Bı	ı Pİ	ld	ir	na	a	\mathbf{C}		11	d-	-N	้อ่	tix	70	S	0	D	at	ał	าล	Se		
		ТМ																			\smile	
Cocł																0						
	kroa	chD	B: So	calat	ole, S	Surv	ivab	le, C	onsi	ister	it, SC	QĽ										
Cocl	kroa	chD	B: Sc	calat	ole, S	Surv	ivab	le, C	onsi	ister	it, SC	QL °										
Cocł	kroa °	chD	B: So	calat	ole, S	Surv	ivab	le, C	onsi	ister	nt, SC	QL o										
Cocł	kroa o o	chD °	B: Sc	calat	ole, S	Surv	ivab °	le, C	onsi	ister	it, SC	}L ○ ○										



@cockroachdb

	W	'h	at	1	S° '	"C	lo	U	d-	N	a	tiv	7e	"?							
0	A C																				
	\mathbb{V}_{\circ}																	000	ockro	bachdk	

What is "Cloud-Native"?							
Horizontally scalable							
 Built to handle failures 							
• No single point of failure							
Survivable (self-healing)							
 Minimal operator overhead (aut 	oma	atab	le)				
Decoupled from underlying plat	forn	n °					
				0	cockro	achdb	0 0

What isn't "Cloud-Native"?						
Can't scale out (monolithic)						
 Not designed to recover from failures 						
 Single point of failure 						
 Can't spread across availability zoi 						
Call (Spleau across availability 20)	16:	20				
 Manual recovery/failover 						
Tied to a particular platform						
			000	ockroa	achdb	0

	Cl	ou	d-1	nat	tiv	e o	dat	ab	as	es	0													
	0	0	So	ur	nds	sa	lo	t li	ike	e m	ıar	ıy	No	oS(QL	D	Bs,	ri	gh	t?				
			0		° Aml		ہ tit						ہ + د	° + ⊃]		。 + i i	o D O	° fai	。]					
			0	Ľ.	epi	IICa	au	JŰI,	SC	all	ing	ι οι	ц,	ίĝι	era	ιIJ	.1 g	ıäı	ıuı	es				
	0	0	Βu	lt v	wh	at	ar	e t	he	ey]	lac	ki	ng	?										
0	×																							
																					00	ockro	achdb)

Data	base Limitations													
• • Exi	sting database solutio	ns p	pla	ce	an	ur	ndu	le l	bu	rde	n (on		
app	olication developers:													
0 0 ∎0	Scale (sql)													
0 0 0	Fault tolerance (sql)													
	Limited transactions (nosq	[l)												
	Limited indexes (nosql)													
	Consistency issues (nosql)													
0 0 0	Consistency 1350c5 (1105q1)													
													o ab db	
· • • •											(aco	CKroa	aprice	

0																					
0																					
ō																					
0																					
0																					
0	W	h	V	sł	10	u	d	V	01]	Ca	r e	2.5	ab	01	ıt					
0														~+							
0 0	C	on	S	IS	te	no	су	0	r	tr	ar	ns	a	ct	io	n	s?				
0 0 0	CC	on	lS]	S	te	n	су	0		tr	ar	15	a	ct	io	n	s?				
0 0 0	CC	on	IS]	İS	te		су	0		tr	ar	າຣ	a	ct	io	n	s?				
0 0 0 0	CC		IS S	IS °	te		су) 1 • •	tr	aı	1S	a(ct	io °		s?				
0 0 0 0 0			IS S		te		сy	0		tr °	aı	1 S		ct :	i0 • •		s?				
0 0 0 0 0 0			S 0 0	IS °	s e		сy) (() () ()()()()()()()()()()		aı	1 S	a.	ct :	io °		s?				

Strong Consistency														
Reasoning about eventual consistency is hard • Wastes developer time														
 Wastes developer time 														
 Causes bugs 														
	0 0													
Avoid stale reads or data loss on failover														
Enables true SQL support														
	a a	cockroa	achdb											

	ہ 																							0
0	Tra	an	sa	CII	OL	IS																		0
0																								0
0																								0
0																								0
0			"We	e be	liev	ve it	is l	bett	er t	o h	ave	apj	olic	atio	n p	rog.	ram	nme	rs c	leal	l wi	th		0
0		p	erfo	orm	and	ce p	rob	lem	s d	ue t	to 0	veri	use	of t	ran	sac	tior	ıs a	s ba	ottle	ene	cks		0
0		0	ari	se.	ratl	her	tha	n al	way	s VS C	odi	na a	aroi	und	the	lac	ck o	f tra	ans	acti	ions	5″		0
0			0	0	0	0	0	0	-			0			0	ö	0	0	0	0	0	0		0
0											0	Aut	tho	rs of	f Go	ogl	e's	Spa	nne	er p	ape	r		0
0																								0
0																								0
Q	×																							0
0																					00	cockro	achdb	0

0																			
0																			
0																			
0																			
0																			
0	W	'h	V	is	t	hi	S	ha	ar	d	20								
Ø	0	0		0	0	0	101	0	10	0	0								
0																			
0																			
0																			
0																			
0																			
0	\mathbf{V}_{\circ}															000	ockro	oachdk	0

Why is getting the best of both worlds so hard? Fundamentally, coordination is very difficult Especially when time is involved Building a database is hard enough as it is It's tough to pair consistency with performance and scalability

0																							
0																							
0																							
0																							
0																							
Ø	Η	٥١	N	d	be	S	C	00	\mathbf{k}	rc)a	cł	٦D)B	C	0	it	?					
0	0	0	0	0	0	0	101	0	0	0	0	0	0	0	0	0	0	0					
Ø																							
0																							
0																							
Ø																							
Ø																							
0	A A																						
0	\mathbf{V}°																			000	ockro	achdl	o c

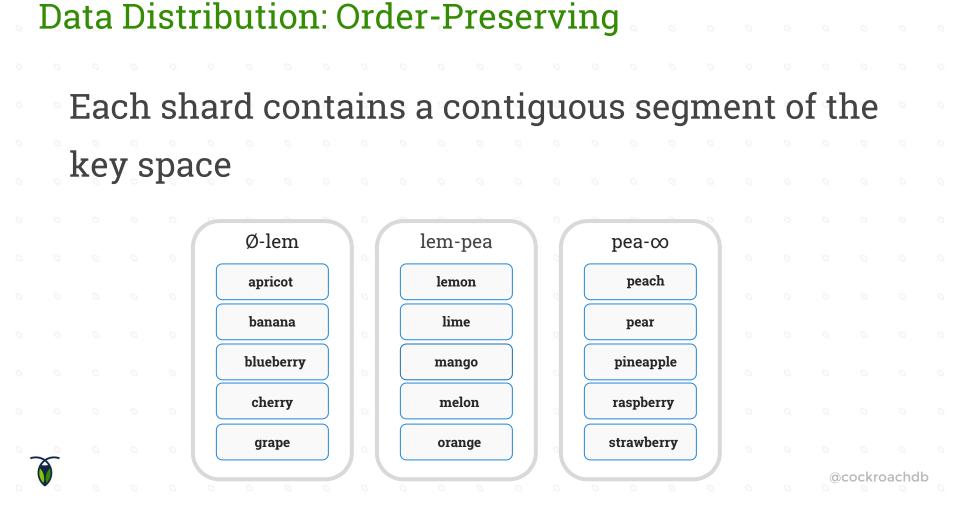
0	Ho	W	do	es	C	00	kro	ba	ch	DE	8 d	oi	t?									
	° 1	0	Da	ta	di	sti	rib	ut	ior	ו a	nd	l re	epl	ica	ati	on						
	0	ò	0						0			0	0									
	ູ2	0	Co	ns	en	ISU	lS	pro	oto	CC		Ra	itt)									
	ິວ	0	Di	etr	ih	unt <i>a</i>	Ба	tra	n	ດີ	-ti	on	m	പ	പ							
		0		0			s N	0	°	Sav			0	°.								
0	0																					
0																			@c	ockro	achdb	0

