E-Commerce database management system

Members:

Chandan Kumar (2019ucs0094)
Anshul Sharma (2019ucs0074)
Abhishek Maurya (2019uee0134)
Abhay Yadav (2019uee0110)
Sachin Naik (2019ucs0091)
Anant Kumar (2019uce0059)
Saurabh Kumar Verma (2019uch0033)
Balwant Yadav (2019uch0009)

Enterprise-Description

Customer

A unique identification number is given to every customer .
Every customer has name , gender , customer address , customer phone number .
And every customer has different id types which will describe about id discount .
Every customer has last purchase details.

Seller

A unique identification number is given to seller.
Every seller has name , address , contact number , gmail id .
Each seller must have one and unique gmail_id

Product

Every product has unique identification number and each product have different product number .
Every product belong to some brand name.
For every product there will be quantity , cost, discount and ratings properties should be listed.

Cart

For every customer a unique identification cart id should be given.
Every cart contains the total number of products and amount of the cart items.

Cart Items:

Every cart items will contain the user chosen products along with their quantity and date of addition.

Payment:

For every payment done by the customer a unique payment id should be given.
Payment date and time should be listed at which time the payment will be done
Total amount & payment mode should be also contained.

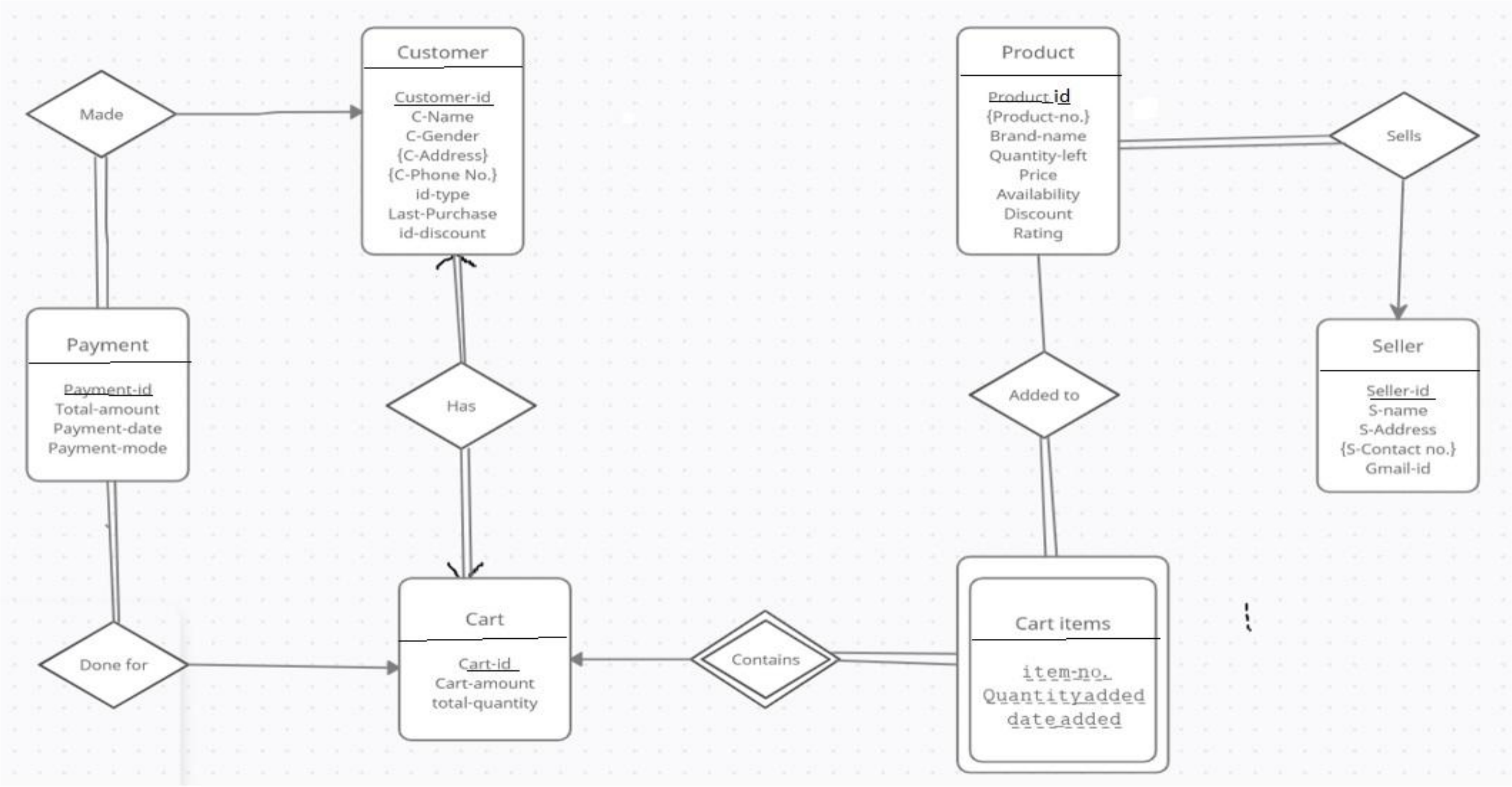
Entities :

