

Retail Sales Analytics Using Excel, SQL, Power BI and Python

CAPSTONE PROJECT

Problem Statement:

In a competitive retail market, businesses need to track sales performance, understand customer behaviour, and identify growth opportunities. This project aims to build an end-to-end analytics solution that enables insights generation from raw transactional data using a multi-tool approach.

Retail company: Gear up

Phase 1: Excel – Data Cleaning & Preparation

1. Standardize Column Headers

Ensure all columns across sheets (like customers, orders, products) are consistently named.

➤ Used Proper function – standardised the Column headers.

2. Remove Duplicates

Identify and remove any duplicate records.

➤ Data Tab – Remove duplicates

3. Handle Missing Values

Use filtering and formulas to find blanks/nulls.

Fill missing ZIP codes or phone numbers with placeholders or “NA”.

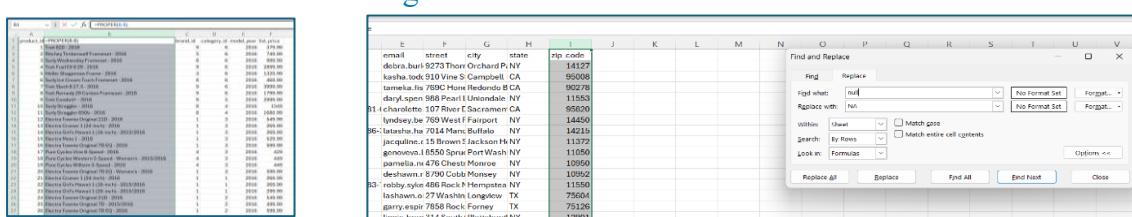
➤ For null values used ctrl + H > replaced null to “NA”

4. Data Type Conversion

Convert order dates, shipped dates, and required dates into Excel Date format.

Ensure numeric fields (quantity, price) are set to number format.

➤ Home tab > Number Formatting



A	B	C	D	E	F	G	H	I	J
order_id	item_id	product_id	Product Name	quantity	list_price	discount	Total Price	Category_Id	Category_Name
1	1439	2	263 Strider Classic 12 Balance Bike - 2018	1	89.99	0.2	89.79	1	Children Bicycles
2	1462	1	263 Strider Classic 12 Balance Bike - 2018	1	89.99	0.2	89.79	1	Children Bicycles
3	1425	2	263 Strider Classic 12 Balance Bike - 2018	1	89.99	0.07	89.92	1	Children Bicycles
4	609	1	84 Sun Bicycles Lil Kitten - 2017	1	109.99	0.2	109.79	1	Children Bicycles
5	1252	2	84 Sun Bicycles Lil Kitten - 2017	1	109.99	0.2	109.79	1	Children Bicycles
6	1472	1	84 Sun Bicycles Lil Kitten - 2017	1	109.99	0.2	109.79	1	Children Bicycles
7	667	2	84 Sun Bicycles Lil Kitten - 2017	1	109.99	0.1	109.89	1	Children Bicycles
8	1210	4	84 Sun Bicycles Lil Kitten - 2017	1	109.99	0.1	109.89	1	Children Bicycles
9	701	1	84 Sun Bicycles Lil Kitten - 2017	1	109.99	0.05	109.94	1	Children Bicycles
10	733	1	84 Sun Bicycles Lil Kitten - 2017	1	109.99	0.05	109.94	1	Children Bicycles
11	1194	4	84 Sun Bicycles Lil Kitten - 2017	1	109.99	0.05	109.94	1	Children Bicycles
12	747	2	86 Trek Girl's Kickster - 2017	1	149.99	0.2	149.79	1	Children Bicycles

5. Data Validation

Use dropdowns for fields like order_status (Pending, Shipped, Delivered) to ensure consistency.

