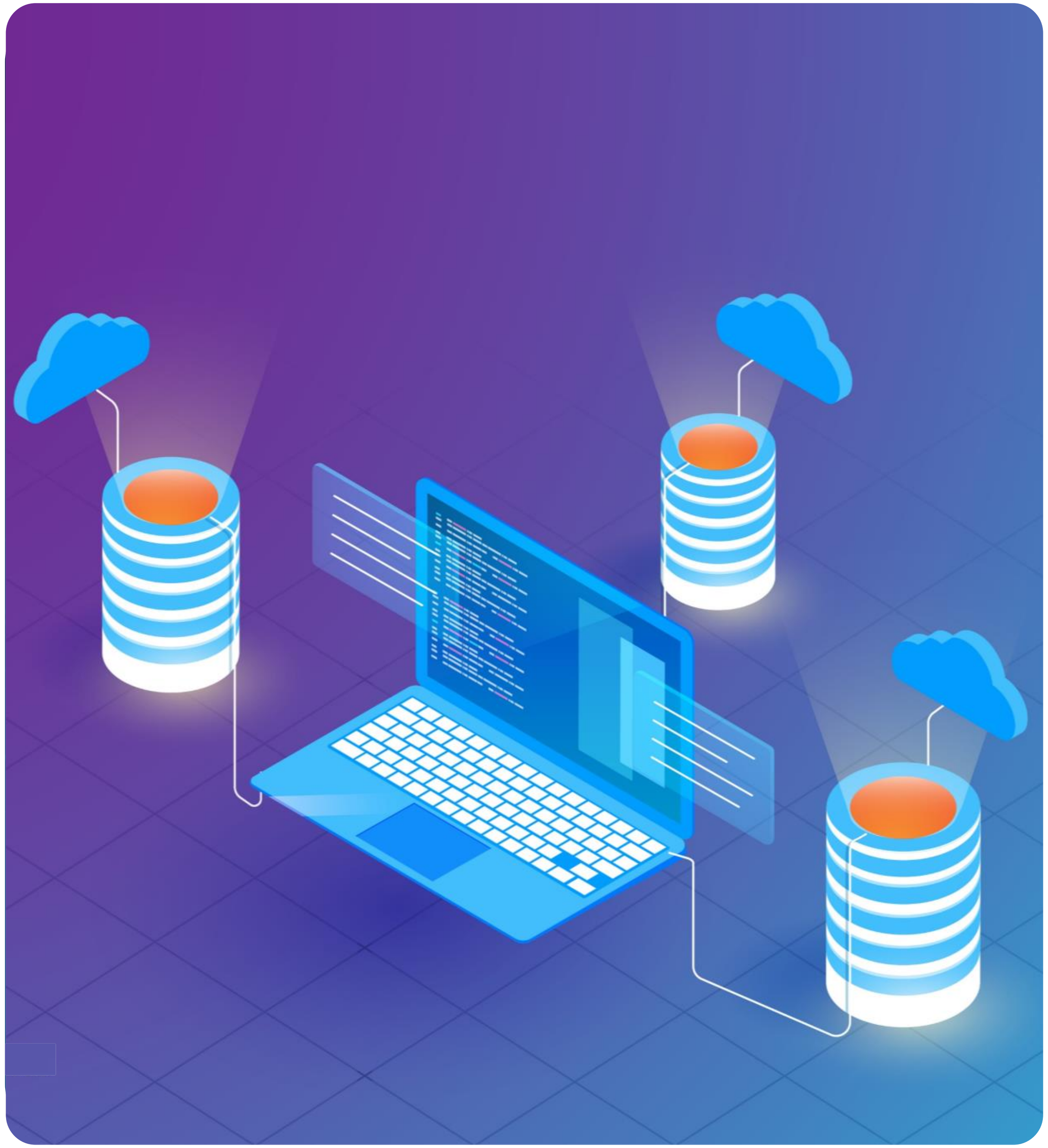
BeSA
Become a Solutions Architect

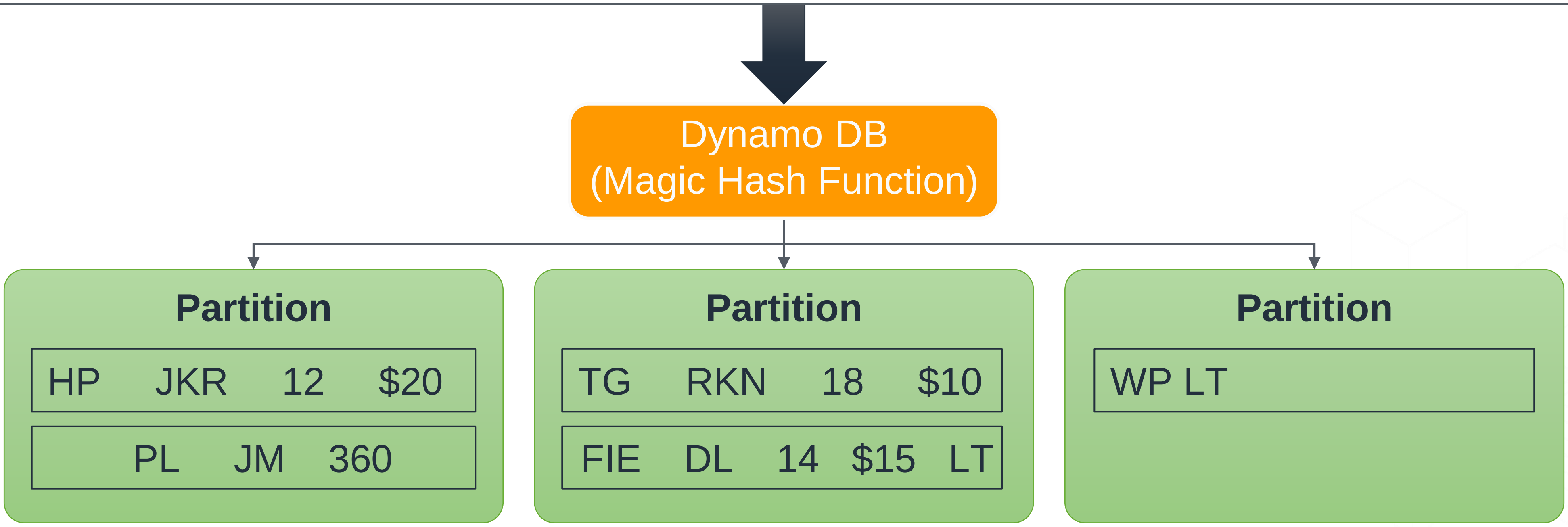
Databases in Cloud



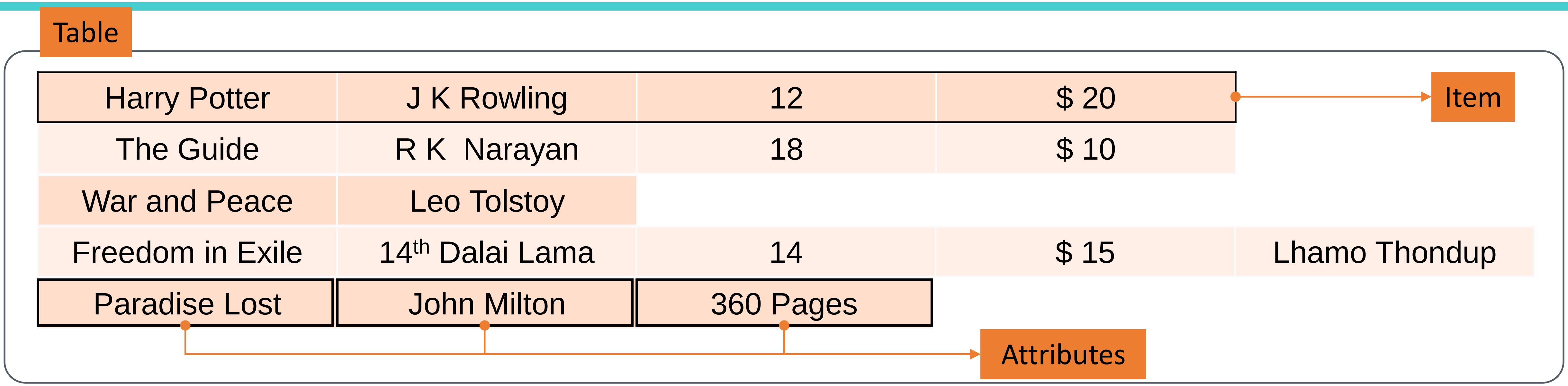
Amazon Dynamo DB

Dynamo DB

Harry Potter	J K Rowling	12	$ 20	
The Guide	R K Narayan	18	$ 10	
War and Peace	Leo Tolstoy			
Freedom in Exile	14th Dalai Lama	14	$ 15	Lhamo Thondup
Paradise Lost	John Milton	360 Pages		

