17th edition SQLDay Conference

12-14 May 2025, WROCŁAW + ONLINE

Platinum sponsors

Gold sponsors

Silver sponsors

Partitioning Approaches in Database Systems:

Insights from Oracle, SQL Server, and PostgreSQL

Sarah Kossakowska, Szymon Lasota

SQL Day

Szymon Lasota

Cloud Architect @ Lingaro

szymon.lasota@lingarogroup.com

https://www.linkedin.com/in/szymon-lasota/

Microsoft CERTIFIED SECURITY OPERATIONS ANALYST ASSOCIATE
Microsoft CERTIFIED AZURE ENTERPRISE DATA ANALYST ASSOCIATE
Microsoft CERTIFIED FABRIC ANALYTICS ENGINEER
Microsoft CERTIFIED DEVOPS ENGINEER EXPERT
Microsoft CERTIFIED CYBERSECURITY ARCHITECT EXPERT
Microsoft CERTIFIED AZURE SOLUTIONS ARCHITECT EXPERT
Microsoft CERTIFIED AZURE DATABASE ADMINISTRATOR ASSOCIATE
Microsoft CERTIFIED AZURE ADMINISTRATOR ASSOCIATE
Microsoft CERTIFIED AZURE DEVELOPER ASSOCIATE
Microsoft CERTIFIED AZURE DATA ENGINEER ASSOCIATE
Microsoft CERTIFIED AZURE NETWORK ENGINEER ASSOCIATE
Microsoft CERTIFIED AZURE SECURITY ENGINEER ASSOCIATE

Microsoft CERTIFIED TRAINER MCT

Since 2020

Sarah Kossakowska

Data Integration Engineer @ Lingaro

sarah.kossakowska@lingarogroup.com

SQLDay 2025 Partitioning Agenda

01
Partitioning intro

02
SQL Server deep dive on Partitioning

03
Oracle deep dive on Partitioning

04
PostgreSQL deep dive on Partitioning

05
What you need to know about partitioning Azure SQL Database

06
QA

01

Partitioning Intro

Working with Large Datasets

Work only with data that you need.

Low maintenance & performance

- Additional effort in design and planning
- More complex data loading

VS

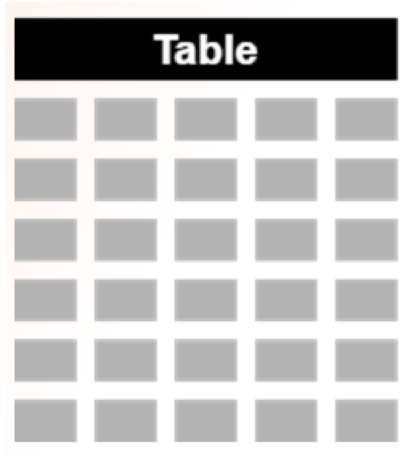
High maintenance & performance?

- Faster data access
- Improved data management flexibility

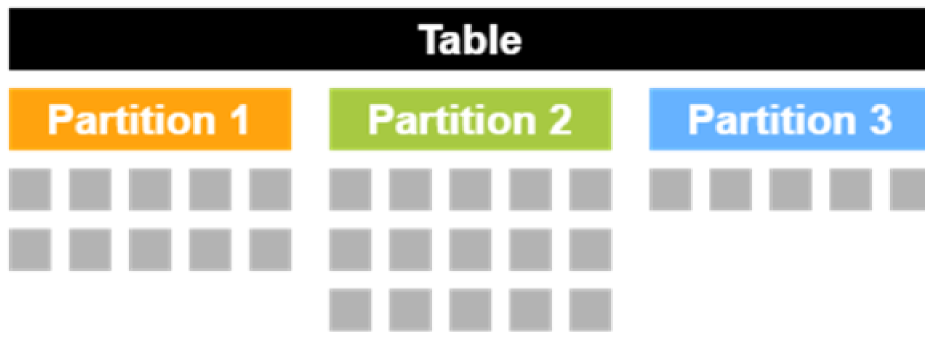
Physically Divided Tables

Patitioning is a physical split of large tables or indexes into smaller manageable units. Each partition can be stored, queried, and maintained separately, though they appear as single logical table to the user.

Non-Partitioned



Partitioned

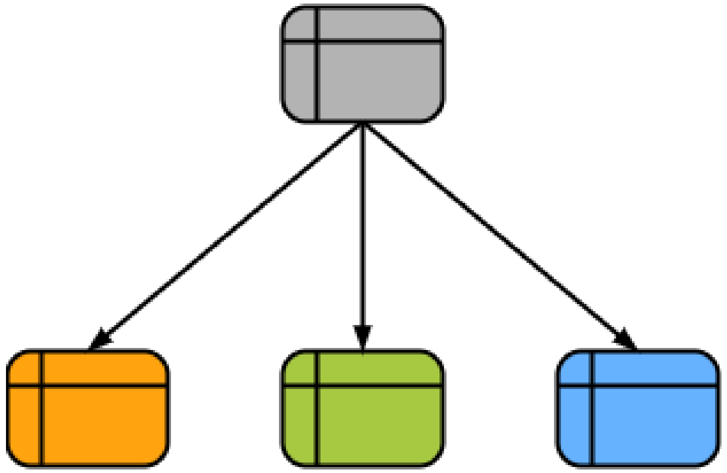


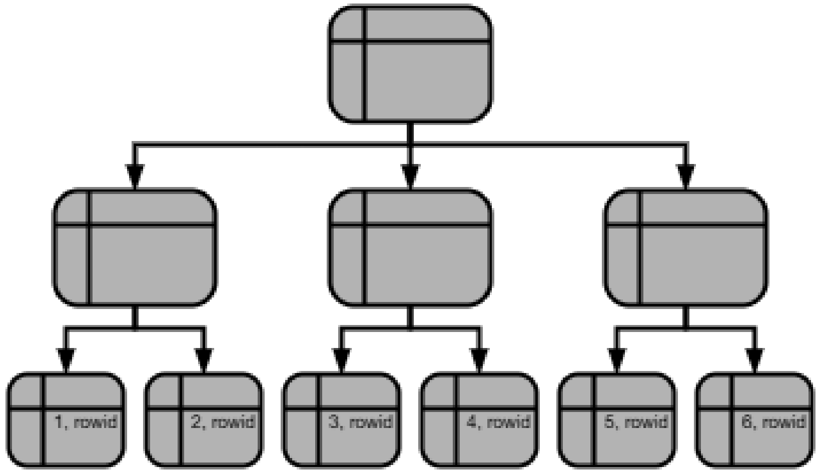
Why Partitions, Not Only Indexes?

They are not alternatives.

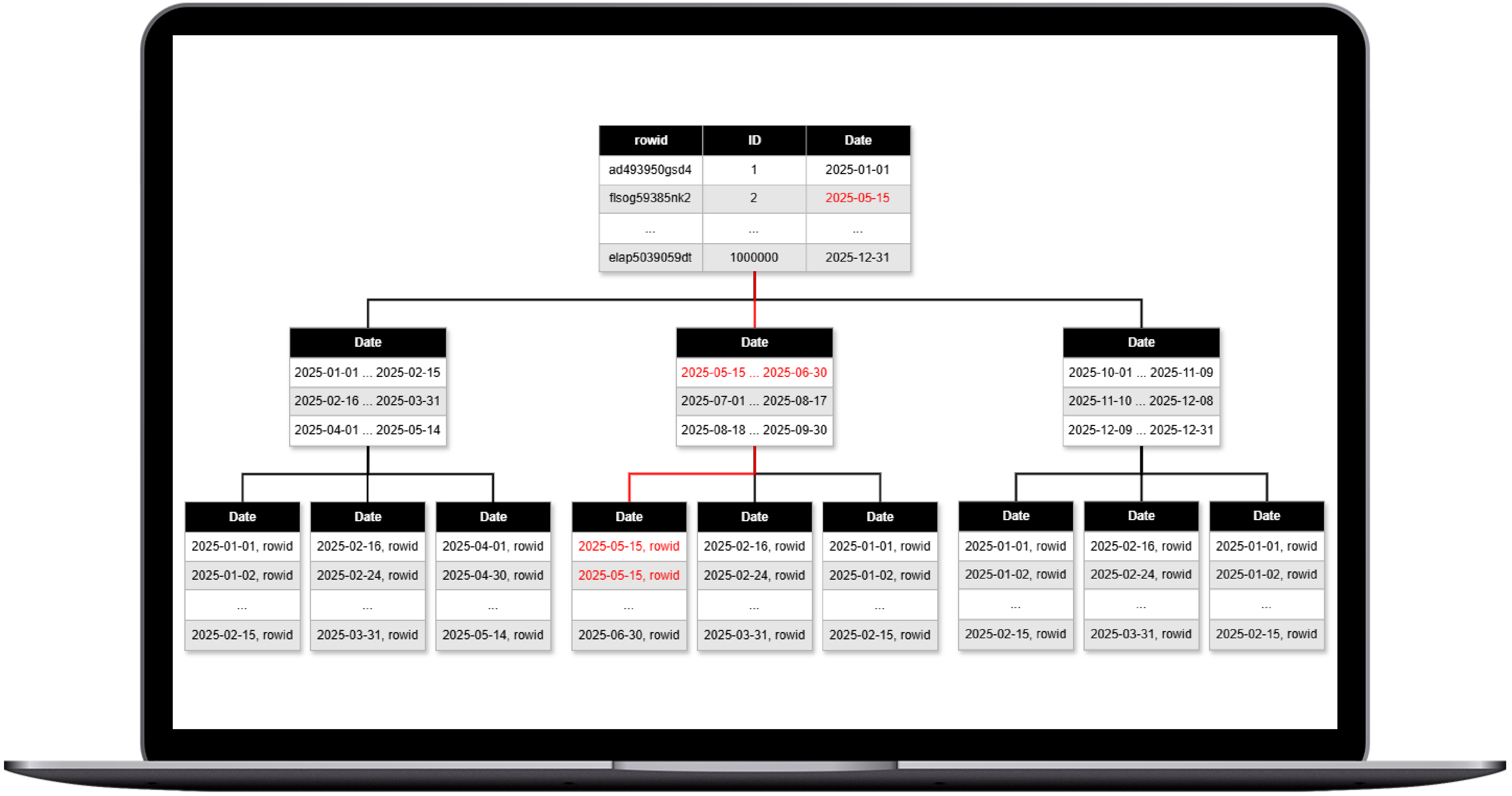
Size of retrieved data portions. Large volumen or specific rows.

Access path. Immediately or rowid by rowid.





