Exot Animal E-Commerce Platform Report

MSA 8040: Data Management for Analytics

Professor Saeid Motevali

Sunday, December 8, 2024 11:59 PM

Lilly Parham
Gracie Rehberg
Pamela Alvarado-Zarate

PURPOSE

Commercial animal distribution has long been a subject of controversy due to concerns about inhumane conditions and safety risks during the holding and transport of animals. Illegal trades, particularly in exotic animals, often disregard care standards and regulations, causing harm to the animals and tarnishing the reputation of the legal distribution industry. To address these challenges, the **Exot Animal E-Commerce platform** was developed to provide a safe, legal, and humane delivery system for exotic wildlife and their necessities, adhering strictly to legal guidelines and upholding rigorous safety standards during the animals' transition to new homes.

The platform offers a transparent and ethical marketplace for consumers while equipping administrative users with a cutting-edge record-keeping and business analytics dashboard. The customer-facing website serves as the primary transactional interface, seamlessly integrated with an interactive, data-driven dashboard for business stakeholders.

THE DATABASE

During the brainstorming phase of our project, we selected the E-Commerce option as our topic because it is highly relevant in today's age of online shopping. Initially, we designed a database with six entities: users, orders, order details, products, categories, and suppliers. However, after reviewing the project guidelines, we realized our database lacked sufficient entities and relationships. To address this, we expanded the design by adding payments, discounts, shipping addresses, wish lists, return requests, and inventory logs.

During the development phase in SQL, our team encountered challenges integrating the new entities due to central dependencies and coding errors. As a result, we revised the database again, removing return requests, wish lists, and inventory logs. Return requests were deemed unnecessary since our products were intended to be final sale, wish lists were excluded to focus solely on actual purchases (e.g., buying a lion rather than wishing for one), and inventory logs were consolidated into attributes within the product entity.

The final database consisted of 10 entities and 11 relationships: users, orders, products, suppliers, payments, order details, shipping addresses, discounts, reviews, and categories. We used MySQL Workbench to develop the **Exot Animal E-Commerce** database.

Figures 1 and 2 illustrate the entity-relationship (ER) diagrams. Figure 1, created with LucidChart, provides an initial representation, while Figure 2, generated by MySQL, depicts a more detailed version. Figure 3 showcases the database's data dictionary, detailing all entities, attributes, data types, and descriptions tailored to the database's requirements.

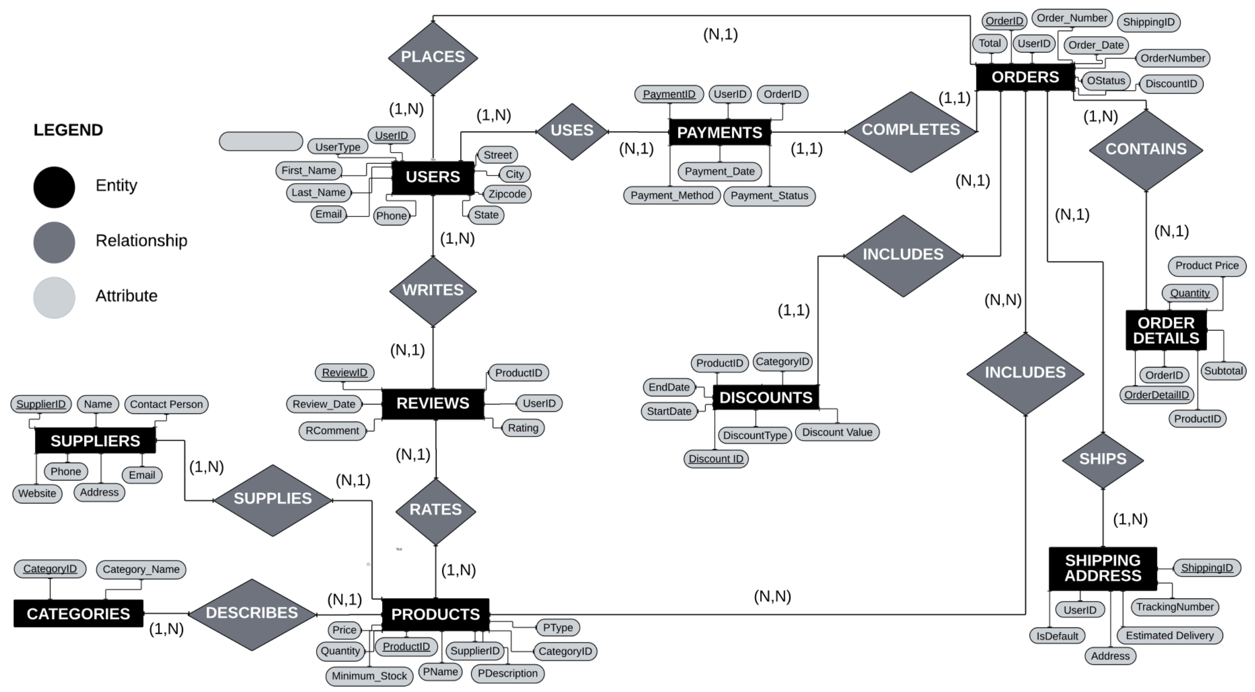


Figure 1. Entity-Relationship (ER) Diagram of Exot Animal E-Commerce database (Created using LucidChart).