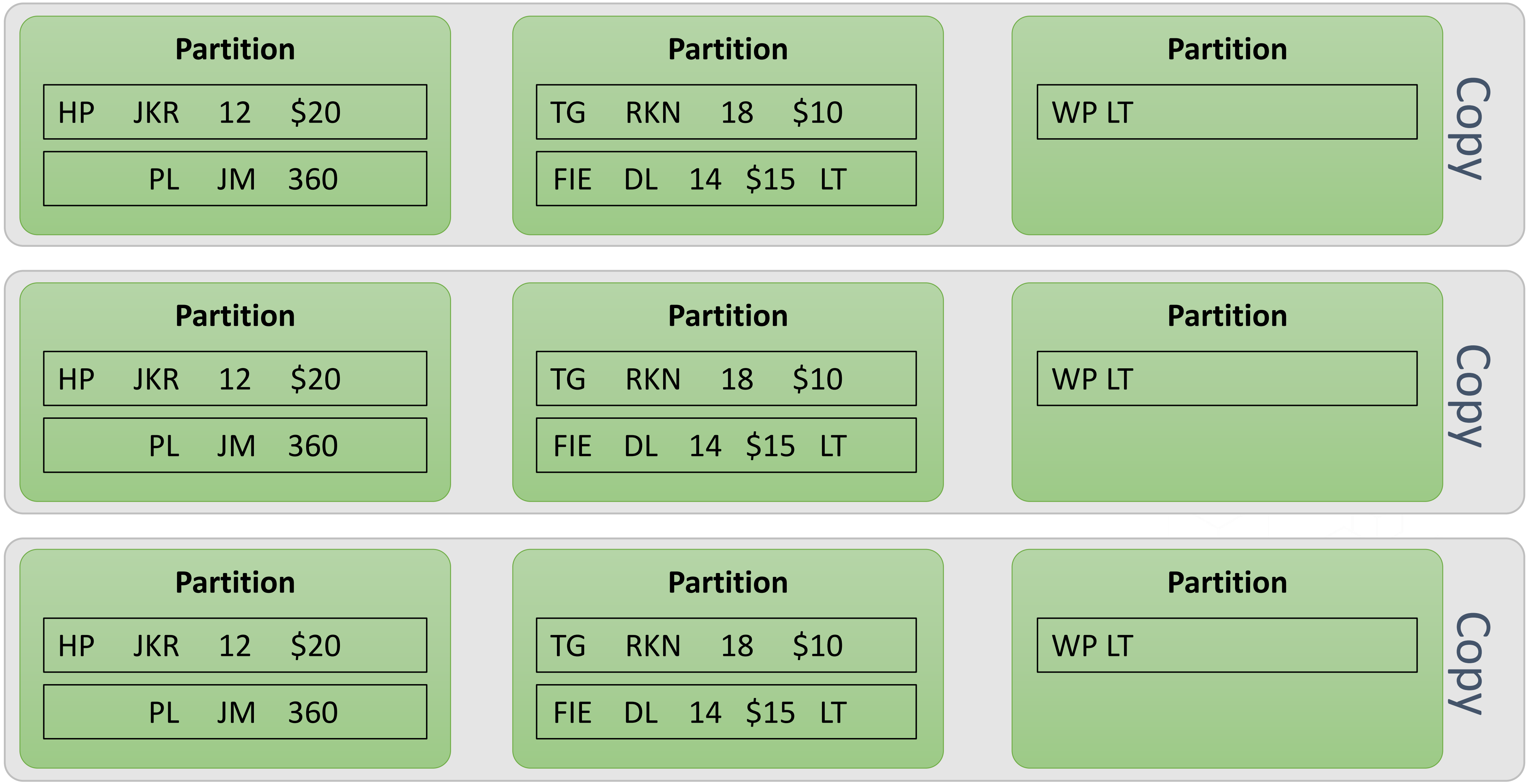


Dynamo DB – Tables, Items, Attributes

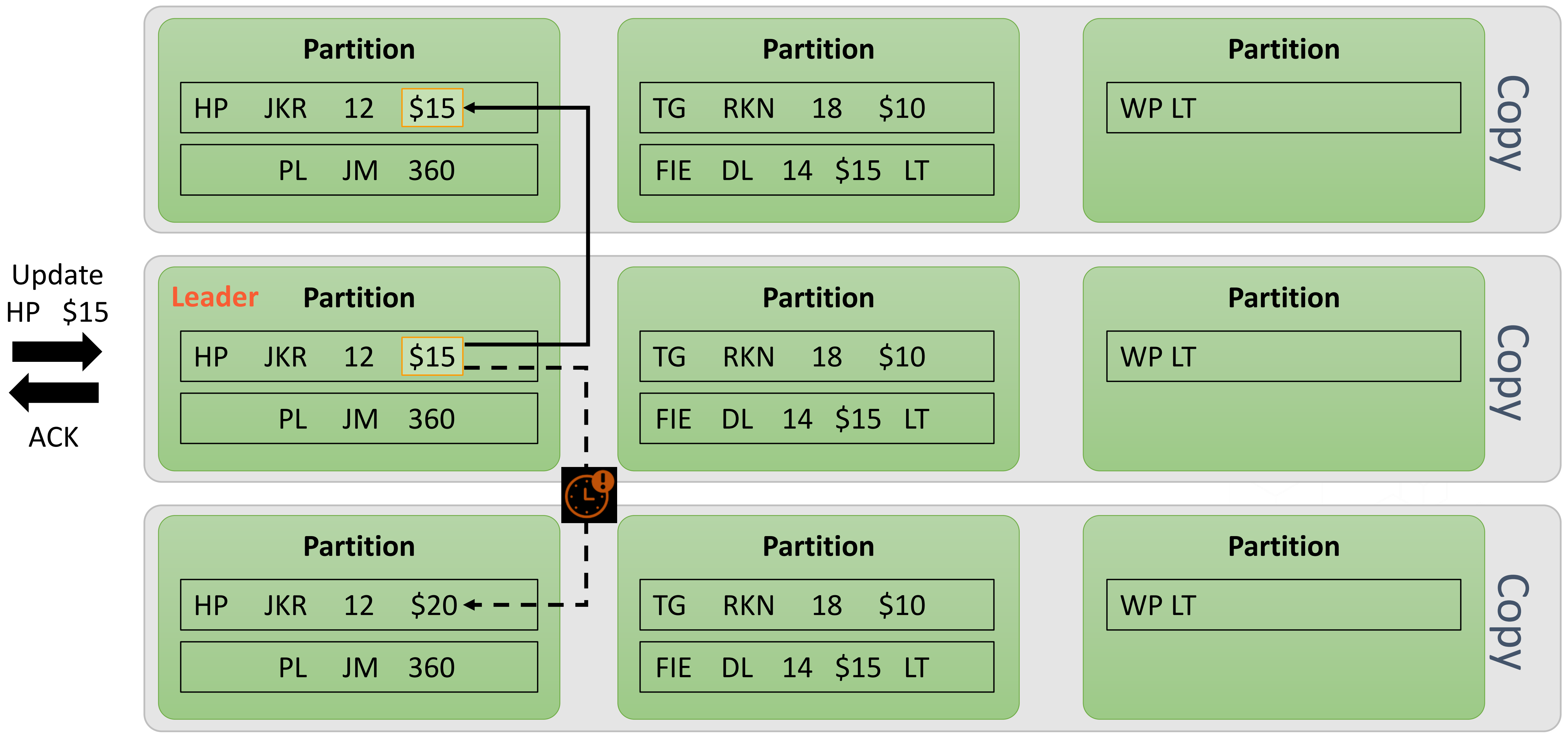


- **Tables** – Similar to other database systems, DynamoDB stores data in tables. A *table* is a collection of data.
- **Items** – Each table contains zero or more items. An *item* is a group of attributes that is uniquely identifiable among all of the other items.
- **Attributes** – Each item is composed of one or more attributes. An *attribute* is a fundamental data element, something that does not need to be broken down any further.

