ADVENTIST UNIVERSITY OF CENTRAL AFRICA

MSDA9215: Big Data Analytics

TECHNICAL REPORT

Distributed Multi-Model Analytics for E-commerce Data

Student: Joseph Tuyambaze

Table of Contents

1. Summary

This report presents a comprehensive multi-model analytics system for e-commerce data, leveraging MongoDB (document model), HBase (wide-column model), and Apache Spark (distributed processing). The system handles:

- • 10,000 users with geographic and registration data
- • 5,000 products across 25 categories
- • 500,000 transactions totaling $470+ million
- • 2,000,000 website sessions spanning 90 days

Key achievements: MongoDB aggregation pipelines, optimized HBase schema with reverse-timestamp row keys, Spark batch processing for cleaning, recommendations, and cohort analysis, cross-database CLV integration, and meaningful visualizations.

2. System Architecture Overview

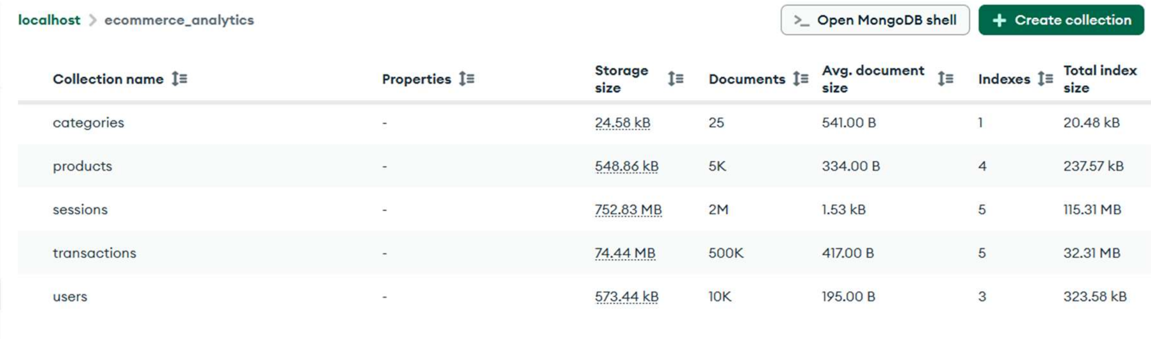
The system follows a polyglot persistence approach:

Technology	Data Stored	Justification
MongoDB	Users, Products, Transactions	Flexible schema for nested documents
HBase	Sessions (2M records)	Optimized for time-series
Apache Spark	Cross-source processing	Distributed processing and integration

3. Data Modeling and Storage

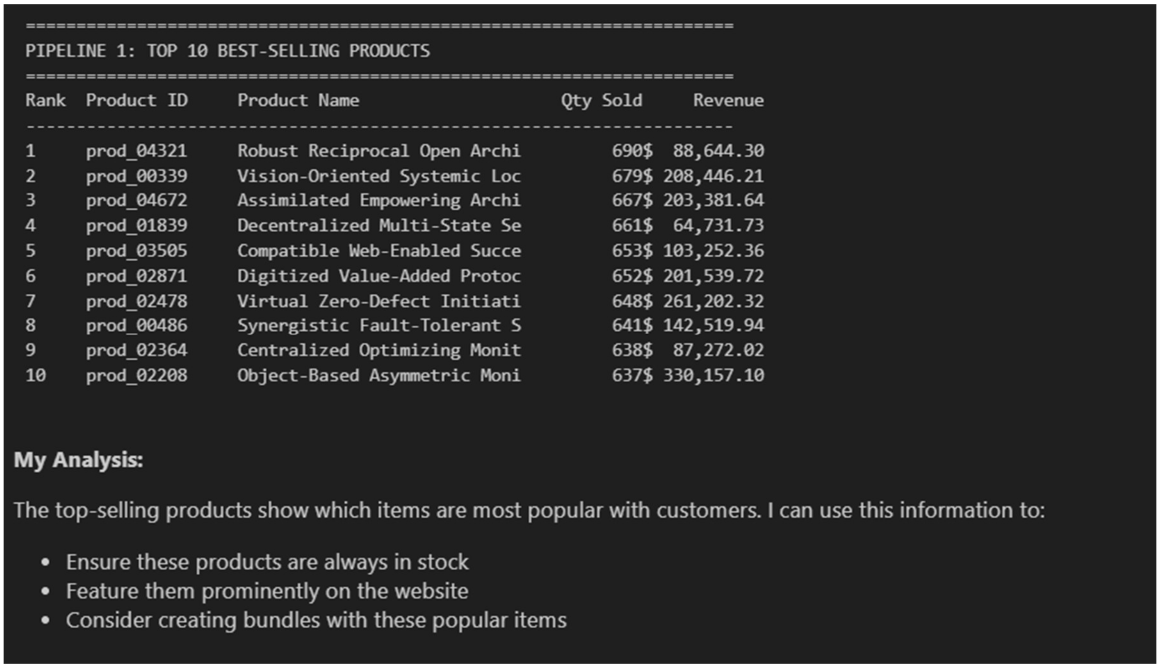
3.1 MongoDB Implementation

MongoDB database **'ecommerce_analytics'** contains:



Collection name	Properties	Storage size	Documents	Avg. document size	Indexes	Total index size
categories	-	24.58 kB	25	541.00 B	1	20.48 kB
products	-	548.86 kB	5K	334.00 B	4	237.57 kB
sessions	-	752.83 MB	2M	1.53 kB	5	115.31 MB
transactions	-	74.44 MB	500K	417.00 B	5	32.31 MB
users	-	573.44 kB	10K	195.00 B	3	323.58 kB

Pipeline Results samples

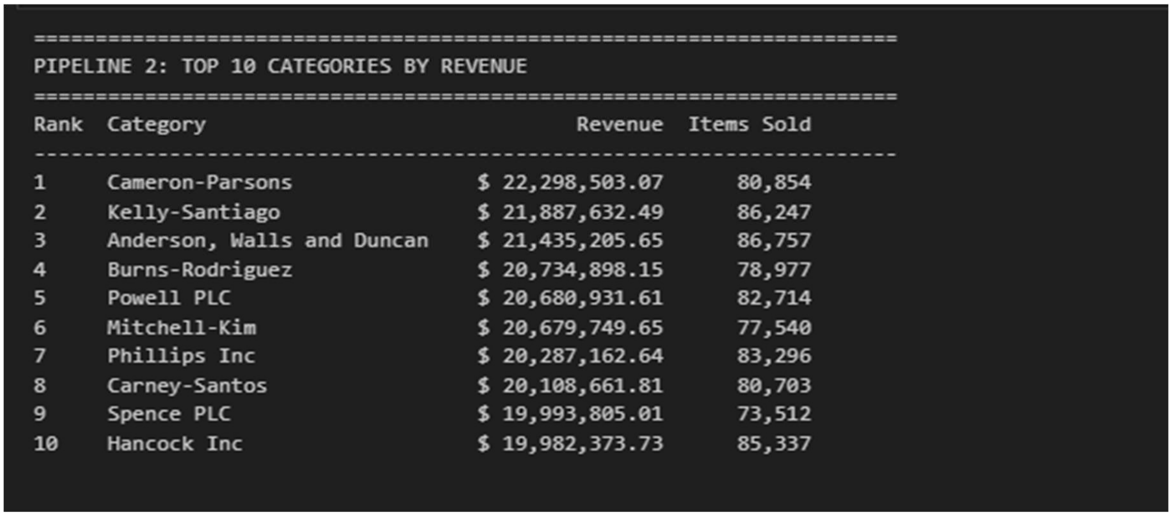


```
======================================================================
PIPELINE 1: TOP 10 BEST-SELLING PRODUCTS
======================================================================
Rank  Product ID     Product Name                    Qty Sold     Revenue
----------------------------------------------------------------------
1     prod_04321     Robust Reciprocal Open Archi         690$  88,644.30
2     prod_00339     Vision-Oriented Systemic Loc         679$ 208,446.21
3     prod_04672     Assimilated Empowering Archi         667$ 203,381.64
4     prod_01839     Decentralized Multi-State Se         661$  64,731.73
5     prod_03505     Compatible Web-Enabled Succe         653$ 103,252.36
6     prod_02871     Digitized Value-Added Protoc         652$ 201,539.72
7     prod_02478     Virtual Zero-Defect Initiati         648$ 261,202.32
8     prod_00486     Synergistic Fault-Tolerant S         641$ 142,519.94
9     prod_02364     Centralized Optimizing Monit         638$  87,272.02
10    prod_02208     Object-Based Asymmetric Moni         637$ 330,157.10
```

