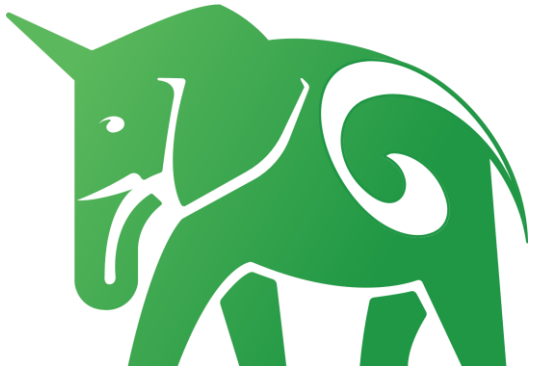
Smooth Sailing: How We Tackled PostgreSQL Migration Challenges from CentOS to Ubuntu

Sena Güngör Tavukçuoğlu

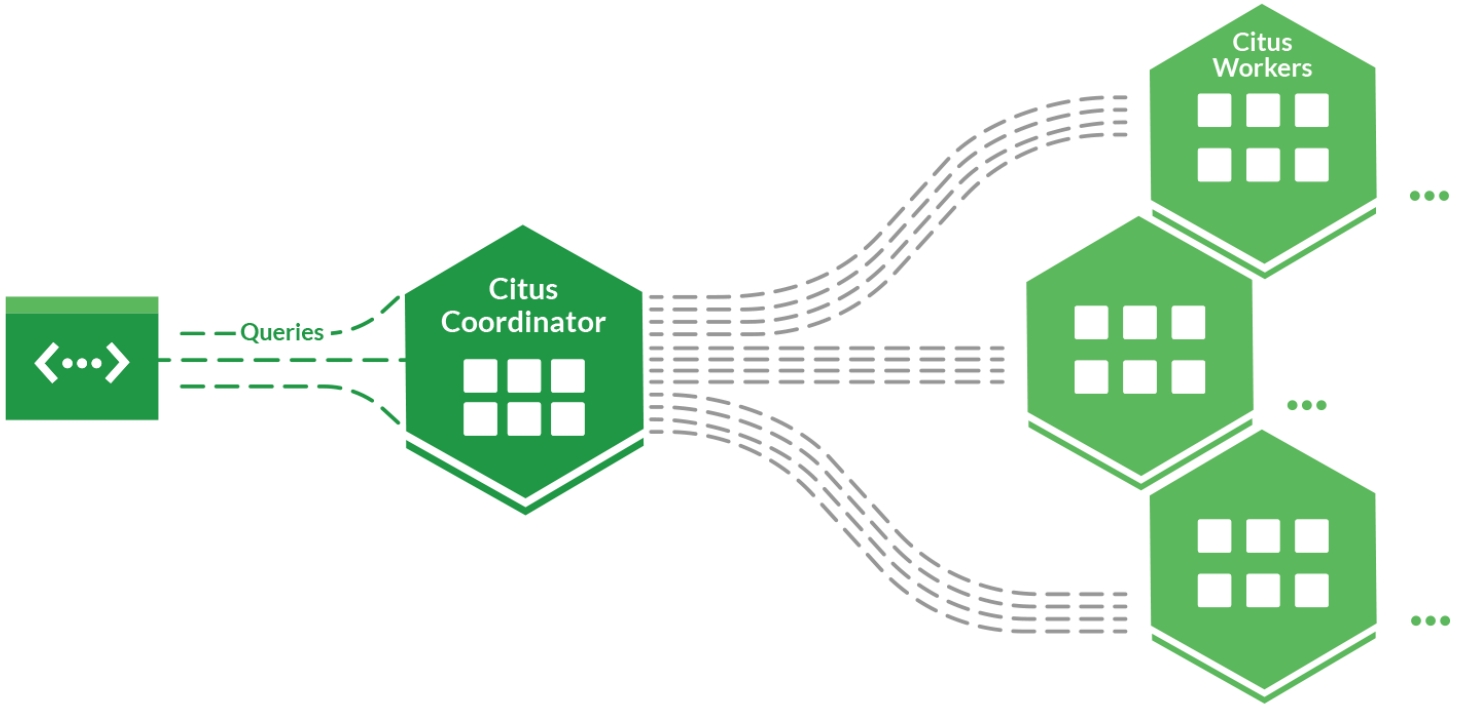
PGConfEU – 2024 Athens

Azure Cosmos DB for PostgreSQL

- fully managed database
- extended with Citus
- horizontally scalable

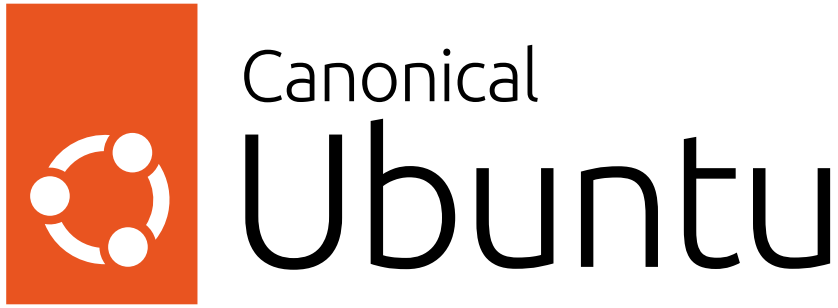


More on Citus



Why Migrate?

- CentOS 7 reached its End of Life.



Why Ubuntu?

Long term support

Package support

Performance improvement

Migration Plan

Create Standby
with Ubuntu

Failover

Error!

```
database "%s" has a collation version mismatch
```

collation?