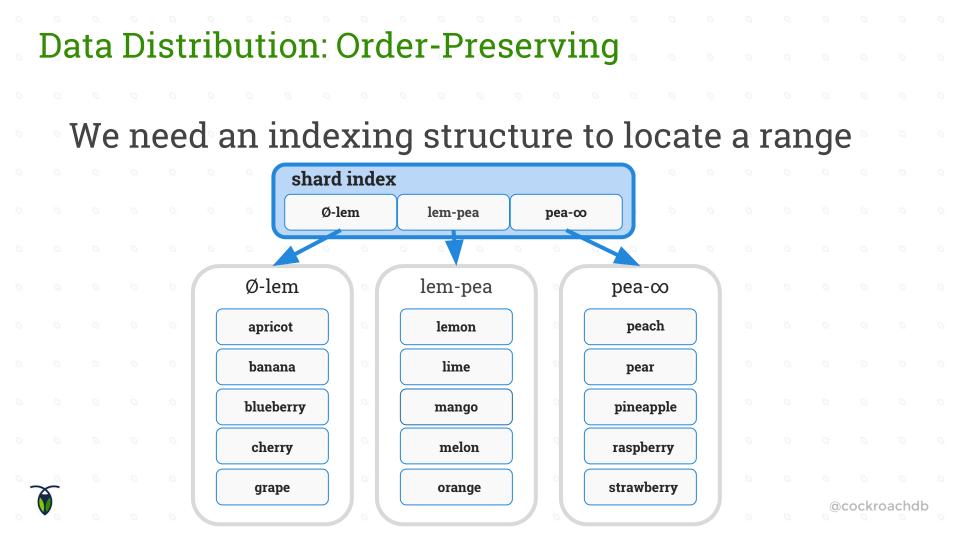
	Ar	ch	ite	ect	ur	。 e。(hi	gh	-le	eve	l)											
										Abs	tract	ion s	stacl	с: С:								
									0	0	0	0	0	0	Q							
												SQL				Q						
										Tr	ansa	actio	nal I	KV		0						
											Dist	ribu	tion			0						
										5	Der	1		R.		Ö						
											кер	olicat	tion			0						
											S	toraç	ge			0						
Ø	0																					
	V																		00	cockro	achdb	0

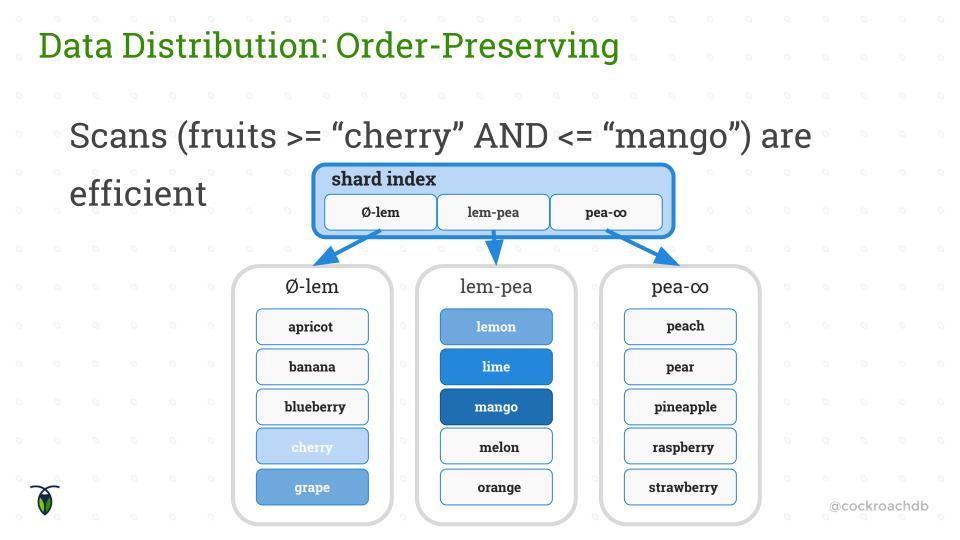
0																							
0																							
0																							
0																							
0					0		0		0				1	- 0	0	0	0	0	7				
0	D	at	a	\mathbf{D}	İS	tr	ib	u	tic	on	0		1			SC)L						
0															Tran	sact	iona	l KV		J			
0															Di	istrik	outio	n					
0															B	enlic	atio	n	0				
0																epiie	Julio.			4			
0																Stor	age						
0													0	~ _	_0	0	9	0	0	10			
0	×°																						
0	\mathbf{V}_{\circ}																			0	cockro	oachdl	b

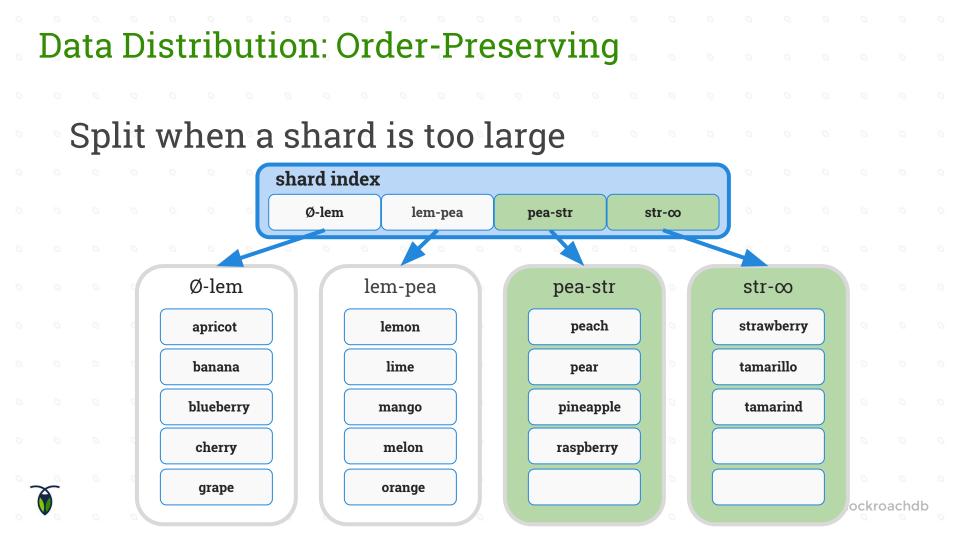
Data Distribution													
Two key questions:													
At what granularit	y is	data	a d	ist	rik	out	ec	<u>]</u> ?					
How do I locate a p	bart	icula	ar I	pie	ce	of	ď	ata	a ?				
The primary options:													
Hashing or Order-Pre	ser												
										000	ockroa	chdb	

			Di													0								
	0	Гh	ea	lte	ern	nat	ive	e t	oh	las	shi	ng	is is	a	1 °0	rd	er	-pi	ces	er	vir	ŋ		
	° (da [.]	ta d	dis	stri	ibι	iti	on	0															
	0	0	0	0	0	0	0	0	0															
	0	0	Pr	0 : (eff	ici	ier	nt° s	SČa	ans	S													
													0											
	0	0	Со	n:	re	qu	ire	es	ad	dit	io	na	lir	nd	ex	ing								
0	X																							
																					0	cockro	achdb	