Indexing



rowid	ID	Date
ad493950gsd4	1	2025-01-01
flsog59385nk2	2	2025-05-15
...	...	...
elap5039059dt	1000000	2025-12-31

Date
2025-01-01 ... 2025-02-15
2025-02-16 ... 2025-03-31
2025-04-01 ... 2025-05-14

Date
2025-05-15 ... 2025-06-30
2025-07-01 ... 2025-08-17
2025-08-18 ... 2025-09-30

Date
2025-10-01 ... 2025-11-09
2025-11-10 ... 2025-12-08
2025-12-09 ... 2025-12-31

Date
2025-01-01, rowid
2025-01-02, rowid
...
2025-02-15, rowid

Date
2025-02-16, rowid
2025-02-24, rowid
...
2025-03-31, rowid

Date
2025-04-01, rowid
2025-04-30, rowid
...
2025-05-14, rowid

Date
2025-05-15, rowid
2025-05-15, rowid
...
2025-06-30, rowid

Date
2025-02-16, rowid
2025-02-24, rowid
...
2025-03-31, rowid

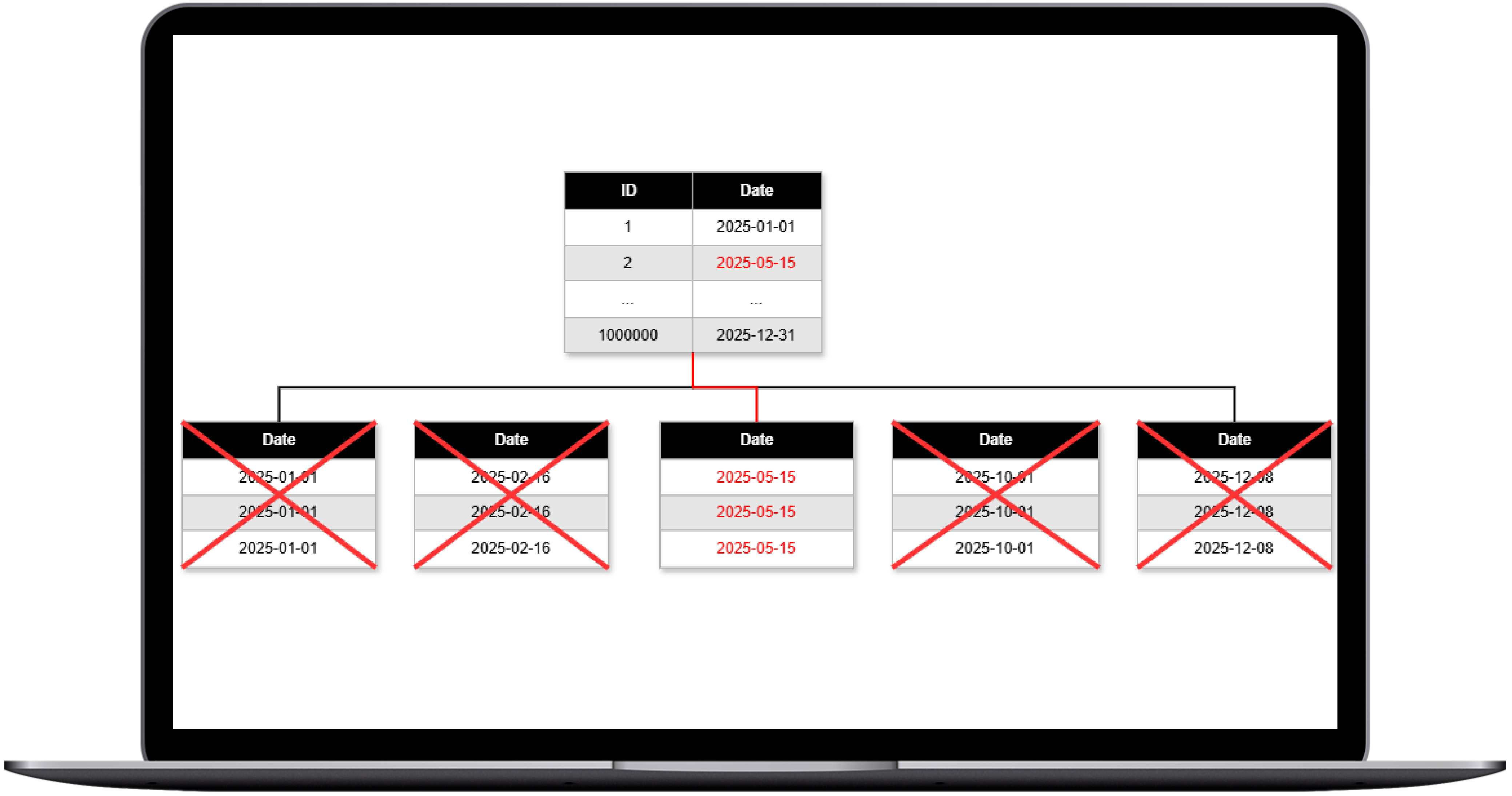
Date
2025-01-01, rowid
2025-01-02, rowid
...
2025-02-15, rowid

Date
2025-01-01, rowid
2025-01-02, rowid
...
2025-02-15, rowid

Date
2025-02-16, rowid
2025-02-24, rowid
...
2025-03-31, rowid

Date
2025-01-01, rowid
2025-01-02, rowid
...
2025-02-15, rowid

Partitioning



ID	Date
1	2025-01-01
2	2025-05-15
...	...
1000000	2025-12-31

Date
~~2025-01-01~~
~~2025-01-01~~
~~2025-01-01~~

Date
~~2025-02-16~~
~~2025-02-16~~
~~2025-02-16~~

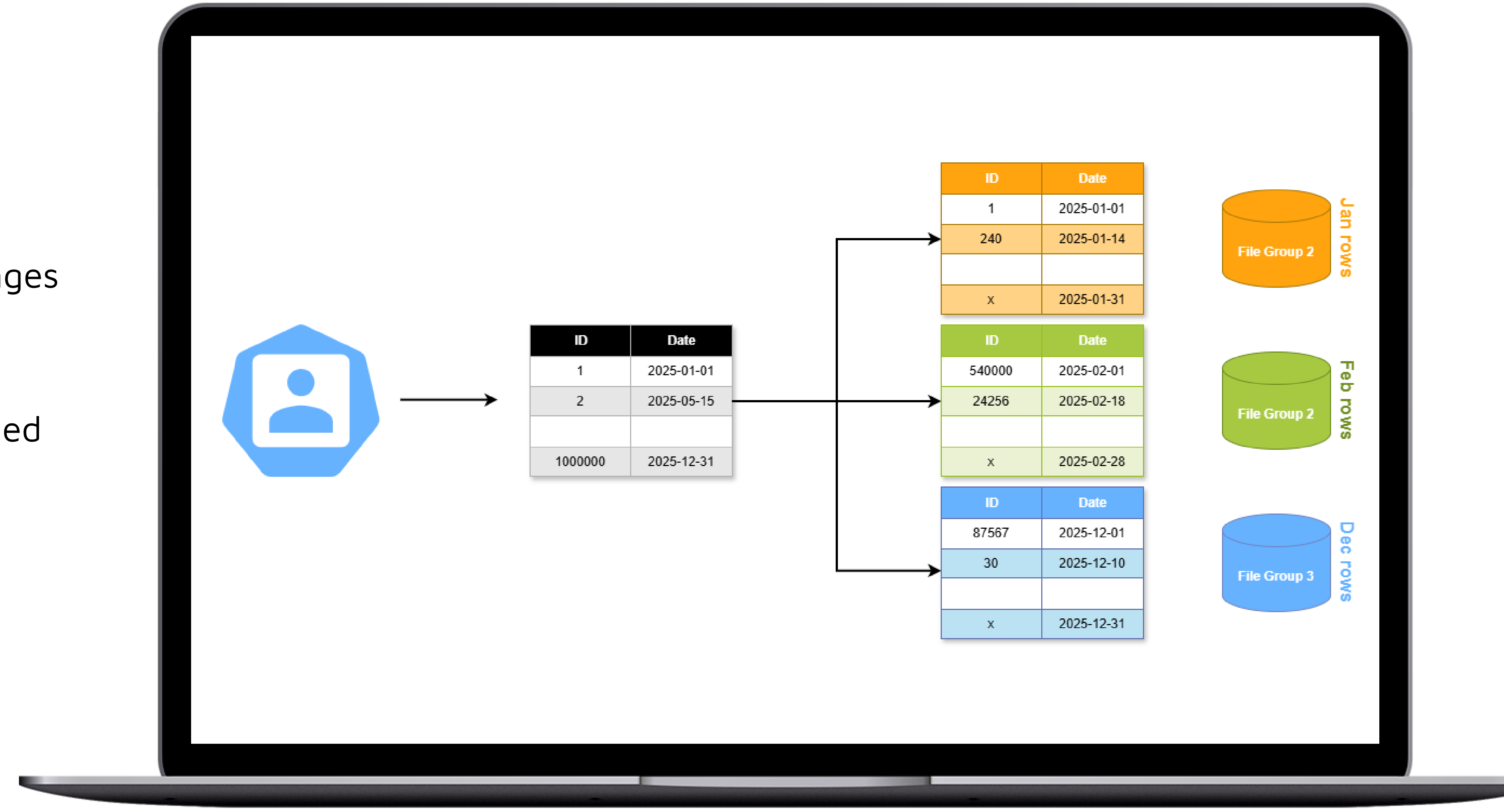
Date
2025-05-15
2025-05-15
2025-05-15

Date
~~2025-10-01~~
~~2025-10-01~~
~~2025-10-01~~

Date
~~2025-12-08~~
~~2025-12-08~~
~~2025-12-08~~

No Impact on User View

Structure remains untouched. No changes required for user or the application. The table is partitioned based on the values of existing columns.



