LUMEN

Report of Yaragala Vikram

Practice Assignment - SQL Basic

Attempt 1 | Submitted on Feb 20 2025 11:03:43

Programming Lab Exercise Result

Powered by UNEXT LABS

Practice Assignment - SQL Basic

Status: graded Attempts: 1/3 taken Validity: Feb 18 2025 - Feb 28 2025 Time Limit: 120 mins Marks: 100

Languages: dbms , dbms , dbms , dbms , dbms Archetypes: mysql_large_dataset , mysql_large_dataset , mysql_large_dataset , mysql_large_dataset , mysql_large_dataset

Overall Performance

107 mins 51 sec
Time Taken to Complete
Start: Feb 20 2025 10:51:34
End: Feb 20 2025 11:03:43

5/5
Projects Attempted
You have attempted 5 projects

100.00/ 100
Exercise Marks
You have received 100.00%

Passed
Exercise Status
Percentage for passing - 60%

Project Performance

1. Extract Domain
Level: Medium DBMS > MYSQL_LARGE_DATASET
20.0/20 marks

2. Customer Loyalty Levels in Retail
Level: Medium DBMS > MYSQL_LARGE_DATASET
20.0/20 marks

3. Classify Orders Based on Amount
Level: Difficult DBMS > MYSQL_LARGE_DATASET
20.0/20 marks

4. Retreive Orders
Level: Easy DBMS > MYSQL_LARGE_DATASET
20.0/20 marks

5. Transaction Details
Level: Difficult DBMS > MYSQL_LARGE_DATASET
20.0/20 marks

Project 1: Extract Domain

Marks	
Compilation: 0%	0
User Black Box: 0%	0
Evaluator Black Box: 0%	0
User White Box: 0%	0
Evaluator White Box: 0%	0
Total Marks Received	**20.00**

Plagiarism- Not Available

Code Quality- Not Available

Final Marks Received	20.00

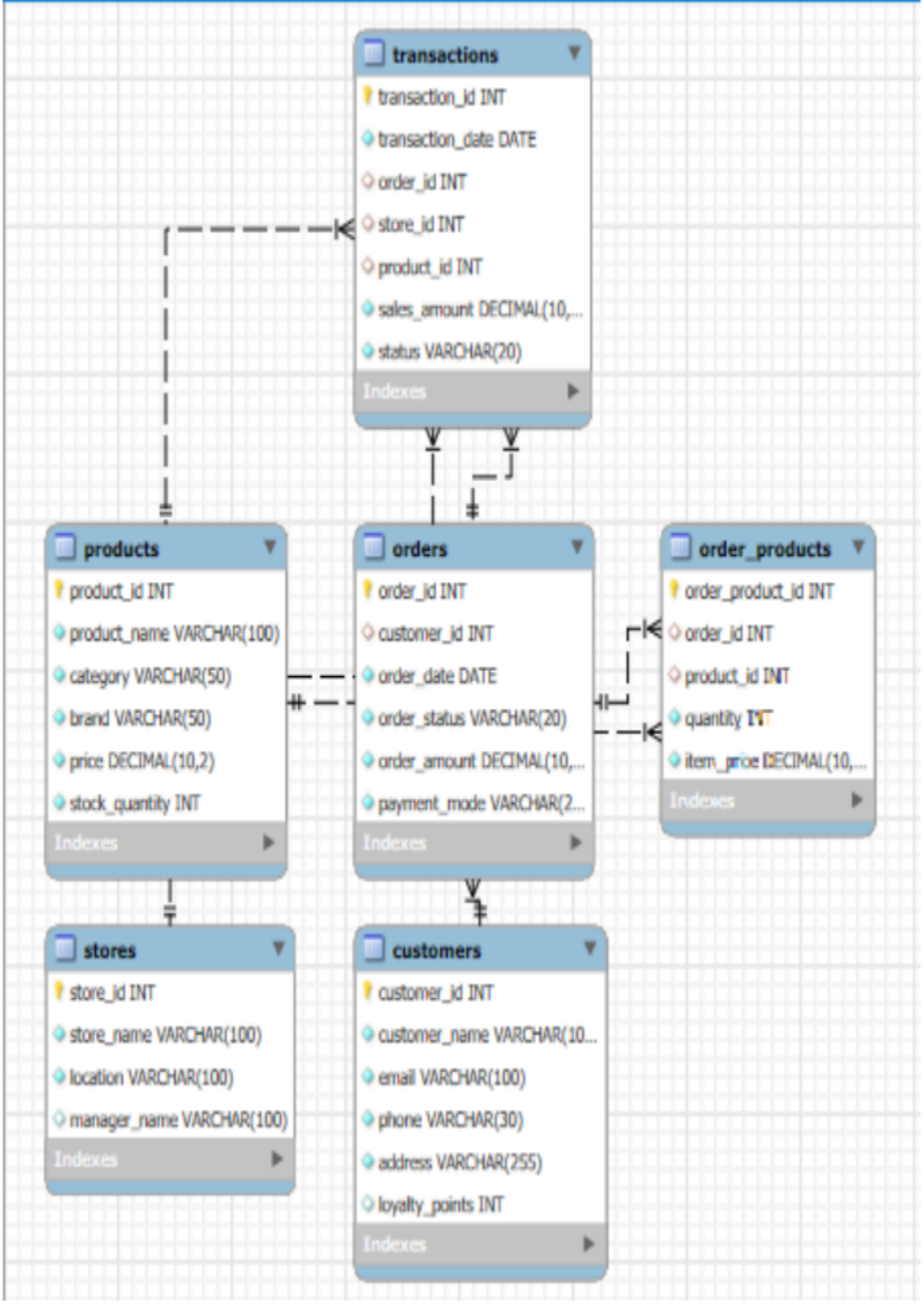
Test Cases
User Black Box
Evaluator Black Box
User White Box
Evaluator White Box