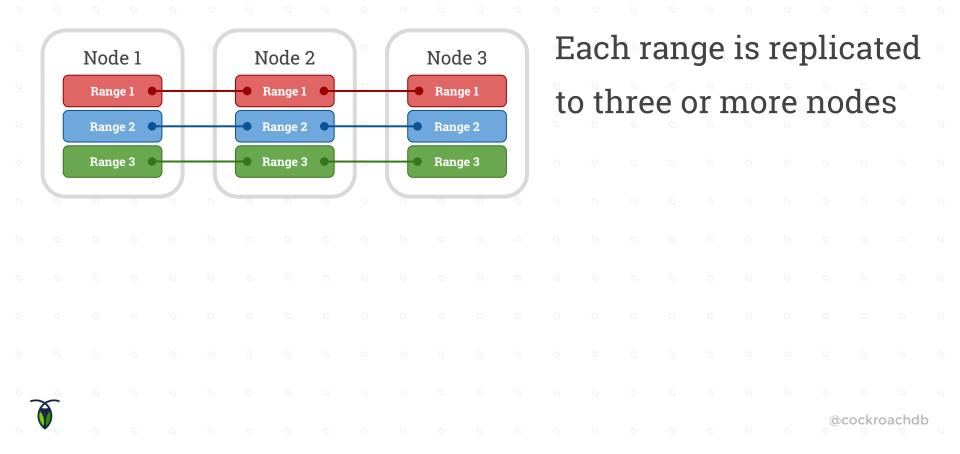


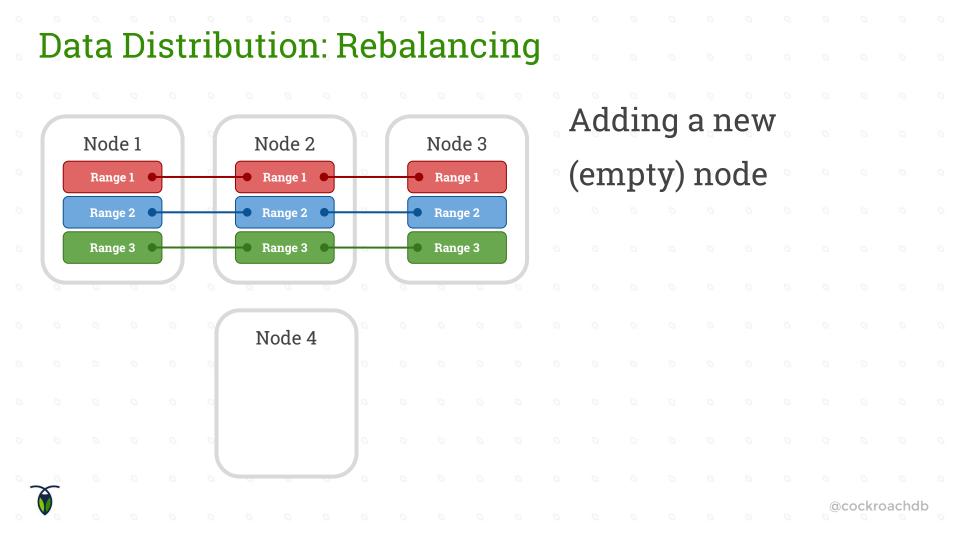


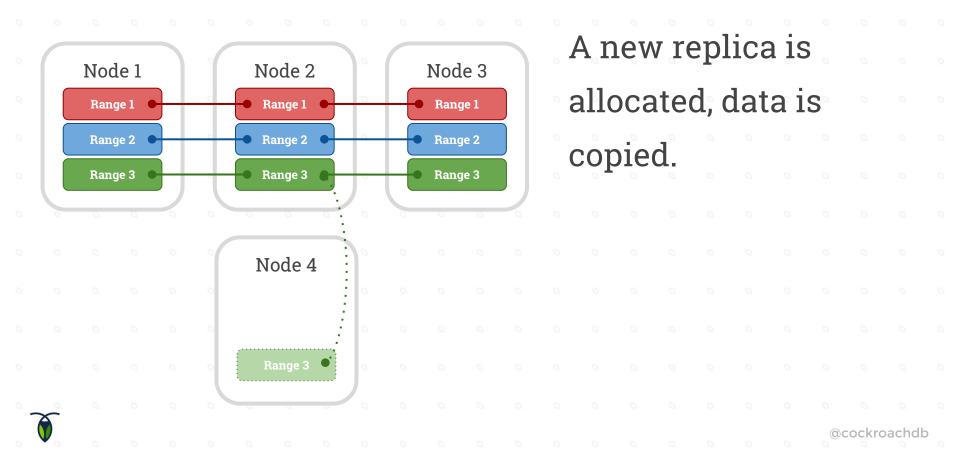


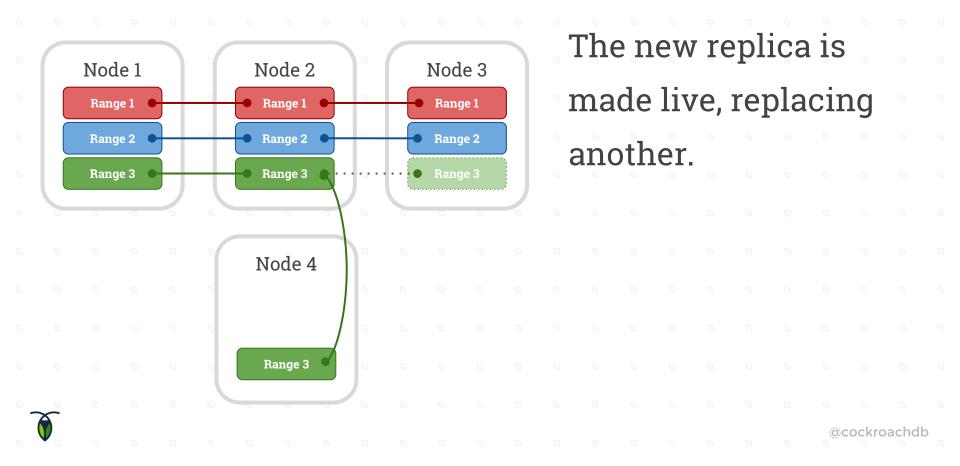


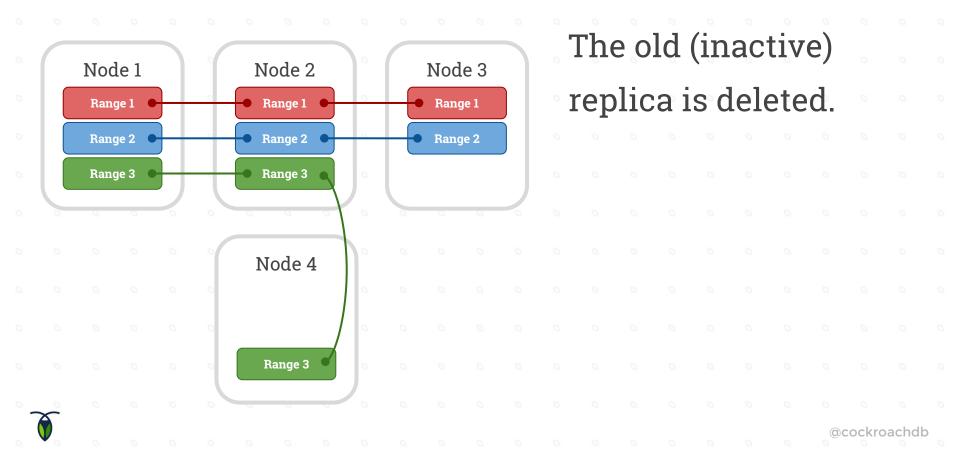
Data Distribution: Placement

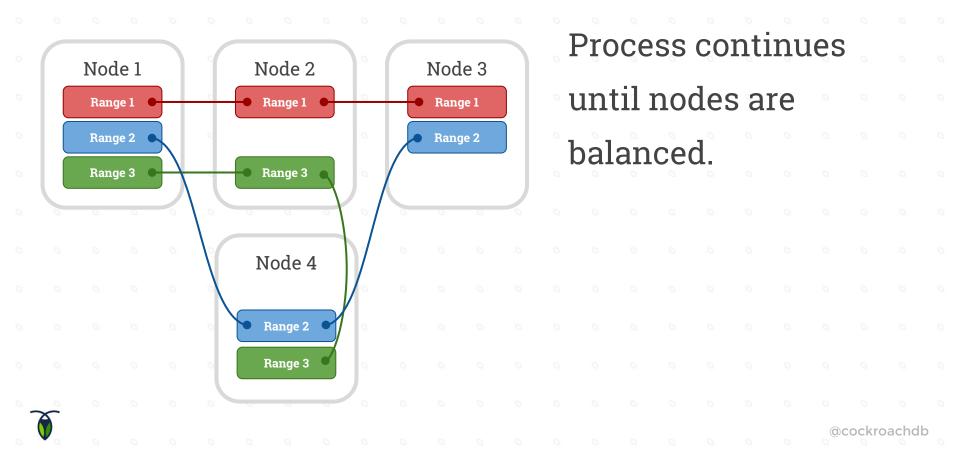


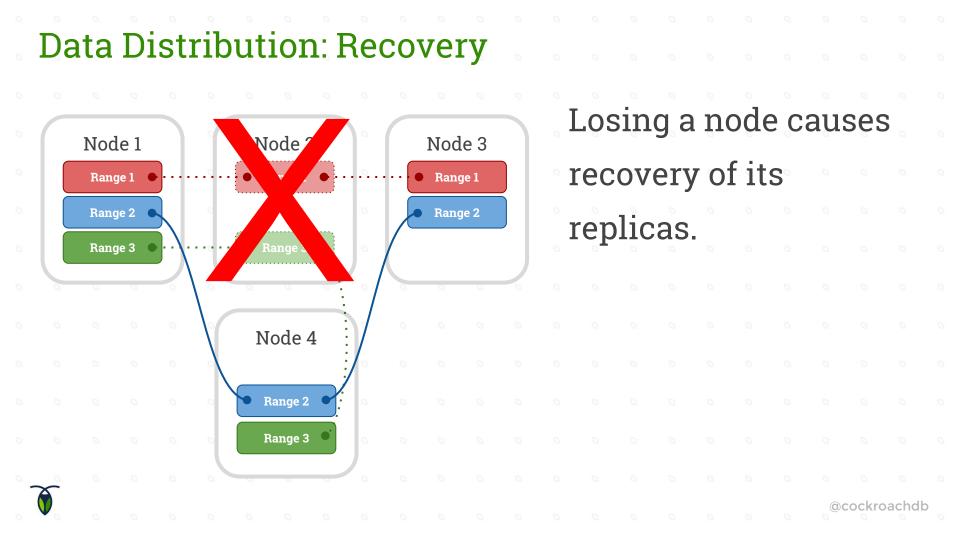












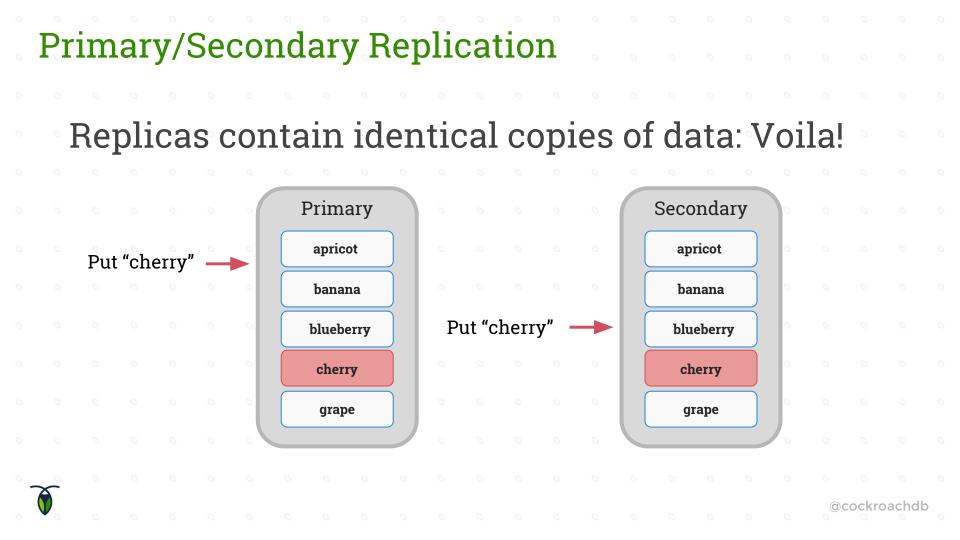
	Da	ta	Di	st	rik	out	ion	: R	ec	° 100	ver	у У												
	0	0	0	0	0	Q	0	0		Q	0	0	0	0										
0		Nod	e 1			N	ode 🖊				Node	e 3		° F	4 11	ew	, TE	:hr	iça	ge	:15			
0		Rang	e1 🌢	L .	c		V,				Rang	e 1		° C	re	ate	ed o	on	an					
0		Rang	e 2 🔍		0				6		Range	e 2				oti	n d		de					
0		Rang	e 3 🔍	$\left \right\rangle$	¢	R	ange J				Range	e 3		° C		διΙ.	ug		oue	0				
	0	0	0	~	0	0	0	0	$\left \right $	0	0	0	0											
				6		N	ode 4																	
						$\$	Range 1																	
						• F	lange 2		0															
					0	F	lange 3																	
~	0				0	-	0	0																
																					@cc	ockroa	achdb	0

Da	ata	Di	st	rił	ou [.]	tio	n: F	Re	CO.	vei	ry										
	Nod		•						0	Nod			One					o] d			
	Rang Rang Rang	ye 2 🗨		0				0		Rang Rang Rang	je 2		ep				, 16	old	a 0		
	0	0			0	0	0			0	0	∘ f	or	got	te	n.					
						Node Range I															
				0		Range 2		0													
Ĩ				0	0	0	0	0										@c	ockroa	achdb	

	Da	ata	Di	st	rib	ut	ioı	n °																
