName	productName	invoicedate	price	currency	shippedby
priya	parvani	HP ENVY	2017-12-16	1000	dollar	Amazon

2. **Products delivered to customer is either on time or delayed which is compared by using requested date and delivery date. If it is delayed then company can immediately take actions by notifying customers about the same or by providing some extra services, or refund the amount which will all help in managing customer relationship.**

Requested Date

orderid	orderdate	customerid	requesteddate
1	2017-12-13	1	2017-12-15
2	2017-12-14	2	2017-12-15
NULL	NULL	NULL	NULL

Date: 14/12/2017

Electronic Store Management System

Result Grid | Filter Rows: | Edit: | Expo

DeliveryId	Deliverydate	shippedby	pickId	customerid
1	2017-12-15	Best Buy	1	1
2	2017-12-16	Amazon	2	2
NULL	NULL	NULL	NULL	NULL

For customer 1, delivery is on time and for customer 2, delivery is delayed

On Time Delivery

```
/*On Time Delivery */
SELECT customer.customerid, person.firstName, person.lastName,
ScheduleDelivery.deliverydate FROM customer_order
INNER JOIN ScheduleDelivery
INNER JOIN customer
INNER JOIN person
on customer.customerid = customer_order.customerid and
person.personid = customer.customerid and
customer_order.customerid = ScheduleDelivery.customerid
where requesteddate >= deliverydate;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customerid	firstName	lastName	deliverydate
1	chitra	parvani	2017-12-15

Delayed Delivery

```
/*Delayed Delivery */
SELECT customer.customerid, person.firstName, person.lastName,
ScheduleDelivery.deliverydate FROM customer_order
INNER JOIN ScheduleDelivery
INNER JOIN customer
INNER JOIN person
on customer.customerid = customer_order.customerid and
person.personid = customer.customerid and
customer_order.customerid = ScheduleDelivery.customerid
where requesteddate < deliverydate;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customerid	firstName	lastName	deliverydate
2	priya	parvani	2017-12-16

Views

Created a temporary inventory table

Date: 14/12/2017 **Electronic Store Management System**

```
/* Created temp inventory table to show stock available in warehouse*/
Create table inventory(id int not null primary key,warehouse varchar(45),
productid int, stockAvailable int,
foreign key (productId) references product(productId));

alter table inventory add Asofdate date;
Update inventory
set Asofdate = '2017-12-13';

Insert into inventory values (1, 'M1',1,100);
Insert into inventory values (2, 'M1',2,100);
Insert into inventory values (3, 'M1',3,100);

select * from inventory;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Co

id	warehouse	productid	stockAvailable	Asofdate
1	M1	1	100	2017-12-13
2	M1	2	100	2017-12-13
3	M1	3	100	2017-12-13
NULL	NULL	NULL	NULL	NULL

View is created to check total stock (which includes stock available in inventory + product ordered by customer)

```
/*Created View to view total stock available including ordered and available */
CREATE View TotalAvailProduct
AS
SELECT orderid, quantity from orderitem
UNION ALL
select productid, stockavailable from inventory;

CREATE view stock
AS
SELECT orderid, sum(quantity) from totalavailproduct group by orderid;

select * from stock;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

orderid	sum(quantity)
1	101
2	101
3	100

INDEX

Index is created in customer_order and orderitem table to access details faster

```
/* CREATE INDEX */
CREATE INDEX OrderIdIndex on customer_order(orderid);

CREATE INDEX OrderItemIndex on OrderItem(orderId);
CREATE INDEX productItemIndex on OrderItem(productId);

desc orderitem;
```

Trigger

Date: 14/12/2017

Electronic Store Management System

I have created order_status_flow field which maintains status of the transactions. Initially, order_status_flow is blank

orderid	orderdate	customerid	requesteddate	order_status_flow
1	2017-12-13	1	2017-12-15	NULL
2	2017-12-14	2	2017-12-15	NULL
NULL	NULL	NULL	NULL	NULL

Once customer places order, order_status_flow changes to Entered by trigger.

```
Insert into customer_order (orderid,orderdate,customerid,requesteddate)
values (3,'2017-12-14',1,'2017-12-20');
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

orderid	orderdate	customerid	requesteddate	order_status_flow
1	2017-12-13	1	2017-12-15	Entered
2	2017-12-14	2	2017-12-15	Entered
3	2017-12-14	1	2017-12-20	Entered
NULL	NULL	NULL	NULL	NULL

Similarly, I have created item_status_flow in orderitem table which is initially null and when customer chooses item and places order, its status changes automatically to 'Ordered' using trigger.

