CPE 366
Team: Pinky and the Brain
Andrew Guenther
Halli Meth
Mitchell Rosen
Haikal Saliba

Lab 4

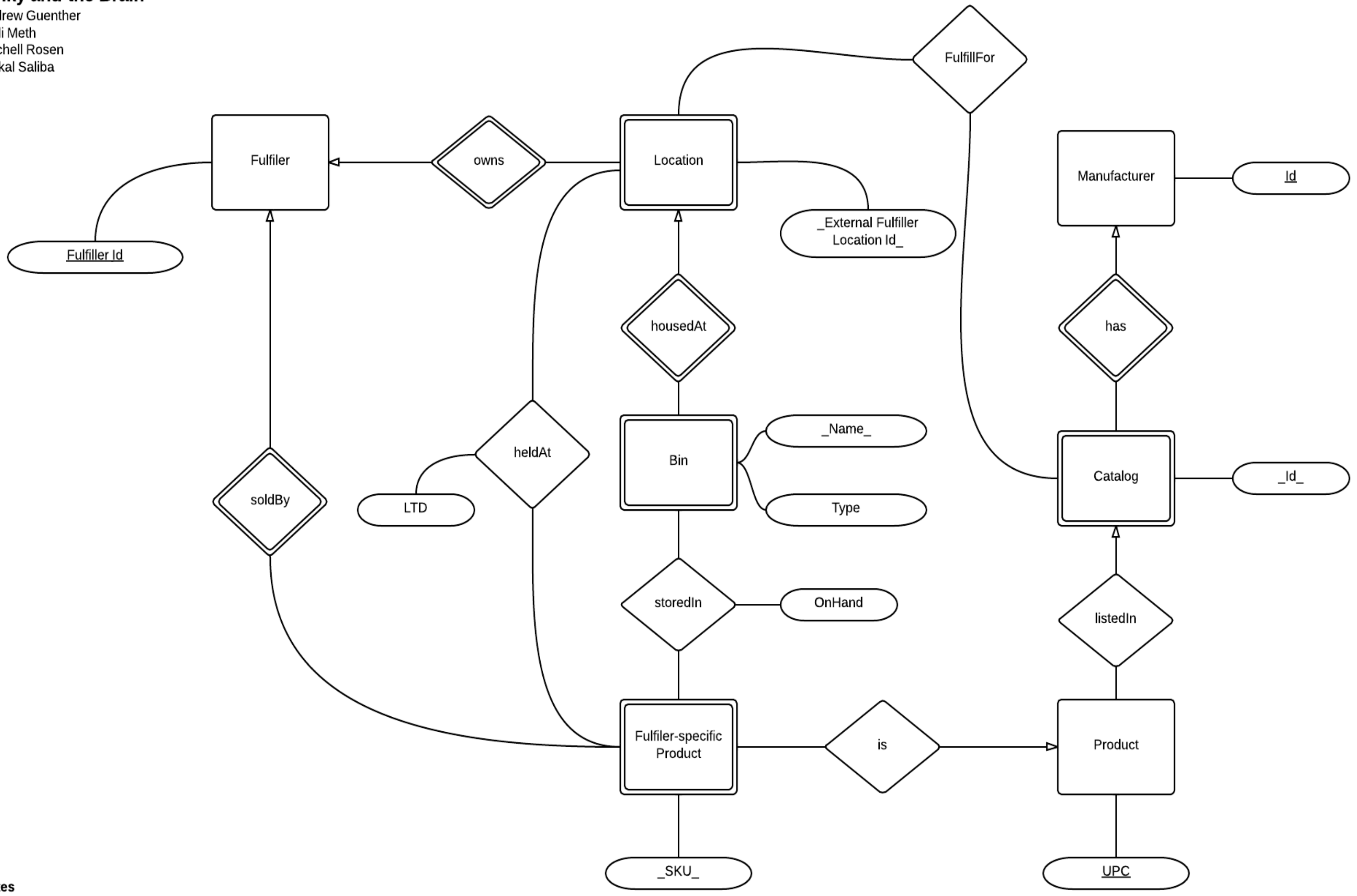
Lab 3 Revision:

Database Constraints:

1. onHand >= numAllocated
2. If a FulfillerSpecificProduct is *HeldAt* a Location, then that Location has to *FulfillFor* the Catalog in which the Product that represents a FulfilerSpecificProduct is *listedIn*
3. If a FulfillerSpecificProduct is *StoredIn* a Bin, there must be at least one Location in which the Bin is *HousedAt* and the FulfillerSpecificProduct is *HeldAt*
4. There can be only one LTD per FulfillerSpecificProduct at each Location
5. The Fulfiller that *Sells* a FullfillerSpecificProduct must be the same Fulfiller that *Owns* the Location where the FulfillerSpecificProduct is *HeldAt*

Pinky and the Brain

Andrew Guenther
Halli Meth
Mitchell Rosen
Haikal Saliba



Notes
Stock information can be aggregated to achieve inventory totals for each product

<name> is a discriminator

$\rightarrow$ referential integrity constraint

Relational Tables:

```
Fulfiller
id STRING PRIMARY KEY

Manufacturer
id STRING PRIMARY KEY

Location
ext_ful_loc_id STRING PRIMARY KEY
int_ful_loc_id INT
fulfiller_id INT PRIMARY KEY
name STRING
type STRING  -- “description” in CSV
latitude DECIMAL
longitude DECIMAL
status STRING
default_safety_stock INT
```

Attribute fulfiller_id is a foreign key referencing table Fulfiller

```
Catalog
id STRING PRIMARY KEY
manufacturer_id STRING PRIMARY KEY
```

Attribute manufacturer_id is a foreign key referencing table Manufacturer

```
FulfillFor
fulfiller_id STRING PRIMARY KEY
ext_ful_loc_id STRING PRIMARY KEY
catalog_id STRING PRIMARY KEY
manufacturer_id STRING PRIMARY KEY
```

Attribute fulfiller_id, ext_ful_loc_id *is a foreign key referencing table Location*
Attributes manufactuer_id, catalog_id is a foreign key referencing table Catalog

```
Product
upc VARCHAR2(12) PRIMARY KEY
catalog_id STRING
manufacturer_id STRING
name STRING
```

Attributes catalog_id, manufacturer_id are foreign keys referencing table Catalog

```
Bin
name STRING PRIMARY KEY
fulfiller_id STRING PRIMARY KEY
ext_ful_loc_id STRING PRIMARY KEY
type STRING
status STRING
```

Attribute ext_ful_loc_id, fulfiller_id is a foreign key referencing table Location

<u>FulfillerSpecificProduct</u>
sku STRING PRIMARY KEY
fulfiller_id STRING PRIMARY KEY
upc STRING

Attribute fulfiller_id is a foreign key referencing table Fulfiller
Attribute UPC is a foreign key referencing table Product
Attributes fulfiller_id and upc are a candidate key

<u>HeldAt</u>
fulfiller_id STRING PRIMARY KEY
ext_ful_loc_id STRING PRIMARY KEY
sku STRING PRIMARY KEY
ltd FLOAT
safety_stock INT

Attributes SKU is a foreign key referencing table FulfillerSpecificProduct
Attributes ext_ful_loc_id, fulfiller_id is a foreign key referencing Location

<u>StoredIn</u>
sku STRING PRIMARY KEY
fulfiller_id STRING PRIMARY KEY
bin_name STRING PRIMARY KEY
ext_ful_loc_id STRING PRIMARY KEY
on_hand INT
num_allocated INT DEFAULT 0

Attribute sku is a foreign key referencing FulfillerSpecificProduct
Attributes fulfiller_id, bin_name and ext_ful_loc_id are foreign keys referencing Bin

Lab 4 Document:

Formalization of Use Cases:

<u>Define Store Locations</u>

1. Identification:

Create store locations for each of the entries in the csv file. Every row corresponds to a single fulfillment store location. The data for each row is used to create the store location with a single bin of “default” using the create store location API.