ID	Date
1	2025-01-01
2	2025-05-15
1000000	2025-12-31

ID	Date
1	2025-01-01
240	2025-01-14
x	2025-01-31

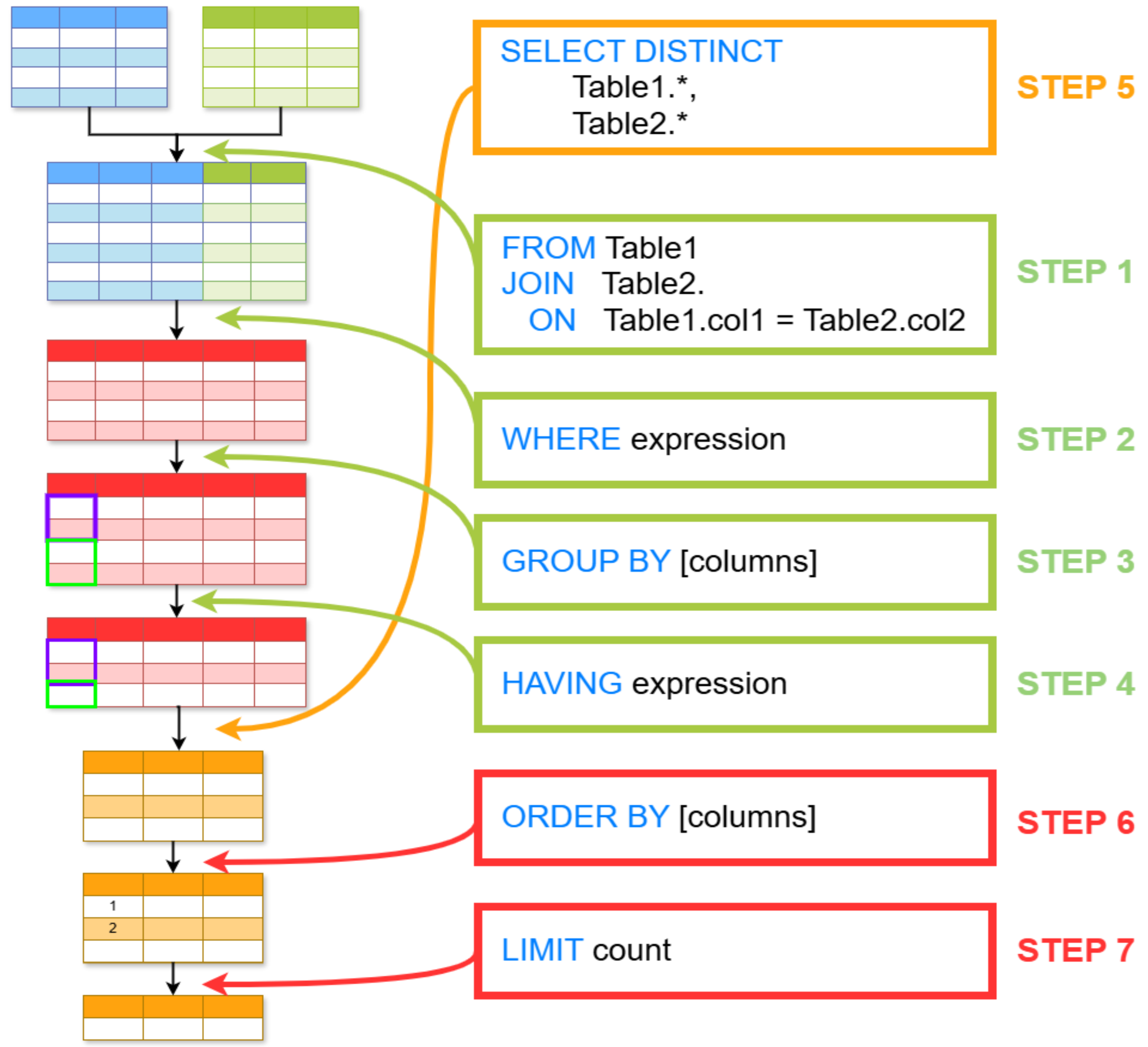
ID	Date
540000	2025-02-01
24256	2025-02-18
x	2025-02-28

ID	Date
87567	2025-12-01
30	2025-12-10
x	2025-12-31

Query Execution Order

Advantage of partitioning through proper query construction.

Query Planner analyzes the query's filters in the WHERE clause and JOIN conditions.



Partition Pruning with WHERE Clause

Filter directly on the partition column using simple comparisons (e.g., =, BETWEEN, <, >) without applying functions or expressions.

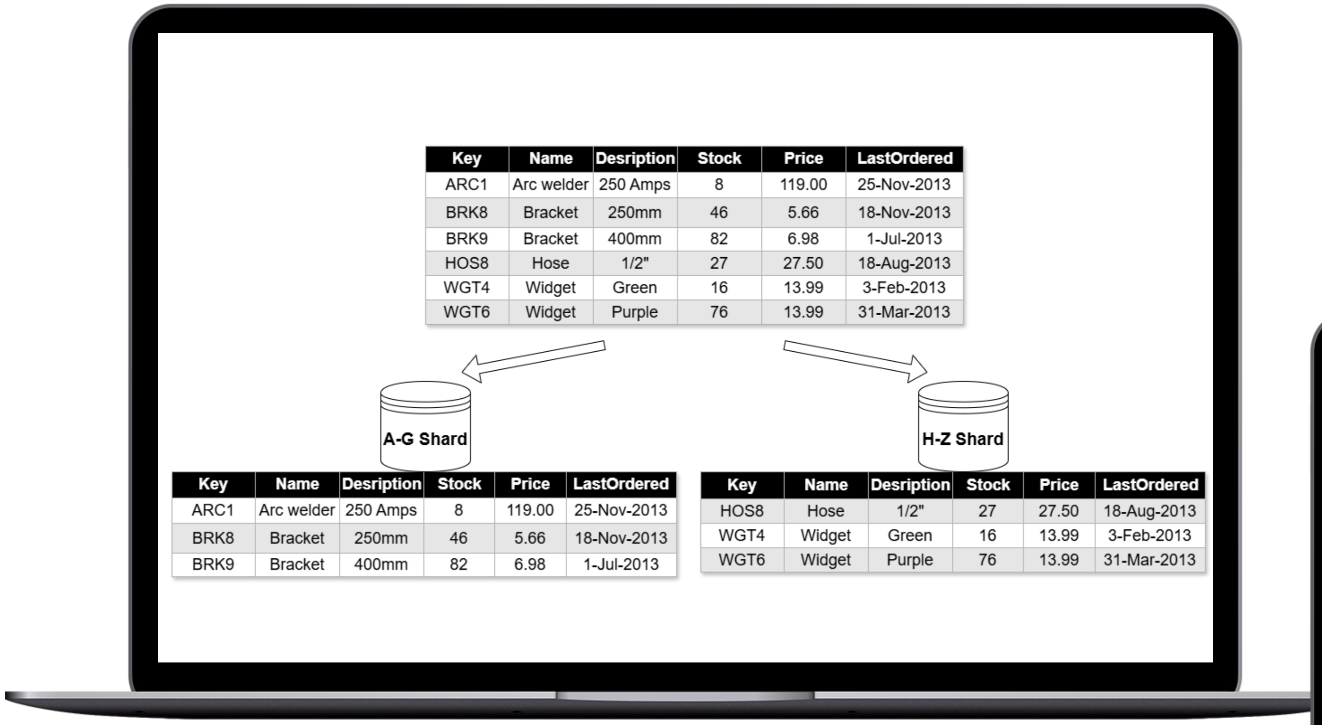
- ✓ WHERE Date = '2024-01-01’
- ✓ WHERE Date BETWEEN '2024-01-01' AND '2024-12-31’
- ✓ WHERE Date >= '2024-01-01' AND order_date <= '2024-12-31’
- ✗ WHERE EXTRACT(YEAR FROM Date) = 2024

Demo

Partitioning Strategies

Concepts of how to physically divide data into separate data stores

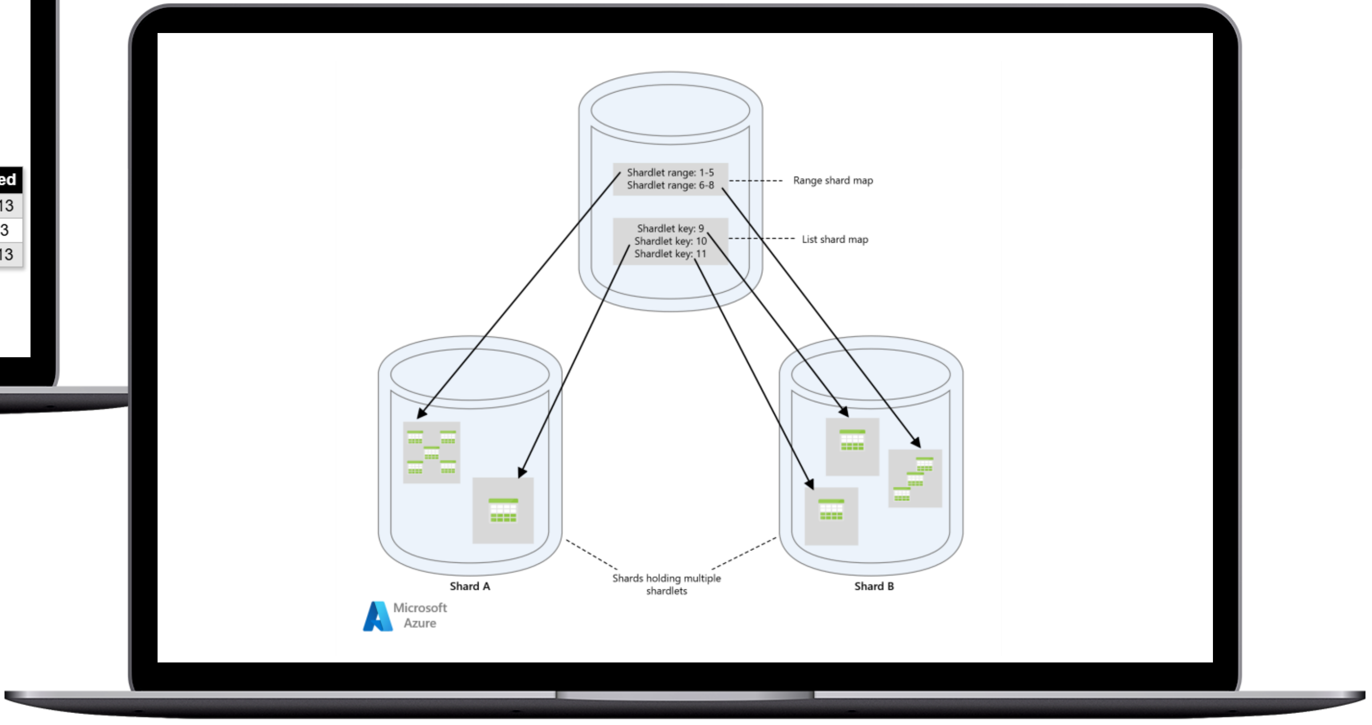
Horizontal partitioning (sharding)