Problem Statement

The retail marketing team is preparing a campaign to send special birthday wishes to random customers. They want to target customers whose email address contains the string "on@" and ends with the ".net" domain. For each of these customers, they need to retrieve the following details:

- `customer_name`
- `email`
- `address`

Instructions


transactions
transaction_id INT
transaction_date DATE
order_id INT
store_id INT
product_id INT
sales_amount DECIMAL(10,...
status VARCHAR(20)
Indexes
products
product_id INT
product_name VARCHAR(100)
category VARCHAR(50)
brand VARCHAR(50)
price DECIMAL(10,2)
stock_quantity INT
Indexes
orders
order_id INT
customer_id INT
order_date DATE
order_status VARCHAR(20)
order_amount DECIMAL(10,...
payment_mode VARCHAR(2...
Indexes
order_products
order_product_id INT
order_id INT
product_id INT
quantity INT
item_price DECIMAL(10,...
Indexes
stores
store_id INT
store_name VARCHAR(100)
location VARCHAR(100)
manager_name VARCHAR(100)
Indexes
customers
customer_id INT
customer_name VARCHAR(10...
email VARCHAR(100)
phone VARCHAR(30)
address VARCHAR(255)
loyalty_points INT
Indexes

Example

Note: Provide table name in uppercase in your query(Eg.,CUSTOMERS)

customer_name	email	address
Joel Rodriguez	hickmanjon@example.net	706 Bailey Ville East Gilbert, WA 79541
Scott Davis	fanderson@example.net	72190 William Manor Breannastad, NH 48419
Daniel Morrow	antoniowilson@example.net	2252 Becker Burgs Suite 724 New Anthonybury, SC 61840
Phillip Hansen	sandramorton@example.net	50047 Matthew Loop Suite 326 North Jonathanburgh, WY 78787
Jeffrey Medina	rrichardson@example.net	29789 Martinez Crossing Port Eric, PR 54091
Taylor Gray	karenharrison@example.net	145 Rita Views Port Jeffrey, TX 13246

Submitted Code

```
select customer_name,email,address from CUSTOMERS
where email like '%on@%' and email like '%.net';
```

Project 2: Customer Loyalty Levels in Retail

Marks	
Compilation: 0%	0
User Black Box: 0%	0
Evaluator Black Box: 0%	0
User White Box: 0%	0
Evaluator White Box: 0%	0
Total Marks Received	**20.00**
Plagiarism- Not Available	
Code Quality- Not Available	
Final Marks Received	**20.00**

