// Create products

USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM "file:///AdventureWorks/products.csv" as row

CREATE (:Product {productName: row.ProductName, productNumber: row.ProductNumber, productId: row.ProductID, modelName: row.ProductModelName, standardCost: row.StandardCost, listPrice: row.ListPrice});

CREATE CONSTRAINT ON (p:Product) ASSERT p.productId IS UNIQUE;

// Create vendors

USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM "file:///AdventureWorks/vendors.csv" as row

CREATE (:Vendor {vendorName: row.VendorName, vendorNumber: row.AccountNumber, vendorId: row.VendorID, creditRating: row.CreditRating, activeFlag: row.ActiveFlag});

CREATE CONSTRAINT ON (v:Vendor) ASSERT v.vendorId IS UNIQUE;

// Create employees

USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM "file:///AdventureWorks/employees.csv" as row

CREATE (:Employee {firstName: row.FirstName, lastName: row.LastName, fullName: row.FullName, employeeId: row.EmployeeID, jobTitle: row.JobTitle, organizationLevel: row.OrganizationLevel, maritalStatus: row.MaritalStatus, gender: row.Gender, territoty: row.Territory, country: row.Country, group: row.Group});

CREATE CONSTRAINT ON (e:Employee) ASSERT e.employeeId IS UNIQUE;

// Create customers

USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM "file:///AdventureWorks/customers.csv" as row

CREATE (:Customer {firstName: row.FirstName, lastName: row.LastName, fullName: row.FullName, customerId: row.CustomerID});

CREATE CONSTRAINT ON (c:Customer) ASSERT c.customerId IS UNIQUE;

// Create categories

USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM "file:///AdventureWorks/productcategories.csv" as row

CREATE (:Category {categoryName: row.CategoryName, categoryId: row.CategoryID});

CREATE CONSTRAINT ON (c:Category) ASSERT c.categoryId IS UNIQUE;

// Create sub-categories

USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM "file:///AdventureWorks/productsubcategories.csv" as row

CREATE (:SubCategory {subCategoryName: row.SubCategoryName, subCategoryId: row.SubCategoryID});

CREATE CONSTRAINT ON (s:SubCategory) ASSERT s.subCategoryId IS UNIQUE;

// Prepare orders

USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM "file:///AdventureWorks/orders.csv" AS row

MERGE (order:Order {orderId: row.SalesOrderID}) ON CREATE SET order.orderDate = row.OrderDate;

CREATE CONSTRAINT ON (o:Order) ASSERT o.orderId IS UNIQUE;

// Create relationships: Order to Product

USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM "file:///AdventureWorks/orders.csv" AS row

MATCH (order:Order {orderId: row.SalesOrderID})

MATCH (product:Product {productId: row.ProductID})

MERGE (order)-[pu:PRODUCT]->(product)

ON CREATE SET pu.unitPrice = toFloat(row.UnitPrice), pu.quantity = toFloat(row.OrderQty);

// Create relationships: Order to Employee

USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM "file:///AdventureWorks/orders.csv" AS row

MATCH (order:Order {orderId: row.SalesOrderID})

MATCH (employee:Employee {employeeId: row.EmployeeID})

MERGE (employee)-[:SOLD]->(order);

// Create relationships: Order to Customer

USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM "file:///AdventureWorks/orders.csv" AS row

MATCH (order:Order {orderId: row.SalesOrderID})

MATCH (customer:Customer {customerId: row.CustomerID})

MERGE (customer)-[:PURCHASED]->(order);

// Create relationships: Product to Vendor

USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM "file:///AdventureWorks/vendorproduct.csv" AS row

MATCH (product:Product {productId: row.ProductID})

MATCH (vendor:Vendor {vendorId: row.VendorID})

MERGE (vendor)-[:SUPPLIES]->(product);

// Create relationships: Product to SubCategory

USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM "file:///AdventureWorks/products.csv" AS row

MATCH (product:Product {productId: row.ProductID})

MATCH (subcategory:SubCategory {subCategoryId: row.SubCategoryID})

MERGE (product)-[:PART\_OF\_SUBCAT]->(subcategory);

// Create relationships: SubCategory to Category

USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM "file:///AdventureWorks/productsubcategories.csv" AS row

MATCH (subcategory:SubCategory {subCategoryId: row.SubCategoryID})

MATCH (category:Category {categoryId: row.CategoryID})

MERGE (subcategory)-[:PART\_OF\_CAT]->(category);

// Create relationship for employee reporting structure

USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM "file:///AdventureWorks/employees.csv" AS row

MATCH (employee:Employee {employeeId: row.EmployeeID})

MATCH (manager:Employee {employeeId: row.ManagerID})

MERGE (employee)-[:REPORTS\_TO]->(manager);