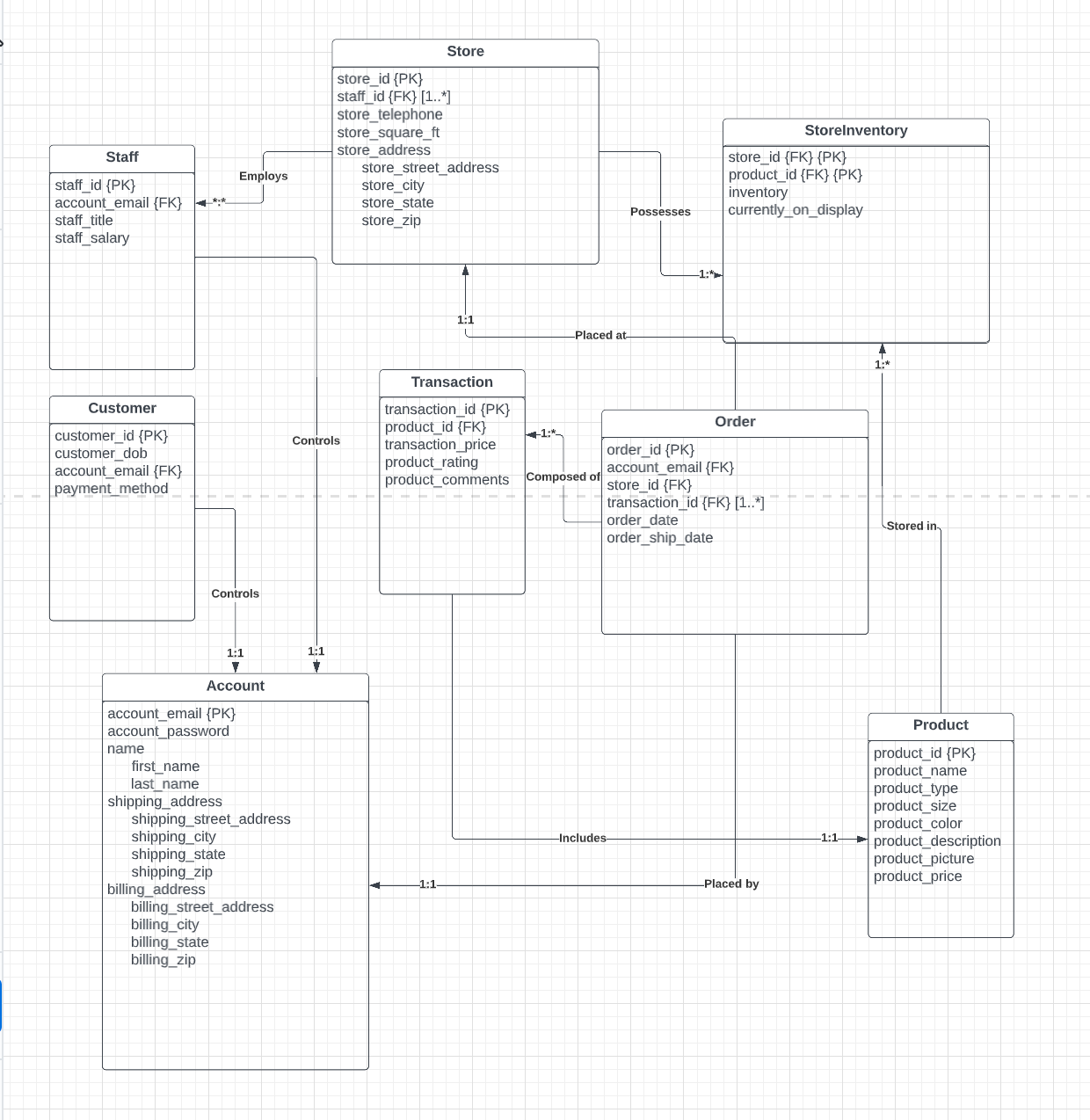
**UML for LL Bean**



**Convert UML to Relations**

Key

* Primary Key is underlined
* Foreign Key is *italic*

Relations

**Customer**(customer\_id, customer\_dob, *account\_email*, payment\_method)

* This relation identifies attributes for the customer.
* There is 1 tuple for every account that has placed an order (in other words, every customer).
* The attribute *account\_email* is a foreign key to the Account relation, which stores more information about each customer.
* degree: 4

