

ELLYPUNK

Foundation

WEEKLY DIGEST / E101

31 August - 6 September 2026



This week

31 August - 6 September 2026

Available archived editions, in their original date order.

THURSDAY / 3 September 2026

Elly Explains: What Is SQL?

3

[Open original online](#)

THURSDAY / 3 September 2026

Kubernetes 101

8

[Open original online](#)

FRIDAY / 4 September 2026

Elly Explains: What Is Kubernetes?

13

[Open original online](#)

ELLY EXPLAINS

PAGE 1

3 SEPTEMBER 2026

WHAT IS SQL?

Hi, I'm Elly!
Let's turn a simple question into an answer from data.

A LANGUAGE FOR DATA

SQL =
Structured Query
Language

SQL is a language for reading and changing data in relational databases.

Which menu items cost less than £4?



You describe the **result** you want. The database system works out how to find it.

Start with a clear question.

NEXT: Tables, rows and columns

MEET A TABLE

A relational database organizes data into **tables**.

menu

item_id	item	price_pence
1	Tea	200
2	Coffee	300
3	Cake	450

Prices are whole pence: 200p = £2.

TABLE

A collection of related data.

ROW

One menu item in this example.

1	Tea	200
2	Coffee	300
3	Cake	450

COLUMN

One kind of detail, such as price.

		200
		300
		450

Each item has an ID, a name and a price.



ID



NAME



PRICE



Rows hold records. **Columns** describe them.

NEXT:

Ask your first question



ASK WITH SELECT ☆



SQL editor • Linux / macOS / Windows

```
SELECT item, price_pence
FROM menu
WHERE price_pence < 400
ORDER BY price_pence;
```

Using the **menu** table from **PAGE 2**.

SELECT

Choose the output columns.



FROM

Choose the table.



WHERE

Keep rows that match.



ORDER BY

Sort the result.



RESULT

item	price_pence
Tea	200
Coffee	300



Tea £2 • Coffee £3



Cake costs 450p, so it does not match.



This **SELECT** reads the data; it does not change the **menu**.

NEXT: A language, many uses →

A LANGUAGE, MANY USES



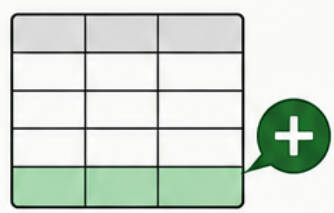
SELECT

Read matching data.



INSERT

Add new rows.



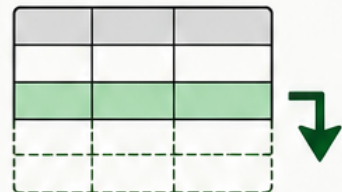
UPDATE

Change existing values.



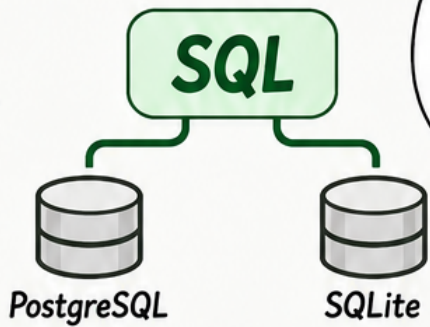
DELETE

Remove rows.

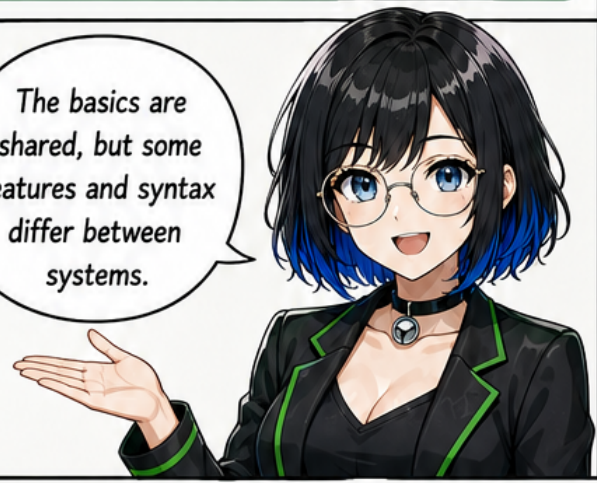


SQL IS THE LANGUAGE

PostgreSQL and SQLite are database systems that understand SQL.