2. WSDL API calls:

```
<wsdl:operation name="createFulfillmentLocation">
   <wsdl:input name="createFulfillmentLocationRequest"
               message="impl:createFulfillmentLocationRequest"/>
   <wsdl:output name="createFulfillmentLocationResponse"
               message="impl:createFulfillmentLocationResponse"/>
</wsdl:operation>
```

3. Input data:

Input

```
<wsdl:message name="createFulfillmentLocationRequest">
   <wsdl:part name="AuthenticationHeader" element="impl:AuthenticationHeader"/>
   <wsdl:part name="parameters" element="impl:createFulfillmentLocation"/>
</wsdl:message>
```

Parameters

```
<element name="createFulfillmentLocation">
   <complexType>
     <sequence>
       <element name="request" type="impl:FulfillmentLocation"/>
   </sequence>
   </complexType>
</element>

<complexType name="FulfillmentLocation">
   <sequence>
      <element name="FulfillerID" type="xsd:positiveInteger"/>
      <element name="ManufacturerLocationID" type="xsd:positiveInteger" nillable="true"/>
      <element name="RetailerLocationID" type="xsd:positiveInteger" nillable="true"/>
      <element name="ExternalLocationID" type="xsd:string" nillable="true"/>
      <element name="LocationName" type="xsd:string" nillable="true"/>
      <element name="TypeID" type="xsd:positiveInteger"/>
      <element name="Latitude" type="xsd:double"/>
      <element name="Longitude" type="xsd:double"/>
      <element name="Status">
         <simpleType>
            <restriction base="xsd:int">
               <enumeration value="1">
                  <annotation>
                     <documentation>Location Active</documentation>
                  </annotation>
               </enumeration>
               <enumeration value="2">
                  <annotation>
```

```
                    <documentation>Location Not Active</documentation>
                  </annotation>
                </enumeration>
              </restriction>
            </simpleType>
          </element>
          <element name="CountryCode" type="xsd:string" nillable="true"/>
        </sequence>
      </complexType>
```

4. Output data expected:

Output

```
<wsdl:message name="createFulfillmentLocationResponse">
   <wsdl:part name="parameters" element="impl:createFulfillmentLocationResponse"/>
</wsdl:message>
```

Parameter

```
<element name="createFulfillmentLocationResponse">
   <complexType>
     <sequence>
       <element name="createFulfillmentLocationReturn" type="xsd:int"/>
     </sequence>
   </complexType>
</element>
```

5. SQL statements:

```
INSERT IGNORE INTO Fulfiller(id) VALUES(FulfillerID);

INSERT IGNORE INTO Manufacturer(id) VALUES(ManufacturerLocationID);

INSERT INTO Location(ext_ful_loc_id, int_ful_loc_id, fulfiller_id, name, type,
latitude, longitude, status) VALUES(ExternalLocationID, RetailerLocationId,
FulfillerId, LocationName, TypeID, Latitude, Longitude, Status);
```

Define Store Bins

1. Identification:

Create bins for each of the entries in the csv file. Every row corresponds to a single bin, linked by name to an external fulfiller location. The data for each row is used to create the bin using the create store location bin API.

2. WSDL API calls:

```
<wsdl:operation name="createBin">
       <wsdl:input name="createBinRequest" message="impl:createBinRequest"/>
   <wsdl:output name="createBinResponse" message="impl:createBinResponse"/>
</wsdl:operation>
```

3. Input data:

Input

```
<wsdl:message name="createBinRequest">
       <wsdl:part name="AuthenticationHeader" element="impl:AuthenticationHeader"/>
   <wsdl:part name="parameters" element="impl:createBin"/>
</wsdl:message>
```

```
Parameters
<element name="createBin">
       <complexType>
      <sequence>
         <element name="request" type="impl:Bin"/>
      </sequence>
   </complexType>
</element>

<complexType name="Bin">
   <sequence>
      <element name="BinID" type="xsd:positiveInteger" nillable="true"/>
      <element name="FulfillerLocationID" type="xsd:positiveInteger"/>
      <element name="BinTypeID" type="xsd:positiveInteger"/>
      <element name="BinStatusID" type="xsd:positiveInteger"/>
      <element name="Name" type="xsd:string" nillable="true"/>
   </sequence>
</complexType>
```