Customer, Payment, Cart, Cart-items, Product, Seller

Relationship :

Entity 1	**Entity 2**	**Relationship**
Customer	Payment	Made
Payment	Cart	Done for
Cart	Cart Items	Contains
Product	Cart items	Added to
Customer	Cart	has
Seller	Product	Sells

E-R Diagram


Customer
Customer-id
C-Name
C-Gender
{C-Address}
{C-Phone No.}
id-type
Last-Purchase
id-discount
Made
Payment
Payment-id
Total-amount
Payment-date
Payment-mode
Has
Done for
Cart
Cart-id
Cart-amount
total-quantity
Contains
Cart items
item-no.
Quantityadded
dateadded
Added to
Product
Product id
{Product-no.}
Brand-name
Quantity-left
Price
Availability
Discount
Rating
Sells
Seller
Seller-id
S-name
S-Address
{S-Contact no.}
Gmail-id

Reduction to relational schema

1. Payment (payment_id, total_amount, payment_date , payment_mode , customer_id , cart_id)
2. Customer (customer_id, customer_name, gender, id_type, last_purchase, id_discount, cart_id)
3. Cart (cart_id, cart_amount, total_amount)
4. Cart_item (item_id, quantity_added, date_added, cart_id)
5. Product (product_id, brand_name, quantity_left, price,availability, discount, rating, seller_id)
6. Seller (seller_id, seller_name, gmail_id)
7. Seller_addr(seller_id, seller_address)
8. Customer_addr (customer_id, customer_address)
9. Customer_phone (customer_id, customer_phone_number)
10. Product_item_number (product_id, product_number)
11. Added_to (product_id, item_id, quantity_added, date_added, cart_id)
12. Seller_phone (seller_id, seller_phone_number)

Payment (payment_id, total_amount, payment_date , payment_mode , customer_id , cart_id)

Non trivial Functional dependencies:

F = {payment_id→ total_amount,
payment_id→ payment_date,
payment_id→ payment_mode,
payment_id→ customer_id,
customer_id→ cart_id,
Cart_id→customer_id }

Closure of F :

Taking, payment_id=A, total_amount=B, payment_date=C, payment_mode=D, customer_id=E, cart_id=F.

Closure of F (F^+) : {A→B, A→C, A→D, A→E, E→F, F→E,A→F, A→BC, A→CD, A→DE, A→EF, A→BD, A→BE, A→BF, A→CE, A→CF, A→BCD, A→CDF, A→BCDE, A→CDEF, A→BCDEF, A→ABC, A→ACD, A→ACDE, A→ABCDE, A→ACDEF, A→ABCDEF, A→ABCD, E→EF, F→EF……}

Closure of attributes :(Payment_id $^+$) → {payment_id, total_amount, payment_date, payment_mode, customer_id, cart_id}

(customer_id$^+$) → {customer_id , cart_id }

(cart_id$^+$) → {cart_id , customer_id}

No extraneous attributes found , so canonical cover of functional dependencies is same as closure of functional dependencies

Candidate key : payment_id

Prime attributes : payment_id

Non prime attributes : total_amount, payment_date , payment_mode , customer_id , cart_id

Normal Form:

- Here all attributes are atomic, so it is in 1st Normal form.
- Here all non prime attributes are fully functional dependent on primary key. So it is in 2nd Normal form.
- non-prime attributes are transitively dependent on primary key so it is not in 3rd NF.
- r1(payment_id, total_amount, payment_date , payment_mode , customer_id)
- r2(customer_id , cart_id)
- Decomposed relation is in BCNF.