My Analysis:

The top-selling products show which items are most popular with customers. I can use this information to:

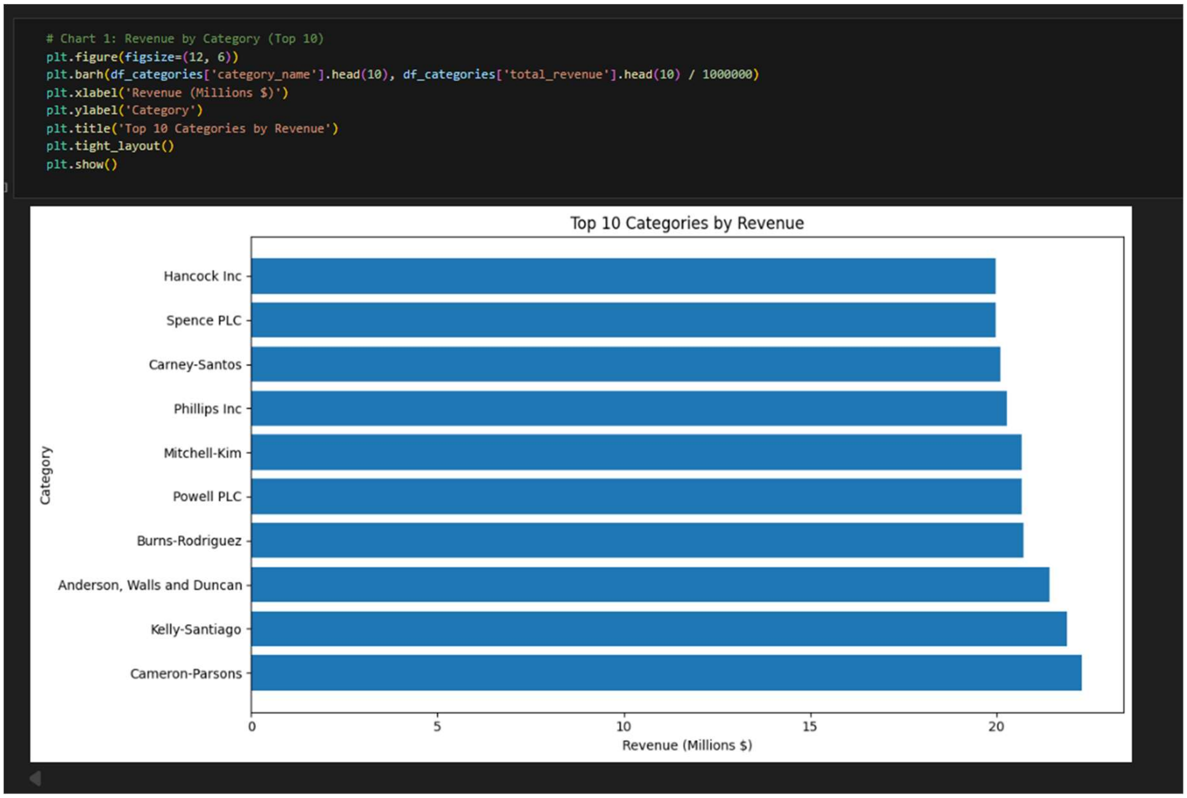
- Ensure these products are always in stock
- Feature them prominently on the website
- Consider creating bundles with these popular items



```
======================================================================
PIPELINE 2: TOP 10 CATEGORIES BY REVENUE
======================================================================
Rank  Category                              Revenue  Items Sold
----------------------------------------------------------------------
1     Cameron-Parsons              $ 22,298,503.07      80,854
2     Kelly-Santiago               $ 21,887,632.49      86,247
3     Anderson, Walls and Duncan   $ 21,435,205.65      86,757
4     Burns-Rodriguez              $ 20,734,898.15      78,977
5     Powell PLC                   $ 20,680,931.61      82,714
6     Mitchell-Kim                 $ 20,679,749.65      77,540
7     Phillips Inc                 $ 20,287,162.64      83,296
8     Carney-Santos                $ 20,108,661.81      80,703
9     Spence PLC                   $ 19,993,805.01      73,512
10    Hancock Inc                  $ 19,982,373.73      85,337
```

n.b: I have attached completed required pipelines in the “**mongodb_ecommerce_analytics.ipynb**”

Sample MongoDB Visualizations

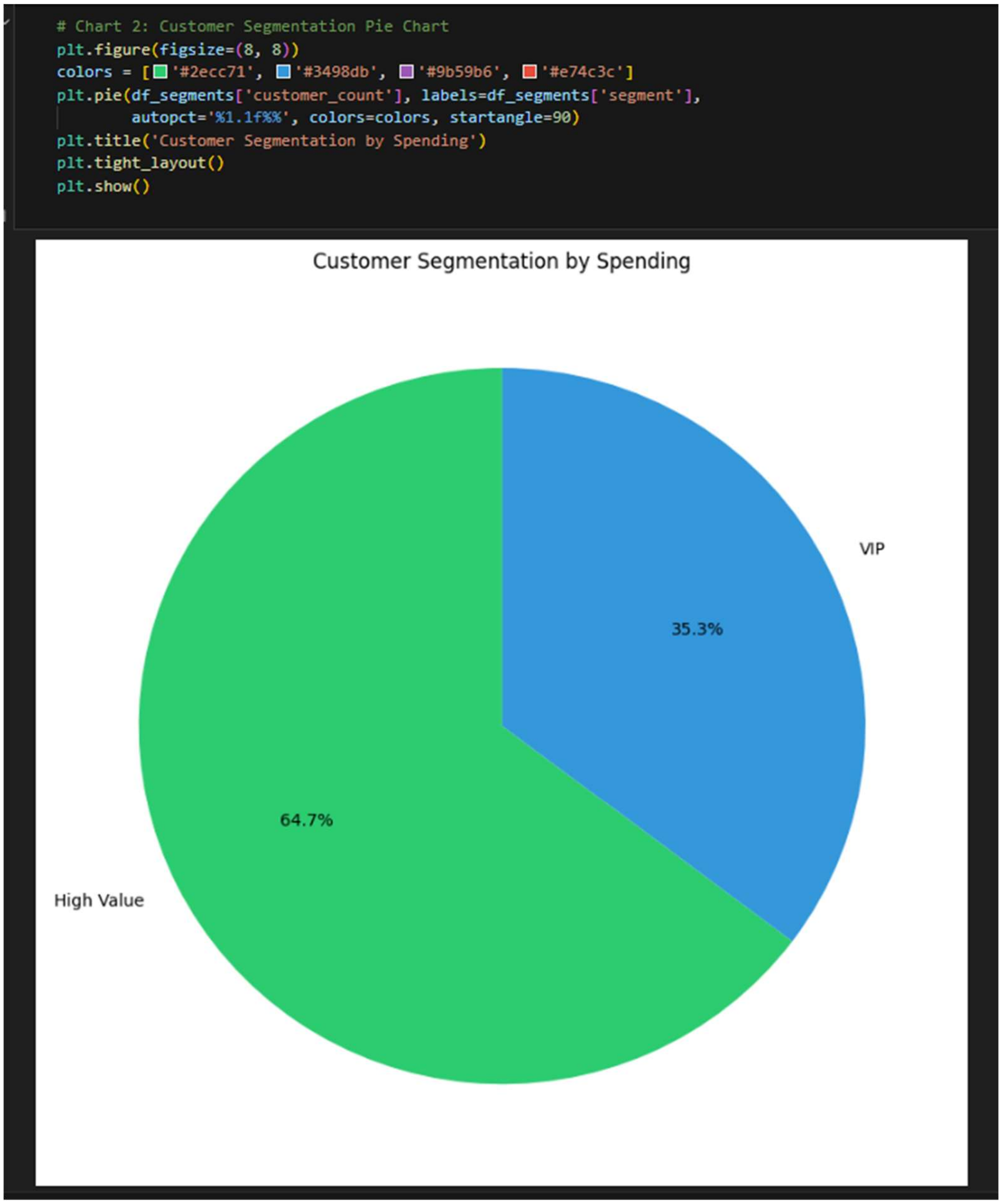


```
# Chart 1: Revenue by Category (Top 10)
plt.figure(figsize=(12, 6))
plt.barh(df_categories['category_name'].head(10), df_categories['total_revenue'].head(10) / 1000000)
plt.xlabel('Revenue (Millions $)')
plt.ylabel('Category')
plt.title('Top 10 Categories by Revenue')
plt.tight_layout()
plt.show()
```