4. Output data expected:

```
Output
<wsdl:message name="createBinResponse">
   <wsdl:part name="parameters" element="impl:createBinResponse"/>
</wsdl:message>

Parameter
<element name="createBinResponse">
   <complexType>
      <sequence>
         <element name="createBinReturn" type="xsd:positiveInteger"/>
          </sequence>
   </complexType>
</element>
```

5. SQL statements:

```
INSERT INTO Bin(name, fulfiller_id, ext_ful_loc_id, type, status) VALUES(Name,
FulfillerID, FulfillerLocationID, BinTypeID, BinStatusID)
```

Bulk Inventory Update

1. Identification:

Create inventory records for each of the entries in the csv file. Every row corresponds to a single inventory record, linked to a bin at a location. The data for each row is used to create the inventory record using the inventory update API. If an inventory record does not exist, the record is created with an allocated count of zero. Otherwise, the on hand quantity defined in the inventory record in the file is passed to the inventory update API.

2. WSDL API calls:

```
<wsdl:operation name="refreshInventory">
   <wsdl:input name="refreshInventoryRequest2" message="impl:RefreshInventorySoapIn"/>
   <wsdl:output name="refreshInventoryResponse2" message="impl:RefreshInventorySoapOut"/>
</wsdl:operation>
```

3. Input data:

```
Input
<wsdl:message name="RefreshInventorySoapIn">
   <wsdl:part name="parameter" element="impl:AuthenticationHeader"/>
   <wsdl:part name="parameters" element="impl:RefreshRequest"/>
</wsdl:message>
Parameters
<complexType name="RefreshRequest">
   <sequence>
      <element name="LocationName" type="xsd:string"/>
      <element name="Items" type="impl:ArrayOf_impl_RefreshItem" nillable="true"/>
   </sequence>
</complexType>

<complexType name="ArrayOf_impl_RefreshItem">
   <sequence>
      <element name="items" type="impl:RefreshItem" minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
</complexType>

<complexType name="RefreshItem">
   <sequence>
      <element name="PartNumber" type="xsd:string" nillable="true"/>
      <element name="UPC" type="xsd:string" nillable="true"/>
      <element name="LocationUPC" type="xsd:string" nillable="true"/>
      <element name="BinID" type="xsd:int"/>
      <element name="Quantity" type="xsd:int"/>
      <element name="LTD" type="xsd:double"/>
      <element name="Floor" type="xsd:int"/>
      <element name="SafetyStock" type="xsd:int"/>
   </sequence>
</complexType>
```

4. Output data expected:

```
Output
<wsdl:message name="RefreshInventorySoapOut">
   <wsdl:part name="parameter" element="impl:RefreshResponse"/>
</wsdl:message>

Parameter
<element name="RefreshResponse" type="xsd:string"/>
```

5. SQL statements:

```
INSERT INTO FulfillerSpecificProduct(sku, fulfiller_id, upc)
VALUES(PartNumber, FulfillerId, UPC);

INSERT INTO HeldAt(fulfiller_id, int_ful_location_id, sku, ltd, safety_stock)
VALUES(FulfillerId, LocationName, PartNumber, LTD, SafetyStock);

INSERT REPLACE StoredIn(sku, fulfiller_id, bin_name, ext_ful_loc_id, on_hand)
VALUES(PartNumber, FulfillerId, BinId, LocationName, Quantity);
```

Trickle Inventory Update

1. Identification:

Update the inventory record for each of the entries in the csv file. Every row corresponds to a single inventory record. Data from each record is sent through the trickle update API with the delta in the on hand quantity.