glibc – GNU C library

- Memory allocation
- Input/output operations
- **String processing**

Sorting and Collation Changes

How collation changes after the migration affected sorting behavior?

Sorting with Special Characters

Before (CentOS 7 – old glibc)

After (Ubuntu 22.0 – new glibc)

“user-2”, “user_1”, “user.3” → “user.3”, “user-2”, “user_1”

Handling Accents

Before (CentOS 7 – old glibc)

After (Ubuntu 22.0 – new glibc)

“resume”, “résumé”, “result” → “resume”, “result”, “résumé”

Sorting Numbers in Strings

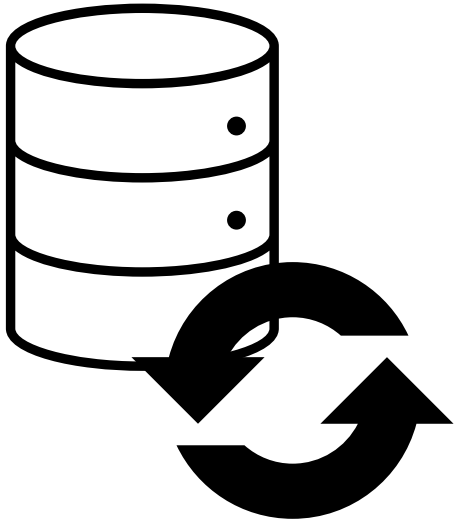
Before (CentOS 7 – old glibc)

After (Ubuntu 22.0 – new glibc)