```
alter table orderitem add item_status_flow varchar(45);

Delimiter //
CREATE TRIGGER changeStatusItem
BEFORE INSERT on orderitem
FOR EACH ROW
BEGIN
SET new.item_status_flow = 'Ordered';
END //
Delimiter ;

Insert into orderitem (Id, orderid, productid, quantity) values
(3,3,1,1);


select * from orderitem;
```

esult Grid | Filter Rows: | Edit: | Export/Import: | Wrap

ID	orderid	productid	quantity	item_status_flow
1	1	1	1	Ordered
2	2	2	1	Ordered
3	3	1	1	Ordered
NULL	NULL	NULL	NULL	NULL

Similarly, trigger is created when item is picked for delivery.

Result Grid | Filter Rows: | Edit:

PickId	pickdate	pickedby	orderid	pick_status_flow
1	2017-12-13	Suresh	1	Picked
2	2017-12-14	Ramesh	2	Picked
NULL	NULL	NULL	NULL	NULL

```
/*Trigger for pick */
select * from pick;

alter table pick add pick_status_flow varchar(45);
alter table pick drop pick_status_flow;
update pick
set pick_status_flow = 'Picked';

Delimiter //
CREATE TRIGGER changeStatusPick
BEFORE INSERT on pick
FOR EACH ROW
BEGIN
SET new.pick_status_flow = 'Picked';
END //
Delimiter ;

Insert into pick (pickid, pickdate, pickedby, orderid) values
(3,'2017-12-14','Mahesh',3);

select * from pick;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

PickId	pickdate	pickedby	orderid	pick_status_flow
1	2017-12-13	Suresh	1	Picked
2	2017-12-14	Ramesh	2	Picked
3	2017-12-14	Mahesh	3	Picked
NULL	NULL	NULL	NULL	NULL

Similarly, when item is delivered status changes to delivered

```
Delimiter //
CREATE TRIGGER changeStatusDelivery
BEFORE INSERT on scheduledelivery
FOR EACH ROW
BEGIN
SET new.delivery_status_flow = 'Delivered';
END //
Delimiter ;

Insert into scheduledelivery
(deliveryid, deliverydate, shippedby, pickid, customerid) values
(3,'2017-12-20','Best Buy',3,1);

delete from scheduledelivery where DeliveryId = 3;

select * from scheduledelivery;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

DeliveryId	Deliverydate	shippedby	pickId	customerid	delivery_status_flow
1	2017-12-15	Best Buy	1	1	Delivered
2	2017-12-16	Amazon	2	2	Delivered
3	2017-12-20	Best Buy	3	1	Delivered
NULL	NULL	NULL	NULL	NULL	NULL

Using this status, entire order to cash can be tracked.

procedure

Procedure is created to find product name and its category

```
/* Procedure to get product Category */
Delimiter //
Create procedure product_cat_view(IN categoryvar int)
BEGIN
Select productcategory.Category, product.Name from product, productcategory
where product.categoryid = productcategory.Categoryid and
productcategory.CategoryId = categoryvar;
END //
Delimiter ;

drop procedure product_cat_view;

Call product_cat_view(1);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Category	Name
Laptop	HP Notebook
Laptop	HP ENVY
Laptop	DELL

Second procedure is created to find total number of products based on category

```
/*count of products based on category */
Delimiter //
Create procedure product_cat_count(IN categoryvar int, OUT total int)
BEGIN
Select productcategory.Category, count(product.Name) from product, productcategory
where product.categoryid = productcategory.Categoryid and
productcategory.CategoryId = categoryvar;
END //
Delimiter ;

Call product_cat_count(1, @total);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Category	count(product.Name)
Laptop	3

Functions

Function is created to determine Vendors rating

```
/*created function to find vendor rating */
Delimiter %%
create function fn_vendorRating(vendorID int)
returns varchar(200)
begin
    DECLARE result varchar(200);
    SET result:=
   (select concat(vendorid, ' | ',repeat('*',rating))
   as rating
   from vendor v,v_rating vr
   where v.vendorid = vr.vendorid and vr.vendorid = vendorId);
   RETURN result;
end %%
DELIMITER ;

select fn_vendorRating(111);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

fn_vendorRating(111)
111 | *****

Customer Rating

Second function is created to determine customer rating

```
/* Create function to find customer rating */
Delimiter %%
create function fn_custRating(customerID int)
returns varchar(200)
begin
    DECLARE result varchar(200);
    SET result:=
   (select concat(customerid, ' | ',repeat('*',rating))
   as rating
   from customer v,c_rating vr
   where v.customerid = vr.customerid and vr.customerid = customerID);
   RETURN result;
end %%
DELIMITER ;

drop function fn_custRating;
select fn_custRating(1);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	fn_custRating(1)
1	*****

User Access

UserAccess provided to ServiceClerk

```
/* Service Clerk */
CREATE USER ServiceClerk  IDENTIFIED BY 'password';

-- Revoke all privileges for the user
REVOKE ALL privileges, grant option from ServiceClerk;

-- Grant needed privileges
GRANT SELECT ON customer to ServiceClerk;

GRANT UPDATE (firstName, lastName, EmailId, PhoneNumber)
ON customer TO ServiceClerk;

GRANT SELECT, DELETE ON customer_order to ServiceClerk;

GRANT INSERT, SELECT, DELETE on orderItem to ServiceClerk;

GRANT SELECT on inventory to ServiceClerk;