Test Cases
User Black Box
Evaluator Black Box
User White Box
Evaluator White Box

Problem Statement

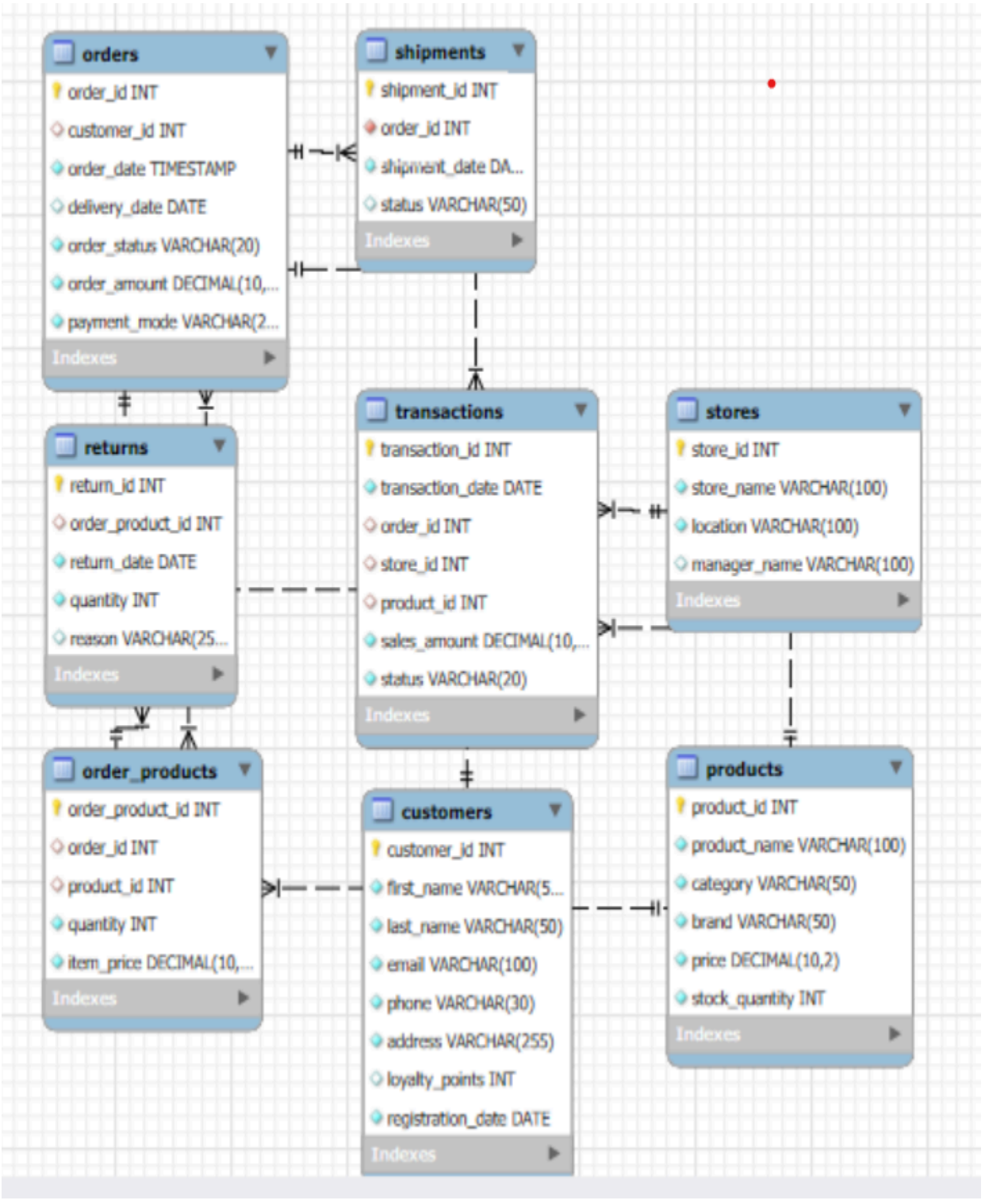
Customer loyalty is critical for business success. Customers accumulate **loyalty points** based on their transactions, which determine their loyalty level. Write a query to classify customers into three loyalty levels based on their `loyalty_points`:

- **Gold**: Customers with 400 or more loyalty points. These customers are considered premium customers and receive special benefits.
- **Silver**: Customers with loyalty points between 200 and 399. These customers are considered regular customers with some benefits.
- **Bronze**: Customers with 199 or less loyalty points. These customers are new or infrequent shoppers.