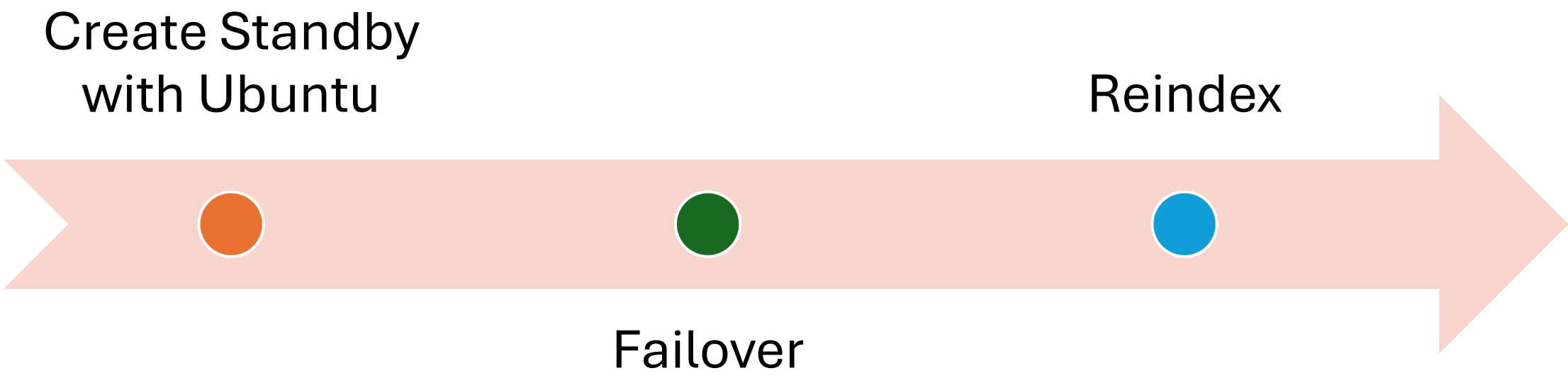
“file1”, “file10”, “file2”  “file1”, “file2”, “file10”

Reindex!

- Rebuild all indexes
- Ensure data integrity
- Query accuracy



Migration Plan



Which indexes are affected?

- Indexes on text columns
- like TEXT, VARCHAR, CHAR, and CITEXT

```
SELECT DISTINCT indrelid::regclass::text, indexrelid::regclass::text,
collname, pg_get_indexdef(indexrelid)

FROM (SELECT indexrelid, indrelid, indcollation[i] coll FROM pg_index,
generate_subscripts(indcollation, 1) g(i)) s

  JOIN pg_collation c ON coll=c.oid

WHERE collprovider IN ('d', 'c') AND collname NOT IN ('C', 'POSIX');
```

Preparing for Reindex

Ẽźźºì 8ĂẢvụ ĂẢẻĝřì Ăẻửĩ ĂÆśźŭĂẴĝĝì źźĂẢvgụ ửì ĝŭźĂ

y.ɔ2y.ỷì È ẻỷừ Ăủv Ăểřì Ăỷì źĝºì ɀ

Why pg_prewarm?

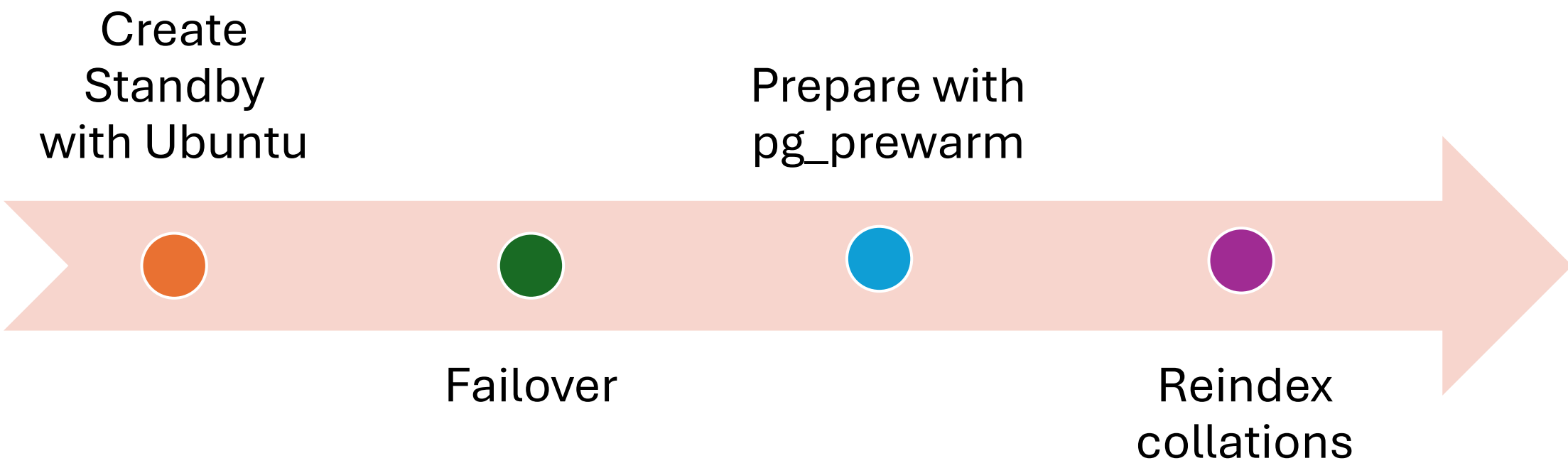
- Loads critical data into memory before users start querying.
- Avoids performance dips caused by fetching data from disk.
- Preloads reindexed tables and indexes, ensuring smooth post-migration performance.

