PySpark Expert Level Problem Set - E-commerce Dataset

🎯 BASIC QUERIES & AGGREGATIONS

1. Customer Analysis

- **Q1.1**: Find the top 10 states with the highest number of customers
- **Q1.2**: Calculate the average number of orders per customer
- **Q1.3**: Find customers who have made more than 3 orders

2. Product Performance

- **Q2.1**: Identify the top 20 best-selling products by quantity
- **Q2.2**: Find the most expensive product in each category
- **Q2.3**: Calculate the average product weight by category

3. Order Statistics

- **Q3.1**: Calculate total revenue by month and year
- **Q3.2**: Find the distribution of order statuses
- **Q3.3**: Calculate the average order value

🔄 INTERMEDIATE JOINS & TRANSFORMATIONS

4. Multi-Table Analysis

- **Q4.1**: Join orders with customers to find the top 5 cities by total order value
- **Q4.2**: Create a complete order summary including customer details, payment info, and items
- **Q4.3**: Find sellers who have sold products in more than 5 different states

5. Payment Analysis

- **Q5.1**: Calculate the most popular payment method by state
- **Q5.2**: Find orders with multiple payment methods and their success rates
- **Q5.3**: Analyze payment installment patterns by order value ranges

6. Geographic Analysis

- **Q6.1**: Calculate the average delivery time by state
- **Q6.2**: Find the distance between seller and customer locations (using lat/lng)
- **Q6.3**: Identify the most active shipping routes (seller_state -> customer_state)

🚀 ADVANCED WINDOW FUNCTIONS

7. Ranking & Percentiles

- **Q7.1**: Rank customers by their total spending within each state
- **Q7.2**: Find the top 3 products by revenue in each category
- **Q7.3**: Calculate the 25th, 50th, and 75th percentile of order values by month

8. Running Totals & Moving Averages

- **Q8.1**: Calculate cumulative revenue by month for each year
- **Q8.2**: Compute 3-month moving average of order counts
- **Q8.3**: Find the percentage contribution of each month to yearly revenue

9. Lead/Lag Analysis

- **Q9.1**: Calculate month-over-month growth rate in revenue
- **Q9.2**: Find the time gap between consecutive orders for each customer
- **Q9.3**: Identify seasonal trends by comparing same month across different years

🎓 EXPERT LEVEL CHALLENGES

10. Customer Segmentation

- **Q10.1**: Implement RFM (Recency, Frequency, Monetary) analysis
- **Q10.2**: Create customer lifetime value calculation
- **Q10.3**: Build a customer churn prediction feature set

11. Advanced Business Intelligence

- **Q11.1**: Create a seller performance dashboard metrics
- **Q11.2**: Build a product recommendation system based on co-purchase patterns
- **Q11.3**: Implement market basket analysis to find frequently bought together items

12. Time Series Analysis

- **Q12.1**: Detect anomalies in daily order patterns
- **Q12.2**: Calculate year-over-year growth rates for different metrics
- **Q12.3**: Implement inventory turnover analysis by product category

🏆 MASTER LEVEL PROJECTS

13. Complex Multi-Dimensional Analysis

- **Q13.1**: Build a comprehensive seller rating system considering multiple factors
- **Q13.2**: Create a dynamic pricing analysis comparing product prices across different regions

- **Q13.3**: Implement a supply chain optimization analysis

14. Machine Learning Feature Engineering

- **Q14.1**: Create features for predicting delivery delays
- **Q14.2**: Build features for customer segmentation clustering
- **Q14.3**: Generate features for product demand forecasting

15. Real-Time Analytics Simulation

- **Q15.1**: Implement sliding window analytics for real-time order monitoring
- **Q15.2**: Create streaming aggregations for live business metrics
- **Q15.3**: Build a real-time recommendation engine data pipeline

📊 PERFORMANCE OPTIMIZATION CHALLENGES

16. Query Optimization

- **Q16.1**: Optimize a complex join query involving all tables
- **Q16.2**: Implement efficient partitioning strategies for large datasets
- **Q16.3**: Create broadcast joins for dimension tables

17. Memory Management

- **Q17.1**: Handle skewed data in state-wise aggregations
- **Q17.2**: Implement efficient caching strategies for frequently accessed data
- **Q17.3**: Optimize window functions for large datasets

🎯 SUGGESTED IMPLEMENTATION ORDER

Phase 1: Foundation (Questions 1-3)

Focus on basic DataFrame operations and simple aggregations

Phase 2: Integration (Questions 4-6)

Master joins and multi-table operations

Phase 3: Advanced Analytics (Questions 7-9)

Deep dive into window functions and analytical operations

Phase 4: Expert Implementation (Questions 10-12)

Build complex business logic and advanced analytics

Phase 5: Mastery (Questions 13-15)

Implement comprehensive solutions and ML features

Phase 6: Optimization (Questions 16-17)

Focus on performance tuning and production readiness

💡 ADDITIONAL CHALLENGES

Bonus Questions:

1. **Data Quality**: Implement comprehensive data quality checks across all tables
2. **ETL Pipeline**: Create a complete ETL pipeline with error handling
3. **Testing**: Build unit tests for your PySpark transformations
4. **Documentation**: Create comprehensive documentation for your solutions

🔧 Technical Skills You'll Master

- **DataFrame Operations**: select, filter, groupBy, agg, orderBy
- **Joins**: inner, outer, left, right, anti, semi joins
- **Window Functions**: rank, dense_rank, row_number, lag, lead
- **Aggregations**: sum, avg, count, collect_list, collect_set
- **UDFs**: Custom functions for complex transformations
- **Performance**: Caching, partitioning, broadcasting
- **SQL Integration**: Mixing SQL with DataFrame API
- **Data Types**: Working with timestamps, decimals, arrays
- **Error Handling**: Null handling, data validation
- **Optimization**: Query plans, catalyst optimizer

📈 Success Metrics

By completing this problem set, you should be able to:

- Handle any real-world PySpark scenario
- Optimize queries for production workloads
- Design efficient data processing pipelines
- Implement complex business logic
- Debug and troubleshoot PySpark applications
- Build scalable analytics solutions

Happy Coding! 🚀