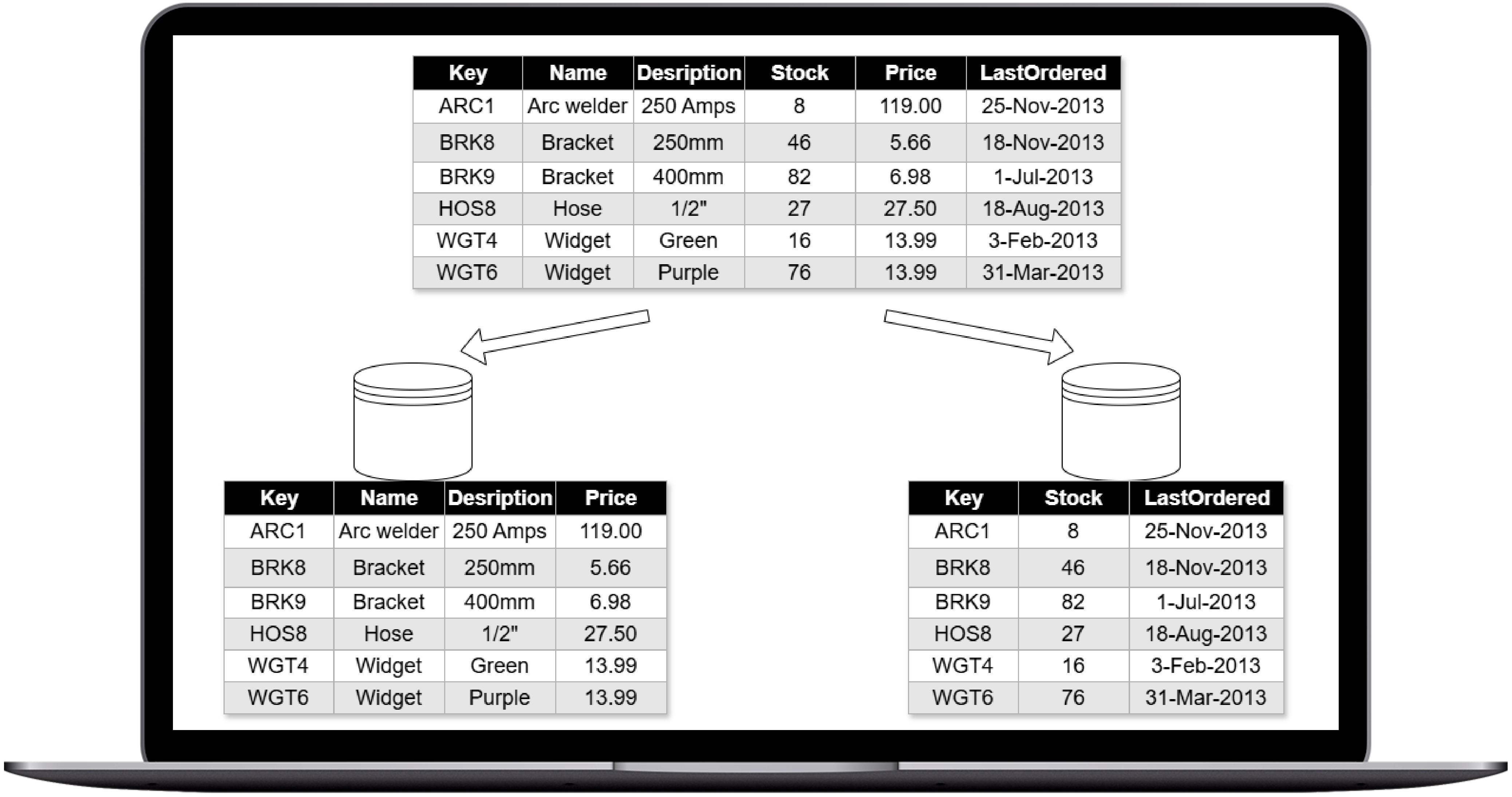
Key	Name	Desription	Stock	Price	LastOrdered
ARC1	Arc welder	250 Amps	8	119.00	25-Nov-2013
BRK8	Bracket	250mm	46	5.66	18-Nov-2013
BRK9	Bracket	400mm	82	6.98	1-Jul-2013
HOS8	Hose	1/2"	27	27.50	18-Aug-2013
WGT4	Widget	Green	16	13.99	3-Feb-2013
WGT6	Widget	Purple	76	13.99	31-Mar-2013

Key	Name	Desription	Stock	Price	LastOrdered
ARC1	Arc welder	250 Amps	8	119.00	25-Nov-2013
BRK8	Bracket	250mm	46	5.66	18-Nov-2013
BRK9	Bracket	400mm	82	6.98	1-Jul-2013

Key	Name	Desription	Stock	Price	LastOrdered
HOS8	Hose	1/2"	27	27.50	18-Aug-2013
WGT4	Widget	Green	16	13.99	3-Feb-2013
WGT6	Widget	Purple	76	13.99	31-Mar-2013



Vertical partitioning

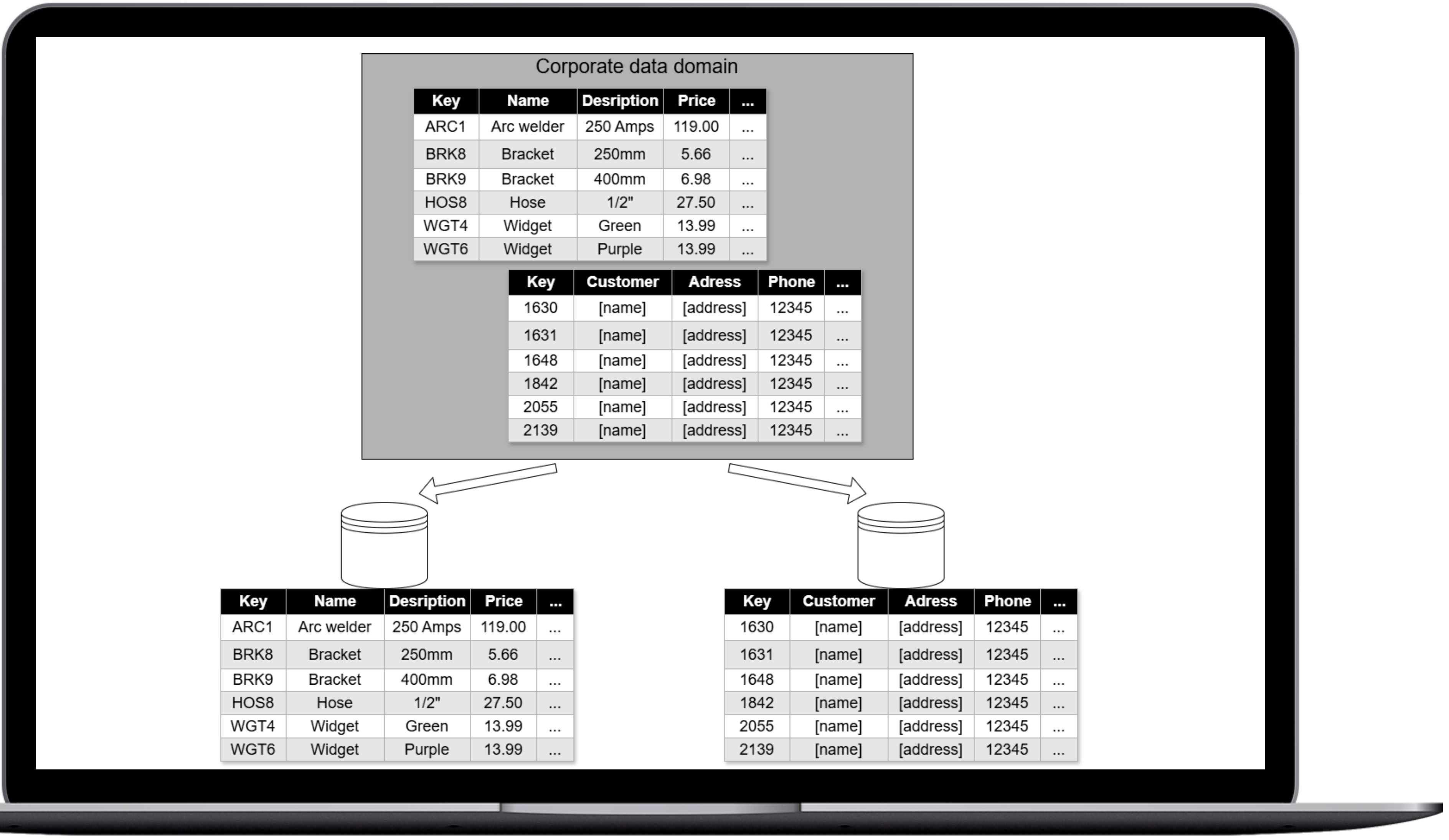


Key	Name	Desription	Stock	Price	LastOrdered
ARC1	Arc welder	250 Amps	8	119.00	25-Nov-2013
BRK8	Bracket	250mm	46	5.66	18-Nov-2013
BRK9	Bracket	400mm	82	6.98	1-Jul-2013
HOS8	Hose	1/2"	27	27.50	18-Aug-2013
WGT4	Widget	Green	16	13.99	3-Feb-2013
WGT6	Widget	Purple	76	13.99	31-Mar-2013

Key	Name	Desription	Price
ARC1	Arc welder	250 Amps	119.00
BRK8	Bracket	250mm	5.66
BRK9	Bracket	400mm	6.98
HOS8	Hose	1/2"	27.50
WGT4	Widget	Green	13.99
WGT6	Widget	Purple	13.99

Key	Stock	LastOrdered
ARC1	8	25-Nov-2013
BRK8	46	18-Nov-2013
BRK9	82	1-Jul-2013
HOS8	27	18-Aug-2013
WGT4	16	3-Feb-2013
WGT6	76	31-Mar-2013

Functional partitioning



Key	Name	Desription	Price	...
ARC1	Arc welder	250 Amps	119.00	...
BRK8	Bracket	250mm	5.66	...
BRK9	Bracket	400mm	6.98	...
HOS8	Hose	1/2"	27.50	...
WGT4	Widget	Green	13.99	...
WGT6	Widget	Purple	13.99	...