pg_prewarm

```
CREATE EXTENSION pg_prewarm;

SELECT pg_prewarm(‘table_name');
```

Migration Plan



Quick Recap: Where We Are So Far

- why migrate?
- initial plan
- need to reindex
- pg_prewarm

Time to migrate!

- How to make sure?

One shot, one opportunity

Look

If you had

One shot

One opportunity

To seize everything you ever wanted

In one moment,

Would you capture it

or just let it slip?

Validation was critical

- Our compass for the sail!

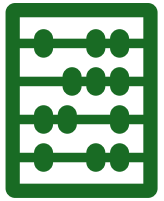
How will we validate?

Fork the cluster

Using Point in Time Recovery

Prepare Standbys

Change any required configuration



Reindex

Measure the downtime

Goals with Validation

ESTIMATE DOWNTIME

TEST THE MIGRATION PROCESS

IDENTIFY EDGE CASES

Goals with Validation

Performance testing

Logging and error reporting

Results with Validation

- We had ~900 clusters to migrate
- We measured the downtime with validators
- 400 of them resulted under 5 minutes
- For the rest, we had to schedule a maintenance window

Challenges

Out of Memory

Increasing Compute Power: Adding vCores

- More vCores allowed for parallel processing
- Challenges: Some nodes required more vCores than expected, and upgrading them was necessary.
- Not all clusters responded

Out-of-Memory Errors

	TOTAL INDEX SIZE	BIGGEST INDEX SIZE	ORIGINAL VCORES	TIME TO REINDEX	INCREASED VCORES	TIME TO REINDEX
Cluster A	49 GB	3,5 GB	4	N/A Out of Memory!	32	4 mins
Cluster B	189 GB	8,7 GB	2	N/A Out of Memory!	64	16 mins
Cluster C	234 GB	22 GB	4	N/A Out of Memory!	64	21

Increasing vm.overcommit_ratio

- What is vm.overcommit_ratio?
- Why Increase It?
- How It Works?

