ecommerse application

SDEC-2)

we have 2 micro services

① order management service ② payment processing microservice.

→ the order management service → received by request to place order.

@Transaction(readOnly=true) → no need transaction not services.

① orders → table (order_no, order_placed_dt customer mobile email address quantity price), shipping_address_id, billing_address_id

② order_line_items } table → in this order what are the products are there.

we insert order_no, name of the customer, amount

→ order_id product_id quantity unit_price

③ address

address_id, address_line1, address_line2, city, state, zip

→ once we complete the order now we need to place request to payment processing service.

→ payment processing service payment information now we make a external service callout to whoom banking application / a payment gateway application.

→ Based on output of payment information know he will insert the data into payment_table know. will make a external service callout to whoom banking application based on the payment output of the payment gate way application he will insert the payment

information of payment_table. (insert data.

Payment_table

order_id payment_id Payment_method date amount status

2 seperate databases.

Payment processing failed. (external failure balance insufficient bank delay payment)

→ when can we say order has to be sucess |

In such a case

→ when all of the operations across the microservices without failure/exception completely completed → then all the tables has to be commited (inserted.

otherwise everything should be rollback.

→ If it has should be own database now how can I make sure that the transactionality can be applied across the database of my application now?

I need to go for what →

→ begin a transaction while performing operation commit a transaction and roll back on each and every database individually. i.e. how we manage transactions now.?

ANS NO It result in data in

How to implement transactionality across the microservices

→ when we choose database per service design pattern, they could be a business transaction that might be spanning across multiple microservices calls in such case one solution we have in applying the transactionality across the microservices is using global / 2 Phase commit transactions.

→ we can use 2-phase commit as global

V.V. Engg.

across transactionality