		0	Ra	ng	es	ar	e~~	-64	M	IB (of	dat	ta											
			0	Sı	ma	ll e	end	oug	gh	to	be	m	o VO	ed/	′sp	lit	qu	icł	۲					
								0	0						0		0		0		0			
			0	La	arg	ee	enc	oug	jh [·]	to	am	or	tiz	ei	nd	ex	ing	01	ver	he	ad			
		0	Th	nis	is	fa	irl	y ື S	sta	nd	lar	d												
								0																
			0	С	ocł	٢O	ac	hD	B/	Big	gta	ble	e/H	IBa	ase	/S	pa	nn	er					
0	0																							
	V																				000	ockro	achdt	

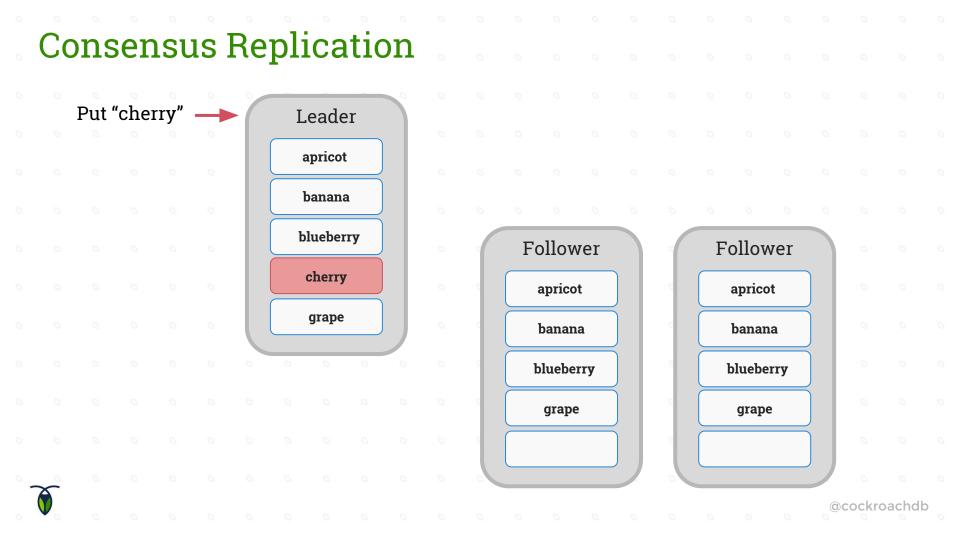
0																					
0																					
Ø																					
0																					
0											1	10	0	0	0	0	~ ~				
0	C	or.	IS	er	1S	U	S							SC)L						
0													Tran	sact	iona	l KV		d			
0													D	istrik	outio	n					
0												0	R	eplic	atio	n	0				
0																					
0														Stor	age			7			
0											0	` _	0	0	9	9	0	Q			
0	×°																				
0	\mathbf{V}_{\circ}																	0	cockrc	achdl	0

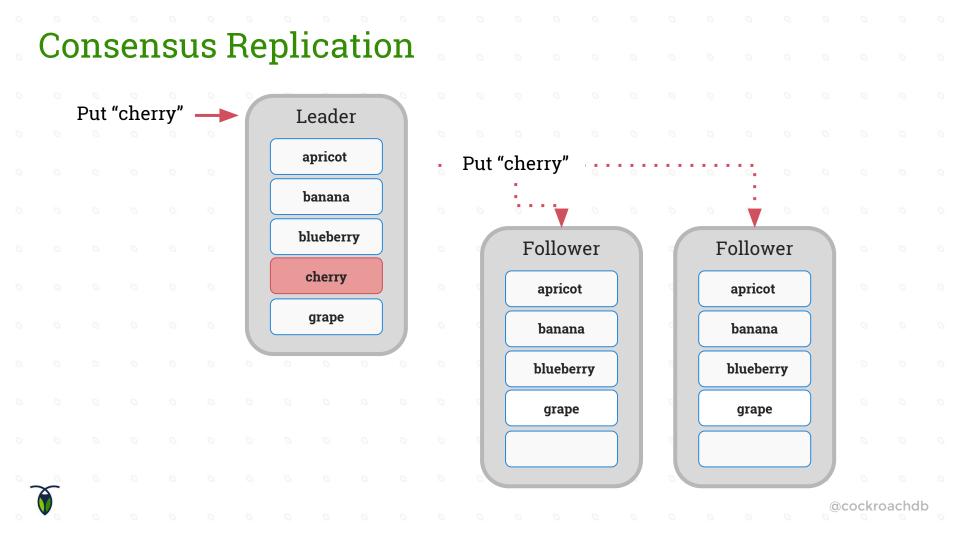
	Ac	h	iev	ing	gC	Cor																		
	0	0	0		0	0	Ø						ø				0	Q.	Ø					
	0	Pro	odı	lCt	10	n l	DB	SI	nu	st	SU	Irv	ive	e n	nao	ch	ine	e fa	ailı	ure	50			
		ò	0	0		0	0	0	о - Т.	Ö.	Q	0	0											
	0	Ag	air	1, T	W		na	IN	alt	er	na	TIV	7es	0										
	0	Ø	Pr			/			ha	o O TI				ot:		0								
	0	0			lai	y /	See		lia	ar	ر x	eþ		al.										
	0	0	Co	ns	ser	າຣເ	lS																	
0	X																							
																					0	ockro	achdk	0