2. WSDL API calls:

```
<wsdl:operation name="adjustInventory">
   <wsdl:input message="impl:AdjustInventorySoapIn"/>
   <wsdl:output message="impl:AdjustInventorySoapOut"/>
</wsdl:operation>
```

3. Input data:

```
Input
<wsdl:message name="AdjustInventorySoapIn">
   <wsdl:part name="AuthenticationHeader" element="impl:AuthenticationHeader"/>
   <wsdl:part name="parameter" element="impl:AdjustRequest"/>
</wsdl:message>

Parameters
<element name="AdjustRequest">
   <complexType>
      <sequence>
         <element name="LocationName" type="xsd:string"/>
             <element name="Items" type="impl:ArrayOf_impl_AdjustItem" nillable="true"/>
      </sequence>
   </complexType>
</element>

<complexType name="ArrayOf_impl_AdjustItem">
   <sequence>
      <element name="items" type="impl:AdjustItem" minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
</complexType>

<complexType name="AdjustItem">
   <sequence>
      <element name="PartNumber" type="xsd:string" nillable="true"/>
      <element name="UPC" type="xsd:string" nillable="true"/>
      <element name="LocationUPC" type="xsd:string" nillable="true"/>
      <element name="BinID" type="xsd:int"/>
      <element name="Quantity" type="xsd:int"/>
   </sequence>
</complexType>
```

4. Output data expected:

```
Output
<wsdl:message name="AdjustInventorySoapOut">
   <wsdl:part name="parameter" element="impl:AdjustResponse"/>
</wsdl:message>

Parameter
```

```
<element name="AdjustResponse" type="xsd:string"/>
```

5. SQL statements:

```
UPDATE StoredIn(sku, fulfiller_id, bin_name, ext_ful_loc_id, on_hand)
   SET on_hand = on_hand + Quantity
   WHERE sku = PartNumber AND fulfiller_id = FulfillerId AND
         bin_name = BinId AND ext_ful_loc_id = LocationName;
```

Get Inventory

1. Identification:

Return detailed inventory information based on a number of parameters.

2. WSDL API calls:

```
<wsdl:operation name="getInventory">
   <wsdl:input name="getInventoryRequest" message="impl:getInventoryRequest"/>
   <wsdl:output name="getInventoryResponse" message="impl:getInventoryResponse"/>
</wsdl:operation>
```

3. Input data:

Input

```
<wsdl:message name="getInventoryRequest">
   <wsdl:part name="AuthenticationHeader" element="impl:AuthenticationHeader"/>
   <wsdl:part name="parameters" element="impl:getInventory"/>
</wsdl:message>
```

Parameters

```
<element name="getInventory">
   <complexType>
      <sequence>
         <element name="request" type="impl:InventoryRequest"/>
      </sequence>
   </complexType>
</element>

<complexType name="InventoryRequest">
   <sequence>
      <element name="Catalog" type="impl:ManufacturerCatalog" nillable="false"/>
      <element name="Quantities" type="impl:ArrayOf_impl_ItemQuantity" nillable="false"/>
      <element name="LocationNames" type="impl:ArrayOfLocationNames" nillable="true"/>
      <element name="Location" type="impl:RequestLocation" nillable="true"/>
      <element name="Type" type="impl:InventoryRequestType" nillable="false"
               minOccurs="1"/>
      <element name="Limit" type="xsd:int" default="10000"/>
      <element name="IgnoreSafetyStock" type="xsd:boolean" default="false"
               nillable="true"/>
      <element name="IncludeNegativeInventory" type="xsd:boolean" default="false"
               nillable="true"/>
      <element name="OrderByLTD" type="boolean"/>
   </sequence>
</complexType>

<complexType name="ManufacturerCatalog">
```

```
   <sequence>
      <element name="ManufacturerID" type="xsd:positiveInteger"/>
      <element name="CatalogID" type="xsd:positiveInteger"/>
   </sequence>
</complexType>

<complexType name="ArrayOf_impl_ItemQuantity">
   <sequence>
      <element name="items" type="impl:ItemQuantity" minOccurs="1"
               maxOccurs="unbounded"/>
   </sequence>
</complexType>

<complexType name="ItemQuantity">
   <sequence>
      <element name="PartNumber" type="xsd:string" nillable="true"/>
      <element name="UPC" type="xsd:string" nillable="true"/>
      <element name="LocationUPC" type="xsd:string" nillable="true"/>
      <element name="Quantity" type="xsd:int"/>
   </sequence>
</complexType>

<complexType name="ArrayOfLocationNames">
   <sequence>
      <element name="LocationNames" type="xsd:string" minOccurs="0"
               maxOccurs="unbounded"/>
   </sequence>
</complexType>

<complexType name="RequestLocation">
   <sequence>
      <element name="Unit" type="xsd:string" default="MILES" nillable="true"/>
      <element name="Radius" type="xsd:positiveInteger" nillable="true"/>
      <element name="PostalCode" type="xsd:string" nillable="true"/>
      <element name="Latitude" type="xsd:double" nillable="true"/>
      <element name="Longitude" type="xsd:double" nillable="true"/>
      <element name="CountryCode" type="xsd:string" nillable="true"/>
   </sequence>
</complexType>

<simpleType name="InventoryRequestType">
   <annotation>
      <documentation/>
   </annotation>
   <restriction base="xsd:string">
      <enumeration value="ALL"/>
      <enumeration value="PARTIAL"/>
      <enumeration value="ANY"/>
      <enumeration value="ALL_STORES"/>
   </restriction>
</simpleType>
```