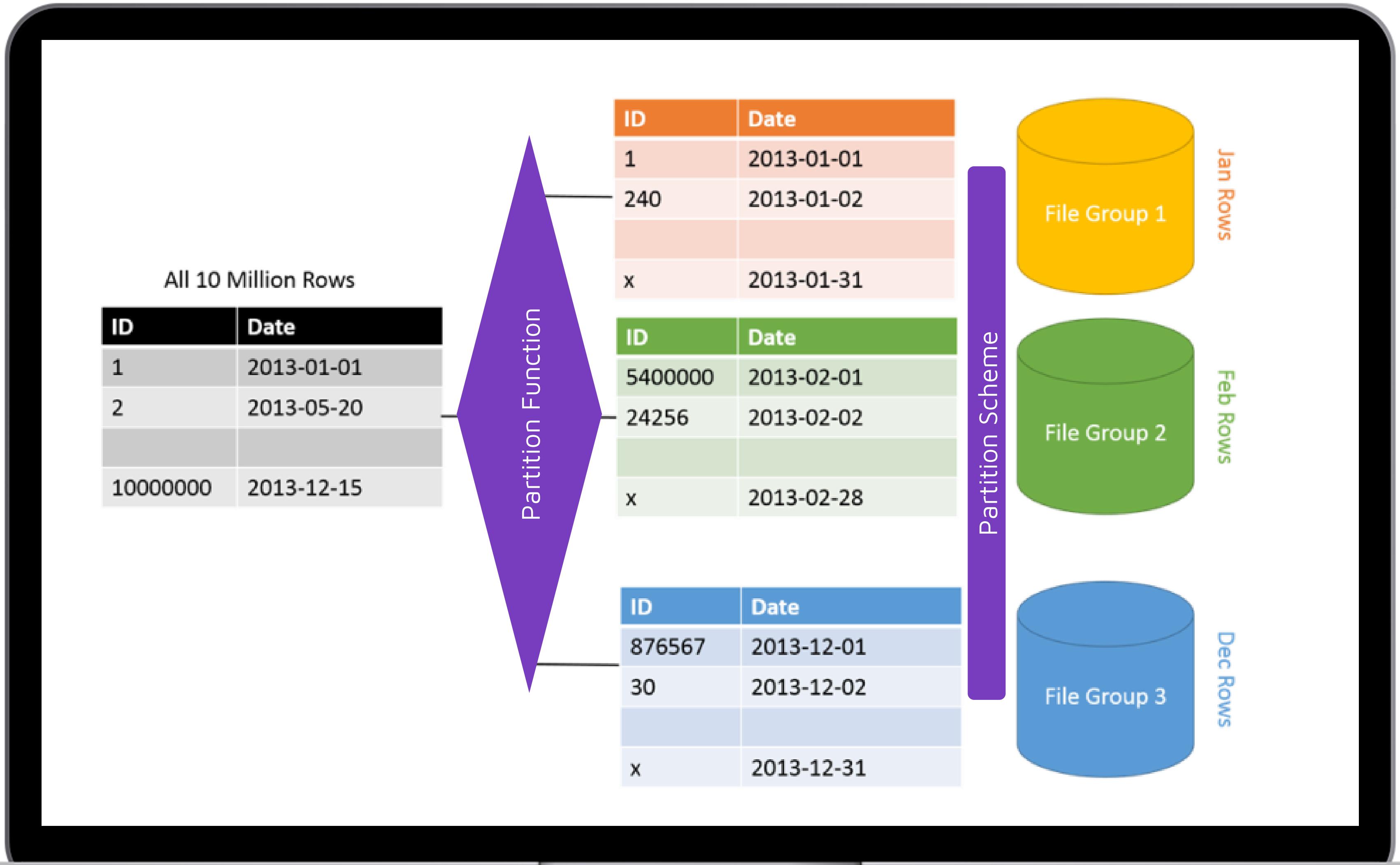
Key	Customer	Adress	Phone	...
1630	[name]	[address]	12345	...
1631	[name]	[address]	12345	...
1648	[name]	[address]	12345	...
1842	[name]	[address]	12345	...
2055	[name]	[address]	12345	...
2139	[name]	[address]	12345	...

Key	Name	Desription	Price	...
ARC1	Arc welder	250 Amps	119.00	...
BRK8	Bracket	250mm	5.66	...
BRK9	Bracket	400mm	6.98	...
HOS8	Hose	1/2"	27.50	...
WGT4	Widget	Green	13.99	...
WGT6	Widget	Purple	13.99	...

Key	Customer	Adress	Phone	...
1630	[name]	[address]	12345	...
1631	[name]	[address]	12345	...
1648	[name]	[address]	12345	...
1842	[name]	[address]	12345	...
2055	[name]	[address]	12345	...
2139	[name]	[address]	12345	...

SQL Server deep dive on Partitioning

Partitioned tables and indexes


All 10 Million Rows

ID	Date
1	2013-01-01
2	2013-05-20
10000000	2013-12-15

ID	Date
1	2013-01-01
240	2013-01-02
x	2013-01-31

ID	Date
5400000	2013-02-01
24256	2013-02-02
x	2013-02-28

ID	Date
876567	2013-12-01
30	2013-12-02
x	2013-12-31

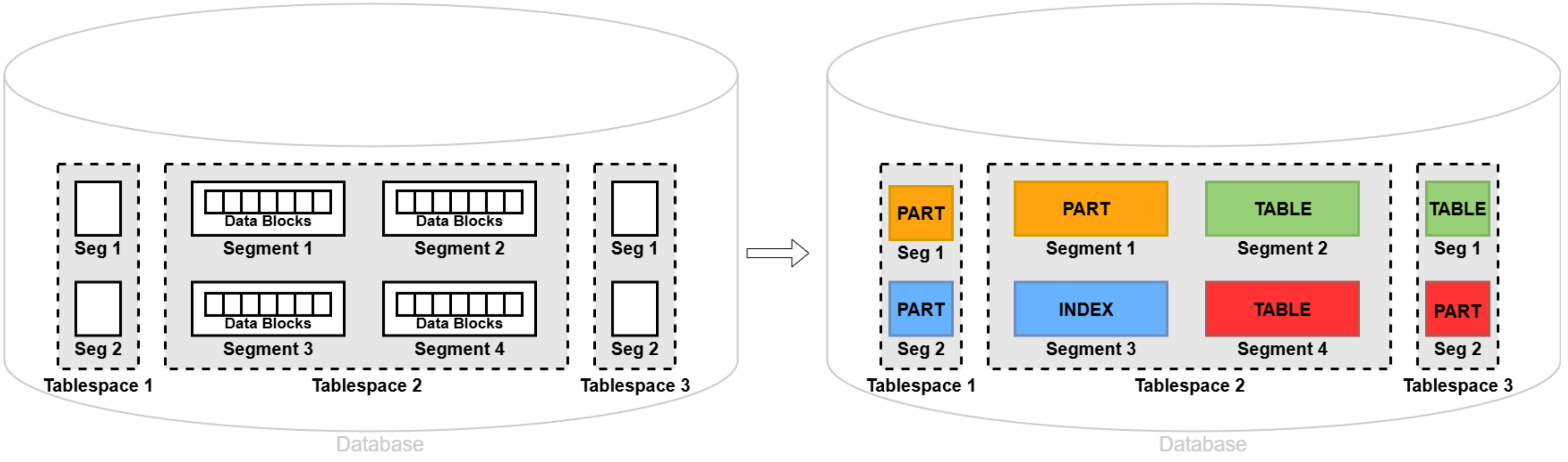
Demo

Oracle deep dive on Partitioning

Oracle: Storage Architecture

Logical data structures are separate from physical data structures.

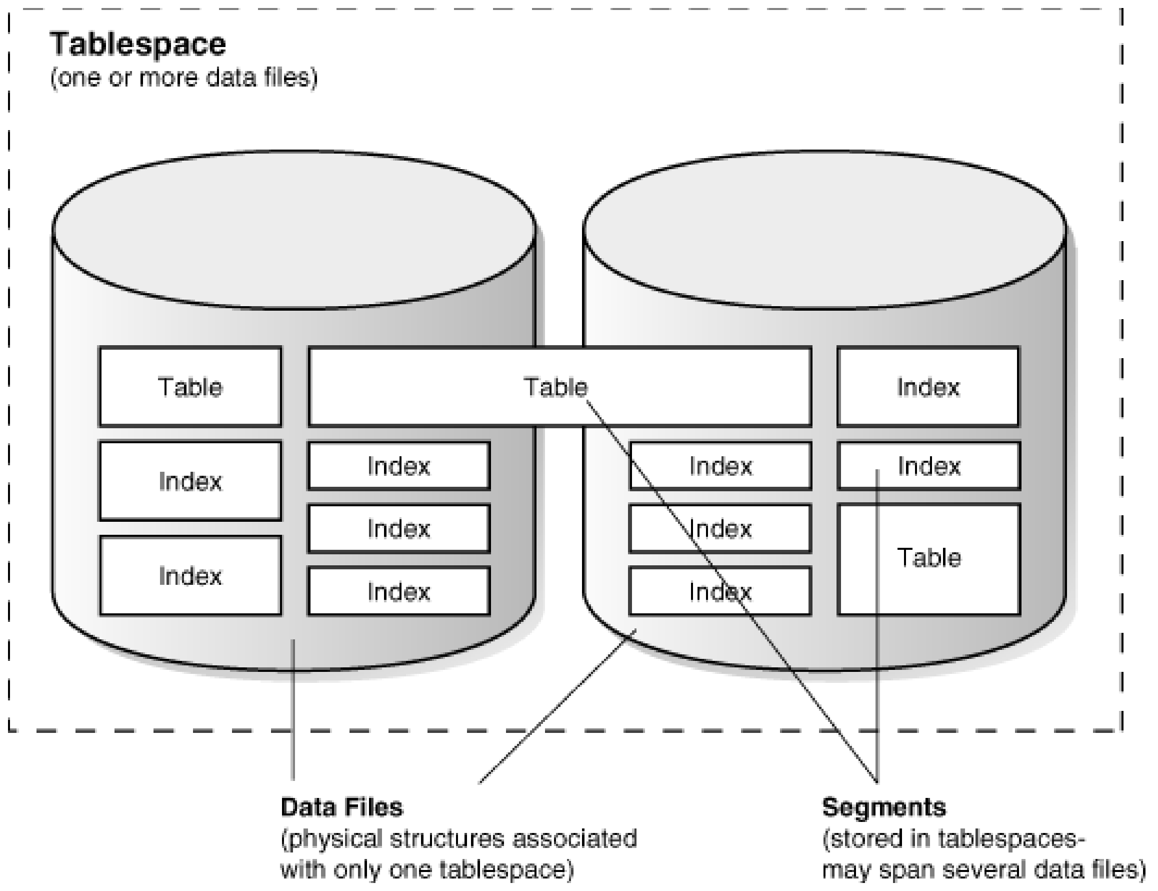
Tables, views, indexes, partitions are logical objects. Each object is stored as a separate segment and can have its own storage settings.



Oracle: Storage Architecture

Segments (partitions) may be located in different tablespaces.

Tablespaces are separate data files or disk groups. Each partition can reside in a separate storage with its own configuration.

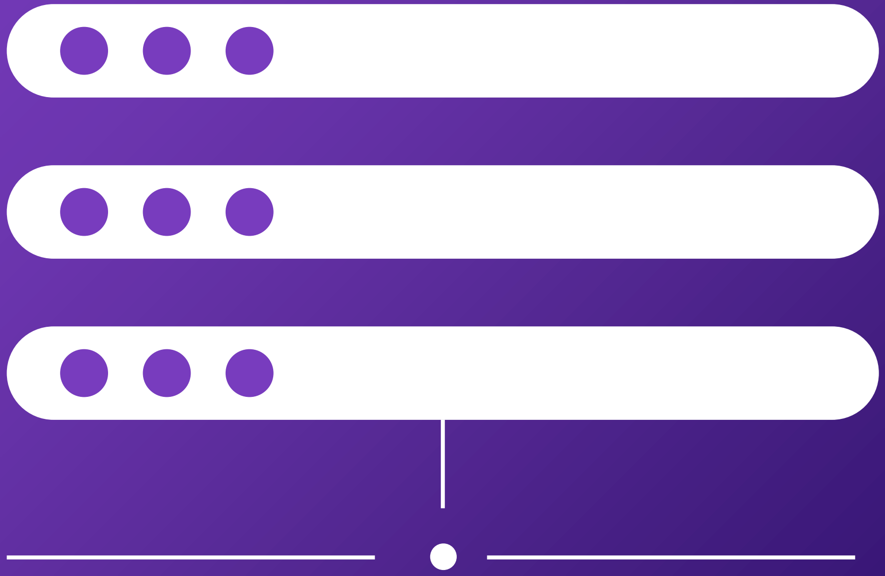


Oracle: Memory

Buffer Cache holds frequently accessed data blocks from any partition. Hot partitions' blocks stay in memory more, effectively prioritizing active data.