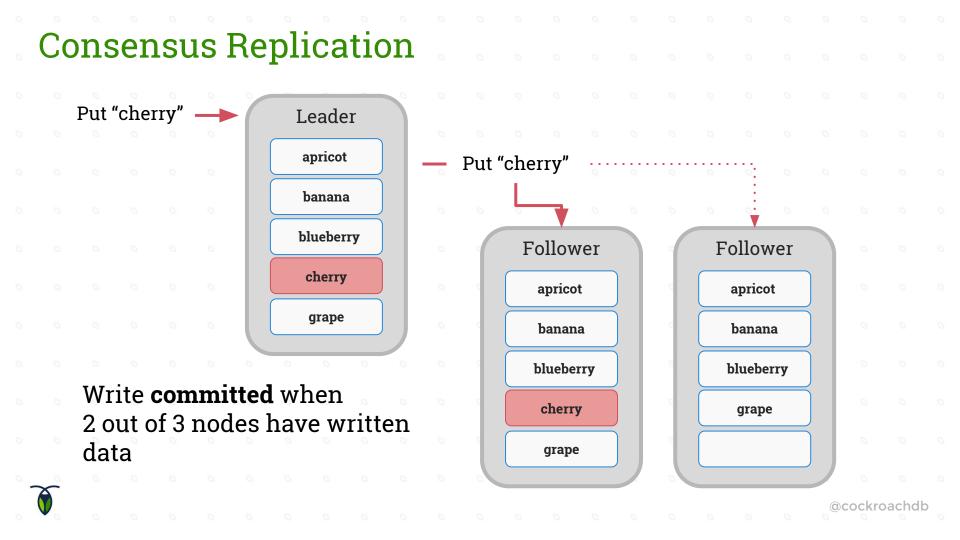


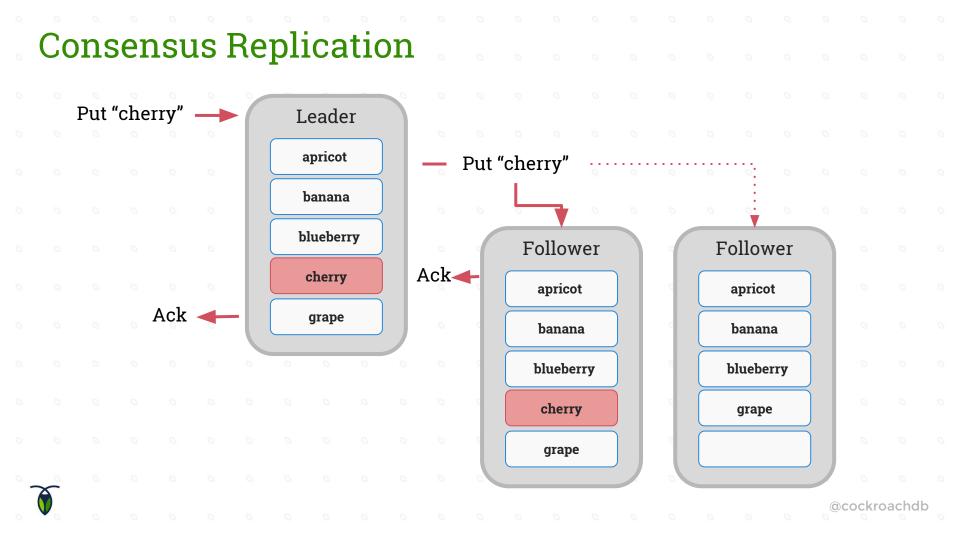
	Pr	in	lar	y/S	Seo	CO]	nd	ar	y F	\er	olio	cat	tio	n										
	0	0	As	syr	neł	nro	n	ous	s =	> 5	sta	le	rea	ad	S									
	0	0		0	° L	0	Q	0	0		0	0	~ .1	ab	:1::	9								
	0	0	Sy			ŐI	ιοι	าร	_>	10	VV	av	all	ap		١y								
	0	Ø	Pr	im	ar	y t	0 3	sec	cor	nd	ary	y fa	ail	OV	er	reo	qu	ire	SC	ar	e			
			0	° Trl	0		0	0	0			0		0	0	0	9	0	0					
			0			a-l	par	τy	ne	ea	St	oʻa			ate	рı	1	lar	y _₀					
0	×																							
																					00	cockro	achdk	C

Consensus Rep														
Doplicato to N	T (ofton													
Replicate to N		° IN-3	5) 110 °	ue	S									
Commit ha	appens	whe	en a	quo	oru	ım	h	av	e v	vri	tte	en		
the data														
CockroachDB	B/Etcd/S	Span	ner/	Άι	llc	ora	/							
													achdb	