Dynamo DB Partitions



Write/Update Operation

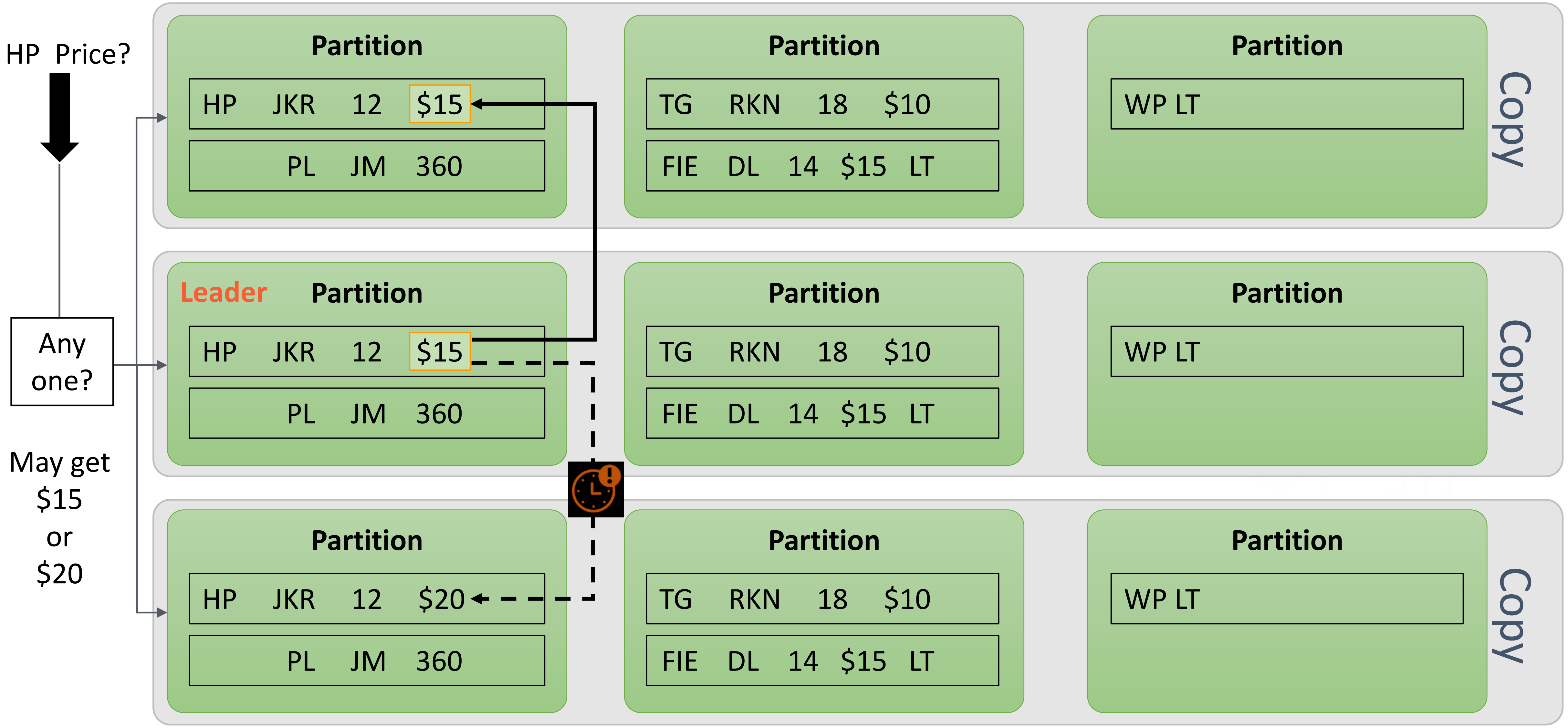


Paying for a dinner

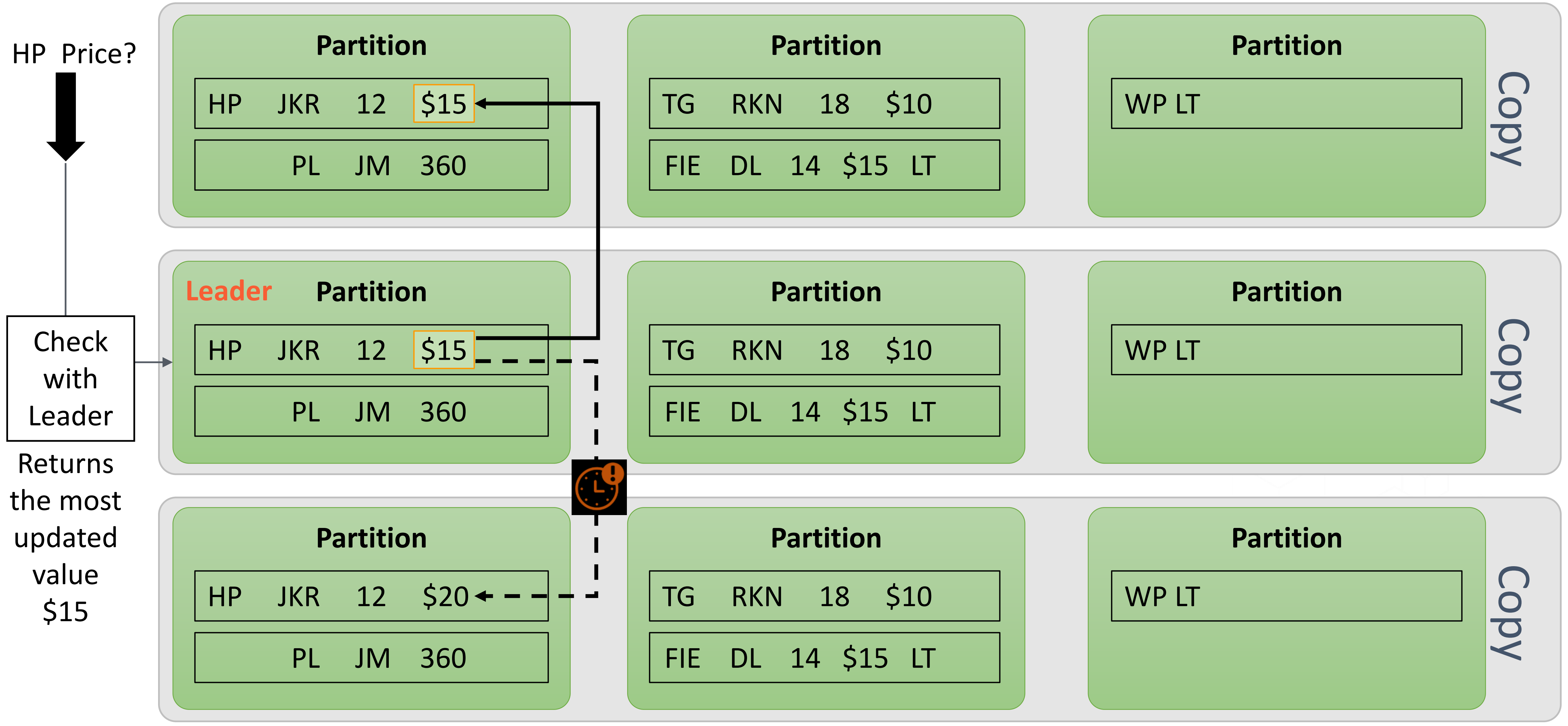
Pay by Cash

Pay by Credit Card

Read - Eventual Consistency

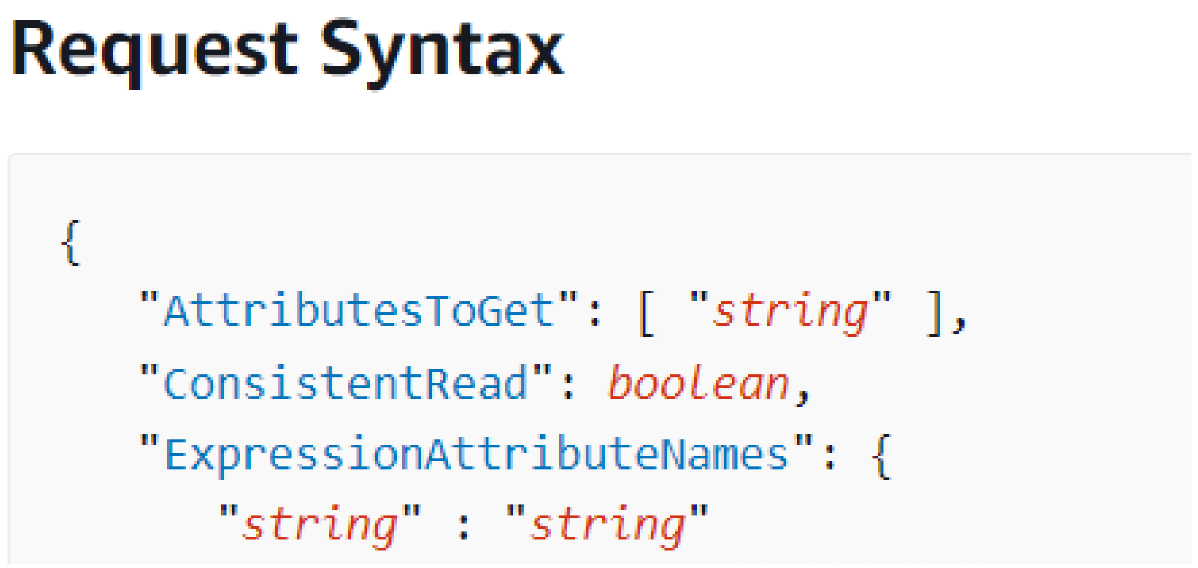


Read - Strong Consistency



Read Operation

- Read (GetItem)
    - Eventual Read
        - Default
    - Strong Read
        - Have to specify
- **ConsistentRead**
    - Determines the read consistency model: If set to true, then the operation uses strongly consistent reads; otherwise, the operation uses eventually consistent reads.
    - Type: Boolean
    - Required: No