The query should retrieve:

- `customer_id`,
- `first_name`,
- `last_name`,
- `loyalty_points`, and
- the derived **loyalty_level** (Gold, Silver, Bronze) for all customers whose last name ends with **son**.

Instructions



Example

customer_id	first_name	last_name	loyalty_points	loyalty_level
87	Breanna	Johnson	445	Gold
67	Lori	Jackson	403	Gold
23	Donna	Ellison	269	Silver
52	Kelly	Johnson	212	Silver
6	Dennis	Anderson	47	Bronze
12	Sandy	Simpson	6	Bronze

Note : Provide starting letter of table name in uppercase in your query(Eg.,Customers)

Submitted Code

```
select customer_id,first_name,last_name,loyalty_points,
case
    when loyalty_points>400 then 'Gold'
    when loyalty_points between 200 and 399 then 'Silver'
    when loyalty_points<199 then 'Bronze'
end as loyalty_level
from Customers
where last_name like '%son'
order by loyalty_points desc;
```

Project 3: Classify Orders Based on Amount

Marks	
Compilation: 0%	0
User Black Box: 0%	0
Evaluator Black Box: 0%	0
User White Box: 0%	0
Evaluator White Box: 0%	0
Total Marks Received	**20.00**
Plagiarism- Not Available	
Code Quality- Not Available	
Final Marks Received	**20.00**

Test Cases
User Black Box
Evaluator Black Box
User White Box
Evaluator White Box

Problem Statement

You are analyzing order data for specific customers in a retail business. Retrieve information from the `Orders` table for all orders placed by customers with `customer_id` 95 or 2.

For each order, the query should include the following columns:

1. **`order_id`**: The unique identifier for the order.
2. **`customer_id`**: The ID of the customer who placed the order.
3. **`order_amount`**: The total monetary value of the order.
4. **`order_date`**: The date when the order was placed.
5. **`order_status`**: The current status of the order (e.g., "Pending", "Delivered").
6. A **derived column named `order_category`** that categorizes each order based on the `order_amount`:
    - **High Value**: Orders with an amount of 4000 or more.
    - **Medium Value**: Orders with an amount between 2000 and 3999.
    - **Low Value**: Orders with an amount below 2000.