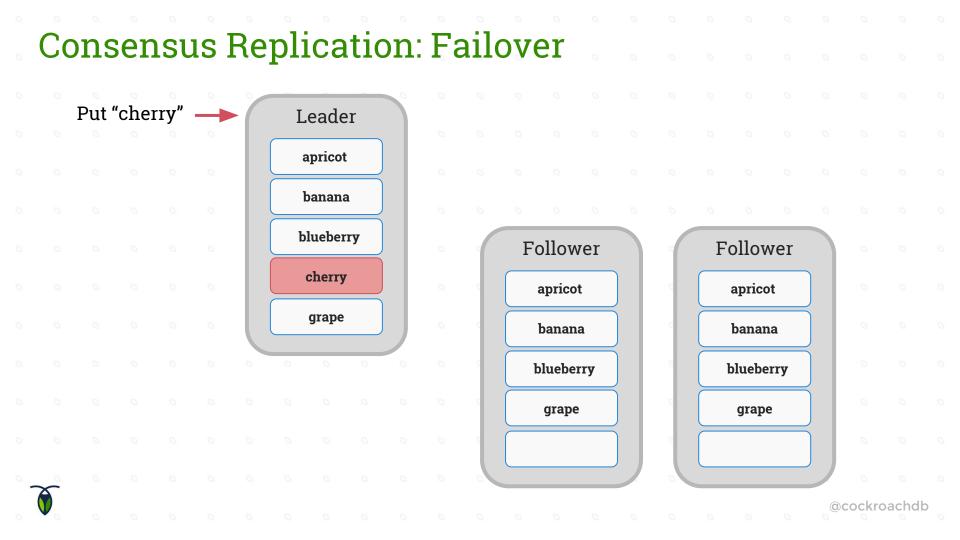


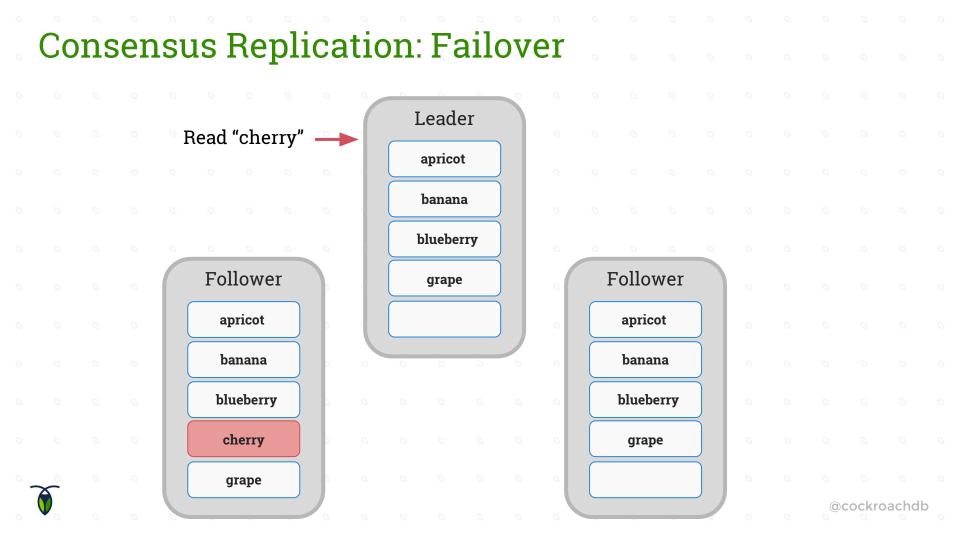


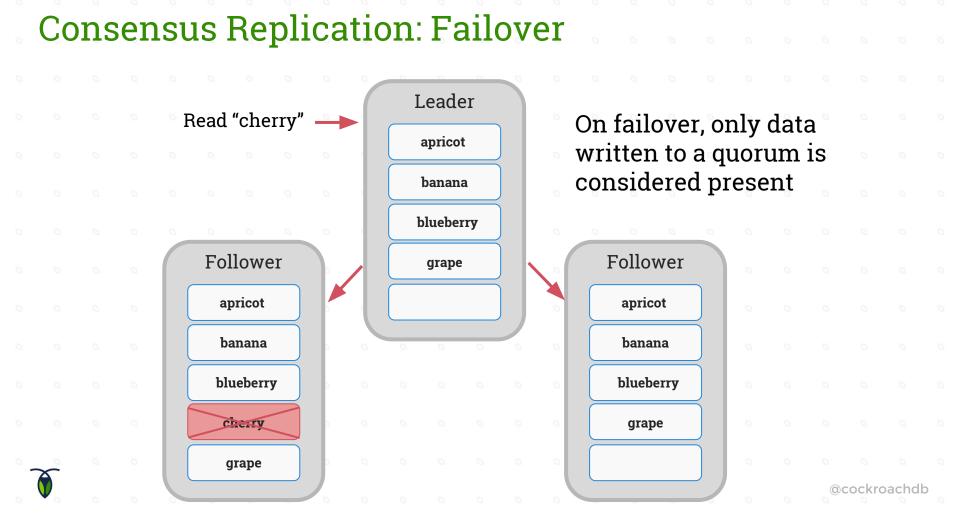


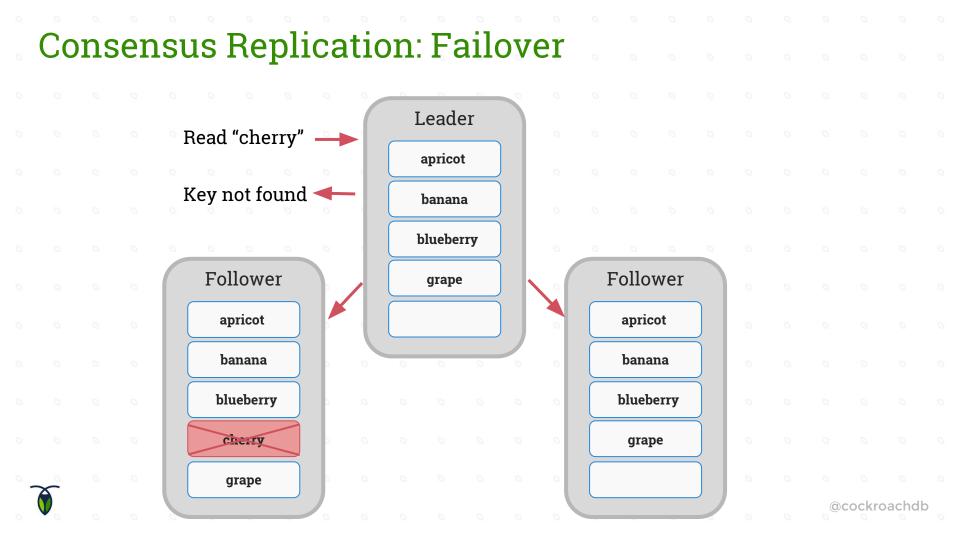


	Со	ns	en	ISU																	
	~ /	Nh	nat	ha	ap	pe	ns	dι	ıri	ng	fa	ilc	Ve	er?							
					0	0				0											
0	A A																				
																		00	ockro	achdb	0









	Сс	ns	ser	ISU	1S .	Re	pli	ica	tic	n														
		0	Ra	ft	is (วนเ	C C C	ons	ser	ISU	IS I	oro	oto	col	of	cł	noi	ce.						
		0	Ru	n	on		on	SPI	ึ้	े 19	ard	זוור	n n	ρr	rar	าต์ค	ں د ∘	f d	ata	0				
		0		0				001			910	Jul				-g	0			0				
		Ø	Pr	act	tica	alo	cor	np	lic	ati	on	S: 1	me	m	bei	cl	nar	nge	e,∘r	an	ge			
			sp	lits	s,°u		ira	des	s, s	ca	lin	d I	านr	nb	er	of	rai	nge	es					
			_		-	6																		
Q	0																							
	()																				00	ockro	achdb)

	Сс	n	sen	ISU	is i																			
		0	Со	ns	en	su	s°p	oro	vid	les	"a	tor	nio	c″≀r	ep	lic	ati	on	0					
			0	Bı	1+ 4	on.	- , , 1	for	ea	ch	ro	na			0									
			0				L N I		Cd			ing	C											
		0	W	nat	t al	00	ut o	ope	era	tio	ns	th	at	hit	m	ul	tip	le :	rar	ıge	es?			
Q	X																							
																					000	ockro	achdb	0

0																				
0																				
0																				
0																				
0											1		0	0	0	0	~~	0		
0	D	is	tr	ib	u	te	d				ł			SC)L					
0	Ť											1	Tran	sact	iona	l KV		J I		
0	1.	la	113	20				12					D	istrik	outio	n				
0											₽	0	B	enlic	atio	n	0			
0													11	epiic	allu	11				
0														Stor	age);		
0 0												` _		Stor	age	<u>a</u>				
0 0 0	0											- 0	0	Stor	age	0	1 0 0			

Transactions

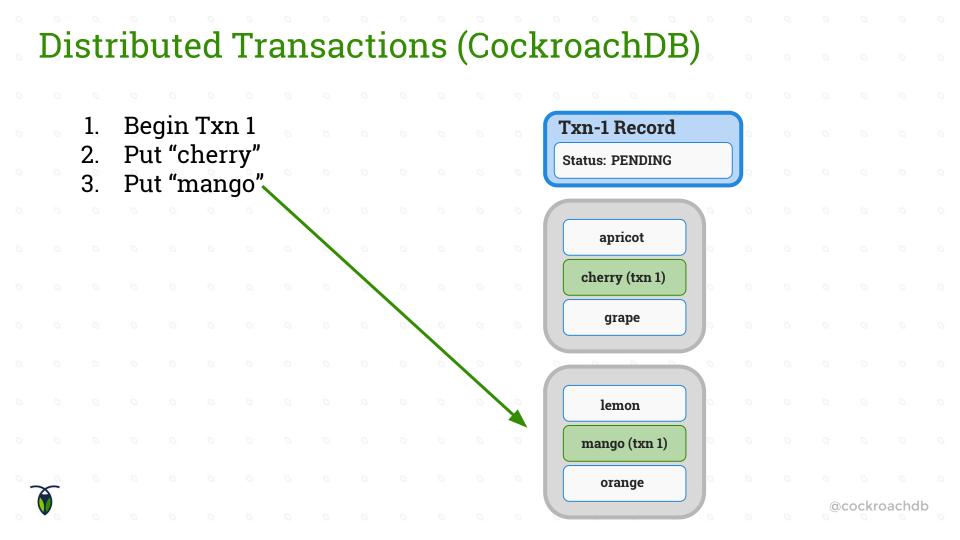
		Ö	Coo	ckr	oac	hD	Bs	up	oor	ts t	rad	itic	ona	l A(CID	se	ma	ntio	S				
			0	"A	ll-c	or-n	oth	ning	g″														
			0	De	efau	ilts	to	the	Se	ria	liza	ble	iso	olat	ior	ı le	vel						
			's fi											-									
	0	trad	ditio	ona	l si	ngl	e-r	lod	e D	B													
0	×																						
0																				@C	ockroa	achdb	

Single-node DB Transactions	
Atomicity and durability are achieved by bootstrapping	
off a lower-level atomic/durable primitive: log writes	
Log entry written prior to mutations being applied	
to the database, tagged with transaction ID	
"Commit" log entry marks the transaction as	
committed	
\sim	
Occkroachdb Occkroachdb	

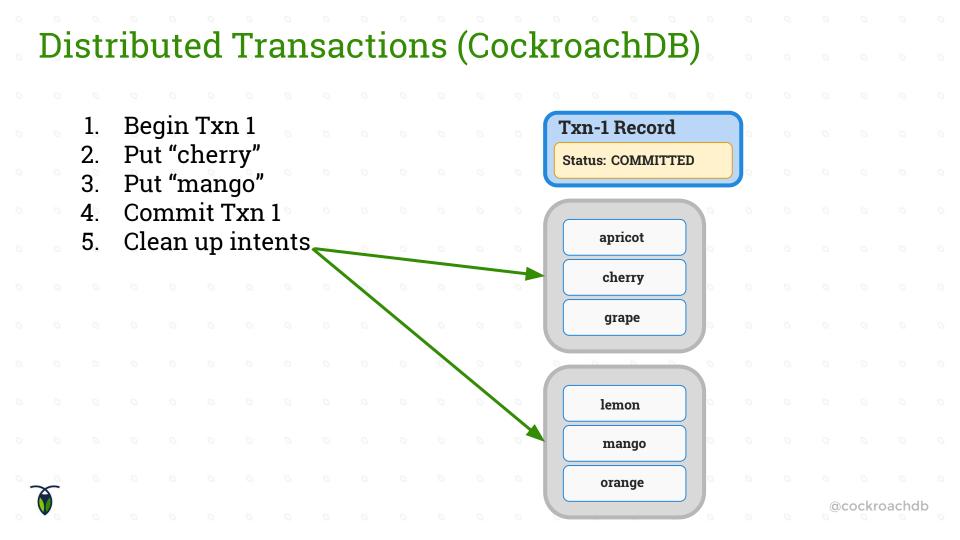
- Need lower-level primitive to bootstrap atomic
 - "commit" of transaction:
 - Write to a range (i.e. a Raft consensus group)
 - A transaction has an associated transaction record
 - keyed by the transaction ID
 - A transaction is atomically committed or aborted by updating the transaction record via a Raft write

0																				
0		1,	Be	gin	Txr	n 1	0	0	D	0	Ø	0		Reco Pendi		Ø				
0													atus. I		NG	 Q				
0																				
0																				
0																				
0																				
Ø																				
0																				
0																				
0	× ×																			
0																	00	ockro	achdb	0

Distributed Transactions (CockroachDB) Begin Txn 1 1. **Txn-1 Record** Put "cherry" 2. Status: PENDING apricot cherry (txn 1) grape