Figure: Revenue by Category - Cameron-Parsons leads with $22.3M

Analysis: Cameron-Parsons leads revenue at $22.3M, followed closely by Kelly-Santiago ($21.9M) and Anderson, Walls and Duncan ($21.4M). The top 10 categories generate relatively similar revenue ($19-22M range), suggesting a well-diversified product portfolio with no single category dominating sales.

Customer Segmentation by Spending

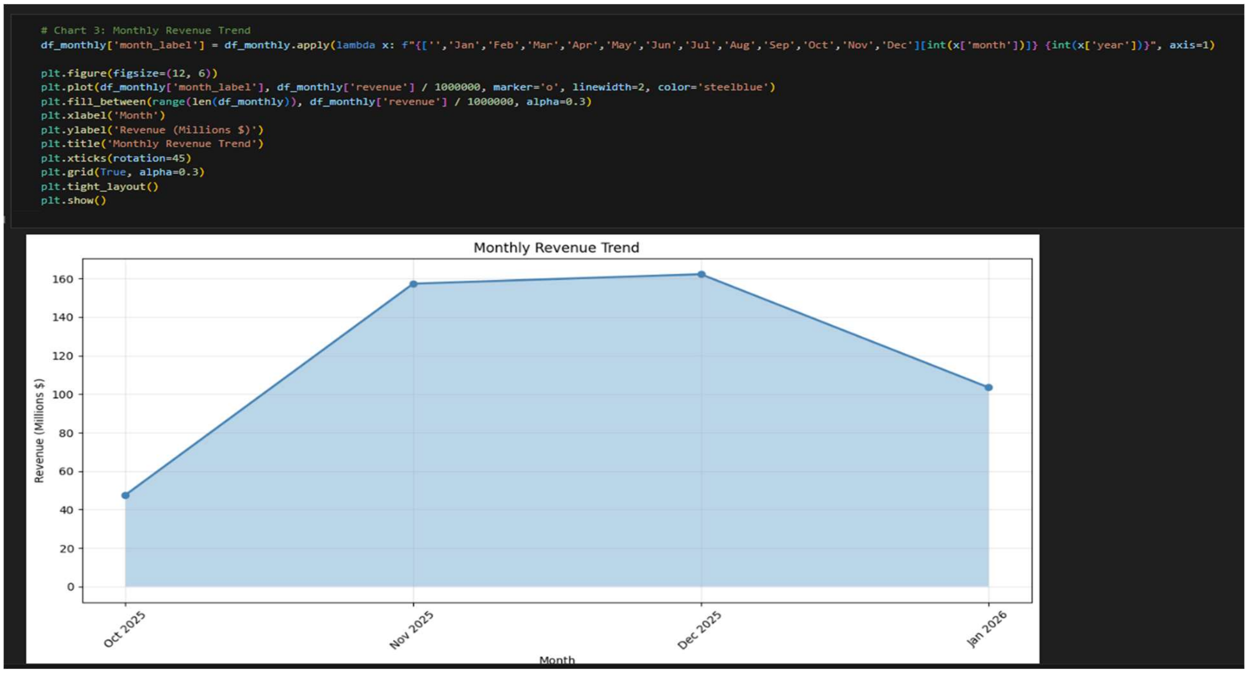


```python
# Chart 2: Customer Segmentation Pie Chart
plt.figure(figsize=(8, 8))
colors = ['#2ecc71', '#3498db', '#9b59b6', '#e74c3c']
plt.pie(df_segments['customer_count'], labels=df_segments['segment'],
        autopct='%1.1f%%', colors=colors, startangle=90)
plt.title('Customer Segmentation by Spending')
plt.tight_layout()
plt.show()
```

Figure: Customer Segmentation - VIP (35.3%) vs High Value (64.7%)

Analysis: VIP customers (35.3%) generate $196.7M with average spending of $55,757. A loyalty program for this segment could increase retention.