➤ New column “Status” > Data Tab > Data Validation > Lists

The screenshot shows the 'Data Validation' dialog box. Under 'Validation criteria', the 'Allow' dropdown is set to 'List'. The 'Source' dropdown is set to 'Pending, Shipped, Delivered'. The 'OK' button is highlighted.

6. Create New Derived Columns

Add a column in order_items to calculate total_price.

➤ Formula – (List_price * quantity) – Discount

The screenshot shows a spreadsheet with columns A through J. Column D contains the formula $=E2*D2-G2$. The formula is highlighted in red, and the cell reference $=E2$ is also highlighted in red in the formula bar.

7. Merge Lookup Data

Use VLOOKUP or XLOOKUP to merge product names into order_items using product_id.

➤ Xlookup function - =Xlookup(product_id, product_id range, product_name range)

The screenshot shows a spreadsheet with columns A through J. Column D contains the formula $=XLOOKUP(C2,Products!$A$2:$A$322,Products!$B$2:$B$322)$. The formula is highlighted in red, and the cell reference $=C2$ is also highlighted in red in the formula bar.

8. Create Basic Pivot Table

Build a pivot table summarizing total sales by product category.

- Using Xlookup – inserted Category_id and Category_name in workbook.
- Insert > Pivot Table > Sales in values & Category_name in rows

The screenshot shows a pivot table with columns C through M. The rows are labeled A and B. The pivot table displays a summary of sales by category, including Product Name, quantity, list price, discount, Total Price, Category_id, and Category_Name.

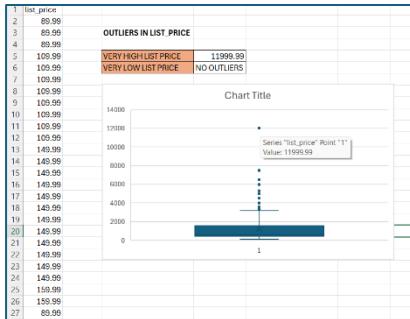
The screenshot shows a pivot table with columns D through N. The rows are labeled A and B. The pivot table displays a summary of sales by category, including Product Name, quantity, list price, discount, Total Price, Category_id, and Category_Name.

The screenshot shows the 'PivotTable Fields' dialog box. The 'Row Labels' section is set to 'Sum of Total_Price'. The 'Value' section is set to 'Sum of Total_Price'. The 'Category Name' section is set to 'Category_Name'. The 'Report Filter' section is set to 'Category_Name'.

9. Sort and Filter for Outliers

Find outliers in pricing

- Select column >Insert> Recommended charts > Boxplot chart



10. Prepare Final CSVs

Save cleaned sheets as CSVs named: customers.csv, orders.csv, etc., for SQL import.

Phase 2: SQL – Database Management and Querying

1. Create Tables Based on ERD

Use CREATE TABLE statements to replicate the exact structure of the

ER diagram (with constraints).

2. Import CSVs into SQL

Load cleaned Excel files using LOAD DATA or MySQL Workbench import feature.

- Database created RETAIL_SALES > Tables > Import Wizard > Clean dataset loaded > Alter table > changed data types and added primary key constraint

3. Inner Join for Order Details

Join orders, order_items, and products to display detailed line items.