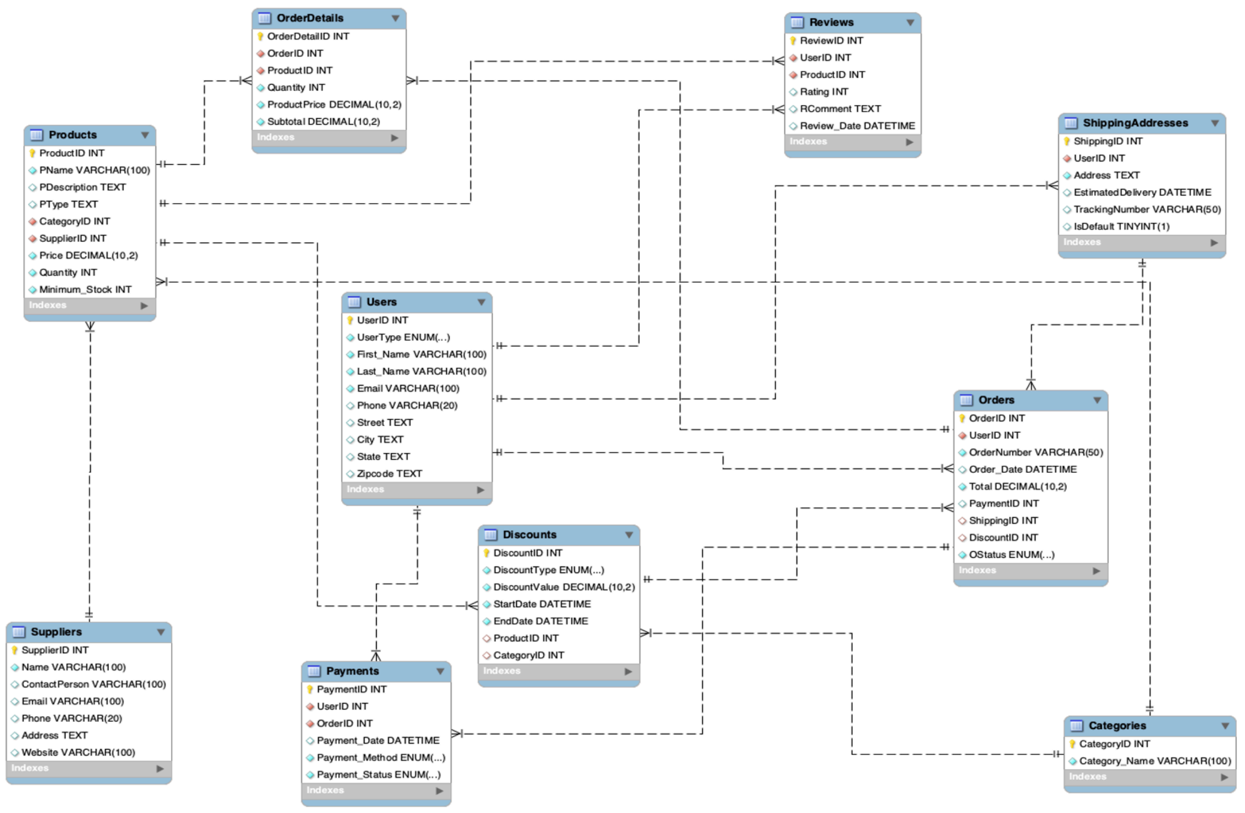


Figure 2. Enhanced Entity-Relationship (EER) Diagram of Exot Animal E-Commerce database.

Entity	Attribute	Data Type	Description
Users	UserID	Integer	Unique identifier for each user.
	Name	String	Full name of the user.
	Email	String	Email address of the user.
	Password	String	Encrypted password for the user's account.
	Address	String	Physical address of the user.
	Phone	String	Contact number of the user.
	UserType	Enum ("Admin", "Customer")	Specifies whether the user is an admin or a customer.
Products	ProductID	Integer	Unique identifier for each product.
	Name	String	Name of the product.
	CategoryID	Integer (Foreign Key)	Identifier linking the product to a category.
	Price	Float	Price of the product.
	StockQuantity	Integer	Number of items available in stock.
	Description	String	Description or details about the product.
	SupplierID	Integer (Foreign Key)	Identifier linking the product to a supplier.

Categories	CategoryID	Integer	Unique identifier for each category.
	CategoryName	String	Name of the category.
Orders	OrderID	Integer	Unique identifier for each order.
	UserID	Integer (Foreign Key)	Identifier linking the order to the user who placed it.
	OrderDate	Date	Date and time when the order was placed.
	TotalAmount	Float	Total monetary value of the order.
	PaymentID	Integer (Foreign Key)	Identifier linking the order to its payment record.
	Status	Enum ("Pending", "Completed", "Cancelled")	Current status of the order.
	ShippingID	Integer (Foreign Key)	Identifier linking the order to its shipping information.
	DiscountID	Integer (Foreign Key)	Identifier linking the order to a discount or coupon applied.
Order Details	OrderDetailID	Integer	Unique identifier for each order detail record.
	OrderID	Integer (Foreign Key)	Identifier linking the detail to an order.
	OrderNumber	String	Unique order number for tracking purposes.
	ProductID	Integer (Foreign Key)	Identifier linking the detail to a product.

	Quantity	Integer	Number of units of the product in the order.
	ProductPrice	Decimal(10,2)	Price of the product at the time of the order.
Reviews	ReviewID	Integer	Unique identifier for each review.
	ProductID	Integer (Foreign Key)	Identifier linking the review to a product.
	UserID	Integer (Foreign Key)	Identifier linking the review to a user.
	Rating	Integer (1-5)	Rating given by the user for the product.
	Comment	String	User's feedback about the product.
	Review Date	Date	Date and time when the review was submitted.
Payments	PaymentID	Integer	Unique identifier for each payment.
	UserID	Integer (Foreign Key)	Identifier linking the payment to a user.
	OrderID	Integer (Foreign Key)	Identifier linking the payment to an order.
	Payment Method	Enum ("Credit Card", "PayPal", "Bank Transfer")	Payment method used.
	Payment Status	Enum ("Successful", "Failed", "Pending")	Current status of the payment.
	Payment Date	Date	Date and time when the payment was made.