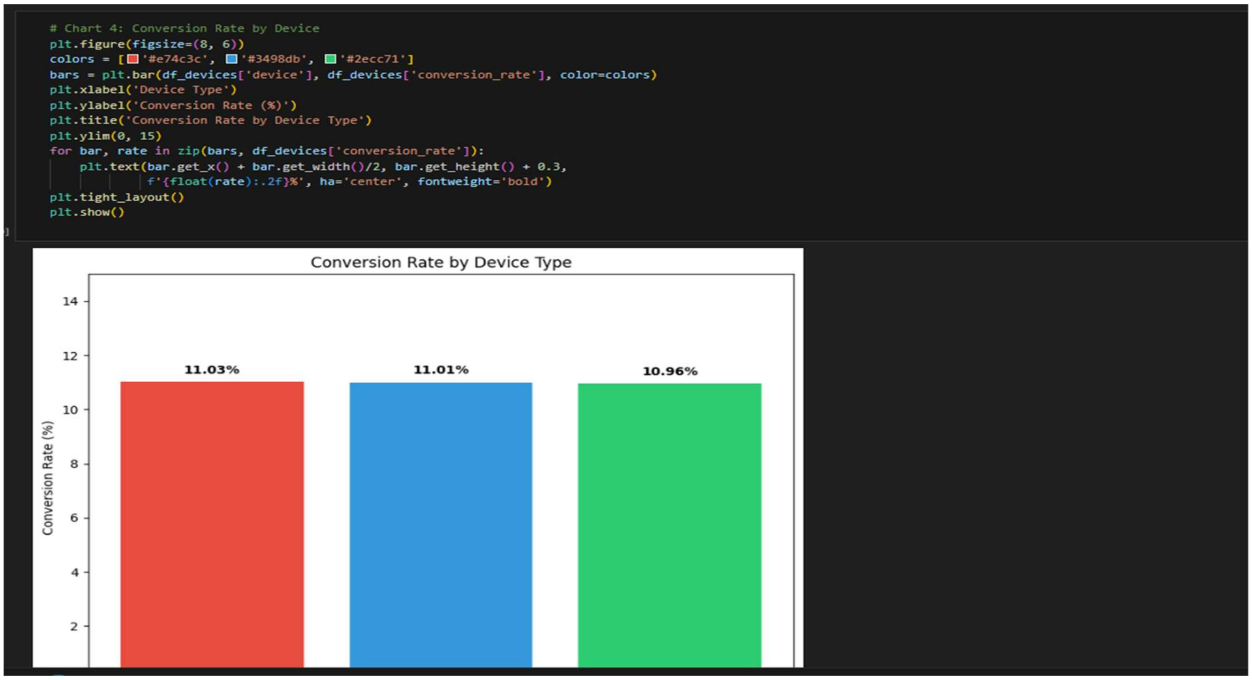
Monthly Revenue Trend



```
# Chart 3: Monthly Revenue Trend
df_monthly['month_label'] = df_monthly.apply(lambda x: f"{['','Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec'][int(x['month'])]} {int(x['year'])}", axis=1)

plt.figure(figsize=(12, 6))
plt.plot(df_monthly['month_label'], df_monthly['revenue'] / 1000000, marker='o', linewidth=2, color='steelblue')
plt.fill_between(range(len(df_monthly)), df_monthly['revenue'] / 1000000, alpha=0.3)
plt.xlabel('Month')
plt.ylabel('Revenue (Millions $)')
plt.title('Monthly Revenue Trend')
plt.xticks(rotation=45)
plt.grid(True, alpha=0.3)
plt.tight_layout()
plt.show()
```

Figure: Monthly Revenue Trend - Peak in December 2025 with $162M

Analysis: Revenue grew from $47.6M (Oct) to $162.3M (Dec), showing strong seasonal patterns. Inventory should anticipate this growth.



```
# Chart 4: Conversion Rate by Device
plt.figure(figsize=(8, 6))
colors = ['#e74c3c', '#3498db', '#2ecc71']
bars = plt.bar(df_devices['device'], df_devices['conversion_rate'], color=colors)
plt.xlabel('Device Type')
plt.ylabel('Conversion Rate (%)')
plt.title('Conversion Rate by Device Type')
plt.ylim(0, 15)
for bar, rate in zip(bars, df_devices['conversion_rate']):
    plt.text(bar.get_x() + bar.get_width()/2, bar.get_height() + 0.3,
             f'{float(rate):.2f}%', ha='center', fontweight='bold')
plt.tight_layout()
plt.show()
```

Figure: Device Conversion Rates - Mobile leads at 11.03%

Analysis: Mobile has highest conversion (11.03%). Continue investing in mobile UX optimization.

3.2 HBase Implementation

HBase was chosen for sessions because:

1. 2 million sessions - HBase handles billions efficiently,

2. Time-series data - HBase excels at time-based queries,

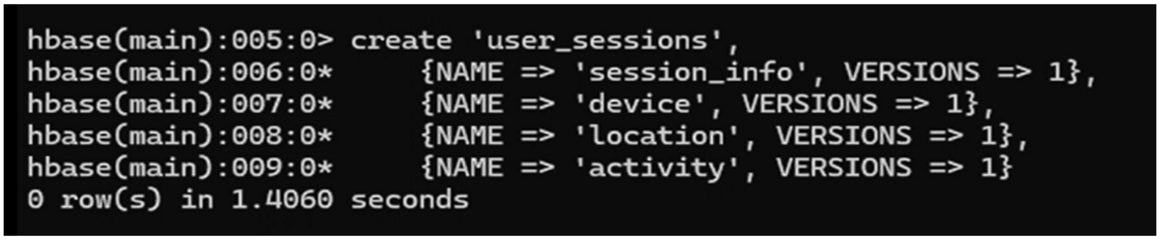
3. Fast user lookups - Row key design enables O(log n) access.

Column Family	Columns	Purpose
session_info	session_id, duration, referrer	Basic session data
device	type, os, browser	Device information
location	city, state, country, ip	Geographic data
activity	page_views, converted	User behavior

Row Key: user_id#reverse_timestamp (e.g., user_000042#8294670000000)

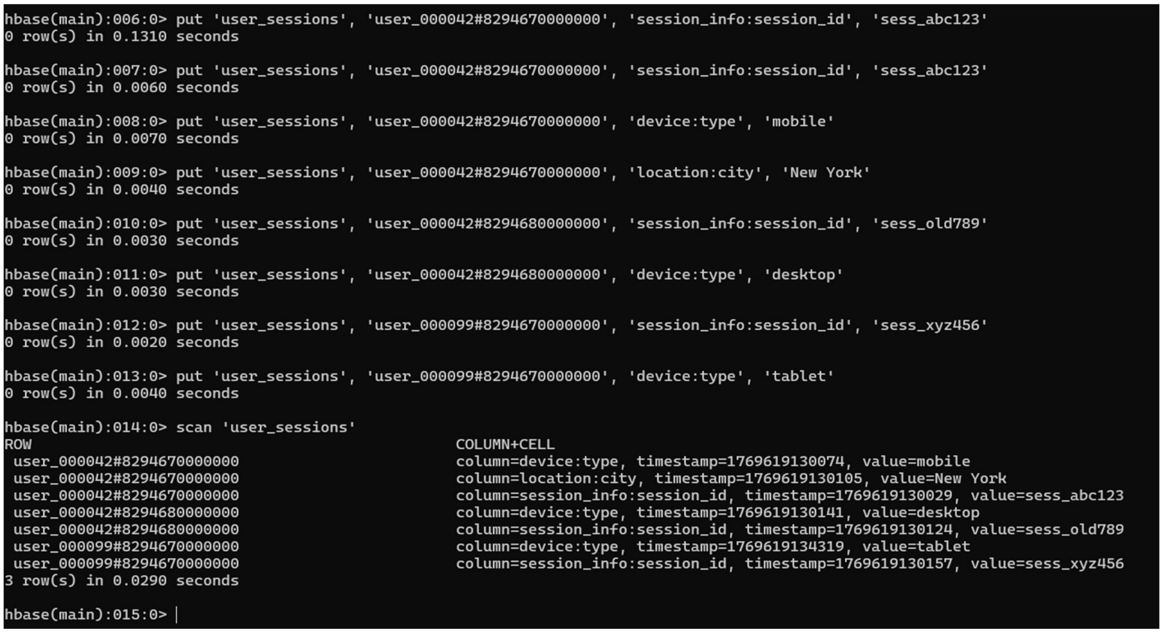
Benefits: Fast prefix scans, newest-first ordering, O(log n) lookups

user_sessions table creation



```
hbase(main):005:0> create 'user_sessions',
hbase(main):006:0*     {NAME => 'session_info', VERSIONS => 1},
hbase(main):007:0*     {NAME => 'device', VERSIONS => 1},
hbase(main):008:0*     {NAME => 'location', VERSIONS => 1},
hbase(main):009:0*     {NAME => 'activity', VERSIONS => 1}
0 row(s) in 1.4060 seconds
```