The basics are shared, but some features and syntax differ between systems.

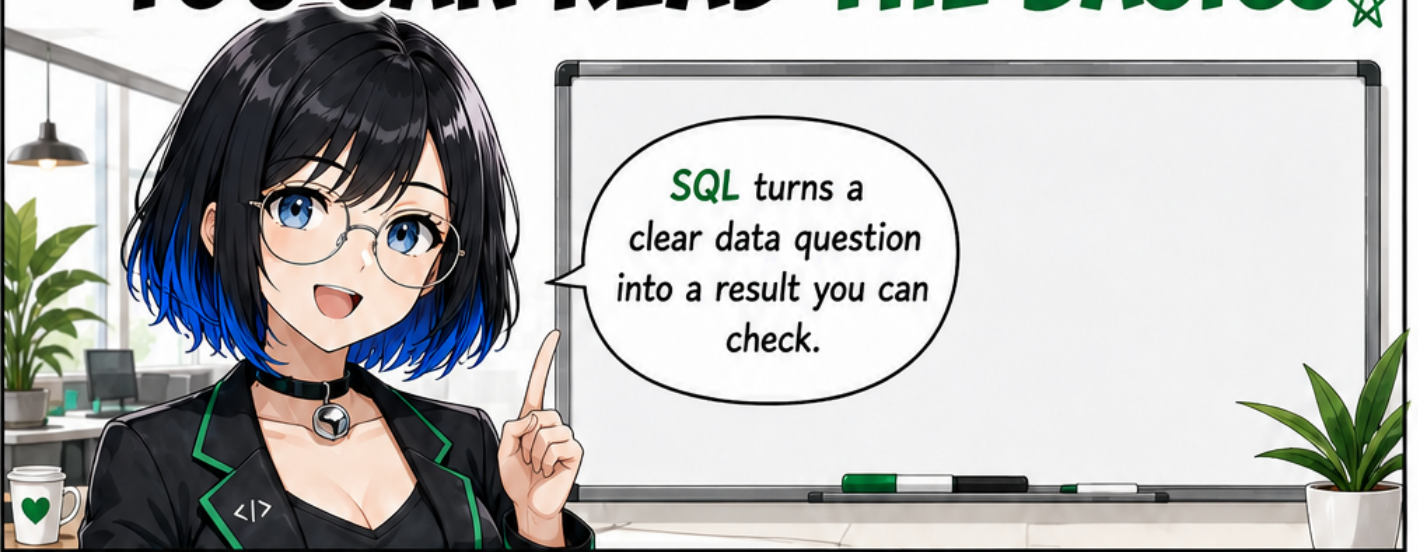


Start by reading sample data.
Learn changes in a practice database.

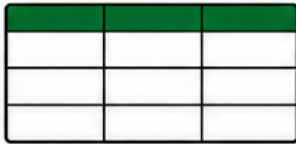


NEXT: Your SQL check-in

YOU CAN READ THE BASICS ☆



1 • KNOW YOUR DATA



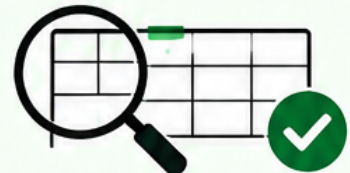
Find the table and its columns.

2 • ASK PRECISELY



Choose columns, filter rows, then sort.

3 • CHECK THE ANSWER



Does the result match the question?



MINI GLOSSARY



Table: related rows and columns.



Query: a request for data.



Result: the data returned.