Shipping Addresses	ShippingID	Integer	Unique identifier for each shipping record.
	UserID	Integer (Foreign Key)	Identifier linking the shipping record to a user.
	Address	String	Shipping address for the order.
	EstimatedDelivery	Date	Estimated delivery date for the order.
	TrackingNumber	String	Tracking number for the shipment.
Discounts	DiscountID	Integer	Unique identifier for each discount or coupon.
	DiscountType	String	Type of discount applied.
	DiscountValue	Float	Value of the discount.
	StartDate	Date	Start date for the discount or coupon.
	EndDate	Date	End date for the discount or coupon.
	ProductID	Integer (Foreign Key)	Identifier linking the discount to a product.
	CategoryID	Integer (Foreign Key)	Identifier linking the discount to a category.
Suppliers	SupplierID	Integer	Unique identifier for each supplier.
	Name	String	Name of the supplier.
	ContactPerson	String	Name of the supplier's contact person.

	Email	String	Email address of the supplier.
	Phone	String	Contact number for the supplier.
	Address	String	Address of the supplier.
	Website	String	Supplier's website.

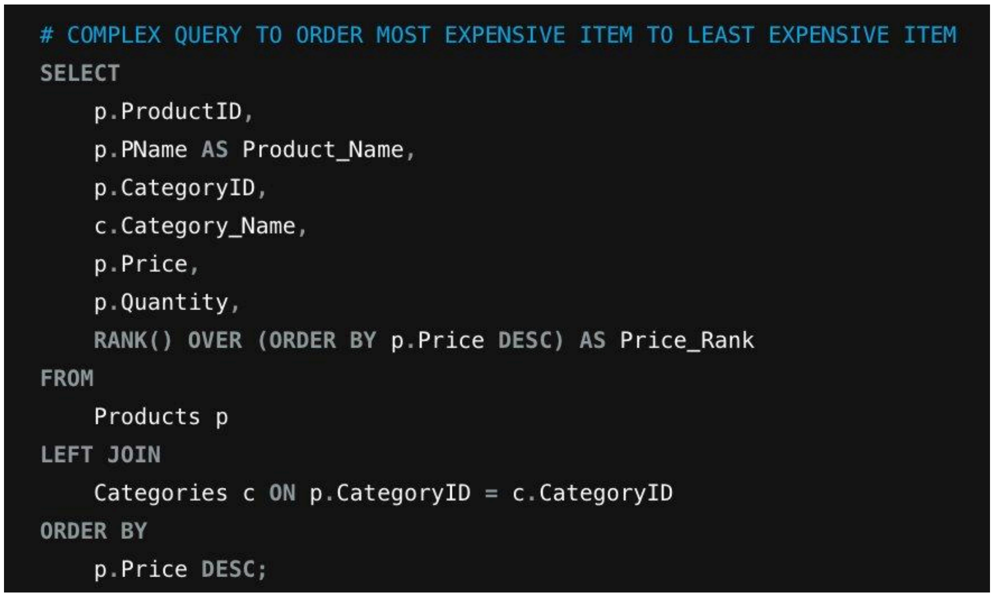
Figure 3. Data Dictionary of the Exot Animal E-Commerce database.

QUERY DEVELOPMENT

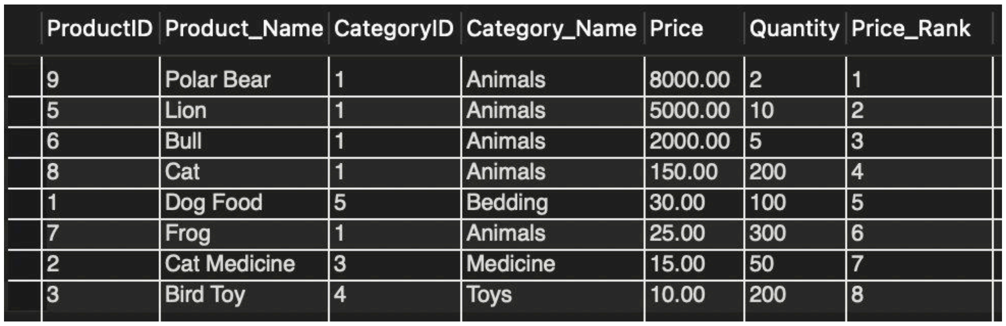
MySQL query functions were utilized for the addition, removal, and retrieval of data. Basic CRUD queries, such as "INSERT," "DELETE," "UPDATE," "SELECT," completed these actions. Complex queries containing "GROUP BY," "ORDER BY," and "JOIN" functions were utilized to view the highest or lowest number of a certain attribute to identify trends, show popular items, or monitor inventory, for example. After developing and adding the entities and attributes into MySQL, we began to use CRUD queries to insert and create dummy data for our e-commerce system.

While brainstorming queries, Gracie and Lilly decided it would be fitting to insert their first customer as their mutual friend, Bhavin Patel, who had introduced them to each other.
By simulating the experience of a real person in our e-commerce scenario, we gained firsthand insights into the customer experience and explored the potential of the business problem more deeply. Once we had our first customer included in the data we then needed to curate a product our friend would want to purchase. In the end, we chose a lion and decided on the development of the Exot Animal E-Commerce platform. We continued to build around the exotic animal concept inserting more exotic animals for purchase, and then continued to expand on categories related to the needs of such exotic animals. Then we implemented CRUD queries to insert more categories of items we wanted to sell including Food, Medicine, Bedding and Toys. Finally we inserted corresponding products available for sale within the categories including Dog Food as a product available for sale within the Food category.

After the basic queries were implemented, we then incorporated complex queries to create various analyses for user and administration purposes. One complex query included the left join and order by function to sort all of the products for sale on Exot Animal E-Commerce from least to most expensive, which we can see in Figure 4. This query is important for our user interface because while shopping, many users want to toggle their product display by costs to be able to look at the products within their budget means.