Retail Grid													
order_id	customer_id	order_status	order_date	required_date	shipped_date	store_id	status	order_id	item_id	product_id	quantity	list_price	
												discount	
1	259	4	2016-01-01	2016-01-03	2016-01-03	1	2	Delivered	1	1	20	1	599.99
1	259	4	2016-01-01	2016-01-03	2016-01-03	1	2	Delivered	1	2	8	2	1799.99
1	259	4	2016-01-01	2016-01-03	2016-01-03	1	2	Delivered	1	3	10	2	1599.05
1	259	4	2016-01-01	2016-01-03	2016-01-03	1	2	Delivered	1	4	16	2	599.99
1	259	4	2016-01-01	2016-01-03	2016-01-03	1	2	Delivered	1	5	4	1	2099.99
2	1212	4	2016-01-01	2016-01-04	2016-01-03	2	6	Delivered	2	2	16	1	599.99
2	1212	4	2016-01-01	2016-01-04	2016-01-03	2	6	Delivered	2	2	16	1	599.99
3	523	4	2016-01-02	2016-01-05	2016-01-03	2	7	Delivered	3	1	3	1	999.99
3	523	4	2016-01-02	2016-01-05	2016-01-03	2	7	Delivered	3	2	20	1	599.99
4	129	4	2016-01-03	2016-01-05	2016-01-03	1	3	Delivered	4	1	2	2	799.99
5	1224	4	2016-01-03	2016-01-06	2016-01-06	2	6	Delivered	5	1	10	2	1599.05
5	1324	4	2016-01-03	2016-01-06	2016-01-06	2	6	Delivered	5	2	17	1	429.05
e	1774	4	2016-01-04	2016-01-06	2016-01-06	2	c	Pending	e	?	?	1	599.99

4. Total Sales by Store

Write a query to group sales (total price) by each store id.

```
-- 4
5 * select o.store_id, sum(oi.total_price) as Total_sales from order_items as oi
6 inner join orders as o on oi.order_id = o.order_id
7 group by o.store_id;
8
```

grid Grid Filter Rows Export Wrap Cell Contents

store_id	Total_sales
1	1790310.499999837
2	5825901.84999975
3	962548.410000039

5. Top 5 Selling Products

Use ORDER BY and LIMIT to get the top 5 most sold products by quantity.

```

77 -- 5 --
78 • select o.product_name,sum(oi.quantity) as sum_qty from order_items as oi
79 inner join products as p on oi.product_id = p.product_id
80 group by o.product_name
81 order by sum_qty desc
82 limit 5;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows: |

product_name	sum_qty
Electra Cruiser 1 (24-inch) - 2016	296
Electra Tonne Original 70 EQ - 2016	290
Electra Tonne Original 21D - 2016	289
Electra Grit Hail 1 (16-inch) - 2015/2016	269
Surly Ice Cream Truck Frameset - 2016	167

6. Customer Purchase Summary

For each customer, return total orders placed, total items purchased, and total revenue.

```

83
84 -- 6 --
85 • select o.customer_id, count(o.order_id) as total_orders, count(oi.item_id) as total_items, sum(oi.total_price) as total_revenue from order_items as oi
86 inner join orders as o on oi.order_id = o.order_id
87 group by o.customer_id;
88

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows: |

customer_id	total_orders	total_items	total_revenue
1	11	11	30644.79
2	10	10	21652.87
3	13	13	26248.659999999993
4	9	9	24700.659999999998
5	8	8	18402.01
6	11	11	38586.829999999994
7	4	4	7707.64
8	3	3	2603.58
9	9	9	2678.97
10	11	11	29052.42
11	5	5	4079.49
12	8	8	27157.21
13	8	8	15830.32
14	n	n	47683.400000000004

7. Segment Customers by Total Spend

Write a query to classify customers into spending brackets (e.g., low, medium, high).

```

89
90 -- 7 -----
91 • select o.customer_id, c.first_name, sum(oi.total_price) as total_spending,
92 case
93 when sum(oi.total_price) >= 15000 then "High"
94 when sum(oi.total_price) between 8000 and 15000 then "Medium"
95 else "Low"
96 end as Spending_brackets
97 from order_items as oi
98 inner join orders as o on oi.order_id = o.order_id
99 inner join customers as c on o.customer_id = c.customer_id
100 group by o.customer_id, c.first_name;
101

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows: |

customer_id	first_name	total_spending	Spending_brackets
1	Debra	30644.79	High
2	Kedra	21652.87	High
3	Taneka	26248.659999999993	High
4	Daryl	24700.659999999998	High
5	Charlotte	18402.01	High
6	Lindsey	38586.829999999994	High
7	Latahah	7707.64	Low
8	Jacqueline	2603.58	Low
9	Genoveva	2678.97	High
10	Pearl	29052.42	High
11	Deshan	4079.49	Low
12	Robby	27157.21	High
13	Lashawn	15830.32	High

8. Staff Performance Analysis

Analyze total revenue generated by each staff member based on their handled orders.

