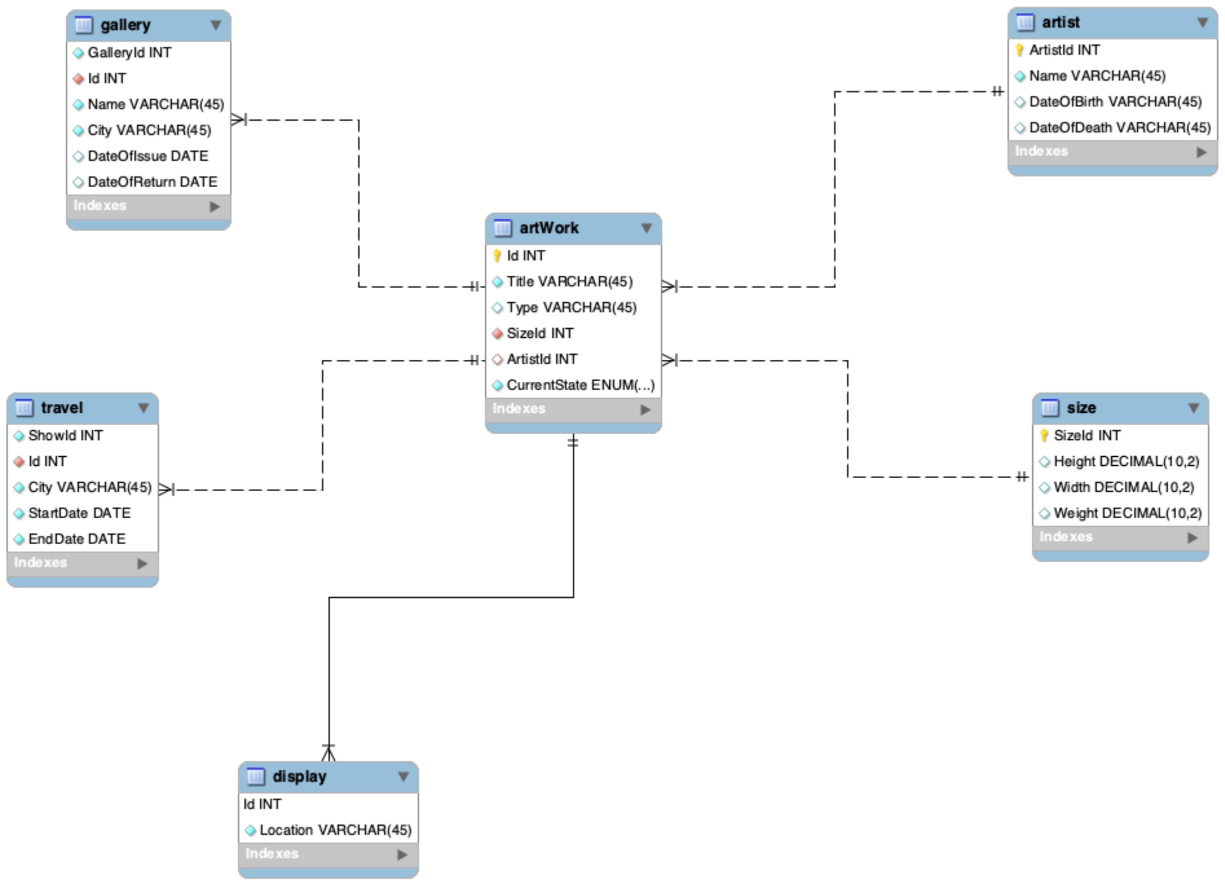
PART 1: ART MUSEUM DATABASE SCHEMA



Tables:

1. **ArtWork:** This is the table that contains info about all the items in the museum. Columns are-
   a. ID - Unique identifier for each item, Primary Key.
   b. Title - Title of the item
   c. Type - Type of the item
   d. SizeId - Size id Foreign key from size table
   e. ArtistId – Artist id Foreign key from artist table
   f. CurrentState – ENUM('display', 'storage', 'travel', 'gallery')

2. **Size:** This is the table that contains info about size of the items in the museum. Columns are-
    a. SizeID- Unique identifier for each type of size, Primary Key (FK in ArtWork).
    b. Height
    c. Weight
    d. Width

3. **Artist:** This is the table that contains info about the artists. Columns are-
    a. ArtistID- Unique identifier for each artist, Primary Key (FK in ArtWork).
    b. Name
    c. DateofBirth
    d. DateofDeath

4. **Display:** This is the table that contains info about the items indispaly. Columns are-
    a. ID- Unique identifier for each item in display, Primary Key (PK in ArtWork).
    b. Location