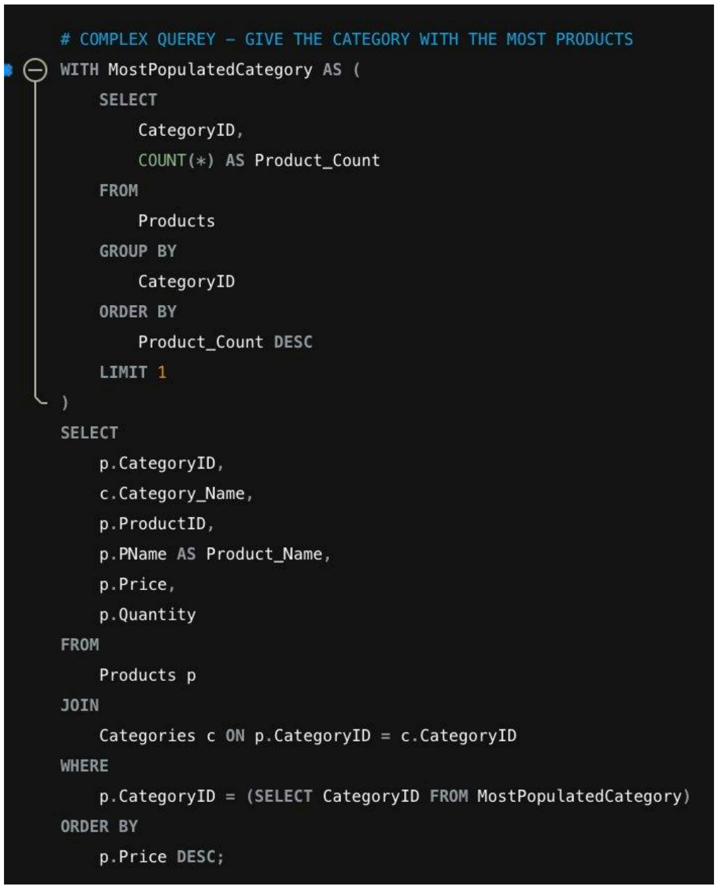
```
# COMPLEX QUERY TO ORDER MOST EXPENSIVE ITEM TO LEAST EXPENSIVE ITEM
SELECT
    p.ProductID,
    p.PName AS Product_Name,
    p.CategoryID,
    c.Category_Name,
    p.Price,
    p.Quantity,
    RANK() OVER (ORDER BY p.Price DESC) AS Price_Rank
FROM
    Products p
LEFT JOIN
    Categories c ON p.CategoryID = c.CategoryID
ORDER BY
    p.Price DESC;
```



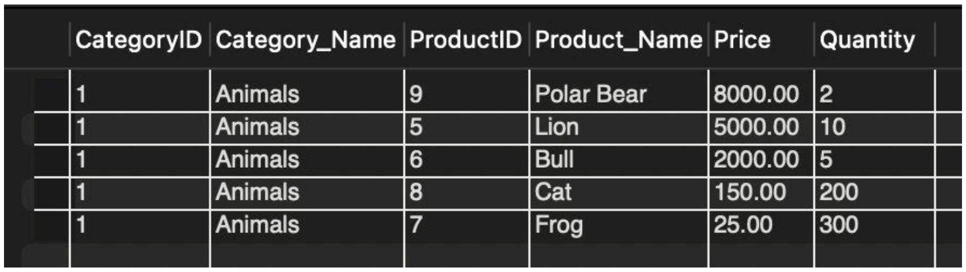
ProductID	Product_Name	CategoryID	Category_Name	Price	Quantity	Price_Rank
9	Polar Bear	1	Animals	8000.00	2	1
5	Lion	1	Animals	5000.00	10	2
6	Bull	1	Animals	2000.00	5	3
8	Cat	1	Animals	150.00	200	4
1	Dog Food	5	Bedding	30.00	100	5
7	Frog	1	Animals	25.00	300	6
2	Cat Medicine	3	Medicine	15.00	50	7
3	Bird Toy	4	Toys	10.00	200	8

Figure 4. Least to expensive product complex query & results.

Two complex queries were designed for user administration purposes, focusing on the database back end. The first query utilized GROUP BY, ORDER BY, and JOIN functions to identify the category with the highest product availability, as shown in Figure 5. This query helps the administrative team analyze inventory results and predict business needs. For example, the data can guide decisions on whether to add new products to certain categories to enhance availability or maintain the current product listings. Additionally, the query results might indicate categories with excessive inventory, prompting administrators to consider reducing replenishment from suppliers or eventually removing overstocked products from availability.



```
# COMPLEX QUEREY - GIVE THE CATEGORY WITH THE MOST PRODUCTS
WITH MostPopulatedCategory AS (
    SELECT
        CategoryID,
        COUNT(*) AS Product_Count
    FROM
        Products
    GROUP BY
        CategoryID
    ORDER BY
        Product_Count DESC
    LIMIT 1
)
SELECT
    p.CategoryID,
    c.Category_Name,
    p.ProductID,
    p.PName AS Product_Name,
    p.Price,
    p.Quantity
FROM
    Products p
JOIN
    Categories c ON p.CategoryID = c.CategoryID
WHERE
    p.CategoryID = (SELECT CategoryID FROM MostPopulatedCategory)
ORDER BY
    p.Price DESC;
```



CategoryID	Category_Name	ProductID	Product_Name	Price	Quantity
1	Animals	9	Polar Bear	8000.00	2
1	Animals	5	Lion	5000.00	10
1	Animals	6	Bull	2000.00	5
1	Animals	8	Cat	150.00	200
1	Animals	7	Frog	25.00	300

Figure 5. Category with the most product availability complex query & results.

The second complex query also utilized GROUP BY, ORDER BY, and JOIN functions to identify the suppliers with the highest product availability. This query plays a critical role in back-end transactions by enabling the administrative team to determine which suppliers the business relies on most for inventory.

Overall, CRUD queries were primarily used to build and manage the database, such as inserting data, while complex queries were employed to analyze and extract key insights, such as purchasing trends, to help predict and guide essential business decisions.

