Database Project

You are about to challenge Amazon.com as a marketer of “all things.” You will begin to reproduce the Amazon experience. You will sell your own products, and each product will be stocked by a single supplier. You will also begin to develop your own customers who will order your products. Each product will have a minimum stock level.

Things go well for a year or two, but then you decide to enter into a partnership with Northwind Traders, AdventureWorks Cycles, and Sakila Films. Each of these partners will also sell your products and in return you can sell their products. Not only that, but you and your three partners will share (not duplicate) customer lists. If your partners add a new product or enroll a new customer, these should be instantly available to your application.

Your application should allow the following basic functionality:

1. Ability to view, add, remove and modify **your** customer information
2. Ability to view **your partners’** customer information
3. Ability to view, add, remove and modify **your** supplier information.
4. Ability to view, add, remove and modify **your** product information
5. Ability to view, **your partners’** product information
6. Ability to view **your** product inventory
7. Ability to generate a restocking order (should be saved in a ”restocking” table) if the supply of any of **your** products falls below the minimum stock level
8. Ability of a customer to place an order, which consists of adding **your or your partners’** items to a shopping cart and then checking out.
9. Ability to browse the product catalog by category (We know that Northwind sells food items, AdventureWorks sells bikes and accessories, and Sakila sells movies. **Your** product catalog should include items from your partners’ categories as well as items from other categories that your partners do not traffic in.

Your application should include the following basic reports (SQL queries)

10. List of all **your** products whose inventory has fallen below the minimum stock level
11. List of customers who have not been “too active”(*you define this*) and for whom special offers should be made.
12. List of products that are not selling “too well”(*you define this*), which might be offered as specials.
13. When the products purchased will ship (Shipping will occur four weekdays from now, e.g., if today is Monday, they will ship on Friday.

The above functionality should be implemented using MySQL on elvis. Additionally, you should create a miniature application in a **different** database package of your choice. That second implementation does not need any insert, update or delete capabilities – but it should include all of the queries to view all of the functionality from 1 to 13 above.

Finally, you will need to enhance your MySQL application to include the following advanced functionality:

14. Ability of a customers to place an item on his/her “wish list.”
15. An algorithm (manifested as a query) to suggest additional products that a customer might be interested in based on their order history, their wish list, or anything else you would like to program.
16. Ability of customers to rate products
17. Ability to view the ratings of products in two ways
    - The average rating based on all rating activity
    - A more intelligent rating that uses an algorithm to weight some customer’s ratings higher than others.

And these additional advanced reports:

18. A report showing the most highly wished for products in every category
19. A report showing wished for products that were never purchased by the customers who wished for them
20. EXTRA CREDIT: What other innovative reports can you think of?

Customer Tables from your Partners

northwind.customer					
Field	**Type**	**Null**	**Key**	**Default**	**Extra**
id	int(11)	NO	PRI		auto _increment
company	varchar(50)	YES	MUL		
last_name	varchar(50)	YES	MUL		
first_name	varchar(50)	YES	MUL		
email_address	varchar(50)	YES			
job_title	varchar(50)	YES			
business_phone	varchar(25)	YES			
home_phone	varchar(25)	YES			
mobile_phone	varchar(25)	YES			
fax_number	varchar(25)	YES			
address	longtext	YES			
city	varchar(50)	YES	MUL		
state_province	varchar(50)	YES	MUL		
zip_postal_code	varchar(15)	YES	MUL		
country_region	varchar(50)	YES			
web_page	longtext	YES			
notes	longtext	YES			
attachments	longblob	YES			
Sample customer: Anna Bedecs					

adventureworks.customer					
Field	**Type**	**Null**	**Key**	**Default**	**Extra**
CustomerID	int(11)	NO	PRI		auto _increment
TerritoryID	int(11)	YES			
AccountNumber	varchar(10)	NO			
CustomerType	varchar(1)	NO			
rowguid	varbinary(16)	NO			
ModifiedDate	timestamp	NO		CURRENT_ TIMESTAMP	
Sample customer: David Robinett					

sakila.customer					
Field	**Type**	**Null**	**Key**	**Default**	**Extra**
customer_id	smallint(5) unsigned	NO	PRI		auto _increment
store_id	tinyint(3) unsigned	NO	MUL		
first_name	varchar(45)	NO			
last_name	varchar(45)	NO	MUL		
email	varchar(50)	YES			
address_id	smallint(5) unsigned	NO	MUL		
active	tinyint(1)	NO		1	
create_date	datetime	NO			
last_update	timestamp	NO		CURRENT_ TIMESTAMP	on update CURRENT_ TIMESTAMP
Sample customer: Mary Smith					

