accountActivities

PK _accountActivityId

accountId
ipAddress
loginDate
metadata
createdAt

sellers

PK _sellerId

accountId
sellerId
businessName
avatarImage
storeBannerImage
privateMobileNumber
lastUpdatedAt

inventories

PK _inventoryId

inventoryId
sellerId
[inventoryItems]
[{ productId, stockCount }]
lastUpdatedAt

accounts

PK _accountId

accountId
[accountType]
username
password
salt
email
isVerified
verifiedAt
createdAt
lastUpdatedAt

products

PK _productId

productId
[productCategory]
sellerId
name
sku
image
description
sellingPrice
avgReviewScore
isDeleted
deleteReason
deletedAt
createdAt
lastUpdatedAt

track products even when deleted, so that the associated reviews will still exist

reviews linked to orderId, why? Because one product from a seller can't always be 100% perfect, there might be defects, and it is common to provide feedbacks per product bought that is related to a specific order, it doesn't make sense to leave a review if you haven't even bought a product.

buyers

PK _buyerId

accountId
buyerId
firstName
lastName
privateMobileNumber
deliveryAddress
deliveryCity
deliveryPostalCode
deliveryCountry
deliveryContactNumber
lastUpdatedAt

orderItemGroups

PK _orderItemGroupId

orderItemGroupId
orderId
[orderItems]
[{ productId, orderCount }]
lastUpdatedAt

reviews

PK _reviewId

reviewId
orderId
productId
buyerId
rating
comment
createdAt
lastUpdatedAt

orders

PK _orderId

orderId
buyerId
taxPercent
totalOrderPrice
[paymentMethod]
paymentResult
isDelivered
deliveredAt
isPaid
paidAt
createdAt
lastUpdatedAt

ProductService

QUERY
+getAllActiveProducts(): Promise<Product[]>
+getProductById(productId: string): Promise<Product>
+getAllProductsBySellerId(sellerId: string): Promise<Product[]>
+getProductsByName(name: string): Promise<Product[]>

MUTATION
+createProduct(createProductInput: CreateProduct): Promise<Product>
+updateProduct(updateProductInput: UpdateProduct): Promise<Product>
+deleteProduct(productId: string): Promise<Product>

AuthService

QUERY
+getAccountByUsername(username: string): Promise<Account>
+login(username: string, password: string): Promise<String>

AccountService

QUERY
+getAccountById(accountId: string): Promise<Account>

MUTATION
+createAccount(createAccountInput: CreateAccount): Promise<Account>
+updateAccount(updateAccountInput: UpdateAccount): Promise<Account>