FUNCTIONALITY

The functionality of the Exot Animal E-Commerce platform is enhanced through the implementation of triggers and stored procedures in the database. A trigger, created using the CREATE TRIGGER command, is executed automatically whenever a new record is inserted into the OrderDetails table. This ensures seamless backend operations, such as updating related tables or performing validations, without requiring manual intervention. Additionally, stored procedures, created using the CREATE PROCEDURE command, provide robust tools for generating key business insights. These include calculating the total amount spent by each customer and producing detailed sales reports for specific time periods. By leveraging these features, the platform delivers efficient and automated database operations for our customers, ensuring reliable and insightful data management to support the E-commerce business model.

USER INTERFACE

The functionality of the e-commerce platform is enhanced through the integration of advanced database features such as triggers and stored procedures, along with a user-friendly and administrative interface. Triggers, implemented using the CREATE TRIGGER command, are designed to automate backend operations. For example, a trigger automatically flags products with low stock by updating a LowStockFlag in the PRODUCTS table whenever stock levels fall below the defined threshold. This eliminates the need for manual inventory checks and ensures real-time stock updates, streamlining inventory management. Additionally, stored procedures, created with the CREATE PROCEDURE command, generate valuable business insights, including the total amount spent by each customer and detailed sales reports for specified time periods. These automated functionalities improve operational efficiency and data accuracy, providing essential support for the e-commerce platform's processes.

Complementing the backend features, the platform includes a comprehensive user interface for both customers and administrators. The customer-facing interface is designed as a website with robust input validation and error handling, ensuring accurate data collection and enhancing the user experience, see Figure 6-9. Input validation includes credit card checks and mandatory address fields to prevent payment and shipping delays, while error handling notifies users of missing or incorrect information, see Figure 10-11. From an administrative perspective, an interactive dashboard built with Tableau provides insights into sales trends, revenue, and customer satisfaction, see Figure 12. Administrators can analyze historical sales data by filtering for specific months and years, enabling them to identify causes of low revenue and implement targeted marketing strategies. Together, these features create a powerful, efficient, and user-friendly platform for managing e-commerce operations.

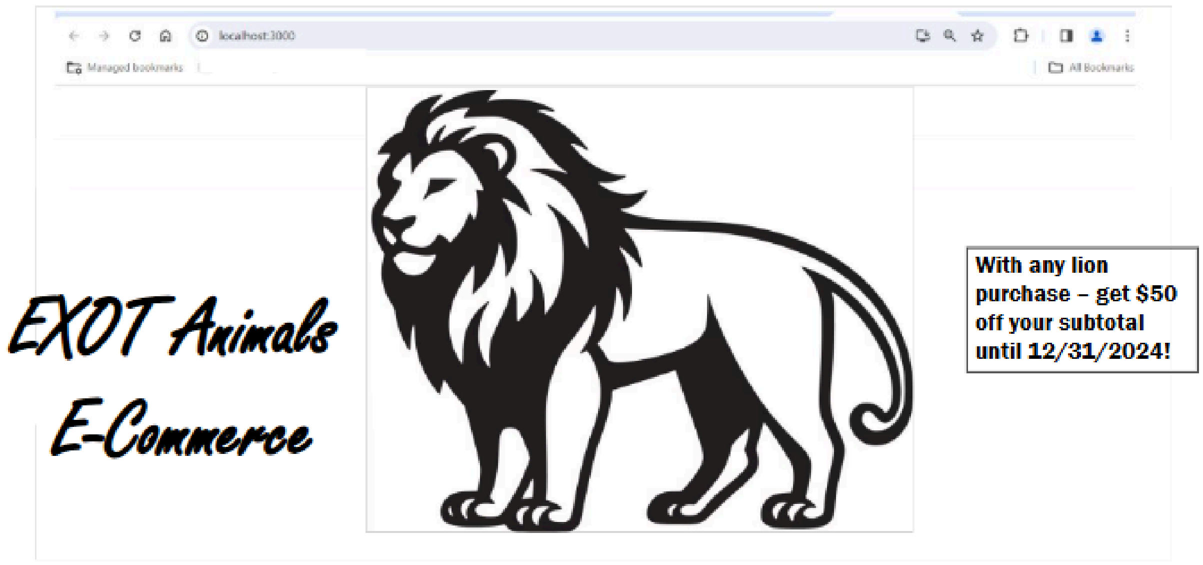
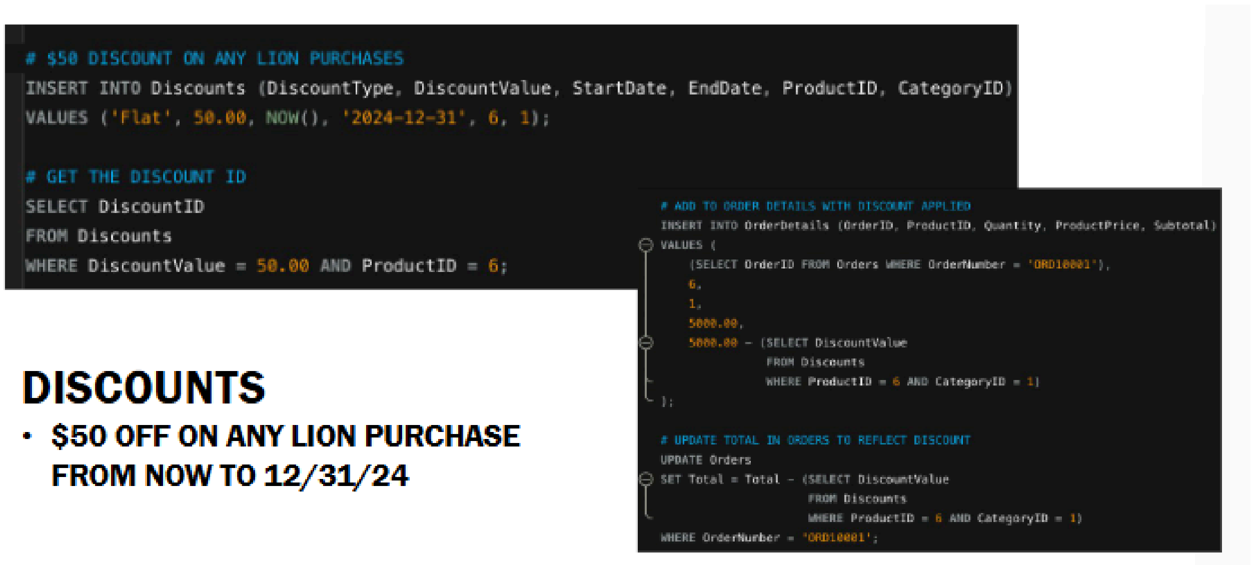