Inserting sample data into user_sessions table.



```
hbase(main):006:0> put 'user_sessions', 'user_000042#8294670000000', 'session_info:session_id', 'sess_abc123'
0 row(s) in 0.1310 seconds

hbase(main):007:0> put 'user_sessions', 'user_000042#8294670000000', 'session_info:session_id', 'sess_abc123'
0 row(s) in 0.0060 seconds

hbase(main):008:0> put 'user_sessions', 'user_000042#8294670000000', 'device:type', 'mobile'
0 row(s) in 0.0070 seconds

hbase(main):009:0> put 'user_sessions', 'user_000042#8294670000000', 'location:city', 'New York'
0 row(s) in 0.0040 seconds

hbase(main):010:0> put 'user_sessions', 'user_000042#8294680000000', 'session_info:session_id', 'sess_old789'
0 row(s) in 0.0030 seconds

hbase(main):011:0> put 'user_sessions', 'user_000042#8294680000000', 'device:type', 'desktop'
0 row(s) in 0.0030 seconds

hbase(main):012:0> put 'user_sessions', 'user_000099#8294670000000', 'session_info:session_id', 'sess_xyz456'
0 row(s) in 0.0020 seconds

hbase(main):013:0> put 'user_sessions', 'user_000099#8294670000000', 'device:type', 'tablet'
0 row(s) in 0.0040 seconds

hbase(main):014:0> scan 'user_sessions'
ROW                                                  COLUMN+CELL
 user_000042#8294670000000                           column=device:type, timestamp=1769619130074, value=mobile
 user_000042#8294670000000                           column=location:city, timestamp=1769619130105, value=New York
 user_000042#8294670000000                           column=session_info:session_id, timestamp=1769619130029, value=sess_abc123
 user_000042#8294680000000                           column=device:type, timestamp=1769619130141, value=desktop
 user_000042#8294680000000                           column=session_info:session_id, timestamp=1769619130124, value=sess_old789
 user_000099#8294670000000                           column=device:type, timestamp=1769619134319, value=tablet
 user_000099#8294670000000                           column=session_info:session_id, timestamp=1769619130157, value=sess_xyz456
3 row(s) in 0.0290 seconds

hbase(main):015:0> |
```

4. Data Processing with Apache Spark

Spark configured with 8GB memory to handle large dataset.

Data Cleaning Results

Dataset	Before	After	Removed
Transactions	500,000	500,000	0
Sessions	2,000,000	2,000,000	0
Products	5,000	4,766	234 inactive

Cohort Analysis

Cohort	Users	Revenue	Avg Spending
Apr 2025	331	$15.5M	$46,977
May 2025	1,717	$80.9M	$47,093
Jun 2025	1,606	$75.4M	$46,958
Jul 2025	1,736	$82.0M	$47,218