Customer (customer_id, customer_name, gender, id_type, last_purchase, id_discount, cart_id)

Non trivial Functional dependencies :

F={ customer_id→customer_name,
customer_id→gender,
customer_id→id_type,
customer_id→last_purchase,
customer_id→cart_id,
id_type→id_discount,
cart_id→customer_id}

Closure of F :

Taking, customer_id=A, customer_name=B, gender=C, id_type=D, last_purchase=E,
id_discount=F, cart_id=G.

Closure of F (F^{+}) :

{A→B, A→C, A→D, A→E, A→F, A→G, D→F, G→A, G→B, G→C, G→D, G→E, G→F, A→BC, A→CD, A→DE, A→EF, A→FG, A→BD, A→BF, A→BE, A→BCD, A→BCDE, A→BCDEF, A→CDE, A→CDEF, A→AB, A→AC, A→AD, A→AE, A→AF, A→ABC, A→ABD, A→ACD, A→ABCD, A→ACDE, A→DEF, A→ABCDEFG, G→BC, G→CD, G→DE, G→EF, G→FA, G→BD, G→BF, G→BE, G→BCD, G→BCDE, G→BCDEF, G→CDE, G→CDEF, G→GB,G→AC, G→AD, G→AE, G→AF, G→ABC, G→ABD, G→ACD, G→ABCD, G→ACDE, G→DEF, G→ABCDEFG, AG→BC, AG→CD, AG→DE, AG→EF, AG→FA, AG→BD, AG→BF, AG→BE, AG→BCD, AG→BCDE, AG→BCDEF, AG→CDE, AG→CDEF, AG→GB, AG→AC, GA→AD, AG→AE, AG→AF, AG→ABC, AG→ABD, AG→ACD, AG→ABCD, AG→ACDE, AG→DEF, AG→ABCDEFG,......}

Closure of attributes :

(customer_id^{+}) → {customer_id, customer_name, gender ,id_type, last_purchase, id_discount, cart_id}

(cart_id^{+}) → {customer_id, cart_id , customer_name , gender ,id_type, last_purchase , id_discount }

(id_type^{+}) → { id_type , id_discount}

Canonical cover of functional dependencies :

{A→B, A→C, A→D, A→E, A→F, A→G, D→F, G→A, G→B, G→C, G→D, G→E, G→F, A→BC, A→CD, A→DE,
A→EF, A→FG, A→BD, A→BF, A→BE, A→BCD, A→BCDE, A→BCDEF, A→CDE, A→CDEF, A→AB, A→AC,
A→AD, A→AE, A→AF, A→ABC, A→ABD, A→ACD, A→ABCD, A→ACDE, A→DEF, A→ABCDEFG, G→BC,
G→CD, G→DE, G→EF, G→FA, G→BD, G→BF, G→BE, G→BCD, G→BCDE, G→BCDEF, G→CDE, G→CDEF,
G→GB, G→AC, G→AD, G→AE, G→AF, G→ABC, G→ABD, G→ACD, G→ABCD, G→ACDE, G→DEF,
G→ABCDEFG,......}

Candidate key: customer_id , cart_id

Prime attributes : customer_id, cart_id

Non-Prime attributes: customer_name, gender, id_type, last_purchase, id_discount

Normal Form:

- Here all attributes are atomic, so it is in 1st Normal form.
- Here all non prime attributes are fully functional dependent on primary key. So it is in 2nd Normal form.
- non-prime attributes are transitively dependent on primary key so it is not in 3rd NF.
- r1(customer_id, customer_name, gender, id_type, last_purchase, cart_id)
- r2(id_type, id_discount)
- r3(customer_id, cart_id)
- Now, all relation also in BCNF.

Cart (cart_id, cart_amount, total_quantity)

Non trivial Functional dependencies:

F= {cart_id→cart_amount ,
cart_id→total_quantity}

Closure of F :

Taking, cart_id=A, cart-amount=B, total_quantity=C

Closure of F (F^{+}) : {A→B, A→C, A→BC, A→AB, A→AC, A→ABC}

Closure of Attributes: (cart_id^{+}) → {cart_id , cart_amount , total_quantity}

No extraneous attributes found , so canonical cover of functional dependencies is same as closure of functional dependencies .

Candidate key : cart_id

Prime attributes : cart_id

Non prime attributes : cart_amount, total_quantity

Normal Form:

- Here all attributes are atomic, so it is in 1st Normal form..
- Here all non prime attributes are fully functional dependent on primary key. So it is in 2nd Normal form.
- No non-prime attributes are transitively dependent on primary key so it is in 3rd NF.
- It is in BCNF form.