localhost:3000
Managed bookmarks
All Bookmarks
EXOT Animals
E-Commerce
With any lion purchase – get $50 off your subtotal until 12/31/2024!

Figure 6. The Front-End: Front page of the Exot Animals E-Commerce Website



```
# $50 DISCOUNT ON ANY LION PURCHASES
INSERT INTO Discounts (DiscountType, DiscountValue, StartDate, EndDate, ProductID, CategoryID)
VALUES ('Flat', 50.00, NOW(), '2024-12-31', 6, 1);

# GET THE DISCOUNT ID
SELECT DiscountID
FROM Discounts
WHERE DiscountValue = 50.00 AND ProductID = 6;
```

DISCOUNTS

- $50 OFF ON ANY LION PURCHASE FROM NOW TO 12/31/24

```
# ADD TO ORDER DETAILS WITH DISCOUNT APPLIED
INSERT INTO OrderDetails (OrderID, ProductID, Quantity, ProductPrice, Subtotal)
VALUES (
    (SELECT OrderID FROM Orders WHERE OrderNumber = 'ORD10001'),
    6,
    1,
    5000.00,
    5000.00 - (SELECT DiscountValue
               FROM Discounts
               WHERE ProductID = 6 AND CategoryID = 1)
);

# UPDATE TOTAL IN ORDERS TO REFLECT DISCOUNT
UPDATE Orders
SET Total = Total - (SELECT DiscountValue
                     FROM Discounts
                     WHERE ProductID = 6 AND CategoryID = 1)
WHERE OrderNumber = 'ORD10001';
```

Figure 7. The Back-End: Discounts

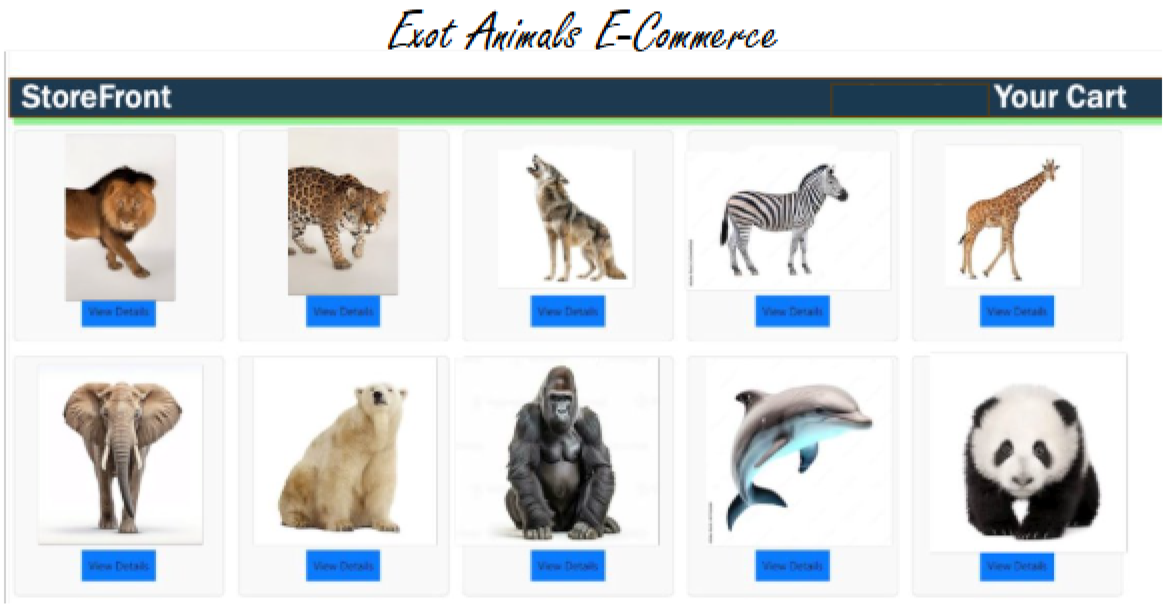
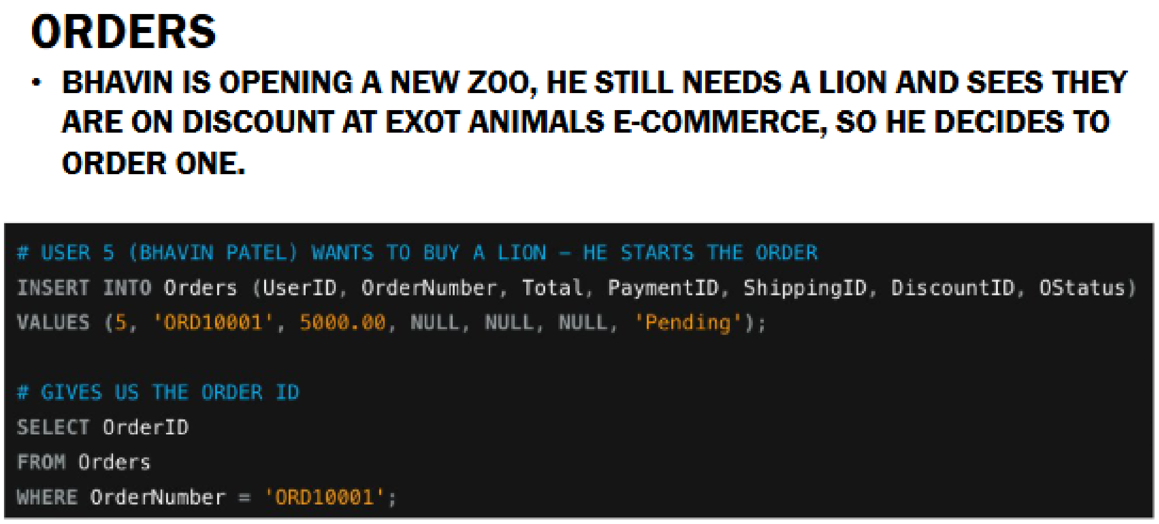