4. Output data expected:

Output
```
<wsdl:message name="getInventoryResponse">
   <wsdl:part name="parameters" element="impl:getInventoryResponse"/>
</wsdl:message>
```

Parameters
```
<element name="getInventoryResponse">
   <complexType>
      <sequence>
         <element name="getInventoryReturn" type="impl:InventoryResponse"
                  maxOccurs="unbounded"/>
      </sequence>
   </complexType>
</element>

<complexType name="InventoryResponse">
   <sequence>
      <element name="LocationName" type="xsd:string" nillable="true"/>
      <element name="CatalogID" type="xsd:int"/>
      <element name="ManufacturerID" type="xsd:int"/>
      <element name="OnHand" type="xsd:int" nillable="true"/>
      <element name="Available" type="xsd:int" nillable="true"/>
      <element name="PartNumber" type="xsd:string" nillable="true"/>
      <element name="UPC" type="xsd:string" nillable="true"/>
      <element name="LocationUPC" type="xsd:string" nillable="true"/>
      <element name="LTD" type="xsd:double" nillable="true"/>
      <element name="Floor" type="xsd:int" nillable="true"/>
      <element name="SafetyStock" type="xsd:int" nillable="true"/>
      <element name="STHEnabled" type="xsd:boolean" nillable="true"/>
      <element name="RestockEnabled" type="xsd:boolean" nillable="true"/>
      <element name="PickupEnabled" type="xsd:boolean" nillable="true"/>
      <element name="CountryCode" type="xsd:string" nillable="true"/>
      <element name="Distance" type="xsd:double" nillable="true"/>
   </sequence>
</complexType>
```

5. SQL statements:
```
SELECT ValidLocation.ext_ful_loc_id, ValidLocation.catalog_id,
       ValidLocation.manufacturer_id, si.on_hand, si.on_hand -
       si.num_allocated, fp.sku, fp.upc, '', ha.ltd, NULL,
       sa.safety_stock, NULL, NULL, NULL, NULL, NULL
   FROM
      (SELECT ext_ful_loc_id, fulfiller_id
         FROM Locations l, FulfillFor ff
         WHERE ff.fullier_id = l.fullier_id AND ff.ext_ful_loc_id =
               l.ext_ful_loc_id AND
               ff. catalog_id = CatalogId AND
               ff.manufacturer_id = ManufacturerID AND
               l.ext_ful_loc_id = LocationName AND l.fulfiller_id =
               FulfillerId
      ) AS ValidLocation
         INNER JOIN Bin b ON(
            b.ext_ful_loc_id = ValidLocation.ext_ful_loc_id AND
            b.fulfiller_id = ValidLocation.fulfiller_id
         )
         INNER JOIN StoredIn si ON(
            si.ext_ful_loc_id = b.ext_ful_loc_id AND
            si.fulfiller_id = b.fulfiller_id AND
            si.bin_name = b.name
         )
         INNER JOIN HeldAt ha ON(
            ha.sku = si.sku AND
            ha.ext_ful_loc_id = si.ext_ful_loc_id AND
            ha.fulfiller_id = si.fulfiller_id
         )
         INNER JOIN FulfillerSpecificProduct fp ON(
            fp.fulfiller_id = ha.fulfiller_id AND
            fp.sku = ha.sku
         )
   WHERE fp.sku = PartNumber AND fp.upc = PC AND
         si.on_hand [- si.num_allocated] >= quantity [- ha.safety_stock]
   [ORDER BY ha.ltd]

-- There will be a separate query that retrieves all location names within a
   certain radius
-- [] added or removed by the flags
```

Allocate Inventory

1. Identification:

Prior to this call, a Get Inventory call is made to determine what locations and bins are available. Allocate Inventory is called with a list of SKU/UPCs, and invokes the allocation API on each SKU/UPC.