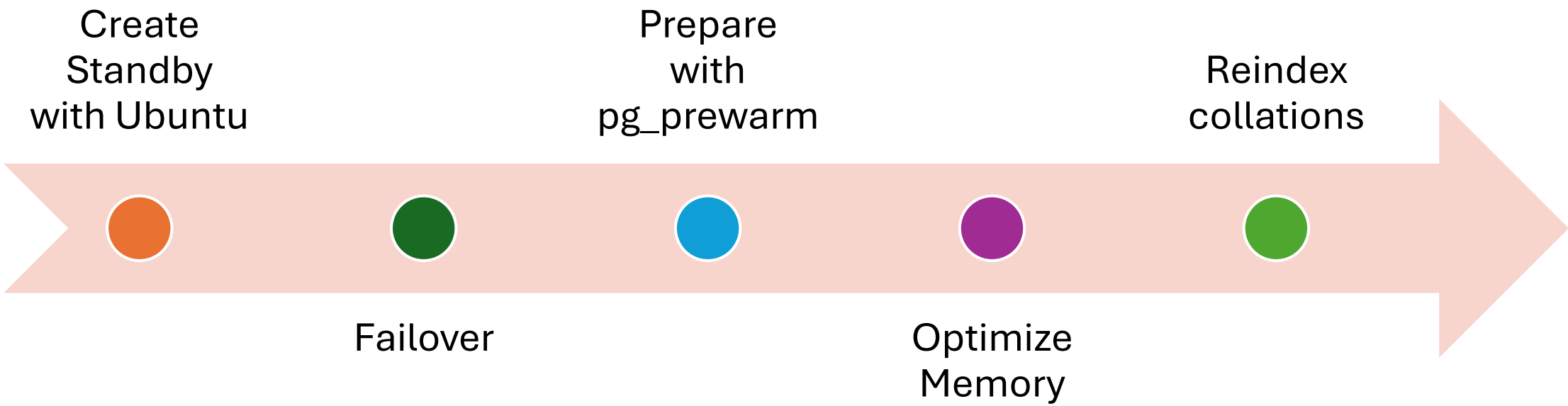
Increasing vm.overcommit_ratio

- Impact of it:
  - Temporarily avoids OOM errors during resource-heavy processes (like reindexing).
  - Can lead to higher memory usage but improves the system's ability to handle short-term spikes in memory demand.
- Trade-Off:
  - Short-term performance gain during intensive tasks.
  - May cause performance issues if swap space is overused, but generally effective for managing spikes

Reducing maintenance_work_mem

- What is maintenance_work_mem?
  - Specifies the amount of memory allocated for maintenance operations like reindexing and vacuuming.
- Allowed the system to operate within available memory limits without hitting OOM issues.
- A balance between **speed and stability.**