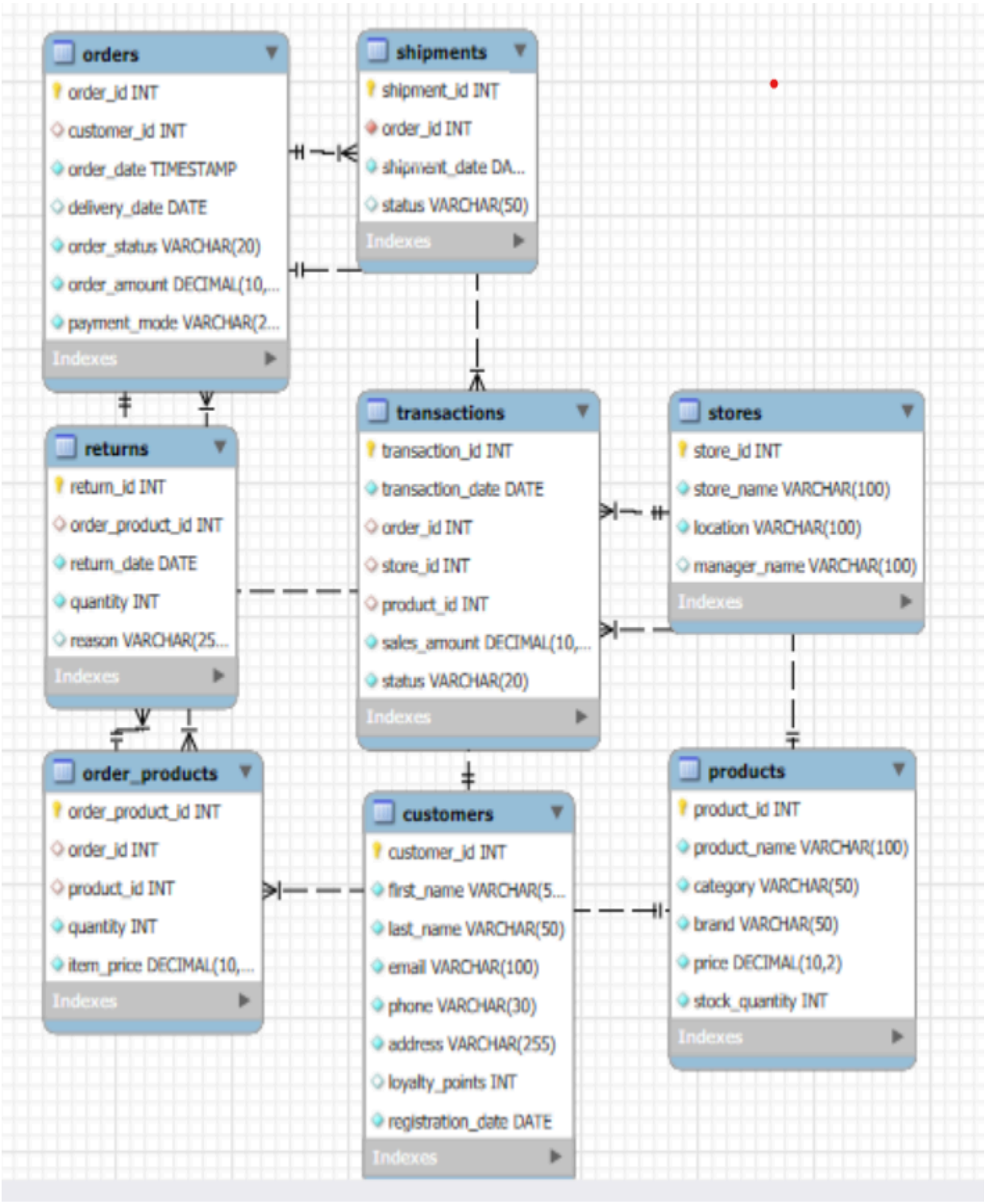
Additionally, filter the results to include only orders from the specified customers (95 and 2).

The output should be sorted:

1. **Primarily** in descending order of `order_amount` to prioritize higher-value orders.

2. **Secondarily** by `order_date` in ascending order for orders with the same value, showing the oldest orders first.

Instructions



Example

order_id	customer_id	order_amount	order_date	order_status	order_category
17	2	4897.6	2021-09-17T00:00:00.000Z	Completed	High Value
27	95	4377.57	2022-08-19T00:00:00.000Z	Delivered	High Value
260	2	3992.78	2024-02-07T00:00:00.000Z	Delivered	Medium Value
479	2	2447.73	2023-08-30T00:00:00.000Z	Completed	Medium Value
268	95	1449.51	2024-05-20T00:00:00.000Z	Delivered	Low Value
164	95	191.76	2023-05-05T00:00:00.000Z	Completed	Low Value

Note : Provide starting letter of table name in uppercase in your query(Eg.,Customers)

Submitted Code

```
        select order_id,customer_id,order_amount,order_date,order_status,
case
   WHEN order_amount>4000 THEN 'High Value'
   when order_amount between 2000 and 3999 then 'Medium Value'
   when order_amount<2000 then 'Low Value'
end as order_category
from Orders
where customer_id in (95,2)
order by order_amount desc,order_date asc;
```

Project 4: Retreive Orders

Marks	
Compilation: 0%	0
User Black Box: 0%	0
Evaluator Black Box: 0%	0
User White Box: 0%	0
Evaluator White Box: 0%	0
Total Marks Received	**20.00**
Plagiarism- Not Available	
Code Quality- Not Available	
Final Marks Received	**20.00**