QUICK CHECK

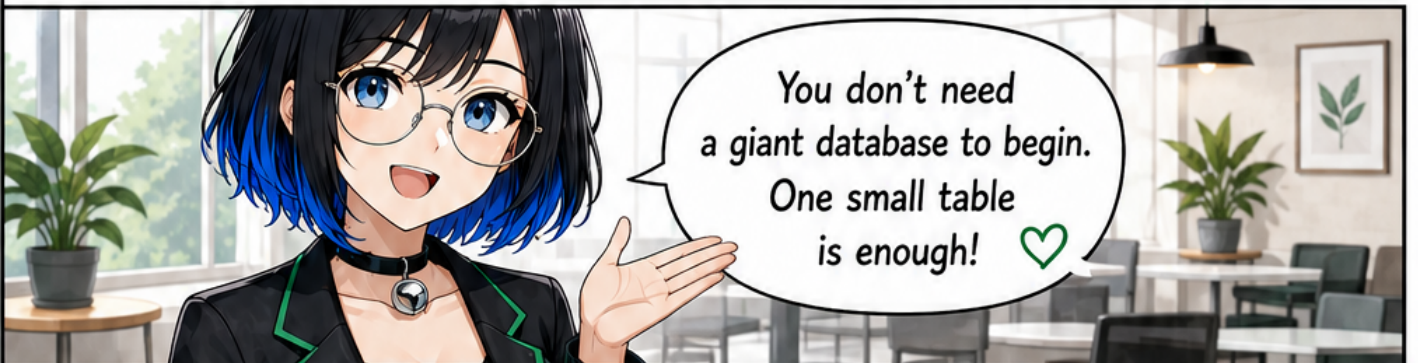
Which keyword filters rows:
SELECT or **WHERE**?



ANSWER



WHERE filters rows.
SELECT chooses the output.



☆ SQL = Structured Query Language ☆

KUBERNETES 101

RUN, REACH, RECOVER

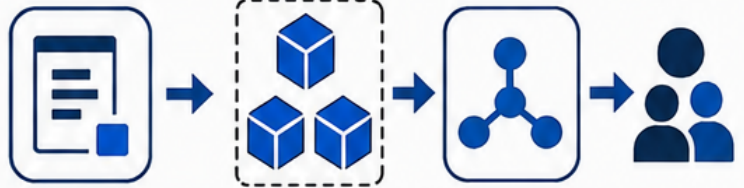


Elly: Kubernetes keeps desired state. You define the desired state.

OBJECTIVE

Map an app from Deployment to Pods to Service.

DEPLOYMENT → PODS → SERVICE → USERS



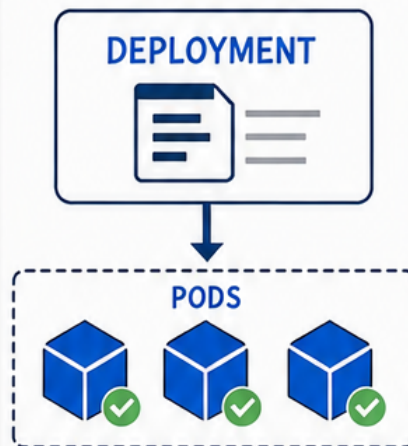
1 START WITH THE USER

What result should people reach?



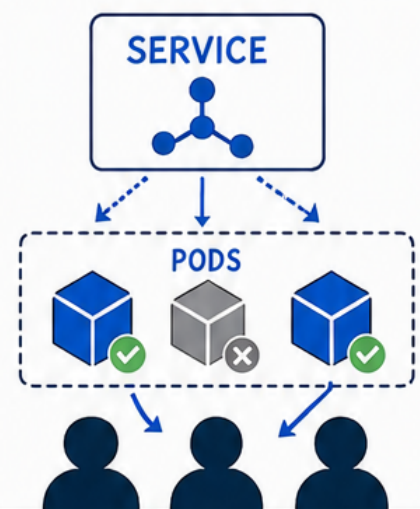
2 DESCRIBE THE WORKLOAD

How many app replicas should exist?



3 CHECK THE PATH

Which ready Pods should receive traffic?