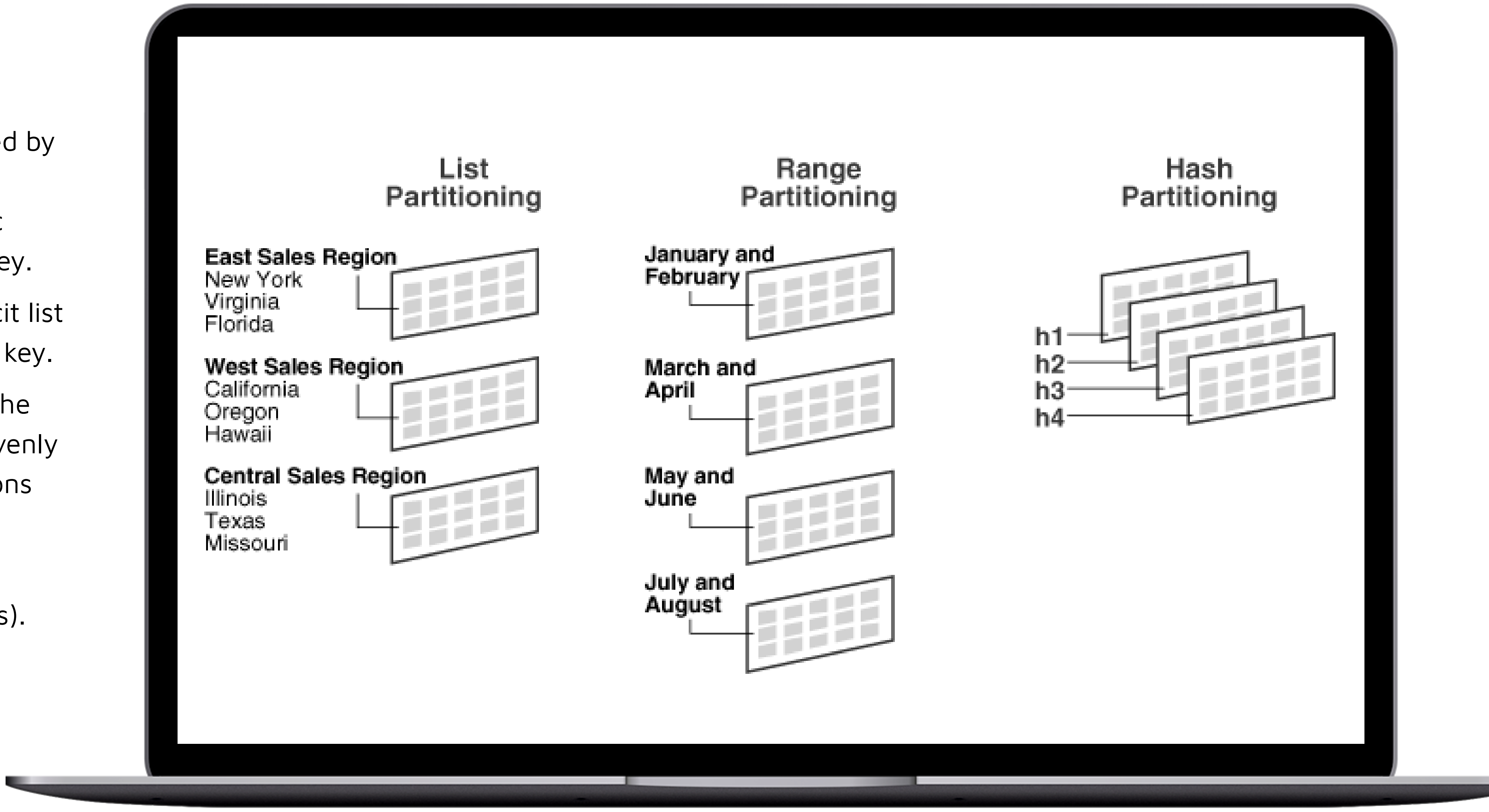
Data dictionary stores partition definitions to determine which partition to access. The mapping from a partition key value to the segment is handled by the optimizer.

Parallel scans, assigning different partitions to different query workers, multiple CPU/IO for partitioned table access. Each partition can be scanned independently in parallel, then results combined.



Partitioning Methods in Oracle

- Range: Data is partitioned by ranges of values. Each partition covers a specific interval of the partition key.
- List: Defined by an explicit list of discrete values for the key.
- Hash: Hash function on the partition key. Rows are evenly distributed across partitions based on hash values.
- Composite: Multi-level partitioning (subpartitions). Combining methods.



Demo

Range

Interval

List

Hash

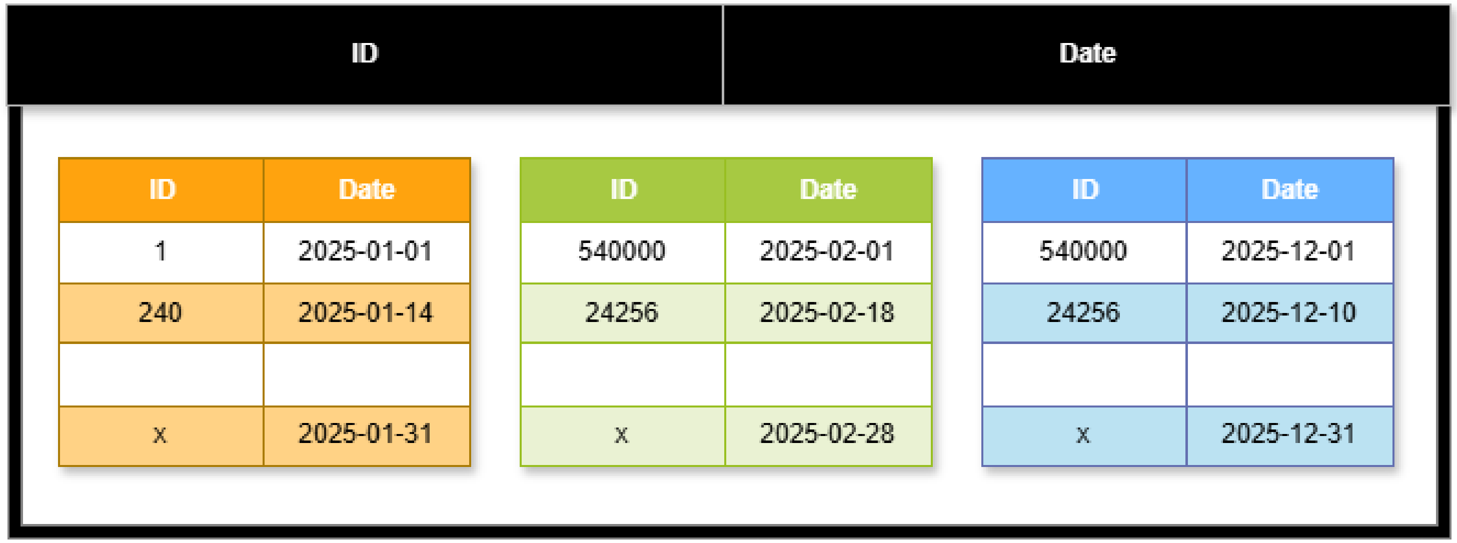
Composite

PostgreSQL deep dive on Partitioning

PostgreSQL: Storage Architecture

Partitioned table is a parent virtual table with no data of its own. All data is stored in its child tables - the partitions.

Each partition is an independent table in the storage engine with its own heap file and index files.



ID	Date

ID	Date
1	2025-01-01
240	2025-01-14
x	2025-01-31

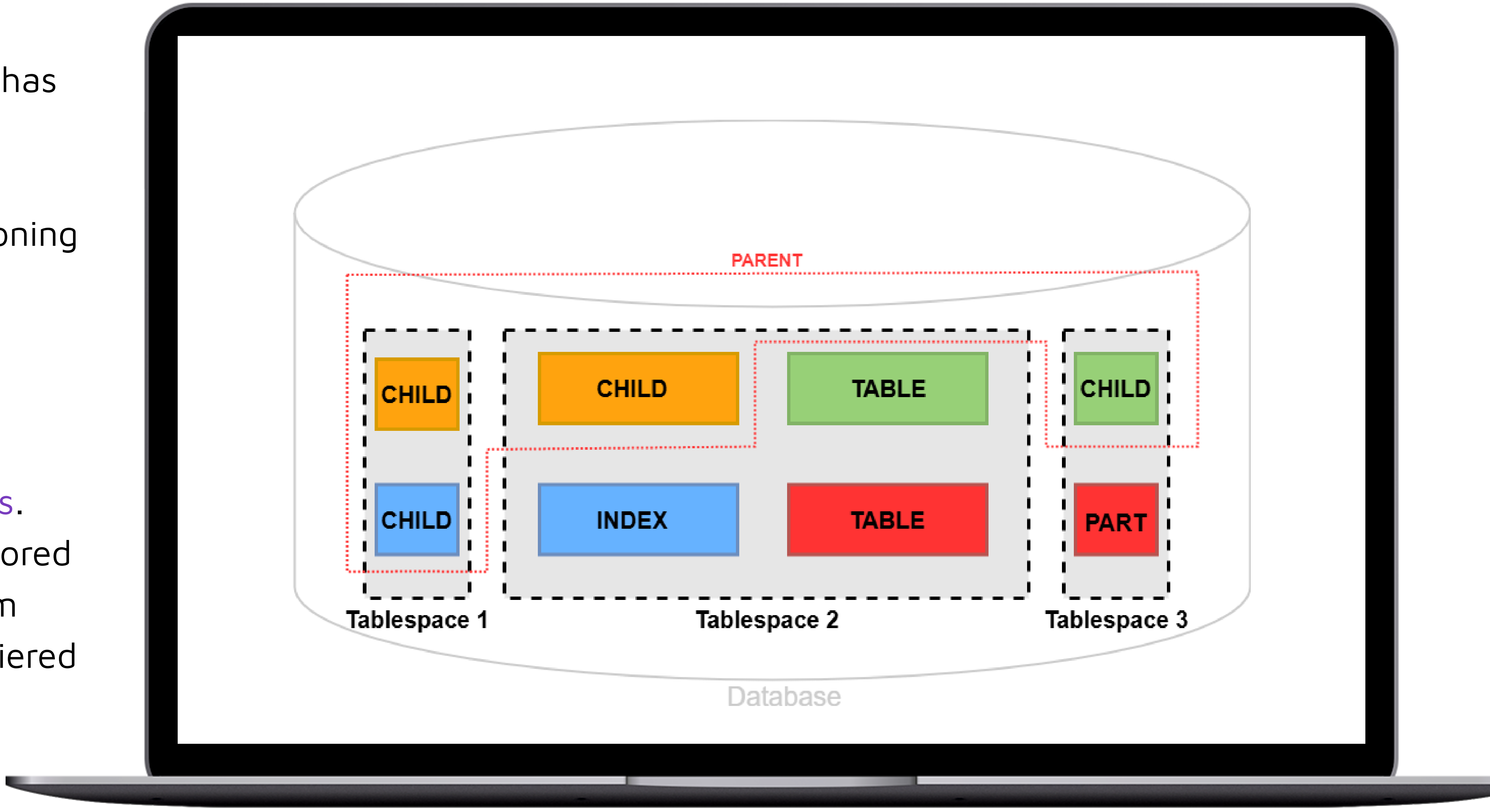
ID	Date
540000	2025-02-01
24256	2025-02-18
x	2025-02-28

ID	Date
540000	2025-12-01
24256	2025-12-10
x	2025-12-31

PostgreSQL: Storage Architecture

Each partition (child) has its own files on disk. Parent table stores metadata like partitioning scheme
and list of partitions.