2. WSDL API calls:

```
<wsdl:operation name="allocateInventory">
   <wsdl:input name="allocateInventoryRequest" message="impl:allocateInventoryRequest"/>
   <wsdl:output name="allocateInventoryResponse"
                message="impl:allocateInventoryResponse"/>
</wsdl:operation>
```

3. Input data:

```
<wsdl:message name="allocateInventoryRequest">
   <wsdl:part name="AuthenticationHeader"
              element="impl:AuthenticationHeader"/>
   <wsdl:part name="parameters" element="impl:allocateInventory"/>
</wsdl:message>

<element name="allocateInventory">
   <complexType>
      <sequence>
         <element name="request" type="impl:UpdateRequest"/>
      </sequence>
   </complexType>
</element>
```

```xml
<complexType name="UpdateRequest">
   <sequence>
      <element name="FulfillerLocationCatalog" type="impl:FulfillmentLocationCatalog"/>
      <element name="Items" type="impl:ArrayOf_impl_UpdateItem"/>
   </sequence>
</complexType>

<complexType name="FulfillmentLocationCatalog">
   <sequence>
      <element name="ManufacturerCatalog" type="impl:ManufacturerCatalog"
               nillable="true"/>
      <element name="FulfillerLocationID" type="xsd:positiveInteger"/>
      <element name="ManufacturerLocation" type="impl:ManufacturerLocation"
               nillable="true"/>
      <element name="RetailerLocation" type="impl:RetailerLocation" nillable="true"/>
   </sequence>
</complexType>

<complexType name="ManufacturerCatalog">
   <sequence>
      <element name="ManufacturerID" type="xsd:positiveInteger"/>
      <element name="CatalogID" type="xsd:positiveInteger"/>
   </sequence>
</complexType>

<complexType name="ManufacturerLocation">
   <sequence>
      <element name="ManufacturerID" type="xsd:positiveInteger"/>
      <element name="ManufacturerLocationID" type="xsd:positiveInteger"/>
   </sequence>
</complexType>

<complexType name="RetailerLocation">
   <sequence>
      <element name="RetailerID" type="xsd:positiveInteger"/>
      <element name="RetailerLocationID" type="xsd:positiveInteger"/>
   </sequence>
</complexType>

<complexType name="ArrayOf_impl_UpdateItem">
   <sequence>
      <element name="items" type="impl:UpdateItem" minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
</complexType>

<complexType name="UpdateItem">
   <sequence>
      <element name="PartNumber" type="xsd:string" nillable="true"/>
      <element name="UPC" type="xsd:string" nillable="true"/>
      <element name="LocationUPC" type="xsd:string" nillable="true"/>
      <element name="Quantity" type="xsd:int"/>
      <element name="OrderID" type="xsd:positiveInteger"/>
      <element name="OrderItemID" type="xsd:positiveInteger"/>
      <element name="ShipmentID" type="xsd:positiveInteger"/>
      <element name="FulfillerLocationID" type="xsd:positiveInteger" nillable="true"/>
   </sequence>
</complexType>
```

4. Output data expected:

```
<wsdl:message name="allocateInventoryResponse">
       <wsdl:part name="parameters" element="impl:allocateInventoryResponse"/>
</wsdl:message>

<element name="allocateInventoryResponse">
   <complexType/>
</element>
```

5. SQL statements:

```
FOR EACH ITEM IN ArrayOf_impl_UpdateItem
      UPDATE StoredIn(sku, fulfiller_id, bin_name, ext_ful_loc_id,
                      num_allocated)
         SET num_allocated = num_allocated + Quantity
         WHERE sku = PartNumber AND fulfiller_id = FulfillerId AND
               bin_name = BinId AND ext_ful_loc_id = LocationName;
```

De-Allocate Inventory

1. Identification:

Companion to the allocate operation, deallocate will be called when inventory should be released back into the pool and available for allocation again. This call will throw an error if the amount which is currently allocated is less than the deallocation request.