**Staff**(staff\_id, *account\_email*, *staff\_title*, staff\_salary)

* This relation identifies attributes for the staff members.
* There is 1 tuple for each LL Bean staff member.
* The attribute *account\_email* is a foreign key to the Account relation, which stores more information about each staff member.
* staff\_title is a foreign key to the Staff\_Title relation, which restricts the domain.
* degree: 4

**Staff\_Title**(staff\_title)

* This relation identifies all possible titles for staff
* This relation serves to restrict the domain of staff\_title to allowable values (manager, sales associate, etc)
* degree: 1

**Store**(store\_id, store\_telephone, store\_square\_ft, store\_street\_address, store\_city, store\_state, store\_zip)

* This relation identifies attributes for both physical stores and the online store.
* This relation does not include the multi-value attribute staff\_id.
* To keep track of online availability, there will be a store named “online,” and attributes such as store\_telephone, store\_square\_ft, and store\_state will be null.
* degree: 7

**StoreInventory**(*store\_id, product\_id*, inventory, currently\_on\_display)

* This relation associates products with stores.
* There is 1 tuple for every instance of a product occurring at an individual store.
* The inventory attribute indicates how much of that particular product is currently available at that particular store. A product sold at a store brings the inventory down by one, while a shipment of a product to a store increases the inventory.
* The currently\_on\_display attribute indicates whether or not the store is currently showcasing the product.
* store\_id and product\_id are composite primary keys.
* degree: 4

**Product**(product\_id, product\_name, *product\_type*, *product\_size*, product\_color, product\_description, product\_picture, product\_price)

* This relation identifies attributes for the products.
* There is 1 tuple for each unique product sold by LL Bean.
* product\_size is a foreign key to the Product\_Size relation, which restricts the domain.
* product\_type is a foreign key to the Product\_Type relation, which restricts the domain.
* degree: 8

**Product\_Size**(product\_size)

* This relation identifies all possible sizes for products
* This relation serves to restrict the domain of product\_size to allowable values (such as S, M, L, and XL)
* degree: 1

**Product\_Type**(product\_type)

* This relation identifies all possible types for products
* This relation serves to restrict the domain of product\_type to allowable values
* degree: 1

**Account**(account\_email, account\_password, first\_name, last\_name, shipping\_street\_address, shipping\_city, shipping\_state, shipping\_zip, billing\_street\_address, billing\_city, billing\_state, billing\_zip)

* This relation identifies attributes for an account holder, which includes both customers and staff members.
* An account holder can also include people who created an account but never made a purchase or worked at a store, and therefore are neither a customer nor a staff member.
* The primary key is the account\_email, which means that each email can only correspond to one account.
* For customers, the account stores their shipping address and billing address. For employees, a shipping address is required as well (to send holiday gifts) and a billing address is needed for paychecks.
* One central account simplifies tasks for employees, as they only need one login (even when ordering products for themselves).
* One central account also makes it easier to identify possible employee fraud, since all purchases will be made using their company account
* degree: 12

**Order**(order\_id, order\_date, order\_ship\_date, *account\_email*, *store\_id*)

* This relation identifies attributes for orders.
* An order is a collection of individual product purchases which share an order\_date, order\_ship\_date, and are placed by the same account\_email
* An order is composed of at least one transaction. This relation does not include the multi-value attribute transaction\_id.
* degree: 5

**Transaction**(transaction\_id, *product\_id*, transaction\_price, product\_rating, product\_comments)

* This relation identifies attributes for transactions.
* There is 1 tuple for each product sold, since transactions correspond to the purchase of an individual product.
* A customer can assign a product\_rating and provide a product description using product\_comments.
* degree: 5