Migration Plan

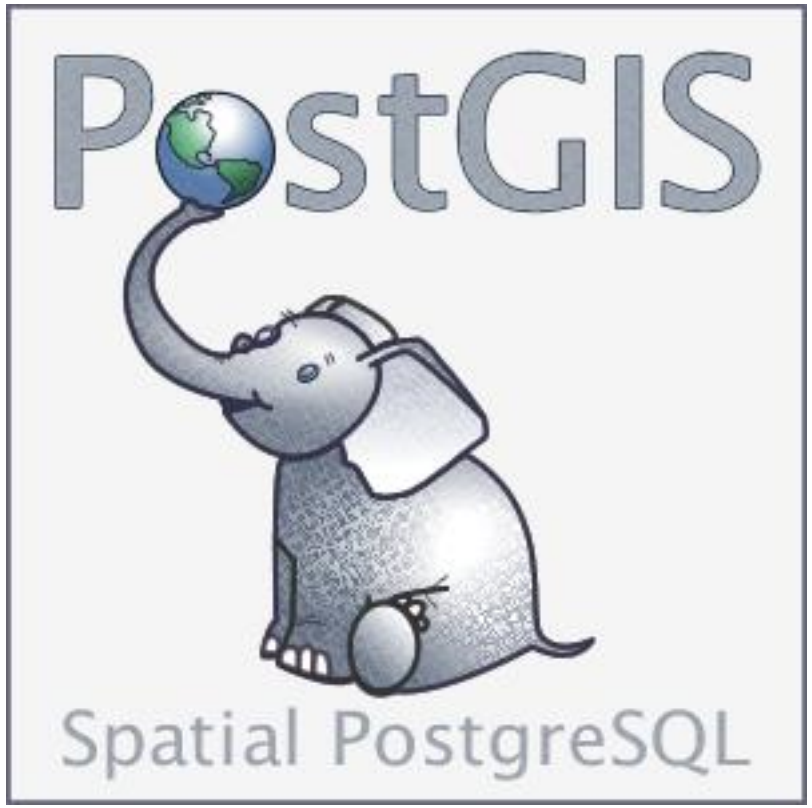


Challenges

- Out of Memory
- PostGIS extension

PostGIS

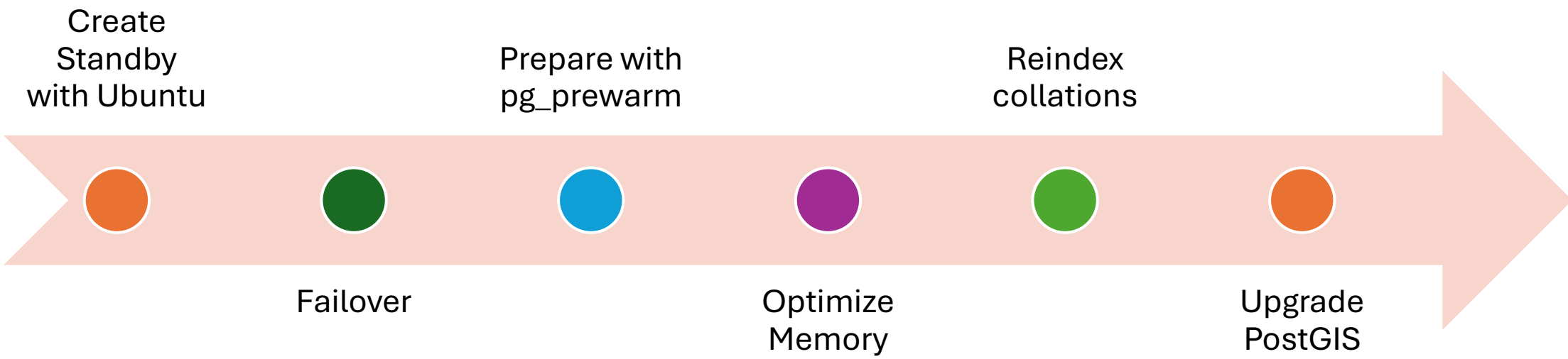
- PostGIS extends capabilities of the PostgreSQL database by adding support for storing, indexing and querying geospatial data.



Upgrading PostGIS

- System library changes
  - PostGIS depends on system libraries like GEOS, GDAL, and PROJ.
- glibc version impact
- Compatibility with Ubuntu libraries
- Smooth migration of spatial data