Request Syntax

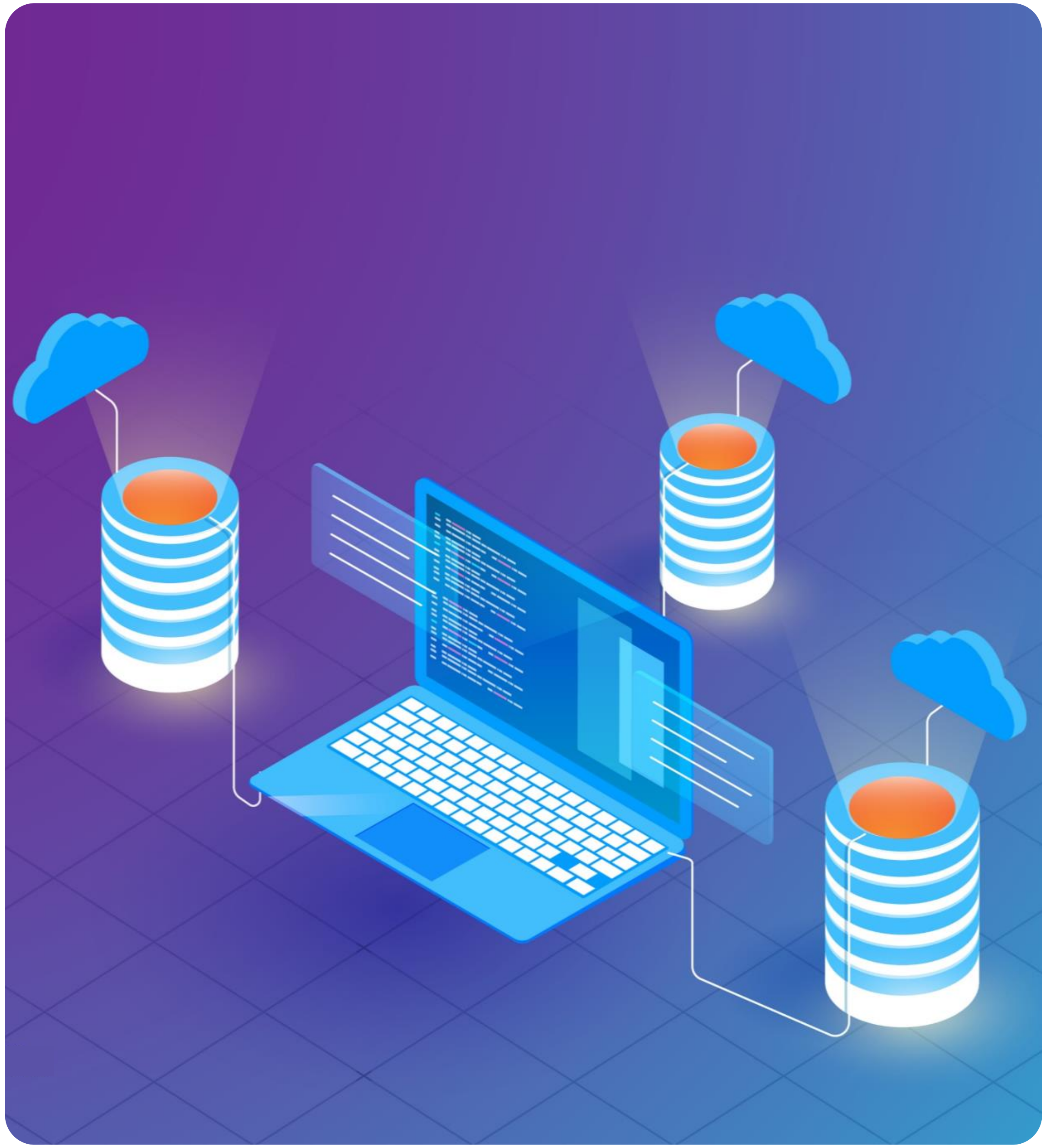
```
{
   "AttributesToGet": [ "string" ],
   "ConsistentRead": boolean,
   "ExpressionAttributeNames": {
      "string" : "string"
```

Attributes Example

```
{
  "Artist": "No One You Know",
  "SongTitle": "My Dog Spot",
  "AlbumTitle": "Hey Now",
  "Price": 1.98,
  "Genre": "Country",
  "CriticRating": 8.4
}
```

```
{
  "PersonID": 102,
  "LastName": "Jones",
  "FirstName": "Mary",
  "Address": {
        "Street": "123 Main",
        "City": "Anytown",
        "State": "OH",
        "ZIPCode": 12345
  }
}
```

- Most of the attributes are scalar, which means that they can have only one value. Strings and numbers are common examples of scalars.
- Some of the items have a nested attribute (Address). DynamoDB supports nested attributes up to 32 levels deep.



Partition Key

and

Sort Key

Partition Key and Sort Key

Partition Key – Title

Primary Key = Partition Key

Harry Potter	J K Rowling	12	$ 20	
The Guide	R K Narayan	18	$ 10	
War and Peace	Leo Tolstoy			
Freedom in Exile	14th Dalai Lama	14	$ 15	Lhamo Thondup
Paradise Lost	John Milton	360 Pages		

Harry Potter	J K Rowling	15	$ 25

2nd Edition

Title – Not Unique, so can't be used as Partition Key

Harry Potter	1st Edition	J K Rowling	12	$ 20
Harry Potter	2nd Edition	J K Rowling	15	$ 25

+

Partition Key

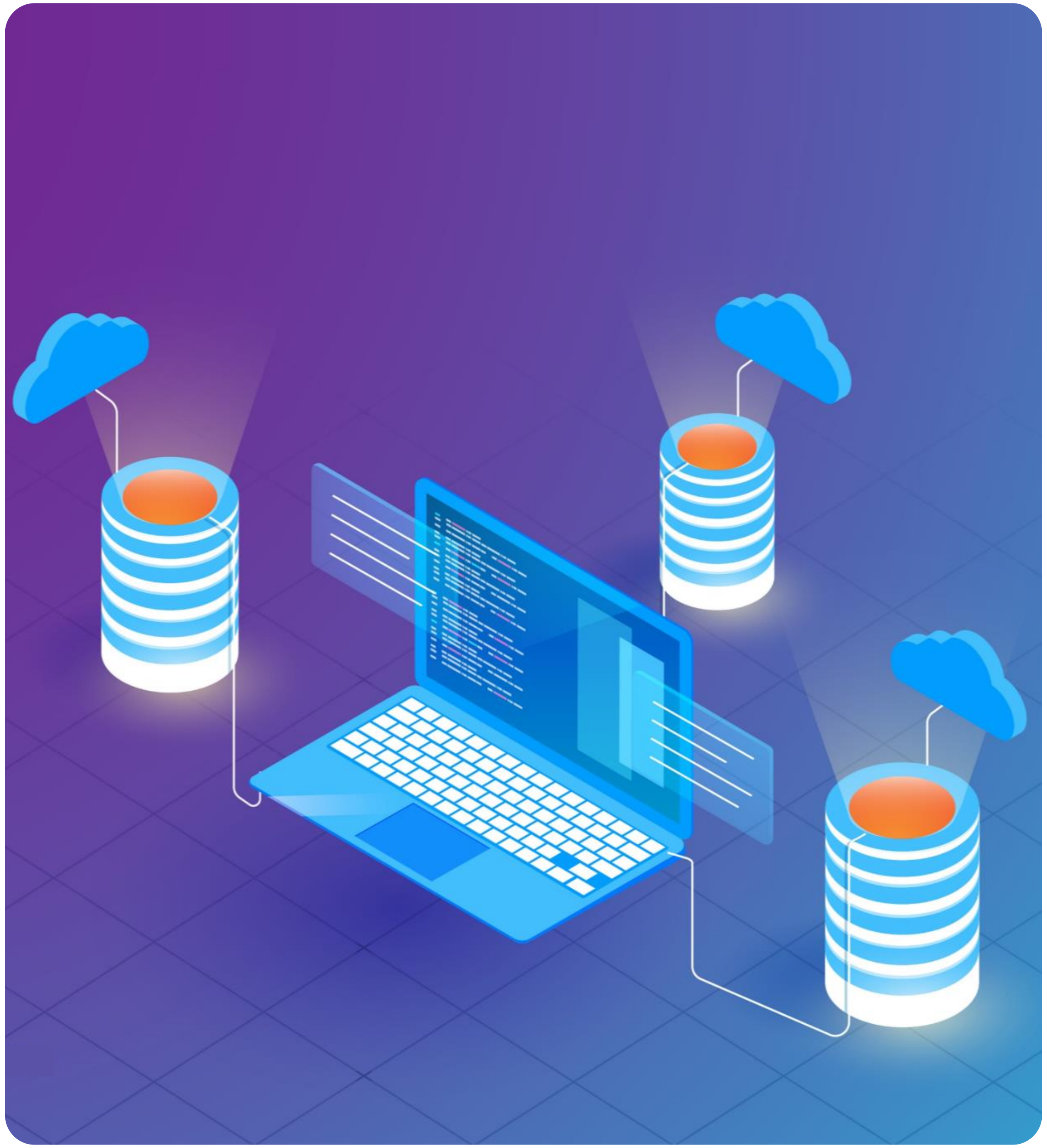
Sort Key

Primary Key = (Partition Key + Sort Key)

Partition Key - Examples

- Use high-cardinality attributes. These are attributes that have distinct values for each item, like `emailid`, `employee_no`, `customerid`, `sessionid`, `orderid`, and so on.

