Cost Model Formulas for Distributed NoSQL Databases

ARROUET–LE BRIGNONEN Aubin, BAUZIN Jérémy et AUPERIN Hugo

1 Query Size (Input & Output)

In this course, the size of a query is modeled as the size of the JSON-like object sent to the servers. It includes:

- Filter fields (key + value),
- Projected fields (key + boolean/int),
- A key overhead of 12 bytes per key,
- An additional 12 bytes for each nested object level.

1.1 Value Sizes

$$\text{int/boolean} = 8\text{ B}, \quad \text{string} = 80\text{ B}, \quad \text{date} = 20\text{ B}, \quad \text{key overhead} = 12\text{ B}$$

1.2 Query Input Size

If the query contains k fields with values v_i:

$$\text{size}_{input} = \sum_{i=1}^{k} (12 + \text{size}(v_i)) + 12 \cdot (\text{nesting levels})$$

1.3 Query Output Size

If each output document contains m fields:

$$\text{size}_{msg} = \sum_{j=1}^{m} (12 + \text{size}(value_j))$$

Total output volume:

$$\text{vol}_{output}(q) = res_q \cdot \text{size}_{msg}$$

1.4 Example: Query Q1

Consider the query:

```
SELECT S.quantity, S.location FROM Stock S WHERE S.IDP = $IDP  AND  S.IDW = $IDW;
```

We assume the following:

- `IDP` is an integer (8 B),
- `IDW` is an integer (8 B),
- `quantity` is projected and modeled as a boolean/int (8 B),
- `location` is projected and modeled as a boolean/int (8 B),
- Each field contributes a key overhead of 12 B,
- The query is wrapped in a `Stock` object, adding one extra key of 12 B.

Input Size Calculation

Value sizes:

$$8 + 8 + 8 + 8 = 32 \text{ B}$$

Key sizes:

$$4 \times 12 = 48 \text{ B}$$

Nesting overhead:

$$12 \text{ B}$$

Total input size:

$$\text{size}_{input} = 32 + 48 + 12 = 92 \text{ B}$$

Output Message Size

Fields returned:

- `quantity`: $12 + 8$ B,
- `location`: $12 + 8$ B.

Thus:

$$\text{size}_{msg} = (12 + 8) + (12 + 8) = 40 \text{ B}$$

Total output volume for res_q results:

$$\text{vol}_{output}(q) = res_q \cdot 40 \text{ B}$$

2 Time Cost

Total execution time:

$$\text{time}_{DB} = \sum_{q=1}^{Q} \left(\frac{\text{vol}_{network}(q)}{\text{bandwidth}_{network}} + \frac{\text{vol}_{RAM}(q)}{\text{bandwidth}_{RAM}} \right) \times freq(q)$$

2.1 Network Volume

Filter queries

$$\text{vol}_{network}(q) = S \cdot \text{size}_{input} + res_q \cdot \text{size}_{msg}$$

Aggregate queries

$$\text{vol}_{network}(q) = S \cdot \text{size}_{input} + shuffle \cdot \text{size}_{msg} + res_q \cdot \text{size}_{msg}$$

2.2 RAM Volume

Per server:

$$\text{vol}_{RAM}(q, n) = index_q + sel_{att} \cdot coll_{q,n} \cdot size_{doc}(q)$$

Global:

$$\text{vol}_{RAM}(q) = \max_n \left(\text{vol}_{RAM}(q, n) \right)$$

3 Environmental Cost

3.1 Carbon from Network

$$impact_{network}(q) = \text{vol}_{network}(q) \cdot CO2_{network}$$

3.2 Carbon from RAM

$$impact_{RAM}(q) = \mathrm{vol}_{RAM}(q) \cdot CO2_{RAM}$$

3.3 Total Carbon Footprint

$$impact(DB) = \sum_{q=1}^{Q} (impact_{network}(q) + impact_{RAM}(q)) \times freq(q)$$

4 Financial Cost

4.1 Monthly Price

$$price_{DB} = price_s \cdot \max\left(\frac{vol_{DB} \cdot 3}{capacity_{storage}}, \frac{vol_{DB}}{capacity_{RAM}} \cdot 2\right) + externalFees \cdot \sum_{q=1}^{Q} vol_{external}(q) \cdot freq(q)$$

5 Sharding and Indexing Strategy

5.1 Sharding

$$S = \begin{cases} 1 & \text{if filtered on the sharding key} \\ \#shards & \text{otherwise} \end{cases}$$

5.2 When to Use Sharding

- Dataset too large for one server.
- Queries often filter on the sharding key.

5.3 When to Avoid Sharding

- Queries mostly filter on non-sharding attributes.
- Aggregations require scanning all shards.

6 Algorithm Choice

6.1 Index Lookup

$$\mathrm{vol}_{RAM}(q) = index_q + sel_{att} \cdot coll \cdot size_{doc}$$

where $index_q \approx 1$ MB.

6.2 Full Scan

Used when no index exists or when selectivity is high:

$$sel_{att} = 1$$