2. WSDL API calls:

```
<wsdl:operation name="deallocateInventory">
   <wsdl:input name="deallocateInventoryRequest”
               message="impl:deallocateInventoryRequest"/>
   <wsdl:output name="deallocateInventoryResponse"
                message="impl:deallocateInventoryResponse"/>
</wsdl:operation>
```

3. Input data:

```
<wsdl:message name="deallocateInventoryRequest">
   <wsdl:part name="AuthenticationHeader" element="impl:AuthenticationHeader"/>
   <wsdl:part name="parameters" element="impl:deallocateInventory"/>
</wsdl:message>

<element name="deallocateInventory">
   <complexType>
      <sequence>
         <element name="request" type="impl:UpdateRequest"/>
      </sequence>
   </complexType>
</element>

<complexType name="UpdateRequest">
   <sequence>
```

```
      <element name="FulfillerLocationCatalog" type="impl:FulfillmentLocationCatalog"/>
      <element name="Items" type="impl:ArrayOf_impl_UpdateItem"/>
   </sequence>
</complexType>

<complexType name="FulfillmentLocationCatalog">
   <sequence>
      <element name="ManufacturerCatalog" type="impl:ManufacturerCatalog"
               nillable="true"/>
      <element name="FulfillerLocationID" type="xsd:positiveInteger"/>
      <element name="ManufacturerLocation" type="impl:ManufacturerLocation"
               nillable="true"/>
      <element name="RetailerLocation" type="impl:RetailerLocation" nillable="true"/>
   </sequence>
</complexType>

<complexType name="ManufacturerCatalog">
   <sequence>
      <element name="ManufacturerID" type="xsd:positiveInteger"/>
      <element name="CatalogID" type="xsd:positiveInteger"/>
   </sequence>
</complexType>

<complexType name="ManufacturerLocation">
   <sequence>
      <element name="ManufacturerID" type="xsd:positiveInteger"/>
      <element name="ManufacturerLocationID" type="xsd:positiveInteger"/>
   </sequence>
</complexType>

<complexType name="RetailerLocation">
   <sequence>
      <element name="RetailerID" type="xsd:positiveInteger"/>
      <element name="RetailerLocationID" type="xsd:positiveInteger"/>
   </sequence>
</complexType>

<complexType name="ArrayOf_impl_UpdateItem">
   <sequence>
      <element name="items" type="impl:UpdateItem" minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
</complexType>

<complexType name="UpdateItem">
   <sequence>
      <element name="PartNumber" type="xsd:string" nillable="true"/>
      <element name="UPC" type="xsd:string" nillable="true"/>
      <element name="LocationUPC" type="xsd:string" nillable="true"/>
      <element name="Quantity" type="xsd:int"/>
      <element name="OrderID" type="xsd:positiveInteger"/>
      <element name="OrderItemID" type="xsd:positiveInteger"/>
      <element name="ShipmentID" type="xsd:positiveInteger"/>
      <element name="FulfillerLocationID" type="xsd:positiveInteger" nillable="true"/>
   </sequence>
</complexType>
```

4. Output data expected:

```
<wsdl:message name="deallocateInventoryResponse">
   <wsdl:part name="parameters" element="impl:deallocateInventoryResponse"/>
</wsdl:message>

<element name="deallocateInventoryResponse">
   <complexType/>
</element>
```

5. SQL statements:

```
FOR EACH ITEM IN ArrayOf_impl_UpdateItem
      UPDATE StoredIn(sku, fulfiller_id, bin_name, ext_ful_loc_id,
                      num_allocated)
         SET num_allocated = num_allocated - Quantity
         WHERE sku = PartNumber AND fulfiller_id = FulfillerId AND
               bin_name = BinId AND ext_ful_loc_id = LocationName;
```