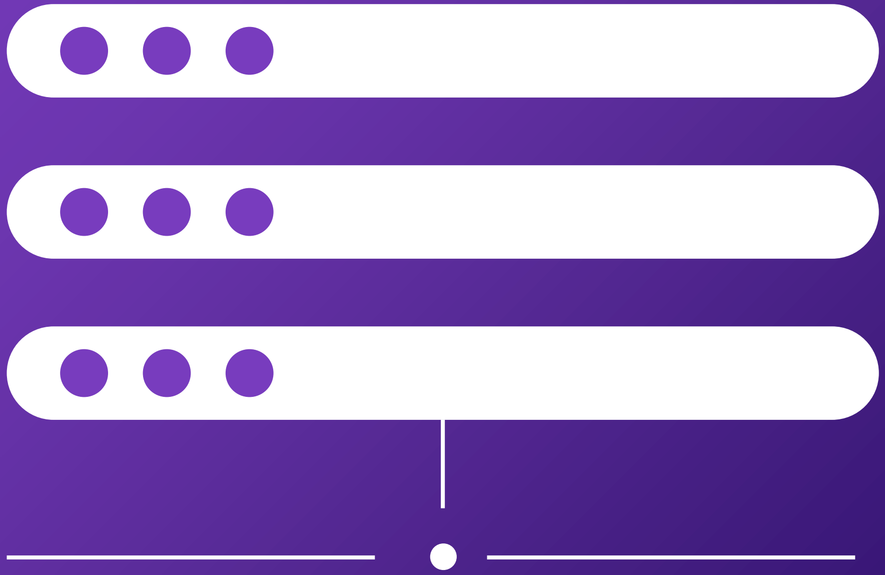
Partitions placed in different tablespaces. Different partitions stored in different file system locations mimicking tiered storage.



PostgreSQL: Memory

Shared buffer pool for caching data. Pages from partition tables are treated the same as any table’s pages. Hot partitions’ pages will occupy more cache due to frequent access, but there’s no dedicated memory per partition.

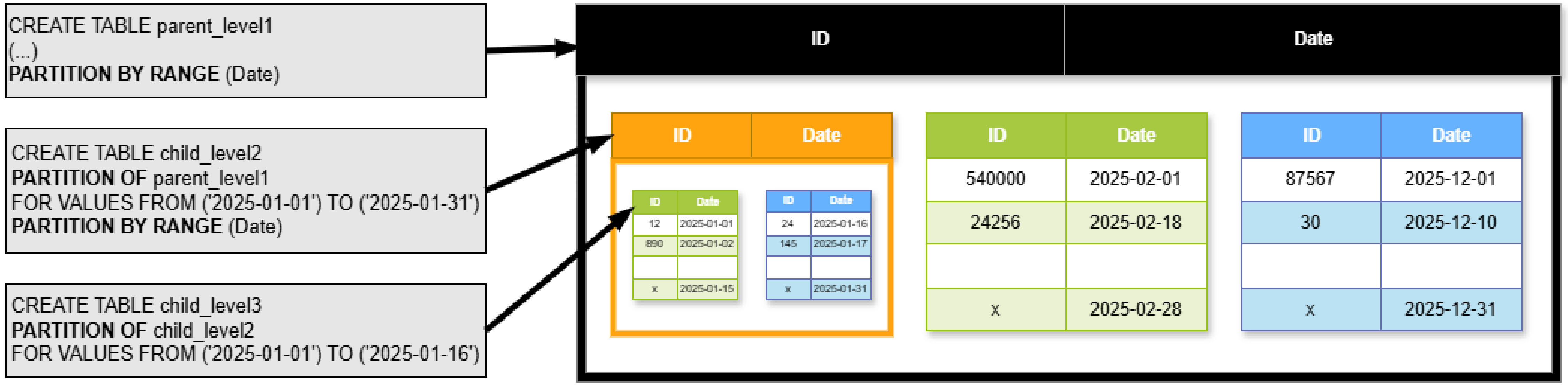
Partition definitions (ranges or lists of values) are stored in the catalog; pg_class for the child tables, pg_inherits links them to the parent, pg_partitioned_table stores partition key info. Executor routes the inserted row to the appropriate child table. For queries, the planner uses the partition constraints to determine which partitions can be pruned.



Partitioning in PostgreSQL

- CREATE TABLE as syntax with one-column partition specified. Individual names.
- Partitioning existing table is not possible. Attaching/detaching existing table as a partition is possible.

- Multi-level partitioning. 3 levels for best performance.
- No indexes possible at TABLE_PARENT, only at TABLE_CHILD (leaf).
- INSERT/SELECT/DELETE possible on TABLE_PARENT and TABLE_CHILD



```
CREATE TABLE parent_level1
(...)
PARTITION BY RANGE (Date)
```

```
CREATE TABLE child_level2
PARTITION OF parent_level1
FOR VALUES FROM ('2025-01-01') TO ('2025-01-31')
PARTITION BY RANGE (Date)
```

```
CREATE TABLE child_level3
PARTITION OF child_level2
FOR VALUES FROM ('2025-01-01') TO ('2025-01-16')
```

ID	Date

ID	Date

ID	Date
12	2025-01-01
890	2025-01-02
x	2025-01-15

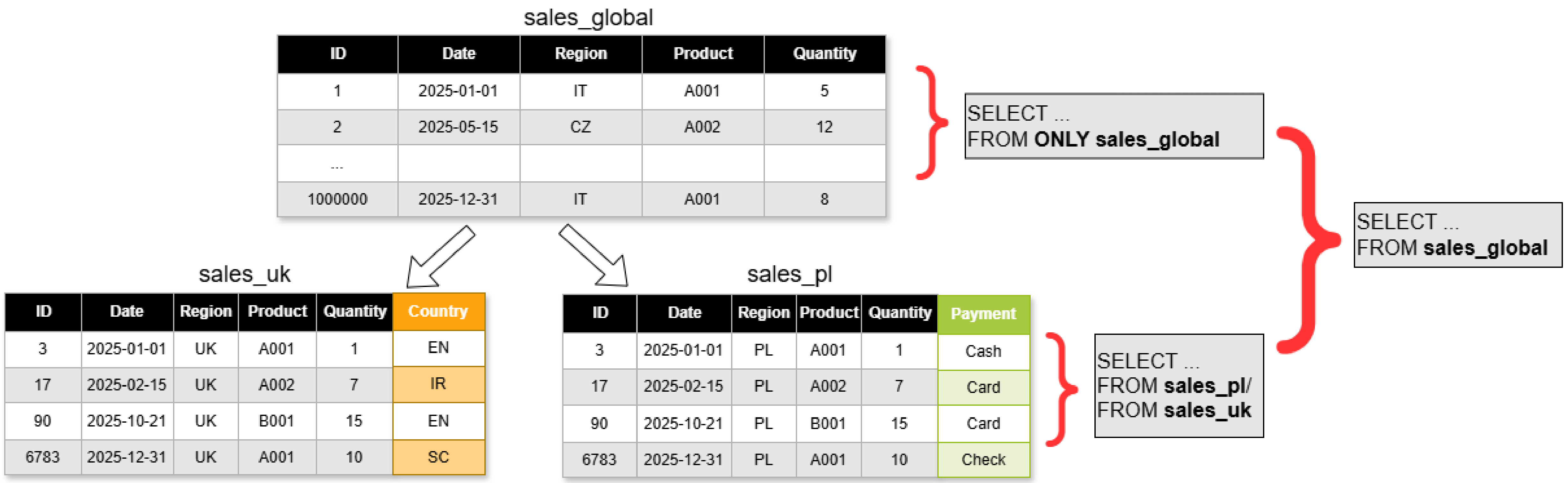
ID	Date
24	2025-01-16
145	2025-01-17
x	2025-01-31

ID	Date
540000	2025-02-01
24256	2025-02-18
x	2025-02-28

ID	Date
87567	2025-12-01
30	2025-12-10
x	2025-12-31

Partitioning in PostgreSQL

Inheritance: Legacy Method where partitioning is done via table inheritance and check constraints manually. This involved parent table + child tables and triggers or application logic for routing inserts. Declarative partitioning automates this.



sales_global

ID	Date	Region	Product	Quantity
1	2025-01-01	IT	A001	5
2	2025-05-15	CZ	A002	12
...				
1000000	2025-12-31	IT	A001	8

sales_uk

ID	Date	Region	Product	Quantity	Country
3	2025-01-01	UK	A001	1	EN
17	2025-02-15	UK	A002	7	IR
90	2025-10-21	UK	B001	15	EN
6783	2025-12-31	UK	A001	10	SC

sales_pl

ID	Date	Region	Product	Quantity	Payment
3	2025-01-01	PL	A001	1	Cash
17	2025-02-15	PL	A002	7	Card
90	2025-10-21	PL	B001	15	Card
6783	2025-12-31	PL	A001	10	Check

