Questions	Users -ShippingAddresses	Users-CartItems	Users-Orders
Would keeping the pieces of information together lead to a simpler data model and simpler code?	Yes -> **EMB**	Yes -> **EMB**	Yes -> **EMB**
Do the pieces of info have a “has a” or “contains a” relationship?	Yes -> **EMB**	Yes -> **EMB**	Yes -> **EMB**
Does the application query the pieces of information together?	Yes -> **EMB** (As shipping address is based on user_id)	Yes -> **EMB** (As CartItems’ is based on user_id)	Yes -> **EMB** (As orders are based on user_id)
Are the entities updated together?	No -> REF (Any change in User doesn’t imply a change in ShippingAddress)	No -> REF (Any change in User doesn’t imply change in CartItem)	No -> REF (Any change in User doesn’t imply any change in Order)
Do the entities need to be archived together?	Yes -> **EMB** (Since they’re related)	No -> REF (Since CartItems are temporary data).	Yes -> **EMB** (Since they’re related).
Is there a high cardinality on the child side of the relationship?	No -> **EMB** (Since a user is unlikely to store hundreds of addresses)	No -> **EMB** (Since a Cart can have upto 20 CartItems)	Yes -> REF (Since a user can have 1000s of orders)
Would data duplication be too complicated to manage and undesired?	No -> **EMB** (Addresses aren’t being duplicated)	No -> **EMB** (CartItems won’t be duplicated)	Yes -> REF (A user may have 1000s of orders)
Would the embedded piece grow without bound?	No -> **EMB** (A user is unlikely to store hundreds of addresses)	No -> **EMB** (A User can have a maximum of 20 items of different kinds in the cart)	Yes -> REF -> (A User can have 1000s of orders)
Would the combined pieces of information take up too much memory or transfer too much bandwidth to the application?	No -> **EMB** (A user is unlikely to store hundreds of addresses)	No -> **EMB** (A User can have a maximum of 20 items of different kinds in the cart)	**Yes -> REF -> (A User can have 1000s of orders) OVRD**

Are the pieces of info written at different times in a write-heavy workload?	No -> **EMB** (They're related)	No -> **EMB** (They're related)	No -> **EMB** (They're related)
For the child side of the relationship, can the pieces exist by themselves without a parent?	No -> **EMB** (A shipping address can't exist without the user)	No -> **EMB** (A cartitem can't exist without a user)	No -> **EMB** (An order can't exist without a user)
SCORE	EMB (10), REF(1) **EMB**	EMB (9), REF(2) **EMB**	EMB(6), REF(5), **OVRD -> REF (Question 8)**

Questions	Orders-OrderItems	OrderItems-Products	OrderItems-ProductItems
Would keeping the pieces of information together lead to a simpler data model and simpler code?	Yes -> **EMB** -> (Embed **OrderItems** in **Orders**. An Order can contain up to 5 OrderItems)	Yes -> **EMB** -> (Embed product_id in the respective **OrderItem**. This is because a **Product** can be present in N orders (where N = millions) and each **Product** Document can't have millions of order_ids)	Yes -> **EMB** (Embed sku in the respective **Orderitem**. This is because a **ProductItem** can be present in N orders (where N = millions) and each **ProductItem** Document can't have millions of order_ids.)
Do the pieces of info have a "has a" or "contains a" relationship?	Yes -> **EMB**	Yes -> **EMB**	Yes -> **EMB**
Does the application query the pieces of information together?	Yes -> **EMB**	Yes -> **EMB**	Yes -> **EMB**
Are the entities updated together?	Yes -> **EMB** (The total_price of the Order has to change if the price of one of the order items changes).	Yes -> **EMB** (The product_id in OrderItem has to change if it is updated in Product).	Yes -> **EMB** (The sku in OrderItem has to change if it is updated in ProductItem).
Do the entities need to be archived together?	Yes -> **EMB** (They're related)	Yes -> **EMB** (They're related)	Yes -> **EMB** (They're related)
Is there a high cardinality (current or growing) on the child side of the relationship?	No **->** **EMB** (Each order can have 5 orderitems only).	No -> **EMB** (Each orderitem **is** 1 product).	No -> **EMB** (Each orderitem **is** 1 productitem).
Would data duplication be too complicated to manage and undesired?	No -> **EMB** (No data will be duplicated).	**Yes -> REF**	**Yes -> REF**

Would the combined size of the pieces of information take up too much memory or transfer too much bandwidth to the application?	No **-> EMB** (Each order can have 5 orderitems only).	No -> **EMB** (Each orderitem **is** 1 product).	No -> **EMB** (Each orderitem **is** 1 productitem).
Would the embedded piece grow without bound?	No **-> EMB** (Each order can have 5 orderitems only).	No -> **EMB** (Each orderitem **is** 1 product).	No -> **EMB** (Each orderitem **is** 1 productitem).
Are the pieces of info written at different times in a write-heavy workload?	No -> **EMB** (They're related)	No -> **EMB** (They're related)	No -> **EMB** (They're related)
For the child side of the relationship, can the pieces exist by themselves without a parent?	No -> **EMB** (An OrderItem can't exist without an Order).	Yes -> REF (A Product can exist without an OrderItem).	Yes -> REF (An OrderItem can exist without a ProductItem).
SCORE	EMB (11), REF (0) **EMB**	EMB (10), REF (1) **REF** (**OVRD**) Reference product_id of **Product** in the respective **OrderItem**.	EMB (10), REF (1) **REF** (**OVRD**) Reference **ProductItem** sku in the respective **Orderitem**.