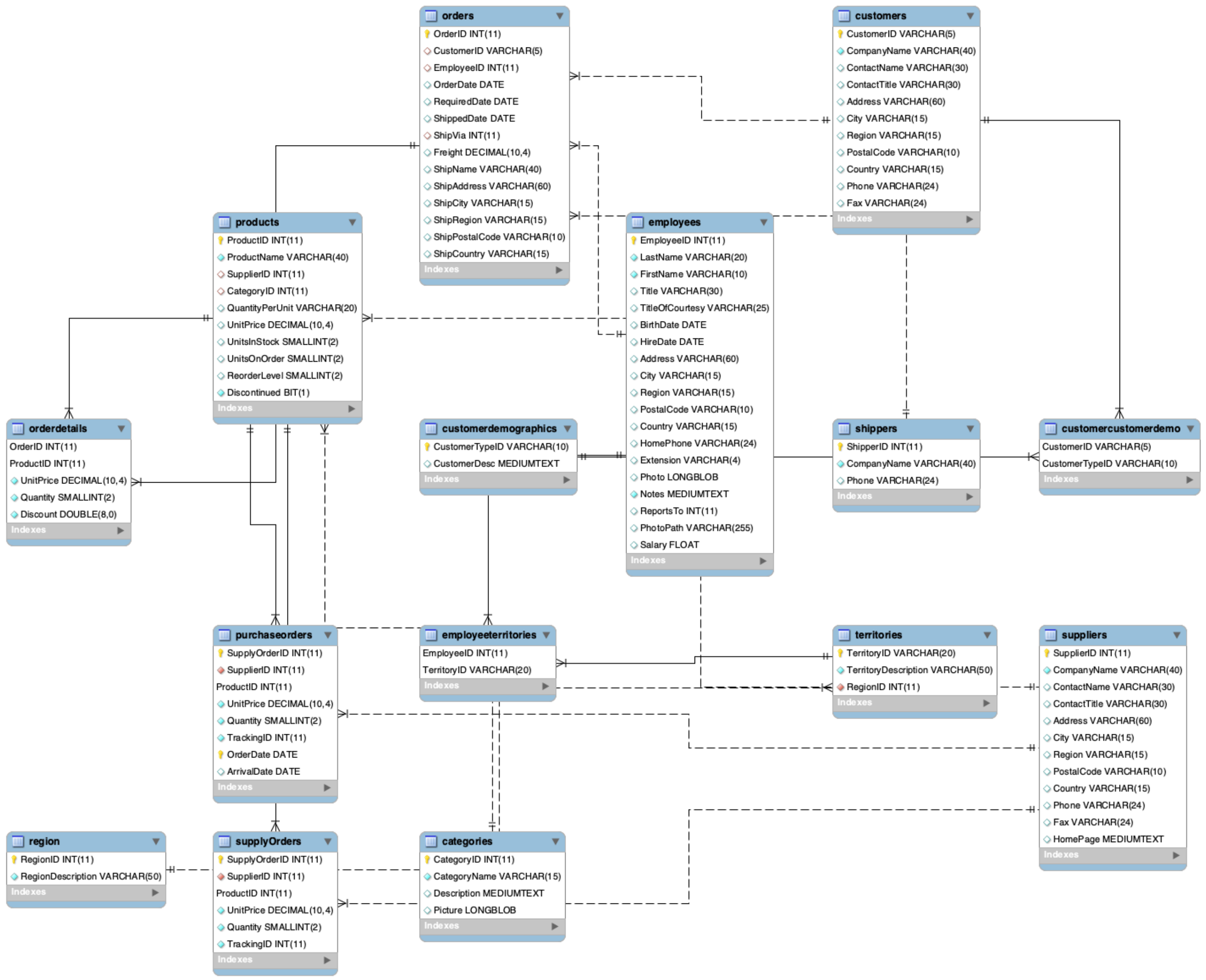
5. **Travel:** This is the table that contains info about the items in travel show. Columns are-
    a. ShowID- Unique identifier for each show, Primary Key (FK in ArtWork).
    b. ID - ID of items in show, Primary Key
    c. City
    d. StartDate
    e. EndDate

6. **Gallery:** This is the table that contains info about the items loaned to the gallery. Columns are-
    a. GalleryID- Unique identifier for each gallery, Primary Key
    b. ID - ID of items in that gallery, Primary Key
    c. City
    d. Name
    e. DateofIssue
    f. DateofReturn

PART 2: INVENTORY CONTROL PROGRAM OVERVIEW

Inventory Control is a program to implement database update for order shipment, reorders, received orders.