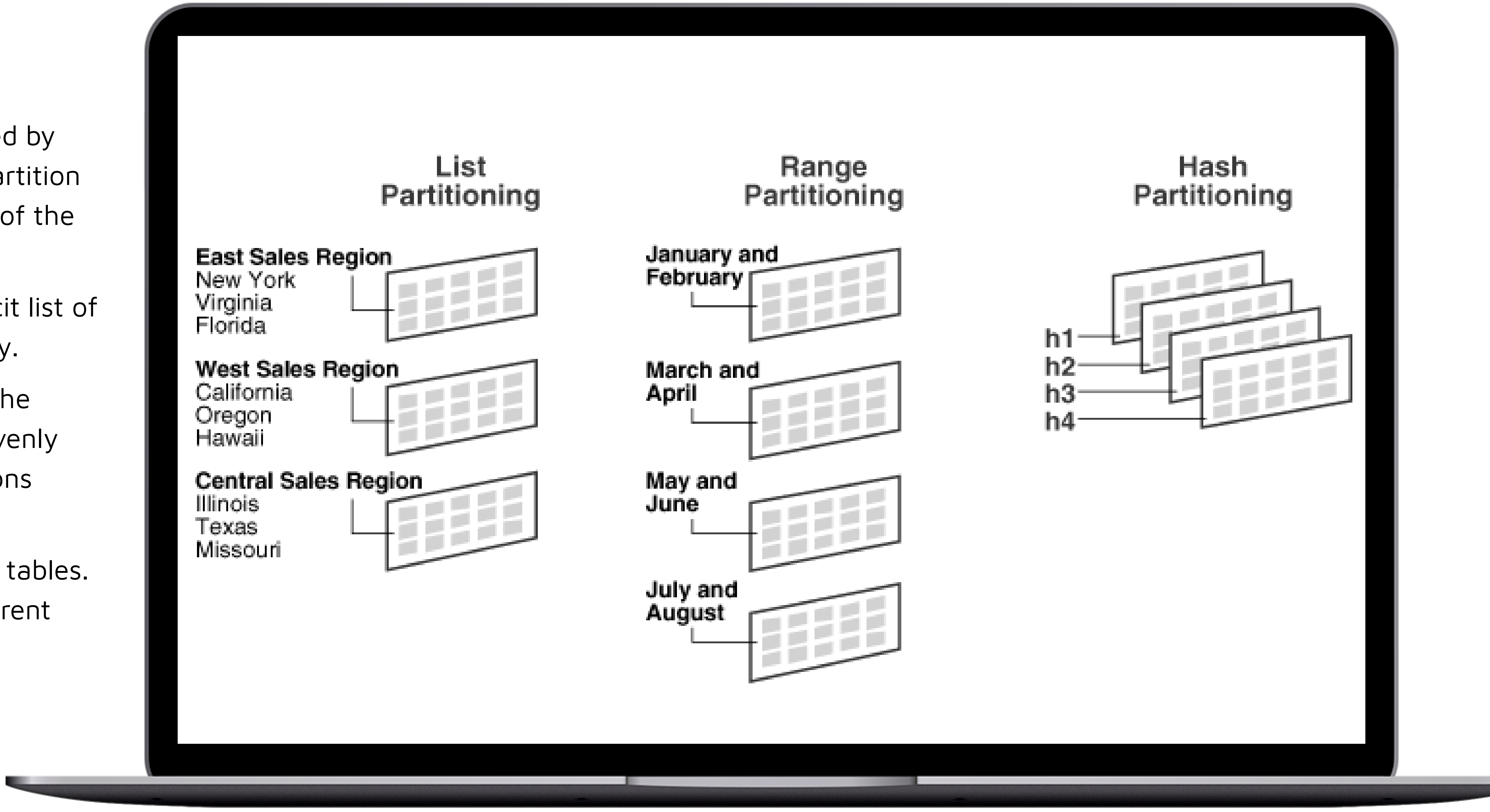
```
SELECT ...
FROM ONLY sales_global
```

```
SELECT ...
FROM sales_global
```

```
SELECT ...
FROM sales_pl/
FROM sales_uk
```

Partitioning in PostgreSQL

- Range: Data is partitioned by ranges of values. Each partition covers a specific interval of the partition key.
- List: Defined by an explicit list of discrete values for the key.
- Hash: Hash function on the partition key. Rows are evenly distributed across partitions based on hash values.
- Multi-level: Nested child tables. Each table can have different method.



Demo

Range

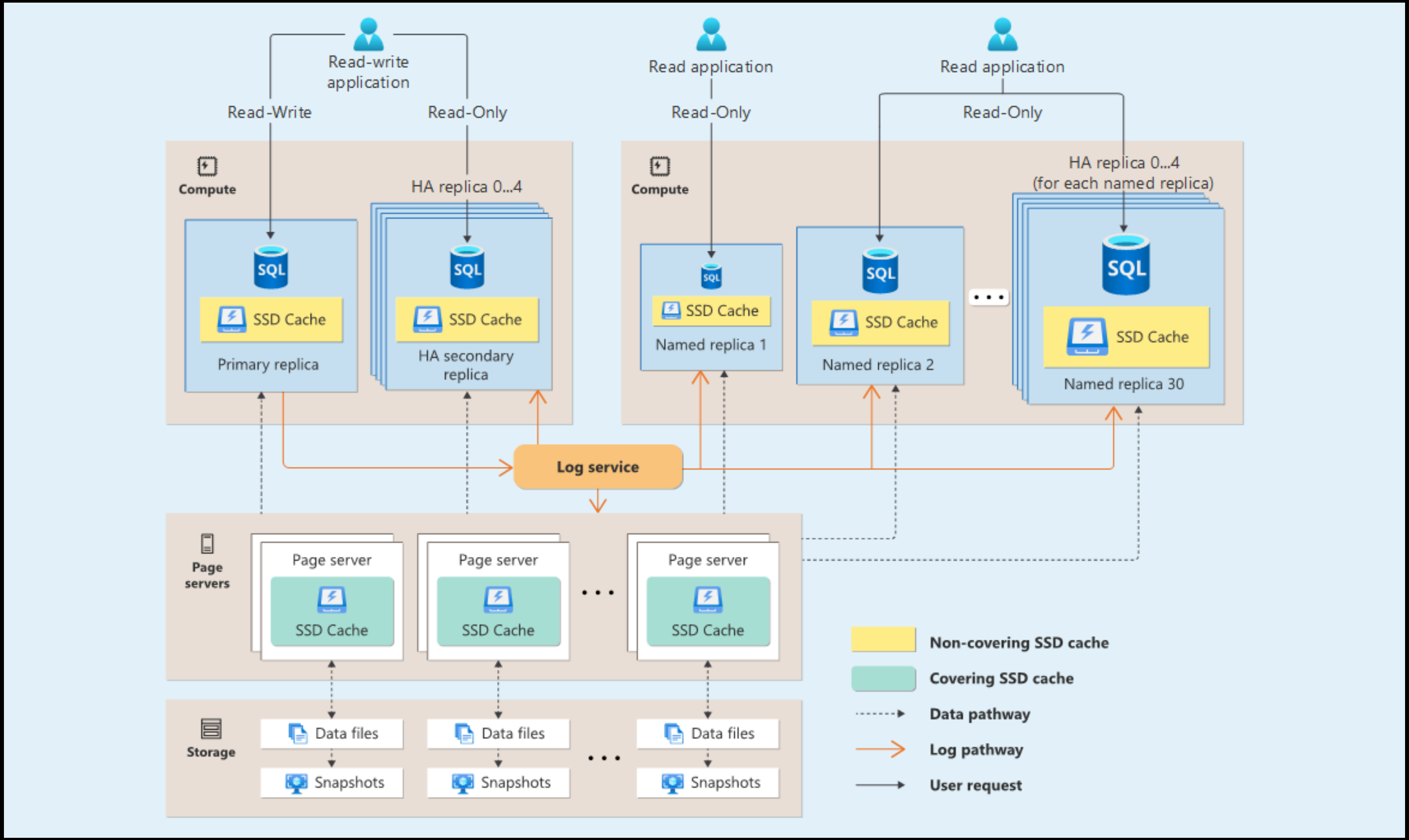
List

Hash

Multi-level

What you need to know about partitioning Azure SQL Database

On Azure SQL DB Hyperscale example



Read-write application
Read-Write
Read-Only
Read application
Read-Only
Read application
Read-Only
Compute
HA replica 0...4
SQL
SSD Cache
Primary replica
SQL
SSD Cache
HA secondary replica
Compute
HA replica 0...4 (for each named replica)
SQL
SSD Cache
Named replica 1
SQL
SSD Cache
Named replica 2
SQL
SSD Cache
Named replica 30
Log service
Page servers
Page server
SSD Cache
Page server
SSD Cache
Page server
SSD Cache
Storage
Data files
Snapshots
Data files
Snapshots
Data files
Snapshots
Non-covering SSD cache
Covering SSD cache
Data pathway
Log pathway
User request

Partitioning Concepts

Table partitioning and table maintenance

- Easier index and statistics maintenance on large tables

Table partitioning and compression

- Implement a table partitioning strategy first, and then implement your compression strategy
- Possibility of using compressions for different sections of the partitioned table

Table partitioning and columnstore indexes

- Each partition has to contain a minimum of 1 million rows for optimal compression and performance of clustered columnstore tables

Table partitioning and tempdb

- When implementing table partitioning, some data may have to be temporarily stored in tempdb. tempdb space is limited, so filling the tempdb completely, will be resulting in a failure.
- Implement the data copy from source table to destination partitioned table in batches

Partitioning Concepts

Table partitioning and Index alignment

- Index is implemented on top of the same partition scheme as its base table.
- Partition switch operations qualify as being metadata-only operations

Table partitioning and statistics

- Possibility to use incremental statistics: computed at partition level

Table partitioning and Replication

- There are requirements and limitations in terms of partition functions and partitions in terms of replication
- Manual administration effort might be needed at subscribers

Table partitioning and Change data capture (CDC)

- Doing operations in a partitioned table can generate inconsistencies if the partitioned table is enabled for change data capture
- Those operations are PARTITION SWITCH, MERGE and SPLIT.

Partitioning Scenarios

Partitioning in place

- The movement of data among partitions happens in-place inside the same table, making this method less space consuming as there is no duplication of tables
- Data movement inside the existing table limits the options to stop and roll back the partitioning operation if desired.
- During the data movement, locks are held on the pages of the table that could block other activities, making it difficult to do online.

Partitioning into a new table

- Original table is available during process
- Possibility to start/stop/rollback if needed
- Can be used for testing to find best partitioning configuration
- Higher complexity doing the data synchronization in comparison with in-place partitioning

Partitioning heap tables

- For a heap, a clustered index over a partition scheme has to be created to actually shuffle the rows among the partitions, and subsequently dropped so the table becomes a heap table again.

Sliding window table partitioning scenario

- Data management scenario for a table in which the same number of partitions is kept over time, and as new time periods start, new partitions are created and brought into the table while the older partitions are evicted

Summary

		Oracle	SQL Server	PostgreSQL
Types	Range	Yes defining upper boundary only, right edge partition open (ptional)	Yes boundaries for inner partitions defined, edge partition open (required)	Yes edge partitions open (optional)
	Interval	Yes	No	No
	List	List with subpartitions	No	List without subpartitions
	Hash	Yes	No	Yes
	Composite	Yes	No	Yes
Subpartitioning		Yes (2 levels)	No (1 level)	Yes (multilevel)
Indexes		Yes, fully suported	Yes, fully suported	Yes, inheritance. On individual partitions, not the parent
Partitioning existing tables		No	No	No
Partition prunning		built-in	built-in	setup enabled/disabled
Separate storage		Yes (on prem)	Yes (on prem)	Yes (on prem)
Parallelism		Yes	Yes	Yes

lingaro