```

11 -- 8 -----
12 • select o.staff_id, s.first_name, count(oi.order_id) as handled_orders, sum(oi.total_price) as total_revenue from order_items as oi
13 inner join orders as o on oi.order_id = o.order_id
14 inner join staffs as s on o.staff_id = s.staff_id
15 group by o.staff_id, s.first_name;
16

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows: |

staff_id	first_name	handled_orders	total_revenue
2	Mireya	462	837375.7800000024
3	Genna	544	952665.7700000049
6	Marcelene	1615	2938714.119999979
7	Venita	1580	2887187.7299999804
8	Kali	269	516667.6600000003
9	Laura	479	48663.7000000001

9. Stock Alert Query

Write a query to list products where stock quantity < 10 in any store.

```

107
108 • select s.store_id, s.store_name, sum(sk.quantity) as total_quantity, p.product_name from stores as s
109 inner join stocks as sk on s.store_id = sk.store_id
110 inner join products as p on p.product_id = sk.product_id
111 group by s.store_id, s.store_name, p.product_name
112 having sum(sk.quantity) < 10;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows: |

store_id	store_name	total_quantity	product_name
1	Santa Cruz Bikes	23	Trek 830 - 2016
1	Santa Cruz Bikes	23	Trek Fuel EX 9.29 - 2016
1	Santa Cruz Bikes	22	Heller Shaganav Frame - 2016
1	Santa Cruz Bikes	11	Trek Conduit + - 2016
1	Santa Cruz Bikes	15	Trek 9.8 2016
1	Santa Cruz Bikes	26	Electra Cruiser 1 (24-inch) - 2016
1	Santa Cruz Bikes	37	Electra Grit Hail 1 (16-inch) - 2015/2016
1	Santa Cruz Bikes	20	Electra Tonne Original 70 EQ - 2016
1	Santa Cruz Bikes	48	Electra Tonne Original 70 EQ - Women's - 2015...
1	Santa Cruz Bikes	26	Electra Tonne Original 70 EQ - Women's - 2016
1	Santa Cruz Bikes	21	Surly Big Dummy Frameset - 2017
1	Santa Cruz Bikes	20	Surly Karate Monkey 27.5+ Frameset - 2017

10. Create Final Segmentation Table

Create a table customer_segments that will be populated from Python ML results later.

Phase 3: Python

1. Load Data from SQL

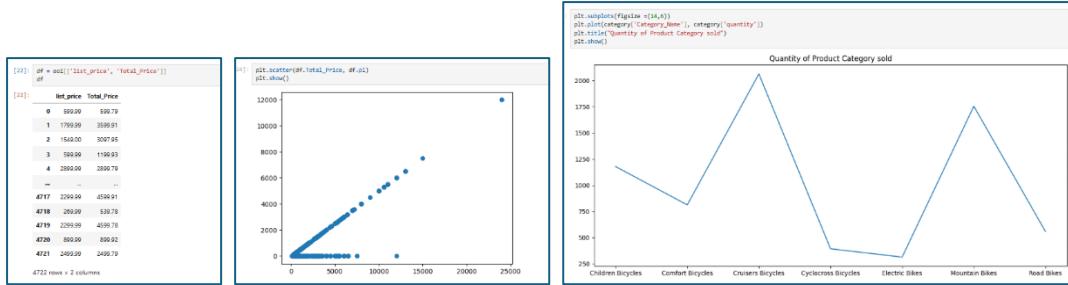
Use pandas.read_sql() to pull the orders, order_items, and customers tables into a DataFrame.