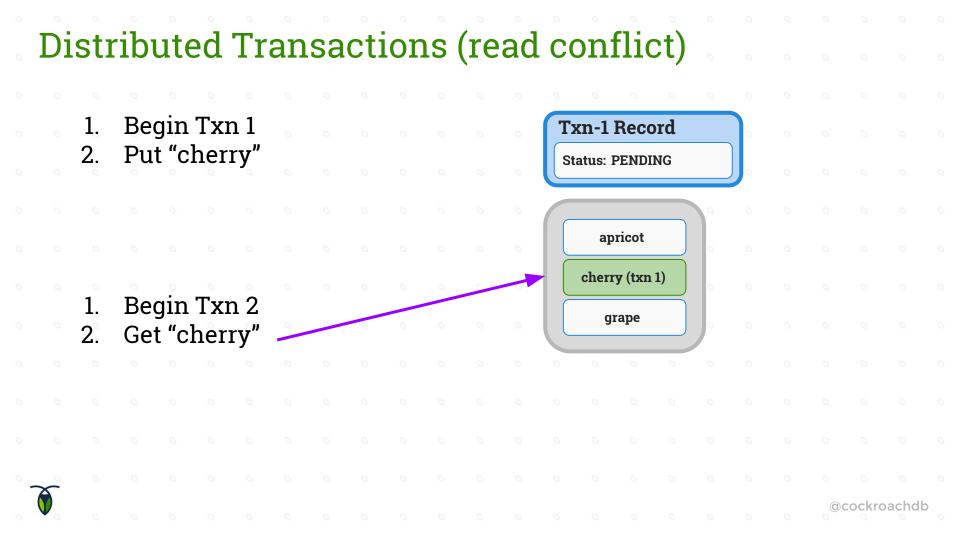


																	0
		1.		gin								Txn-1 Record	0				
			Pu Pu					<u>o</u>	9	0	-	Status: COMMITTED					0
		4 .		mm			0	0									0
												apricot					0
												cherry (txn 1)					0
												grape					0
																	0
												lemon					0
												mango (txn 1)					0
																	0
Ī	S											orange		@c	ockro	achdk	С
	0																



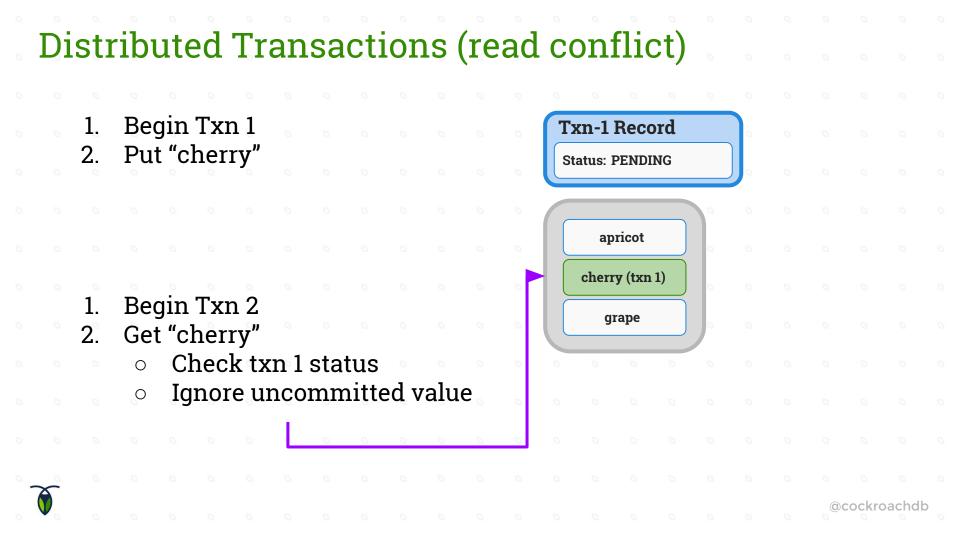
	Di	stı	cib	ute	ed	Tr	an	sa	cti														
			ہ احت	Q.	<u>o</u>	0	0																
	0	0	Th	lat	S 1	the	e h	ap	ру	C C	ase	90											
	0	0	W]	ha	t a	bo	ut	° CC	n	lic	tir	DU	tra	ans	sac	cti	on	s?					
	0	0	0	0	0	0	0	0	0	0	0	-9	0	0	0	Ø.	0	0					
0																							
	\mathbf{V}_{\circ}																			000	ockro	achdb)

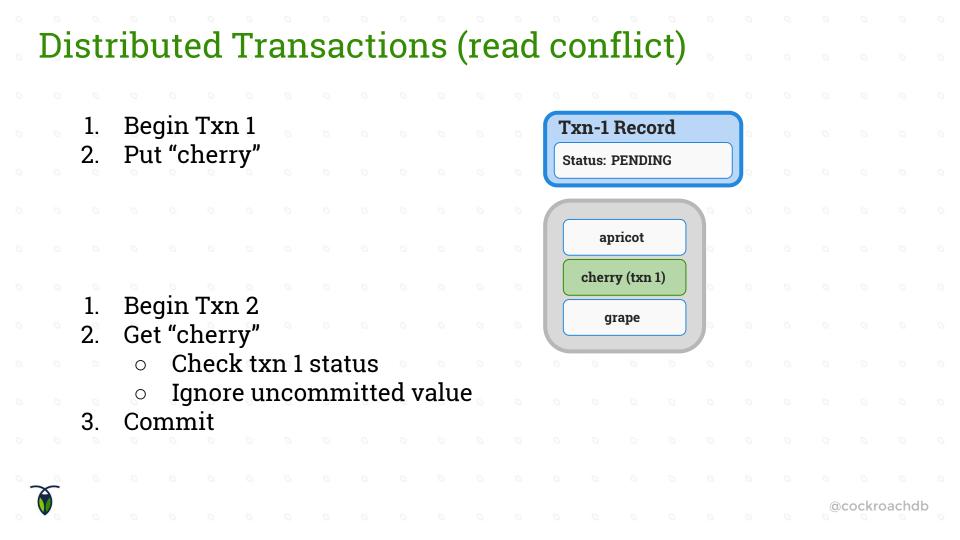
Distributed Transactions (read conflict) Begin Txn 1 1. **Txn-1 Record** Put "cherry" 2. Status: PENDING apricot cherry (txn 1) grape



Distributed Transactions (read conflict)

																					0
		1. 2	Begin Put "cl						0	Ø	0		kn-1]				0				0
		Д.		leiiy					9		<u>0</u>	St	atus: F	PENDII	NG		9				0
									0							0					0
									0					ricot		0					0
		1.	Begin	Tvn (2				0				cherr	y (txn	1)	0:					0
			Get "cł	herry	<i>n o</i>				0			l	gı	аре		0					0
				neck		stat	tus	0	0			0.	0	0	0						0
																					0
																					0
0	A A																				0
0																		@C	ockro	achdb	0