Figure 8. The Front-End: Storefront of the website, with options to add animals to your cart



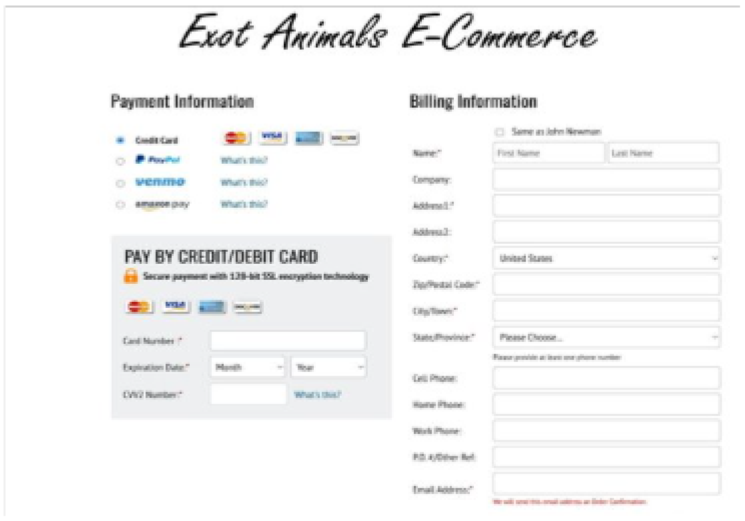
ORDERS

- **BHAVIN IS OPENING A NEW ZOO, HE STILL NEEDS A LION AND SEES THEY ARE ON DISCOUNT AT EXOT ANIMALS E-COMMERCE, SO HE DECIDES TO ORDER ONE.**

```
# USER 5 (BHAVIN PATEL) WANTS TO BUY A LION - HE STARTS THE ORDER
INSERT INTO Orders (UserID, OrderNumber, Total, PaymentID, ShippingID, DiscountID, OStatus)
VALUES (5, 'ORD10001', 5000.00, NULL, NULL, NULL, 'Pending');

# GIVES US THE ORDER ID
SELECT OrderID
FROM Orders
WHERE OrderNumber = 'ORD10001';
```

Figure 9. The Back-End: Inserting a new order



Exot Animals E-Commerce

Payment Information

- Credit Card
- PayPal What's this?
- venmo What's this?
- amazon pay What's this?

PAY BY CREDIT/DEBIT CARD

Secure payment with 128-bit SSL encryption technology

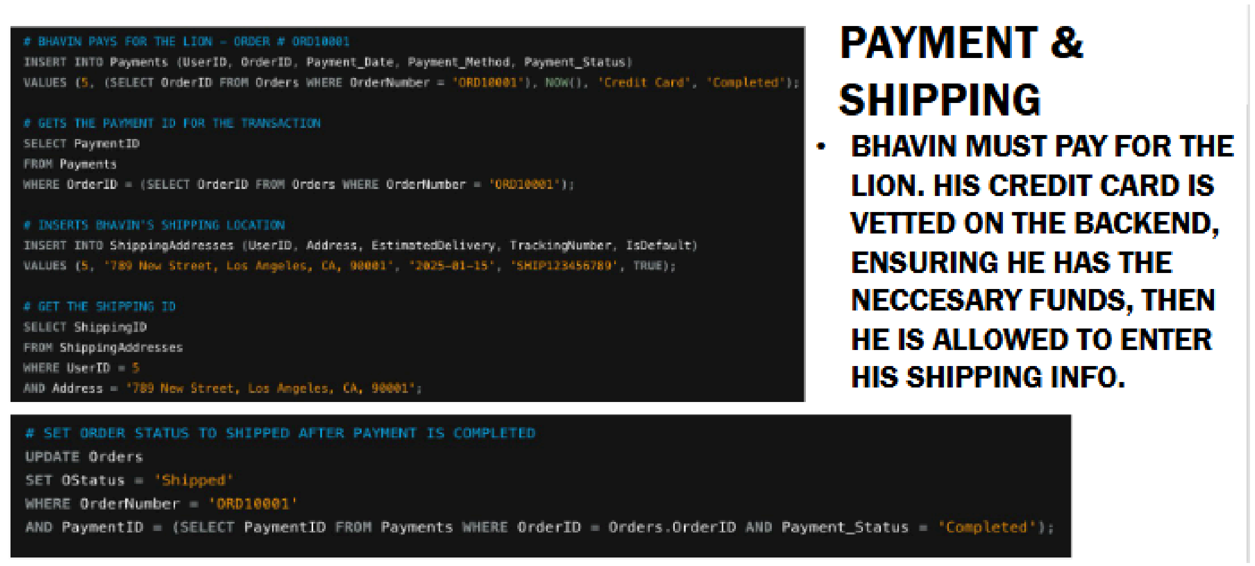
Card Number: *		
Expiration Date:*	Month	Year
CVV2 Number:*		What's this?