```
from sqlalchemy import create_engine
engine = create_engine("mysql+pymysql://root:12345@localhost:3306/retail_sales")
customers = pd.read_sql("SELECT * FROM customers;", engine)
order_items = pd.read_sql("SELECT * FROM order_items;", engine)
orders = pd.read_sql("SELECT * FROM orders;", engine)
```

2. Basic EDA (Exploratory Data Analysis)

Use df.describe(), df.info(), and df.value_counts() to summarize the dataset.

customer_id	zip_code
1	10001
2	10002
3	10003
4	10004
5	10005
6	10006
7	10007
8	10008
9	10009
10	10010
11	10011
12	10012
13	10013
14	10014
15	10015
16	10016
17	10017
18	10018
19	10019
20	10020
21	10021
22	10022
23	10023
24	10024
25	10025
26	10026
27	10027
28	10028
29	10029
30	10030
31	10031
32	10032
33	10033
34	10034
35	10035
36	10036
37	10037
38	10038
39	10039
40	10040
41	10041
42	10042
43	10043
44	10044
45	10045
46	10046
47	10047
48	10048
49	10049
50	10050
51	10051
52	10052
53	10053
54	10054
55	10055
56	10056
57	10057
58	10058
59	10059
60	10060
61	10061
62	10062
63	10063
64	10064
65	10065
66	10066
67	10067
68	10068
69	10069
70	10070
71	10071
72	10072
73	10073
74	10074
75	10075
76	10076
77	10077
78	10078
79	10079
80	10080
81	10081
82	10082
83	10083
84	10084
85	10085
86	10086
87	10087
88	10088
89	10089
90	10090
91	10091
92	10092
93	10093
94	10094
95	10095
96	10096
97	10097
98	10098
99	10099
100	10100
101	10101
102	10102
103	10103
104	10104
105	10105
106	10106
107	10107
108	10108
109	10109
110	10110
111	10111
112	10112
113	10113
114	10114
115	10115
116	10116
117	10117
118	10118
119	10119
120	10120
121	10121
122	10122
123	10123
124	10124
125	10125
126	10126
127	10127
128	10128
129	10129
130	10130
131	10131
132	10132
133	10133
134	10134
135	10135
136	10136
137	10137
138	10138
139	10139
140	10140
141	10141
142	10142
143	10143
144	10144
145	10145
146	10146
147	10147
148	10148
149	10149
150	10150
151	10151
152	10152
153	10153
154	10154
155	10155
156	10156
157	10157
158	10158
159	10159
160	10160
161	10161
162	10162
163	10163
164	10164
165	10165
166	10166
167	10167
168	10168
169	10169
170	10170
171	10171
172	10172
173	10173
174	10174
175	10175
176	10176
177	10177
178	10178
179	10179
180	10180
181	10181
182	10182
183	10183
184	10184
185	10185
186	10186
187	10187
188	10188
189	10189
190	10190
191	10191
192	10192
193	10193
194	10194
195	10195
196	10196
197	10197
198	10198
199	10199
200	10200