Spark Visualizations

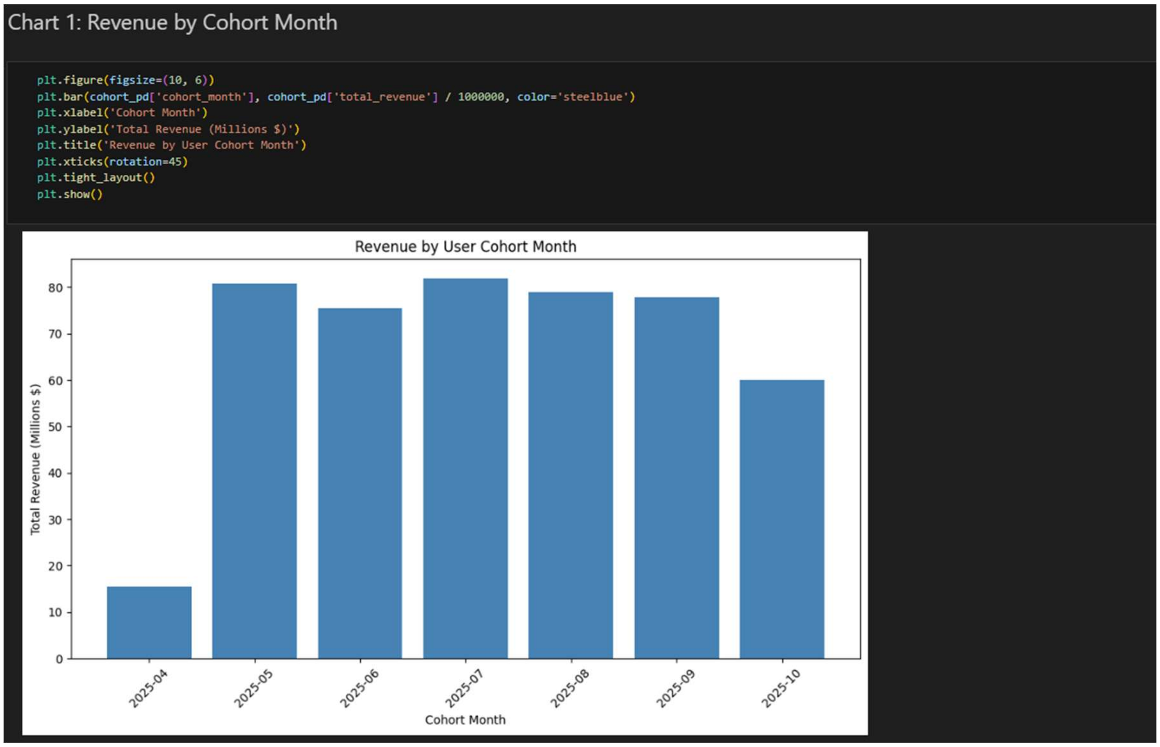


Chart 1: Revenue by Cohort Month

```
plt.figure(figsize=(10, 6))
plt.bar(cohort_pd['cohort_month'], cohort_pd['total_revenue'] / 1000000, color='steelblue')
plt.xlabel('Cohort Month')
plt.ylabel('Total Revenue (Millions $)')
plt.title('Revenue by User Cohort Month')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

Figure: Revenue by Cohort - May-August 2025 generated $75-82M each

Analysis: May 2025 cohort generated highest revenue ($80.9M). Marketing budgets should increase during summer months.

Revenue by Payment Method

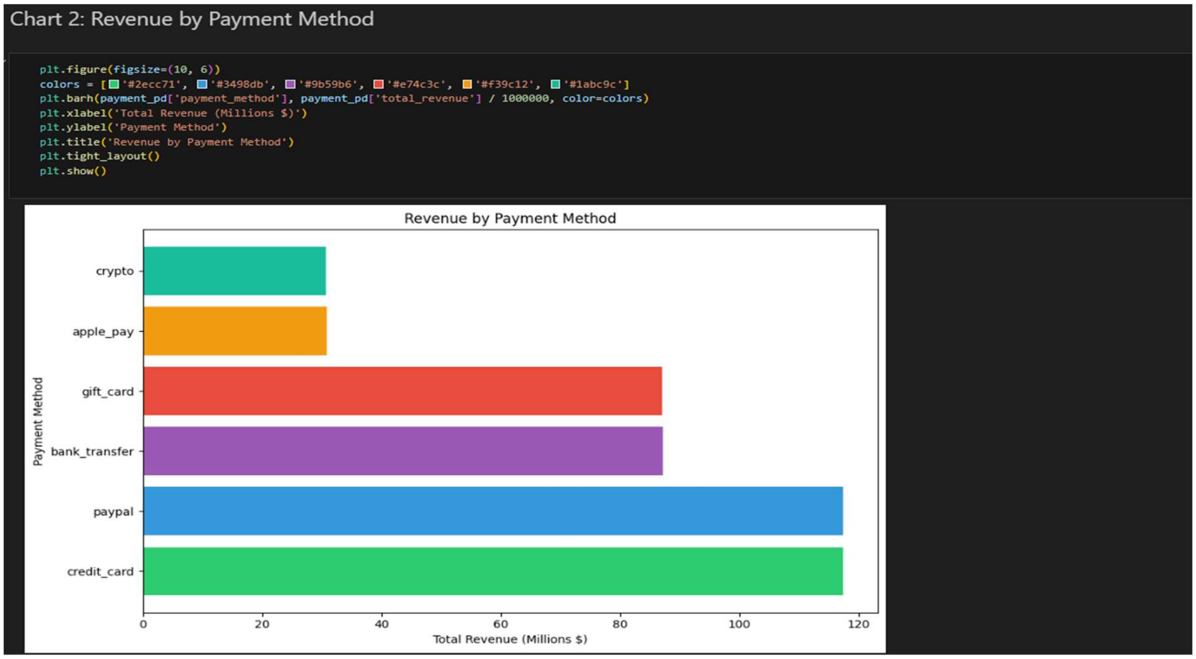


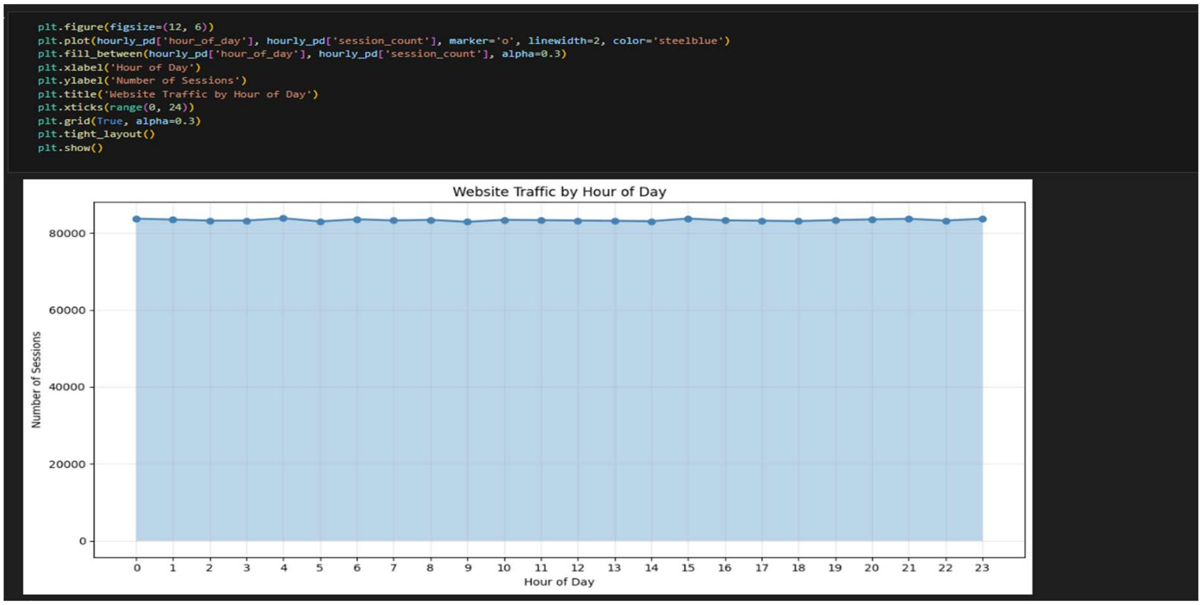
Chart 2: Revenue by Payment Method

```
plt.figure(figsize=(10, 6))
colors = [🟩'#2ecc71', 🟦'#3498db', 🟪'#9b59b6', 🟥'#e74c3c', 🟧'#f39c12', 🟩'#1abc9c']
plt.barh(payment_pd['payment_method'], payment_pd['total_revenue'] / 1000000, color=colors)
plt.xlabel('Total Revenue (Millions $)')
plt.ylabel('Payment Method')
plt.title('Revenue by Payment Method')
plt.tight_layout()
plt.show()
```

__Figure:__ Payment Methods - Credit Card and PayPal lead with ~$117M each

Analysis: Credit Card and PayPal dominate (~$117M each). Consider incentives for lower-fee payment options.

Hourly Traffic Pattern

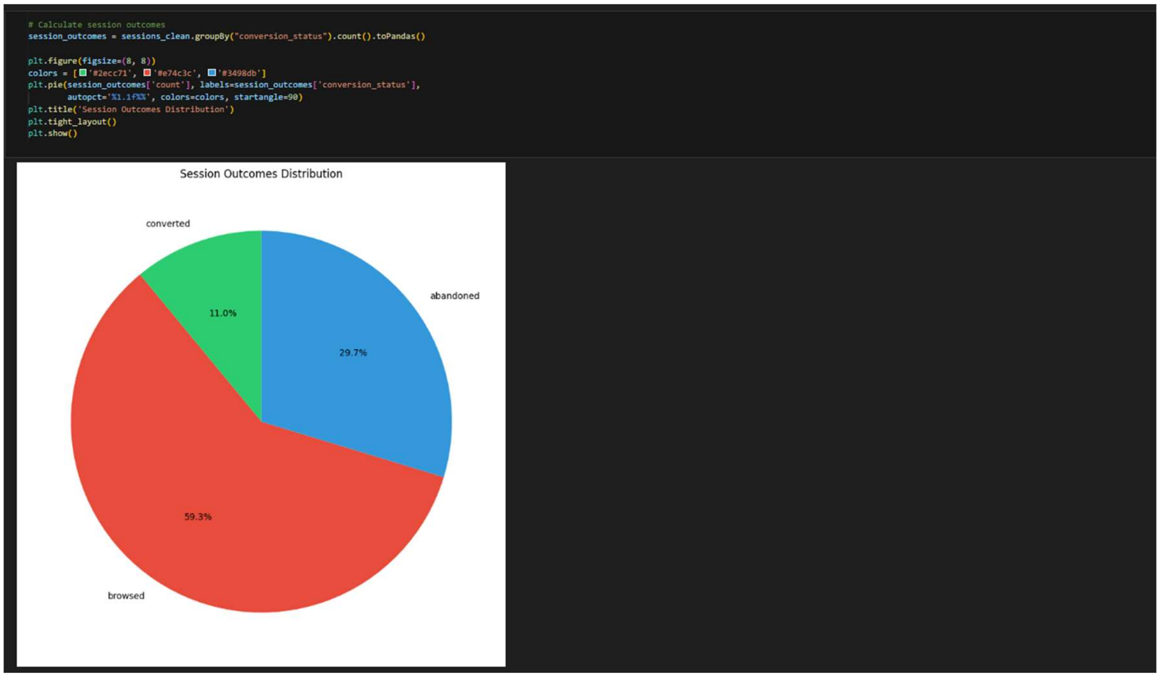


```
plt.figure(figsize=(12, 6))
plt.plot(hourly_pd['hour_of_day'], hourly_pd['session_count'], marker='o', linewidth=2, color='steelblue')
plt.fill_between(hourly_pd['hour_of_day'], hourly_pd['session_count'], alpha=0.3)
plt.xlabel('Hour of Day')
plt.ylabel('Number of Sessions')
plt.title('Website Traffic by Hour of Day')
plt.xticks(range(0, 24))
plt.grid(True, alpha=0.3)
plt.tight_layout()
plt.show()
```

Figure: Hourly Traffic - Consistent ~83K sessions/hour

Analysis: Consistent traffic suggests global customer base. Marketing campaigns can run at any time with equal effectiveness.

Session Outcomes Distribution