- Use composite attributes. Try to combine more than one attribute to form a unique key, if that meets your access pattern. For example, consider an orders table with `customerid#productid#countrycode` as the partition key and `order_date` as the sort key, where the symbol # is used to split different field.



Secondary Indexes

Why we need indexes?

Partition Key – Title

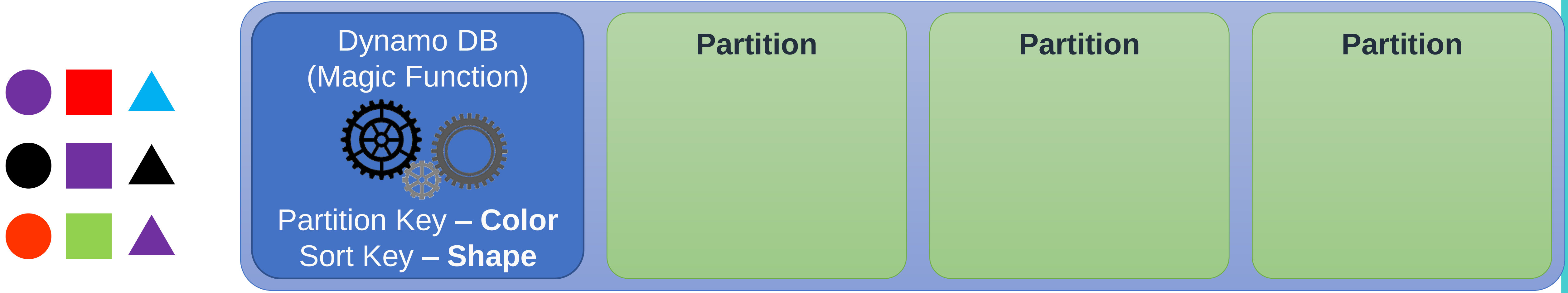
Sort Key – Author

Primary Key = Partition Key + Sort Key

Harry Potter	J K Rowling	12	$ 20	
The Guide	R K Narayan	18	$ 10	
War and Peace	Leo Tolstoy			
Freedom in Exile	14th Dalai Lama	14	$ 15	Lhamo Thondup
Paradise Lost	John Milton	360 Pages		

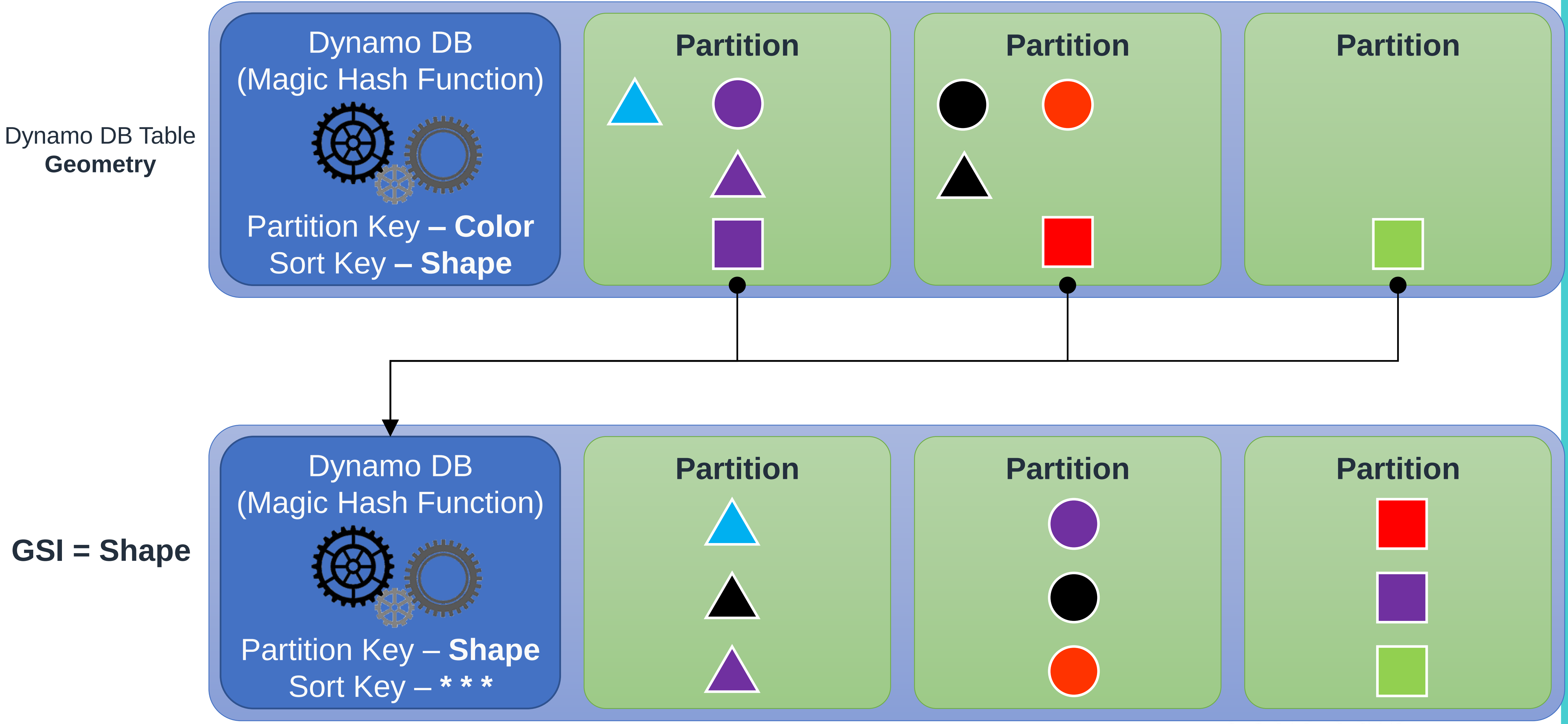
- Questions?
  - Which book has 18 chapters?
  - Give me all the books of price less than $15?