201	10201
202	10202
203	10203
204	10204
205	10205
206	10206
207	10207
208	10208
209	10209
210	10210
211	10211
212	10212
213	10213
214	10214
215	10215
216	10216
217	10217
218	10218
219	10219
220	10220
221	10221
222	10222
223	10223
224	10224
225	10225
226	10226
227	10227
228	10228
229	10229
230	10230
231	10231
232	10232
233	10233
234	10234
235	10235
236	10236
237	10237
238	10238
239	10239
240	10240
241	10241
242	10242
243	10243
244	10244
245	10245
246	10246
247	10247
248	10248
249	10249
250	10250
251	10251
252	10252
253	10253
254	10254
255	10255
256	10256
257	10257
258	10258
259	10259
260	10260
261	10261
262	10262
263	10263
264	10264
265	10265
266	10266
267	10267
268	10268
269	10269
270	10270
271	10271
272	10272
273	10273
274	10274
275	10275
276	10276
277	10277
278	10278
279	10279
280	10280
281	10281
282	10282
283	10283
284	10284
285	10285
286	10286
287	10287
288	10288
289	10289
290	10290
291	10291
292	10292
293	10293
294	10294
295	10295
296	10296
297	10297
298	10298
299	10299
300	10300
301	10301
302	10302
303	10303
304	10304
305	10305
306	10306
307	10307
308	10308
309	10309
310	10310
311	10311
312	10312
313	10313
314	10314
315	10315
316	10316
317	10317
318	10318
319	10319
320	10320
321	10321
322	10322
323	10323
324	10324
325	10325
326	10326
327	10327
328	10328
329	10329
330	10330
331	10331
332	10332
333	10333
334	10334
335	10335
336	10336
337	10337
338	10338
339	10339
340	10340
341	10341
342	10342
343	10343
344	10344
345	10345
346	10346
347	10347
348	10348
349	10349
350	10350
351	10351
352	10352
353	10353
354	10354
355	10355
356	10356
357	10357
358	10358
359	10359
360	10360
361	10361
362	10362
363	10363
364	10364
365	10365
366	10366
367	10367
368	10368
369	10369
370	10370
371	10371
372	10372
373	10373
374	10374
375	10375
376	10376
377	10377
378	10378
379	10379
380	10380
381	10381
382	10382
383	10383
384	10384
385	10385
386	10386
387	10387
388	10388
389	10389
390	10390
391	10391
392	10392
393	10393
394	10394
395	10395
396	10396
397	10397
398	10398
399	10399
400	10400
401	10401
402	10402
403	10403
404	10404
405	10405
406	10406
407	10407
408	10408
409	10409
410	10410
411	10411
412	10412
413	10413
414	10414
415	10415
416	10416
417	10417
418	10418
419	10419
420	10420
421	10421
422	10422
423	10423
424	10424
425	10425
426	10426
427	10427
428	10428
429	10429
430	10430
431	10431
432	10432
433	10433
434	10434