/* Entry Clerk */
```

Output

Action Output

#	Time	Action	Message
105	03:04:15	GRANT INSERT, SELECT, DELETE on orderItem to ServiceClerk	0 row(s) affected
106	03:04:44	GRANT SELECT on inventory to ServiceClerk	0 row(s) affected
107	03:06:52	CREATE USER EntryClerk IDENTIFIED BY 'password'	0 row(s) affected
108	03:06:55	REVOKE ALL privileges, grant option from ServiceClerk	0 row(s) affected
109	03:07:54	GRANT INSERT, SELECT on customer TO EntryClerk	0 row(s) affected

Date: 14/12/2017

Electronic Store Management System

Tables, Data Types and Domain

Person Table

```
314 • desc person;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
PersonId	int(11)	NO	PRI	NULL	
firstName	varchar(45)	YES		NULL	
lastName	varchar(45)	YES		NULL	
emailid	varchar(45)	YES		NULL	
phonenumber	varchar(15)	YES		NULL	

Address Table

```
316 • desc address;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
AddressId	int(11)	NO	PRI	NULL	
Street1	varchar(45)	YES		NULL	
Street2	varchar(45)	YES		NULL	
country	varchar(45)	YES		NULL	
city	varchar(45)	YES		NULL	
zipcode	varchar(45)	YES		NULL	
personId	int(11)	YES	MUL	NULL	
state	varchar(45)	YES		NULL	

Customer Table

```
318 • desc customer;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Co

Field	Type	Null	Key	Default	Extra
CustomerId	int(11)	NO	PRI	NULL	
creditlimit	int(11)	YES		NULL	
lastrevised	date	YES		NULL	

Customer_Order Table

```
320 • desc customer_order;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
orderid	int(11)	NO	PRI	NULL	
orderdate	date	YES		NULL	
customerid	int(11)	YES	MUL	NULL	
requesteddate	date	YES		NULL	
order_status_flow	varchar(45)	YES		NULL	

Date: 14/12/2017

Electronic Store Management System

OrderItem Table

```
322 • desc orderItem;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
ID	int(11)	NO	PRI	NULL	
orderid	int(11)	NO	MUL	NULL	
productid	int(11)	NO	MUL	NULL	
quantity	int(11)	NO		NULL	
item_status_flow	varchar(45)	YES		NULL	

Product Table

```
324 • desc product;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Conte

Field	Type	Null	Key	Default	Extra
ProductId	int(11)	NO	PRI	NULL	
Name	varchar(45)	NO		NULL	
Description	varchar(45)	YES		NULL	
vendorid	int(11)	YES	MUL	NULL	
price	int(11)	YES		NULL	
currency	varchar(10)	YES		NULL	
categoryid	int(11)	YES		NULL	

Vendor Table

```
326 • desc vendor;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Cor

Field	Type	Null	Key	Default	Extra
VendorId	int(11)	NO	PRI	NULL	

StockPick Table

```
328 • desc stockpick;
```

Result Grid | Filter Rows: | Export: | Wrap Cell C

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	
pickId	int(11)	YES	MUL	NULL	
productId	int(11)	YES	MUL	NULL	
quantity	int(11)	NO		NULL	

Pick Table

Date: 14/12/2017 **Electronic Store Management System**

```
330 • desc pick;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
PickId	int(11)	NO	PRI	NULL	
pickdate	date	YES		NULL	
pickedby	varchar(45)	YES		NULL	
orderid	int(11)	YES	MUL	NULL	
pick_status_flow	varchar(45)	YES		NULL	

ScheduleDelivery Table

```
332 • desc scheduledelivery;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
DelivervId	int(11)	NO	PRI	NULL	
Delivervdate	date	YES		NULL	
shippedby	varchar(45)	YES		NULL	
pickId	int(11)	YES	MUL	NULL	
customerid	int(11)	YES	MUL	NULL	
delivery_status_flow	varchar(45)	YES		NULL	

DeliveryInvoice Table

```
333
334 • desc deliveryinvoice;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Conte

Field	Type	Null	Key	Default	Extra
Id	int(11)	NO	PRI	NULL	
delivervid	int(11)	YES	MUL	NULL	
invoiceid	int(11)	YES	MUL	NULL	
invoicedate	date	YES		NULL	

Invoice Table

```
336 • desc invoice;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Conter

Field	Type	Null	Key	Default	Extra
invoiceId	int(11)	NO	PRI	NULL	
invoicedate	date	YES		NULL	
delivervid	int(11)	YES	MUL	NULL	
invoicetotal	int(11)	YES		NULL	
customerid	int(11)	YES	MUL	NULL	

ProductCategory Table

Date: 14/12/2017 **Electronic Store Management System**

```
338 • desc productcategory;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
CategoryId	int(11)	YES		NULL	
Category	varchar(45)	YES		NULL	

Vendor Rating Table

```
340 • desc v_rating;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Cont

Field	Type	Null	Key	Default	Extra
vendorid	int(11)	YES	MUL	NULL	
rating	int(11)	YES		NULL	

Customer Rating Table

```
342 • desc c_rating;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Co

Field	Type	Null	Key	Default	Extra
customerid	int(11)	YES	MUL	NULL	
rating	int(11)	YES		NULL	