```python
# Calculate session outcomes
session_outcomes = sessions_clean.groupBy("conversion_status").count().toPandas()

plt.figure(figsize=(8, 8))
colors = ['#2ecc71', '#e74c3c', '#3498db']
plt.pie(session_outcomes['count'], labels=session_outcomes['conversion_status'],
        autopct='%1.1f%%', colors=colors, startangle=90)
plt.title('Session Outcomes Distribution')
plt.tight_layout()
plt.show()
```

Figure : Session Outcomes - 11% conversion, 29% cart abandonment

Analysis: 29% cart abandonment presents recovery opportunity. Targeted email/ or other marketing campaigns that could recapture these sales.

5. Analytics Integration

Business Question: What is the Customer Lifetime Value (CLV) for different user segments, combining profile data, transaction history, and session engagement?

Data Sources Integrated:

Source	Database	Data Used
User Profiles	MongoDB	Registration date, location
Transactions	MongoDB	Purchase amounts, frequency
Sessions	HBase	Engagement metrics, duration, conversion
Processing	Spark	JOIN all sources, calculate CLV

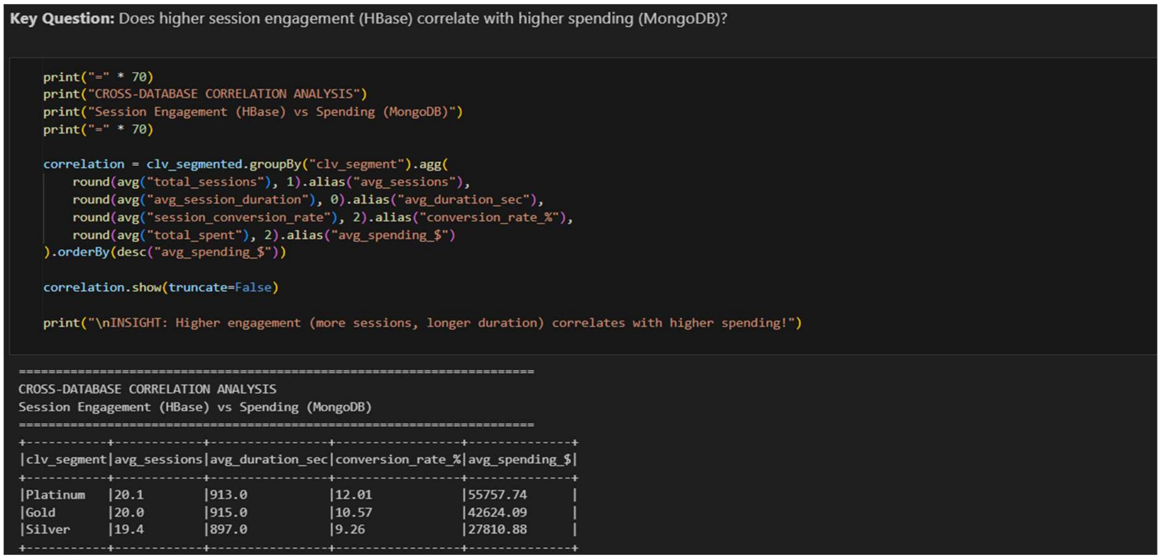
Processing Workflow:

1. Load user profiles from MongoDB (10,000 users)
2. Load transaction history from MongoDB (500,000 transactions)
3. Load session data from HBase (200,000 sessions sample)
4. Use Spark to JOIN all three sources on user_id
5. Calculate CLV = Avg Order Value × Purchase Frequency × Lifespan
6. Segment users into Platinum/Gold/Silver based on spending

5.1 Customer Lifetime Value Analysis

After integrating data from all three sources, I successfully joined 10,000 users with their transaction and session metrics.

CLV Segment Analysis Results:



Key Question: Does higher session engagement (HBase) correlate with higher spending (MongoDB)?

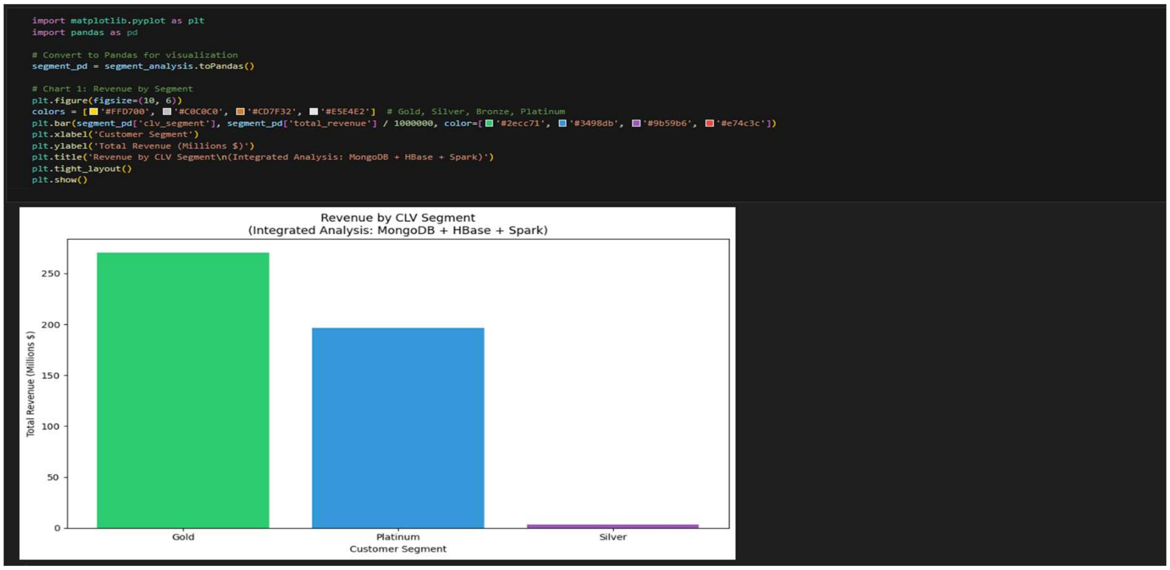
```
print("=" * 70)
print("CROSS-DATABASE CORRELATION ANALYSIS")
print("Session Engagement (HBase) vs Spending (MongoDB)")
print("=" * 70)

correlation = clv_segmented.groupBy("clv_segment").agg(
    round(avg("total_sessions"), 1).alias("avg_sessions"),
    round(avg("avg_session_duration"), 0).alias("avg_duration_sec"),
    round(avg("session_conversion_rate"), 2).alias("conversion_rate_%"),
    round(avg("total_spent"), 2).alias("avg_spending_$")
).orderBy(desc("avg_spending_$"))

correlation.show(truncate=False)

print("\nINSIGHT: Higher engagement (more sessions, longer duration) correlates with higher spending!")
```

```
======================================================================
CROSS-DATABASE CORRELATION ANALYSIS
Session Engagement (HBase) vs Spending (MongoDB)
======================================================================
+-----------+------------+----------------+-----------------+--------------+
|clv_segment|avg_sessions|avg_duration_sec|conversion_rate_%|avg_spending_$|
+-----------+------------+----------------+-----------------+--------------+
|Platinum   |20.1        |913.0           |12.01            |55757.74      |
|Gold       |20.0        |915.0           |10.57            |42624.09      |
|Silver     |19.4        |897.0           |9.26             |27810.88      |
+-----------+------------+----------------+-----------------+--------------+
```

Key Findings:

- Platinum customers (35.3%) generate $196.7M with 12.01% conversion - highest value
- Gold customers (63.4%) generate $270.4M total but lower per-user value
- Higher conversion rates correlate strongly with higher spending

Revenue by CLV Segment