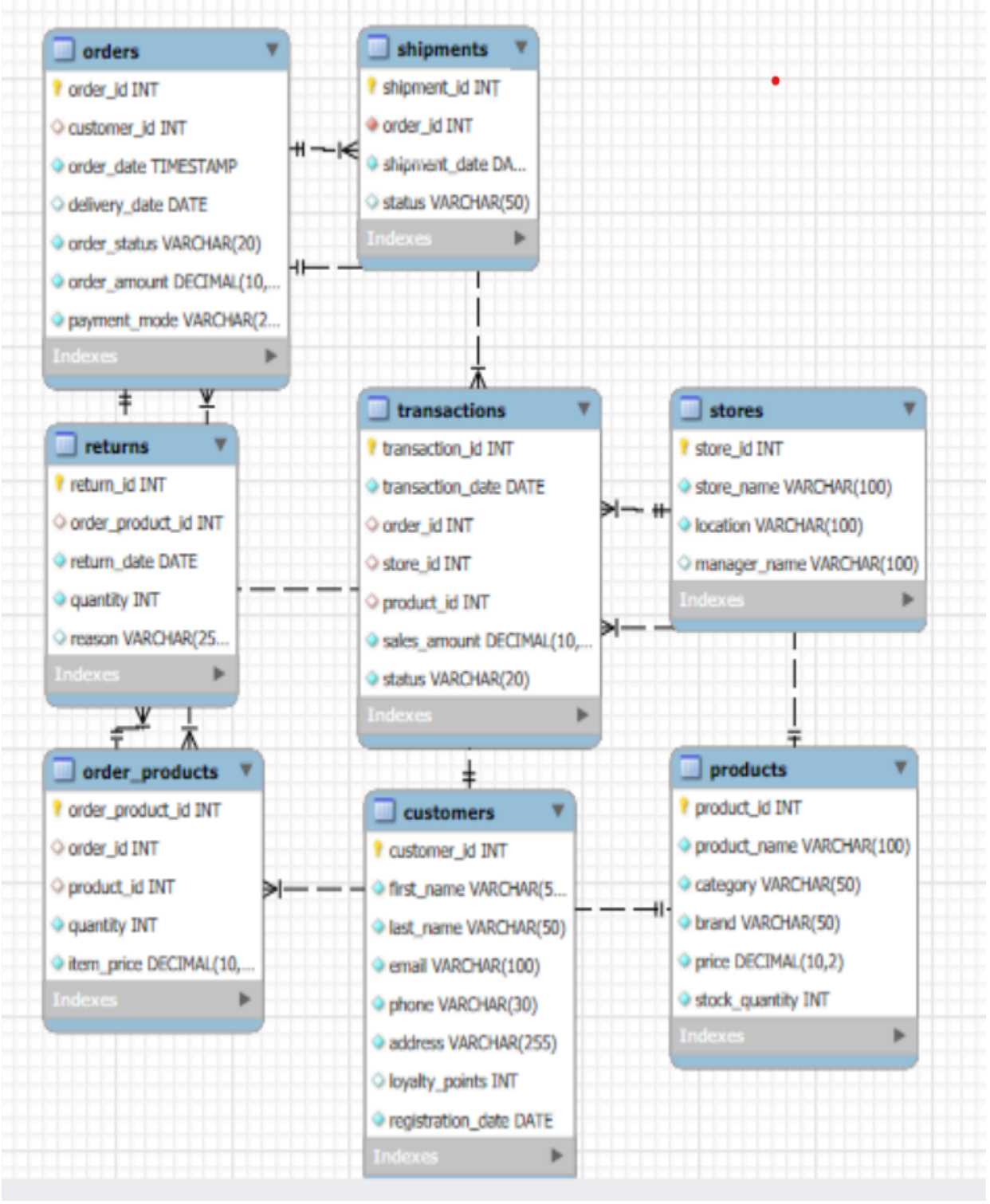
Test Cases
User Black Box
Evaluator Black Box
User White Box
Evaluator White Box

Problem Statement

The retail management team needs to retrieve details of orders for analysis. They are specifically looking for orders placed by customers with IDs 95, 2, or 51, and that have an order status of either 'Delivered' or 'Shipped'. Additionally, they want the results to be sorted as follows:

1. Orders should be ordered primarily by the `order_date` in descending order, so the most recent orders appear first.
2. If multiple orders were placed on the same date, they should be further sorted by the `order_amount` in ascending order.