435	10435
436	10436
437	10437
438	10438
439	10439
440	10440
441	10441
442	10442
443	10443
444	10444
445	10445
446	10446
447	10447
448	10448
449	10449
450	10450
451	10451
452	10452
453	10453
454	10454
455	10455
456	10456
457	10457
458	10458
459	10459
460	10460
461	10461
462	10462
463	10463
464	10464
465	10465
466	10466
467	10467
468	10468
469	10469
470	10470
471	10471
472	10472
473	10473
474	10474
475	10475
476	10476
477	10477
478	10478
479	10479
480	10480
481	10481
482	10482
483	10483
484</td	



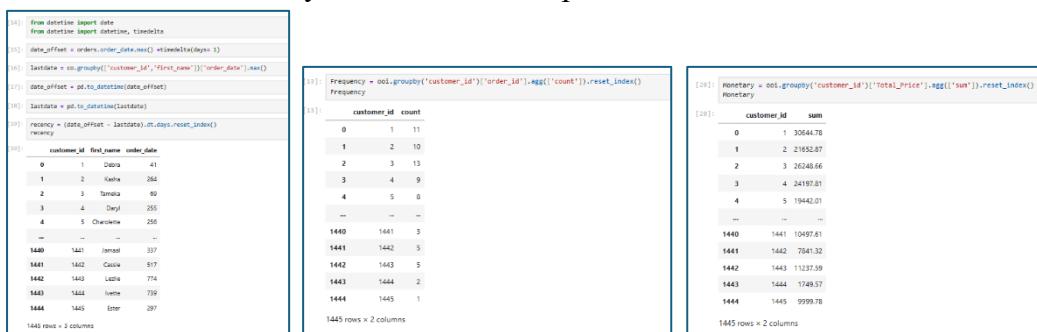
3. Calculate RFM Features for Customers

Compute Recency, Frequency, and Monetary values for each customer.

Recency: Days since last order

Frequency: Number of orders

Monetary: Total value of all purchases



4. Export Segmentation Results to SQL

Save the rfm_data with the segment label to SQL as customer_segments table.

customer_id	first_name	Recency	Frequency	Monetary
1	Debra	41	11	20644.78
2	Kosha	264	10	21652.87
3	Tamela	69	13	26248.66
4	Daryl	255	9	24197.81
5	Charlotte	256	8	19442.01
6	Lynsey	114	11	35856.83
7	Latahah	195	4	7707.64
8	Jacqueline	261	3	2602.58
9	Genevieve	246	9	26678.97
10	Hannah	128	21	37000.42
11	Deshawn	247	5	4079.49
12	Robby	250	8	27157.21
13	Ladaini	261	8	15830.32

Phase 4: Power BI – Visualization & Dashboarding Mandatory Tasks

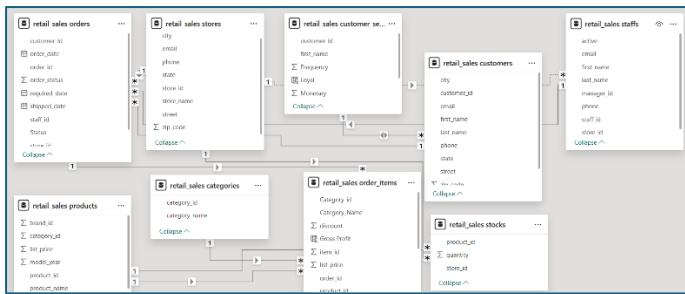
1. Connect Power BI to SQL

Import key tables like orders, products, order_items, customers, stores, and staffs from the SQL database.

- [Get data> Database> MySQL Database> Server name & database name> Navigator \(selection of tables\)> transform data](#)

2. Create Relationships Between Tables

Use Model View to define foreign key relationships based on the ER diagram (e.g., link orders.customer_id to customers.customer_id).



3. Sales Overview Report

Create visuals for:

Total sales over time (line or area chart)

Monthly sales trend

Total orders placed



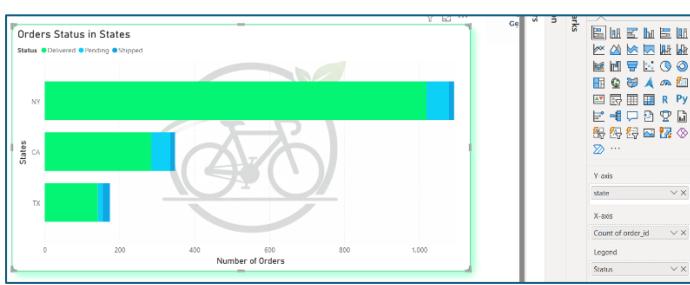
4. Top Products by Sales

Bar chart or table listing top-selling products by revenue or quantity.



5. Customer Purchase Analysis

Add a stacked bar chart to show purchase patterns by city or state using the customers and orders table.



6. Sales by Store Map

Use the map visual with store.state or store.zip_code to show sales distribution geographically.



7. Low Stock Alert Dashboard