Linux / macOS / Windows

Concepts only. No platform-specific commands.



KUBERNETES 101

PAGE 2

THE CLUSTER MAP



CONTROL PLANE

Stores cluster state and makes scheduling decisions.



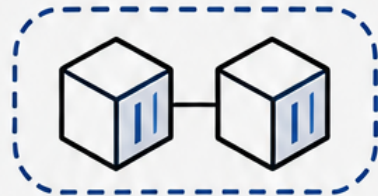
NODE

A machine that runs Pods.



POD

The smallest deployable unit. Containers in a Pod can share network and storage.



CONTAINER

Runs one part of the application.



FLOW



CONTROL PLANE



NODE



CONTAINER

Elly: Learn the map before you debug the traffic.



Linux / macOS / Windows

Concepts only. No platform-specific commands.

KUBERNETES 101

PAGE 3

Elly: Desired state is a contract. Labels connect the contract.



DEPLOYMENT



Declares a Pod template and the desired number of replicas.



REPLICASET



Maintains the requested number of matching Pods.

LABELS

Attach simple key-value tags, such as `app=web`.

`app=web`

SELECTORS

Choose Pods by matching labels.

`app=web`

POD



`app=web`

POD



`app=web`

POD



`app=web`

RULE

A Service selector must match the labels on the Pods it should reach.



`app=web`



`app=web`



`app=web`



`app=web`



Linux / macOS / Windows

Concepts only. No platform-specific commands.

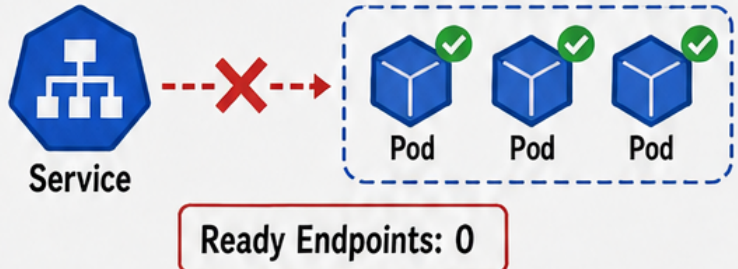
KUBERNETES 101

PAGE 4

CASE FILE: RUNNING, BUT UNREACHABLE

PROBLEM

Three Pods are running, but the Service has zero ready endpoints.



EVIDENCE

Service selector:
app=web

Pod label:
app=api

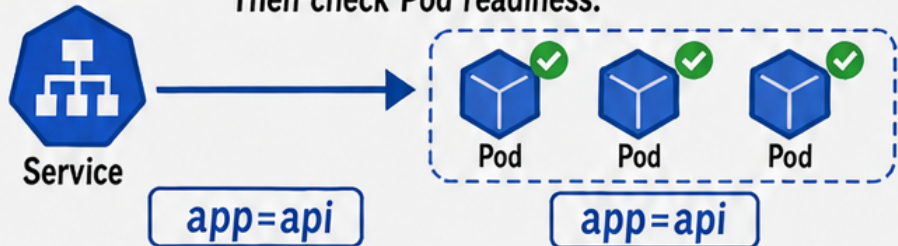
CAUSE

The selector does not match the Pods.

app=web --- $\neq$ --- **app=api**

ACTION

Align the Service selector and Pod labels.
Then check Pod readiness.



CHECK

The Service lists ready matching endpoints.
A user request reaches an app replica.



Linux / macOS / Windows

Concepts only. No platform-specific commands.

KUBERNETES 101

PAGE 5

RUN, REACH, RECOVER

YOUR CHECKLIST

- 1 Describe the workload with a Deployment. 
- 2 Confirm the requested number of Pods. 
- 3 Keep labels and Service selectors aligned. 
- 4 Check that matching Pods are ready. 
- 5 Verify the user can reach the application. 

QUICK DRILL



Pods are running. The Service has no ready endpoints. What do you compare first?



ANSWER

The Service selector, Pod labels, and Pod readiness.



ELLY'S FINAL TIP



Desired state is a promise.
Observations prove it.