Product Tables from your Partners

northwind.product					
Field	**Type**	**Null**	**Key**	**Default**	**Extra**
supplier_ids	longtext	YES			
id	int(11)	NO	PRI		auto_increment
product_code	varchar(25)	YES	MUL		
product_name	varchar(50)	YES			
description	longtext	YES			
standard_cost	decimal(19,4)	YES		0	
list_price	decimal(19,4)	NO		0	
reorder_level	int(11)	YES			
target_level	int(11)	YES			
quantity_per_unit	varchar(50)	YES			
discontinued	tinyint(1)	NO		0	
minimum_reorder_quantity	int(11)	YES			
category	varchar(50)	YES			
attachments	longblob	YES			
Sample product: Northwind Traders Curry Sauce					

adventureworks.product					
Field	**Type**	**Null**	**Key**	**Default**	**Extra**
ProductID	int(11)	NO	PRI		auto_increment
Name	varchar(50)	NO			
ProductNumber	varchar(25)	NO			
MakeFlag	bit(1)	NO			
FinishedGoodsFlag	bit(1)	NO			
Color	varchar(15)	YES			
SafetyStockLevel	smallint(6)	NO			
ReorderPoint	smallint(6)	NO			
StandardCost	double	NO			
ListPrice	double	NO			
Size	varchar(5)	YES			
SizeUnitMeasureCode	varchar(3)	YES			
WeightUnitMeasure Code	varchar(3)	YES			
Weight	decimal(8,2)	YES			
DaysToManufacture	int(11)	NO			
ProductLine	varchar(2)	YES			
Class	varchar(2)	YES			
Style	varchar(2)	YES			
ProductSubcategoryID	int(11)	YES			
ProductModelID	int(11)	YES			
SellStartDate	datetime	NO			
SellEndDate	datetime	YES			
DiscontinuedDate	datetime	YES			
rowguid	varbinary(16)	NO			
ModifiedDate	timestamp	NO		CURRENT_TIMESTAMP	
Sample product: Seat Tube					

sakila.film					
Field	**Type**	**Null**	**Key**	**Default**	**Extra**
film_id	smallint(5) unsigned	NO	PRI		auto_increment
title	varchar(255)	NO	MUL		
description	text	YES			
release_year	year(4)	YES			
language_id	tinyint(3) unsigned	NO	MUL		
original_language_id	tinyint(3) unsigned	YES	MUL		
rental_duration	tinyint(3) unsigned	NO		3	
rental_rate	decimal(4,2)	NO		4.99	
length	smallint(5) unsigned	YES			
replacement_cost	decimal(5,2)	NO		19.99	
rating	enum('G','PG','PG-13','R','NC-17')	YES		G	
special_features	set('Trailers','Commentaries','Deleted Scenes','Behind the Scenes')	YES			
last_update	timestamp	NO		CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP
Sample product (film): Sorority Queen					

Your deliverable

Each team will host the MySQL application on elvis and provide a URL so that the functionality can be demonstrated.

Each team will also demonstrate their non-MySQL version of the application

A design document must be produced which should include:

- An ER diagram or an EER diagram
- The physical schema diagram from MySQL
- All SQL statements that support the basic and advanced functionality and reports of the system
- All create table scripts for **your** MySQL tables
- All index creation scripts for **your** MySQL tables
- All create view scripts for **your** MySQL views
- All grant scripts for your MySQL tables and views
- A description of the algorithms you used for "suggested products" and more accurate product ratings (Note: these should be PROCEDURE based as much as possible).
- Source code for any database procedures or triggers

- All database statements from your second database that support the basic functionality and reports of the system.
- A <u>comprehensive</u> description of the differences between the MySQL implementation and the other implementation. This should include well articulated pros and cons of each implementation.