Questions	Products - ProductImages	Products - ProductItems	CartItems - ProductItems
Would keeping the pieces of information together lead to a simpler data model and simpler code?	Yes -> **EMB**	Yes -> **EMB**	Yes -> **EMB**
Do the pieces of info have “has a” or “contains a” relationship?	Yes -> **EMB**	Yes -> **EMB**	Yes -> **EMB**
Does the application query the pieces of information together?	Yes -> **EMB**	Yes -> **EMB**	Yes -> **EMB**
Are the entities updated together?	No -> REF (An update to an ProductImage doesn’t mean update to Product)	No -> REF (Any change in ProductItem doesn’t lead to a change in Product)	Yes -> **EMB** (Any change in the sku of the ProductItem will reflect in the sku of the CartItem).
Do the entities need to be archived together?	Yes -> **EMB** (They’re related)	Yes -> **EMB** (They’re related)	Yes -> **EMB** (They’re related)
Is there a high cardinality (current or growing) on the child side of the relationship?	No -> **EMB** (A Product can have upto 5 ProductImages only).	No -> **EMB** (A Product can have upto 19 ProductItems only).	No -> **EMB** (A CartItem **is** a ProductItem).
Would data duplication be too complicated to manage and undesired?	No -> **EMB** (No data is being duplicated).	**Yes -> REF**	**Yes -> REF**
Would the combined size of the pieces of information take up too much memory or transfer too much bandwidth to the application.	No -> **EMB** (A Product can have upto 5 ProductImages only).	No -> **EMB** (A Product can have upto 19 ProductItems only).	No -> **EMB** (A CartItem **is** a ProductItem).
Would the embedded piece	No -> **EMB** (A Product can have	No -> **EMB** (A Product can have	No -> **EMB** (A CartItem **is** a

grow without bound?	upto 5 ProductImages only).	upto 19 ProductItems only).	ProductItem).
Are the pieces of info written at different times in a write-heavy workload?	No -> **EMB** (They're related)	No -> **EMB** (They're related)	No -> **EMB** (They're related)
For the child side of the relationship, can the pieces exist by themselves without a parent?	No -> **EMB** (A ProductImage can't exist without a Product).	No -> **EMB** (A ProductItem can't exist without a Product).	Yes -> REF (A ProductItem can exist without a CartItem).
SCORE	EMB (10), REF(1) **EMB**	EMB (10), REF(1) **REF** (**OVRD**) Reference **Product** in **ProductItem**.	EMB (10), REF(1) **REF** (**OVRD**) Reference respective **ProductItem** sku in each **CartItem**.

Question	CartItems - Products	Products - Reviews
Would keeping the pieces of information together lead to a simpler data model and simpler code?	Yes -> **EMB**	Yes -> **EMB**
Do the pieces of info have "has a" or "contains a" relationship?	Yes -> **EMB**	Yes -> **EMB**
Does the application query the pieces of information together?	Yes -> **EMB**	Yes -> **EMB**
Are the entities updated together?	Yes -> **EMB**	No -> REF (A Review can be updated without updating a product).
Do the entities need to be archived together?	Yes -> **EMB** (They're related)	Yes -> **EMB** (They're related)
Is there a high cardinality (current or growing) on the child side of the relationship?	No -> **EMB** (A CartItem **is** a Product).	Yes -> REF (A Product can have millions of Reviews).
Would data duplication be too complicated to manage and undesired?	**Yes -> REF**	Yes -> REF (A Product can have millions of Reviews).

Would the combined size of the pieces of information take up too much memory or transfer too much bandwidth to the application.	No -> **EMB** (A CartItem **is** a Product).	Yes -> REF (A Product can have millions of Reviews).
Would the embedded piece grow without bound?	No -> **EMB** (A CartItem **is** a Product).	Yes -> REF (A Product can have millions of Reviews).
Are the pieces of info written at different times in a write-heavy workload?	No -> **EMB** (A CartItem **is** a Product).	Yes -> REF (A Review can be updated without updating a product).
For the child side of the relationship (author), can the pieces exist by themselves without a parent?	Yes -> REF (A Product can exist without a CartItem).	No -> **EMB** (A Review can’t exist without the Product).
SCORE	EMB (10) REF(1) **REF** (**OVRD**) Reference product_id of **Product** in each **CartItem**.	EMB (5) REF(6) **REF**