Use conditional formatting and cards to display products with stock levels below a threshold (e.g., 10 units).

8. Interactive Filters and Slicers

Add slicers for:

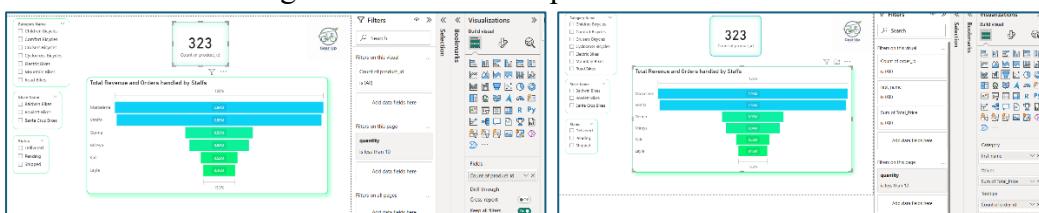
Order Status

Product Category

Store

9. Staff Performance Report

Table or chart showing total sales/revenue per staff member based on handled orders.



10. Consolidated Dashboard Page

Final report page with KPIs:

Total Revenue

Active Customers

Avg Order Value

Total Orders



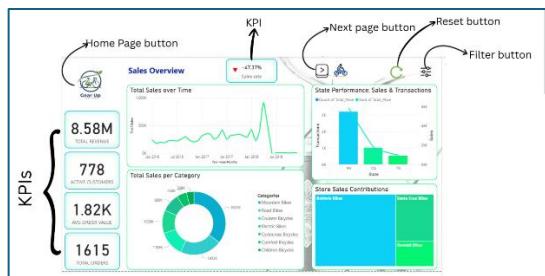
Buttons:



Sales Overview:

- Total sales Over time
- Total sales per category
- State performance: Sales & Transactions

• Store Sales Contribution



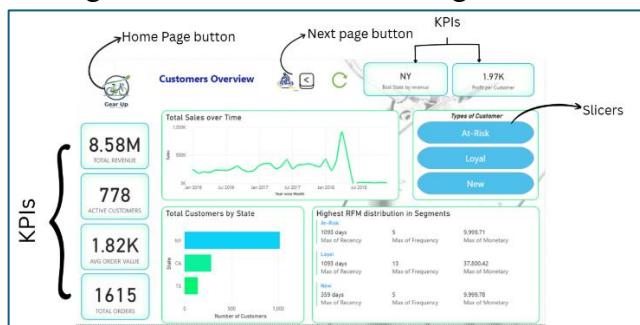
Products Overview:

- Total Sales over Time
 - Product sales by stores
 - Top 5 best product performers



Customers overview:

- Total sales over time
 - Total customers by state
 - Highest RFM distribution in segments



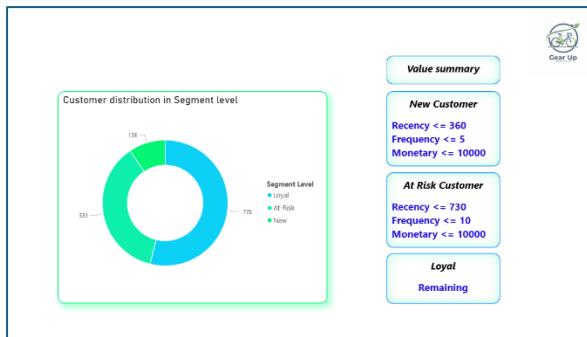
11. Import customer segments Table

Load the segmentation result (exported from Python) from SQL into Power BI.

- Get data> Database> MySQL Database> Server name & database name> Navigator (selection of tables)> transform data

12. Visualize Customer Segments

Use Pie Chart / Donut / TreeMap to show the distribution of segments (e.g., Loyal, New, At-Risk).



13. Segment-Level Revenue Breakdown

Add a bar chart that shows total revenue per customer segment.



14. Use Segments as Report Filters

Enable filtering across dashboards by customer segment using slicers.

Overall insights:

- Sales was high during mid (April, June, September) of the calendar year, which is the summer season.

Q1	Jan, Feb, Mar	Low Sales
Q2	April, May, June	High Sales
Q3	July, Aug, Sept	High sales
Q4	Oct, Nov, Dec	Low Sales

- Start of the year and end of the year has low sales because of fall and winter seasons.
- Most sold product category is Mountain bikes: which clear mountain riders rides during summer.
- But least sales are of children category bicycles: in past three years they have bought during summer season.
- In the last one year (2018) more than 100 new customers have come which gave average profit of 1.27k.
- Since 2016 to 2018, loyal customers are higher (778) but at-risk customers are little closer to the loyal customers' count.