Instructions


orders
order_id INT
customer_id INT
order_date TIMESTAMP
delivery_date DATE
order_status VARCHAR(20)
order_amount DECIMAL(10,...
payment_mode VARCHAR(2...
Indexes
shipments
shipment_id INT
order_id INT
shipment_date DA...
status VARCHAR(50)
Indexes
returns
return_id INT
order_product_id INT
return_date DATE
quantity INT
reason VARCHAR(25...
Indexes
transactions
transaction_id INT
transaction_date DATE
order_id INT
store_id INT
product_id INT
sales_amount DECIMAL(10,...
status VARCHAR(20)
Indexes
stores
store_id INT
store_name VARCHAR(100)
location VARCHAR(100)
manager_name VARCHAR(100)
Indexes
order_products
order_product_id INT
order_id INT
product_id INT
quantity INT
item_price DECIMAL(10,...
Indexes
customers
customer_id INT
first_name VARCHAR(5...
last_name VARCHAR(50)
email VARCHAR(100)
phone VARCHAR(30)
address VARCHAR(255)
loyalty_points INT
registration_date DATE
Indexes
products
product_id INT
product_name VARCHAR(100)
category VARCHAR(50)
brand VARCHAR(50)
price DECIMAL(10,2)
stock_quantity INT
Indexes

Example

Note: Provide starting letter of table name in uppercase in your query(Eg.,Customers).

order_id	customer_id	order_amount	order_date	order_status
268	95	1449.51	2024-05-20T00:00:00.000Z	Delivered
260	2	3992.78	2024-02-07T00:00:00.000Z	Delivered
163	51	4468.13	2022-09-24T00:00:00.000Z	Delivered
27	95	4377.57	2022-08-19T00:00:00.000Z	Delivered
357	51	3605.05	2022-04-05T00:00:00.000Z	Shipped
407	51	3427.52	2022-04-01T00:00:00.000Z	Shipped

Submitted Code

```
select order_id,customer_id,order_amount,order_date,order_status
from Orders
where customer_id in(95,2,51) and order_status in('Delivered','Shipped')
order by order_date desc,order_amount asc;
```

Project 5: Transaction Details

Marks	
Compilation: 0%	0
User Black Box: 0%	0
Evaluator Black Box: 0%	0
User White Box: 0%	0
Evaluator White Box: 0%	0
Total Marks Received	**20.00**
Plagiarism- Not Available	
Code Quality- Not Available	
Final Marks Received	**20.00**

Test Cases
User Black Box
Evaluator Black Box
User White Box
Evaluator White Box

Problem Statement

A retail chain is conducting a review of certain **pending transactions** that are under manual verification. The management team wants to classify these transactions based on their **sales amount** into categories to prioritize processing:

- **High Sales** for transactions with a sales amount greater than $1000.
- **Moderate Sales** for transactions between $500 and $1000.
- **Low Sales** for transactions below $500.