Database Modification: Added purchase orders table to add information from suppliers orders



Query: CREATE TABLE `purchaseorders` (

`SupplyOrderID` int(11) NOT NULL,

```
  `SupplierID` int(11) NOT NULL,

  `ProductID` int(11) NOT NULL,

  `UnitPrice` decimal(10,4) NOT NULL DEFAULT '0.0000',

  `Quantity` smallint(2) NOT NULL DEFAULT '1',

  `TrackingID` int(11) NOT NULL,

  `OrderDate` date DEFAULT NULL,

  `ArrivalDate` date DEFAULT NULL,

  PRIMARY KEY (`SupplyOrderID`,`ProductID`,`OrderDate`),

  KEY `FK_purchase_orders_products` (`ProductID`),

  CONSTRAINT `FK_purchase_orders_products` FOREIGN KEY (`ProductID`) REFERENCES
 `products` (`ProductID`),

  CONSTRAINT `FK_purchase_orders_supplier` FOREIGN KEY (`SupplierID`) REFERENCES
 `suppliers` (`SupplierID`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Inventory Control Methods:

- Ship Order:
    - **Update shipDate:** *"update orders set* ShippedDate =<ShipDate> where OrderID = <orderNumbe>"
    - **Get order details:** *"Select * from orderdetails where OrderID = <orderNumber>"*
    - **Get products in order:** *"Select * from products where ProductID in(<product_list>)"*
    - **For each product update stock:** *"Update products set UnitsInStock=<(Quantity-value)> where* ProductId=<key>"
- Issue reorder:
    - **Get all products that need reordering:** *"Select * from products where (UnitsInStock + UnitsOnOrder) <= ReorderLevel and Discontinued = 0"*
    - **For each product in above list get unit price in latest order:** *"with latest_order as (Select orderID as latestOrderId from orders where orderID in (Select distinct orderID from orderdetails where productID=<prodId>) order by orderDate desc limit 1) Select **

from orderdetails where orderId = (Select latestOrderId from latest_order) and productid =<prodId>"

- **Insert rows into purchase table:** *"insert into purchaseorders values(purchaseId, supId, prodId unitPrice, Quantity, trackid, orderDate , null)"*
- **For each product update stock:** *"Update products set UnitsOnOrder= UnitsOnOrder+ Quantity where ProductID=<prodId>"*

➢ Receive_order:

- **Update arrival date for given purchase order:** *"Update purchaseorders set ArrivalDate=<ArrivalDate> where SupplyOrderID=<internal_order_reference>"*
- **Get all the orders for given order id:** *"Select * from purchaseorders where SupplyOrderID=<internal_order_reference>"*
- **Insert rows into purchase table:** *"insert into purchaseorders values(purchaseId, supId, prodId unitPrice, Quantity, trackid, orderDate , null)"*
- **For each product update stock:** *"update products set UnitsInStock = < purchaseOrder.getInt("Quantity") >, UnitsOnOrder=0 where ProductID=<prodId> ";*

❖ Approach used:

- After connecting with database using JDBC driver.
- Perform queries that are explained above.
- Once result set for each query is obtained.
- Each result set is saved in global variables.
- Each table is updated to keep inventory up to date.
- Custom exception is used to handle cases

❖ Why this solution is ready to be deployed:

1. **Scalable:** Variable length data structures are used to save data from the queries which means that even if database gets scaled to any further number of rows, summary generator can handle it without any issue.
2. **Flexibility:** There is no hard coding for any query or data processing that means any new field can be queried. There is also use of subqueries that means more subqueries can be added to enhance the results.
3. **Performance:** Implemented solution is time-efficient as the data is stored globally and is easily accessible and even loops are run on dynamic data-structure. It can work well in the company
4. **Reusability:** There are two generic programs written that implement all update, insert and select queries. Which enables code maintainability and removes redundancy.

❖ Testing Information:

- Run the script in northwind_pkaur.sql which will create local data base named pkaur.
- Then set units in stock to zero for certain products for testing purposes. Use query: *Update products set UnitsInStock = 0, UnitsOnOrder=0 where ProductID in(1,2,3,4,5);*
- Run : *select * from purchaseorders;* in database and table should be empty.
- Now run the program in northwindInventory: Ship_order(10248), Issue_reorders(2019, 03, 13), Receive_order(125)
- Run : *select * from purchaseorders;* in database and table should have four rows now.
- Call methods with Ids that do not exist, it should throw exception saying order id does not exist.

Submitted by:

Prabhjot Kaur
B00843735