**StoreStaff**(*staff\_id*, *store\_id*)

* This relation associates staff and stores.
* staff\_id and store\_id are composite primary keys.
* A store employs at least one staff member. There is no restriction against staff working at multiple stores.
* There is 1 tuple for every instance of a staff member working at a store.
* degree: 2

**OrderTransaction**(*order\_id*, *transaction\_id*)

* This relation associates orders and transactions.
* order\_id and transaction\_id are composite primary keys.
* A transaction can only be a part of a single order. An order must have at least one transaction.
* degree: 2

**Domain**

* customer\_dob, order\_date, order\_ship\_date: Datetime value
* store\_telephone: 10 digit number
* store\_state: 50 state codes plus can be NULL for online
* store\_zip, shipping\_zip, billing\_zip: valid zip codes
* currently\_on\_display: true/false
* account\_email: email value
* shipping\_state, billing\_state: 50 state codes
* product\_rating: integer between 1 and 5

1. Identify the 12 month purchase history for Jesper Bratt. Display the customer name, product, size, price, date of purchase.

last12 <- 𝛿order\_date > “2021-10-16 00:00:00” (Order)

bratt <- 𝛿first\_name = “Jesper” ⋀ last\_name = “Bratt” (Account)

brattlast12 <- 𝛿bratt.account\_email = last12.account\_email (bratt x last12)

bratt12ot <- 𝛿brattlast12.order\_id = OrderTransaction.order\_id (brattlast12 x OrderTransaction)

bratt12t <- 𝛿bratt12ot.transaction\_id = Transaction.transaction\_id (bratt12ot x Transaction)

bratt12p <- 𝛿bratt12t.product\_id = Product.product\_id (bratt12t x Product)

Answer <- Πfirst\_name,last\_name,product\_name,product\_size,transaction\_price,order\_date(bratt12p)

1. Identify polo shirts, size medium available now at the Jeffersonville, Ohio location. Display the product name, product code, available colors and price.

jo <- 𝛿store\_city = “Jeffersonville” ⋀ store\_state = “OH” (Store)

joinven <- 𝛿jo.store\_id = StoreInventory.store\_id (jo x StoreInventory)

joavail <- 𝛿inventory > 0 ⋀ currently\_on\_display = TRUE (joinven)

joprod <- 𝛿joavail.product\_id = Product.product\_id (joavail x Product)

jomedpolo <- 𝛿product\_type = “Polo Shirt” ⋀ product\_size = “M” (joprod)

Answer <- Πproduct\_name,product\_id,product\_color,product\_price (jomedpolo)

1. Identify when the recent order for Jesper Bratt will be shipped. Display the products ordered, price and ship date.

bratt <- 𝛿first\_name = “Jesper” ⋀ last\_name = “Bratt” (Account)

brattorders <- 𝛿bratt.account\_email = Order.account\_email(bratt x Order)

brattordersclone <- 𝛿bratt.account\_email = Order.account\_email(bratt x Order)

brattoldorders <- 𝛿brattorders.order\_date < brattordersclone.order\_date(brattorders x brattordersclone)

brattrecentorder <- Πorder\_date(brattorders) - Πbrattorders.order\_date(brattoldorders)

a <- 𝛿brattrecentorder.order\_id = OrderTransaction.order\_id(brattrecentorder x OrderTransaction)

brattrecenttrans <- 𝛿a.transaction\_id = Transaction.transaction\_id(a x Transaction)

brattrecentprods <- 𝛿brattrecenttrans.product\_id = Product.product\_id(brattrecenttrans x Product)

Answer <- Πproduct\_name,transaction\_price,order\_ship\_date(brattrecentprods)