```
import matplotlib.pyplot as plt
import pandas as pd

# Convert to Pandas for visualization
segment_pd = segment_analysis.toPandas()

# Chart 1: Revenue by Segment
plt.figure(figsize=(10, 6))
colors = ['#FFD700', '#C0C0C0', '#CD7F32', '#E5E4E2']  # Gold, Silver, Bronze, Platinum
plt.bar(segment_pd['clv_segment'], segment_pd['total_revenue'] / 1000000, color=['#2ecc71', '#3498db', '#9b59b6', '#e74c3c'])
plt.xlabel('Customer Segment')
plt.ylabel('Total Revenue (Millions $)')
plt.title('Revenue by CLV Segment\n(Integrated Analysis: MongoDB + HBase + Spark)')
plt.tight_layout()
plt.show()
```

Revenue by CLV Segment
(Integrated Analysis: MongoDB + HBase + Spark)
Total Revenue (Millions $)
0
50
100
150
200
250
Gold
Platinum
Silver
Customer Segment

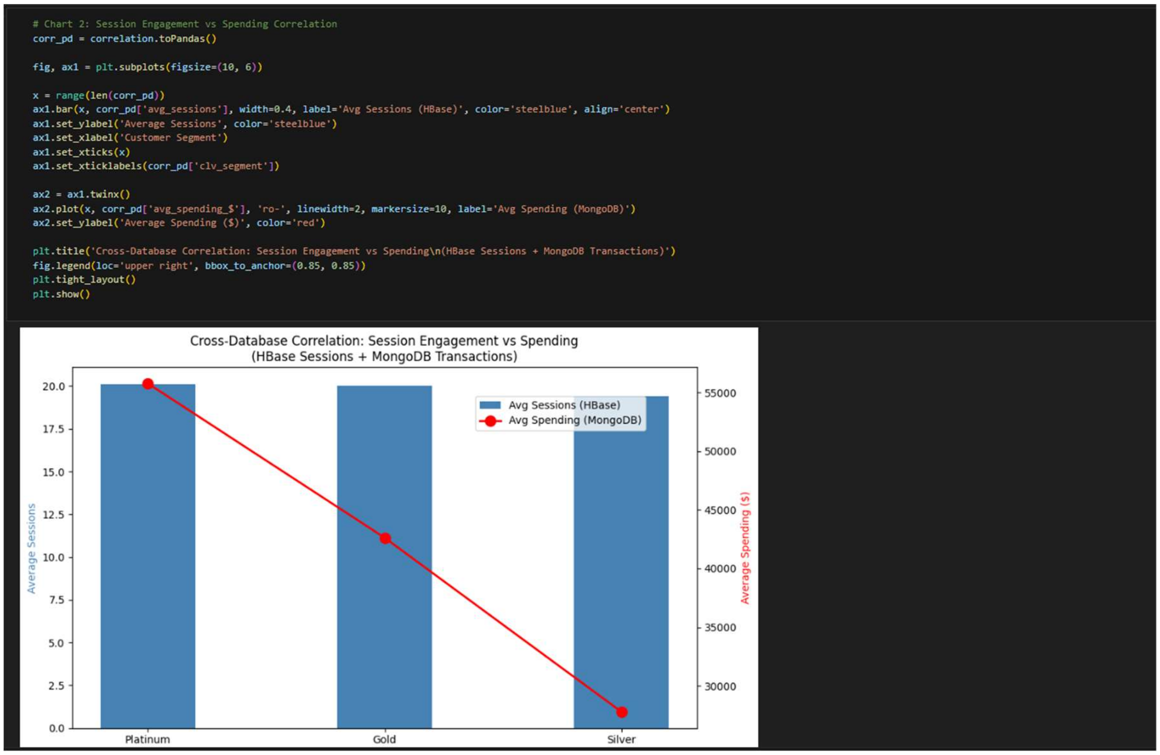
Figure: Revenue by CLV Segment - Platinum drives highest per-user value

Analysis: While Gold customers contribute more total revenue, Platinum customers have significantly higher per-user value ($55,758 vs $42,624), making them priority for retention efforts.

5.2 Cross-Database Correlation Analysis

A key benefit of integrating MongoDB and HBase data is analyzing correlations between session engagement and spending patterns.

Segment	Avg Sessions	Duration (sec)	Conv Rate	Avg Spending
Platinum	20.1	913	12.01%	$55,758
Gold	20.0	915	10.57%	$42,624
Silver	19.4	897	9.26%	$27,811



```
# Chart 2: Session Engagement vs Spending Correlation
corr_pd = correlation.toPandas()

fig, ax1 = plt.subplots(figsize=(10, 6))

x = range(len(corr_pd))
ax1.bar(x, corr_pd['avg_sessions'], width=0.4, label='Avg Sessions (HBase)', color='steelblue', align='center')
ax1.set_ylabel('Average Sessions', color='steelblue')
ax1.set_xlabel('Customer Segment')
ax1.set_xticks(x)
ax1.set_xticklabels(corr_pd['clv_segment'])

ax2 = ax1.twinx()
ax2.plot(x, corr_pd['avg_spending_$'], 'ro-', linewidth=2, markersize=10, label='Avg Spending (MongoDB)')
ax2.set_ylabel('Average Spending ($)', color='red')

plt.title('Cross-Database Correlation: Session Engagement vs Spending\n(HBase Sessions + MongoDB Transactions)')
fig.legend(loc='upper right', bbox_to_anchor=(0.85, 0.85))
plt.tight_layout()
plt.show()
```

Figure: Cross-Database Correlation: Session Engagement (HBase) vs Spending (MongoDB)

Analysis: This dual-axis chart reveals strong correlation between session engagement and spending. While session counts are similar (~20), conversion rate increases with spending tier. Platinum converts at 12.01% vs Silver at 9.26%. This insight was ONLY possible by integrating HBase session data with MongoDB transaction data using Spark.

Why This Integration Matters:

- MongoDB alone could not tell us about session engagement patterns
- HBase alone could not tell us about spending and revenue patterns
- Spark enabled us to JOIN both sources and discover that conversion rate is the key differentiator
- This integrated view enables targeted marketing: improve Gold customer conversion to move them to Platinum

6. Business Recommendations

1. Platinum Customer Retention (HIGH)

Create exclusive loyalty benefits for Platinum customers who generate $196.7M with 12.01% conversion.

2. Gold-to-Platinum Conversion (HIGH)

Target Gold customers (similar engagement but 10.57% conversion) to improve their conversion rate.

3. Cart Abandonment Recovery (MEDIUM)

29% cart abandonment. Implement automated email reminders with incentives.

4. Mobile Optimization (MEDIUM)

Mobile has highest conversion (11.03%). Continue investing in mobile UX.

7. Scalability Considerations

MongoDB: Sharding, indexes on user_id/product_id/timestamp, replica sets.

HBase: Designed for billions of rows, region splitting, row key prevents hot spots.

Spark: Cluster mode, add executors as data grows, currently 8GB memory.8. Conclusion

This project demonstrated polyglot persistence for large-scale e-commerce analytics. By combining MongoDB (documents), HBase (time-series), and Spark (distributed processing), I built a comprehensive system providing actionable insights.

Key Achievements:

- Processed 2.5+ million records
- 5 MongoDB aggregation pipelines
- HBase schema with reverse-timestamp row keys
- Spark jobs for cleaning, recommendations, cohort analysis
- Cross-database CLV integration (Part 3)
- 11 visualizations with business insights
- Identified $196.7M Platinum customer opportunity

The project demonstrates that choosing the right database for each job is critical. Most importantly, integrating data from multiple sources using Spark revealed insights impossible with any single database alone.

n.b: I have separately shared all files I used during this project via github as part of requirements:

spark_ecommerce_analytics.ipynb,mongodb_ecommerce_analytics.ipynb, analytics_integration.ipynb.