Q3)

Performance

INFERENCES :

- **Create:**
    - Single inserts and writes perform nearly the same across both designs.
    - Bulk operations show that the **transaction-centric model is faster**, especially for bulk writes.
- **Read:**
    - Looking up customer-related data is faster in the **customer-centric design**.
    - Invoice lookups are significantly quicker in the **transaction-centric design**.
    - General item searches and top-customer queries show almost identical performance in both designs.
    - Counting transactions is faster in the **transaction-centric design**.
- **Update:**
    - Updating customer details is quicker in the **customer-centric design**.
    - Updating invoice-specific fields (like status) is faster in the **transaction-centric design**.
    - Country-based updates are slightly faster in the **customer-centric model**.
- **Delete:**
    - Customer deletions are more efficient in the **customer-centric design**.
    - Deleting low-value entries is faster in the **customer-centric design**, while country-based deletes are quicker in the **transaction-centric model**.

Conclusion

- Use the **transaction-centric design** if your application frequently handles bulk operations, invoice lookups, or transaction counts.

- Use the **customer-centric design** if the main focus is on managing customer profiles, retrieving all customer data, or efficiently updating/deleting customer-level information.

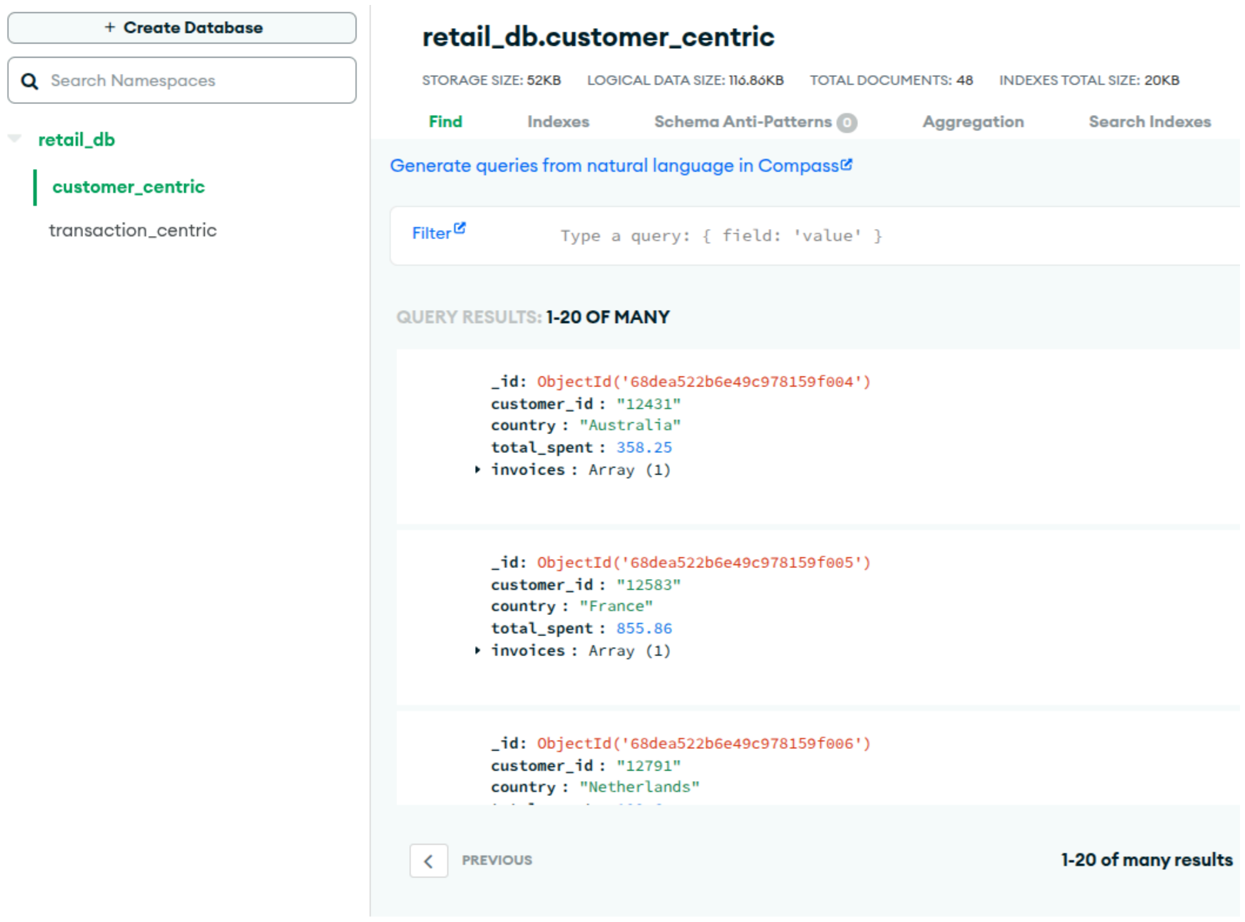
- For workloads that mix both invoice-heavy and customer-heavy operations, a hybrid approach or careful indexing strategy may provide the best balance.

```
========================== RESULTS ==========================

Operation                    Transaction-Centric   Customer-Centric
----------------------------------------------------------------------
CREATE - Single Insert              0.000337            0.000335
```

CREATE - Single Write	0.000237	0.000317
CREATE - Bulk Insert	0.003462	0.004345
CREATE - Bulk Write	0.004449	0.012009
READ - Customer Data Lookup	0.000737	0.000347
READ - Invoice Lookup	0.000136	0.001277
READ - Item Search	0.003403	0.003401
READ - Transaction Count	0.000782	0.001959
READ - Top Customers	0.000734	0.000736
UPDATE - Customer Data	0.000487	0.000301
UPDATE - Invoice Status	0.000514	0.000389
UPDATE - Flag Country	0.002344	0.002152
DELETE - Customer Data	0.000337	0.000261
DELETE - Low Value	0.000795	0.000568
DELETE - By Country	0.000959	0.001434