1. Identify products with no inventory offered at the web store. Display the product name, product code, size and color.

online <- 𝛿store\_state = NULL(Store)

onlineinven <- 𝛿online.store\_id = StoreInventory.store\_id(online x StoreInventory)

onlineavail <- 𝛿inventory > 0 ⋀ currently\_on\_display = TRUE (onlineinven)

onlinenotavail <- Πproduct\_id(Product) - Πproduct\_id(onlineavail)

onlinenotavailp <- 𝛿onlinenotavail.product\_id =Product.product\_id(onlinenotavail x Product)

Answer <- Πproduct\_name,product\_id,product\_size,product\_color(onlinenotavailp)

1. Identify button-down shirts not sold in the last month at the web store. Display the product name, product code, size, color and price.

online <- 𝛿store\_state = NULL(Store)

onlineorders <- 𝛿online.store\_id = Order.store\_id(online x Order)

monthorders <- 𝛿order\_date>”2022-09-16 00:00:00” (onlineorders)

monthordertrans <- 𝛿monthorders.order\_id = OrderTransaction.order\_id(monthorders x OrderTransaction)

monthtrans <- 𝛿monthordertrans.transaction\_id = Transaction.transaction\_id(monthordertrans x Transaction)

monthprods <- 𝛿monthtrans.product\_id = Product.product\_id(monthtrans x Product)

monthbdshirts <- 𝛿product\_type = “Button-down Shirt”(monthprods)

allbdshirts <- 𝛿product\_type = “Button-down Shirt” (Product)

bdshirtsnotsold <- Πproduct\_id(allbdshirts) - Πproduct\_id(monthbdshirts)

bdshirtsnotsoldinfo <- 𝛿bdshirtsnotsold.product\_id =Product.product\_id(bdshirtsnotsold x Product)

Answer <- Πproduct\_name,product\_id,product\_size,product\_color,product\_price(bdshirtsnotsoldinfo)

1. Identify customers without a purchase in the last year. Display the customer name and email.

yearorder <- 𝛿order\_date>”2021-10-16 00:00:00” (Order)

customers <- 𝛿Customer.account\_email = Account.account\_email(Customer x Account)

yearnoorder <- Πaccount\_email(customers) - Πaccount\_email(yearorder)

yearnoorderinfo <- 𝛿yearnoorder.account\_email =Account.account\_email(yearnoorder x Account)

Answer <- Πfirst\_name,last\_name,account\_email(yearnoorderinfo)

1. Identify sales by state since 10/16/2021. Display 3 columns: state, number of sales and total dollar sales.

a <- 𝛿Order.order\_id = OrderTransaction.order\_id (Order x OrderTransaction)

ordertrans <- 𝛿a.transaction\_id = Transaction.transaction\_id(a x Transaction)

storeordertrans <- 𝛿ordertrans.store\_id = Store.store\_id(ordertrans x Store)

storeordertransyear <- 𝛿order\_date > “2021-10-16 00:00:00”(storeordertrans)

ρanswer(State, Number of Sales, Total Dollar Sales) store\_state ℑcount transaction\_id, sum transaction\_price (storeordertransyear)

1. Identify the total number of active New York customers. Display the number.

(assume “active” means at least one purchase in the last year, and a customer is considered to be a “New York” customer if they are billed in NY)

yearorder <- 𝛿order\_date>”2021-10-16 00:00:00” (Order)

nyaccounts <- 𝛿billing\_state = “NY” (Account)

nycustomers <- 𝛿Customer.account\_email = nyaccounts.account\_email(Customer x nyaccounts)

nyactive <- 𝛿nycustomers.account\_email = yearorder.account\_email (nycustomers x yearorder)

Answer <- ℑcount account\_email(nyactive)

1. Identify sales by product since 10/16/2021. Display 3 columns: Product name, number of products sold and total dollar amount.

yearorder <- 𝛿order\_date>”2021-10-16 00:00:00” (Order)

yearordertrans <- 𝛿yearorder.order\_id = OrderTransaction.order\_id (yearorder x OrderTransaction)

yeartrans <- 𝛿yearordertrans.transaction\_id = Transaction.transaction\_id(yearordertrans x Transaction)

yearprod <- 𝛿yeartrans.product\_id = Product.product\_id(yeartrans x Product)

ρanswer(Product Name, Number of Product Sold, Total Dollar Amount) product\_name ℑcount transaction\_id, sum transaction\_price(yearprod)

1. Identify winter hats with low customer ratings in the last 6 months. Display the

product, size, color, price, rating, and the customer description.