Linux / macOS / Windows



Concepts only.
No platform-specific commands.

ELLY EXPLAINS

PAGE 1 / 7



WHAT IS KUBERNETES?



Our cafe menu website runs in Docker.

WHAT IS DOCKER?



Docker is a tool for building and running containers.

A **container** runs your app and what it needs in an **isolated environment**.



ONE COPY CAN BE ENOUGH



For a small app, this **simple setup** may be all you need.

BUT WHAT IF...

It crashes?

Nobody can reach the app.



More users arrive?

One copy has limits.



You update it?

Stopping it can interrupt users.



THE NEXT IDEA

Run **several copies** of the same app.



NEXT: Why several copies help.

WHY SEVERAL COPIES HELP

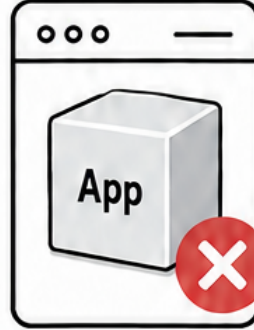
Same app. Separate running copies.



KEEP SERVING

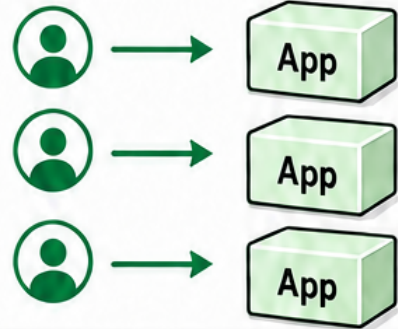
If one copy fails, healthy copies can keep serving users.

Spread copies across machines to survive a machine failure.



HANDLE MORE TRAFFIC

Copies can handle different requests at the same time.



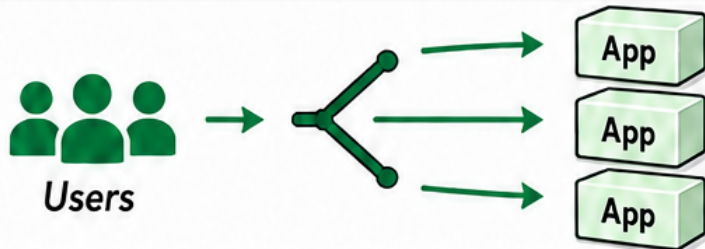
UPDATE GRADUALLY

Replace old copies with new ones while others keep serving.



SHARE THE TRAFFIC

Traffic can be shared between healthy copies. This is load balancing.



NOW SOMEONE MUST...

Keep enough running. Replace failures. Choose machines. Send traffic to healthy copies.

NEXT:

Enter Kubernetes.

ELLY EXPLAINS

PAGE 3 / 7

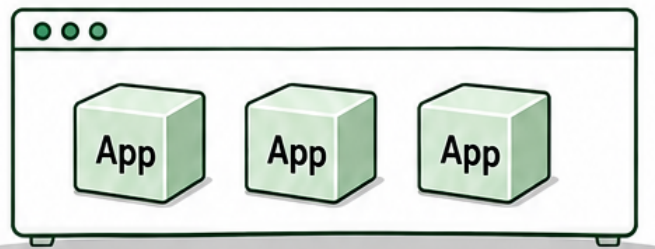


ENTER KUBERNETES

Running copies is one job.
Coordinating them is another.

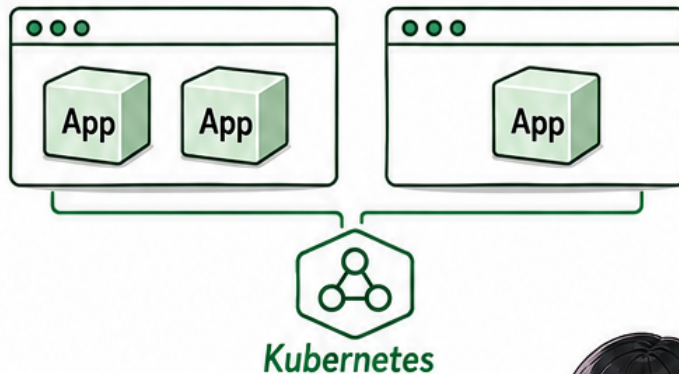
DOCKER RUNS CONTAINERS

Docker can run
one container
or **many**.