Billing Information

Same as John Newman

Name:*	First Name	Last Name
Company:		
Address1:*		
Address2:		
Country:*	United States	
Zip/Postal Code:*		
City/Town:*		
State/Province:*	Please Choose...	
	Please provide at least one phone number	
Cell Phone:		
Home Phone:		
Work Phone:		
P.O. #/Other Ref:		
Email Address:*		
	We will send this email address an Order Confirmation	

Figure 10. The Front-End: Payment input page, where input validation and error handling ensure accurate information for front and back-end success



```
# BHAVIN PAYS FOR THE LION - ORDER # ORD10001
INSERT INTO Payments (UserID, OrderID, Payment_Date, Payment_Method, Payment_Status)
VALUES (5, (SELECT OrderID FROM Orders WHERE OrderNumber = 'ORD10001'), NOW(), 'Credit Card', 'Completed');

# GETS THE PAYMENT ID FOR THE TRANSACTION
SELECT PaymentID
FROM Payments
WHERE OrderID = (SELECT OrderID FROM Orders WHERE OrderNumber = 'ORD10001');

# INSERTS BHAVIN'S SHIPPING LOCATION
INSERT INTO ShippingAddresses (UserID, Address, EstimatedDelivery, TrackingNumber, IsDefault)
VALUES (5, '789 New Street, Los Angeles, CA, 90001', '2025-01-15', 'SHIP123456789', TRUE);

# GET THE SHIPPING ID
SELECT ShippingID
FROM ShippingAddresses
WHERE UserID = 5
AND Address = '789 New Street, Los Angeles, CA, 90001';
```

PAYMENT & SHIPPING

- BHAVIN MUST PAY FOR THE LION. HIS CREDIT CARD IS VETTED ON THE BACKEND, ENSURING HE HAS THE NECCESARY FUNDS, THEN HE IS ALLOWED TO ENTER HIS SHIPPING INFO.

```
# SET ORDER STATUS TO SHIPPED AFTER PAYMENT IS COMPLETED
UPDATE Orders
SET OStatus = 'Shipped'
WHERE OrderNumber = 'ORD10001'
AND PaymentID = (SELECT PaymentID FROM Payments WHERE OrderID = Orders.OrderID AND Payment_Status = 'Completed');
```

Figure 11. The Back-End: Input Validation and Error Handling

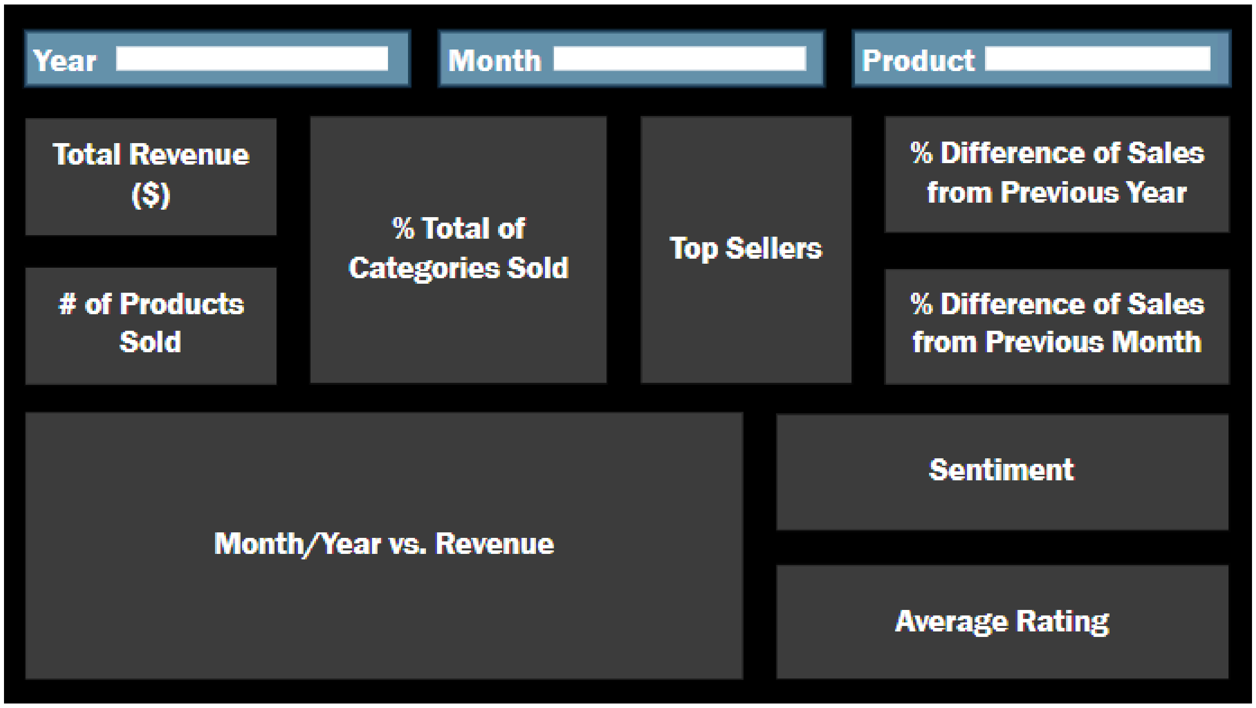

Year
Month
Product
Total Revenue ($)
% Total of Categories Sold
Top Sellers
% Difference of Sales from Previous Year
# of Products Sold
% Difference of Sales from Previous Month
Month/Year vs. Revenue
Sentiment
Average Rating

Figure 12. The Back End: Interactive dashboard built for administration and analytics

CONCLUSION

The development of the Exot Animal E-Commerce platform was created to be an integration of ethical considerations, advanced database design, and user-friendly functionality to address the challenges of exotic animal distribution. By adhering to humane, legal, and safety standards, the platform offers a transparent marketplace for consumers while equipping administrators with tools for effective inventory management and data-driven decision-making. Through the use of CRUD and complex queries, triggers, stored procedures, and a robust user interface, the platform delivers seamless operational efficiency and valuable business insights. This project highlights the potential of technology to not only solve logistical challenges but also promote responsible commerce, setting a strong precedent for innovation in the e-commerce space.