Migration Plan

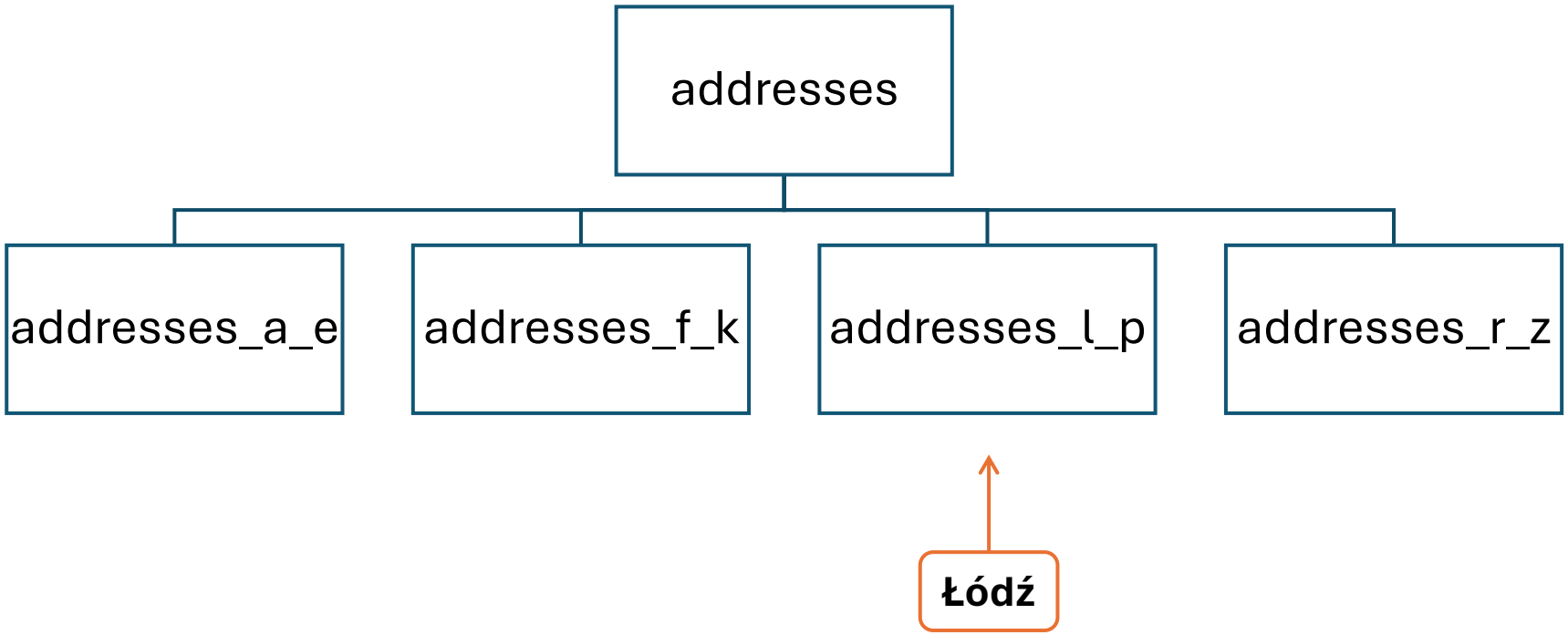


Challenges

- Out of Memory
- PostGIS extension
- Range partition

Range Partition

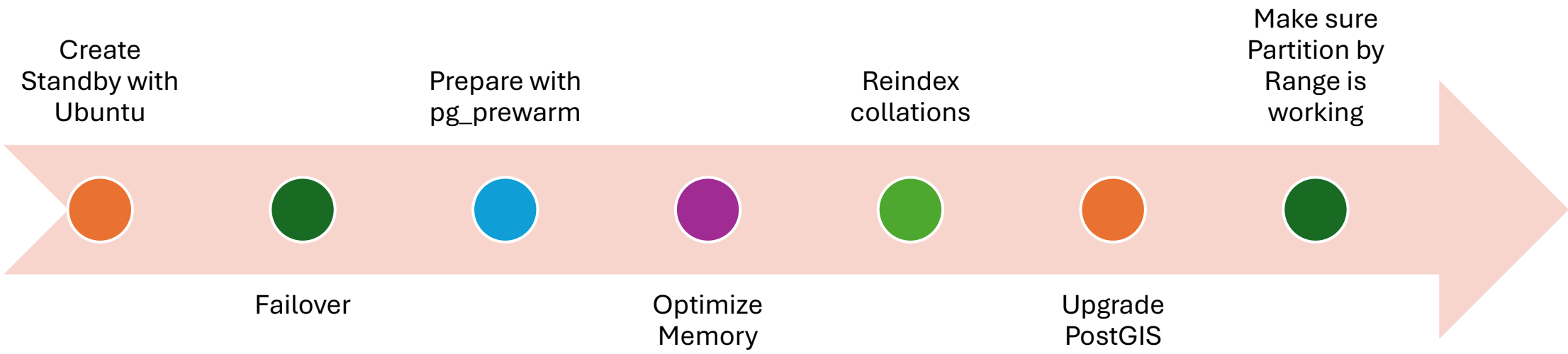
partitioned by **city names**



How to make sure?

- Check if there is range partitioning
- If yes, check if it’s partitioned by a text column

Migration Plan



It was anything, but smooth!

- Memory optimization
- Extension management
- Range partitioning

It was anything, but smooth!

- Migrated over 900 clusters
- Customer discussions
- Night shifts to watch over migrations
- Thanks to my teammates Rıdvan and Volkan

Thank you!

PGCONF.EU
2024

PGConf.EU 2024

22-25 Oct 2024

Microsoft

Microsoft

Meet our Postgres team at PGConf EU 2024

Microsoft

Got 3 minutes? We'd love your input on some of our Postgres work

Get your FREE socks
@ Microsoft booth

Have you listened to TalkingPostgres.com?

Save the date
June 10-12, 2025

POSETTE: An Event for Postgres

2025 Now in it's 4th year!

A free & virtual developer event

Subscribe to news → aka.ms/posette-subscribe

Thank you!