KUBERNETES COORDINATES

Kubernetes manages
many containers
across machines.



MORE THAN STARTING THEM



Starting containers alone
does not coordinate
failures, placement, traffic
and updates across
machines.

Kubernetes is **one way**
to manage that
extra work.



YOU SET THE GOAL



Keep 3
copies
running.



Kubernetes
keeps **checking**
and **acting**
to maintain
that goal.



Simple setup
enough? You
may not need
Kubernetes.

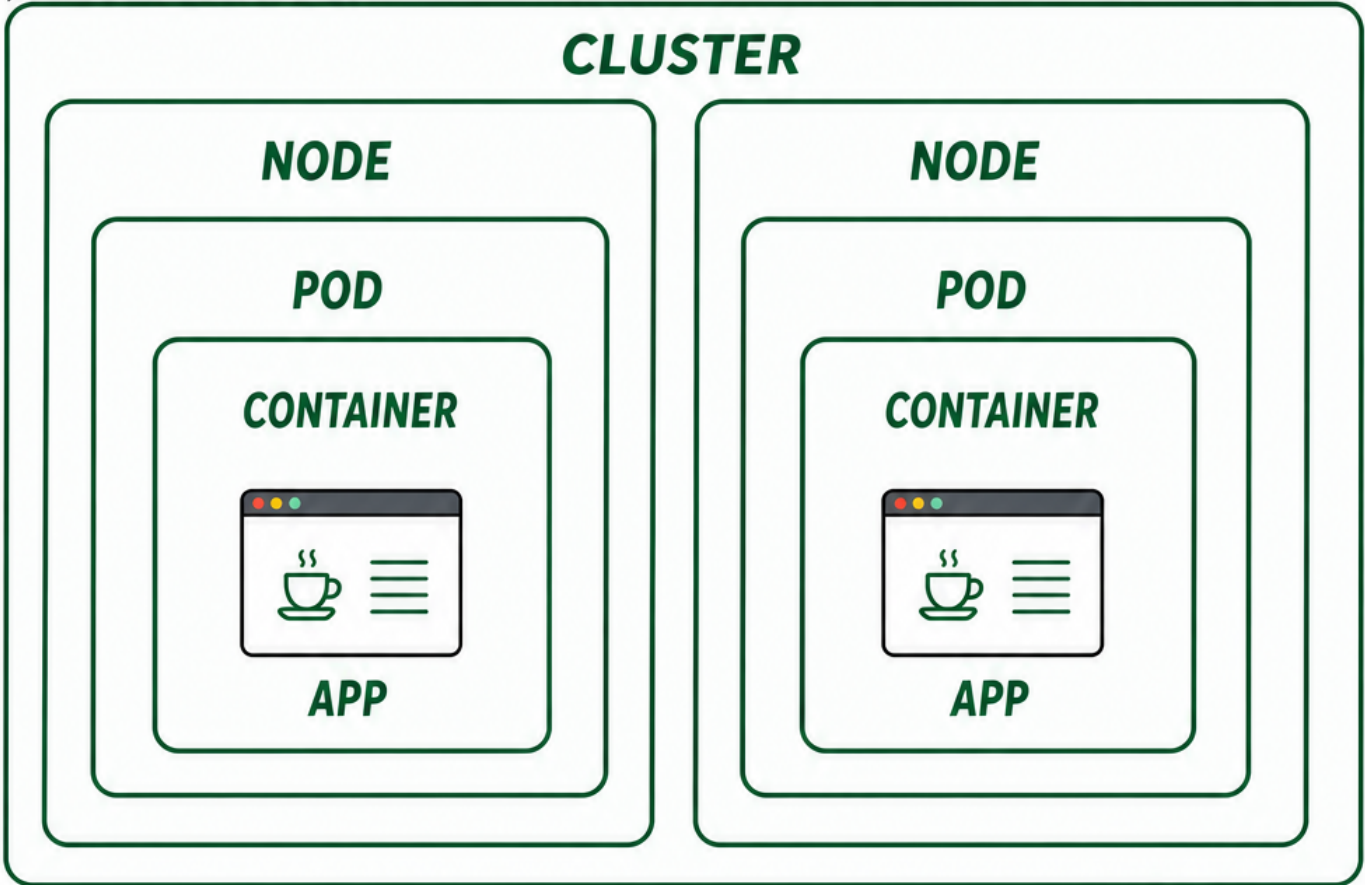
NEXT: Meet the basic parts.