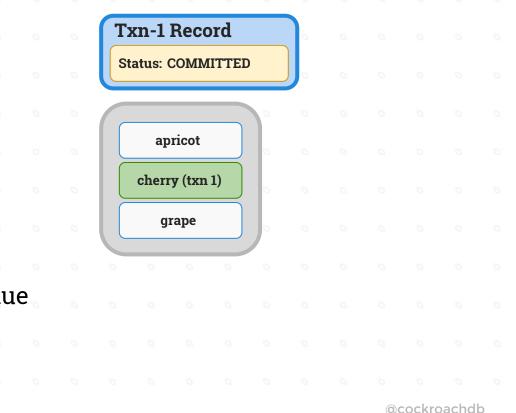


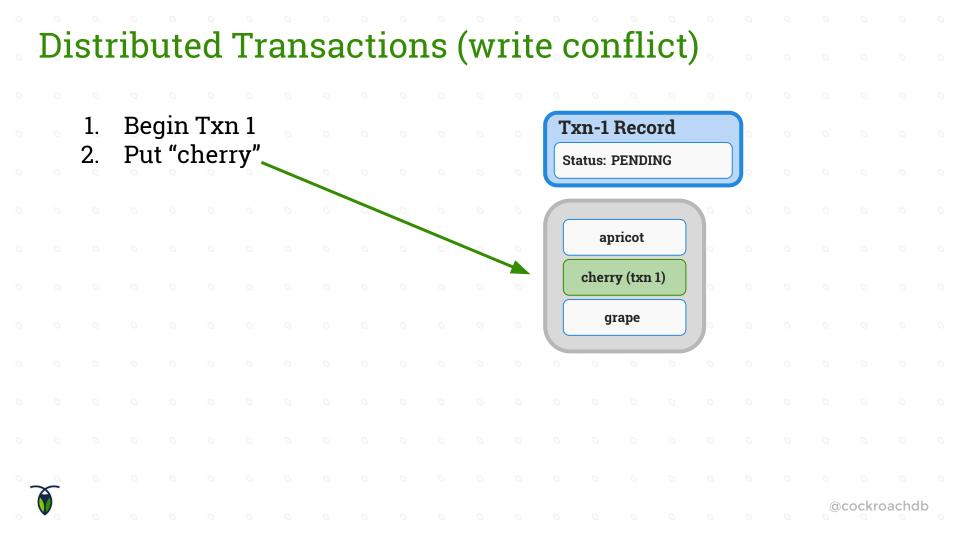


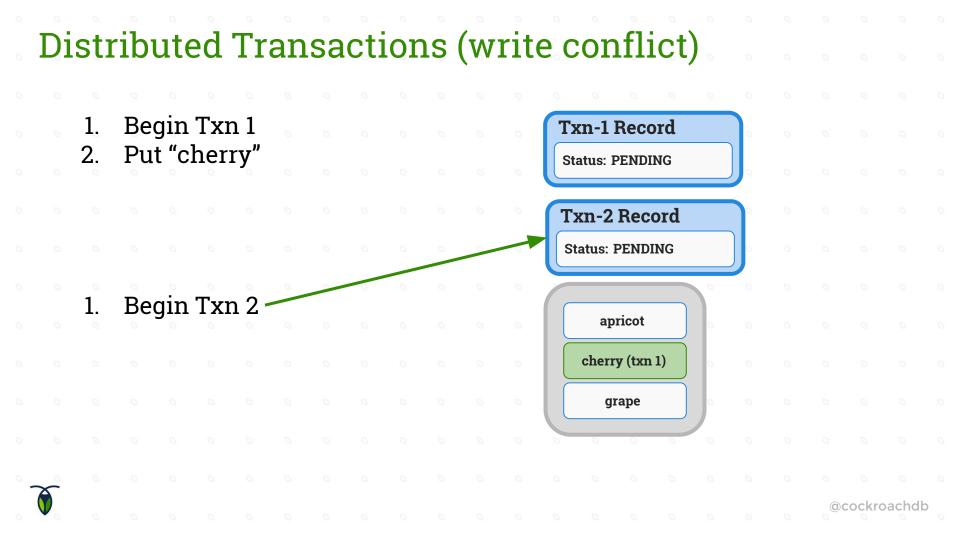
Distributed Transactions (read conflict)

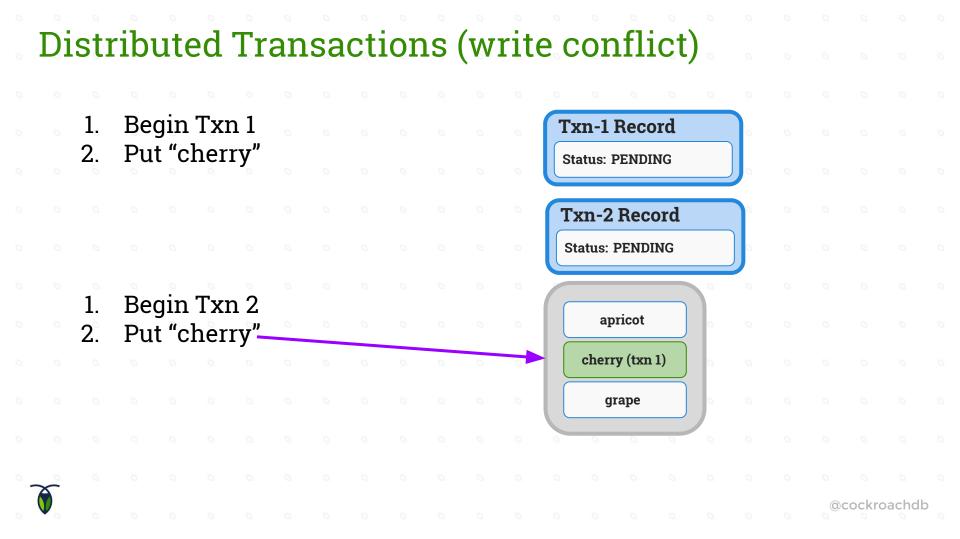
- 1. Begin Txn 1
- 2. Put "cherry"
- 3. Commit (potentially at later
- timestamp)

- 1. Begin Txn 2
- 2. Get "cherry"
 - Check txn 1 status
 - Ignore uncommitted value
- 3. Commit









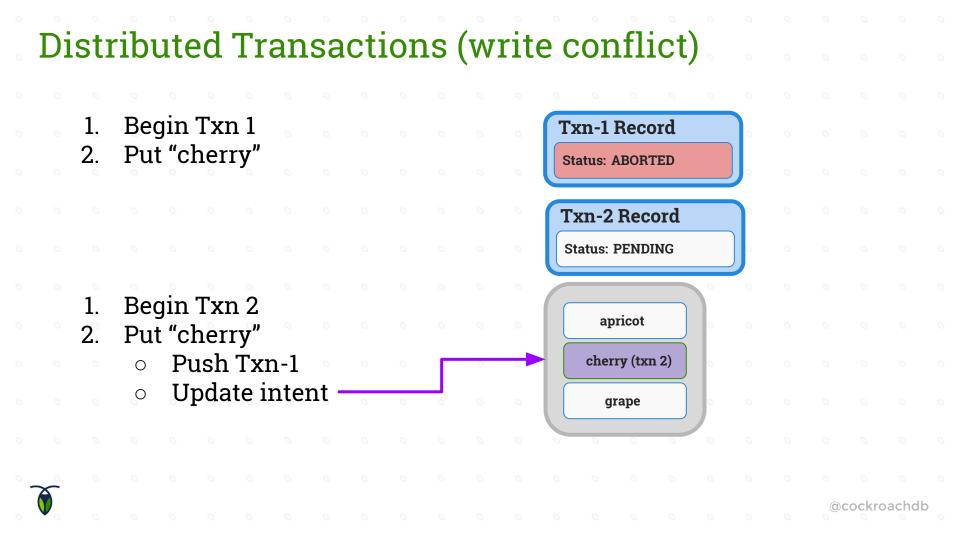
Distributed Transactions (write conflict) Begin Txn 1 **Txn-1 Record** 1. Put "cherry" 2. **Status: ABORTED**

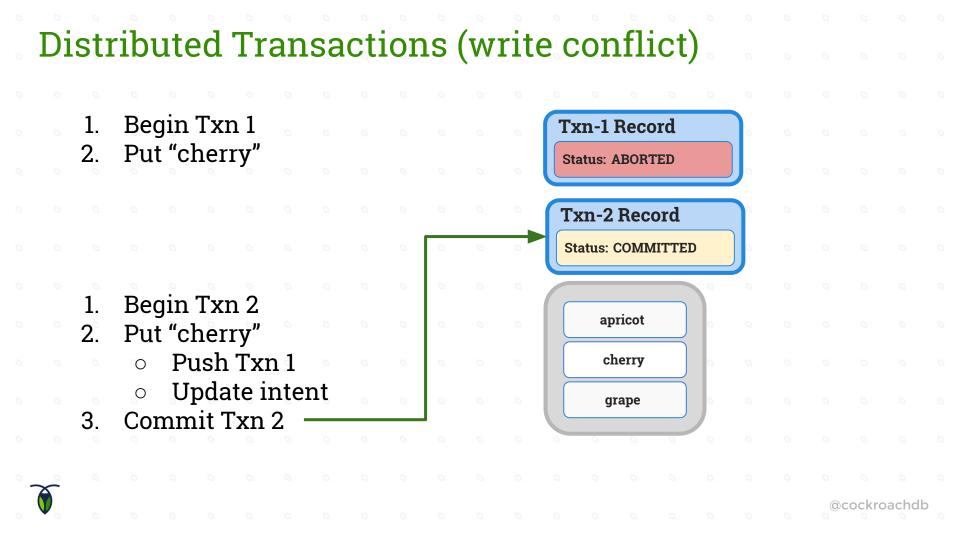
Begin Txn 2 1. Put "cherry" 2. 0

apricot Push Txn-1 cherry (txn 1) grape

Txn-2 Record

Status: PENDING





- Transaction atomicity is bootstrapped on top of Raft atomicity
 - Isolation, MVCC, other conflicts: ignored in this description
 - \circ More details on the Cockroach Labs blog¹

-	A 1	httpc		okro	aabla	m/bl	rioliz	obla	h oo	iotribu	ited i	laalat	ion o	ookro	aabd	h/		
	0																	
																		Q
																		0
																		0

Su	m	ma	ary	0		

	Building a SQL database for cloud environments																						
			0	Distribute data to support SQL workloads and																			
				fault-tolerance																			
			0	े R€	Replicate with Raft as foundation of atomicity																		
				Distributed transactions built on top																			
			0		SU.	01.		°	ua °	ns													
0	Ň																						
0																				@c	ockro	achdb	

			ık																					
0																								
0								Ĉ	oc]	kro	Dac	hL	al)S.	CO	m								
0				0	° Nit	hu	h	0	0	Ó.	ckı	0	0	0	0	0	, b	° ()2	ch	Ö.				
0				0	JIL				11/		CUI	UC		<u>IU</u>			, Û I	Uc	LC1	0				
0																								
0							al	ex	<u>@</u>	:00	kr	oa	ch	la	bs.	CO	m							
0							Ö	iitł	านไ	b.c	on	n∕a) -T	ob	in	SO]	n							
0							0	0.	0	0	Q	0	Ø	Q	Q	Ø	0							
0											No.	0												
0																								
0	X										0	0												
Q		0	0	0	0	0	0	0	0	0	Ø	0	0	0	0	0	0	0	Ø	0	0	ockro	achdl	b