Q4)



retail_db.customer_centric

STORAGE SIZE: 52KB LOGICAL DATA SIZE: 116.86KB TOTAL DOCUMENTS: 48 INDEXES TOTAL SIZE: 20KB

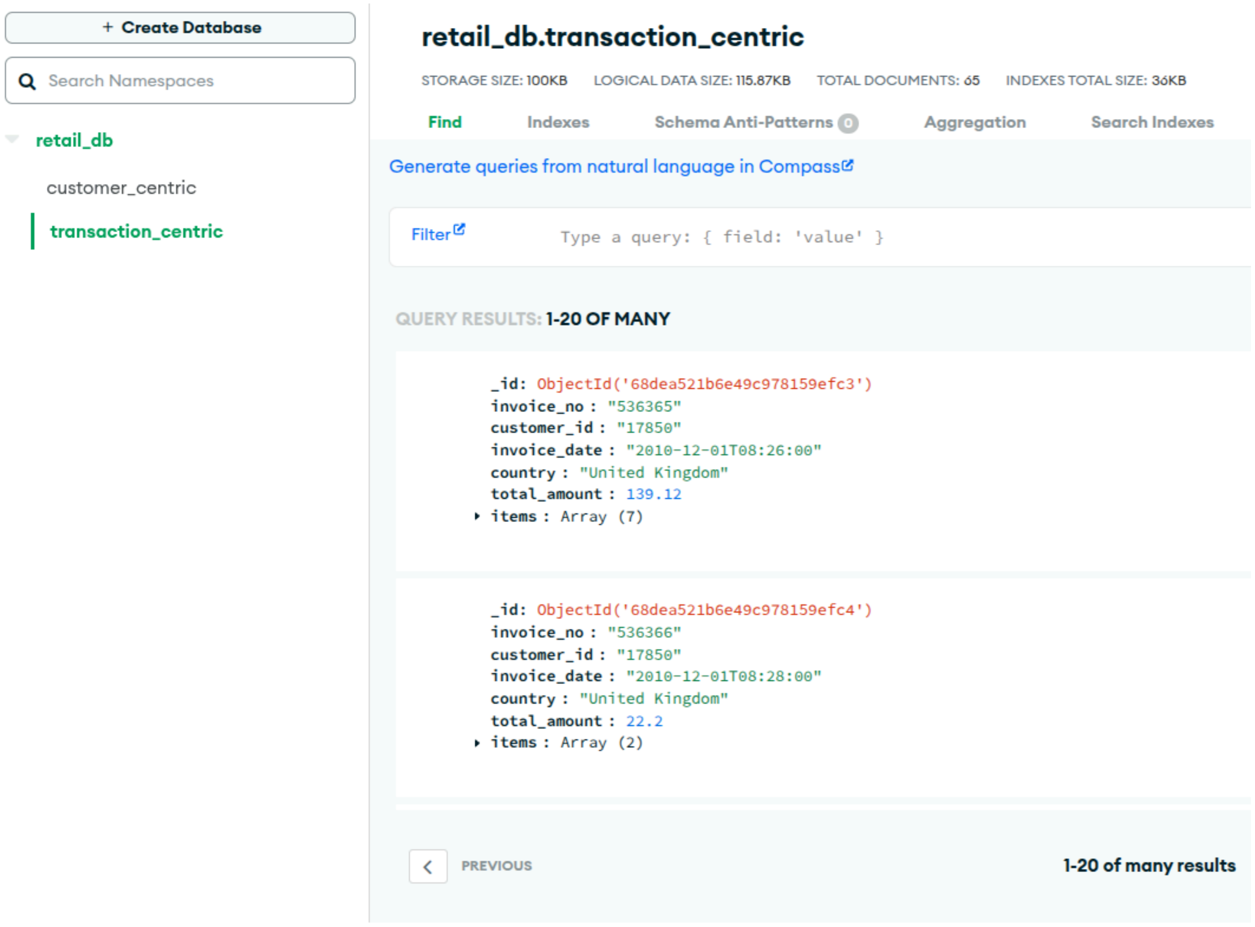
QUERY RESULTS: 1-20 OF MANY

```
_id: ObjectId('68dea522b6e49c978159f004')
customer_id : "12431"
country : "Australia"
total_spent : 358.25
invoices : Array (1)

_id: ObjectId('68dea522b6e49c978159f005')
customer_id : "12583"
country : "France"
total_spent : 855.86
invoices : Array (1)

_id: ObjectId('68dea522b6e49c978159f006')
customer_id : "12791"
country : "Netherlands"
```

1-20 of many results



retail_db.transaction_centric

STORAGE SIZE: 100KB LOGICAL DATA SIZE: 115.87KB TOTAL DOCUMENTS: 65 INDEXES TOTAL SIZE: 36KB

Find
Indexes
Schema Anti-Patterns 0
Aggregation
Search Indexes
Generate queries from natural language in Compass
Filter
Type a query: { field: 'value' }

QUERY RESULTS: 1-20 OF MANY

```
_id: ObjectId('68dea521b6e49c978159efc3')
invoice_no : "536365"
customer_id : "17850"
invoice_date : "2010-12-01T08:26:00"
country : "United Kingdom"
total_amount : 139.12
items : Array (7)
```

```
_id: ObjectId('68dea521b6e49c978159efc4')
invoice_no : "536366"
customer_id : "17850"
invoice_date : "2010-12-01T08:28:00"
country : "United Kingdom"
total_amount : 22.2
items : Array (2)
```

1-20 of many results