(assume low customer rating means 1 star or 2 stars out of 5 stars)

allhats <- 𝛿product\_type = “Winter Hat” (Product)

allhatssold <- 𝛿Transaction.product\_id = allhats.product\_id (Transaction x allhats)

allhatsot <- 𝛿allhatssold.transaction\_id = OrderTransaction.transaction\_id(allhatssold x OrderTransaction)

allhatso <- 𝛿allhatsot.order\_id = Order.order\_id(allhatsot x Order)

allhatssixmonths <- 𝛿order\_date > “2022-04-16 00:00:00”(allhatso)

badreviewhats <- 𝛿product\_rating < 3(allhatssixmonths)

Answer <- Πproduct\_name,product\_size,product\_color,product\_price,product\_rating,product\_comments(badreviewhats)

1. Identify employees who are not customers. Display the employee name, employee ID and address.

customers <- 𝛿Customer.account\_email = Account.account\_email(Customer x Account)

staffnotcustomers <- Πaccount\_email(Staff) - Πaccount\_email(customers)

a <- 𝛿Staff.account\_email = staffnotcustomers.account\_email(Staff x staffnotcustomers)

b <- 𝛿Account.account\_email = a.account\_email(Account x a)

Answer <- Πfirst\_name,last\_name,staff\_id,billing\_street\_address,billing\_city,billing\_state,billing\_zip(b)

1. Identify products sold 9/6/2022 at the Jeffersonville, Ohio location. Display the customer name, product purchased, price and address.

jo <- 𝛿store\_city = “Jeffersonville” ⋀ store\_state = “OH” (Store)

jeforder <- 𝛿jo.store\_id = Order.store\_id(jo x Order)

jeforderss <- 𝛿order\_date = “2022-09-06 00:00:00”(jeforder)

a <- 𝛿jeforderss.account\_email = Account.account\_email(jeforderss x Account)

b <- 𝛿a.order\_id = OrderTransaction.order\_id(a x OrderTransaction)

c <- 𝛿b.transaction\_id = Transaction.transaction\_id(b x Transaction)

d <- 𝛿c.product\_id = Product.product\_id(c x Product)

Answer<-Πfirst\_name,last\_name,product\_name,transaction\_price,shipping\_street\_address,shipping\_city,shipping\_state,shipping\_zip)(d)

1. Identify customers who live in warm weather states (FL, CA, AZ, GA, NV, NM) and also recently purchased winter jackets. Display the customer name, product, date of purchase and price

(assume recently purchased means last month)

monthorders <- 𝛿order\_date>”2022-09-16 00:00:00” (Order)

monthordertrans <- 𝛿monthorders.order\_id = OrderTransaction.order\_id(monthorders x OrderTransaction)

monthtrans <- 𝛿monthordertrans.transaction\_id = Transaction.transaction\_id(monthordertrans x Transaction)

monthprods <- 𝛿monthtrans.product\_id = Product.product\_id(monthtrans x Product)

monthjackets <- 𝛿product\_type = “Winter Jacket”(monthprods)

customers <- 𝛿Customer.account\_email = Account.account\_email(Customer x Account)

warmweatheracc <- 𝛿billing\_state = “FL” ∨ billing\_state = “CA” ∨ billing\_state = “AZ” ∨ billing\_state = “GA” ∨ billing\_state = “NV” ∨ billing\_state = “NM”(customers)

wwjackets <- 𝛿monthjackets.account\_email = warmweatheracc.account\_email(monthjackets x warmweatheracc)

Answer <- Πfirst\_name,last\_name,product\_name,order\_date,transaction\_price(wwjackets)