Product (product_id, brand_name, quantity_left, price,availability, discount, rating, seller_id)

Non trivial Functional dependencies:

F = {product_id→ brand_name,

product_id→ quantity_left,

product_id→ price,

product_id→ discount,

product_id→rating,

product_id→ seller_id,

quatity_left→availability}

Closure of F :

Taking, product_id=A, brand_name=B, quantity_left=C, price=D, availability=E, discount=F, rating=G, seller_id=H.

Closure of F (F^+) :

{A→B, A→C, A→D, A→F, A→G, A→H, C→E, A→E,A→BC, A→CD, A→DE, A→EF, A→FG, A→BD, A→BF, A→BE, A→BCD, A→BCDE, A→BCDEF, A→CDE, A→CDEF, A→AB, A→AC, A→AD, A→AE, A→AF, A→ABC, A→ABD, A→ACD, A→ABCD, A→ACDE, A→DEF, A→ABCDEFG,A→BCH, A→CDH, A→DEH, A→EFH, A→FGH, A→BDH, A→BFH, A→BEH, A→BCDH, A→BCDEH, A→BCDEFH, A→AD, A→AE,A→ABCH, A→ABDH, A→ACDH, A→ABCDH, A→ACDEH, A→DEFH, A→ABCDEFGH, C→CE, AC→BCDE, AC→BCDEF, AC→CDE, AC→CDEF, AC→AB, AC→AC, AC→AD, AC→AE, AC→AF, AC→ABC, AC→ABD, AC→ACD, AC→ABCD, AC→ACDE, AC→DEF, AC→ABCDEFG,AC→BCH, AC→CDH, AC→DEH, AC→EFH, AC→FGH, AC→BDH, AC→BFH, AC→BEH, AC→BCDH, AC→BCDEH,....}

Closure of attributes:

(product_id$^+$) → { product_id ,brand_name , quantity_left , price , availability , discount , rating , seller_id}

(quantity_left$^+$) → { quantity_left , availability}

Canonical cover of functional dependencies

{A→B, A→C, A→D, A→F, A→G, A→H, C→E,A→E, A→BC, A→CD, A→DE, A→EF, A→FG, A→BD, A→BF, A→BE, A→BCD,
A→BCDE, A→BCDEF, A→CDE, A→CDEF, A→AB, A→AC, A→AD, A→AE, A→AF, A→ABC, A→ABD, A→ACD, A→ABCD,
A→ACDE, A→DEF, A→ABCDEFG,A→BCH, A→CDH, A→DEH, A→EFH, A→FGH, A→BDH, A→BFH, A→BEH, A→BCDH,
A→BCDEH, A→BCDEFH, A→AD, A→AE,A→ABCH, A→ABDH, A→ACDH, A→ABCDH, A→ACDEH, A→DEFH,
A→ABCDEFGH, C→CE, ...}

Candidate key : product_id

Prime attributes : product_id

Non prime attributes : brand_name, quantity_left, price, availability, discount, rating, seller_id

Normal Form:

- Here all attributes are atomic, so it is in 1st Normal form.
- Here all non prime attributes are fully functional dependent on primary key. So it is in 2nd Normal form.
- non-prime attributes are transitively dependent on primary key so it is not in 3rd NF.
- r1(product_id, brand_name, quantity_left, price, discount, rating, seller_id)
- r2(quantity_left, availability)
- Decomposed relation is in BCNF.

Seller (seller_id, seller_name, gmail_id)

Non trivial Functional dependencies:

F={seller_id→seller_name,

Seller_id→gmail_id,

gmail_id→Seller_id,

gmail_id→Seller_name}

Closure of F :

Taking, seller_id=A, seller_name=B, gmail_id=C

Closure of F (F^{+}) :

{A→B, A→C,C→A, C→B, C→AB, C→ABC, A→BC, A→AB, A→AC, A→ABC }

Closure of attributes :

(seller_id^{+}) → { seller_id , seller_name , gmail_id}

(gmail_id^{+}) → { seller_id , seller_name , gmail_id}

No extraneous attributes found , so canonical cover of functional dependencies is same as closure of functional dependencies

Candidate key : seller_id, gmail_id

Prime attributes : seller_id, gmail_id

Non prime attributes : seller_name

Normal Form:

- Here all attributes are atomic, so it is in 1st Normal form.
- Here all non prime attributes are fully functional dependent on primary key. So it is in 2nd Normal form.
- No non-prime attributes are transitively dependent on primary key so it is in 3rd NF.
- It is in BCNF form.

THANK YOU