Dynamo DB Table – Geometry

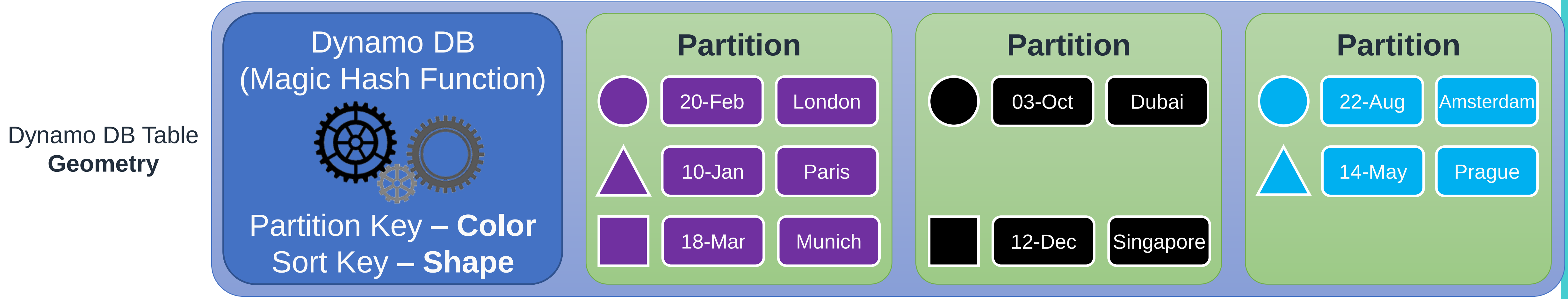


- Query
  - How many Purple Color Shapes we have?
  - Give me details of Purple Color Square?
  - How many Squares we have?
    - Current Primary Key is not efficient for this question. That’s where Global Secondary Index is useful.

Global Secondary Index



Local Secondary Index



- Query
  - How many Purple Color items for London?
  - Give me details of Purple Color items on 18-Mar?
- A single partition may have lots of items and we need an efficient way to find specific items. That’s where Local Secondary Index is useful.

Indexes

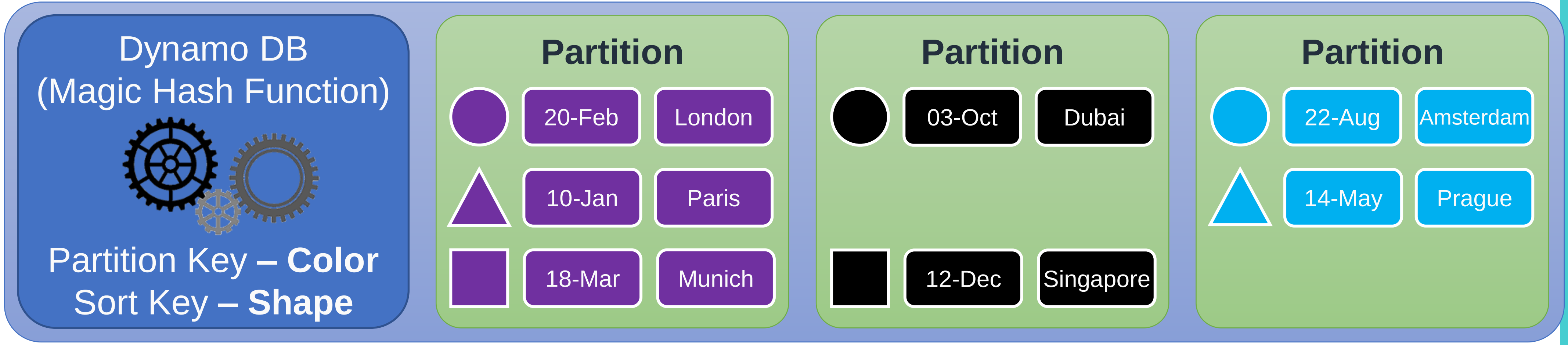
- DynamoDB supports two kinds of indexes:

- Global secondary index (GSI) – An index with a partition key and sort key that can be different from those on the table.

- Local secondary index (LSI) – An index that has the same partition key as the table, but a different sort key.

Local Secondary Index

Dynamo DB Table
Geometry



Partition Key + Sort Key (of your choice)

LSI 1	Color + Date
LSI 2	Color + City

GSI vs LSI

Compare	Global Secondary Index (GSI)	Local Secondary Index (LSI)
Queries	Across all partitions	In a single partition
Size Limit	No size limitations	Can’t exceed 10 GB
Provisioned throughput	Separate from table	Shares with the tables
Read Consistency	Only Eventual	Strong or Eventual
Maximum	20	5
Creation	Anytime	Only with table creation
Deletion	Anytime	Only with table deletion