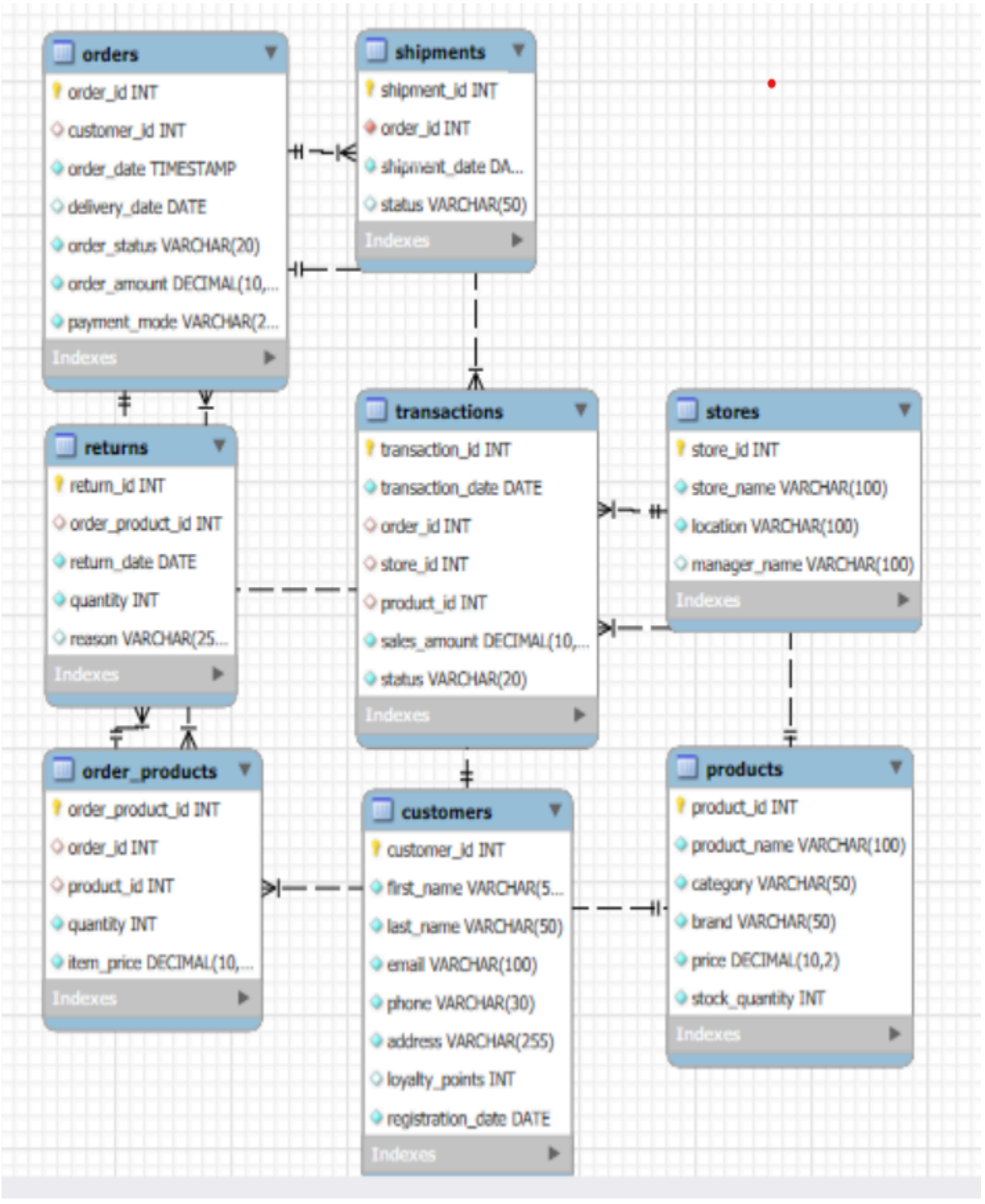
The focus is only on the **pending transactions** with specific transaction IDs (185, 16, and 210) to streamline the investigation process. The results must display the transaction details and be sorted to show the highest-priority transactions first.

Question:

Write a query to display details of pending transactions with transaction IDs 185, 16, and 210. For each transaction, categorize the sales amount into **High Sales**, **Moderate Sales**, or **Low Sales** and sort the results:

1. First by **Sales Category** in the order: High > Moderate > Low.
2. Then by **Sales Amount** in descending order within each category.

Instructions



Example

transaction_id	sales_amount	sales_category
185	1997.15	High Sales
16	869.26	Moderate Sales
210	848.47	Moderate Sales

Note: Provide starting letter of table name in uppercase in your query(Eg.,Customers)

Submitted Code

```
        select transaction_id,sales_amount,
case
   when sales_amount>1000 then 'High Sales'
   when sales_amount between 500 and 1000 then 'Moderate Sales'
   else 'Low Sales'
end as sales_category
from Transactions
where transaction_id in(185,16,210)
order by sales_amount desc;
```