MEET THE BASIC PARTS

Zoom in: Cluster → Node → Pod → Container

CLUSTER



CLUSTER

The machines and management that work together.



NODE

A machine where Pods run.



POD

A running home for one or more containers in Kubernetes.



CONTAINER

An isolated environment that runs the app and its dependencies.



APP

The software, such as our cafe menu website.



In this example: one app container per Pod.



More copies means more Pods, not more apps squeezed into one Pod.

NEXT:

Desired state in action.



DESIRED STATE IN ACTION



A **Deployment** describes which app to run and how many Pods you want.

“ You declare a goal.
Kubernetes continually works toward it. ”

DESIRED: 3 PODS

A

B

C

1 • ACTUAL: 3

A

B

C

✓ 3 ready

2 • POD B IS GONE
ACTUAL: 2

A

C

✓ 2 ready

3 • CREATE A REPLACEMENT
ACTUAL: 3

A

C

D

✓ 3 ready



Kubernetes sees 2 instead of 3, so it **creates** a replacement Pod.



D is **new**. It is not B moved elsewhere.



This takes **time** and available resources; it is not instant.

NEXT: Helpful, not magic.



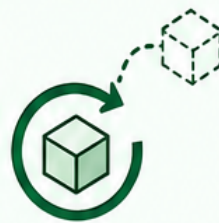
HELPFUL, NOT MAGIC

WHAT KUBERNETES HELPS WITH



KEEP THE COUNT

Maintain the number of Pods you asked for.



REPLACE FAILURES

Replace lost Pods; restart failed containers.



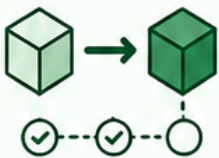
CHOOSE MACHINES

Place Pods on suitable Nodes and help spread the work.



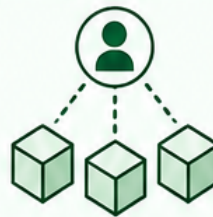
SCALE UP OR DOWN

Request more or fewer copies. Autoscaling needs configuration.



ROLL OUT UPDATES

Introduce new versions gradually.



SHARE TRAFFIC

Send requests to healthy, ready copies.



YOU STILL DO THE THINKING



You configure resources, health checks and placement rules.



Kubernetes does not build your app, write your code or magically fix bugs.



Not every app needs Kubernetes. Choose it when the coordination is worth the extra complexity.

NEXT:

Check your understanding.

ELLY EXPLAINS

PAGE 7 / 7



CAN YOU EXPLAIN WHY? WHY SEVERAL COPIES?



Reliability.



More capacity.



Gradual updates.



WHY NOT JUST START CONTAINERS?



Lots of copies need ongoing coordination across machines.

QUICK CHECK

A **Deployment** wants 3 Pods. Only 2 remain. What happens next?



ANSWER



Kubernetes creates a **replacement Pod** to work toward 3 again.



It needs **resources** and **time** to become ready.

THE DIFFERENCE



Docker runs containers.



Kubernetes manages many containers.

THE BIG IDEA

“ You declare a goal. Kubernetes continually works toward it.

Not every app needs Kubernetes. ”



END:

Keep learning, one question at a time.