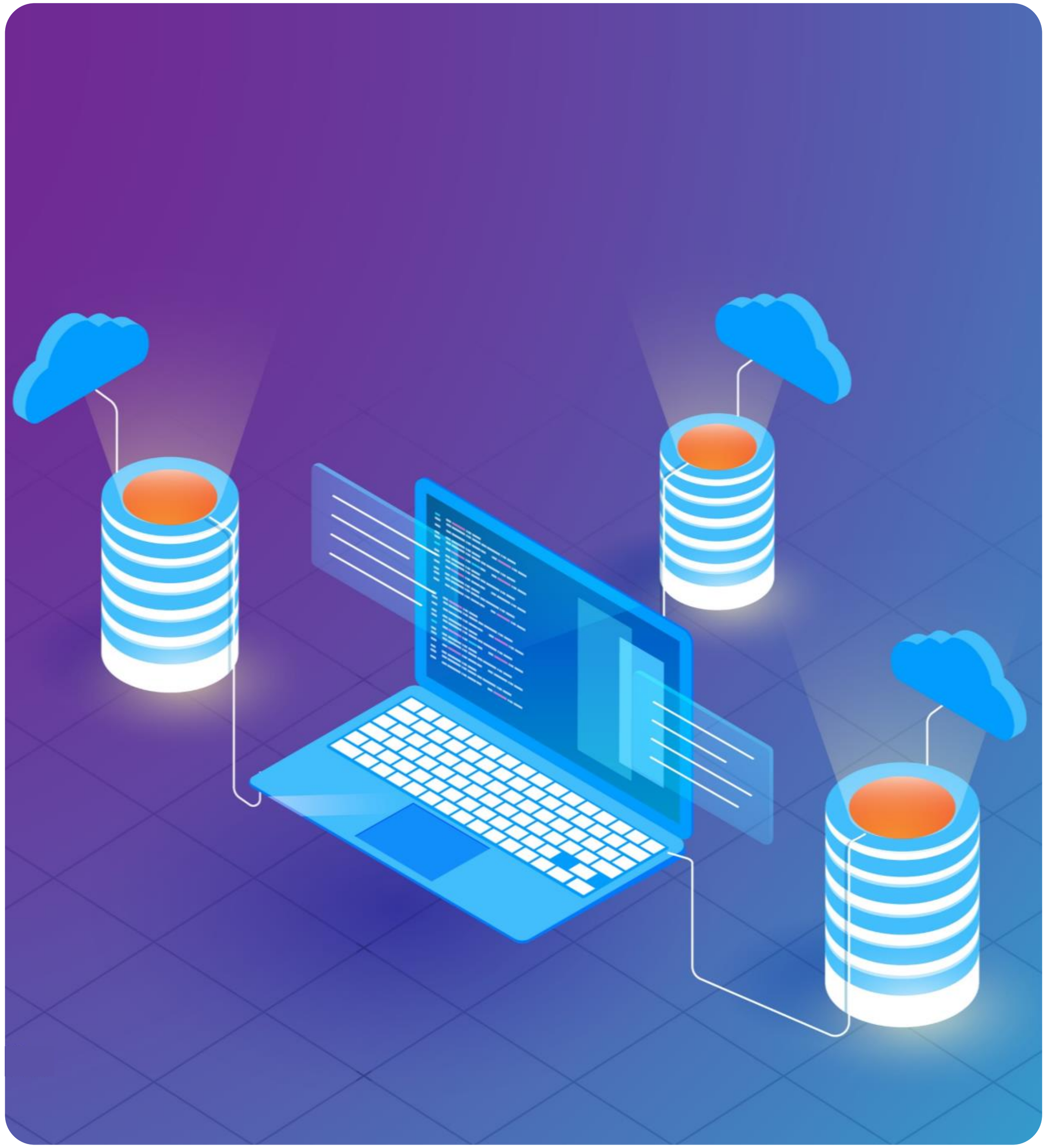
Query vs Scan

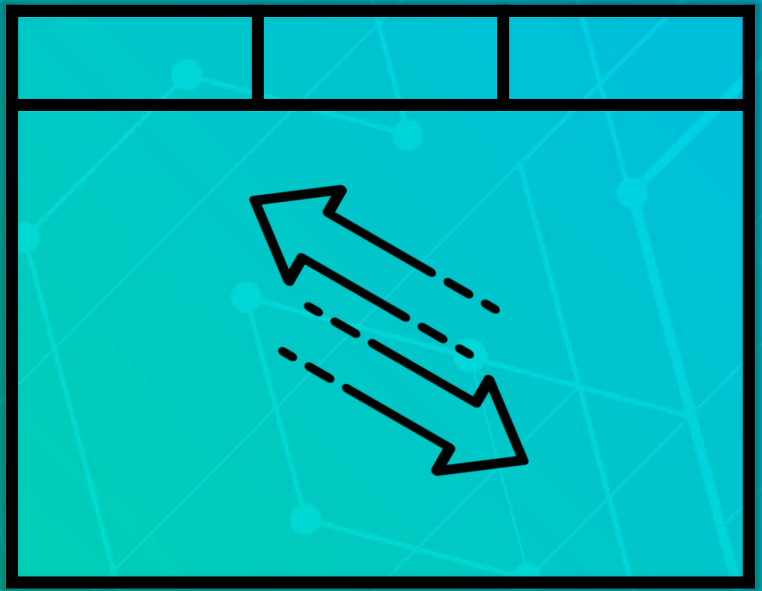
Partition Key – Title

Primary Key = Partition Key

Harry Potter	J K Rowling	12	$ 20	
The Guide	R K Narayan	18	$ 10	
War and Peace	Leo Tolstoy			
Freedom in Exile	14th Dalai Lama	14	$ 15	Lhamo Thondup
Paradise Lost	John Milton	360 Pages		

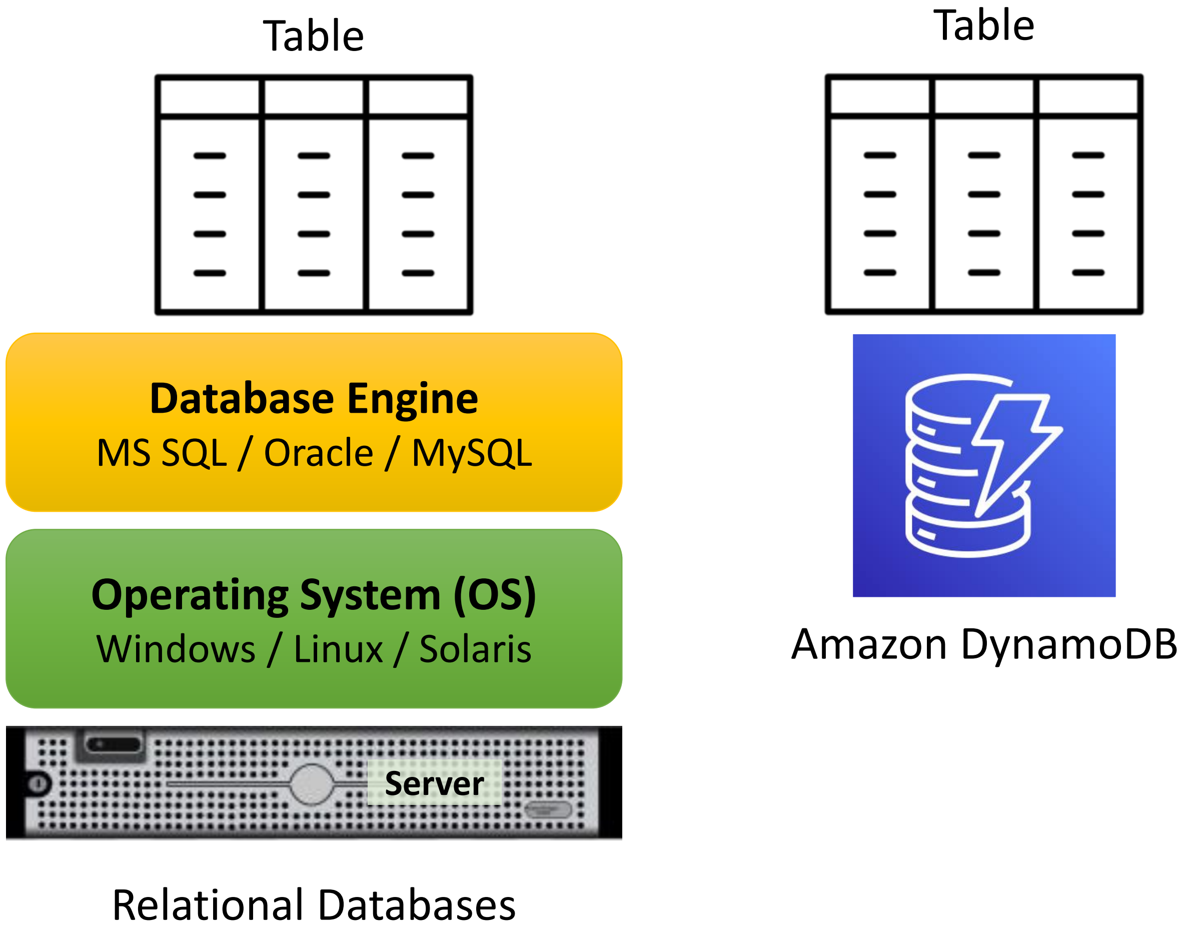
- Give me the book with title – War and Peace?
  - Query (Faster)
- Give me the book whose author is John Milton?
  - Scan (Slower)





DynamoDB Read/Write Capacity

Dynamo DB Performance



- Dynamo DB Table performance is controlled by configuring:
  - Read Capacity Unit (RCU)
  - Write Capacity Unit (WCU)
- One RCU = one strongly consistent read per second, or two eventually consistent reads per second, for an item up to 4 KB in size.
- One WCU = one write per second for an item up to 1 KB in size.

Dynamo DB capacity modes

- Amazon DynamoDB has two read/write capacity modes for processing reads and writes on your tables:

	Provisioned Mode	**On-Demand Mode**
What?	Provision the capacity (RCU/WCU) to run at a specific limit	No limit scaling, serving thousands of requests per second without capacity planning
Charges	Pay for provisioned capacity (whether you use it or not)	Pay only for read and write you perform
Benefit	Controls cost and supports capacity reservation	Instantly accommodates your workload as traffic ramps up or down
Suitable for	Steady state and predictable traffic	Random and unpredictable traffic
Floor and ceiling	Can be setup using Auto Scaling	Can scale to zero, no ceiling